TECHNISCHE
UNIVERSITÄT
DRESDEN

# Duality, Derivative-Based Training Methods and Hyperparameter Optimization for Support Vector Machines 

M.Sc. Nico Strasdat

## Dissertation

to achieve the academic degree
Doctor rerum naturalium (Dr. rer. nat.)
submitted to
the School of Science of
the Technische Universität Dresden

First referee
Prof. Dr. rer. nat. habil. Andreas Fischer
Second referee
Prof. Marco Sciandrone
Submitted on: June 13, 2023
Defended on: October 11, 2023

## Danksagung

Die hier vorliegende Doktorarbeit wäre wohl nie ohne die Hilfe meines wissenschaftlichen Betreuers Prof. Dr. Andreas Fischer entstanden. Ich danke inm herzlich für die Begleitung im Studium, die Weitergabe der grundlegenden mathematischen Kenntnisse in den Vorlesungen, die Ermöglichung der Arbeit in vielen interessanten Projekten und die Beratung bei der Erstellung der Dissertationsschrift.

Im vergleichbaren Grad wichtig auf dem Weg zur Anfertigung der Arbeit war Gerd Langensiepen. Ich möchte ihm besonders dafür danken, dass er mich auf die Thematik aufmerksam gemacht, mit mir erste wissenschaftliche Schritte gegangen ist und den Kontakt zu den Herren Thorsten Thies, Frank Weber und Klaus Luig von der Cognitec Systems GmbH aufgebaut hat. Letzteren bin ich dankbar für Kooperation in Projekten, die mein Wissen über die Theorie und Anwendung von SVMs wesentlich erweitert haben und meine anfängliche Arbeit an der TU Dresden finanzierten.

Ebenfalls durfte ich von den erfahreneren Kollegen am Institut lernen. Ich danke insbesondere Guntram Scheithauer für die Heranführung an Fragestellungen der diskreten Optimierung - die mich manche Zeit vom Promovieren abgehalten, aber sehr bereichert, haben - und Klaus Schönefeld für die gemeinsame Arbeit an Problemen der kontinuierlichen Optimierung. Außerdem durfte ich neben der Promotion einige Artikel zusammen mit John Martinovic erarbeiten und veröffentlichen, dem ich an dieser Stellen auch danken darf.

Neben der wissenschaftlichen Weiterentwicklung am Institut durfte ich auch einige mitmenschliche Höhepunkte gemeinsam mit den Kollegen genießen. Gedankt sei dafür besonders John Martinovic und Gunar Matthies, aber auch Torsten Buchwald, Isabella Käming, Sebastian Franz und Patrick Jaap (keine vollständige Aufzählung!).

Nicht zuletzt darf ich meiner Frau Claudia für die ständige Begleitung und Umsorgung in allen Lebenslagen danken. Zusammen mit unseren Kindern Theodor und Jonne hat sie die finale Phase meiner Promotion zu etwas ganz Besonderem gemacht.

