Design and Training of Support Vector Machines

by

Alistair Shilton

Submitted in total fulfilment of
the requirements for the degree of

Doctor of Philosophy

Department Of Electrical and Electronic Engineering
The University of Melbourne
Australia

2006

ABSTRACT

SUPPORT Vector Machines (SVMs), which were first introduced by Vapnik in early 90s [28], have found applications in a wide variety of areas. As binary classifiers, SVMs have been used in face recognition [62], speaker identification [69] and text categorisation [49], to name a few. As regressors, they have been used in control systems [30] [89] and communications [77] [24], amongst others.

In this thesis I introduce a new and novel form of SVM known as regression with inequalities, in addition to the standard SVM formulations of binary classification and regression. This extension encompasses both binary classification and regression, reducing the workload when extending the general form; and also provides theoretical insight into the underlying connections between the two formulations.

This new SVM formulation is extended to cover general cost functions and general tube shrinking functions. Particular attention has been paid to the quadric cost functions, which present a number of pleasing properties not present in the traditional formulations. The quadric ν-SVR formulation presented provides a novel combination of the useful features of the least-squares SVM and the standard SVM.

It may be argued that the sheer generality of the SVM methodology is both a blessing and a curse. In particular, the great freedom allowed by the kernel function and the various parameters (C and ϵ, for example) is that one may construct an SVM to suit pretty much any situation. The downside of this, however, is that choosing these parameters can be a long and time consuming process.

Many attempts have been made to tackle this problem using different performance bounds with varying degrees of success (for example, [22]). Of these, the work of Schölkopf and Smola [76] stands apart, in-so-far as it represents an attempt to bridge the gap between the emerging field of support vector regression and the theoretically rich, well explored area of maximum-likelihood estimation. This thesis extends this approach significantly to cover a very general set of SVM formulations under a range of conditions, and explores the theoretical and practical issues involved.

As SVMs gained wider acceptance in the academic and industrial communities

during the late 90s, there arose for incremental training algorithms, which would allow application of SVMs in setting where the training set was non-constant, for example, adaptive control systems or equalization of time-varying communications channels. Our work on this problem is presented in the later chapters of this thesis.

ACKNOWLEDGMENTS

First I would like to thank my supervisor, Associate Professor Marimuthu Palaniswami, whose knowledge, guidance and patience made this research possible.

Second, I would also like to thank the many people who helped me along the way. Andrew Sanderson for convincing me to take the SVM project in my final year of undergrad, kick-starting the whole process. Daniel Lai, whose willingness to debate ideas and provide constructive criticism invariably led to a better understanding of the issues. Sophia, for being Sophia. Brendan Owen - thanks for all the fish! Laurence Park for introducing me to LaTeX, saving me from a life of torture by MS Word. Lachlan Andrew, for his offbeat sense of humour, knowledge of wide-ranging fields and friendship. The whole SEECS team over the years - Jayavardhana "JV" Rama, Bharat Sundaram, Dave Jahshan, Malcolm Peh, Tao Peng, Micheal Ngo, Leigh Palmer, Louis Lombard and all the others whose names I have unfairly left out or forgotten.

Last but not least, I would like to thank mum, dad and Anthony, who have always been there for me. In particular, I would like to thank my mother, Heather Shilton, whose strength, love and compassion through thick and thin to the very end were an inspiration to all who knew her. This one's for you, Mum.

PREFACE

During the course of this project, a number of public presentations have been made which are based on the work presented in this thesis. They are listed here for reference.

REFEREED PUBLICATIONS

[81] A. Shilton, M. Palaniswami, D. Ralph and A. C. Tsoi. Incremental Training of Support Vector Machines. *IEEE Transactions on Neural Networks*, vol. 16, no. 1, page 114–131, Jan 2005.

[78] A. Shilton, D. Lai and M. Palaniswami. A Monomial ν-SV Method For Regression. *Submitted to PAMI*, 2004.

CONFERENCE PROCEEDINGS

[80] A. Shilton, M. Palaniswami, D. Ralph and A. C. Tsoi. Incremental Training of Support Vector Machines. *Proceedings of International Joint Conference on Neural Networks, IJCNN'01 (CD version)*, 2001.

[79] A. Shilton and M. Palaniswami. A Modified ν-SV Method for Simplified Regression. *Proceedings of the International Conference on Intelligent Sensing and Information Processing*, page 422–427, 2004.

[64] M. Palaniswami, A. Shilton, D. Ralph and B. D. Owen. Machine Learning using Support Vector Machines. *Proceedings of International Conference on Artificial Intelligence in Science and Technology, AISAT2000*, 2000.

[63] M. Palaniswami and A. Shilton. Adaptive Support Vector Machines for Regression. *Proceedings of the 9th International Conference on Neural Information Processing*, vol. 2, page 1043–1049, November 2000.

[53] D. Lai, A. Shilton, N. Mani and M. Palaniswami. A Convergence Rate Estimate for the SVM Decomposition Method. *Proceedings of the International Joint Conference on Neural Networks, IJCNN05*, page 931–936, August 2005.

[19] S. Challa, M. Palaniswami and A. Shilton, Distributed data fusion using Support Vector Machines. *Proceedings of the Fifth International Conference on Information Fusion*, vol. 2, page 881–885, July 2002.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Nomenclature

Symbol	Definition
$\Re$	The set of reals.
$\Re^+$	The set of positive reals.
$\Re^-$	The set of negative reals.
$\mathbb{Z}$	The set of integers.
$\mathbb{Z}^+$	The set of positive integers.
$\mathbb{Z}^-$	The set of negative integers.
$\emptyset$	The empty set.
$\#(X)$	The number of elements in the set X.
$f^{\leftarrow}$	The inverse of an invertible function f.
$f\|\, Q_X$	The restriction of a map $f : Q \to R$ to $f : Q_X \to R_X$, $Q_X \subseteq Q$.
$\neg$	The logical negation (NOT) unary operator.
$\wedge$	The logical conjunction (AND) binary operator.
$\vee$	The logical inclusive disjunction (OR) binary operator.
$\veebar$	The logical exclusive disjunction (XOR) binary operator.
$\exists$	There exists.
$\forall$	For all.
g_{ij}	The metric tensor.
${\Gamma_{ij}}^k$	The affine connection.
${R^i}_{jkl}$	The curvature tensor.
$\ldots$	(see also appendix B for more on tensor notations)
$\delta_{x,y}$	The Kronecker-delta symbol.
$\delta(x)$	The Dirac-delta function.
$u(x)$	The Heaviside step function.
$\operatorname{sgn}(x)$	The sign function ($\operatorname{sgn}(0) = 0$).
$\Gamma(x)$	The gamma function.
$\psi_0(x)$	The digamma function.
$\psi_n(x)$	The polygamma function.
$\Gamma(x, y)$	The incomplete gamma function.
$B(x, y)$	The beta function.
$\daleth_m(x, y)$	The backgamma function (see appendix C).
$a, b, \ldots$	Scalars.
$\mathbf{a}, \mathbf{b}, \ldots$	Column vectors.
$\mathbf{A}, \mathbf{B}, \ldots$	Matrices.
$\mathbf{a}^T$	The transpose of column vector $\mathbf{a}$ (a row vector).
$\mathbf{A}^T$	The transpose of matrix $\mathbf{A}$.

Symbol	Definition
a_i	Element i of column vector $\mathbf{a}$.
$A_{i,j}$	Element i,j of matrix $\mathbf{A}$.
$\mathbf{0},\mathbf{1},\mathbf{2},\ldots$	Vectors where each element is $0,1,2,\ldots$.
$\mathbf{1}_{\pm},\mathbf{2}_{\pm},\ldots$	Vectors where each element is $\pm1,\pm2,\ldots$ (individual signs may differ).
$\mathbf{I}_l$	The $l\times l$ identity matrix (subscript optional).
$\mathbf{0}$	The zero matrix.
$\lvert\mathbf{a}\rvert$	Elementwise mod: $\mathbf{c}=\lvert\mathbf{a}\rvert$, $c_i=\lvert a_i\rvert$.
$\mathrm{sgn}\left(\mathbf{a}\right)$	Elementwise sign: $\mathbf{c}=\mathrm{sgn}\left(\mathbf{a}\right)$, $c_i=\mathrm{sgn}\left(a_i\right)$.
$\mathbf{a}\mathbf{b}^T$	Outer product: $\mathbf{C}=\mathbf{a}\mathbf{b}^T$, $C_{i,j}=a_ib_j$.
$\mathbf{a}\mathbf{b}$	Elementwise product: $\mathbf{c}=\mathbf{a}\mathbf{b}$, $c_i=a_ib_i$
$\mathbf{a}^T\mathbf{b}$	Inner product: $c=\mathbf{a}^T\mathbf{b}$, $c=a_1b_1+a_2b_2+\ldots+a_nb_n$.
$\lVert\mathbf{a}\rVert$	Euclidean vector norm: $\lVert\mathbf{a}\rVert=\sqrt{\mathbf{a}^T\mathbf{a}}$.
$\lVert\mathbf{a}\rVert_p$	Vector p-norm: $\lVert\mathbf{a}\rVert_p=\left(\sum_{i=1}^{n}\lvert a_i\rvert^p\right)^{\frac{1}{p}}$.
$\mathrm{diag}\left(\mathbf{a}\right)$	A diagonal matrix where $A_{i,i}=a_i$.
$\Pr\left(x\right)$	The probability of event x (if x is drawn from a finite or countably infinite set).
$p\left(x\right)$	The probability density function (if x is drawn from an uncountably infinite set).
$P\left(x\right)$	Some other description of the probability of event x.
$\underset{\Delta}{p}\left(x\right)$	Generalized density function: $\underset{\Delta}{p}\left(x\in X\right)=\left\{\begin{array}{ll}\Pr\left(x\right) & \text{if } \#\left(X\right) \text{ is finite or countable}\\ p\left(x\right)dx & \text{otherwise}\end{array}\right.$ Using this notation, the integral is to be interpreted thusly: $\int f\left(x\right)\underset{\Delta}{p}\left(x\right)=\left\{\begin{array}{ll}\sum f\left(x\right)\Pr\left(x\right) & \text{if } \#\left(X\right) \text{ is finite or countable}\\ \int f\left(x\right)p\left(x\right)dx & \text{otherwise}\end{array}\right.$
$\mathbf{E}\left(a\middle\vert b\right)$	The expectation of a given b.
Υ	The set of objects which are to be classified.
J	The set of classes to which elements of Υ belong.
$\hat{\Delta}$	The real classification map ($\hat{\Delta}:\Upsilon\to J$).
$\Upsilon_{\hat{\Delta}_j}$	The set of objects $\upsilon\in\Upsilon$ belonging to class $j\in J$.
$\hat{\mathbf{o}}$	The (noiseless) observation map to input space ($\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$).
Ξ	The image of Υ under $\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$.
$\Xi_{\hat{\Delta}_j}$	The image of $\Upsilon_{\hat{\Delta}_j}$ under $\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$.
Ξ_j	The set of elements of Ξ corresponding unambiguously only to objects of class $j\in J$ under $\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$.
$\Xi_{\checkmark}$	The set of elements of Ξ wherein each element $\mathbf{x}$ corresponds unambiguously to a single class $j\left(\mathbf{x}\right)\in J$ under $\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$.
$\Xi_{?}$	The set of elements of Ξ corresponding to objects of more than one class $j\in J$ under $\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$ (i.e. $\Xi_{?}=\Xi\backslash\Xi_{\checkmark}$).
Υ_j	The inverse image of Ξ_j under $\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$.
$\Upsilon_{\checkmark}$	The inverse image of $\Xi_{\checkmark}$ under $\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$.
$\Upsilon_{?}$	The inverse image of $\Xi_{?}$ under $\hat{\mathbf{o}}:\Upsilon\to\Re^{d_L}$.

Symbol	Definition
$P_{\Upsilon}(\upsilon)$	The probability of drawing the object υ from the set of all objects Υ.
$\mathbf{o}(\upsilon)$	The result of noisy observation of an object $\upsilon \in \Upsilon$.
$P_{\Xi}(\mathbf{x}\|\,\upsilon)$	The probability that (noisy) observation of an object $\upsilon \in \Upsilon$ will give the result $\mathbf{x} = \mathbf{o}(\upsilon) \in \Re^{d_L}$.
$\Delta(\upsilon)$	The result of classification of an object $\upsilon \in \Upsilon$ by the supervisor.
$P_{\Xi}(j\|\,\upsilon)$	The probability that the supervisor will classify an object $\upsilon \in \Upsilon$ as belonging to class $j = \Delta(\upsilon) \in J$.
$\gamma(\lambda)$	The (trained) classification function, parametrised by $\lambda \in \Lambda$.
Λ	The parameter set from which λ (in $\gamma(\lambda)$) may be selected.
T	The set of classifiers $\gamma(\lambda)$, $\lambda \in \Lambda$.
$E(\lambda)$	The probability that the classifier $\gamma(\lambda)$ will misclassify an object.
$R(\lambda)$	The risk associated with a classifier $\gamma(\lambda)$.
c	The cost function used to calculate risk ($c : \Re^{d_L} \times J^2 \to \Re$).
λ^*	The parameter choice $\lambda \in \Lambda$ which minimises $R(\lambda)$.
$\mathbf{Y}$	The set of training pairs $(\mathbf{x}_i, q_i)$ - $\mathbf{x}_i$ is the input and q_i the target.
$\mathbf{Y}_{=}$	The set of all training pairs corresponding to equality constraints in the regression training set.
$\mathbf{Y}_{\geq}$	Either the set of all training pairs corresponding to lower bound constraints in the regression case, or the set of points classed $+1$ in the binary classification case.
$\mathbf{Y}_{\leq}$	Either the set of all training pairs corresponding to upper bound constraints in the regression case, or the set of points classed -1 in the binary classification case.
$\mathbf{X}$	The set of all inputs $\mathbf{x}_i$ contained in the training set $\mathbf{Y}$.
$\mathbf{Q}$	The set of all targets q_i contained in the training set $\mathbf{Y}$.
$\mathbf{D}$	Alternative notation for $\mathbf{Q}$ used for classification.
$\mathbf{Z}$	Alternative notation for $\mathbf{Q}$ used for regression.
$R_{emp}(\lambda\|\,\mathbf{Y})$	The empirical risk associated with $\gamma(\lambda)$ for some $\mathbf{Y}$.
$R_{reg}(\lambda\|\,\mathbf{Y})$	The regularised empirical risk associated with $\gamma(\lambda)$ for some $\mathbf{Y}$.
$\phi(\lambda)$	The regularisation term used in $R_{reg}(\lambda\|\,\mathbf{Y})$.
C	The tradeoff constant in the regularised risk function.
$\mathcal{L}[\lambda\|\,\mathbf{Y}]$	The log-likelihood of the experimental setup shown in figure 3.3 generating the training set $\mathbf{Y}$.
e	The asymptotic efficiency of an estimator.
$e_q(\omega)$	The asymptotic efficiency of a q^{th} order monomial SVR.
$e_{q,p}(\omega)$	$e_q(\omega)$ in the presence of p^{th} order polynomial noise.
e_{LS}	Asymptotic efficiency of the LS-SVR.
d_L	The dimension of input space ($d_L \in \mathbb{Z}^+$).
d_H	The dimension of feature space ($d_H \in \mathbb{Z}^+ \cup \{\infty\}$).
$\boldsymbol{\varphi}$	The feature map ($\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$).
K	The kernel function associated with $\boldsymbol{\varphi}$ ($K : \Re^{d_L} \times \Re^{d_L} \to \Re$).
$\mathcal{M}_{\kappa}$	The set of all Mercer kernels.

Symbol	Definition
g	The trained SVM ($g : \Re^{d_L} \to \Re$).
$R_{\boldsymbol{\epsilon}}$	The set of possible (but not necessarily optimal) $g : \Re^{d_L} \to \Re$ which satisfy the primal constraints for the training set (usually (5.2)).
ρ	The margin of separation for a SVC.
ρ_b	The margin of separation for a SVC in augmented feature space.
${}_{d}R_{d^*}$	SVM primal: General convex cost.
${}_{d}L_{d^*}$	SVM Lagrangian: General convex cost.
${}_{d}Q\!\!\!L_{d^*}$	SVM partial dual: General convex cost.
${}_{d}Q_{d^*}$	SVM (Wolfe) dual: General convex cost.
R_q	SVM primal: q^{th} order monomial cost.
L_q	SVM Lagrangian: q^{th} order monomial cost.
$Q\!\!\!L_q$	SVM partial dual: q^{th} order monomial cost.
Q_q	SVM (Wolfe) dual: q^{th} order monomial cost.
${}_{d}R_{d^*,c}$	SVM primal: General convex cost and tube shrinking.
${}_{d}L_{d^*,c}$	SVM Lagrangian: General convex cost and tube shrinking.
${}_{d}Q\!\!\!L_{d^*,c}$	SVM partial dual: General convex cost and tube shrinking.
${}_{d}Q_{d^*,c}$	SVM (Wolfe) dual: General convex cost and tube shrinking.
$R_{q,n}$	SVM primal: q^{th} order monomial cost, n^{th} order shrinking.
$L_{q,n}$	SVM Lagrangian: q^{th} order monomial cost, n^{th} order shrinking.
$Q\!\!\!L_{q,n}$	SVM partial dual: q^{th} order monomial cost, n^{th} order shrinking.
$Q_{q,n}$	SVM (Wolfe) dual: q^{th} order monomial cost, n^{th} order shrinking.
ν	Tube shrinking constant.
β	Bias regularisation constant.
$\mathbf{K}$	The kernel matrix.
$\mathbf{H}$	The Hessian matrix.
E	Classification: overall hyperplane distance scale.
$\boldsymbol{\epsilon}$	Classification: ϵ_i sets relative hyperplane distance for $\mathbf{x}_i \in \mathbf{X}_{\geq}$.
$\boldsymbol{\epsilon}^*$	Classification: ϵ_i^* sets relative hyperplane distance for $\mathbf{x}_i \in \mathbf{X}_{\leq}$.
$\mathbf{t}$	Classification: t_i sets relative empirical risk weight for $\mathbf{x}_i \in \mathbf{X}_{\geq}$.
$\mathbf{t}^*$	Classification: t_i^* sets relative empirical risk weight for $\mathbf{x}_i \in \mathbf{X}_{\leq}$.
E	Regression: overall width of the ϵ-insensitive region.
$\boldsymbol{\epsilon}$	Regression: ϵ_i sets rel. positive ϵ-insensitive width for $\mathbf{x}_i \in \mathbf{X}_{=} \cup \mathbf{X}_{\geq}$.
$\boldsymbol{\epsilon}^*$	Regression: ϵ_i^* sets rel. negative ϵ-insensitive width for $\mathbf{x}_i \in \mathbf{X}_{=} \cup \mathbf{X}_{\leq}$.
$\mathbf{t}$	Regression: t_i sets rel. +ve empirical risk weight for $\mathbf{x}_i \in \mathbf{X}_{=} \cup \mathbf{X}_{\geq}$.
$\mathbf{t}^*$	Regression: t_i^* sets rel. −ve empirical risk weight for $\mathbf{x}_i \in \mathbf{X}_{=} \cup \mathbf{X}_{\leq}$.
$\mathbf{w}, b$	The primal weight vector ($\mathbf{w} \in \Re^{d_H}$) and bias ($b \in \Re$) specifying the decision surface in feature space.
$\boldsymbol{\alpha}$	Lagrange multipliers (dual variables) used instead of $\mathbf{w}$.
$\boldsymbol{\beta}, \boldsymbol{\beta}^*$	Alternative notation for $\boldsymbol{\alpha}$, splitting positive and negative values.
$\boldsymbol{\xi}, \boldsymbol{\xi}^*$	Slack (error) vectors.
$\mathbf{e}$	Gradient of the (partial) dual with respect to $\boldsymbol{\alpha}$.
f	Gradient of the (partial) dual with respect to b.

Symbol	Definition
N	The number of training vectors ($N = \#(\mathbf{Y})$).
N_S	The number of support vectors.
N_B	The number of boundary vectors.
N_E	The number of error vectors.
N_Z	The number of α's at zero.
N_L	The number of α's at a (non-zero) lower bound.
N_U	The number of α's at a (non-zero) upper bound.
N_F	The number of α's not at a constraint boundary.
N_{F+}	The number of positive α's not at a constraint boundary.
N_{F-}	The number of negative α's not at a constraint boundary.
$\lvert x\rvert_\epsilon$	The ϵ-insensitive magnitude ($\lvert x\rvert_\epsilon = \max(0, \lvert x\rvert - \epsilon)$).
$\underline{\mathbf{a}}$	Ordered notation - see chapter 7.
$\boldsymbol{\tau}$	The type vector.
$\mathbf{m}$	The order vector.
$\mathbf{T}$	The abstract training set.
$\boldsymbol{\vartheta}$	The abstract training data.
$\maltese$	The abstract solution to the SVM optimisation problem.
$\gimel$	The optimisation state.
$a^{(k)}, b^{(k)}, \ldots$	The value of something at iteration k.
F	Algorithm dependent information.
$\beth_{\boldsymbol{\tau}}$	The hessian factorization.

Acronym	Definition
i.i.d.	Independent and identically distributed.
KKT	Karush-Kuhn-Tucker optimality conditions.
LM	Learning machine.
LS	Least-squares.
ML	Maximum likelihood.
MSE	Mean squared error.
QP	Quadratic programme.
RBF	Radial-basis function.
RMSE	Root mean squared error.
SMO	Sequential minimal optimisation.
SRM	Structural risk minimisation.
SSE	Sum squared error.

Acronym	Definition
SV	Support vector.
SVM	Support vector machine.
SVC	Support vector classifier.
SVR	Support vector regressor.
C-SVM	Support vector machine without tube shrinking.
C-SVR	Support vector regressor without tube shrinking.
C-SVC	Support vector classifier without tube shrinking.
ν-SVM	Support vector machine with tube shrinking.
ν-SVR	Support vector regressor with tube shrinking.
ν-SVC	Support vector classifier with tube shrinking.
LS-SVM	Least-squares support vector machine.
LS-SVR	Least-squares support vector regressor.
VC	Vapnik-Chervonenkis dimension.

Chapter 1

INTRODUCTION

From the moment I picked your book up until I laid it down I was convulsed with laughter. Someday I intend reading it.

– Groucho Marx

BINARY pattern recognition involves constructing a decision rule to classify vectors into one of two classes based on a training set of vectors whose classification is known a-priori. Support vector machines (SVMs [28] [46]) do this by implicitly mapping the training data into a higher-dimensional feature space. A hyperplane (decision surface) is then constructed in this feature space that bisects the two categories and maximises the margin of separation between itself and those points lying nearest to it (called the support vectors). This decision surface can then be used as a basis for classifying vectors of unknown classification.

The main advantages of the SVM approach are:

- SVMs implement a form of structural risk minimisation (SRM [94]) - They attempt to find a compromise between the minimisation of empirical risk and the prevention of overfitting.
- The problem is a convex quadratic programming (QP) problem. So there are no non-global minima, and the problem is readily solvable using quadratic programming techniques.
- The resulting classifier can be specified completely in terms of its support vectors and kernel function type.

Support Vector regressors (SVRs [34] [97] [84]) are a class of non-linear regressors inspired by Vapnik's SV methods for pattern classification. Like Vapnik's method, SVRs first implicitly map all data into a (usually) higher dimensional feature space.

In this feature space, the SVR attempts to construct a linear function of position that mimics the relationship between input (position in feature space) and output observed in the training data by minimising a measure of the empirical risk. To prevent overfitting a regularisation term is included to bias the result toward functions with smaller gradient in feature space.

Two major advantages that SVRs have over competing methods (unregularised least-squares methods, for example) are sparseness and simplicity [84] [13]. SVRs are able to give accurate results based only on a sparse subset of the complete training set, making them ideal for problems with large training sets. Moreover, such results are achievable without excessive algorithmic complexity, and use of the kernel "trick" makes the dual form of the SVR problem particularly simple.

Roughly speaking, SVR methods may be broken into ϵ-SVR [34] [97] and ν-SVR methods [70] [75], both of which require a-priori selection of certain parameters. Of particular interest is the ϵ (or ν in ν-SVR methods) parameter, which controls the sensitivity of the SVR to presence of noise in the training data. In both cases, this parameter controls the threshold ϵ (directly for ϵ-SVR, indirectly for ν-SVR) of insensitivity of the cost function to noise through use of Vapnik's ϵ-insensitive loss function.

The standard ϵ-SVR approach is associated with a simple dual problem, but unfortunately selection of ϵ requires knowledge of the noise present in the training data (and its variance in particular) which may not be available [83]. Conversely, the standard ν-SVR method has a more complex dual form, but has the advantage that selection of ν requires less knowledge of the noise process [83] (only the form of the noise is required, not the variance). Thus both forms have certain difficulties associated with them.

Yet another approach is that of Suykens' least-squares SVR (LS-SVR [87]), which uses the normal least-squares cost function with an added regularisation term inspired by Vapnik's original SV method. The two main advantages of this approach are the simplicity of the resulting dual cost function, which is even simpler than ϵ-SVR; and having one less constant to choose a-priori. The disadvantages include loss of sparsity and robustness in the solution. These problems may be ameliorated somewhat through use of a weighted LS-SVR scheme [88]. However, while this method is noticeably superior when extreme outliers are present in the training data, in our experience the performance of the weighted LS-SVR may not be significantly better than the standard LS-SVR if such outliers are not present.

Usually, support vector machines are trained using a batch model. Under this model, all training data is given *a priori* and training is performed in one batch. If more training data is later obtained, or parameters are modified, the SVM must be re-trained from scratch. But if only a small amount of data is to be added to a large training set (assuming that the problem is well posed) then it will likely have only a minimal effect on the decision surface. Re-solving the problem from scratch seems computationally wasteful.

An alternative is to "warm-start" the solution process by using the old solution as a starting point to find a new solution. This approach is at the heart of active set optimisation methods [40], [27] and, in fact, incremental learning is a natural extension of these methods. While many papers have been published on SVM training, relatively few have considered the problem of incremental training.

1.1 Focus of Thesis

There are three main areas forming the focus of these thesis, which can be roughly summarised as theory, analysis and implementation.

The theoretical focus of the thesis is the consolidation and extension of the traditional SVM methodologies. Firstly, when the standard C-SVC and C-SVR methodologies are introduced, the optimisation dual is formulated in such a way that the SVC and SVR problems are (essentially) indistinguishable. This motivates the introduction of the novel "regression with inequalities" SVM formulation, of which regression and pattern classification are shown to be special cases. The focus is then shifted to extensions of this generic formulation, including both well-known (e.g. LS-SVM) and novel (e.g. quadric ν-SVM) approaches.

The analytical focus of the thesis is the extension of Schölkopf and Smola's work on asymptotic efficiency and it's application to parameter selection to the more general formulations introduced previously.

Finally, the practical focus of the thesis is the description of our work in the field of incremental training of SVMs. Once again, the aim is to design a training algorithm which may be applied to the "regression with inequalities" formulation of the SVM problem, which is able to implement as many of the extensions as practical without compromising training speed.

In addition to this, I have included a brief survey of the relevant background material pertinent to the thesis, which forms chapters 2 and 3 and the appendices.

1.2 Overview of Chapters

Chapter 2: Classification and Regression gives a very general introduction to the problems of classification and regression with which SVM methods are concerned. The problems are formalised and some discussion is given about the difficulties which must be overcome by a classifier or regressor. Of particular importance in this chapter is the definition of the cost function and the concept of risk, as associated with a classifier or regressor.

Chapter 3: The Learning Paradigm goes into greater detail on the concepts of training a classifier or regressor. In particular, the idea of a training set is introduced, as well as the empirical risk associated with such a training set for a classifier. As an adjunct to this, generalisation, overfitting and regularisation methodologies are described. Finally, the concepts of maximum likelihood estimation and efficiency are introduced, which will be required later in chapter 6.

Chapter 4: Support Vector Machine Basics sees the introduction of the fundamental SVM methodologies. Following the standard recipe, the SVM classifier is introduced first, initially assuming separable data and then extending to the inseparable case. This work in then related back to the risk minimisation (and in particular structural risk minimisation) and regularisation to demonstrate the solid theoretical foundations of the SVM approach. The kernel trick is analysed in some detail, including the connection between the kernel and ideas of curvature of input space. Finally, the SVM regressor is introduced.

It will be noted that, while the SVM formulations given in this chapter are, in essence, just the standard C-SVC and C-SVR formulations, the notation used is slightly non-standard. Also, a number of parameters are included which are normally not present. This is done to make plainly apparent the connection between these standard formulations, as discussed in this chapter, and used to advantage in subsequent chapters.

Chapter 5: Extensions to the Basic SVM Method begins with the introduction of the novel "regression with inequalities" SVM formulation, which is shown to be a superset of the standard SVM forms of binary classification and regression. Using this as a foundation, a series of extensions and generalisations of this formulation are then introduced, namely:

Fixed and Automatic Biasing: SVMs without a bias term, Mangasarian's automatic biasing (bias regularisation) scheme, and the connection between these through the kernel.

Generalised Empirical Risk: the use of arbitrary convex cost functions, with a particular focus on the case of monomial and quadric cost functions.

Tube Shrinking: Schölkopf et al.'s tube shrinking SVM formulation and the ν-SVC.

Generalised Tube Shrinking: the use of arbitrary convex tube shrinking functions (combined with arbitrary convex cost functions), with a particular focus on monomial and quadric cost functions.

At the end of the chapter an overall summary is given describing how the various extensions and concepts may be combined in the dual form of the SVM optimisation problem.

Chapter 6: Asymptotic Analysis of SV Regression gives an asymptotic analysis of the monomial SVM regressor in terms of efficiency, and then describes how this may be used to optimally select parameters for the SVM. Particular attention is paid to the case of polynomial noise, and a number of theorems are presented showing the intimate connection between the degree of the monomial cost function and the degree of the (polynomial) noise present in the training data.

Chapter 7: Training the SVM - Preliminaries shows how the various SVM formulations may be combined in the form of an abstract (dual) optimisation problem. The remainder of this chapter is spent analysing the properties of this abstract formulation and constructing the underlying mechanisms required to solve the abstract optimisation problem presented.

Chapter 8: Training the SVM - An Active Set Approach presents in some detail our work on the incremental (and batch) training of SVMs using an active set method. First the algorithm is described in some detail. Then the algorithm is analysed, a proof of convergence is given, and some discussion is given of the relative merits of the algorithm. Finally, it is demonstrated how the algorithm may be used to implement incremental learning.

Appendix A: Detailed derivations of Dual and Partial Dual Forms contains detailed derivations for the SVM formulations given in chapters 4 and 5, which may be skipped or abbreviated in the body of the thesis for reasons of clarity and brevity.

Appendix B: Differential Geometry gives a summary of the theory of differential geometry, which is used in chapter 4 when analysing the geometry induced on input space by kernel functions.

Appendix C: The Gamma and Related Functions gives details of the gamma, digamma, polygamma, incomplete gamma, beta and backgamma functions required by chapter 6. Theorems relevant to the thesis are summarised and relevant references given.

Appendix D: Algorithms gives the matrix factorisation algorithms required by chapter 8.

Appendix E: A Useful Property of the Determinant contains a proof of theorem 4.4, which is a property of the determinant of diagonally perturbed rank one matrices.

1.3 Publications

[**81**] A. Shilton, M. Palaniswami, D. Ralph and A. C. Tsoi. Incremental Training of Support Vector Machines. *IEEE Transactions on Neural Networks*, vol. 16, no. 1, page 114–131, Jan 2005.

[**78**] A. Shilton, D. Lai and M. Palaniswami. A Monomial ν-SV Method For Regression. *Submitted to PAMI*, 2004.

[**80**] A. Shilton, M. Palaniswami, D. Ralph and A. C. Tsoi. Incremental Training of Support Vector Machines. *Proceedings of International Joint Conference on Neural Networks, IJCNN'01 (CD version)*, 2001.

[**79**] A. Shilton and M. Palaniswami. A Modified ν-SV Method for Simplified Regression. *Proceedings of the International Conference on Intelligent Sensing and Information Processing*, page 422–427, 2004.

[**64**] M. Palaniswami, A. Shilton, D. Ralph and B. D. Owen. Machine Learning using Support Vector Machines. *Proceedings of International Conference on Artificial Intelligence in Science and Technology, AISAT2000*, 2000.

[**63**] M. Palaniswami and A. Shilton. Adaptive Support Vector Machines for Regression. *Proceedings of the 9th International Conference on Neural Information Processing*, vol. 2, page 1043–1049, November 2000.

[**53**] D. Lai, A. Shilton, N. Mani and M. Palaniswami. A Convergence Rate Estimate for the SVM Decomposition Method. *Proceedings of the International Joint Conference on Neural Networks, IJCNN05*, page 931–936, August 2005.

[**19**] S. Challa, M. Palaniswami and A. Shilton, Distributed data fusion using Support Vector Machines. *Proceedings of the Fifth International Conference on*

Information Fusion, vol. 2, page 881–885, July 2002.

Chapter 2

CLASSIFICATION AND REGRESSION

It is a mistake to think you can solve any major problems just with potatoes.

– Douglas Adams

2.1 The Classification Problem

One of the most common problems in the machine learning field is the *classification problem*. A common form of this is the following:

> For some set of objects Υ, each belonging uniquely to one of M distinct classes, construct a machine that is able to correctly ascertain the classification of any object $\upsilon \in \Upsilon$ when presented with a set of observations of that object.

Some examples of the classification problem include face recognition [62], speaker identification [69] and text categorisation [49]. The machine learning approach to this problem is one of "learning from data", wherein one starts with a general machine which then learns (or is trained) from a finite set of examples (training data). This has the practical advantages of not assuming any particular knowledge of the problem, nor the existence of an infinite pool of training data from which samples may be drawn.

2.1.1 Basic Definitions - Sets and Noiseless Measurements

Let Υ denote the set of all possible distinct instances of object under consideration. Each element $\upsilon \in \Upsilon$ is assumed belong to precisely one of M possible classes, labelled by the index set J (where $\#(J) = M$). This classification is given by a many-to-one *target map*, $\hat{\Delta} : \Upsilon \to J$, and an object $\upsilon \in \Upsilon$ is said to belong to the class $d = \hat{\Delta}(\upsilon)$.

In most cases the target map will not be known, although indirect access to this map is assumed during the training process, as will be detailed later. Using the target map it is possible to divide the set Υ into M subsets $\Upsilon_{\hat{\Delta}_j}$, $j \in J$, such that all the elements of $\Upsilon_{\hat{\Delta}_j}$ belong to the same class, j. Symbolically:

$$\Upsilon_{\hat{\Delta}_j} = \left\{ \upsilon \in \Upsilon \middle| \, \hat{\Delta}(\upsilon) = j \right\} \forall j \in J \tag{2.1}$$

Lemma 2.1. *The set* $\left\{ \Upsilon_{\hat{\Delta}_j} \middle| j \in J \right\}$ *forms a disjoint cover for* Υ*.*

The elements of the set Υ are physical objects or processes (for example, faces, or signals in a communications channel). To classify these objects, the classifier must have some way of measuring the relevant attributes of the object in question to obtain something a computer can work with (i.e. a finite set of numbers). Ignoring the potential for measurement error and variation, this process is captured by the idealised *observation map*, $\hat{\mathbf{o}} : \Upsilon \to \Re^{d_L}$, $d_L \in \mathbb{Z}^+$. In a face recognition system, for example, the process may involve obtaining, scanning and pixellating an image of the face in question.

The observation map $\hat{\mathbf{o}} : \Upsilon \to \Re^{d_L}$ maps objects $\upsilon \in \Upsilon$ to points in d_L (where d_L is assumed to be finite) dimensional *input space*. For an object $\upsilon \in \Upsilon$, $\hat{\mathbf{o}}(\upsilon)$ may be thought of as a distillation of (some part of) the readily observable information about the object, whereas finding $\hat{\Delta}(\upsilon)$ presumably requires some sort of non-trivial insight (although how "non-trivial" this insight is is, of course, problem dependant).

Using this notation, the aim of pattern classification may be stated thusly:

> The aim of pattern classification is to construct a function $\gamma : \Re^{d_L} \to J$ from input space to a putative classification, such that $\gamma \circ \hat{\mathbf{o}} = \hat{\Delta}$.

However, as will be shown in the following sections, this may not be possible in reality, or even (in some cases) in principle.

2.1.2 Performance Limiting Factors

There are a number of factors that limit the potential best case performance of all classifiers. These factors can be broken into two types. The first type of factor is random error, which includes observation noise and variation, and can be countered by experimental repetition (e.g. by classifying an object multiple times and finding a solution using a voting method). The second type of factor is information loss. This cannot be countered without re-designing the observation procedure, and as such forms a hard limit on the best-case performance of the classifier.

Observation Noise and Variation

As defined previously, the observation map is an idealisation that is not generally achievable in reality. Usually, the process of observation will be affected by noise and variation (by which I mean the possibility that many different observations may correctly identify a given object $\upsilon \in \Upsilon$).

Given an object $\upsilon \in \Upsilon$, the result of a real measurement of the characteristics of this object will be denoted $\mathbf{o}(\upsilon)$, where $\mathbf{o}(\upsilon)$ is a random variable characterised by some conditional probability distribution $P_{\Xi}(\mathbf{x}|\upsilon)$. If no noise is present, $\mathbf{o}(\upsilon) = \hat{\mathbf{o}}(\upsilon)$ for all observations and for all $\upsilon \in \Upsilon$.

Hence even if γ is selected optimally for the noiseless case (i.e. $\gamma \circ \hat{\mathbf{o}} = \hat{\Delta}$), there is still a possibility of incorrect classification (i.e. for an object, $\upsilon \in \Upsilon$, $\gamma(\mathbf{o}(\upsilon)) \neq \hat{\Delta}(\upsilon)$ for some fraction of measurements).

Information Loss and Observability

Consider the process of observing objects in Υ (ignoring measurement errors and variations). The observation map implicitly defines the following regions in input space:

$$\Xi_{\hat{\Delta}_j} = \hat{\mathbf{o}}\left(\Upsilon_{\hat{\Delta}_j}\right) \forall j \in J$$

Lemma 2.2. *The set* $\left\{ \Xi_{\hat{\Delta}_j} \middle| j \in J \right\}$ *forms a cover for* Ξ.

Any element $\mathbf{x} \in \Xi_{\hat{\Delta}_i}$ is the image of some object (or objects) $\upsilon \in \Upsilon_{\hat{\Delta}_i}$. As the observation map is in general not one-to-one, the sets $\Xi_{\hat{\Delta}_j}$ will not in general be disjoint. Hence for any element $\mathbf{x} \in \Xi_{\hat{\Delta}_i}$, there is no guarantee that there is not some object $\omega \in \Upsilon_{\hat{\Delta}_j}$, $j \neq i$, such that $\mathbf{x} = \hat{\mathbf{o}}(\omega)$, making unambiguous classification of $\mathbf{x}$ impossible.

Because of this, it is convenient to define:

$$\begin{aligned}
\Xi_{?} &= \bigcup_{i \neq j \in J} \left(\Xi_{\hat{\Delta}_i} \cap \Xi_{\hat{\Delta}_j} \right) \\
\Xi_j &= \Xi_{\hat{\Delta}_j} \backslash \Xi_{?} \\
\Xi_{\checkmark} &= \bigcup_{j \in J} \Xi_j \\
\Upsilon_{?} &= \hat{\mathbf{o}}^{\leftarrow}\left(\Xi_{?}\right) \\
\Upsilon_j &= \hat{\mathbf{o}}^{\leftarrow}\left(\Xi_j\right) \\
\Upsilon_{\checkmark} &= \hat{\mathbf{o}}^{\leftarrow}\left(\Xi_{\checkmark}\right)
\end{aligned}$$

Lemma 2.3. *The set* $\{\Xi_j | \, j \in J\}$ *forms a disjoint cover for* $\Xi_{\checkmark}$.

Lemma 2.4. *The set* $\{\Xi_{?}, \Xi_{\checkmark}\}$ *forms a disjoint cover for* Ξ.

Theorem 2.5. $\Upsilon_j \subseteq \Upsilon_{\hat{\Delta}_j}$.

Proof. Suppose there exists some $\upsilon \in \Upsilon_j$ such that $\upsilon \notin \Upsilon_{\hat{\Delta}_j}$. Then by lemma 2.1 $\upsilon \in \Upsilon_{\hat{\Delta}_i}$ for some $i \neq j$, and so $\mathbf{x} = \hat{\mathbf{o}}(\upsilon) \in \Xi_{\hat{\Delta}_i}$. But by definition of Υ_j there must exist some $\mathbf{y} \in \Xi_j$ such that $\mathbf{y} = \hat{\mathbf{o}}(\upsilon)$, and as $\hat{\mathbf{o}}$ is many-to-one, it follows that $\mathbf{x} = \mathbf{y}$. So $\mathbf{x} \in \Xi_{\hat{\Delta}_i}$ and $\mathbf{x} \in \Xi_j$, which is not possible, as Ξ_j and $\Xi_{\hat{\Delta}_i}$ are disjoint. So $\Upsilon_j \subseteq \Upsilon_{\hat{\Delta}_j}$. □

The set $\Upsilon_{\checkmark}$ defined here may be thought of as those objects which may (in principle) be unambiguously classified using the noiseless observation map $\hat{\mathbf{o}}$ and an appropriately constructed classification function γ. In other words:

Lemma 2.6. $\exists \gamma : \Re^{d_L} \to J$ *such that* $\gamma \circ \hat{\mathbf{o}} | \, \Upsilon_{\checkmark} = \hat{\Delta} \Big| \, \Upsilon_{\checkmark}$

Conversely, for any $\mathbf{x} \in \Xi_{?}$ there will be at least two objects $\upsilon, \omega \in \Upsilon$ such that $\mathbf{x} = \hat{\mathbf{o}}(\upsilon) = \hat{\mathbf{o}}(\omega)$ but $\hat{\Delta}(\upsilon) \neq \hat{\Delta}(\omega)$, as shown in figure 2.1. Clearly no classifier will be able to correctly classify both υ and ω. Hence:

Lemma 2.7. *If* $\Upsilon_{?} \neq \emptyset$, $\neg \exists \gamma : \Re^{d_L} \to J$ *such that* $\gamma \circ \hat{\mathbf{o}} | \, \Upsilon_{?} = \hat{\Delta} \Big| \, \Upsilon_{?}$

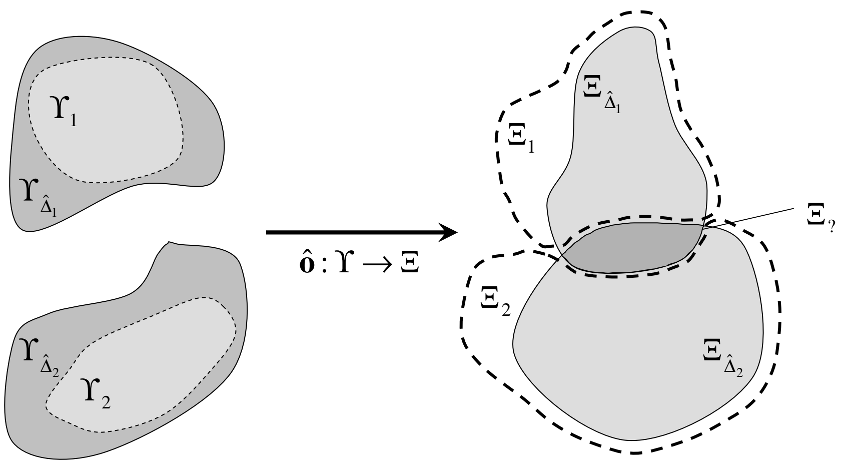

Υ_1
$\Upsilon_{\hat{\Delta}_1}$
$\Upsilon_{\hat{\Delta}_2}$
Υ_2
$\hat{\mathbf{o}} : \Upsilon \to \Xi$
$\Xi_{\hat{\Delta}_1}$
Ξ_1
$\Xi_?$
Ξ_2
$\Xi_{\hat{\Delta}_2}$

Figure 2.1: Unclassifiability of points due to measurement (2 class example).

So, even if noiseless observation is achievable it may not be possible to achieve perfect classification due to the inherent limitations of the observation process. Specifically, it will not be possible to construct a perfect classifier unless $\Upsilon_? = \emptyset$. This limitation may be the result of flawed or limited measurements or even the physical indistinguishability of objects in Υ.

2.1.3 Sampling, Risk and Optimal Classification

In the previous section, it was shown why in general it is not possible to construct a perfect classifier, even if "perfect" observation is possible. Given this fact, and given that the observation procedure will almost certainly introduce some noise and/or variation, it is still sensible when constructing a classifier to attempt to make the performance of that classifier as close to perfect as possible (or practical). To do this, it is necessary to have some means of gauging the performance of classifiers. Specifically, it would be helpful to be able to state the probability of misclassification of some randomly selected object $\upsilon \in \Upsilon$.

Unless otherwise stated, it will be assumed throughout this thesis that objects are drawn from Υ in an i.i.d. manner according to the probability distribution $P_\Upsilon(\upsilon)$ (this will be referred to subsequently as "the usual procedure"). To gauge the performance of the classifier (and also, as will be discussed later, to train the classifier), the presence of a supervisor (either a person or another machine), who can classify objects to some degree of accuracy, is assumed. The action of this supervisor is modelled by the noisy assignment procedure $\Delta(\Upsilon)$, which is a random variable characterised by the conditional probability distribution $P_{\Xi}(d|\,\upsilon)$. For a perfect supervisor, $\Delta(\upsilon) = \hat{\Delta}(\upsilon)$ for all classifications and for all $\upsilon \in \Upsilon$.

The supervisor may be thought of as a person who can be present during training and testing, but for practical reasons cannot act as a classifier at other times. For example, a face-in-a-crowd recognition system may need to study thousands of faces every minute, a task which would be impractical if carried out by human operators.

In this section, it is necessary differentiate between two different phases of operation. First is the *training/evaluation phase*. During this phase, it is assumed that a supervisor is present, and hence both the observed characteristics of an object $\mathbf{o}(\upsilon)$ and also the postulated classification $\Delta(\upsilon)$ are available. The second phase of operation is *use*, during which only the observed characteristics of an object, $\mathbf{o}(\upsilon)$, are available.

Equivalent Models - Training and Use

Suppose some element $\upsilon \in \Upsilon$ is selected according to the usual procedure, and its characteristics observed to obtain $\mathbf{o}(\upsilon)$. The only observable result from this process is the result of the noisy observation, which will be seen to be an i.i.d. random variable satisfying some probability function $\underset{\Delta}{p}(\mathbf{x})$.[1]

Suppose again that some element $\upsilon \in \Upsilon$ is chosen and characterised in this manner. However, in addition to noisy observation, the supervisor classifies the object as $\Delta(\upsilon)$ in accordance with the conditional distribution $P_{\Xi}(d|\,\upsilon)$. This may be thought of as an augmentation of the observation process, where the observable part is the combination of noisy observation and classification, characterised by the

[1]The notation $\underset{\Delta}{p}(\mathbf{x})$ is intended to incorporate both the discrete and continuous probability cases. Specifically, if $x \in X$ is drawn from a finite or countable set then $\underset{\Delta}{p}(\mathbf{x}) = \Pr(x)$, where $\Pr(x)$ is the probability of event x. Otherwise, $\underset{\Delta}{p}(\mathbf{x}) = p(x)\,dx$, where $p(x)$ is the density function. Using this notation, the integral is to be interpretted thusly:

$$\int f(x)\,\underset{\Delta}{p}(x) = \begin{cases} \sum f(x)\Pr(x) & \text{if } \#(X) \text{ is finite or countable} \\ \int f(x)\,p(x)\,dx & \text{otherwise} \end{cases}$$

probability function $\underset{\Delta}{p}(\mathbf{x}, d)$.

Finally, suppose that only the supervisors classification is all that is available to the observer. In this case, the observer will see an i.i.d. random variable characterised by $\underset{\Delta}{p}(d)$.

The important thing to note is that in all cases one could dispense with the element selection/observation process and replace it with a simple random process satisfying the relevant probability function ($\underset{\Delta}{p}(\mathbf{x})$, $\underset{\Delta}{p}(\mathbf{x}, d)$ or $\underset{\Delta}{p}(d)$). Given this, it is not difficult to see that the following scenarios are indistinguishable during training/evaluation:

- An object $\upsilon \in \Upsilon$ is selected randomly according to $P_{\Upsilon}(\upsilon)$. An observation is made of the object to obtain $\mathbf{x} = \mathbf{o}(\upsilon)$, and the supervisor postulates that the object belongs to class $d = \Delta(\upsilon)$.

- Some vector $\mathbf{x} \in \Re^{d_L}$ is selected randomly according to $\underset{\Delta}{p}(\mathbf{x})$, and an associated class $d \in J$ is randomly chosen based on the conditional probability $\underset{\Delta}{p}(d|\, \mathbf{x})$, where:
$$\underset{\Delta}{p}(d|\, \mathbf{x}) = \frac{\underset{\Delta}{p}(\mathbf{x}, d)}{\underset{\Delta}{p}(\mathbf{x})}$$

- A pair $(\mathbf{x}, d) \in \Re^{d_L} \times J$ is randomly chosen based on the probability $\underset{\Delta}{p}(\mathbf{x}, d)$.

Any one of these models may be used at any given time. When the classifier is being used (i.e. no supervisor is present), there are two similarly indistinguishable models, namely:

- An object $\upsilon \in \Upsilon$ is selected randomly according to $P_{\Upsilon}(\upsilon)$ and an observation made to obtain $\mathbf{x} = \mathbf{o}(\upsilon)$.

- A vector $\mathbf{x} \in \Re^{d_L}$ is randomly chosen based on the probability $\underset{\Delta}{p}(\mathbf{x})$.

It will be noted that during the training/evaluation phase the only available classification for any given object is provided by the supervisor, and may not be correct. However, as there is (presumably) no way to obtain a more accurate assessment as to which class an element belongs to, the classification given by the supervisor must be assumed correct, *even though it may not be.* This is important when assessing the operation of a classification machine, and essentially rules out the possibility of perfect performance (unless both noiseless observations and a perfect supervisor are available, which is unlikely in most cases).

Risk and Optimality

Throughout this thesis, it will be assumed that the classifier $\gamma : \Re^{d_L} \to J$ is selected from the parametrised set $T = \left\{ \gamma(\lambda) : \Re^{d_L} \to J \middle| \lambda \in \Lambda \right\}$. So the classifier may be written $\gamma(\lambda) : \Re^{d_L} \to J$ and is completely characterised by the parameter $\lambda \in \Lambda$. Consequently training involves selecting the parameter λ. What is needed is some function $E(\lambda)$ to measure the performance of the classifier for any given λ.

The probability that a classifier $\gamma(\lambda)$ will misclassify a point $\mathbf{x}$ in input space randomly selected according to $\underset{\Delta}{p}(\mathbf{x})$ is:

$$E(\lambda) = \int\limits_{(\mathbf{x},d) \in \Re^{d_L} \times J} \frac{1}{2} \left(1 - \delta_{d,\gamma(\lambda)(\mathbf{x})}\right) \underset{\Delta}{p}(\mathbf{x}, d)$$

which is the performance measure required. It is not, however, the only valid performance measure, nor the most convenient in all cases. More generally, the *risk* associated with the classifier $\gamma(\lambda)$ is defined to be:

$$R(\lambda) = \int\limits_{(\mathbf{x},d) \in \Re^{d_L} \times J} c\left(\mathbf{x}, d, \gamma(\lambda)(\mathbf{x})\right) \underset{\Delta}{p}(\mathbf{x}, d) \tag{2.2}$$

where the function $c : \Re^{d_L} \times J^2 \to \Re$ is called the cost function, and must be integrable for any γ under consideration and satisfy the additional constraint $c(\mathbf{x}, a, a) = 0$. Clearly, if $c(\mathbf{x}, a, b) = \frac{1}{2}(1 - \delta_{a,b})$, $R(\lambda) = E(\lambda)$.

The risk functional $R(\lambda)$ gives a measure of the "badness" (or riskiness) of the classifier $\gamma(\lambda)$ (the higher the risk, the worse the classifier). The *optimal* classifier (or classifiers) is the classifier $\gamma(\lambda)$ which minimises the risk $R(\lambda)$. Specifically, if $\lambda^* = \underset{\lambda \in \Lambda}{\arg\min}\, R(\lambda)$, then the optimal classifier will be $\gamma(\lambda^*)$.

The process of constructing (or training) a classifier is a process of finding a classifier which is as near to optimal as possible, while still being practical (i.e. does not take too long to classify objects, and is not otherwise too large or complex); and completing the construction/training process within an acceptable period of time.

2.2 The Regression Problem

The aim of regression may be stated thusly:

> Given a memoryless time-invariant system H with input $\hat{\mathbf{x}} \in \Re^{d_L}$ and output $z_H \in \Re$, construct a machine to mimic the behaviour of H as closely as possible.

It is assumed that the input $\hat{\mathbf{x}} \in \Re^{d_L}$ is measurable at all times, while the output $z_H \in \Re$ may be measured during training or evaluation. In general, both processes may be prone to noise. Given an input $\hat{\mathbf{x}}$, the result of a noisy observation of this input is denoted $\mathbf{x} = \mathbf{o}(\hat{\mathbf{x}})$, and is a random variable characterised by the conditional probability $\underset{\Delta}{p_{\Xi}}(\mathbf{x}|\,\hat{\mathbf{x}})$. If the measurement process is noiseless, $\underset{\Delta}{p_{\Xi}}(\mathbf{x}|\,\hat{\mathbf{x}}) = \delta_{\mathbf{x},\hat{\mathbf{o}}(\hat{\mathbf{x}})}$ where $\hat{\mathbf{o}} = \mathbf{i} : \Re^{d_L} \to \Re^{d_L}$ is the identity map, and is retained for notational consistency. For simplicity, it will be assumed throughout that $\underset{\Delta}{p_{\Xi}}(\mathbf{x}|\,\hat{\mathbf{x}}) = \delta_{\mathbf{x},\hat{\mathbf{o}}(\hat{\mathbf{x}})}$ (i.e. I will not consider the error-in-variables problem).

The system H is assumed to be constructed as shown in 2.2, where $\hat{f}_H : \Re^{d_L} \to \Re$ is completely deterministic process (and may therefore be treated as a many-to-one map), and the noise is generated according to $\underset{\Delta}{p_{int}}(n|\,\hat{\mathbf{x}})$ (zero mean assumed), where $\hat{\mathbf{x}}$ is the input to H. The output of H is denoted $z_H = f_H(\hat{\mathbf{x}})$, and is a random variable characterised by the conditional probability $\underset{\Delta}{p_H}(z_H|\,\hat{\mathbf{x}})$. If no internal noise is present:

$$\underset{\Delta}{p_H}(z_H|\,\hat{\mathbf{x}}) = \delta_{z_H,\hat{f}_H(\hat{\mathbf{x}})}$$

The result of a noisy measurement of the output z_H of H, denoted $z = o_{out}(z_H)$ is a random variable characterised by the conditional probability density function $\underset{\Delta}{p_{obs}}(z|\,z_H)$. In the noiseless case:

$$\underset{\Delta}{p_{obs}}(z|\,z_H) = \delta_{z,z_H}$$

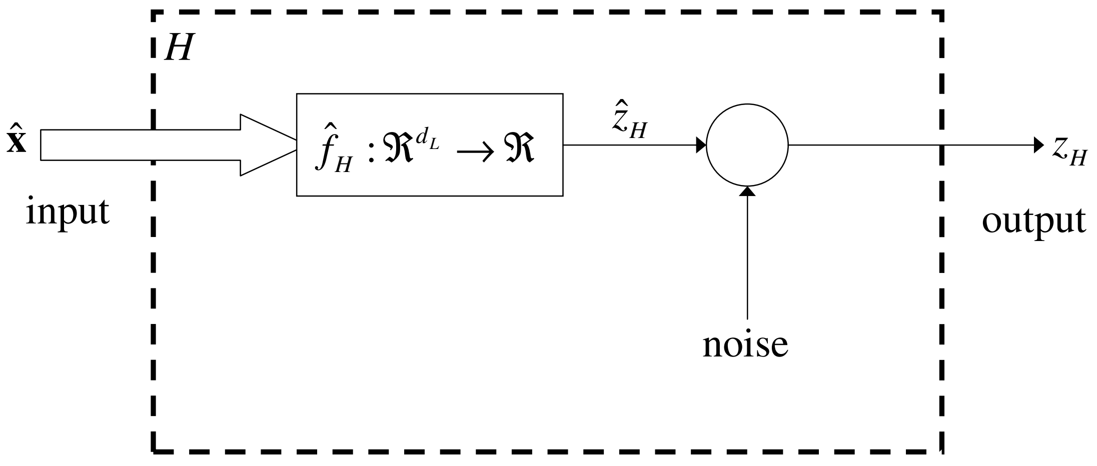


Figure 2.2: Basic regression problem.

Note that all observations of the output of the system H must be done using the noisy observation process described above. Hence there is no way to distinguish between noise produced by the internal noise source, and noise resulting from the noisy observation process. Motivated by this, rather than attempt the to emulate the behaviour of the entire system H, it is preferable to emulate only the deterministic part of H, namely the map $\hat{f}_H : \Re^{d_L} \to \Re$. Defining $\hat{\Delta} = \hat{f}_H$, the regression problem may be re-stated thusly:

> Given a deterministic, noiseless system H with inputs $\hat{\mathbf{x}} \in \Re^{d_L}$ and outputs $\hat{z} \in \Re$, construct a machine to mimic the behaviour of H as closely as possible.

In this formulation the result of a noisy observation of the output $\hat{z}$ (given input $\hat{\mathbf{x}}$) of the system H is a random variable denoted $z = \Delta(\hat{\mathbf{x}})$, which is characterised by the conditional probability function $\underset{\Delta}{p_\Xi}(z|\,\hat{\mathbf{x}})$, where:

$$\underset{\Delta}{p_\Xi}(z|\,\hat{\mathbf{x}}) = \int\limits_{z_H \in \Re} \underset{\Delta}{p_{obs}}(z|\,z_H)\,\underset{\Delta}{p_H}(z_H|\,\hat{\mathbf{x}})$$

Using slightly different notation, the aim of regression may be stated thusly:

> The aim of regression is to construct a function $\gamma : \Re^{d_L} \to \Re$ from input space to a real number, such that $\gamma \circ \hat{\mathbf{o}} = \hat{\Delta}$.

Note the similarities between this problem and the previous statement of the pattern classification problem given at the end of section 2.1.1.

2.2.1 Regression as ∞-class Classification

Consider the regression problem. Define $\Upsilon \subseteq \Re^{d_L}$ to be the set of all permissable inputs $\hat{\mathbf{x}}$ to the system H. For this set of inputs, the image of Υ under the many-to-one map $\hat{\Delta} : \Re^{d_L} \to \Re$ is $J = \hat{\Delta}(\Upsilon)$. It follows that $J \subseteq \Re$.

So, for every permissable input $\upsilon \in \Upsilon$, there will be an associated label (or, loosely speaking, class) $\hat{\Delta}(\upsilon) \in J$. Now, if $\#(J)$ is finite, it is not difficult to see that the regression problem is equivalent to the classification problem discussed in section 2.1, where $\Xi = \Upsilon$ and $\hat{\mathbf{o}} = \mathbf{i}$.

Suppose $\#(J)$ is infinite. In this case, it is still possible to associate a "class" with every "object" $\upsilon \in \Upsilon$ using $z = \hat{\Delta}(\upsilon)$, except that in this case there will be infinitely many such classes. Given this, if the possibility of an infinite set of classes is permitted, it is not difficult to see that sections 2.1.1 and 2.1.2 may be applied directly to the problem of regression, defining $\Xi = \Upsilon$ and $\hat{\mathbf{o}} = \mathbf{i}$. Because $\hat{\mathbf{o}}$ is the identity map, it follows that:

$$\begin{aligned}
&\Upsilon_{?} = \Xi_{?} = \emptyset \\
&\Upsilon_{\checkmark} = \Upsilon = \Xi_{\checkmark} = \Xi \\
&\Upsilon_{\hat{\Delta}_z} = \Upsilon_z = \Xi_{\hat{\Delta}_z} = \Xi_z, z \in J
\end{aligned}$$

Applying the same argument as for the pattern classification case, it is easy to see that, if noiseless observation is possible, and the system itself is noiseless, it should be possible to construct an arbitrarily accurate regressor (although it may not be possible to construct a perfect regressor due to the inherent limitations of hardware. e.g. a computer can only approximate a real output to within finite accuracy).

2.2.2 Risk and Optimal Regression

Suppose that the inputs to the regression system are selected in an i.i.d. manner according to the probability function $\underset{\Delta}{p_{\Upsilon}}(\mathbf{y})$. It follows that:

$$\begin{aligned}
\underset{\Delta}{p}(\mathbf{x}) &= \int\limits_{\mathbf{y}\in\Upsilon} \underset{\Delta}{p_{\Xi}}(\mathbf{x}|\,\mathbf{y})\,\underset{\Delta}{p_{\Upsilon}}(\mathbf{y}) \\
\underset{\Delta}{p}(z) &= \int\limits_{\mathbf{y}\in\Upsilon} \underset{\Delta}{p_{\Xi}}(z|\,\mathbf{y})\,\underset{\Delta}{p_{\Upsilon}}(\mathbf{y}) \\
\underset{\Delta}{p}(\mathbf{x},z) &= \int\limits_{\mathbf{y}\in\Upsilon} \underset{\Delta}{p_{\Xi}}(\mathbf{x}|\,\mathbf{y})\,\underset{\Delta}{p_{\Xi}}(z|\,\mathbf{y})\,\underset{\Delta}{p_{\Upsilon}}(\mathbf{y}) \\
\underset{\Delta}{p}(z|\,\mathbf{x}) &= \frac{\int\limits_{\mathbf{y}\in\Upsilon} \underset{\Delta}{p_{\Xi}}(\mathbf{x}|\mathbf{y})\,\underset{\Delta}{p_{\Xi}}(z|\mathbf{y})\,\underset{\Delta}{p_{\Upsilon}}(\mathbf{y})}{\int\limits_{\mathbf{y}\in\Upsilon} \underset{\Delta}{p_{\Xi}}(\mathbf{x}|\mathbf{y})\,\underset{\Delta}{p_{\Upsilon}}(\mathbf{y})} \\
\underset{\Delta}{p}(\mathbf{x}|\,z) &= \frac{\int\limits_{\mathbf{y}\in\Upsilon} \underset{\Delta}{p_{\Xi}}(\mathbf{x}|\mathbf{y})\,\underset{\Delta}{p_{\Xi}}(z|\mathbf{y})\,\underset{\Delta}{p_{\Upsilon}}(\mathbf{y})}{\int\limits_{\mathbf{y}\in\Upsilon} \underset{\Delta}{p_{\Xi}}(z|\mathbf{y})\,\underset{\Delta}{p_{\Upsilon}}(\mathbf{y})}
\end{aligned}$$

Note that this is notationally identical to the pattern recognition problem, except that d has been replaced by z. Just as happened in the pattern classification problem, for the regression problem observations may be generated in a number of indistinguishable ways. The risk associated with a regressor is:

$$R(\lambda) = \int\limits_{(\mathbf{x},z)\in\Re^{d_L}\times\Re} c(\mathbf{x},z,\gamma(\lambda)(\mathbf{x}))\,\underset{\Delta}{p}(\mathbf{x},z)$$

where once again it is assumed that the regressor γ is selected from a set of possible regressors, $T = \left\{\gamma(\lambda) : \Re^{d_L} \to \Re\right|\,\lambda\in\Lambda\big\}$, indexed by $\lambda \in \Lambda$. The function $c : \Re^{d_L} \times \Re^2 \to \Re$ is the cost function, as before. For example, if $c(\mathbf{x},x,y) = \frac{1}{2}(x-y)^2$, $R(\lambda)$ will be the mean-squared error. As for pattern classification, the optimal regressor (or regressors) is the regressor $\gamma(\lambda)$ that has the smallest risk associated with it. So if $\lambda^* = \underset{\lambda\in\Lambda}{\arg\min}\, R(\lambda)$, the optimal regressor is $\gamma(\lambda^*)$.

Chapter 3

THE LEARNING PARADIGM

Science may be described as the art of systematic over-simplification.

– Karl Popper

In the previous chapter, the problems of pattern classification and regression were introduced. The regression problem was shown to be equivalent to pattern classification where the set of classes was infinite, allowing many of the concepts from pattern classification to be used directly in the context of regression.

Risk was also introduced as a measure of the "badness" (riskiness/error) of a given classifier/regressor. However, the definition given is of little practical use in most cases, as evaluation of the risk requires knowledge of (and access to) not only the entire set of objects that are to be classified (the set Υ), but also the classification of each of these objects.

3.1 The Learning Paradigm

This thesis is based on the idea of learning from data. The usual model for this kind of learning is shown in figure 3.1. The elements of the model are:

1. The generator of samples, Σ.

2. The observer, O.

3. The target generator (supervisor), S.

4. The learning machine, LM.

The generator Σ takes samples from the set Υ in an i.i.d. manner according to the probability distribution $P_\Upsilon(\upsilon)$. Based on this sample, the observer O generates

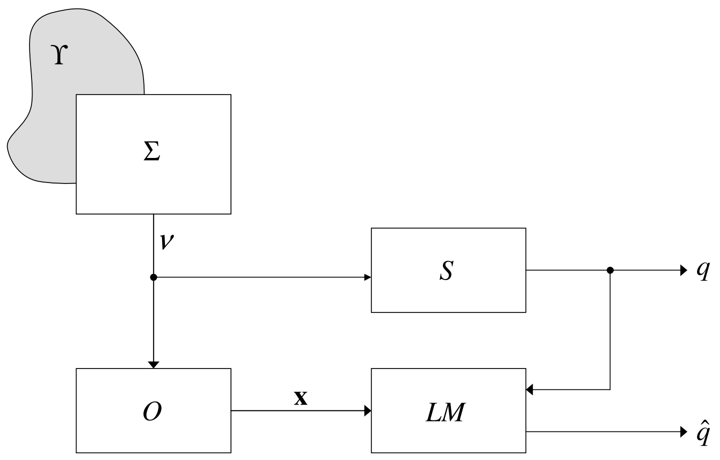


Figure 3.1: The learning model. During training the learning machine LM observes pairs $(\mathbf{x}, q)$, which are generated from the sampled object υ using the noisy observation function $\mathbf{o}$ (generated by the observer, O) and the supervision function Δ (generated by the supervisor S). After training the machine will attempt to classify any given object ω based on the observation $\mathbf{x}$ of ω, giving the result $\hat{q}$.

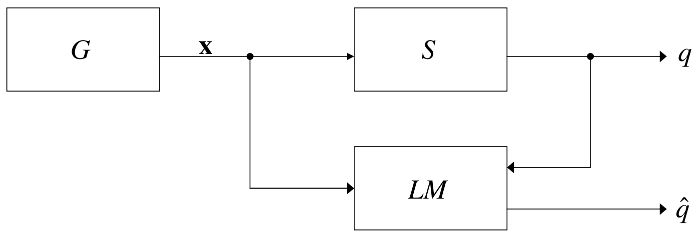


Figure 3.2: Simplified learning model ([95]). The operation is essentially identical to figure 3.1, except that the sampling and observation processes are not shown.

a noisy observation $\mathbf{x} = \mathbf{o}(\upsilon) \in \Re^{d_L}$ using the conditional probability distribution, $P_{\Xi}(\mathbf{x}|\, \upsilon)$. Finally, the target generator (supervisor) provides a possible classification (output in the regression setting) $q = \Delta(\upsilon) \in J$ for the object, which is taken to be correct, according to $P_{\Xi}(q|\, \upsilon)$.

As the action of the sample generator in figure 3.1 is invisible to the learning machine, it is usual to consider the simplified version of this schema shown in figure 3.2. In this case the training data is generated randomly by the generator G in accordance with the probability function $\underset{\Delta}{p}(\mathbf{x})$. The target generator then generates the classification (or output) q in accordance with the conditional probability function $\underset{\Delta}{p}(q|\, \mathbf{x})$.

During the *training phase*, the LM will accumulate N training pairs:

$$\mathbf{Y} = \left(\left(\mathbf{x}_1, q_1 \right), \left(\mathbf{x}_2, q_2 \right), \ldots, \left(\mathbf{x}_N, q_N \right) \right)$$

where each pair $(\mathbf{x}_i, q_i) \in \Re^{d_L} \times J$ contains a noisy observation $\mathbf{x}_i$ of some object υ generated by the observer O and a corresponding classification (output) q_i of that object generated by the supervisor S. Based on this training set, the aim is to construct a machine to mimic the behaviour of the supervisor S. Mathematically speaking, the aim is to find the classifier $\gamma\left(\lambda^*\right) \in \left\{ \gamma\left(\lambda\right) : \Re^{d_L} \rightarrow J \middle| \, \lambda \in \Lambda \right\}$ that minimises the risk functional, $R\left(\lambda\right)$, using the information contained in the training set $\mathbf{Y}$. The classifier thus selected is called the *trained machine.*

Notation

For classification, the training set is defined as:

$$\begin{aligned}
\mathbf{Y} &= \left(\left(\mathbf{x}_1, d_1 \right), \left(\mathbf{x}_2, d_2 \right), \ldots, \left(\mathbf{x}_N, d_N \right) \right) \\
\mathbf{X} &= \left(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \right) \\
\mathbf{D} &= \left(d_1, d_2, \ldots, d_N \right) \\
\mathbf{x}_i &\in \Re^{d_L} \\
d_i &\in J \subset \mathbb{Z}
\end{aligned}$$

Likewise, for regression:

$$\begin{aligned}
\mathbf{Y} &= \left(\left(\mathbf{x}_1, z_1 \right), \left(\mathbf{x}_2, z_2 \right), \ldots, \left(\mathbf{x}_N, z_N \right) \right) \\
\mathbf{X} &= \left(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \right) \\
\mathbf{Z} &= \left(z_1, z_2, \ldots, z_N \right) \\
\mathbf{x}_i &\in \Re^{d_L} \\
z_i &\in J \subset \Re
\end{aligned}$$

When referring to either or both:

$$\begin{aligned}
\mathbf{Y} &= \left(\left(\mathbf{x}_1, q_1 \right), \left(\mathbf{x}_2, q_2 \right), \ldots, \left(\mathbf{x}_N, q_N \right) \right) \\
\mathbf{X} &= \left(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \right) \\
\mathbf{Q} &= \left(q_1, q_2, \ldots, q_N \right) \\
\mathbf{x}_i &\in \Re^{d_L} \\
q_i &\in J
\end{aligned}$$

where q and $\mathbf{Q}$ are understood to be either d and $\mathbf{D}$ for the classification problem or z and $\mathbf{Z}$ for the regression problem, respectively.

3.2 A Risk Minimisation Approach

3.2.1 Risk, Empirical Risk and the Regularisation

As stated previously, the optimal classifier $\gamma\left(\lambda^*\right) \in \left\{ \gamma\left(\lambda\right) : \Re^{d_L} \to J \middle| \lambda \in \Lambda \right\}$ is the classifier that minimises the risk functional, $R\left(\lambda\right)$. Unfortunately, neither $\underset{\Delta}{p}\left(\mathbf{x}, q\right)$ nor $\Delta : \Re^{d_L} \to J$ is known (obviously, or the problem would be trivial), so it is not possible to directly calculate the appropriate λ to minimise (2.2). Hence (2.2) is of little practical help in finding λ. An alternative way of tackling this problem is to minimise an approximation of the actual risk based on the information contained in the training set. This empirical risk, $R_{emp}\left(\lambda\middle|\mathbf{Y}\right)$, is defined as follows:

$$R_{emp}\left(\lambda\middle|\mathbf{Y}\right) = \frac{1}{N} \sum_{\left(\mathbf{x}_i, q_i\right) \in \mathbf{Y}} c\left(\mathbf{x}_i, q_i, \gamma\left(\lambda\right)\left(\mathbf{x}_i\right)\right) \tag{3.1}$$

Note that $R\left(\lambda\right) = R_{emp}\left(\lambda \middle| \left\{ \left(\upsilon, \hat{\Delta}\left(\upsilon\right)\right) \middle| \upsilon \in \Upsilon \right\}\right)$.

However, it turns out that selecting λ by directly minimising the empirical risk $R_{emp}\left(\lambda\middle|\mathbf{Y}\right)$ does not ensure that the actual risk $R\left(\lambda\right)$ is small (unless N is very large, which is not generally the case). The difficulty is the potential for overfitting, which occurs when the set of possible training functions $T = \left\{ \gamma\left(\lambda\right) : \Re^{d_L} \to J \middle| \lambda \in \Lambda \right\}$ has too much capacity (a concept that will be quantified later) for the problem at hand, resulting in a classifier that generalises badly. To overcome this difficulty, one can select λ to minimise a *regularised* risk, $R_{reg}\left(\lambda|\,\mathbf{Y}\right)$:

$$R_{reg}\left(\lambda|\,\mathbf{Y}\right) = CR_{emp}\left(\lambda|\,\mathbf{Y}\right) + \phi\left(\lambda\right) \tag{3.2}$$

where the term $\phi\left(\lambda\right)$ is a measure of the capacity of some subset $T^* \subset T$ of all possible classifiers, where $\gamma\left(\lambda\right) \in T^*$. In the regression case, $\phi\left(\lambda\right)$ is known as a regularisation term. The parameter C controls the trade-off between simplicity (small $\phi\left(\lambda\right)$ term, caused by selecting C small) and empirical accuracy (small empirical risk, caused by selecting C large).

3.3 A Maximum Likelihood Approach

Note that the risk $R\left(\lambda\right)$ associated with a classifier or regressor $\gamma\left(\lambda\right)$ is just the expectation of the cost function given $(\mathbf{x}, q)$ drawn from $\underset{\Delta}{p}\left(\mathbf{x}, q\right)$. Likewise, the empirical risk $R_{emp}\left(\lambda|\,\mathbf{Y}\right)$ is an approximation of this based on a finite set $\mathbf{Y}$ of examples drawn from the same distribution. Mathematically:

$$\begin{aligned} R\left(\lambda\right) &= \mathbf{E}\left(c\left(\mathbf{x}, q, \gamma\left(\lambda\right)\left(\mathbf{x}\right)\right)\middle|\, \underset{\Delta}{p}\left(\mathbf{x}, q\right)\right) \\ R_{emp}\left(\lambda|\,\mathbf{Y}\right) &= \mathbf{E}\left(c\left(\mathbf{x}, q, \gamma\left(\lambda\right)\left(\mathbf{x}\right)\right)|\left(\mathbf{x}, q\right) \in \mathbf{Y}\right) \end{aligned}$$

The approach to the learning problem discussed up until now has been to minimise the empirical risk approximation (possibly with an additional regularisation term to prevent overfitting), thus hopefully minimising the actual risk. The present section considers an alternative approach based on the statistical concept of maximum likelihood. While this is not directly applicable to the SVM method, in later chapters the theory underpinning this approach will be used to tackle the problem of optimal parameter selection.

The following section follows closely the material in [76].

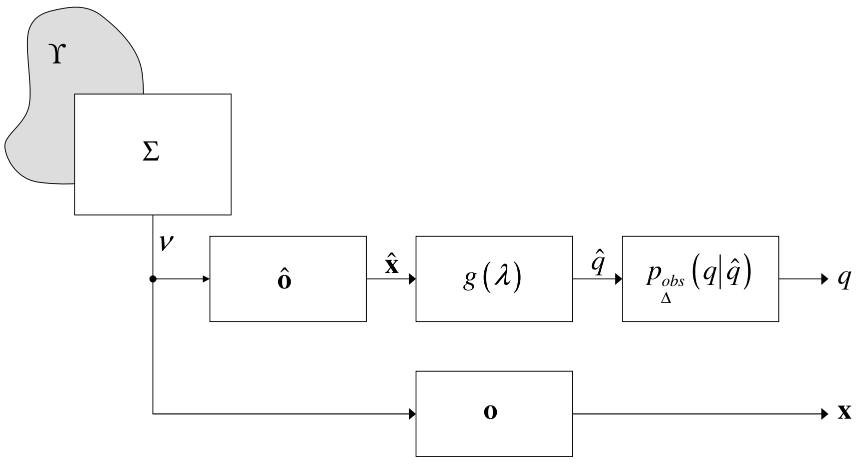


Figure 3.3: Experimental setup for statistical learning.

3.3.1 Maximum Likelihood Estimation

Consider the training set $\mathbf{Y}$. Given a machine $\gamma(\lambda)$ and experimental setup as in figure 3.3, the probability of this experimental setup generating these results is $\underset{\Delta}{p}(\mathbf{Y}|\,\lambda)$, and may be infinitesimal. The aim of maximum likelihood estimation is to find the value of λ (and hence also the machine $\gamma(\lambda)$) that is most likely to have resulted in this setup generating the training set $\mathbf{Y}$. Mathematically, the likelihood of a training set $\mathbf{Y}$ being generated by the setup shown in figure 3.3 is:

$$\begin{aligned}
\underset{\Delta}{p}(\mathbf{Y}|\,\lambda) &= \prod_{(\mathbf{x}_i,q_i)\in\mathbf{Y}} \underset{\Delta}{p}(q_i,\mathbf{x}_i|\,\lambda) \\
&= \prod_{(\mathbf{x}_i,q_i)\in\mathbf{Y}} \underset{\Delta}{p}(q_i|\,\mathbf{x}_i,\lambda)\,\underset{\Delta}{p}(\mathbf{x}_i)
\end{aligned}$$

where $\underset{\Delta}{p}(q_i|\,\mathbf{x}_i,\lambda)$ is the probability of observing output q_i, given observed input $\mathbf{x}_i$ and machine parameters λ using experimental setup as per figure 3.3. The aim of the maximum likelihood approach is to find λ^* to maximise $\underset{\Delta}{p}(\mathbf{Y}|\,\lambda^*)$.

This expression may be re-written:

$$\begin{aligned}
\underset{\Delta}{p}(\mathbf{Y}|\,\lambda) &= \prod_{(\mathbf{x}_i,q_i)\in\mathbf{Y}} \underset{\Delta}{p}(q_i|\,\mathbf{x}_i,\lambda) \prod_{(\mathbf{x}_i,q_i)\in\mathbf{Y}} \underset{\Delta}{p}(\mathbf{x}_i) \\
&= \left(\prod_{(\mathbf{x}_i,q_i)\in\mathbf{Y}} \underset{\Delta}{p}(\mathbf{x}_i)\right) \exp\left(\ln \prod_{(\mathbf{x}_i,q_i)\in\mathbf{Y}} \underset{\Delta}{p}(q_i|\,\mathbf{x}_i,\lambda)\right) \\
&= c_0 \exp\left(\sum_{(\mathbf{x}_i,q_i)\in\mathbf{Y}} \ln\left(\underset{\Delta}{p}(q_i|\,\mathbf{x}_i,\lambda)\right)\right)
\end{aligned}$$

where c_0 is a constant dependant on $\underset{\Delta}{p}(\mathbf{x})$, and may be neglected for simplicity. Hence to maximise the likelihood of a training set, one can instead minimise the *log-likelihood* for that training set, which is defined as:

$$\underset{\Delta}{\mathcal{L}}[\lambda|\,\mathbf{Y}] = \sum_{(\mathbf{x}_i,q_i)\in\mathbf{Y}} -\ln\left(\underset{\Delta}{p}(q_i|\,\mathbf{x}_i,\lambda)\right)$$

Regression

As defined here, in the regression problem the training set $\mathbf{Z}$ is related to $\mathbf{X}$ by the functional relationship $z_i = \hat{f}_H(\mathbf{x}_i) + \xi_i$ for all $1 \leq i \leq N$, where ξ_i is a noise term generated according to $\underset{\Delta}{p_{int}}(\xi_i|\,\mathbf{x}_i)$. If the noise is not dependent on the input $\mathbf{x}$ then $\underset{\Delta}{p_{int}}(\xi|\,\mathbf{x})$ may be re-written $\underset{\Delta}{p_{int}}(\xi)$, and the log-likelihood expressed thusly:

$$\underset{\Delta}{\mathcal{L}}[\lambda|\,\mathbf{Y}] = \sum_{(\mathbf{x}_i,z_i)\in\mathbf{Y}} -\ln\left(\underset{\Delta}{p_{int}}(z_i - \gamma(\lambda)(\mathbf{x}_i))\right)$$

and will be infinitesimal in most cases. Consider the definition of empirical risk, (3.1). If the cost function used in this expression is:

$$c(\mathbf{x},y,z) = -\ln\left(\underset{\Delta}{p_{int}}(y-z)\right)$$

then:

$$R_{emp}(\lambda|\,\mathbf{Y}) = \underset{\Delta}{\mathcal{L}}[\lambda|\,\mathbf{Y}]$$

Note, however, that this may result in an infinitesimal cost and empirical risk (which could be awkward from a computational standpoint). In this case, the associated density function $p_{int}(\xi)$ may be used in place of the probability function $\underset{\Delta}{p_{int}}(\xi)$. So:

$$\begin{aligned}
c(\mathbf{x},y,z) &= -\ln(p_{int}(y-z)) \\
\mathcal{L}[\lambda|\,\mathbf{Y}] &= \sum_{(\mathbf{x}_i,z_i)\in\mathbf{Y}} -\ln(p_{int}(z_i - \gamma(\lambda)(\mathbf{x}_i))) \\
R_{emp}(\lambda|\,\mathbf{Y}) &= \mathcal{L}[\lambda|\,\mathbf{Y}]
\end{aligned}$$

Table 3.1 and figure 3.4 show some typical cost functions and the associated

Table 3.1: Cost functions and associated maximum-likelihood density functions.

	Cost function $c(\mathbf{x}, y, z)$	Density function $p(\xi)$
ϵ-insensitive	$\lvert y-z \rvert_\epsilon$	$\frac{1}{2(1+\epsilon)} e^{-\lvert\xi\rvert_\epsilon}$
Laplacian	$\lvert y-z \rvert$	$\frac{1}{2} e^{-\lvert\xi\rvert}$
ϵ-insensitive polynomial	$\frac{1}{n} \lvert y-z \rvert_\epsilon^n$	$\frac{1}{2\left(n^{\frac{1}{n}-1}\Gamma\left(\frac{1}{n}\right)+\epsilon\right)} e^{-\frac{1}{n}\lvert\xi\rvert_\epsilon^n}$
Polynomial	$\frac{1}{n} \lvert y-z \rvert^n$	$\frac{1}{2n^{\frac{1}{n}-1}\Gamma\left(\frac{1}{n}\right)} e^{-\frac{1}{n}\lvert\xi\rvert^n}$
Gaussian	$\frac{1}{2}(y-z)^2$	$\frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}\xi^2}$
Piecewise polynomial	$\begin{cases} \frac{1}{n\epsilon^{n-1}} \lvert y-z \rvert^n & \text{if } \lvert y-z \rvert \leq \epsilon \\ \lvert y-z \rvert - \frac{(n-1)}{n}\epsilon & \text{otherwise} \end{cases}$	$\propto \begin{cases} e^{-\frac{1}{n\epsilon^{n-1}}\lvert\xi\rvert^n} & \text{if } \lvert\xi\rvert \leq \epsilon \\ e^{-\lvert\xi\rvert+\frac{(n-1)}{n}\epsilon} & \text{otherwise} \end{cases}$
Huber's robust loss	$\begin{cases} \frac{1}{2\epsilon}(y-z)^2 & \text{if } \lvert y-z \rvert \leq \epsilon \\ \lvert y-z \rvert - \frac{1}{2}\epsilon & \text{otherwise} \end{cases}$	$\propto \begin{cases} e^{-\frac{1}{2\epsilon}\xi^2} & \text{if } \lvert\xi\rvert \leq \epsilon \\ e^{-\lvert\xi\rvert+\frac{1}{2}\epsilon} & \text{otherwise} \end{cases}$

maximum likelihood density functions (source, [76]).

Classification

As for regression, an appropriate choice of cost function will result in a risk minimisation based classifier that will "mimic" the behaviour of a maximum-likelihood approach. Specifically, using the cost function:

$$c(\mathbf{x}, d, e) = -\ln(\Pr(d|\, e))$$

where $\Pr(d|\, e)$ is the probability that the $\gamma(\lambda)(\mathbf{x}) = d$ given that $\mathbf{x}$ is selected according to $\underset{\Delta}{p}(\mathbf{x}|\, e)$ (in words, the probability that the machine will say an object belongs to class d, given that the classifier (trainer) has said the same object belongs to class e). Using this cost function, it follows that:

$$R_{emp}(\lambda|\, \mathbf{Y}) = \mathcal{L}[\lambda|\, \mathbf{Y}]$$

A common model for $\Pr(d|\, e)$ in the binary case is the logistic model, where:

$$\begin{aligned} \Pr(1|\, d) &= \tfrac{e^d}{1+e^d} \\ \Pr(-1|\, d) &= \tfrac{1}{1+e^d} \end{aligned}$$

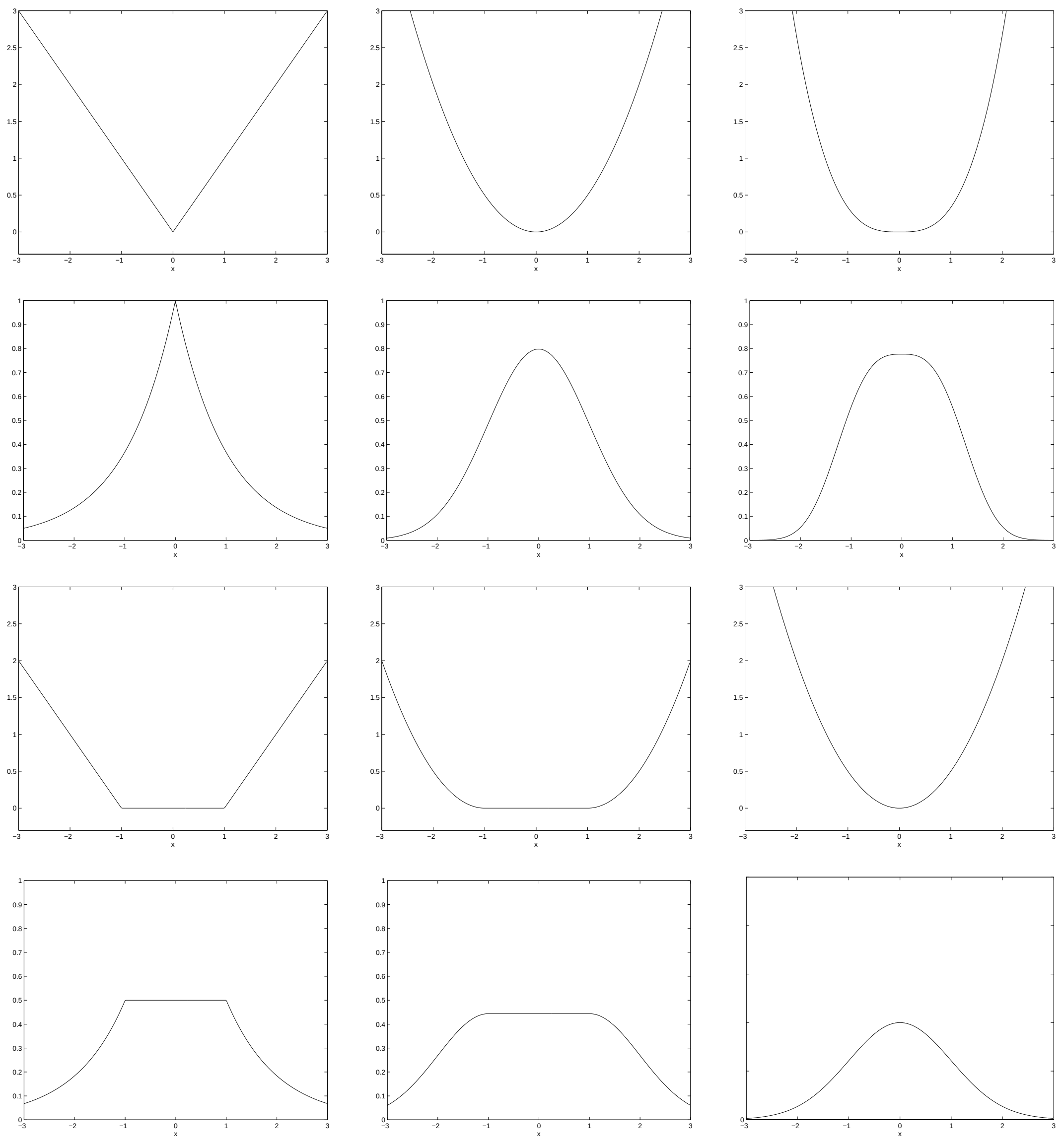


Figure 3.4: Typical cost functions and associated density functions. Shown are: linear, quadratic and cubic cost (top row, left to right), with associated density shown below; and linear and quadratic ϵ-insensitive cost where $\epsilon = 1$ and finally Huber's robust loss function with $\epsilon = 1$ (second bottom row, left to right), again with associated density shown below.

3.3.2 Efficiency

In the previous subsection it was shown that maximum likelihood estimation can be reduced to risk minimisation provided that an appropriate loss function is selected. However, this assumes that the noise model is known, which is not true in general. Even if the noise model is known, the resultant loss function may not be useful in-so-far as it may have local minima, or be "difficult" in some other way. Hence there is a need for some other way of comparing the performance of different loss functions for different noise models. Asymptotic efficiency is one such method.

Assume the training set $\mathbf{Y}$ is drawn in an i.i.d. method based on the probability function $\underset{\Delta}{p}\left(\mathbf{y}\middle|\,\theta\right)$ (this does not imply that $\underset{\Delta}{p}\left(\mathbf{y}\middle|\,\theta\right)$ is only dependent on θ, just that it is dependent on θ among other parameters). Let $\hat{\theta}\left(\mathbf{Y}\right)$ be an estimator of the parameters θ based on the training set $\mathbf{Y}$.

For notational simplicity, define:

$$\mathbf{E}_\theta\left[\xi\left(\mathbf{Y}\right)\right] = \int\limits_{\mathbf{Y}} \xi\left(\mathbf{Y}\right) \underset{\Delta}{p}\left(\mathbf{Y}\middle|\,\theta\right)$$

as the expectation of a random variable $\xi\left(\mathbf{Y}\right)$ with respect to $\underset{\Delta}{p}\left(\mathbf{Y}\middle|\,\theta\right)$. An unbiased estimator is defined as an estimator satisfying:

$$\mathbf{E}_\theta\left(\hat{\theta}\left(\mathbf{Y}\right)\right) = \theta$$

Given an unbiased estimator, one measure of the goodness of that estimator is the variance of the estimator. If the variance is low, the probability that the error in the estimate $\hat{\theta}\left(\mathbf{Y}\right)$ of θ will be large should also be low.

The score $V_\theta\left(\mathbf{Y}\right)$ of $\underset{\Delta}{p}\left(\mathbf{Y}\middle|\,\theta\right)$ is defined as:

$$V_\theta\left(\mathbf{Y}\right) = \frac{\partial \ln \underset{\Delta}{p}\left(\mathbf{Y}\middle|\theta\right)}{\partial\theta}$$

The score gives a measure of how dependent the data is on the components of the parameter set θ, and hence how much the data will affect the choice of θ when using the estimator $\hat{\theta}\left(\mathbf{Y}\right)$. The Fisher information matrix $\mathbf{I}$ is the covariance of this score. If the individual components of θ are written θ_i (i.e. treat θ as a vector) then:

$$I_{i,j} = \mathbf{E}_\theta \left[\left(\frac{\partial \ln \underset{\Delta}{p}(\mathbf{Y}|\theta)}{\partial \theta_i} - \mathbf{E}_\theta \left[\frac{\partial \ln \underset{\Delta}{p}(\mathbf{Y}|\theta)}{\partial \theta_i} \right] \right) \left(\frac{\partial \ln \underset{\Delta}{p}(\mathbf{Y}|\theta)}{\partial \theta_j} - \mathbf{E}_\theta \left[\frac{\partial \ln \underset{\Delta}{p}(\mathbf{Y}|\theta)}{\partial \theta_j} \right] \right) \right]$$

Noting that:

$$\begin{aligned}
\mathbf{E}_\theta \left[V_\theta (\mathbf{Y}) \right] &= \int\limits_{\mathbf{Y}} V_\theta (\mathbf{Y}) \underset{\Delta}{p} (\mathbf{Y}|\, \theta) \\
&= \int\limits_{\mathbf{Y}} \frac{\partial \left(\ln \underset{\Delta}{p}(\mathbf{Y}|\theta) \right)}{\partial \theta} \underset{\Delta}{p} (\mathbf{Y}|\, \theta) \\
&= \int\limits_{\mathbf{Y}} \frac{\partial \underset{\Delta}{p}(\mathbf{Y}|\theta)}{\partial \theta} \\
&= \frac{\partial}{\partial \theta} \int\limits_{\mathbf{Y}} \underset{\Delta}{p} (\mathbf{Y}|\, \theta) \\
&= \frac{\partial}{\partial \theta} 1 \\
&= 0
\end{aligned}$$

It follows that:

$$I_{i,j} = \mathbf{E}_\theta \left[\frac{\partial \ln \underset{\Delta}{p}(\mathbf{Y}|\theta)}{\partial \theta_i} \frac{\partial \ln \underset{\Delta}{p}(\mathbf{Y}|\theta)}{\partial \theta_j} \right]$$

The *covariance matrix* $\mathbf{B}$ of $\hat{\theta}(\mathbf{Y})$ is:

$$B_{i,j} = \mathbf{E}_\theta \left[\left(\hat{\theta}_i (\mathbf{Y}) - \mathbf{E}_\theta \left[\hat{\theta}_i (\mathbf{Y}) \right] \right) \left(\hat{\theta}_j (\mathbf{Y}) - \mathbf{E}_\theta \left[\hat{\theta}_j (\mathbf{Y}) \right] \right) \right]$$

This covariance provides a useful measure of the "goodness" of the estimator, as described previously. The Cramer-Rao bound states that, for any unbiased estimator $\hat{\theta}(\mathbf{Y})$:

$$\det (\mathbf{IB}) \geq 1 \tag{3.3}$$

Now, a good unbiased estimator should have minimal variance. Hence the smaller $|\mathbf{IB}|$ is, the better. This leads to the definition of the *efficiency* of an estimator, namely:

$$e = \det (\mathbf{IB})^{-1} \leq 1$$

The larger the efficiency, the lower the variance of the estimator and hence the better the estimator. The key advantage of this form is that e can be computed efficiently (see [60], lemma 3). Assuming:

$$\begin{aligned}\hat{\theta}(\mathbf{Y}) &= \arg\min_{\theta} d(\mathbf{Y},\theta) \\ &= \arg\min_{\theta} \sum_{i=1}^{N} d(\mathbf{y}_i,\theta)\end{aligned}$$

where $d(\mathbf{Y},\theta)$ is a twice differentiable function of θ then as $N \to \infty$:

$$e = \frac{|\mathbf{Q}|^2}{|\mathbf{IG}|} \tag{3.4}$$

where:

$$\begin{aligned}G_{i,j} &= \mathbf{E}_\theta\left[\frac{\partial d\left(\mathbf{Y},\hat{\theta}\right)}{\partial \hat{\theta}_i}\frac{\partial d\left(\mathbf{Y},\hat{\theta}\right)}{\partial \hat{\theta}_j}\right] \\ Q_{i,j} &= \mathbf{E}_\theta\left[\frac{\partial^2 d\left(\mathbf{Y},\hat{\theta}\right)}{\partial \hat{\theta}_i \partial \hat{\theta}_j}\right]\end{aligned}$$

In other words, given such an estimator, the asymptotic efficiency can be calculated directly from $d(\mathbf{Y},\theta)$. It can be shown that for a maximum likelihood estimator ($d\left(\mathbf{Y},\hat{\theta}\right) = \ln \underset{\Delta}{p}(\mathbf{Y}|\,\theta)$) the asymptotic efficiency is $e = 1$. However, this result is only useful if the noise model is known.

If there is only one parameter:

$$\begin{aligned}I &= \int_{\mathbf{Y}} \left(\frac{\partial \ln \underset{\Delta}{p}(\mathbf{Y}|\theta)}{\partial \theta}\right)^2 \underset{\Delta}{p}(\mathbf{Y}|\,\theta) \\ G &= \int_{\mathbf{Y}} \left(\frac{\partial d(\mathbf{Y},\theta)}{\partial \theta}\right)^2 \underset{\Delta}{p}(\mathbf{Y}|\,\theta) \\ Q &= \int_{\mathbf{Y}} \frac{\partial^2 d(\mathbf{Y},\theta)}{\partial \theta^2} \underset{\Delta}{p}(\mathbf{Y}|\,\theta)\end{aligned}$$

Noting that:

$$\underset{\Delta}{p}(\mathbf{Y}|\,\lambda) = \prod_{\mathbf{y}\in\mathbf{Y}} \underset{\Delta}{p}(\mathbf{y}_i|\,\lambda)$$

it follows that [76]:

$$\begin{aligned}I &= N\int_{\mathbf{y}} \left(\frac{\partial \ln \underset{\Delta}{p}(\mathbf{y}|\theta)}{\partial \theta}\right)^2 \underset{\Delta}{p}(\mathbf{y}|\,\theta) \\ G &= N\int_{\mathbf{y}} \left(\frac{\partial d(\mathbf{y},\theta)}{\partial \theta}\right)^2 \underset{\Delta}{p}(\mathbf{y}|\,\theta) \\ Q &= N\int_{\mathbf{y}} \frac{\partial^2 d(\mathbf{y},\theta)}{\partial \theta^2} \underset{\Delta}{p}(\mathbf{y}|\,\theta)\end{aligned} \tag{3.5}$$

Note that the factor N in (3.5) will cancel in the expression (3.4) for the efficiency e. Also, it will assumed in later chapters that all distributions will be continuous. So, for the purposes of this thesis, (3.5) may be re-written:

$$\begin{aligned} I &= \int\limits_{\mathbf{y}} \left(\tfrac{\partial \ln p(\mathbf{y}|\theta)}{\partial \theta} \right)^2 p\left(\mathbf{y}\right|\theta) d\mathbf{y} \\ G &= \int\limits_{\mathbf{y}} \left(\tfrac{\partial d(\mathbf{y},\theta)}{\partial \theta} \right)^2 p\left(\mathbf{y}\right|\theta) d\mathbf{y} \\ Q &= \int\limits_{\mathbf{y}} \tfrac{\partial^2 d(\mathbf{y},\theta)}{\partial \theta^2} p\left(\mathbf{y}\right|\theta) d\mathbf{y} \end{aligned} \tag{3.6}$$

Chapter 4

SUPPORT VECTOR MACHINE BASICS

A child of five could understand this. Fetch me a child of five. I can't make heads or tails of it.

- Groucho Marx

4.1 Support Vector Machine Classifiers

THE simplest form of pattern classification is binary (or 2 class) pattern classification. For notational simplicity, assume the label set is $J = \{\pm 1\}$. The observation map $\hat{\mathbf{o}} : \Upsilon \to \Re^{d_L}$ maps objects to points in d_L dimensional input space. Extending the notation of chapter 2, each point $\mathbf{x}$ in input space may be assigned a label $d = \hat{\Delta}_{\Xi}(\mathbf{x})$ from the set $\bar{J} = \{\pm 1, 0\}$, such that:

$$\hat{\Delta}_{\Xi}(\mathbf{x}) = \begin{cases} +1 & \text{if } \mathbf{x} \in \Xi_{+1} \\ -1 & \text{if } \mathbf{x} \in \Xi_{-1} \\ 0 & \text{if } \mathbf{x} \notin \Xi_{\checkmark} \end{cases}$$

4.1.1 Separating Hyperplanes and Feature Space

One possible set of classifiers is the set of all functions of the form:

$$\gamma(\lambda)(\mathbf{x}) = \operatorname{sgn}(g(\lambda)(\mathbf{x}))$$

where:

$$g(\lambda)(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

and the parameter set is $\lambda = (\mathbf{w}, b) \in \Lambda = \left\{ (\mathbf{w}, b) \in \Re^{d_L+1} \middle| \mathbf{w} \neq \mathbf{0} \right\}$. In this case, any given selection of parameters $\lambda \in \Lambda$ defines an oriented hyperplane (decision surface) in input space, namely $\{\mathbf{x}| g(\lambda)(\mathbf{x}) = 0\}$. The two sets $\Xi_{\hat{\Delta}_{+1}}$ and $\Xi_{\hat{\Delta}_{-1}}$ are said to be *linearly separable* if, for some $\lambda \in \Lambda$, $\hat{\Delta}_{\Xi}(\mathbf{x}) g(\lambda)(\mathbf{x}) > 0$ for all $\mathbf{x} \in \Xi$ (or, equivalently, $\hat{\Delta}_{\Xi}(\mathbf{x}) = \gamma(\lambda)(\mathbf{x})$ for all $\mathbf{x} \in \Xi$). Geometrically speaking, this means that the oriented hyperplane defined by λ shatters $\Xi_{\hat{\Delta}_{+1}}$ and $\Xi_{\hat{\Delta}_{-1}}$. Note that such a plane can only exist if $\Xi \neq \Re^{d_L}$ and $\Xi_{?} = \emptyset$.

More generally, if $\Xi_{\hat{\Delta}_{+1}}$ and $\Xi_{\hat{\Delta}_{-1}}$ are not linearly separable then it may be possible to define a map $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$ into some d_H-dimensional *feature space* such that $\boldsymbol{\varphi}\left(\Xi_{\hat{\Delta}_{+1}}\right)$ and $\boldsymbol{\varphi}\left(\Xi_{\hat{\Delta}_{-1}}\right)$ are linearly separable in feature space. In other words, using the set of classifiers of the form:

$$g(\lambda)(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) + b \tag{4.1}$$

(implicitly ignoring all but the sign of the output) it may be possible to find some $\lambda \in \Lambda$ such that $\hat{\Delta}_{\Xi}(\mathbf{x}) g(\lambda)(\mathbf{x}) > 0$ for all $\mathbf{x} \in \Xi$. This may be thought of as either being a hyperplane in d_H-dimensional feature space or a non-linear dividor in input space. In this case, $\Xi_{\hat{\Delta}_{+1}}$ and $\Xi_{\hat{\Delta}_{-1}}$ are said to be *nonlinearly separable* under the map $\boldsymbol{\varphi}$. Once again, this is possible only if $\Xi \neq \Re^{d_L}$ and $\Xi_{?} = \emptyset$.

4.1.2 The Separable Case

Consider a set of training data consisting of a set of points in $\Re^{d_L}$ dimensional input space, labelled in a binary manner, ie:

$$\begin{aligned}
&\mathbf{Y} = \{(\mathbf{x}_1, d_1), (\mathbf{x}_2, d_2), \ldots, (\mathbf{x}_N, d_N)\} \\
&\mathbf{x}_i \in \Re^{d_L} \\
&d_i \in \{+1, -1\}
\end{aligned} \tag{4.2}$$

Define:

$$\begin{aligned}
\mathbf{Y}_{\geq} &= \{\mathbf{x}_i | (\mathbf{x}_i, d_i) \in \mathbf{Y}, d_i = +1\} \\
\mathbf{Y}_{\leq} &= \{\mathbf{x}_i | (\mathbf{x}_i, d_i) \in \mathbf{Y}, d_i = -1\}
\end{aligned}$$

noting that the indices of the elements $\mathbf{x}_i$ remain unchanged - for example, if $(\mathbf{x}_4, d_4) \in \mathbf{Y}$, $d_4 = +1$, then $\mathbf{x}_4 \in \mathbf{Y}_{\geq}$, $\mathbf{x}_4 \notin \mathbf{Y}_{\leq}$, with the same index 4 used in all cases. This is purely a matter of notational convenience.

It is assumed that there is some pre-defined map $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$ from d_L dimensional input space to d_H dimensional feature space (where d_H may be very large or even infinite). Furthermore, it is assumed that $\Xi_{\hat{\Delta}_{+1}}$ and $\Xi_{\hat{\Delta}_{-1}}$ are nonlinearly separable under $\boldsymbol{\varphi}$. It follows that there exists an infinite set R of classifiers (4.1) satisfying:

$$\begin{aligned} \operatorname{sgn}\left(g\left(\lambda\right)\left(\mathbf{x}_i\right)\right) &= +1 \forall \mathbf{x}_i \in \mathbf{Y}_{\geq} \\ \operatorname{sgn}\left(g\left(\lambda\right)\left(\mathbf{x}_i\right)\right) &= -1 \forall \mathbf{x}_i \in \mathbf{Y}_{\leq} \end{aligned} \tag{4.3}$$

Consider one such classifier, $g\left(\lambda_1\right) \in R$, where $\lambda_1 = \{\mathbf{w}_1, b_1\}$. For some $\kappa > 0$, define $\lambda_\kappa = \{\kappa\mathbf{w}_1, \kappa b_1\}$. Clearly $g\left(\lambda_\kappa\right) \in R$ (as $\gamma\left(\lambda_\kappa\right) = \operatorname{sgn}\left(g\left(\lambda_\kappa\right)\right) = \operatorname{sgn}\left(\kappa g\left(\lambda_1\right)\right) = \operatorname{sgn}\left(g\left(\lambda_1\right)\right) = \gamma\left(\lambda_1\right)$). Furthermore, the decision surfaces associated with the classifiers $g\left(\lambda_1\right)$ and $g\left(\lambda_\kappa\right)$ ($\{\mathbf{x} | g\left(\lambda_1\right)\left(\mathbf{x}\right) = 0\}$ and $\{\mathbf{x} | g\left(\lambda_\kappa\right)\left(\mathbf{x}\right) = 0\}$, respectively), must be identical. It follows that $g\left(\lambda_1\right)$ and $g\left(\lambda_\kappa\right)$ define the same classifier $\gamma\left(\lambda_1\right) = \gamma\left(\lambda_\kappa\right)$. So, without loss of generality, the set of allowable classifiers R may be restricted to the subset $R_{\boldsymbol{\epsilon}}$ of classifiers satisfying:

$$\begin{aligned} g\left(\lambda\right)\left(\mathbf{x}_i\right) &\geq -E\epsilon_i \forall \mathbf{x}_i \in \mathbf{Y}_{\geq} \\ g\left(\lambda\right)\left(\mathbf{x}_i\right) &\leq E\epsilon_i^* \forall \mathbf{x}_i \in \mathbf{Y}_{\leq} \end{aligned} \tag{4.4}$$

where $\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon}^*$ are negative constant vectors, each element ϵ_i (or ϵ_i^*) being defined in adjunct to a single training point $\mathbf{x}_i \in \mathbf{Y}_{\geq}$ (or $\mathbf{x}_i \in \mathbf{Y}_{\leq}$) and $E \geq 0$ is a constant. There exists some redundancy here, as for all i, only one of ϵ_i and ϵ_i^* will be relevant to (i.e. effect) (4.4). To eliminate this redundancy, the vectors $\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon}^*$ may be consolidated into a single vector, $\boldsymbol{\epsilon}^{(*)}$, where:

$$\epsilon_i^{(*)} = \begin{cases} \epsilon_i & \text{if } \mathbf{x}_i \in \mathbf{Y}_{\geq} \\ \epsilon_i^* & \text{if } \mathbf{x}_i \in \mathbf{Y}_{\leq} \end{cases}$$

In the standard formulation, $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}^* = -\mathbf{1}$ and $E = 1$. $\boldsymbol{\epsilon}^{(*)}$ may be used to differentiate between points based on the relative importance (i.e. important points should lie further from the boundary to make misclassification less likely). It should be noted that at this juncture that there is, strictly speaking, no need to include both E and $\boldsymbol{\epsilon}^{(*)}$. It should be quite clear from the construction of $R_{\boldsymbol{\epsilon}}$ that the constant E

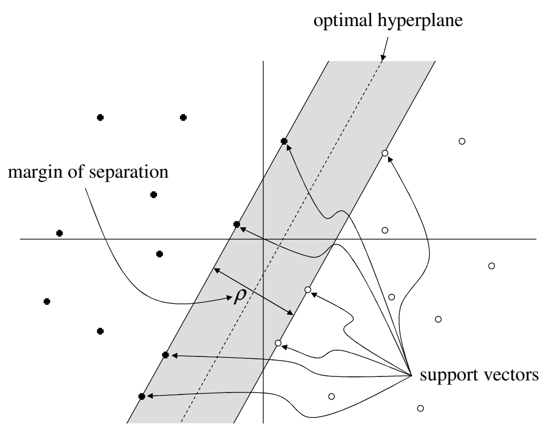


Figure 4.1: Optimal hyperplane selection via max-margin.

will have no effect on the form of the decision surface (its effect is analogous to that of κ in the above construction). The *mean* of $\boldsymbol{\epsilon}^{(*)}$, $\frac{1}{N}\mathbf{1}^T\boldsymbol{\epsilon}^{(*)}$ is similarly inneffectual. However, there is no harm in including some redundancy at this point (the choice is arbitrary, so an arbitrary selection may be made). The two main reasons for including these constants (E in particular) are:

1. To maintain notational consistency when constructing the combined classification and regression SVM formulation (regression with inequalities) in section 5.1.

2. E is included for simplicity when introducing *tube shrinking* in section 5.5.

The set $R_{\boldsymbol{\epsilon}}$ of potential classifiers is still infinite in size. What is required is some way to select the "best" classifier $g(\lambda) \in R_{\boldsymbol{\epsilon}}$. The basic concept underpinning the SVM approach is to select the classifier whose associated decision surface maximises the distance between the decision surface and those points lying closest to it (the support vectors), as shown in figure 4.1. As will be shown in section 4.2.3, this is the closely related to selecting a classifier from the subset $R^*_{\boldsymbol{\epsilon}} \subset R_{\boldsymbol{\epsilon}}$ with the least capacity such that $\exists \lambda \in R^*_{\boldsymbol{\epsilon}}$ satisfying (4.4) (i.e. $R^*_{\boldsymbol{\epsilon}} \neq \emptyset$).

It is not difficult to see that the shortest Euclidean distance r_i between any training point $\mathbf{x}_i \in \mathbf{Y}_{\geq} \cup \mathbf{Y}_{\leq}$ and a decision surface defined by (4.4) in feature space is:

$$r_i(\mathbf{x}_i) = \frac{|g(\mathbf{x}_i)|}{\|\mathbf{w}\|_2}$$

Hence the distance between the decision surface and the training point(s) of a particular class lying closest to it is:

$$\begin{aligned}\tau^{\geq} &= \min_{\mathbf{x}_i \in \mathbf{Y}_{\geq}} r_i\left(\mathbf{x}_i\right) \\ \tau^{\leq} &= \min_{\mathbf{x}_i \in \mathbf{Y}_{\leq}} r_i\left(\mathbf{x}_i\right)\end{aligned}$$

Using the fact that an arbitrary scaling $\mathbf{w}' = \kappa \mathbf{w}$ and $b' = \kappa b$, $\kappa > 0$, does not change the decision surface, it follows that it is always possible to scale $\mathbf{w}$ and b so that:

$$\begin{aligned}\tau^{\geq} &= \min_{\mathbf{x}_i \in \mathbf{Y}_{\geq}} \tfrac{-E\epsilon_i}{\|\mathbf{w}\|_2} \\ \tau^{\leq} &= \min_{\mathbf{x}_j \in \mathbf{Y}_{\leq}} \tfrac{-E\epsilon_j^*}{\|\mathbf{w}\|_2}\end{aligned}$$

(if only one of these conditions can be met, then the decision surface may be shifted (without changing its orientation) toward the class for which this condition fails until both are met). Based on this, the *margin of separation* between the two classes is defined to be:

$$\rho = \tau^{\geq} + \tau^{\leq} = \frac{E\left|\epsilon_i + \epsilon_j^*\right|}{\|\mathbf{w}\|_2} \tag{4.5}$$

for some $1 \leq i, j \leq N$.