## Contents

1 Introduction ..... 9
2 Basics of Support Vector Machines ..... 12
2.1 Common Problem Settings ..... 12
2.2 Basic Idea of Support Vector Machines ..... 13
2.2.1 Separable Case ..... 14
2.2.2 Non-Separable Case ..... 16
2.2.3 Duality ..... 17
2.2.4 Kernel Trick ..... 17
2.3 Other Variants of Support Vector Machines ..... 19
2.3.1 Kernel Functions ..... 19
2.3.2 Loss Functions ..... 20
2.3.3 Regularization ..... 20
2.4 A General Training Problem ..... 21
2.4.1 Definition of the General Training Problem ..... 21
2.4.2 Separable Loss Functionals ..... 23
2.4.3 Standard Decision Operator ..... 23
2.4.4 Standard Regularization Functional ..... 24
2.4.5 Standard Training Problem ..... 24
3 Application of Convex Duality Theory ..... 26
3.1 Dual Training Problem and Optimality Conditions ..... 27
3.2 Separable Loss Functionals ..... 28
3.3 Regularization with Standard Decisions ..... 30
3.4 Duality for the Standard Training Problem ..... 33
3.5 Finite-dimensional Representation of the Solution ..... 34
3.6 Interpretation of Optimality Conditions ..... 36
3.7 Notes on the Existence of Solutions ..... 38
3.8 Distance Estimation and Uniqueness of Solutions ..... 40
4 Examples of Classification and Regression Problems ..... 43
4.1 Binary Classification Problems ..... 43
4.1.1 Classical Support Vector Classification ..... 43
4.1.2 Maximal Margin Classifier ..... 45
4.1.3 Logistic Regression Support Vector Classification ..... 46
4.2 Regression Problems ..... 47
4.2.1 Least-Squares Support Vector Machine ..... 48
4.2.2 $\varepsilon$-Support Vector Regression ..... 48
4.2.3 Hard $\varepsilon$-Support Vector Regression ..... 49
4.3 Interpretation of Optimality Conditions Revisited ..... 50
4.3.1 ... for Support Vector Classification ..... 51
4.3.2 ... for $\varepsilon$-insensitive Support Vector Regression ..... 53
4.3.3 ... for Least-Squares Support Vector Machine ..... 54
4.4 Differentiable Approximations of Loss Functions ..... 54
4.4.1 A Construction Principle ..... 55
4.4.2 Examples of Smooth Approximation of the Maximum Function ..... 57
4.4.3 Convex Conjugate of Approximations ..... 58
4.5 Dual View on Smoothed Optimality Conditions ..... 61
5 Examples of Non-Standard Training Problems ..... 68
5.1 Multi-Class Classification Problems ..... 68
$5.2 \nu$-Support Vector Machines ..... 74
5.3 Subspace Support Vector Machines ..... 79
6 Reformulations of the Primal Training Problem ..... 83
6.1 First Finite-Dimensional Formulation ..... 83
6.2 Equivalent Formulation using a Bijection ..... 85
6.3 Second Finite-Dimensional Formulation ..... 87
6.4 Review of Dual Optimality Conditions ..... 89
6.5 Formulation as a System of Nonlinear Equations ..... 91
7 Sequential Minimal Optimization for the Dual Training Problem ..... 94
7.1 Basic Idea of Decomposition Methods ..... 94
7.2 Optimality Conditions Revisited ..... 95
7.3 Maximal Violating Pair and Directional Derivative ..... 98
7.4 A General Convergence Framework ..... 100
7.4.1 First-Order Optimality Measure ..... 100
7.4.2 Convergence under Sufficient Decrease Condition ..... 103
7.4.3 Exact Line Search ..... 104
7.4.4 Armijo-Type Step Size Selection ..... 105
7.5 Violating Pairs and First-Order Optimality ..... 107
7.6 Definition and Properties of a Bound-Aware Violation Measure ..... 110
7.7 Working Set Selection Rules ..... 114
7.8 Convergence of SMO ..... 116
7.9 Implementation and Complexity Analysis ..... 118
7.9.1 Discussion of Complexity ..... 118
7.9.2 Notes on the Convergence Speed ..... 121
7.10 Detection of Fixed Variables for Shrinking ..... 122
8 Optimization Methods for the Solution of the Primal Training Problem ..... 127
8.1 Basic Idea of Descent-Based Methods ..... 127
8.2 Simple Gradient Descent Method ..... 128
8.3 Preconditioned Gradient Descent Method ..... 130
8.3.1 General Idea and Derivation ..... 130
8.3.2 Practical Implementation ..... 131
8.3.3 Discussion of the Computational Effort ..... 132
8.3.4 Shrinking of the Training Problem ..... 134
8.3.5 Potential Application of Error Bound Conditions ..... 135
8.4 Application of Newton's Method ..... 136
8.4.1 Analysis of the Newton System ..... 136
8.4.2 Efficient Solution of the Newton System ..... 138
8.4.3 Formulation of Newton's Method ..... 140
8.4.4 Practical Remarks ..... 142
8.5 Properties of the Jacobian ..... 143
9 Hyperparameter Optimization for Support Vector Machines ..... 146
9.1 Formulation of a Bilevel Optimization Problem ..... 146
9.1.1 Summary of Approaches in the Literature ..... 147
9.1.2 Formulation of the General Idea ..... 148
9.1.3 Parameter-Dependent Standard Training Problem ..... 149
9.1.4 Formulation of the Upper Level Problem ..... 151
9.2 An Implicit Function Approach ..... 152
9.2.1 Application of the Implicit Function Theorem ..... 153
9.2.2 Sensitivity Analysis for Particular Training Samples ..... 154
9.2.3 Derivatives of the Validation Error Measure ..... 155
10 Computational Study ..... 157
10.1 Definition of Dataset and Training Problem ..... 157
10.2 Sequential Minimal Optimization ..... 159
10.2.1 Working Set Selection ..... 160
10.2.2 Comparison of Optimality Measures ..... 163
10.2.3 Behavior of Directional Derivative ..... 164
10.2.4 Relations between the Measures ..... 165
10.2.5 Parameter Dependency of Relations ..... 165
10.2.6 Observation for Problems with Smoothed Maximum Function ..... 167
10.2.7 Number of Support Vectors and Active Set ..... 169
10.2.8 Application of the Shrinking Approach ..... 170
10.2.9 Performance for Increasing Problem Size ..... 172
10.3 Methods for the Primal Training Problem ..... 174
10.3.1 Performance of First-Order Methods ..... 174
10.3.2 Empirical Evaluation of an Error Bound Condition ..... 175
10.3.3 Comparison of Newton's Method and SMO ..... 176
10.4 Practical Application of Hyperparameter Optimization ..... 178
10.4.1 Consideration of the Artificial Dataset ..... 178
10.4.2 Application to Real-World Regression Problems ..... 179
11 Summary and Outlook ..... 185
A Definitions and Basic Theory ..... 187
A. 1 Convexity and Convex Subdifferential ..... 187
A. 2 Convex Conjugates ..... 188
A. 3 Convex Duality ..... 189
A. 4 Bregman Distance ..... 191
A. 5 Strict-Smooth Duality ..... 192
Bibliography ..... 195

## 1 Introduction

The application of machine learning methods for real-world problems is omnipresent nowadays. In many practical settings large models, such as deep neural networks, are used, which lead to non-convex optimization problems with many parameters and hence are often difficult to handle both theoretically and practically. In contrast, methods derived from convex optimization problems usually possess a distinct theoretical foundation that enables an easier interpretation of the resulting models. This is mainly possible because the solution of a convex optimization problem can be characterized in a rather simple way. Moreover, convexity also helps to construct efficient solution methods, for which rather strong convergence results can be proven.

Support Vector Machines constitute a particular class of machine learning methods which is based on the consideration of convex optimization problems. Though there is a vast amount of works handling particular formulations of the problem in theory and practice, we aim to formalize the main structure into a general training problem for which theoretical investigations are still possible. This will be done in Chapter 2 after an introductory overview over some well-known examples is given.

In Chapter 3, methods from convex analysis are used to derive a dual formulation of the general training problem. This formulation is complemented by optimality conditions which are useful in the main part of the thesis and provide an additional insight into the properties of the solution of the training problem and the resulting decision function. In that context, conditions being sufficient for the existence and the partial uniqueness of the solution are provided.

The considered duality approach is applied to some particular classification and regression problems in Chapter 4 which yields a descriptive interpretation of the optimality conditions. Moreover, a smoothing approach for the primal training problem is proposed, and the consequence of dual smoothing for the primal training problem is analyzed. In Chapter 5 the duality theory is further used to derive dual formulations and optimality conditions for non-standard training problems, in particular, for multi-class SVMs, for a generalized version of $\nu$-SVMs, and for Subspace SVMs.