Noting that $\rho \propto \frac{2}{\|\mathbf{w}\|_2}$, and neglecting $E\left|\epsilon_i + \epsilon_j^*\right|$,[1] it may be noted that the problem of finding the decision surface $g(\mathbf{x}) = 0$ satisfying (4.4) that maximises the margin of separation (4.5) is equivalent to solving the following optimisation problem:

$$\begin{aligned}&\min_{\mathbf{w},b} R_0\left(\mathbf{w}, b\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} \\ &\text{such that:}\quad \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq -E\epsilon_i \forall \mathbf{x}_i \in \mathbf{Y}_{\geq} \\ &\qquad\qquad\quad \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq E\epsilon_i^* \forall \mathbf{x}_i \in \mathbf{Y}_{\leq}\end{aligned} \tag{4.6}$$

At this juncture, it is worthwhile comparing (4.6) and the regularised risk functional, (3.2). Firstly, note that the constraints in (4.6) ensure that $R_{emp}\left(\lambda \middle| \mathbf{Y}\right) = 0$.

[1] Alternatively, remembering that E is essentially superfluous (it may be scaled arbitrarily without effecting the decision surface) E may be selected so that $E\left|\epsilon_i + \epsilon_j^*\right| = 2$ *without changing the decision surface itself.* Hence the term $E\left|\epsilon_i + \epsilon_j^*\right|$ is, truly, irrelevant.

Tentatively, therefore, it seems reasonable to identify:

$$\phi(\lambda) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

This issue will be re-visited in section 4.2.3.

The Dual Formulation

Unfortunately, the primal optimisation problem (4.6) is rather difficult to solve (especially if d_H is large or infinite). For this reason, it is convenient to convert (4.6) into a dual (or at least partially dual) form. To do this, introduce a vector of Lagrange multipliers, $\boldsymbol{\alpha}$, such that each element of this vector $\alpha_i \geq 0$ corresponds to a single inequality constraint, $\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq -E\epsilon_i$ if $\mathbf{x}_i \in \mathbf{Y}_\geq$ ($\alpha_i \leq 0$ to $\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq E\epsilon_i^*$ otherwise) in (4.6). The Lagrangian (maximin) form of (4.6) is then:

$$\begin{aligned}
\min_{\mathbf{w},b}\max_{\boldsymbol{\alpha}} L_0(\mathbf{w},b,\boldsymbol{\alpha}) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} - &\sum_{\mathbf{x}_i\in\mathbf{Y}_\geq} \alpha_i\left(\left(\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)+b\right)+E\epsilon_i\right)\\
- &\sum_{\mathbf{x}_i\in\mathbf{Y}_\leq} \alpha_i\left(\left(\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)+b\right)-E\epsilon_i^*\right)\\
\text{such that:}\quad & \alpha_i \geq 0 \forall i : \mathbf{x}_i \in \mathbf{Y}_\geq\\
& \alpha_i \leq 0 \forall i : \mathbf{x}_i \in \mathbf{Y}_\leq
\end{aligned} \tag{4.7}$$

It follows that, for optimality:

$$\begin{aligned}
\tfrac{\partial L_0}{\partial \mathbf{w}} &= \mathbf{0} \Rightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i\boldsymbol{\varphi}(\mathbf{x}_i)\\
\tfrac{\partial L_0}{\partial b} &= 0 \Rightarrow \mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned}$$

Hence the dual form is:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} Q_0(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon}^{(*)}\\
&\text{such that:}\quad \alpha_i \geq 0 \forall i : \mathbf{x}_i \in \mathbf{X}_\geq\\
&\qquad\qquad\quad\ \alpha_i \leq 0 \forall i : \mathbf{x}_i \in \mathbf{X}_\leq\\
&\qquad\qquad\quad\ \mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned} \tag{4.8}$$

where $\mathbf{K} \in \Re^{N \times N}$, $K_{i,j} = K\left(\mathbf{x}_i, \mathbf{x}_j\right)$ and $K\left(\mathbf{x}_i, \mathbf{x}_j\right) = \boldsymbol{\varphi}\left(\mathbf{x}_i\right)^T \boldsymbol{\varphi}\left(\mathbf{x}_j\right)$ is known as the *kernel function.* The resulting classification function may be expressed in terms of $\boldsymbol{\alpha}$ and b thusly:

$$\operatorname{sgn}\left(g\left(\lambda\right)\left(\mathbf{y}\right)\right) = \operatorname{sgn}\left(\sum_{\substack{i=1\\ \alpha_i \neq 0}}^{N} \alpha_i K\left(\mathbf{x}_i, \mathbf{y}\right) + b\right) \tag{4.9}$$

where:

$$b = -E \operatorname{sgn}\left(\alpha_j\right) \epsilon_j^{(*)} - \sum_{\substack{i=1\\ \alpha_i \neq 0}}^{N} \alpha_i K_{i,j} \forall j \in \left\{ 1 \leq j \leq N \middle| \, \alpha_j \neq 0 \right\}$$

The appearance of $|\boldsymbol{\alpha}|$ in (4.8) may cause some consternation at first. However, noting that

$$\operatorname{sgn}\left(\alpha_i\right) = \begin{cases} d_i & \text{if } \alpha_i \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

(4.8) may be re-written thusly:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} Q_0\left(\boldsymbol{\alpha}\right) &= \tfrac{1}{2}\boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + E \boldsymbol{\alpha}^T \left(\mathbf{d} \boldsymbol{\epsilon}^{(*)}\right) \\ \text{such that: } & \alpha_i \geq 0 \forall \mathbf{x}_i \in \mathbf{X}_{\geq} \\ & \alpha_i \leq 0 \forall \mathbf{x}_i \in \mathbf{X}_{\leq} \\ & \mathbf{1}^T \boldsymbol{\alpha} = 0 \end{aligned}$$

which is just a standard quadratic programming problem. I have chosen to give the dual of the SVM problem in form (4.8) because it does not present any real difficulty when solving the dual problem and, as will be seen later, makes the relationship between the regression and pattern recognition forms of the SVM more plainly apparent. To convert to the standard form, replace α_i with $d_i \alpha_i$ and set $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}^* = -\mathbf{1}$, $E = 1$.

Note that the matrix $\mathbf{K}$ is positive semi-definite (this will be proven in theorem 7.2) and the constraints are linear. Hence the optimisation problem is convex (there are no non-global minima), which greatly simplifies the process of finding a solution. It is also worth noting that there is exactly one variable α_i associated with each training point $\mathbf{x}_i$, and that only those α_i's corresponding to support vectors will have non-zero values. Formally, a training point $\mathbf{x}_i \in \mathbf{X}$ is called a support vector if $\alpha_i \neq 0$.

The Kernel Trick

The function $K : \Re^{d_L} \times \Re^{d_L} \to \Re$ is known as the kernel function (strictly, the Mercer kernel - however, as I only consider Mercer kernels here, I will usually leave the Mercer part implicit). As it has been defined here, it is the result of mapping two vector arguments into some other flat space (feature space) and then evaluating a standard dot product in this space.

It should be noted that in neither the optimisation problem (4.8), nor in the resulting classification function (4.9), is it necessary to evaluate the result of mapping a point into feature space using $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$. The kernel function hides this detail, resulting in a problem of dimension N, rather than d_H. Thus it is possible to use very high (or even infinite) dimensional feature spaces and complex mappings without increasing the complexity of the dual problem (4.8). This hiding of problem complexity using kernel functions is known generally as the *kernel trick*.

Rather than starting with a feature map and then calculating the kernel function, it is usual to simply define a kernel function (or look one up in a table of kernel functions). To see if a function $K : \Re^{d_L} \times \Re^{d_L} \to \Re$ may be used as a kernel function, *Mercer's theorem* [28], [58] may be applied:

Theorem 4.1. *For a function $K : \Re^{d_L} \times \Re^{d_L} \to \Re$, there exists a map $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$ such that*

$$K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\varphi}(\mathbf{x}_i)^T \boldsymbol{\varphi}(\mathbf{x}_j)$$

if and only if, for all $g : \Re^{d_L} \to \Re$ such that $\int g(\mathbf{x})^2 d\mathbf{x}$ is finite:

$$\int K(\mathbf{x}, \mathbf{y}) g(\mathbf{x}) g(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0$$

For later reference, define:

Definition 4.1. *The set of all Mercer kernels is denoted $\mathcal{M}_\kappa$.*

4.1.3 The Inseparable Case

If the training classes are not separable, it is necessary to relax the inequalities in (4.6) using slack variables and modify the cost function to penalise any failure to meet the original (strict) inequalities. Using the standard (linear) penalty function (see [28]), the problem becomes:

$$\begin{aligned}
&\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} R_1\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \\
&\text{such that:} \quad \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq -E\epsilon_i - \xi_i \forall \mathbf{x}_i \in \mathbf{Y}_\geq \\
&\qquad\qquad\quad \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq E\epsilon_i^* + \xi_i^* \forall \mathbf{x}_i \in \mathbf{Y}_\leq \\
&\qquad\qquad\quad \boldsymbol{\xi} \geq \mathbf{0} \\
&\qquad\qquad\quad \boldsymbol{\xi}^* \geq \mathbf{0}
\end{aligned} \tag{4.10}$$

as shown graphically in figure 4.2, where each $t_i^{(*)} > 0$ (where $\mathbf{t}^{(*)}$ is defined by $\mathbf{t}$ and $\mathbf{t}^*$ in a manner directly analogous to the manner in which $\boldsymbol{\epsilon}^{(*)}$ is defined, with similar redundancy), like $\epsilon_i^{(*)}$, is defined in adjunct to the training point $\mathbf{x}_i$, and provides a means of differentiating between more and less important training points (heuristically, misclassifying an "important" training point should cost more than misclassifying an unimportant or unreliable one, and hence $t_i^{(*)}$ would typically be larger for more important points). Like E, C is a "global" scaling factor for $\mathbf{t}$. However, unlike E, C is in no way redundant. Note that, as $\xi_i\xi_i^* = 0$ for all i, it is possible to unambiguously define a vector $\boldsymbol{\xi}^{(*)}$ by:

$$\xi_i^{(*)} = \begin{cases} \xi_i & \text{if } \mathbf{x}_i \in \mathbf{Y}_\geq \\ \xi_i^* & \text{if } \mathbf{x}_i \in \mathbf{Y}_\leq \end{cases}$$

so that:

$$R_1\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N}\mathbf{t}^{(*)T}\boldsymbol{\xi}^{(*)}$$

In (4.10), identifying:

$$\begin{aligned}
&\phi\left(\lambda\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} \\
&R_{emp}\left(\lambda\right|\mathbf{Y}) = \tfrac{1}{N}\mathbf{t}^{(*)T}\boldsymbol{\xi}^{(*)}
\end{aligned}$$

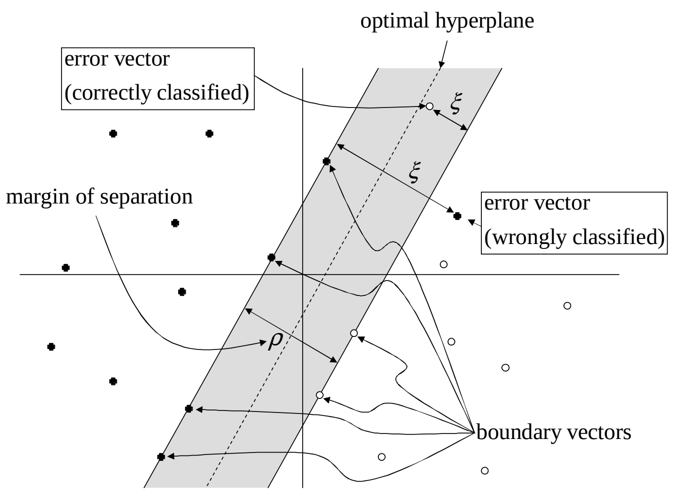


Figure 4.2: Optimal hyperplane selection via max-margin, (inseparable case).

using the notation of (3.2) it is clear that $R_1(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*)$ is essentially a measure of regularised risk. The parameter C controls the trade-off between the dual objectives of maximising the margin (regularisation) of separation and minimising the misclassification error (empirical risk minimisation).

The Lagrangian (maximin) form of (4.10) is:[2]

$$\begin{aligned}\min_{\mathbf{w},b,\boldsymbol{\xi}^{(*)}} \max_{\boldsymbol{\alpha},\boldsymbol{\gamma}^{(*)}} L_1 = {} & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^{(*)T}\boldsymbol{\xi}^{(*)} + \boldsymbol{\gamma}^{(*)T}\boldsymbol{\xi}^{(*)} \\ & - \sum_{i:\mathbf{x}_i\in\mathbf{Y}_\geq} \alpha_i\left(\left(\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)+b\right) + E\epsilon_i^{(*)} + \xi_i^{(*)}\right) \\ & - \sum_{i:\mathbf{x}_i\in\mathbf{Y}_\leq} \alpha_i\left(\left(\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)+b\right) - E\epsilon_i^{(*)} - \xi_i^{(*)}\right) \qquad (4.11)\\ \text{such that: } & \alpha_i \geq 0 \forall i : \mathbf{x}_i \in \mathbf{Y}_\geq \\ & \alpha_i \leq 0 \forall i : \mathbf{x}_i \in \mathbf{Y}_\leq \\ & \boldsymbol{\gamma}^{(*)} \geq \mathbf{0}\end{aligned}$$

[2]Noting that $L_1 = L_1\left(\mathbf{w}, b, \boldsymbol{\xi}^{(*)}, \boldsymbol{\alpha}, \boldsymbol{\gamma}^{(*)}\right)$. The arguments of the Lagrangian will often be neglected in this way to avoid unnecessary clutter in expressions.

Applying the usual methods, it follows that:

$$\begin{aligned}
&\tfrac{\partial L_1}{\partial \mathbf{w}} = \mathbf{0} \Rightarrow \mathbf{w} = \textstyle\sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}(\mathbf{x}_i) \\
&\tfrac{\partial L_1}{\partial b} = 0 \Rightarrow \mathbf{1}^T \boldsymbol{\alpha} = 0 \\
&\tfrac{\partial L_1}{\partial \boldsymbol{\xi}^{(*)}} = 0 \Rightarrow \alpha_i + \gamma_i^{(*)} = \tfrac{C}{N} t_i^{(*)} \forall i : \mathbf{x}_i \in \mathbf{Y}_{\geq} \\
&\tfrac{\partial L_1}{\partial \boldsymbol{\xi}^{(*)}} = 0 \Rightarrow -\alpha_i + \gamma_i^{(*)} = \tfrac{C}{N} t_i^{(*)} \forall i : \mathbf{x}_i \in \mathbf{Y}_{\leq}
\end{aligned}$$

Hence the dual form is:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} Q_1(\boldsymbol{\alpha}) = \tfrac{1}{2} \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} + E |\boldsymbol{\alpha}|^T \boldsymbol{\epsilon}^{(*)} \\
&\text{such that:} \quad 0 \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : \mathbf{x}_i \in \mathbf{Y}_{\geq} \\
&\qquad\qquad\quad -\tfrac{C}{N} t_i^* \leq \alpha_i \leq 0 \forall i : \mathbf{x}_i \in \mathbf{Y}_{\leq} \\
&\qquad\qquad\quad \mathbf{1}^T \boldsymbol{\alpha} = 0
\end{aligned} \tag{4.12}$$

where $\mathbf{K} \in \Re^{N \times N}$ and $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$. This form differs from (4.8) only in the existence of the upper bound on $|\boldsymbol{\alpha}|$. Once again:

$$\operatorname{sgn}(g(\lambda)(\mathbf{y})) = \operatorname{sgn}\left(\sum_{\substack{i=1 \\ \alpha_i \neq 0}}^{N} \alpha_i K(\mathbf{x}_i, \mathbf{y}) + b \right)$$

where:

$$b = -E \operatorname{sgn}(\alpha_j) \epsilon_j^{(*)} - \sum_{\substack{i=1 \\ \alpha_i \neq 0}}^{N} \alpha_i K_{i,j} \forall j \in \left\{ 1 \leq j \leq N \middle| 0 < |\alpha_j| < \frac{C t_j^{(*)}}{N} \right\}$$

4.1.4 The Partially Dual Form

While the dual form of the SVM is not overly complex, the presence of the equality constraint $\mathbf{1}^T \boldsymbol{\alpha} = 0$ does add some unwanted complexity to the problem. Furthermore, the need to calculate b afterwards is an unnecessary inconvenience. Rather than use this form, it is desirable to use instead a partially-dual form of the problem. In particular, do not eliminate b from the problem when constructing the dual. This gives the (partial-)dual problem:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} \max_{b} Q_{L_1}(\boldsymbol{\alpha}, b) = \frac{1}{2} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} + E \begin{vmatrix} b \\ \boldsymbol{\alpha} \end{vmatrix}^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{bmatrix} \\
&\text{such that:} \quad 0 \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : \mathbf{x}_i \in \mathbf{Y}_{\geq} \\
&\qquad\qquad -\tfrac{C}{N} t_i^* \leq \alpha_i \leq 0 \forall i : \mathbf{x}_i \in \mathbf{Y}_{\leq}
\end{aligned} \tag{4.13}$$

where:

$$\mathbf{H} = \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{bmatrix}$$

There are two main advantages (4.13) has over (4.12). Most obvious is the fact that b and $\boldsymbol{\alpha}$ are calculated together, rather than finding $\boldsymbol{\alpha}$ first and then b based on this. The other big advantage this form has (which will be especially noticeable when considering incremental training issues) is that there is no equality constraint present in the constraint set. The disadvantages of this form are the indefiniteness of the hessian and the increased dimensionality ($N+1$, as opposed to N). It will be seen in chapters 7 and 8 that the indefiniteness issue is not serious, and unless N is particularly small, the increase in dimension will not be noticeable.

The stationary or KKT (Karush-Kuhn-Tucker) conditions for the program (4.13), see [40], are as follows:

$$\begin{aligned}
\alpha_i &\leq \begin{cases} \frac{C}{N} t_i & \text{if } \mathbf{x}_i \in \mathbf{Y}_{\geq} \\ 0 & \text{if } \mathbf{x}_i \in \mathbf{Y}_{\leq} \end{cases} \\
\alpha_i &\geq \begin{cases} 0 & \text{if } \mathbf{x}_i \in \mathbf{Y}_{\geq} \\ -\frac{C}{N} t_i^* & \text{if } \mathbf{x}_i \in \mathbf{Y}_{\leq} \end{cases} \\
e_i &\begin{cases} \geq -E\epsilon_i & \text{if } \alpha_i = 0 \wedge \mathbf{x}_i \in \mathbf{Y}_{\geq} \\ = -E\epsilon_i & \text{if } 0 < \alpha_i < \frac{C}{N} t_i \\ \leq -E\epsilon_i & \text{if } \alpha_i = \frac{C}{N} t_i \\ \leq E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge \mathbf{x}_i \in \mathbf{Y}_{\leq} \\ = E\epsilon_i^* & \text{if } -\frac{C}{N} t_i^* < \alpha_i < 0 \\ \geq E\epsilon_i^* & \text{if } \alpha_i = -\frac{C}{N} t_i^* \end{cases} \\
f &= 0
\end{aligned} \tag{4.14}$$

where:

$$\begin{bmatrix} f \\ \mathbf{e} \end{bmatrix} = \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} \tag{4.15}$$

At this point, it is important to note that $e_i = g(\mathbf{x}_i)$ for all $\mathbf{x}_i \in \mathbf{Y}$. Given this, one immediately sees that the KKT conditions on $\mathbf{e}$ in (4.14) are just a relaxed form of the original conditions (4.4) in the separable case. Indeed (see [13]), α_i may be interpreted as the amount of force exerted on the decision surface by a training point to keep it the required distance away from this point (the decision is considered to be a solid plate under this interpretation). Hence $\frac{C}{N} t_i^{(*)}$ may be seen as an effective upper bound on the amount of force any one training point may impart on the decision surface, thereby limiting the amount of influence that point may have on the position of the decision surface (it also prevents the force becoming infinite in the inseparable case). Under this interpretation, the condition $f = 0$ is simply a statement that the total force on the decision surface is 0, placing the decision surface in a position of mechanical equilibrium.

It is useful to split the set of support vectors into boundary vectors, for which $0 < |\alpha_i| < \frac{C}{N} t_i^{(*)}$, and error vectors, for which $|\alpha_i| = \frac{C}{N} t_i^{(*)}$. The major difference between the two subclasses is that boundary vectors must be on the correct side of the decision surface (i.e. be correctly classified by the classifier), while error vectors need not be. Note also that the margin of separation (4.5) is, in the inseparable case, measured between the boundary vectors of the relevant classes. For later use, N_S is defined to be the number of support vectors, N_B the number of boundary vectors and N_E the number of error vectors. Hence $N_S = N_B + N_E$.

4.2 SVMs and Risk Minimisation

In previous sections, I tentatively identified:

$$\begin{aligned} \phi(\lambda) &= \tfrac{1}{2} \mathbf{w}^T \mathbf{w} \\ R_{emp}(\lambda | \mathbf{Y}) &= \tfrac{1}{N} \mathbf{t}^{(*)T} \boldsymbol{\xi}^{(*)} \end{aligned}$$

so that:

$$R(\mathbf{w}, b) = R_{reg}(\lambda | \mathbf{Y}) = \phi(\lambda) + CR_{emp}(\lambda | \mathbf{Y})$$

Assuming this identification is reasonable, it places the SVM approach to binary classification on a firm theoretical footing, and helps to explain the excellent results that have been achieved using SVMs. In this section this identification is shown to be theoretically valid.

4.2.1 The VC Dimension and the Risk Bound

An important first step to understanding the link between max-margin methods and the regularisation principle is to find a concrete means of measuring the capacity of a set of classifiers. The Vapnik-Chervonenkis (VC) dimension of a class of classifiers is one such measure.

The VC dimension is a characteristic of the complete set of possible classification functions, $T = \left\{ \gamma(\lambda) : \Re^{d_L} \to \bar{J} \middle| \lambda \in \Lambda \right\}$. Before defining the VC dimension, it is necessary to define the concept of shattering of points in an d_L-dimensional space.

Definition 4.2. *A set of l distinct points (vectors) is said to be* shattered *by a set of functions T if, for each of the 2^l possible ways of labelling the points in an arbitrary binary manner, there exists a member of the classifier set T which can correctly assign these labels.*

Definition 4.3. *The* VC dimension *h of a set of binary functions T is defined to be the maximum number of distinct points h which can be shattered by T.*

It should be noted that the definition of the VC dimension of a set of binary functions T does not imply that *all* sets of h points can be shattered by T, just that at least one such set of points exists which can be. Generally speaking, the following observations may be made:

- Learning networks selected from sets with a high VC dimensions are able to learn very complex tasks.

- However, they have a tendency to overfit. That is, they learn incidental (irrelevant) information and noise as though it was relevant, and as a result may not generalise well.

- Conversely, if the VC dimension is too low, the classifier may generalise well, but will generally not work well when applied to a difficult task.

The following result due to Vapnik [94] helps to explain these observations:

Theorem 4.2. *For any $0 \leq \eta \leq 1$ there is a probability of $1-\eta$ that the following bound will hold:*

$$R(\lambda) \leq R_{emp}(\lambda|\mathbf{Y}) + \sqrt{\left(\frac{h\left(\log\left(\frac{2N}{h}\right) - \log\left(\frac{\eta}{4}\right)\right)}{N}\right)}$$

This is known as the risk bound, and the second term on the right is the *VC confidence*. If h is too small then the empirical risk $R_{emp}(\lambda)$ may be large, and so the actual risk $R(\lambda)$ may be large. Conversely, if h is too large, the VC confidence will be large, and so the above bound will be ineffectual in limiting the size of the actual risk. Ideally, h will be chosen to minimise the risk bound. One method of doing just that is to use *structural risk minimisation.*

4.2.2 Structural Risk Minimisation

Consider a set of functions $T = \left\{ \gamma(\lambda) : \Re \to \bar{J} \middle| \lambda \in \Lambda \right\}$. If the choice of λ is further restricted in some manner, then so to is the VC dimension associated with the (restricted) set of functions. Indeed, the following nested structure may be defined:

$$\begin{aligned}
&T_i = \{\gamma(\lambda)|\, \lambda \in \Lambda_i\} \\
&T = \bigcup_i T_i \\
&T_1 \subset T_2 \subset \ldots \subset T_i \subset T_{i+1} \subset \ldots \\
&h_1 \leq h_2 \leq \ldots \leq h_i \leq h_{i-1} \leq \ldots
\end{aligned}$$

where h_i is the VC dimension of the set of functions T_i.

For each function subset T_i it is in principle possible to minimise the empirical risk to find an optimal parameter choice $\lambda_i \in \Lambda$. Using this, it is possible to calculate the risk bound of each of these function subsets, from which one may select the solution associated with the minimal risk bound, namely:

$$i_{\min} = \arg\min_i \left(R_{emp}\left(\lambda_i \middle| \mathbf{Y}\right) + \sqrt{\left(\frac{h_i\left(\log\left(\frac{2N}{h_i}\right) - \log\left(\frac{\eta}{4}\right)\right)}{N}\right)}\right)$$

The classifier $\gamma\left(\lambda_{i_{min}}\right) : \Re^{d_L} \to \bar{J}$ is in a sense the "optimal" trade-off between empirical risk minimisation and capacity control, as it minimises the risk bound. This approach to finding $g\left(\lambda^*\right)$ is known as structural risk minimisation (SRM). As will be seen, SVMs implement a form of SRM.

4.2.3 SVMs and Structural Risk Minimisation

For SVMs the set of functions $T = \left\{ \gamma\left(\lambda\right) : \Re \to \bar{J} \middle| \lambda \in \Lambda \right\}$ takes the form of a set of oriented hyperplanes in some d_H dimensional feature space, i.e.:

$$\gamma\left(\lambda\right)\left(\mathbf{x}\right) = \operatorname{sgn}\left(\mathbf{w}^T \boldsymbol{\varphi}\left(\mathbf{x}\right) + b\right)$$

where $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$ is some pre-defined map and $\lambda = \left(\mathbf{w}, b\right)$. It is not difficult to show that the VC dimension of the set of all linear classifiers in $\Re^{d_H}$ is $d_H + 1$. So the capacity of a SVM is directly related to the dimension of its feature space.

This leads to an apparent contradiction. Support vector machines have been observed to generalise very well in many cases. However, the VC dimension corresponding to many SVM kernels is very high (or even infinite), which at first glance would appear to imply that they should generalise badly. The key to understanding this apparent contradiction lies in the relationship between the SVM method and the theory of structural risk minimisation.

Consider the primary cost function of the basic SVM:

$$R_1\left(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N}\mathbf{t}^{(*)T}\boldsymbol{\xi}^{(*)}$$

The second term is just a measure of empirical risk, as stated previously. To understand the first term, the following theorem due to Vapnik [94] may be used:

Theorem 4.3. *Let D denote the diameter of the smallest ball in feature space containing the images of all training vectors $\mathbf{x} \in \mathbf{X}$ mapped to feature space. The set of hyperplanes described by the equation $g(\lambda)(\mathbf{x}) = 0$ has a VC dimension h bounded above by:*

$$h \leq \min \left\{ \left\lceil \frac{D^2}{\rho^2} \right\rceil, d_H \right\} + 1$$

where ρ is the margin of separation, defined previously.

Using this, and assuming that $\left\lceil \frac{D^2}{\rho^2} \right\rceil \leq d_H$, the risk bound may be re-formulated thusly:

$$R(\lambda) < R_{emp}(\lambda|\mathbf{Y}) + \sqrt{\frac{\left(\left\lceil \frac{D^2}{\rho^2} \right\rceil + 1\right)\left(\log\left(\frac{2N}{\left\lceil \frac{D^2}{\rho^2} \right\rceil + 1}\right) - \log\left(\frac{\eta}{4}\right)\right)}{N}}$$

The term $\frac{1}{2}\mathbf{w}^T\mathbf{w}$ in the primal formulation of the SVM optimisation problem attempts to maximise ρ, thereby making it more likely that classifiers with larger margins of separation will be chosen in preference to those with small margins of separation, which is basically SRM at work (i.e. preferential consideration given to those classifiers which minimise the VC dimension, h, and therefore minimise the risk bound).

So the identification:

$$\begin{aligned} &\phi(\lambda) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} \\ &R_{emp}(\lambda|\mathbf{Y}) = \tfrac{1}{N}\mathbf{t}^{(*)T}\boldsymbol{\xi}^{(*)} \end{aligned}$$

so that:

$$R(\mathbf{w}, b) = R_{reg}(\lambda|\mathbf{Y}) = \phi(\lambda) + C R_{emp}(\lambda|\mathbf{Y})$$

is reasonable.

This explains why SVMs are able to generalise well despite their potentially excessive VC dimension - during training, they tune the VC dimension to be sufficient to avoid underfitting but not large enough to cause the risk bound to blow out and

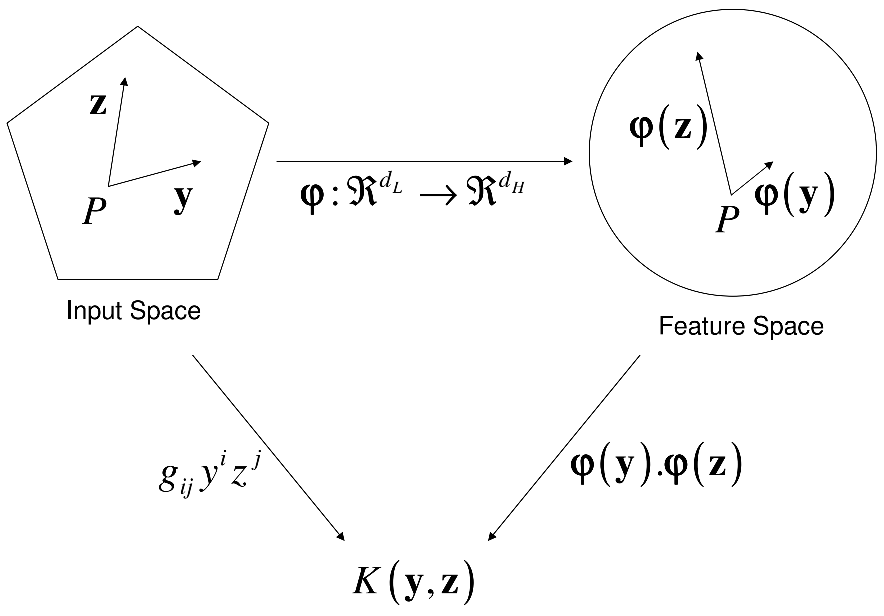


Figure 4.3: The alternative views of kernel mapping.

cause overfitting. It also highlights the importance of choosing C with care.

4.3 Kernel Geometry

As already stated, the standard interpretation of the Mercer kernel in the SVM setting is to treat it as a standard dot product between two vectors that are first mapped to feature space using a non-linear feature map, as shown schematically in figure 4.3. Thus, although SVM methods are non-linear in nature, they correspond to a linear problem in some higher dimensional feature space. Under this interpretation, the kernel function is simply a means of hiding the complexity of this feature space, thereby circumventing (to some degree) the so-called "curse of dimensionality".

While this is, strictly speaking, all that is necessarily to understand SVM methods, it is none-the-less instructive to look at the non-Euclidean geometry induced on input space from feature space by the non-linear feature map in more detail (following the work of [12]) to gain further insight into the structure of input space, which can be useful when choosing an appropriate kernel function for a given training set.

It is assumed that $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H} \subset C^\infty$. As usual, feature space is assumed to be Euclidean (and hence equipped with the usual Euclidean metric). The input space is assumed to be a (torsionless) Riemann manifold with coordinates inherited from the implicit coordinate system already existing on the input space. For notational simplicity, S is defined to be the input space, and F the associated feature space. So $\boldsymbol{\varphi} : S \to F$. Lower case Latin indices (i, j etc) range from 1 to d_L, and lower case greek indices (α, β etc) from 1 to d_H. The Einstein summation convention of automatic summation over repeated indices of different types (one raised, one lowered) will be used. Appendix B gives a short introduction to the necessary background to the essentials of differential geometry.

4.3.1 Constructing the Metric Tensor on Input Space

The following is largely based on [12]. For an SVM there exist two spaces, input space S and feature space F, connected by some map $\boldsymbol{\varphi} : S \to F$. Feature space is Euclidean with standard orthonormal coordinates, and has metric $\delta_{\alpha\beta}$. Given two points ξ^α and τ^α in feature space, the shortest path from ξ^α to τ^α (the geodesic) is the curve $\sigma^\alpha(t) = t\xi^\alpha + (1-t)\tau^\alpha$, where $0 \le t \le 1$. If ξ^α and τ^α both lie on the decision surface, so will all points on the geodesic.

Moreover, any smooth curve $\sigma^\alpha(t)$, $0 \le t \le 1$ between ξ^α and τ^α in feature space will have length:

$$l(\sigma) = \int_0^1 \sqrt{\delta_{\alpha\beta} \frac{\partial \sigma^\alpha}{\partial t} \frac{\partial \sigma^\beta}{\partial t}} dt$$

and the geodesic represents the shortest path, using this measure.

Now consider two points x^i and y^i in input space, where $\xi^\alpha = \varphi^\alpha(x^i)$ and $\tau^\alpha = \varphi^\alpha(y^i)$, along with a smooth curve $c^i(t)$, $0 \le t \le 1$, between these points in input space. The length of this curve in (non-Euclidean) input space is:

$$l(c) = \int_0^1 \sqrt{g_{\alpha\beta} \frac{\partial x^\alpha}{\partial t} \frac{\partial x^\beta}{\partial t}} dt$$

where $g_{\alpha\beta}$ is the metric tensor in input space.

Suppose that the curve $\sigma^\alpha(t)$, $0 \le t \le 1$ in feature space is the image of the curve $c^i(t)$, $0 \le t \le 1$ in input space under the feature map $\boldsymbol{\varphi} : S \to F$. Now, if x^i were a training vector, and y^i some point on the decision surface, then the distances

$l(c)$ in input space and $l(\sigma)$ in feature space should be the same. In general, this does not restrict either x^i or y^i, and so it is sensible to require that the length of all curves c in input space must correspond to the length of the image of that curve in feature space.

Consider the infinitesimal path from x^i to $x^i + dx^i$ in input space. The squared length of the image of this path mapped to feature space is:

$$\begin{aligned} ds^2 &= \delta_{\alpha\beta}\left(\varphi^\alpha\left(x^i+dx^i\right)-\varphi^\alpha\left(x^i\right)\right)\left(\varphi^\beta\left(x^i+dx^i\right)-\varphi^\beta\left(x^i\right)\right) \\ &= K\left(\mathbf{x}+d\mathbf{x},\mathbf{x}+d\mathbf{x}\right)-K\left(\mathbf{x}+d\mathbf{x},\mathbf{x}\right)-K\left(\mathbf{x},\mathbf{x}+d\mathbf{x}\right)+K\left(\mathbf{x},\mathbf{x}\right) \end{aligned}$$

To second order:

$$\begin{aligned} K\left(\mathbf{x}+d\mathbf{x},\mathbf{x}+d\mathbf{x}\right) &= K\left(\mathbf{x},\mathbf{x}\right)+\left(\tfrac{\partial}{\partial x_i}K\left(\mathbf{x},\mathbf{y}\right)\right)\Big|_{\mathbf{y}=\mathbf{x}}dx^i+\left(\tfrac{\partial}{\partial x^i}K\left(\mathbf{y},\mathbf{x}\right)\right)\Big|_{\mathbf{y}=\mathbf{x}}dx^i \\ &\quad+\tfrac{1}{2}\left(\tfrac{\partial^2}{\partial x^i\partial x^j}K\left(\mathbf{x},\mathbf{x}\right)\right)dx^idx^j \\ K\left(\mathbf{x}+d\mathbf{x},\mathbf{x}\right) &= K\left(\mathbf{x},\mathbf{x}\right)+\left(\tfrac{\partial}{\partial x^i}K\left(\mathbf{x},\mathbf{y}\right)\right)\Big|_{\mathbf{y}=\mathbf{x}}dx^i \\ &\quad+\tfrac{1}{2}\left(\tfrac{\partial^2}{\partial x^i\partial x^j}K\left(\mathbf{x},\mathbf{y}\right)\right)\Big|_{\mathbf{y}=\mathbf{x}}dx^idx^j \\ K\left(\mathbf{x},\mathbf{x}+d\mathbf{x}\right) &= K\left(\mathbf{x},\mathbf{x}\right)+\left(\tfrac{\partial}{\partial x^i}K\left(\mathbf{y},\mathbf{x}\right)\right)\Big|_{\mathbf{y}=\mathbf{x}}dx^i \\ &\quad+\tfrac{1}{2}\left(\tfrac{\partial^2}{\partial x^i\partial x^j}K\left(\mathbf{y},\mathbf{x}\right)\right)\Big|_{\mathbf{y}=\mathbf{x}}dx^idx^j \end{aligned}$$

Using the symmetry $K(\mathbf{x},\mathbf{y}) = K(\mathbf{y},\mathbf{x})$, it follows that in feature space:

$$ds^2 = \left(\frac{1}{2}\left(\frac{\partial}{\partial x^i}\frac{\partial}{\partial x^j}K\left(\mathbf{x},\mathbf{x}\right)\right)-\left(\frac{\partial}{\partial y^i}\frac{\partial}{\partial y^j}K\left(\mathbf{x},\mathbf{y}\right)\right)\right)_{\mathbf{y}=\mathbf{x}}dx^idx^j$$

Therefore the induced metric in input space (at $\mathbf{x}$) is:[3]

$$g_{ij} = \left(\frac{1}{2}\left(\frac{\partial}{\partial x^i}\frac{\partial}{\partial x^j}K\left(\mathbf{x},\mathbf{x}\right)\right)-\left(\frac{\partial}{\partial y^i}\frac{\partial}{\partial y^j}K\left(\mathbf{x},\mathbf{y}\right)\right)\right)_{\mathbf{y}=\mathbf{x}} \tag{4.16}$$

So, given any kernel function, (4.16) may be used to calculate the metric g_{ij} in

[3]Amari and Wu [4] obtain the result:

$$g_{ij} = \left(\frac{\partial}{\partial x^i}\frac{\partial}{\partial y^j}K\left(\mathbf{x},\mathbf{y}\right)\right)_{\mathbf{x}=\mathbf{y}}$$

However, this result is only correct for kernels of the form $K(f(\mathbf{x}-\mathbf{y}))$, and may fail otherwise. Note however that as [4] only considers the RBF kernel in detail, this mistake does not effect the validity of the method presented in [4].

input space induced by the feature map implicitly defined by this kernel function. While the reverse process (using a metric to construct a kernel function) is in principle possible using equation (4.16), care must be taken to ensure that the result satisfies Mercer's condition. Furthermore, there is no guarantee of uniqueness (for example, the RBF kernel and the linear dot-product kernel are both associated with the same (up to linear scaling) metric).

It should be noted that geodesics in input space do not necessarily map to geodesics in feature space (as there is no guarantee that the shortest path in feature space will even lie in the image of the input space in feature space). Furthermore, the decision surface in input space need not be geodesic (flat) in input space - it is the pre-image of the intersection of the decision surface in feature space and the image of input space in feature space.

4.3.2 The Metric Kernel for Euclidean Difference Kernels

A Euclidean distance kernel function $K(\mathbf{x}, \mathbf{y})$ is kernel which is a function of the Euclidean distance $\|\mathbf{x} - \mathbf{y}\|^2$ - that is, $K(\mathbf{x}, \mathbf{y}) = K\left(\|\mathbf{x} - \mathbf{y}\|^2\right)$ (i.e. $K(\mathbf{x}, \mathbf{y}) = K\left(\delta_{mn}(x^m - y^m)(x^n - y^n)\right)$). For this case, it can be shown [12] that:

$$\begin{aligned} g_{ij} &= -2\delta_{ij} K'(0) \\ g^{ij} &= \tfrac{-\delta^{ij}}{2K'(0)} \\ \Gamma_{ijk} &= 0 \\ \Gamma_{ij}{}^{k} &= 0 \end{aligned}$$

where $K'(r) = \frac{\partial K(r)}{\partial r}$. Note that this implies that the curvature induced by the kernel mapping is zero, and hence the space is flat ($R^i{}_{jkl} = 0$). The volume element is:

$$dV = \sqrt{\left(-2K'(0)\right)^{d_L}} dx^{d_L}$$

An example of this kind of kernel function is the standard RBF kernel:

$$\begin{aligned} K(z) &= e^{-\frac{z}{2\sigma^2}} \\ K'(z) &= -\tfrac{1}{2\sigma^2} e^{-\frac{z}{2\sigma^2}} \end{aligned}$$

wherein:

$$\begin{aligned}
g_{ij} &= \sigma^{-2}\delta_{ij} \\
g^{ij} &= \sigma^{2}\delta^{ij} \\
\Gamma_{ijk} &= 0 \\
\Gamma_{ij}{}^{k} &= 0
\end{aligned}$$

Note also that $dV = \sigma^{-d_L} dx^{d_L}$. Thus the parameter σ in the standard RBF kernel may be interpreted as a scale on the volume of input space. The smaller σ is, the more compressed the induced input space will be (i.e. more volume in an apparently small area). Alternately, and equivalently, it may be interpreted as meaning that a smaller σ means the scale on the axis of input space is more stretched, thereby increasing the volume encompassed by an arbitrary cube constructed on these axis.

4.3.3 The Metric Kernel for Dot Product Kernels

A dot product kernel is a kernel function $K(\mathbf{x}, \mathbf{y})$ which is actually just a function of the dot product $\mathbf{x}^T\mathbf{y}$ - that is, $K(\mathbf{x}, \mathbf{y}) = K\left(\mathbf{x}^T\mathbf{y}\right) = K(\delta_{mn}x^m y^n)$. Hence [12]:[4]

$$\begin{aligned}
g_{ij} &= \delta_{ij}K' + \delta_{mi}\delta_{nj}x^m x^n K'' \\
g^{ij} &= \delta^{ij}\tfrac{1}{K'} - x^i x^j \tfrac{K''}{K'(K'+zK'')} \\
\Gamma_{ijk} &= \delta_{mi}\delta_{nj}\delta_{qk}x^m x^n x^q K''' + 2\delta_{q\{i}\delta_{j\}k}x^q K'' \\
\Gamma_{ij}{}^{k} &= \delta_{mi}\delta_{nj}x^m x^n x^k \tfrac{K'''K'-2K''^2}{K'(K'+zK'')} + 2\delta_{q\{i}\delta_{j\}}{}^{k}x^q \tfrac{K'''}{K'}
\end{aligned}$$

where $K = K(z)$, $K' = \frac{\partial K}{\partial z}(z)$, $K'' = \frac{\partial^2 K}{\partial z^2}(z)$, etc; and $z = \delta_{mn}x^m x^n$. So, as expected, the metric and connection are both spherically symmetric about the origin for all dot product kernels.

Calculation of the curvature tensor for the general case here is rather difficult due to the number of terms involved, and not particularly informative in any case. Therefore I will not give the exact expression for the curvature in the general case. Suffice to say that it is non-zero.

[4]There is a minor abuse of notation in [12], wherein the definition $x_i = \delta_{ij}x^j$ is used rather than the standard $x_i = g_{ij}x^j$ (the former is only true if $g_{ij}(\mathbf{x}) = \delta_{ij}$).

One example of a dot product kernel is the quadratic kernel:

$$\begin{aligned}
K(z) &= (1+z)^2 \\
K'(z) &= 2(1+z) \\
K''(z) &= 2 \\
K'''(z) &= 0 \\
K''''(z) &= 0
\end{aligned}$$

therefore:

$$\begin{aligned}
g_{ij} &= 2(1+z)\,\delta_{ij} + 2\delta_{im}\delta_{jn}x^m x^n \\
g^{ij} &= \tfrac{\delta^{ij}}{2(1+z)} - \tfrac{x^i x^j}{2(1+z)(1+2z)} \\
\Gamma_{ijk} &= 4\delta_{q\{i}\delta_{j\}k}x^q \\
\Gamma_{ij}{}^{k} &= -\tfrac{2\delta_{im}\delta_{jn}x^m x^n x^k}{(1+z)(1+2z)} \\
R^i{}_{jkl} &= 4\left(\tfrac{\delta_{jm}\delta_{n[k}\delta_{l]}{}^{i}x^m x^n - \delta_{j[k}\delta_{l]m}x^m x^i}{(1+z)(1+2z)}\right)
\end{aligned}$$

The volume element for a general dot product kernel is:

$$dV = \sqrt{\det\left(\delta_{ij}K' + \delta_{im}\delta_{jn}x^m x^n K''\right)}dx^{d_L}$$

or, writing in matrix notation:

$$dV = \sqrt{\det\left(K'\left(\mathbf{x}^T\mathbf{x}\right)\mathbf{I} + K''\left(\mathbf{x}^T\mathbf{x}\right)\mathbf{x}\mathbf{x}^T\right)}dx^{d_L}$$

This may be simplified further using the following theorem:

Theorem 4.4. *For any n-dimensional vector $\mathbf{b}$,* $\det\left(\mathbf{I} + \mathbf{b}\mathbf{b}^T\right) = 1 + \mathbf{b}^T\mathbf{b}$.

Proof. See appendix E. □

Hence:

$$dV = \sqrt{\left(K'\left(\mathbf{x}^T\mathbf{x}\right)\right)^{d_L}\left(1 + \frac{K''\left(\mathbf{x}^T\mathbf{x}\right)}{K'\left(\mathbf{x}^T\mathbf{x}\right)}\mathbf{x}^T\mathbf{x}\right)}dx^{d_L}$$

For the quadratic kernel given above, then, the volume element will be:

$$dV = \sqrt{2^{d_L}\left(1 + \mathbf{x}^T\mathbf{x}\right)^{d_L - 1}\left(1 + 2\mathbf{x}^T\mathbf{x}\right)}dx^{d_L}$$

4.4 Support Vector Machine Regression

For regression, the training set is:

$$\begin{aligned}&\mathbf{Y} = \mathbf{Y}_= = \{(\mathbf{x}_1, z_1), (\mathbf{x}_2, z_2), \ldots, (\mathbf{x}_N, z_N)\}\\&\mathbf{x}_i \in \Re^{d_L}\\&z_i \in \Re\end{aligned} \tag{4.17}$$

Motivated by the SVM method for pattern recognition, it is usual to define a map $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$ from input space to feature space. The system is approximated using the following linear function of position (in feature space):

$$g(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) + b$$

where $\lambda = (\mathbf{w}, b)$ is the parameter set. Following the template of the pattern recognition case, the cost function is modelled on the empirical risk formula, namely:

$$R(\mathbf{w}, b) = R_{reg}(\lambda | \mathbf{Y}) = C R_{emp}(\lambda | \mathbf{Y}) + \phi(\lambda)$$

where:

$$\phi(\lambda) = \frac{1}{2}\mathbf{w}^T \mathbf{w}$$

To understand this choice, note that:

$$\begin{aligned}\frac{\partial g(\mathbf{x})}{\partial \mathbf{x}} &= \sum_{i=1}^{d_H} w_i \frac{\partial \varphi_i(\mathbf{x})}{\partial \mathbf{x}}\\&= \sqrt{\mathbf{w}^T \mathbf{w}} \sum_{i=1}^{d_H} \hat{w}_i \frac{\partial \varphi_i(\mathbf{x})}{\partial \mathbf{x}}\end{aligned}$$

where $\hat{\mathbf{w}}$ is a unit vector in the direction of $\mathbf{w}$. So, to first order:

$$g(\mathbf{x} + \delta\mathbf{x}) = g(\mathbf{x}) + \sqrt{\mathbf{w}^T \mathbf{w}} \left(\sum_{i=1}^{d_H} \hat{w}_i \frac{\partial \varphi_i(\mathbf{x})}{\partial \mathbf{x}} \right)^T \delta\mathbf{x} + \ldots$$

If the inputs $\mathbf{x}$ (training data or otherwise) are affected by noise $\delta\mathbf{x}$, the magnitude of the error seen at the output will be proportional to $\sqrt{\mathbf{w}^T \mathbf{w}}$. Hence minimising

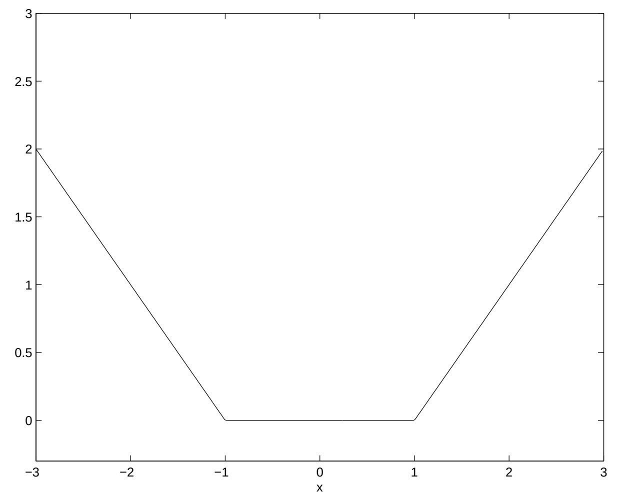


Figure 4.4: Vapnik's ϵ-insensitive cost function ($\epsilon = 1$ in this case).

$\phi(\lambda)$ will minimise the noise sensitivity of the of the regressor. Geometrically, this term flattens the function in feature space (minimises the gradient).

The choice of cost function is motivated by the dual objectives of computational simplicity and machine veracity. Ideally, motivated by the maximum likelihood theory of section 3.3, the cost function used would be the maximum likelihood cost for the type of noise affecting the training data. This is not practical for two main reasons:

- Incomplete knowledge - the type of noise present in the training data may not be known.
- Computational complexity - it may not be practical to use this cost function for reasons of computational complexity.

Vapnik [94] [95] suggests the following loss function (called the ϵ-insensitive loss function) as a compromise:

$$c(\mathbf{x}, y, z) = |y - z|_\epsilon = \begin{cases} 0 & \text{if } |y - z| \leq \epsilon \\ |y - z| - \epsilon & \text{otherwise} \end{cases} \tag{4.18}$$

where it is assumed that $\epsilon \geq 0$. This is shown in figure 4.4.

The parameter ϵ inserts a "dead-zone" so that training errors below this threshold do not contribute the cost. This is beneficial when dealing with noisy data, as it lends a degree of noise insensitivity to the cost function. Hence a simple choice of cost function [47] is:

$$R_{emp}\left(\lambda|\, \mathbf{Y}\right) = \frac{1}{N} \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}} t_i \left|\mathbf{w}^T \boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b - z_i\right|_{E\epsilon_i}$$

for $t_i > 0$, $\epsilon_i \geq 0$ for all i, and $E \geq 0$.

So, overall:

$$R_1\left(\mathbf{w}, b\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N} \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}} t_i \left|\mathbf{w}^T \boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b - z_i\right|_{E\epsilon_i} \tag{4.19}$$

For mathematical convenience, this equation is usually re-written in terms of the non-negative slack variables $\boldsymbol{\xi}$ defined by:

$$\begin{aligned}&\xi_i \geq \left|\mathbf{w}^T \boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b - z_i\right|_{E\epsilon_i} \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}\\ &\boldsymbol{\xi} \geq \mathbf{0}\end{aligned}$$

thusly:

$$R_1\left(\mathbf{w}, b, \boldsymbol{\xi}\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N}\mathbf{t}^T\boldsymbol{\xi}$$

so the primal form of the SVM regression problem is:

$$\begin{aligned}&\min_{\mathbf{w},b,\boldsymbol{\xi}} R_1\left(\mathbf{w}, b, \boldsymbol{\xi}\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi}\\ &\text{such that: } \xi_i \geq \left|\mathbf{w}^T \boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b - z_i\right|_{E\epsilon_i} \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_=\\ &\qquad\qquad\quad \boldsymbol{\xi} \geq 0\end{aligned} \tag{4.20}$$

More generally, it may be desirable to make the cost function asymmetrical - that is, apply a different penalty if $g\left(\lambda\right)$ is to large than would be applied if it was too small. This gives the final primal form:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \; & R_1\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \\
\text{such that:} \; & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \\
& \boldsymbol{\xi} \geq \mathbf{0} \\
& \boldsymbol{\xi}^* \geq \mathbf{0}
\end{aligned} \tag{4.21}$$

Note that, for regression, both $\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon}^*$ must be positive vectors, whereas for pattern recognition they were negative (more on this later). Note also that one of ξ_i and ξ_i^* will be zero for all i. Of course, there is (in this case) no reason to define $\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon}^*$ separately, as the effect of having $\epsilon_i \neq \epsilon_i^*$ will be the same as that of adding a bias to z_i. However, this differentiation of terms will be required later.

Geometrically, this regression formulation may be visualised as shown in figure 4.5. In this interpretation, feature space is extended by 1 dimension, namely z. The training points in this (extended) feature space are $\left(\boldsymbol{\varphi}\left(\mathbf{x}_i\right), z_i\right)$, and one aims to construct an ϵ-tube (a tube whose width along the z axis is ϵ, as shown in the figure), such that all (extended) training points lie inside this tube (or as close to the tube as possible if this is not achievable), and the tube itself is as flat as possible with respect to the z axis. The empirical risk term of the cost function works toward the former goal, and the regularisation term the latter.

4.4.1 The Dual and Partially Dual Forms

As for pattern recognition, the primal form (4.21) of the regression problem may not be convenient to solve directly, especially if d_H is large. Hence the dual form is usually used instead, and is derived as follows.

There are three inequalities in (4.21). Following the same approach as was used when constructing the dual form for SVMs for pattern recognition, introduce four vectors of Lagrange multipliers, $\boldsymbol{\beta} \geq \mathbf{0}$, $\boldsymbol{\beta}^* \leq \mathbf{0}$ and $\boldsymbol{\gamma} \geq \mathbf{0}$ and $\boldsymbol{\gamma}^* \geq \mathbf{0}$ associated with the four constraints in the primal (4.21) in the usual fashion. Using this

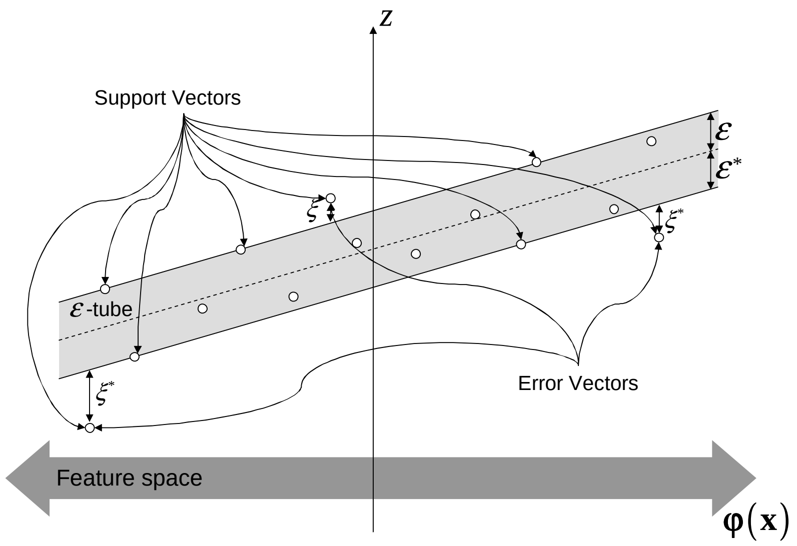


Figure 4.5: Geometric interpretation of the standard SV regressor.

notation, the maximin form of (4.21) will be:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*} L_1 = & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=} \beta_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)+b\right) - z_i + E\epsilon_i + \xi_i\right) \\
& - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=} \beta_i^* \left(\left(\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i)+b\right) - z_i - E\epsilon_i^* - \xi_i^*\right)
\end{aligned} \tag{4.22}$$

1

$$\begin{aligned}
\text{such that: } & \boldsymbol{\beta} \geq \mathbf{0} \\
& \boldsymbol{\beta}^* \leq \mathbf{0} \\
& \boldsymbol{\gamma} \geq \mathbf{0} \\
& \boldsymbol{\gamma}^* \geq \mathbf{0}
\end{aligned}$$

Now, given that $E \geq 0$, $\boldsymbol{\epsilon} \geq \mathbf{0}$ and $\boldsymbol{\epsilon}^* \geq \mathbf{0}$, it follows from the primal (4.21) that one or both of ξ_i or ξ_i^* must be zero for all i. Therefore one may define two new vectors, $\boldsymbol{\xi}^{(*)}$ and $\boldsymbol{\gamma}^{(*)}$, where:

$$\xi_i^{(*)} = \begin{cases} \xi_i & \text{if } \xi_i^* = 0 \\ \xi_i^* & \text{otherwise} \end{cases}$$

$$\gamma_i^{(*)} = \begin{cases} \gamma_i & \text{if } \xi_i^* = 0 \\ \gamma_i^* & \text{otherwise} \end{cases}$$

Furthermore, for any given i, at most one of the first two constraints on the primal (4.21) can be met exactly (where by "met exactly" I mean that the two sides of the expression are equal).[5] Hence at least one of β_i and β_i^* must be zero for all i. This makes it possible to define a new vector $\boldsymbol{\alpha}$ (or, alternatively, $\boldsymbol{\beta}^{(*)}$) where:

$$\alpha_i = \beta_i^{(*)} = \begin{cases} \beta_i & \text{if } \beta_i^* = 0 \\ \beta_i^* & \text{otherwise} \end{cases}$$

Using this notation, (4.22) may be re-written thusly:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}^{(*)}} \max_{\boldsymbol{\alpha},\boldsymbol{\gamma}^{(*)}} L_1 = {} & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^{(*)T}\boldsymbol{\xi}^{(*)} - \boldsymbol{\gamma}^{(*)T}\boldsymbol{\xi}^{(*)} \\ & - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_= \wedge \alpha_i>0} \alpha_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + E\epsilon_i + \xi_i^{(*)} \right) \\ & - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_= \wedge \alpha_i<0} \alpha_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i - E\epsilon_i^* - \xi_i^{(*)} \right) \\ \text{such that: } & \boldsymbol{\gamma}^{(*)} \geq \mathbf{0} \end{aligned}$$

Solving for the primal variables:

$$\begin{aligned} \tfrac{\partial L_1}{\partial \mathbf{w}} = \mathbf{0} & \Rightarrow \mathbf{w} = \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=} \alpha_i \boldsymbol{\varphi}(\mathbf{x}_i) \\ \tfrac{\partial L_1}{\partial b} = \mathbf{0} & \Rightarrow \mathbf{1}^T\boldsymbol{\alpha} = 0 \\ \tfrac{\partial L_1}{\partial \boldsymbol{\xi}^{(*)}} = \mathbf{0} & \Rightarrow \alpha_i + \gamma_i^{(*)} = \tfrac{C}{N} t_i \text{ if } \alpha_i \geq 0 \\ & \Rightarrow -\alpha_i + \gamma_i^{(*)} = \tfrac{C}{N} t_i^* \text{ if } \alpha_i < 0 \end{aligned}$$

Hence the dual form of (4.21) may be written:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} Q_1(\boldsymbol{\alpha}) = {} & \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon}^{(*)} \\ \text{such that: } & -\tfrac{C}{N}\mathbf{t}^* \leq \boldsymbol{\alpha} \leq \tfrac{C}{N}\mathbf{t} \\ & \mathbf{1}^T\boldsymbol{\alpha} = 0 \end{aligned} \tag{4.23}$$

[5]Technically, this assumes that $E(\epsilon_i + \epsilon_i^*) > 0$ for all $1 \leq i \leq N$, which may not be true in all cases. For a complete derivation including this case, see appendix A. This has no effect on the end result.

where $\mathbf{K}$ is defined in the same way as for pattern recognition, and:

$$\epsilon_i^{(*)} = \begin{cases} \epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i < 0 \end{cases}$$

As for pattern recognition:

$$g(\lambda)(\mathbf{y}) = \sum_{\substack{i=1 \\ \alpha_i \neq 0}}^{N} \alpha_i K(\mathbf{x}_i, \mathbf{y}) + b \tag{4.24}$$

and:

$$b = -E \operatorname{sgn}(\alpha_j) \epsilon_j^{(*)} - \sum_{\substack{i=1 \\ \alpha_i \neq 0}}^{N} \alpha_i K_{i,j} \forall j \in \left\{ 1 \leq j \leq N \,|\, 0 < |\alpha_j| < \frac{Ct_j}{N} \right\}$$

The presence of $|\boldsymbol{\alpha}|$, and also the vector $\boldsymbol{\epsilon}^{(*)}$ (which is implicitly a function of $\operatorname{sgn}(\boldsymbol{\alpha})$) in the dual form (4.23) may cause concern. However, re-writing (4.23) in terms of $\boldsymbol{\beta}$ and $\boldsymbol{\beta}^*$ the dual becomes:

$$\begin{aligned} \min_{\boldsymbol{\beta}, \boldsymbol{\beta}^*} Q_1(\boldsymbol{\beta}, \boldsymbol{\beta}^*) = \tfrac{1}{2} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \end{bmatrix}^T \begin{bmatrix} \mathbf{K} & \mathbf{K} \\ \mathbf{K} & \mathbf{K} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \end{bmatrix} - \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \end{bmatrix}^T \begin{bmatrix} \mathbf{z} - E\boldsymbol{\epsilon} \\ \mathbf{z} + E\boldsymbol{\epsilon}^* \end{bmatrix} \\ \text{such that:} \quad & \mathbf{0} \leq \boldsymbol{\beta} \leq \tfrac{C}{N}\mathbf{t} \\ & -\tfrac{C}{N}\mathbf{t}^* \leq \boldsymbol{\beta}^* \leq \mathbf{0} \\ & \mathbf{1}^T\boldsymbol{\beta} - \mathbf{1}^T\boldsymbol{\beta}^* = 0 \end{aligned} \tag{4.25}$$

Now, $\begin{bmatrix} \mathbf{K} & \mathbf{K} \\ \mathbf{K} & \mathbf{K} \end{bmatrix}$ is positive semidefinite (as $\mathbf{K}$ is positive semidefinite) and the constraints are linear, so the (4.23) will be convex. I choose to consider (4.23) because, like (4.12), it does not present any real computational difficulty and also makes the relationship between pattern recognition and regression forms of the SVM clearer. Also, the number of variables $\boldsymbol{\alpha}$ is half the number contained in (4.25).

As for pattern recognition, the equality constraint present in (4.23) may be removed by considering instead the partially dual form, which is:

$$\begin{aligned}&\min_{\boldsymbol{\alpha}} \max_{b} QL_1\left(\boldsymbol{\alpha}, b\right) = \tfrac{1}{2} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} + E \left| \begin{matrix} b \\ \boldsymbol{\alpha} \end{matrix} \right|^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{bmatrix} \\ &\text{such that:} \quad -\tfrac{C}{N}\mathbf{t}^* \leq \boldsymbol{\alpha} \leq \tfrac{C}{N}\mathbf{t}\end{aligned} \tag{4.26}$$

where:

$$\mathbf{H} = \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{bmatrix}$$

The stationary or KKT (Karush-Kuhn-Tucker) conditions for the program (4.13), see [40], are as follows:

$$\begin{aligned} &\boldsymbol{\alpha} \geq -\tfrac{C}{N}\mathbf{t}^* \\ &\boldsymbol{\alpha} \leq \tfrac{C}{N}\mathbf{t} \\ &f = 0 \\ &e_i \begin{cases} \in \left[z_i - E\epsilon_i, z_i + E\epsilon_i^*\right] & \text{if } \alpha_i = 0 \\ = z_i - E\epsilon_i & \text{if } 0 < \alpha_i < \tfrac{C}{N}t_i \\ \leq z_i - E\epsilon_i & \text{if } \alpha_i = \tfrac{C}{N}t_i \\ = z_i + E\epsilon_i^* & \text{if } -\tfrac{C}{N}t_i^* < \alpha_i < 0 \\ \geq z_i + E\epsilon_i^* & \text{if } \alpha_i = -\tfrac{C}{N}t_i^* \end{cases} \end{aligned} \tag{4.27}$$

where:

$$\begin{bmatrix} f \\ \mathbf{e} \end{bmatrix} = \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} \tag{4.28}$$

Once again, $e_i = g\left(\mathbf{x}_i\right)$ for all training vectors.

Chapter 5

EXTENSIONS TO THE BASIC SVM METHOD

With sufficient thrust, pigs fly just fine. However, this is not necessarily a good idea. It is hard to be sure where they are going to land, and it could be dangerous sitting under them as they fly overhead.

– RFC 1925

5.1 Regression with Inequalities

One simple extension of the SVM regression problem is the inclusion of inequalities in the training data. For example, it may be necessary to impose constraints on a system so that the output, given a particular input, stays within certain physically realisable bounds. In this section, I describe how this may be achieved.

First, extend the training set thusly.

$$\begin{aligned}
&\mathbf{Y} = \{(\mathbf{x}_1, z_1), (\mathbf{x}_2, z_2), \ldots, (\mathbf{x}_N, z_N)\} \\
&\mathbf{Y} = \mathbf{Y}_= \cup \mathbf{Y}_\geq \cup \mathbf{Y}_\leq \\
&\mathbf{Y}_\geq \cap \mathbf{Y}_\leq = \mathbf{Y}_= \cap \mathbf{Y}_\leq = \mathbf{Y}_= \cap \mathbf{Y}_\geq = \emptyset \\
&\mathbf{x}_i \in \Re^{d_L} \\
&z_i \in \Re
\end{aligned} \tag{5.1}$$

The set $\mathbf{Y}_=$ is the usual regression training set. Points in the set $\mathbf{Y}_\geq$ are those points which are to be used as lower bounds (that is, it is required that $g(\lambda)(\mathbf{x}_i) \geq z_i - E\epsilon_i$, where $E\epsilon_i$ here is a "margin of error" akin to the ϵ-insensitive region of (4.18)) and likewise points in $\mathbf{Y}_\leq$ are to be used as upper bounds (i.e. $g(\lambda)(\mathbf{x}_i) \leq$

$z_i + E\epsilon_i)$. The primal is:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} & R_1\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \\
\text{such that: } & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\
& \boldsymbol{\xi} \geq \mathbf{0} \\
& \boldsymbol{\xi}^* \geq \mathbf{0}
\end{aligned} \tag{5.2}$$

where $\epsilon_i \geq 0$, $\epsilon_i^* \geq 0$ for all $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ (there are two reasons for this assumption. Practically, it is a convenience factor which makes forming the dual problem significantly simpler. Philosophically, as regression with inequalities is an extension of the standard regression problem, it is required to maintain consistency between the two). No constraint is placed on ϵ_i when $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$.

The maximin form of (5.2) is:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*} L_1 = & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=\cup\mathbf{Y}_\geq} \beta_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \right) - z_i + E\epsilon_i + \xi_i \right) \\
& - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=\cup\mathbf{Y}_\leq} \beta_i^* \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \right) - z_i - E\epsilon_i^* - \xi_i^* \right) \\
\text{such that: } & \boldsymbol{\beta} \geq \mathbf{0} \\
& \boldsymbol{\beta}^* \leq \mathbf{0} \\
& \boldsymbol{\gamma} \geq \mathbf{0} \\
& \boldsymbol{\gamma}^* \geq \mathbf{0}
\end{aligned} \tag{5.3}$$

As for regression, it may be noted that for all $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ only one of the constraints:

$$\begin{aligned}
\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b &\geq z_i - E\epsilon_i - \xi_i \\
\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b &\leq z_i + E\epsilon_i^* + \xi_i^*
\end{aligned}$$

may be met exactly (i.e. left side equals right side).[1] So at least one of β_i and β_i^*

[1]Technically, this assumes that $E\left(\epsilon_i + \epsilon_i^*\right) > 0$ for all $1 \leq i \leq N$ such that $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$, which

will be zero for all $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$. Likewise, at least one of ξ_i and ξ_i^* will be zero for all $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$. Hence it is possible to define the vectors $\boldsymbol{\alpha}$ (alternatively written $\boldsymbol{\beta}^{(*)}$), $\boldsymbol{\xi}^{(*)}$, $\boldsymbol{\gamma}^{(*)}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\xi_i^{(*)} = \begin{cases} \xi_i & \text{if } (\xi_i^* = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_=) \vee (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ \xi_i^* & \text{otherwise} \end{cases}$$

$$\gamma_i^{(*)} = \begin{cases} \gamma_i & \text{if } (\xi_i^* = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_=) \vee (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ \gamma_i^* & \text{otherwise} \end{cases}$$

$$\epsilon_i^{(*)} = \begin{cases} \epsilon_i & \text{if } (\beta_i^* = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_=) \vee (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ \epsilon_i^* & \text{otherwise} \end{cases}$$

$$\alpha_i = \begin{cases} \beta_i & \text{if } (\beta_i^* = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_=) \vee (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ \beta_i^* & \text{otherwise} \end{cases}$$

such that the maximin problem may be re-written as:

$$\begin{aligned} \min_{\mathbf{w}, b, \boldsymbol{\xi}^{(*)}} \max_{\boldsymbol{\alpha}, \boldsymbol{\gamma}^{(*)}} L_1 = {} & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^{(*)T}\boldsymbol{\xi}^{(*)} - \boldsymbol{\gamma}^{(*)T}\boldsymbol{\xi}^{(*)} \\ & - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=\cup\mathbf{Y}_\geq,\alpha_i\geq 0} \alpha_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + E\epsilon_i^{(*)} + \xi_i^{(*)} \right) \\ & - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=\cup\mathbf{Y}_\leq,\alpha_i< 0} \alpha_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i - E\epsilon_i^{(*)} - \xi_i^{(*)} \right) \\ \text{such that: } & \alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ & \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \\ & \boldsymbol{\gamma}^{(*)} \geq \mathbf{0} \end{aligned}$$

Following the usual method (details in appendix A) to remove the primal variables, it is not difficult to show that the dual form of regression with inequalities

may not be true in all cases. For a complete derivation including this case, see appendix A. This has no effect on the end result.

is:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} Q_1(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon}^{(*)} \\
&\text{such that:} \quad -\tfrac{C}{N}t_i^* \le \alpha_i \le 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\le \\
&\qquad\qquad\quad 0 \le \alpha_i \le \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\ge \\
&\qquad\qquad\quad -\tfrac{C}{N}t_i^* \le \alpha_i \le \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \\
&\qquad\qquad\quad \mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned} \tag{5.4}$$

As was the case for regression, the presence of both $|\boldsymbol{\alpha}|$ and the vector $\boldsymbol{\epsilon}^{(*)}$ (which is implicitly dependent on $\operatorname{sgn}(\boldsymbol{\alpha})$) is merely a convenience factor. They may be removed by appropriately re-expressing (5.4) in terms of $\boldsymbol{\beta}$ and $\boldsymbol{\beta}^*$. The form of $g(\lambda)$ and b is the same as for regression.

The partially dual form ($\mathbf{H}$ as before) of the problem is:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} \max_{b} \mathcal{L}_1(\boldsymbol{\alpha}, b) = \tfrac{1}{2}\begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} + E \left| \begin{matrix} b \\ \boldsymbol{\alpha} \end{matrix} \right|^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{bmatrix} \\
&\text{such that:} \quad -\tfrac{C}{N}t_i^* \le \alpha_i \le 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\le \\
&\qquad\qquad\quad 0 \le \alpha_i \le \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\ge \\
&\qquad\qquad\quad -\tfrac{C}{N}t_i^* \le \alpha_i \le \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_=
\end{aligned} \tag{5.5}$$

and has the KKT conditions:

$$\begin{aligned}
&\alpha_i \le \begin{cases} \tfrac{C}{N}t_i & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\ge \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\le \end{cases} \\
&\alpha_i \ge \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\ge \\ -\tfrac{C}{N}t_i^* & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\le \end{cases} \\
&e_i \begin{cases} \ge z_i - E\epsilon_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\ge \\ = z_i - E\epsilon_i & \text{if } 0 < \alpha_i < \tfrac{C}{N}t_i \\ \le z_i - E\epsilon_i & \text{if } \alpha_i = \tfrac{C}{N}t_i \\ \le z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\le \\ = z_i + E\epsilon_i^* & \text{if } -\tfrac{C}{N}t_i^* < \alpha_i < 0 \\ \ge z_i + E\epsilon_i^* & \text{if } \alpha_i = -\tfrac{C}{N}t_i^* \end{cases} \\
&f = 0
\end{aligned} \tag{5.6}$$

where:

$$\begin{bmatrix} f \\ \mathbf{e} \end{bmatrix} = \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} \tag{5.7}$$

As before, $e_i = g(\mathbf{x}_i)$.

5.1.1 Regression With Inequalities as a Superset of Regression and Pattern Recognition

It turns out that both regression and pattern recognition may be done under the general framework of regression with inequalities - that is, the regression with inequalities SVM formulation is a *superset* of the regression and pattern classification SVM formulations.

Firstly, note that the final form of $g(\lambda)$ (in terms of $\boldsymbol{\alpha}$ and b) is the same for all the types of SVM considered thus far. Hence to show equivalence all that is required is to show equivalence of $\boldsymbol{\alpha}$ and b obtained using the different methods.

Now consider the regression with inequalities primal, (5.2). In the special case when $\mathbf{Y}_{\leq} = \mathbf{Y}_{\geq} = \emptyset$ this may be reduced to:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \ & R_1(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \\
\text{such that:} \ & \mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \\
& \mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \\
& \boldsymbol{\xi} \geq \mathbf{0} \\
& \boldsymbol{\xi}^* \geq \mathbf{0}
\end{aligned}$$

But this is precisely identical to the regression primal problem, (4.21), and so will have same global solution (or solutions). Clearly, then, it is possible to use the general regression with inequalities SVM for the task of regression by the simple expediency of choosing $\mathbf{Y}_{\leq} = \mathbf{Y}_{\geq} = \emptyset$.

Alternatively, consider the regression with inequalities primal, (5.2), but this time with the requirements $\mathbf{Y}_= = \emptyset$, $\mathbf{z} = \mathbf{0}$ and $\boldsymbol{\epsilon}, \boldsymbol{\epsilon}^* \leq \mathbf{0}$. Then the primal becomes:

$$\begin{aligned}
\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \; & R_1 \left(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^* \right) = \tfrac{1}{2} \mathbf{w}^T \mathbf{w} + \tfrac{C}{N} \mathbf{t}^T \boldsymbol{\xi} + \tfrac{C}{N} \mathbf{t}^{*T} \boldsymbol{\xi}^* \\
\text{such that:} \; & \mathbf{w}^T \boldsymbol{\varphi} \left(\mathbf{x}_i \right) + b \geq -E \epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, 0 \right) \in \mathbf{Y}_\geq \\
& \mathbf{w}^T \boldsymbol{\varphi} \left(\mathbf{x}_i \right) + b \leq E \epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, 0 \right) \in \mathbf{Y}_\leq \\
& \boldsymbol{\xi} \geq \mathbf{0} \\
& \boldsymbol{\xi}^* \geq \mathbf{0}
\end{aligned}$$

which is precisely identical to the pattern classification primal, (4.10), except for a minor notational inconsistency in the definition of the sets $\mathbf{Y}_\geq$ and $\mathbf{Y}_\leq$, namely that in the pattern classification formulation, elements of these sets are vectors $\mathbf{x}_i$, whereas for regression with inequalities, these elements are *pairs* of the form $(\mathbf{x}_i, z_i)$. This notational inconsistency may be overcome by defining the shorthand that $\mathbf{x}_i \in \mathbf{Y}_{\overset{\geq}{\underset{\leq}{=}}}$ is equivalent to $(\mathbf{x}_i, z_i) \in \mathbf{Y}_{\overset{\geq}{\underset{\leq}{=}}}$ - that is, unless z_i is given explicitly, assume that it is zero.

To summarise:

- A (pure) regression SVM (or SV regressor (SVR)) is equivalent to a regression with inequalities SVM where $\mathbf{Y}_\leq = \mathbf{Y}_\geq = \emptyset$.

- A pattern classification SVM (or SV classifier (SVC)) is equivalent to a regression with inequalities SVM where $\mathbf{Y}_= = \emptyset$, $\mathbf{z} = \mathbf{0}$ and $\boldsymbol{\epsilon}, \boldsymbol{\epsilon}^* \leq \mathbf{0}$.

An important implication of this is that many results and techniques may be applied directly to regression with inequalities, rather than separately to pattern classification and regression, thereby halving the amount of work needed.

Given this "equivalence" between the various forms of SVM, the obvious question is: what is the relationship between the max-margin principle (for pattern recognition) and gradient flattening (for regression)? To answer this question, consider figure 5.1, which shows the graph of $g(\mathbf{x})$ as distance above (or below) feature space. For the two boundary vectors shown, the perpendicular distance between the support vector $\mathbf{x}_i$ in feature space and the graph $g(\mathbf{x}_i)$ is fixed. If the margin of separation is ρ then the gradient is $\frac{\epsilon_i^{(*)} + \epsilon_j^{(*)}}{\rho}$. Clearly, minimising this gradient will

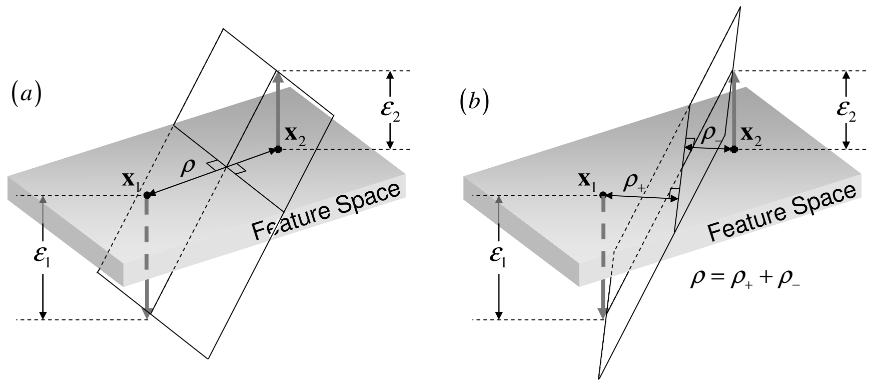


Figure 5.1: Relationship between the concepts of max-margin (for the pattern recognition case and graph flattening (for the regression case). (a) shows the max-margin case, and (b) a non-optimal margin. Note that the gradient of the surface is smaller in case (a) - i.e. it is "flatter".

maximise ρ, and vice-versa. This explains the connection between the two concepts.

5.2 Fixed Bias SVMs

In some cases, it is convenient to make the bias term b in the SVM problem a constant, rather than a variable for optimisation. Indeed, in section 5.3 it will be shown that the bias term b is essentially superfluous for the pattern recognition problem if the kernel function is selected appropriately.

The effect of making b constant is minimal (for details, see appendix A). The primal problem (5.2) becomes:

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi},\boldsymbol{\xi}^*} & R_1\left(\mathbf{w},\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \\ \text{such that:} \quad & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\ & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\ & \boldsymbol{\xi} \geq 0 \\ & \boldsymbol{\xi}^* \geq 0 \end{aligned} \tag{5.8}$$

which is unchanged, except that b is now a constant. Skipping the intermediate steps, the dual problem is:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} Q_1(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathbf{1}b - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon}^{(*)} \\
&\text{such that: } -\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
&\qquad\qquad\quad 0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\
&\qquad\qquad\quad -\tfrac{C}{N}t_i^* \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=}
\end{aligned} \tag{5.9}$$

There is no partially dual form in this case. The KKT conditions are as follows:

$$\begin{aligned}
&\alpha_i \leq \begin{cases} \frac{C}{N}t_i & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \end{cases} \\
&\alpha_i \geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ -\frac{C}{N}t_i^* & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \end{cases} \\
&e_i \begin{cases} \geq z_i - E\epsilon_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ = z_i - E\epsilon_i & \text{if } 0 < \alpha_i < \frac{C}{N}t_i \\ \leq z_i - E\epsilon_i & \text{if } \alpha_i = \frac{C}{N}t_i \\ \leq z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \\ = z_i + E\epsilon_i^* & \text{if } -\frac{C}{N}t_i^* < \alpha_i < 0 \\ \geq z_i + E\epsilon_i^* & \text{if } \alpha_i = -\frac{C}{N}t_i^* \end{cases}
\end{aligned} \tag{5.10}$$

where:

$$\mathbf{e} = \mathbf{K}\boldsymbol{\alpha} + \mathbf{1}b \tag{5.11}$$

As before, $e_i = g(\mathbf{x}_i)$.

5.3 Automatic Biasing

Consider the primal form of the SVM problem, (5.2). As was shown in section 4.2, the first term in the cost is actually a regularisation term for $\mathbf{w}$. In [55], Mangasarian showed that by adding an additional regularisation term for the bias, b, it is possible to simplify the dual form (5.4) so that the equality constraint $\mathbf{1}^T\boldsymbol{\alpha} = 0$ is removed (making the partially dual form of the problem superfluous).

Mangasarian's extension of (5.2) is:

$$\begin{aligned}
&\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} R_1\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{\beta}{2}b^2 + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \\
&\text{such that:} \quad \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
&\qquad\qquad\quad \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\
&\qquad\qquad\quad \boldsymbol{\xi} \geq 0 \\
&\qquad\qquad\quad \boldsymbol{\xi}^* \geq 0
\end{aligned} \tag{5.12}$$

and differs from (5.2) by the addition of the new term $\frac{\beta}{2}b^2$ ($\beta > 0$), which is a regularisation term for b (Mangasarian further required that $\beta = 1$). It is essentially trivial (appendix A) to show that the dual form of this is:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} Q_1\left(\boldsymbol{\alpha}\right) = \tfrac{1}{2}\boldsymbol{\alpha}^T\left(\mathbf{K} + \tfrac{1}{\beta}\mathbf{1}\mathbf{1}^T\right)\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon}^{(*)} \\
&\text{such that:} \quad -\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\leq \\
&\qquad\qquad\quad 0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\geq \\
&\qquad\qquad\quad -\tfrac{C}{N}t_i^* \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_=
\end{aligned} \tag{5.13}$$

where $b = \frac{1}{\beta}\mathbf{1}^T\boldsymbol{\alpha}$. The discriminant $g\left(\lambda\right)$ takes its usual form (4.24). As (5.13) contains no equality constraints, there is no need to form the partial dual. This form has the advantage of the partially dual form of the standard SVM, insofar as it lacks an equality constraint. Furthermore, the hessian in this case is positive semidefinite, not indefinite as was the case for the partially dual form, which makes optimisation marginally simpler.

While this form certainly has advantages over the usual form, some care is required. In particular, the use of automatic biasing in the context of SVM regression is not advised for reasons that will be discussed in the following section.

5.3.1 Geometric Interpretation

In [79], I showed how this idea may be interpreted geometrically. Given a feature map $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$, an augmented feature map $\boldsymbol{\varphi}_b : \Re^{d_L} \to \Re^{d_H+1}$ may be defined to thusly:

$$\boldsymbol{\varphi}_b(\mathbf{x}) = \begin{bmatrix} \boldsymbol{\varphi}(\mathbf{x}) \\ 1 \end{bmatrix}$$

by identifying $\mathbf{w}_b = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$. Using this notation, it follows that:

$$\begin{aligned} g(\lambda)(\mathbf{x}) &= \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}) + b \\ &= \mathbf{w}_b^T \boldsymbol{\varphi}_b(\mathbf{x}) \end{aligned}$$

and hence the modified primal problem (5.12) may be re-written:

$$\begin{aligned} \min_{\mathbf{w}_b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} R_1(\mathbf{w}_b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) &= \tfrac{1}{2}\mathbf{w}_b^T \mathbf{Q} \mathbf{w}_b + \tfrac{C}{N}\mathbf{t}^T \boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T} \boldsymbol{\xi}^* \\ \text{such that: } & \mathbf{w}_b^T \boldsymbol{\varphi}_b(\mathbf{x}_i) \geq z_i - E\epsilon_i - \xi_i \forall (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\ & \mathbf{w}_b^T \boldsymbol{\varphi}_b(\mathbf{x}_i) \leq z_i + E\epsilon_i^* + \xi_i^* \forall (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\ & \boldsymbol{\xi} \geq 0 \\ & \boldsymbol{\xi}^* \geq 0 \end{aligned} \tag{5.14}$$

where:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0}^T & \beta \end{bmatrix}$$

Clearly, $\frac{1}{2}\mathbf{w}_b^T \mathbf{Q} \mathbf{w}_b = \frac{1}{2}\|\mathbf{w}_b\|_{\mathbf{Q}}^2$ is just a mildly non-Euclidean norm (and is precisely Euclidean if $\beta = 1$), and so (5.14) is essentially just the usual SVM primal with explicit bias terms removed (as biasing is now implicit in the feature map). Geometrically, this formulation is shown in figure 5.2 from a pattern recognition standpoint (the regression interpretation will be dealt with later). In this diagram, the original "feature space" is now a hyperplane in augmented feature space, lying at a fixed point on the $(d_H + 1)^{th}$ axis. The decision surface associated with $g(\lambda)$ in augmented feature space passes through the origin in this space, but the intersection with (non-augmented) feature space will *not* pass though the origin in this (non-augmented) feature space. Thus there is an *effective (automatic) bias* in feature space, which is selected implicitly when constructing the decision surface in

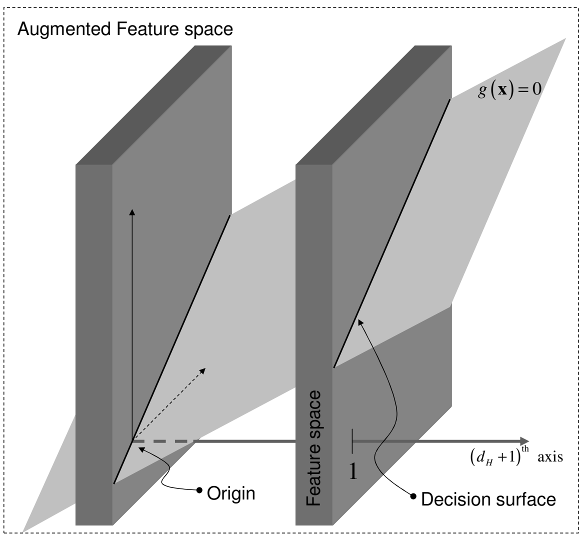


Figure 5.2: Geometry of Automatic Biasing.

augmented feature space.

It is informative to consider the relationship between the margin of separation ρ_b in augmented feature space and the margin of separation ρ in non-augmented feature space. From (4.5):

$$\begin{aligned}\rho &= \frac{E\left(-\epsilon_i^{(*)}+-\epsilon_j^{(*)}\right)}{\|\mathbf{w}\|_2} \\ \rho_b &= \frac{E\left(-\epsilon_i^{(*)}+-\epsilon_j^{(*)}\right)}{\|\mathbf{w}_b\|_{\mathbf{Q}}}\end{aligned}$$

Hence:

$$\rho = \frac{1}{\sqrt{1-\beta\frac{\rho_b^2 b^2}{E^2\left(-\epsilon_i^{(*)}+-\epsilon_j^{(*)}\right)^2}}}\rho_b \tag{5.15}$$

$$\rho_b = \frac{1}{\sqrt{1+\beta\frac{\rho^2 b^2}{E^2\left(-\epsilon_i^{(*)}+-\epsilon_j^{(*)}\right)^2}}}\rho \tag{5.16}$$

Now, in the pattern recognition case, it is my experience that the affect of including an “automatic biasing” term is essentially neutral in terms of performance (although the parameter C may need to be adjusted to compensate for the narrower margin of separation in augmented feature space). In the regression case, however, this is not true. This is because, for the regression case, not only the sign, but also the magnitude of $g(\lambda)$ is of interest. The βb^2 term in the cost function (5.13) will pull the mean of $\{g(\lambda)(\mathbf{x}_i)\}$ toward 0, causing a systematic bias in the result.

Therefore use of automatic biasing for regression is not recommended.

From (5.16) it can bee seen that:

$$\lim_{\beta \to 0} \rho_b = \rho$$

5.3.2 The Connection with Fixed Bias SVMs

Consider the special case of automatic biasing where $\beta = 1$. The kernel function associated with the augmented feature map is then:

$$K_b\left(\mathbf{x},\mathbf{y}\right) = K\left(\mathbf{x},\mathbf{y}\right) + 1$$

so the dual form of the automatically biased SVM may be re-written:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} Q_1\left(\boldsymbol{\alpha}\right) &= \tfrac{1}{2}\boldsymbol{\alpha}^T \mathbf{K}_b \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{z} + E|\boldsymbol{\alpha}|^T \boldsymbol{\epsilon}^{(*)} \\ \text{such that:} \quad & -\tfrac{C}{N} t_i^* \leq \alpha_i \leq 0 \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{\leq} \\ & 0 \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{\geq} \\ & -\tfrac{C}{N} t_i^* \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{=} \end{aligned} \tag{5.17}$$

where $\mathbf{K}_b$ is the kernel matrix associated with the augmented kernel mapping (so $K_{bi,j} = K_b\left(\mathbf{x}_i, \mathbf{x}_j\right) = K\left(\mathbf{x}_i, \mathbf{x}_j\right) + 1$). Furthermore, it is not difficult to show that:

$$g\left(\lambda\right)\left(\mathbf{y}\right) = \sum_{\substack{\mathbf{x}_i \in \mathbf{X} \\ \alpha_i \neq 0}} \alpha_i K_b\left(\mathbf{x}_i, \mathbf{y}\right)$$

But this is identical to the fixed bias SVM constructed using the kernel function $K_b\left(\mathbf{x},\mathbf{y}\right)$ with fixed bias $b = 0$. So, in the special case $\beta = 1$, a fixed bias SVM with $b = 0$ using the augmented kernel function will give the same result as an automatically biased SVM using the original kernel.

Consider a Mercer kernel $K\left(\mathbf{x},\mathbf{y}\right)$ (for example, $K\left(\mathbf{x},\mathbf{y}\right) = \left(1 + \mathbf{x}^T\mathbf{y}\right)^n$) such that there exists a Mercer kernel $K_{-b}\left(\mathbf{x},\mathbf{y}\right)$ and an associated positive constant $\eta > 0$ such that:

$$K\left(\mathbf{x},\mathbf{y}\right) = K_{-b}\left(\mathbf{x},\mathbf{y}\right) + \eta \tag{5.18}$$

A fixed bias SVM with $b = 0$ constructed using this Mercer kernel $K(\mathbf{x}, \mathbf{y})$ will be equivalent to an automatically biased SVM with $\beta = 1$ constructed using the Mercer kernel $K_{-b}(\mathbf{x}, \mathbf{y})$ (the constant term η makes no practical difference, so long as $\eta > 0$. Essentially, $b \to \eta b$). So, in a sense, the fixed bias SVM so constructed contains an effective automatic (non-fixed) biasing term that is *implicit* in the Mercer kernel. Indeed, this may be said of any fixed bias SVM constructed using a Mercer kernel that may be split in this way.

5.4 Generalised Empirical Risk

Consider the primal form of the SVM regression problem, (4.21). If $\boldsymbol{\epsilon} = \mathbf{0}$ and $\mathbf{t} = \mathbf{1}$ then the empirical risk component of this is just:

$$R_{emp}\left(\lambda \middle| \mathbf{Y}\right) = \frac{1}{N} \sum_{i:(\mathbf{x}_i, z_i) \in \mathbf{Y}} \left| \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b - z_i \right|$$

From table 3.1, it can be seen that this cost function is optimal if the training set is affected by Laplacian noise. If the training set is affected by non-Laplacian noise, however, better results may be achieved by selecting a different cost function. However, the selection process is somewhat limited by the practical requirement that the resultant optimisation problem should not be too difficult or impractical to solve. Therefore the process of selecting a cost function is a trade-off between optimality and practicality, which is what motivated the original selection of:

$$R_{emp}\left(\lambda \middle| \mathbf{Y}\right) = \frac{1}{N} \sum_{i:(\mathbf{x}_i, z_i) \in \mathbf{Y}} t_i \left| \mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b - z_i \right|_{E\epsilon_i}$$

This is not, however, the only usable choice, nor even necessarily the best. Suykens [87], for example, suggests a least-squares cost function which is optimal for Gaussian noise. More generally still, Smola et. al. [85] consider a whole family of symmetric, convex loss functions with discontinuity at $\pm\epsilon$, which may be viewed as the general extension of the ϵ-insensitive cost function discussed in the previous chapter. In general, when using these more general forms, one is faced with a trade-off between accuracy and practicallity. In particular, the optimisation that may arise when attempting to optimally match the empirical risk component of the cost function and the form of the training noise may be too complex to be solved practically, and hence a compromise may be necessary to make the problem practical. Similarly, as will be seen in chapter 6, to maintain sparsity it may be necessary to choose a non-optimal empirical risk component.

In the present section, I will introduce the formulation given [85] and then concentrate in particular on the monomial form of this, which will be analysed in some detail in chapter 6.

5.4.1 General Convex Cost Functions

In [85], the following cost function is considered:

$$R_{emp}\left(\lambda|\,\mathbf{Y}\right) = \frac{1}{N} \sum_{i:(\mathbf{x}_i, z_i)\in\mathbf{Y}} t_i d\left(\left|\mathbf{w}^T \boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b - z_i\right|_{E\epsilon_i}\right)$$

or, equivalently:

$$R_{emp}\left(\lambda|\,\mathbf{Y}\right) = \frac{1}{N} \sum_{i:(\mathbf{x}_i, z_i)\in\mathbf{Y}} t_i \left(d\left(\xi_i\right) + d\left(\xi_i^*\right)\right)$$

where $d \in C^1$ is a convex function with $d\left(0\right) = 0$. In this form, ϵ_i and t_i serve the same purpose as in the standard formulation, except that the form of the cost function outside of the "dead-zone" is now arbitrary. The special case $d\left(\xi\right) = \xi$ gives the standard formulation. For complete generality, the following extension will be considered here:

$$R_{emp}\left(\lambda|\,\mathbf{Y}\right) = \frac{1}{N} \sum_{i:(\mathbf{x}_i, z_i)\in\mathbf{Y}} t_i d\left(\xi_i\right) + \frac{1}{N} \sum_{i:(\mathbf{x}_i, z_i)\in\mathbf{Y}} t_i^* d^*\left(\xi_i^*\right)$$

where $d, d^* \in C^1$ are both convex functions with $d\left(0\right) = d^*\left(0\right) = 0$.

The associated primal form of the SVM problem is:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} {}_{d}R_{d^*}\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) &= \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\sum_{i=1}^{N} t_i d\left(\xi_i\right) + \tfrac{C}{N}\sum_{i=1}^{N} t_i^* d^*\left(\xi_i^*\right) \\
\text{such that:}\quad & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\
& \boldsymbol{\xi} \geq 0 \\
& \boldsymbol{\xi}^* \geq 0
\end{aligned} \tag{5.19}$$

In the general case $d\left(\xi\right) \neq d^*\left(\xi\right)$ it can be seen that the variable $\boldsymbol{\epsilon}^*$ is no longer superfluous. Indeed, it may be necessary to choose $\boldsymbol{\epsilon}^* \neq \boldsymbol{\epsilon}$ so-as to *not* skew the training vector $\mathbf{z}$.

The maximin form of (5.19) is:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*} {}_{d}L_{d^*} = {} & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\sum_{i=1}^{N} t_i d\left(\xi_i\right) + \tfrac{C}{N}\sum_{i=1}^{N} t_i^* d^*\left(\xi_i^*\right) - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i:\left(\mathbf{x}_i,z_i\right)\in\mathbf{Y}_=\cup\mathbf{Y}_\geq} \beta_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i + E\epsilon_i + \xi_i\right) \\
& - \sum_{i:\left(\mathbf{x}_i,z_i\right)\in\mathbf{Y}_=\cup\mathbf{Y}_\leq} \beta_i^* \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i - E\epsilon_i^* - \xi_i^*\right) \\
\text{such that:}\quad & \boldsymbol{\beta} \geq \mathbf{0} \\
& \boldsymbol{\beta}^* \leq \mathbf{0} \\
& \boldsymbol{\gamma} \geq \mathbf{0} \\
& \boldsymbol{\gamma}^* \geq \mathbf{0}
\end{aligned} \tag{5.20}$$

Skipping the usual workings (the derivation follows much the same template as previous derivations of duals - details can be found in appendix A), the dual can be shown to be:

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} {}_{d}Q_{d^*}\left(\boldsymbol{\alpha}\right) = {} & \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E\left|\boldsymbol{\alpha}\right|^T\boldsymbol{\epsilon}^{(*)} - \tfrac{C}{N}\sum_{i=1}^{N} t_i T\left(\xi_i\right) - \tfrac{C}{N}\sum_{i=1}^{N} t_i^* T^*\left(\xi_i^*\right) \\
\text{such that:}\quad & \alpha_i \leq 0 \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\leq \\
& \alpha_i \geq 0 \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\geq \\
& \mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned} \tag{5.21}$$

where:

$$\xi_i = \begin{cases} \inf\left\{ \xi \left| \, \frac{C}{N} t_i \frac{\partial d}{\partial \xi}\left(\xi\right) > \left|\alpha_i\right| \right. \right\} & \text{if } \alpha_i > 0 \\ 0 & \text{if } \alpha_i \leq 0 \end{cases}$$
$$\xi_i^* = \begin{cases} 0 & \text{if } \alpha_i \geq 0 \\ \inf\left\{ \xi \left| \, \frac{C}{N} t_i^* \frac{\partial d^*}{\partial \xi}\left(\xi\right) > \left|\alpha_i\right| \right. \right\} & \text{if } \alpha_i < 0 \end{cases}$$

and:

$$T\left(\xi\right) = d\left(\xi\right) - \xi \frac{\partial d}{\partial \xi}\left(\xi\right)$$
$$T^*\left(\xi\right) = d^*\left(\xi\right) - \xi \frac{\partial d^*}{\partial \xi}\left(\xi\right)$$

It will be noted that this general form is more complex than previous forms of SVM given. This is the price paid for generality. In [85] an algorithm is given for solving this general optimisation problem. However, in the present thesis I will not be looking at this general form in any detail. Instead, I consider some special cases, focussing in particular on the case where $d\left(\xi\right) = d^*\left(\xi\right) = \frac{1}{q}\xi^q$, which in the special case $q = 2$, $E = 0$ corresponds to the LS-SVM [87]. Unless otherwise stated, it is assumed that $d\left(\xi\right) = d^*\left(\xi\right)$.

5.4.2 Monomial ϵ-Insensitive Risk and LS-SVMs

One variant of the standard SVM method known to have a particularly simple dual form is the LS-SVM [87], which (assuming $E = 0$) uses the cost function $d\left(\xi\right) = \frac{1}{2}\xi^2$. This is optimal (in the maximum likelihood sense) if the training data is effected by Gaussian noise. More generally, in [79] we proposed the *monomial cost function* $d\left(\xi\right) = \frac{1}{q}\xi^q$, where $q \in \mathbb{Z}^+$ is a constant. This corresponds to the ML cost function for degree q polynomial noise, where polynomial noise of degree q (unit variance, zero mean assumed) is characterised by the density function $p_{\mathrm{std}}\left(\tau\right) = c_q e^{-c_q' |\tau|^q}$, where:

$$c_q = \frac{1}{2} \frac{q}{\Gamma\left(\frac{1}{q}\right)} \sqrt{\frac{\Gamma\left(\frac{3}{q}\right)}{\Gamma\left(\frac{1}{q}\right)}}$$
$$c_q' = \left(\sqrt{\frac{\Gamma\left(\frac{3}{q}\right)}{\Gamma\left(\frac{1}{q}\right)}} \right)^q$$

In terms of the usual slack variables, the SVM primal problem corresponding to the monomial cost function may be written:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} & R_q\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{qN}\sum_{i=1}^{N} t_i\xi_i^q + \tfrac{C}{qN}\sum_{i=1}^{N} t_i^*\xi_i^{*q} \\
\text{such that: } & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\
& \boldsymbol{\xi} \geq 0 \\
& \boldsymbol{\xi}^* \geq 0
\end{aligned} \tag{5.22}$$

I shall refer to a regressor of form (5.22) as a monomial ϵ-SVM. Unfortunately, for the general case $q \neq 1, 2$ the dual form of (5.22) is rather complicated. For this reason I will restrict myself to the special case $q = 2$ (quadric ϵ-SVR), in which case it turns out that the dual problem is mathematically "nice". Moreover, as will be seen in chapter 6, the quadric ϵ-SVR retains many of the useful properties of the LS-SVR with additional advantages in the presense of non-gaussian training noise (the parameter E may be tuned to maximise the efficiency of the emulator) and also large data sets (in this case, E is useful for ensuring sparsity in the solution).

Before constructing the dual form of (5.22), it is convenient to show that if $q = 2$ the positivity constraints $\boldsymbol{\xi}, \boldsymbol{\xi}^* \geq \mathbf{0}$ in the primal are superfluous, giving the simplified problem:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} & R_2\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{2N}\sum_{i=1}^{N} t_i\xi_i^2 + \tfrac{C}{2N}\sum_{i=1}^{N} t_i^*\xi_i^{*2} \\
\text{such that: } & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq
\end{aligned} \tag{5.23}$$

Theorem 5.1. *For every solution* $\left(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*\right)$ *of (5.23),* $\boldsymbol{\xi}, \boldsymbol{\xi}^* \geq \mathbf{0}$.

Proof. Suppose there exists a solution $\left(\bar{\mathbf{w}}, \bar{b}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\xi}}^*\right)$ of (5.23) such that $\bar{\xi}_i < 0$ for some $1 \leq i \leq N$. Then for all other $\left(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*\right)$ satisfying the constraints contained in (5.23), $R_2\left(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*\right) \geq R_2\left(\bar{\mathbf{w}}, \bar{b}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\xi}}^*\right)$ by definition.

Consider $(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*)$ defined by:

$$\begin{aligned}
\mathbf{w} &= \bar{\mathbf{w}} \\
b &= \bar{b} \\
\xi_j &= \begin{cases} \bar{\xi}_j & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \\
\boldsymbol{\xi}^* &= \bar{\boldsymbol{\xi}}^*
\end{aligned}$$

First, note that as $\left(\bar{\mathbf{w}}^T \boldsymbol{\varphi}(\mathbf{x}_i) + \bar{b}\right) \geq z_i - E\epsilon_i - \bar{\xi}_i$, $\mathbf{w} = \bar{\mathbf{w}}$, $b = \bar{b}$, $\bar{\xi}_i < 0$ and $\xi_i = 0$, it follows that $\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b\right) \geq z_i - E\epsilon_i - \xi_i$. Hence $(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*)$ satisfies the constraints in (5.23).

Second, note that:

$$\begin{aligned}
&R_2\left(\bar{\mathbf{w}}, \bar{b}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\xi}}^*\right) = R_2(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) + \tfrac{C}{2N} t_i \bar{\xi}_i^2 \\
&\therefore R_2(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) < R_2\left(\bar{\mathbf{w}}, \bar{b}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\xi}}^*\right)
\end{aligned}$$

These two observations contradict the original statement that $\left(\bar{\mathbf{w}}, \bar{b}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\xi}}^*\right)$ with $\bar{\xi}_i < 0$ for some $1 \leq i \leq N$ was a solution of (5.23). Hence, for all solutions $(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*)$ of (5.23), $\boldsymbol{\xi} \geq \mathbf{0}$.

The proof of the non-negativity of $\boldsymbol{\xi}^*$ follows from an analogous argument for the elements of this vector. □

Theorem 5.2. *In the case $q = 2$, any solution $(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*)$ of (5.23) will also be a solution of (5.22), and vice-versa.*

Proof. This follows trivially from theorem 5.1. □

Using (5.23) it is straightforward to construct the dual form of (5.22) when $q = 2$ via the usual method (see appendix A). The dual is:

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} Q_2(\boldsymbol{\alpha}) &= \tfrac{1}{2}\boldsymbol{\alpha}^T \left(\mathbf{K} + \operatorname{diag}\left(\tfrac{N}{C t_i^{(*)}}\right)\right) \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{z} + E|\boldsymbol{\alpha}|^T \boldsymbol{\epsilon}^{(*)} \\
\text{such that: } & \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
& \alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\
& \mathbf{1}^T \boldsymbol{\alpha} = 0
\end{aligned} \tag{5.24}$$

where $\mathbf{K}$ is as before. The trained machine takes the same form as always. Note that there are now fewer constraints on the vector $\boldsymbol{\alpha}$. Furthermore, it can be shown that (see appendix A for details):

$$\begin{aligned}\xi_i &= \max\left\{0, \tfrac{N}{Ct_i}\alpha_i\right\}\\ \xi_i^* &= \max\left\{0, -\tfrac{N}{Ct_i^*}\alpha_i\right\}\end{aligned}$$

It is instructive to re-write (5.24) in terms of $\boldsymbol{\beta}$ and $\boldsymbol{\beta}^*$, as done previously in section 4.4.1. Substituting $\boldsymbol{\alpha} = \boldsymbol{\beta} + \boldsymbol{\beta}^*$ (5.24) becomes:

$$\begin{aligned}\min_{\boldsymbol{\alpha}} Q_2(\boldsymbol{\alpha}) = \tfrac{1}{2}&\begin{bmatrix}\boldsymbol{\beta}\\ \boldsymbol{\beta}^*\end{bmatrix}^T\begin{bmatrix}\mathbf{K}+\mathbf{D} & \mathbf{K}\\ \mathbf{K} & \mathbf{K}+\mathbf{D}^*\end{bmatrix}\begin{bmatrix}\boldsymbol{\beta}\\ \boldsymbol{\beta}^*\end{bmatrix} - \begin{bmatrix}\boldsymbol{\beta}\\ \boldsymbol{\beta}^*\end{bmatrix}^T\begin{bmatrix}\mathbf{z}-E\boldsymbol{\epsilon}\\ \mathbf{z}+E\boldsymbol{\epsilon}^*\end{bmatrix}\\ \text{such that: }\; &\beta_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq}\\ &\beta_i = 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq}\\ &\beta_i^* \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq}\\ &\beta_i^* = 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq}\\ &\mathbf{1}^T\boldsymbol{\alpha} = 0\end{aligned}$$

where:

$$\begin{aligned}D_{i,j} &= \delta_{ij}\tfrac{N}{Ct_i}\\ D_{i,j}^* &= \delta_{ij}\tfrac{N}{Ct_i^*}\end{aligned}$$

From this, it is immediately clear that not only are there no non-global solutions to (5.24), because the Hessian is positive definite (which follows from the fact that it is a positive semidefinite matrix with positive diagonals added), the solution must be *unique*. This makes quadric ϵ-SVMs particularly simple to solve. In particular, selecting $E = 0$ and $\mathbf{t} = \mathbf{t}^* = \mathbf{1}$, (5.24) is precisely Suykens' LS-SVM formulation (although LS-SVM is usually formulated in partially dual form, not fully dual), which is known to be particularly easy to solve [87].

The partially dual form of (5.24) is:

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} \max_{b} Q_{L_2} (\boldsymbol{\alpha}, b) = \frac{1}{2} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \hat{\mathbf{H}} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} + E \left| \begin{matrix} b \\ \boldsymbol{\alpha} \end{matrix} \right|^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{bmatrix} \\
\text{such that:} \quad \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
\alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq}
\end{aligned} \tag{5.25}$$

where:

$$\begin{aligned}
\hat{\mathbf{H}} &= \mathbf{H} + \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{D}^{(*)} \end{bmatrix} \\
D_{i,j}^{(*)} &= \delta_{ij} \frac{N}{C t_i^{(*)}}
\end{aligned}$$

and has the KKT conditions:

$$\begin{aligned}
\alpha_i &\leq \begin{cases} \infty & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \end{cases} \\
\alpha_i &\geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ -\infty & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \end{cases} \\
e_i &\begin{cases} \geq z_i - E\epsilon_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ = z_i - E\epsilon_i & \text{if } \alpha_i > 0 \\ \leq z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \\ = z_i + E\epsilon_i^* & \text{if } \alpha_i < 0 \end{cases} \\
f &= 0
\end{aligned} \tag{5.26}$$

where:

$$\begin{bmatrix} f \\ \mathbf{e} \end{bmatrix} = \hat{\mathbf{H}} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} \tag{5.27}$$

Unlike previously, in this case $e_i = g(\mathbf{x}_i) + D_{i,i}^{(*)} \alpha_i$ (rather than $e_i = g(\mathbf{x}_i)$).

5.5 Tube Shrinking

Thus far in my thesis I have not considered the issue of parameter selection, where by parameter here I mean the constants C and E.[2] Consider E in particular. If the problem comes with some specified target accuracy, this may be used to select E (i.e. in regression, if the regressor is to be accurate to within $\pm 10^{-2}$, then one may reasonably select $E \approx 10^{-2}$). In many cases, however, such a benchmark will not be provided - rather, one may simply want the regressor to be "as accurate as possible".

It will be shown in chapter 6 that optimal selection of this parameter (to make the regressor "as accurate as possible") actually requires knowledge of the type and amount of noise present in the training data. But it is quite possible that this information will not be known, making the selection a matter of trial and error. To overcome this problem, Schölkopf et. al. [70] introduced the ν-SV regression formulation based on the concept of "shrinking the ϵ-tube". This was subsequently extended to cover pattern recognition by Schölkopf and Smola in [75], the extension being called ν-SV classification (ν-SVC) (as opposed to "standard" C-SV classification (C-SVC)).

The basic idea behind the "tube shrinking" concept is to change $E > 0$ from a constant to a variable to be minimised w.r.t. when constructing the SVM. However, simply doing this without changing the primal formulation of the problem will not work. For regression (assuming $\boldsymbol{\epsilon}, \boldsymbol{\epsilon}^* > \mathbf{0}$) this will simply stretch the width of the dead-zone (the "ϵ-tube") by increasing $E > 0$ until $\boldsymbol{\xi} = \boldsymbol{\xi}^* = \mathbf{0}$, resulting in very poor generalisation. For pattern recognition, $E > 0$ will be reduced in a analogous manner, reducing the margin of separation ρ to (effectively) 0 in any non-separable case and thereby negating the efficacy of the max-margin concept central to SVM pattern recognition.