In Chapter 6, different finite-dimensional formulations of the primal training problem are introduced, and their relationships are discussed in detail. This preliminary work provides the necessary foundation for the application of practical solution methods to the primal training problem. Furthermore, an important connection to the solution of the dual problem is highlighted.

Both views on the training problem are then used to derive and analyze potential solution methods. The well-known and state-of-the-art Sequential Minimal Optimization (SMO) method is examined in a general formulation in Chapter 7. In the course of that chapter, a general convergence framework for methods solving convex optimization problems is proposed and used to prove convergence of the dual SMO method under certain rather non-restrictive assumptions. Following the basic findings, a theoretically founded approach for the detection of fixed variables in the dual training problem is given which makes it possible to reduce the computational effort in practice.

In Chapter 8, the application of derivative-based solution approaches for the primal training problem is considered. After two realizations of first-order methods are introduced and analyzed, it is argued that an implementation of Newton's method is both theoretically possible and practically efficient under certain conditions.

The previously derived properties of the primal training problem are also used afterwards in Chapter 9 for the derivation of a bilevel hyperparameter optimization approach. It is shown how the formulation of a training problem as system of nonlinear equations can be exploited by means of the implicit function theorem to reduce the bilevel structure of the hyperparameter optimization problem. This idea makes it possible to handle large problems efficiently in practice.

The practical methods of Chapters 7 to 9 are finally evaluated in a variety of computational tests in Chapter 10. In particular, the SMO method is empirically analyzed with respect to the newly introduced optimality measures, its dependence on hyperparameters, and the practical performance of the proposed detection method for fixed variables. Moreover, the first- and second-order methods for the solution of the primal training problem are evaluated practically. Finally, the derived hyperparameter optimization approach is applied to selected problems.

Note that the structure of the present thesis is also visualized in Figure 1.1.
Let us briefly summarize the used notation. Throughout the thesis, we follow the convention to use lowercase bold characters to denote vectors and uppercase bold characters to denote linear operator or matrices. If the underlying vector space is $\mathbb{R}^{n}$, the components of vector and matrices are denoted by lower indices which are appended to the corresponding plain characters instead of the bold ones. Upper indices are used to indicate elements of sequences mostly for vectors.

For a particular size $n \in \mathbb{N}$, the vectors $\mathbf{0} \in \mathbb{R}^{n}$ and $\mathbf{1} \in \mathbb{R}^{n}$ consist of zeros and ones, respectively. The $i$ th standard basis vector is denoted by $\boldsymbol{e}^{i} \in \mathbb{R}^{n}$. The identity matrix of size $n$ is denoted by $\boldsymbol{I} \in \mathbb{R}^{n \times n}$. Finally, the term $\boldsymbol{O}$ denotes a zero matrix of appropriate (not necessarily quadratic) size, which should become clear from the context.

In a general vector space the term $\mathbf{0}$ is also used to denote the zero vector. Whenever it is needed, $\langle\cdot, \cdot\rangle$ denotes the inner product associated with the particular inner product space at hand.


Figure 1.1: Overview of the structure of the work

## 2 Basics of Support Vector Machines

In this chapter, we give a brief introduction into the idea of Support Vector Machines. First, we start with the summary of different kinds of learning problems. Afterwards, some more specific topics associated with Support Vector Machines are discussed. In the last subsection, we formulate a general training problem which is the basis in the rest of the work. Additionally, some particular choices for the parts of the general training problem are considered.

### 2.1 Common Problem Settings

Of course, there is a broad range of possible applications of machine learning algorithms. Subsequently, we introduce a set of particular tasks which will be referred to in the course of this chapter.

Commonly, one differentiates machine learning tasks into two problem classes: supervised and unsupervised learning problems. In unsupervised learning problems the aim is to extract common patterns from a given dataset. The concrete meaning of the term pattern depends on the particular application. Typical examples of unsupervised learning problems are clustering problems, outlier detection and density estimation.

In contrast, for supervised learning problems a relationship between input and output values is to be modeled by means of a dataset which contains samples of inputoutput pairs. The two main examples of supervised learning problems are classification and regression problems.

As a particular example of an unsupervised learning problem, consider the problem of outlier detection. Here, the main task is to generate a model for the structure of a given unlabeled dataset

$$
\left\{\boldsymbol{x}_{1}, \ldots, \boldsymbol{x}_{n}\right\} \subseteq \mathcal{X}
$$

of $n$ data points. The model should be constructed such that it is able to predict whether a particular data point is similar (in a specific sense) to the points in the dataset or not. If the given data point does not fit well into the model, it is called an outlier. Of course, the measure of similarity depends on the application at hand. Note that the input set $\mathcal{X}$ is not expected to have any special structure a priori.

A simple approach to model similarity is to determine a set which encloses the given dataset. This set should be chosen in a way that it is as tight as possible (in a certain
sense) but does not overfit the dataset. One possibility to compute such a set is the search for the smallest enclosing hypersphere, see [STC04, Section 7.1]. A similar idea is also used in the Support Vector method for novelty detection, see [SWS ${ }^{+} 99$ ].

On the other hand, there are supervised learning problems. Probably the most frequently considered problem in supervised learning is the binary classification problem. In this problem setting, a dataset

$$
\left\{\left(\boldsymbol{x}_{1}, y_{1}\right), \ldots,\left(\boldsymbol{x}_{n}, y_{n}\right)\right\} \subseteq \mathcal{X} \times \mathcal{Y}
$$

with input values $\boldsymbol{x}_{i} \in \mathcal{X}$ and associated labels $y_{i} \in \mathcal{Y}$ is given. Each label can only have one of two predefined labels. Typically, the set of labels is chosen to be $\mathcal{Y}=\{-1,1\}$. The classification problem then lies in the construction of a function $f: \mathcal{X} \rightarrow \mathcal{Y}$ that captures the relation between the input and the associated output values of a given dataset and generalizes well to unseen data points. A particular approach to treat this problem by means of Support Vector Machines is described in detail in the next subsection. Note that binary classification problems are an important benchmark task for the present thesis.