In general, the signs of the elements in $\boldsymbol{\epsilon}$ and $\boldsymbol{\epsilon}^*$ are restricted only by the requirement that $\epsilon_i \geq 0$ for all $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$. For the purposes of tube-shrinking, however,

[2] $\mathbf{t}^{(*)}$ and $\boldsymbol{\epsilon}^{(*)}$ are also, in a sense, parameters. However, whereas E and C have a "global" effect (that is, they effect the training process as a whole), the elements of these vectors have only a "local" effect, in-so-far as they are intended for fine-tuning of the SVM, which is best done by hand using knowledge of individual training points. As a general rule, $\epsilon_i^{(*)} \approx \pm 1$ and $t_i^{(*)} \approx 1$.

it is necessary to make the further requirement that either $\operatorname{sgn}(\boldsymbol{\epsilon}) = \operatorname{sgn}(\boldsymbol{\epsilon}^*) = \mathbf{1}$ or $\operatorname{sgn}(\boldsymbol{\epsilon}) = \operatorname{sgn}(\boldsymbol{\epsilon}^*) = -\mathbf{1}$. This will be assumed true for all tube-shrinking SVMs discussed in this thesis. For convenience, define:

$$\chi = \operatorname{sgn}(\epsilon_i) \, \forall 1 \leq i \leq N$$

Consequently, if $\chi = -1$, $\mathbf{Y}_= = \emptyset$, from which it may be concluded that if $\chi = -1$ then the SVM is essentially a pattern recognition SVM (ignoring, for the moment, the potential influence of $\mathbf{z} \neq \mathbf{0}$). Similarly, if $\chi = 1$, although there may be inequalities in the training set, the problem is clearly one of regression, as the $\epsilon_i^{(*)}$ parameters associated with these inequalities will be associated with an ϵ-tube of positive width.

Consider the pattern recognition case $\chi = -1$ (i.e. $\boldsymbol{\epsilon}, \boldsymbol{\epsilon}^* < \mathbf{0}$). Rather than thinking of this as pattern recognition with a positive margin of separation ρ, this may be thought of as regression *only* in inequalities, where the width of the ϵ-tube is negative with magnitude proportional to E.

Given this, it can be seen that for both pattern recognition and regression naively converting E from a constant to a variable will result in *stretching* of the ϵ-tube. To counteract this, an additional term $\chi C \nu E$ may be added to the primal formulation (5.2) to penalise this stretching (by *shrinking the ϵ-tube.* i.e. applying a counteracting force), thusly:

$$\begin{aligned}
\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, E} \quad & R_{1,1}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, E) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \chi C \nu E + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \\
\text{such that:} \quad & \mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\
& \boldsymbol{\xi} \geq 0 \\
& \boldsymbol{\xi}^* \geq 0 \\
& E > 0
\end{aligned} \tag{5.28}$$

where $\nu \geq 0$ is a constant. Whilst it may seem that (5.28) has simply moved the parameter selection problem from selecting E to selecting ν, this turns out to be somewhat less arduous, as the effect of ν has a more convenient interpretation in the SVM context.

The maximin form of (5.28) is:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E}\ \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*,\rho}\ L_{1,1} = {} & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \chi C\nu E + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=\cup\mathbf{Y}_\geq} \beta_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right)+b\right) - z_i + E\epsilon_i + \xi_i\right) \\
& - \sum_{i:(\mathbf{x}_i,z_i)\in\mathbf{Y}_=\cup\mathbf{Y}_\leq} \beta_i^* \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right)+b\right) - z_i - E\epsilon_i^* - \xi_i^*\right) \\
& -\rho E \\
\text{such that: } & \boldsymbol{\beta} \geq \mathbf{0} \\
& \boldsymbol{\beta}^* \leq \mathbf{0} \\
& \boldsymbol{\gamma} \geq \mathbf{0} \\
& \boldsymbol{\gamma}^* \geq \mathbf{0} \\
& \rho \geq 0 \\
& E \neq 0
\end{aligned} \tag{5.29}$$

Solving this is mostly routine (see appendix A), following the usual template. However ([25]), note that ρ (the lagrange multiplier for the constraint $E \geq 0$) must be 0 unless $E = 0$. However, $E \neq 0$ is a constraint in the maximin formulation (and is retained for this very reason, namely that $E > 0$, not just $E \geq 0$), so $\rho = 0$. Optimising (5.29) with respect to E, it must be true that:

$$\begin{aligned}
& \tfrac{\partial L_{1,1}}{\partial E} = 0 \\
& \Rightarrow \left|\boldsymbol{\epsilon}^{(*)}\right|^T \left|\boldsymbol{\alpha}\right| = C\nu - \chi\rho = C\nu
\end{aligned}$$

Hence the dual form of (5.28) is:

$$\begin{aligned}
& \min_{\boldsymbol{\alpha}} Q_{1,1}\left(\boldsymbol{\alpha}\right) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} \\
& \text{such that: } -\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \\
& \qquad 0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\
& \qquad -\tfrac{C}{N}t_i^* \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \\
& \qquad \mathbf{1}^T\boldsymbol{\alpha} = 0 \\
& \qquad \left|\boldsymbol{\epsilon}^{(*)}\right|^T \left|\boldsymbol{\alpha}\right| = C\nu
\end{aligned} \tag{5.30}$$

The partially dual form (leaving both E and b in the dual) is:

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} \max_{b,E} Q\!\!\!L_{1,1}\left(\boldsymbol{\alpha}, b, E\right) = \tfrac{1}{2} \begin{bmatrix} E \\ b \\ \boldsymbol{\alpha} \end{bmatrix}^T \bar{\mathbf{H}} \begin{bmatrix} E \\ b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} E \\ b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} C\nu \\ 0 \\ \mathbf{z} \end{bmatrix} \\
\text{such that:} \quad & -\tfrac{C}{N} t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
& 0 \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\
& -\tfrac{C}{N} t_i^* \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \\
& E > 0
\end{aligned} \tag{5.31}$$

where:

$$\bar{\mathbf{H}} = \begin{bmatrix} 0 & 0 & \left(\left|\boldsymbol{\epsilon}^{(*)}\right| \operatorname{sgn}\left(\boldsymbol{\alpha}\right)\right)^T \\ 0 & 0 & \mathbf{1}^T \\ \left(\left|\boldsymbol{\epsilon}^{(*)}\right| \operatorname{sgn}\left(\boldsymbol{\alpha}\right)\right) & \mathbf{1} & \mathbf{K} \end{bmatrix}$$

and has the KKT conditions:

$$\begin{aligned}
& \alpha_i \leq \begin{cases} \frac{C}{N} t_i & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \end{cases} \\
& \alpha_i \geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ -\frac{C}{N} t_i^* & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \end{cases} \\
& e_i \begin{cases} \geq z_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ = z_i & \text{if } 0 < \alpha_i < \frac{C}{N} t_i \\ \leq z_i & \text{if } \alpha_i = \frac{C}{N} t_i \\ \leq z_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \\ = z_i & \text{if } -\frac{C}{N} t_i^* < \alpha_i < 0 \\ \geq z_i & \text{if } \alpha_i = -\frac{C}{N} t_i^* \end{cases} \\
& f = 0 \\
& g = C\nu
\end{aligned} \tag{5.32}$$

where:

$$\begin{bmatrix} g \\ f \\ \mathbf{e} \end{bmatrix} = \bar{\mathbf{H}} \begin{bmatrix} E \\ b \\ \boldsymbol{\alpha} \end{bmatrix} \tag{5.33}$$

In this case, $e_i = g(\mathbf{x}_i) + E\left|\epsilon_i^{(*)}\alpha_i\right|$.

Unfortunately, neither the dual nor the partially dual forms of this problem are particularly "nice" to solve. This is (partially) what motivated us to construct the generalised tube shrinking formulation, as will be described in the next subsection. However, before considering this, it is worthwhile to note the salient features of (5.31):

Theorem 5.3. *Assuming $|\boldsymbol{\epsilon}| = |\boldsymbol{\epsilon}^*| = \mathbf{1}$ and $\mathbf{t} = \mathbf{t}^* = \mathbf{1}$:*

1. *ν is an upper bound on the fraction $\frac{N_E}{N}$ of error vectors.*

2. *ν is an lower bound on the fraction $\frac{N_S}{N}$ of support vectors.*

3. *Suppose $\mathbf{Y}_=$, $\mathbf{Y}_\geq$ and $\mathbf{Y}_\leq$ were all generated iid from a distribution $p(\mathbf{x}, z) = p(\mathbf{x})\, p(z|\mathbf{x})$ with $p(z|\mathbf{x})$ continuous. With probability 1, asymptotically, $\frac{N_E}{N} = \frac{N_S}{N} = \nu$.*

Proof. Note that under the assumptions of the theorem the equality constraint $\left|\boldsymbol{\epsilon}^{(*)}\right|^T |\boldsymbol{\alpha}| = C\nu$ in the dual form (5.30) reduces to $\mathbf{1}^T |\boldsymbol{\alpha}| = C\nu$. Furthermore, $0 \leq |\alpha_i| \leq \frac{C}{N}$. Given this:

1. At most $N_E = \nu N$ of all training examples can satisfy $|\alpha_i| = \frac{C}{N}$ (i.e. be error vectors), and so $\frac{N_E}{N} \leq \nu$.

2. At least $N_S = \nu N$ of all training samples must have $|\alpha_i| > 0$ (i.e. be support vectors), and so $\frac{N_S}{N} \geq \nu$.

3. See [70] and [75] for proof of item 3. In essence, the proof relies on the fact the number of points lying exactly on the boundary of the decision region cannot increase linearly with the training set size, and hance the ratio $\frac{N_B}{N} = \frac{N_S - N_E}{N}$ must go to zero at $N \to \infty$.

$\square$

The following theorems from [70] and [75] are special cases of theorem 5.3:

Theorem 5.4 (Proposition 1, [70]). *Assuming* $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}^* = \mathbf{1}$, $\mathbf{t} = \mathbf{t}^* = \mathbf{1}$ *and* $\mathbf{Y}_{\geq} = \mathbf{Y}_{\leq} = \emptyset$ *(pure regression):*

1. *ν is an upper bound on the fraction $\frac{N_E}{N}$ of error vectors.*

2. *ν is an lower bound on the fraction $\frac{N_S}{N}$ of support vectors.*

3. *Suppose $\mathbf{Y}$ was generated iid from a distribution $p(\mathbf{x}, z) = p(\mathbf{x}) p(z|\mathbf{x})$ with $p(z|\mathbf{x})$ continuous. With probability 1, asymptotically, $\frac{N_E}{N} = \frac{N_S}{N} = \nu$.*

Theorem 5.5 (Proposition 12, [75]). *Assuming* $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}^* = -\mathbf{1}$ *and* $\mathbf{t} = \mathbf{t}^* = \mathbf{1}$ *(pattern recognition):*

1. *ν is an upper bound on the fraction $\frac{N_E}{N}$ of error vectors.*

2. *ν is an lower bound on the fraction $\frac{N_S}{N}$ of support vectors.*

3. *Suppose $\mathbf{Y}$ was generated iid from a distribution $p(\mathbf{x}, d) = p(\mathbf{x}) \Pr(d|\mathbf{x})$ such that neither $\Pr(+1|\mathbf{x})$ nor $\Pr(-1|\mathbf{x})$ contains any discrete component. With probability 1, asymptotically, $\frac{N_E}{N} = \frac{N_S}{N} = \nu$.*

In the literature [75], it is usual to call tube-shrinking when applied to regression ν-SV regression (ν-SVR) (and standard regression ϵ-SV regression (ϵ-SVR)). When applied to pattern recognition, tube shrinking is referred to as ν-SV classification (ν-SVC) (and standard pattern recognition C-SV classification (C-SVC)).

5.5.1 Generalised Tube Shrinking

Of course, the concept of tube-shrinking may be extended to the more general case of generalised convex empirical risk SVMs, as considered in section 5.4. As for standard tube shrinking, it is necessary to assume that $\mathrm{sgn}(\boldsymbol{\epsilon}) = \mathrm{sgn}(\boldsymbol{\epsilon}^*) = \chi\mathbf{1}$. Adding the general tube-shrinking term $\chi C \nu c(E)$, where $c \in C^1$ is convex (with $c(0) = 0$), to the primal form of the generalised convex empirical risk SVM (5.19) gives:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E} {}_dR_{d^*,c}\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E\right) &= \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \chi C\nu c\left(E\right) + \tfrac{C}{N}\sum_{i=1}^{N} t_i d\left(\xi_i\right) + \tfrac{C}{N}\sum_{i=1}^{N} t_i^* d^*\left(\xi_i^*\right) \\
\text{such that: } & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\
& \boldsymbol{\xi} \geq 0 \\
& \boldsymbol{\xi}^* \geq 0 \\
& E \geq 0
\end{aligned} \tag{5.34}$$

where $d, d^*, c \in C^1$ are convex functions, $d\left(0\right) = d^*\left(0\right) = c\left(0\right) = 0$) and $\nu > 0$ is a constant as before.

The maximin form of (5.34) is then:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*,\rho} {}_dL_{d^*,c} = & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \chi C\nu c\left(E\right) + \tfrac{C}{N}\sum_{i=1}^{N} t_i d\left(\xi_i\right) + \tfrac{C}{N}\sum_{i=1}^{N} t_i^* d^*\left(\xi_i^*\right) \\
& -\rho E - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i:\left(\mathbf{x}_i,z_i\right)\in\mathbf{Y}_=\cup\mathbf{Y}_\geq} \beta_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i + E\epsilon_i + \xi_i\right) \\
& - \sum_{i:\left(\mathbf{x}_i,z_i\right)\in\mathbf{Y}_=\cup\mathbf{Y}_\leq} \beta_i^* \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i - E\epsilon_i^* - \xi_i^*\right)
\end{aligned} \tag{5.35}$$

$$\begin{aligned}
\text{such that: } & \boldsymbol{\beta} \geq \mathbf{0} \\
& \boldsymbol{\beta}^* \leq \mathbf{0} \\
& \boldsymbol{\gamma} \geq \mathbf{0} \\
& \boldsymbol{\gamma}^* \geq \mathbf{0} \\
& \rho \geq 0
\end{aligned}$$

from which the dual can be derived as (see appendix A for details):

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} {}_dQ_{d^*,c}\left(\boldsymbol{\alpha}\right) &= \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} - \chi C\nu S\left(E\right) - \tfrac{C}{N}\sum_{i=1}^{N} t_i T\left(\xi_i\right) - \tfrac{C}{N}\sum_{i=1}^{N} t_i^* T^*\left(\xi_i^*\right) \\
\text{such that: } & \alpha_i \leq 0 \forall i : \left(\mathbf{x}_i, d_i\right) \in \mathbf{Y}_\leq \\
& \alpha_i \geq 0 \forall i : \left(\mathbf{x}_i, d_i\right) \in \mathbf{Y}_\geq \\
& \mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned} \tag{5.36}$$

where:

$$\xi_i = \begin{cases} \inf\left\{ \xi | \, \frac{C}{N} t_i \frac{\partial d}{\partial \xi}\left(\xi\right) > \left|\alpha_i\right| \right\} & \text{if } \alpha_i > 0 \\ 0 & \text{if } \alpha_i \leq 0 \end{cases}$$
$$\xi_i^* = \begin{cases} 0 & \text{if } \alpha_i \geq 0 \\ \inf\left\{ \xi | \, \frac{C}{N} t_i^* \frac{\partial d^*}{\partial \xi}\left(\xi\right) > \left|\alpha_i\right| \right\} & \text{if } \alpha_i < 0 \end{cases}$$
$$E = \begin{cases} \inf\left\{ E | \, C\nu \frac{\partial c}{\partial E}\left(E\right) > \left|\boldsymbol{\epsilon}^{(*)}\right|^T \left|\boldsymbol{\alpha}\right| \right\} & \text{if } \chi = +1 \\ \inf\left\{ E | \, C\nu \frac{\partial c}{\partial E}\left(E\right) < \left|\boldsymbol{\epsilon}^{(*)}\right|^T \left|\boldsymbol{\alpha}\right| \right\} & \text{if } \chi = -1 \end{cases}$$

and:

$$T\left(\xi\right) = d\left(\xi\right) - \xi \frac{\partial d}{\partial \xi}\left(\xi\right)$$
$$T^*\left(\xi\right) = d^*\left(\xi\right) - \xi \frac{\partial d^*}{\partial \xi}\left(\xi\right)$$
$$S\left(E\right) = c\left(E\right) - E \frac{\partial c}{\partial E}\left(E\right)$$

It will be noted that, like the dual form of generalised empirical risk SVM, this dual form is much more complex than the standard SVM dual. To the best of my knowledge no attempt has been made to solve this general form of tube shrinking - indeed, I am unaware of any papers dealing with generalised tube shrinking (our paper on monomial ν-SV regression [78] deals with the special case $d = d^* = c$, $d\left(E\right) = E^q$, and only considers $d\left(E\right) = E^2$ in detail). However, it should not be too difficult to extend the algorithm presented in [85] to cover this situation. In the present thesis I will be concentrating on the case $d = d^* = c$, $d\left(E\right) = \frac{1}{q}E^q$, $q \in \mathbb{Z}^+$.

5.5.2 Monomial ν-SV Regressors

The following is an extension of monomial the ϵ-SVM method given in section 5.4.2 to encompass tube shrinking. Taking the monomial ϵ-SVM cost $d\left(\xi\right) = \frac{1}{q}\xi^q$, $q \in \mathbb{Z}^+$, the monomial ν-SVM is defined by taking $c\left(E\right) = \frac{1}{n}E^n$, $n \in \mathbb{Z}^+$, as in [78]. Once again, this corresponds to the ML cost function for degree q polynomial noise, but with tube-shrinking applied. Unfortunately, for the pattern classification case $\chi = -1$ certain difficulties arise which make this approach impractical in this case. Hence only regression will be considered in this section, where $\chi = +1$.

The monomial ν-SVR primal cost function is:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E} \; & R_{q,n}\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C\nu}{n}E^n + \tfrac{C}{qN}\sum_{i=1}^{N} t_i\xi_i^q + \tfrac{C}{qN}\sum_{i=1}^{N} t_i^*\xi_i^{*q} \\
\text{such that:} \; & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\
& \boldsymbol{\xi} \geq 0 \\
& \boldsymbol{\xi}^* \geq 0 \\
& E \geq 0
\end{aligned} \tag{5.37}$$

In this thesis I will often restrict myself to the special case $q = n = 2$ (quadric ν-SVR), in which case it turns out that the dual problem is not only mathematically "nice" but retains the property that ν is, in a sense, "easier to select" than the alternative parameter E in the ϵ-SVM case (the cases $q = 1$, $n = 2$ and $q = 2$, $n = 1$ are also mathematically tractable. However, they lose something of the elegance of ν selection by comparison). From this point on, assume that $q = n = 2$.

From theorem 5.1 (the proof of which holds in this case), it is clear that the positivity constraints $\boldsymbol{\xi}, \boldsymbol{\xi}^* \geq \mathbf{0}$ in the primal are superfluous when $q = n = 2$. I will also show shortly that the positivity constraint $E \geq 0$ is superfluous, giving the simplified primal problem:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E} \; & R_{2,2}\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C\nu}{2}E^2 + \tfrac{C}{2N}\sum_{i=1}^{N} t_i\xi_i^2 + \tfrac{C}{2N}\sum_{i=1}^{N} t_i^*\xi_i^{*2} \\
\text{such that:} \; & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq
\end{aligned} \tag{5.38}$$

The following theorems hold if $\chi = +1$:[3]

Theorem 5.6. *For every solution* $(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, E)$ *of (5.38),* $E \geq 0$.

Proof. The proof is directly analogous to that of theorem 5.1. □

Theorem 5.7. *Any solution* $(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, E)$ *of (5.38) will also be a solution of (5.37) when* $q = n = 2$*, and vice-versa.*

[3]It is the failure of the first of these theorems (theorem 5.6) when $\chi = -1$ that makes this working non-applicable to the pattern recognition case.

Proof. This follows trivially from theorems 5.1 and 5.6. □

Using (5.38) it is straightforward to construct the dual form of (5.37) when $q = n = 2$ via the usual method (see appendix A). The dual is:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} Q_{2,2}(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + \tfrac{1}{2}\sum_{i=1}^{N}\alpha_i^2\tfrac{N}{Ct_i^{(*)}} + \tfrac{1}{2C\nu}|\boldsymbol{\alpha}|^T\left(\left|\boldsymbol{\epsilon}^{(*)}\right|\left|\boldsymbol{\epsilon}^{(*)}\right|^T\right)|\boldsymbol{\alpha}| \\
&\text{such that:}\quad \alpha_i \le 0\forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\le} \\
&\qquad\qquad\quad\ \alpha_i \ge 0\forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\ge} \\
&\qquad\qquad\quad\ \mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned}\tag{5.39}$$

where $\mathbf{K}$ is as before. The trained machine takes the same form as always. Note the absence of any additional constraints due to the presence if the tube-shrinking term in the primal. As before:

$$\begin{aligned}
\xi_i &= \max\left\{0, \tfrac{N}{Ct_i}\alpha_i\right\} \\
\xi_i^* &= \max\left\{0, -\tfrac{N}{Ct_i^*}\alpha_i\right\}
\end{aligned}$$

It is instructive to re-write (5.39) in terms of $\boldsymbol{\beta}$ and $\boldsymbol{\beta}^*$, as done previously in section 4.4.1. Substituting $\boldsymbol{\alpha} = \boldsymbol{\beta} + \boldsymbol{\beta}^*$ (5.39) becomes:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} Q_{2,2}(\boldsymbol{\alpha}) = \tfrac{1}{2}\begin{bmatrix}\boldsymbol{\beta}\\ \boldsymbol{\beta}^*\end{bmatrix}^T\begin{bmatrix}\mathbf{K}+\mathbf{N}+\mathbf{D} & \mathbf{K}-\mathbf{N}^* \\ \mathbf{K}-{}^*\mathbf{N} & \mathbf{K}+{}^*\mathbf{N}^*+\mathbf{D}^*\end{bmatrix}\begin{bmatrix}\boldsymbol{\beta}\\ \boldsymbol{\beta}^*\end{bmatrix} - \begin{bmatrix}\boldsymbol{\beta}\\ \boldsymbol{\beta}^*\end{bmatrix}^T\begin{bmatrix}\mathbf{z}\\ \mathbf{z}\end{bmatrix} \\
&\text{such that:}\quad \beta_i \ge 0\forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\ge} \\
&\qquad\qquad\quad\ \beta_i = 0\forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\le} \\
&\qquad\qquad\quad\ \beta_i^* \le 0\forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\le} \\
&\qquad\qquad\quad\ \beta_i^* = 0\forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\ge} \\
&\qquad\qquad\quad\ \mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned}$$

where:

$$\begin{aligned}
D_{i,j} &= \delta_{ij}\tfrac{N}{Ct_i} \\
D_{i,j}^* &= \delta_{ij}\tfrac{N}{C^*t_i^*}
\end{aligned}$$

and:

$$\begin{aligned}
\mathbf{N} &= \tfrac{1}{C\nu}|\boldsymbol{\epsilon}||\boldsymbol{\epsilon}|^T \\
\mathbf{N}^* &= \tfrac{1}{C\nu}|\boldsymbol{\epsilon}||\boldsymbol{\epsilon}^*|^T \\
{}^*\mathbf{N} &= \tfrac{1}{C\nu}|\boldsymbol{\epsilon}^*||\boldsymbol{\epsilon}|^T \\
{}^*\mathbf{N}^* &= \tfrac{1}{C\nu}|\boldsymbol{\epsilon}^*||\boldsymbol{\epsilon}^*|^T
\end{aligned}$$

From this, it is immediately clear that there are no non-global solutions, and furthermore that the solution must be *unique*, a feature in common with quadric ϵ-SVR.

The partially dual form of (5.39) is:

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} \max_{b} QL_{2,2}(\boldsymbol{\alpha}, b) = \tfrac{1}{2}\begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \hat{\mathbf{H}} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} \\
&\text{such that: } \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
&\qquad\qquad\quad \alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq}
\end{aligned} \tag{5.40}$$

where:

$$\begin{aligned}
\hat{\mathbf{H}} &= \mathbf{H} + \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{N}^{(*)} + \mathbf{D}^{(*)} \end{bmatrix} \\
D_{i,j}^{(*)} &= \delta_{ij} \frac{N}{Ct_i^{(*)}} \\
N_{i,j}^{(*)} &= \begin{cases} \frac{1}{C\nu}\left|\epsilon_i^{(*)}\right|\left|\epsilon_j^{(*)}\right| & \text{if } \operatorname{sgn}(\alpha_i\alpha_j) = +1 \\ -\frac{1}{C\nu}\left|\epsilon_i^{(*)}\right|\left|\epsilon_j^{(*)}\right| & \text{if } \operatorname{sgn}(\alpha_i\alpha_j) \neq +1 \end{cases}
\end{aligned}$$

and has the KKT conditions:

$$\begin{aligned}
\alpha_i &\leq \begin{cases} \infty & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_{\geq} \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \end{cases} \\
\alpha_i &\geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ -\infty & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_{\leq} \end{cases} \\
e_i &\begin{cases} \geq z_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_{\geq} \\ = z_i & \text{if } \alpha_i \neq 0 \\ \leq z_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_{\leq} \end{cases} \\
f &= 0
\end{aligned} \tag{5.41}$$

where:

$$\begin{bmatrix} f \\ \mathbf{e} \end{bmatrix} = \hat{\mathbf{H}} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} \tag{5.42}$$

The only major weakness of this formulation is that the solution $\boldsymbol{\alpha}$ may not be as sparse as for standard SV regressors (as will be shown in section 6.4). However, unless there is some given criteria which may be used to determine E (for example, some pre-defined accuracy requirement, or detailed knowledge of any noise in the training data), in my experience this formulation gives excellent results without the difficulties, such as singular hessians, sometimes encountered with standard SVR techniques.

5.6 Combining Methodologies

It is, of course, possible to combine multiple methodologies (for example, automatic biasing and quadric tube shrinking) into a single SVM. In general this is easier if the SVM is symmetric (that is, $\mathbf{t} = \mathbf{t}^*$, $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}^*$, etc.), in which case $\mathbf{t}^{(*)}$, $\boldsymbol{\epsilon}^{(*)}$ are simply constant vectors, and not functions of $\operatorname{sgn}(\boldsymbol{\alpha})$. Making this assumption, the effects of different methodologies are summarised below, starting with either the dual base formulation (5.4):

$$\begin{aligned}
&\min_{\boldsymbol{\alpha}} Q_1(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon} \\
&\text{such that:} \quad -\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
&\qquad\qquad\quad 0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\
&\qquad\qquad\quad -\tfrac{C}{N}t_i^* \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \\
&\qquad\qquad\quad \mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned}$$

or the partially dual base formulation (5.5):

$$\begin{aligned}\min_{\boldsymbol{\alpha}} \max_{b} Q_{L_1}(\boldsymbol{\alpha}, b) = \frac{1}{2} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} + E \left| \begin{matrix} b \\ \boldsymbol{\alpha} \end{matrix} \right|^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon} \end{bmatrix} \\ \text{such that:} \quad -\tfrac{C}{N} t_i^* \le \alpha_i \le 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\le} \\ 0 \le \alpha_i \le \tfrac{C}{N} t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\ge} \\ -\tfrac{C}{N} t_i^* \le \alpha_i \le \tfrac{C}{N} t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=}\end{aligned}$$

where:

$$\mathbf{H} = \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{bmatrix}$$

5.6.1 Fixed Bias

Formulation choice: Use the dual formulation, not the partial dual.

Necessary change: Remove the equality constraint $\mathbf{1}^T \boldsymbol{\alpha} = 0$ from the problem.

Caveat: May not give satisfactory results.

Note: The resulting SVM will be automatically biased if the Mercer kernel chosen satisfies (5.18).

5.6.2 Automatic Bias

Formulation choice: Use the dual formulation, not the partial dual.

Necessary change: Remove the equality constraint $\mathbf{1}^T \boldsymbol{\alpha} = 0$ from the problem.

Necessary change: Add the matrix $\frac{1}{\beta} \mathbf{1}\mathbf{1}^T$ to $\mathbf{K}$.

Caveat: May not give satisfactory results for the regression problem.

5.6.3 Quadric Empirical Risk

Necessary change: Add a positive diagonal perturbation $\mathrm{diag}\left(\frac{N}{Ct_i}\right)$ to $\mathbf{K}$.

Necessary change: Remove upper bounds on the magnitude of $\boldsymbol{\alpha}$.

Caveat: May make solution less sparse.

5.6.4 Linear Tube Shrinking

Necessary change: Remove all $\boldsymbol{\epsilon}$-dependant terms from the cost Q.

Necessary change: Add the new constraint $|\boldsymbol{\epsilon}|^T |\boldsymbol{\alpha}| = C\nu$ to the problem.

Caveat: Cannot be combined with other forms of tube shrinking.

Note: Alternatively, rather than adding a new constraint, the (partial) dual may be extended as shown in (5.31), section 5.5. In the case of fixed or automatic biasing, the row/column relevant to b is removed.

5.6.5 Quadric Tube Shrinking

Necessary change: Remove all $\boldsymbol{\epsilon}$-dependant terms from the cost Q.

Necessary change: Add the matrix $\frac{1}{C\nu}(\mathrm{sgn}\,(\boldsymbol{\alpha})\,\boldsymbol{\epsilon})(\mathrm{sgn}\,(\boldsymbol{\alpha})\,\boldsymbol{\epsilon})^T$ to $\mathbf{K}$.

Caveat: Must remember that the perturbation to $\mathbf{K}$ is a function of $\mathrm{sgn}\,(\boldsymbol{\alpha})$, and therefore not constant during optimisation.

Chapter 6

ASYMPTOTIC ANALYSIS OF SV REGRESSION

If we knew what we were doing, it wouldn't be called research!

– Albert Einstein

IN chapters 4 and 5 I introduced the basic methodologies of SVM pattern recognition and regression. Subsequently, I introduced a common framework for both of these as well as some extensions to the basic models. In particular, in section 5.5 I spent some time discussing the *tube shrinking* re-formulation of the SVR (ν-SVR) whereby the E parameter may be replaced by an alternative parameter, ν, which may be interpreted directly via theorem 5.3.

I consider two issues in this chapter, which is based on our work in [78]. First, I perform some analysis of the SVR when the training set is arbitrarily large (the asymptotic case). I consider the issues of sparsity (what fraction of the training vectors are likely to be support vectors) and also the connection between the E and ν parameters. Based on this work, I consider the issue of optimal parameter selection for the parameters E and ν by applying the concept of *efficiency* introduced in section 3.3.2.

In this section, it will be convenient to differentiate between the type of noise present in the training data (specified by the distribution type) and the amount present (which is assumed to be specified solely by the variance, σ, of the noise distribution). The reason for this is that in many cases the type of noise (e.g. gaussian) may well be known, whereas the amount may not be. It will be shown that optimal selection of E requires knowledge of both type and amount, whereas optimal selection of ν requires only knowledge of the type of noise. I will only be considering noise processes with well defined variance.

6.1 Assumptions

The following assumptions are based on [83]. While the assumptions made in [83] are not strictly met by SV regressors, experimental results demonstrate that they provide both a useful "first guess" of the correct results (e.g. what value ν is optimal) (throughout this chapter, these will be referred to as "the usual assumptions"):

1. The training set is infinitely large ($N \to \infty$).

2. The general SVR model is replaced by an unregularised location parameter estimator ($\lim\limits_{N\to\infty} \frac{C}{N} = \infty$, $K \to 0$).

3. The empirical risk function used has the general form introduced in section 5.4, with the proviso that $d^* = d$ (symmetry) and:

$$d(ab) = d(a)\, d(b) \tag{6.1}$$

4. $\mathbf{t} = \mathbf{t}^* = \mathbf{1}$ and $\boldsymbol{\epsilon} = \boldsymbol{\epsilon}^* = \mathbf{1}$.

It is worthwhile considering the implications of these assumptions. Assumption 1 is just the common assumption that the training set is arbitrarily large - the asymptotic case. Assumption 2 states that the SVR is an unbiased mean estimator, which is strictly false in all practical situations. Hence it is clearly unreasonable to expect an approach making these assumptions to yield exact results. However, given that the training set is, by definition, finite in size, it may be argued that assumption 2 is no more false than the oft used assumption 1.

Indeed, it is not hard to imagine conditions where assumption 2 is a reasonable approximation of reality. For example, if $\frac{C}{N}$ is sufficiently large the regularisation term in the primal will be insignificant, in which case the SVR is unregularised for practical purposes. Similarly, the role of the bias term b is significantly more important than $\boldsymbol{\alpha}$ in-so-far as b incorporates the "mean" of the training data z, whereas $\boldsymbol{\alpha}$ deals with what are essentially higher-order moments. In this light assumption 2 appears somewhat less tenuous.

Assumptions 3 and 4 are more technical in nature. In particular, assumption 3 is necessary to allow the process of E selection to be split into two steps, one using the *type* of training noise present and the other the *amount*, as will be shown shortly. It should be noted that assumption 3 is satisfied by the standard SVM, the LS-SVM and the general monomial cost function from section 5.4.2.[1] In fact, (6.1) is just the Cauchy Functional equation, and it has been known since 1821 [16] that the only continuous functions satisfying this have the form [2]:

$$d(x) = x^a$$

for some $a \in \Re$. So, as multiplicativity is assumed, it is sufficient to consider the case of monomial cost. I will restrict myself to the case $a = q \in \mathbb{Z}^+$.

6.2 Asymptotically Optimal Selection of E

In [83], Smola describes how the parameter E (ϵ in the original notation of [83]) may be selected in an "optimal" fashion using the concept of maximum efficiency introduced in section 3.3.2. In this section I show how the same essential concept may be extended to cover the whole class of monomial SVRs and give a detailed analysis of the polynomial noise case.

From equations (3.4) and (3.6), the efficiency of e of an unbiased mean estimator is for z_i is:

$$e = \frac{Q^2}{IG}$$

where:

$$\begin{aligned} I &= \int_{-\infty}^{\infty} \left(\frac{\partial \ln p(z|\theta)}{\partial \theta} \right)^2 p(z|\theta) dz \\ G &= \int_{-\infty}^{\infty} \left(\frac{\partial d(z,\theta)}{\partial \theta} \right)^2 p(z|\theta) dz \\ Q &= \int_{-\infty}^{\infty} \left(\frac{\partial^2 d(z,\theta)}{\partial \theta^2} \right) p(z|\theta) dz \end{aligned}$$

and $p(z|\theta)$ is the probability density from which training samples z_i are selected. Under the assumption that the SVR is acting simply as a mean estimator, the parameter $\theta = b$ is just the mean of $p(z|\theta)$. σ is defined to be the variance of

[1]Neglecting the superfluous scaling factor of $\frac{1}{q}$, which in any case may be incorporating into C.

$p(z|\theta)$ (i.e. the amount of noise present in the training data).

As discussed previously, the efficiency provides a convenient means of measuring the goodness of a regressor. This is what motivated Smola [83] to select E to maximise the efficiency for the standard ϵ-SVR. In [78] we extended this idea to cover all monomial SVRs, as will be described shortly.

Before proceeding, it is convenient to define the standardised (zero mean, unit variance) version of $p(z|\theta)$, namely:

$$p_{\text{std}}(\tau) = \sigma p(\sigma\tau - \theta|\theta) \tag{6.2}$$

which characterises the type of noise present in the training data; and also:

$$q_{\text{std}}(\tau) = \frac{1}{2}\left(p_{\text{std}}(-\tau) + p_{\text{std}}(\tau)\right) \tag{6.3}$$

which is the symmetrised ($q_{\text{std}}(\tau) = q_{\text{std}}(-\tau)$) and standardised (zero mean, unit variance) version of $p(z|\theta)$.

6.2.1 Training Noise - Splitting Type and Amount

Consider the expression for I:

$$\begin{aligned} I &= \int_{-\infty}^{\infty} \left(\frac{\partial \ln p(z|\theta)}{\partial\theta}\right)^2 p(z|\theta) dz \\ &= \int_{-\infty}^{\infty} \left(\frac{\partial p(z|\theta)}{\partial\theta}\right)^2 \frac{1}{p(z|\theta)} dz \end{aligned}$$

Substituting $\tau = \frac{z+\theta}{\sigma}$:

$$\begin{aligned} I &= \int_{-\infty}^{\infty} \left(\frac{\partial(\sigma p(\sigma\tau-\theta|\theta))}{\partial\theta}\right)^2 \frac{1}{\sigma p(\sigma\tau-\theta|\theta)} d\tau \\ &= \int_{-\infty}^{\infty} \left(\frac{\partial(\sigma p(\sigma\tau-\theta|\theta))}{\partial\tau}\right)^2 \left(\frac{\partial\tau}{\partial\theta}\right)^2 \frac{1}{\sigma p(\sigma\tau-\theta|\theta)} d\tau \\ &= \frac{1}{\sigma^2}\int_{-\infty}^{\infty} \left(\frac{\partial p_{\text{std}}(\tau)}{\partial\tau}\right)^2 \frac{1}{p_{\text{std}}(\tau)} d\tau \\ &= \frac{1}{\sigma^2} I_{\text{std}} \end{aligned}$$

where:

$$I_{\text{std}} = \int_{-\infty}^{\infty} \left(\frac{\partial \ln p_{\text{std}}(\tau)}{\partial\tau}\right)^2 p_{\text{std}}(\tau) d\tau$$

is dependent on the *type* of noise affecting the training data, $p_{\rm std}(\tau)$, but not the *amount*, σ. Using this notation:

$$e = \sigma^2 \frac{Q^2}{I_{\rm std} G}$$

By analogy with the construction of $I_{\rm std}$, define:

$$\begin{aligned}
\tilde{G}_{\rm std} &= \left(\int_{-\infty}^{0} \left(\frac{\partial d\left(\sigma \max\left(-\tau - \frac{E}{\sigma}, 0\right)\right)}{\partial \tau} \right)^2 p_{\rm std}(\tau) d\tau + \int_{0}^{\infty} \left(\frac{\partial d\left(\sigma \max\left(\tau - \frac{E}{\sigma}, 0\right)\right)}{\partial \tau} \right)^2 p_{\rm std}(\tau) d\tau \right) \\
\tilde{Q}_{\rm std} &= \left(\int_{-\infty}^{0} \left(\frac{\partial^2 d\left(\sigma \max\left(-\tau - \frac{E}{\sigma}, 0\right)\right)}{\partial \tau^2} \right) p_{\rm std}(\tau) d\tau + \int_{0}^{\infty} \left(\frac{\partial^2 d\left(\sigma \max\left(\tau - \frac{E}{\sigma}, 0\right)\right)}{\partial \tau^2} \right) p_{\rm std}(\tau) d\tau \right)
\end{aligned}$$

Using assumption (6.1), this becomes:

$$\begin{aligned}
\tilde{G}_{\rm std} &= 2 d^2(\sigma) \int_{0}^{\infty} \left(\frac{\partial d\left(\max\left(\tau - \frac{E}{\sigma}, 0\right)\right)}{\partial \tau} \right)^2 q_{\rm std}(\tau) d\tau \\
\tilde{Q}_{\rm std} &= 2 d(\sigma) \int_{0}^{\infty} \left(\frac{\partial^2 d\left(\max\left(\tau - \frac{E}{\sigma}, 0\right)\right)}{\partial \tau^2} \right) q_{\rm std}(\tau) d\tau
\end{aligned} \tag{6.4}$$

As for $I_{\rm std}$, it is not difficult to show that $G = \frac{1}{\sigma^2} \tilde{G}_{\rm std}$ and $Q = \frac{1}{\sigma^2} \tilde{Q}_{\rm std}$. Hence:

$$e = \frac{\tilde{Q}_{\rm std}^2}{I_{\rm std} \tilde{G}_{\rm std}}$$

Noting that all factors of $d(\sigma)$ cancel out of this expression, (6.4) may be simplified by dropping the $d(\sigma)$ factors to give:

$$\begin{aligned}
G_{\rm std} &= 2 \int_{0}^{\infty} \left(\frac{\partial d\left(\max\left(\tau - \frac{E}{\sigma}, 0\right)\right)}{\partial \tau} \right)^2 q_{\rm std}(\tau) d\tau \\
Q_{\rm std} &= 2 \int_{0}^{\infty} \left(\frac{\partial^2 d\left(\max\left(\tau - \frac{E}{\sigma}, 0\right)\right)}{\partial \tau^2} \right) q_{\rm std}(\tau) d\tau \\
I_{\rm std} &= 2 \int_{0}^{\infty} \left(\frac{\partial \ln q_{\rm std}(\tau)}{\partial \tau} \right)^2 q_{\rm std}(\tau) d\tau
\end{aligned} \tag{6.5}$$

and hence:

$$e = \frac{Q_{\rm std}^2}{I_{\rm std} G_{\rm std}} \tag{6.6}$$

Consider (6.5). It has already been mentioned that $I_{\rm std}$ is independent of the amount of noise, σ. In the definitions of $G_{\rm std}$ and $Q_{\rm std}$ the variance σ appears only in the ratio $\frac{E}{\sigma}$. Furthermore, the only place that E appears is in the same ratio. So, using the shorthand $\omega = \frac{E}{\sigma}$, (6.5) may be re-written thusly:

$$\begin{aligned} G_{\text{std}}(\omega) &= 2\int_0^\infty \left(\frac{\partial d(\max(\tau-\omega,0))}{\partial\tau}\right)^2 q_{\text{std}}(\tau)d\tau \\ Q_{\text{std}}(\omega) &= 2\int_0^\infty \left(\frac{\partial^2 d(\max(\tau-\omega,0))}{\partial\tau^2}\right) q_{\text{std}}(\tau)d\tau \\ I_{\text{std}} &= 2\int_0^\infty \left(\frac{\partial \ln q_{\text{std}}(\tau)}{\partial\tau}\right)^2 q_{\text{std}}(\tau)d\tau \end{aligned} \tag{6.7}$$

or, equivalently:[2]

$$\begin{aligned} G_{\text{std}}(\omega) &= 2\int_0^\infty \left(\frac{\partial d(\max(\tau,0))}{\partial\tau}\right)^2 q_{\text{std}}(\tau+\omega)d\tau \\ Q_{\text{std}}(\omega) &= 2\int_0^\infty \left(\frac{\partial^2 d(\max(\tau,0))}{\partial\tau^2}\right) q_{\text{std}}(\tau+\omega)d\tau \\ I_{\text{std}} &= 2\int_0^\infty \left(\frac{\partial \ln q_{\text{std}}(\tau)}{\partial\tau}\right)^2 q_{\text{std}}(\tau)d\tau \end{aligned} \tag{6.8}$$

and subsequently:

$$e(\omega) = \frac{Q_{\text{std}}^2(\omega)}{I_{\text{std}} G_{\text{std}}(\omega)}$$

is a function of the ratio $\omega = \frac{E}{\sigma}$, but neither E nor σ alone.

Now, the aim of this section is to investigate how E may be selected to maximise the efficiency given the type of noise effecting training data, $p_{\text{std}}(\tau)$, and also the amount, σ. But e is only dependent on E through ω. Hence, using only the type of noise effecting the training data, it should in principle be possible to find the value $\omega = \omega_{\text{opt}}$ to maximise $e(\omega)$. Mathematically:

$$\omega_{\text{opt}} = \arg\min_{\omega} \frac{1}{e(\omega)} \tag{6.9}$$

Given this, E_{opt} (the optimal (for maximum efficiency) value for E) is just:

$$E_{\text{opt}} = \sigma\omega_{\text{opt}} \tag{6.10}$$

[2]Note that $\max(\tau, 0)$ must remain in this expression, as the behaviour of the expressions inside the integrals about $\tau = 0$ will be of some importance.

6.2.2 Deriving the Efficiency

As has previously been mentioned, the only possible cost functions satisfying the multiplicativity assumption made are the monomial cost functions of the general form:

$$d\left(\max\left(\tau,0\right)\right)=\max\left(\tau,0\right)^{a}$$

where $a\in\Re$ is a constant, and it is assumed that $a=q\in\mathbb{Z}^{+}$.

It is easy to see that:[3]

$$\begin{aligned}
d\left(\max\left(\tau,0\right)\right) &= \max\left(\tau,0\right)^{q} \\
\tfrac{\partial d(\max(\tau,0))}{\partial\tau} &= \begin{cases} q\max\left(\tau,0\right)^{q-1} & \text{if } q\neq 1 \\ u\left(\tau\right) & \text{if } q=1 \end{cases} \\
\tfrac{\partial^{2} d(\max(\tau,0))}{\partial\tau^{2}} &= \begin{cases} q\left(q-1\right)\max\left(\tau,0\right)^{q-2} & \text{if } q\neq 1 \\ \delta\left(\tau\right) & \text{if } q=1 \end{cases}
\end{aligned}$$

and hence:

$$\begin{aligned}
G_{\text{std}}\left(\omega\right) &= 2q^{2}\textstyle\int_{0}^{\infty}\tau^{2(q-1)}q_{\text{std}}\left(\tau+\omega\right)d\tau \\
Q_{\text{std}}\left(\omega\right) &= \begin{cases} 2q\left(q-1\right)\int_{0}^{\infty}\tau^{q-2}q_{\text{std}}\left(\tau+\omega\right)d\tau & \text{if } q\neq 1 \\ p_{\text{std}}\left(-\omega\right)+p_{\text{std}}\left(\omega\right) & \text{if } q=1 \end{cases} \\
I_{\text{std}} &= 2\textstyle\int_{0}^{\infty}\left(\frac{\partial\ln q_{\text{std}}(\tau)}{\partial\tau}\right)^{2}q_{\text{std}}\left(\tau\right)d\tau
\end{aligned}$$

which, substituting into (6.6), gives:

$$e_{q}\left(\omega\right)=\frac{1}{I_{\text{std}}}\begin{cases} 2\left(q-1\right)^{2}\frac{\left(\int_{\omega}^{\infty}(\tau-\omega)^{q-2}q_{\text{std}}(\tau)d\tau\right)^{2}}{\int_{\omega}^{\infty}(\tau-\omega)^{2(q-1)}q_{\text{std}}(\tau)d\tau} & \text{if } q\neq 1 \\ \frac{(p_{\text{std}}(-\omega)+p_{\text{std}}(\omega))^{2}}{1-2\int_{0}^{\omega}q_{\text{std}}(\tau)d\tau} & \text{if } q=1 \end{cases} \tag{6.11}$$

where the subscript q has been added for completeness.

[3]Where $u\left(x\right)$ is the Heaviside step function:

$$u\left(x\right)=\begin{cases} 0 & \text{if } x<0 \\ 1 & \text{if } x\geq 0 \end{cases}$$

Efficiency of the Least-Squares SV Regressor

It may be noted that this formula allows us to calculate the efficiency of the LS-SVR, e_{LS}. Specifically, this corresponds to the case $q = 2$, $E = \omega = 0$, i.e.:

$$e_{\mathrm{LS}} = e_2\left(0\right) = \frac{1}{I_{\mathrm{std}} \int_{-\infty}^{\infty} \tau^2 p_{\mathrm{std}}\left(\tau\right) d\tau} \tag{6.12}$$

6.2.3 The Polynomial Noise Case

Suppose that the training noise is polynomial. That is, $p_{\mathrm{std}}\left(\tau\right) = c_p e^{-c'_p |\tau|^p}$, $p \in \mathbb{Z}^+$, where:

$$\begin{aligned} c_p &= \tfrac{1}{2} \tfrac{p}{\Gamma\left(\frac{1}{p}\right)} \sqrt{\tfrac{\Gamma\left(\frac{3}{p}\right)}{\Gamma\left(\frac{1}{p}\right)}} \\ c'_p &= \left(\sqrt{\tfrac{\Gamma\left(\frac{3}{p}\right)}{\Gamma\left(\frac{1}{p}\right)}} \right)^p \end{aligned}$$

where $\Gamma\left(x\right)$ is the (complete) gamma function (C.1). Note that this is just Laplacian noise if $p = 1$, and Gaussian if $p = 2$.

For polynomial noise

$$\begin{aligned} G_{\mathrm{std}}\left(\omega\right) &= q^2 \tfrac{\Gamma^{q-2}\left(\frac{1}{p}\right)}{\Gamma^{q-1}\left(\frac{3}{p}\right)} \daleth_{2q-2}\left(\tfrac{1}{p}, c'_p \omega^p\right) \\ Q_{\mathrm{std}}\left(\omega\right) &= \begin{cases} q\left(q-1\right) \dfrac{\Gamma^{\frac{q-4}{2}}\left(\frac{1}{p}\right)}{\Gamma^{\frac{q-2}{2}}\left(\frac{3}{p}\right)} \daleth_{q-2}\left(\tfrac{1}{p}, c'_p \omega^p\right) & \text{if } q \geq 2 \\ \dfrac{p}{\Gamma\left(\frac{1}{p}\right)} \sqrt{\dfrac{\Gamma\left(\frac{3}{p}\right)}{\Gamma\left(\frac{1}{p}\right)}} e^{-c'_p \omega^p} & \text{if } q = 1 \end{cases} \\ I_{\mathrm{std}} &= \tfrac{p^2 \Gamma\left(\frac{3}{p}\right) \Gamma\left(2-\frac{1}{p}\right)}{\Gamma^2\left(\frac{1}{p}\right)} \end{aligned}$$

where $\daleth_m\left(x, y\right)$ is the backgamma function (C.17), and use has been made of result (C.16). Using results (C.20) and (C.18), it follows that:

$$e_{q,p}\left(\omega\right) = \frac{1}{\Gamma\left(2 - \frac{1}{p}\right)} \begin{cases} \left(\frac{q-1}{p}\right)^2 \frac{\daleth^2_{q-2}\left(\frac{1}{p}, c'_p \omega^p\right)}{\daleth_{2q-2}\left(\frac{1}{p}, c'_p \omega^p\right)} & \text{if } q \geq 2 \\ \frac{e^{-2 c'_p \omega^p}}{\Gamma\left(\frac{1}{p}, c'_p \omega^p\right)} & \text{if } q = 1 \end{cases} \tag{6.13}$$

where the subscript p is included for completeness.

Table 6.1 shows the optimal value for $\frac{E}{\sigma}$ found from the above in the special cases $q = 1, 2$ for polynomial noise of degrees $p = 1$ to 6. Unsurprisingly, the optimal value

Table 6.1: Optimal $\frac{\epsilon}{\sigma}$ and ν for standard and quadric (labelled (S) and (Q), respectively) ϵ-SVR and ν-SVR methods with polynomial additive noise of degree $1 \leq p \leq 6$, and asymptotic support vector ratio's at optimality.

Polynomial degree	1	2	3	4	5	6
Optimal $\frac{\epsilon}{\sigma}$ (S)	0	0.61	1.12	1.36	1.48	1.56
Optimal ν (S)	1	0.54	0.29	0.19	0.14	0.11
$\lim\limits_{N \to \infty} \frac{N_S}{N}$ (S)	1	0.54	0.29	0.19	0.14	0.11
Optimal $\frac{\epsilon}{\sigma}$ (Q)	0	0	0.61	0.97	1.17	1.30
Optimal ν (Q)	∞	∞	0.56	0.18	0.09	0.05
$\lim\limits_{N \to \infty} \frac{N_S}{N}$ (Q)	1	1	0.59	0.39	0.30	0.23

for E for the quadric ϵ-SVR in the presence of Gaussian noise ($p = 2$) is 0, as in this case the quadric ϵ-SVR primal and the maximum-likelihood estimator both take approximately the same form if $E = 0$ and C is large (just as $E_{\text{opt}} = 0$ for standard ϵ-SVR in the presence of Laplacian noise ($p = 1$)). Note also that, for the cases represented in the table, $E_{\text{opt}} > 0$ for all $p > q$ and $E_{\text{opt}} = 0$ for all $p \leq q$. Generally:

Theorem 6.1. *Under the usual assumptions, given a set of training data affected by polynomial noise of degree $p = q$ with non-zero variance, the optimal value (E_{opt} which maximises e) for the parameter E as defined by (6.13) will be zero.*

Proof. From (6.13), if $p = q$ then, using (C.18) and (C.3):

$$\begin{aligned}
e_{p,p}(0) &= \frac{1}{\Gamma\left(2-\frac{1}{p}\right)} \begin{cases} \frac{1}{\Gamma(1)} & \text{if } p = 1 \\ \left(\frac{p-1}{p}\right)^2 \frac{\daleth_{p-2}^2\left(\frac{1}{p},0\right)}{\daleth_{2p-2}\left(\frac{1}{p},0\right)} & \text{if } p \geq 2 \end{cases} \\
&= \begin{cases} 1 & \text{if } p = 1 \\ \left(\frac{p-1}{p}\right)^2 \frac{\Gamma^2\left(1-\frac{1}{p}\right)}{\Gamma^2\left(2-\frac{1}{p}\right)} & \text{if } p \geq 2 \end{cases} \\
&= \begin{cases} 1 & \text{if } p = 1 \\ \left(1-\frac{1}{p}\right)^2 \frac{\Gamma^2\left(1-\frac{1}{p}\right)}{\Gamma^2\left(2-\frac{1}{p}\right)} & \text{if } p \geq 2 \end{cases} \\
&= 1
\end{aligned}$$

But $e_{p,q}(\omega) \leq 1$ by the Cramer-Rao bound (3.3) [67], so optimal efficiency is achieved for $\omega = \frac{E}{\sigma} = 0$. Hence $E_{\text{opt}} = 0$ is a solution. □

Theorem 6.2. *Under the usual assumptions, given a set of training data affected* *by polynomial noise of degree $p > q$ with non-zero variance, the optimal value (E_{opt} which maximises e) for the parameter E as defined by (6.13) will be positive.*

Proof. Note that $e_{p,q}(\omega) \in C^2$ for $\omega \geq 0$. Note also that the range of ω is $\omega \in [0, \infty)$. If $E_{\mathrm{opt}} = 0$, and hence $\omega_{\mathrm{opt}} = 0$, $e_{p,q}(\omega)$ must have a (global) maxima at $\omega = 0$, which implies that the gradient $\frac{de_{p,q}(\omega)}{d\omega}$ of $e_{p,q}(\omega)$ must be non-positive at 0.[4] Given this, to prove the theorem, it is sufficient to prove that the gradient $\frac{de_{p,q}(\omega)}{d\omega}$ of $e_{p,q}(\omega)$ at $\omega = 0$ is positive for all $p > q$ (intuitively it may be seen that if the gradient is positive at zero then increasing ω must result in an increase in $e_{p,q}(\omega)$, and so $\omega = 0$ cannot be a maxima of $e_{p,q}(\omega)$, global or otherwise).

To simplify the mathematics, define $x = {c'_p}^{\frac{1}{p}}\omega$. Hence:

$$\begin{aligned} \frac{de_{p,q}(\omega)}{d\omega} &= {c'_p}^{\frac{1}{p}} \frac{de_{p,q}(x)}{dx} \\ \operatorname{sgn}\left(\frac{de_{p,q}(\omega)}{d\omega}\right) &= \operatorname{sgn}\left(\frac{de_{p,q}(x)}{dx}\right) \end{aligned}$$

Using (C.21), it is straightforward to show that $\frac{de_{p,q}(x)}{dx} = d_{p,q}(x)\, \bar{e}'_{p,q}(x)$, where:

$$\bar{e}'_{p,q}(x) = \begin{cases} e^{-x^p} - 2x^{p-1}\Gamma\left(\frac{1}{p}, x^p\right) & q = 1 \\ \frac{1}{p}\daleth_0\left(\frac{1}{p}, x^p\right) - e^{-x^p}\frac{\daleth_2\left(\frac{1}{p}, x^p\right)}{\daleth_1\left(\frac{1}{p}, x^p\right)} & q = 2 \\ \frac{(q-1)}{(q-2)}\frac{\daleth_{q-2}\left(\frac{1}{p}, x^p\right)}{\daleth_{q-3}\left(\frac{1}{p}, x^p\right)} - \frac{\daleth_{2q-2}\left(\frac{1}{p}, x^p\right)}{\daleth_{2q-3}\left(\frac{1}{p}, x^p\right)} & q \geq 3 \end{cases} \tag{6.14}$$

and:

$$d_{q,p}(x) = \begin{cases} \frac{pe^{-2x^p}}{\Gamma\left(2-\frac{1}{p}\right)\Gamma^2\left(\frac{1}{p}, x^p\right)} & q = 1 \\ \frac{2\daleth_0\left(\frac{1}{p}, x^p\right)\daleth_1\left(\frac{1}{p}, x^p\right)}{p\Gamma\left(2-\frac{1}{p}\right)\daleth_2^2\left(\frac{1}{p}, x^p\right)} & q = 2 \\ \frac{2(q-1)^2(q-2)\daleth_{q-2}\left(\frac{1}{p}, x^p\right)\daleth_{q-3}\left(\frac{1}{p}, x^p\right)\daleth_{2q-3}\left(\frac{1}{p}, x^p\right)}{p^2\Gamma\left(2-\frac{1}{p}\right)\daleth_{2q-2}^2\left(\frac{1}{p}, x^p\right)} & q \geq 3 \end{cases}$$

is a positive function for $\omega \geq 0$, $p, q \in \mathbb{Z}^+$. Hence if $\bar{e}'_{p,q}(0) > 0$ for $p > q \in \mathbb{Z}^+$ then $\left.\frac{de_{p,q}(\omega)}{d\omega}\right|_{\omega=0} > 0$ for $p > q \in \mathbb{Z}^+$, which is sufficient to prove the theorem. Using

[4]As $\omega = 0$ is at the end of the range $\omega \in [0, \infty)$ it follows that if the gradient is negative at $\omega = 0$ then there will be a local maxima at that point.

(C.18), for all $p, q \in \mathbb{Z}^+$:

$$\bar{e}'_{p,q}(0) = \begin{cases} 1 & \text{if } q = 1 \\ \frac{1}{p}\Gamma\left(\frac{1}{p}\right) - \frac{\Gamma\left(\frac{3}{p}\right)}{\Gamma\left(\frac{2}{p}\right)} & \text{if } q = 2 \\ \frac{q-1}{q-2}\frac{\Gamma\left(\frac{q-1}{p}\right)}{\Gamma\left(\frac{q-2}{p}\right)} - \frac{\Gamma\left(\frac{2q-1}{p}\right)}{\Gamma\left(\frac{2q-2}{p}\right)} & \text{if } q \geq 3 \end{cases} \tag{6.15}$$

Which proves the theorem in the case $q = 1$. Consider the case $p > q > 2$. Writing $q = 2 + m$, $p = 2 + m + n$, where $m, n \in \mathbb{Z}^+$, and using (C.3), if $q \geq 3$:

$$\begin{aligned} \bar{e}'_{p,q}(0) &= \frac{\frac{q-1}{p}}{\frac{q-2}{p}}\frac{\Gamma\left(\frac{q-1}{p}\right)}{\Gamma\left(\frac{q-2}{p}\right)} - \frac{\Gamma\left(\frac{2q-1}{p}\right)}{\Gamma\left(\frac{2q-2}{p}\right)} \\ &= \frac{\Gamma\left(\frac{2m+n+3}{m+n+2}\right)}{\Gamma\left(\frac{2m+n+2}{m+n+2}\right)} - \frac{\Gamma\left(\frac{2m+3}{m+n+2}\right)}{\Gamma\left(\frac{2m+2}{m+n+2}\right)} \\ &= \frac{\Gamma(a+c)}{\Gamma(a)} - \frac{\Gamma(b+c)}{\Gamma(b)} \\ &= \frac{\Gamma(a+b+c)}{\Gamma(a)\Gamma(b)}\left(B\left(a+c,b\right) - B\left(a,b+c\right)\right) \end{aligned}$$

where $a = \frac{2m+n+2}{m+n+2} > 0$, $b = \frac{2m+2}{m+n+2} > 0$, $c = \frac{1}{m+n+2} > 0$ and $B(a,b)$ is the beta function (see appendix C). As $a > b$, it follows from theorem C.2 that $B(a+c,b) > B(a,b+c)$, and hence $\bar{e}'(0) > 0$, which proves the theorem for the case $q \geq 3$.

Suppose that m (and subsequently q) is treated as a real number such that $m \geq 0$ (so $q \geq 2$). The inequality $B(a+c,b) > B(a,b+c)$ will still hold, as $a > b$ for all $m \in [0,\infty)$, $n > 0$. Hence $\lim_{q\to 2^+} \bar{e}'_{p,q}(0) > 0$. Furthermore, using theorem C.1:

$$\lim_{q\to 2^+} \frac{q-1}{q-2}\frac{\Gamma\left(\frac{q-1}{p}\right)}{\Gamma\left(\frac{q-2}{p}\right)} - \frac{\Gamma\left(\frac{2q-1}{p}\right)}{\Gamma\left(\frac{2q-2}{p}\right)} = \frac{1}{p}\Gamma\left(\frac{1}{p}\right) - \frac{\Gamma\left(\frac{3}{p}\right)}{\Gamma\left(\frac{2}{p}\right)}$$

which implies that:

$$\bar{e}'_{p,2}(0) = \lim_{q\to 2^+} \bar{e}'_{p,q}(0) > 0$$

and so $\bar{e}'_{p,q}(0) > 0$ for all $p > q \in \mathbb{Z}^+$. Hence $\left.\frac{de_{p,q}(\omega)}{d\omega}\right|_{\omega=0} > 0$ for all $p > q \in \mathbb{Z}^+$, which proves the theorem. □

Conjecture 6.3. *Under the usual assumptions, given a set of training data affected by polynomial noise of degree $p < q$ with non-zero variance, the optimal value (E_{opt} which maximises e) for the parameter E as defined by (6.13) will be zero.*

Partial proof of conjecture 6.3. To prove that $E_{\text{opt}} = 0$ it is necessary (although insufficient) to prove that $e_{p,q}(\omega)$ has a local maxima at $\omega = 0$. By analogy with the proof of theorem 6.2 it is necessary to prove that the gradient $\frac{de_{p,q}(\omega)}{d\omega}$ of $e_{p,q}(\omega)$ at $\omega = 0$ is non-positive for all $p < q$. From the proof of theorem 6.2, $\frac{de_{p,q}(\omega)}{d\omega} = d_{p,q}(\omega)\,\bar{e}'_{p,q}(\omega)$, $d_{p,q}(\omega) > 0$, for all $\omega \geq 0$, $p, q \in \mathbb{Z}^+$, where $\bar{e}'_{p,q}(\omega)$ is given by (6.14). Hence, if $\bar{e}'_{p,q}(0) \leq 0$, $\left.\frac{de_{p,q}(\omega)}{d\omega}\right|_{\omega=0} \leq 0$, and therefore $e_{p,q}(\omega)$ will have a local maxima at $\omega = 0$.

As $1 < p < q$, $q \geq 2$. If $q = 2$, $p = 1$, and hence by (6.15) $\bar{e}'_{1,2}(0) = -1$. Therefore $e_{p,2}(\omega)$ has a local maxima at $\omega = 0$. If $q \geq 3$, writing $q = 2 + m$, $p = 2 + m - n$, where $m \in \mathbb{Z}^+$, $n \in \{1, 2, \ldots, m+1\}$, it can be seen that if $p > q > 2$:

$$\bar{e}'_{p,q}(0) = \frac{\Gamma(a+b+c)}{\Gamma(a)\Gamma(b)} \left(B(a+c, b) - B(a, b+c)\right)$$

where $a = \frac{2m-n+2}{m-n+2} > 0$, $b = \frac{2m+2}{m-n+2} > 0$ and $c = \frac{1}{m-n+2} > 0$. As $a < b$, it follows from theorem C.2 that $B(a+c, b) < B(a, b+c)$, and hence $\bar{e}'(0) < 0$. Therefore, in general, $e_{p,q}(\omega)$ has a local maxima at $\omega = 0$ for all $p > q \in \mathbb{Z}^+$. $\square$

Unfortunately, this is not sufficient to prove the theorem. To do this, it would be necessary to prove that the maxima at $\omega = 0$ is *global* (or unique, which would imply globality), and hence $\omega_{\text{opt}} = E_{\text{opt}} = 0$. Alternatively, proving that $e'_{p,q}(\omega) \leq 0$ for all $\omega \geq 0$ would demonstrate that the maxima at $\omega = 0$ is global and thereby prove the conjecture. Unfortunately this part of the proof remains incomplete. Empirically (using a search approach on the PC) it has been observed [78] that conjecture 6.3 appears to hold for $1 \leq q \leq 1000$.

Figures 6.1 to 6.6 show the inverse efficiency $\frac{1}{e_{q,p}(\omega)}$ as a function of $\omega = \frac{E}{\sigma}$ for same. Note that with the exception of Laplacian noise ($p = 1$) the optimal theoretical efficiency of the quadratic ϵ-SVR exceeds the optimal theoretical efficiency of the standard ϵ-SVR. Also note that the efficiency of the monomial ϵ-SVR method exceeds that of the LS-SVR for all $p > 2$.

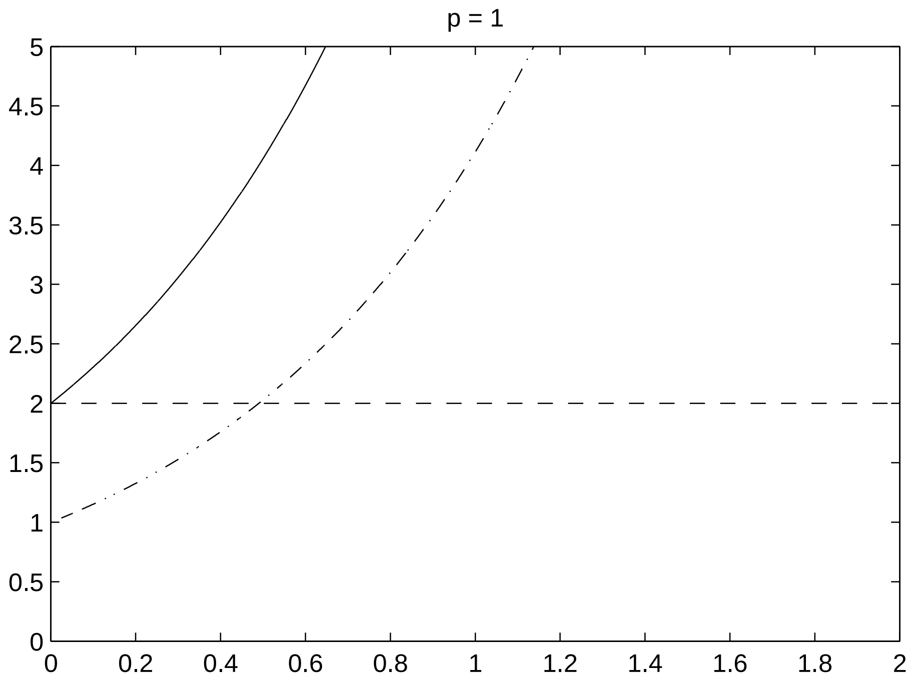


Figure 6.1: Comparative inverse asymptotic efficiency versus $\frac{E}{\sigma}$ of standard ϵ-SVR, quadric ϵ-SVR and LS-SVR for polynomial noise of degree 1 (Laplacian noise). The solid line represents the efficiency of quadric ϵ-SVR, the dotted line the efficiency of standard ϵ-SVR, and the dashed line the efficiency of the LS-SVR.

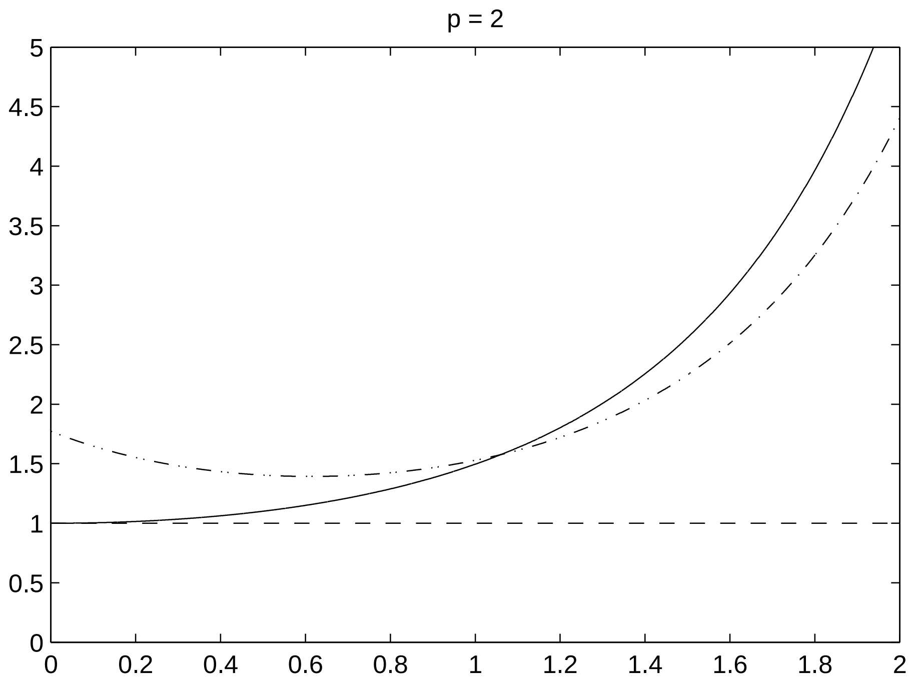


Figure 6.2: Comparative inverse asymptotic efficiency versus $\frac{E}{\sigma}$ of standard ϵ-SVR, quadric ϵ-SVR and LS-SVR for polynomial noise of degree 2 (Gaussian noise). The solid line represents the efficiency of quadric ϵ-SVR, the dotted line the efficiency of standard ϵ-SVR, and the dashed line the efficiency of the LS-SVR.

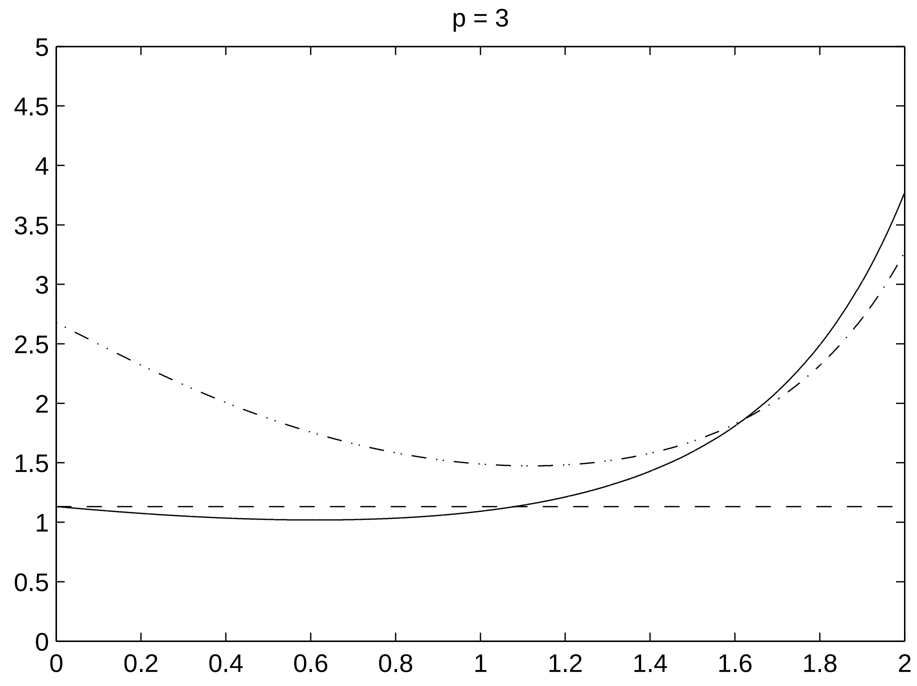


Figure 6.3: Comparative inverse asymptotic efficiency versus $\frac{E}{\sigma}$ of standard ϵ-SVR, quadric ϵ-SVR and LS-SVR for polynomial noise of degree 3. The solid line represents the efficiency of quadric ϵ-SVR, the dotted line the efficiency of standard ϵ-SVR, and the dashed line the efficiency of the LS-SVR.

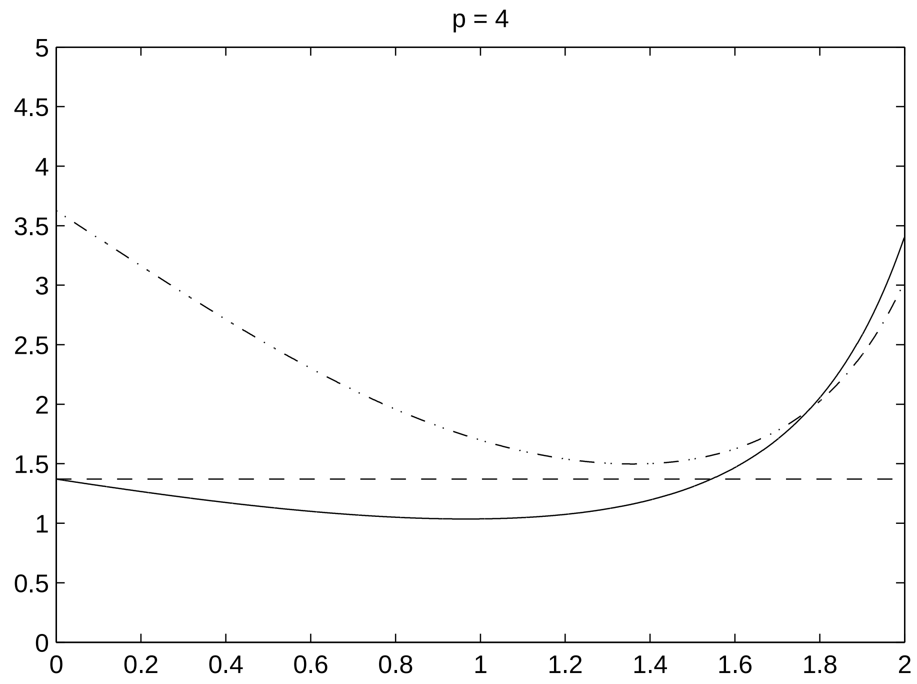


Figure 6.4: Comparative inverse asymptotic efficiency versus $\frac{E}{\sigma}$ of standard ϵ-SVR, quadric ϵ-SVR and LS-SVR for polynomial noise of degree 4. The solid line represents the efficiency of quadric ϵ-SVR, the dotted line the efficiency of standard ϵ-SVR, and the dashed line the efficiency of the LS-SVR.

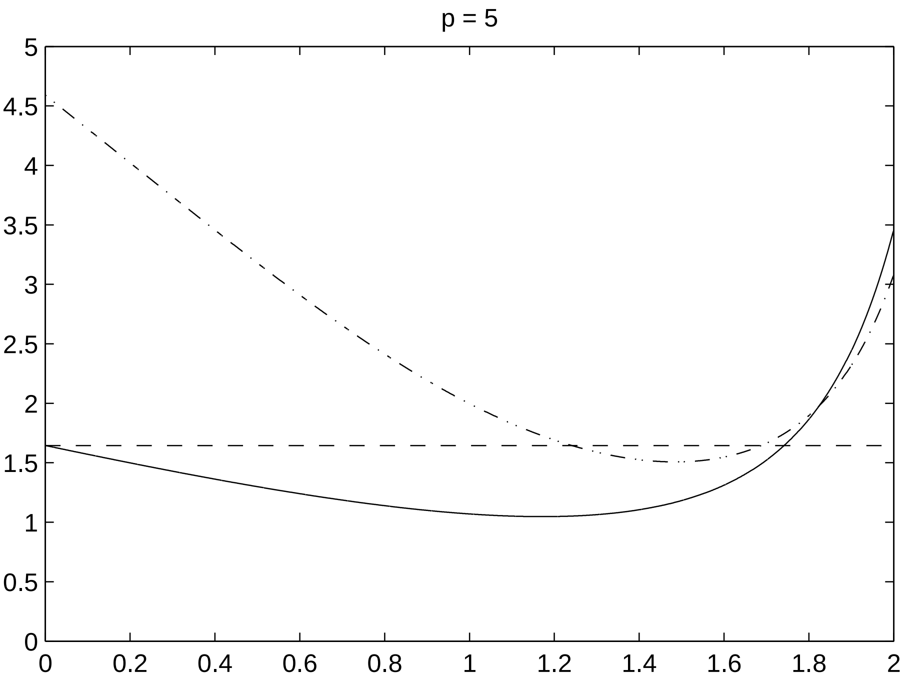


Figure 6.5: Comparative inverse asymptotic efficiency versus $\frac{E}{\sigma}$ of standard ϵ-SVR, quadric ϵ-SVR and LS-SVR for polynomial noise of degree 5. The solid line represents the efficiency of quadric ϵ-SVR, the dotted line the efficiency of standard ϵ-SVR, and the dashed line the efficiency of the LS-SVR.

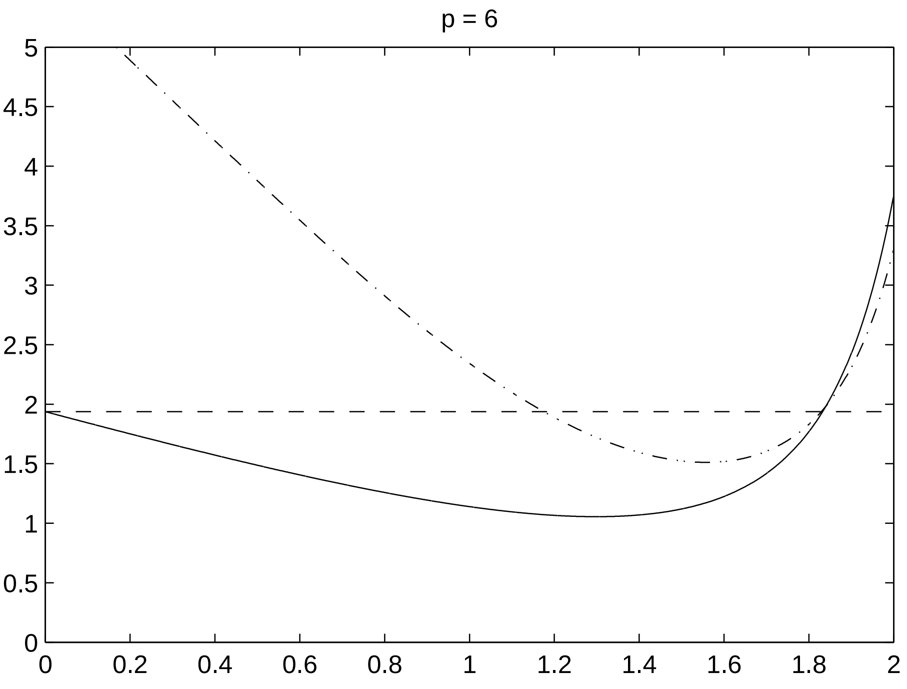


Figure 6.6: Comparative inverse asymptotic efficiency versus $\frac{E}{\sigma}$ of standard ϵ-SVR, quadric ϵ-SVR and LS-SVR for polynomial noise of degree 6. The solid line represents the efficiency of quadric ϵ-SVR, the dotted line the efficiency of standard ϵ-SVR, and the dashed line the efficiency of the LS-SVR.

Efficiency of the Least-Squares SV Regressor - Polynomial Training Noise

If the training data is affected by polynomial noise:

$$e_{\mathrm{LS}} = \frac{\Gamma^2\left(\frac{1}{p}\right)}{p^2\Gamma\left(2-\frac{1}{p}\right)\Gamma\left(\frac{3}{p}\right)} \tag{6.16}$$

Note that if $p = 2$ then $e_{\mathrm{LS}} = 1$ (i.e. the least-squares support vector regressor is optimal in the presence of gaussian training noise, as expected).

6.3 Asymptotically Optimal Selection of ν

6.3.1 The Connection Between ν and E

It has been noted in section 6.2 that the efficiency, e, of an ϵ-SVR is, under the usual assumptions, a function of $\omega = \frac{E}{\sigma}$. The implication of this was that E may be selected using a two stage process. First, ω_{opt} is selected to maximise the efficiency based on the noise type (independent of the noise variance σ), according to (6.9). Then E_{opt} is given by:

$$E_{\mathrm{opt}} = \sigma\omega_{\mathrm{opt}}$$

I now show that e may also be expressed (somewhat indirectly) as a function of ν (making the usual assumptions), independent of σ if $q = n$. The advantage of this form is that, unlike E_{opt}, calculation of ν_{opt} does not require knowledge of σ.

Consider the regularised cost function (5.37) in its primal form, making the usual assumptions:[5]

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E} \; & R_{q,n}\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E\right) = \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C\nu}{n}E^n + \tfrac{C}{qN}\sum_{i=1}^{N}\xi_i^q + \tfrac{C}{qN}\sum_{i=1}^{N}\xi_i^{*q} \\
\text{such that:} \; & \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E - \xi_i \forall \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\
& \mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E + \xi_i^* \forall \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\
& \boldsymbol{\xi} \geq 0 \\
& \boldsymbol{\xi}^* \geq 0 \\
& E \geq 0
\end{aligned}$$

where $n \geq 1$ and $q \geq 1$ are integers. First, note that:

$$\frac{\partial R_{q,n}}{\partial E} = C\left(\nu E^{n-1} + \frac{1}{N}\sum_{i=1}^{N}\left(\xi_i^{q-1}\frac{\partial \xi_i}{\partial E} + \xi_i^{*q-1}\frac{\partial \xi_i^*}{\partial E}\right)\right)$$

where, using the optimality result:

$$\begin{aligned}
\xi_i &= \max\left(0, -g\left(\mathbf{x}\right) + z_i - E\right) \\
\xi_i^* &= \max\left(0, g\left(\mathbf{x}\right) - z_i - E\right)
\end{aligned}$$

it follows that:[6]

$$\begin{aligned}
\tfrac{\partial \xi_i}{\partial E} &= \begin{cases} 0 & \text{if } g\left(\mathbf{x}_i\right) > z_i - E \text{ or } \left(g\left(\mathbf{x}_i\right) = z_i - E, \delta E > 0\right) \\ -1 & \text{otherwise} \end{cases} \\
\tfrac{\partial \xi_i^*}{\partial E} &= \begin{cases} 0 & \text{if } g\left(\mathbf{x}_i\right) < z_i + E \text{ or } \left(g\left(\mathbf{x}_i\right) = z_i + E, \delta E > 0\right) \\ -1 & \text{otherwise} \end{cases}
\end{aligned}$$

[5]It is important to note that the additional term $\frac{C\nu}{n}E^n$ will not vanish in the limit $C \to \infty$. However, this term will *not* make the resulting SVM biased in the limit, as this term will only affect the form of the empirical risk term in the cost function (via the width of the insensitive region), not the form of the resulting trained machine.

[6]I have taken some liberties here when calculating this derivative, which is not actually well defined. However, the rate of change as E is either increased or decreased is well defined, which is what is indicated here.

and hence:

$$\begin{aligned}\frac{\partial R_{q,n}}{\partial E} &= C\begin{cases}\nu E^{n-1} - \frac{N_S}{N} & q=1, \delta E<0 \\ \nu E^{n-1} - \frac{N_E}{N} & q=1, \delta E>0 \\ \nu E^{n-1} - \frac{1}{N}\sum\limits_{i=1}^{N}\left(\xi_i^{q-1}+\xi_i^{*q-1}\right) & \text{otherwise}\end{cases} \\ &= C\begin{cases}\nu E^{n-1} - \frac{N_S}{N} & q=1, \delta E<0 \\ \nu E^{n-1} - \frac{N_E}{N} & q=1, \delta E>0 \\ \nu E^{n-1} - \frac{1}{N}\sum\limits_{i=1}^{N}\max\left(\left|z_i - g\left(\mathbf{x}_i\right)\right| - E, 0\right)^{q-1} & \text{otherwise}\end{cases}\end{aligned} \tag{6.17}$$

The process of selection E_{opt} has already been given previously. In the tube shrinking case, however, E will be selected automatically based on ν. The optimality condition for E is that $\frac{\partial R_{q,n}}{\partial E} = 0$, where $\frac{\partial R_{q,n}}{\partial E}$ is given by (6.17). So, the optimal value ν_{opt} for ν to achieve maximum efficiency is the value which, in the solution to the primal, gives $E = E_{\text{opt}}$; and the relationship between ν_{opt} and E_{opt} is given by (6.17). If $q \geq 2$ then the functional relationship between E and ν is:

$$\begin{aligned}\nu &= E^{1-n}\frac{1}{N}\sum_{i=1}^{N}\left(\xi_i^{q-1}+\xi_i^{*q-1}\right) \\ &= E^{1-n}\frac{1}{N}\sum_{i=1}^{N}\max\left(\left|z_i - g\left(\mathbf{x}_i\right)\right| - E, 0\right)^{q-1}\end{aligned}$$

Applying the usual assumptions ($g\left(\mathbf{x}\right) = b$, $N \to \infty$), it follows that in all cases:[7]

$$\nu = \sigma^{q-n}\left[2\omega^{1-n}\int_{\omega}^{\infty}\left(\tau-\omega\right)^{q-1} q_{\text{std}}\left(\tau\right) d\tau\right]$$

[7]If $q = 1$ then in the limit $N \to \infty$ (6.17) becomes:

$$\lim_{N\to\infty}\frac{\partial R_{1,r}}{\partial \epsilon} = C\begin{cases}\nu E^{n-1} - \lim\limits_{N\to\infty}\frac{N_S}{N} & \text{if } \delta\epsilon < 0 \\ \nu E^{n-1} - \lim\limits_{N\to\infty}\frac{N_E}{N} & \text{if } \delta\epsilon > 0\end{cases}$$

which is well defined, as $\lim\limits_{N\to\infty}\frac{N_S}{N} = \lim\limits_{N\to\infty}\frac{N_E}{N}$. For optimality, $\frac{\partial R_{q,r}}{\partial\epsilon} = 0$, and so using theorem 6.4 it follows that, if $q = 1$:

$$\nu = \sigma^{1-n}\left[2\omega^{1-n}\int_{\omega}^{\infty} q_{\text{std}}\left(\tau\right) d\tau\right]$$

where, as usual, $\omega = \frac{E}{\sigma}$. If $q = n$ then:

$$\nu = t_q\left(\omega\right) = 2\omega^{1-q} \int_{\omega}^{\infty} \left(\tau - \omega\right)^{q-1} q_{\mathrm{std}}\left(\tau\right) d\tau \tag{6.18}$$

which is also 1-1 (invertible) if $q_{\mathrm{std}}\left(\tau\right)$ is positive for all $\tau \geq 0$.

This implies that, if $q = n$, ν may be selected to achieve a particular efficiency e_{select} (assuming that efficiency is achievable) by finding ω required to achieve this and then substituting this into the relevant expression for ν. At no point is it necessary to use σ, so only the noise type is required. If $q = n$:

$$\nu = t_q\left(e_q^{\leftarrow}\left(E\right)\right) \tag{6.19}$$

Note that if $q = n = 1$, (6.18) reduces to Smola's original result [70]:

$$\nu = 1 - \int_{-\omega}^{\omega} p_{\mathrm{std}}\left(\tau\right) d\tau$$

Hence, for a standard ν-SVR, if $\nu = 1$ then, in the limit $N \to \infty$, $\omega = E = 0$. Generally, if $q = n$ then in order to achieve maximum efficiency one should choose:

$$\nu_{\mathrm{opt}} = t_q\left(\arg\min_{\omega} \frac{1}{e_q\left(\omega\right)}\right) \tag{6.20}$$

In the special case of polynomial noise:

$$\nu = \sigma^{q-n} \left[\frac{\Gamma^{\frac{q-3}{2}}\left(\frac{1}{p}\right)}{\Gamma^{\frac{q-1}{2}}\left(\frac{3}{p}\right)\omega^{n-1}} \urcorner_{q-1}\left(\frac{1}{p}, c'_p \omega^p\right) \right]$$

and, if $q = n$:

$$\nu = \frac{\Gamma^{\frac{q-3}{2}}\left(\frac{1}{p}\right)}{\Gamma^{\frac{q-1}{2}}\left(\frac{3}{p}\right)\omega^{q-1}} \urcorner_{q-1}\left(\frac{1}{p}, c'_p \omega^p\right) \tag{6.21}$$

Table 6.1 shows the optimal value for ν found from the above in the special cases $q = 1, 2$ for polynomial noise of degrees $p = 1$ to 6. Figures 6.7 to 6.12 show the inverse asymptotic efficiency versus ν for both standard and quadratic ν-SVRs, as well as LS-SVR. The important thing to note from these graphs is that, although

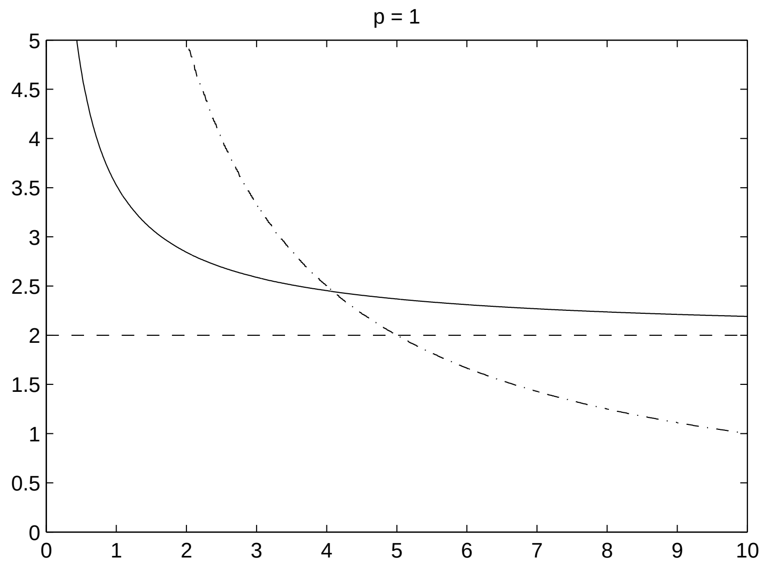


Figure 6.7: Comparative inverse asymptotic efficiency versus ν of standard ν-SVR, quadric ν-SVR and LS-SVR for polynomial noise of degree 1 (Laplacian noise). The solid line represents the efficiency of quadric ν-SVR, the dotted line the efficiency of standard ν-SVR and the dashed line the efficiency of the LS-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

the range of ν is much larger for quadratic ν-SVR methods than for standard ν-SVR methods, the efficiency itself quickly flattens out. In particular, although the theoretically optimal value for Gaussian noise is $\nu \to \infty$, it can be seen that even when $\nu = 1$ the efficiency is very close to it's maximum, $e = 1$. Also note the comparative flatness of the efficiency curves for quadratic ν-SVR.

6.4 Sparsity Analysis

As discussed previously, part of the motivation for developing the framework for monomial ϵ-SVR is Suykens' LS-SVR technique. Indeed, if C is sufficiently large and the noise affecting measurements Gaussian, LS-SVR corresponds approximately to the ML estimator for the parameters $\mathbf{w}$ and b. However, a downside of the LS-SVR approach is the lack of sparsity in the solution [87], where by sparsity I am referring here to the number of non-zero elements in the vector $\boldsymbol{\alpha}$ or, equivalently as $N \to \infty$ (see theorem 5.3), the fraction of training vectors that are also support

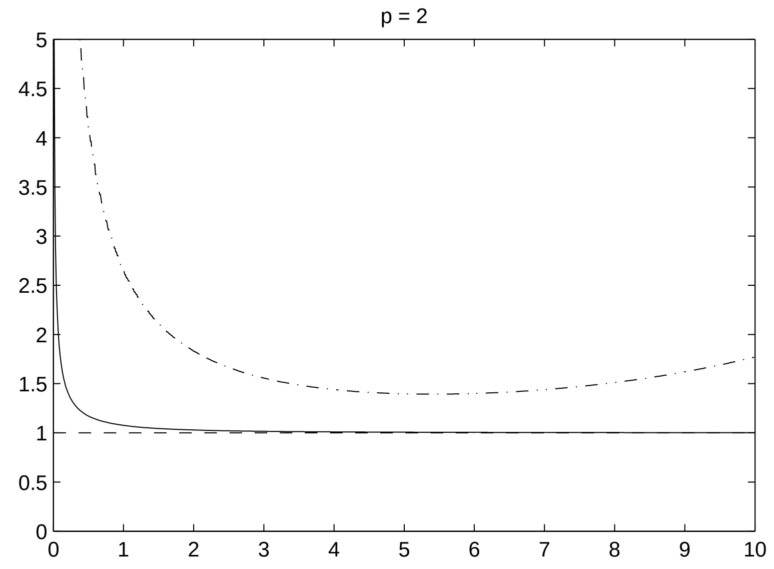


Figure 6.8: Comparative inverse asymptotic efficiency versus ν of standard ν-SVR, quadric ν-SVR and LS-SVR for polynomial noise of degree 2 (Gaussian noise). The solid line represents the efficiency of quadric ν-SVR, the dotted line the efficiency of standard ν-SVR and the dashed line the efficiency of the LS-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

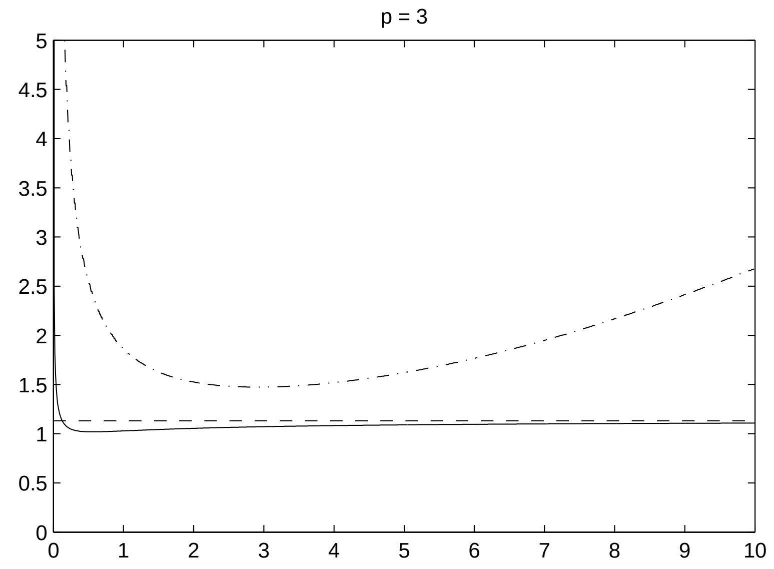


Figure 6.9: Comparative inverse asymptotic efficiency versus ν of standard ν-SVR, quadric ν-SVR and LS-SVR for polynomial noise of degree 3. The solid line represents the efficiency of quadric ν-SVR, the dotted line the efficiency of standard ν-SVR and the dashed line the efficiency of the LS-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

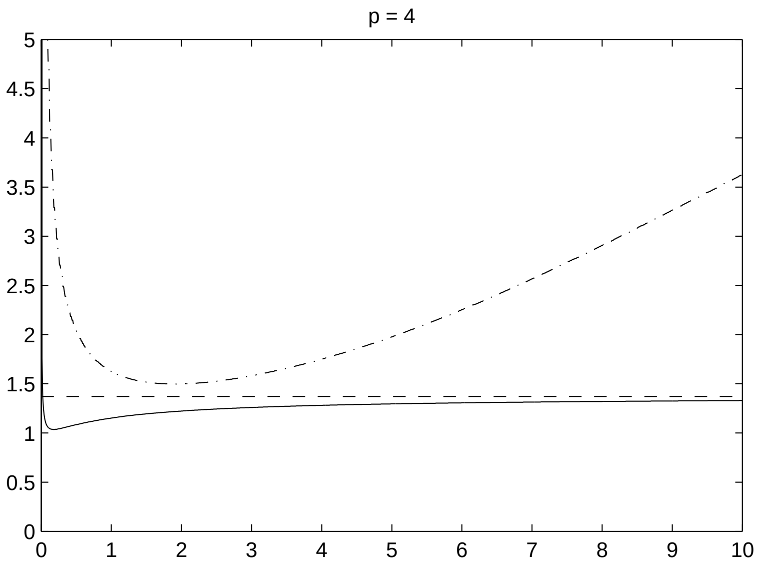


Figure 6.10: Comparative inverse asymptotic efficiency versus ν of standard ν-SVR, quadric ν-SVR and LS-SVR for polynomial noise of degree 4. The solid line represents the efficiency of quadric ν-SVR, the dotted line the efficiency of standard ν-SVR and the dashed line the efficiency of the LS-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

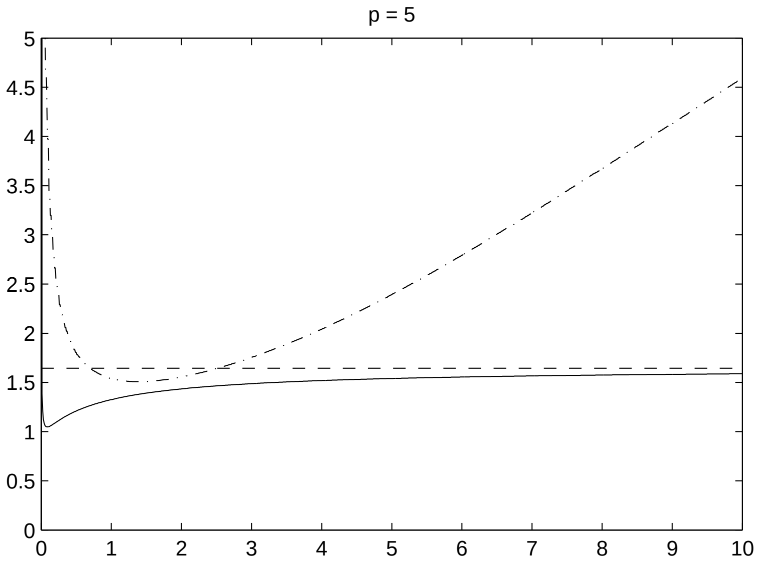


Figure 6.11: Comparative inverse asymptotic efficiency versus ν of standard ν-SVR, quadric ν-SVR and LS-SVR for polynomial noise of degree 5. The solid line represents the efficiency of quadric ν-SVR, the dotted line the efficiency of standard ν-SVR and the dashed line the efficiency of the LS-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

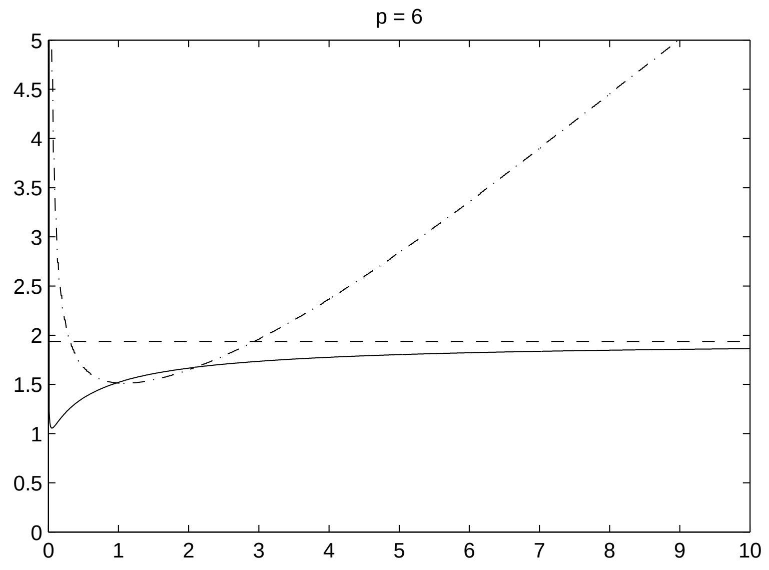


Figure 6.12: Comparative inverse asymptotic efficiency versus ν of standard ν-SVR, quadric ν-SVR and LS-SVR for polynomial noise of degree 6. The solid line represents the efficiency of quadric ν-SVR, the dotted line the efficiency of standard ν-SVR and the dashed line the efficiency of the LS-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

vectors. In the LS-SVR case, all training vectors are support vectors, i.e. $N_S = N$. While the sparsity problems of the LS-SVR may be overcome to a degree using the weighted LS-SVR approach [88], this approach is somewhat heuristic in nature, requiring some external arbiter (either human or machine) to decide when to cease pruning the dataset. However, by maintaining the ϵ-insensitive component of the cost function, the need for such heuristics (at least at this level of abstraction) is removed.[8]

So long as $E > 0$, one would expect that the solution $\boldsymbol{\alpha}$ to the monomial ϵ-SVR dual problem will contain some non-zero fraction of non-support vectors. Under the usual assumptions, in [78] we showed that:

Theorem 6.4. *The fraction of support vectors found by the ϵ-SVR under the usual*

[8]Of course, some heuristic input will still be required for each approach, either for E selection or when deciding how much compromise is acceptable during pruning. The advantage of the former is that there exist alternative criteria which may be used to select E (i.e. optimal performance for a given noise model), with sparsity properties being just a useful side affect of this choice. If the dataset is very large, however, sparsity may be of primary importance, in which case neither approach will have a clear advantage.

assumptions will, asymptotically, be:

$$\lim_{N \to \infty} \frac{N_S}{N} = 2 \int_{\omega}^{\infty} q_{\rm std}\left(\tau\right) d\tau$$

where $\omega = \frac{E}{\sigma}$.

Proof. Theorem 5.3 says that:

$$\lim_{N \to \infty} \frac{N_S}{N} = \lim_{N \to \infty} \frac{N_E}{N}$$

Furthermore, since error vectors are those for which $\left| g\left(\mathbf{x}_i\right) - z_i \right| > E$ (i.e. those lying outside the ϵ-tube):

$$\lim_{N \to \infty} \frac{N_S}{N} = \Pr\left(\left| g\left(\mathbf{x}\right) - \hat{g}\left(\mathbf{x}\right) \right| > E \right)$$

Therefore:

$$\begin{aligned} \lim_{N \to \infty} \tfrac{N_S}{N} &= \int\limits_{z \in \Re \backslash [\theta - E, \theta + E]} p\left(z \right| \theta) \, dz \\ &= \int\limits_{z \in \Re \backslash [-E, E]} p\left(z + \theta \right| \theta) \, dz \\ &= \int\limits_{z \in \Re \backslash [-\omega, \omega]} \sigma p\left(\sigma\tau + \theta \right| \theta) \, d\tau \\ &= \int\limits_{z \in \Re \backslash [-\omega, \omega]} p_{\rm std}\left(\tau\right) d\tau \\ &= 2 \textstyle\int_{\omega}^{\infty} q_{\rm std}\left(\tau\right) d\tau \end{aligned}$$

This proves the theorem. □

Note that this implies:

$$\lim_{\substack{N \to \infty \\ E \to 0}} \frac{N_S}{N} = 1$$

as expected for LS-SVR. It also implies that any decrease in E is likely to lead to a decrease in the sparsity of the solution. So, in general, if the training set is large then E should be chosen to be as large as possible while still maintaining acceptable performance to maximise the sparsity of the solution.

6.4.1 Sparsity of the Monomial ν-SVR

Theorem 6.4 showed that, under the usual assumptions:

$$s(\omega) = \lim_{N \to \infty} \frac{N_S}{N} = 2 \int_{\omega}^{\infty} q_{\mathrm{std}}(\tau)\, d\tau \tag{6.22}$$

which is a 1-1 (invertible) function if the distribution $q_{\mathrm{std}}(\tau)$ is everywhere positive. It was also shown in the previous section that if $q = n$ then from (6.18), $\nu = t_q(\omega)$, where $t_q(\omega)$ is invertible if $q_{\mathrm{std}}(\tau)$ is positive everywhere.

These two functions demonstrate that in general there exists a direct relation between the asymptotic value $s(\omega)$ of the fraction of support vectors in the training set (and hence the sparsity of the solution) and the parameter ν. Specifically:

$$\lim_{N \to \infty} \frac{N_S}{N} = s\left(t_q^{\leftarrow}(\nu)\right)$$

In general, the exact nature of this relation will be dependent of the form of the noise process affecting the training data, and may not be expressible in closed form. In the special case $q = n = 1$ (i.e. standard ν-SVR), however, it is clear that:

$$\nu = \lim_{N \to \infty} \frac{N_S}{N}$$

which coincides with theorem 5.3.

6.4.2 Sparsity in Polynomial Noise

In the case of polynomial noise then, from (6.22):

$$s(\omega) = \frac{\Gamma\left(\frac{1}{p}, c'_p \omega^p\right)}{\Gamma\left(\frac{1}{p}\right)}$$

Assuming $q = n$, it follows from (6.21) that:

$$\nu(\omega) = \frac{\Gamma^{\frac{q-3}{2}}\left(\frac{1}{p}\right)}{\Gamma^{\frac{q-1}{2}}\left(\frac{3}{p}\right)\omega^{q-1}} \daleth_{q-1}\left(\frac{1}{p}, c'_p \omega^p\right)$$

Or, in the special cases $q = 1$ and $q = 2$:

$$\nu\left(\omega\right) = \begin{cases} \frac{\Gamma\left(\frac{1}{p}, c'_p \omega^p\right)}{\Gamma\left(\frac{1}{p}\right)} & \text{if } q = 1 \\ \frac{\daleth_1\left(\frac{1}{p}, c'_p \omega^p\right)}{\omega\sqrt{\Gamma\left(\frac{1}{p}\right)\Gamma\left(\frac{3}{p}\right)}} & \text{if } q = 2 \end{cases}$$

In the case $q = 1$, as expected, this implies:

$$\lim_{N \to \infty} \frac{N_S}{N} = \nu$$

The case $q = 2$ is slightly more complex, and best illustrated graphically as in figures 6.13 to 6.18, which show the predictions for the fraction of support vectors found as a function of ν for both standard and quadratic ν-SVR methods for polynomial noise of degree $1 \leq p \leq 6$. Note that the general shape of the curves is essentially identical in all cases. Generally, the fraction of support vectors found by the quadratic ν-SVR will increase quickly while ν is small and then level out, approaching 1 as $\nu \to \infty$ (as expected, given that the LS case corresponds with $\nu \to \infty$ and treats all vectors as support vectors).

Table 6.1 gives the expected asymptotic ratio of support vectors to training vectors when ν is optimally selected for the usual degrees of polynomial noise. On average, the results given in the table imply that quadratic ν-SVRs may require approximately twice as many support vectors as standard ν-SVRs to achieve optimal accuracy on the same dataset. This may be understood by realising that the act of extracting support vectors is essentially a form of lossy compression. The modified ν-SVR is (theoretically) able to achieve more accurate results than standard ν-SVR because it can handle more information (by using less compression or, equivalently, finding more support vectors) before over-fitting (and subsequent degradation in performance) begins.