Aside from binary classification problems, many applications result in classification problems in which more than one class label is possible for a specific input point. This directly leads to the notion of multi-class classification problems. Note that it is possible to treat such problems by means of a set of binary classification problem employing different techniques. Two popular approaches to reduce multi-class classification problems to binary classification problems are the strategies one-vs-rest and one-vsone. For more details on these transformations we refer to [Bis06, Subsection 4.1.2]. It should be noted that such a transformation implies a certain complexity of the resulting problem if the number of classes is high. In these cases, alternative methods can be more efficient. A different solution strategy will be discussed in Section 5.1.

Another important class of supervised learning problems are regression problems. The difference to classification problems is that the output values $y_{i} \in \mathcal{Y}$ for the given dataset are not discrete but real-valued. This means that $\mathcal{Y} \subseteq \mathbb{R}$.
Then, the aim of regression is to determine a function $f: \mathcal{X} \rightarrow \mathcal{Y}$ such that $f\left(\boldsymbol{x}_{i}\right)$ is close to $y_{i}$ for all $i \in\{1, \ldots, n\}$ and generalizes well for unseen data points. Of course, the measure of closeness depends on the application at hand. Regression problems form the second type of benchmark tasks of this thesis.

### 2.2 Basic Idea of Support Vector Machines

Before we start with an abstract definition of Support Vector Machines (SVMs) we want to explain the basic idea step by step with the help of simple examples. The following derivation is roughly based on [SS02, Sections 7.4 and 7.5] and [STC04, Section 7.2].

To start with, let us assume that we are given a data set $\left\{\left(\boldsymbol{x}_{1}, y_{1}\right), \ldots,\left(\boldsymbol{x}_{n}, y_{n}\right)\right\}$ consisting of real input vectors $\boldsymbol{x}_{i} \in \mathbb{R}^{d}$ and associated output values $y_{i} \in\{-1,+1\}$ for $i \in\{1, \ldots, n\}$. A simple visual example is given in Figure 2.2.1a.


Figure 2.2.1: Example classification problem

Our task is then to solve the binary classification problem, i.e., to find some function $f: \mathbb{R}^{n} \rightarrow \mathbb{R}$ such that the sign of $f\left(\boldsymbol{x}_{i}\right)$ is equal to $y_{i}$ for most of the data points. In Figure 2.2.1 a this task is represented by a data point (black circle) with unknown class label that should be inferred from the given dataset.

In order to derive the classical formulation of SVMs we proceed in two steps: First, we assume that the dataset is linearly separable, and we seek for a separating hyperplane with a certain property. Second, we consider the general case where no separating hyperplane exists, and we derive an alternative approach based on the first case. Afterwards, we discuss two other topics related to the classical SVM formulation.

### 2.2.1 Separable Case

If the data points are linearly separable, a reasonable approach would be to construct a hyperplane such that all points of the same class lie on the same side of the hyperplane. In general, there are many possibilities to choose separating hyperplanes which result in different prediction qualities, see Figure 2.2.1b. One can easily see that the predicted class label for the given data points depends on the choice of the hyperplane. Obviously, not every possible selection is equally good in this example.

The idea of SVMs is to construct a maximal margin classifier (see [Vap95]), i.e., a hyperplane that has a maximal distance to the points from each class. This problem is typically formulated as an optimization problem. However, there are several formulations for that problem in the literature which are equivalent under certain conditions. Subsequently, we formulate some of these optimization problems and discuss their basic properties and relationships.

For a fixed vector $\boldsymbol{w} \in \mathbb{R}^{d}$ and a number $b \in \mathbb{R}$ a hyperplane is given by the equation $\langle\boldsymbol{w}, \boldsymbol{x}\rangle+b=0$. The hyperplane separates the two classes if and only if

$$
\left(\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right) y_{i}>0 \quad \text { for all } \quad i \in\{1, \ldots, n\} .
$$

Subsequently, we assume that $\boldsymbol{w} \neq \mathbf{0}$ as otherwise the hyperplane cannot separate the classes in a non-trivial case. It is well known that the distance of some point $\boldsymbol{x}$ to
the hyperplane can be computed as $|\langle\boldsymbol{w}, \boldsymbol{x}\rangle+b| /\|\boldsymbol{w}\|$ if $\boldsymbol{w} \neq \mathbf{0}$. This means that the problem of finding a maximal margin classifier can be modeled as

$$
\begin{array}{ll}
\max _{\boldsymbol{w}, b} & \frac{1}{\|\boldsymbol{w}\|} \min _{i \in\{1, \ldots, n\}}\left|\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right|  \tag{2.2.1}\\
\text { s.t. } & \left(\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right) y_{i}>0 \quad \text { for all } \quad i \in\{1, \ldots, n\} .
\end{array}
$$

Note that this problem is only solvable if there is at least one point in each class. Otherwise, the objective function is unbounded.