6.5 Experimental Results

Due to the restrictive assumptions made when deriving the results for theoretical efficiency given in the preceding sections (which will, in the strictest sense, never

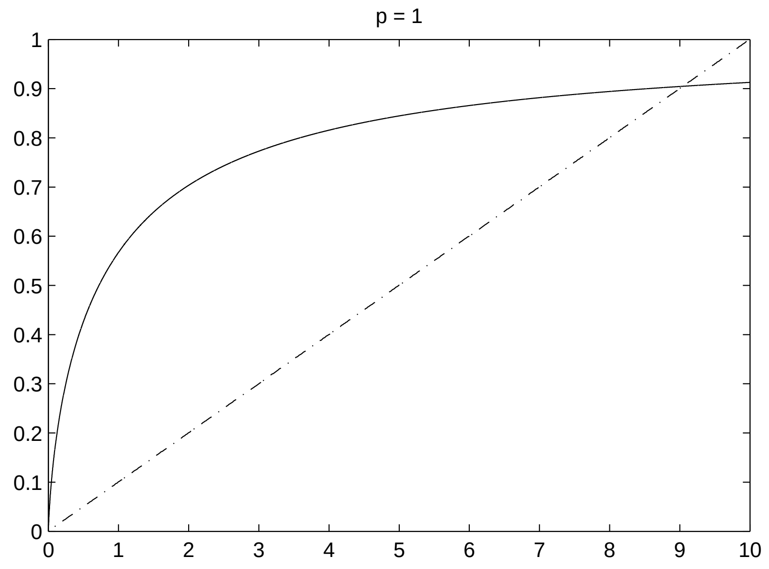


Figure 6.13: Comparative asymptotic fraction of support vectors versus ν of standard and quadric ν-SVR for polynomial noise of degree 1 (Laplacian noise). The solid line represents the efficiency of modified ν-SVR and the dotted line standard ν-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

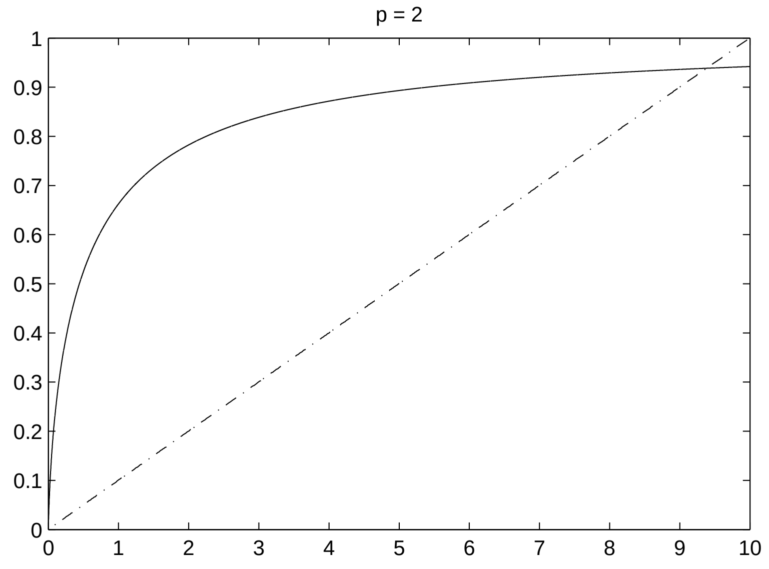


Figure 6.14: Comparative asymptotic fraction of support vectors versus ν of standard and quadric ν-SVR for polynomial noise of degree 2 (Gaussian noise). The solid line represents the efficiency of modified ν-SVR and the dotted line standard ν-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

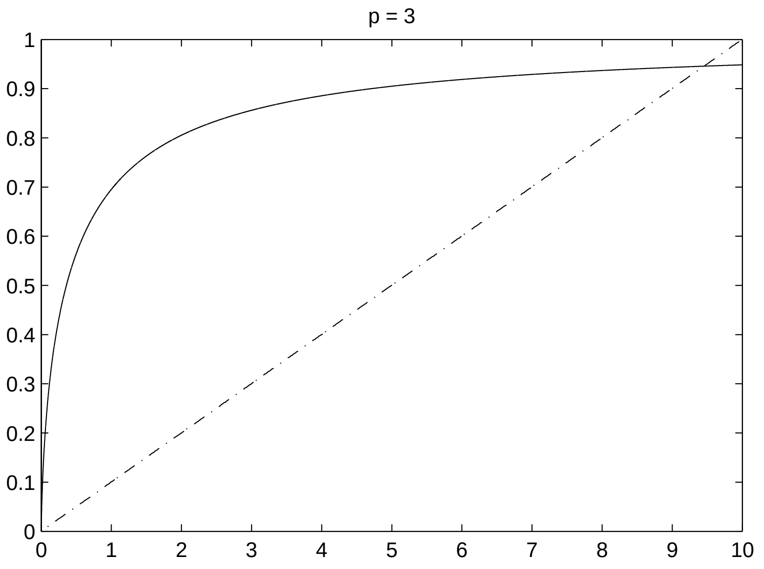


Figure 6.15: Comparative asymptotic fraction of support vectors versus ν of standard and quadric ν-SVR for polynomial noise of degree 3. The solid line represents the efficiency of modified ν-SVR and the dotted line standard ν-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

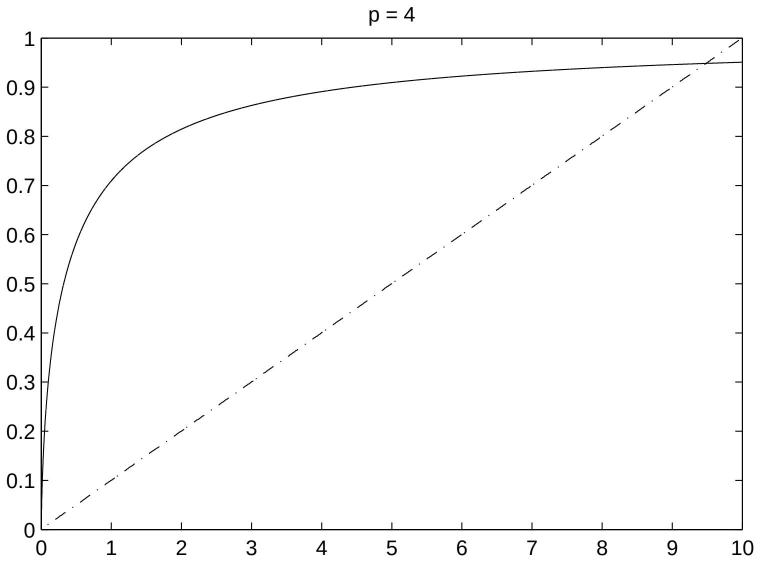


Figure 6.16: Comparative asymptotic fraction of support vectors versus ν of standard and quadric ν-SVR for polynomial noise of degree 4. The solid line represents the efficiency of modified ν-SVR and the dotted line standard ν-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

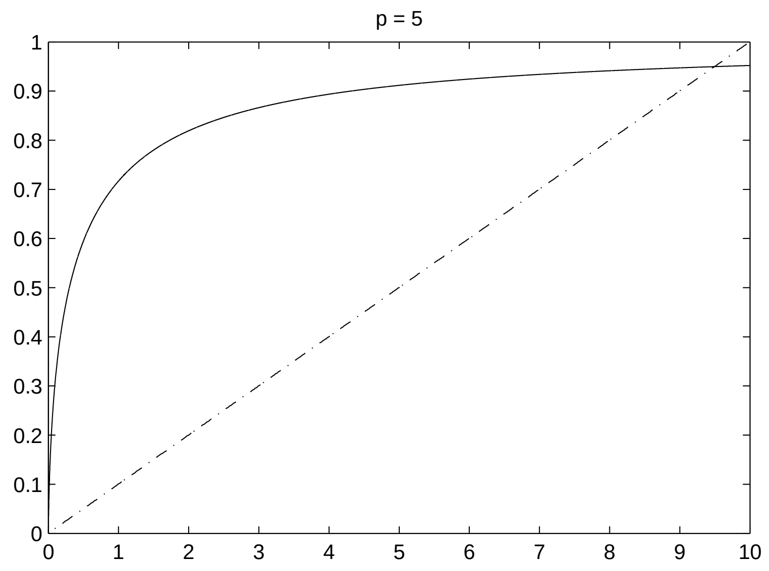


Figure 6.17: Comparative asymptotic fraction of support vectors versus ν of standard and quadric ν-SVR for polynomial noise of degree 5. The solid line represents the efficiency of modified ν-SVR and the dotted line standard ν-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

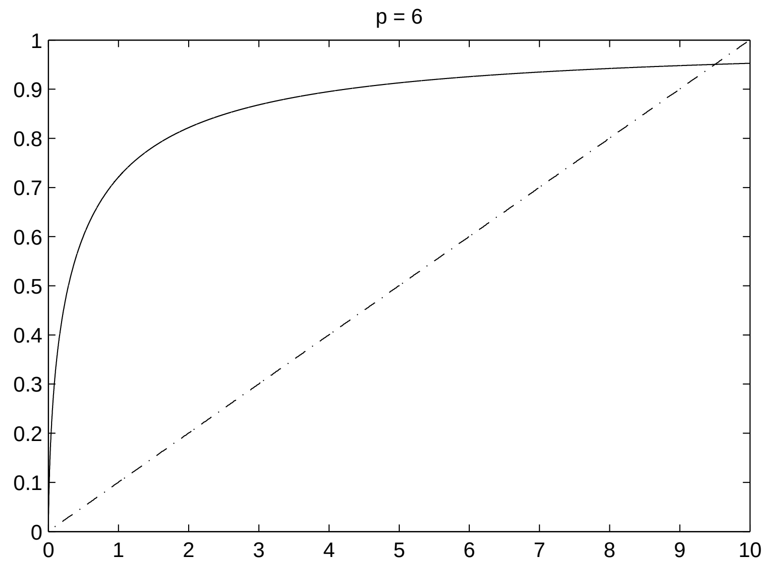


Figure 6.18: Comparative asymptotic fraction of support vectors versus ν of standard and quadric ν-SVR for polynomial noise of degree 6. The solid line represents the efficiency of modified ν-SVR and the dotted line standard ν-SVR. Note that the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

be satisfied for any SVR), it is important to seek some form of experimental confirmation of these results. The experimental results presented in this section were previously published in [78].

For ease of comparison the experimental procedure used here is modelled on that of [18]. The risk (in the form of root mean squared error (RMSE)) has been numerically computed as a function of ν for both standard and quadric ν-SVR methods, allowing clear comparison between the two methods. Furthermore, the RMSE for LS-SVR (which is the limiting case of quadric ν-SVR as $\nu \to \infty$) and (non-sparse) weighted LS-SVR has also been computed [88]. Plots of risk versus ν are given for polynomial noise of degree $1 \leq p \leq 6$ to compare the theoretical and experimental results, and some relevant results are given for the effect of other parameters on the ν curves.

Finally, the sparsity of standard and quadric ν-SVR for different orders of polynomial noise $1 \leq p \leq 6$ has been computed, and comparisons made with the theoretical predictions.

As in [18], the training set consisted of 100 examples (x_i, z_i) where x_i is drawn uniformly from the range $[-3, 3]$ and z_i is given by the noisy sinc function:

$$\begin{aligned} z_i &= \operatorname{sinc}(x_i) + \zeta_i \\ &= \tfrac{\sin(\pi x_i)}{\pi x_i} + \zeta_i \end{aligned}$$

where ζ_i represents additive polynomial noise. The test set consisted of 500 pairs (x_i, z_i) where the x_i's were equally spaced over the interval $[-3, 3]$ and z_i was given by the noiseless sinc function. 500 trials were carried out for each result.

By default, we set the parameter $C = 100$ and noise variance $\sigma = 0.5$. In all experiments we use the Gaussian RBF kernel function $K(\mathbf{x}, \mathbf{y}) = e^{-\frac{1}{2\sigma^2_{\text{kernel}}}\|\mathbf{x}-\mathbf{y}\|^2}$ where $2\sigma^2_{\text{kernel}} = 1$ by default.

6.5.1 Additive Polynomial Noise

In the first experiment, the training data has been affected by polynomial noise of degree $1 \leq p \leq 6$. Plots of RMSE for standard ν-SVR, quadric ν-SVR, LS-SVR

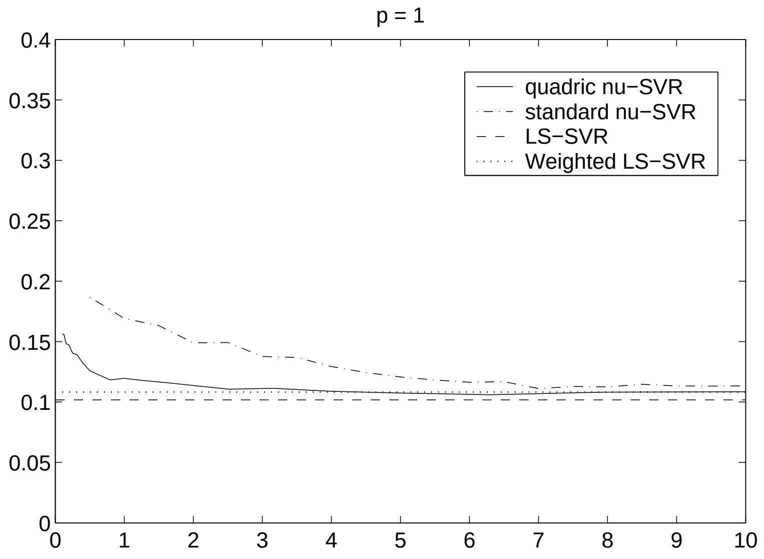


Figure 6.19: Risk (RMSE) versus ν for sinc data with polynomial noise of degree 1 (c.f. [18], figure 2). In these experiments, $\sigma = 0.5$, $C = 100$ and $2\sigma_{\mathrm{kernel}}^2 = 1$. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

and weighted LS-SVR are shown in figures 6.19 to 6.24. With the exception of Laplacian noise ($p = 1$), these results resemble the theoretical predictions shown in figures 6.7 to 6.12. Note that the quadric ν-SVR outperforms both standard ν-SVR and (weighted and unweighted) LS-SVR in all cases except Laplacian noise ($p = 1$).

Whilst the general shape of the RMSE curves closely resembles the shape of the predicted efficiency curves, it is important to note that the sharp optimum predicted by the theory for $p \geq 4$ (see figures 6.10 to 6.12) is not present in the experimental results. Instead, the region of optimality is somewhat to the right of this and also somewhat blunter.

From an application point of view, this is actually an advantage, as it means that results are far less sensitive to ν than expected. Indeed, from these results one may empirically say that the "sweet spot" for ν lies between 0.5 and 1 for polynomial noise of degree $p \geq 3$ and anything above 2 otherwise. But, roughly speaking, selecting $\nu = 1$ will in all cases presented give results superior to the standard ν-SVR method and at least comparable to the LS-SVR methods (better if $p \geq 3$).

In the case of Laplacian noise the actual performance (in terms of RMSE) of

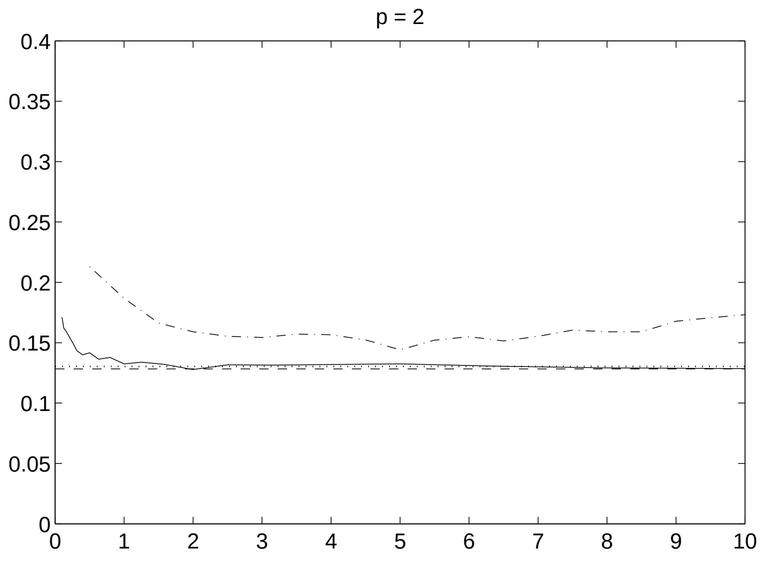


Figure 6.20: Risk (RMSE) versus ν for sinc data with polynomial noise of degree 2 (c.f. [18], figure 2). In these experiments, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

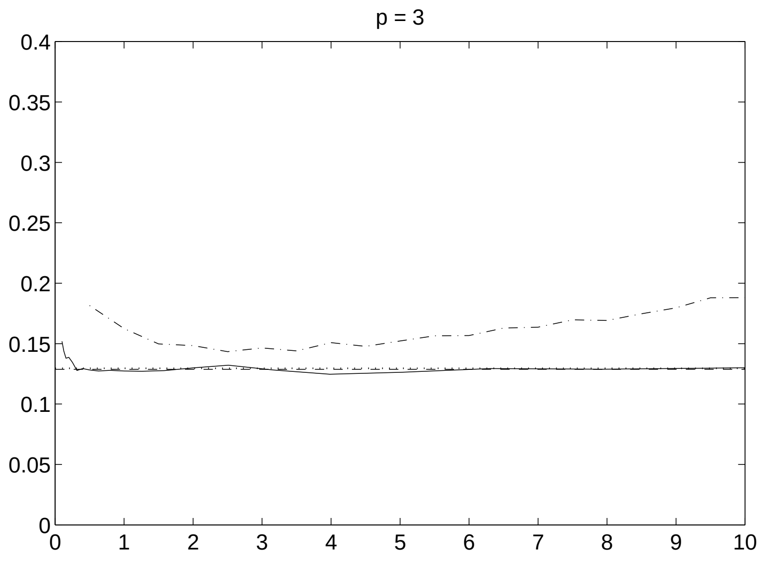


Figure 6.21: Risk (RMSE) versus ν for sinc data with polynomial noise of degree 3 (c.f. [18], figure 2). In these experiments, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

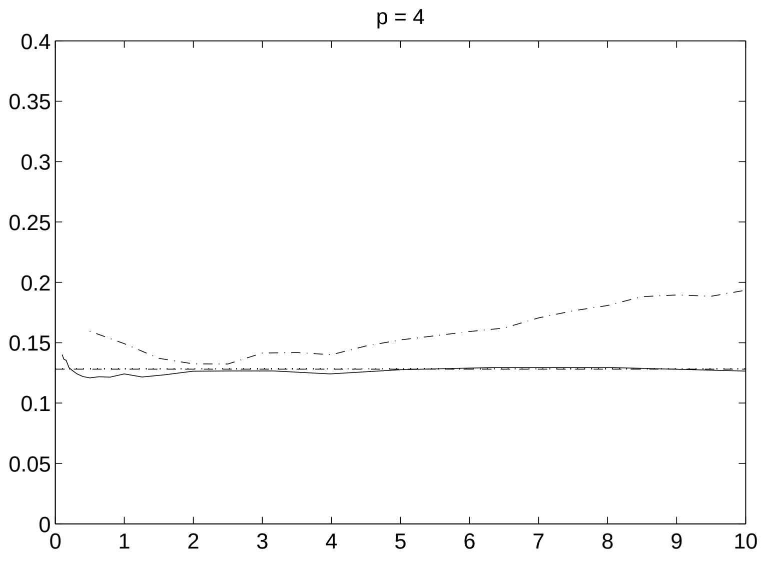


Figure 6.22: Risk (RMSE) versus ν for sinc data with polynomial noise of degree 4 (c.f. [18], figure 2). In these experiments, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

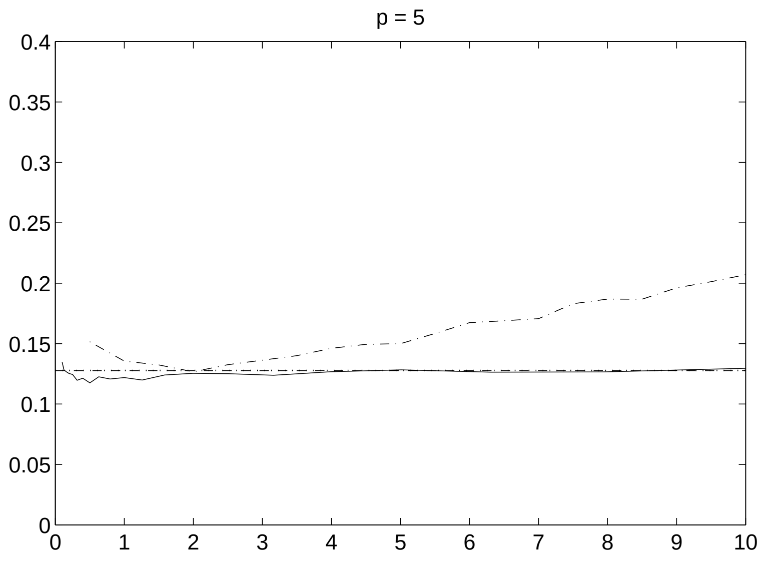


Figure 6.23: Risk (RMSE) versus ν for sinc data with polynomial noise of degree 5 (c.f. [18], figure 2). In these experiments, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

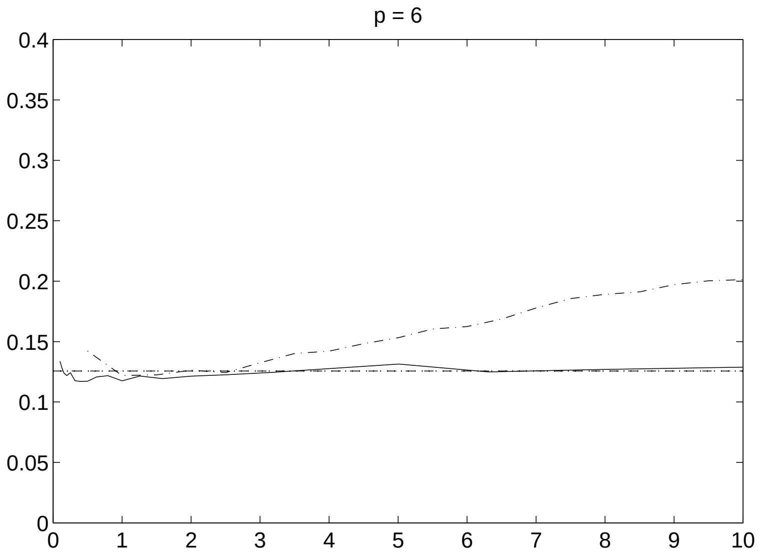


Figure 6.24: Risk (RMSE) versus ν for sinc data with polynomial noise of degree 6 (c.f. [18], figure 2). In these experiments, $\sigma = 0.5$, $C = 100$ and $2\sigma_{\text{kernel}}^2 = 1$. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual value of ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

both the quadric ν-SVR and the LS-SVRs is better than that of the standard ν-SVR, whereas theory would suggest that this should not be the case. The reason for this anomaly remain unclear.

Figures 6.25 to 6.30 show the ratio of support vectors to training vectors for both standard and quadric ν-SVRs as a function of ν. These curves closely match the predictions given in figures 6.13 to 6.18. It is clear from figures 6.19 to 6.24 and 6.25 to 6.30 that, as expected, the number of support vectors found by the quadric ν-SVR is substantially larger than the number found by the standard ν-SVR. However, this is still substantially less than the number of support vectors found by the LS-SVR (which uses all training vectors as support vectors).

6.5.2 Parameter Variation with Additive Gaussian Noise

In the second experiment, the performance of the standard and quadric ν-SVR in the presence of additive Gaussian noise is analysed as other parameters of the problem (particularly σ (the noise variance), C and σ_{kernel}) are varied, the aim being to see if the RMSE versus ν curves retains the same general form as these parameters are

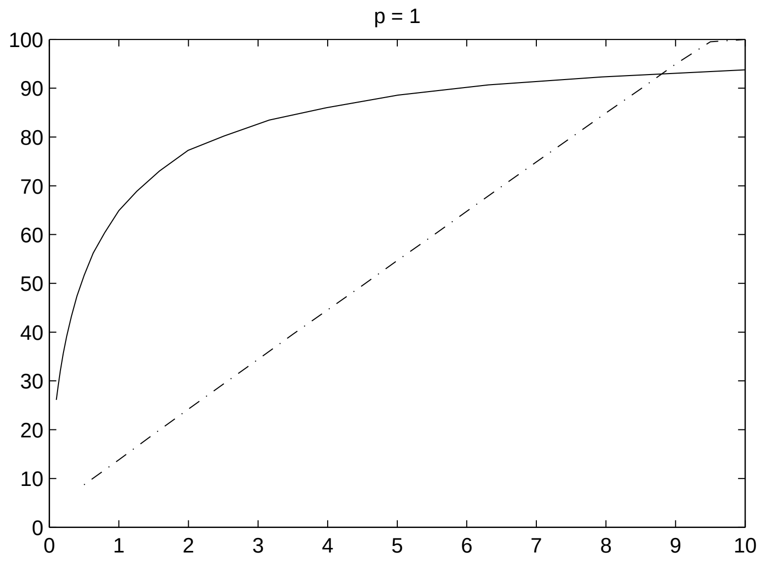


Figure 6.25: Number of support vectors (out of 100 training vectors) versus ν for sinc data with polynomial noise of degree 1. For this experiment, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\mathrm{kernel}} = 1$. The dotted line shows standard ν-SVR, and the solid line quadric ν-SVR. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual setting for ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

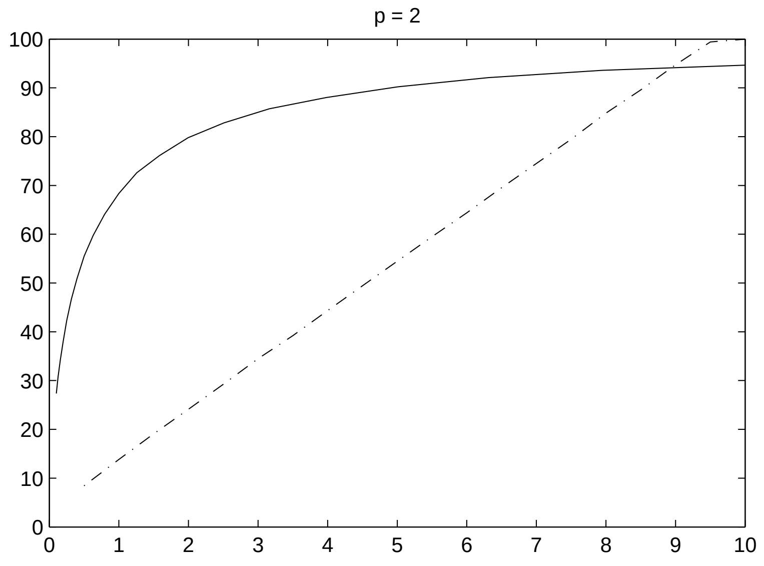


Figure 6.26: Number of support vectors (out of 100 training vectors) versus ν for sinc data with polynomial noise of degree 2. For this experiment, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\mathrm{kernel}} = 1$. The dotted line shows standard ν-SVR, and the solid line quadric ν-SVR. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual setting for ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

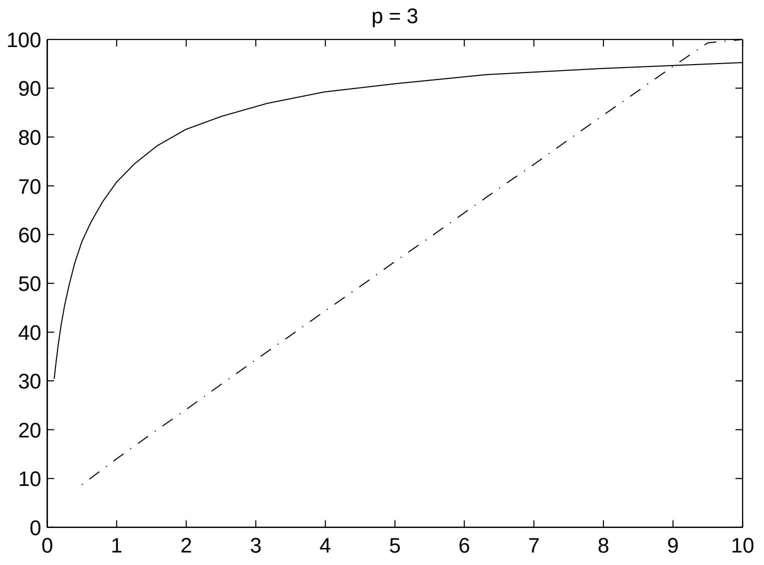


Figure 6.27: Number of support vectors (out of 100 training vectors) versus ν for sinc data with polynomial noise of degree 3. For this experiment, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. The dotted line shows standard ν-SVR, and the solid line quadric ν-SVR. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual setting for ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

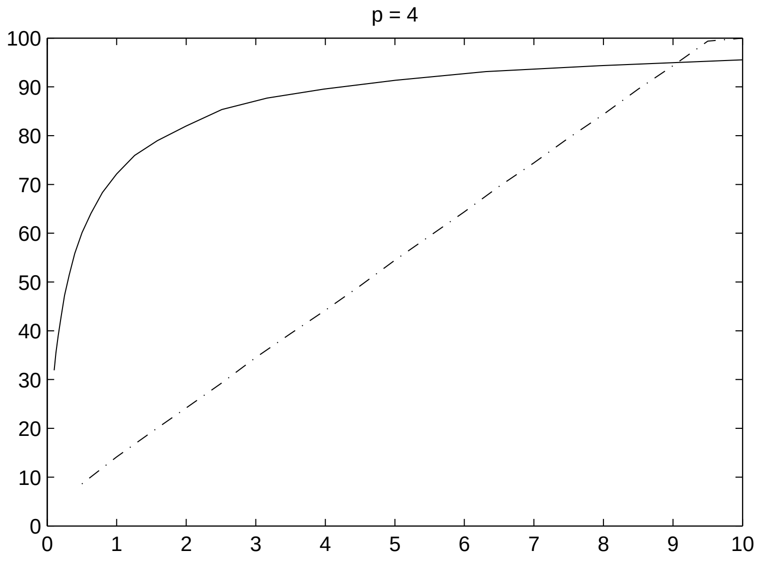


Figure 6.28: Number of support vectors (out of 100 training vectors) versus ν for sinc data with polynomial noise of degree 4. For this experiment, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. The dotted line shows standard ν-SVR, and the solid line quadric ν-SVR. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual setting for ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

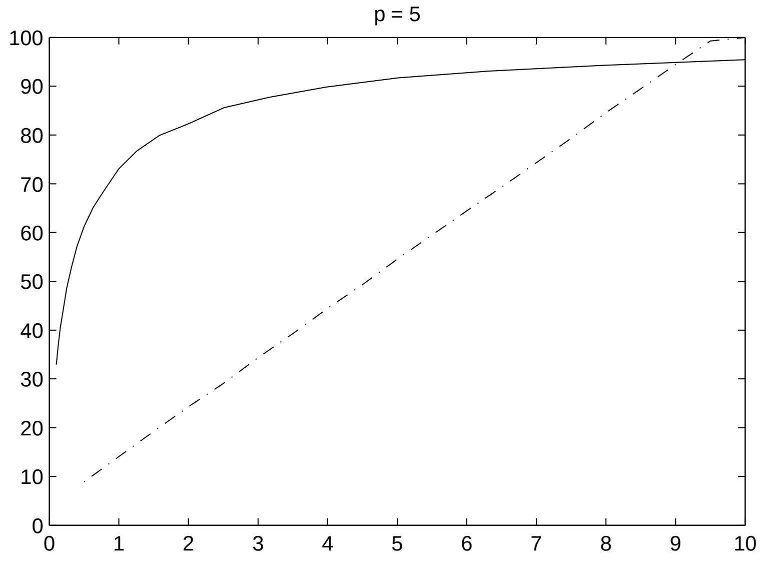


Figure 6.29: Number of support vectors (out of 100 training vectors) versus ν for sinc data with polynomial noise of degree 5. For this experiment, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. The dotted line shows standard ν-SVR, and the solid line quadric ν-SVR. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual setting for ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

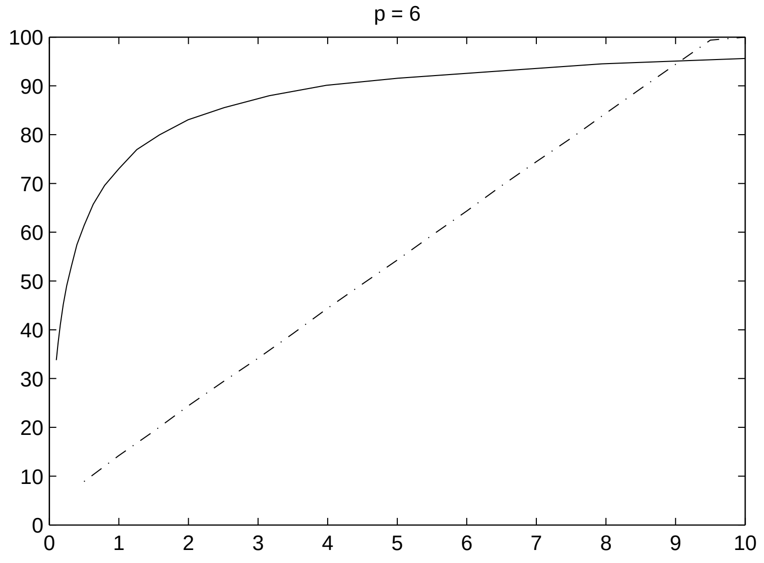


Figure 6.30: Number of support vectors (out of 100 training vectors) versus ν for sinc data with polynomial noise of degree 6. For this experiment, $\sigma = 0.5$, $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. The dotted line shows standard ν-SVR, and the solid line quadric ν-SVR. Note that the ν-scale differs for standard and modified ν-SVRs. For standard ν-SVRs, the actual setting for ν is one tenth of that indicated on the x axis (for modified ν-SVRs, ν is as indicated).

varied.

The top row of figure 6.31 shows the form of the RMSE versus ν curve for a range of noise levels. In all cases the form of the curves remains essentially unchanged. It will be noted, however, that for the lowest noise case ($\sigma = 0.1$) the standard ν-SVR is able to out-perform the quadric ν-SVR. Once again, selecting $\nu > 1$ gives reasonable performance in all cases (c.f. Smola's result for standard ν-SVR [83], where the "optimal area" is $\nu \in [0.3, 0.8]$).

The middle row of figure 6.31 shows the same curve with the noise variance σ fixed for different values of C, namely $C \in \{10, 100, 1000\}$. Once again, the RMSE curve for quadric ν-SVR is roughly as predicted, and $\nu > 1$ gives reasonable results. In this case, $C = 10$ provides an anomalous result.

Finally, in the bottom row of figure 6.31, shows the RMSE versus ν curves when the kernel parameter $2\sigma^2_{\text{kernel}}$ is varied. These results follow the same pattern as for variation of σ and C.

It is interesting to note here that, while the RMSE versus ν curve obtained using the quadric ν-SVR fits more closely the predicted curve than does the same curve for standard ν-SVR when parameters are chosen badly, this does not imply that the performance of the quadric ν-SVR will necessarily be better than the standard ν-SVR in this case. Indeed, the two cases where other SV parameters (i.e. not ν) have been chosen badly (i.e. $C = 10$ and $2\sigma^2_{\text{kernel}} = 0.1$ in figure 6.31) are the two cases where the standard ν-SVR most outperforms the quadric ν-SVR. However, as one should continue to search until appropriate parameters are found, this should not be too much of a problem in practical situations.

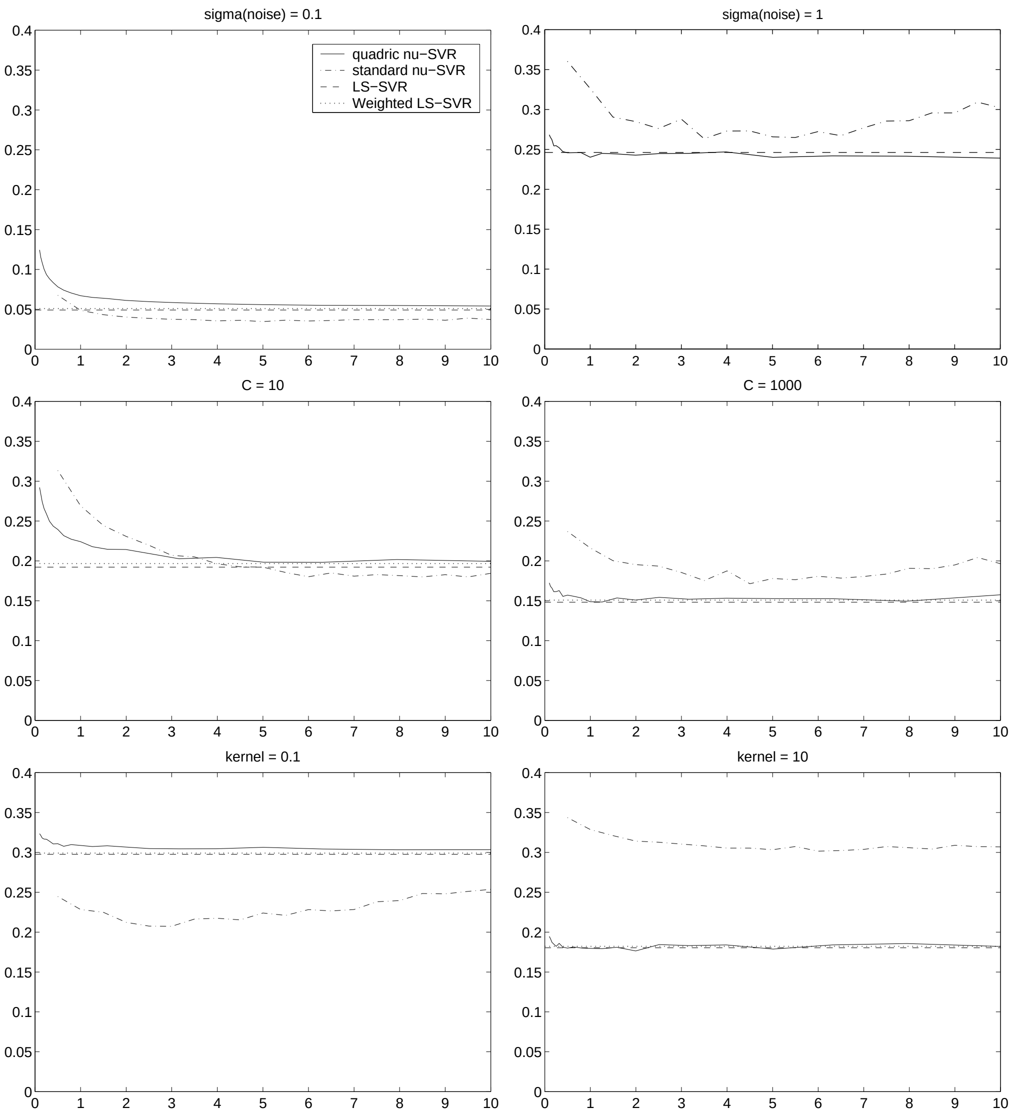


Figure 6.31: Risk (RMSE) versus ν for sinc data with Gaussian noise (c.f. [18], figure 1). The top row shows performance for different noise variances, $\sigma \in \{0.1, 1\}$ from left to right, with $C = 100$ and $2\sigma^2_{\text{kernel}} = 1$. The middle row gives performance for $C \in \{10, 1000\}$, respectively, with $\sigma = 0.5$ and $2\sigma^2_{\text{kernel}} = 1$. Finally, the bottom row shows performance for $2\sigma^2_{\text{kernel}} \in \{0.1, 10\}$, respectively, with $\sigma = 0.5$ and $C = 100$. Note that for all graphs the ν-scale differs for standard and quadric ν-SVRs. For standard ν-SVRs, the actual setting for ν is one tenth of that indicated on the x axis (for quadric ν-SVRs, ν is as indicated).

Chapter 7

TRAINING THE SVM - PRELIMINARIES

You know, I always say that a day without an autopsy is like a day without sunshine.

– Buffy

IMPLEMENTATION of the various SVM formulations introduced in chapters 4 and 5 invariably requires the solving of some form of quadratic program (although it should be noted that not all SVM formulations result in quadratic programs, e.g. [86] [50] [11] [98]), whether this be the SVM optimisation primal, the dual or the partial dual (where applicable). Over the next two chapters I will introduce the SVM optimisation algorithm presented in [81] (and prior to that, in somewhat limited form, [80]), which was later extended [63] to cover regression with inequalities. This chapter will introduce the necessary background material and definitions. Details of the algorithm itself are contained in chapter 8.

7.1 Scope and Limitations

Because there exist such a wide range of SVM formulations and modifications (some of which have been introduced in chapters 4 and 5), it is important to clearly define the scope of applicability for any algorithm dealing with the SVM optimisation problem. In this section I outline the limitations of my algorithm and the reasoning behind these limits

7.1.1 Primal, Dual and Partially Dual

In the basic regression with inequalities SVM introduced in section 5.1 (which covers both regression and pattern classification in their basic form), there are (usually) three forms in which the problem may be solved, namely primal, dual and partially dual.

The primal form of the problem is a quadratic program in d_H dimensions with complex linear constraints. However, as has been discussed previously, this form makes high dimensional feature spaces difficult to deal with, and infinite dimensional feature spaces impossible. Furthermore one cannot use the kernel trick here. This makes direct solution of the primal problem unattractive, so I have chosen to ignore this method.

The dual form overcomes many of the shortcomings of the primal form. It to is a quadratic programming problem. However the feature map is now hidden by the kernel function, and the size of the problem is N (where N is the size of the training set). Each of the dual variables α_i has at least a lower bound, and there are between 0 (fixed bias) and 2 (variable bias, linear tube shrinking) additional linear equality constraints. Furthermore, the hessian is at least positive semidefinite (and may be positive definite, depending on the exact form of the problem - for example, the LS-SVR is guaranteed to have a positive definite hessian). I have chosen to use this form for the fixed-bias SVM (where there are no additional linear equality constraints) because in this case the dual is particularly simple.

Finally, in all but the fixed bias case (without linear tube shrinking) there is the partially dual form. This form is very similar to the dual form, except that the size of the quadratic program is increased from N to either $N + 1$ or $N + 2$, and the constraint set now consists only of simple bounds on each of the dual variables $\boldsymbol{\alpha}$. The downside to this is that the positive semi-definite nature of the hessian is lost (although if the hessian of the dual is positive definite then the hessian of the partially dual must be non-singular, as will be shown later). This is the form I will be using for all but the fixed bias SVM.

7.1.2 Limits on the SVM Formulation

In developing a training algorithm for SVM methods (both regression and pattern classification), it is necessary to trade-off between the conflicting requirements of generality, speed and space. Specifically:

- *Generality*: The algorithm should cover as many of the diverse SVM types as possible, from the standard pattern classification and regression methods of chapter 4 to the general cost function regressor with inequalities and tube shrinking of section 5.5.1.

- *Speed*: The algorithm should be find the optimal (or close enough for practical purposes) solution as quickly as possible.

- *Memory use*: Computer memory is finite, and SVM training sets can be extremely large, so memory conservation is an important design factor, especially given that one may wish to use whatever algorithms are developed in applications where memory resources are scarce (mobile telephony, for example).

Of course, the requirement for speed is essentially in conflict with the requirement for generality. To optimise an algorithm's speed, one must be able to take advantage of the specific properties of the problem at hand, avoiding all unnecessary calculations and tests. But to increase generality some calculations and checks must be done "just in case", even though they may not be needed for the particular case at hand, compromising the algorithm's speed.

Similarly, the requirements of speed and minimum memory usage are in some ways conflicting. Storing very large matrices, for example, takes a lot of memory. However, if the elements of such matrices are needed repeatedly then re-calculating them at regular intervals may result in a noticeable degradation of performance. Of course, if memory usage is excessive (i.e. exceeds the available conventional memory) then there will be an impact on speed due to the practical necessity of swapping blocks of data in and out of physical memory, resulting in thrashing, so these requirements are not entirely at odds.

As a trade-off I have restricted my work thusly:

1. Only SVMs with linear or quadratic empirical risk functions will be dealt with.

2. Linear (conventional) tube shrinking will not be considered.

3. It will be assumed that the training dataset is sufficiently small, and the computers memory sufficiently large that all data will fit into conventional memory,

avoiding thrashing problems due to disk/memory swapping.

So I will be not be considering the SVM optimisation problem associated with a general cost function, as described in section 5.5.1, nor the standard tube-shrinking problem. I will, however, be considering the case of quadratic cost (including LS-SVMs, although it should be noted that my approach in this special case is not optimal), and quadratic tube shrinking. It should be noted that assumption 3 is not intended to imply that I pay no attention to memory usage. It simply means that I will be ignoring the complex issues of thrashing and other effects due to memory limitations that, while important for very large datasets, are beyond the scope of this thesis.

7.2 Abstract Problem Formulation

In section 5.1 I introduced the new SVM formulation of regression with inequalities and (in subsection 5.1.1) demonstrated how the standard SVM formulations of binary pattern recognition and regression may be viewed as simply special cases of regression with inequalities. Specifically, the special cases are retrieved by applying restrictions to the general training set, namely:

1. Pattern recognition: $\mathbf{Y}_= = \emptyset$, $\mathbf{z} = \mathbf{0}$, $\boldsymbol{\epsilon}, \boldsymbol{\epsilon}^* \leq \mathbf{0}$.

2. Standard regression: $\mathbf{Y}_\leq = \mathbf{Y}_\geq = \emptyset$.

One key advantage of taking this approach is that it reduces the amount of repetition required when dealing with extensions to the basic model that are applicable to both the pattern recognition and regression forms of the SVM - rather than repeating essentially the same method for both pattern recognition and classification, one need only describe the method once for the more general formulation.

The aim of the abstract problem formulation is much the same. Rather than describing several essentially identical variants of the same algorithm to deal with the variants of the SVM problem, a single algorithm is described which may be applied to many of the variants by appropriately mapping the original (specific) training set onto an abstract training set and then using the generic training algorithm, with appropriate interpretation of the resulting function $g(\mathbf{x})$.

7.2.1 The Abstract Training Set

The abstract training set $\mathbf{T}$ consists of the training data:

$$\begin{aligned}
& \mathbf{T} = (\boldsymbol{\vartheta}, \text{BiasType}, b_{\text{fix}}, K) \\
\text{where : } \quad & \boldsymbol{\vartheta} = (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
& \vartheta_i = (\mathbf{x}_i, z_i, \rho_i, \rho_i^*, h_i, v_i, \gamma_i, \gamma_i^*, \mu_i, \mu_i^*) \\
& \mathbf{x}_i \in \Re^{d_L} \\
& b_{\text{fix}} \in \Re \\
& z_i, \rho_i, \rho_i^*, v_i, h_i, \gamma_i, \gamma_i^*, \mu_i, \mu_i^* \in \Re \\
& \rho_i, \rho_i^* \geq 0 \text{ if } v_i < 0 < h_i \\
& \gamma_i, \gamma_i^*, \mu_i, \mu_i^* \geq 0 \\
& v_i \leq 0 \\
& h_i \geq 0 \\
& \text{FixBias} \in \{\text{Var}, \text{Fix}\} \\
& K \in \mathcal{M}_\kappa
\end{aligned} \tag{7.1}$$

where the binary variable BiasType is Fix for fixed bias operation ($b = b_{\text{fix}}$) or Var for normal (variable bias) operation, and K is the usual kernel function. The exact connection between this training set and the standard SVM training sets will be described shortly. Roughly speaking, however:

- $\mathbf{x}_i$ and z_i are the standard training pair for regression with inequalities.
- ρ_i and μ_i play a role similar to $E\epsilon_i$. However, in this case ρ_i refers specifically to the non tube-shrinking case, whereas μ_i relates specifically to the quadratic tube shrinking case. While it is possible to have give both elements non-zero values, this will not correspond to any of the SVM formulations given previously. The positivity requirement made on ρ_i if $v_i < 0 < h_i$ (which, as will be seen, is only applicable to regression) is a technicality which will be required later.

- ρ_i^* and μ_i^* play the analogous role for $E\epsilon_i^*$.
- v_i and h_i act as lower and upper bounds on α_i. That is, $\alpha_i \in [v_i, h_i]$. Thus these play the role of $\frac{C}{N}t_i^*$ and $\frac{C}{N}t_i$ in the standard SVM formulation. While they must be finite, for quadratic cost they are typically set to a very large value, so providing an effectively infinite range. This requirement of finiteness is both technically (as it makes the proof of convergence easier) and also practically (as computers typically cannot handle arbitrarily large numbers well) useful.
- γ_i and γ_i^* provide a diagonal offset to the Hessian, making possible the use of quadratic cost functions.

7.2.2 The Abstract Optimisation Problem

Given a training set $\mathbf{T}$, consider the following optimisation problem:

$$\{\boldsymbol{\beta}, \boldsymbol{\beta}^*, b\} = \arg \min_{\boldsymbol{\beta}, \boldsymbol{\beta}^* \in \Re^N} \max_{b \in \Re} Q_{\mathrm{BiasType}}\left(\boldsymbol{\beta}, \boldsymbol{\beta}^*, b\right) \tag{7.2}$$

$$\begin{aligned} \text{such that:} \quad & b = b_{\mathrm{fix}} \text{ if BiasType} = \text{Fix} \\ & \mathbf{0} \leq \boldsymbol{\beta} \leq \mathbf{h} \\ & \mathbf{v} \leq \boldsymbol{\beta}^* \leq \mathbf{0} \end{aligned}$$

where:

$$Q_{\mathrm{Var}}\left(\boldsymbol{\beta}, \boldsymbol{\beta}^*, b\right) = \frac{1}{2} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \\ b \end{bmatrix}^T \begin{bmatrix} \mathbf{Q} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \\ b \end{bmatrix} - \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \\ b \end{bmatrix}^T \begin{bmatrix} \mathbf{z} - \boldsymbol{\rho} \\ \mathbf{z} + \boldsymbol{\rho}^* \\ 0 \end{bmatrix} \tag{7.3}$$

$$Q_{\mathrm{Fix}}\left(\boldsymbol{\beta}, \boldsymbol{\beta}^*, b\right) = \frac{1}{2} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \end{bmatrix}^T \mathbf{Q} \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \end{bmatrix} - \begin{bmatrix} \boldsymbol{\beta} \\ \boldsymbol{\beta}^* \end{bmatrix}^T \begin{bmatrix} \mathbf{z} - \mathbf{1}b - \boldsymbol{\rho} \\ \mathbf{z} - \mathbf{1}b + \boldsymbol{\rho}^* \end{bmatrix} \tag{7.4}$$

and:

$$\begin{aligned} \mathbf{Q} &= \begin{bmatrix} \mathbf{K} + \boldsymbol{\mu}\boldsymbol{\mu}^T + \mathrm{diag}\left(\boldsymbol{\gamma}\right) & \mathbf{K} - \boldsymbol{\mu}\boldsymbol{\mu}^{*T} \\ \mathbf{K} - \boldsymbol{\mu}^*\boldsymbol{\mu}^T & \mathbf{K} + \boldsymbol{\mu}^*\boldsymbol{\mu}^{*T} + \mathrm{diag}\left(\boldsymbol{\gamma}^*\right) \end{bmatrix} \\ K_{ij} &= K\left(\mathbf{x}_i, \mathbf{x}_j\right) \end{aligned}$$

Now:

Theorem 7.1. *For any training set* $\mathbf{T}$ *as defined by (7.1) and an associated solution* $\{\boldsymbol{\beta}, \boldsymbol{\beta}^*, b\}$ *to (7.2), one or both of* β_i *and* β_i^* *must be zero for all* $1 \leq i \leq N$.

Proof. Note that:

$$\begin{aligned}
\mathbf{e} &= \frac{\partial Q_{\text{BiasType}}}{\partial \boldsymbol{\beta}} \\
&= \mathbf{k} - \mathbf{z} + E\boldsymbol{\mu} + \boldsymbol{\gamma}\boldsymbol{\beta} + \boldsymbol{\rho} \qquad (7.5) \\
\mathbf{e}^* &= \frac{\partial Q_{\text{BiasType}}}{\partial \boldsymbol{\beta}^*} \\
&= \mathbf{k} - \mathbf{z} - E\boldsymbol{\mu}^* + \boldsymbol{\gamma}^*\boldsymbol{\beta}^* - \boldsymbol{\rho}^* \qquad (7.6)
\end{aligned}$$

where the following are defined for convenience:

$$\begin{aligned}
E &= \boldsymbol{\mu}^T \boldsymbol{\beta} - \boldsymbol{\mu}^{*T} \boldsymbol{\beta}^* \\
\mathbf{k} &= \mathbf{K}\left(\boldsymbol{\beta} + \boldsymbol{\beta}^*\right) + \mathbf{1}b
\end{aligned}$$

noting that this implies that $E \geq 0$.

Suppose the theorem is false. That is, suppose $0 < \beta_i \leq h_i$ and $v_i \leq \beta_i^* < 0$ for some $1 \leq i \leq N$. Then, for optimality, $e_i \leq 0$ and $e_i^* \geq 0$ or, equivalently, $e_i + r_i = e_i^* - r_i^* = 0$ for some $r_i, r_i^* \geq 0$. Substituting into (7.5) and (7.6) it may be seen that $\beta_i = \frac{\gamma_i^*}{\gamma_i}\beta_i^* - \frac{E\mu_i + E\mu_i^* + \rho_i + \rho_i^* + r_i + r_i^*}{\gamma_i}$. As $E, \gamma_i, \gamma_i^*, \mu_i, \mu_i^*, \rho_i, \rho_i^*, r_i, r_i^* \geq 0$ (the constraint $\rho_i, \rho_i^* \geq 0$ follows from the fact that $v_i < 0 < h_i$, which in turn follows from the supposition that $0 < \beta_i \leq h_i$ and $v_i \leq \beta_i^* < 0$) and $\beta_i^* < 0$, it follows that $\beta_i < 0$, which contradicts the original supposition $0 < \beta_i \leq h_i$. Hence the solution cannot satisfy both $0 < \beta_i \leq h_i$ and $v_i \leq \beta_i^* < 0$ for any $1 \leq i \leq N$.

It follows that for any optimal solution it cannot be true that both $0 < \beta_i \leq h_i$ and $v_i \leq \beta_i^* < 0$ for any $1 \leq i \leq N$, which means that one or both of β_i and β_i^* must be zero for all $1 \leq i \leq N$. □

Consider the optimisation functions (7.3) and (7.4). It will be noted that both are particularly simple $2N$-dimensional (or $(2N+1)$-dimensional) quadratic equations, with none of the (apparent) discontinuities found in the SVM regression with inequalities partial dual (5.5). However, the (apparent) dimensionality of (7.3) and (7.4) is roughly twice that of (5.5), which is not a good sign if we intend using (7.4) for SVM optimisation work. Theorem 7.1 holds the key to reducing this dimensionality without losing the underlying smoothness of the problem.

Before doing so, however, some useful notation must be introduced. The *type* vector $\boldsymbol{\tau} \in \{-2,-1,0,1,2\}^N$ is defined thusly:

$$\tau_i = \begin{cases} -2 & \text{if } v_i = \alpha_i \wedge v_i < 0 \\ -1 & \text{if } v_i \leq \alpha_i \leq 0 \wedge v_i < 0 \\ 0 & \text{if } \alpha_i = 0 \\ +1 & \text{if } h_i \geq \alpha_i \geq 0 \wedge h_i > 0 \\ +2 & \text{if } h_i = \alpha_i \wedge h_i > 0 \end{cases} \tag{7.7}$$

neglecting (for now) the apparent ambiguity in this definition. Based on this, it is also convenient to define the vectors $\boldsymbol{\rho}_{\boldsymbol{\tau}}^{(*)} \in \Re^N$, $\boldsymbol{\gamma}_{\boldsymbol{\tau}}^{(*)} \in \Re^N$ and $\boldsymbol{\mu}_{\boldsymbol{\tau}}^{(*)} \in \Re^N$ using:

$$\begin{aligned} \rho_{\boldsymbol{\tau}}^{(*)}{}_i &= \begin{cases} \rho_i & \text{if } \tau_i > 0 \\ 0 & \text{if } \tau_i = 0 \\ \rho_i^* & \text{if } \tau_i < 0 \end{cases} \\ \gamma_{\boldsymbol{\tau}}^{(*)}{}_i &= \begin{cases} \gamma_i & \text{if } \tau_i > 0 \\ 0 & \text{if } \tau_i = 0 \\ \gamma_i^* & \text{if } \tau_i < 0 \end{cases} \\ \mu_{\boldsymbol{\tau}}^{(*)}{}_i &= \begin{cases} \mu_i & \text{if } \tau_i > 0 \\ 0 & \text{if } \tau_i = 0 \\ \mu_i^* & \text{if } \tau_i < 0 \end{cases} \end{aligned} \tag{7.8}$$

where the subscript $\boldsymbol{\tau}$ indicates the implicit dependence on the type vector $\boldsymbol{\tau}$.

Noting that $g(\mathbf{y})$ is dependent only on $\boldsymbol{\beta}+\boldsymbol{\beta}^*$ and b it can be seen that, defining

$\boldsymbol{\alpha} = \boldsymbol{\beta} + \boldsymbol{\beta}^*$, the abstract optimisation problem may be re-written:

$$\begin{aligned} \{\boldsymbol{\alpha}, b\} &= \arg\min_{\boldsymbol{\alpha}\in[\mathbf{v},\mathbf{h}]} \arg\max_{b\in\Re} Q_{\mathrm{BiasType}}(\boldsymbol{\alpha}, b) && (7.9)\\ &\text{such that: } b = b_{\mathrm{fix}} \text{ if BiasType} = \mathrm{Fix} && (7.10) \end{aligned}$$

where:

$$\begin{aligned} Q_{\mathrm{Var}}(\boldsymbol{\alpha}, b) &= \frac{1}{2}\begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix}^T \begin{bmatrix} \mathbf{G}_{\boldsymbol{\mathcal{T}}} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} - \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix}^T \begin{bmatrix} \mathbf{z} - \mathrm{sgn}(\boldsymbol{\tau})\,\boldsymbol{\rho}_{\boldsymbol{\mathcal{T}}}^{(*)} \\ 0 \end{bmatrix} && (7.11)\\ Q_{\mathrm{Fix}}(\boldsymbol{\alpha}, b) &= \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{G}_{\boldsymbol{\mathcal{T}}}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathbf{1}b - \boldsymbol{\alpha}^T\left(\mathbf{z} - \mathrm{sgn}(\boldsymbol{\tau})\,\boldsymbol{\rho}_{\boldsymbol{\mathcal{T}}}^{(*)}\right) && (7.12) \end{aligned}$$

and:

$$\mathbf{G}_{\boldsymbol{\mathcal{T}}} = \mathbf{K} + \mathrm{diag}\left(\boldsymbol{\gamma}_{\boldsymbol{\mathcal{T}}}^{(*)}\right) + \left(\mathrm{sgn}(\boldsymbol{\tau})\,\boldsymbol{\mu}_{\boldsymbol{\mathcal{T}}}^{(*)}\right)\left(\mathrm{sgn}(\boldsymbol{\tau})\,\boldsymbol{\mu}_{\boldsymbol{\mathcal{T}}}^{(*)}\right)^T$$

There are several things to note here. Firstly, note that theorem 7.1 allows us to reconstruct the complete solution $\{\boldsymbol{\beta}, \boldsymbol{\beta}^*, b\}$ to (7.2) using the solution $\{\boldsymbol{\alpha}, b\}$ to the N-dimensional (or $(N+1)$-dimensional) optimisation problem (7.9) using the type vector $\boldsymbol{\tau}$ thusly:

$$\beta_i = \begin{cases} \alpha_i & \text{if } \tau_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\beta_i^* = \begin{cases} \alpha_i & \text{if } \tau_i < 0 \\ 0 & \text{otherwise} \end{cases}$$

or, reversing this logic:

$$\alpha_i = \begin{cases} \beta_i & \text{if } \tau_i > 0 \\ 0 & \text{if } \tau_i = 0 \\ \beta_i^* & \text{if } \tau_i < 0 \end{cases} \tag{7.13}$$

Secondly, note that whereas (7.2) is a smooth optimisation problem, (7.9) would appear to be discontinuous due to the discontinuous implicit dependencies on $\boldsymbol{\tau}$ throughout. Fortunately, this presents no real difficulties so long one views (7.9) not as a problem in its own right, but simply as a shorthand way of writing (7.2) to

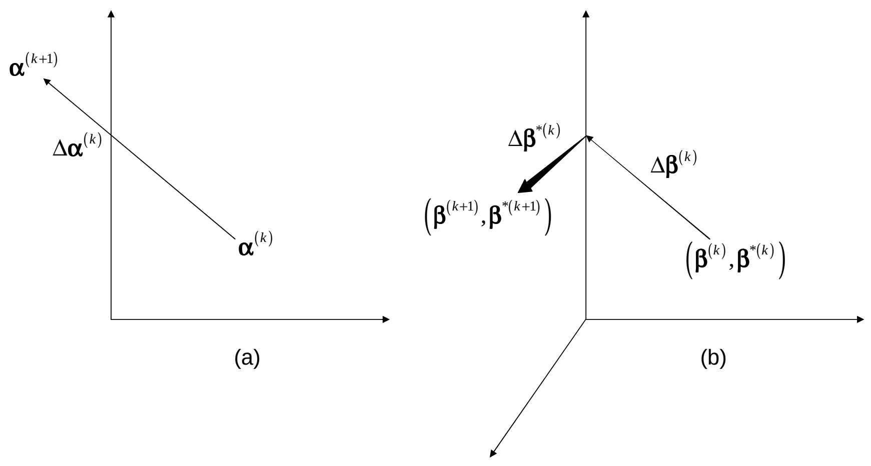


Figure 7.1: Effect of changing the sign of $\boldsymbol{\alpha}$ in one step. (a) shows the apparent change in $\boldsymbol{\alpha}$, while (b) shows (schematically) the actual two steps involved in terms of $\boldsymbol{\beta}$ and $\boldsymbol{\beta}^*$.

highlight the sparseness given by theorem 7.1.

Of course, when solving (7.9) some care must be taken. In particular, noting that α_i is simply a placeholder for a variable which is either strictly non-negative or strictly non-positive, if an iterative method is used to find the solution care must be taken to ensure that the sign of α_i does not go from positive to negative (or vice-versa) for any given iteration, as this would imply that what α_i is acting as a placeholder for is changing mid-step, which could lead to difficulties. This problem is shown schematically in figure 7.1. As will be seen shortly, this is the same as ensuring that $\boldsymbol{\tau}$ is held constant for any given iteration.

Given a solution $\{\boldsymbol{\alpha}, b\}$ of (7.9), the "trained machine" associated with this is defined to be the function:

$$g(\mathbf{y}) = \sum_{i=1}^{N} \alpha_i K(\mathbf{x}_i, \mathbf{y}) + b$$

7.3 The Order Vector

When formulating the abstract optimisation problem (7.9) it has been assumed that the ordering of the elements in the solution $\{\boldsymbol{\alpha}, b\}$ is of little import, and hence for convenience corresponds to the ordering of elements in the training set $\mathbf{T}$ (and in particular $\boldsymbol{\vartheta}$). However, this ordering may not be convenient when actually attempting to find this solution - for example, it may be more convenient to group the support vectors together. Of course, if an iterative method is used to find the solution then this order will not be constant, but will instead vary from one iteration to the next.

While it is convenient to re-order the elements of the problem after each iteration, actually doing so would be an unnecessary exercise in bookkeeping which may have a serious impact of the speed of the training algorithm. Fortunately it is possible to attain this convenience without the added book-keeping by making the re-ordering implicit. One way to do this is to define an *order vector* $\mathbf{m} \in \mathbb{Z}^N$, where $1 \leq m_i \leq N$ for all $1 \leq i \leq N$ is an integer, and $m_i \neq m_j$ for all $1 \leq i, j \leq N$, $i \neq j$.

Using this order vector, one may associate with any other vector $\mathbf{q}$ a alternative "view" $\underline{\mathbf{q}}$ of this vector, where:

$$\underline{q}_i = q_{m_i}$$

similarly, for any $N \times N$ matrix $\mathbf{R}$, one may define $\underline{\mathbf{R}}$ using:

$$\underline{R}_{i,j} = R_{m_i,m_j}$$

then, rather than re-ordering $\underline{\mathbf{q}}$, $\underline{\mathbf{R}}$, and any other vectors and matrices in use for each iteration, one can simply re-order the order vector $\mathbf{m}$ instead, which implicitly re-orders the other objects (via the above definition).

Using this notation, the abstract optimisation problem (7.9) can be trivially re-written:

$$\begin{aligned} \{\underline{\boldsymbol{\alpha}}, b\} &= \arg\min_{\underline{\boldsymbol{\alpha}} \in [\underline{\mathbf{v}}, \underline{\mathbf{h}}]} \arg\max_{b \in \Re} Q_{\text{BiasType}}\left(\underline{\boldsymbol{\alpha}}, b\right) && (7.14) \\ &\text{such that: } b = b_{\text{fix}} \text{ if BiasType} = \text{Fix} && (7.15) \end{aligned}$$

where:

$$\begin{aligned} Q_{\text{Var}}\left(\underline{\boldsymbol{\alpha}}, b\right) &= \frac{1}{2} \begin{bmatrix} \underline{\boldsymbol{\alpha}} \\ b \end{bmatrix}^T \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \underline{\boldsymbol{\alpha}} \\ b \end{bmatrix} - \begin{bmatrix} \underline{\boldsymbol{\alpha}} \\ b \end{bmatrix}^T \begin{bmatrix} \underline{\mathbf{z}} - \operatorname{sgn}\left(\underline{\boldsymbol{\tau}}\right) \underline{\boldsymbol{\rho}}_{\boldsymbol{\mathcal{T}}}^{(*)} \\ 0 \end{bmatrix} && (7.16) \\ Q_{\text{Fix}}\left(\underline{\boldsymbol{\alpha}}, b\right) &= \frac{1}{2} \underline{\boldsymbol{\alpha}}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}} \underline{\boldsymbol{\alpha}} + \underline{\boldsymbol{\alpha}}^T \mathbf{1} b - \underline{\boldsymbol{\alpha}}^T \left(\underline{\mathbf{z}} - \operatorname{sgn}\left(\underline{\boldsymbol{\tau}}\right) \underline{\boldsymbol{\rho}}_{\boldsymbol{\mathcal{T}}}^{(*)} \right) && (7.17) \end{aligned}$$

7.3.1 Standard Form

For notational convenience, unless otherwise stated I will assume that $\mathbf{m}$ is chosen such that:

$$\underline{\boldsymbol{\tau}} = \begin{bmatrix} \mathbf{0} \\ -\mathbf{2} \\ +\mathbf{2} \\ \mathbf{1}_{\pm} \end{bmatrix} \tag{7.18}$$

This is *standard form.* By definition, in (7.18), $\mathbf{0} \in \Re^{N_Z}$, $-\mathbf{2} \in \Re^{N_L}$, $+\mathbf{2} \in \Re^{N_U}$ and $\mathbf{1}_{\pm} \in \Re^{N_F}$. Hence $N = N_Z + N_L + N_U + N_F$. Finally, define N_{F-} is the number of elements $\underline{\tau}_i = -1$, and N_{F+} the number for which $\underline{\tau}_i = +1$. So $N_F = N_{F+} + N_{F-}$.

Variables $\underline{\alpha}_i$ for which $\underline{\tau}_i \in \{-2, 0, +2\}$ are called *actively constrained* variables, and the corresponding inequality constraints which hold as equality constraints (i.e. $\underline{\alpha}_i = \underline{v}_i$, $\underline{\alpha}_i = 0$ or $\underline{\alpha}_i = \underline{h}_i$) are called *active constraints*, and are treated as equality constraints. Constraints that are not active are called *inactive* constraints. The total number of actively constrained variables is $N_C = N_Z + N_L + N_U$.

All other variables (i.e. variables $\underline{\alpha}_i$ for which $\underline{\tau}_i = \pm 1$) are called *free* variables. The number of free variables is $N_F = N_{F+} + N_{F-}$, where N_{F+} is the number of free positive variables ($\underline{\tau}_i = +1$), and N_{F-} is the number of free negative variables ($\underline{\tau}_i = -1$). All of the constraints relating to the free variables are inactive.

Note that the free variables correspond to boundary vectors, and the constrained variables for which $\underline{\tau}_i = \pm 2$ to error vectors. So there are $N_B = N_F$ boundary vectors, $N_E = N_L + N_U$ error vectors and $N_S = N_L + N_U + N_F$ support vectors in total.

When in standard form, all vectors (for example, $\underline{\mathbf{z}}$) can be split thusly:

$$\underline{\mathbf{z}} = \begin{bmatrix} \underline{\mathbf{z}}_Z \\ \underline{\mathbf{z}}_L \\ \underline{\mathbf{z}}_U \\ \underline{\mathbf{z}}_F \end{bmatrix}, \text{ etc.}$$

where $\underline{\mathbf{z}}_Z \in \Re^{N_Z}$, $\underline{\mathbf{z}}_L \in \Re^{N_L}$, $\underline{\mathbf{z}}_U \in \Re^{N_U}$, and $\underline{\mathbf{z}}_F \in \Re^{N_F}$ correspond consecutively to those variables actively constrained at (Z)ero, those actively constrained at the (L)ower bound $\underline{\mathbf{v}}$, those actively constrained at an (U)pper bound $\underline{\mathbf{h}}$ and the (F)ree variables. Hence:

$$\begin{aligned} \underline{\boldsymbol{\tau}}_Z &= \mathbf{0} & \underline{\boldsymbol{\alpha}}_Z &= \mathbf{0} \\ \underline{\boldsymbol{\tau}}_L &= -\mathbf{2} & \underline{\boldsymbol{\alpha}}_L &= \underline{\mathbf{v}}_L \\ \underline{\boldsymbol{\tau}}_U &= +\mathbf{2} & \underline{\boldsymbol{\alpha}}_U &= \underline{\mathbf{h}}_U \\ \underline{\boldsymbol{\tau}}_F &= \mathbf{1}_{\pm} & \underline{\boldsymbol{\alpha}}_F &\in [\underline{\mathbf{v}}_F, \underline{\mathbf{h}}_F] \end{aligned}$$

Likewise, symmetric matrices (which includes all matrices of interest here, for example, $\underline{\mathbf{K}}$) may be split thusly:

$$\underline{\mathbf{K}} = \begin{bmatrix} \underline{\mathbf{K}}_Z & \underline{\mathbf{K}}_{ZL} & \underline{\mathbf{K}}_{ZU} & \underline{\mathbf{K}}_{ZF} \\ \underline{\mathbf{K}}_{ZL}^T & \underline{\mathbf{K}}_L & \underline{\mathbf{K}}_{LU} & \underline{\mathbf{K}}_{LF} \\ \underline{\mathbf{K}}_{ZU}^T & \underline{\mathbf{K}}_{LU}^T & \underline{\mathbf{K}}_U & \underline{\mathbf{K}}_{UF} \\ \underline{\mathbf{K}}_{ZF}^T & \underline{\mathbf{K}}_{LF}^T & \underline{\mathbf{K}}_{UF}^T & \underline{\mathbf{K}}_F \end{bmatrix}$$

where $\underline{\mathbf{K}}_Z \in \Re^{N_Z \times N_Z}$, $\underline{\mathbf{K}}_{ZL} \in \Re^{N_L \times N_Z}$ etc. Using this notation:

$$\underline{\mathbf{G}}_{\boldsymbol{\tau}} = \left[\begin{array}{cccc} \underline{\mathbf{K}}_Z & \underline{\mathbf{K}}_{ZL} & \underline{\mathbf{K}}_{ZU} & \dots \\ \underline{\mathbf{K}}_{ZL}^T & \underline{\mathbf{K}}_L + \boldsymbol{\mu}_L^* \boldsymbol{\mu}_L^{*T} + \operatorname{diag}(\boldsymbol{\gamma}_L^*) & \underline{\mathbf{K}}_{LU} - \boldsymbol{\mu}_L^* \boldsymbol{\mu}_U^T & \dots \\ \underline{\mathbf{K}}_{ZU}^T & \underline{\mathbf{K}}_{LU}^T - \boldsymbol{\mu}_U \boldsymbol{\mu}_L^{*T} & \underline{\mathbf{K}}_U + \boldsymbol{\mu}_U \boldsymbol{\mu}_U^T + \operatorname{diag}(\boldsymbol{\gamma}_U) & \dots \\ \underline{\mathbf{K}}_{ZF}^T & \underline{\mathbf{K}}_{LF}^T - \left(\operatorname{sgn}(\boldsymbol{\tau}_F) \boldsymbol{\mu}_{\boldsymbol{\tau} F}^{(*)} \right) \boldsymbol{\mu}_L^{*T} & \underline{\mathbf{K}}_{UF}^T + \left(\operatorname{sgn}(\boldsymbol{\tau}_F) \boldsymbol{\mu}_{\boldsymbol{\tau} F}^{(*)} \right) \boldsymbol{\mu}_U^T & \dots \end{array} \right.$$
$$\left. \begin{array}{cc} \dots & \underline{\mathbf{K}}_{ZF} \\ \dots & \underline{\mathbf{K}}_{LF} - \boldsymbol{\mu}_L^* \left(\operatorname{sgn}(\boldsymbol{\tau}_F) \boldsymbol{\mu}_{\boldsymbol{\tau} F}^{(*)} \right)^T \\ \dots & \underline{\mathbf{K}}_{UF} + \boldsymbol{\mu}_U \left(\operatorname{sgn}(\boldsymbol{\tau}_F) \boldsymbol{\mu}_{\boldsymbol{\tau} F}^{(*)} \right)^T \\ \dots & \underline{\mathbf{K}}_F + \left(\operatorname{sgn}(\boldsymbol{\tau}_F) \boldsymbol{\mu}_{\boldsymbol{\tau} F}^{(*)} \right) \left(\operatorname{sgn}(\boldsymbol{\tau}_F) \boldsymbol{\mu}_{\boldsymbol{\tau} F}^{(*)} \right)^T + \operatorname{diag}\left(\boldsymbol{\gamma}_{\boldsymbol{\tau} F}^{(*)} \right) \end{array} \right]$$

Note that:

$$g(\mathbf{y}) = \sum_{i=N_Z+1}^{N} \underline{\alpha}_i K(\underline{\mathbf{x}}_i, \mathbf{y}) + b$$

The active set approach [40] to solving (7.14) is an iterative method whereby one approaches the solution by testing a sequence of possible active sets for optimality. For each iteration, an active set is proposed (and $\boldsymbol{\tau}$ set appropriately and put in standard form via pivoting of $\mathbf{m}$) and, treating the constrained variables as constants, the optimal value for the free variables is found. This proposed solution is tested for optimality (for both free and constrained variables), and if it is non-optimal then the active set is modified ($\boldsymbol{\tau}$ is changed) and the process is repeated. The key advantage of this approach is that typically when dealing with SVMs $N_F \ll N$. Hence the optimisation problem to be solved for each iteration of the optimisation procedure should be small and therefore quickly solvable.

7.4 Optimality Conditions

Consider the abstract optimisation problem (7.14) in standard form.

$$\begin{gathered}\{\underline{\boldsymbol{\alpha}}, b\} = \arg\min_{\underline{\boldsymbol{\alpha}} \in [\underline{\mathbf{v}}, \underline{\mathbf{h}}]} \arg\max_{b \in \Re} Q_{\text{BiasType}} \left(\underline{\boldsymbol{\alpha}}, b\right) \\ \text{such that: } b = b_{\text{fix}} \text{ if BiasType} = \text{Fix}\end{gathered}$$

where:

$$\begin{aligned} Q_{\text{Var}}\left(\underline{\boldsymbol{\alpha}}, b\right) &= \frac{1}{2} \begin{bmatrix} \underline{\boldsymbol{\alpha}} \\ b \end{bmatrix}^T \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\tau}} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \underline{\boldsymbol{\alpha}} \\ b \end{bmatrix} - \begin{bmatrix} \underline{\boldsymbol{\alpha}} \\ b \end{bmatrix}^T \begin{bmatrix} \underline{\mathbf{z}} - \operatorname{sgn}\left(\underline{\boldsymbol{\tau}}\right) \underline{\boldsymbol{\rho}}_{\boldsymbol{\tau}}^{(*)} \\ 0 \end{bmatrix} \\ Q_{\text{Fix}}\left(\underline{\boldsymbol{\alpha}}, b\right) &= \frac{1}{2} \underline{\boldsymbol{\alpha}}^T \underline{\mathbf{G}}_{\boldsymbol{\tau}} \underline{\boldsymbol{\alpha}} + \underline{\boldsymbol{\alpha}}^T \mathbf{1} b - \underline{\boldsymbol{\alpha}}^T \left(\underline{\mathbf{z}} - \operatorname{sgn}\left(\underline{\boldsymbol{\tau}}\right) \underline{\boldsymbol{\rho}}_{\boldsymbol{\tau}}^{(*)}\right) \end{aligned}$$

Before deriving the optimality conditions it is useful to define:

$$E = \underline{\boldsymbol{\mu}}_{\boldsymbol{\tau}}^{(*)T} \left|\underline{\boldsymbol{\alpha}}\right| \tag{7.19}$$

$$\begin{bmatrix} \underline{\mathbf{k}} \\ f \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{K}} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \underline{\boldsymbol{\alpha}} \\ b \end{bmatrix} \tag{7.20}$$

$$\begin{bmatrix} \underline{\mathbf{e}}_{\boldsymbol{\tau}} \\ f \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\tau}} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} \begin{bmatrix} \underline{\boldsymbol{\alpha}} \\ b \end{bmatrix} - \begin{bmatrix} \underline{\mathbf{z}} - \operatorname{sgn}\left(\underline{\boldsymbol{\tau}}\right) \underline{\boldsymbol{\rho}}_{\boldsymbol{\tau}}^{(*)} \\ 0 \end{bmatrix} \tag{7.21}$$

$$= \begin{bmatrix} \underline{\mathbf{k}} \\ f \end{bmatrix} - \begin{bmatrix} \underline{\mathbf{z}} - \operatorname{sgn}\left(\underline{\boldsymbol{\tau}}\right) \left(\underline{\boldsymbol{\rho}}_{\boldsymbol{\tau}}^{(*)} + E\underline{\boldsymbol{\mu}}_{\boldsymbol{\tau}}^{(*)}\right) - \underline{\boldsymbol{\gamma}}_{\boldsymbol{\tau}}^{(*)}\underline{\boldsymbol{\alpha}} \\ 0 \end{bmatrix} \tag{7.22}$$

$$\begin{bmatrix} \underline{\mathbf{e}}_{\boldsymbol{\tau}Z} \\ \underline{\mathbf{e}}_{\boldsymbol{\tau}L} \\ \underline{\mathbf{e}}_{\boldsymbol{\tau}U} \\ \underline{\mathbf{e}}_{\boldsymbol{\tau}F} \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{k}}_Z \\ \underline{\mathbf{k}}_L \\ \underline{\mathbf{k}}_U \\ \underline{\mathbf{k}}_F \end{bmatrix} - \begin{bmatrix} \underline{\mathbf{z}}_Z \\ \underline{\mathbf{z}}_L \\ \underline{\mathbf{z}}_U \\ \underline{\mathbf{z}}_F \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \underline{\boldsymbol{\gamma}}_L^{*}\underline{\mathbf{v}}_L \\ \underline{\boldsymbol{\gamma}}_U\underline{\mathbf{h}}_U \\ \underline{\boldsymbol{\gamma}}_{\boldsymbol{\tau}F}^{(*)}\underline{\boldsymbol{\alpha}}_F \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ -\left(\underline{\boldsymbol{\rho}}_L^{*} + E\underline{\boldsymbol{\mu}}_L^{*}\right) \\ \left(\underline{\boldsymbol{\rho}}_U + E\underline{\boldsymbol{\mu}}_U\right) \\ \underline{\boldsymbol{\tau}}_F\left(\underline{\boldsymbol{\rho}}_{\boldsymbol{\tau}F}^{(*)} + E\underline{\boldsymbol{\mu}}_{\boldsymbol{\tau}F}^{(*)}\right) \end{bmatrix} \tag{7.23}$$

noting that $\underline{k}_i = g\left(\underline{\mathbf{x}}_i\right)$.

Returning to the longform version of the abstract optimisation problem (7.2), it is easy to see that the optimality conditions for $\boldsymbol{\beta}$, $\boldsymbol{\beta}^*$ and b are simply:

$$\frac{\partial Q_{\mathrm{BiasType}}}{\partial \underline{\beta}_i} \begin{cases} \geq 0 & \text{if } \underline{\beta}_i = 0 \wedge h_i > 0 \\ = 0 & \text{if } 0 < \underline{\beta}_i < h_i \wedge h_i > 0 \\ \leq 0 & \text{if } \underline{\beta}_i = h_i \wedge h_i > 0 \end{cases}$$

$$\frac{\partial Q_{\mathrm{BiasType}}}{\partial \underline{\beta}_i^*} \begin{cases} \geq 0 & \text{if } \underline{\beta}_i^* = v_i \wedge v_i < 0 \\ = 0 & \text{if } v_i < \underline{\beta}_i^* < 0 \wedge v_i < 0 \\ \leq 0 & \text{if } \underline{\beta}_i^* = 0 \wedge v_i < 0 \end{cases}$$

$$\frac{\partial Q_{\mathrm{BiasType}}}{\partial b} = 0 \text{ if BiasType} = \text{Var}$$

where:

$$\begin{aligned}
\frac{\partial Q_{\text{BiasType}}}{\partial \underline{\boldsymbol{\beta}}} &= \underline{\mathbf{k}} - \underline{\mathbf{z}} + E\underline{\boldsymbol{\mu}} + \underline{\boldsymbol{\gamma}\boldsymbol{\beta}} + \underline{\boldsymbol{\rho}} \\
\frac{\partial Q_{\text{BiasType}}}{\partial \underline{\boldsymbol{\beta}}^*} &= \underline{\mathbf{k}} - \underline{\mathbf{z}} - E\underline{\boldsymbol{\mu}}^* + \underline{\boldsymbol{\gamma}}^* \underline{\boldsymbol{\beta}}^* - \underline{\boldsymbol{\rho}}^* \\
\frac{\partial Q_{\text{Var}}}{\partial b} &= f
\end{aligned}$$

Now, note that:

$$\begin{aligned}
\frac{\partial Q_{\text{BiasType}}}{\partial \underline{\beta}_i} &= \underline{e}_{\boldsymbol{\tau} i} \text{ if } \underline{\tau}_i > 0 \\
\frac{\partial Q_{\text{BiasType}}}{\partial \underline{\beta}_i^*} &= \underline{e}_{\boldsymbol{\tau} i} \text{ if } \underline{\tau}_i < 0
\end{aligned}$$

Also, as $\underline{\rho}_i, \underline{\rho}_i^* \geq 0$ when $\underline{v}_i < 0 < \underline{h}_i$, it follows that:

$$\frac{\partial Q_{\text{BiasType}}}{\partial \underline{\beta}_i^*} \leq \frac{\partial Q_{\text{BiasType}}}{\partial \underline{\beta}_i} \;\forall i : \underline{v}_i < 0 < \underline{h}_i$$

Using these results, it is not too difficult to see that in terms of $\underline{\mathbf{e}}_{\boldsymbol{\tau}}$ and f, the optimality conditions (KKT conditions) for the abstract optimisation problem (7.14) are just:

$$\begin{aligned}
\underline{e}_{\boldsymbol{\tau} Zi} &\in \begin{cases} \left[-\left(\underline{\rho}_{Zi} + E\underline{\mu}_{Zi} \right), \left(\underline{\rho}_{Zi}^* + E\underline{\mu}_{Zi}^* \right) \right] & \text{if } \underline{v}_i < 0 < \underline{h}_i \\ \left[-\left(\underline{\rho}_{Zi} + E\underline{\mu}_{Zi} \right), \infty \right) & \text{if } \underline{v}_i = 0 < \underline{h}_i \\ \left(-\infty, \left(\underline{\rho}_{Zi}^* + E\underline{\mu}_{Zi}^* \right) \right] & \text{if } \underline{v}_i < 0 = \underline{h}_i \\ (-\infty, \infty) & \text{if } \underline{v}_i = 0 = \underline{h}_i \end{cases} \\
\underline{\mathbf{e}}_{\boldsymbol{\tau} L} &\geq \mathbf{0} \\
\underline{\mathbf{e}}_{\boldsymbol{\tau} U} &\leq \mathbf{0} \\
\underline{\mathbf{e}}_{\boldsymbol{\tau} F} &= \mathbf{0} \\
f &= 0 \text{ if BiasType} = \text{Var} \\
\underline{\boldsymbol{\alpha}} &\in [\underline{\mathbf{v}}, \underline{\mathbf{h}}]
\end{aligned} \tag{7.24}$$

It is this form (7.24) which will be used most in this thesis. The KKT conditions may be expressed in terms of $\underline{\mathbf{k}}$ thusly:

$$\begin{aligned}
\underline{k}_{Zi} &\in \begin{cases} \left[\underline{z}_{Zi} - \left(\underline{\rho}_{Zi} + E\underline{\mu}_{Zi} \right), \underline{z}_{Zi} + \left(\underline{\rho}^*_{Zi} + E\underline{\mu}^*_{Zi} \right) \right] & \text{if } \underline{v}_i < 0 < \underline{h}_i \\ \left[\underline{z}_{Zi} - \left(\underline{\rho}_{Zi} + E\underline{\mu}_{Zi} \right), \infty \right) & \text{if } \underline{v}_i = 0 < \underline{h}_i \\ \left(-\infty, \underline{z}_{Zi} + \left(\underline{\rho}^*_{Zi} + E\underline{\mu}^*_{Zi} \right) \right] & \text{if } \underline{v}_i < 0 = \underline{h}_i \\ \left(-\infty, \infty \right) & \text{if } \underline{v}_i = 0 = \underline{h}_i \end{cases} \\
\underline{\mathbf{k}}_L &\geq \underline{\mathbf{z}}_L - \underline{\boldsymbol{\gamma}}^* \underline{\mathbf{v}}_L + \left(\underline{\boldsymbol{\rho}}^*_L + E\underline{\boldsymbol{\mu}}^*_L \right) \\
\underline{\mathbf{k}}_U &\leq \underline{\mathbf{z}}_U - \underline{\boldsymbol{\gamma}} \underline{\mathbf{h}}_U - \left(\underline{\boldsymbol{\rho}}_U + E\underline{\boldsymbol{\mu}}_U \right) \\
\underline{\mathbf{k}}_F &= \underline{\mathbf{z}}_F - \underline{\boldsymbol{\gamma}}^{(*)}_{\boldsymbol{\tau}} \underline{\boldsymbol{\alpha}}_F - \underline{\boldsymbol{\tau}}_F \left(\underline{\boldsymbol{\rho}}_{\boldsymbol{\tau}}{}^{(*)}_{F} + E\underline{\boldsymbol{\mu}}_{\boldsymbol{\tau}}{}^{(*)}_{F} \right) \\
f &= 0 \text{ if BiasType} = \text{Var} \\
\underline{\boldsymbol{\alpha}} &\in \left[\underline{\mathbf{v}}, \underline{\mathbf{h}} \right]
\end{aligned}$$

Note that this is really just a statement about the range in which the output of the system $g(\mathbf{y})$ must lie for the various elements of the training set. In particular, for the non-support vectors it states that the output must lie within some margin $\pm \left(\underline{\rho}_i^{(*)} + E\underline{\mu}_i^{(*)} \right)$ of the target value, z_i. Likewise, for boundary vectors, the output must lie at the edge of this region with some margin for variation if quadratic cost is used (i.e. $\underline{\gamma}_i^{(*)}$ is non-zero). Only the error vectors may lie outside this region.

7.5 Abstract Solution

For completeness, given an abstract training set $\mathbf{T}$ as in (7.1), the abstract solution of the optimisation problem (7.14) is defined to be:

$$\maltese = (\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \mathbf{m})$$

7.6 Connection with Standard SV Classifiers

To use the abstract problem formulation given above for binary pattern classification, start with a training set of the standard form (4.2):

$$\begin{aligned}
&\mathbf{Y} = \{(\mathbf{x}_1, d_1), (\mathbf{x}_2, d_2), \ldots, (\mathbf{x}_N, d_N)\} \\
&\mathbf{x}_i \in \Re^{d_L} \\
&d_i \in \{+1, -1\}
\end{aligned}$$

as well as the additional (implicitly defined) data $C > 0$, $E \geq 0$, $t_i > 0$, $t_i^* > 0$, $\epsilon_i < 0$ and $\epsilon_i^* < 0$ $(1 \leq i \leq N)$. Then the abstract training set (7.1) is constructed using the recipe appropriate to the formulation under consideration.

In all cases, $\mathcal{H}$ is assumed to be a very large number. Ideally, $\mathcal{H} = \infty$. In reality, however, $\mathcal{H}$ represents the largest value which the computer solving the generalised optimisation problem can deal with appropriately.

It is convenient to define $\tilde{C} = \frac{C}{N}$.

Standard SVM - Linear Risk, Variable Bias

$$\begin{aligned}
&\mathbf{T} = (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
&\boldsymbol{\vartheta} = (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
&\vartheta_i = \begin{cases} \left(\mathbf{x}_i, 0, E\epsilon_i, 0, \tilde{C}t_i, 0, 0, 0, 0, 0\right) & \text{if } d_i = +1 \\ \left(\mathbf{x}_i, 0, 0, E\epsilon_i^*, 0, -\tilde{C}t_i^*, 0, 0, 0, 0\right) & \text{if } d_i = -1 \end{cases}
\end{aligned} \tag{7.25}$$

Linear Risk, Fixed Bias

$$\begin{aligned}
&\mathbf{T} = (\boldsymbol{\vartheta}, \mathrm{Fix}, b_{\mathrm{fix}}) \\
&\boldsymbol{\vartheta} = (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
&\vartheta_i = \begin{cases} \left(\mathbf{x}_i, 0, E\epsilon_i, 0, \tilde{C}t_i, 0, 0, 0, 0, 0\right) & \text{if } d_i = +1 \\ \left(\mathbf{x}_i, 0, 0, E\epsilon_i^*, 0, -\tilde{C}t_i^*, 0, 0, 0, 0\right) & \text{if } d_i = -1 \end{cases}
\end{aligned} \tag{7.26}$$

Quadratic Risk, Variable Bias

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, 0, E\epsilon_i, 0, \mathcal{H}, 0, \frac{1}{\tilde{C}t_i}, 0, 0, 0\right) & \text{if } d_i = +1 \\ \left(\mathbf{x}_i, 0, 0, E\epsilon_i^*, 0, -\mathcal{H}, 0, \frac{1}{\tilde{C}t_i^*}, 0, 0\right) & \text{if } d_i = -1 \end{cases}
\end{aligned} \tag{7.27}$$

Quadratic Risk, Fixed Bias

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Fix}, b_{\mathrm{fix}}) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, 0, E\epsilon_i, 0, \mathcal{H}, 0, \frac{1}{\tilde{C}t_i}, 0, 0, 0\right) & \text{if } d_i = +1 \\ \left(\mathbf{x}_i, 0, 0, E\epsilon_i^*, 0, -\mathcal{H}, 0, \frac{1}{\tilde{C}t_i^*}, 0, 0\right) & \text{if } d_i = -1 \end{cases}
\end{aligned} \tag{7.28}$$

7.7 Connection with Standard SV Regressors

To use the abstract problem formulation given above for nonlinear regression, start with a training set of the standard form (4.17):

$$\begin{aligned}
&\mathbf{Y}_= = \{(\mathbf{x}_1, z_1), (\mathbf{x}_2, z_2), \ldots, (\mathbf{x}_N, z_N)\} \\
&\mathbf{x}_i \in \Re^{d_L} \\
&z_i \in \Re
\end{aligned}$$

as well as the additional (implicitly defined) data $C > 0$ (and possible $\nu > 0$ for tube shrinking), $E \geq 0$, $t_i > 0$, $t_i^* > 0$, $\epsilon_i > 0$ and $\epsilon_i^* > 0$ $(1 \leq i \leq N)$. Then the abstract training set (7.1) is constructed using the recipe appropriate to the formulation under consideration.

Once again, $\mathcal{H}$ is assumed to be a very large number, and $\tilde{C} = \frac{C}{N}$. Also, in this case, $\tilde{\nu} = N\nu$

Standard SVR - Linear Risk, Variable Bias, No Tube Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \left(\mathbf{x}_i, z_i, E\epsilon_i, E\epsilon_i^*, \tilde{C}t_i, -\tilde{C}t_i^*, 0, 0, 0, 0\right)
\end{aligned} \tag{7.29}$$

Linear Risk, Fixed Bias, No Tube Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Fix}, b_{\mathrm{fix}}) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \left(\mathbf{x}_i, z_i, E\epsilon_i, E\epsilon_i^*, \tilde{C}t_i, -\tilde{C}t_i^*, 0, 0, 0, 0\right)
\end{aligned} \tag{7.30}$$

Quadratic Risk, Variable Bias, No Tube Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \left(\mathbf{x}_i, z_i, E\epsilon_i, E\epsilon_i^*, \mathcal{H}, -\mathcal{H}, \tfrac{1}{\tilde{C}t_i}, \tfrac{1}{\tilde{C}t_i^*}, 0, 0\right)
\end{aligned} \tag{7.31}$$

Quadratic Risk, Fixed Bias, No Tube Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Fix}, b_{\mathrm{fix}}) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \left(\mathbf{x}_i, z_i, E\epsilon_i, E\epsilon_i^*, \mathcal{H}, -\mathcal{H}, \tfrac{1}{\tilde{C}t_i}, \tfrac{1}{\tilde{C}t_i^*}, 0, 0\right)
\end{aligned} \tag{7.32}$$

Linear Risk, Variable Bias, Quadratic Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \left(\mathbf{x}_i, z_i, 0, 0, \tilde{C}t_i, -\tilde{C}t_i^*, 0, 0, \tfrac{\epsilon_i}{\tilde{C}\tilde{\nu}}, \tfrac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right)
\end{aligned} \tag{7.33}$$

Linear Risk, Fixed Bias, Quadratic Shrinking

$$\begin{aligned} \mathbf{T} &= (\boldsymbol{\vartheta}, \text{Fix}, b_{\text{fix}}) \\ \boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\ \vartheta_i &= \left(\mathbf{x}_i, z_i, 0, 0, \tilde{C} t_i, -\tilde{C} t_i^*, 0, 0, \tfrac{\epsilon_i}{\tilde{C}\tilde{\nu}}, \tfrac{\epsilon_i^*}{\tilde{C}\tilde{\nu}} \right) \end{aligned} \tag{7.34}$$

Quadratic Risk, Variable Bias, Quadratic Shrinking

$$\begin{aligned} \mathbf{T} &= (\boldsymbol{\vartheta}, \text{Var}, 0) \\ \boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\ \vartheta_i &= \left(\mathbf{x}_i, z_i, 0, 0, \mathcal{H}, -\mathcal{H}, \tfrac{1}{\tilde{C} t_i}, \tfrac{1}{\tilde{C} t_i^*}, \tfrac{\epsilon_i}{\tilde{C}\tilde{\nu}}, \tfrac{\epsilon_i^*}{\tilde{C}\tilde{\nu}} \right) \end{aligned} \tag{7.35}$$

Quadratic Risk, Fixed Bias, Quadratic Shrinking

$$\begin{aligned} \mathbf{T} &= (\boldsymbol{\vartheta}, \text{Fix}, b_{\text{fix}}) \\ \boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\ \vartheta_i &= \left(\mathbf{x}_i, z_i, 0, 0, \mathcal{H}, -\mathcal{H}, \tfrac{1}{\tilde{C} t_i}, \tfrac{1}{\tilde{C} t_i^*}, \tfrac{\epsilon_i}{\tilde{C}\tilde{\nu}}, \tfrac{\epsilon_i^*}{\tilde{C}\tilde{\nu}} \right) \end{aligned} \tag{7.36}$$

7.8 Connection with SV Regression With Inequalities

To use the abstract problem formulation given above nonlinear regression with inequalities, start with a training set of the standard form (5.1):

$$\begin{aligned} &\mathbf{Y} = \{(\mathbf{x}_1, z_1), (\mathbf{x}_2, z_2), \ldots, (\mathbf{x}_N, z_N)\} \\ &\mathbf{Y} = \mathbf{Y}_= \cup \mathbf{Y}_\geq \cup \mathbf{Y}_\leq \\ &\mathbf{Y}_\geq \cap \mathbf{Y}_\leq = \mathbf{Y}_= \cap \mathbf{Y}_\leq = \mathbf{Y}_= \cap \mathbf{Y}_\geq = \emptyset \\ &\mathbf{x}_i \in \Re^{d_L} \\ &z_i \in \Re \end{aligned}$$

as well as the additional (implicitly defined) data $C > 0$ (and possible $\nu > 0$ for tube shrinking), $E \geq 0$, $t_i > 0$, $t_i^* > 0$, ϵ_i and ϵ_i^* $(1 \leq i \leq N)$. Then the abstract training set (7.1) is constructed using the recipe appropriate to the formulation under consideration.

$\mathcal{H}$, $\tilde{C}$ and $\tilde{\nu}$ are defined as before.

Standard SVR - Linear Risk, Variable Bias, No Tube Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, z_i, E\epsilon_i, E\epsilon_i^*, \tilde{C}t_i, -\tilde{C}t_i^*, 0, 0, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_= \\ \left(\mathbf{x}_i, z_i, E\epsilon_i, 0, \tilde{C}t_i, 0, 0, 0, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_\geq \\ \left(\mathbf{x}_i, z_i, 0, E\epsilon_i^*, 0, -\tilde{C}t_i^*, 0, 0, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_\geq \end{cases}
\end{aligned} \tag{7.37}$$

Linear Risk, Fixed Bias, No Tube Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Fix}, b_{\mathrm{fix}}) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, z_i, E\epsilon_i, E\epsilon_i^*, \tilde{C}t_i, -\tilde{C}t_i^*, 0, 0, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_= \\ \left(\mathbf{x}_i, z_i, E\epsilon_i, 0, \tilde{C}t_i, 0, 0, 0, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_\geq \\ \left(\mathbf{x}_i, z_i, 0, E\epsilon_i^*, 0, -\tilde{C}t_i^*, 0, 0, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_\geq \end{cases}
\end{aligned} \tag{7.38}$$

Quadratic Risk, Variable Bias, No Tube Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, z_i, E\epsilon_i, E\epsilon_i^*, \mathcal{H}, -\mathcal{H}, \frac{1}{\tilde{C}t_i}, \frac{1}{\tilde{C}t_i^*}, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_= \\ \left(\mathbf{x}_i, z_i, E\epsilon_i, 0, \mathcal{H}, 0, \frac{1}{\tilde{C}t_i}, 0, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_\geq \\ \left(\mathbf{x}_i, z_i, 0, E\epsilon_i^*, 0, -\mathcal{H}, 0, \frac{1}{\tilde{C}t_i^*}, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_\geq \end{cases}
\end{aligned} \tag{7.39}$$

Quadratic Risk, Fixed Bias, No Tube Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Fix}, b_{\mathrm{fix}}) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, z_i, E\epsilon_i, E\epsilon_i^*, \mathcal{H}, -\mathcal{H}, \frac{1}{\tilde{C}t_i}, \frac{1}{\tilde{C}t_i^*}, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_= \\ \left(\mathbf{x}_i, z_i, E\epsilon_i, 0, \mathcal{H}, 0, \frac{1}{\tilde{C}t_i}, 0, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_\geq \\ \left(\mathbf{x}_i, z_i, 0, E\epsilon_i^*, 0, -\mathcal{H}, 0, \frac{1}{\tilde{C}t_i^*}, 0, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_\geq \end{cases}
\end{aligned} \tag{7.40}$$

Linear Risk, Variable Bias, Quadratic Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, z_i, 0, 0, \tilde{C}t_i, -\tilde{C}t_i^*, 0, 0, \frac{\epsilon_i}{\tilde{C}\tilde{\nu}}, \frac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_= \\ \left(\mathbf{x}_i, z_i, 0, 0, \tilde{C}t_i, 0, 0, 0, \frac{\epsilon_i}{\tilde{C}\tilde{\nu}}, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_\geq \\ \left(\mathbf{x}_i, z_i, 0, 0, 0, -\tilde{C}t_i^*, 0, 0, 0, \frac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_\geq \end{cases}
\end{aligned} \tag{7.41}$$

Linear Risk, Fixed Bias, Quadratic Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Fix}, b_{\mathrm{fix}}) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, z_i, 0, 0, \tilde{C}t_i, -\tilde{C}t_i^*, 0, 0, \frac{\epsilon_i}{\tilde{C}\tilde{\nu}}, \frac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_= \\ \left(\mathbf{x}_i, z_i, 0, 0, \tilde{C}t_i, 0, 0, 0, \frac{\epsilon_i}{\tilde{C}\tilde{\nu}}, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_\geq \\ \left(\mathbf{x}_i, z_i, 0, 0, 0, -\tilde{C}t_i^*, 0, 0, 0, \frac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_\geq \end{cases}
\end{aligned} \tag{7.42}$$

Quadratic Risk, Variable Bias, Quadratic Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Var}, 0) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, z_i, 0, 0, \mathcal{H}, -\mathcal{H}, \frac{1}{\tilde{C}t_i}, \frac{1}{\tilde{C}t_i^*}, \frac{\epsilon_i}{\tilde{C}\tilde{\nu}}, \frac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_= \\ \left(\mathbf{x}_i, z_i, 0, 0, \mathcal{H}, 0, \frac{1}{\tilde{C}t_i}, 0, \frac{\epsilon_i}{\tilde{C}\tilde{\nu}}, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_\geq \\ \left(\mathbf{x}_i, z_i, 0, 0, 0, -\mathcal{H}, 0, \frac{1}{\tilde{C}t_i^*}, 0, \frac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z_i}) \in \mathbf{Y}_\geq \end{cases}
\end{aligned} \tag{7.43}$$

Quadratic Risk, Fixed Bias, Quadratic Shrinking

$$\begin{aligned}
\mathbf{T} &= (\boldsymbol{\vartheta}, \mathrm{Fix}, b_{\mathrm{fix}}) \\
\boldsymbol{\vartheta} &= (\vartheta_1, \vartheta_2, \ldots, \vartheta_N) \\
\vartheta_i &= \begin{cases} \left(\mathbf{x}_i, z_i, 0, 0, \mathcal{H}, -\mathcal{H}, \frac{1}{\tilde{C}t_i}, \frac{1}{\tilde{C}t_i^*}, \frac{\epsilon_i}{\tilde{C}\tilde{\nu}}, \frac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_= \\ \left(\mathbf{x}_i, z_i, 0, 0, \mathcal{H}, 0, \frac{1}{\tilde{C}t_i}, 0, \frac{\epsilon_i}{\tilde{C}\tilde{\nu}}, 0\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_\geq \\ \left(\mathbf{x}_i, z_i, 0, 0, 0, -\mathcal{H}, 0, \frac{1}{\tilde{C}t_i^*}, 0, \frac{\epsilon_i^*}{\tilde{C}\tilde{\nu}}\right) & \text{if } (\mathbf{x}_\mathrm{i}, \mathrm{z}_\mathrm{i}) \in \mathbf{Y}_\geq \end{cases}
\end{aligned} \tag{7.44}$$

7.9 Virtual Bounds

As noted in theorem 7.1 that at least one of β_i and β_i^* must be zero for all $1 \leq i \leq N$. Using this, the type vector $\boldsymbol{\tau}$ can be interpreted in terms of the values of the elements of $\boldsymbol{\beta}$ and $\boldsymbol{\beta}^*$ thusly:

- If $\tau_i = -2$ then $\beta_i = 0$ and $\beta_i^* = v_i$.
- If $\tau_i = -1$ then $\beta_i = 0$ and $v_i \leq \beta_i^* \leq 0$.
- If $\tau_i = 0$ then $\beta_i = 0$ and $\beta_i^* = 0$.
- If $\tau_i = +1$ then $0 \leq \beta_i \leq h_i$ and $\beta_i^* = 0$.
- If $\tau_i = +2$ then $\beta_i = h_i$ and $\beta_i^* = 0$.

So, if α_i is viewed as shorthand for "whichever one of β_i and β_i^* is non-zero", then the type τ_i is the key that defines explicitly which one it is (unless $\tau_i = 0$, in which case the answer is neither, both are zero and we have no preference either way).

As will be discussed later, the process of finding the optimal solution to the abstract optimisation problem (7.14) is an iterative one. As the optimisation process proceeds the value of the type vector $\boldsymbol{\tau}$ (and hence meaning of $\boldsymbol{\alpha}$ and the values of elements with subscript $\boldsymbol{\tau}$) will change. Because of this, some care is required when the value of $\boldsymbol{\tau}$ is changed. For example, the method described here is an active set method, which basically means that at each iteration the solution to a subproblem where only some fraction of the vector $\boldsymbol{\alpha}$ is involved must be calculated. If the sign of $\boldsymbol{\alpha}$ is allowed to change during such a calculation then the problem of finding this "step" becomes complex.

The solution to this apparent difficulty is to note that if the sign of some element α_i is allowed to change then in reality both β_i and β_i^* have changed sequentially (first one goes to zero, then the other becomes non-zero). In other words, this is not a single "step", but rather two steps, as shown in figure 7.1. Such problems can be avoided by only taking steps that either maintain the sign of α_i for all $1 \leq i \leq N$ or send it to/from zero. In this way, only one of β_i and β_i^* will ever change for a single step. Changes in sign are achieved by first taking a step such that $\alpha_i = 0$ afterwards, and then another step such that the $\operatorname{sgn}(\alpha_i)$ is the negative of its original value.

Formally, this can be achieved by defining the virtual bound vectors $\bar{\mathbf{v}}_{\boldsymbol{\tau}}$ and $\bar{\mathbf{h}}_{\boldsymbol{\tau}}$:

$$\bar{v}_{\boldsymbol{\tau} i} = \begin{cases} 0 & \text{if } \tau_i = +1 \\ v_i & \text{otherwise} \end{cases}$$
$$\bar{h}_{\boldsymbol{\tau} i} = \begin{cases} 0 & \text{if } \tau_i = -1 \\ h_i & \text{otherwise} \end{cases}$$

which are used as effective replacements for the bounds $\mathbf{v}$ and $\mathbf{h}$ when calculating the step, thereby preventing a change in sign during a step. To change the sign, the value of $\boldsymbol{\tau}$ must be changed first in a manner compatible with $\boldsymbol{\alpha}$.

7.10 Optimisation State

Because of the inequality constraints in the optimisation problem (7.14) it is necessary to solve it iteratively. To this end, it is necessary to have a way of defining the "state" of the problem at any point before, during or after the optimisation. Each iteration of the solution process is then simply means of mapping the present state to the next state of the problem.

The state of the optimisation process can be unambiguously defined by:

$$(\maltese, \mathbf{T})$$

However, for reasons which will become apparent later, the following definition of state is used instead:

$$\daleth = (\digamma, \maltese, \mathbf{T}) \tag{7.45}$$

where $\digamma$ describes whatever "other" information may be kept by a particular algorithm.

As the state will be evolving as the optimisation process proceeds, the superscript $k \in \mathbb{Z}^+$ is used to indicate the iteration to which the state refers, so:

$$\begin{aligned} \daleth^{(k)} &= \left(\digamma^{(k)}, \maltese^{(k)}, \mathbf{T} \right) \\ \maltese^{(k)} &= \left(\boldsymbol{\alpha}^{(k)}, b^{(k)}, \boldsymbol{\tau}^{(k)}, \mathbf{m}^{(k)} \right) \end{aligned}$$

refers to the state directly after iteration k (where $\mathbf{T}$ is assumed to be constant throughout). Hence the process of solving (7.14) can be summarised by a sequence of T states:

$$\left(\daleth^{(1)}, \daleth^{(2)}, \ldots, \daleth^{(T)} \right)$$

where T is the number of iterations to reach the final solution. $\daleth^{(1)}$ is the initial state, and for the final (optimal) solution is also denoted $\daleth^{(*)}$ (i.e. $\daleth^{(*)} = \daleth^{(T)}$).

7.11 Pivoting

During an iterative algorithm, it will be necessary to change the active set, and in particular $\boldsymbol{\tau}$ and $\mathbf{m}$, while ensuring that the $\maltese$ remains in standard form. There are two basic operations involved here, namely:

1. Activation of constraints on previously free variables. That is, changing an element of the type vector $\boldsymbol{\tau}$ from $\tau_i = \pm 1$ to $\tau \in \{-2, 0, +2)$ whilst simultaneously re-ordering the order vector $\mathbf{m}$ to retain standard form.

2. De-activation of constraints on previously actively constrained variables. This is essentially the reverse of the previous operation.

The first of operations will be denoted constrain (i), which is to be read as "activate the relevant constraint on variable $\underline{\alpha}_i$" (where it is assumed that $\underline{\tau}_i = \pm 1$, and $\underline{\alpha}_i \in \{\underline{v}_i, 0, \underline{h}_i\}$). Similarly, item 2 is encapsulated by the operations $\mathrm{free}_L\,(i)$ and $\mathrm{free}_U\,(i)$, which is to be read as "de-activate the current active constraint on $\underline{\alpha}_i$" (where it is assumed that $\underline{\alpha}_i \in \{-2, 0, +2\}$, and hence that $\underline{\alpha}_i \in \{\underline{v}_i, 0, \underline{h}_i\}$). The difference between $\mathrm{free}_L\,(i)$ and $\mathrm{free}_U\,(i)$ is that $\mathrm{free}_L\,(i)$ will convert $\underline{\alpha}_i$ into a *negative* free variable, whereas $\mathrm{free}_U\,(i)$ will make $\underline{\alpha}_i$ into a *positive* free variable. Note that none of these operations will change the value of $\underline{\alpha}_i$ itself.

The effect of the constrain (i) operation, assuming $N_Z + N_L + N_U + 1 \leq i \leq N$ and $\underline{\alpha}_i \in \{\underline{v}_i, 0, \underline{h}_i\}$, is defined by:

$$\begin{aligned}
\mathrm{constrain}\,(i)\,\gimel &= \mathrm{constrain}\,(i)\,(\digamma, \maltese, \mathbf{T}) \\
&= (\mathrm{constrain}\,(i)\,\digamma, \mathrm{constrain}\,(i)\,\maltese, \mathrm{constrain}\,(i)\,\mathbf{T})
\end{aligned}$$

where, neglecting for the moment the operation $\mathrm{constrain}\,(i)\,\digamma$:

$$\begin{aligned}
\mathrm{constrain}\,(i)\,\mathbf{T} &= \mathbf{T} \\
\mathrm{constrain}\,(i)\,\maltese &= \mathrm{constrain}\,(i)\,(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \mathbf{m}) \\
&= \mathrm{constrain}\,(i)\left(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \begin{bmatrix} m_1 & m_2 & \ldots & m_{i-1} & m_i & m_{i+1} & \ldots & m_N \end{bmatrix}^T\right) \\
&= \left(\boldsymbol{\alpha}, b, \boldsymbol{\tau}', \begin{bmatrix} m_1 & m_2 \ldots & m_{N_X} & m_i & m_{N_X+1} & \ldots & m_{i-1} & m_{i+1} & \ldots & m_N \end{bmatrix}^T\right)
\end{aligned}$$

and:

$$\begin{aligned}
N_X &= \begin{cases} N_Z & \text{if } \alpha_{m_i} = 0 \\ N_Z + N_L & \text{if } \alpha_{m_i} = v_{m_i} \wedge v_{m_i} < 0 \\ N_Z + N_L + N_U & \text{if } \alpha_{m_i} = h_{m_i} \wedge h_{m_i} > 0 \end{cases} \\
\tau'_j &= \begin{cases} \tau_j & \text{if } m_i \neq j \\ 0 & \text{if } m_i = j \wedge \alpha_{m_i} = 0 \\ -2 & \text{if } m_i = j \wedge \alpha_{m_i} = v_{m_i} \wedge v_{m_i} < 0 \\ +2 & \text{if } m_i = j \wedge \alpha_{m_i} = h_{m_i} \wedge h_{m_i} > 0 \end{cases}
\end{aligned}$$

The effect of the $\mathrm{free}_L\,(i)$ operation, assuming $1 \leq i \leq N_Z + N_L$ and $v_{m_i} < 0$, is defined by:

$$\begin{aligned}
\mathrm{free}_L\,(i)\,\gimel &= \mathrm{free}_L\,(i)\,(\digamma, \maltese, \mathbf{T}) \\
&= (\mathrm{free}_L\,(i)\,\digamma, \mathrm{free}_L\,(i)\,\maltese, \mathrm{free}_L\,(i)\,\mathbf{T})
\end{aligned}$$

where, neglecting for the moment the operation $\mathrm{free}_L\,(i)\,\digamma$:

$$\begin{aligned}
\mathrm{free}_L\,(i)\,\mathbf{T} &= \mathbf{T} \\
\mathrm{free}_L\,(i)\,\maltese &= \mathrm{free}_L\,(i)\,(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \mathbf{m}) \\
&= \mathrm{free}_L\,(i) \left(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \left[\begin{array}{cccccccc} m_1 & m_2 & \ldots & m_{i-1} & m_i & m_{i+1} & \ldots & m_N \end{array} \right]^T \right) \\
&= \left(\boldsymbol{\alpha}, b, \boldsymbol{\tau}', \left[\begin{array}{ccccccc} m_1 & m_2 \ldots & m_{i-1} & m_{i+1} & \ldots & m_N & m_i \end{array} \right]^T \right)
\end{aligned}$$

and:

$$\tau'_j = \left\{ \begin{array}{ll} \tau_j & \text{if } m_i \neq j \\ -1 & \text{if } m_i = j \end{array} \right.$$

Finally, the effect of the $\mathrm{free}_U\,(i)$ operation, assuming $1 \leq i \leq N_Z$ or $N_Z + N_L + 1 \leq i \leq N_Z + N_L + N_U$; and $h_{m_i} > 0$, is defined by:

$$\begin{aligned}
\mathrm{free}_U\,(i)\,\gimel &= \mathrm{free}_U\,(i)\,(\digamma, \maltese, \mathbf{T}) \\
&= (\mathrm{free}_U\,(i)\,\digamma, \mathrm{free}_U\,(i)\,\maltese, \mathrm{free}_U\,(i)\,\mathbf{T})
\end{aligned}$$

where, neglecting for the moment the operation $\mathrm{free}_U\,(i)\,\digamma$:

$$\begin{aligned}
\mathrm{free}_U\,(i)\,\mathbf{T} &= \mathbf{T} \\
\mathrm{free}_U\,(i)\,\maltese &= \mathrm{free}_U\,(i)\,(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \mathbf{m}) \\
&= \mathrm{free}_U\,(i) \left(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \left[\begin{array}{cccccccc} m_1 & m_2 & \ldots & m_{i-1} & m_i & m_{i+1} & \ldots & m_N \end{array} \right]^T \right) \\
&= \left(\boldsymbol{\alpha}, b, \boldsymbol{\tau}', \left[\begin{array}{ccccccc} m_1 & m_2 \ldots & m_{i-1} & m_{i+1} & \ldots & m_N & m_i \end{array} \right]^T \right)
\end{aligned}$$

and:

$$\tau_j' = \begin{cases} \tau_j & \text{if } m_i \neq j \\ +1 & \text{if } m_i = j \end{cases}$$

For technical reasons, the following pivoting operation is also necessary to allow one to modify the order vector $\mathbf{m}$ without modifying the active set. The $\text{eswap}\,(i,j)$ operation (which is only defined if either $\underline{\tau}_i = \pm 1$ and $\underline{\tau}_j = \pm 1$; or $\underline{\tau}_i = \underline{\tau}_j$) is defined by:

$$\begin{aligned} \text{eswap}\,(i,j)\,\gimel &= \text{eswap}\,(i,j)\,(\digamma, \maltese, \mathbf{T}) \\ &= (\text{eswap}\,(i,j)\,\digamma, \text{eswap}\,(i,j)\,\maltese, \text{eswap}\,(i,j)\,\mathbf{T}) \end{aligned}$$

where, neglecting for the moment the operation $\text{eswap}\,(i,j)\,\digamma$:

$$\begin{aligned} \text{eswap}\,(i,j)\,\mathbf{T} &= \mathbf{T} \\ \text{eswap}\,(i,j)\,\maltese &= \text{eswap}\,(i,j)\,(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \mathbf{m}) \\ &= \text{eswap}\,(i,j) \left(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \begin{bmatrix} m_1 \\ m_2 \\ \ldots \\ m_{i-1} \\ m_i \\ m_{i+1} \\ \ldots \\ m_{j-1} \\ m_j \\ m_{j+1} \\ \ldots \\ m_N \end{bmatrix} \right) \\ &= \left(\boldsymbol{\alpha}, b, \boldsymbol{\tau}, \begin{bmatrix} m_1 & m_2 & \ldots & m_{i-1} & m_j & m_{i+1} & \ldots & m_{j-1} & m_i & m_{j+1} & \ldots & m_N \end{bmatrix}^T \right) \end{aligned}$$

7.11.1 Effect of Pivoting on $\digamma$

The $\digamma$ part of the training state $\gimel$ represents any "other" not strictly necessary, but potentially useful, information used by the training algorithm. It is assumed that the order of any matrices and vectors therein is fixed, being aligned with the order of elements in $\mathbf{T}$ (i.e. the same order as the original training data), and dereferenced in the usual manner using the order vector $\mathbf{m}$. Such vectors may also be implicit function of $\boldsymbol{\tau}$.

Unless explicitly stated otherwise, it may be assumed that $\digamma = \{E, \mathbf{e}_{\boldsymbol{\tau}}, f, \mathbf{G}_{\boldsymbol{\tau}}, \boldsymbol{\beth}_{\boldsymbol{\tau}}\}$, where $\boldsymbol{\beth}_{\boldsymbol{\tau}}$ is defined in chapter 8. Clearly, the value of E and f are both independent of any pivoting operations. However, the gradient vector $\mathbf{e}_{\boldsymbol{\tau}}$ and the hessian matrix $\mathbf{G}_{\boldsymbol{\tau}}$ are both implicit functions of $\boldsymbol{\tau}$, and hence will both be affected by pivoting operations $\mathrm{constrain}\,(i)$, $\mathrm{free}_L\,(i)$, $\mathrm{free}_U\,(i)$ and $\mathrm{eswap}\,(i,j)$ thusly:

$$\begin{aligned}
\mathrm{constrain}\,(i)\,\digamma &= \{E, \mathrm{constrain}\,(i)\,\mathbf{e}_{\boldsymbol{\tau}}, f, \mathrm{constrain}\,(i)\,\mathbf{G}_{\boldsymbol{\tau}}, \mathrm{constrain}\,(i)\,\boldsymbol{\beth}_{\boldsymbol{\tau}}\} \\
\mathrm{free}_L\,(i)\,\digamma &= \{E, \mathrm{free}_L\,(i)\,\mathbf{e}_{\boldsymbol{\tau}}, f, \mathrm{free}_L\,(i)\,\mathbf{G}_{\boldsymbol{\tau}}, \mathrm{free}_L\,(i)\,\boldsymbol{\beth}_{\boldsymbol{\tau}}\} \\
\mathrm{free}_U\,(i)\,\digamma &= \{E, \mathrm{free}_U\,(i)\,\mathbf{e}_{\boldsymbol{\tau}}, f, \mathrm{free}_U\,(i)\,\mathbf{G}_{\boldsymbol{\tau}}, \mathrm{free}_U\,(i)\,\boldsymbol{\beth}_{\boldsymbol{\tau}}\} \\
\mathrm{eswap}\,(i,j)\,\digamma &= \{E, \mathbf{e}_{\boldsymbol{\tau}}, f, \mathbf{G}_{\boldsymbol{\tau}}, \mathrm{eswap}\,(i,j)\,\boldsymbol{\beth}_{\boldsymbol{\tau}}\}
\end{aligned}$$

assuming the operations $\mathrm{constrain}\,(i)\,\boldsymbol{\beth}_{\boldsymbol{\tau}}$, $\mathrm{free}_L\,(i)\,\boldsymbol{\beth}_{\boldsymbol{\tau}}$, $\mathrm{free}_U\,(i)\,\boldsymbol{\beth}_{\boldsymbol{\tau}}$ and $\mathrm{eswap}\,(i,j)\,\boldsymbol{\beth}_{\boldsymbol{\tau}}$ are appropriately defined.

The affect of the operations on $\mathbf{e}_{\boldsymbol{\tau}}$ and $\mathbf{G}_{\boldsymbol{\tau}}$ is:

$$\mathrm{constrain}\,(i)\,e_{\boldsymbol{\tau}j} = \begin{cases} e_{\boldsymbol{\tau}j} & \text{if } m_i \neq j \vee \alpha_{m_i} \neq 0 \\ e_{\boldsymbol{\tau}j} - \tau_j\left(\rho_j^{(*)} + E\mu_j^{(*)}\right) & \text{if } m_i = j \wedge \alpha_{m_i} = 0 \end{cases}$$

$$\mathrm{free}_L\,(i)\,e_{\boldsymbol{\tau}j} = \begin{cases} e_{\boldsymbol{\tau}j} & \text{if } m_i \neq j \vee \alpha_{m_i} \neq 0 \\ e_{\boldsymbol{\tau}j} - \left(\rho_j^{*} + E\mu_j^{*}\right) & \text{if } m_i = j \wedge \alpha_{m_i} = 0 \end{cases}$$

$$\mathrm{free}_U\,(i)\,e_{\boldsymbol{\tau}j} = \begin{cases} e_{\boldsymbol{\tau}j} & \text{if } m_i \neq j \vee \alpha_{m_i} \neq 0 \\ e_{\boldsymbol{\tau}j} + \left(\rho_j + E\mu_j\right) & \text{if } m_i = j \wedge \alpha_{m_i} = 0 \end{cases}$$

$$\text{constrain}\,(i)\,G_{\boldsymbol{\mathcal{T}}j,k} = \begin{cases} G_{\boldsymbol{\mathcal{T}}j,k} & (\text{if } m_i \neq j \wedge m_i \neq k) \vee \alpha_{m_i} \neq 0 \\ G_{\boldsymbol{\mathcal{T}}j,k} - \tau_j \tau_k \mu_j^{(*)} \mu_k^{(*)} & (\text{if } m_i = j \veebar m_i = k) \wedge \alpha_{m_i} = 0 \\ G_{\boldsymbol{\mathcal{T}}j,k} - \gamma_j^{(*)} - \mu_j^{(*)} \mu_k^{(*)} & (\text{if } m_i = j \wedge m_i = k) \wedge \alpha_{m_i} = 0 \end{cases}$$

$$\text{free}_L\,(i)\,G_{\boldsymbol{\mathcal{T}}j,k} = \begin{cases} G_{\boldsymbol{\mathcal{T}}j,k} & (\text{if } m_i \neq j \wedge m_i \neq k) \vee \alpha_{m_i} \neq 0 \\ G_{\boldsymbol{\mathcal{T}}j,k} - \tau_k \mu_j^{*} \mu_k^{(*)} & (\text{if } m_i = j \wedge m_i \neq k) \wedge \alpha_{m_i} = 0 \\ G_{\boldsymbol{\mathcal{T}}j,k} - \tau_j \mu_j^{(*)} \mu_k^{*} & (\text{if } m_i \neq j \wedge m_i = k) \wedge \alpha_{m_i} = 0 \\ G_{\boldsymbol{\mathcal{T}}j,k} + \gamma_j^{*} + \mu_j^{*} \mu_k^{*} & (\text{if } m_i = j \wedge m_i = k) \wedge \alpha_{m_i} = 0 \end{cases}$$

$$\text{free}_U\,(i)\,G_{\boldsymbol{\mathcal{T}}j,k} = \begin{cases} G_{\boldsymbol{\mathcal{T}}j,k} & (\text{if } m_i \neq j \wedge m_i \neq k) \vee \alpha_{m_i} \neq 0 \\ G_{\boldsymbol{\mathcal{T}}j,k} + \tau_k \mu_j \mu_k^{(*)} & (\text{if } m_i = j \wedge m_i \neq k) \wedge \alpha_{m_i} = 0 \\ G_{\boldsymbol{\mathcal{T}}j,k} + \tau_j \mu_j^{(*)} \mu_k & (\text{if } m_i \neq j \wedge m_i = k) \wedge \alpha_{m_i} = 0 \\ G_{\boldsymbol{\mathcal{T}}j,k} + \gamma_j + \mu_j \mu_k & (\text{if } m_i = j \wedge m_i = k) \wedge \alpha_{m_i} = 0 \end{cases}$$

7.12 Properties of the Hessian

The Hessian of the abstract optimisation problem (7.14) is:

$$\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}} = \begin{cases} \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix} & \text{if BiasType} = \text{Var} \\ \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}} & \text{if BiasType} = \text{Fix} \end{cases}$$

As usual, $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}}$ may be split into:

$$\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}} = \begin{bmatrix} \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}Z} & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}ZL} & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}ZU} & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}ZF} \\ \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}ZL}^T & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}L} & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}LU} & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}LF} \\ \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}ZU}^T & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}LU}^T & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}U} & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}UF} \\ \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}ZF}^T & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}LF}^T & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}UF}^T & \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F} \end{bmatrix}$$

except that, in this case, if BiasType = Var then $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F} \in \Re^{(N_F+1)\times(N_F+1)}$, $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}ZF} \in \Re^{N_Z\times(N_F+1)}$, $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}LF} \in \Re^{N_L\times(N_F+1)}$ and $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}UF} \in \Re^{N_U\times(N_F+1)}$.

Theorem 7.2. $\mathbf{G}_{\boldsymbol{\mathcal{T}}}$ *is a positive semidefinite matrix.*

Proof. Recall that by definition a positive semidefinite matrix $\mathbf{A}$ is a matrix for which $\mathbf{a}^T \mathbf{A} \mathbf{a} \geq 0$ for all vectors $\mathbf{a}$. Consider $\mathbf{K}$. This may be re-written:

$$\mathbf{K} = \begin{bmatrix} \boldsymbol{\varphi}(\underline{\mathbf{x}}_1)^T \\ \boldsymbol{\varphi}(\underline{\mathbf{x}}_2)^T \\ \vdots \\ \boldsymbol{\varphi}(\underline{\mathbf{x}}_N)^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\varphi}(\underline{\mathbf{x}}_1) & \boldsymbol{\varphi}(\underline{\mathbf{x}}_2) & \cdots & \boldsymbol{\varphi}(\underline{\mathbf{x}}_N) \end{bmatrix}$$

where $\boldsymbol{\varphi} : \Re^{d_L} \to \Re^{d_H}$ is the feature map associated with the kernel function. For an arbitrary vector $\mathbf{a} \in \Re^N$:

$$\begin{aligned}
\mathbf{a}^T \mathbf{K} \mathbf{a} &= \mathbf{a}^T \left(\begin{bmatrix} \boldsymbol{\varphi}(\underline{\mathbf{x}}_1)^T \\ \boldsymbol{\varphi}(\underline{\mathbf{x}}_2)^T \\ \vdots \\ \boldsymbol{\varphi}(\underline{\mathbf{x}}_N)^T \end{bmatrix} \begin{bmatrix} \boldsymbol{\varphi}(\underline{\mathbf{x}}_1) & \boldsymbol{\varphi}(\underline{\mathbf{x}}_2) & \cdots & \boldsymbol{\varphi}(\underline{\mathbf{x}}_N) \end{bmatrix} \right) \mathbf{a} \\
&= \left(a_1 \boldsymbol{\varphi}(\underline{\mathbf{x}}_1) + a_2 \boldsymbol{\varphi}(\underline{\mathbf{x}}_2) + \cdots + a_N \boldsymbol{\varphi}(\underline{\mathbf{x}}_N) \right)^T \left(a_1 \boldsymbol{\varphi}(\underline{\mathbf{x}}_1) + a_2 \boldsymbol{\varphi}(\underline{\mathbf{x}}_2) + \cdots + a_N \boldsymbol{\varphi}(\underline{\mathbf{x}}_N) \right) \\
&= \mathbf{w}^T \mathbf{w} \geq 0
\end{aligned}$$

Hence $\mathbf{K}$ is positive semidefinite. Now consider $\mathbf{G}_{\boldsymbol{\mathcal{T}}}$. Again, for an arbitrary vector $\mathbf{a}$:

$$\begin{aligned}
\mathbf{a}^T \mathbf{G}_{\boldsymbol{\mathcal{T}}} \mathbf{a} &= \mathbf{a}^T \mathbf{K} \mathbf{a} + \mathbf{a}^T \operatorname{diag}(\boldsymbol{\gamma}) \mathbf{a} + \mathbf{a}^T \left(\left(\operatorname{sgn}(\boldsymbol{\tau}) \boldsymbol{\mu}_{\boldsymbol{\mathcal{T}}}^{(*)} \right) \left(\operatorname{sgn}(\boldsymbol{\tau}) \boldsymbol{\mu}_{\boldsymbol{\mathcal{T}}}^{(*)} \right)^T \right) \mathbf{a} \\
&= \mathbf{a}^T \mathbf{K} \mathbf{a} + \mathbf{a}^T \operatorname{diag}(\boldsymbol{\gamma}) \mathbf{a} + \left(\mathbf{a}^T \left(\operatorname{sgn}(\boldsymbol{\tau}) \boldsymbol{\mu}_{\boldsymbol{\mathcal{T}}}^{(*)} \right) \right)^2
\end{aligned}$$

Given that $\boldsymbol{\gamma} \geq \mathbf{0}$, it follows that $\mathbf{a}^T \mathbf{G}_{\boldsymbol{\mathcal{T}}} \mathbf{a} \geq 0$ for any $\mathbf{a}$. Hence $\mathbf{G}_{\boldsymbol{\mathcal{T}}}$ is a positive semidefinite matrix. □

Corollary 7.3. *There are no non-global solutions to the abstract optimisation problem (7.14).*

Proof. Compare (7.2) with (10.2.2) in [40]. If we identify b with the lagrange multiplier $\mathbf{b}$ in this equation then the two are equivalent. Theorem 7.2 says that $\mathbf{G}$ is positive semidefinite, so the reasoning on page 79 of [40] applies, and hence all solutions will be global. □

Theorem 7.4. *If* $N_F = 1$ *and* BiasType = Var *then* $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$ *is non-singular.*

Proof. If $N_F = 1$ then $\det\left(\underline{\mathbf{H}}_{\boldsymbol{\tau}F}\right) = \det\left(\begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\tau}F} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix}\right) = -1$, so $\underline{\mathbf{H}}_{\boldsymbol{\tau}F}$ is non-singular. □

7.12.1 Singular Hessian Decomposition

Variable Bias Case (BiasType = Var)

Suppose $\underline{\mathbf{H}}_{\boldsymbol{\tau}F}$ is singular and $N_F > 1$. Then it follows from theorem 7.4 that the free variables may be repartitioned (without re-ordering) as follows:

$$\underline{\boldsymbol{\alpha}}_F = \begin{bmatrix} \underline{\boldsymbol{\alpha}}_{FN} \\ \underline{\alpha}_{FB} \\ \underline{\boldsymbol{\alpha}}_{FS} \end{bmatrix}, \text{ etc.}$$

$$\underline{\mathbf{G}}_{\boldsymbol{\tau}F} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\tau}FN} & \underline{\mathbf{g}}_{\boldsymbol{\tau}FBN} & \underline{\mathbf{G}}_{\boldsymbol{\tau}FSN}^T \\ \underline{\mathbf{g}}_{\boldsymbol{\tau}FBN}^T & \underline{g}_{\boldsymbol{\tau}FB} & \underline{\mathbf{g}}_{\boldsymbol{\tau}FSB}^T \\ \underline{\mathbf{G}}_{\boldsymbol{\tau}FSN} & \underline{\mathbf{g}}_{\boldsymbol{\tau}FSB} & \underline{\mathbf{G}}_{\boldsymbol{\tau}FS} \end{bmatrix}$$

such that:

- $\underline{\mathbf{H}}_{\boldsymbol{\tau}FN} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\tau}FN} & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix}$ is non-singular.
- $\underline{\mathbf{H}}_{\boldsymbol{\tau}FNB} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\tau}FN} & \underline{\mathbf{g}}_{\boldsymbol{\tau}FBN} & \mathbf{1} \\ \underline{\mathbf{g}}_{\boldsymbol{\tau}FBN}^T & \underline{g}_{\boldsymbol{\tau}FB} & 1 \\ \mathbf{1}^T & 1 & 0 \end{bmatrix}$ is singular.

So, $\underline{\boldsymbol{\alpha}}_{FN} \in \Re^{N_{FN}}$ $(0 \leq N_{FN} \leq N_F - 1)$ represents those free variables corresponding to the (N)on-singular sub-Hessian $\underline{\mathbf{H}}_{\boldsymbol{\tau}FN}$ and $\underline{\alpha}_{FB} \in \Re$ may be thought of as lying on the (B)oundary of the non-singular sub-Hessian $\underline{\mathbf{H}}_{\boldsymbol{\tau}FNB}$. By definition, $\underline{\boldsymbol{\alpha}}_{FS} \in \Re^{N_F - N_{FN} - 1}$.

If $N_F = 0$ then $\underline{\mathbf{H}}_{\boldsymbol{\tau}F} = [0]$ will be singular, in which case $\underline{\mathbf{H}}_{\boldsymbol{\tau}FN}$ is defined to be the empty matrix. It is also useful to define:

$$\underline{\mathbf{H}}_{\boldsymbol{\tau}R} = \begin{cases} \underline{\mathbf{H}}_{\boldsymbol{\tau}F} & \text{if } \underline{\mathbf{H}}_{\boldsymbol{\tau}F} \text{ is non-singular.} \\ \underline{\mathbf{H}}_{\boldsymbol{\tau}FN} & \text{otherwise.} \end{cases}$$

as a shorthand for the (R)elevant part of the Hessian. $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R} \in \Re^{N_R \times N_R}$, $\underline{\boldsymbol{\alpha}}_R \in \Re^{N_R}$ etc. are defined analogously, where $N_R = N_F$ if $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$ is non-singular, $N_R = N_{FN}$ otherwise.

Fixed Bias Case (BiasType = Fix**)**

Suppose $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}F}$ is singular and $N_F \geq 1$. Then it follows that it is possible to write:

$$\underline{\boldsymbol{\alpha}}_F = \begin{bmatrix} \underline{\boldsymbol{\alpha}}_{FN} \\ \underline{\alpha}_{FB} \\ \underline{\boldsymbol{\alpha}}_{FS} \end{bmatrix}, \text{ etc.}$$
$$\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}F} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}FN} & \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}FBN} & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}FSN}^T \\ \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}FBN}^T & \underline{g}_{\boldsymbol{\mathcal{T}}FB} & \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}FSB}^T \\ \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}FSN} & \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}FSB} & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}FS} \end{bmatrix}$$

such that:

- $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}FN} = \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}FN}$ is either non-singular or empty.
- $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}FNB} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}FN} & \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}FBN} \\ \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}FBN}^T & \underline{g}_{\boldsymbol{\mathcal{T}}FB} \end{bmatrix}$ is singular.

  Once again:

$$\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R} = \begin{cases} \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F} & \text{if } \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F} \text{ is non-singular.} \\ \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}FN} & \text{otherwise.} \end{cases}$$

is a shorthand for the (R)elevant part of the Hessian.

7.13 A Note on the Least-Squares Case

It should be noted that, while the least-squares regressor (quadratic symmetric empirical risk ($\boldsymbol{\gamma}^* = \boldsymbol{\gamma} > \mathbf{0}$), no tube shrinking ($\boldsymbol{\mu}^{(*)} = \mathbf{0}$), zero margin ($\boldsymbol{\rho}^{(*)} = \mathbf{0}$), variable bias (BiasType = Var)) may be placed in the abstract form, the resulting optimisation problem is significantly more complicated than is necessary. In particular, noting that $\mathbf{v} = -\infty\mathbf{1}$ and $\mathbf{h} = +\infty\mathbf{1}$ (and subsequently no variables will be constrained at a bound, upper or lower), (7.24) reduces to:

$$\begin{aligned} \underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}Z} &= \mathbf{0} \\ \underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}F} &= \mathbf{0} \\ f &= 0 \end{aligned}$$

which shows that the KKT condition for variables constrained at zero is precisely the same as for free variables. Indeed, as the range within which $\underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}Z}$ may optimally vary has zero width, *any* change to the free variables will invalidate any existing optimality of $\underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}Z}$. So there is nothing to be gained from having any variables constrained at zero, meaning that a better approach may be to treat *all* variables as free variables.

The problem with this is that the algorithm presented here is designed based on the assumption that the number of support vectors is relatively small. But for LS-SVRs $N_S = N_B = N$, so this is clearly a bad assumption in this case. So while the algorithm described here will successfully deal with the LS-SVR case, it will do so rather badly (in terms of speed and memory use - of course, the resulting LS-SVR will be exactly the same as for any other optimisation algorithm which can successfully find the unique global minima).

For this reason, it is preferable (where feasible) to solve the LS-SVR optimisation problem directly by computing:

$$\begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{K} + \mathrm{diag}\left(\boldsymbol{\gamma}\right) & \mathbf{1} \\ \mathbf{1}^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{z} \\ 0 \end{bmatrix}$$

Chapter 8

TRAINING THE SVM - AN ACTIVE SET APPROACH

"Ford, you're turning into a penguin. Stop it."

– Arthur Dent

As noted previously, only those α_i's associated with support vectors will have non-zero values. An attractive feature of SVMs is that support vectors usually make up only a small fraction of the total training set (the ratio of support vectors to training set size may increase if the training set is noisy, but even in this case the ratio will tend to remain relatively small). Of all the methods of solving linearly constrained quadratic programming problems, active set methods [40] seem best suited to take advantage of this feature of SVMs. This is because, by using an active set method, they are able to reduce the effective dimensionality of the problem from the number of training points, which may be very large, to the number of support vectors (or some interim guess of the number of support vectors), which is typically small.

In an active set method, constraints are divided into two sets, namely the set of active constraints (the active set) and the set of inactive constraints, as described previously. The algorithm then iteratively steps towards the solution, adjusting the active set after each step, until the optimal active set (and hence optimal solution) is arrived at. For any given iteration, the step is calculated by treating the active constraints as equality constraints, temporarily discarding the inactive constraints, and solving the resultant unconstrained optimisation problem (for a detailed introduction to active set methods, see for example [40]).

Unless otherwise stated, it is assumed throughout that the state $\gimel$ is in standard form.

8.1 The Hessian Factorisation $\beth_{\boldsymbol{\mathcal{T}}}$

For (nearly) every iteration of the algorithm, it will be necessary to solve the equation $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}\underline{\mathbf{r}} = \underline{\mathbf{s}}$ for $\underline{\mathbf{r}}$, where $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}$ and $\underline{\mathbf{s}}$ are known. However, actually calculating the inverse of $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}$ is too computationally expensive to be feasible. To overcome this problem, it is useful to calculate (and maintain throughout the algorithm) a factorisation, $\beth_{\boldsymbol{\mathcal{T}}} \in \digamma$, of $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}$. In principle any number of factorisations could be used, the only criterion being the ability to quickly solve $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}\underline{\mathbf{r}} = \underline{\mathbf{s}}$ for $\underline{\mathbf{r}}$ using $\beth_{\boldsymbol{\mathcal{T}}}$ and also that the factorisation itself may be quickly modified to reflect changes in $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}$ (through pivoting operations). Two factorisations considered here are the inverse and Cholesky factorisations.

8.2 Overall Structure of the Active Set Algorithm

The basic structure of my algorithm is shown in figure 8.1. In future, the algorithm in figure 8.1 will be referred to simply as the algorithm. Structurally, the algorithm is typical of active set methods. A Newton step (or some other step if the Hessian is singular) for the free variables is calculated. This step is then scaled ($\beta^{(k)}$ being the scale factor) to prevent moving outside of the feasible region, and the scaled step is taken. If the KKT conditions are not met after this, the active set is modified in a minimal fashion (a single constraint is activated or deactivated using the pivoting operations) and the process is repeated. Otherwise, the algorithm terminates, the optimal solution being:

$$\begin{bmatrix} \underline{\boldsymbol{\alpha}}^{(*)} \\ b^{(*)} \end{bmatrix} = \begin{bmatrix} \hat{\underline{\boldsymbol{\alpha}}}^{(k)} \\ \hat{b}^{(k)} \end{bmatrix}$$

Notes on the algorithm:

- The relevant section number for each block is given beside that block for easy reference.

- The default state $\gimel^{(1)}$ upon entering the algorithm (if none has already been defined) is:

$$\begin{aligned} m_i^{(1)} &= i \,\forall\, 1 \leq i \leq N \\ \boldsymbol{\tau}^{(1)} &= \mathbf{0} \\ \boldsymbol{\alpha}^{(1)} &= \mathbf{0} \\ b^{(1)} &= 0 \\ E &= 0 \\ \mathbf{e} &= -\mathbf{z} \\ f &= 0 \end{aligned}$$

- $\left(\Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} Z}^{(k)}, \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} L}^{(k)}, \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} U}^{(k)}\right)$ is always calculated using the formula:

$$\begin{bmatrix} \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} Z}^{(k)} \\ \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} L}^{(k)} \\ \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} U}^{(k)} \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\tau} ZF} & \mathbf{1} \\ \underline{\mathbf{G}}_{\boldsymbol{\tau} LF} & \mathbf{1} \\ \underline{\mathbf{G}}_{\boldsymbol{\tau} UF} & \mathbf{1} \end{bmatrix} \begin{bmatrix} \Delta \underline{\boldsymbol{\alpha}}_F^{(k)} \\ \Delta b^{(k)} \end{bmatrix} \tag{8.1}$$

- $\Delta E^{(k)}$ is always calculated using:

$$\Delta E^{(k)} = \underline{\boldsymbol{\mu}}_F^{(*)T} \left(\Delta \underline{\boldsymbol{\alpha}}_F^{(k)} \underline{\boldsymbol{\tau}}_F \right)$$

- The largest possible scaling factor $\beta^{(k)}$ (which is used here) satisfying the constraint $-\underline{\bar{\mathbf{v}}}_{\boldsymbol{\tau} F}^{(k)} \leq \underline{\boldsymbol{\alpha}}_F^{(k)} \leq \underline{\bar{\mathbf{h}}}_{\boldsymbol{\tau} F}^{(k)}$ will be:

$$\beta^{(k)} = \min_{i:\underline{\tau}_i = \pm 1} \left(1, \min_{i:\Delta \underline{\alpha}_i^{(k)} < 0} \frac{\underline{\bar{v}}_{\boldsymbol{\tau} i}^{(k)} - \underline{\alpha}_i^{(k)}}{\Delta \underline{\alpha}_i^{(k)}}, \min_{i:\Delta \underline{\alpha}_i^{(k)} > 0} \frac{\underline{\bar{h}}_{\boldsymbol{\tau} i}^{(k)} - \underline{\alpha}_i^{(k)}}{\Delta \underline{\alpha}_i^{(k)}} \right) \tag{8.2}$$

- As will be shown in section 8.2.5, it is sufficient, when checking the KKT conditions (7.24), to check:

$$\begin{aligned}
\hat{\underline{e}}^{(k)}_{\boldsymbol{\tau} Zi} &\in \begin{cases} \left[-\left(\underline{\rho}_{Zi} + \hat{E}^{(k)}\underline{\mu}_{Zi}\right) - \varepsilon, \left(\underline{\rho}^*_{Zi} + \hat{E}^{(k)}\underline{\mu}^*_{Zi}\right) + \varepsilon\right] & \text{if } \underline{v}_i < 0 < \underline{h}_i \\ \left[-\left(\underline{\rho}_{Zi} + \hat{E}^{(k)}\underline{\mu}_{Zi}\right) - \varepsilon, \infty\right) & \text{if } \underline{v}_i = 0 < \underline{h}_i \\ \left(-\infty, \left(\underline{\rho}^*_{Zi} + \hat{E}^{(k)}\underline{\mu}^*_{Zi}\right) + \varepsilon\right] & \text{if } \underline{v}_i < 0 = \underline{h}_i \\ (-\infty, \infty) & \text{if } \underline{v}_i = 0 = \underline{h}_i \end{cases} \\
\hat{\underline{\mathbf{e}}}^{(k)}_{\boldsymbol{\tau} L} &\geq \varepsilon \mathbf{1} \\
\hat{\underline{\mathbf{e}}}^{(k)}_{\boldsymbol{\tau} U} &\leq -\varepsilon \mathbf{1} \\
\hat{f}^{(k)} &\in [-\varepsilon, \varepsilon] \text{ if BiasType} = \text{Var}
\end{aligned} \tag{8.3}$$

where the constant $0 < \varepsilon \ll 1$ is necessary to prevent cycling due to cumulative numerical errors. Typically, $\varepsilon \simeq 10^{-3}$ was found to be sufficient for pattern classification. For regression, ε typically must be of the order of $\varepsilon \simeq 10^{-6}$, depending on the width of the ϵ-insensitive zone (note that $\epsilon \neq \varepsilon$).

8.2.1 Modifying the Active Set

In the algorithm, modification of the active set always takes the form of activating or deactivating a single constraint. The heuristic of [40] (used here) is particularly simple, namely:

1. If the most recent step was scaled (i.e. $\beta^{(k)} < 1$) then $\gimel^{(k+1)} = \text{constrain}\left(p^{(k)}\right) \hat{\gimel}^{(k)}$ where:

$$p^{(k)} = \mathop{\arg\min}_{i:\underline{\tau}_i = \pm 1} \left(\min_{i:\Delta\underline{\alpha}^{(k)}_i < 0} \frac{\bar{\underline{v}}^{(k)}_{\boldsymbol{\tau} i} - \underline{\alpha}^{(k)}_i}{\Delta \underline{\alpha}^{(k)}_i}, \min_{i:\Delta\underline{\alpha}^{(k)}_i > 0} \frac{\bar{\underline{h}}^{(k)}_{\boldsymbol{\tau} i} - \underline{\alpha}^{(k)}_i}{\Delta \underline{\alpha}^{(k)}_i} \right) \tag{8.4}$$

   This constrains whichever element $\underline{\alpha}^{(k)}_i$ has "run into" a boundary.

2. Otherwise (if $\beta^{(k)} = 1$), find the element $\hat{\underline{e}}^{(k)}_{\boldsymbol{\tau} q^{(k)}}$ corresponding to an actively constrained $\hat{\underline{\alpha}}^{(k)}_{q^{(k)}}$ (i.e. $\underline{\tau}^{(k)}_{q^{(k)}} \in \{-2, 0, +2\}$) that most violates the simplified KKT conditions (8.3) (according to the criteria detailed below). Then $\gimel^{(k+1)} = \text{free}_X\left(q^{(k)}\right) \hat{\gimel}^{(k)}$.

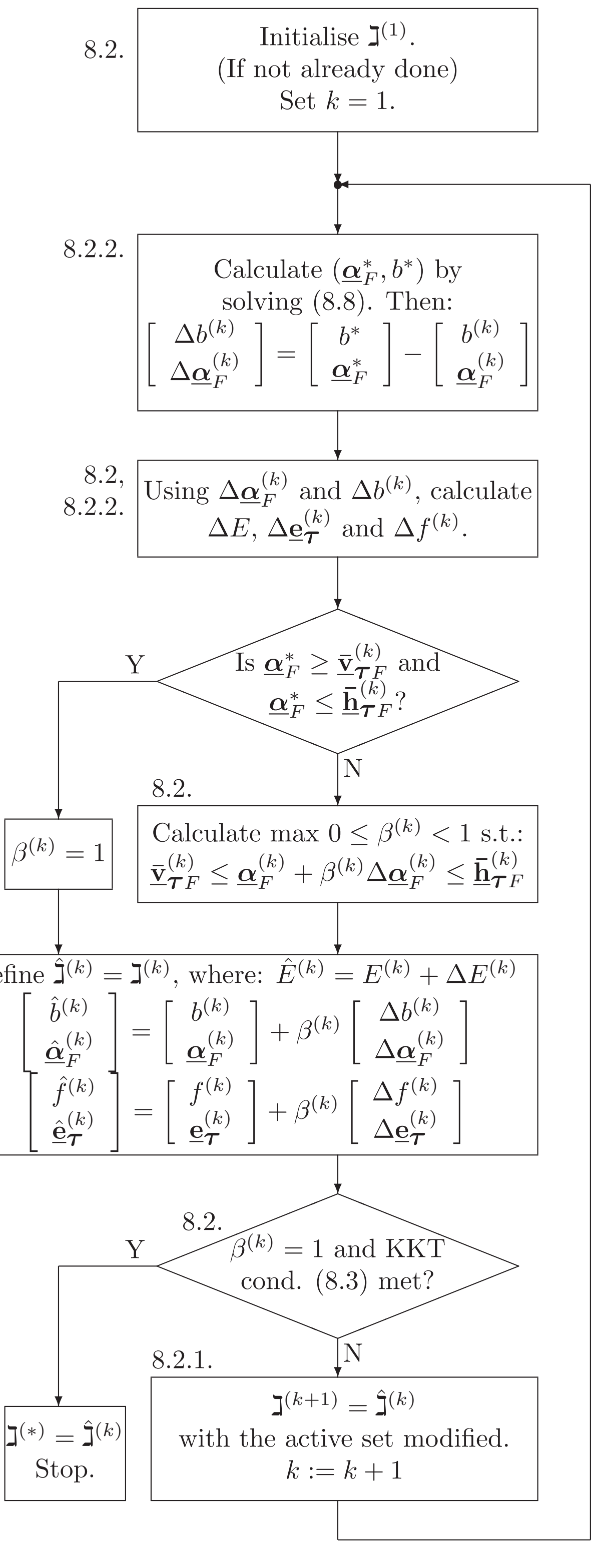

8.2.
Initialise $\daleth^{(1)}$. (If not already done) Set $k = 1$.
8.2.2.
Calculate $(\underline{\boldsymbol{\alpha}}_F^*, b^*)$ by solving (8.8). Then:
$\begin{bmatrix} \Delta b^{(k)} \\ \Delta \underline{\boldsymbol{\alpha}}_F^{(k)} \end{bmatrix} = \begin{bmatrix} b^* \\ \underline{\boldsymbol{\alpha}}_F^* \end{bmatrix} - \begin{bmatrix} b^{(k)} \\ \underline{\boldsymbol{\alpha}}_F^{(k)} \end{bmatrix}$
8.2, 8.2.2.
Using $\Delta \underline{\boldsymbol{\alpha}}_F^{(k)}$ and $\Delta b^{(k)}$, calculate ΔE, $\Delta \underline{\mathbf{e}}_{\boldsymbol{\tau}}^{(k)}$ and $\Delta f^{(k)}$.
Is $\underline{\boldsymbol{\alpha}}_F^* \geq \bar{\underline{\mathbf{v}}}_{\boldsymbol{\tau} F}^{(k)}$ and $\underline{\boldsymbol{\alpha}}_F^* \leq \bar{\underline{\mathbf{h}}}_{\boldsymbol{\tau} F}^{(k)}$?
Y
N
8.2.
$\beta^{(k)} = 1$
Calculate max $0 \leq \beta^{(k)} < 1$ s.t.: $\bar{\underline{\mathbf{v}}}_{\boldsymbol{\tau} F}^{(k)} \leq \underline{\boldsymbol{\alpha}}_F^{(k)} + \beta^{(k)} \Delta \underline{\boldsymbol{\alpha}}_F^{(k)} \leq \bar{\underline{\mathbf{h}}}_{\boldsymbol{\tau} F}^{(k)}$
Define $\hat{\daleth}^{(k)} = \daleth^{(k)}$, where: $\hat{E}^{(k)} = E^{(k)} + \Delta E^{(k)}$
$\begin{bmatrix} \hat{b}^{(k)} \\ \hat{\underline{\boldsymbol{\alpha}}}_F^{(k)} \end{bmatrix} = \begin{bmatrix} b^{(k)} \\ \underline{\boldsymbol{\alpha}}_F^{(k)} \end{bmatrix} + \beta^{(k)} \begin{bmatrix} \Delta b^{(k)} \\ \Delta \underline{\boldsymbol{\alpha}}_F^{(k)} \end{bmatrix}$
$\begin{bmatrix} \hat{f}^{(k)} \\ \hat{\underline{\mathbf{e}}}_{\boldsymbol{\tau}}^{(k)} \end{bmatrix} = \begin{bmatrix} f^{(k)} \\ \underline{\mathbf{e}}_{\boldsymbol{\tau}}^{(k)} \end{bmatrix} + \beta^{(k)} \begin{bmatrix} \Delta f^{(k)} \\ \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau}}^{(k)} \end{bmatrix}$
8.2.
$\beta^{(k)} = 1$ and KKT cond. (8.3) met?
Y
N
8.2.1.
$\daleth^{(*)} = \hat{\daleth}^{(k)}$ Stop.
$\daleth^{(k+1)} = \hat{\daleth}^{(k)}$ with the active set modified. $k := k + 1$

Figure 8.1: Outline of active set algorithm.

In case 2, if $N_F^{(k)} > 0$ or BiasType = Fix then $q^{(k)}$ is selected by finding the element $\hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} q^{(k)}}$ that most violates the KKT condition associated with it. This is equivalent to using the simple rule:

$$q^{(k)} = \mathop{\arg\min}_{i:\underline{\mathcal{T}}_i \in \{-2,0,+2\}} \left(\begin{array}{l} \displaystyle\min_{i:\underline{\mathcal{T}}_i=+2} -\hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} i}, \\ \displaystyle\min_{i:\underline{\mathcal{T}}_i=0 \wedge \bar{\underline{v}}_i<0} -\left(\hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} i} - \left(\underline{\rho}_i^* + \hat{E}^{(k)} \underline{\mu}_i^*\right)\right), \\ \displaystyle\min_{i:\underline{\mathcal{T}}_i=0 \wedge \bar{\underline{h}}_i>0} \left(\hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} i} + \left(\underline{\rho}_i + \hat{E}^{(k)} \underline{\mu}_i\right)\right), \\ \displaystyle\min_{i:\underline{\mathcal{T}}_i=-2} \hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} i} \end{array} \right) \tag{8.5}$$

$$X = \begin{cases} L & \text{if } \underline{\mathcal{T}}^{(k)}_{q^{(k)}} = -2 \vee \left(\underline{\mathcal{T}}^{(k)}_{q^{(k)}} = 0 \wedge \bar{\underline{v}}_i < 0 \wedge -\left(\hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} q^{(k)}} - \left(\underline{\rho}^*_{q^{(k)}} + \hat{E}^{(k)} \underline{\mu}^*_{q^{(k)}} \right)\right) < 0 \right) \\ U & \text{otherwise.} \end{cases}$$

If BiasType = Var it is not necessary to check that $\hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} q^{(k)}}$ violates a KKT condition in (8.3), because, as will be shown later, if $N_F^{(k)} > 0$ and $\beta^{(k)} = 1$, it follows that $\hat{f}^{(k)} = 0$, and hence some constrained element will fail to meet the KKT conditions (8.3) which, by virtue of the minima used in (8.5), is sufficient to ensure that $\hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} q^{(k)}}$ violates a KKT condition. Likewise, if BiasType = Fix then the simplified KKT conditions (8.3) are sufficient to ensure that $\hat{\underline{e}}^{(k)}_{\boldsymbol{\mathcal{T}} q^{(k)}}$ will violate a KKT condition in (8.3).

Continuing case 2, if $N_F^{(k)} = 0$ and BiasType = Var then there is no guarantee that $\hat{f}^{(k)} = 0$. This is a potential problem as it is possible that the algorithm may enter an endless loop. To see why this may occur, consider the explicit form of the next step, in the variable bias case, from equation (8.11):[1]

$$\left[\begin{array}{c} \Delta b^{(k+1)} \\ \Delta \alpha^{(k+1)}_{m_{q^{(k)}}} \end{array} \right] = \left[\begin{array}{c} G_{\boldsymbol{\mathcal{T}} m_{q^{(k)}} m_{q^{(k)}}} \hat{f}^{(k)} - e^{(k)}_{\boldsymbol{\mathcal{T}} m_{q^{(k)}}} \\ -\hat{f}^{(k)} \end{array} \right] \tag{8.6}$$

Note that $\alpha^{(k+1)}_{m_{q^{(k)}}} = \alpha^{(k)}_{m_{q^{(k)}}}$ (and hence by definition $\alpha^{(k+1)}_{m_{q^{(k)}}} = \underline{\alpha}^{(k)}_{q^{(k)}}$). Given this, if either $\underline{\alpha}^{(k)}_{q^{(k)}} = \bar{\underline{v}}^{(k+1)}_{\boldsymbol{\mathcal{T}}}$ and $\hat{f}^{(k)} > 0$ or $\underline{\alpha}^{(k)}_{q^{(k)}} = \bar{\underline{h}}^{(k+1)}_{\boldsymbol{\mathcal{T}}}$ and $\hat{f}^{(k)} < 0$, then it can

[1]Note that the indexing used here is with respect to the order of the original training set, not the standard form.

be seen from (8.2) and (8.6) that $\beta^{(k+1)} = 0$, and hence the magnitude of $f^{(k+1)}$ will not decrease in the next iteration. Subsequently $\alpha^{(k+1)}_{m_{q^{(k)}}}$ will be immediately reconstrained (as it is the only free variable, this is the only option), and, as $\gimel^{(k+2)} = \gimel^{(k)}$ up to pivoting, an infinite loop will result. To avoid this problem, the following (slightly modified) definition of $q^{(k)}$ is used when $N_F^{(k)} = 0$ and BiasType = Var:

$$\begin{aligned} q^{(k)} = \mathop{\arg\min}_{i:\underline{\tau}_i \in \{-2,0,+2\}} \Bigg(& \min_{i:\underline{\tau}_i=+2 \wedge \hat{f}^{(k)} \geq 0} -\hat{\underline{e}}^{(k)}_{\boldsymbol{\tau}i}, \\ & \min_{i:\underline{\tau}_i=0 \wedge \bar{\underline{v}}_i<0 \wedge \hat{f}^{(k)} \geq 0} -\left(\hat{\underline{e}}^{(k)}_{\boldsymbol{\tau}i} - \left(\underline{\rho}^*_i + \hat{E}^{(k)}\underline{\mu}^*_i\right)\right), \\ & \min_{i:\underline{\tau}_i=0 \wedge \bar{\underline{h}}_i>0 \wedge \hat{f}^{(k)} \leq 0} \left(\hat{\underline{e}}^{(k)}_{\boldsymbol{\tau}i} + \left(\underline{\rho}_i + \hat{E}^{(k)}\underline{\mu}_i\right)\right), \\ & \min_{i:\underline{\tau}_i=-2 \wedge \hat{f}^{(k)} \leq 0} \hat{\underline{e}}^{(k)}_{\boldsymbol{\tau}i} \Bigg) \end{aligned} \tag{8.7}$$

$$X = \begin{cases} L & \text{if } \underline{\tau}^{(k)}_{q^{(k)}} = -2 \vee \left(\underline{\tau}^{(k)}_{q^{(k)}} = 0 \wedge \bar{\underline{v}}_i < 0 \wedge \hat{f}^{(k)} \geq 0 \wedge -\left(\hat{\underline{e}}^{(k)}_{\boldsymbol{\tau}q^{(k)}} - \left(\underline{\rho}^*_{q^{(k)}} + \hat{E}^{(k)}\underline{\mu}^*_{q^{(k)}}\right)\right) < 0\right) \\ U & \text{otherwise.} \end{cases}$$

where the additional constraint on $\hat{f}^{(k)}$ ensures that a loop condition cannot occur. Note that equations (8.5) and (8.7) are equivalent if $\hat{f}^{(k)} = 0$. When proving the convergence of the algorithm (section 8.2.5), the following technical result will be needed:

Theorem 8.1. *If* $N_F^{(k)} = 0$ *and* BiasType = Var *then the solution to equation (8.7) is well defined.*

Proof. First, suppose that $\hat{f}^{(k)} = 0$. In this case, the solution to equation (8.7) must be well defined, as it is equivalent to (8.5) in this case, and hence some constrained element must have failed to meet the KKT conditions (8.3) which, by virtue of the minima used in (8.5), is sufficient to ensure that $\hat{\underline{e}}^{(k)}_{\boldsymbol{\tau}q^{(k)}}$ is well defined and violates a KKT condition.

Otherwise, suppose that the theorem is not correct when $\hat{f}^{(k)} \neq 0$ - i.e. the solution to (8.7) is not well defined. It must be true that $N_U^{(k)} + N_L^{(k)} > 0$ (otherwise $\hat{f}^{(k)} = 0$).

If $\hat{f}^{(k)} > 0$ then, by the definition of $\hat{f}^{(k)}$, $N_U^{(k)} > 0$, and so there must be some i for which $\underline{\tau}_i = +2$. But this makes the solution to (8.7) well-defined. Hence if the solution to (8.7) is not well defined, it must be true that $\hat{f}^{(k)} < 0$. But in this case, $N_L^{(k)} > 0$, so there must be some i such that $\underline{\tau}_i = -2$, which once again, this makes the solution to (8.7) well-defined.

Hence if $N_F^{(k)} = 0$ and BiasType $=$ Var then the solution to equation (8.7) must be well defined. □

Theorem 8.2. *In the algorithm, if* $N_F^{(k)} = 0$ *and* BiasType $=$ Var *then* $\Delta_{\min} \leq \left| f^{(k+2)} - f^{(k)} \right| \leq \left| f^{(k)} \right|$ *and* $\left| f^{(k+2)} \right| \leq \left| f^{(k)} \right|$*, where:*

$$\Delta_{\min} = \min \left(\left| f^{(k)} \right|, \min_{i:\underline{\bar{v}}_i < 0} \left(\left| \underline{\bar{v}}_i \right| \right), \min_{i:\underline{\bar{h}}_i > 0} \left(\left| \underline{\bar{h}}_i \right| \right) \right) > 0$$

Proof. First, suppose $f^{(k)} = 0$. As $\Delta f^{(k)} = \mathbf{1}^T \Delta \underline{\boldsymbol{\alpha}}^{(k)}$, it follows that $\Delta f^{(k)} = 0$, $f^{(k+1)} = 0$, and, from equation (8.6), $f^{(k+2)} = 0$. Hence $\left| f^{(k+2)} - f^{(k)} \right| = \min \left(\left| f^{(k)} \right|, \Delta_{\min} \right)$ and $\left| f^{(k+2)} \right| \leq \left| f^{(k)} \right|$, as required.

Now suppose $f^{(k)} > 0$. Following the same reasoning as for the previous case it is clear that $\Delta f^{(k)} = 0$. It can be seen from (8.6) that $\Delta \underline{\boldsymbol{\alpha}}_F^{(k+1)} = -\mathbf{1} f^{(k)}$, which implies that $\Delta f^{(k+1)} = -f^{(k)}$. Given that $0 < \beta^{(k+1)} \leq 1$, $\left| f^{(k+2)} \right| \leq \left| f^{(k)} \right|$. Also, as $\underline{\alpha}_{q^{(k)}}^{(k+1)} = \underline{\bar{h}}_{q^{(k+1)}}^{(k+1)}$ and $\underline{\bar{\mathbf{v}}}^{(k+1)} \leq \underline{\boldsymbol{\alpha}}^{(k+1)} \leq \underline{\bar{\mathbf{h}}}^{(k+1)}$, it follows from (8.2) that $\left| f^{(k+2)} - f^{(k)} \right| = \min \left(\left| f^{(k)} \right|, \underline{\bar{h}}_{q^{(k+1)}}^{(k+1)} - \underline{\bar{v}}_{q^{(k+1)}}^{(k+1)} \right)$. Furthermore, as one of $\underline{\bar{v}}_{q^{(k+1)}}^{(k+1)}$ and $\underline{\bar{h}}_{q^{(k+1)}}^{(k+1)}$ must be zero, $\underline{\bar{v}}_{q^{(k+1)}}^{(k+1)} \leq 0$ and $\underline{\bar{h}}_{q^{(k+1)}}^{(k+1)} \geq 0$, it follows that:

$$\min \left(\left| f^{(k)} \right|, \left| \underline{\bar{h}}_{q^{(k+1)}}^{(k+1)} \right|, \left| \underline{\bar{v}}_{q^{(k+1)}}^{(k+1)} \right| \right) \leq \left| f^{(k+2)} - f^{(k)} \right| \leq \left| f^{(k)} \right|$$

Clearly $\Delta_{\min} \leq \min \left(\left| f^{(k)} \right|, \left| \underline{\bar{h}}_{q^{(k+1)}}^{(k+1)} \right|, \left| \underline{\bar{v}}_{q^{(k+1)}}^{(k+1)} \right| \right)$, so, if $f^{(k)} > 0$:

$$\Delta_{\min} \leq \left| f^{(k+2)} - f^{(k)} \right| \leq \left| f^{(k)} \right|$$

The argument for the case $f^{(k)} < 0$ is essentially the same and has been omitted for brevity. □

8.2.2 Calculating the Step

Treating the active constraints as equality constraints, and ignoring the inactive constraints, the abstract optimisation problem (7.14) reduces to the following unconstrained quadratic programming problem:

$$\begin{aligned} \{\underline{\boldsymbol{\alpha}}_F, b\} &= \mathop{\arg\min}_{\underline{\boldsymbol{\alpha}}_F \in \Re^{N_F}} \mathop{\arg\max}_{b \in \Re} Q_{F\text{BiasType}}\left(\underline{\boldsymbol{\alpha}}_F, b\right) \\ &\text{such that: } b = b_{\text{fix}} \text{ if BiasType} = \text{Fix} \end{aligned} \tag{8.8}$$

where:

$$Q_{F\text{Var}}\left(\underline{\boldsymbol{\alpha}}_F, b\right) = \frac{1}{2} \begin{bmatrix} \underline{\boldsymbol{\alpha}}_F \\ b \end{bmatrix}^T \underline{\mathbf{H}}_{\boldsymbol{\tau} F} \begin{bmatrix} \underline{\boldsymbol{\alpha}}_F \\ b \end{bmatrix} + \begin{bmatrix} \underline{\boldsymbol{\alpha}}_F \\ b \end{bmatrix}^T \begin{bmatrix} \underline{\mathbf{r}}_F \\ r_b \end{bmatrix} \tag{8.9}$$

$$Q_{F\text{Fix}}\left(\underline{\boldsymbol{\alpha}}_F, b\right) = \frac{1}{2} \underline{\boldsymbol{\alpha}}_F^T \underline{\mathbf{H}}_{\boldsymbol{\tau} F} \underline{\boldsymbol{\alpha}}_F + \underline{\boldsymbol{\alpha}}_F^T \mathbf{1} b + \underline{\boldsymbol{\alpha}}_F^T \underline{\mathbf{r}}_F \tag{8.10}$$

and:

$$\begin{bmatrix} \underline{\mathbf{r}}_F \\ r_b \end{bmatrix} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\tau} FU}^T \underline{\mathbf{h}}_U + \underline{\mathbf{G}}_{\boldsymbol{\tau} FL}^T \underline{\mathbf{v}}_L - \underline{\mathbf{z}}_F + \underline{\boldsymbol{\tau}}_F \underline{\boldsymbol{\rho}}_{\boldsymbol{\tau} F}^{(*)} \\ \mathbf{1}^T \underline{\mathbf{h}}_U + \mathbf{1}^T \underline{\mathbf{v}}_L \end{bmatrix}$$

The solution to (8.8) (assuming that it is well defined) is denoted $(\underline{\boldsymbol{\alpha}}_F^*, b^*)$. The aim, if possible, is to calculate a finite step $\left(\Delta \underline{\boldsymbol{\alpha}}_F^{(k)}, \Delta b^{(k)}\right) = (\underline{\boldsymbol{\alpha}}_F^*, b^*) - \left(\underline{\boldsymbol{\alpha}}_F^{(k)}, b^{(k)}\right)$ or, if $(\underline{\boldsymbol{\alpha}}_F^*, b^*)$ is ill-defined (due to singularity of the hessian), move in an appropriate direction towards the optimal solution.

8.2.3 Variable Bias Case (BiasType = Var)

If the Hessian $\underline{\mathbf{H}}_{\boldsymbol{\tau} F}$ is non-singular (which implies $N_F^{(k)} > 0$) then:

$$\begin{bmatrix} \Delta \underline{\boldsymbol{\alpha}}_F^{(k)} \\ \Delta b^{(k)} \end{bmatrix} = -\underline{\mathbf{H}}_{\boldsymbol{\tau} R}^{-1} \begin{bmatrix} \underline{\mathbf{e}}_{\boldsymbol{\tau} F}^{(k)} \\ f^{(k)} \end{bmatrix} \tag{8.11}$$

where the matrix inversion is avoided by using $\beth_{\boldsymbol{\tau}}$, as described in section 8.3. It follows that:

$$\begin{bmatrix} \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} F}^{(k)} \\ \Delta f^{(k)} \end{bmatrix} = - \begin{bmatrix} \underline{\mathbf{e}}_{\boldsymbol{\tau} F}^{(k)} \\ f^{(k)} \end{bmatrix} \tag{8.12}$$

If the Hessian is singular then either $N_F^{(k)} = 0$ or $N_F^{(k)} \geq 2$ (it follows from theorem 7.4 that $N_F^{(k)} \neq 1$ here). In either case, it is clear that either the current active set is not optimal or there exists an alternative optimal active set with fewer free variables than the present active set. If $N_F^{(k)} = 0$, noting that the quadratic programming problem (8.8) is an unconstrained linear programming problem, it is clear that either the solution will not be unique (if $f^{(k)} = 0$), or will lie at $b = \pm\infty$, which is not reachable with a finite step (obviously ;). So, for simplicity, in this case:

$$\begin{aligned} \Delta b^{(k)} &= 0 \\ \Delta f^{(k)} &= 0 \end{aligned}$$

If the Hessian is singular and $N_F^{(k)} \geq 2$, one approach is to move in a direction of linear non-ascent with respect to $\underline{\boldsymbol{\alpha}}_F$ and linear non-descent with respect to b. The step should be large enough to lead to the activation of a constraint (i.e. $\beta^{(k)} < 1$), thereby preventing termination of the algorithm in a non-optimal state. Consider the step:

$$\begin{aligned} \begin{bmatrix} \Delta \underline{\boldsymbol{\alpha}}_{FN}^{(k)} \\ \Delta b^{(k)} \end{bmatrix} &= -\theta \underline{\mathbf{H}}_{\boldsymbol{\tau} R}^{-1} \begin{bmatrix} \underline{\mathbf{g}}_{\boldsymbol{\tau} FBN} \\ 1 \end{bmatrix} = -\theta \begin{bmatrix} \underline{\mathbf{s}}_{\boldsymbol{\tau}\boldsymbol{\alpha}}^{(k)} \\ s_{\boldsymbol{\tau} b}^{(k)} \end{bmatrix} \\ \begin{bmatrix} \Delta \underline{\alpha}_{FB}^{(k)} \\ \Delta \underline{\boldsymbol{\alpha}}_{FS}^{(k)} \end{bmatrix} &= \begin{bmatrix} \theta \\ \mathbf{0} \end{bmatrix} \end{aligned} \tag{8.13}$$

where once again the matrix inversion is avoided by using $\beth_{\boldsymbol{\tau}}$.

Consequently:

$$\begin{aligned} \begin{bmatrix} \Delta \underline{\mathbf{e}}_{FN}^{(k)} \\ \Delta f^{(k)} \end{bmatrix} &= \begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix} \\ \begin{bmatrix} \Delta \underline{e}_{FB}^{(k)} \\ \Delta \underline{\mathbf{e}}_{FS}^{(k)} \end{bmatrix} &= \begin{bmatrix} 0 \\ \mathbf{1} \Delta b^{(k)} + \underline{\mathbf{G}}_{\boldsymbol{\tau} FSN} \Delta \underline{\boldsymbol{\alpha}}_{FN}^{(k)} + \underline{\mathbf{g}}_{\boldsymbol{\tau} FSB}^{(k)} \Delta \underline{\alpha}_{FB}^{(k)} \end{bmatrix} \end{aligned} \tag{8.14}$$

It is easy to see that this step is in a direction of linear descent/ascent with respect to both $\boldsymbol{\alpha}_F$ and b. When selecting θ, the aim is to ensure that the magnitude of the step is sufficiently large to ensure that the $\beta^{(k)} < 1$, and furthermore that the direction is one of linear non-ascent with respect to $\underline{\boldsymbol{\alpha}}_F$ and linear non-descent with respect to b. Unfortunately, it is not in general possible to satisfy both of the constraints on the direction of the step. Because of this, ignore the requirement of linear non-descent with respect to b if $f^{(k)} \neq 0$ (if $f^{(k)} = 0$ then this requirement will be met regardless of our choice of θ). This makes implementing the algorithm simpler and, as will be shown in section 8.2.5, does not affect the convergence of the algorithm (whereas choosing to ignore the requirement of non-ascent with respect to $\underline{\boldsymbol{\alpha}}_F$ would lead to problems of cycling when $f^{(k)} = 0$ and subsequent failure of the algorithm to convergence).

It is not difficult to see that the following simple definition of θ will satisfy the requirement of linear non-ascent with respect to $\underline{\boldsymbol{\alpha}}_F$:

$$\theta = \begin{cases} -(1+\nu)\left|\underline{\overline{v}}^{(k)}_{\boldsymbol{\mathcal{T}}FB} + \underline{\overline{h}}^{(k)}_{\boldsymbol{\mathcal{T}}FB}\right| & \text{if } \underline{e}^{(k)}_{\boldsymbol{\mathcal{T}}FB} \geq \underline{\mathbf{s}}^{(k)}_{\boldsymbol{\mathcal{T}}\boldsymbol{\alpha}}{}^{T} \underline{\mathbf{e}}^{(k)}_{\boldsymbol{\mathcal{T}}FN} \\ (1+\nu)\left|\underline{\overline{v}}^{(k)}_{\boldsymbol{\mathcal{T}}FB} + \underline{\overline{h}}^{(k)}_{\boldsymbol{\mathcal{T}}FB}\right| & \text{if } \underline{e}^{(k)}_{\boldsymbol{\mathcal{T}}FB} < \underline{\mathbf{s}}^{(k)}_{\boldsymbol{\mathcal{T}}\boldsymbol{\alpha}}{}^{T} \underline{\mathbf{e}}^{(k)}_{\boldsymbol{\mathcal{T}}FN} \end{cases} \tag{8.15}$$

In this equation, $\varepsilon < \nu < 1$ is a small positive constant.

8.2.4 Fixed Bias Case (BiasType = Fix)

If the Hessian $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}F}$ is non-singular and $N_F^{(k)} > 0$ then:

$$\Delta\underline{\boldsymbol{\alpha}}_F^{(k)} = -\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}^{-1}\underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}F}^{(k)} \tag{8.16}$$

where the matrix inversion is avoided by using $\beth_{\boldsymbol{\mathcal{T}}}$, as described in section 8.3. It follows that:

$$\Delta\underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}F}^{(k)} = -\underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}F}^{(k)} \tag{8.17}$$

and by definition, $\Delta b^{(k)} = \Delta f^{(k)} = 0$ (as $f^{(k)}$ is not used here there is little point in updating it).

If the Hessian is singular then, as for the variable bias case, it is possible to find a direction of linear non-ascent w.r.t. $\underline{\boldsymbol{\alpha}}_F$ using the step:

$$\begin{aligned}
\Delta \underline{\boldsymbol{\alpha}}_{FN}^{(k)} &= -\theta \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}^{-1} \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}FBN} = -\theta \underline{\mathbf{s}}_{\boldsymbol{\mathcal{T}}\boldsymbol{\alpha}}^{(k)} \\
\begin{bmatrix} \Delta \underline{\alpha}_{FB}^{(k)} \\ \Delta \underline{\boldsymbol{\alpha}}_{FS}^{(k)} \end{bmatrix} &= \begin{bmatrix} \theta \\ \mathbf{0} \end{bmatrix}
\end{aligned} \tag{8.18}$$

where once again the matrix inversion by utilising $\boldsymbol{\beth}_{\boldsymbol{\mathcal{T}}}$.

Consequently:

$$\begin{aligned}
\Delta \underline{\mathbf{e}}_{FN}^{(k)} &= \mathbf{0} \\
\begin{bmatrix} \Delta \underline{e}_{FB}^{(k)} \\ \Delta \underline{\mathbf{e}}_{FS}^{(k)} \end{bmatrix} &= \begin{bmatrix} 0 \\ \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}FSN} \Delta \underline{\boldsymbol{\alpha}}_{FN}^{(k)} + \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}FSB}^{(k)} \Delta \underline{\alpha}_{FB}^{(k)} \end{bmatrix}
\end{aligned} \tag{8.19}$$

It is easy to see that this step is in a direction of linear ascent/descent with respect to $\underline{\boldsymbol{\alpha}}_F$. To make sure that the direction is one of non-ascent, choose:

$$\theta = \begin{cases} -(1+\nu) \left| \overline{\underline{v}}_{\boldsymbol{\mathcal{T}}FB}^{(k)} + \overline{\underline{h}}_{\boldsymbol{\mathcal{T}}FB}^{(k)} \right| & \text{if } \underline{e}_{\boldsymbol{\mathcal{T}}FB}^{(k)} \geq \underline{\mathbf{s}}_{\boldsymbol{\mathcal{T}}\boldsymbol{\alpha}}^{(k)\,T} \underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}FN}^{(k)} \\ (1+\nu) \left| \overline{\underline{v}}_{\boldsymbol{\mathcal{T}}FB}^{(k)} + \overline{\underline{h}}_{\boldsymbol{\mathcal{T}}FB}^{(k)} \right| & \text{if } \underline{e}_{\boldsymbol{\mathcal{T}}FB}^{(k)} < \underline{\mathbf{s}}_{\boldsymbol{\mathcal{T}}\boldsymbol{\alpha}}^{(k)\,T} \underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}FN}^{(k)} \end{cases} \tag{8.20}$$

where once again $\varepsilon < \nu < 1$ is a small positive constant.

In the special case $N_F^{(k)} = 0$, no step will be taken.

8.2.5 Properties of the Algorithm

In this section I provide a proof of convergence for the algorithm. This is split into two parts. First, I consider the standard variable bias case (BiasType = Var). This proof was previously published in [81]. Then I give an essentially trivial proof of convergence for the fixed bias case (BiasType = Fix).

Variable Bias Case BiasType = Var**.**

The proof of convergence for the variable bias case is split into two parts. Firstly, it is shown that after a finite number of iterations of the algorithm certain conditions must be met. Having arrived at a point where these conditions are satisfied, it is possible to directly apply a well-known convergence result, thus preventing unnecessary repetition of results readily available in the literature. In particular, reference will be made to proofs contained in [40] and [27].

Theorem 8.3. *If* BiasType = Var *and* $f^{(1)} \neq 0$ *then after a finite number* l *of iterations of the algorithm,* $f^{(l)} = 0$*. Furthermore, for all subsequent iterations* $l+n$*,* $n \geq 0$*,* $f^{(l+n)} = 0$*.*

Proof. First, assume that $N_F^{(1)} > 0$. For each subsequent iteration k with $N_F^{(k)} > 0$ either a constraint will be activated ($N_F^{(k+1)} < N_F^{(k)}$); or $\beta^{(k)} = 1$, implying that the most recent step was calculated using (8.11) and hence that $f^{(k+1)} = 0$. As N_F is bounded and decreasing, after at most $N_F^{(1)}$ iterations either $f^{\left(N_F^{(1)}+1\right)} = 0$ or $N_F^{\left(N_F^{(1)}+1\right)} = 0$, but not both.

Consider the latter possibility. By theorem 8.2, $\left|f^{(k+2)} - f^{(k)}\right| \geq \Delta_{\min}$ where $\Delta_{\min} > 0$, and $\left|f^{(k+2)}\right| \leq \left|f^{(k)}\right|$. Given that $f^{(1)}$ must be finite, it follows that $f^{(1)} \leq K\Delta_{\min}$ for some finite positive integer $K \leq \left\lceil \frac{f^{(1)}}{\Delta_{\min}} \right\rceil$. So, starting from the first iteration where $N_F = 0$, f will become zero after at most $m \leq 2K$ iterations. So, if $f^{(1)} \neq 0$, after some $l \leq 2\left\lceil \frac{f^{(1)}}{\Delta_{\min}} \right\rceil + N_F^{(1)} + 2$ iterations of the algorithm, $f^{(l)} = 0$. This proves the first part of the theorem.

Consider all possible methods of calculating subsequent steps, namely (8.11) and (8.13). In either case, $f^{(k)} = 0$ implies $\Delta f^{(k)} = 0$ and hence $f^{(k+1)} = 0$, proving the second part of the theorem. □

Having obtained this result, one may now obtain the main result of this section, namely:

Theorem 8.4. *If* BiasType = Var *then given any valid initial state* $\daleth^{(1)}$*, the algorithm will find an optimal solution* $\daleth^{(*)}$ *to (7.14) in a finite time, where an optimal solution is one satisfying (7.24).*

Proof. There are two main steps to this proof:

1. Show that the algorithm will not terminate until an optimal solution has been found.

2. Show that an optimal solution will be found in a finite time.

Consider item 1. From (8.3) it is clear that the algorithm will not terminate unless $f^{(k+1)}$, $\underline{\mathbf{e}}_Z^{(k+1)}$, $\underline{\mathbf{e}}_L^{(k+1)}$ and $\underline{\mathbf{e}}_U^{(k+1)}$ all satisfy the appropriate optimality conditions given in (7.24) (to within precision ε). Furthermore, $\underline{\mathbf{v}} \leq \underline{\boldsymbol{\alpha}} \leq \underline{\mathbf{h}}$ throughout the algorithm. All that remains to be shown is that the algorithm will not terminate unless $\underline{\mathbf{e}}_F^{(k+1)} = \mathbf{0}$ (or $N_F^{(k)} = 0$).

If $N_F^{(k)} > 0$ then the final step of the algorithm must be calculated using (8.11) and, furthermore, it must be the case that $\beta^{(k)} = 1$ (which will not be true for the singular step). Therefore from (8.12) it follows that if $N_F^{(k)} > 0$ then on termination of the algorithm $\mathbf{e}_F^{(k+1)} = \mathbf{0}$. So the algorithm cannot terminate unless the solution is optimal (i.e. the KKT conditions (7.24) are met).

Now consider item 2. Clearly, each iteration of the algorithm will take a finite time to complete. Hence proving that the algorithm will terminate after a finite time is equivalent to proving that it will terminate after a finite number of iterations.

Firstly, suppose that $f^{(1)} \neq 0$. It is known from theorem 8.3 that after some finite number of iterations, l, $f^{(l)} = 0$, and that for all subsequent iterations, $l + n$, $n \geq 0$, $f^{(l+n)} = 0$.

As already noted, if $f = 0$ (7.14) can be identified with Equations (10.2.2) and (10.3.1) in [40]. Assuming that $\mathbf{G}$ is positive definite then the proof of convergence given in [40] can be directly applied to the algorithm. For the more general case where $\mathbf{G}$ is positive semidefinite the algorithm may be identified as a more general form of that given in [27]. By analogy with the proof given in [27] it is straightforward to prove that the algorithm will terminate after a finite number of iterations.

So, in general, after some finite number of iterations, l, $f^{(l)} = 0$. Furthermore, for all subsequent iterations of the algorithm, $f^{(k)} = 0$. Given this condition the algorithm will terminate after a finite number of iterations (after the first iteration where $f^{(l)} = 0$).

This proves the theorem. $\square$

Fixed Bias Case BiasType = Fix**.**

Theorem 8.5. *If* BiasType = Fix *then given any valid initial state* $\gimel^{(1)}$*, the algorithm will find an optimal solution* $\gimel^{(*)}$ *to (7.14) in a finite time, where an optimal solution is one satisfying (7.24).*

Proof. In this case, simply note that the algorithm reduces to a standard active set algorithm, as in [40], [27], with a positive semidefinite Hessian, so the standard convergence proof applies. □

8.3 Factoring the Hessian

As defined previously, $\beth_{\mathcal{T}}$ is the inverse or some other factorisation of $\underline{\underline{\mathbf{H}}}_{\mathcal{T}R}$ defined in such a way as to facilitate the fast calculation of $\underline{\mathbf{r}}$ when $\underline{\mathbf{s}}$ is known and $\underline{\underline{\mathbf{H}}}_{\mathcal{T}R}\underline{\mathbf{r}} = \underline{\mathbf{s}}$. In this section two such factorisations will be considered in detail, viz., the inverse and the Cholesky factorisation respectively. For both cases, the following issues will be considered:

- How to initially calculate $\beth_{\mathcal{T}}$ upon entering the algorithm (unless $\beth_{\mathcal{T}}$ is already known).
- How to quickly find $\underline{\mathbf{r}}$ when $\underline{\mathbf{s}}$ is known and $\underline{\underline{\mathbf{H}}}_{\mathcal{T}R}\underline{\mathbf{r}} = \underline{\mathbf{s}}$ using $\beth_{\mathcal{T}}$ (fast matrix inversion).
- How to re-invert or re-factorise $\beth_{\mathcal{T}}$ quickly when constraints are activated and de-activated.

This section is organised as follows. In subsection 8.3.1 some fundamental results related to the Hessian $\underline{\underline{\mathbf{H}}}_{\mathcal{T}R}$ are given, and in particular how the form of $\underline{\underline{\mathbf{H}}}_{\mathcal{T}R}$ may (or may not) change when the active set is modified.

In subsections 8.3.2 and 8.3.3 two factorisation methods are introduced, the necessary background information given and notation required in subsequent sections. In subsections 8.3.4, 8.3.5, 8.3.6 and 8.3.7 how the factorisation is setup, modified and used is considered in some detail. Finally, subsection 8.3.8 considers how the algorithm may be optimised, and 8.3.9 compares the merits of the inverse and Cholesky factorisations.

8.3.1 Properties of the Hessian Matrix

Variable Bias Case BiasType = Var.

For the purposes of this subsection alone, it is useful, if $N_R > 1$, to partition $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$, $\underline{\mathbf{z}}_R$, etc. as follows:

$$\underline{\mathbf{z}}_R = \begin{bmatrix} \underline{\mathbf{z}}_{Ra} \\ \underline{z}_{Rb} \end{bmatrix}, \text{ etc.}$$

$$\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}Ra} & \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rab} \\ \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rab}^T & \underline{g}_{\boldsymbol{\mathcal{T}}Rb} \end{bmatrix}$$

If $N_F = 0$ then $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F} = [0]$ is singular by definition. If $N_F = 1$ then, as shown in theorem 7.4, $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$ is non-singular. Suppose $N_F > 1$. By definition, $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$ is non-singular. If $N_R = 1$ then the form of $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ is restricted only by the requirement of positive semi-definiteness of $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$. Otherwise:

Theorem 8.6. *If $N_R > 1$ and* BiasType = Var *then either $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ is non-singular or, after re-ordering, $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ is singular, $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}Ra}$ is non-singular ($\underline{g}_{\boldsymbol{\mathcal{T}}Rb} = \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rab}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}Ra}^{-1} \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rab}$) and $\underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rab}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}Ra}^{-1} \mathbf{1} \neq 1$.*

Proof. Suppose $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ is non-singular. Given that $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ is positive definite in this case, it follows that $\mathbf{1}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1} \mathbf{1} \neq 0$. Therefore $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$ is non-singular as required as its inverse can be formed, i.e.:

$$\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}^{-1} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1} - \frac{\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1} \mathbf{1}\mathbf{1}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1}}{\mathbf{1}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1} \mathbf{1}} & \frac{\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1} \mathbf{1}}{\mathbf{1}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1} \mathbf{1}} \\ \frac{\mathbf{1}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1}}{\mathbf{1}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1} \mathbf{1}} & -\frac{1}{\mathbf{1}^T \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}^{-1} \mathbf{1}} \end{bmatrix}$$

Now suppose that $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ is singular. First, consider the case where $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R} = \mathbf{0}$. As $N_R > 1$ it follows that $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$ is singular, which contradicts the defined non-singularity of $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$. So $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R} \neq \mathbf{0}$. Hence it must be possible (after some pivoting) to write

$\underline{\mathbf{G}}_{\mathcal{T}R}$ in the form:

$$\underline{\mathbf{G}}_{\mathcal{T}R} = \begin{bmatrix} \underline{\mathbf{G}}_{\mathcal{T}RX} & \underline{\mathbf{G}}_{\mathcal{T}RY} \\ \underline{\mathbf{G}}_{\mathcal{T}RY}^T & \underline{\mathbf{G}}_{\mathcal{T}RY}^T \underline{\mathbf{G}}_{\mathcal{T}RX}^{-1} \underline{\mathbf{G}}_{\mathcal{T}RY} \end{bmatrix}$$

where $\underline{\mathbf{G}}_{\mathcal{T}RX}$ is positive definite. As $\underline{\mathbf{H}}_{\mathcal{T}R}$ is non-singular its inverse must be:

$$\underline{\mathbf{H}}_{\mathcal{T}R}^{-1} = \begin{bmatrix} \underline{\mathbf{G}}_{\mathcal{T}RX}^{-1} + \mathbf{M}^T\mathbf{D}^{-1}\mathbf{M} & -\mathbf{M}^T\mathbf{D}^{-1} \\ -\mathbf{D}^{-1}\mathbf{M} & \mathbf{D}^{-1} \end{bmatrix}$$

where:

$$\begin{aligned} \mathbf{D} &= \begin{bmatrix} \mathbf{0} & \mathbf{1} - \underline{\mathbf{G}}_{\mathcal{T}RY}^T \underline{\mathbf{G}}_{\mathcal{T}RX}^{-1}\mathbf{1} \\ \mathbf{1}^T - \mathbf{1}^T\underline{\mathbf{G}}_{\mathcal{T}RX}^{-1}\underline{\mathbf{G}}_{\mathcal{T}RY} & -\mathbf{1}^T\underline{\mathbf{G}}_{\mathcal{T}RX}^{-1}\mathbf{1} \end{bmatrix} \\ \mathbf{M} &= \begin{bmatrix} \underline{\mathbf{G}}_{\mathcal{T}RY}^T \\ \mathbf{1}^T \end{bmatrix} \underline{\mathbf{G}}_{\mathcal{T}RX}^{-1} \end{aligned}$$

So, $\mathbf{D}$ is singular iff $\underline{\mathbf{H}}_{\mathcal{T}R}$ is singular. The two necessary conditions for $\mathbf{D}$ to be non-singular are as stated in the theorem. □

There are three basic forms which $\underline{\mathbf{H}}_{\mathcal{T}R}$ may take, namely:

1. $N_F = 0$ and $\underline{\mathbf{H}}_{\mathcal{T}R}$ is empty.
2. $N_R = 1$ and/or $\underline{\mathbf{G}}_{\mathcal{T}R}$ is positive definite.
3. $N_R > 1$, $\underline{\mathbf{G}}_{\mathcal{T}R}$ is singular and, after appropriate re-ordering, $\underline{\mathbf{G}}_{\mathcal{T}Ra}$ is non-singular and $\underline{\mathbf{g}}_{\mathcal{T}Rab}^T \underline{\mathbf{G}}_{\mathcal{T}Ra}^{-1}\mathbf{1} \neq 1$.

It is important to consider the effect of the pivoting operations $\mathrm{free}_L(i)$, $\mathrm{free}_U(i)$ and $\mathrm{constrain}(i)$ on the singular nature of the Hessian matrix.

Theorem 8.7. *If* BiasType = Var *then for all* $k \geq 1$, $\daleth^{(k+1)} = \mathrm{free}_X\left(q^{(k)}\right)\hat{\daleth}^{(k)}$ *it follows that* $N_R^{(k+1)} \geq N_R^{(k)}$.

Proof. If $N_R^{(k)} = 0$ then the theorem is trivial. Otherwise, note that the operation $\mathrm{free}_X\left(q^{(k)}\right)\hat{\daleth}^{(k)}$ will result in a new row and column being added to the end of $\underline{\mathbf{H}}_{\mathcal{T}F}$, but otherwise $\underline{\mathbf{H}}_{\mathcal{T}F}$ will not be modified, so N_R cannot decrease. □

Theorem 8.8. *If* BiasType = Var *then for all* $k \geq 1$, $\daleth^{(k+1)} = \text{constrain}\left(p^{(k)}\right) \hat{\daleth}^{(k)}$ *it follows that* $N_R^{(k+1)} \geq N_R^{(k)} - 1$.

Proof. Clearly if constrain $\left(p^{(k)}\right)$ is a well defined operation then $N_F^{(k)} \geq 1$ and hence, from theorem 7.4, $N_R^{(k)} \geq 1$.

The cases $N_R^{(k)} = 1$ and $N_R^{(k)} = 2$ are essentially trivial. Specifically, if $N_R^{(k)} = 1$ then, as $N_R^{(k)} \geq 1$ by definition, it follows that $N_R^{(k+1)} \geq N_R^{(k)} - 1$. Likewise, if $N_R^{(k)} = 2$ then $N_F^{(k+1)} \geq 1$, and hence by theorem 7.4, $N_R^{(k)} \geq 1$, so $N_R^{(k+1)} \geq N_R^{(k)} - 1$.

The case $N_R^{(k)} \geq 3$ is not so simple. First off, suppose that $p^{(k)} > N_R^{(k)}$. As $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}^{(k)}$ is unaffected by the operation, it follows that its non-singular nature will not change and hence $N_R^{(k+1)} \geq N_R^{(k)} - 1$.

Now suppose $1 \leq p^{(k)} \leq N_R^{(k)}$. First, note that for the positive semidefinite matrix $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ if $\underline{G}_{\boldsymbol{\mathcal{T}}Ri,i} = 0$ then, due to the condition of positive semidefiniticity, $\underline{G}_{\boldsymbol{\mathcal{T}}Ri,j} = \underline{G}_{\boldsymbol{\mathcal{T}}Rj,i} = 0$ for all $1 \leq j \leq N$. So, at most 1 element on the diagonal of $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ may be zero. To see why this is so, suppose n diagonals of $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ are zero. Then re-order things so that all of the zero diagonals of $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ lie at the bottom right-hand of the matrix, thus:

$$\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R} = \begin{bmatrix} \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}RX} & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}RY} \\ \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}RY}^T & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}RZ} \end{bmatrix}$$

where $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}RZ} = \mathbf{0} \in \Re^{n \times n}$ and $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}RY} = \mathbf{0}$. But by theorem 8.6, this is not possible, so at most one element on the diagonal of $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ may be zero. Assume that $\underline{G}_{\boldsymbol{\mathcal{T}}R1,1} \neq 0$ (if this is not true, then it may be made so by appropriate re-ordering, so there is no loss of generality involved in making such an assumption), and also assume that $p^{(k)} = N_R^{(k)}$ (again, if this is not true, it may be made so by appropriate re-ordering).

Borrowing the notation of section 8.3.3, define:

$$\underline{\bar{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R} = \begin{bmatrix} \underline{g}_{\boldsymbol{\mathcal{T}}Ra} & 1 & \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rba}^T \\ 1 & 0 & \mathbf{1}^T \\ \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rba} & \mathbf{1} & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}Rb} \end{bmatrix}$$

It is not difficult to show that under these circumstances it is always possible to find a lower triangular matrix $\bar{\mathbf{L}}$ with positive diagonals such that:

$$\underline{\bar{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R} = \bar{\mathbf{L}} \begin{bmatrix} 1 & 0 & \mathbf{0}^T \\ 0 & -1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \bar{\mathbf{L}}^T$$

Partition $\underline{\bar{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}$ and $\bar{\mathbf{L}}$ as follows:

$$\underline{\bar{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R} = \begin{bmatrix} \underline{\bar{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}Rn} & \underline{\bar{\mathbf{h}}}_{\boldsymbol{\mathcal{T}}Rnl} \\ \underline{\bar{\mathbf{h}}}^T_{\boldsymbol{\mathcal{T}}Rnl} & \underline{\bar{h}}_{\boldsymbol{\mathcal{T}}Rl} \end{bmatrix}$$
$$\bar{\mathbf{L}} = \begin{bmatrix} \bar{\mathbf{L}}_n & \mathbf{0} \\ \bar{\mathbf{l}}^T_{nl} & \bar{l}_l \end{bmatrix}$$

But this implies that it is possible to write:

$$\underline{\bar{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}Rn} = \bar{\mathbf{L}}_n \begin{bmatrix} 1 & 0 & \mathbf{0}^T \\ 0 & -1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \bar{\mathbf{L}}_n^T$$

where $\bar{\mathbf{L}}_n$ is a lower triangular matrix with positive diagonals, and so $\underline{\bar{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}Rn}$ must be non-singular. Hence it is possible to conclude that if $1 \leq p^{(k)} \leq N_R^{(k)}$ then $N_R^{(k+1)} \geq N_R^{(k)} - 1$. $\square$

On a practical note, it is necessary to allow for the finite numerical precision of the computer. One implication of this is that theorem 8.8 may not, in practice, be seen to be correct. However, the results do provide a rough guide as to what may be expected from an implementation.

Fixed Bias Case BiasType = Fix**.**

The following two theorems are extend theorems 8.7 and 8.8 for the general case BiasType $\in \{\text{Fix}, \text{Var}\}$:

Theorem 8.9. *For all $k \geq 1$, $\mathfrak{I}^{(k+1)} = \text{free}_X \left(q^{(k)}\right) \mathfrak{I}^{(k)}$ it follows that $N_R^{(k+1)} \geq N_R^{(k)}$.*

Proof. The variable bias case has been proven in theorem 8.7. The proof for the fixed bias case is trivially analogous to the proof of theorem 8.7. $\square$

Theorem 8.10. *For all $k \geq 1$, $\daleth^{(k+1)} = \mathrm{constrain}\left(p^{(k)}\right) \hat{\daleth}^{(k)}$ it follows that $N_R^{(k+1)} \geq N_R^{(k)} - 1$.*

Proof. The variable bias case has been proven in theorem 8.8. The proof for the fixed bias case follows from the fact that removing a row or column from a positive definite matrix will give another positive definite matrix. $\square$

8.3.2 The Inverse Update Method

The obvious method of factorising the Hessian matrix $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$ is by direct inversion.[2] This makes the solution of the fast matrix inversion problem trivial (i.e. use matrix multiplication), and, as will be shown in subsequent sections, updating the factorisation may be done efficiently using a rank-1 updating technique.

Define:

$$\breve{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R} = \begin{cases} \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R} \end{bmatrix} & \text{if BiasType} = \text{Var} \\ \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R} & \text{if BiasType} = \text{Fix} \end{cases}$$

Define:

Definition 8.11. *For the* inverse update *method $\beth_{\boldsymbol{\mathcal{T}}} = \{\mathbf{V}\}$, where $\mathbf{V} = \breve{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}R}^{-1}$ (the inverse of an empty matrix is defined to be an empty matrix).*

By default, $\mathbf{V}$ is assumed to be empty and $N_R = 0$.

8.3.3 Cholesky Update Method

It is well known [44] that, for any positive definite and symmetric matrix $\mathbf{M}$, there exists a lower triangular matrix $\mathbf{L}$ with positive diagonal entries such that $\mathbf{M} = \mathbf{L}\mathbf{L}^T$. The matrix $\mathbf{L}$ is known as the Cholesky factorisation of $\mathbf{M}$, denoted $\mathrm{chol}\left(\mathbf{M}\right)$, and has some nice numerical properties [103].

[2] [17] uses this approach, but does not consider the case where $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$ is singular.

Suppose that BiasType = Var. So long as it exists, $\underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}R}$ must be positive semidefinite and symmetric. If $N_F \geq 1$ define:

$$\begin{aligned}\underline{\mathbf{z}}_R &= \begin{bmatrix} \underline{z}_{Ra} \\ \underline{\mathbf{z}}_{Rb} \end{bmatrix}, \text{ etc.} \\ \underline{\mathbf{G}}_R &= \begin{bmatrix} \underline{g}_{\boldsymbol{\mathcal{T}}Ra} & \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rba}^T \\ \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rba} & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}Rb} \end{bmatrix}\end{aligned}$$

Assuming $\underline{g}_{\boldsymbol{\mathcal{T}}Ra} \neq 0$ (for numerical purposes, $\underline{g}_{\boldsymbol{\mathcal{T}}Ra} \geq \varepsilon$) and using theorem 8.6, it is not difficult to show that it is possible to define $\mathbf{L}$ such that:

$$\begin{bmatrix} \underline{g}_{\boldsymbol{\mathcal{T}}Ra} & 1 & \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rba}^T \\ 1 & 0 & \mathbf{1}^T \\ \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rba} & \mathbf{1} & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}Rb} \end{bmatrix} = \mathbf{LJL}^T \tag{8.21}$$

where $\mathbf{L}$ is a lower triangular matrix with positive diagonal elements, and:

$$\mathbf{J} = \begin{bmatrix} 1 & 0 & \mathbf{0}^T \\ 0 & -1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}$$

Relation (8.21) is analogous to the standard Cholesky factorisation, except that the matrix that is being factorised not positive definite (although it is non-singular), and $J_{2,2} = -1$. If $\underline{g}_{\boldsymbol{\mathcal{T}}Ra} < \varepsilon$ then $\mathbf{L}$ cannot be defined to satisfy (8.21). In this case there are two distinct possibilities:

1. $N_F = 1$ or $\underline{G}_{\boldsymbol{\mathcal{T}}F2,2} < \varepsilon$. In this case $N_R = 1$, but there is no way to factorise $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R} \in \Re^{2\times 2}$ using a Cholesky type factorisation. Fortunately, however, the inversion of $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$ is trivial in this case.

2. $N_F > 1$ and $\underline{G}_{\boldsymbol{\mathcal{T}}F2,2} \geq \varepsilon$. In this case, after the pivot eswap $(N_C + 1, N_C + 2)$ it is possible to find a lower triangular matrix $\mathbf{L}$ with positive diagonal elements satisfying (8.21).

To deal with this technical difficulty, it is convenient to define a binary variable

NoChol $\in \{\mathrm{TRUE}, \mathrm{FALSE}\}$ to indicate whether $\mathbf{L}$ may (NoChol $=$ FALSE, default) or may not (NoChol $=$ TRUE, case 1) be formed to satisfy equation (8.21). If at any point $\underline{G}_{\boldsymbol{\mathcal{T}}F1,1} < \varepsilon$ then the algorithm will either pivot using eswap$\,(N_C + 1, N_C + 2)$ if $N_F > 1$ and $\underline{G}_{\boldsymbol{\mathcal{T}}F2,2} \geq \varepsilon$, or set NoChol $=$ TRUE.

Formally, define the Cholesky factorisation as follows:

Definition 8.12. *For the* Cholesky update *method,* $\daleth_{\boldsymbol{\mathcal{T}}} = \{\mathbf{L}, \mathit{NoChol}\}$. *So long as* $N_F > 0$ *and NoChol= FALSE:*

$$\mathbf{LJL}^T = \begin{cases} \begin{bmatrix} \underline{g}_{\boldsymbol{\mathcal{T}}Ra} & 1 & \underline{\mathbf{g}}^T_{\boldsymbol{\mathcal{T}}Rba} \\ 1 & 0 & \mathbf{1}^T \\ \underline{\mathbf{g}}_{\boldsymbol{\mathcal{T}}Rba} & \mathbf{1} & \underline{\mathbf{G}}_{\boldsymbol{\mathcal{T}}Rb} \end{bmatrix} & \textit{if } \mathrm{BiasType} = \mathrm{Var} \\ \underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R} & \textit{if } \mathrm{BiasType} = \mathrm{Fix} \end{cases}$$

where $\mathbf{L}$ *is a lower triangular matrix with positive diagonals, and:*

$$\mathbf{J} = \begin{cases} \begin{bmatrix} 1 & 0 & \mathbf{0}^T \\ 0 & -1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} & \textit{if } \mathrm{BiasType} = \mathrm{Var} \\ \mathbf{I} & \textit{if } \mathrm{BiasType} = \mathrm{Fix} \end{cases}$$

If $N_F = 0$ *or NoChol* $=$ *TRUE then for completeness* $\mathbf{L}$ *is defined to be an empty matrix.*

By default, $\mathbf{L}$ is assumed to be empty, NoChol $=$ FALSE and $N_R = 0$.

Basic Operations with Cholesky Factorisations

The following basic Cholesky operations will be needed. Firstly, given the column vector $\mathbf{z}$, it is possible to find $\mathbf{r}$ such that either $\mathbf{Lr} = \mathbf{z}$ or $\mathbf{L}^T\mathbf{r} = \mathbf{z}$. This may be done using *forward elimination* and *back substitution*, respectively, as described in algorithms 4.1-1 and 4.1-2 of [44]. As both algorithms are quadratic time algorithms, combining these operations allows one to perform fast matrix inversion. For convenience these algorithms are reproduced in appendix D (see algorithms 1 and 2).

Another necessary algorithm is the Cholesky rank-1 update algorithm. Given some matrix $\bar{\mathbf{M}} = \bar{\mathbf{Q}}\bar{\mathbf{Q}}^T$ where $\bar{\mathbf{Q}} = \mathrm{chol}\left(\bar{\mathbf{M}}\right)$ is known, it is possible to find $\mathbf{Q} = \mathrm{chol}\left(\mathbf{M}\right)$ such that $\mathbf{M} = \mathbf{Q}\mathbf{Q}^T = \bar{\mathbf{M}} + \mathbf{h}\mathbf{h}^T$ quickly using a rank-1 update algorithm. Appropriate algorithms may be found in the standard texts [44], [42].

8.3.4 Setup algorithm

In this section, I detail how the factorisation $\beth_{\boldsymbol{\tau}}$ is calculated, if necessary, upon entering the algorithm. Algorithms 3 ($\mathrm{InvFact}\left(\underline{\mathbf{H}}_{\boldsymbol{\tau}F}\right)$) and 4 ($\mathrm{CholFact}\left(\underline{\mathbf{H}}_{\boldsymbol{\tau}F}\right)$) are used to calculate $\beth_{\boldsymbol{\tau}}$ during the initialisation phase of the optimisation algorithm, and also to update $\beth_{\boldsymbol{\tau}}$ when the active set is modified. The algorithms themselves are very slight variants of standard algorithms from the literature (for example, [44]), and are therefore presented in appendix D without commentary.

Algorithm 3 is used if an inverse factorisation method is chosen, and algorithm 4 if a Cholesky factorisation is chosen (see [44], for example). Both algorithms share two important features, namely:

1. If, as a result of a constraint being either activated or deactivated, $\underline{\mathbf{H}}_{\boldsymbol{\tau}R}$ is improperly defined in such a way that $\underline{\mathbf{H}}_{\boldsymbol{\tau}FNB}$ is non-singular, the setup algorithm may be called to extend $\underline{\mathbf{H}}_{\boldsymbol{\tau}R}$ to its correct size.

2. If as a result of running the algorithm it is found that $N_F \neq N_R$ then $(\underline{s}_b, \underline{\mathbf{s}}_{\boldsymbol{\alpha}})$ in (8.13) (or $\underline{\mathbf{s}}_{\boldsymbol{\alpha}}$ in (8.18)) will already be, at least partially, calculated. Hence most of the computational cost involved in calculating a step using equation (8.13) (or (8.18)) is avoided.

Indeed, based on point 1 above, the setup algorithm may be thought of as a means maximising N_R. Any prior knowledge (in the form of a partial factorisation) is used to "kick-start" the algorithm, minimising computational cost. As the default active set definition upon entering the algorithm in a "cold-start" situation is for all variables α_i to be actively constrained at a lower bound (i.e. $N_F = 0$), the setup algorithm will rarely be required to increase N_R significantly, and hence contributes little to the computational cost of the algorithm.

Consider algorithm 3. If this algorithm terminates with $N_F \neq N_R$ then, as noted previously, the variable $(\underline{s}_b, \underline{\mathbf{s}}_{\boldsymbol{\alpha}})$ (or just $\underline{\mathbf{s}}_{\boldsymbol{\alpha}}$ for the fixed bias case) calculated most recently in algorithm 3 may be used directly during the subsequent iteration of the main algorithm when calculating the step using equation (8.13).

The analogous situation when using a Cholesky update method is not quite so simple. However, if BiasType = Var and algorithm 4 terminates with $N_F \neq N_R$ a significant computational cost saving may still be had when calculating $(\underline{s}_b, \underline{\mathbf{s}}_{\boldsymbol{\alpha}})$ by solving:

$$\begin{bmatrix} s_{\boldsymbol{\alpha} a} \\ s_b \\ \mathbf{s}_{\boldsymbol{\alpha} b} \end{bmatrix} = \text{BackwardSub}\left(\mathbf{L}^T, \mathbf{a}\right)$$

where $\mathbf{a}$ is as calculated during the final iteration of algorithm 4, and:

$$\underline{\mathbf{s}}_{\boldsymbol{\alpha}} = \begin{bmatrix} \underline{s}_{\boldsymbol{\alpha} a} \\ \underline{\mathbf{s}}_{\boldsymbol{\alpha} b} \end{bmatrix}$$

is the vectorial part of $(\underline{s}_b, \underline{\mathbf{s}}_{\boldsymbol{\alpha}})$ to be used directly in equation (8.13).

Likewise, if BiasType = Fix (the fixed bias case) one may solve:

$$\underline{\mathbf{s}}_{\boldsymbol{\alpha}} = \text{BackwardSub}\left(\mathbf{L}^T, \mathbf{a}\right)$$

to achieve the same end.

8.3.5 Fast Matrix Inversion

One of the key aims in defining the factorisation $\beth_{\boldsymbol{\mathcal{T}}}$ of the relevant part of the Hessian matrix $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$ was to be able to quickly calculate the vector $\mathbf{r}$, given $\mathbf{p}$, where $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}\mathbf{r} = \mathbf{p}$ and, by assumption, $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$ is non-singular and $N_R > 0$.

Inverse Factorisation

In the fixed bias case BiasType = Fix:

$$\mathbf{r} = \mathbf{V}\mathbf{p}$$

For the variable bias case BiasType = Var, define:

$$\mathbf{r} = \begin{bmatrix} \mathbf{r}_{\boldsymbol{\alpha}} \\ r_b \end{bmatrix}$$
$$\mathbf{p} = \begin{bmatrix} \mathbf{p}_{\boldsymbol{\alpha}} \\ p_b \end{bmatrix}$$

then:

$$\begin{bmatrix} r_b \\ \mathbf{r}_{\boldsymbol{\alpha}} \end{bmatrix} = \mathbf{V} \begin{bmatrix} p_b \\ \mathbf{p}_{\boldsymbol{\alpha}} \end{bmatrix}$$

Cholesky Factorisation

In the fixed bias case BiasType = Fix:

$$\mathbf{t} = \text{ForwardElim}\left(\mathbf{L}, \mathbf{p}\right) \tag{8.22}$$
$$\mathbf{r} = \text{BackwardSub}\left(\mathbf{L}^T, \mathbf{t}\right) \tag{8.23}$$

Likewise for the variable bias case BiasType = Var, if NoChol = FALSE, define:

$$\mathbf{r} = \begin{bmatrix} r_{\boldsymbol{\alpha}_a} \\ \mathbf{r}_{\boldsymbol{\alpha}_b} \\ r_b \end{bmatrix}$$
$$\mathbf{p} = \begin{bmatrix} p_{\boldsymbol{\alpha}_a} \\ \mathbf{p}_{\boldsymbol{\alpha}_b} \\ p_b \end{bmatrix}$$
$$\mathbf{t} = \begin{bmatrix} t_{\boldsymbol{\alpha}_a} \\ \mathbf{t}_{\boldsymbol{\alpha}_b} \\ t_b \end{bmatrix}$$

then:

$$\begin{bmatrix} t_{\boldsymbol{\alpha}_a} \\ t_b \\ \mathbf{t}_{\boldsymbol{\alpha}_b} \end{bmatrix} = \mathrm{ForwardElim}\left(\mathbf{L}, \begin{bmatrix} s_{\boldsymbol{\alpha}_a} \\ s_b \\ \mathbf{s}_{\boldsymbol{\alpha}_b} \end{bmatrix} \right) \tag{8.24}$$

$$\begin{bmatrix} r_{\boldsymbol{\alpha}_a} \\ r_b \\ \mathbf{r}_{\boldsymbol{\alpha}_b} \end{bmatrix} = \mathrm{BackwardSub}\left(\mathbf{L}^T, \begin{bmatrix} t_{\boldsymbol{\alpha}_a} \\ -t_b \\ \mathbf{t}_{\boldsymbol{\alpha}_b} \end{bmatrix} \right) \tag{8.25}$$

Otherwise, if NoChol = TRUE then, solving explicitly:

$$\begin{bmatrix} r_1 \\ r_2 \end{bmatrix} = \begin{bmatrix} p_2 \\ p_1 - \underline{G}_{\boldsymbol{\mathcal{T}}F1,1} p_2 \end{bmatrix}$$

8.3.6 Activating a Constraint

Consider the activation of a constraint, as in $\gimel^{(k+1)} = \mathrm{constrain}\left(p^{(k)}\right) \hat{\gimel}^{(k)}$. In this section I consider the affect of this operation on the factorisation $\daleth_{\boldsymbol{\mathcal{T}}}$. The following possibilities must be considered:

1. $p^{(k)} > N_R^{(k)} + 1$.
2. $p^{(k)} = N_R^{(k)} + 1$.
3. $p^{(k)} = N_R^{(k)} = N_F^{(k)} = 1$.
4. $p^{(k)} = N_R^{(k)} = 1$ and $N_F^{(k)} > 1$.
5. $p^{(k)} \leq N_R^{(k)}$ and $N_R^{(k)} > 1$.

Where the Hessian $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}F}$ must be singular in cases 1, 2 and 4, and may be singular in case 5. For the first 4 cases, the modification to $\daleth_{\boldsymbol{\mathcal{T}}}$ is as follows:

1: As neither $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}FN}$ nor $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}FNB}$ are affected, ${\daleth_{\boldsymbol{\mathcal{T}}}}^{(k+1)} = {\daleth_{\boldsymbol{\mathcal{T}}}}^{(k)}$.

2: While $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}FN}$ is not affected, $\underline{\underline{\mathbf{H}}}_{\boldsymbol{\mathcal{T}}FNB}$ is, and so it may be possible to increase N_R if $N_R^{(k)} \neq N_F^{(k)} - 1$. Hence if $N_R^{(k)} \neq N_F^{(k)} - 1$ then, after making the appropriate changes to $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$, algorithm 3 (or 4, depending on the update method used) is called to update $\daleth_{\boldsymbol{\mathcal{T}}}$ and maximise N_R.

3: After this operation, $\daleth_{\boldsymbol{\mathcal{T}}}$ will revert to its default form. Hence all matrices in $\daleth_{\boldsymbol{\mathcal{T}}}{}^{(k+1)}$ will be empty, $N_R^{(k+1)} = 0$ and NoChol = FALSE if a Cholesky update method is used.

4: $\daleth_{\boldsymbol{\mathcal{T}}}$ must be re-built from scratch after the appropriate changes have been made to $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$.

For case 5, the modification is dependent on the factorisation used, as will now be described.

Inverse update method

The inverse of $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$ at iteration k, $\mathbf{V}^{(k)}$, may be partitioned as follows:

$$\mathbf{V}^{(k)} = \begin{bmatrix} \mathbf{V}_a & \mathbf{v}_d & \mathbf{V}_b^T \\ \mathbf{v}_d^T & v_e & \mathbf{v}_f^T \\ \mathbf{V}_b & \mathbf{v}_f & \mathbf{V}_c \end{bmatrix}$$

where $\mathbf{V}_a \in \Re^{\left(p^{(k)}-1\right)\times\left(p^{(k)}-1\right)}$ if BiasType = Fix, or $\mathbf{V}_a \in \Re^{p^{(k)}\times p^{(k)}}$ if BiasType = Var.

If $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$ was non-singular prior the activation of the constraint (i.e. $N_R^{(k)} = N_F^{(k)}$) then:

$$\mathbf{V}^{(k+1)} = \begin{bmatrix} \mathbf{V}_a & \mathbf{V}_b^T \\ \mathbf{V}_b & \mathbf{V}_c \end{bmatrix} - \frac{1}{v_e} \begin{bmatrix} \mathbf{v}_d \\ \mathbf{v}_f \end{bmatrix} \begin{bmatrix} \mathbf{v}_d \\ \mathbf{v}_f \end{bmatrix}^T \tag{8.26}$$

Otherwise, equation (8.26) may still be used to calculate an interim form of $\daleth_{\boldsymbol{\mathcal{T}}}{}^{(k+1)}$, and then algorithm 3 may be called to increase (to a maximum) $N_R^{(k+1)}$.

According to theorem 8.8, equation (8.26) must be well defined, and, in particular, $v_e \neq 0$. Unfortunately, due to cumulative numerical errors, this may not be true in practice. Even if $v_e \neq 0$, if $|v_e| < \varepsilon$ it would be inadvisable to use equation (8.26), as the likely result would be large numerical errors in $\mathbf{V}$.

Our strategy [81] for dealing with this problem is to attempt to reduce $N_R^{(k)}$ by some minimal amount to ensure that $\mathbf{V}^{(k+1)}$ is non-singular to working precision (i.e. $|v_e| \geq \varepsilon$ in equation (8.26)).

Assume $p^{(k)} = N_R^{(k)}$ (if this is not true then, noting that $\mathbf{V}$ may be re-ordered in the same way as $\underline{\mathbf{H}}_{\boldsymbol{\tau}F}$, use the operation eswap $\left(N_C + p^{(k)}, N_C + N_R^{(k)}\right)$ to make it so). Hence $\mathbf{V}_c$ has zero size. If $N_R^{(k)} \geq 4$ the aim is to partition $\mathbf{V}$ as follows:

$$\mathbf{V}^{(k)} = \left[\begin{array}{cc} \mathbf{V}_\alpha & \mathbf{V}_\beta^T \\ \mathbf{V}_\beta & \mathbf{V}_\gamma \end{array} \right]$$

where $\mathbf{V}_\gamma \in \Re^{N_\gamma \times N_\gamma}$, and $2 \leq N_\gamma \leq \left\lfloor \frac{N_R^{(k)}}{2} \right\rfloor$ is as small as possible such that $\det(\mathbf{V}_\gamma) >= \varepsilon$. If such an N_γ exists then:

$$\mathbf{V}^{(k+1)} = \mathbf{V}_\alpha - \mathbf{V}_\beta^T \mathbf{V}_\gamma^{-1} \mathbf{V}_\beta$$

where $\mathbf{V}_\gamma^{-1}$ may be calculated using a simple variant of algorithm 3. Note that it is not computationally worthwhile to increase N_γ past the limit set above, as the computational cost of calculating $\mathbf{V}_\gamma^{-1}$ would then exceed the computational cost of re-calculating $\mathbf{V}^{(k+1)}$ from scratch using algorithm 3. If this method fails set $N_R := 0$, $\mathbf{V}$ empty and use algorithm 3 to calculate $\mathbf{V}^{(k+1)}$ from scratch.

From a computational perspective, this method is far from optimal if N_γ becomes large. However, in our experience cases where $N_\gamma > 2$ are extremely rare, and in any case indicates that a significant cumulative numerical error has occurred that is best rectified by re-calculating $\mathbf{V}$ from scratch.

Cholesky update method

Firstly, let suppose that $p^{(k)} \neq 1$. Hence $\mathbf{L}^{(k)}$ may be partitioned as follows:

$$\mathbf{L}^{(k)} = \left[\begin{array}{ccc} \mathbf{L}_a & \mathbf{0} & \mathbf{0} \\ \mathbf{l}_d^T & l_e & \mathbf{0}^T \\ \mathbf{L}_b & \mathbf{l}_f & \mathbf{L}_c \end{array} \right]$$

where $\mathbf{L}_a \in \Re^{\left(p^{(k)}-1\right)\times\left(p^{(k)}-1\right)}$ if BiasType = Fix and $\mathbf{L}_a \in \Re^{p^{(k)}\times p^{(k)}}$ if BiasType = Var. Assuming that $N_R^{(k+1)} = N_R^{(k)} - 1$:

$$\mathbf{L}^{(k+1)} := \begin{bmatrix} \mathbf{L}_a & \mathbf{0} \\ \mathbf{L}_b & \bar{\mathbf{L}}_c \end{bmatrix}$$

where:

$$\bar{\mathbf{L}}_c \bar{\mathbf{L}}_c^T = \mathbf{L}_c \mathbf{L}_c^T + \mathbf{l}_f \mathbf{l}_f^T$$

If $N_R^{(k)} \neq N_F^{(k)}$ then algorithm 4 is called subsequently to maximise N_R.

If, however, $p^{(k)} = 1$, then the only recourse is make $\mathbf{L}$ empty, $N_R = 0$ and NoChol = FALSE, make the necessary changes to $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$ etc., and then use algorithm 4 to make $\mathbf{L}^{(k+1)}$ from scratch. In extreme cases, it may be found that (as a result of numerical errors) that $N_R^{(k+1)} < N_R^{(k)} - 1$. However, unlike the similar problem when dealing with an inverse update, this may occur only if $p^{(k)} = 1$, and even then is, in our experience, quite rare.

8.3.7 De-activating a Constraint

When a constraint is de-activated, as in $\gimel^{(k+1)} = \mathrm{free}_X\left(q^{(k)}\right)\hat{\gimel}^{(k)}$, $\beth_{\boldsymbol{\mathcal{T}}}$ must be updated to reflect the modification to the active set. Clearly, if $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$ is singular and $N_F^{(k)} > 1$ then adding a row and column to the end of this matrix will not effect $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}R}$. Hence if $N_R^{(k)} \neq N_F^{(k)}$ and $N_F^{(k)} > 1$, ${\beth_{\boldsymbol{\mathcal{T}}}}^{(k+1)} = {\beth_{\boldsymbol{\mathcal{T}}}}^{(k)}$. Otherwise, the re-entrant properties of algorithms 3 and 4 may be used to advantage by simply updating $\underline{\mathbf{H}}_{\boldsymbol{\mathcal{T}}F}$ etc. appropriately and calling the relevant factorisation algorithm, 3 or 4.

8.3.8 Optimisations

Computational cost optimisation

In theorem 8.3 it was shown if $f^{(1)} \neq 0$ then there exists some l such that $f^{(l+n)} = 0$ for all $n \geq 0$. Furthermore, for most of these iterations, if $N_F^{(l+n)} > 0$ then $\underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}F}^{(l+n)}$

will have the form:

$$\underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}F}^{(l+n)} = \left[\begin{array}{c} \mathbf{0} \\ \underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}Fnz}^{(l+n)} \end{array} \right]$$

where $\underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}Fnz}^{(l+n)} \in \Re^{N_{Fnz}^{(l+n)}}$. Indeed, in most cases (all if theorem 8.8 holds true in spite of numerical errors), $N_{Fnz}^{(l+n)} = 1$. Using this fact, it is possible to significantly accelerate the calculation of equation (8.11) using $\beth_{\boldsymbol{\mathcal{T}}}$.

First, suppose an inverse factorisation is used. In this case, if $f^{(k)} = 0$ and BiasType = Var, (8.11) reduces to:

$$\left[\begin{array}{c} \Delta b^{(k)} \\ \Delta \underline{\boldsymbol{\alpha}}_F^{(k)} \end{array} \right] = -\mathbf{V}_{nz} \underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}Fnz}^{(k)}$$

where $\mathbf{V}_{nz} \in \Re^{\left(N_F^{(k)}+1\right) \times N_{Fnz}^{(k)}}$ and:

$$\mathbf{V} = \left[\begin{array}{cc} \mathbf{V}_z & \mathbf{V}_{nz} \end{array} \right]$$

Similarly, if $f^{(k)} = 0$ and BiasType = Fix, (8.11) reduces to:

$$\Delta \underline{\boldsymbol{\alpha}}_F^{(k)} = -\mathbf{V}_{nz} \underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}Fnz}^{(k)}$$

where $\mathbf{V}_{nz} \in \Re^{N_F^{(k)} \times N_{Fnz}^{(k)}}$ and:

$$\mathbf{V} = \left[\begin{array}{cc} \mathbf{V}_z & \mathbf{V}_{nz} \end{array} \right]$$

Using a Cholesky factorisation then, if $f^{(k)} = 0$, BiasType = Fix, NoChol = FALSE, $N_F^{(k)} > 1$ and $N_{Fnz}^{(k)} < N_F^{(k)}$, (8.22) and (8.23) can be replaced by:

$$\mathbf{t}_{nz} = \text{ForwardElim}\left(\mathbf{L}_{nz}, \underline{\mathbf{e}}_{\boldsymbol{\mathcal{T}}Fnz}^{(k)}\right) \quad (8.27)$$

$$\mathbf{r} = \text{BackwardSub}\left(\mathbf{L}^T, \left[\begin{array}{c} \mathbf{0} \\ \mathbf{t}_{nz} \end{array} \right]\right) \quad (8.28)$$

where $\mathbf{L}_{nz} \in \Re^{N_{Fnz}^{(k)} \times N_{Fnz}^{(k)}}$ and:

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_z & \mathbf{0} \\ \mathbf{L}_{nzz} & \mathbf{L}_{nz} \end{bmatrix}$$

The variable bias case is much the same, except that (8.28) is replaced by:

$$\begin{bmatrix} r_{\boldsymbol{\alpha} a} \\ r_b \\ \mathbf{r}_{\boldsymbol{\alpha} b} \end{bmatrix} = \text{BackwardSub}\left(\mathbf{L}^T, \begin{bmatrix} \mathbf{0} \\ \mathbf{t}_{nz} \end{bmatrix} \right) \tag{8.29}$$

Another area where some computational cost savings may be made is the calculation of $\left(\Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} U}^{(k)}, \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} L}^{(k)} \right)$ when $\underline{\mathbf{H}}_{\boldsymbol{\tau} F}$ is singular. As was shown in equation (8.13), $\Delta \underline{\boldsymbol{\alpha}}_{FS}^{(k)} = \mathbf{0}$ in this case. Extending the notation in the obvious manner, it can be seen that equation (8.1) may be simplified to:

$$\begin{bmatrix} \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} U}^{(k)} \\ \Delta \underline{\mathbf{e}}_{\boldsymbol{\tau} L}^{(k)} \end{bmatrix} = \begin{bmatrix} \mathbf{1} & \underline{\mathbf{G}}_{\boldsymbol{\tau} UFN} & \underline{\mathbf{g}}_{\boldsymbol{\tau} UFB} \\ \mathbf{1} & \underline{\mathbf{G}}_{\boldsymbol{\tau} LFN} & \underline{\mathbf{g}}_{\boldsymbol{\tau} LFB} \end{bmatrix} \begin{bmatrix} \Delta b^{(k)} \\ \Delta \underline{\boldsymbol{\alpha}}_{FN}^{(k)} \\ \Delta \underline{\alpha}_{FB}^{(k)} \end{bmatrix}$$

Memory usage optimisation

As our algorithm stores $\mathbf{G}_{\boldsymbol{\tau}} \in \Re^{N \times N}$ in full and also any matrices associated with the factorisation $\beth_{\boldsymbol{\tau}}$, it will naturally use significant amounts of memory if our training set size N is excessively large. Hence, the algorithm we have described is intended mainly for problems where the amount of training data is small to moderate. Because of this, it is important to consider issues of memory use, and how it may be minimised.