Now, one can observe that $\left|\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right|=\left(\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right) y_{i}>0$ because $y_{i} \in\{-1,+1\}$. In order to replace the minimum in the objective function one can introduce an additional variable $\xi$ and use the relation

$$
\min _{i \in\{1, \ldots, n\}}\left|\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right|=\max \left\{\xi\left|\xi \leq\left|\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right| \quad \text { for all } \quad i \in\{1, \ldots, n\}\right\} .\right.
$$

This shows that the problem (2.2.1) is equivalent to

$$
\begin{equation*}
\max _{\boldsymbol{w}, b, \xi} \frac{\xi}{\|\boldsymbol{w}\|} \quad \text { s.t. } \quad\left(\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right) y_{i} \geq \xi \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{2.2.2}
\end{equation*}
$$

Note that it is not necessary to add the constraint $\xi>0$ explicitly to problem (2.2.2) because it will be satisfied at a solution of the problem if the classes are linearly separable. On the other hand, whenever the classes are not linearly separable, the optimal value of (2.2.2) is not positive. Nevertheless, any solution of (2.2.2) defines a hyperplane which could be used as classifier though it lacks the maximal margin property.
There are two issues with the formulation of problem (2.2.2) that we are going to discuss in the following. First, the objective function of the problem is non-convex. This means that theoretical and practical treatment can be tough. Second, it is possible to scale the variables by some positive factor without changing the value of the objective function and the constraints. In particular, for each solution of the problem there are infinitely many other solutions. That is because the representation of the hyperplane is not unique. To avoid this ambiguity one can normalize either one of the variables $\boldsymbol{w}, b$ or $\xi$. For instance, using the normalizing equation $\xi=1$, the problem (2.2.2) can be written as

$$
\begin{equation*}
\max _{\boldsymbol{w}, b} \frac{1}{\|\boldsymbol{w}\|} \quad \text { s.t. } \quad\left(\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right) y_{i} \geq 1 \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{2.2.3}
\end{equation*}
$$

Now, the maximization of the term $1 /\|\boldsymbol{w}\|$ is equivalent to the minimization of $\frac{1}{2}\|\boldsymbol{w}\|^{2}$. This gives rise to the convex optimization problem

$$
\begin{equation*}
\min _{\boldsymbol{w}, b} \frac{1}{2}\|\boldsymbol{w}\|^{2} \quad \text { s.t. } \quad\left(\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right) y_{i} \geq 1 \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{2.2.4}
\end{equation*}
$$

With the additional constraint $\xi=1$ the problem does not have a feasible point if the dataset is not linearly separable. This means that the problems (2.2.2) and (2.2.4) are not equivalent in this case. Depending on the application it may be more reasonable to at the constraint $\|\boldsymbol{w}\|=1$ to problem (2.2.2) instead.

In Figure 2.2.1c the optimal hyperplane (solid black) for the example dataset is presented. Additionally, there are two other hyperplanes (dashed blue and red) shown. These supporting hyperplanes arise directly from the maximization of the margin and are defined by the equations $\langle\boldsymbol{w}, \boldsymbol{x}\rangle+b=1$ and $\langle\boldsymbol{w}, \boldsymbol{x}\rangle+b=-1$, respectively. The points which are touched by the hyperplanes are commonly called support vectors (SVs) and play an important role in the study of SVMs. In the next subsection, we consider the modification of SVMs to the more practical case of non-separable datasets.

### 2.2.2 Non-Separable Case

In general, a given dataset will not be separable by a simple hyperplane. An example for this case is shown in Figure 2.2.2a.

(a) Given dataset and point to be classified

(b) Optimal hyperplane and Support Vectors

Figure 2.2.2: Example classification problem with dataset that is not linearly separable
If the dataset is not linearly separable, it is necessary to modify problem (2.2.2) as it does not correspond to maximizing the margin anymore. This is the case because it is not possible to satisfy the constraint $\xi>0$. Thus, one cannot expect to obtain a reasonable classifier from the solution of (2.2.3).

On the other hand, the problem (2.2.4) has no feasible point. A typical approach to treat this problem is to relax the constraint which ensures that a separating hyperplane is computed, see [CV95]. A measure for the violation of the constraint is then added to the objective function multiplied by a suitable factor $C>0$. This leads to the problem

$$
\begin{array}{ll}
\min _{\boldsymbol{w}, b, \boldsymbol{\xi}} & \frac{1}{2}\|\boldsymbol{w}\|^{2}+C \sum_{i=1}^{n} \xi_{i}  \tag{2.2.5}\\
\text { s.t. } & \left(\left\langle\boldsymbol{w}, \boldsymbol{x}_{i}\right\rangle+b\right) y_{i} \geq 1-\xi_{i}, \xi_{i} \geq 0 \quad \text { for all } \quad i \in\{1, \ldots, n\} .
\end{array}
$$

Now, the objective of the optimization problem is made of two conflicting goals: maximization of the margin (first term) and minimization of the misclassification (second term). This structure is very characteristic of SVMs.

Note that the term margin has a slightly different meaning in the context of problem (2.2.5). While there was an empty space between the two supporting hyperplanes in Figure 2.2.1c, now some data points lie in between these hyperplanes. This means
that the set of support vectors now consists of all points between the supporting hyperplanes. The solution of problem (2.2.5) for the example dataset (and a particular choice of the parameter $C$ ) is shown in Figure 2.2.2b. There, also the supporting hyperplanes and the support vectors are highlighted.

### 2.2.3 Duality

Observe that problem (2.2.5) is a convex quadratic optimization problem. Hence, it can be solved quite efficiently by means of standard software. However, if the input dimension of the dataset is large, it can be helpful to consider the dual optimization problem instead. This is a very common approach in the literature as it provides several advantages.

We do not want to provide a detailed derivation of the dual problem here because the usual way using Lagrange duality is quite elaborate. Instead, in Chapter 4, we follow a more general approach to obtain the dual problem using Fenchel's duality theory. We refer to [SS02, Section 7.5] for the classical derivation of the dual problem. The dual problem associated with (2.2.5) is given by

$$
\begin{array}{cl}
\min _{\boldsymbol{\alpha}} & \frac{1}{2} \boldsymbol{\alpha}^{\top} \boldsymbol{Q} \boldsymbol{\alpha}-\mathbf{1}^{\top} \boldsymbol{\alpha} \\
\text { s.t. } & \boldsymbol{y}^{\top} \boldsymbol{\alpha}=0  \tag{2.2.6}\\
& 0 \leq \alpha_{i} \leq C \text { for all } i \in\{1, \ldots, n\},
\end{array}
$$

where the matrix $\boldsymbol{Q} \in \mathbb{R}^{n \times n}$ is defined by $Q_{i j}:=y_{i} y_{j}\left\langle\boldsymbol{x}_{i}, \boldsymbol{x}_{j}\right\rangle$ for $i, j \in\{1, \ldots, n\}$.
In the same way as its primal problem (2.2.5), the problem (2.2.6) has a convex quadratic objective function and only linear constraints on the variables. Hence, it can also be solved by means of conventional optimization software. However, some very efficient methods were constructed which exploit the simple structure with only box constraints and a single linear equation. In particular, the Sequential Minimal Optimization method proposed by Platt [Pla99] or some of its extension is used very often due to its efficiency and simplicity.