It will be noted that all matrices used in our algorithm are either lower triangular ($\mathbf{L}$, as used in the Cholesky update method) or symmetric (the large matrix $\mathbf{G}_{\boldsymbol{\tau}}$, and also $\mathbf{V}$, as used in the inverse update method). To take advantage of this symmetry, in [81] we stored all of these matrices in lower triangular form, thus reducing matrix memory usage from $N^2 + O(N)$ (assuming $N_R \ll N$) to $\frac{1}{2}N^2 + O(N)$.

8.3.9 Comparative Merits

Inverse update method

The advantages and disadvantages of the inverse update method are:

- Advantages:
    1. Speed - updating the inverse Hessian is significantly faster than calculating it from scratch.
    2. Simplicity - the algorithms to form and update the factorisation are simpler than their Cholesky update counterparts.
- Disadvantages:
    1. Numerical stability - as was seen in section 8.3.6, numerical errors can lead to significant inaccuracies in our inverse factorisation $\mathbf{V}$ and, consequently, the "optimal" solution.

Cholesky update method

The advantages and disadvantages of the Cholesky update method are:

- Advantages:
    1. Speed - calculating the step is significantly faster than calculating from scratch.
    2. Numerical stability - while it is still possible that, due to numerical errors, we may have problems with previously known non-singular matrices "appearing" non-singular as a result of the de-activation of a constraint, the likelihood of such an event is much lower when using a Cholesky update method than when using an inverse update method.
- Disadvantages:
    1. Complexity - the Cholesky update method is significantly more complex than the inverse update method.

Computational complexity comparisons

Consider the computational cost of an iteration. This may be split into factorisation dependent and factorisation independent components, where "factorisation" refers to either the inverse or Cholesky approach. The factorisation independent component of each iteration is the cost of calculating $\Delta\underline{\mathbf{e}}_U^{(k)}$ and $\Delta\underline{\mathbf{e}}_L^{(k)}$, as well as performing the update. This operation takes in the order of $2N_F(N_U + N_L) + 2N + 2N_F$ flops (a flop is defined here as a floating point multiplication, addition or square root). The factorisation dependent cost of each iteration is the cost of calculating $\Delta\underline{\boldsymbol{\alpha}}_F^{(k)}$ and $\Delta b^{(k)}$ and also the cost of updating the factorisation whenever a constraint is activated or de-activated.

Let us consider the factorisation dependent cost in more detail. For simplicity, assume $N_{Fnz} = 1$ (as is most often the case), and also that the Hessian is always non-singular. With the exception of the final step, each such step calculation will be followed by a constraint activation or de-activation. Hence it makes sense to look at the combined step plus constraint activation/de-activation cost.

For a step followed by a constraint activation, the computational cost of the inverse update method is approximately $3N_F + \left(N - p^{(k)}\right)^2$ flops, compared to $N_F^2 + 4\left(N - p^{(k)}\right)^2$ flops for the Cholesky method. Clearly, the inverse update method is significantly faster here. For a step followed by a constraint de-activation, however, the computational cost of the inverse update method is approximately $3N_F^2$ flops, which makes it significantly slower than the Cholesky method, which takes approximately N_F^2 flops.

From this, it may be observed that neither algorithm has a significant computational advantage over the other. In [81] we found the Cholesky method to be marginally faster than the inverse update method. However, it is known that usually $N_F \ll N$, so the factorisation dependent cost tends to be swamped by the factorisation independent cost. Therefore, in most cases, the computational cost issues of the two factorisation methods are likely to be less important than the complexity versus numerical stability trade-off.

8.4 Incremental Learning and Forgetting

It is straightforward to apply the algorithm presented here to incremental learning and forgetting. Suppose that $\bar{\gimel} = \left\{\bar{\digamma}, \bar{\maltese}, \bar{\mathbf{T}}\right\}$ (where $\bar{\mathbf{T}} = \left(\bar{\boldsymbol{\vartheta}}, \mathrm{Bias\bar{T}ype}, \bar{b}_{\mathrm{fix}}\right)$, $\bar{\maltese} = \left\{\bar{\boldsymbol{\alpha}}, \bar{b}, \bar{\boldsymbol{\tau}}, \bar{\mathbf{m}}\right\}$ and $\bar{\digamma} = \left\{\bar{E}, \bar{\mathbf{e}}_{\boldsymbol{\mathcal{T}}}, \bar{f}, \bar{\mathbf{G}}_{\boldsymbol{\mathcal{T}}}, \bar{\daleth}_{\boldsymbol{\mathcal{T}}}\right\}$) is the abstract state of some SVM, which may or may not be optimal but must be in standard form. Incremental learning then involves incorporating the additional training points $\hat{\boldsymbol{\vartheta}} = \left(\hat{\vartheta}_1, \hat{\vartheta}_2, \ldots, \hat{\vartheta}_{\hat{N}}\right)$ into the existing abstract training set $\bar{\mathbf{T}} = \left(\bar{\boldsymbol{\vartheta}}, \mathrm{Bias\bar{T}ype}, \bar{b}_{\mathrm{fix}}\right)$, where $\bar{\boldsymbol{\vartheta}} = \left(\bar{\vartheta}_1, \bar{\vartheta}_2, \ldots, \bar{\vartheta}_{\bar{N}}\right)$, and then suitably extending $\bar{\digamma}$ and $\bar{\maltese}$ to give a new machine $\gimel = \{\digamma, \maltese, \mathbf{T}\}$, which is optimal.

The approach taken here is to split this process into two stages. Firstly, $\bar{\gimel}$ is extended to some intermediate $\bar{\bar{\gimel}}$ which includes the new training points and is in standard form, but is not optimal. Define:

$$\begin{aligned}
\bar{\bar{\gimel}} &= \left\{\bar{\bar{\digamma}}, \bar{\bar{\maltese}}, \bar{\bar{\mathbf{T}}}\right\} \\
\bar{\bar{\mathbf{T}}} &= \left\{\bar{\bar{\boldsymbol{\vartheta}}}, \mathrm{Bias\bar{T}ype}, \bar{b}_{\mathrm{fix}}\right\} \\
\bar{\bar{\maltese}} &= \left\{\begin{bmatrix} \bar{\boldsymbol{\alpha}} \\ \mathbf{0} \end{bmatrix}, \bar{b}, \begin{bmatrix} \bar{\boldsymbol{\tau}} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \bar{\mathbf{m}} + \mathbf{1}\hat{N} \\ \hat{\mathbf{m}} \end{bmatrix}\right\} \\
\bar{\bar{\digamma}} &= \left\{\bar{E}, \begin{bmatrix} \bar{\mathbf{e}}_{\boldsymbol{\mathcal{T}}} \\ \hat{\mathbf{e}}_{\boldsymbol{\mathcal{T}}} \end{bmatrix}, \bar{f}, \begin{bmatrix} \bar{\mathbf{G}}_{\boldsymbol{\mathcal{T}}} & \hat{\mathbf{G}}_{\boldsymbol{\mathcal{T}}}^T \\ \hat{\mathbf{G}}_{\boldsymbol{\mathcal{T}}} & \hat{\hat{\mathbf{G}}}_{\boldsymbol{\mathcal{T}}} \end{bmatrix}, \bar{\daleth}_{\boldsymbol{\mathcal{T}}}\right\} \\
\bar{\bar{\boldsymbol{\vartheta}}} &= \left(\bar{\vartheta}_1, \bar{\vartheta}_2, \ldots, \bar{\vartheta}_{\bar{N}}, \hat{\vartheta}_1, \hat{\vartheta}_2, \ldots, \hat{\vartheta}_{\hat{N}}\right)
\end{aligned}$$

where:

$$\begin{aligned}
\hat{\mathbf{e}} &= \hat{\mathbf{G}}_{\boldsymbol{\mathcal{T}}}\bar{\boldsymbol{\alpha}} + \mathbf{1}\bar{b} - \hat{\mathbf{z}} \\
\hat{m}_i &= i \\
\hat{G}_{\boldsymbol{\mathcal{T}}i,j} &= K\left(\hat{\mathbf{x}}_i, \mathbf{x}_j\right) \\
\hat{\hat{G}}_{\boldsymbol{\mathcal{T}}i,j} &= K\left(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j\right)
\end{aligned}$$

Note that the new abstract training points are all actively constrained to zero in $\bar{\bar{\gimel}}$, and that $\bar{\bar{\gimel}}$ is in standard form. Also, there is a small problem here. Technically speaking, as $\boldsymbol{\gamma}$, $\boldsymbol{\gamma}^*$, $\mathbf{h}$ and $\mathbf{v}$ are all proportional to $\frac{C}{N}$, these should be modified appropriately here. However, this is not practical at this point. If $\bar{N} \gg \hat{N}$ this will not be a problem, but if this is not true then it may be necessary to rescale $\boldsymbol{\gamma}$, $\boldsymbol{\gamma}^*$, $\mathbf{h}$ and $\mathbf{v}$ as described in the next section.

If $\bar{\bar{\gimel}}$ satisfies the KKT conditions (8.3) then $\bar{\bar{\gimel}}$ is optimal, so set $\gimel = \bar{\bar{\gimel}}$. Otherwise, the algorithm may be re-entered with $\gimel^{(1)} = \bar{\bar{\gimel}}$. Note that it is not necessary to re-calculate the factorisation $\daleth^{(1)}$, as the old factorisation is correct already.

Forgetting may be implemented in a similar manner. Simply remove the ϑ_i's corresponding to the training points that are to be removed, re-calculate f, E and $\mathbf{e}_{\boldsymbol{\mathcal{T}}}$ appropriately, and re-enter the optimisation algorithm if the KKT conditions are no longer satisfied. Note that this will only happen if support vectors are removed.

During forgetting, the inverse or factorisation $\daleth^{(1)}$ may be altered due to the removal of free variables. The effect of the factorisation is the same as for the activation of a constraint on that variable, and so the factorisation can be updated in the same manner as would be used for constraint activation.

8.5 Constraint Parameter Variation

When designing and training an SVM, it is necessary to select a number of parameters, including C, the kernel function K and the width of the ϵ-insensitive region. Traditionally, this has been done by trial-and-error for kernel selection and using a search method for other parameters. More recently, [22] has given an algorithm for automatically selecting the kernel parameters in order to minimise an upper bound on the generalisation error. In either case, one must optimise the SVM for a large number of different choices of parameter. Using a batch re-solve method tends to make the process extremely slow.

In [80], [81], we demonstrated the efficacy of an alternative approach to this problem, namely incremental training. Suppose that we have optimised our SVM for a given set of parameters, and want to repeat the process for a slightly different set of parameters. It is reasonable to expect that, assuming the difference between the parameters is sufficiently small, the solution for the new parameters will be close to the solution for the old parameter set. As we demonstrated, it is possible to take advantage of this feature in a warm-start approach.

8.5.1 $\boldsymbol{\rho}$, $\boldsymbol{\rho}^*$, z and b_{fix} Variation

These parameters are the easiest to adjust. One simply needs to adjust f, E and $\mathbf{e}_{\boldsymbol{\mathcal{T}}}$ appropriately to allow for any change and then re-enter the algorithm if the optimality conditions are no longer met. For example, if we wish to scale $\boldsymbol{\rho}$ and $\boldsymbol{\rho}^*$ by some scale factor κ then:

$$\begin{aligned}
\mathbf{e}_{\boldsymbol{\mathcal{T}}} &:= \mathbf{e}_{\boldsymbol{\mathcal{T}}} + (\kappa - 1)\,\mathrm{sgn}\,(\boldsymbol{\tau})\,\boldsymbol{\rho}^{(*)}_{\boldsymbol{\mathcal{T}}} \\
f &:= f \\
E &:= E
\end{aligned}$$

8.5.2 v and h Variation

The bounds are also not too difficult to adjust - one simply adjusts f, E and $\mathbf{e}_{\boldsymbol{\mathcal{T}}}$ appropriately and re-enters the algorithm if the KKT conditions are no longer met. However, some care must be taken if either there are points actively constrained at an upper or lower (non-zero) bound which is changing, or the change in the bounds will cause one of the free variables to move outside one of these bounds.

In the first case (i.e. when a point is actively constrained at a non-zero boundary which is being changed) it is necessary to take this into account when adjusting f, E or $\mathbf{e}_{\boldsymbol{\mathcal{T}}}$. In the latter case, the variation may need to be done in several stages. For each stage, the bounds will be changed as far as possible, and then any free variables lying on the new bound must be actively constrained using the appropriate pivoting operations. In practice, it is uncommon to require more then two such stages.

8.5.3 γ, γ^*, μ, μ^*, K and $\mathbf{x}_i$ Variation

As for other parameters, when adjusting these parameters is necessary to first adjust f, E and $\mathbf{e}_{\mathcal{T}}$ appropriately, and then re-enter the algorithm if the KKT conditions are no longer met. However, there is an additional complication here. Adjustment of these parameters will also affect $\mathbf{G}_{\mathcal{T}}$ (and appropriate adjustments must be made) and, potentially, the Hessian factorisation $\beth_{\mathcal{T}}$.

Unfortunately, the cost of re-factorising the Hessian is likely to be significant, especially if the size of the support vector set is large. So, while there is no in-principle problem with adjusting these parameters using this method, the computational cost of such operations may be prohibitive.

8.5.4 Connection with Standard SVM Parameter Variation

The connection between the standard SVM parameters (like C, ν etc) and the abstract parameters given here will, of course, be dependent on the SVM formulation being used. However, using the information given in sections 7.6, 7.7 and 7.8, it should be straightforward to see how adjustments to these parameters are to be mapped to adjustment of the abstract parameters in each case.

8.6 Implementation Details

This algorithm, and Platt's SMO algorithm [66], were both implemented in C++,[3] and experiments were carried out on a 1GHz Pentium III with 512MB running Windows 2000, using DJGPP for code compilation.

To ensure the comparison was based on algorithmic efficiency rather than the efficiency of the implementation, ability of the compiler to optimise the code and the speed of the computer on which the experiment was done, computational cost has been measured explicitly in terms of flops, where one flop is defined here as one floating point addition, multiplication or square root.

An accuracy of $\epsilon = 10^{-2}$ was chosen. The kernel function used for all experiments

[3]code available at http://www2.ee.mu.oz.au/pgrad/apsh/svm/

was the quadratic kernel $K(\mathbf{x}, \mathbf{y}) = \left(1 + \frac{1}{d_L}\mathbf{x}^T\mathbf{y}\right)^2$. Some of the following results were first published in [81].

8.7 Experimental Methodology

The aim here is to investigate the advantages and disadvantages of using an incremental training algorithm instead of a batch training algorithm in those situations where a partial solution (either in the form of an SVM trained on some subset of the complete training set, or an SVM trained using different parameters) is available. As such, the accuracy of the resultant SVM (which is essentially independent of the training algorithm in any case) is of secondary importance, with the computational cost of updating (or re-training) the SVM being our primary concern.

When discussing the computational cost of updating an SVM, either through incremental learning or incremental parameter variation, it is assumed that the "old" solution is given a-priori. So the computational cost of updating this SVM will include only the cost of modifying this old solution in an appropriate manner. Specifically, when investigating incremental training, one is concerned with the cost of updating an SVM with a *base* training set of N training pairs, to which M additional training pairs have been added.

Three incremental training experiments are described here. In the first, the problem at hand is relatively simple, and a high degree of accuracy may be achieved using only a small fraction of the complete training set (UCI mushroom toxicity dataset [10]). In this case, an error rate of less than 5% (tested using 800 vectors not contained in the training set) was achieved using a training set of less than 100 vectors out of a possible 7000. In this way it is possible to investigate the "steady state" computational cost where the decision surface is essentially static, suffering only occasional perturbations when new training data is added to the training set. The accuracy of the SVM for different training set sizes is shown in figure 8.2.

In the second experiment, the training problem is significantly more difficult (the adult dataset from the delve repository[4]). In this case the lowest achievable error

[4]see http://www.cs.toronto.edu/~delve/data/datasets.html

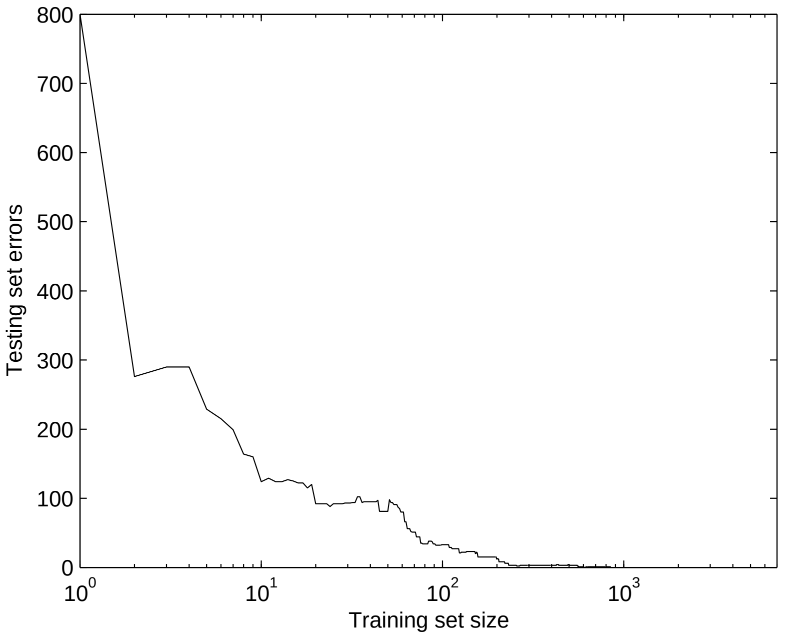


Figure 8.2: Testing set error versus training set size - mushroom dataset

rate was approximately 20%, as shown by figure 8.3 (measured using 150 vectors not contained in the training set). This allowed investigation of the properties of the algorithm when the solution changes significantly with each increment.

The third experiment considers the computational cost of incrementally varying the constraint parameter C, as may be required to find the optimal value for this constraint parameter. The dataset used here is the smaller adult dataset.

8.8 Experimental results

8.8.1 Incremental Training

Figure 8.4 is typical of the computational cost for the SMO algorithm (adult dataset). The dashed line in this figure shows the computational cost of batch optimising the SVM using an SMO algorithm for different training set sizes, and the solid line the computational cost of incrementally adding $M = 1$ training points to an existing SVM (optimised for an existing training set size) using the SMO algorithm.[5]

[5]Incremental SMO learning may be achieved by simply keeping the old $\boldsymbol{\alpha}$ value, setting $\hat{\boldsymbol{\alpha}} = \mathbf{0}$, and starting at this value. This method is not applicable to decremental learning or constraint parameter variation, as in general we must start the SMO algorithm with $f = 0$.

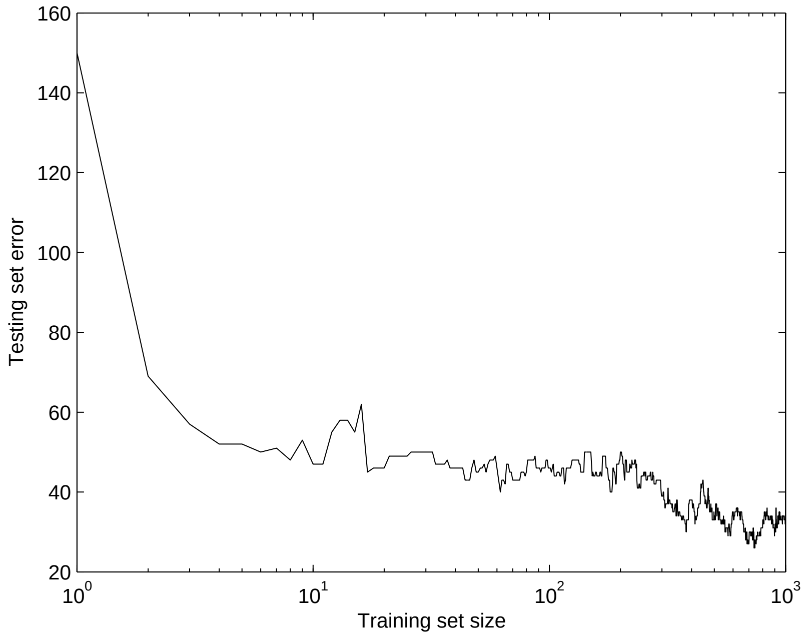


Figure 8.3: Testing set error versus training set size - adult dataset

As can be seen from this graph, there is no advantage to be had using incremental SMO methods. Figure 8.7 shows the same result for the mushroom toxicity dataset using the same increment size ($M = 1$). For both datasets, the active set algorithm was significantly faster than SMO (compare, for example, figures 8.4 and 8.11). However, it must be born in mind that the datasets considered in the present paper are relatively small, and the SMO algorithm is optimised for much larger datasets where it becomes impractical to store the entire Hessian in memory. On average, it was found that the number of flops taken by the active set algorithm to train the SVM (batch method) was usually around $\frac{1}{20}^{\text{th}}$ of the number of flops required by the SMO to complete the same problem.

Figure 8.11 shows a comparison between batch and incremental cost using the active set algorithm (adult dataset) for an increment size of $M = 1$. In this figure, the *base cost* is the cost of extending $\mathbf{G}_{\boldsymbol{\mathcal{T}}}$, $\mathbf{e}_{\boldsymbol{\mathcal{T}}}$ etc. and thus represents the minimum possible incremental update cost for an increment of size $M = 1$ given an existing training set of the size indicated on the x-axis. As shown by the graph, the computational cost for a significant proportion (52%) of updates is exactly equal to the base cost, which indicates that the optimal SVM has not been changed when the new training data was added. Even when the incremental cost exceeds this base

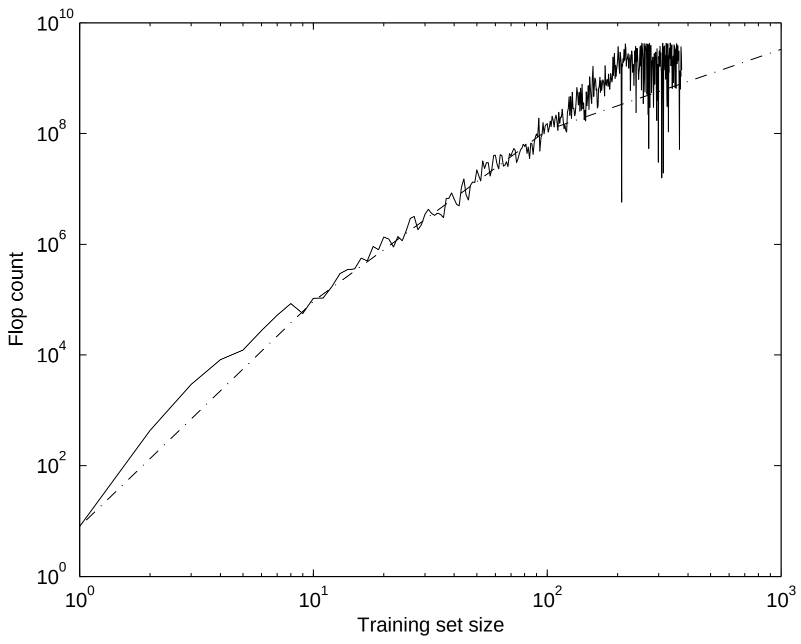


Figure 8.4: Flop count vs. training set size (adult dataset) - SMO algorithm (dashed line batch, solid line incremental ($M = 1$)).

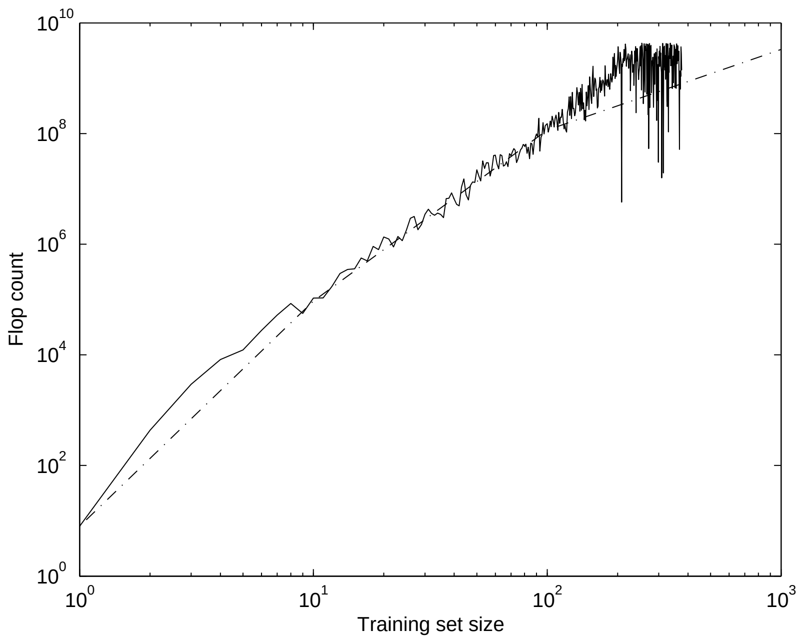


Figure 8.5: Flop count vs. training set size (adult dataset) - SMO algorithm (dashed line batch, solid line incremental ($M = 10$)).

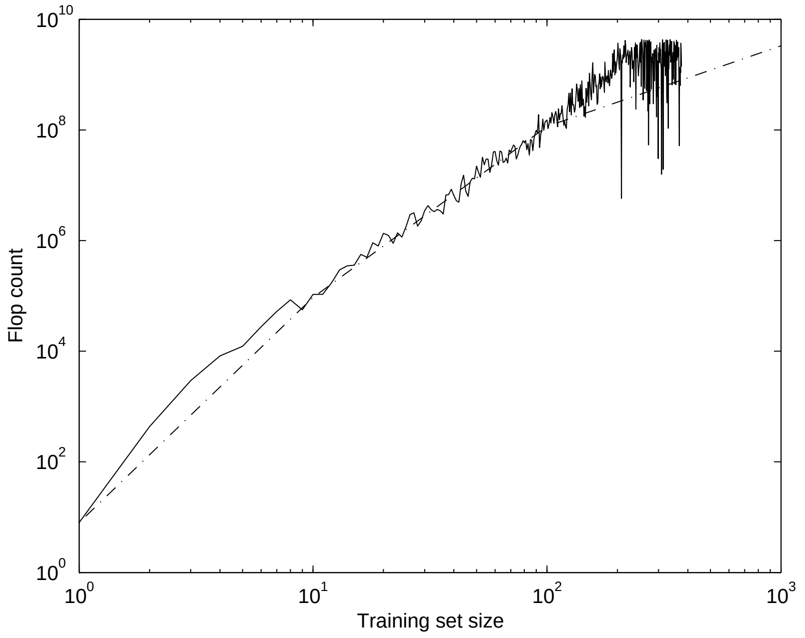


Figure 8.6: Flop count vs. training set size (adult dataset) - SMO algorithm (dashed line batch, solid line incremental ($M = 100$)).

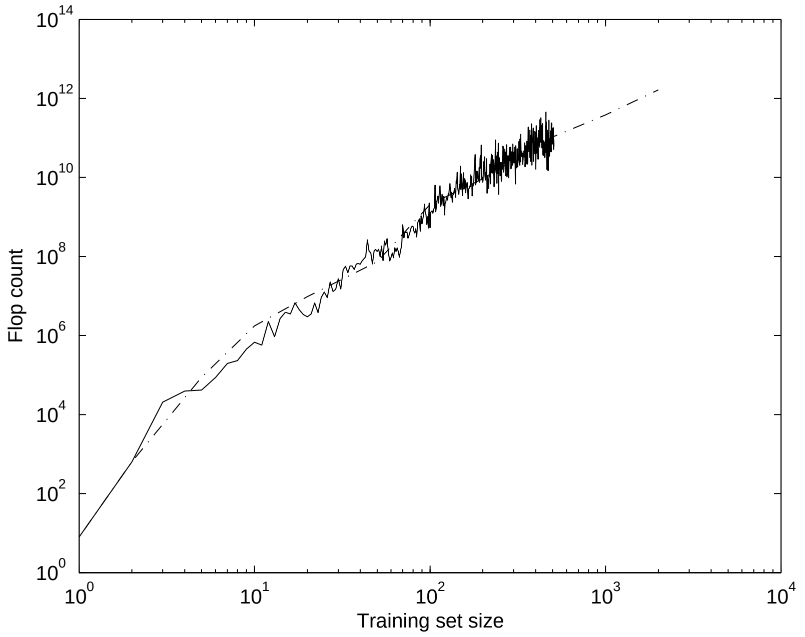


Figure 8.7: Flop count vs. training set size (mushroom dataset) - SMO algorithm (dashed line batch, solid line incremental ($M = 1$)).

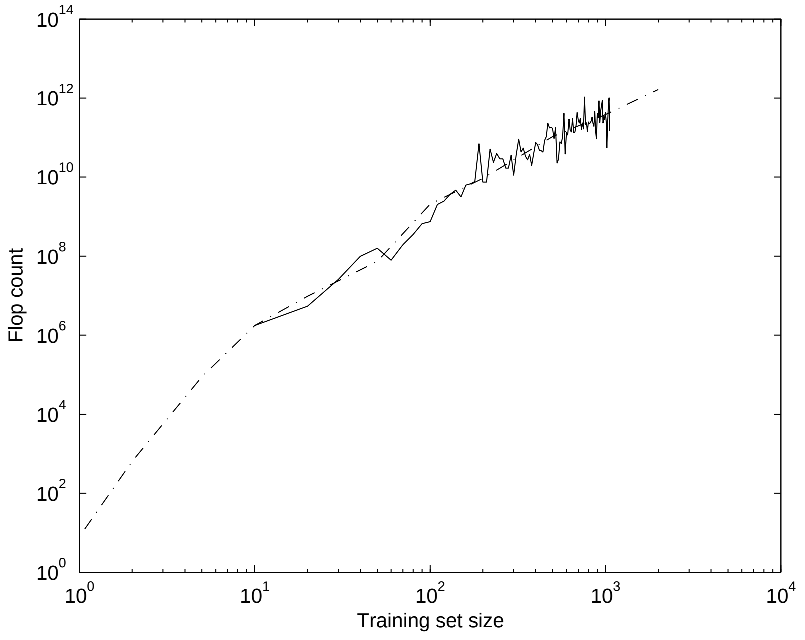


Figure 8.8: Flop count vs. training set size (mushroom dataset) - SMO algorithm (dashed line batch, solid line incremental ($M = 10$)).

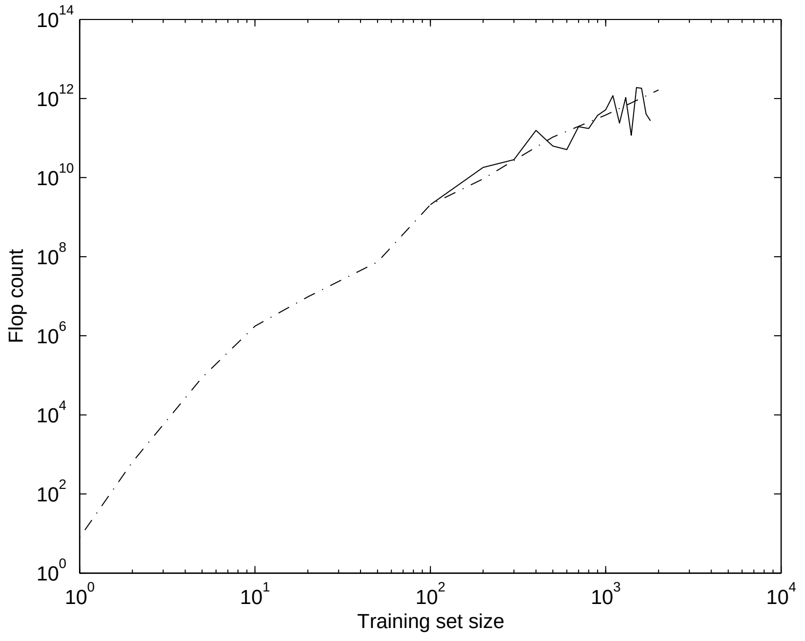


Figure 8.9: Flop count vs. training set size (mushroom dataset) - SMO algorithm (dashed line batch, solid line incremental ($M = 100$)).

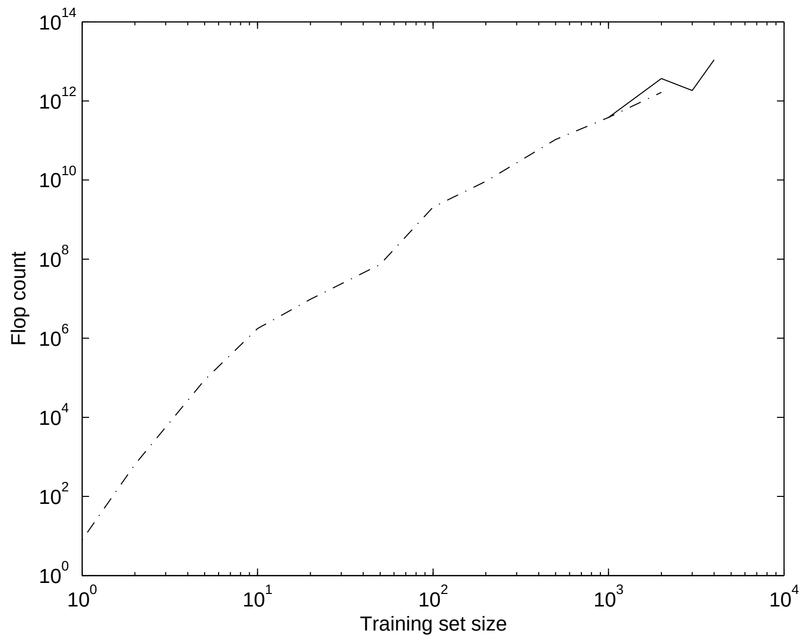


Figure 8.10: Flop count vs. training set size (mushroom dataset) - SMO algorithm (dashed line batch, solid line incremental ($M = 1000$)).

cost, the incremental computational cost is almost always significantly less (and always, for the datasets used here, at least slightly less) than the comparable batch optimisation cost. On average, it was found that the incremental update method was 870 times faster than the comparable batch method for this example. If only those increments which modified the SVM in a non-trivial manner were considered, this dropped to 37 times faster on average, which is still a significant improvement.

For larger increment sizes (for example, $M = 100$ for the adult dataset is shown in figure 8.16, $M = 1000$ for mushroom toxicity dataset in figure 8.17), the incremental cost is more likely to exceed the base cost. However, it was still found that the incremental method was faster than the batch method in all cases (for the datasets in question).

For this first part of the experiment, no significant differences were found between the performance of the two proposed factorisation methods (inverse and Cholesky), either in computational cost or the optimal SVM found. However, as will be seen in the following section, this does not indicate that there are no differences in general.

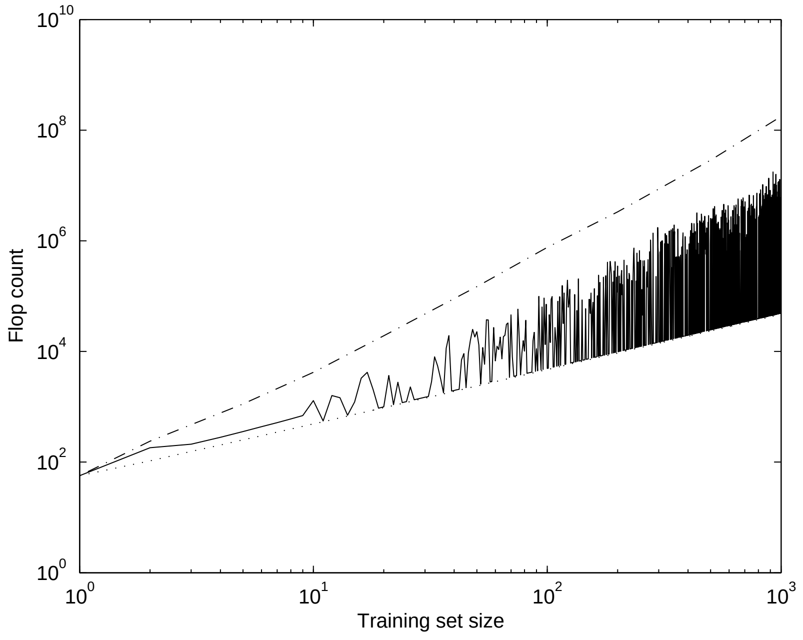


Figure 8.11: Flop count vs. training set size (adult dataset) - active set algorithm (Cholesky factorisation) (in order of increasing magnitude: base cost, incremental cost ($M = 1$) and batch cost).

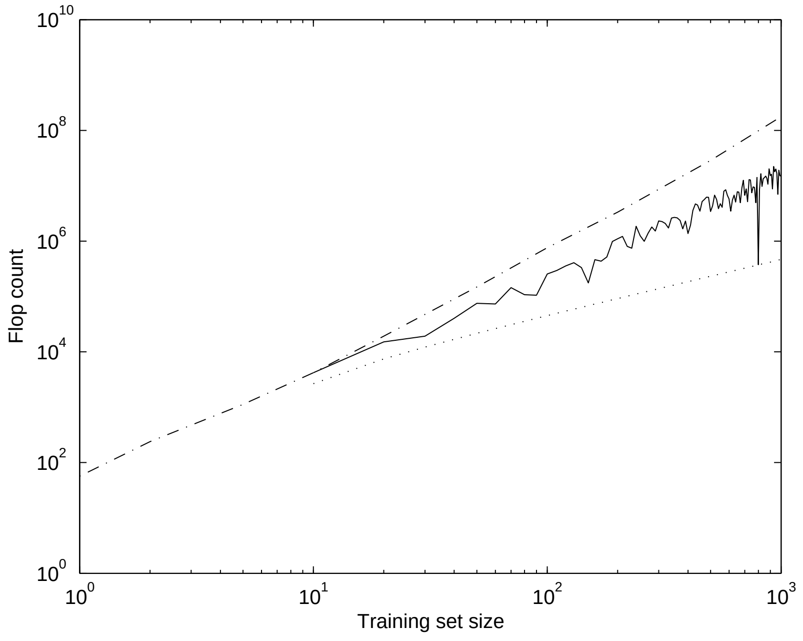


Figure 8.12: Flop count vs. training set size (adult dataset) - active set algorithm (Cholesky factorisation) (in order of increasing magnitude: base cost, incremental cost ($M = 10$) and batch cost).

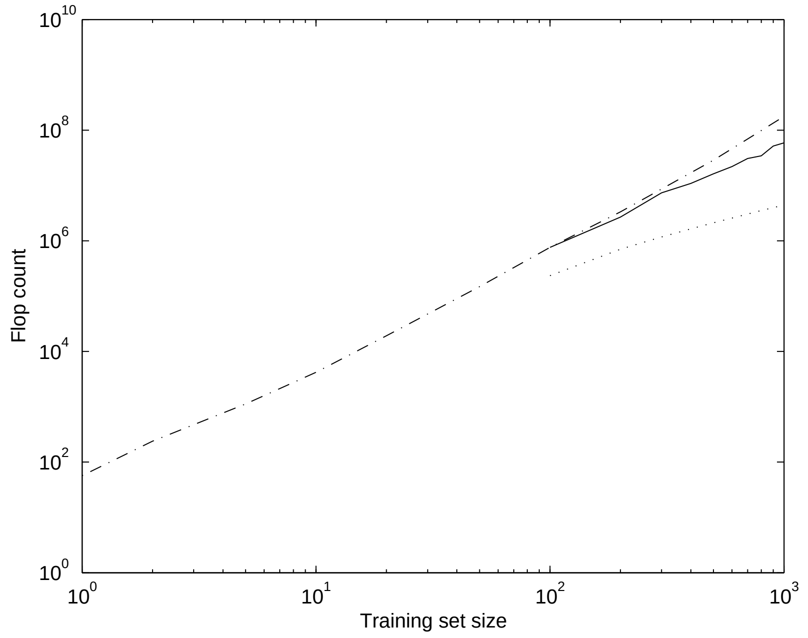


Figure 8.13: Flop count vs. training set size (adult dataset) - active set algorithm (Cholesky factorisation) (in order of increasing magnitude: base cost, incremental cost ($M = 100$) and batch cost).

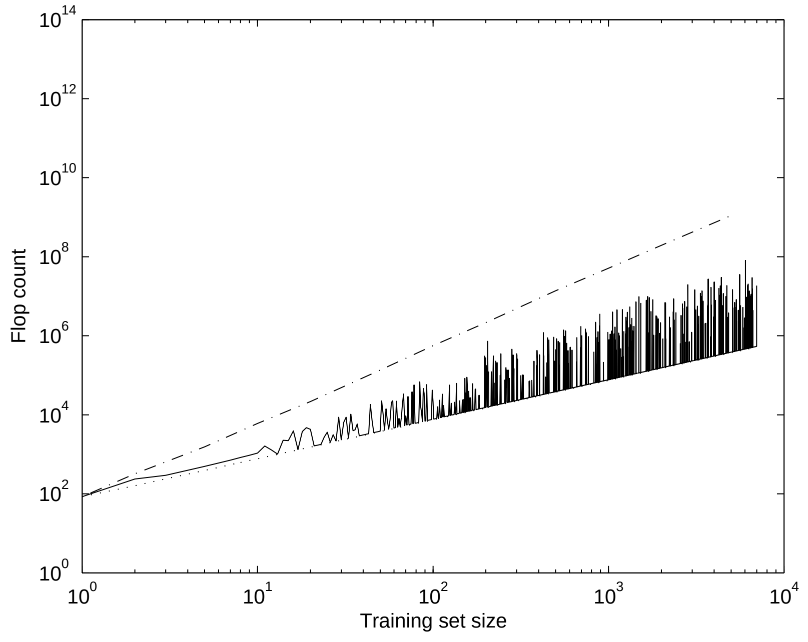


Figure 8.14: Flop count vs. training set size (mushroom dataset) - active set algorithm (Cholesky factorisation) (in order of increasing magnitude: base cost, incremental cost ($M = 1$) and batch cost).

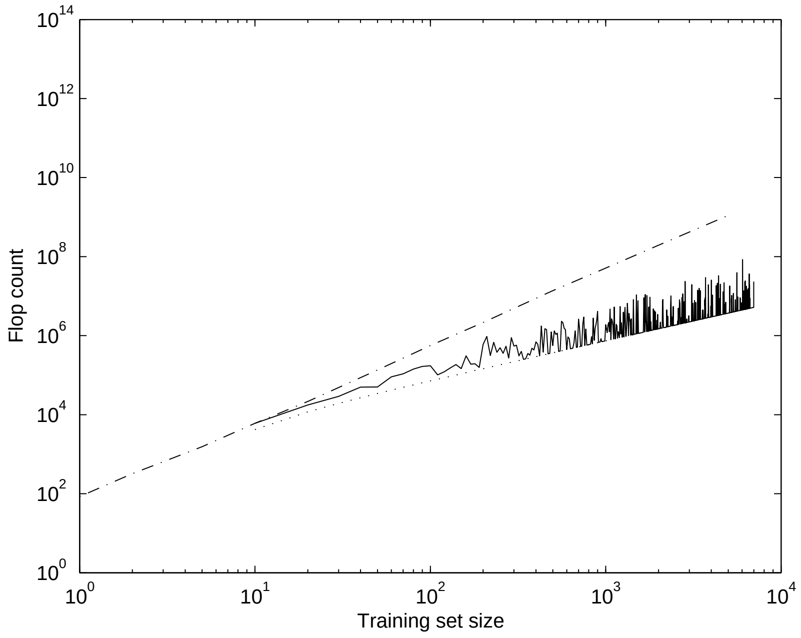


Figure 8.15: Flop count vs. training set size (mushroom dataset) - active set algorithm (Cholesky factorisation) (in order of increasing magnitude: base cost, incremental cost ($M = 10$) and batch cost).

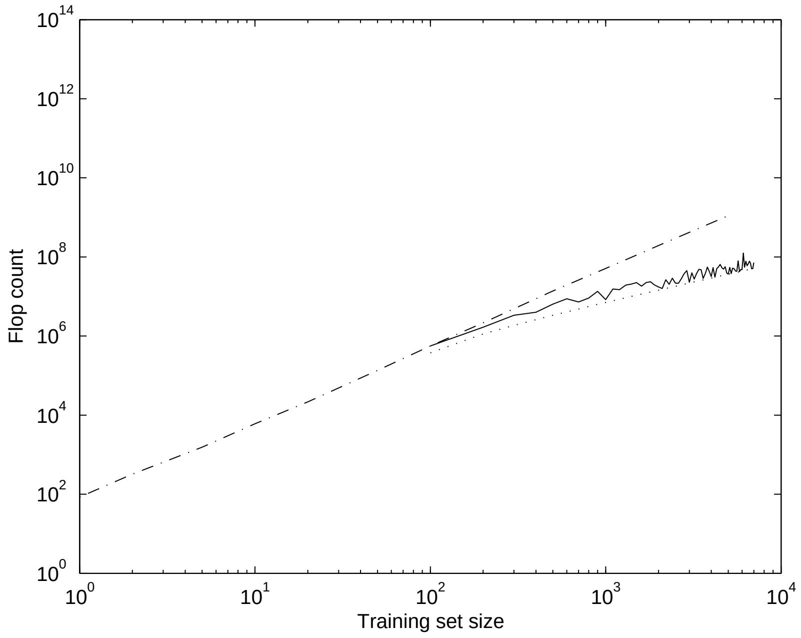


Figure 8.16: Flop count vs. training set size (mushroom dataset) - active set algorithm (inverse factorisation) (in order of increasing magnitude: base cost, incremental cost ($M = 100$) and batch cost).

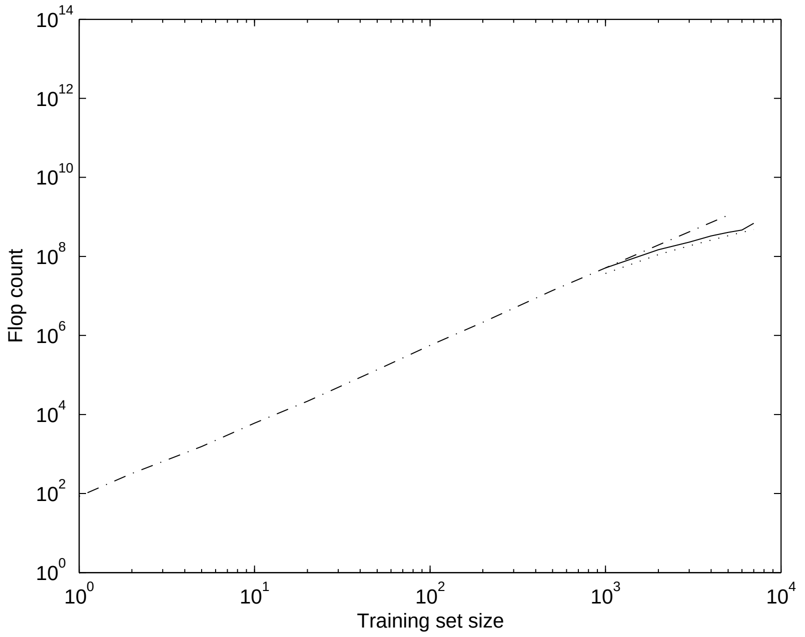


Figure 8.17: Flop count vs. training set size (mushroom dataset) - active set algorithm (Cholesky factorisation) (in order of increasing magnitude: base cost, incremental cost ($M = 1000$) and batch cost).

8.8.2 Incremental Constraint Parameter Variation

Table 6.1 gives the computational cost of incrementally changing C from some initial value (given at the top of the column) to new value (given at the left of the row), as well as the batch computational cost of batch-optimising for the C value in question along the diagonal, for the adult dataset.

It will be noted that, so long as C is being increased, the computational cost of the incrementally modifying an SVM is usually smaller than the computational

Table 8.1: Computational cost matrix (MFlops) - C variation (Active set method - Cholesky factorisation). Initial C values are shown in the top row, target C values in the left column. Diagonal entries show batch costs.

batch	0.01	0.1	1	10	100	1000	10000
0.01	*35.3*	82.3	133	159	178	185	184
0.1	8.79	*38.7*	123	156	180	188	187
1	21.6	18.6	*73*	156	192	200	200
10	69.7	64.8	50.6	*177*	181	215	227
100	143	160	148	81	*424*	214	274
1000	320	335	299	180	87.3	*1020*	326
10000	426	451	468	257	167	107	*1810*

cost associated with batch re-training for the new value of C. Indeed, for most cases shown in table 6.1 it is computationally cheaper to batch train an SVM for a small value of C (C_{small}) and then incrementally modify C to some larger value (C_{large}) than it is to batch train the SVM using C_{large}.

When decreasing C, however, in many cases (especially when either the change in C was large or the target C was small) it was computationally cheaper to batch re-train the SVM using the new value of C rather than using an incremental approach. This is not too surprising, as when C is decreased one is more likely to have to modify the Hessian factorisation $\daleth$ than when C is increased, resulting in a higher computational cost for the former.

It was not possible to complete the table using an inverse factorisation, as the numerical errors resulting from the inverse update factorisation procedure quickly became unacceptably high, leading to convergence problems. This result, combined with the similarity of computational cost between the two factorisation methods, would appear to indicate that, in general, the Cholesky factorisation method is superior to the inverse factorisation method, despite the additional algorithmic complications involved in its implementation.

Chapter 9

CONCLUSION

Ring the bells that still can ring

Forget your perfect offering

There is a crack in everything

That's how the light gets in.

– Leonard Cohen, "Anthem"

9.1 Summary of Work

The main contributions of this thesis is the formulation, asymptotic analysis and implementation of a number of novel SVM formulations. In particular, the quadric ν-SVM formulation I have described has been shown to be an effective regressor with many favorable properties, both in terms of simplicity of formulation and performance. On a broader scale, I have demonstrated that by simple modification of the base SVM optimisation problem it is possible to implement many different forms of SVM, ranging from Suykens' LS-SVM formulation, to Mangasarian's automatically biased SVM and the tube-shrinking methods of Schölkopf.

The necessary background material in pattern recognition and regression has been provided in chapters 2 and 3 of this thesis to give the reader a feel for the problem at hand and an understanding of the difficulties which must be overcome, as well as the underlying theory. In chapter 4 I have introduced the SVM approach to these problems, giving the standard formulations in slightly non-standard notation.

In this thesis I have presented the theory of Support Vector Machines for binary pattern classification and regression in a unified manner. To this end, at the start of chapter 5 I introduced the new SVM formulation to solve the "regression with inequalities" problem, and demonstrated that this effectively contained (as special cases) both the pattern classification and regression problems.

I then proceeded to present a series of generalisations to this base formulation, including the novel monomial ν-SVM, which incorporates features drawn from LS-SVM theory and tube-shrinking and combines them in such a way as to capture many of the useful features of both. Using the dual and partially dual formulations, these modifications were then summarised, and it was shown that most may be viewed as relatively straightforward modifications to the traditional optimisation problem.

In chapter 6 the problem of parameter selection was considered for the SVR using a monomial cost function, with particular attention being paid to the quadric case. The theoretical asymptotic efficiency of the monomial C-SVR and the monomial ν-SVR was computed in the asymptotic case. By doing so, I was able to demonstrate that the quadric ν-SVR method not only shares the standard ν-SVR method's property of (theoretical) noise variance insensitivity, but is also more efficient in many cases (in particular, when the training data is affected by polynomial noise of degree $p \geq 2$). These predictions have been experimentally tested, and comparisons have been made between the performances of quadric ν-SVR, standard ν-SVR, standard LS-SVR and weighted LS-SVR methods in the presence of higher order polynomial noise. Both theoretical and experimental results indicate that performance of the quadric ν-SVR method in many cases exceeds that of both standard ν-SVR and LS-SVR.

Finally, in chapters 7 and 8 I described our approach to the training problem, and in particular the incremental training problem. In doing so, the abstract optimisation problem was formed, which is a generic SVM optimisation method which may be applied to many of the SVM formulations given in chapter 5. An active set optimiser was then developed to take advantage of the particular properties of the SVM formulation. By choosing an active set method, I was able to construct an algorithm directly applicable to the problem of incremental training and parameter variation.

9.2 Future Directions

There are a number of questions which are raised in this thesis which warrant further investigation, including:

Chapter 4: Support Vector Machine Basics The issue of the geometry of the various SVM kernels remains relatively obscure and little explored in the literature. However, understanding the geometry associated with the various Mercer kernels is in many ways fundamental to understanding when a particular kernel may be (or may not be) applicable.

Chapter 5 - Extensions to the Basic SVM Method: The most general SVM formulation given in this chapter (generalised tube shrinking) will require further investigation. At present I have not proceeded past formulating the dual optimisation problem, and so as yet I have not been able to actually test this formulation.

Chapter 6 - Asymptotic Analysis of SV Regression: Most obviously, Conjecture 6.3 remains unproven, which needs to be rectified. More generally, the assumptions made in this chapter are still far to strict to be able to give confident predictions of optimal values for the various parameters (although the experimental results presented indicate that they do provide a good first guess). Further analysis must be carried out in order to see what effect the relaxation of some of these assumptions may make.

Chapter 7/8 - Training the SVM: The work presented in this thesis on this area was first completed in early 2002 (although it was not published until January 2005). In the intervening time, significant advances have occurred in the area of SVM training algorithms. Using the knowledge gained in this period, it is probable that a significantly faster algorithm could be constructed to tackle the problem of incremental training.

BIBLIOGRAPHY

[1] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, chapter 6.3: Psi (Digamma) Function. Dover, 1972.

[2] J. Aczél. *Lectures on Functional Equations and their Applications.* Academic Press, 1966.

[3] E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 9–16, 2000.

[4] S. Amari and S. Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, pages 783–789, 1999.

[5] P. Amore. Asymptotic and exact series representations for the incomplete gamma function. *Europhysics Letters*, 71(1):1–7, 2005.

[6] F. Ayres. *Theory and Problems of Matrices.* Schaum Publishing Co., New York, 1962.

[7] H. Bateman. *Higher Transcendental Functions*, volume 1. McGraw-Hill Book Company Inc., 1953.

[8] K. P. Bennett and E. J. Bredensteiner. Duality and geometry in SVM classifiers. In *Proc. 17th International Conf. on Machine Learning*, pages 57–64. Morgan Kaufmann, San Francisco, CA, 2000.

[9] E. Biglieri, R. Gitlin, and T. Lim. Adaptive cancellation of nonlinear intersymbol interference for voiceband data transmission. *IEEE Journal on Selected Areas in Communications*, SAC-2(5), September 1984.

[10] C.L. Blake and C.J. Merz. UCI repository of machine learning databases (http://www.ics.uci.edu/ mlearn/MLRepository.html), 1998.

[11] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. Shavlik, editor, *Proceedings of the 15th International Conference on Machine Learning (ICML'98)*, pages 82–90, San Fansisco, 1998. Kaufmann.

[12] C. J. C. Burges. *Advances in Kernel Methods - Support Vector Learning*, chapter 7: Geometry and Invariance in Kernel Based Methods. MIT Press, Cambridge, Massachusetts, 1998.

[13] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Knowledge Discovery and Data Mining*, 2(2):121–167, 1998.

[14] M. Carozza and S. Rampone. Function approximation from noisy data by an incremental RBF network. *Pattern Recognition*, 32(12):2081–2083, 1999.

[15] M. Carozza and S. Rampone. Towards an incremental SVM for regression. In *Proceedings of the IJCNN-2000*, 2000.

[16] A. L. Cauchy. Cours d'analyse de l'école polytechnique. *Analyse algebrique, V. Paris*, 1, 1821. (OEuvres, Ser. 2, Vol. 3, pp 98-113 and 220. Paris, 1897). 6,6,31,31,37,117,117.

[17] G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning. *NIPS2000*, pages 409–415, 2000.

[18] A. Chalimourda, B. Schölkopf, and A. Smola. Experimentally optimal ν in support vector regression for different noise models and parameter settings. *Neural Networks*, 17(1):127–141, 2004.

[19] S. Challa, M. Palaniswami, and A. Shilton. Distributed data fusion using support vector machines. In *Proceedings of the Fifth International Conference on Information Fusion*, volume 2, pages 881–885, July 2002.

[20] C. Chang and C. Lin. LIBSVM: a library for support vector machines (version 2.3), 2001.

[21] C.-C Chang and C.-J Lin. Training ν-support vector regression: Theory and algorithms. *Neural Computation*, 14(8):1959–1977, 2002.

[22] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46:131, 2002.

[23] S. Chen, G. J. Gibson, C. F. N. Cowan, and P. M. Grant. Adaptive equalization of finite non-linear channels using multilayer perceptrons. *EURASIP Signal Processing*, 20:107–119, June 1990.

[24] S. Chen, A. K. Samingan, and L. Hanzo. Adaptive multiuser receiver using a support vector machine technique. In *IEEE VEH TECHNOL CONF. Vol. 1 no. 53*, pages 604–608, 2001.

[25] H.-G. Chew, R. E. Bogner, and C.-C. Lim. Dual ν-support vector machine with error rate and training size biasing. In *Proceedings of the 26th International Conference on Acoustics, Speech and Signal Processing (ICASSP 2001)*, pages 1269–1272, 2001.

[26] V. Chvatal. *Linear Programming.* Freeman, 1983.

[27] T. F. Coleman and L. A. Hubert. A direct active set method for large sparse quadratic programs with simple bounds. *Mathematical Programming*, 45:373–406, 1989.

[28] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20(3):273–297, 1995.

[29] D.J. Crisp and C.J.C. Burges. *A Geometric Interpretation of ν-SVM Classifiers*, chapter 7: Geometry and Invariance in Kernel Based Methods. MIT Press, 1998.

[30] B. J. de Kruif. *Function Approximation for Learning Control.* PhD thesis, 2004.

[31] C. P. Diehl and G. Cauwenberghs. SVM incremental learning, adaptation and optimization. In *Proceedings of the 2003 International Joint Conference on Neural Networks*, pages 2685–2690, 2003.

[32] P. A. M. Dirac. *General Theory of Relativity.* Wiley-Interscience, 1975.

[33] C. Domeniconi and D. Gunopulos. Incremental support vector machine construction. In *ICDM*, pages 589–592, 2001.

[34] H. Drucker, C. J. C Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9, page 155. The MIT Press, 1997.

[35] A. Erdélyi, W. Magnus, F. Oberhettinger, and F. G. Tricomi. *Higher Transcendental Functions*, volume 1. Krieger, New York, 1981.

[36] S. Fine and K. Scheinberg. Efficient implementation of interior point methods for quadratic problems arising in support vector machines. *NIPS2000*, 2000.

[37] S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

[38] S. Fine and K. Scheinberg. Incremental learning and selective sampling via parametric optimization framework for SVM. In *NIPS 2001*, pages 705–711, 2001.

[39] S. Fine and K. Scheinberg. *Advances in Neural Information Processing Systems 14*, chapter Incremental Learning and Selective Sampling via Parametric Optimization Framework for SVM. MIT Press, Cambridge MA, 2002.

[40] R. Fletcher. *Practical Methods of Optimisation Volume 2: Constrained Optimisation.* John Wiley and Sons, Chichester, 1981.

[41] W. Gautschi. The incomplete gamma functions since tricomi. *Tricomi's Ideas and Contemporary Applied Mathematics*, (147):203–237, 1998.

[42] P. E. Gill, G. H. Golub, W. Murray, and M. Saunders. Methods for modifying matrix factorizations. *Mathematics of Computation*, 28(126):505–535, April 1974.

[43] R. Goldblatt. *Lectures on the Hyperreals.* Springer-Verlag, New York, 1998.

[44] G. H. Golub and C. F. Van Loan. *Matrix Computations.* John Hopkins University Press, Baltimore, 1983.

[45] I. S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products.* Academic Press, London, 2000.

[46] I. Guyon, B. Boser, and V. Vapnik. Automatic capacity tuning of very large VC-dimension classifiers. In S. Hanson, J. Cowan, and C. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 147–155. Morgan Kaufmann, 1993.

[47] S. Haykin. *Neural Networks - A Comprehensive Foundation.* Prentice Hall, Upper Saddle River, New Jersey, 2 edition, 1999.

[48] R. Herbrich and J. Weston. Adaptive margin support vector machines for classification learning. In *Proceedings of the Ninth International Conference on Artificial Neural Networks*, pages 880–885, 1999.

[49] T. Joachims. Text categorization with support vector machines. Technical Report LS VIII 23k, University of Dortmund, 1997.

[50] Vojislav Kecman and Ivana Hadzic. Support vectors selection by linear programming. In *Proceedings of the International Joing Conference on Neural Networks (IJCNN 2000)*, volume 5, pages 193–198, Italy, 2000.

[51] S. G. Krantz. *Handbook of Complex Variables.* Birkhusers, 1999.

[52] D. Lai, M. Palaniswami, and N. Mani. Fast linear stationarity methods for automatically biased support vector machines. *IJCNN03*, 2003.

[53] D. Lai, A. Shilton, N. Mani, and M. Palaniswami. A convergence rate estimate for the SVM decomposition method. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN05*, pages 931–936, August 2005.

[54] K. Levenberg. A method for the solution of certain problems in least squares. *Quart. Appl. Math.*, 2:164–168, 1944.

[55] O. Mangasarian and D. Musciant. Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks*, 10(5), 1999.

[56] O. L. Mangasarian. Generalized support vector machines. Technical Report 98-14, University of Wisconsin, Computer Sciences Department, Madison, WI, USA, October 1998.

[57] D. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 11:431–441, 1963.

[58] J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Transactions of the Royal Society of London*, 209(A), 1909.

[59] L. M. Milne-Thomson and J. Comrie, L. *Standard Four-Figure Mathematical Tables.* MacMillan and Co., London, 1948.

[60] Noboru Murata, Shuji Yoshizawa, and Shun-Ichi Amari. Network Information Criterion—determining the number of hidden units for an artificial neural network model. *IEEE Transactions on Neural Networks*, 5(6):865–872, November 1994.

[61] K. Ogata. *Modern Control Engineering.* Prentice Hall, New Jersey, 1970.

[62] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing*, pages 276–285, 1997.

[63] M. Palaniswami and A. Shilton. Adaptive support vector machines for regression. In *Proceedings of the 9th International Conference on Neural Information Processing, Singapore*, volume 2, pages 1043–1049, November 2000.

[64] M. Palaniswami, A. Shilton, D. Ralph, and B. D. Owen. Machine learning using support vector machines. In *Proceedings of International Conference on Artificial Intelligence in Science and Technology, AISAT2000*, 2000.

[65] S. Personick. Receiver design for digital fiber optic communication systems i. *The Bell Technical Journal*, 52(6), July-August 1973.

[66] J. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report 98-14, Microsoft, Redmond, Washington, April 1998.

[67] C. R. Rao. *Linear Statistical Inference and its Applications.* Wiley, New York, 1973.

[68] A. Sanderson and A. Shilton. Image tracking and recognition with applications to fisheries - an investigation of support vector machines. Final year honours thesis, Department of Electrical and Electronic Engineering, The University of Melbourne, 1999.

[69] M. Schmidt. Identifying speaker with support vector networks. In *Interface 96 Proceedings*, 1996.

[70] B. Schölkopf, P. Bartlett, A. Smola, and R. Williamson. Shrinking the tube: A new support vector regression algorithm. *Advances in Neural Information Processing Systems*, 11:330–336, 1999.

[71] B. Schölkopf, P. Bartlett, A. J. Smola, and R. Williamson. Support vector regression with automatic accuracy control. In *Proceedings of ICANN98, Perspectives in Neural Computing*, pages 111–116. Springer Verlag, 1998.

[72] B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In *Proceedings, First International Conference on Knowledge Discovery and Data Mining*, pages 252–257, Menlo Park, CA, 1995. AAAI Press.

[73] B. Schölkopf, J. C. Burges, and A. J. Smola. *Advances in Kernel Methods: Support Vector Machines.* MIT Press, Cambridge, Massachusetts, 1999.

[74] B. Schölkopf, P. Simard, A. J. Smola, and V. Vapnik. *Advances in Neural Information Processing Systems*, volume 10, chapter Prior Knowledge in Support Vector Kernels, pages 640–646. MIT Press, Cambridge, MA, 1998.

[75] B. Schölkopf and A. J. Smola. New support vector algorithms. Technical Report NeuroCOLT2 Technical Report Series, NC2-TR-1998-031, Royal Holloway College, University of London, UK, November 1998.

[76] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, Cambridge, Massachusetts, 2001.

[77] D. J. Sebald and J. A. Bucklew. Support vector machine techniques for nonlinear equalization. *IEEE Transactions on Signal Processing*, 48(11), November 2000.

[78] A. Shilton, D. Lai, and M. Palaniswami. A monomial ν-SV method for regression. 2004. Submitted to PAMI.

[79] A. Shilton and M. Palaniswami. A modified ν-SV method for simplified regression. In *Proceedings of the International Conference on Intelligent Sensing and Information Processing*, pages 422–427, 2004.

[80] A. Shilton, M. Palaniswami, D. Ralph, and A. C. Tsoi. Incremental training of support vector machines. In *Proceedings of International Joint Conference on Neural Networks, IJCNN'01 (CD version)*, 2001.

[81] A. Shilton, M. Palaniswami, D. Ralph, and A. C. Tsoi. Incremental training of support vector machines. *IEEE Transactions on Neural Networks*, 16(1):114–131, January 2005.

[82] W. L. Shirer. *The Rise and Fall of the Third Reich – A History of Nazi Germany.* Simon and Schuster, New York, 1960.

[83] A. Smola, N. Murata, B. Schölkopf, and K.-R. Muller. Asymptotically optimal choice of ϵ-loss for support vector machines. In L. Niklasson, M. Boden, and T. Ziemke, editors, *Proceedings of the 8th International Conference on Artificial Neural Networks - Perspectives in Neural Computing*, pages 105–110. Springer Verlag, 1998.

[84] A. Smola and B. Schölkopf. A tutorial on support vector regression. Technical Report NeuroCOLT2 Technical Report Series, NC2-TR-1998-030, Royal Holloway College, University of London, UK, October 1998.

[85] Alex Smola, Bernhard Schölkopf, and K. Muller. General cost functions for support vector regression. In T. Downs, M. Frean, and M. Gallagher, editors, *Proceedings of the Ninth Australian Conf. on Neural Networks*, pages 79 – 83, 1998.

[86] Alex Smola, Bernhard Schölkopf, and Gunnar Rätsch. Linear programs for automatic accuracy control in regression. In *Proceedings ICANN'99, Int. Conf. on Artificial Neural Networks*, Berlin, 1999. Springer.

[87] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. *Least Squares Support Vector Machines.* World Scientific Publishing, New Jersey, 2002.

[88] J. A. K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle. Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing, Special issue on fundamental and information processing aspects of neurocomputing*, 48(1–4):85–105, October 2002.

[89] J.A.K. Suykens, Vandewalle J., and B. De Moor. Optimal control by least squares support vector machines. *IEEE Transactions on Neural Networks*, 14(1):23–35, January 2001.

[90] J.A.K. Suykens and J. Vandewalle. Recurrent least squares support vector machines. *IEEE Transactions on Circuits and Systems-I*, 47(7):1109–1114, July 2000.

[91] N. A. Syed, H. Liu, and K. K. Sung. Incremental learning with support vector machines. In *Proceedings of the International Joint Conference on Artificial intelligence (IJCAI-99)*, 1999.

[92] J. Synge and A. Schild. *Tensor Calculus.* University of Toronto Press, 1966.

[93] A. Uncini, L. Vecci, P. Campolucci, and F. Piazza. Complex-valued neural networks with adaptive spline activation function for digital radio links nonlinear equalization. *IEEE Transactions On Signal Processing*, 47(2), February 1999.

[94] V. Vapnik. *Statistical Learning Theory.* Springer-Verlag, New York, 1995.

[95] V. Vapnik. *The Nature of Statistical Learning Theory.* John Wiley and sons, 1998.

[96] V. Vapnik and O. Chapelle. Bounds on error expectation for support vector machines. *Neural Computation*, 12(9), 2000.

[97] V. Vapnik, S. Golowich, and A. Smola. *Advances in Neural Information Processing Systems*, volume 9, chapter Support Vector Methods for Function Approximation, Regression Estimation, and Signal Processing, pages 281–187. MIT Press, 1997.

[98] V. Vapnik and S. Mukherjee. *Advances in Neural Information Processing Systems*, volume 12, chapter Support Vector Method for Multivariate Density Estimation. MIT Press, 2000.

[99] E. W. Weisstein. Beta function. From MathWorld–A Wolfram Web Resource http://mathworld.wolfram.com/BetaFunction.html.

[100] E. W. Weisstein. Gamma function. From MathWorld–A Wolfram Web Resource http://mathworld.wolfram.com/GammaFunction.html.

[101] E. W. Weisstein. Polygamma function. From MathWorld–A Wolfram Web Resource http://mathworld.wolfram.com/PolygammaFunction.html.

[102] J. Weston and R. Herbrich. *Advances in Large Margin Classifiers*, chapter Adaptive margin support vector machines, pages 281–295. MIT Press., Cambridge, MA, 2000.

[103] J. Wilkinson. A priori error analysis of algebraic processes. In *Proc. International Congress Math.*, pages 629–639, 1968.

[104] J. M. Zurada. *Introduction to Artificial Neural Networks.* West Publishing Co., St. Paul, 1992.

Appendix A

DETAILED DERIVATIONS OF (PARTIAL) DUAL FORMS

For a moment, nothing happened. Then, after a second or so, nothing continued to happen.

– Douglas Adams

For reasons of clarity and brevity, the derivations of the dual and partially dual forms of some of the SVM formulations given in chapters 4 and 5 were either not presented in full or not presented at all. This is because the method varies little from form to form, and hence it should not be too difficult to construct using previous derivations as a template. However, for completeness, the derivations are included in full in this appendix.

In all cases, the training set is defined thusly:

$$\begin{array}{l} \mathbf{Y} = \{(\mathbf{x}_1, z_1), (\mathbf{x}_2, z_2), \ldots, (\mathbf{x}_N, z_N)\} \\ \mathbf{Y} = \mathbf{Y}_= \cup \mathbf{Y}_\geq \cup \mathbf{Y}_\leq \\ \mathbf{Y}_\geq \cap \mathbf{Y}_\leq = \mathbf{Y}_= \cap \mathbf{Y}_\leq = \mathbf{Y}_= \cap \mathbf{Y}_\geq = \emptyset \\ \mathbf{x}_i \in \Re^{d_L} \\ z_i \in \Re \end{array} \tag{A.1}$$

Each formulation also has training parameters, which are summarised in table A.1.

A.1 Binary Classification (section 4.1)

In this case, $\mathbf{Y}_= = \emptyset$. The primal form (4.10) is:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} R_1(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \frac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \tag{A.2}$$

such that:

$$\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \tag{A.3}$$
$$\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \tag{A.4}$$
$$\xi_i \geq 0 \forall 1 \leq i \leq N \tag{A.5}$$
$$\xi_i^* \geq 0 \forall 1 \leq i \leq N \tag{A.6}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.3) is associated with a Lagrange multiplier $\beta_i \geq 0$, each constraint (A.4) with a

Parameter	Range	Additional constraints	Section
C	$\Re^+$	–	All.
E	$\Re \backslash \Re^-$	For tube shrinking forms, $E > 0$ is a variable.	All.
β	$\Re^+$	–	A.5
χ	$\{-1, +1\}$	–	A.8, A.9, A.10
ν	$\Re^+$	–	A.8, A.9, A.10
K	$\mathcal{M}_\kappa$	–	All.
$d : \Re \to \Re^+$	C^1	Must be convex, and $d(0) = 0$.	A.6, A.9
$d^* : \Re \to \Re^+$	C^1	Must be convex, and $d^*(0) = 0$.	A.6, A.9
$c : \Re \to \Re^+$	C^1	Must be convex, and $c(0) = 0$.	A.9
$\mathbf{t}$	$\Re^{+N}$	–	All.
$\mathbf{t}^*$	$\Re^{+N}$	–	All.
$\boldsymbol{\epsilon}$	$\Re^N$	$\epsilon_i \in \begin{cases} \Re & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \cup \mathbf{Y}_\geq \\ \Re \backslash \Re^- & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \end{cases}$ For tube shrinking forms, $\operatorname{sgn}(\boldsymbol{\epsilon}) = \chi \mathbf{1}$	All.
$\boldsymbol{\epsilon}^*$	$\Re^N$	$\epsilon_i^* \in \begin{cases} \Re & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \cup \mathbf{Y}_\geq \\ \Re \backslash \Re^- & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \end{cases}$ For tube shrinking forms, $\operatorname{sgn}(\boldsymbol{\epsilon}^*) = \chi \mathbf{1}$	All.

Table A.1: Training parameters used by various SVM formulations.

.

Lagrange multiplier $\beta_i^* \leq 0$, each constraint (A.5) with a Lagrange multiplier $\gamma_i^* \geq 0$ and each constraint (A.6) with a Lagrange multiplier $\gamma_i^* \geq 0$. By definition:

$$\begin{aligned} \beta_i &= 0 \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \notin \mathbf{Y}_\geq \\ \beta_i^* &= 0 \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \notin \mathbf{Y}_\leq \end{aligned}$$

Hence the maximin form of (A.2) may be written:

$$\begin{aligned} \min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \max_{\boldsymbol{\beta}, \boldsymbol{\beta}^*, \boldsymbol{\gamma}, \boldsymbol{\gamma}^*} L_1 = & \tfrac{1}{2} \mathbf{w}^T \mathbf{w} + \tfrac{C}{N} \mathbf{t}^T \boldsymbol{\xi} + \tfrac{C}{N} \mathbf{t}^{*T} \boldsymbol{\xi}^* - \boldsymbol{\gamma}^T \boldsymbol{\xi} - \boldsymbol{\gamma}^{*T} \boldsymbol{\xi}^* \\ & - \sum_{i=1}^N \beta_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + E \epsilon_i + \xi_i \right) \\ & - \sum_{i=1}^N \beta_i^* \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i - E \epsilon_i^* - \xi_i^* \right) \end{aligned} \tag{A.7}$$

such that:

$$\begin{aligned} \boldsymbol{\beta} &\geq \mathbf{0} \\ \boldsymbol{\beta}^* &\leq \mathbf{0} \\ \boldsymbol{\gamma} &\geq \mathbf{0} \\ \boldsymbol{\gamma}^* &\geq \mathbf{0} \end{aligned}$$

Note that for all $1 \leq i \leq N$ one of β_i and β_i^* must be zero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\alpha_i = \left\{ \begin{array}{ll} \beta_i & \text{if } \beta_i \geq 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \end{array} \right.$$
$$\epsilon_i^{(*)} = \left\{ \begin{array}{ll} \epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0 \end{array} \right.$$

Now:

$$\begin{aligned} \tfrac{\partial L_1}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} \left(\beta_i + \beta_i^*\right) \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\ \tfrac{\partial L_1}{\partial \boldsymbol{\xi}} &= \tfrac{C}{N}\mathbf{t} - \boldsymbol{\gamma} - \boldsymbol{\beta} \\ \tfrac{\partial L_1}{\partial \boldsymbol{\xi}^*} &= \tfrac{C}{N}\mathbf{t}^* - \boldsymbol{\gamma}^* + \boldsymbol{\beta}^* \end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\ \boldsymbol{\gamma} &= \tfrac{C}{N}\mathbf{t} - \boldsymbol{\beta} \\ \boldsymbol{\gamma}^* &= \tfrac{C}{N}\mathbf{t}^* + \boldsymbol{\beta}^* \end{aligned} \tag{A.8}$$

Substituting back into (A.7) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the partially dual form may be derived, namely:

$$\min_{\boldsymbol{\alpha}} \max_{b} Q_{L_1}\left(\boldsymbol{\alpha}, b\right) = \tfrac{1}{2} \left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array} \right]^T \mathbf{H} \left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array} \right] - \left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array} \right]^T \left[\begin{array}{c} 0 \\ \mathbf{z} \end{array} \right] + E \left| \begin{array}{c} b \\ \boldsymbol{\alpha} \end{array} \right|^T \left[\begin{array}{c} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{array} \right] \tag{A.9}$$

such that:

$$\begin{aligned} -\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{\leq} \\ 0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{\geq} \end{aligned} \tag{A.10}$$

where:

$$\mathbf{H} = \left[\begin{array}{cc} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{array} \right]$$

and $K_{i,j} = K\left(\mathbf{x}_i, \mathbf{x}_j\right)$.

To obtain the dual form, note that for optimality $\frac{\partial L_1}{\partial b} = 0$, where it is trivial to show that $\frac{\partial L_1}{\partial b} = \mathbf{1}^T\boldsymbol{\alpha}$. Substituting into the partial dual (A.9), the dual form is seen to be:

$$\min_{\boldsymbol{\alpha}} Q_1\left(\boldsymbol{\alpha}\right) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E\left|\boldsymbol{\alpha}\right|^T\boldsymbol{\epsilon}^{(*)} \tag{A.11}$$

such that:

$$\begin{aligned}
&-\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
&0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\
&\mathbf{1}^T \boldsymbol{\alpha} = 0
\end{aligned}$$

In either case, the optimality conditions are:

$$\begin{aligned}
&\alpha_i \leq \begin{cases} \tfrac{C}{N}t_i & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \end{cases} \\
&\alpha_i \geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ -\tfrac{C}{N}t_i^* & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \end{cases} \\
&e_i \begin{cases} \geq z_i - E\epsilon_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ = z_i - E\epsilon_i & \text{if } 0 < \alpha_i < \tfrac{C}{N}t_i \\ \leq z_i - E\epsilon_i & \text{if } \alpha_i = \tfrac{C}{N}t_i \\ \leq z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\ = z_i + E\epsilon_i^* & \text{if } -\tfrac{C}{N}t_i^* < \alpha_i < 0 \\ \geq z_i + E\epsilon_i^* & \text{if } \alpha_i = -\tfrac{C}{N}t_i^* \end{cases} \\
&f = 0
\end{aligned}$$

where:

$$\begin{bmatrix} f \\ \mathbf{e} \end{bmatrix} = \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}$$

A.2 Regression (section 4.4)

In this case, $\mathbf{Y}_{\geq} = \mathbf{Y}_{\leq} = \emptyset$. The primal form (4.20) is:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} R_1(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \frac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \tag{A.12}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.13)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.14)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.15)} \\
&\xi_i \geq 0 \forall 1 \leq i \leq N && \text{(A.16)} \\
&\xi_i^* \geq 0 \forall 1 \leq i \leq N && \text{(A.17)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.13) is associated with a Lagrange multiplier $\beta_i \geq 0$, each constraint (A.14) with a Lagrange multiplier $\beta_i^* \leq 0$, each constraint (A.15) with a Lagrange multiplier $\lambda_i \in \Re$, each constraint (A.16) with a Lagrange multiplier $\gamma_i^* \geq 0$ and each constraint

(A.17) with a Lagrange multiplier $\gamma_i^* \geq 0$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : E\epsilon_i + E\epsilon_i^* = 0 \\
\beta_i^* &= 0 \forall 1 \leq i \leq N : E\epsilon_i + E\epsilon_i^* = 0 \\
\lambda_i &= 0 \forall 1 \leq i \leq N : E\epsilon_i + E\epsilon_i^* > 0
\end{aligned}$$

Hence the maximin form of (A.12) may be written:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*} L_1 = & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i=1}^{N} \beta_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + E\epsilon_i + \xi_i \right) \\
& - \sum_{i=1}^{N} \beta_i^* \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i - E\epsilon_i^* - \xi_i^* \right) \\
& - \sum_{i=1}^{N} \lambda_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + \xi_i - \xi_i^* \right)
\end{aligned} \tag{A.18}$$

such that:

$$\begin{aligned}
\boldsymbol{\beta} &\geq \mathbf{0} \\
\boldsymbol{\beta}^* &\leq \mathbf{0} \\
\boldsymbol{\gamma} &\geq \mathbf{0} \\
\boldsymbol{\gamma}^* &\geq \mathbf{0}
\end{aligned}$$

For any $1 \leq i \leq N$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.13) and (A.14) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). Otherwise, if $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \leq i \leq N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\begin{aligned}
\alpha_i &= \begin{cases} \beta_i & \text{if } \beta_i > 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \\ \lambda_i & \text{otherwise} \end{cases} \\
\epsilon_i^{(*)} &= \begin{cases} \epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0 \end{cases}
\end{aligned}$$

Now:

$$\begin{aligned}
\tfrac{\partial L_1}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} \left(\beta_i + \beta_i^* + \lambda_i \right) \boldsymbol{\varphi}(\mathbf{x}_i) \\
\tfrac{\partial L_1}{\partial \boldsymbol{\xi}} &= \tfrac{C}{N}\mathbf{t} - \boldsymbol{\gamma} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\
\tfrac{\partial L_1}{\partial \boldsymbol{\xi}^*} &= \tfrac{C}{N}\mathbf{t}^* - \boldsymbol{\gamma}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda}
\end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned}\mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}(\mathbf{x}_i) \\ \boldsymbol{\gamma} &= \tfrac{C}{N}\mathbf{t} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\ \boldsymbol{\gamma}^* &= \tfrac{C}{N}\mathbf{t}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda}\end{aligned}$$

Substituting back into (A.18) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the partially dual form may be derived, namely:

$$\min_{\boldsymbol{\alpha}} \max_{b} Q_{L_1}(\boldsymbol{\alpha}, b) = \tfrac{1}{2}\begin{bmatrix} b \\ \boldsymbol{\alpha}\end{bmatrix}^T \mathbf{H}\begin{bmatrix} b \\ \boldsymbol{\alpha}\end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha}\end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z}\end{bmatrix} + E\left|\begin{matrix} b \\ \boldsymbol{\alpha}\end{matrix}\right|^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)}\end{bmatrix} \tag{A.19}$$

such that:

$$-\tfrac{C}{N}\mathbf{t}^* \leq \boldsymbol{\alpha} \leq \tfrac{C}{N}\mathbf{t}$$

where:

$$\mathbf{H} = \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K}\end{bmatrix}$$

and $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

To obtain the dual form, note that for optimality $\frac{\partial L_1}{\partial b} = 0$, where it is trivial to show that $\frac{\partial L_1}{\partial b} = \mathbf{1}^T\boldsymbol{\alpha}$. Substituting into the partial dual (A.19), the dual form is seen to be:

$$\min_{\boldsymbol{\alpha}} Q_1(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon}^{(*)} \tag{A.20}$$

such that:

$$\begin{aligned} -\tfrac{C}{N}\mathbf{t}^* \leq \boldsymbol{\alpha} \leq \tfrac{C}{N}\mathbf{t} \\ \mathbf{1}^T\boldsymbol{\alpha} = 0 \end{aligned}$$

In either case, the optimality conditions are:

$$\begin{aligned}&\alpha_i \leq \tfrac{C}{N}t_i \\ &\alpha_i \geq -\tfrac{C}{N}t_i^* \\ &e_i \begin{cases} \geq z_i - E\epsilon_i & \text{if } \alpha_i = 0 \\ = z_i - E\epsilon_i & \text{if } 0 < \alpha_i < \frac{C}{N}t_i \\ \leq z_i - E\epsilon_i & \text{if } \alpha_i = \frac{C}{N}t_i \\ \leq z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \\ = z_i + E\epsilon_i^* & \text{if } -\frac{C}{N}t_i^* < \alpha_i < 0 \\ \geq z_i + E\epsilon_i^* & \text{if } \alpha_i = -\frac{C}{N}t_i^* \end{cases} \\ &f = 0\end{aligned}$$

where:

$$\begin{bmatrix} f \\ \mathbf{e}\end{bmatrix} = \mathbf{H}\begin{bmatrix} b \\ \boldsymbol{\alpha}\end{bmatrix}$$

A.3 Regression with inequalities (section 5.1)

The primal form (5.2) is:

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} R_1\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \frac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \tag{A.21}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.22)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.23)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.24)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\geq && \text{(A.25)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\leq && \text{(A.26)}\\
&\xi_i \geq 0 \forall 1 \leq i \leq N && \text{(A.27)}\\
&\xi_i^* \geq 0 \forall 1 \leq i \leq N && \text{(A.28)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.22) or (A.25) is associated with a Lagrange multiplier $\beta_i \geq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.23) or (A.26) with a Lagrange multiplier $\beta_i^* \leq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.24), with a Lagrange multiplier $\lambda_i \in \Re$, each constraint (A.27) with a Lagrange multiplier $\gamma_i^* \geq 0$ and each constraint (A.28) with a Lagrange multiplier $\gamma_i^* \geq 0$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0\right) \wedge \left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_\geq\\
\beta_i^* &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0\right) \wedge \left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_\leq\\
\lambda_i &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* > 0\right)
\end{aligned}$$

Hence the maximin form of (A.21) may be written:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*} L_1 = {} & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^*\\
& - \sum_{i=1}^N \beta_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i + E\epsilon_i + \xi_i\right)\\
& - \sum_{i=1}^N \beta_i^* \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i - E\epsilon_i^* - \xi_i^*\right)\\
& - \sum_{i=1}^N \lambda_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i + \xi_i - \xi_i^*\right)
\end{aligned} \tag{A.29}$$

such that:

$$\begin{aligned}
\boldsymbol{\beta} &\geq \mathbf{0}\\
\boldsymbol{\beta}^* &\leq \mathbf{0}\\
\boldsymbol{\gamma} &\geq \mathbf{0}\\
\boldsymbol{\gamma}^* &\geq \mathbf{0}
\end{aligned}$$

For any $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.22) and (A.23) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). This result also holds (by definition) for the case $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$. For $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \leq i \leq N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\alpha_i = \begin{cases} \beta_i & \text{if } \beta_i > 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \\ \lambda_i & \text{otherwise} \end{cases}$$

$$\epsilon_i^{(*)} = \begin{cases} \epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0 \end{cases}$$

Now:

$$\begin{aligned} \frac{\partial L_1}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} (\beta_i + \beta_i^* + \lambda_i)\, \boldsymbol{\varphi}(\mathbf{x}_i) \\ \frac{\partial L_1}{\partial \boldsymbol{\xi}} &= \frac{C}{N}\mathbf{t} - \boldsymbol{\gamma} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\ \frac{\partial L_1}{\partial \boldsymbol{\xi}^*} &= \frac{C}{N}\mathbf{t}^* - \boldsymbol{\gamma}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda} \end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}(\mathbf{x}_i) \\ \boldsymbol{\gamma} &= \frac{C}{N}\mathbf{t} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\ \boldsymbol{\gamma}^* &= \frac{C}{N}\mathbf{t}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda} \end{aligned}$$

Substituting back into (A.29) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the partially dual form may be derived, namely:

$$\min_{\boldsymbol{\alpha}} \max_{b} Q_{L_1}(\boldsymbol{\alpha}, b) = \frac{1}{2} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} + E \begin{vmatrix} b \\ \boldsymbol{\alpha} \end{vmatrix}^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{bmatrix} \tag{A.30}$$

such that:

$$\begin{aligned} &-\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \\ &0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ &-\tfrac{C}{N}t_i^* \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \end{aligned}$$

where:

$$\mathbf{H} = \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{bmatrix}$$

and $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

To obtain the dual form, note that for optimality $\frac{\partial L_1}{\partial b} = 0$, where it is trivial to show that $\frac{\partial L_1}{\partial b} = \mathbf{1}^T\boldsymbol{\alpha}$. Substituting into the partial dual (A.30), the dual form is seen to be:

$$\min_{\boldsymbol{\alpha}} Q_1(\boldsymbol{\alpha}) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon}^{(*)} \tag{A.31}$$

such that:

$$\begin{aligned}
&-\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
&0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\
&-\tfrac{C}{N}t_i^* \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \\
&\mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned}$$

In either case, the optimality conditions are:

$$\begin{aligned}
&\alpha_i \leq \begin{cases} \frac{C}{N}t_i & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \end{cases} \\
&\alpha_i \geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ -\frac{C}{N}t_i^* & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \end{cases} \\
&e_i \begin{cases} \geq z_i - E\epsilon_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ = z_i - E\epsilon_i & \text{if } 0 < \alpha_i < \frac{C}{N}t_i \\ \leq z_i - E\epsilon_i & \text{if } \alpha_i = \frac{C}{N}t_i \\ \leq z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \\ = z_i + E\epsilon_i^* & \text{if } -\frac{C}{N}t_i^* < \alpha_i < 0 \\ \geq z_i + E\epsilon_i^* & \text{if } \alpha_i = -\frac{C}{N}t_i^* \end{cases} \\
&f = 0
\end{aligned}$$

where:

$$\left[\begin{array}{c} f \\ \mathbf{e} \end{array}\right] = \mathbf{H}\left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array}\right]$$

A.4 Fixed Bias SVM (section 5.2)

The primal form (5.8) is:

$$\min_{\mathbf{w},\boldsymbol{\xi},\boldsymbol{\xi}^*} R_1(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\xi}^*) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \frac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \tag{A.32}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.33)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.34)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{=} \wedge E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.35)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} && \text{(A.36)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} && \text{(A.37)} \\
&\xi_i \geq 0 \forall 1 \leq i \leq N && \text{(A.38)} \\
&\xi_i^* \geq 0 \forall 1 \leq i \leq N && \text{(A.39)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.33) or (A.36) is associated with a Lagrange multiplier $\beta_i \geq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.34) or (A.37) with a Lagrange multiplier $\beta_i^* \leq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.35), with a Lagrange multiplier $\lambda_i \in \Re$, each constraint (A.38) with a Lagrange multiplier $\gamma_i^* \geq 0$ and each constraint (A.39) with a Lagrange multiplier $\gamma_i^* \geq 0$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0) \wedge (\mathbf{x}_i, z_i) \notin \mathbf{Y}_\geq \\
\beta_i^* &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0) \wedge (\mathbf{x}_i, z_i) \notin \mathbf{Y}_\leq \\
\lambda_i &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* > 0)
\end{aligned}$$

Hence the maximin form of (A.32) may be written:

$$\begin{aligned}
\min_{\mathbf{w},\boldsymbol{\xi},\boldsymbol{\xi}^*} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*} L_1 = & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i=1}^N \beta_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + E\epsilon_i + \xi_i \right) \\
& - \sum_{i=1}^N \beta_i^* \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i - E\epsilon_i^* - \xi_i^* \right) \\
& - \sum_{i=1}^N \lambda_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + \xi_i - \xi_i^* \right)
\end{aligned} \tag{A.40}$$

such that:

$$\begin{aligned}
\boldsymbol{\beta} &\geq \mathbf{0} \\
\boldsymbol{\beta}^* &\leq \mathbf{0} \\
\boldsymbol{\gamma} &\geq \mathbf{0} \\
\boldsymbol{\gamma}^* &\geq \mathbf{0}
\end{aligned}$$

For any $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.33) and (A.34) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). This result also holds (by definition) for the case $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$. For $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \leq i \leq N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\alpha_i = \begin{cases} \beta_i & \text{if } \beta_i > 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \\ \lambda_i & \text{otherwise} \end{cases}$$

$$\epsilon_i^{(*)} = \begin{cases} \epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0 \end{cases}$$

Now:

$$\begin{aligned}
\frac{\partial L_1}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} \left(\beta_i + \beta_i^* + \lambda_i\right) \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\
\frac{\partial L_1}{\partial \boldsymbol{\xi}} &= \tfrac{C}{N}\mathbf{t} - \boldsymbol{\gamma} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\
\frac{\partial L_1}{\partial \boldsymbol{\xi}^*} &= \tfrac{C}{N}\mathbf{t}^* - \boldsymbol{\gamma}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda}
\end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned}
\mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\
\boldsymbol{\gamma} &= \tfrac{C}{N}\mathbf{t} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\
\boldsymbol{\gamma}^* &= \tfrac{C}{N}\mathbf{t}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda}
\end{aligned}$$

Substituting back into (A.40) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the dual form may be derived, namely:

$$\min_{\boldsymbol{\alpha}} Q_1\left(\boldsymbol{\alpha}\right) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\mathbf{1}b - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T\boldsymbol{\epsilon}^{(*)} \tag{A.41}$$

such that:

$$\begin{aligned}
&-\tfrac{C}{N}t_i^* \leq \alpha_i \leq 0 \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{\leq} \\
&0 \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{\geq} \\
&-\tfrac{C}{N}t_i^* \leq \alpha_i \leq \tfrac{C}{N}t_i \forall i : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{=}
\end{aligned} \tag{A.42}$$

where $K_{i,j} = K\left(\mathbf{x}_i, \mathbf{x}_j\right)$.

The optimality conditions are:

$$\begin{aligned}
\alpha_i &\leq \begin{cases} \tfrac{C}{N}t_i & \text{if } \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ 0 & \text{if } \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{\leq} \end{cases} \\
\alpha_i &\geq \begin{cases} 0 & \text{if } \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{\geq} \\ -\tfrac{C}{N}t_i^* & \text{if } \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \end{cases} \\
e_i &\begin{cases} \geq z_i - E\epsilon_i & \text{if } \alpha_i = 0 \wedge \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\geq} \\ = z_i - E\epsilon_i & \text{if } 0 < \alpha_i < \tfrac{C}{N}t_i \\ \leq z_i - E\epsilon_i & \text{if } \alpha_i = \tfrac{C}{N}t_i \\ \leq z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_{=} \cup \mathbf{Y}_{\leq} \\ = z_i + E\epsilon_i^* & \text{if } -\tfrac{C}{N}t_i^* < \alpha_i < 0 \\ \geq z_i + E\epsilon_i^* & \text{if } \alpha_i = -\tfrac{C}{N}t_i^* \end{cases}
\end{aligned}$$

where:

$$\mathbf{e} = \mathbf{K}\boldsymbol{\alpha} + \mathbf{1}b$$

A.5 Automatically Biased SVM (section 5.3)

The primal form (5.12) is:

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} R_1\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{\beta}{2}b^2 + \frac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \frac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \tag{A.43}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.44)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.45)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.46)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\geq && \text{(A.47)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\leq && \text{(A.48)}\\
&\xi_i \geq 0 \forall 1 \leq i \leq N && \text{(A.49)}\\
&\xi_i^* \geq 0 \forall 1 \leq i \leq N && \text{(A.50)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.44) or (A.47) is associated with a Lagrange multiplier $\beta_i \geq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.45) or (A.48) with a Lagrange multiplier $\beta_i^* \leq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.46), with a Lagrange multiplier $\lambda_i \in \Re$, each constraint (A.49) with a Lagrange multiplier $\gamma_i^* \geq 0$ and each constraint (A.50) with a Lagrange multiplier $\gamma_i^* \geq 0$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0\right) \wedge \left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_\geq\\
\beta_i^* &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0\right) \wedge \left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_\leq\\
\lambda_i &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* > 0\right)
\end{aligned}$$

Hence the maximin form of (A.43) may be written:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*} L_1 = &\tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{\beta}{2}b^2 + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^*\\
&- \sum_{i=1}^N \beta_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i + E\epsilon_i + \xi_i\right)\\
&- \sum_{i=1}^N \beta_i^* \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i - E\epsilon_i^* - \xi_i^*\right)\\
&- \sum_{i=1}^N \lambda_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i + \xi_i - \xi_i^*\right)
\end{aligned} \tag{A.51}$$

such that:

$$\begin{aligned}
\boldsymbol{\beta} &\geq \mathbf{0}\\
\boldsymbol{\beta}^* &\leq \mathbf{0}\\
\boldsymbol{\gamma} &\geq \mathbf{0}\\
\boldsymbol{\gamma}^* &\geq \mathbf{0}
\end{aligned}$$

For any $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.44) and (A.45) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). This result also holds (by definition) for the case $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$. For $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \le i \le N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\begin{aligned} \alpha_i &= \begin{cases} \beta_i & \text{if } \beta_i > 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \\ \lambda_i & \text{otherwise} \end{cases} \\ \epsilon_i^{(*)} &= \begin{cases} \epsilon_i & \text{if } \alpha_i \ge 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0 \end{cases} \end{aligned}$$

Now:

$$\begin{aligned} \frac{\partial L_1}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} \left(\beta_i + \beta_i^* + \lambda_i\right) \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\ \frac{\partial L_1}{\partial b} &= \beta b - \sum_{i=1}^{N} \left(\beta_i + \beta_i^* + \lambda_i\right) \\ \frac{\partial L_1}{\partial \boldsymbol{\xi}} &= \frac{C}{N}\mathbf{t} - \boldsymbol{\gamma} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\ \frac{\partial L_1}{\partial \boldsymbol{\xi}^*} &= \frac{C}{N}\mathbf{t}^* - \boldsymbol{\gamma}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda} \end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\ b &= \frac{1}{\beta}\mathbf{1}^T\boldsymbol{\alpha} \\ \boldsymbol{\gamma} &= \frac{C}{N}\mathbf{t} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\ \boldsymbol{\gamma}^* &= \frac{C}{N}\mathbf{t}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda} \end{aligned}$$

Substituting back into (A.51) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the dual form may be derived, namely:

$$\min_{\boldsymbol{\alpha}} Q_1\left(\boldsymbol{\alpha}\right) = \frac{1}{2}\boldsymbol{\alpha}^T \left(\mathbf{K} + \frac{1}{\beta}\mathbf{1}\mathbf{1}^T\right) \boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} + E|\boldsymbol{\alpha}|^T \boldsymbol{\epsilon}^{(*)} \tag{A.52}$$

such that:

$$\begin{aligned} &-\frac{C}{N}t_i^* \le \alpha_i \le 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\le \\ &0 \le \alpha_i \le \frac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\ge \\ &-\frac{C}{N}t_i^* \le \alpha_i \le \frac{C}{N}t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \end{aligned} \tag{A.53}$$

where $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

The optimality conditions are:

$$\begin{aligned}
\alpha_i &\leq \begin{cases} \frac{C}{N} t_i & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \end{cases} \\
\alpha_i &\geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ -\frac{C}{N} t_i^* & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \end{cases} \\
e_i &\begin{cases} \geq z_i - E\epsilon_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\ = z_i - E\epsilon_i & \text{if } 0 < \alpha_i < \frac{C}{N} t_i \\ \leq z_i - E\epsilon_i & \text{if } \alpha_i = \frac{C}{N} t_i \\ \leq z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\ = z_i + E\epsilon_i^* & \text{if } -\frac{C}{N} t_i^* < \alpha_i < 0 \\ \geq z_i + E\epsilon_i^* & \text{if } \alpha_i = -\frac{C}{N} t_i^* \end{cases}
\end{aligned}$$

where:

$$\begin{aligned}
\mathbf{e} &= \mathbf{K}\boldsymbol{\alpha} + \mathbf{1}b \\
b &= \mathbf{1}^T \boldsymbol{\alpha}
\end{aligned}$$

A.6 General Convex Cost Functions (section 5.4.1)

The primal form (5.19) is:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} {}_d R_{d^*}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{N}\sum_{i=1}^N t_i d(\xi_i) + \frac{C}{N}\sum_{i=1}^N t_i^* d^*(\xi_i^*) \tag{A.54}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.55)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.56)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.57)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq && \text{(A.58)} \\
&\mathbf{w}^T\boldsymbol{\varphi}(\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq && \text{(A.59)} \\
&\xi_i \geq 0 \forall 1 \leq i \leq N && \text{(A.60)} \\
&\xi_i^* \geq 0 \forall 1 \leq i \leq N && \text{(A.61)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.55) or (A.58) is associated with a Lagrange multiplier $\beta_i \geq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.56) or (A.59) with a Lagrange multiplier $\beta_i^* \leq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.57), with a Lagrange multiplier $\lambda_i \in \Re$, each constraint (A.60) with a Lagrange multiplier

$\gamma_i^* \geq 0$ and each constraint (A.61) with a Lagrange multiplier $\gamma_i^* \geq 0$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0) \wedge (\mathbf{x}_i, z_i) \notin \mathbf{Y}_\geq \\
\beta_i^* &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0) \wedge (\mathbf{x}_i, z_i) \notin \mathbf{Y}_\leq \\
\lambda_i &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* > 0)
\end{aligned}$$

Hence the maximin form of (A.54) may be written:

$$\begin{aligned}
\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \max_{\boldsymbol{\beta}, \boldsymbol{\beta}^*, \boldsymbol{\gamma}, \boldsymbol{\gamma}^*} {}_dL_{d^*} = & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C}{N}\sum_{i=1}^N t_i d(\xi_i) + \tfrac{C}{N}\sum_{i=1}^N t_i^* d^*(\xi_i^*) - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i=1}^N \beta_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + E\epsilon_i + \xi_i \right) \\
& - \sum_{i=1}^N \beta_i^* \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i - E\epsilon_i^* - \xi_i^* \right) \\
& - \sum_{i=1}^N \lambda_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi}(\mathbf{x}_i) + b \right) - z_i + \xi_i - \xi_i^* \right)
\end{aligned} \tag{A.62}$$

such that:

$$\begin{aligned}
\boldsymbol{\beta} &\geq \mathbf{0} \\
\boldsymbol{\beta}^* &\leq \mathbf{0} \\
\boldsymbol{\gamma} &\geq \mathbf{0} \\
\boldsymbol{\gamma}^* &\geq \mathbf{0}
\end{aligned}$$

For any $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.55) and (A.56) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). This result also holds (by definition) for the case $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$. For $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \leq i \leq N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\alpha_i = \begin{cases} \beta_i & \text{if } \beta_i > 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \\ \lambda_i & \text{otherwise} \end{cases}$$

$$\epsilon_i^{(*)} = \begin{cases} \epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0 \end{cases}$$

Now:

$$\begin{aligned}
\frac{\partial_d L_{d^*}}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^N (\beta_i + \beta_i^* + \lambda_i) \boldsymbol{\varphi}(\mathbf{x}_i) \\
\frac{\partial_d L_{d^*}}{\partial \xi_i} &= \tfrac{C}{N} t_i \tfrac{\partial d}{\partial \xi}(\xi_i) - \gamma_i - (\beta_i + \lambda_i) \\
\frac{\partial_d L_{d^*}}{\partial \xi_i^*} &= \tfrac{C}{N} t_i^* \tfrac{\partial d^*}{\partial \xi^*}(\xi_i^*) - \gamma_i^* + (\beta_i^* + \lambda_i)
\end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned}
\mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}(\mathbf{x}_i) \\
\gamma_i &= \tfrac{C}{N} t_i \tfrac{\partial d}{\partial \xi}(\xi_i) - (\beta_i + \lambda_i) \\
\gamma_i^* &= \tfrac{C}{N} t_i^* \tfrac{\partial d^*}{\partial \xi^*}(\xi_i^*) + (\beta_i^* + \lambda_i)
\end{aligned}$$

Substituting back into (A.62) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the partially dual form may be derived, namely:

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} \max_{b} {}_dQL_{d^*}(\boldsymbol{\alpha}, b) &= \tfrac{1}{2} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} + E \left| \begin{matrix} b \\ \boldsymbol{\alpha} \end{matrix} \right|^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{bmatrix} \\
&\quad - \tfrac{C}{N} \sum_{i=1}^{N} t_i T(\xi_i) - \tfrac{C}{N} \sum_{i=1}^{N} t_i^* T^*(\xi_i^*)
\end{aligned} \tag{A.63}$$

such that:

$$\begin{aligned}
&\alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
&\alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq}
\end{aligned} \tag{A.64}$$

where:

$$\mathbf{H} = \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{bmatrix}$$

$$\begin{aligned}
\xi_i &= \begin{cases} \inf \left\{ \xi | \tfrac{C}{N} t_i \tfrac{\partial d}{\partial \xi}(\xi) > |\alpha_i| \right\} & \text{if } \alpha_i > 0 \\ 0 & \text{if } \alpha_i \leq 0 \end{cases} \\
\xi_i^* &= \begin{cases} 0 & \text{if } \alpha_i \geq 0 \\ \inf \left\{ \xi^* | \tfrac{C}{N} t_i^* \tfrac{\partial d^*}{\partial \xi^*}(\xi^*) > |\alpha_i| \right\} & \text{if } \alpha_i < 0 \end{cases}
\end{aligned}$$

$$\begin{aligned}
T(\xi) &= d(\xi) - \xi \tfrac{\partial d}{\partial \xi}(\xi) \\
T^*(\xi^*) &= d^*(\xi^*) - \xi^* \tfrac{\partial d^*}{\partial \xi^*}(\xi^*)
\end{aligned}$$

and $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

To obtain the dual form, note that for optimality $\frac{\partial {}_dL_{d^*}}{\partial b} = 0$, where it is trivial to show that $\frac{\partial {}_dL_{d^*}}{\partial b} = \mathbf{1}^T \boldsymbol{\alpha}$. Substituting into the partial dual (A.63), the dual form is seen to be:

$$\min_{\boldsymbol{\alpha}} {}_dQ_{d^*}(\boldsymbol{\alpha}) = \tfrac{1}{2} \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{z} + E |\boldsymbol{\alpha}|^T \boldsymbol{\epsilon}^{(*)} - \tfrac{C}{N} \sum_{i=1}^{N} t_i T(\xi_i) - \tfrac{C}{N} \sum_{i=1}^{N} t_i^* T^*(\xi_i^*) \tag{A.65}$$

such that:

$$\begin{aligned}
&\alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
&\alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\
&\mathbf{1}^T \boldsymbol{\alpha} = 0
\end{aligned}$$

The optimality conditions are not simple in this case.