### 2.2.4 Kernel Trick

In general, one cannot expect that a hyperplane is suitable to build a classifier because there is usually no simple linear relationship in the dataset. This is in particular important if the input data does not possess a linear structure at all. In these cases it is helpful to introduce the notion of feature maps. Suppose that the given input points $\boldsymbol{x}_{i}$ are taken from some input space $\mathcal{X}$ without special structure and let $\varphi: \mathcal{X} \rightarrow \mathcal{F}$ be a function which maps elements of the input space into a particular inner product space $\mathcal{F}$. Commonly, the function $\boldsymbol{\varphi}$ is called feature map and the vectors $\boldsymbol{\varphi}\left(\boldsymbol{x}_{i}\right)$ are called feature vectors.

Using these definitions, the problems (2.2.5) and (2.2.6) can be considered with $\boldsymbol{x}_{i}$ replaced by $\boldsymbol{\varphi}\left(\boldsymbol{x}_{i}\right)$. After that transformation, problem (2.2.5) determines a hyperplane in the feature space $\mathcal{F}$ and the classification is done by evaluating the function

$$
f: \mathcal{X} \rightarrow\{-1,1\}, \quad \boldsymbol{x} \mapsto \operatorname{sign}(\langle\boldsymbol{w}, \boldsymbol{\varphi}(\boldsymbol{x})\rangle+b)
$$

for any given point $\boldsymbol{x} \in \mathcal{X}$.
The dimension of the variables $(\boldsymbol{w}, b, \boldsymbol{\xi})$ in the primal problem (2.2.5) will be $\operatorname{dim} \mathcal{F}+$ $1+n$ instead of $d+1+n$, whereas the number of variables in the dual problem (2.2.6) does not change. Moreover, the entries in the matrix $\boldsymbol{Q}$ in problem (2.2.6) depend only on inner products between features vectors. This important observation leads to the idea of replacing the inner products by values of a so-called kernel function. Let a kernel function $\kappa: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be defined by $\kappa\left(\boldsymbol{x}, \boldsymbol{x}^{\prime}\right):=\left\langle\boldsymbol{\varphi}(\boldsymbol{x}), \boldsymbol{\varphi}\left(\boldsymbol{x}^{\prime}\right)\right\rangle$ for all $\boldsymbol{x}, \boldsymbol{x}^{\prime} \in \mathcal{X}$. Then, the matrix $\boldsymbol{Q}$ can be computed by means of $Q_{i j}=y_{i} y_{j} \kappa\left(\boldsymbol{x}_{i}, \boldsymbol{x}_{j}\right)$ for all $i, j \in\{1, \ldots, n\}$. Note that it is not necessary to have an explicit formulation of the feature map if a procedure to evaluate the kernel function is available.

This approach of introducing the feature map and using a kernel function to obtain a nonlinear classification function is commonly called kernel trick. It is quite powerful if the feature map is defined in a way that the transformed data is linearly separable.

An example is shown in Figure 2.2.3. There, two datasets which are not linearly separable are shown on the left-hand side. On the right-hand side the images of the datasets under specially chosen feature maps are presented. In these simple examples it is obvious that there exists a separating hyperplane in the feature space.


Figure 2.2.3: Examples of polynomial feature maps and separating hyperplane in the feature space

Note that, the kernel trick is not only applicable in the dual problem. As we will observe in Section 3.5, it is also possible to replace the feature map in the primal problem under non-restrictive assumptions.

### 2.3 Other Variants of Support Vector Machines

Support Vector Machines are a means to treat various problems in machine learning. The original idea is due to Vapnik (see [Vap63, Vap06]), where he introduced an algorithm for the construction of an optimal separating hyperplane. Later on, in [BGV92] the notion of optimal margin classifiers was introduced. The optimization problem proposed in that paper is very close to that of the usual formulation for SVMs. A similar idea was developed by Mangasarian, see [Man65, Man68]. Later on, the approach of separating hyperplanes was extended to regression problems in [Vap95, VGS96]. The latter approach is sometimes called Support Vector Regression (SVR) in the literature. A vast amount of different ideas extending SVMs to other settings were proposed. Now, it is merely impossible to give a comprehensive overview of the field of SVMs. However, a reasonable introduction to kernel methods and SVMs is given in the book [STC04]. There, different aspects of kernel methods are summarized.

If one wants to apply SVMs to some practical learning problem, there are several building blocks to choose in a suitable way. In the following, we describe each of these blocks briefly. A sketch of the relevant parts is presented in Figure 2.3.1 below.


Figure 2.3.1: Overview of a practical supervised learning problem

### 2.3.1 Kernel Functions

The learning problem typically starts with the definition of the input and the output data format. An important step in the application of kernel methods is the definition of kernel function which measures the similarity of two data points. In some special cases
(in particular if the input data consists of high-dimensional real vectors) it is sufficient to consider linear functions. This assumption led to the development of specialized training method, see, for instance, $\left[\mathrm{FCH}^{+} 08\right]$. However, in many cases the input data is not suitable for this approach since the set of input data has no vector space structure in general. Several examples for kernel functions for images, graphs and text data are given in [STCO4]. Beyond that, there is a vast amount of research articles concerning the definition of custom kernel functions for special applications.

### 2.3.2 Loss Functions

Besides the choice of a kernel function which is related to the input data, the definition of a suitable loss function is necessary. This is closely related to the format of the considered output data. Obviously, the choice of the loss function depends on the decision problem at hand. For example, the output of the decision function for a binary classification problem is a number whose sign predicts the class of the input point. A typical choice for a loss function in this case is the so-called hinge loss introduced in [BGV92]. But also the squared error loss function can be employed, which is used as a basic approach, for instance, in [SV99]. On the other hand, in regression problems the value of the decision function is directly related to the target value of an input point. Here, typical choices for loss functions are the $\varepsilon$-insensitive loss function, see [Vap98], and the squared error loss function, see [SGV98, SVGDB+02]. Some other loss functions are proposed, for instance, in [SSM98]. In a general setting, the loss function has to measure the degree of discrepancy between two output values depending on the particular learning problem at hand.

### 2.3.3 Regularization

A third important part in the definition of a training problem is the setting of a regularization functional. In most practical applications, it is necessary to not only minimize a loss function as this would lead to severe overfitting to the given dataset. In order to improve generalization of the resulting decision function, usually a weighted sum of loss functions (error term) and a suitable regularization term is minimized. We have already seen an example for such a problem in (2.2.5). Certainly, coming from the idea of margin-maximization as described in the previous section, the typical choice for a regularization functional is the term $\frac{1}{2}\|\boldsymbol{w}\|^{2}$. This choice is also plausible for regression functions of the form

$$
\boldsymbol{x} \mapsto\langle\boldsymbol{w}, \boldsymbol{\varphi}(\boldsymbol{x})\rangle+b
$$

because the term $\|\boldsymbol{w}\|^{2}$ can be interpreted as a measure for the non-flatness or nonsmoothness (see [SS02, Section 4.3]) of the regression function which should not be too high.

Of course, it is also possible to use another norm in the regularization term which is chosen based on some prior knowledge about the learning problem. This is especially interesting if a linear feature map is considered. Then, it may be practical to add some term measuring the non-sparseness of the vector $\boldsymbol{w}$. In particular, for linear feature
maps, enforcing a certain sparseness of $\boldsymbol{w}$ implies a selection of features. The most prominent choice to obtain this is $\|\boldsymbol{w}\|_{0}$, the so-called zero-norm, which counts the number of non-zero entries in the vector $\boldsymbol{w}$, cf. [WEST03]. Note that the term $\|\boldsymbol{w}\|_{0}$ is not a norm in the mathematical sense. Moreover, it is not convex such that solving the training problem with zero-norm regularization practically is hard in general. To circumvent this problem, one often uses a convex approximation for the zero-norm in practice. A prevalent example for this is the one-norm, see [BM98, FM04].

Instead of introducing a penalty on the norm of $\boldsymbol{w}$ it is also possible to add a regularization term which forces the variables of the learning problem to lay in a certain subset or subspace. We elaborate on this idea briefly in Section 5.3.

Note that the weights associated with each term in the objective function of the training problem need to be chosen in a suitable way. This problem will be further discussed in the context of hyperparameter optimization in Chapter 9.

### 2.4 A General Training Problem

Now that we have discussed some examples for modeling a particular training problem, our aim is to unify the resulting problems into a common structure. In particular, we want to see different unsupervised and supervised training problems as special cases of one abstract optimization problem. As we derive properties of the training problem later on, we will state the problem in a rather general structure here. Once the theory is established, the results can be applied to special training problems easily.

### 2.4.1 Definition of the General Training Problem

In Section 2.2, we introduced the classical formulation of an SVM for classification. In the following, we generalize the framework to a more general training problem which is not necessarily a classification problem. By training problem we mean the problem of finding a model function $f: \mathcal{X} \rightarrow \mathcal{Y}$ which generates an output value $y \in \mathcal{Y}$ for any given input point $\boldsymbol{x} \in \mathcal{X}$ such that an unknown relation is approximated.

Some prototypical examples for this kind of problem include classification and regression problems as presented in Section 2.1. Recall that for binary classification problems $\mathcal{Y}:=\{-1,1\}$ and for regression problems $\mathcal{Y}:=\mathbb{R}$, whereas for multi-class classification problems, the set $\mathcal{Y}$ is some discrete set. Moreover, unsupervised training problems like density estimation problems can also be seen as examples. For those problems, the output value is equal to the density, i.e., $\mathcal{Y}:=\mathbb{R}_{+}$.

All these kinds of problems can be described in a common form. To see this, we consider again the example classification problem in (2.2.5). There, we saw that the training problem can be formulated as an optimization problem and that the model function has a particular form. The objective function of the problem is composed of two parts: one part quantifying the closeness of the model function to the given dataset and another one measuring the regularity of the model function. Subsequently, those
parts are described by two functionals $\mathcal{L}$ and $\mathcal{R}$. A general training problem can be stated as follows.

Definition 2.4.1 (general training problem)
Let $\mathcal{H}$ and $\mathcal{D}$ be Banach spaces. Suppose that we are given

- a regularization functional $\mathcal{R}: \mathcal{H} \rightarrow \mathbb{R} \cup\{+\infty\}$,
- a linear operator $\boldsymbol{T}: \mathcal{H} \rightarrow \mathcal{D}$ and
- a loss functional $\mathcal{L}: \mathcal{D} \rightarrow \mathbb{R} \cup\{+\infty\}$.

Then, we say that an optimization problem of the form

$$
\begin{equation*}
\min _{\boldsymbol{\omega} \in \mathcal{H}} \psi_{\mathrm{p}}(\boldsymbol{\omega}) \tag{2.4.1}
\end{equation*}
$$

with the objective function

$$
\begin{aligned}
\psi_{\mathrm{p}}: \mathcal{H} & \rightarrow \mathbb{R} \cup\{+\infty\}, \\
\boldsymbol{\omega} & \mapsto \psi_{\mathrm{p}}(\boldsymbol{\omega}):=\mathcal{R}(\boldsymbol{\omega})+\mathcal{L}(\boldsymbol{T} \boldsymbol{\omega})
\end{aligned}
$$

is a general training problem.