A.7 Quadric Cost Functions (section 5.4.2)

The primal form (5.23) is:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} R_2 (\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{C}{2N} \sum_{i=1}^{N} t_i \xi_i^2 + \frac{C}{2N} \sum_{i=1}^{N} t_i^* \xi_i^{*2} \tag{A.66}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T \boldsymbol{\varphi} (\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.67)} \\
&\mathbf{w}^T \boldsymbol{\varphi} (\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.68)} \\
&\mathbf{w}^T \boldsymbol{\varphi} (\mathbf{x}_i) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.69)} \\
&\mathbf{w}^T \boldsymbol{\varphi} (\mathbf{x}_i) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} && \text{(A.70)} \\
&\mathbf{w}^T \boldsymbol{\varphi} (\mathbf{x}_i) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} && \text{(A.71)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.67) or (A.70) is associated with a Lagrange multiplier $\beta_i \geq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.68) or (A.71) with a Lagrange multiplier $\beta_i^* \leq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.69), with a Lagrange multiplier $\lambda_i \in \Re$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0) \wedge (\mathbf{x}_i, z_i) \notin \mathbf{Y}_{\geq} \\
\beta_i^* &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0) \wedge (\mathbf{x}_i, z_i) \notin \mathbf{Y}_{\leq} \\
\lambda_i &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* > 0)
\end{aligned}$$

Hence the maximin form of (A.66) may be written:

$$\begin{aligned}
\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*} \max_{\boldsymbol{\beta}, \boldsymbol{\beta}^*} L_2 = \tfrac{1}{2} \mathbf{w}^T \mathbf{w} &+ \tfrac{C}{2N} \sum_{i=1}^{N} t_i \xi_i^2 + \tfrac{C}{2N} \sum_{i=1}^{N} t_i^* \xi_i^{*2} \\
&- \sum_{i=1}^{N} \beta_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi} (\mathbf{x}_i) + b \right) - z_i + E\epsilon_i + \xi_i \right) \\
&- \sum_{i=1}^{N} \beta_i^* \left(\left(\mathbf{w}^T \boldsymbol{\varphi} (\mathbf{x}_i) + b \right) - z_i - E\epsilon_i^* - \xi_i^* \right) \\
&- \sum_{i=1}^{N} \lambda_i \left(\left(\mathbf{w}^T \boldsymbol{\varphi} (\mathbf{x}_i) + b \right) - z_i + \xi_i - \xi_i^* \right)
\end{aligned} \tag{A.72}$$

such that:

$$\begin{aligned} \boldsymbol{\beta} &\geq \mathbf{0} \\ \boldsymbol{\beta}^* &\leq \mathbf{0} \end{aligned} \tag{A.73}$$

For any $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.67) and (A.68) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). This result also holds (by definition) for the case $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$. For $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \leq i \leq N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\begin{aligned} \alpha_i &= \begin{cases} \beta_i & \text{if } \beta_i > 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \\ \lambda_i & \text{otherwise} \end{cases} \\ \epsilon_i^{(*)} &= \begin{cases} \epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0 \end{cases} \end{aligned}$$

Now:

$$\begin{aligned} \frac{\partial L_2}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} \left(\beta_i + \beta_i^* + \lambda_i\right) \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\ \frac{\partial L_2}{\partial \xi_i} &= \frac{C}{N} t_i \xi_i - \beta_i - \lambda_i \\ \frac{\partial L_2}{\partial \xi_i^*} &= \frac{C}{N} t_i^* \xi_i^* - \beta_i^* + \lambda_i \end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\ \xi_i &= \max\left(0, \frac{N}{Ct_i}\alpha_i\right) \\ \xi_i^* &= \max\left(0, -\frac{N}{Ct_i^*}\alpha_i\right) \end{aligned}$$

Substituting back into (A.72) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the partially dual form may be derived, namely:

$$\min_{\boldsymbol{\alpha}} \max_{b} Q_{L_2}\left(\boldsymbol{\alpha}, b\right) = \frac{1}{2} \left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array}\right]^T \hat{\mathbf{H}} \left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array}\right] - \left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array}\right]^T \left[\begin{array}{c} 0 \\ \mathbf{z} \end{array}\right] + E \left|\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array}\right|^T \left[\begin{array}{c} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{array}\right] \tag{A.74}$$

such that:

$$\begin{aligned} &\alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\ &\alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \end{aligned} \tag{A.75}$$

where:

$$\begin{aligned} \hat{\mathbf{H}} &= \mathbf{H} + \left[\begin{array}{cc} 0 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{D}^{(*)} \end{array} \right] \\ \mathbf{H} &= \left[\begin{array}{cc} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{array} \right] \\ D_{i,j}^{(*)} &= \delta_{ij} \frac{N}{Ct_i^{(*)}} \end{aligned}$$

and $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

To obtain the dual form, note that for optimality $\frac{\partial L_2}{\partial b} = 0$, where it is trivial to show that $\frac{\partial L_2}{\partial b} = \mathbf{1}^T \boldsymbol{\alpha}$. Substituting into the partial dual (A.74), the dual form is seen to be:

$$\min_{\boldsymbol{\alpha}} Q_2(\boldsymbol{\alpha}) = \tfrac{1}{2} \boldsymbol{\alpha}^T \left(\mathbf{K} + \mathbf{D}^{(*)} \right) \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{z} + E |\boldsymbol{\alpha}|^T \boldsymbol{\epsilon}^{(*)} \tag{A.76}$$

such that:

$$\begin{aligned} &\alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\ &\alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ &\mathbf{1}^T \boldsymbol{\alpha} = 0 \end{aligned}$$

In either case, the optimality conditions are:

$$\begin{aligned} &\alpha_i \leq \begin{cases} \infty & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_{\geq} \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \end{cases} \\ &\alpha_i \geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\ -\infty & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_{\leq} \end{cases} \\ &e_i \begin{cases} \geq z_i - E\epsilon_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_{\geq} \\ = z_i - E\epsilon_i & \text{if } \alpha_i > 0 \\ \leq z_i + E\epsilon_i^* & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_{\leq} \\ = z_i + E\epsilon_i^* & \text{if } \alpha_i < 0 \end{cases} \\ &f = 0 \end{aligned}$$

where:

$$\left[\begin{array}{c} f \\ \mathbf{e} \end{array} \right] = \hat{\mathbf{H}} \left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array} \right]$$

A.8 Linear Tube Shrinking (section 5.5)

The primal form (5.28) is:

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E} R_{1,1}\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \chi C\nu E + \frac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \frac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* \tag{A.77}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.78)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.79)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.80)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\geq && \text{(A.81)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\leq && \text{(A.82)}\\
&\xi_i \geq 0 \forall 1 \leq i \leq N && \text{(A.83)}\\
&\xi_i^* \geq 0 \forall 1 \leq i \leq N && \text{(A.84)}\\
&E > 0 && \text{(A.85)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.78) or (A.81) is associated with a Lagrange multiplier $\beta_i \geq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.79) or (A.82) with a Lagrange multiplier $\beta_i^* \leq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.80), with a Lagrange multiplier $\lambda_i \in \Re$, each constraint (A.83) with a Lagrange multiplier $\gamma_i^* \geq 0$, each constraint (A.84) with a Lagrange multiplier $\gamma_i^* \geq 0$ and constraint (A.85) with a Lagrange multiplier $\rho \geq 0$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0\right) \wedge \left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_\geq \\
\beta_i^* &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0\right) \wedge \left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_\leq \\
\lambda_i &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* > 0\right)
\end{aligned}$$

Hence the maximin form of (A.77) may be written:

$$\begin{aligned}
\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*,\rho} L_{1,1} = & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \chi C\nu E + \tfrac{C}{N}\mathbf{t}^T\boldsymbol{\xi} + \tfrac{C}{N}\mathbf{t}^{*T}\boldsymbol{\xi}^* - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\
& - \sum_{i=1}^{N} \beta_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i + E\epsilon_i + \xi_i\right) \\
& - \sum_{i=1}^{N} \beta_i^* \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i - E\epsilon_i^* - \xi_i^*\right) \\
& - \sum_{i=1}^{N} \lambda_i \left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b\right) - z_i + \xi_i - \xi_i^*\right) \\
& - \rho E
\end{aligned} \tag{A.86}$$

such that:

$$\begin{aligned}
&\boldsymbol{\beta} \geq \mathbf{0} \\
&\boldsymbol{\beta}^* \leq \mathbf{0} \\
&\boldsymbol{\gamma} \geq \mathbf{0} \\
&\boldsymbol{\gamma}^* \geq \mathbf{0} \\
&\rho \geq 0 \\
&E \neq 0
\end{aligned}$$

For any $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.78) and (A.79) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). This result also holds (by definition) for the case $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$. For $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \leq i \leq N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\begin{aligned}
\alpha_i &= \begin{cases} \beta_i & \text{if } \beta_i > 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \\ \lambda_i & \text{otherwise} \end{cases} \\
\epsilon_i^{(*)} &= \begin{cases} \epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0 \end{cases}
\end{aligned}$$

Now:

$$\begin{aligned}
\frac{\partial L_{1,1}}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} \left(\beta_i + \beta_i^* + \lambda_i\right) \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\
\frac{\partial L_{1,1}}{\partial \boldsymbol{\xi}} &= \frac{C}{N}\mathbf{t} - \boldsymbol{\gamma} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\
\frac{\partial L_{1,1}}{\partial \boldsymbol{\xi}^*} &= \frac{C}{N}\mathbf{t}^* - \boldsymbol{\gamma}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda}
\end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned}
\mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\
\boldsymbol{\gamma} &= \frac{C}{N}\mathbf{t} - \boldsymbol{\beta} - \boldsymbol{\lambda} \\
\boldsymbol{\gamma}^* &= \frac{C}{N}\mathbf{t}^* + \boldsymbol{\beta}^* + \boldsymbol{\lambda}
\end{aligned}$$

As noted in section 5.5, $\rho = 0$ due to the constraint $E > 0$ (rather than $E \geq 0$). Hence:

$$\begin{aligned}
&\frac{\partial L_{1,1}}{\partial E} = 0 \\
&\Rightarrow \left|\boldsymbol{\epsilon}^{(*)}\right|^T \left|\boldsymbol{\alpha}\right| = C\nu - \chi\rho = C\nu
\end{aligned}$$

Substituting back into (A.86) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the partially dual form may be derived, namely:

$$\min_{\boldsymbol{\alpha}} \max_{b,E} QL_{1,1}(\boldsymbol{\alpha}, b, E) = \frac{1}{2} \begin{bmatrix} E \\ b \\ \boldsymbol{\alpha} \end{bmatrix}^T \bar{\mathbf{H}} \begin{bmatrix} E \\ b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} E \\ b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} C\nu \\ 0 \\ \mathbf{z} \end{bmatrix} \tag{A.87}$$

such that:

$$\begin{aligned} &-\tfrac{C}{N} t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \\ &0 \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ &-\tfrac{C}{N} t_i^* \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \end{aligned}$$

where:

$$\bar{\mathbf{H}} = \begin{bmatrix} 0 & 0 & \left(\left|\boldsymbol{\epsilon}^{(*)}\right| \operatorname{sgn}(\boldsymbol{\alpha})\right)^T \\ 0 & 0 & \mathbf{1}^T \\ \left(\left|\boldsymbol{\epsilon}^{(*)}\right| \operatorname{sgn}(\boldsymbol{\alpha})\right) & \mathbf{1} & \mathbf{K} \end{bmatrix}$$

and $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

To obtain the dual form, note that for optimality $\frac{\partial L_{1,1}}{\partial b} = 0$, where it is trivial to show that $\frac{\partial L_{1,1}}{\partial b} = \mathbf{1}^T \boldsymbol{\alpha}$, and similarly $\frac{\partial L_{1,1}}{\partial E} = \left|\boldsymbol{\epsilon}^{(*)}\right|^T |\boldsymbol{\alpha}| - C\nu = 0$. Substituting into the partial dual (A.87), the dual form is seen to be:

$$\min_{\boldsymbol{\alpha}} Q_{1,1}(\boldsymbol{\alpha}) = \tfrac{1}{2} \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{z} \tag{A.88}$$

such that:

$$\begin{aligned} &-\tfrac{C}{N} t_i^* \leq \alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \\ &0 \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ &-\tfrac{C}{N} t_i^* \leq \alpha_i \leq \tfrac{C}{N} t_i \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \\ &\mathbf{1}^T \boldsymbol{\alpha} = 0 \\ &\left|\boldsymbol{\epsilon}^{(*)}\right|^T |\boldsymbol{\alpha}| = C\nu \end{aligned}$$

In either case, the optimality conditions are:

$$\begin{aligned} &\alpha_i \leq \begin{cases} \frac{C}{N} t_i & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \end{cases} \\ &\alpha_i \geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ -\frac{C}{N} t_i^* & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \end{cases} \\ &e_i \begin{cases} \geq z_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\ = z_i & \text{if } 0 < \alpha_i < \frac{C}{N} t_i \\ \leq z_i & \text{if } \alpha_i = \frac{C}{N} t_i \\ \leq z_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\ = z_i & \text{if } -\frac{C}{N} t_i^* < \alpha_i < 0 \\ \geq z_i & \text{if } \alpha_i = -\frac{C}{N} t_i^* \end{cases} \\ &f = 0 \\ &g = C\nu \end{aligned}$$

where:

$$\begin{bmatrix} g \\ f \\ \mathbf{e} \end{bmatrix} = \bar{\mathbf{H}} \begin{bmatrix} E \\ b \\ \boldsymbol{\alpha} \end{bmatrix}$$

A.9 Generalised Tube Shrinking (section 5.5.1)

The primal form (5.34) is:

$$\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E} {}_dR_{d^*,c}\left(\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \chi C\nu c\left(E\right) + \frac{C}{N}\sum_{i=1}^{N} t_i d\left(\xi_i\right) + \frac{C}{N}\sum_{i=1}^{N} t_i^* d^*\left(\xi_i^*\right) \tag{A.89}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.90)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.91)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.92)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\geq && \text{(A.93)}\\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : \left(\mathbf{x}_i, z_i\right) \in \mathbf{Y}_\leq && \text{(A.94)}\\
&\xi_i \geq 0 \forall 1 \leq i \leq N && \text{(A.95)}\\
&\xi_i^* \geq 0 \forall 1 \leq i \leq N && \text{(A.96)}\\
&E \geq 0 && \text{(A.97)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.90) or (A.93) is associated with a Lagrange multiplier $\beta_i \geq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.91) or (A.94) with a Lagrange multiplier $\beta_i^* \leq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.92), with a Lagrange multiplier $\lambda_i \in \Re$, each constraint (A.95) with a Lagrange multiplier $\gamma_i^* \geq 0$, each constraint (A.96) with a Lagrange multiplier $\gamma_i^* \geq 0$, and constraint (A.97) with a Lagrange multiplier $\rho \geq 0$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0\right) \wedge \left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_\geq\\
\beta_i^* &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0\right) \wedge \left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_\leq\\
\lambda_i &= 0 \forall 1 \leq i \leq N : \left(\left(\mathbf{x}_i, z_i\right) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* > 0\right)
\end{aligned}$$

Hence the maximin form of (A.89) may be written:

$$\begin{aligned}\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*,E}\ \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*,\boldsymbol{\gamma},\boldsymbol{\gamma}^*,\rho}\ {}_dL_{d^*,c} = {} & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \chi C\nu c\left(E\right) + \tfrac{C}{N}\sum_{i=1}^{N} t_i d\left(\xi_i\right) + \tfrac{C}{N}\sum_{i=1}^{N} t_i^* d^*\left(\xi_i^*\right) \\ & -\rho E - \boldsymbol{\gamma}^T\boldsymbol{\xi} - \boldsymbol{\gamma}^{*T}\boldsymbol{\xi}^* \\ & -\sum_{i=1}^{N}\beta_i\left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right)+b\right) - z_i + E\epsilon_i + \xi_i\right) \\ & -\sum_{i=1}^{N}\beta_i^*\left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right)+b\right) - z_i - E\epsilon_i^* - \xi_i^*\right) \\ & -\sum_{i=1}^{N}\lambda_i\left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right)+b\right) - z_i + \xi_i - \xi_i^*\right)\end{aligned} \tag{A.98}$$

such that:

$$\begin{aligned}\boldsymbol{\beta} &\geq \mathbf{0}\\ \boldsymbol{\beta}^* &\leq \mathbf{0}\\ \boldsymbol{\gamma} &\geq \mathbf{0}\\ \boldsymbol{\gamma}^* &\geq \mathbf{0}\\ \rho &\geq 0\end{aligned}$$

For any $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.90) and (A.91) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). This result also holds (by definition) for the case $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$. For $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \leq i \leq N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\begin{aligned}\alpha_i &= \begin{cases}\beta_i & \text{if } \beta_i > 0\\ \beta_i^* & \text{if } \beta_i^* < 0\\ \lambda_i & \text{otherwise}\end{cases}\\ \epsilon_i^{(*)} &= \begin{cases}\epsilon_i & \text{if } \alpha_i \geq 0\\ \epsilon_i^* & \text{if } \alpha_i^* < 0\end{cases}\end{aligned}$$

Now:

$$\begin{aligned}\frac{\partial\, {}_dL_{d^*,c}}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N}\left(\beta_i + \beta_i^* + \lambda_i\right)\boldsymbol{\varphi}\left(\mathbf{x}_i\right)\\ \frac{\partial\, {}_dL_{d^*,c}}{\partial \xi_i} &= \tfrac{C}{N}t_i\tfrac{\partial d}{\partial \xi}\left(\xi_i\right) - \gamma_i - \left(\beta_i + \lambda_i\right)\\ \frac{\partial\, {}_dL_{d^*,c}}{\partial \xi_i^*} &= \tfrac{C}{N}t_i^*\tfrac{\partial d^*}{\partial \xi^*}\left(\xi_i^*\right) - \gamma_i^* + \left(\beta_i^* + \lambda_i\right)\\ \frac{\partial\, {}_dL_{d^*,c}}{\partial E} &= \chi C\nu\tfrac{\partial c}{\partial E}\left(E\right) - \rho - \sum_{i=1}^{N}\beta_i\epsilon_i + \sum_{i=1}^{N}\beta_i^*\epsilon_i^*\end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned}
\mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}(\mathbf{x}_i) \\
\gamma_i &= \tfrac{C}{N} t_i \tfrac{\partial d}{\partial \xi}(\xi_i) - (\beta_i + \lambda_i) \\
\gamma_i^* &= \tfrac{C}{N} t_i^* \tfrac{\partial d^*}{\partial \xi^*}(\xi_i^*) + (\beta_i^* + \lambda_i) \\
\rho &= \chi C \nu \tfrac{\partial c}{\partial E}(E) - \sum_{i=1}^{N} \beta_i \epsilon_i + \sum_{i=1}^{N} \beta_i^* \epsilon_i^*
\end{aligned}$$

Substituting back into (A.98) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the partially dual form may be derived, namely:

$$\begin{aligned}
\min_{\boldsymbol{\alpha}} \max_{b} {}_d\mathcal{Q}L_{d^*,c}(\boldsymbol{\alpha}, b) &= \tfrac{1}{2} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \mathbf{H} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} + E \left| \begin{matrix} b \\ \boldsymbol{\alpha} \end{matrix} \right|^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{bmatrix} \\
&\quad -\chi C \nu S(E) - \tfrac{C}{N} \sum_{i=1}^{N} t_i T(\xi_i) - \tfrac{C}{N} \sum_{i=1}^{N} t_i^* T^*(\xi_i^*)
\end{aligned} \tag{A.99}$$

such that:

$$\begin{aligned}
\alpha_i &\le 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\le} \\
\alpha_i &\ge 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\ge}
\end{aligned} \tag{A.100}$$

where:

$$\mathbf{H} = \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{bmatrix}$$

$$\begin{aligned}
\xi_i &= \begin{cases} \inf \left\{ \xi | \tfrac{C}{N} t_i \tfrac{\partial d}{\partial \xi}(\xi) > |\alpha_i| \right\} & \text{if } \alpha_i > 0 \\ 0 & \text{if } \alpha_i \le 0 \end{cases} \\
\xi_i^* &= \begin{cases} 0 & \text{if } \alpha_i \ge 0 \\ \inf \left\{ \xi^* | \tfrac{C}{N} t_i^* \tfrac{\partial d^*}{\partial \xi^*}(\xi^*) > |\alpha_i| \right\} & \text{if } \alpha_i < 0 \end{cases} \\
E &= \begin{cases} \inf \left\{ E | C \nu \tfrac{\partial c}{\partial E}(E) > \left| \boldsymbol{\epsilon}^{(*)} \right|^T |\boldsymbol{\alpha}| \right\} & \text{if } \chi = +1 \\ \inf \left\{ E | C \nu \tfrac{\partial c}{\partial E}(E) < \left| \boldsymbol{\epsilon}^{(*)} \right|^T |\boldsymbol{\alpha}| \right\} & \text{if } \chi = -1 \end{cases}
\end{aligned}$$

$$\begin{aligned}
T(\xi) &= d(\xi) - \xi \tfrac{\partial d}{\partial \xi}(\xi) \\
T^*(\xi^*) &= d^*(\xi^*) - \xi^* \tfrac{\partial d^*}{\partial \xi^*}(\xi^*) \\
S(E) &= c(E) - E \tfrac{\partial d}{\partial E}(E)
\end{aligned}$$

and $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

To obtain the dual form, note that for optimality $\frac{\partial_d L_{d^*,c}}{\partial b} = 0$, where it is trivial to show that $\frac{\partial_d L_{d^*,c}}{\partial b} = \mathbf{1}^T \boldsymbol{\alpha}$. Substituting into the partial dual (A.99), the dual form is seen to be:

$$\min_{\boldsymbol{\alpha}} {}_dQ_{d^*,c}\left(\boldsymbol{\alpha}\right) = \tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{z} - \chi C\nu S\left(E\right) - \tfrac{C}{N}\sum_{i=1}^{N} t_i T\left(\xi_i\right) - \tfrac{C}{N}\sum_{i=1}^{N} t_i^* T^*\left(\xi_i^*\right) \tag{A.101}$$

such that:

$$\begin{aligned}
&\alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} \\
&\alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} \\
&\mathbf{1}^T\boldsymbol{\alpha} = 0
\end{aligned}$$

The optimality conditions are not simple in this case.

A.10 Quadric ν-SVR (section 5.5.2)

It is assumed here that $\chi = +1$. The primal form (5.23) is:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, E} R_{2,2}\left(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C\nu}{2}E^2 + \frac{C}{2N}\sum_{i=1}^{N} t_i \xi_i^2 + \frac{C}{2N}\sum_{i=1}^{N} t_i^* \xi_i^{*2} \tag{A.102}$$

such that:

$$\begin{aligned}
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.103)} \\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* > 0 && \text{(A.104)} \\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b = z_i - \xi_i + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \wedge E\epsilon_i + E\epsilon_i^* = 0 && \text{(A.105)} \\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \geq z_i - E\epsilon_i - \xi_i \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\geq} && \text{(A.106)} \\
&\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right) + b \leq z_i + E\epsilon_i^* + \xi_i^* \forall 1 \leq i \leq N : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\leq} && \text{(A.107)}
\end{aligned}$$

Lagrange multipliers are defined for each constraint thusly. Each constraint (A.103) or (A.106) is associated with a Lagrange multiplier $\beta_i \geq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.104) or (A.107) with a Lagrange multiplier $\beta_i^* \leq 0$ (noting that at most one of these constraints will be relevant to each $1 \leq i \leq N$), each constraint (A.105) and with a Lagrange multiplier $\lambda_i \in \Re$. By definition:

$$\begin{aligned}
\beta_i &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0) \wedge (\mathbf{x}_i, z_i) \notin \mathbf{Y}_{\geq} \\
\beta_i^* &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* = 0) \wedge (\mathbf{x}_i, z_i) \notin \mathbf{Y}_{\leq} \\
\lambda_i &= 0 \forall 1 \leq i \leq N : ((\mathbf{x}_i, z_i) \notin \mathbf{Y}_= \vee E\epsilon_i + E\epsilon_i^* > 0)
\end{aligned}$$

Hence the maximin form of (A.102) may be written:

$$\begin{aligned}\min_{\mathbf{w},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \max_{\boldsymbol{\beta},\boldsymbol{\beta}^*} L_{2,2} = &\tfrac{1}{2}\mathbf{w}^T\mathbf{w} + \tfrac{C\nu}{2}E^2 + \tfrac{C}{2N}\sum_{i=1}^{N} t_i\xi_i^2 + \tfrac{C}{2N}\sum_{i=1}^{N} t_i^*\xi_i^{*2} \\ &-\sum_{i=1}^{N}\beta_i\left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right)+b\right)-z_i+E\epsilon_i+\xi_i\right) \\ &-\sum_{i=1}^{N}\beta_i^*\left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right)+b\right)-z_i-E\epsilon_i^*-\xi_i^*\right) \\ &-\sum_{i=1}^{N}\lambda_i\left(\left(\mathbf{w}^T\boldsymbol{\varphi}\left(\mathbf{x}_i\right)+b\right)-z_i+\xi_i-\xi_i^*\right)\end{aligned} \tag{A.108}$$

such that:

$$\begin{aligned}\boldsymbol{\beta} &\geq \mathbf{0} \\ \boldsymbol{\beta}^* &\leq \mathbf{0}\end{aligned} \tag{A.109}$$

For any $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* > 0$, only one of the constraints (A.103) and (A.104) may be met exactly (i.e. left side equals right side). Hence at least one of β_i and β_i^* must be zero for all such cases (and $\lambda_i = 0$ by definition). This result also holds (by definition) for the case $(\mathbf{x}_i, z_i) \notin \mathbf{Y}_=$. For $(\mathbf{x}_i, z_i) \in \mathbf{Y}_=$ where $E\epsilon_i + E\epsilon_i^* = 0$, both of β_i and β_i^* will be zero by definition. Hence for all $1 \leq i \leq N$, only one of β_i, β_i^* and λ_i may be nonzero. Hence it is possible to unambiguously define the vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\epsilon}^{(*)}$ using:

$$\begin{aligned}\alpha_i &= \begin{cases}\beta_i & \text{if } \beta_i > 0 \\ \beta_i^* & \text{if } \beta_i^* < 0 \\ \lambda_i & \text{otherwise}\end{cases} \\ \epsilon_i^{(*)} &= \begin{cases}\epsilon_i & \text{if } \alpha_i \geq 0 \\ \epsilon_i^* & \text{if } \alpha_i^* < 0\end{cases}\end{aligned}$$

Now:

$$\begin{aligned}\tfrac{\partial L_{2,2}}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N}\left(\beta_i + \beta_i^* + \lambda_i\right)\boldsymbol{\varphi}\left(\mathbf{x}_i\right) \\ \tfrac{\partial L_{2,2}}{\partial \xi_i} &= \tfrac{C}{N}t_i\xi_i - \beta_i - \lambda_i \\ \tfrac{\partial L_{2,2}}{\partial \xi_i^*} &= \tfrac{C}{N}t_i^*\xi_i^* - \beta_i^* + \lambda_i \\ \tfrac{\partial L_{2,2}}{\partial E} &= C\nu E - \sum_{i=1}^{N}\beta_i\epsilon_i + \sum_{i=1}^{N}\beta_i^*\epsilon_i^*\end{aligned}$$

must, for optimality, be zero. Hence:

$$\begin{aligned}
\mathbf{w} &= \sum_{i=1}^{N} \alpha_i \boldsymbol{\varphi}(\mathbf{x}_i) \\
\xi_i &= \max\left(0, \tfrac{N}{Ct_i}\alpha_i\right) \\
\xi_i^* &= \max\left(0, -\tfrac{N}{Ct_i^*}\alpha_i\right) \\
E &= \tfrac{1}{C\nu} |\boldsymbol{\alpha}|^T \boldsymbol{\epsilon}^{(*)}
\end{aligned}$$

Substituting back into (A.108) (noting the constraints placed on the various multipliers) negating and expressing the result in terms of $\boldsymbol{\alpha}$, the partially dual form may be derived, namely:

$$\min_{\boldsymbol{\alpha}} \max_{b} QL_{2,2}(\boldsymbol{\alpha}, b) = \tfrac{1}{2} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \hat{\mathbf{H}} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} - \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}^T \begin{bmatrix} 0 \\ \mathbf{z} \end{bmatrix} + E \left| \begin{matrix} b \\ \boldsymbol{\alpha} \end{matrix} \right|^T \begin{bmatrix} 0 \\ \boldsymbol{\epsilon}^{(*)} \end{bmatrix} \tag{A.110}$$

such that:

$$\begin{aligned}
\alpha_i \le 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\le} \\
\alpha_i \ge 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_{\ge}
\end{aligned} \tag{A.111}$$

where:

$$\begin{aligned}
\hat{\mathbf{H}} &= \mathbf{H} + \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{N}^{(*)} + \mathbf{D}^{(*)} \end{bmatrix} \\
\mathbf{H} &= \begin{bmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & \mathbf{K} \end{bmatrix} \\
N_{i,j}^{(*)} &= \begin{cases} \frac{1}{C\nu} \left|\epsilon_i^{(*)}\right| \left|\epsilon_j^{(*)}\right| & \text{if } \operatorname{sgn}(\alpha_i \alpha_j) = +1 \\ -\frac{1}{C\nu} \left|\epsilon_i^{(*)}\right| \left|\epsilon_j^{(*)}\right| & \text{if } \operatorname{sgn}(\alpha_i \alpha_j) \neq +1 \end{cases} \\
D_{i,j}^{(*)} &= \delta_{ij} \frac{N}{Ct_i^{(*)}}
\end{aligned}$$

and $K_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j)$.

To obtain the dual form, note that for optimality $\frac{\partial L_{2,2}}{\partial b} = 0$, where it is trivial to show that $\frac{\partial L_{2,2}}{\partial b} = \mathbf{1}^T \boldsymbol{\alpha}$. Substituting into the partial dual (A.110), the dual form is seen to be:

$$\min_{\boldsymbol{\alpha}} Q_2(\boldsymbol{\alpha}) = \tfrac{1}{2} \boldsymbol{\alpha}^T \left(\mathbf{K} + \mathbf{N}^{(*)} + \mathbf{D}^{(*)}\right) \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{z} + E |\boldsymbol{\alpha}|^T \boldsymbol{\epsilon}^{(*)} \tag{A.112}$$

such that:

$$\begin{aligned}
&\alpha_i \leq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \\
&\alpha_i \geq 0 \forall i : (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\
&\mathbf{1}^T \boldsymbol{\alpha} = 0
\end{aligned}$$

In either case, the optimality conditions are:

$$\begin{aligned}
&\alpha_i \leq \begin{cases} \infty & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\ 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\leq \end{cases} \\
&\alpha_i \geq \begin{cases} 0 & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_\geq \\ -\infty & \text{if } (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \end{cases} \\
&e_i \begin{cases} \geq z_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\geq \\ = z_i & \text{if } \alpha_i > 0 \\ \leq z_i & \text{if } \alpha_i = 0 \wedge (\mathbf{x}_i, z_i) \in \mathbf{Y}_= \cup \mathbf{Y}_\leq \\ = z_i & \text{if } \alpha_i < 0 \end{cases} \\
&f = 0
\end{aligned}$$

where:

$$\begin{bmatrix} f \\ \mathbf{e} \end{bmatrix} = \hat{\mathbf{H}} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix}$$

Appendix B

DIFFERENTIAL GEOMETRY

I reject your reality, and substitute my own.

- Adam Savage

In this appendix, I give a quick introduction to the theory of differential geometry. This is not intended to be a rigorous treatment of the subject. Instead, I hope to give sufficient detail to give a feel for the subject without going too far into obscurities. The notation used here is the older style indexed notation of [92] [32], and infinitesimals will often be used somewhat carelessly, ignoring the more modern mathematical treatment which avoids this approach at the expense (in my opinion) or clarity and insight.[1]

B.1 Vector Space, Space and Tensors

B.1.1 Vector Spaces

I will begin with the formal definitions of a (real) *vector space* and a (real) *vector subspace* of a vector space:

Definition B.1.1. *A (real)* vector space *consists of a set V together with two operations, namely* addition *of the form $+ : V \times V \to V$ and* scalar multiplication *of the form $V \times \Re \to V$, satisfying the following axioms:*

Properties of addition:

1. Associativity of vector addition: $\mathbf{u}+(\mathbf{v}+\mathbf{w}) = (\mathbf{u}+\mathbf{v})+\mathbf{w}$ for all $\mathbf{u}, \mathbf{v}, \mathbf{w} \in V$.

2. Commutativity of vector addition: $\mathbf{u}+\mathbf{v} = \mathbf{v}+\mathbf{u}$ for all $\mathbf{u}, \mathbf{v} \in V$.

3. Existence of additive identity in V: there exists $\mathbf{0} \in V$ such that $\mathbf{0}+\mathbf{u} = \mathbf{u}$ for all $\mathbf{u} \in V$.

4. Existence of additive inverse in V: for every $\mathbf{u} \in V$ there exists $-\mathbf{u} \in V$ such that $\mathbf{u} + -\mathbf{u} = \mathbf{0}$.

Properties of scalar multiplication:

1. Distributivity with respect to vector addition: $a(\mathbf{u}+\mathbf{v}) = a\mathbf{u} + a\mathbf{v}$ for all $a \in \Re$, $\mathbf{u}, \mathbf{v} \in V$.

[1]Which is not to say that infinitesimals cannot be used in a rigorous approach, as in [43].

2. Distributivity with respect to scalar addition: $(a+b)\,\mathbf{u} = a\mathbf{u} + b\mathbf{u}$ for all $a, b \in \Re$, $\mathbf{u} \in V$.

3. Associativity of scalar multiplication in V: $(ab)\,\mathbf{u} = a\,(b\mathbf{u})$ for all $a, b \in \Re$, $\mathbf{u} \in V$.

4. Neutrality of 1: $1\mathbf{u} = \mathbf{u}$ for all $\mathbf{u} \in V$.

Definition B.1.2. *A (real)* vector subspace *W of a (real) vector space V is a subset of $W \subseteq V$ such that W, along with the operations of addition and multiplication inherited from V, is itself a (real) vector space.*

All vectors will be written in bold type, e.g. $\mathbf{u}$. For any vector space, it can be shown that one may construct a *basis* for that vector space, defined by:

Definition B.1.3. *A* basis *for a (real) vector space V is a set of vectors $\{\mathbf{e}^i \in V |\, i \in I\}$ satisfying:*

1. Span: For all $\mathbf{u} \in V$ there exists $a^i \in \Re$, $i \in I$ such that $\mathbf{u} = \sum\limits_{i \in I} a^i \mathbf{e}^i$.

2. Independence: $\sum\limits_{i \in I} a^i \mathbf{e}^i = \mathbf{0}$ if and only if $a^i = 0$ for all $i \in I$.

There may, of course, be many different basis for a given vector space V. However, it can be shown that all such basis will have the same number of elements (assuming finiteness). This number of elements is known as the *dimension* of the vector space. Formally:

Definition B.1.4. *The* dimension *of a vector space V is defined to be the number of elements, N, in some basis $\{\mathbf{e}^i \in V |\, i \in I\}$ of that vector space.*

As only real finite dimensional vector spaces are of interest here, it may be assumed that any vector spaces mentioned herein are real and finite dimensional.

Given an N-dimensional vector space V and a basis $\{\mathbf{e}^i \in V |\, 1 \leq i \leq N\}$ for that vector space, any vector $\mathbf{u}$ in that vector space may be completely and uniquely defined by the N-tuple $\left(u^1, u^2, \ldots, u^N\right) \in \Re^N$ using the relation:

$$\mathbf{u} = \sum_{i=1}^{N} u^i \mathbf{e}^i \tag{B.1}$$

Taking the definition of the basis set $\{\mathbf{e}^i \in V |\, 1 \leq i \leq N\}$ to be implicitly along with the vector space V the shorthand notation u^i may be used for the vector $\mathbf{u}$, where use of equation (B.1) is implicit, along with the range convention:

Notation B.1.1 (Range convention). *When a small Latin suffix (superscript or subscript) occurs, it is understood to take all values* 1, 2, …, N, *where N is the dimension of the vector space.*

Given a different basis $\left\{ \bar{\mathbf{e}}^{\bar{i}} \in V \middle| 1 \leq \bar{i} \leq N \right\}$ for the same vector space, the same vector $\mathbf{u}$ is written $\bar{u}^{\bar{i}}$ with the obvious interpretation, where the differing basis is denoted by the overbars shared with the basis vectors. Suppose that:

$$\bar{\mathbf{e}}^{\bar{i}} = \sum_{i=1}^{N} a^{i}{}_{\bar{i}} \mathbf{e}^{i}$$

This is known as a *change of basis*. The restriction which must be made here is that the matrix $[a^{i}{}_{\bar{i}}]$ must be non-singular (otherwise independence will be lost, and subsequently $\left\{ \bar{\mathbf{e}}^{\bar{i}} \in V \right\}$ will not form a basis for V). Noting that:

$$\mathbf{u} = \sum_{i=1}^{N} u^{i} \mathbf{e}^{i}$$

and:

$$\mathbf{u} = \sum_{\bar{i}=1}^{N} \bar{u}^{\bar{i}} \bar{\mathbf{e}}^{\bar{i}} = \sum_{i=1}^{N} a^{i}{}_{\bar{i}} \bar{u}^{\bar{i}} \mathbf{e}^{i}$$

it follows that:

$$\bar{u}^{\bar{i}} = b_{i}{}^{\bar{i}} u^{i}$$

where $\left[b_{i}{}^{\bar{i}} \right] = [a^{i}{}_{\bar{i}}]^{-1}$, and use has been made of the Einstein summation convention:

Notation B.1.2 (Einstein summation convention). *If a small Latin suffix is repeated twice in a term, where one instance is a subscript and the other a superscript, there is an implied summation with respect to that suffix over the range* 1 *to* N.

It should be noted that summation over a repeated indices where both incidences are of the same type (both superscript or both subscript) is not defined, and hence should be avoided. The term will also be ill-defined if an index is repeated more than twice.

B.1.2 Spaces, Subspaces, Points and Curves

I will now move away from the concept of a real vector space to a more general, abstract concept of space. Specifically, consider a set X of *points* $p \in X$. A set of values $\left(x^1, x^2, \ldots, x^N \right)$ is assigned to each point, and the variables $x^1, x^2, \ldots, x^N$ are called the *coordinates* of that point (and are *not* vectors). Each coordinate is assumed to have some range (usually $-\infty$ to ∞, but more restricted ranges are possible, e.g. 0 to 2π), which is assumed to be an open subset of $\Re$. Assuming that every $\left(x^1, x^2, \ldots, x^N \right)$ where each x^i is within it's given range corresponds to a unique point, X is called an N-dimensional *space*, and is denoted by, for example, V_N, where the capital Latin subscript indicates the dimensionality of the space. As there is a one-to-one correspondence between points p and coordinate sets $\left(x^1, x^2, \ldots, x^N \right)$, the coordinate set $\left(x^1, x^2, \ldots, x^N \right)$ may itself be unambiguously called a point. Using the usual range convention, a point p may be written x^i.[2]

[2]There are also certain smoothness assumptions here too, which I will neglect for simplicity.

A *curve* is defined to be all points given by the set of N continuous functions:

$$x^i = f^i\left(u\right)$$

as u varies over its range $\acute{u} \leq u \leq \grave{u}$.

More generally, an M *dimensional subspace* W_M of V_N is defined to be the set of all points given by the set of N continuous functions:

$$x^i = f^i\left(u^1, u^2, \ldots, u^M\right)$$

as u^1, u^2, $\ldots$, u^M vary over the ranges $\acute{u}^1 \leq u^1 \leq \grave{u}^1$, $\acute{u}^2 \leq u^2 \leq \grave{u}^2$, $\ldots$, $\acute{u}^M \leq u^M \leq \grave{u}^M$. Hence a curve is just a 1-dimensional subspace. Another set of subspaces of special interest are the $N-1$-dimensional subspaces, which are called *hypersurfaces* by way of analogy with 2-dimensional surfaces defined in 3-dimensional space. Hypersurfaces are of interest because they can be used to *bisect* a space (divide it into two separate regions).

The following piece of shorthand will be useful:

Definition B.1.5. *By definition,* $\partial_i = \frac{\partial}{\partial x^i}$ *and likewise* $\bar{\partial}_{\bar{i}} = \frac{\partial}{\partial \bar{x}^{\bar{i}}}$.

B.1.3 Change of Coordinates

The choice of coordinates in a space V_N is, of course, entirely arbitrary (so long as neighbouring points remain neighbouring), and in no way influences the properties of that space. Indeed, selection of coordinates is little more than a matter of convenience. Hence it would be unwise to restrict ones self to any single coordinate system. To this end, it is necessary to consider the issue of changing coordinates.

Suppose V_N is equipped with coordinates x^i. To change coordinates to a different set $\bar{x}^{\bar{i}}$ requires a differentiable bijective map $\bar{f}^{\bar{i}} : \Re^N \to \Re^N$ so that:

$$\bar{x}^{\bar{i}} = \bar{f}^{\bar{i}}\left(x^i\right) \tag{B.2}$$

which will assign to a point x^i a new set of coordinates $\bar{x}^{\bar{i}}$. It follows immediately from (B.2) that:

$$d\bar{x}^{\bar{i}} = \left(\partial_i \bar{x}^{\bar{i}}\right) dx^i$$

Where the overbar is used to implicitly indicate the coordinate system. That is, an object T^i is implicitly with respect to the coordinate system x^i. The same object with respect to a different coordinate system $\hat{x}^{\hat{i}}$ will be written $\hat{T}^{\hat{i}}$.

As the coordinate change f^i is a bijection, there must exist a coordinate change map $f^i = \bar{f}^{\bar{i}\leftarrow} : \Re^N \to \Re^N$ to reverse the process, so:

$$x^i = f^i\left(\bar{x}^{\bar{i}}\right)$$

is also a change of coordinates; and $\bar{f}^{\bar{i}} \bullet f^i$ and $f^i \bullet \bar{f}^{\bar{i}}$ are identity maps.

The following definitions will be useful later:

Definition B.1.6. *The* Jacobian *of the change of coordinates $\bar{x}^{\bar{i}}$ to $\bar{\bar{x}}^{\bar{\bar{i}}}$ is defined to be:*

$$\underline{\bar{\bar{J}}} = \det\left(\left[\begin{array}{c} \bar{\partial}_{\bar{i}}\bar{\bar{x}}^{\bar{\bar{j}}} \end{array}\right]\right) = \det\left(\left[\begin{array}{cccc} \bar{\partial}_{\bar{1}}\bar{\bar{x}}^{\bar{\bar{1}}} & \bar{\partial}_{\bar{2}}\bar{\bar{x}}^{\bar{\bar{1}}} & \cdots & \bar{\partial}_{\bar{N}}\bar{\bar{x}}^{\bar{\bar{1}}} \\ \bar{\partial}_{\bar{1}}\bar{\bar{x}}^{\bar{\bar{2}}} & \bar{\partial}_{\bar{2}}\bar{\bar{x}}^{\bar{\bar{2}}} & \cdots & \bar{\partial}_{\bar{N}}\bar{\bar{x}}^{\bar{\bar{2}}} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{\partial}_{\bar{1}}\bar{\bar{x}}^{\bar{\bar{N}}} & \bar{\partial}_{\bar{2}}\bar{\bar{x}}^{\bar{\bar{N}}} & \cdots & \bar{\partial}_{\bar{N}}\bar{\bar{x}}^{\bar{\bar{N}}} \end{array}\right]\right)$$

Definition B.1.7. *An object which remains unchanged by any change of coordinates is known as an* invariant.

B.1.4 Scalars, Contravariant Vectors and Covariant Vectors

It is constructive to investigate the set of "geometric objects" which may exist in a space, where a geometric object is something whose physical structure is independent of the coordinate system used, but whose representation may not be.

The simplest geometric object is the scalar, which is simply a number associated with a point P. For example, the height of a point of land above sea level is a scalar. Mathematically, a scalar is a number which, under a change of coordinates from x^i to $\bar{x}^{\bar{i}} = \bar{x}^{\bar{i}}\left(x^i\right)$, transforms as:

$$\bar{T} = T$$

The next most basic geometric object is the vector, which is analogous to a vector in a vector space. Consider two infinitesimally separated points x^i (point P) and $x^i + dx^i$ (point Q). Under change of coordinates, these points become $\bar{x}^{\bar{i}}$ and $\bar{x}^{\bar{i}} + d\bar{x}^{\bar{i}}$, where:

$$d\bar{x}^{\bar{i}} = \left(\partial_i \bar{x}^{\bar{i}}\right) dx^i \tag{B.3}$$

where $\partial_i \bar{x}^{\bar{i}}$ is a function of P but independent of Q. This indicates that, on an infinitesimal scale, the general space V_N has certain properties in common with a real vector space (where dx^i is analogous to a vector).

Indeed, it is possible to construct a *tangent (vector) space* T_P to the point P in V_N. To do this, choose some arbitrary infinitesimal ds. Based on this, construct a set of points Q_I (with coordinates y_I) infinitesimally close to P with coordinates:

$${y_I}^i = x^i + {\delta_I}^i ds$$

where ${\delta_I}^i$ is the Kronecker-delta function:

$${\delta_I}^i = \left\{ \begin{array}{ll} 1 & \text{if } I = i \\ 0 & \text{if } I \neq i \end{array} \right.$$

This gives us a set of arrows $\overrightarrow{PQ_I}$ from the point P to the neighbouring points Q_I. Identifying P with the origin $\mathbf{0}$ of the tangent (vector) space T_P, the arrows $\overrightarrow{PQ_I}$ can then be used to construct a basis $\{\mathbf{e}^i | \, 1 \leq i \leq N\}$ for T_P by identifying:

$$\mathbf{e}^I = \frac{1}{ds}\overrightarrow{PQ_I}$$

Formally, the vector space T_P is then:

$$T_P = \left\{ \sum_{i=1}^{N} a^i \mathbf{e}^i \middle| a^i \in \Re \right\}$$

Using this, the offset dx^i from the point x^i to the point $x^i + dx^i$ may be identified with an infinitesimal vector $d\mathbf{x}$ in the tangent space T_P using the construction:

$$d\mathbf{x} = \sum_{i=1}^{N} dx^i \mathbf{e}^i$$

Roughly speaking, the tangent space T_P is to the space V_N at point P what the tangent is to a curve at a point. The tangent space is a vector space obeying the usual expectations of Cartesian space, as is any infinitesimal region of the space V_N. A vector $\mathbf{u}$ in the tangent space T_P of some point P is known as *contravariant vector* at this point. Formally, a contravariant vector is defined thusly:

Definition B.1.8. *An array T^i at a point P is said to be a* contravariant vector *at P if, under change of coordinates from x^i to $\bar{x}^{\bar{i}} = \bar{x}^{\bar{i}}\left(x^i\right)$:*

$$\bar{T}^{\bar{i}} = \left(\partial_i \bar{x}^{\bar{i}}\right) T^i$$

where $\partial_i \bar{x}^{\bar{i}}$ is evaluated at P.

The following notational convention is useful:

Notation B.1.3. *A* contravariant vector *T^i is always written with an upper (superscript) index.*

It is important to note, however, that whereas the infinitesimal vector dx^i at a point x^i may be validly thought of as a arrow from x^i to $x^i + dx^i$, a finite vector a^i at the same point is *not* an arrow from x^i to $x^i + a^i$. Furthermore, regardless of the notational irregularity, the coordinate x^i is *not* a contravariant vector. Examples of contravariant vectors include the infinitesimal displacement dx^i and the gradient $\frac{dx^i}{du}$ of a curve $x^i\left(u\right)$.

A concept closely related to the contravariant vector is the covariant vector. Whereas the contravariant vector may be thought of as an arrow at a point P, a covariant vector may be thought of as the gradient (at P) of some scalar function defined in the neighbourhood of P, rather like a topographical map about P.

Let ψ be a differentiable scalar function defined about P. At the set of points Q_I defined previously:

$$\psi\left({y_I}^i\right) = \psi\left(x^i\right) + \frac{\partial \psi}{\partial x^I}\left(x^i\right) ds$$

Moreover, $\frac{\partial \psi}{\partial x^I}$ is the rate of change of ψ moving in the direction of the basis vector $\mathbf{e}^I$ in the infinitesimal neighbourhood about P. Using these observations, a linear homogeneous function $\psi_P : T_P \to \Re$ can be constructed using:

$$\psi_P \left(u^i \right) = c_i u^i + d$$

where $u^i \in T_P$, $d = \psi \left(x^i \right)$ and:

$$c_i = \frac{\partial \psi}{\partial x^i} \left(x^k \right)$$

Now, under a change of coordinates:

$$\psi_P \left(\bar{u}^{\bar{i}} \right) = \bar{c}_{\bar{i}} \bar{u}^{\bar{i}} + \bar{d}$$

It is not difficult to show that:

$$\begin{aligned} \bar{d} &= d \\ \bar{c}_{\bar{i}} &= \left(\bar{\partial}_{\bar{i}} x^i \right) c_i \end{aligned}$$

The array c_i is the gradient of a linear homogeneous function in the tangent space T_P. As this is a constant throughout T_P (for a given set of coordinates), c_i may be thought of as a single *covariant vector* defined at the point P. Formally speaking, a covariant vector is defined as:

Definition B.1.9. *An array T_i at a point P is said to be a* covariant vector *at P if, under change of coordinates from x^i to $\bar{x}^{\bar{i}} = \bar{x}^{\bar{i}} \left(x^i \right)$:*

$$\bar{T}_{\bar{i}} = \left(\bar{\partial}_{\bar{i}} x^i \right) T_i$$

where $\bar{\partial}_{\bar{i}} x^i$ is evaluated at P.

The following notational convention is useful:

Notation B.1.4. *A* covariant vector *T_i is always written with an lower (subscript) index.*

Note that given a contravariant vector T^i and a covariant vector S_i at a point the construction:

$$U = T^i S_i$$

is a scalar at that point. Interpreting $S_i = \frac{\partial \psi}{\partial x^i}$, it can be seen that:

$$U = \psi_P \left(T^i \right) - \psi_P \left(\mathbf{0} \right)$$

B.1.5 Relative Tensors and Densities

Scalars, contravariant vectors and covariant vectors are simply special cases of a more general object known as a *relative tensor*, which is defined thusly:

Definition B.1.10. *A* relative tensor *of weight w and order $n + m$ at a point P is* *an ordered array* $\underset{w}{a}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ *defined at* P *that transforms according to the rule:*

$$\underset{w}{\bar{a}}{}^{\bar{i}_1 \bar{i}_2 \ldots \bar{i}_n}_{\bar{j}_1 \bar{j}_2 \ldots \bar{j}_m} = \bar{J}^w \left(\bar{\partial}_{\bar{j}_1} x^{j_1}\right) \left(\bar{\partial}_{\bar{j}_2} x^{j_2}\right) \ldots \left(\bar{\partial}_{\bar{j}_m} x^{j_m}\right) \underset{w}{a}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} \left(\partial_{i_1} \bar{x}^{\bar{i}_1}\right) \left(\partial_{i_2} \bar{x}^{\bar{i}_2}\right) \ldots \left(\partial_{i_n} \bar{x}^{\bar{i}_n}\right)$$

where $\bar{\partial}_{\bar{k}} x^k$, $\partial_k \bar{x}^{\bar{k}}$ *and* $\bar{J}$ *are evaluated at* P.

Note that in an actual relative tensor there is a definite ordering of upper and lower indices, for example $\underset{w}{a}{}^{i}{}_{j}{}^{k}$. This is not done in this prototype to maintain notational simplicity. The underscript w is used to indicate the weight w of the relative tensor. If $w = 0$ this may be left out. The following objects are special cases of relative tensor:

Definition B.1.11. *A* contravariant relative tensor *is a relative tensor order* $n+0$*.*

Definition B.1.12. *A* covariant relative tensor *is a relative tensor of order* $0 + m$*.*

Definition B.1.13. *A* tensor density *is a relative tensor of weight* $+1$*.*

Definition B.1.14. *A* contravariant tensor density *is a tensor density of order* $n+0$*.*

Definition B.1.15. *A* covariant tensor density *is a tensor density of order* $0 + m$*.*

Definition B.1.16. *A* tensor *is a relative tensor of weight* 0*.*

Definition B.1.17. *A* contravariant tensor *is a tensor of order* $n + 0$*.*

Definition B.1.18. *A* covariant tensor *is a tensor of order* $0 + m$*.*

Definition B.1.19. *A* contravariant vector *is a tensor of order* $1 + 0$*.*

Definition B.1.20. *A* covariant vector *is a tensor of order* $0 + 1$*.*

Definition B.1.21. *A* relative scalar *is a relative tensor of order* $0 + 0$*,*

Definition B.1.22. *A* scalar density *is a relative scalar of weight* $+1$*.*

Definition B.1.23. *A* scalar *is a relative scalar of weight* 0*.*

It is essentially trivial to prove that relative tensors defined at some common point P may be combined according to the following rules:

- Addition: Any two relative tensors, $\underset{w}{a}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ and $\underset{w}{b}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ of the same weight w and the same order $n + m$ may be added to form another relative tensor, $\underset{w}{c}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$, also of weight w and order $n + m$, using:

$$\underset{w}{c}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \underset{w}{a}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} + \underset{w}{b}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$$

- Multiplication: Any two relative tensors, $\underset{w}{a}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ and $\underset{v}{b}{}^{i_1 i_2 \ldots i_p}_{j_1 j_2 \ldots j_q}$ of weight w and v and order $n + m$ and $p + q$, respectively, may be multiplied to give a another relative tensor, $\underset{w+v}{c}{}^{i_1 i_2 \ldots i_{n+p}}_{j_1 j_2 \ldots j_{m+q}}$, of weight $w + v$ and order $(n + m) + (p + q)$ using:

$$\underset{w+v}{c}{}^{i_1 i_2 \ldots i_{n+p}}_{j_1 j_2 \ldots j_{m+q}} = \underset{w}{a}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} \underset{v}{b}{}^{i_{n+1} i_{n+2} \ldots i_{n+p}}_{j_{m+1} j_{m+2} \ldots j_{m+q}}$$

- Contraction: Given a relative tensor, $\underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$ of weight w and order $n+m$, another relative tensor, $\underset{w}{c}^{i_1 i_2 \ldots i_{n-1}}_{j_1 j_2 \cdots j_{m-1}}$, of weight w and order $(n-1)+(m-1)$ may be formed by contracting an upper and lower index thusly:

$$\underset{w}{c}^{i_1 i_2 \ldots i_{n-1}}_{j_1 j_2 \cdots j_{m-1}} = \underset{w}{a}^{i_1 i_2 \ldots i_{i-1} k i_{i+1} \ldots i_n}_{j_1 j_2 \cdots j_{j-1} k j_{j+1} \cdots j_m}$$

Note, however, that contraction of indices of the same type (both upper or both lower) is not a well-defined operation, as the result of such a (naively applied) contraction may not, in general, be a tensor density.

The reason such emphasis is placed on tensors is that tensors transform, under change of coordinates, in a linear and homogeneous manner, so if a purely tensorial equation holds in one coordinate system, it will hold in all others. This is known as transitivity of the tensorial character.

Finally, a *relative tensor field* is defined thusly:

Definition B.1.24. *A* relative tensor field *is a tensor assigned to all points in a subspace of* V_N *in a smooth manner.*

So, if the value of the field at a point x^i is $\underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$ then the value at $x^i + dx^i$ for some infinitesimal displacement dx^i (assuming that $x^i + dx^i$ is not outside the subspace in question) must be $\underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m} + d\underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$, where $d\underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$ is infinitesimal. However, as will be seen in subsequent sections, this infinitesimal need not be a tensor in general.

Other fields (tensor fields, relative scalar fields, etc.) are defined in an analogous manner.

A property of all relative tensors which will have some importance later is that any relative tensor $\underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$ of weight w may be re-written thusly:

$$\underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m} = \underset{w}{b} \sum_{k=1}^{N^{n+m}} c_{(k,1)}{}^{i_1} c_{(k,2)}{}^{i_2} \ldots c_{(k,n)}{}^{i_n} d_{(k,1) j_1} d_{(k,2) j_2} \ldots d_{(k,m) j_m} \tag{B.4}$$

where $\underset{w}{b}$ is a (non-zero) relative scalar of weight w, $c_{(k,l)}{}^{i_l}$ (where $1 \leq k \leq N^{n+m}$, $1 \leq l \leq n$) is a contravariant vector and $d_{(k,l) i_l}$ (where $1 \leq k \leq N^{n+m}$, $1 \leq l \leq m$) is a covariant vector.

To see why the decomposition (B.4) must be possible, first note that for any non-zero relative scalar $\underset{w}{b}$ of weight w:

$$\underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m} = \underset{w}{b} h^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$$

where:

$$h^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m} = \underset{w}{b}^{-1} \underset{w}{a}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$$

is a tensor of order $n+m$ (it is trivial to prove that if $\underset{w}{b}$ is a relative scalar of weight w, $\underset{w}{b}^{-1}$ must be a relative scalar of weight $-w$. The fact that $h^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$ is a tensor follows directly).

Now, for the current coordinates, if $n \geq 1$, define:

$$k_{(l)}{}^{i} = \delta^{i}{}_{l}$$

where $1 \leq l \leq N$, and extend these to other coordinates by *defining* their behaviour under change of coordinates to be that of a contravariant vector. Then, also in the current coordinates, define $h^{(l)i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ by:

$$h^{(l)i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \delta_{i_1}{}^{l} h^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$$

where $1 \leq l \leq N$, and once again extend to other coordinates by defining the behaviour under change of coordinates to be that of a tensor of order $(n-1)+m$. Then:

$$h^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \sum_{l=1}^{N} k_{(l)}{}^{i_1} h^{(l)i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$$

The same procedure may then be applied to each of $h^{(l)i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$, and so on for subsequent decompositions until only contravariant vectors and covariant tensors remain. The end result of this process will be the decomposition:

$$h^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \sum_{k=1}^{N^n} c_{(k,1)}{}^{i_1} c_{(k,2)}{}^{i_2} \ldots c_{(k,n)}{}^{i_n} p_{(k) j_1 j_2 \ldots j_m}$$

where $p_{(k) j_1 j_2 \ldots j_m}$ $(1 \leq k \leq N^n)$ are covariant tensors of order $0+m$, and $c_{(k,l)}{}^{i_l}$ $(1 \leq k \leq N^n,\ 1 \leq l \leq n)$ are contravariant vectors. An analogous method may then be applied to decompose these covariant tensors into covariant vectors.

Specifically, suppose $p_{j_1 j_2 \ldots j_m}$, $m > 1$ is a covariant tensor. Then define:

$$\kappa^{(l)}{}_{i} = \delta_{i}{}^{l}$$

where $1 \leq l \leq N$, and extend these to other coordinates by defining their behaviour under change of coordinates to be that of a covariant vector. Then, also in the current coordinates, define $p_{(l) j_2 \ldots j_m}$ by:

$$p_{(l) j_2 \ldots j_m} = \delta^{j_1}{}_{l} p_{j_1 j_2 \ldots j_m}$$

where $1 \leq l \leq N$, and once again extend to other coordinates by defining the behaviour under change of coordinates to be that of a covariant tensor of order $0+(m-1)$. Then:

$$p_{j_1 j_2 \ldots j_m} = \sum_{l=1}^{N} \kappa^{(l)}{}_{i} p_{(l) j_2 \ldots j_m}$$

Applying this recursively, one will eventually arrive at the required decomposition, namely:

$$h^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \sum_{k=1}^{N^{n+m}} {c_{(k,1)}}^{i_1} {c_{(k,2)}}^{i_2} \ldots {c_{(k,n)}}^{i_n} d_{(k,1) j_1} d_{(k,2) j_2} \ldots d_{(k,m) j_m}$$

where ${c_{(k,l)}}^{i_l}$ $(1 \leq k \leq N^{n+m}, 1 \leq l \leq n)$ are contravariant vectors and where $d_{(k,l) j_l}$ $(1 \leq k \leq N^{n+m}, 1 \leq l \leq m)$ are covariant vectors, from which the required result (B.4) follows directly.

The importance of this result is not in the mathematical detail, however. The above construction is not the only possible construction, and definitely not the most useful. The importance lies in the *interpretation* of this result. Specifically, it tells us that any relative tensor is really just a construction of contravariant vectors (arrows), covariant vectors (gradients) which are in some way connected together; and a single relative scalar whose weight w defines how the magnitude of the whole construction depends on the coordinate system of choice. Specifically, under a change of coordinates, the arrows will still point in the same directions (although their representation will change), the gradients will still have the same slope in the same direction (although, again, their representations will change) and any change in the scalar magnitude will be completely dependent on w.

It is worthwhile at this point to explain the use of the term "density" in this context. Physically, densities represent the density of something (tensor or scalar) per unit volume. The Jacobian measures the (inverse) change in the local "scale" of the coordinates under the transform. Hence the $\bar{J}$ factor in the conversion of densities is included to correct for this change in scale. For example, consider a Euclidean 3 dimensional space with orthonormal axis, where 1 unit on any axis represents 1 cm. Further, suppose $\underset{1}{a} = 2000$ is a scalar density constant everywhere representing the number gas molecules per cm^3. Now suppose the coordinate system is changed so that 1 unit on any axis represents 1 mm, rather than 1 cm. This is achieved by the change of coordinates $\bar{x} = \frac{x}{10}$, $\bar{y} = \frac{y}{10}$, $\bar{z} = \frac{z}{10}$. Subsequently:

$$\bar{J} = \det\left(\begin{bmatrix} \frac{1}{10} & 0 & 0 \\ 0 & \frac{1}{10} & 0 \\ 0 & 0 & \frac{1}{10} \end{bmatrix}\right) = \frac{1}{1000}$$

and hence:

$$\underset{1}{\bar{a}} = \bar{J} \underset{1}{a} = 2$$

which just means that there are 2 gas molecules per mm^3, which is of course identical to 2000 gas molecules per cm^3. Hence use of the term "density" is appropriate in this case.

More generally, the term "relative" tensor simply indicates that the value measured is in some way relative to the scale of the axis, where the exact nature of the relationship is controlled by w.

Oriented Relative Tensors

As well as relative tensors (and the related special cases), there is one other set of object which will be of interest here. These are the *oriented* relative tensors, which are defined as follows:

Definition B.1.25. *An* oriented relative tensor *of weight w and order $n+m$ at a point P is an ordered array $\underset{w+}{a}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ defined at P that transforms according to the rule:*

$$\underset{w+}{\bar{a}}{}^{\bar{i}_1 \bar{i}_2 \ldots \bar{i}_n}_{\bar{j}_1 \bar{j}_2 \ldots \bar{j}_m} = \operatorname{sgn}\left(\bar{J}\right) \bar{J}^w \left(\bar{\partial}_{\bar{j}_1} x^{j_1}\right) \left(\bar{\partial}_{\bar{j}_2} x^{j_2}\right) \ldots \left(\bar{\partial}_{\bar{j}_m} x^{j_m}\right) \underset{w+}{a}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} \left(\partial_{i_1} \bar{x}^{\bar{i}_1}\right) \left(\partial_{i_2} \bar{x}^{\bar{i}_2}\right) \ldots \left(\partial_{i_n} \bar{x}^{\bar{i}_n}\right)$$

where $\bar{\partial}_{\bar{k}} x^k$, $\partial_k \bar{x}^{\bar{k}}$ and $\bar{J}$ are evaluated at P.

Oriented tensor densities, oriented scalars etc. are all defined by analogy with definitions B.1.11 to B.1.23. The usual rules of combination of oriented (or non-oriented) relative tensors are defined, noting that:

- The product of two oriented relative tensors is a (non-oriented) relative tensor.
- The product of an oriented relative tensor and a (non-oriented) relative tensor is an oriented relative tensor.
- The result of contracting the indices of a oriented relative tensor is an oriented relative tensor.
- The sum of two oriented relative tensors is an oriented relative tensor. However, the sum of an oriented relative tensor and a (non-oriented) tensor is a non-tensorial object.

Like (non-oriented) relative tensors, oriented relative tensors can be decomposed into a set of covariant and contravariant vectors in the form (B.4). However, in this case b must be an *oriented* relative scalar of weight w. Physically, this means that, in addition to the properties described previously for the relative tensor, an oriented relative tensor has a definite orientation with respect to the coordinate axis.[3]

B.1.6 The Generalised Kronecker-Delta Symbol

I will now present the formal definitions of three objects which play a central role in tensor theory, namely the *Kronecker-Delta symbol*, the *generalised Kronecker-Delta symbol* and the *permutation symbols.*

The Kronecker-Delta symbol is defined to be:

Definition B.1.26. *The* Kronecker-Delta symbol $\delta_i{}^j = \delta^j{}_i$ *is defined to be, for all coordinate systems:*

$$\delta_i{}^j = \delta^j{}_i = \begin{cases} 1 & \textit{if } i = j \\ 0 & \textit{if } i \neq j \end{cases}$$

[3]Technically speaking, the previous example was actually an oriented scalar density (consider a change of coordinates with a negative Jacobian.

Now, by definition, $\delta_i{}^j = \bar{\delta}_{\bar{i}}{}^{\bar{j}}$. Given this, it is not difficult to see that the Kronecker-delta symbol $\delta_i{}^j = \delta^j{}_i$ is an invariant tensor. The invariant nature of the Kronecker delta symbol follows from the definition. To see that it is also a tensor, one need only note that:

$$\left(\bar{\partial}_{\bar{i}} x^i\right)\left(\partial_j \bar{x}^{\bar{j}}\right)\delta_i{}^j = \delta_{\bar{i}}{}^{\bar{j}}$$

The Kronecker-delta symbol may be generalised thusly:[4]

Definition B.1.27. *The* generalised Kronecker-Delta symbol $\delta_{i_1 i_2 \ldots i_n}{}^{j_1 j_2 \ldots j_n}$ *is defined to be, for all coordinate systems:*

$$\begin{aligned}\delta_{i_1 i_2 \ldots i_n}{}^{j_1 j_2 \ldots j_n} &= \begin{cases} 0 & \text{if } i_k = i_l, k \neq l \text{ or } \{i_1, i_2, \ldots, i_n\} \neq \{j_1, j_2, \ldots, j_n\} \\ +1 & \text{if } (i_1, i_2, \ldots, i_n) \text{ is an even perm. of } (j_1, j_2, \ldots, j_n) \\ -1 & \text{if } (i_1, i_2, \ldots, i_n) \text{ is an odd perm. of } (j_1, j_2, \ldots, j_n) \end{cases} \\ &= \delta^{j_1 j_2 \ldots j_n}{}_{i_1 i_2 \ldots i_n}\end{aligned} \tag{B.5}$$

It is not difficult to see that:

$$\delta_{i_1 i_2 \ldots i_n}{}^{j_1 j_2 \ldots j_n} = \det\left(\begin{bmatrix} \delta_{i_1}{}^{j_1} & \delta_{i_2}{}^{j_1} & \cdots & \delta_{i_n}{}^{j_1} \\ \delta_{i_1}{}^{j_2} & \delta_{i_2}{}^{j_2} & \cdots & \delta_{i_n}{}^{j_2} \\ \vdots & \vdots & \ddots & \vdots \\ \delta_{i_1}{}^{j_n} & \delta_{i_2}{}^{j_n} & \cdots & \delta_{i_n}{}^{j_n} \end{bmatrix}\right)$$

from which it follows that $\delta_{i_1 i_2 \ldots i_n}{}^{j_1 j_2 \ldots j_n}$ is a invariant tensor of order $n + n$. It can be shown that:

$$\delta^{k_1 k_2 \ldots k_p j_1 j_1 \ldots j_{N-p}}{}_{j_1 j_2 \ldots j_{N-p} i_1 i_1 \ldots i_p} = p!(N-p)!\delta^{k_1}{}_{i_1}\delta^{k_2}{}_{i_2}\ldots\delta^{k_p}{}_{i_p} \tag{B.6}$$

$$\delta_{k_1 k_2 \ldots k_p j_1 j_1 \ldots j_{N-p}}{}^{j_1 j_2 \ldots j_{N-p} i_1 i_1 \ldots i_p} = p!(N-p)!\delta_{k_1}{}^{i_1}\delta_{k_2}{}^{i_2}\ldots\delta_{k_p}{}^{i_p} \tag{B.7}$$

In the special case $n = N$, it is not difficult to see that one may write:

$$\delta_{i_1 i_2 \ldots i_N}{}^{j_1 j_2 \ldots j_N} = \epsilon_{i_1 i_2 \ldots i_N}\epsilon^{j_1 j_2 \ldots j_N} \tag{B.8}$$

where $\epsilon_{i_1 i_2 \ldots i_N}$ and $\epsilon^{j_1 j_2 \ldots j_N}$ are the *permutation symbols*, which are defined thusly:

Definition B.1.28. *The* permutation symbols $\epsilon_{i_1 i_2 \ldots i_N}$ *and* $\epsilon^{j_1 j_2 \ldots j_N}$ *are defined to*

[4] An ordered set $(a_1, a_2, \ldots, a_m)$ is said to be an odd permutation of some other set $(b_1, b_2, \ldots, b_m)$ of like elements (i.e. $\{a_1, a_2, \ldots, a_m\} = \{b_1, b_2, \ldots, b_m\}$) if the particular ordering $(a_1, a_2, \ldots, a_m)$ nay be constructed from $(b_1, b_2, \ldots, b_m)$ by choosing two elements, swapping them, and repeating the process an odd number of times. Otherwise, the permutation is said to be even.

be, for all coordinate systems:

$$\begin{aligned}
\epsilon_{i_1 i_2 \ldots i_N} &= \begin{cases} 0 & \textit{if } i_k = i_l, k \neq l \\ +1 & \textit{if } (i_1, i_2, \ldots, i_N) \textit{ is an even perm. of } (1,2,\ldots,N) \\ -1 & \textit{if } (i_1, i_2, \ldots, i_N) \textit{ is an odd perm. of } (1,2,\ldots,N) \end{cases} \qquad \text{(B.9)} \\
&= \delta_{i_1 i_2 \ldots i_N}{}^{1\ 2\ \ldots\ N} \\
\epsilon^{j_1 j_2 \ldots j_N} &= \begin{cases} 0 & \textit{if } j_k = j_l, k \neq l \\ +1 & \textit{if } (j_1, j_2, \ldots, j_N) \textit{ is an even perm. of } (1,2,\ldots,N) \\ -1 & \textit{if } (j_1, j_2, \ldots, j_N) \textit{ is an odd perm. of } (1,2,\ldots,N) \end{cases} \qquad \text{(B.10)} \\
&= \delta_{1\ 2\ \ldots\ N}{}^{j_1 j_2 \ldots j_N}
\end{aligned}$$

I will now demonstrate that $\epsilon^{i_1 i_2 \ldots i_N}$ is an invariant relative tensor of weight -1 and order $N+0$, and $\epsilon_{j_1 j_2 \ldots j_N}$ an invariant relative tensor of weight $+1$ and order $0+N$ (i.e. an invariant tensor density). To do so, consider a tensor a_{ij}, such that the matrix of its elements $[a_{ij}]$ has a positive determinant. Define $\underset{-2}{a}$ to be this determinant - that is, $\underset{-2}{a} = \det([a_{ij}])$. Then, under the usual change of coordinates:

$$\bar{a}_{\bar{i}\bar{j}} = \left(\bar{\partial}_{\bar{i}} x^i\right)\left(\bar{\partial}_{\bar{j}} x^j\right) a_{ij}$$

or, in matrix notation:

$$\left[\bar{a}_{\bar{i}\bar{j}}\right] = \left[\bar{\partial}_{\bar{i}} x^i\right][a_{ij}]\left[\bar{\partial}_{\bar{j}} x^j\right]$$

Hence:

$$\underset{-2}{\bar{a}} = \bar{J}^{-2}\underset{-2}{a} \qquad \text{(B.11)}$$

and so $\underset{-2}{a}$ must be a relative scalar of weight -2 which is positive in all coordinate systems. However, by definition of the determinant:

$$\begin{aligned}
\underset{-2}{a} &= \epsilon^{i_1 i_2 \ldots i_N} \epsilon^{j_1 j_2 \ldots j_N} a_{i_1 j_1} a_{i_2 j_2} \ldots a_{i_N j_N} \qquad \text{(B.12)} \\
\underset{-2}{\bar{a}} &= \bar{\epsilon}^{\bar{i}_1 \bar{i}_2 \ldots \bar{i}_N} \bar{\epsilon}^{\bar{j}_1 \bar{j}_2 \ldots \bar{j}_N} \bar{a}_{\bar{i}_1 \bar{j}_1} \bar{a}_{\bar{i}_2 \bar{j}_2} \ldots \bar{a}_{\bar{i}_N \bar{j}_N} \\
&= \bar{\epsilon}^{\bar{i}_1 \bar{i}_2 \ldots \bar{i}_N} \bar{\epsilon}^{\bar{j}_1 \bar{j}_2 \ldots \bar{j}_N} \ldots \\
&\quad \ldots \left(\bar{\partial}_{\bar{i}_1} x^{i_1}\right)\left(\bar{\partial}_{\bar{j}_1} x^{j_1}\right) a_{i_1 j_1} \left(\bar{\partial}_{\bar{i}_2} x^{i_2}\right)\left(\bar{\partial}_{\bar{j}_2} x^{j_2}\right) a_{i_2 j_2} \ldots \left(\bar{\partial}_{\bar{i}_N} x^{i_N}\right)\left(\bar{\partial}_{\bar{j}_N} x^{j_N}\right) a_{i_N j_N} \\
&= \left(\left(\bar{\partial}_{\bar{i}_1} x^{i_1}\right)\left(\bar{\partial}_{\bar{i}_2} x^{i_2}\right)\left(\bar{\partial}_{\bar{i}_N} x^{i_N}\right) \bar{\epsilon}^{\bar{i}_1 \bar{i}_2 \ldots \bar{i}_N}\right) \ldots \\
&\quad \ldots \left(\left(\bar{\partial}_{\bar{j}_1} x^{j_1}\right)\left(\bar{\partial}_{\bar{j}_2} x^{j_2}\right)\left(\bar{\partial}_{\bar{j}_N} x^{j_N}\right) \bar{\epsilon}^{\bar{j}_1 \bar{j}_2 \ldots \bar{j}_N}\right) a_{i_1 j_1} a_{i_2 j_2} \ldots a_{i_N j_N} \qquad \text{(B.13)}
\end{aligned}$$

It follows from (B.11) and (B.12) that:

$$\begin{aligned}
\underset{-2}{\bar{a}} &= \bar{J}^{-2}\underset{-2}{a} \\
&= \left(\bar{J}^{-1}\epsilon^{i_1 i_2 \ldots i_N}\right)\left(\bar{J}^{-1}\epsilon^{j_1 j_2 \ldots j_N}\right) a_{i_1 j_1} a_{i_2 j_2} \ldots a_{i_N j_N} \qquad \text{(B.14)}
\end{aligned}$$

Comparing (B.13) and (B.14), it can be seen that:

$$\bar{\epsilon}^{\bar{i}_1 \bar{i}_2 \ldots \bar{i}_N} = \bar{J}^{-1}\left(\partial_{i_1} \bar{x}^{\bar{i}_1}\right)\left(\partial_{i_2} \bar{x}^{\bar{i}_2}\right) \ldots \left(\partial_{i_N} \bar{x}^{\bar{i}_N}\right) \epsilon^{i_1 i_2 \ldots i_N}$$

which proves that $\epsilon^{i_1 i_2 \ldots i_N}$ is an invariant relative tensor of weight -1 and order $N+0$ (the invariant nature following from the definition). An analogous argument based on a tensor b^{ij} may be used to demonstrate that $\epsilon_{j_1 j_2 \ldots j_N}$ is an invariant relative tensor of weight $+1$ and order $0+N$ (an invariant tensor density).

B.1.7 Symmetries and Symmetrisations

Given a tensor (or some other indexed object, in general), it will occasionally be useful to extract the completely symmetric (unchanged by the swapping of any 2 indices) and completely anti-symmetric (negated by the swapping of any 2 indices) parts. For example, given T_{ij}, the following:

$$\begin{aligned} T_{\{ij\}} &= \frac{1}{2}\left(T_{ij} + T_{ji}\right) \\ T_{[ij]} &= \frac{1}{2}\left(T_{ij} - T_{ji}\right) \end{aligned}$$

are, respectively, the completely symmetric and completely anti-symmetric components of T_{ij} (noting that $T_{\{ij\}} = T_{\{ji\}}$ and $T_{[ij]} = -T_{[ji]}$). Note that $T_{ij} = T_{\{ij\}} + T_{[ij]}$, and that if T_{ij} is a tensor, so too will be $T_{\{ij\}}$ and $T_{[ij]}$.

The same shorthand is applied to upper indices. For example:

$$\begin{aligned} S^{\{ij\}} &= \frac{1}{2}\left(S^{ij} + S^{ji}\right) \\ S^{[ij]} &= \frac{1}{2}\left(S^{ij} - S^{ji}\right) \end{aligned}$$

and once again $S^{ij} = S^{\{ij\}} + S^{[ij]}$, and any tensorial character is retained.

For mixed indices, some care is needed. Specifically, the symmetrisation and antisymmetrisation operations can be defined only if the relevant raising/lowering operation is defined (see section B.2). The same shorthand is used, with a dot $\bullet$ acting as a placeholder for the indices not at the same level as the relevant bracket. For example:

$$\begin{aligned} U^{\{i\bullet\}}{}_{j} &= \frac{1}{2}\left(U^{i}{}_{j} + U_{j}{}^{i}\right) \\ V_{[i\bullet]}{}^{j} &= \frac{1}{2}\left(V_{i}{}^{j} - V^{j}{}_{i}\right) \end{aligned}$$

where as usual either tensor is the sum of it's symmetric and anti-symmetric parts, and tensorial character is retained.

Given an object with more than two indices the same shorthand may be applied, for example:

$$\begin{aligned} S^{[ij]k} &= \frac{1}{2}\left(S^{ijk} - S^{jik}\right) \\ T_{i\{j\bullet\}}^{\ \ k} &= \frac{1}{2}\left(T_{ij}^{\ \ k} + T_{i\ \,j}^{\ k}\right) \\ V_{[i}^{\ \,j}W^{k}_{\ \,l]m} &= \frac{1}{2}\left(V_i^{\ j}W^k_{\ lm} - V_l^{\ j}W^k_{\ im}\right) \end{aligned}$$

where the final example illustrates the need for the placeholder, •, to avoid any possible ambiguity. In all of these examples (and indeed all cases where only two indices are affected), the original indexed object can be completely reconstructed by adding together its completely symmetric and completely anti-symmetric components, and tensorial character is retained.

More generally, much the same procedure may be applied to more than two indices at once. However, it should be noted that if more than two indices are involved then there is no way to re-construct the original object using only its completely symmetric and completely anti-symmetric parts (although the any tensorial character will be retained). This is done using the generalised kronecker-delta symbols thusly:

$$\begin{aligned} T_{[i_1 i_2 \ldots i_m]} &= \frac{1}{m!}\delta_{i_1 i_2 \ldots i_m}^{\ \ \ \ \ \ \ j_1 j_2 \ldots j_m} T_{j_1 j_2 \ldots j_m} \\ S^{\{i_1 i_2 \ldots i_m\}} &= \frac{1}{m!}\left|\delta_{j_1 j_2 \ldots j_m}^{\ \ \ \ \ \ \ i_1 i_2 \ldots i_m}\right| S^{j_1 j_2 \ldots j_m} \end{aligned}$$

which can be extended to mixed indices (tensors only), etc., as required.

To avoid confusion, the following alternative notation will occasionally be used:

$$\begin{aligned} T_{\underbrace{i_1 i_2 \ldots i_m}} &= \frac{1}{m!}\delta_{i_1 i_2 \ldots i_m}^{\ \ \ \ \ \ \ j_1 j_2 \ldots j_m} T_{j_1 j_2 \ldots j_m} \\ S^{\underline{i_1 i_2 \ldots i_m}} &= \frac{1}{m!}\left|\delta_{j_1 j_2 \ldots j_m}^{\ \ \ \ \ \ \ i_1 i_2 \ldots i_m}\right| S^{j_1 j_2 \ldots j_m} \end{aligned}$$

so:

$$\begin{aligned} T_{\underbrace{i_1 i_2 \ldots i_m}} &= T_{[i_1 i_2 \ldots i_m]} \\ S^{\underline{i_1 i_2 \ldots i_m}} &= T_{\{i_1 i_2 \ldots i_m\}} \end{aligned}$$

The symmetry properties of an array can often be used to advantage by the use of some common "tricks". In particular, it is not uncommon to encounter expressions of the form:

$$A^{\cdots}_{\cdots} = B^{\cdots ij \cdots}_{\cdots} C^{\cdots}_{\cdots ij \cdots}$$

Now, suppose that $B^{\cdots ij \cdots}_{\cdots}$ is completely symmetric in the indices ij, and $C^{\cdots}_{\cdots ij \cdots}$

completely anti-symmetric in the indices ij. Then:

$$\begin{aligned} A^{\cdots}_{\cdots} &= B^{\cdots ij \cdots}_{\cdots} C^{\cdots}_{\cdots ij \cdots} \\ &= -B^{\cdots ij \cdots}_{\cdots} C^{\cdots}_{\cdots ji \cdots} \\ &= -B^{\cdots ji \cdots}_{\cdots} C^{\cdots}_{\cdots ji \cdots} \end{aligned}$$

but the indices ij are just placeholders in the implied summation. So they may be swapped, and hence:

$$\begin{aligned} A^{\cdots}_{\cdots} &= B^{\cdots ij \cdots}_{\cdots} C^{\cdots}_{\cdots ij \cdots} \\ &= -B^{\cdots ji \cdots}_{\cdots} C^{\cdots}_{\cdots ji \cdots} \\ &= -B^{\cdots ij \cdots}_{\cdots} C^{\cdots}_{\cdots ij \cdots} \end{aligned}$$

which can only be true if $A^{\cdots}_{\cdots} = 0$.

Exactly the same reasoning may be applied if $B^{\cdots ij \cdots}_{\cdots}$ is completely anti-symmetric in the indices ij, and $C^{\cdots}_{\cdots ij \cdots}$ completely symmetric in the indices ij. The upshot of this is that any such double contraction, where one pair (either both superscript or both subscript) is anti-symmetric and the other symmetric will give a result of zero.

It is also worth noting that, given A^i, the product $A^i A^j$ is completely symmetric. Likewise, given B_i, the product $B_i B_j$ is completely symmetric.

B.1.8 p-Cells, p-Forms and Duality

So far, I have concentrated on the contravariant and covariant vectors as the primitives from which other objects (relative tensors, oriented densities etc.) may be constructed. However, this picture is somewhat misleading, as arrows (contravariant vectors) and gradients (covariant vectors) are most definitely not the only geometric objects which one may be interested in. In this section I will introduce more general objects, known as forms that, while trivially expressible in terms of vectorial primitives, represent distinct geometrical objects in themselves.

I will start with contravariant p-forms. An infinitesimal contravariant vector a^i represents the primitive of the directed line element - it has a beginning and an end (in tangent space), and also a direction (the beginning and end are distinct). Moreover, the infinitesimal offset dx^i may be seen to be the primitive for 1-dimensional objects (curves) - the tuple dx^i at a point x^i tells how to get from this point to another point nearby, using an implicitly defined set of tangent axis at this point. Another “simple” shape in tangent space is the p dimensional parallelogram, or p-cell, as shown for example in figure B.1. Using infinitesimal p-cells, an arbitrary p-dimensional region (subspace) in space may be defined. That is, p-cells are the primitive objects from which such structures may be made.

Consider the infinitesimal 2-dimensional parallelogram (2-surface/cell) shown in figure B.2. This is partially characterised by the offset vectors $d_{(1)}x^i$ and $d_{(2)}x^i$ from which it was constructed. However, it also has a characteristic sign (direction) (one could say that a particular side is facing forwards, and then reverse this), which is dependent on the precise ordering of the offsets $d_{(1)}x^i$ and $d_{(2)}x^i$. Hence, the parallelogram corresponding to $d_{(2)}x^i$ and $d_{(1)}x^i$ (i.e. order reversed) is in a sense

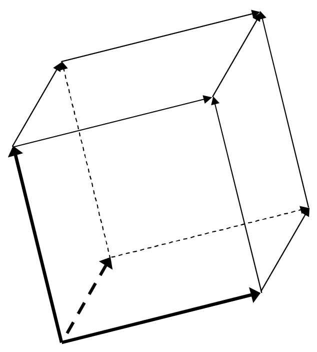


Figure B.1: A simple 3-cell.

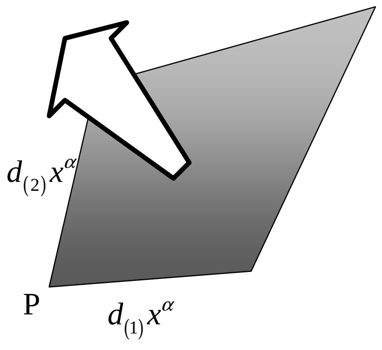


Figure B.2: Construction of an infinitesimal parallelogram. The direction here can be determined by taking your right hand, pointing the thumb in the direction of the first offset $d_{(1)}x^i$ and your fingers in the direction of the second offset $d_{(2)}x^i$. The palm of your hand is then facing in the "direction" of the parallelogram.

the negative of the original.

Mathematically, the infinitesimal 2-surface defined by the ordered pair $d_{(1)}x^i$ and $d_{(2)}x^i$ (at P) is defined to be $dA^{i_1 i_2} = d_{(1)}x^{[i_1}d_{(2)}x^{i_2]}$, which has $\frac{N(N+1)}{2}$ independent components. The basis for these components are ordered pairs of tangent axis, so any element of $dA^{I_1 I_2}$ is the signed area of the complete parallelogram when projected onto the 2-surface defined by that (ordered) pair of tangent axis $\left(\mathbf{e}^{I_1}, \mathbf{e}^{I_2}\right)$. In this way, any anti-symmetric $2+0$ order tensor A^{ij} may be thought to represent a 2-surface like object in tangent space.

Note that:

$$\begin{aligned} d_{(1)}x^{[i_1}d_{(2)}x^{i_2]} &= -d_{(2)}x^{[i_1}d_{(1)}x^{i_2]} \\ d_{(1)}x^{[i_1}d_{(1)}x^{i_2]} &= 0 \end{aligned}$$

which implies that this object is indeed signed and, furthermore, that a parallelogram constructed by two parallel offsets is not a parallelogram (insofar as it is a 1-dimensional object with 0 area).

This can be extended to an arbitrary infinitesimal p-cell constructed from p infinitesimal offsets $d_{(i)}x^j$ via:

$$d\tau_{(p)}{}^{i_1 i_2 \ldots i_p} = d_{(1)}x^{[i_1}d_{(2)}x^{i_2} \ldots d_{(p)}x^{i_p]}$$

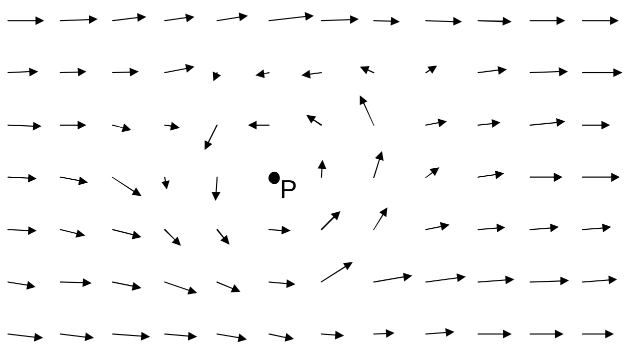


Figure B.3: The rotation of a fluid in a fluid in a 2 dimensional space about a point P

which has as it's basis the ordered p-tuples of tangent axis $(\mathbf{e}^{i_1}, \mathbf{e}^{i_2}, \ldots, \mathbf{e}^{i_p})$. Note that unless the set of offsets $d_{(i)}x^j$ are independent $d\tau_{(p)}{}^{i_1 i_2 \ldots i_p} = 0$, and hence it is not possible to construct nonzero p-cells where $p > N$. This is reasonable, as such a p-cell would be unphysical.

From this, it may be seen that any completely antisymmetric $p+0$ order tensor will represent a finite p-cell in tangent space. Such an object is technically known as a *contravariant p-form* of weight 0, where a contravariant p-form of weight w is defined thusly:

Definition B.1.29. *A* contravariant p-form *of weight w is a completely antisymmetric relative tensor of weight w and order $p+0$.*

Closely related to this is the *covariant p-form.* Now, a covariant vector r_i represents the (local) rate of change of some scalar function. Physically, a more useful way of picturing this is to imagine a fluid permeating our N-dimensional space. Then a covariant vector r_i at a point P represents the rate of flow of that fluid going past P - that is, each element r_I represents that rate of flow of the fluid along the axis $\mathbf{e}^I$ at P.

Of course, a covariant vector r_i at a point cannot completely capture the flow of a fluid at that point. For example, suppose that space is 2 dimensional, and that our fluid is rotating about the point P, as shown in figure B.3. In this case, there is no fluid flow along any axis at P, but none-the-less there is something non-trivial happening at this point!

Once again, the key to understanding and describing such situations lies in considering objects whose basis are ordered sets of basis vectors, for example $(\mathbf{e}^i, \mathbf{e}^j)$. The flow in the above example may be described using a completely antisymmetric covariant tensor A_{ij}. In this simple 2-dimensional example, $A_{I_1 I_2}$ represents that rate of flow of the fluid around the 2-cell defined by the ordered pair $\left(\mathbf{e}^{I_1}, \mathbf{e}^{I_2}\right)$ - leaving P in the direction of $\mathbf{e}^{I_1}$ and returning along $-\mathbf{e}^{I_2}$ after circumnavigating the 2-cell. So, using standard cartesian coordinates, A_{xy} is the anticlockwise rate of rotation of the fluid, and $A_{yx} = -A_{xy}$ the clockwise rate of rotation.

This extends to the rate of flow (rotation) around arbitrary p-cells in N dimensional space (which is unfortunately rather difficult to visualise for $p \geq 4$), this being represented by a *covariant p-form* of weight 0, where a covariant p-form of weight w is defined by:

Definition B.1.30. *A* covariant p-form *of weight w is a completely antisymmetric relative tensor of weight w and order* $0+p$.

There is a close connection between contravariant and covariant p-forms, known as duality. Suppose $\underset{w}{f}^{i_1i_2\ldots i_p}$ is a contravariant p-form of weight w. The *dual* $\underset{w}{\overset{d}{f}}{}^{i_1i_2\ldots i_p} = \underset{w+1}{f}{}_{j_1j_2\ldots j_{N-p}}$ of $\underset{w}{f}^{i_1i_2\ldots i_p}$ is defined to be:

$$\underset{w+1}{f}{}_{j_1j_2\ldots j_{N-p}} = \underset{w}{\overset{d}{f}}{}^{i_1i_2\ldots i_p} = \frac{1}{p!}\epsilon_{j_1j_2\ldots j_{N-p}i_1i_1\ldots i_p}\underset{w}{f}^{i_1i_2\ldots i_p}$$

and is a covariant $(N-p)$-form of weight $w+1$. Similarly, if $\underset{w}{f}{}_{i_1i_2\ldots i_p}$ is a covariant p-form of weight w, the *dual* $\underset{w-1}{f}{}^{j_1j_2\ldots j_{N-p}} = \underset{w}{\overset{d}{f}}{}_{i_1i_2\ldots i_p}$ of $\underset{w}{f}{}_{i_1i_2\ldots i_p}$ is defined to be:

$$\underset{w-1}{f}{}^{j_1j_2\ldots j_{N-p}} = \underset{w}{\overset{d}{f}}{}_{i_1i_2\ldots i_p} = \frac{1}{p!}\epsilon^{j_1j_2\ldots j_{N-p}i_1i_1\ldots i_p}\underset{w}{f}{}_{i_1i_2\ldots i_p}$$

and is a contravariant $(N-p)$-form of weight $w-1$. Because of the notational ambiguity involved here, the use of overscript d to indicate duality will be avoided where possible. The dual of an oriented object will also be oriented.

Using (B.6) and (B.7), it can be seen that:

$$\begin{aligned}
\underset{w}{\overset{dd}{f}}{}^{k_1k_2\ldots k_p} &= \frac{1}{(N-p)!}\epsilon^{k_1k_2\ldots k_pj_1j_1\ldots j_{N-p}}\underset{w}{\overset{d}{f}}{}^{j_1j_2\ldots j_p} \\
&= \frac{1}{p!\,(N-p)!}\epsilon^{k_1k_2\ldots k_pj_1j_1\ldots j_{N-p}}\epsilon_{j_1j_2\ldots j_{N-p}i_1i_1\ldots i_p}\underset{w}{f}^{i_1i_2\ldots i_p} \\
&= \frac{1}{p!\,(N-p)!}\delta^{k_1k_2\ldots k_pj_1j_1\ldots j_{N-p}}{}_{j_1j_2\ldots j_{N-p}i_1i_1\ldots i_p}\underset{w}{f}^{i_1i_2\ldots i_p} \\
&= \underset{w}{f}^{k_1k_2\ldots k_p} \\
\underset{w}{\overset{dd}{f}}{}_{k_1k_2\ldots k_p} &= \frac{1}{(N-p)!}\epsilon_{k_1k_2\ldots k_pj_1j_1\ldots j_{N-p}}\underset{w}{\overset{d}{f}}{}_{j_1j_2\ldots j_p} \\
&= \frac{1}{p!\,(N-p)!}\epsilon_{k_1k_2\ldots k_pj_1j_1\ldots j_{N-p}}\epsilon^{j_1j_2\ldots j_{N-p}i_1i_1\ldots i_p}\underset{w}{f}{}_{i_1i_2\ldots i_p} \\
&= \frac{1}{p!\,(N-p)!}\delta_{k_1k_2\ldots k_pj_1j_1\ldots j_{N-p}}{}^{j_1j_2\ldots j_{N-p}i_1i_1\ldots i_p}\underset{w}{f}{}_{i_1i_2\ldots i_p} \\
&= \underset{w}{f}{}_{k_1k_2\ldots k_p}
\end{aligned}$$

Clearly, computing the dual of a p-form leaves the geometric information stored by that p-form essentially untouched (otherwise the operation would not be reversible). So what is the connection between a p-form and its dual?

This question can be answered by considered some examples. First off, consider the infinitesimal contravariant 1-form a^i in a 3-dimensional space, which is a contravariant vector in tangent space. The dual of this is a covariant 2-form in the same tangent space, representing a fluid rotation around (but not along) the direction a^i - that is, a flow around an 2-cell perpendicular to a^i. More generally, the dual of any contravariant 1-form in an N dimensional space is a covariant $(N-1)$-form representing the flow of some fluid about an $(N-1)$-cell which is perpendicular to direction of the original contravariant 1-form (which in the 2-dimensional case just means flow at right angles to the original contravariant 1-form).

Similarly, it is not hard to see that the dual of an contravariant $(N-1)$-form will be a covariant 1-form representing flow perpendicularly through the original contravariant $(N-1)$-cell in the direction of the $(N-1)$-cell. More generally, the duals of all contravariant forms of order 0 to $N-1$ may be visualised similarly as flows essentially "perpendicular" to some cell (be they linear flows, 2 dimensional rotations or higher order rotations), represented by the original contravariant form.

A case of special interest is the dual of the infinitesimal contravariant N-form $d\tau_{(N)}{}^{i_1 j_2 \ldots j_N}$ defined previously, which is an infinitesimal scalar density:

$$\begin{aligned}
\overset{d}{d\tau}_{(N)}{}^{i_1 i_2 \ldots i_N} &= \epsilon_{i_1 i_2 \ldots i_N} d\tau_{(N)}{}^{i_1 i_2 \ldots i_N} \\
&= \epsilon_{i_1 i_2 \ldots i_N} dx_1^{[i_1} dx_2^{i_2} \ldots dx_N^{i_N]} \\
&= \frac{1}{N!} \epsilon_{i_1 i_2 \ldots i_N} \delta^{i_1 i_2 \ldots i_N}{}_{j_1 j_2 \ldots j_N} dx_1^{j_1} dx_2^{j_2} \ldots dx_N^{j_N} \\
&= \frac{1}{N!} \delta_{i_1 i_2 \ldots i_N}{}^{i_1 i_2 \ldots i_N} \epsilon_{j_1 j_2 \ldots j_N} dx_1^{j_1} dx_2^{j_2} \ldots dx_N^{j_N} \\
&= \epsilon_{i_1 i_2 \ldots i_N} dx_1^{i_1} dx_2^{i_2} \ldots dx_N^{i_N}
\end{aligned}$$

Now, if $dx_I{}^j = \delta_I{}^j dy$ then:

$$\overset{d}{d\tau}_{(N)}{}^{i_1 i_2 \ldots i_N} = dy^N \tag{B.15}$$

For the standard Cartesian basis, (B.15) is obviously the standard infinitesimal volume element. However, for other basis, this is not true, which is clear from the fact that $\overset{d}{d\tau}_{(N)}{}^{i_1 i_2 \ldots i_N}$ is a density, whereas one would expect the volume of an N-cell to be an oriented scalar. The drive to construct a scalar measure of the volume of an N-cell leads to the construction of a much more practical definition of duality in section B.2.4.

B.2 The Metric Tensor

B.2.1 Magnitude and Angle in Vector Spaces

Consider a real vector space with standard Cartesian basis. In this vector space the length of a vector a^i is usually (i.e. in the usual Euclidean setting) defined to be:

$$\left\| a^i \right\| = \sqrt{\delta_{ij} a^i a^j}$$

Now, under an arbitrary change of basis:

$$\left|\bar{a}^{\bar{i}}\right|^2 = \left|a^i\right|^2 = \bar{g}_{\bar{i}\bar{j}}\bar{a}^{\bar{i}}\bar{a}^{\bar{j}}$$

where:

$$\bar{g}_{\bar{i}\bar{j}} = \delta_{ij}\bar{b}_{\bar{i}}^{\ i}\bar{b}_{\bar{j}}^{\ j}$$

or, in matrix notation:

$$\left[\bar{g}_{\bar{i}\bar{j}}\right] = \left[\bar{b}_{\bar{i}}^{\ i}\right]\left[\bar{b}_{\bar{j}}^{\ i}\right]^T$$

The only limitation on the change of coordinates is that the matrix $\left[\bar{b}_{\bar{i}}^{\ i}\right]$ is a non-singular. Given this, $\left[\bar{g}_{\bar{i}\bar{j}}\right]$ can be made to be (by choosing an appropriate change of coordinates) any *symmetric positive definite* matrix.

Rather than starting with a standard Cartesian basis, one may instead simply define the length of a vector a^i in some arbitrary coordinate system to be:

$$\left\|a^i\right\| = \sqrt{g_{ij}a^ia^j}$$

where g_{ij} is some array defined such that $[g_{ij}]$ is symmetric and positive definite. Changing to another basis this becomes:

$$\begin{aligned}\left|\bar{a}^{\bar{i}}\right|^2 &= \left|a^i\right|^2\\ &= \bar{g}_{\bar{i}\bar{j}}\bar{a}^{\bar{i}}\bar{a}^{\bar{j}}\end{aligned}$$

where:

$$\bar{g}_{\bar{i}\bar{j}} = g_{ij}\bar{b}_{\bar{i}}^{\ i}\bar{b}_{\bar{j}}^{\ j} \tag{B.16}$$

In this manner, one may define a cartesian basis (and cartesian coordinates) to be the basis for which $g_{ij} = \delta_{ij}$. So in this sense g_{ij} *defines* the basis of the vector space, and if $g_{ij} = \delta_{ij}$ then the basis so defined is the usual orthonormal Cartesian basis.

More generally (when considering the geometry of special relativity, for example) a vector may have a negative magnitude (i.e. $\left|a^i\right|^2 < 0$). To construct this magnitude, start with:

$$\left|a^i\right|^2 = \gamma_{ij}a^ia^j$$

where:

$$\gamma_{ij} = \begin{cases} \pm 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

It is useful to define:

Definition B.2.1. *The* signature *of a vector space is defined to be* $(\gamma_{11}, \gamma_{22}, \ldots, \gamma_{NN})$.

The effect of this modification is that g_{ij} need not be positive definite so long as $[g_{ij}]$ is non-singular and symmetric. Of course, if $\left|a^i\right|^2 < 0$ then one cannot simply take the square root to find the length. Instead, the length may be defined thusly:

$$\left\| a^i \right\| = \sqrt{\left| g_{ij} a^i a^j \right|}$$

However, it should be noted that in this case the sign of the magnitude $g_{ij} a^i a^j$ is also informative. In special relativity, for example, this sign defines the difference between a timelike vectors (for which $g_{ij} a^i a^j < 0$) and spacelike vectors (for which $g_{ij} a^i a^j > 0$).

Given two tensors a^i and b^i the inner (dot) product for a general basis is:

$$\left\langle a^i, b^j \right\rangle = g_{ij} a^i b^j$$

which has the usual interpretation as the squared length of the vector a^i projected onto b^i. Using this, the angle θ between the vectors a^i and b^i is given by:

$$\cos\theta = \frac{g_{ij} a^i b^j}{\left\| a^k \right\| \left\| b^k \right\|}$$

Some care is warranted here, however, as if $g_{ij} a^i b^j < 0$ then the resultant angle θ will be complex as defined by the above formula.

B.2.2 The Metric Tensor (Field)

As discussed previously, in a general space each point P has associated with it a tangent vector space T_P, which is a vector space at that point rather like the tangent to a curve at a point. Furthermore, in the infinitesimal neighbourhood of a point P, the offsets dx^i act like the vectors in this tangent space.

One may take this one step further and require that the space be *locally Euclidean* or, more generally, *locally metric* (making the space a *metric* space equipped with *Reimannian geometry*). This is done by defining the length ds of an offset dx^i (which is the distance between the points x^i and $x^i + dx^i$) using the standard formulas of the previous section. That is:

$$ds^2 = g_{ij} dx^i dx^j$$

where g_{ij} is defined at the point x^i such that $[g_{ij}]$ is symmetric and non-singular. As length is a property of the vector $d\mathbf{x}$ and not the coordinate system, it follows that the magnitude ds^2 should be a scalar. Using this requirement, it is not difficult to show that the g_{ij} must be a tensor of order $0+2$.

The metric tensor is a tensor field of order $0+2$, which assigns to every point a tensor g_{ij} (which is implicitly a function of position) as above, thereby defining the concept of length throughout the space, and allowing one to measure the length of curves in a space. For example, if $x^i = f^i(u)$, $\acute{u} \leq u \leq \grave{u}$, defines a curve in space then the length of that curve $l(f^i)$ will be:

$$l\left(f^i\right) = \int_{u=\acute{u}}^{\grave{u}} \sqrt{\left| g_{ij} \frac{df^i}{du} \frac{df^j}{du} \right|} du$$

By analogy with the definition of the metric tensor, the conjugate tensor g^{ij} may be defined to give the magnitude of a covariant vector a_i at a point x^i via $\|a_i\|^2 = g^{ij} a_i a_j$. For example, the N-dimensional Laplacian is defined by:

$$\nabla^2 = g^{ij} \partial_i \partial_j$$

Naturally, for an orthonormal Cartesian basis at a point x^i one would expect that $g^{ij} = \gamma^{ij}$ at that point. Using an analogous argument to the construction of the metric tensor, it follows logically that g^{ij} should be a symmetric tensor of order $2+0$.

Consider two tensors a_i and b^i at a point x^i. Then $f = a_i b^i$ will be a scalar at x^i. Choosing an orthonormal basis for T_{x^i}, it follows that:

$$f = a_i b^i = c^i d_i$$

where:

$$\begin{aligned} c^i &= \gamma^{ij} a_j \\ &= g^{ij} a_j \qquad (B.17) \\ d_i &= \gamma_{ij} b^j \\ &= g_{ij} b^j \qquad (B.18) \end{aligned}$$

are, by definition, tensors. For arbitrary coordinates:

$$f = c^i d_i = a_i g^{ij} g_{jk} b^k$$

but by definition $f = a_i b^i$, so it may be concluded that:

$$g_{ij} g^{jk} = \gamma_{ij} \gamma^{jk} = \delta_i{}^k$$

or, in matrix notation:

$$[g_{ij}] \left[g^{jk} \right] = \mathbf{I}$$

and therefore:

$$\left[g^{ij} \right] = [g_{ij}]^{-1}$$

The tensor g^{ij} is the contravariant tensor, and g_{ij} is its covariant conjugate.

Consider the change in g_{ij} under change of coordinates:

$$\bar{g}_{\bar{i}\bar{j}} = \left(\bar{\partial}_{\bar{i}} x^i \right) g_{ij} \left(\bar{\partial}_{\bar{j}} x^j \right)$$

Defining $g = \det([g_{ij}])$, it follows that:

$$\bar{g} = \bar{J}^{-2} g$$

which implies that g is a relative scalar of weight -2 (although for brevity the underscript -2 is not included). Furthermore, in Cartesian coordinates, $g = \gamma_{11} \gamma_{22} \ldots \gamma_{NN} = \pm 1$, and hence $\operatorname{sgn}(g) = \gamma_{11} \gamma_{22} \ldots \gamma_{NN}$. Being careful to ensure that the relevant square-roots are defined (both argument and result are positive), it follows that:

$$\sqrt{\operatorname{sgn}(\bar{g})\,\bar{g}} = \operatorname{sgn}\left(\bar{J}\right)\bar{J}^{-1}\sqrt{\operatorname{sgn}(g)\,g} \tag{B.19}$$

so $\sqrt{\operatorname{sgn}(g)\,g}$ is an oriented relative tensor of weight -1.

Another useful result (which will not be proven here) is:

$$\frac{\partial g}{\partial g_{ij}} = g g^{ij}$$

or:

$$\frac{\partial \ln(\operatorname{sgn}(g)\,g)}{\partial g_{ij}} = g^{ij}$$

and:

$$\frac{\partial \ln(\operatorname{sgn}(g)\,g)}{\partial x^k} = \frac{\partial \ln(\operatorname{sgn}(g)\,g)}{\partial g_{ij}}\frac{\partial g_{ij}}{\partial x^k} = g^{ij}\frac{\partial g_{ij}}{\partial x^k} \tag{B.20}$$

B.2.3 Raising and Lowering Indices

When defining general relative tensors (and subsequently tensor densities and tensors) in definition B.1.10, I noted that, in general, superscripts should not be placed directly over subscripts and vice-versa, giving a definite order to the indices. I will now show why this was done.

Consider the definitions of covariant and contravariant vectors. Clearly, they are expressing much that same thing (namely, the concept of directed magnitude), but in different ways (namely, a movement of something in a given direction (contravariant vectors) versus the rate of change of something (covariant vectors)).

The expression $g_{ij}a^i a^j$ holds the clue to this connection. Suppose this is rewritten $a_i a^i$, where a_i has been defined by $a_i = g_{ij}a^j$. Clearly, a_i is a covariant vector, defined by the contravariant vector a^i and a characteristic of space, namely the metric tensor g_{ij}. So, in a very real sense, a_i and a^i are the same thing. Formally, the operation of moving from a^i to a_i using $a_i = g_{ij}a^j$ is called *lowering* the index i.

Likewise, given a covariant vector b_i, it is possible to *raise* the index to get a contravariant vector $b^i = g^{ij}b_j$, and as before, $g^{ij}b_i b_j = b^i b_i$. Note the raising and then lowering an index will not change the tensor. Hence:

$$\begin{aligned} a^i &= g^{ij}a_j = g^{ij}g_{jk}a^k = \delta^i{}_k a^k \\ b_i &= g_{ij}b^j = g_{ij}g^{jk}b_k = \delta_i{}^k b_k \end{aligned}$$

More generally, any index of a relative tensor (or an oriented relative tensor) may be unambiguously raised or lowered in this way. However, some care is required when dealing with "special" objects. For example, $\delta^i{}_j$ may be treated like a tensor (which it is), but δ^{ij} may not (and neither may γ^{ij}). In this case, raising j in $\delta^i{}_j$ will give the conjugate metric tensor, g^{ij}, and lowering i will give the metric tensor g_{ij}. Likewise, raising and lowering indices of $\delta_{j_1 j_2 \ldots j_n}{}^{i_1 i_2 \ldots i_n}$, $\epsilon^{i_1 i_2 \ldots i_N}$ and $\epsilon_{j_1 j_2 \ldots j_N}$ is to be avoided.

B.2.4 Duality and Volume Revisited

In section B.1.8 it was noted (see equation (B.23)) that the dual $\overset{d}{d\tau}_{(N)}{}^{i_1 j_2 \ldots j_N}$ of an N-cell is closely related to the concept of volume. In this section an alternative definition of duality will be given that makes this connection exact.

First, define:

$$\eta_{i_1 i_2 \ldots i_N} = \sqrt{\operatorname{sgn}(g)\, g}\, \epsilon_{i_1 i_2 \ldots i_N} \tag{B.21}$$
$$\eta^{i_1 i_2 \ldots i_N} = \frac{1}{\sqrt{\operatorname{sgn}(g)\, g}} \epsilon^{i_1 i_2 \ldots i_N} \tag{B.22}$$

noting that $\eta_{i_1 i_2 \ldots i_N}$ is an oriented tensor of order $0+N$ and $\eta^{i_1 i_2 \ldots i_N}$ an oriented tensor of order $N+0$. These tensors may be used in an alternative (more practical) definition of duality, namely:

- The *dual* $\underset{w+}{f}{}_{j_1 j_2 \ldots j_{N-p}} = \underset{w}{\overset{D}{f}}{}^{i_1 i_2 \ldots i_p}$ of a contravariant p-form $\underset{w}{f}{}^{i_1 i_2 \ldots i_p}$ of weight w is defined to be:
$$\underset{w}{\overset{D}{f}}{}^{i_1 i_2 \ldots i_p} = \frac{1}{p!} \eta_{j_1 j_2 \ldots j_{N-p} i_1 i_2 \ldots i_p} \underset{w}{f}{}^{i_1 i_2 \ldots i_p}$$
and is an oriented covariant $(N-p)$-form of weight w.

- The *dual* $\underset{w+}{f}{}^{j_1 j_2 \ldots j_{N-p}} = \underset{w}{\overset{D}{f}}{}_{i_1 i_2 \ldots i_p}$ of a covariant p-form $\underset{w}{f}{}_{i_1 i_2 \ldots i_p}$ of weight w is defined to be:
$$\underset{w}{\overset{D}{f}}{}_{j_1 j_2 \ldots j_p} = \frac{1}{p!} \eta^{j_1 j_2 \ldots j_{N-p} i_1 i_2 \ldots i_p} \underset{w}{f}{}_{i_1 i_2 \ldots i_p}$$
and is an oriented contravariant $(N-p)$-form of weight w.

- The *dual* $\underset{w}{f}{}_{j_1 j_2 \ldots j_{N-p}} = \underset{w+}{\overset{D}{f}}{}^{i_1 i_2 \ldots i_p}$ of an oriented contravariant p-form $\underset{w+}{f}{}^{i_1 i_2 \ldots i_p}$ of weight w is defined to be:
$$\underset{w+}{\overset{D}{f}}{}^{i_1 i_2 \ldots i_p} = \frac{1}{p!} \eta_{j_1 j_2 \ldots j_{N-p} i_1 i_2 \ldots i_p} \underset{w+}{f}{}^{i_1 i_2 \ldots i_p}$$
and is a covariant $(N-p)$-form of weight w.

- The *dual* $\underset{w}{f}{}^{j_1 j_2 \ldots j_{N-p}} = \underset{w+}{\overset{D}{f}}{}_{i_1 i_2 \ldots i_p}$ of an oriented covariant p-form $\underset{w+}{f}{}_{i_1 i_2 \ldots i_p}$ of weight w is defined to be:
$$\underset{w+}{\overset{D}{f}}{}_{j_1 j_2 \ldots j_p} = \frac{1}{p!} \eta^{j_1 j_2 \ldots j_{N-p} i_1 i_2 \ldots i_p} \underset{w+}{f}{}_{i_1 i_2 \ldots i_p}$$
and is a contravariant $(N-p)$-form of weight w.

Once again, it can be shown that the duality operation is its own inverse. That is:

$$\begin{aligned}
\overset{DD}{\underset{w}{f}}{}_{i_1 i_2 \ldots i_p} &= \underset{w}{f}{}_{i_1 i_2 \ldots i_p} \\
\overset{DD}{\underset{w+}{f}}{}_{i_1 i_2 \ldots i_p} &= \underset{w+}{f}{}_{i_1 i_2 \ldots i_p} \\
\overset{DD}{\underset{w}{f}}{}^{i_1 i_2 \ldots i_p} &= \underset{w}{f}{}^{i_1 i_2 \ldots i_p} \\
\overset{DD}{\underset{w+}{f}}{}^{i_1 i_2 \ldots i_p} &= \underset{w+}{f}{}^{i_1 i_2 \ldots i_p}
\end{aligned}$$

Applying this definition of duality to the N-cell $d\tau_{(N)}{}^{i_1 i_2 \ldots i_N}$, it is clear that:

$$\overset{D}{d\tau}_{(N)}{}^{i_1 i_2 \ldots i_N} = \eta_{i_1 i_2 \ldots i_N} dx_1^{i_1} dx_2^{i_2} \ldots dx_N^{i_N}$$

is an oriented scalar. Furthermore, if $dx_I{}^j = \delta_I{}^j dy$ and the coordinates are cartesian (i.e. $g_{ij} = \gamma_{ij}$) then:

$$\overset{D}{d\tau}_{(N)}{}^{i_1 i_2 \ldots i_N} = dy^N \tag{B.23}$$

from which one may conclude that $\overset{D}{d\tau}_{(N)}{}^{i_1 i_2 \ldots i_N}$ is an appropriate measure of the volume of the N-cell $d\tau_{(N)}{}^{i_1 i_2 \ldots i_N}$, namely an oriented scalar which, for cartesian coordinates, corresponds to the standard infinitesimal volume measure.

Moving from Cartesian coordinates to arbitrary coordinates $\bar{x}^{\bar{i}}$ then using (B.19) it is possible to derive the standard formula:

$$\begin{aligned}
\overset{D}{d\tau}_{(N)}{}^{i_1 i_2 \ldots i_N} &= dy^N \\
&= \sqrt{\operatorname{sgn}(\bar{g})\, \bar{g}}\, d\bar{x}^1 d\bar{x}^2 \ldots d\bar{x}^N \\
&= \left|\bar{J}^{-1}\right| d\bar{x}^1 d\bar{x}^2 \ldots d\bar{x}^N
\end{aligned}$$

So in general the infinitesimal volume element is:

$$dV = \sqrt{\operatorname{sgn}(g)\, g}\, dx^1 dx^2 \ldots dx^N$$

B.2.5 Of Geodesics and Christoffel Symbols

Until the present I have only considered the local properties of a space. In the present section, I will be considering a non-local property of space for the first time, namely the concept of *shortest paths.* Before proceeding however it is important to correct a common misconception. It is not uncommon for books, when discussing geodesics, to confuse the quite distinct concepts of *shortest paths* and *straightest paths.* This confusion of concepts is not problematic if space is assumed to be torsionless (torsion will be defined in section B.3), and consequently difficulties do not typically arise in books which assume (sometimes explicitly, often implicitly) that torsion is zero. However, in general it is important to be clear from the start.

Given two non-coincident points x^i and y^i:

- A *geodesic path* between x^i and y^i is the curve of extreme (either shortest, longest or inflected) length between the two points.
- An *autoparallel path* between x^i and y^i is a straight path between the two points, as will be defined in section B.3.
- The geodesic and autoparallel paths between x^i and y^i are not, in general, the same.

Consider two points y^i and z^i. A path from y^i to z^i is a curve $x^i = f^i(u)$, $\acute{u} \leq u \leq \grave{u}$, such that $y^i = f^i(\acute{u})$ and $z^i = f^i(\grave{u})$. The length $l\left(f^k\right)$ is defined to be:

$$l\left(f^k\right) = \int_{u=\acute{u}}^{\grave{u}} \sqrt{\left| g_{ij} \frac{df^i}{du} \frac{df^j}{du} \right|} du$$

Geodesics in Locally Euclidean Spaces

To begin, let us assume that the space is locally Euclidean - that is, the signature of the space is $(+1, +1, \ldots, +1)$. Suppose that the curve $f^i(u)$ is a member of a family of curves $h^i(v^n)(u)$ (assuming continuity in v^n, which is not a tensor) where every choice $v^n \in \mathbb{V}$ defines a curve from y^i to z^i, and the curve so chosen is parametrised by u in the usual manner. The length of a particular curve v^n is defined as:

$$l_h(v^n) = l\left(h^k(v^n)\right) = \int_{u=\acute{u}}^{\grave{u}} \sqrt{\left| g_{ij} \frac{dh^i(v^n)}{du} \frac{dh^j(v^n)}{du} \right|} du$$

If $h^i(v^n)(u)$ is the set of all paths from y^i to z^i then the *geodesic path* is the curve whose length is an extrema of this family. i.e.

$$\frac{\partial l_h(v^n)}{\partial v^q} = 0$$

Defining $p^i = \frac{dh^i(v^n)}{du}$, it follows that:

$$\frac{\partial l_h(v^n)}{\partial v^q} = \int_{u=\acute{u}}^{\grave{u}} \left(\frac{\partial}{\partial v^q} \sqrt{\left| g_{ij} p^i p^j \right|} \right) du = 0$$

Now:

$$\begin{aligned}
\frac{\partial}{\partial v^q} \sqrt{\left| g_{ij} p^i p^j \right|} &= \left(\frac{\partial}{\partial x^k} \sqrt{\left| g_{ij} p^i p^j \right|} \right) \frac{\partial x^k}{\partial v^q} + \left(\frac{\partial}{\partial p^k} \sqrt{\left| g_{ij} p^i p^j \right|} \right) \frac{\partial p^k}{\partial v^q} \\
&= \left(\frac{\partial}{\partial x^k} \sqrt{\left| g_{ij} p^i p^j \right|} \right) \frac{\partial x^k}{\partial v^q} + \left(\frac{\partial}{\partial p^k} \sqrt{\left| g_{ij} p^i p^j \right|} \right) \frac{\partial}{\partial v^q} \frac{dx^k}{du} \\
&= \left(\frac{\partial}{\partial x^k} \sqrt{\left| g_{ij} p^i p^j \right|} \right) \frac{\partial x^k}{\partial v^q} + \left(\frac{\partial}{\partial p^k} \sqrt{\left| g_{ij} p^i p^j \right|} \right) \frac{d}{du} \frac{\partial x^k}{\partial v^q}
\end{aligned}$$

Noting that:

$$\frac{d}{du}\left(\left(\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{\partial x^k}{\partial v^q}\right) = \left(\frac{d}{du}\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{\partial x^k}{\partial v^q}+\left(\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{d}{du}\frac{\partial x^k}{\partial v^q}$$

it follows that:

$$\begin{aligned}\frac{\partial}{\partial v^q}\sqrt{|g_{ij}p^ip^j|} &= \left(\frac{\partial}{\partial x^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{\partial x^k}{\partial v^q}+\frac{d}{du}\left(\left(\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{\partial x^k}{\partial v^q}\right)\\ &- \left(\frac{d}{du}\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{\partial x^k}{\partial v^q}\end{aligned}$$

and hence:

$$\begin{aligned}\frac{\partial l_h\left(v^n\right)}{\partial v^q} &= \left[\left(\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{\partial x^k}{\partial v^q}\right]_{u=\acute{u}}^{\grave{u}}\\ &-\int_{u=\acute{u}}^{\grave{u}}\left(\frac{d}{du}\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}-\frac{\partial}{\partial x^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{\partial x^k}{\partial v^q}du\end{aligned}$$

However, by definition, $\frac{\partial x^i}{\partial v^q}=0$ at $u=\acute{u},\grave{u}$ (as $h^i\left(v^n\right)\left(\acute{u}\right)=y^i$ and $h^i\left(v^n\right)\left(\grave{u}\right)=z^i$ for all $v^n\in\mathbb{V}$). Therefore:

$$\frac{\partial l_h\left(v^n\right)}{\partial v^q} = -\int_{u=\acute{u}}^{\grave{u}}\left(\frac{d}{du}\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}-\frac{\partial}{\partial x^k}\sqrt{|g_{ij}p^ip^j|}\right)\frac{\partial x^k}{\partial v^q}du$$

This gives us the rate of change of the length of the members of the curve family $h^i\left(v^n\right)\left(u\right)$ as the continuous selector $v^n\in\mathbb{V}$ is varied. For a geodesic, $\frac{\partial l_h\left(v^n\right)}{\partial v^q}=0$, so:

$$\frac{d}{du}\frac{\partial}{\partial p^k}\sqrt{|g_{ij}p^ip^j|}-\frac{\partial}{\partial x^k}\sqrt{|g_{ij}p^ip^j|}=0$$

Using the assumption that the signature of the space is $(+1,+1,\ldots,+1)$ (and hence $[g_{ij}]$ must be positive definite everywhere), it is not too difficult to show that this implies:

$$\frac{d}{du}\frac{\partial}{\partial p^k}\left(g_{ij}p^ip^j\right)-\frac{\partial}{\partial x^k}\left(g_{ij}p^ip^j\right)=\frac{1}{2\left(g_{ij}p^ip^j\right)}\left[\frac{d}{du}\left(g_{ij}p^ip^j\right)\right]\frac{\partial}{\partial p^k}\left(g_{ij}p^ip^j\right) \quad \text{(B.24)}$$

Define $s=s\left(u\right)$ be the length of the path (specified by v^q) from $\acute{u}$ to u. Suppose that $s=u$. Then $p^i=\frac{dx^i}{ds}$, and so:

$$\begin{aligned}\frac{d}{ds}\left(g_{ij}p^ip^j\right) &= \frac{d}{ds}\left(g_{ij}\frac{dx^i}{ds}\frac{dx^j}{ds}\right)\\ &= \frac{d}{ds}\left(\frac{g_{ij}dx^idx^j}{ds^2}\right)\\ &= \frac{d}{ds}\left(\frac{ds^2}{ds^2}\right)=0\end{aligned}$$

Hence (B.24) becomes:

$$\frac{d}{ds}\frac{\partial}{\partial p^k}\left(g_{ij}p^ip^j\right)-\frac{\partial}{\partial x^k}\left(g_{ij}p^ip^j\right)=0$$

and so:

$$2\frac{d}{ds}\left(g_{ik}p^i\right)-\frac{\partial g_{ij}}{\partial x^k}p^ip^j=0 \tag{B.25}$$

The first term can be simplified thusly:

$$\begin{aligned}2\frac{d}{ds}\left(g_{ik}p^i\right) &= 2\frac{\partial}{\partial x^l}\left(g_{ik}p^i\right)\frac{dx^l}{ds}+2\frac{\partial}{\partial p^l}\left(g_{ik}p^i\right)\frac{dp^l}{ds}\\ &= 2\frac{\partial g_{ik}}{\partial x^j}p^ip^j+2g_{ik}\frac{dp^i}{ds}\end{aligned}$$

so (B.25) may be re-written:

$$2\frac{\partial g_{ik}}{\partial x^j}p^ip^j+2g_{ik}\frac{dp^i}{ds}-\frac{\partial g_{ij}}{\partial x^k}p^ip^j=0$$

Trivially, this implies the following:

$$\frac{1}{2}g_{ik}\frac{dp^i}{ds}+\frac{1}{2}\frac{\partial g_{ik}}{\partial x^j}p^ip^j-\frac{1}{4}\frac{\partial g_{ij}}{\partial x^k}p^ip^j = 0 \tag{B.26}$$

$$\frac{1}{2}g_{ik}\frac{dp^i}{ds}+\frac{1}{2}\frac{\partial g_{kj}}{\partial x^i}p^ip^j-\frac{1}{4}\frac{\partial g_{ij}}{\partial x^k}p^ip^j = 0 \tag{B.27}$$

Adding (B.26) and (B.26) it follows that a curve $x^i=f^i(s)$ is a geodesic if:

$$g_{ik}\frac{dp^i}{ds}+[ij\ \ k]\,p^ip^j=0$$

where:

$$[ij\ \ k]=\frac{1}{2}\left(\partial_jg_{ik}+\partial_ig_{kj}-\partial_kg_{ij}\right)$$

is called the *Christoffel symbol of the first kind.* It is customary to define a *Christoffel symbol of the second kind* by:

$$\left\{\begin{array}{c}k\\ij\end{array}\right\}=g^{kl}\,[ij\ \ l]$$

Then a curve $x^i=f^i(s)$ is a geodesic if:

$$\frac{d^2x^k}{ds^2}+\left\{\begin{array}{c}k\\ij\end{array}\right\}\frac{dx^i}{ds}\frac{dx^j}{ds}=0 \tag{B.28}$$

everywhere along the curve, where s is the length of the curve measured from some arbitrary point on the curve (noting that an arbitrary offset for s will not change (B.28)). This is known as the *geodesic equation.*

It should be noted that as (B.28) is a second order differential equation, to determine a geodesic everywhere it is sufficient to have a point which the geodesic will starting point (or intercept), say y^i, and the gradient at that point (say $\frac{dy^i}{ds}$) - that is, a starting point (or intercept) and a direction.

Now, in general, $u = u(s)$, where s is the length of the path (specified by v^n) from $\acute{u}$ to u. Then:

$$\begin{aligned}
&\frac{d^2x^k}{ds^2} + \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \frac{dx^i}{ds}\frac{dx^j}{ds} = 0 \\
\therefore\ &\frac{d}{du}\left(\frac{dx^k}{du}\frac{du}{ds}\right)\frac{du}{ds} + \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \frac{dx^i}{du}\frac{du}{ds}\frac{dx^j}{du}\frac{du}{ds} = 0 \\
\therefore\ &\frac{d^2x^k}{du^2}\left(\frac{du}{ds}\right)^2 + \frac{dx^i}{du}\left(\frac{d}{du}\frac{du}{ds}\right)\frac{du}{ds} + \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \frac{dx^i}{du}\frac{dx^j}{du}\left(\frac{du}{ds}\right)^2 = 0 \\
\therefore\ &\frac{d^2x^k}{du^2}\left(\frac{du}{ds}\right)^2 + \frac{dx^i}{du}\frac{d^2u}{ds^2} + \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \frac{dx^i}{du}\frac{dx^j}{du}\left(\frac{du}{ds}\right)^2 = 0
\end{aligned}$$

which gives an alternate form of the geodesic equation:

$$\frac{d^2x^k}{du^2} + \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \frac{dx^i}{du}\frac{dx^j}{du} = \lambda\frac{dx^k}{du} \tag{B.29}$$

where:

$$\lambda = -\frac{\frac{d^2u}{ds^2}}{\left(\frac{du}{ds}\right)^2}$$

Now, note that in (B.29) the left-hand side is a tensor (specifically, contravariant vector), as is $\frac{dx^i}{du}$ on the right hand side. Given that u can be *any* function of the distance s from y^i, λ can be made to be any scalar. So, in general, a curve with an arbitrary parameter u is a geodesic if it satisfies:

$$\frac{d^2x^i}{du^2} + \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \frac{dx^i}{du}\frac{dx^j}{du} \propto \frac{dx^i}{du} \tag{B.30}$$

Geodesics in Generic Spaces

While the construction of the geodesic given above makes the assumption that the space is locally Euclidean, it will be noted that throughout the argument the only assumption that is strictly necessary is that the curves are not null. That is, at no point does the tangent dx^i to the curve satisfy the equality $ds^2 = g_{ij}dx^idx^j = 0$.

For the case $ds = 0$, a *geodesic null line* is defined to be a curve satisfying (B.29) where $\lambda = 0$.

B.2.6 Properties of the Christoffel Symbols

The Christoffel symbols of the first and second kind, respectively, were defined previously as:

$$[ij\ \ k] = \frac{1}{2}\left(\partial_j g_{ik} + \partial_i g_{kj} - \partial_k g_{ij}\right) \tag{B.31}$$

$$\left\{ \begin{matrix} k \\ ij \end{matrix} \right\} = g^{kl}\,[ij\ \ l] \tag{B.32}$$

Under change of basis, it can be shown that:

$$\overline{[\bar{i}\bar{j}\ \ \bar{k}]} = \left(\bar{\partial}_{\bar{i}} x^i\right)\left(\bar{\partial}_{\bar{j}} x^j\right)\left(\bar{\partial}_{\bar{k}} x^k\right)[ij\,,\ k] + g_{kj}\left(\bar{\partial}_{\bar{k}} x^k\right)\left(\bar{\partial}_{\bar{i}}\bar{\partial}_{\bar{j}} x^j\right) \tag{B.33}$$

$$\overline{\left\{ \begin{matrix} \bar{k} \\ \bar{i}\bar{j} \end{matrix} \right\}} = \left(\bar{\partial}_{\bar{i}} x^i\right)\left(\bar{\partial}_{\bar{j}} x^j\right)\left(\partial_k \bar{x}^{\bar{k}}\right)\left\{ \begin{matrix} k \\ ij \end{matrix} \right\} + \left(\partial_k \bar{x}^{\bar{k}}\right)\left(\bar{\partial}_{\bar{i}}\bar{\partial}_{\bar{j}} x^k\right) \tag{B.34}$$

from which it is clear that neither symbol is a tensor.

For later reference, the following properties are presented without proof:

$$[ij\ \ k] = \frac{1}{2}\left(\partial_j g_{ik} + \partial_i g_{kj} - \partial_k g_{ij}\right) \tag{B.35}$$

$$[[ij]\ \ k] = 0 \tag{B.36}$$

$$[i[j\ \ k]] = \frac{1}{2}\left(\partial_j g_{ik} - \partial_k g_{ij}\right) \tag{B.37}$$

$$\left[\begin{matrix} [\bullet & \bullet] \\ ij & k \end{matrix} \right] = \frac{1}{2}\left(\partial_i g_{kj} - \partial_k g_{ij}\right) \tag{B.38}$$

$$[\{ij\}\ \ k] = [ij\ \ k] \tag{B.39}$$

$$[i\{j\ \ k\}] = \frac{1}{2}\partial_i g_{jk} \tag{B.40}$$

$$\left[\begin{matrix} \{\bullet & \bullet\} \\ ij & k \end{matrix} \right] = \frac{1}{2}\partial_j g_{ik} \tag{B.41}$$

$$\left[\underbrace{ij\ \ k}\right] = 0 \tag{B.42}$$

$$[\underline{ij\ \ k}] = \frac{1}{3!}\left(\partial_k g_{ij} + \partial_i g_{jk} + \partial_j g_{ki}\right) \tag{B.43}$$

B.3 The Affine Connection

B.3.1 Non-tensorial Tensor Gradients

Consider a contravariant tensor field, A^i. At points x^i and $x^i + \delta x^i$, define the value of the field to be A^i and $A^i + dA^i$, respectively. Then:

$$dA^i = \left(\partial_j A^i\right) dx^j$$

Naively, it would seem reasonable to expect that dA^i should be a contravariant tensor at x^i, namely the difference between the value of the field at x^i and $x^i + \delta x^i$, as measured at x^i. However, this is not in general true. To see this, consider what happens under a change of coordinates:

$$\begin{aligned}
d\bar{A}^{\bar{i}} &= \left(\bar{\partial}_{\bar{j}}\bar{A}^{\bar{i}}\right)\delta\bar{x}^{\bar{j}} \\
&= \left(\left(\bar{\partial}_{\bar{j}}x^j\right)\partial_j\left(\left(\partial_i\bar{x}^{\bar{i}}\right)A^i\right)\right)\left(\partial_k\bar{x}^{\bar{j}}\right)\delta x^k \\
&= \left(\left(\bar{\partial}_{\bar{j}}x^j\right)\left(\partial_i\bar{x}^{\bar{i}}\right)\left(\partial_j A^i\right)+\left(\bar{\partial}_{\bar{j}}x^j\right)\left(\partial_j\partial_i\bar{x}^{\bar{i}}\right)A^i\right)\left(\partial_k\bar{x}^{\bar{j}}\right)\delta x^k \\
&= \left(\partial_i\bar{x}^{\bar{i}}\right)dA^i+\left(\partial_i\partial_j\bar{x}^{\bar{i}}\right)A^i\delta x^j
\end{aligned} \tag{B.44}$$

The presence of the second term, $\left(\partial_i\partial_j\bar{x}^{\bar{i}}\right)A^i\delta x^j$, in the above expression clearly demonstrates that, in general, the difference between the values of a contravariant tensor field at two infinitesimally separated points x^i and $x^i+\delta x^i$ is not a tensor.

Indeed, the gradient of a generic relative tensor field $\underset{w}{A}{}^{i_1i_2\ldots i_n}_{j_1j_2\ldots j_m}$ is in general non-tensorial. It is straightforward (although admittedly rather tedious) to show that, under change of coordinates:[5]

$$\begin{aligned}
&\bar{\partial}_{\bar{k}}\underset{w}{\bar{A}}{}^{\bar{i}_1\bar{i}_2\ldots\bar{i}_n}_{\bar{j}_1\bar{j}_2\ldots\bar{j}_m} = \ldots \\
&\quad \bar{J}^w\left(\partial_{i_1}\bar{x}^{\bar{i}_1}\right)\left(\partial_{i_2}\bar{x}^{\bar{i}_2}\right)\ldots\left(\partial_{i_n}\bar{x}^{\bar{i}_n}\right)\left(\bar{\partial}_{\bar{j}_1}x^{j_1}\right)\left(\bar{\partial}_{\bar{j}_2}x^{j_2}\right)\ldots\left(\bar{\partial}_{\bar{j}_m}x^{j_m}\right)\left(\bar{\partial}_{\bar{k}}x^k\right)\partial_k\underset{w}{A}{}^{i_1i_2\ldots i_n}_{j_1j_2\ldots j_m} \\
&- w\bar{J}^w\bar{\Upsilon}^{\bar{q}}{}_{\bar{q}\bar{k}}\left(\partial_{i_1}\bar{x}^{\bar{i}_1}\right)\left(\partial_{i_2}\bar{x}^{\bar{i}_2}\right)\ldots\left(\partial_{i_n}\bar{x}^{\bar{i}_n}\right)\left(\bar{\partial}_{\bar{j}_1}x^{j_1}\right)\left(\bar{\partial}_{\bar{j}_2}x^{j_2}\right)\ldots\left(\bar{\partial}_{\bar{j}_m}x^{j_m}\right)\underset{w}{A}{}^{i_1i_2\ldots i_n}_{j_1j_2\ldots j_m} \\
&+ \Big(\left(\bar{\partial}_{\bar{k}}x^k\right)\left(\partial_k\partial_{i_1}\bar{x}^{\bar{i}_1}\right)\left(\partial_{i_2}\bar{x}^{\bar{i}_2}\right)\ldots\left(\partial_{i_n}\bar{x}^{\bar{i}_n}\right)\left(\bar{\partial}_{\bar{j}_1}x^{j_1}\right)\left(\bar{\partial}_{\bar{j}_2}x^{j_2}\right)\ldots\left(\bar{\partial}_{\bar{j}_m}x^{j_m}\right) \\
&\quad +\left(\bar{\partial}_{\bar{k}}x^k\right)\left(\partial_{i_1}\bar{x}^{\bar{i}_1}\right)\left(\partial_k\partial_{i_2}\bar{x}^{\bar{i}_2}\right)\ldots\left(\partial_{i_n}\bar{x}^{\bar{i}_n}\right)\left(\bar{\partial}_{\bar{j}_1}x^{j_1}\right)\left(\bar{\partial}_{\bar{j}_2}x^{j_2}\right)\ldots\left(\bar{\partial}_{\bar{j}_m}x^{j_m}\right) \\
&\quad +\ldots \\
&\quad +\left(\bar{\partial}_{\bar{k}}x^k\right)\left(\partial_{i_1}\bar{x}^{\bar{i}_1}\right)\left(\partial_{i_2}\bar{x}^{\bar{i}_2}\right)\ldots\left(\partial_k\partial_{i_n}\bar{x}^{\bar{i}_n}\right)\left(\bar{\partial}_{\bar{j}_1}x^{j_1}\right)\left(\bar{\partial}_{\bar{j}_2}x^{j_2}\right)\ldots\left(\bar{\partial}_{\bar{j}_m}x^{j_m}\right) \\
&\quad +\left(\partial_{i_1}\bar{x}^{\bar{i}_1}\right)\left(\partial_{i_2}\bar{x}^{\bar{i}_2}\right)\ldots\left(\partial_{i_n}\bar{x}^{\bar{i}_n}\right)\left(\bar{\partial}_{\bar{k}}\bar{\partial}_{\bar{j}_1}x^{j_1}\right)\left(\bar{\partial}_{\bar{j}_2}x^{j_2}\right)\ldots\left(\bar{\partial}_{\bar{j}_m}x^{j_m}\right) \\
&\quad +\left(\partial_{i_1}\bar{x}^{\bar{i}_1}\right)\left(\partial_{i_2}\bar{x}^{\bar{i}_2}\right)\ldots\left(\partial_{i_n}\bar{x}^{\bar{i}_n}\right)\left(\bar{\partial}_{\bar{j}_1}x^{j_1}\right)\left(\bar{\partial}_{\bar{k}}\bar{\partial}_{\bar{j}_2}x^{j_2}\right)\ldots\left(\bar{\partial}_{\bar{j}_m}x^{j_m}\right) \\
&\quad +\ldots \\
&\quad +\left(\partial_{i_1}\bar{x}^{\bar{i}_1}\right)\left(\partial_{i_2}\bar{x}^{\bar{i}_2}\right)\ldots\left(\partial_{i_n}\bar{x}^{\bar{i}_n}\right)\left(\bar{\partial}_{\bar{j}_1}x^{j_1}\right)\left(\bar{\partial}_{\bar{j}_2}x^{j_2}\right)\ldots\left(\bar{\partial}_{\bar{k}}\bar{\partial}_{\bar{j}_m}x^{j_m}\right)\Big)\bar{J}^w\underset{w}{A}{}^{i_1i_2\ldots i_n}_{j_1j_2\ldots j_m}
\end{aligned}$$

where for convenience I have defined:

$$\begin{aligned}
\bar{\Upsilon}^{\bar{k}}{}_{\bar{i}\bar{j}} &= \left(\partial_k\bar{x}^{\bar{k}}\right)\left(\bar{\partial}_{\bar{i}}\bar{\partial}_{\bar{j}}x^k\right) \\
&= -\left(\bar{\partial}_{\bar{i}}x^i\right)\left(\bar{\partial}_{\bar{j}}x^j\right)\left(\partial_i\partial_j\bar{x}^{\bar{k}}\right)
\end{aligned}$$

[5]To derive this, you will need the result:

$$\partial_k\left(\ln\left|\det\left[\partial_q\bar{x}^{\bar{q}}\right]\right|\right)=\left(\bar{\partial}_{\bar{q}}x^q\right)\left(\partial_q\partial_k\bar{x}^{\bar{q}}\right)$$

which is be obtained by considering the equation:

$$\partial_k \delta^i{}_j = \partial_k \left(\left(\partial_j \bar{x}^{\bar{k}} \right) \left(\bar{\partial}_{\bar{k}} x^i \right) \right) = 0$$

This is all rather inconvenient, and seems to preclude the possibility of comparing any objects not positioned at exactly the same point. The problem with the above is not serious, however - it is simply that indicative of a failure to account for the fundamentally non-local character of such a comparison. Once the non-local character is taken into account, the problem evaporates.

B.3.2 Parallel Transport and Covariant Differentiation

An approach to the problem of defining the actual difference between (relative) tensors not defined at the same position is to use *parallel transport*, defining how the numerical representation of such an object with respect to a given basis changes as it is moved from position to position.

Consider a relative tensor $\underset{w}{A}{}^{\cdots}_{\cdots}$ of weight w and order $n+m$ defined at x^i, and another infinitesimal different relative tensor $\underset{w}{A}{}^{\cdots}_{\cdots} + d\underset{w}{A}{}^{\cdots}_{\cdots}$, also of weight w and order $n+m$, defined at the infinitesimally close point $x^i + \delta x^i$. To obtain a relative tensor representing the physical difference between the two relative tensors, we first parallel transport $\underset{w}{A}{}^{\cdots}_{\cdots}$ to $x^i + \delta x^i$. i.e.:

1. Start with a relative tensor $\underset{w}{A}{}^{\cdots}_{\cdots}$ at x^i.

2. Move $\underset{w}{A}{}^{\cdots}_{\cdots}$ parallelly by the infinitesimal displacement δx^i from x^i to $x^i + \delta x^i$.

3. This leaves the relative tensor physically unchanged, but results in a numerical change $\delta\underset{w}{A}{}^{\cdots}_{\cdots}$.

4. We now have a relative tensor $\underset{w}{A}{}^{\cdots}_{\cdots} + \delta\underset{w}{A}{}^{\cdots}_{\cdots}$ at $x^i + \delta x^i$ which is physically identical (by definition) to the original relative tensor $\underset{w}{A}{}^{\cdots}_{\cdots}$ at x^i.

The physical difference between the two infinitesimally separated relative tensors is defined to be the numerical difference between $\underset{w}{A}{}^{\cdots}_{\cdots} + \delta\underset{w}{A}{}^{\cdots}_{\cdots}$ and $\underset{w}{A}{}^{\cdots}_{\cdots} + d\underset{w}{A}{}^{\cdots}_{\cdots}$ at $x^i + \delta x^i$, namely:

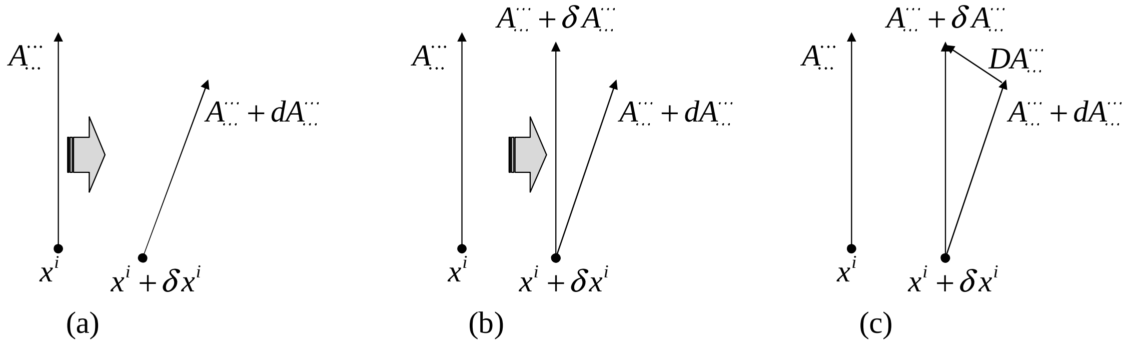
$$D\underset{w}{A}{}^{\cdots}_{\cdots} = d\underset{w}{A}{}^{\cdots}_{\cdots} - \delta\underset{w}{A}{}^{\cdots}_{\cdots}$$

and is a relative tensor of the same type at $x^i + \delta x^i$. This is shown schematically in figure B.4

Let us define the parallel transport process using the smooth function:

$$\delta\underset{w}{A}{}^{\cdots}_{\cdots} = f\left(x^i\right)\left(\underset{w}{A}{}^{\cdots}_{\cdots}, \delta x^i\right)$$

where both $\underset{w}{A}{}^{\cdots}_{\cdots}$ and $\delta\underset{w}{A}{}^{\cdots}_{\cdots}$ are relative tensors of weight w and order $n+m$.



$A^{\cdots}_{\cdots}$
$A^{\cdots}_{\cdots}+dA^{\cdots}_{\cdots}$
x^i
$x^i+\delta x^i$
(a)
$A^{\cdots}_{\cdots}+\delta A^{\cdots}_{\cdots}$
$A^{\cdots}_{\cdots}$
$A^{\cdots}_{\cdots}+dA^{\cdots}_{\cdots}$
x^i
$x^i+\delta x^i$
(b)
$A^{\cdots}_{\cdots}+\delta A^{\cdots}_{\cdots}$
$A^{\cdots}_{\cdots}$
$DA^{\cdots}_{\cdots}$
$A^{\cdots}_{\cdots}+dA^{\cdots}_{\cdots}$
x^i
$x^i+\delta x^i$
(c)

Figure B.4: Schematic illustration of parallel transport. (a) shows the starting situation, with $A^{\cdots}_{\cdots}$ at x^i and $\underset{w}{A}^{\cdots}_{\cdots}+d\underset{w}{A}^{\cdots}_{\cdots}$ at $x^i+\delta x^i$. $\underset{w}{A}^{\cdots}_{\cdots}$ is then parallel transported to $x^i+\delta x^i$ to give $A^{\cdots}_{\cdots}+\delta\underset{w}{A}^{\cdots}_{\cdots}$ at this point. The physical difference $D\underset{w}{A}^{\cdots}_{\cdots}$ between the relative tensors, both located at $x^i+\delta x^i$, is shown in figure (c).

Now, in reality this function will only ever be encountered in path integrals of the form:

$$\int_{u=\acute{u}}^{\grave{u}} f\left(\frac{dh^i}{du}du\right)\left(\underset{w}{A}^{\cdots}_{\cdots},\delta x^i\right)$$

where $h^i\left(u\right)$ defines some curve. This is basically the finite change in $\underset{w}{A}^{\cdots}_{\cdots}$ due to parallel transport along the curve $h^i\left(u\right)$. To ensure a finite answer, it must be true that:

$$f\left(x^i\right)\left(\underset{w}{A}^{\cdots}_{\cdots},\delta x^i\right)=o\left(\delta x^i\right)$$

Furthermore, as none of the higher order components (that is, $o\left(\left(\delta x^i\right)^2\right)$) will have any effect on the integral, *only* the behaviour of f to first order is of interest.

It is also reasonable to expect that parallel transport would satisfy the following conditions (technically, this defines the affine connection):

- If $\underset{w}{A}^{\cdots}_{\cdots}=\beta\underset{w}{B}^{\cdots}_{\cdots}+\gamma\underset{w}{C}^{\cdots}_{\cdots}$ then:

$$f\left(x^i\right)\left(\underset{w}{A}^{\cdots}_{\cdots},\delta x^i\right)=\beta f\left(x^i\right)\left(\underset{w}{B}^{\cdots}_{\cdots},\delta x^i\right)+\gamma f\left(x^i\right)\left(\underset{w}{C}^{\cdots}_{\cdots},\delta x^i\right)+o\left(\left(\delta x^i\right)^2\right)$$

1

- If $\underset{w}{A}^{\cdots}_{\cdots}=\underset{w}{B}^{\cdots}_{\cdots}\underset{w}{C}^{\cdots}_{\cdots}$ then:

$$f\left(x^i\right)\left(\underset{w}{A}^{\cdots}_{\cdots},\delta x^i\right)=f\left(x^i\right)\left(\underset{w}{B}^{\cdots}_{\cdots},\delta x^i\right)f\left(x^i\right)\left(\underset{w}{C}^{\cdots}_{\cdots},\delta x^i\right)+o\left(\left(\delta x^i\right)^2\right)$$

- If $\underset{w}{A}^{\cdots}_{\cdots}=\underset{w}{B}^{\cdots i\cdots}_{\cdots i\cdots}$ then:

$$f\left(x^i\right)\left(\underset{w}{A}^{\cdots}_{\cdots},\delta x^i\right)=\delta_i{}^j f\left(x^i\right)\left(\underset{w}{B}^{\cdots i\cdots}_{\cdots j\cdots},\delta x^i\right)+o\left(\left(\delta x^i\right)^2\right)$$

- If $\delta x^i = \beta \delta y^i + \gamma \delta z^i$ then:

$$f\left(x^i\right)\left(\underset{w}{A}{}^{\cdots}_{\cdots}, \delta x^i\right) = \beta f\left(x^i\right)\left(\underset{w}{A}{}^{\cdots}_{\cdots}, \delta y^i\right) + \gamma f\left(x^i\right)\left(\underset{w}{A}{}^{\cdots}_{\cdots}, \delta z^i\right) + o\left(\left(\delta x^i\right)^2\right)$$

- If $\underset{w}{A}$ is a relative scalar:

$$f\left(x^i\right)\left(\underset{w}{A}, \delta x^i\right) = o\left(\left(\delta x^i\right)^2\right)$$

Ignoring terms of order $o\left(\left(\delta x^i\right)^2\right)$, the following special cases turn out to be particularly simple:

- If B^i is a contravariant vector then:

$$\delta B^k = \Omega_{ij}{}^k B^i \delta x^j$$

  where $\Omega_{ij}{}^k$ is implicitly a function of position.

- If C_i is a covariant vector then:

$$\delta C_k = \Xi_{kj}{}^i C_i \delta x^j$$

  where $\Xi_{ij}{}^k$ is implicitly a function of position.

Now, it was shown previously (B.4) that any relative tensor $\underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ of weight w and order $n+m$ may be written thusly:

$$\underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \underset{w}{B} \sum_{k=1}^{N^{n+m}} c_{(k,1)}{}^{i_1} c_{(k,2)}{}^{i_2} \ldots c_{(k,n)}{}^{i_n} d_{(k,1) j_1} d_{(k,2) j_2} \ldots d_{(k,m) j_m}$$

where $\underset{w}{B}$ is any (non-zero) relative scalar of weight w, $c_{(k,l)}{}^{i_l}$ (where $1 \leq k \leq N^{n+m}$, $1 \leq l \leq n$) is a contravariant vector and $d_{(k,l) i_l}$ (where $1 \leq k \leq N^{n+m}$, $1 \leq l \leq m$) is a covariant vector. Given this, it follows immediately that, neglecting all terms of order $o\left(\left(\delta x^i\right)^2\right)$:

$$\delta \underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \left(\sum_{p=1}^{n} \Omega_{kl}{}^{i_p} \underset{w}{A}{}^{i_1 i_2 \ldots i_{p-1} k i_{p+1} \ldots i_n}_{j_1 j_2 \ldots j_m} + \sum_{q=1}^{m} \Xi_{j_q l}{}^{k} \underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_{q-1} k j_{q+1} \ldots j_m} \right) \delta x^l$$

So the effect of parallel transportation on *any* relative tensor is entirely specified by the arrays $\Omega_{ij}{}^k$ and $\Xi_{ij}{}^k$, which were originally defined with the parallel transport of contravariant and covariant vectors in mind contact me and I'll shout you a beer, limited offer. This is particularly useful, as it means that one only need deal with such vectors (and scalars) in order to divine the properties of parallel transport in general.

Suppose A^i is a contravariant vector and B_i a covariant vector. Then $A^i B_j$ is a tensor, and:

$$\delta\left(A^i B_j\right) = \left({\Omega_{kl}}^i A^k B_j + {\Xi_{jl}}^k A^i B_k\right)\delta x^l \tag{B.45}$$

By contraction, define $C = A^i B_i$, which is a scalar, and hence $\delta C = 0$. Now, from (B.45) it follows that:

$$\begin{aligned}
\delta C &= {\delta_i}^j \delta\left(A^i B_j\right) \\
&= \left({\Omega_{kl}}^i A^k B_i + {\Xi_{il}}^k A^i B_k\right)\delta x^l \\
&= \left({\Omega_{kl}}^i A^k B_i + {\Xi_{kl}}^i A^k B_i\right)\delta x^l \\
&= \left({\Omega_{kl}}^i + {\Xi_{kl}}^i\right) A^k B_i \delta x^l \\
&= 0
\end{aligned}$$

Therefore:

$${\Xi_{ij}}^k = -{\Omega_{ij}}^k$$

This leads us to define the *affine connection* ${\Gamma_{ij}}^k$ thusly:

$${\Gamma_{ij}}^k = {\Xi_{ij}}^k = -{\Omega_{ij}}^k$$

Because only the affine connection will be of interest here, ${\Gamma_{ij}}^k$ will sometimes be referred to as simply the *connection*. Strictly speaking, however, the connection is a more general object, so some care is required when reading elsewhere where the more general object may be the object of interest.

In terms of the affine connection, the parallel transport operation may be written thusly:

$$\delta \underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \left(\sum_{q=1}^{m} {\Gamma_{j_q l}}^k \underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_{q-1} k j_{q+1} \ldots j_m} - \sum_{p=1}^{n} {\Gamma_{kl}}^{i_p} \underset{w}{A}{}^{i_1 i_2 \ldots i_{p-1} k i_{p+1} \ldots i_n}_{j_1 j_2 \ldots j_m}\right)\delta x^l \tag{B.46}$$

This summarises the effect of parallel transport on general relative tensors in terms of a single array field ${\Gamma_{ij}}^k$. The next task is to divine how ${\Gamma_{ij}}^k$ will behave under change of coordinates.

Suppose A^i is a contravariant tensor field with value A^i at x^i and value $A^i + dA^i$ at $x^i + \delta x^i$. Then by definition:

$$DA^i = dA^i - \delta A^i = dA^i + {\Gamma_{kl}}^i A^k \delta x^l \tag{B.47}$$

must be a contravariant tensor. That is, under change of coordinates:

$$D\bar{A}^{\bar{i}} = \left(\partial_i \bar{x}^{\bar{i}}\right) DA^i \tag{B.48}$$

Combining (B.44), (B.47) and (B.48), it follows that:

$$\begin{aligned} D\bar{A}^{\bar{i}} &= \left(\partial_i \bar{x}^{\bar{i}}\right) dA^i + \left(\partial_k \partial_l \bar{x}^{\bar{i}}\right) A^k \delta x^l + \bar{\Gamma}_{\bar{k}\bar{l}}{}^{\bar{i}} \bar{A}^{\bar{k}} \delta \bar{x}^{\bar{l}} \\ &= \left(\partial_i \bar{x}^{\bar{i}}\right) dA^i + \left(\left(\partial_k \bar{x}^{\bar{k}}\right)\left(\partial_l \bar{x}^{\bar{l}}\right) \bar{\Gamma}_{\bar{k}\bar{l}}{}^{\bar{i}} + \left(\partial_k \partial_l \bar{x}^{\bar{i}}\right)\right) A^k \delta x^l \end{aligned}$$

and:

$$D\bar{A}^{\bar{i}} = \left(\partial_i \bar{x}^{\bar{i}}\right) dA^i + \left(\partial_i \bar{x}^{\bar{i}}\right) \Gamma_{kl}{}^i A^k \delta x^l$$

Therefore, under change of coordinates:

$$\begin{aligned} \bar{\Gamma}_{\bar{k}\bar{l}}{}^{\bar{i}} &= \left(\bar{\partial}_{\bar{k}} x^k\right)\left(\bar{\partial}_{\bar{l}} x^l\right)\left(\partial_i \bar{x}^{\bar{i}}\right) \Gamma_{kl}{}^i - \left(\bar{\partial}_{\bar{k}} x^k\right)\left(\bar{\partial}_{\bar{l}} x^l\right)\left(\partial_k \partial_l \bar{x}^{\bar{i}}\right) && \text{(B.49)} \\ &= \left(\bar{\partial}_{\bar{k}} x^k\right)\left(\bar{\partial}_{\bar{l}} x^l\right)\left(\partial_i \bar{x}^{\bar{i}}\right) \Gamma_{kl}{}^i + \left(\partial_i \bar{x}^{\bar{i}}\right)\left(\bar{\partial}_{\bar{k}} \bar{\partial}_{\bar{l}} x^i\right) && \text{(B.50)} \end{aligned}$$

which clearly demonstrates that the connection $\Gamma_{ij}{}^k$ is *not* a tensor.

B.3.3 The Covariant Derivative

Using the connection, the covariant derivative of a generic relative tensor field $\underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ of weight w is defined to be:

$$\begin{aligned} D_k \underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} &= \frac{D \underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}}{\delta x^k} \\ &= \partial_k \underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} + \sum_{p=1}^{n} \Gamma_{kl}{}^{i_p} \underset{w}{A}{}^{i_1 i_2 \ldots i_{p-1} k i_{p+1} \ldots i_n}_{j_1 j_2 \ldots j_m} - \sum_{q=1}^{m} \Gamma_{j_q l}{}^{k} \underset{w}{A}{}^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_{q-1} k j_{q+1} \ldots j_m} \end{aligned}$$

which represents the physical rate of change of that field, and can be shown to be a relative tensor field of weight w and order $n + (m+1)$.

The usual (non-covariant) derivative ∂_i has certain useful properties, including:

1. $[\partial_i, \partial_j] \underset{w}{A}{}^{\cdots}_{\cdots} = 0$ (commutativity).

2. $\partial_i \left(\underset{w}{A}{}^{\cdots}_{\cdots} \underset{w}{B}{}^{\cdots}_{\cdots}\right) = \left(\partial_i \underset{w}{A}{}^{\cdots}_{\cdots}\right) \underset{w}{B}{}^{\cdots}_{\cdots} + \underset{w}{A}{}^{\cdots}_{\cdots} \partial_i \underset{w}{B}{}^{\cdots}_{\cdots}$.

3. $\partial_i \left(\underset{w}{A}{}^{\cdots}_{\cdots} + \underset{w}{B}{}^{\cdots}_{\cdots}\right) = \partial_i \underset{w}{A}{}^{\cdots}_{\cdots} + \partial_i \underset{w}{B}{}^{\cdots}_{\cdots}$.

where $[a,b] = ab - ba$ is the usual commutator. The question is whether these relations will hold for the covariant derivative, D_i. The issue of commutativity (item 1) will be dealt with later (in general, the covariant derivative is non-commutative). Otherwise, it is straightforward to show that:

1. $D_i \left(\underset{w}{A}{}^{\cdots}_{\cdots} \underset{w}{B}{}^{\cdots}_{\cdots}\right) = \left(D_i \underset{w}{A}{}^{\cdots}_{\cdots}\right) \underset{w}{B}{}^{\cdots}_{\cdots} + \underset{w}{A}{}^{\cdots}_{\cdots} D_i \underset{w}{B}{}^{\cdots}_{\cdots}$.

2. $D_i \left(\underset{w}{A}{}^{\cdots}_{\cdots} + \underset{w}{B}{}^{\cdots}_{\cdots} \right) = D_i \underset{w}{A}{}^{\cdots}_{\cdots} + D_i \underset{w}{B}{}^{\cdots}_{\cdots}$.

Note that item 1 indicates that raising and lowering of indices can safely be applied to a covariant derivative. For example:

$$\begin{aligned} g_{im} D_n A^{mj}{}_k &= D_n A_i{}^j{}_k \\ g^{ij} D_n A_{ij}{}^k &= D_n A_i{}^i{}_k \\ \delta^i{}_j D_n A_i{}^{jk} &= D_n A_i{}^i{}_k \\ D^n A^i &= g^{nm} D_m A^i \\ &\dots \end{aligned}$$

Consider the derivative:

$$\begin{aligned} D_k \delta_i{}^j &= \partial_k \delta_i{}^j - \Gamma_{ik}{}^l \delta_l{}^j + \Gamma_{lk}{}^j \delta_i{}^l \\ &= -\Gamma_{ik}{}^j + \Gamma_{ik}{}^j \\ &= 0 \end{aligned}$$

So, given that $g_{im} g^{mj} = \delta_i{}^j$, it follows that:

$$\begin{aligned} D_k \left(g_{im} g^{mj} \right) &= g^{mj} D_k g_{im} + g_{im} D_k g^{mj} \\ &= 0 \end{aligned}$$

lowering j, it may be seen that:

$$\delta^m{}_j D_k g_{im} = -g_{im} D_k \delta^m{}_j = 0$$
$$\therefore D_k g_{ij} = 0 \tag{B.51}$$

and likewise, raising i:

$$-g^{mj} D_k \delta^i{}_m = \delta^i{}_m D_k g^{mj} = 0$$
$$\therefore D_k g^{ij} = 0$$

which implies that both the metric tensor g_{ij} and its conjugate g^{ij} are physically the same everywhere, although they may vary numerically.

B.3.4 Torsion, Contorsion and the Christoffel Symbols

For reasons which will become apparent shortly, it is convenient to split the connection $\Gamma_{ij}{}^k$ into completely symmetric and completely antisymmetric parts, based on the indices ij. Specifically:

$$\Gamma_{ij}{}^k = T_{ij}{}^k + Q_{ij}{}^k$$

where $T_{ij}{}^k = \Gamma_{\{ij\}}{}^k$ is the completely symmetric component of $\Gamma_{ij}{}^k$, and $Q_{ij}{}^k = \Gamma_{[ij]}{}^k$ the completely antisymmetric component. The completely antisymmetric component $Q_{ij}{}^k$ is known as the *torsion*.

Consider (B.49). For the torsion component:

$$\begin{aligned}
\bar{Q}_{\bar{k}\bar{l}}{}^{\bar{i}} &= \left(\bar{\partial}_{\bar{k}} x^k\right)\left(\bar{\partial}_{\bar{l}} x^l\right)\left(\partial_i \bar{x}^{\bar{i}}\right) Q_{kl}{}^i - \left(\bar{\partial}_{[\bar{k}} x^k\right)\left(\bar{\partial}_{\bar{l}]} x^l\right)\left(\partial_k \partial_l \bar{x}^{\bar{i}}\right) \\
&= \left(\bar{\partial}_{\bar{k}} x^k\right)\left(\bar{\partial}_{\bar{l}} x^l\right)\left(\partial_i \bar{x}^{\bar{i}}\right) Q_{kl}{}^i
\end{aligned}$$

where the symmetry "trick" introduced in section B.1.7 has been used to remove the second term. This shows that the torsion component $Q_{ij}{}^k$ of the connection is a tensor, called the *torsion tensor*. So, the connection is the sum of a completely symmetric, nontensorial component $T_{ij}{}^k$ and a completely anti-symmetric tensorial component $Q_{ij}{}^k$ (the torsion).

Now, consider the Christoffel symbol of the second kind, $\left\{ \begin{array}{c} k \\ ij \end{array} \right\}$, as defined in (B.32). Combining (B.34) and (B.49), it follows that:

$$\left[\bar{\Gamma}_{\bar{i}\bar{j}}{}^{\bar{k}} - \overline{\left\{ \begin{array}{c} \bar{k} \\ \bar{i}\bar{j} \end{array} \right\}} \right] = \left(\bar{\partial}_{\bar{i}} x^i\right)\left(\bar{\partial}_{\bar{j}} x^j\right)\left(\partial_k \bar{x}^{\bar{k}}\right) \left[\Gamma_{ij}{}^k - \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \right]$$

From which it may be concluded that:

$$\Gamma_{ij}{}^k = \left\{ \begin{array}{c} k \\ ij \end{array} \right\} - K_{ij}{}^k \tag{B.52}$$

where $K_{ij}{}^k$ is a tensor, called the *contorsion tensor*.

Now, defining:

$$\begin{aligned}
\Gamma_{ijk} &= g_{kl} \Gamma_{ij}{}^l \\
T_{ijk} &= g_{kl} T_{ij}{}^l
\end{aligned}$$

it follows that:

$$\begin{aligned}
D_k g_{ij} &= \partial_k g_{ij} - \Gamma_{ik}{}^l g_{lj} - \Gamma_{jk}{}^l g_{li} \\
&= \partial_k g_{ij} - \Gamma_{ikj} - \Gamma_{jki}
\end{aligned}$$

Using (B.51) and permuting:

$$\begin{aligned}
\partial_k g_{ij} &= \Gamma_{ikj} + \Gamma_{jki} && \text{(B.53)} \\
\partial_i g_{kj} &= \Gamma_{kij} + \Gamma_{jik} && \text{(B.54)} \\
\partial_j g_{ik} &= \Gamma_{ijk} + \Gamma_{kji} && \text{(B.55)}
\end{aligned}$$

Computing the sum −(B.53)+(B.54)+(B.55), it can be seen that:

$$[ij\ \ k] = Q_{kij} + Q_{kji} + T_{ijk} \tag{B.56}$$

Using (B.52), it is trivial to see that:

$$T_{ijk} = [ij\ \ k] - K_{ijk} - Q_{ijk} \tag{B.57}$$

Combining (B.56) and (B.57), using the symmetries of the various objects and raising k, it follows that:

$$\begin{aligned} K_{ij}{}^{k} &= -Q_{ij}{}^{k} - Q_{i}{}^{k}{}_{j} - Q_{j}{}^{k}{}_{i} \\ &= -Q_{ij}{}^{k} + 2Q^{k}{}_{\{ij\}} \end{aligned}$$

The important point to note here is that if the space is torsionless, the connection is just the Christoffel symbol of the second kind. If, on the other hand, torsion is present, then connection is determined completely by the Christoffel symbol and the completely anti-symmetric (in indices ij) torsion tensor $Q_{ij}{}^{k}$. Note, however, that in this case the symmetric part is not simply the Christoffel symbol, but includes an additional component dependent on the torsion.

So, noting that:

$$\begin{aligned} K_{[ij]}{}^{k} &= -Q_{ij}{}^{k} \\ K_{\{ij\}}{}^{k} &= -Q_{i}{}^{k}{}_{j} - Q_{j}{}^{k}{}_{i} \\ K_{ij}{}^{k} &= -K^{k}{}_{ji} \end{aligned}$$

it follows that:

$$\begin{aligned} T_{ij}{}^{k} &= \left\{ \begin{array}{c} k \\ ij \end{array} \right\} + Q_{i}{}^{k}{}_{j} + Q_{j}{}^{k}{}_{i} \\ &= \left\{ \begin{array}{c} k \\ ij \end{array} \right\} - 2Q^{k}{}_{\{ij\}} \end{aligned}$$

For future reference, consider:

$$\begin{aligned} \Gamma_{ij}{}^{i} &= \left\{ \begin{array}{c} i \\ ij \end{array} \right\} - K_{ij}{}^{i} \\ &= \left\{ \begin{array}{c} i \\ ij \end{array} \right\} + Q_{i}{}^{i}{}_{j} + Q_{j}{}^{i}{}_{i} + Q_{ij}{}^{i} \\ &= \left\{ \begin{array}{c} i \\ ij \end{array} \right\} - Q^{i}{}_{ji} + Q_{ij}{}^{i} \\ &= \left\{ \begin{array}{c} i \\ ij \end{array} \right\} \end{aligned}$$

Now:

$$\begin{aligned} \left\{ \begin{array}{c} i \\ ij \end{array} \right\} &= g^{ik}\frac{1}{2}\left(\partial_i g_{kj} + \partial_j g_{ik} - \partial_k g_{ij}\right) \\ &= \frac{1}{2}\left(\partial^k g_{kj} + g^{ik}\partial_j g_{ik} - \partial^i g_{ij}\right) \\ &= \frac{1}{2}g^{ik}\partial_i g_{kj} \end{aligned}$$

Using (B.20), it follows that:

$$\Gamma_{ij}{}^{i} = \left\{ \begin{array}{c} i \\ ij \end{array} \right\} = \partial_j \left(\ln \sqrt{|g|} \right)$$

For later reference, the following properties are presented without proof:

$$\begin{aligned}
Q_{ij}{}^{k} &= Q_{ij}{}^{k} && \text{(B.58)}\\
Q_{[ij]}{}^{k} &= Q_{ij}{}^{k} && \text{(B.59)}\\
Q_{i[j\overset{k}{\bullet}]} &= \frac{1}{2}\left(Q_{ij}{}^{k} + Q^{k}{}_{ij}\right) && \text{(B.60)}\\
Q^{[\bullet\ k]}_{\ i\ j} &= \frac{1}{2}\left(Q_{ij}{}^{k} - Q^{k}{}_{ji}\right) && \text{(B.61)}\\
Q_{\{ij\}}{}^{k} &= 0 && \text{(B.62)}\\
Q_{i\{j\overset{k}{\bullet}\}} &= \frac{1}{2}\left(Q_{ij}{}^{k} - Q^{k}{}_{ij}\right) && \text{(B.63)}\\
Q^{\{\bullet\ k\}}_{\ i\ j} &= \frac{1}{2}\left(Q_{ij}{}^{k} + Q^{k}{}_{ji}\right) && \text{(B.64)}\\
Q_{\underbrace{ij\overset{k}{\bullet}}} &= \frac{1}{3}\left(Q_{ij}{}^{k} + Q^{k}{}_{ij} - Q^{k}{}_{ji}\right) && \text{(B.65)}\\
Q_{\underline{ij\overset{k}{\bullet}}} &= 0 && \text{(B.66)}
\end{aligned}$$

$$\begin{aligned}
K_{ij}{}^{k} &= -Q_{ij}{}^{k} + Q^{k}{}_{ij} + Q^{k}{}_{ji} && \text{(B.67)}\\
K_{[ij]}{}^{k} &= -Q_{ij}{}^{k} + Q^{k}{}_{ij} + Q^{k}{}_{ji} && \text{(B.68)}\\
K_{i[j\overset{k}{\bullet}]} &= Q^{k}{}_{ji} && \text{(B.69)}\\
K^{[\bullet\ k]}_{\ i\ j} &= -Q_{ij}{}^{k} + Q^{k}{}_{ij} + Q^{k}{}_{ji} && \text{(B.70)}\\
K_{\{ij\}}{}^{k} &= 0 && \text{(B.71)}\\
K_{i\{j\overset{k}{\bullet}\}} &= -Q_{ij}{}^{k} + Q^{k}{}_{ij} && \text{(B.72)}\\
K^{\{\bullet\ k\}}_{\ i\ j} &= 0 && \text{(B.73)}\\
K_{\underbrace{ij\overset{k}{\bullet}}} &= -\frac{1}{3}\left(Q_{ij}{}^{k} + Q^{k}{}_{ij} - Q^{k}{}_{ji}\right) && \text{(B.74)}\\
K_{\underline{ij\overset{k}{\bullet}}} &= 0 && \text{(B.75)}
\end{aligned}$$

B.3.5 Autoparallels and Geodesics

In section B.2.5 the concept of a geodesic curve was introduced as a curve of shortest length in a general space. In particular, it was shown that a curve $x^i(u)$ is geodesic if, for all $\acute{u} \leq u \leq \grave{u}$:

$$\frac{d^2x^k}{du^2} + \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \frac{dx^i}{du}\frac{dx^j}{du} = \lambda \frac{dx^k}{du}$$

for some $\lambda = \lambda(u)$. This is called the geodesic equation.

I also noted that this does not, in general, represent the straightest path. Now, having precisely defined the concept of parallelism between objects at different locations it is possible to make this argument formally, by constructing the set of equations which must be satisfied by a straight line.

Consider the curve C given by $x^i = x^i(u)$ for $\acute{u} \leq u \leq \grave{u}$. For some vector field w^i, the physical rate of change of w^i as one moves along C (by varying u) is:

$$\begin{aligned} \frac{Dw^k}{du} &= \left(D_i w^k\right) \frac{dx^i}{du} \\ &= \left(\partial_i w^k\right) \frac{dx^i}{du} + \Gamma_{ij}{}^k w^i \frac{dx^j}{du} \\ &= \frac{dw^k}{du} + \Gamma_{ij}{}^k w^i \frac{dx^j}{du} \end{aligned}$$

Now, $\frac{dx^i}{du}$ is tangent to (i.e. points in the direction of) the curve C at the point $x^i(u)$. Suppose then that $w^i = \frac{dx^i}{du}$. It follows that:

$$\frac{D}{du}\frac{dx^k}{du} = \frac{d^2x^k}{du^2} + \Gamma_{ij}{}^k \frac{dx^i}{du}\frac{dx^j}{du} \tag{B.76}$$

is the physical rate of change of the tangent of the curve.

The curve C is said to be *autoparallel* if at all points $x^i(u)$ the tangent vector $\frac{dx^i}{du}$ is physically parallel to the tangent at every other point $x^i(v)$. That is, if the direction of the curve is constant. This will be achieved if:

$$\frac{D}{du}\frac{dx^k}{du} = \lambda \frac{dx^k}{du}$$

for some $\lambda = \lambda(u)$. This implies that the direction of the curve is constant.

Plugging this into (B.76) it follows that a curve $x^i(u)$ is autoparallel (straight) if, for all u:

$$\frac{d^2x^k}{du^2} + \Gamma_{ij}{}^k \frac{dx^i}{du}\frac{dx^j}{du} = \lambda \frac{dx^k}{du}$$

for some $\lambda = \lambda(u)$.

Applying symmetry arguments, this may be rewritten:

$$\frac{d^2x^k}{du^2} + \left\{ \begin{array}{c} k \\ ij \end{array} \right\} \frac{dx^i}{du}\frac{dx^j}{du} - 2Q^k{}_{\{ij\}}\frac{dx^i}{du}\frac{dx^j}{du} = \lambda \frac{dx^k}{du} \tag{B.77}$$

which is known as the *autoparallel equation.*

Comparing the geodesic equation (B.29) and the autoparallel equation (B.77) it is clear that a geodesic curve will be autoparallel (straight) if the torsion $Q_{ij}{}^k$ is zero. Indeed, the connection between the two is rather more intimate than this. However, as this is only an appendix, and the subject of torsion rather tangential to the thesis as a whole, it would be inappropriate to go into too much detail here.

Briefly, however, consider construction of a parallelogram, as shown in figure B.5. This is done by starting with two infinitesimal vectors $d_{(1)}x^i$ and $d_{(2)}x^i$ at y^i.

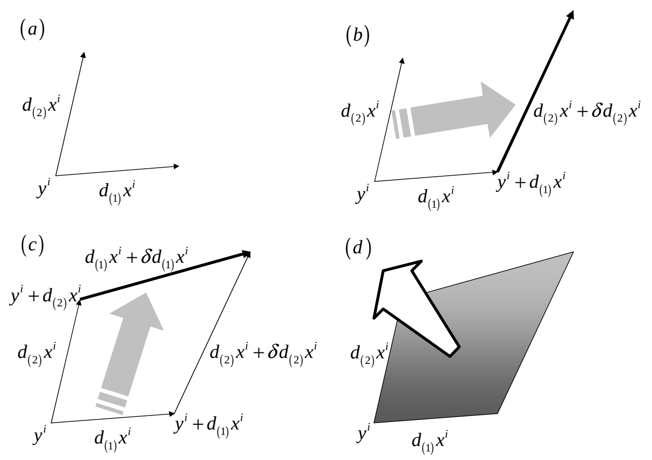


Figure B.5: Construction of a parallelogram in curved space.

These form two sides of the parallelogram. To construct the remaining sides of the parallelogram, the infinitesimal vector $d_{(1)}x^i$ is parallel transported to $y^i + d_{(2)}x^i$ and the similarly the infinitesimal $d_{(2)}x^i$ is parallel transported to $y^i + d_{(1)}x^i$.

Now:

$$\begin{aligned} \delta d_{(1)}x^i &= -\Gamma^i_{jk} d_{(1)}x^j d_{(2)}x^k \\ \delta d_{(2)}x^i &= -\Gamma^i_{jk} d_{(2)}x^j d_{(1)}x^k \end{aligned}$$

and hence:

$$\begin{aligned} d_{(1)}x^i + d_{(2)}x^i + \delta d_{(2)}x^i &= d_{(1)}x^i + d_{(2)}x^i - \Gamma^i_{jk} d_{(2)}x^j d_{(1)}x^k \\ &= d_{(1)}x^i + d_{(2)}x^i - T^i_{jk} d_{(2)}x^j d_{(1)}x^k - Q^i_{jk} d_{(2)}x^j d_{(1)}x^k \\ d_{(2)}x^i + d_{(1)}x^i + \delta d_{(1)}x^i &= d_{(2)}x^i + d_{(1)}x^i - \Gamma^i_{jk} d_{(1)}x^j d_{(2)}x^k \\ &= d_{(2)}x^i + d_{(1)}x^i - \Gamma^i_{kj} d_{(2)}x^j d_{(1)}x^k \\ &= d_{(2)}x^i + d_{(1)}x^i - T^i_{kj} d_{(2)}x^j d_{(1)}x^k - Q^i_{kj} d_{(2)}x^j d_{(1)}x^k \\ &= d_{(2)}x^i + d_{(1)}x^i - T^i_{jk} d_{(2)}x^j d_{(1)}x^k + Q^i_{jk} d_{(2)}x^j d_{(1)}x^k \\ &= \left(d_{(1)}x^i + d_{(2)}x^i + \delta d_{(2)}x^i \right) + 2Q^i_{jk} d_{(2)}x^j d_{(1)}x^k \end{aligned}$$

But this implies that, if the torsion tensor is nonzero at y^i, the parallelogram will not be closed as the two sides furthest from y^i will not meet. While it would be inappropriate to go into too much detail here, the implication of this result is that if one was to try to navigate in such a space, following what would appear to be a closed path from a Euclidean space perspective would in fact result in a finite displacement in a space with torsion. Thus the effect of torsion is to convert closed paths to open paths. In the present context, however, it is convenient to simply assume that torsion is zero (an assumption which is also made in most texts on the subject of differential geometry).

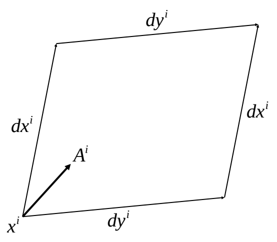


Figure B.6: Basic setup for parallel transport around an infinitesimal loop.

B.3.6 Curvature

Consider a tourist visiting one of the general spaces described here. Assume that this tourist, while accustomed to existence in a more prosaic Euclidean space, is keen to experience the strange wonders of non-Euclidean space first hand. The question is: how might he do so? Indeed, apart from the peculiarity of shortest paths not being straight as seen in the previous section, what oddities should our tourist be on the lookout for?

Consider a vector A^i at a point P in space. To parallel transport this vector to another point Q, choose a path C from P to Q and parallel transport A^i along this path. The result of this process will be a vector $A^i + \Delta_C A^i$ at Q, where $\Delta_C A^i$ is some finite (numerical) change due to parallel transport, the subscript C indicating the path used to calculate $\Delta_C A^i$. In a "normal" space one would expect that this change $\Delta_C A^i$ would be independent of the path C - that is, there is a concept of absolute parallelism over finite distances, not just infinitesimal distances. Failure to meet this condition is an indication that the space is *curved*.

Now, if parallel transporting a tensor from P to Q gives a different result depending on what path is taken, then parallel transport around a closed loop (i.e. from P to Q along one path, and then back to P along another) should result in a change in the tensor.

To begin, consider the infinitesimal closed 2-region shown in figure B.6 (which, it should be noted, is not necessarily a parallelogram). Suppose the vector A^i at x^i is parallelly transported around the edge of this region in a clockwise direction. Note that:

$$\begin{aligned}
\Gamma_{ij}{}^{k}\left(x^i\right) &= \Gamma_{ij}{}^{k} \\
\Gamma_{ij}{}^{k}\left(x^i + dy^i\right) &= \Gamma_{ij}{}^{k} + \left(\partial_l \Gamma_{ij}{}^{k}\right) dy^l \\
\Gamma_{ij}{}^{k}\left(x^i + dx^i + dy^i\right) &= \Gamma_{ij}{}^{k} + \left(\partial_l \Gamma_{ij}{}^{k}\right) dx^l + \left(\partial_l \Gamma_{ij}{}^{k}\right) dy^l \\
\Gamma_{ij}{}^{k}\left(x^i + dx^i\right) &= \Gamma_{ij}{}^{k} + \left(\partial_l \Gamma_{ij}{}^{k}\right) dx^l
\end{aligned}$$

Neglecting terms of order $o\left(dx^i dx^j\right)$, $o\left(dy^i dy^j\right)$ for reasons which will become apparent shortly:

- Start with the value A^k at x^i.

- Moving from x^i to $x^i + dy^i$ results in a numerical change:

$$\delta A^k = -\Gamma_{ij}{}^k A^i dy^j$$

  to give the value $B^k = A^k - \Gamma_{ij}{}^k A^i dy^j$ at $x^i + dy^i$.

- Moving from $x^i + dy^i$ to $x^i + dx^i + dy^i$ will result in a numerical change:

$$\delta B^k = -\Gamma_{ij}{}^k A^i dx^j + \Gamma_{mn}{}^k \Gamma_{ij}{}^m A^i dx^n dy^j - \left(\partial_n \Gamma_{ij}{}^k\right) A^i dx^j dy^n$$

  to give the value $C^k = A^k - \Gamma_{ij}{}^k A^i dx^j - \Gamma_{ij}{}^k A^i dy^j + \Gamma_{mn}{}^k \Gamma_{ij}{}^m A^i dx^n dy^j - \left(\partial_n \Gamma_{ij}{}^k\right) A^i dx^j dy^n$ at $x^i + dx^i + dy^i$.

- Moving from $x^i + dx^i + dy^i$ to $x^i + dx^i$ will result in a numerical change:

$$\delta C^k = \Gamma_{ij}{}^k A^i dy^j - \Gamma_{mn}{}^k \Gamma_{ij}{}^m A^i dx^j dy^n + \left(\partial_n \Gamma_{ij}{}^k\right) A^i dx^n dy^j$$

  to give the value $D^k = A^k - \Gamma_{ij}{}^k A^i dx^j + \Gamma_{mn}{}^k \Gamma_{ij}{}^m A^i dx^n dy^j - \Gamma_{mn}{}^k \Gamma_{ij}{}^m A^i dx^j dy^n - \left(\partial_n \Gamma_{ij}{}^k\right) A^i dx^j dy^n + \left(\partial_n \Gamma_{ij}{}^k\right) A^i dx^n dy^j$ at $x^i + dx^i$.

- Moving from $x^i + dx^i$ to x^i will result in a numerical change:

$$\delta D^k = \Gamma_{ij}{}^k A^i dx^j$$

  to give the value $E^k = A^k + \Gamma_{mn}{}^k \Gamma_{ij}{}^m A^i dx^n dy^j - \Gamma_{mn}{}^k \Gamma_{ij}{}^m A^i dx^j dy^n - \left(\partial_n \Gamma_{ij}{}^k\right) A^i dx^j dy^n + \left(\partial_n \Gamma_{ij}{}^k\right) A^i dx^n dy^j$ at x^i.

From which it may concluded that parallel transporting the vector A^i around the infinitesimal closed counterclockwise loop in figure B.6 will result in an infinitesimal change in the vector, namely:

$$\delta A^k = R^k{}_{lij} A^l dx^i dy^j$$

where $R^k{}_{mij}$ is called the *curvature tensor* (the tensorial character will be demonstrated shortly) defined as:

$$R^k{}_{lij} = \left(\partial_i \Gamma_{lj}{}^k\right) - \left(\partial_j \Gamma_{li}{}^k\right) + \Gamma_{lj}{}^m \Gamma_{mi}{}^k - \Gamma_{li}{}^m \Gamma_{mj}{}^k$$

Similar reasoning for a vector A_i at x^i leads to the conclusion that parallel transported clockwise around the same closed loop will result in the change:

$$\delta_C A_k = R^l{}_{kij} A_l dx^i dy^j$$

and hence, using (B.4), it can be seen that parallelly transporting an arbitrary (oriented or non-oriented) relative tensor $A^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m}$ around the same loop will result in the change (once again neglecting higher order terms):

$$\delta_C A^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_m} = \left(\sum_{p=1}^{n} R^{i_p}{}_{lij} A^{i_1 i_2 \ldots i_{p-1} l i_{p+1} \ldots i_n}_{j_1 j_2 \ldots j_m} + \sum_{q=1}^{m} R^l{}_{j_q ij} A^{i_1 i_2 \ldots i_n}_{j_1 j_2 \ldots j_{q-1} l j_{q+1} \ldots j_m} \right) dx^i dy^j$$

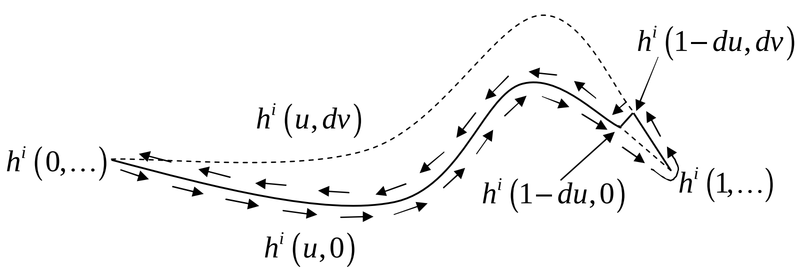


Figure B.7: First step in constructing the path $h^i(dv)$ from the edge path $h^i(0)$.

However, our primary interest is in finite loops, not infinitesimal loops. To extend this reasoning to a finite loop, consider a clockwise loop ∂C starting and ending at a point x^i. Selecting another point y^i on this loop, construct a family $h^i(v,u)$ of curves from x^i to y^i in a manner similar to section B.2.5 (i.e. v specifies the member of the family, u the position along that curve) such that $\partial C = h_i(0,u) - h_i(1,u)$. For simplicity, assume $0 \leq u, v \leq 1$. Thus $h^i(v,u)$ defines a 2-surface C bounded by ∂C and parametrised by (u,v), where $h^i(v,0)$ corresponds to point x^i for all v and $h^i(v,1)$ corresponds to point y^i for all v.

Note that:

- Fixing v gives a path from x^i to y^i.
- Fixing u gives a path from $h^i(0,u)$ to $h^i(1,u)$.
- $\frac{\partial h^i}{\partial v}(v,0) = \frac{\partial h^i}{\partial v}(v,1) = 0$.

Consider the path $h^i(0,u)$. Given a vector A^i at x^i, the change $\Delta_{\partial C_0} A^i$ in A^i due to parallel transport around the loop $\partial C_0 = h^i(0,u) - h^i(0,u)$ will be zero. Now suppose that we slightly modify this loop as shown in figure B.7. Then the change δA^i due to parallel transport around this modified loop will be:

$$\begin{aligned}\delta A^k &= \Delta_{\partial C_0} A^k + R^k{}_{lij}\left(h^m(0,u-du)\right) A^l \frac{dh^i}{du}(0,u-du)\frac{dh^j}{dv}(0,u-du)\,dudv \\ &= R^k{}_{lij}\left(h^m(0,u-du)\right) A^l \frac{dh^i}{du}(0,u-du)\frac{dh^j}{dv}(0,u-du)\,dudv\end{aligned}$$

Repeating this process moving from y^i to x^i, it is easy to see that the change in A^i due to parallel transport around the loop $\partial C_{dv} = h^i(0,u) - h^i(dv,u)$ is:

$$\Delta_{\partial C_{dv}} A^k = \left.\left(\int_{u=0}^{1} R^k{}_{lij} A^l \frac{dh^i}{du}\frac{dh^j}{dv}du\right)\right|_{v=0} dv$$

from which it is straightforward to show that the change $\Delta_{\partial C} A^i$ in the vector A^i

due to parallel transport around ∂C will be:[6]

$$\begin{aligned}\Delta_{\partial C} A^k &= \int_{v=0}^{1} \int_{u=0}^{1} R^k{}_{lij} A^l \frac{dh^i}{du} \frac{dh^j}{dv} du dv \\ &= \iint_C R^k{}_{lij} A^l dx^i dx^j\end{aligned}$$

More generally, given a closed clockwise loop ∂C bounding some (arbitrary) 2-surface C, and some (possibly oriented) relative tensor $A^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$ defined at some point x^i on this loop, parallel transporting this object once around the loop will result in a change $\Delta_{\partial C} A^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m}$ to this object, where:

$$\Delta_{\partial C} A^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_m} = \iint_C \left(\sum_{p=1}^{n} R^{i_p}{}_{lij} A^{i_1 i_2 \ldots i_{p-1} l i_{p+1} \ldots i_n}_{j_1 j_2 \cdots j_m} + \sum_{q=1}^{m} R^l{}_{j_q ij} A^{i_1 i_2 \ldots i_n}_{j_1 j_2 \cdots j_{q-1} l j_{q+1} \cdots j_m} \right) dx^i dx^j$$

is independent of C (so long as C is bounded by δC).

What does this mean? Consider our tourist. Suppose they set out one morning from the hotel after carefully orienting their maps to match their compass bearings. They then set off on their days adventures, but are careful to ensure that their map continues to point in the same direction wherever they go - that is, they parallel transport it with them. After their days adventures (which take them on a most circuitous route) they arrive back at their hotel to find that their map is no longer pointing in the correct direction!

[6]Note that higher order terms will not effect this expression, which is why they were neglected earlier.

Appendix C

THE GAMMA AND RELATED FUNCTIONS

"We must have some kind of amnesia."
"I don't know what that is, but I'm certain I don't have it. I bathe quite often."

- Xander and Buffy

I think I speak for everyone here when I say, "huh?"

- Buffy

WHEN the asymptotic analysis of monomial support vector regressors was carried out in chapter 5, a number of special functions were required. This appendix introduces these functions and provides the relevant properties.

C.1 The Gamma Function

The gamma function [7] [35] [51] is a continuous extension of the discrete factorial function $n! = \prod_{i=1}^{n} i$ (which is defined for $n \in \mathbb{Z}^+$). For all $x \in \Re$, the gamma function is defined to be:

$$\Gamma\left(x\right) = \int_0^\infty t^{x-1} e^{-t} dt \tag{C.1}$$

Or equivalently, in the Euler limit form [7]:

$$\Gamma\left(x\right) = \frac{1}{x} \prod_{n=1}^{\infty} \left[\left(1 + \frac{1}{n}\right)^x \left(1 + \frac{x}{n}\right)^{-1} \right] \tag{C.2}$$

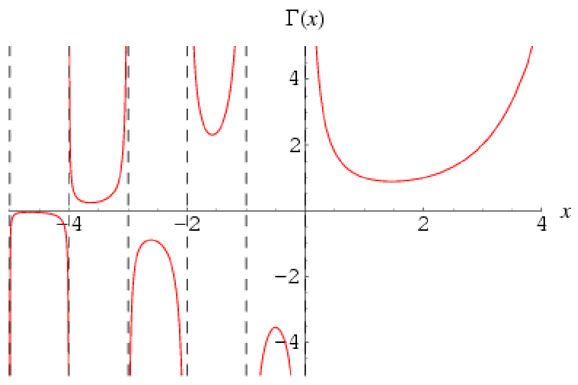


Figure C.1: The gamma function (source [100]).

C.1.1 Results

It can be shown that [35] for all $x \in \Re$:

$$\Gamma(x) = (x-1)\Gamma(x-1) \tag{C.3}$$

Also, for $n \in \mathbb{Z}^+$ [35]:

$$\Gamma(n) = (n-1)! \tag{C.4}$$

The following integral will be useful (equation 3.326, [45]). For $m, p \in \mathbb{Z}^+$, $\beta \in \Re^+$:

$$\int_0^\infty x^m e^{-\beta x^p} dx = \frac{\Gamma(\gamma)}{p\beta^\gamma} \tag{C.5}$$

where:

$$\gamma = \frac{m+1}{p}$$

The derivative of the gamma function is:

$$\frac{d\Gamma(x)}{dx} = \Gamma(x)\psi_0(x)$$

where $\psi_0(x)$ is the digamma (zeroth order polygamma) function, as in section C.2.

Theorem C.1. *$\lim_{x\to 0^+} x\Gamma(ax) = \frac{1}{a}$ for all $a > 0$.*

Proof. Using (C.2) it can be seen that:

$$\begin{aligned} x\Gamma(ax) &= x\frac{1}{ax}\prod_{n=1}^{\infty}\left[\left(1+\frac{1}{n}\right)^{ax}\left(1+\frac{ax}{n}\right)^{-1}\right] \\ &= \frac{1}{a}\prod_{n=1}^{\infty}\left[\left(1+\frac{1}{n}\right)^{ax}\left(1+\frac{ax}{n}\right)^{-1}\right] \end{aligned}$$

and therefore $\lim_{x\to 0^+} x\Gamma(ax) = \frac{1}{a}$. □

C.2 The Polygamma Function

The *polygamma function* is defined to be the $(n+1)^{\text{th}}$ derivative of the (natural) logarithm of the gamma function, i.e.:

$$\psi_n(x) = \frac{d^{n+1}}{dx^{n+1}} \ln[\Gamma(x)] \tag{C.6}$$

The zeroth order polygamma function (also known as the digamma function) is of particular interest here. Specifically, note that [7] the digamma function $\psi_0(x)$ is a monotonically increasing function in the range $(0, \infty)$.

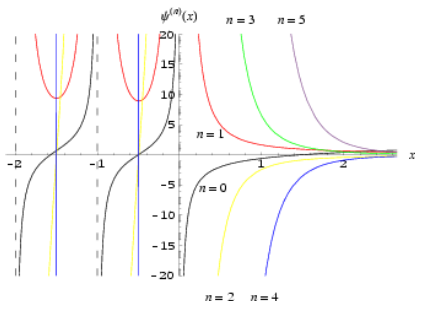


Figure C.2: The polygamma function (source [101]).

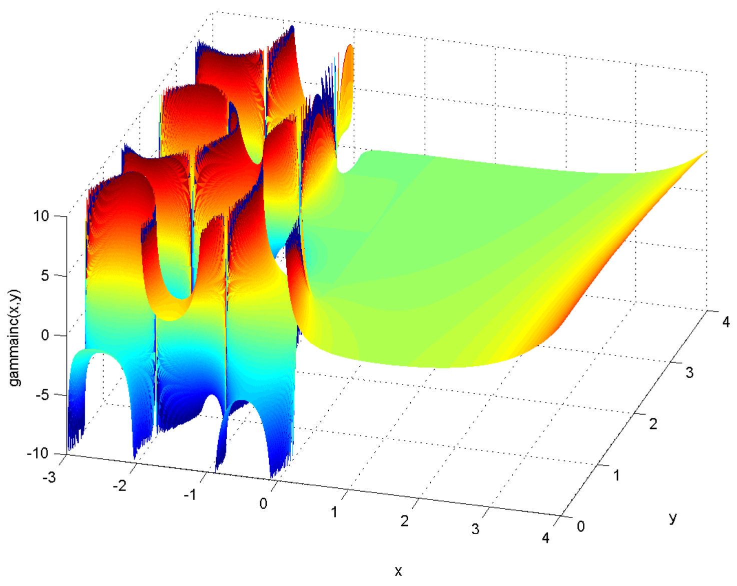


Figure C.3: The incomplete gamma function.

C.3 The Incomplete Gamma Function

The (upper) incomplete gamma function $\Gamma(x, y)$ is a simple extension of the gamma function defined by:

$$\Gamma(x, y) = \int_y^\infty t^{x-1} e^{-t} dt \tag{C.7}$$

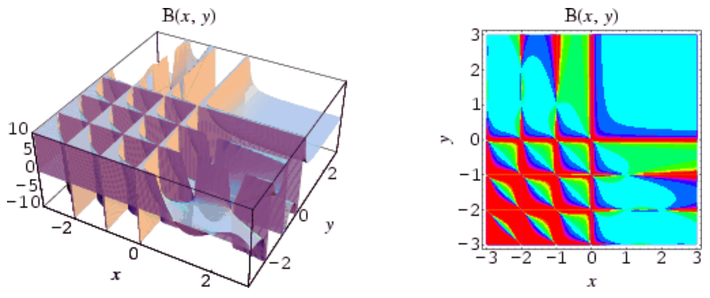


Figure C.4: The beta function (source [99]).

C.3.1 Results

It is not difficult to show that:

$$
\begin{aligned}
\Gamma(x,y) &= y^{x-1}e^{-y} + (x-1)\,\Gamma(x-1,y) && \text{(C.8)}\\
\frac{\partial}{\partial x}\Gamma(x,y) &= \int_y^\infty \ln(t)\, t^{x-1}e^{-t}dt && \text{(C.9)}\\
\frac{\partial}{\partial y}\Gamma(x,y) &= -y^{x-1}e^{-y} && \text{(C.10)}\\
\Gamma(x,0) &= \Gamma(x) && \text{(C.11)}\\
\lim_{y\to\infty}\Gamma(x,y) &= 0 && \text{(C.12)}\\
\int_\omega^\infty \tau^m e^{-\beta\tau^p}d\tau &= \frac{\Gamma(\gamma,\beta\omega^p)}{p\beta^\gamma} && \text{(C.13)}
\end{aligned}
$$

where:

$$\gamma = \frac{m+1}{p}$$

C.4 The Beta Function

The *Beta function* is defined to be ([35], section 1.5, equations (1) and (5)):

$$
\begin{aligned}
B(x,y) &= \frac{\Gamma(x)\,\Gamma(x)}{\Gamma(x+y)}\\
&= \int_0^1 t^{x-1}(1-t)^{y-1}\,dt && \text{(C.14)}
\end{aligned}
$$

C.4.1 Results

Theorem C.2. *$B(a+c,b) > B(a,b+c)$ for all $a > b$, $a,b,c > 0$; and $B(a+c,b) < B(a,b+c)$ for all $a < b$, $a,b,c > 0$.*

Proof. From (C.14), it can be seen that:

$$\begin{aligned} B(a+c,b) &= \frac{B(c,b)}{B(a,c)} B(a,b+c) \\ &= \frac{\frac{\Gamma(b)}{\Gamma(b+c)}}{\frac{\Gamma(a)}{\Gamma(a+c)}} B(a,b+c) \end{aligned} \tag{C.15}$$

But the gradient $\psi_0(x)$ of $\ln(\Gamma(x))$ is a monotonically increasing function of $x > 0$, and so for all $a > b$, $a, b, c > 0$:

$$\ln(\Gamma(a+c)) - \ln(\Gamma(a)) > \ln(\Gamma(b+c)) - \ln(\Gamma(b))$$

so:

$$\ln\left(\frac{\Gamma(a+c)}{\Gamma(a)}\right) > \ln\left(\frac{\Gamma(b+c)}{\Gamma(b)}\right)$$

and hence $\frac{\Gamma(b)}{\Gamma(b+c)} > \frac{\Gamma(a)}{\Gamma(a+c)}$ for all $a > b$, $a, b, c > 0$. Using (C.15), it follows that $B(a+c,b) > B(a,b+c)$ for all $a > b$, $a, b, c > 0$, which proves the first part of the theorem. The proof of the second part is essentially the same, except that, as $a < b$, $\frac{\Gamma(b)}{\Gamma(b+c)} < \frac{\Gamma(a)}{\Gamma(a+c)}$, and so $B(a+c,b) < B(a,b+c)$ for all $a < b$, $a, b, c > 0$. □

C.5 The Backgamma Function

C.5.1 Definition

In this thesis, the following integral will have some importance:

$$\int_\omega^\infty (\tau - \omega)^m e^{-\beta\tau^p} d\tau$$

where $m \in \mathbb{Z} \backslash \mathbb{Z}^-$ and $p \in \mathbb{Z}^+$.

This can be re-expressed (using (C.13)) as:

$$\begin{aligned} \int_\omega^\infty (\tau - \omega)^m e^{-\beta\tau^p} d\tau &= \int_\omega^\infty \left(\sum_{i=0}^m \binom{m}{i} (-\omega)^{m-i} \tau^i \right) e^{-\beta\tau^p} d\tau \\ &= \sum_{i=0}^m \binom{m}{i} (-\omega)^{m-i} \int_\omega^\infty \tau^i e^{-\beta\tau^p} d\tau \\ &= \sum_{i=0}^m \binom{m}{i} (-\omega)^{m-i} \frac{\Gamma\left(\frac{i+1}{p}, \beta\omega^p\right)}{p\beta^{\frac{i+1}{p}}} \end{aligned}$$

This may be re-written:

$$\int_\omega^\infty (\tau - \omega)^m e^{-\beta\tau^p} d\tau = \frac{1}{p\beta^{\frac{m+1}{p}}} \daleth_m \left(\frac{1}{p}, \beta\omega^p \right) \tag{C.16}$$

by defining the *backgamma function*:

$$\begin{aligned} ⅂_m(x,y) &= \frac{1}{x}\int_{y^x}^{\infty} (\tau - y^x)^m e^{-\tau^{1/x}} d\tau \\ &= \int_{y}^{\infty} (\tau^x - y^x)^m \tau^{x-1} e^{-\tau} d\tau \\ &= \sum_{i=0}^{m} \binom{m}{i} (-y^x)^{m-i} \Gamma((i+1)x, y) \end{aligned} \tag{C.17}$$

which is defined for $m \in \mathbb{Z}\backslash\mathbb{Z}^-$, $x \in \Re^+$, $y \in \Re\backslash\Re^-$. Note that for all arguments in this range, $⅂_m(x,y) > 0$.

C.5.2 Basic Properties

It is not too difficult to show that:

$$⅂_m(x,0) = \Gamma((m+1)x) \tag{C.18}$$

$$\lim_{y\to\infty} ⅂_m(x,y) = 0 \tag{C.19}$$

$$⅂_0(x,y) = \Gamma(x,y) \tag{C.20}$$

C.5.3 Differentiation

If $m = 0$ then by (C.9), (C.10):

$$\begin{aligned} \frac{\partial}{\partial x} ⅂_0(x,y) &= \int_y^{\infty} \ln(\tau)\, \tau^{x-1} e^{-\tau} d\tau \\ \frac{\partial}{\partial y} ⅂_0(x,y) &= -y^{x-1} e^{-y} \end{aligned}$$

If $m \geq 1$, using (C.10) it follows that:

$$\begin{aligned} \frac{\partial}{\partial (y^x)} ⅂_m(x,y) &= \frac{\partial}{\partial (y^x)} \int_{y^x}^{\infty} (\tau - y^x)^m e^{-\tau^{\frac{1}{x}}} d\tau \\ &= (y^x - y^x)^m e^{-y} + \int_{y^x}^{\infty} (-m)(\tau - y^x)^{m-1} e^{-\tau^{\frac{1}{x}}} d\tau \\ &= -m \int_{y^x}^{\infty} (\tau - y^x)^{m-1} e^{-\tau^{\frac{1}{x}}} d\tau \\ &= -m ⅂_{m-1}(x,y) \end{aligned}$$

and hence:

$$\begin{aligned}\frac{\partial}{\partial y}⅂_m(x,y) &= \frac{\partial y^x}{\partial y}\frac{\partial}{\partial(y^x)}⅂_m(x,y)\\ &= -mxy^{x-1}⅂_{m-1}(x,y)\end{aligned}$$

So, to summarise:

$$\frac{\partial}{\partial y}⅂_m(x,y) = -xy^{x-1}\left\{\begin{array}{ll} m⅂_{m-1}(x,y) & \text{if } m \neq 0\\ \frac{1}{x}e^{-y} & \text{if } m = 0\end{array}\right. \tag{C.21}$$

Note also that:

$$\frac{\partial}{\partial(y^x)}⅂_m(x,y) = -\left\{\begin{array}{ll} m⅂_{m-1}(x,y) & \text{if } m \neq 0\\ \frac{1}{x}e^{-y} & \text{if } m = 0\end{array}\right. \tag{C.22}$$

C.5.4 Ratio Limits

Consider the limit:

$$\lim_{y\to\infty}\frac{⅂_m(a,cy)}{⅂_n(b,dy)}$$

where $m,n \in \mathbb{Z}\backslash\mathbb{Z}^-$ and $a,b,c,d \in \Re^+$. Using L'opital's rule and (C.22), it may be seen that if $m > n$:

$$\begin{aligned}\lim_{y\to\infty}\frac{⅂_m(a,cy)}{⅂_n(b,dy)} &= \lim_{y^x\to\infty}\frac{⅂_m(a,cy)}{⅂_n(b,dy)}\\ &= \left(\frac{c}{d}\right)^n\left\{\begin{array}{ll}\frac{\prod_{i=m-n+1}^m i}{\prod_{i=1}^n i} & \text{if } n>0\\ 1 & \text{if } n=0\end{array}\right\}\lim_{y^x\to\infty}\frac{⅂_{m-n}(a,cy)}{⅂_0(b,dy)}\\ &= b\left(\frac{c}{d}\right)^{n+1}\frac{\prod_{i=m-n}^m i}{n!}\lim_{y^x\to\infty}\frac{⅂_{m-n-1}(a,cy)}{e^{-dy}}\\ &= b\left(\frac{c}{d}\right)^m\frac{m!}{n!}\lim_{y^x\to\infty}\frac{⅂_0(a,cy)}{e^{-dy}}\\ &= \frac{m!}{n!}\frac{b}{a}\left(\frac{c}{d}\right)^{m+1}\lim_{y^x\to\infty}\frac{e^{-cy}}{e^{-dy}}\\ &= \left\{\begin{array}{ll}0 & \text{if } c>d\\ \frac{m!}{n!}\frac{b}{a} & \text{if } c=d\\ \infty & \text{if } c<d\end{array}\right.\end{aligned}$$

Essentially the same argument may be applied for $m < n$ and $m = n$, and is skipped for brevity. In general, for all $m,n \in \mathbb{Z}\backslash\mathbb{Z}^-$:

$$\lim_{y\to\infty}\frac{⅂_m(a,cy)}{⅂_n(b,dy)} = \left\{\begin{array}{ll}0 & \text{if } c>d\\ \frac{m!}{n!}\frac{b}{a} & \text{if } c=d\\ \infty & \text{if } c<d\end{array}\right.$$

Appendix D

ALGORITHMS

Things involving the computer fill me with a childlike terror. Now, if it were a nice ogre or some such, I'd be more in my element.

- Giles

I'm not ashamed. It's the computer age. Nerds are in. They're still in, right?

- Willow

The four algorithms presented in this appendix relate to the use and upkeep of the factorisation $\boldsymbol{\beth}_{\boldsymbol{\iota}}$, as in chapter 8. They deal with the details of constructing and using the inverse and Cholesky factorisations. Specifically, the algorithms are:

Alg 1 Forward elimination algorithm to solve $\mathbf{Lr} = \mathbf{z}$ quickly for $\mathbf{r}$ given a lower triangular $\mathbf{L} \in \Re^{n \times n}$ with positive diagonals. This is algorithm 4.1-1 in [44]. Requires $\frac{n^2}{2}$ additions and $\frac{n^2}{2}$ multiplications (neglecting $o(n)$ terms).

Alg 2 Back substitution algorithm to solve $\mathbf{Ur} = \mathbf{z}$ quickly for $\mathbf{r}$ given a upper triangular $\mathbf{U} \in \Re^{n \times n}$ with positive diagonals. This is algorithm 4.1-2 in [44]. Requires $\frac{n^2}{2}$ additions and $\frac{n^2}{2}$ multiplications (neglecting $o(n)$ terms).

Alg 3 Inverse factorisation setup algorithm as described in section 8.3.4.

Alg 4 Cholesky factorisation setup algorithm as described in section 8.3.4.

Algorithm 1: Forward Elimination Algorithm.
Input: A lower triangular matrix **L** and a vector **b**.
Output: A vector **y** such that $\mathbf{Ly} = \mathbf{b}$.

```
ForwardElim(L,b)
(1)    for $i = 1$ to $n$
(2)        $\mathbf{y}_i := \mathbf{b}_i$
(3)        for $j = 1$ to $i - 1$
(4)            $\mathbf{y}_i := \mathbf{y}_i - \mathbf{L}_{ij}\mathbf{y}_j$
(5)        $\mathbf{y}_i := \frac{\mathbf{y}_i}{\mathbf{L}_{ii}}$
```

Algorithm 2: Back Substitution Algorithm.
Input: An upper triangular matrix **U** and a vector **y**.
Output: A vector **x** such that $\mathbf{Ux} = \mathbf{y}$.

```
BackwardSub(U,y)
(1)    for $i = n$ to 1
(2)        $\mathbf{x}_i := \mathbf{y}_i$
(3)        for $j = i + 1$ to $n$
(4)            $\mathbf{x}_i := \mathbf{x}_i - \mathbf{U}_{ij}\mathbf{x}_j$
(5)        $\mathbf{x}_i := \frac{\mathbf{x}_i}{\mathbf{U}_{ii}}$
```

Algorithm 3: Calculate $\mathbf{V}$.

```
InvFact($\underline{\mathbf{H}}_{\boldsymbol{\iota}F}$)
(1)  if BiasType = Var
(2)     if $N_F = 0$
(3)        $N_R := 0$
(4)        $\mathbf{V}$ := empty.
(5)        return
(6)     if $N_R = 0$
(7)        $N_R := 1$
(8)        $\mathbf{V} := \begin{bmatrix} -\underline{G}_{\boldsymbol{\iota}F1,1} & 1 \\ 1 & 0 \end{bmatrix}$
(9)     while $N_R < N_F$
(10)       $\begin{bmatrix} \underline{s}_b \\ \underline{\mathbf{s}}_{\boldsymbol{\alpha}} \end{bmatrix} := \mathbf{V} \begin{bmatrix} 1 \\ \underline{\mathbf{g}}_{\boldsymbol{\iota}FBN} \end{bmatrix}$
(11)       $x := \underline{g}_{FB} - \left( \underline{s}_b + \underline{\mathbf{g}}^T_{\boldsymbol{\iota}FBN} \underline{\mathbf{s}}_{\boldsymbol{\alpha}} \right)$
(12)       if $|x| < \varepsilon$ then return
(13)       $\mathbf{V} := \begin{bmatrix} \mathbf{V} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} + \frac{1}{x} \begin{bmatrix} \underline{s}_b \\ \underline{\mathbf{s}}_{\boldsymbol{\alpha}} \\ 1 \end{bmatrix} \begin{bmatrix} \underline{s}_b \\ \underline{\mathbf{s}}_{\boldsymbol{\alpha}} \\ 1 \end{bmatrix}^T$
(14)       $N_R := N_R + 1$
(15)    return
(16) else
(17)    if $N_F = 0$
(18)       $N_R := 0$
(19)       $\mathbf{V}$ := empty.
(20)       return
(21)    while $N_R < N_F$
(22)       $\underline{\mathbf{s}}_{\boldsymbol{\alpha}} := \mathbf{V} \underline{\mathbf{g}}_{\boldsymbol{\iota}FBN}$
(23)       $x := \underline{g}_{FB} - \underline{\mathbf{g}}^T_{\boldsymbol{\iota}FBN} \underline{\mathbf{s}}_{\boldsymbol{\alpha}}$
(24)       if $|x| < \varepsilon$ then return
(25)       $\mathbf{V} := \begin{bmatrix} \mathbf{V} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} + \frac{1}{x} \begin{bmatrix} \underline{\mathbf{s}}_{\boldsymbol{\alpha}} \\ 1 \end{bmatrix} \begin{bmatrix} \underline{\mathbf{s}}_{\boldsymbol{\alpha}} \\ 1 \end{bmatrix}^T$
(26)       $N_R := N_R + 1$
(27)    return
```

Algorithm 4: Calculate **L**.

```
CholFact($\underline{\mathbf{H}}_{\boldsymbol{\iota}F}$)
(1)  if BiasType = Var
(2)     if $N_F = 0$
(3)        $N_R := 0$
(4)        NoChol := FALSE
(5)        $\mathbf{L}$ := empty.
(6)        return
(7)     if $N_R = 0$ or NoChol = TRUE
(8)        if $\underline{G}_{\boldsymbol{\iota}F1,1} < \varepsilon$
(9)           if $\underline{G}_{\boldsymbol{\iota}F2,2} < \varepsilon$ or $N_F = 1$
(10)             $N_R := 1$
(11)             NoChol := TRUE
(12)             $\mathbf{L}$ := empty.
(13)             return
(14)          $\daleth := \mathrm{eswap}\,(N_C + 1, N_C + 2)\daleth$
(15)       $N_R := 1$
(16)       NoChol := FALSE
(17)       $\mathbf{L} := \begin{bmatrix} \sqrt{\underline{G}_{\boldsymbol{\iota}F1,1}} & 0 \\ \frac{1}{\sqrt{\underline{G}_{\boldsymbol{\iota}F1,1}}} & \frac{1}{\sqrt{\underline{G}_{\boldsymbol{\iota}F1,1}}} \end{bmatrix}$
(18)    while $N_R < N_F$
(19)       $\mathbf{a} = \mathrm{ForwardElim}\left(\mathbf{L}, \begin{bmatrix} \underline{g}_{\boldsymbol{\iota}FBNa} \\ 1 \\ \underline{\mathbf{g}}_{\boldsymbol{\iota}FBNb} \end{bmatrix}\right)$.
(20)       $\mathbf{b} := \mathbf{Ja}$
(21)       $c := \underline{g}_{\boldsymbol{\iota}FB} - \mathbf{a}^T\mathbf{b}$
(22)       if $c < \varepsilon$ then return
(23)       $c := \sqrt{c}$
(24)       $\mathbf{L} := \begin{bmatrix} \mathbf{L} & \mathbf{0} \\ \mathbf{b}^T & c \end{bmatrix}$
(25)       $N_R := N_R + 1$
(26)    return
(27) else
(28)    if $N_F = 0$
(29)       $N_R := 0$
(30)       NoChol := FALSE
(31)       $\mathbf{L}$ := empty.
(32)       return
(33)    while $N_R < N_F$
(34)       $\mathbf{a} = \mathrm{ForwardElim}\left(\mathbf{L}, \underline{\mathbf{g}}_{\boldsymbol{\iota}FBN}\right)$.
(35)       $c := \underline{g}_{\boldsymbol{\iota}FB} - \mathbf{a}^T\mathbf{a}$
(36)       if $c < \varepsilon$ then return
(37)       $c := \sqrt{c}$
(38)       $\mathbf{L} := \begin{bmatrix} \mathbf{L} & \mathbf{0} \\ \mathbf{b}^T & c \end{bmatrix}$
(39)       $N_R := N_R + 1$
(40)    return
```

Appendix E

A USEFUL PROPERTY OF THE DETERMINANT

I don't believe it. Prove it to me and I still won't believe it.
- Douglas Adams

In this appendix I present a proof for theorem 4.4. Before proceeding, it should be noted that this is likely a standard result in matrix theory of which I am unaware, and as such I make no claim to the following being an original result or method. It is presented for completeness in the absence of an appropriate reference.

Theorem 4.4 *For any n-dimensional vector* $\mathbf{b}$, $\det\left(\mathbf{I}+\mathbf{b}\mathbf{b}^T\right)=1+\mathbf{b}^T\mathbf{b}$.

Proof. Define:

$$\begin{aligned}
\mathbf{A}^{(j_1,j_2,\ldots j_l)}_{(i_1,i_2,\ldots i_m)} &= \begin{bmatrix} c_{i_1}+b_{i_1}b_{i_1} & b_{i_1}b_{i_2} & \ldots & b_{i_1}b_{i_m} \\ b_{i_2}b_{i_1} & c_{i_2}+b_{i_2}b_{i_2} & \ldots & b_{i_2}b_{i_m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_2} & \ldots & c_{i_m}+b_{i_m}b_{i_m} \end{bmatrix} \\
a^{(j_1,j_2,\ldots,j_l)}_{(i_1,i_2,\ldots i_m)} &= \det\left(\mathbf{A}^{(j_1,j_2,\ldots j_l)}_{(i_1,i_2,\ldots i_m)}\right)
\end{aligned}$$

where $l \leq m$, $l,m \in \mathbb{Z}^+$, $(i_1,i_2,\ldots i_m)$ is an ordered set of m elements such that $i_p < i_q$ for all $p<q$, $(j_1,j_2,\ldots j_l)$ is an ordered set of $l \geq 0$ elements such that $j_p < j_q$ for all $p<q$, $\{j_1,j_2,\ldots,j_l\} \subseteq \{i_1,i_2,\ldots,i_m\}$; and:

$$c_{i_k} = \begin{cases} 1 & \text{if } i_k \notin \{j_1,j_2,\ldots j_l\} \\ 0 & \text{otherwise} \end{cases}$$

Consider the matrix $\mathbf{A}^{(j_1,j_2,\ldots,j_l)}_{(i_1,i_2,\ldots i_m)}$. This may be re-written:

$$\mathbf{A}^{(j_1,j_2,\ldots j_l)}_{(i_1,i_2,\ldots i_m)} = \operatorname{diag}\left(c_{i_1},c_{i_2},\ldots c_{i_m}\right) + \begin{bmatrix} b_{i_1} \\ b_{i_2} \\ \vdots \\ b_{i_m} \end{bmatrix} \begin{bmatrix} b_{i_1} \\ b_{i_2} \\ \vdots \\ b_{i_m} \end{bmatrix}^T$$

which, given that $c_{i_p} \geq 0$ for all p, must be a positive semidefinite matrix. If $l=0$ then:

$$\mathbf{A}^{(j_1,j_2,\ldots,j_l)}_{(i_1,i_2,\ldots i_m)}{}^{-1} = \mathbf{A}^{\emptyset}_{(i_1,i_2,\ldots i_m)}{}^{-1} = \mathbf{I} - \frac{1}{1+\mathbf{d}^T\mathbf{d}}\mathbf{d}\mathbf{d}^T$$

where:

$$\mathbf{d} = \begin{bmatrix} b_{i_1} \\ b_{i_2} \\ \vdots \\ b_{i_m} \end{bmatrix}$$

from which we may conclude that $\mathbf{A}^{(j_1,j_2,\dots j_l)}_{(i_1,i_2,\dots i_m)}$ is non-singular if $l = 0$. More generally, if $m \geq l > 0$ then the matrix $\mathbf{A}^{(j_1,j_2,\dots j_l)}_{(i_1,i_2,\dots i_m)}$ may be trivially re-ordered so that:

$$\begin{aligned}
\mathbf{A}^{\cdots}_{\cdots} &= \begin{bmatrix} \mathbf{I}_{m-l} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{d} \\ \mathbf{f} \end{bmatrix} \begin{bmatrix} \mathbf{d} \\ \mathbf{f} \end{bmatrix}^T \\
&= \begin{bmatrix} \mathbf{I}_{m-l} + \mathbf{d}\mathbf{d}^T & \mathbf{d}\mathbf{f}^T \\ \mathbf{f}\mathbf{d}^T & \mathbf{f}\mathbf{f}^T \end{bmatrix}
\end{aligned}$$

Now, if $l = 1$ and $\mathbf{f} \neq \mathbf{0}$ (or equivalently $b_{j_l} \neq 0$) then this will be non-singular. For all other cases $l \geq 2$ this will be singular. From this, it may be immediately seen that $a^{(j_1,j_2,\dots,j_l)}_{(i_1,i_2,\dots i_m)} = 0$ for all $l \geq 2$.

Expanding $a^{\emptyset}_{(i_1,i_2,\dots i_m)}$ along its top row:

$$\begin{aligned}
a^{\emptyset}_{(i_1,i_2,\dots i_m)} &= \det \begin{bmatrix} 1 + b_{i_1}b_{i_1} & b_{i_1}b_{i_2} & \dots & b_{i_1}b_{i_m} \\ b_{i_2}b_{i_1} & 1 + b_{i_2}b_{i_2} & \dots & b_{i_2}b_{i_m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_2} & \dots & 1 + b_{i_m}b_{i_m} \end{bmatrix} \\
&= (1 + b_{i_1}b_{i_1}) \det \begin{bmatrix} 1 + b_{i_2}b_{i_2} & b_{i_2}b_{i_3} & \dots & b_{i_2}b_{i_m} \\ b_{i_3}b_{i_2} & 1 + b_{i_3}b_{i_3} & \dots & b_{i_3}b_{i_m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_2} & b_{i_m}b_{i_3} & \dots & 1 + b_{i_m}b_{i_m} \end{bmatrix} \\
&\quad - b_{i_2}b_{i_1} \det \begin{bmatrix} b_{i_2}b_{i_1} & b_{i_2}b_{i_3} & \dots & b_{i_2}b_{i_m} \\ b_{i_3}b_{i_1} & 1 + b_{i_3}b_{i_3} & \dots & b_{i_3}b_{i_m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_3} & \dots & 1 + b_{i_m}b_{i_m} \end{bmatrix} + \dots \\
&= \det \begin{bmatrix} 1 + b_{i_2}b_{i_2} & b_{i_2}b_{i_3} & \dots & b_{i_2}b_{i_m} \\ b_{i_3}b_{i_2} & 1 + b_{i_3}b_{i_3} & \dots & b_{i_3}b_{i_m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_2} & b_{i_m}b_{i_3} & \dots & 1 + b_{i_m}b_{i_m} \end{bmatrix} \\
&\quad + \det \begin{bmatrix} b_{i_1}b_{i_1} & b_{i_1}b_{i_2} & \dots & b_{i_1}b_{i_m} \\ b_{i_2}b_{i_1} & 1 + b_{i_2}b_{i_2} & \dots & b_{i_2}b_{i_m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_2} & \dots & 1 + b_{i_m}b_{i_m} \end{bmatrix} \\
&= a^{\emptyset}_{(i_2,i_3,\dots i_m)} + a^{(i_1)}_{(i_1,i_2,\dots i_m)}
\end{aligned}$$

Similarly, expanding $a^{(i_1)}_{(i_1,i_2,\dots i_m)}$ along its second row:

$$\begin{aligned}
a^{(i_1)}_{(i_1,i_2,\dots i_m)} &= \det\begin{bmatrix} b_{i_1}b_{i_1} & b_{i_1}b_{i_2} & b_{i_1}b_{i_3} & \dots & b_{i_1}b_{i_m} \\ b_{i_2}b_{i_1} & 1+b_{i_2}b_{i_2} & b_{i_2}b_{i_3} & \dots & b_{i_2}b_{i_m} \\ b_{i_3}b_{i_1} & b_{i_3}b_{i_2} & 1+b_{i_3}b_{i_3} & \dots & b_{i_3}b_{i_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_2} & b_{i_m}b_{i_3} & \dots & 1+b_{i_m}b_{i_m} \end{bmatrix} \\
&= -b_{i_2}b_{i_1}\det\begin{bmatrix} b_{i_1}b_{i_2} & b_{i_1}b_{i_3} & b_{i_1}b_{i_4} & \dots & b_{i_1}b_{i_m} \\ b_{i_3}b_{i_2} & 1+b_{i_3}b_{i_3} & b_{i_3}b_{i_4} & \dots & b_{i_3}b_{i_m} \\ b_{i_4}b_{i_2} & b_{i_4}b_{i_3} & 1+b_{i_4}b_{i_4} & \dots & b_{i_4}b_{i_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_2} & b_{i_m}b_{i_3} & b_{i_m}b_{i_4} & \dots & 1+b_{i_m}b_{i_m} \end{bmatrix} \\
&\quad + (1+b_{i_2}b_{i_2})\det\begin{bmatrix} b_{i_1}b_{i_1} & b_{i_1}b_{i_3} & b_{i_1}b_{i_4} & \dots & b_{i_1}b_{i_m} \\ b_{i_3}b_{i_1} & 1+b_{i_3}b_{i_3} & b_{i_3}b_{i_4} & \dots & b_{i_3}b_{i_m} \\ b_{i_4}b_{i_1} & b_{i_4}b_{i_3} & 1+b_{i_4}b_{i_4} & \dots & b_{i_4}b_{i_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_3} & b_{i_m}b_{i_4} & \dots & 1+b_{i_m}b_{i_m} \end{bmatrix} \\
&\quad -b_{i_2}b_{i_3}\det\begin{bmatrix} b_{i_1}b_{i_1} & b_{i_1}b_{i_2} & b_{i_1}b_{i_4} & \dots & b_{i_1}b_{i_m} \\ b_{i_3}b_{i_1} & b_{i_3}b_{i_2} & b_{i_3}b_{i_4} & \dots & b_{i_3}b_{i_m} \\ b_{i_4}b_{i_1} & b_{i_4}b_{i_2} & 1+b_{i_4}b_{i_4} & \dots & b_{i_4}b_{i_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_2} & b_{i_m}b_{i_4} & \dots & 1+b_{i_m}b_{i_m} \end{bmatrix} + \dots \\
&= -b_{i_2}b_{i_1}\det\begin{bmatrix} b_{i_1}b_{i_2} & b_{i_1}b_{i_3} & b_{i_1}b_{i_4} & \dots & b_{i_1}b_{i_m} \\ b_{i_3}b_{i_2} & 1+b_{i_3}b_{i_3} & b_{i_3}b_{i_4} & \dots & b_{i_3}b_{i_m} \\ b_{i_4}b_{i_2} & b_{i_4}b_{i_3} & 1+b_{i_4}b_{i_4} & \dots & b_{i_4}b_{i_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_2} & b_{i_m}b_{i_3} & b_{i_m}b_{i_4} & \dots & 1+b_{i_m}b_{i_m} \end{bmatrix} \\
&\quad +b_{i_2}b_{i_2}\det\begin{bmatrix} b_{i_1}b_{i_1} & b_{i_1}b_{i_3} & b_{i_1}b_{i_4} & \dots & b_{i_1}b_{i_m} \\ b_{i_3}b_{i_1} & 1+b_{i_3}b_{i_3} & b_{i_3}b_{i_4} & \dots & b_{i_3}b_{i_m} \\ b_{i_4}b_{i_1} & b_{i_4}b_{i_3} & 1+b_{i_4}b_{i_4} & \dots & b_{i_4}b_{i_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_3} & b_{i_m}b_{i_4} & \dots & 1+b_{i_m}b_{i_m} \end{bmatrix} + \dots \\
&\quad + \det\begin{bmatrix} b_{i_1}b_{i_1} & b_{i_1}b_{i_3} & b_{i_1}b_{i_4} & \dots & b_{i_1}b_{i_m} \\ b_{i_3}b_{i_1} & 1+b_{i_3}b_{i_3} & b_{i_3}b_{i_4} & \dots & b_{i_3}b_{i_m} \\ b_{i_4}b_{i_1} & b_{i_4}b_{i_3} & 1+b_{i_4}b_{i_4} & \dots & b_{i_4}b_{i_m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{i_m}b_{i_1} & b_{i_m}b_{i_3} & b_{i_m}b_{i_4} & \dots & 1+b_{i_m}b_{i_m} \end{bmatrix} \\
&= a^{(i_1)}_{(i_1,i_3,i_4,\dots i_m)} + a^{(i_1,i_2)}_{(i_1,i_2,\dots i_m)}
\end{aligned}$$

But it has already been shown that $a^{(i_1,i_2)}_{(i_1,i_2,\ldots i_m)} = 0$. Therefore:

$$a^{(i_1)}_{(i_1,i_2,i_3,\ldots i_m)} = a^{(i_1)}_{(i_1,i_3,i_4,\ldots i_m)}$$

Applying this rule recursively, it can be seen that:

$$\begin{aligned} a^{(i_1)}_{(i_1,i_2,\ldots i_m)} &= a^{(i_1)}_{(i_1,i_m)} \\ &= \det \begin{bmatrix} b_{i_1}b_{i_1} & b_{i_1}b_{i_m} \\ b_{i_m}b_{i_1} & 1 + b_{i_m}b_{i_m} \end{bmatrix} \\ &= b_{i_1}b_{i_1} \end{aligned}$$

And so:

$$a^{\emptyset}_{(i_1,i_2,\ldots i_m)} = a^{\emptyset}_{(i_2,i_3,\ldots i_m)} + b_{i_1}b_{i_1} \tag{E.1}$$

Now, by definition:

$$\det\left(\mathbf{I} + \mathbf{b}\mathbf{b}^T\right) = a^{\emptyset}_{(1,2,\ldots n)}$$

so, using (E.1) it may be seen that:

$$\begin{aligned} \det\left(\mathbf{I} + \mathbf{b}\mathbf{b}^T\right) &= a^{\emptyset}_{(1,2,\ldots n)} \\ &= a^{\emptyset}_{(2,3,\ldots n)} + b_1b_1 \\ &= a^{\emptyset}_{(3,4,\ldots n)} + b_1b_1 + b_2b_2 \\ &= \ldots \\ &= a^{\emptyset}_{(n)} + b_1b_1 + b_2b_2 + \ldots + b_{n-1}b_{n-1} \\ &= \det\left[\ 1 + b_nb_n\ \right] + b_1b_1 + b_2b_2 + \ldots + b_{n-1}b_{n-1} \\ &= 1 + b_1b_1 + b_2b_2 + \ldots + b_nb_n \\ &= 1 + \mathbf{b}^T\mathbf{b} \end{aligned}$$

which proves the theorem. □