In the setting of a general training problem, the variable vector $\boldsymbol{\omega}$ defines the model function $f$ which we are trying to compute. The regularization functional measures the degree of regularity of the corresponding model function. The linear operator $\boldsymbol{T}$ models the decisions associated with each input point and the loss functional measures the loss which is associated with the decisions depending on the choice of the vector $\boldsymbol{\omega}$.

In the abstract definition of the general training problem the space $\mathcal{D}$ has no predefined meaning. Depending on the application at hand, different choices may be suitable. For supervised learning problem which we have in mind, the decisions are usually one real number for each element of the dataset, i.e., the space $\mathcal{D}$ is equal to $\mathbb{R}^{n}$, where $n$ is the number of training samples. In this case, the vector $\boldsymbol{t}:=\boldsymbol{T} \boldsymbol{\omega}$ consists of entries $t_{i}=[\boldsymbol{T} \boldsymbol{\omega}]_{i}$ for $i \in\{1, \ldots, n\}$, and each of the entries can be seen as a decision for the $i$ th training sample.

Note that there can also be a difference between the decision $[\boldsymbol{T} \boldsymbol{\omega}]_{i}$ and the value of the model function $f\left(\boldsymbol{x}_{i}\right)$ as post-processing might be necessary. For example, for binary classification problems, the decisions are typically real values (i.e., $\mathcal{D}=\mathbb{R}^{n}$ ), whereas the decision itself is binary. Then, the model function could be defined as $f\left(\boldsymbol{x}_{i}\right)=\operatorname{sign}\left([\boldsymbol{T} \boldsymbol{\omega}]_{i}\right)$.

In the course of the discussion we will assume additional properties on the structure of the spaces $\mathcal{H}$ and $\mathcal{D}$ when it is necessary. For instance, with the special problem (2.2.5) in mind, one can think of $\mathcal{H}$ to be the product space of the feature space $\mathcal{F}$ and $\mathbb{R}$ such that $\boldsymbol{\omega}=(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$. Furthermore, for the formulation of some other problems it is convenient to introduce additional variables. In this case the space $\mathcal{H}$ can be a product space of even more factors.

### 2.4.2 Separable Loss Functionals

The general training problem (2.4.1) captures a wide range of possible applications. Subsequently, we summarize only a small subset of problems and mention other applications and formulations if it is convenient. Particularly, we often focus on separable loss functionals as follows.

Definition 2.4.2 (separable loss functional and loss functions)
Let $\left(\mathcal{D}_{i}\right)_{i=1}^{n}$ be a sequence of linear spaces and let $\mathcal{D}:=\mathcal{D}_{1} \times \cdots \times \mathcal{D}_{n}$ be the associated product space. Suppose that $\ell_{i}: \mathcal{D}_{i} \rightarrow \mathbb{R} \cup\{+\infty\}$ is convex and lower semi-continuous for $i \in\{1, \ldots, n\}$.
The loss functional

$$
\begin{aligned}
\mathcal{L}: \mathcal{D}_{1} \times \cdots \times \mathcal{D}_{n} & \rightarrow \mathbb{R} \cup\{+\infty\} \\
\left(\boldsymbol{t}_{1}, \ldots, \boldsymbol{t}_{n}\right) & \mapsto \sum_{i=1}^{n} \ell_{i}\left(\boldsymbol{t}_{i}\right)
\end{aligned}
$$

is called a separable loss functional. The functions $\ell_{i}$ for $i \in\{1, \ldots, n\}$ are called loss functions.

Note that a separable loss functional is convex and lower semi-continuous by definition because the loss functions are assumed to be convex and lower semi-continuous. This means that they are indeed suitable for the definition of a general training problem.

Separable loss functionals are present in most of the supervised and unsupervised learning problems. Moreover, in many applications the spaces $\mathcal{D}_{i}$ are chosen to be the same space. In this case, we can write $\mathcal{D}_{i}:=\mathcal{D}_{0}$ for $i \in\{1, \ldots, n\}$. Particularly, for supervised learning problems the loss functions typically have the form

$$
\ell_{i}(\boldsymbol{t})=\ell\left(\boldsymbol{t}, y_{i}\right) \quad \text { for all } \quad i \in\{1, \ldots, n\},
$$

where $\ell: \mathcal{D}_{0} \times \mathcal{Y} \rightarrow \mathbb{R} \cup\{+\infty\}$ is some generic loss function which depends on the output value.

### 2.4.3 Standard Decision Operator

In the following, we introduce a special case in which the decision operator is also defined in a separable way. The basic idea is to determine a common structure for that the application of separable loss function makes sense. This idea leads to the following definition of a standard decision operator.

Definition 2.4.3 (standard decision operator)
Suppose that $\mathcal{F}$ is a real Hilbert space. Let $\mathcal{H}=\mathcal{F} \times \mathbb{R}$ and $\mathcal{D}=\mathbb{R}^{n}$. Suppose that
feature vectors $\boldsymbol{\varphi}_{i} \in \mathcal{F}$ are given for $i \in\{1, \ldots, n\}$. The linear operator defined by

$$
\begin{aligned}
\boldsymbol{T}: \mathcal{F} \times \mathbb{R} & \rightarrow \mathbb{R}^{n} \\
\quad(\boldsymbol{w}, b) & \mapsto\left[\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right]_{i=1}^{n}
\end{aligned}
$$

is called standard decision operator. The space $\mathcal{F}$ is called feature space. Moreover, the function

$$
\begin{aligned}
d: \mathcal{F} & \rightarrow \mathbb{R}, \\
\boldsymbol{\varphi} & \mapsto\langle\boldsymbol{w}, \boldsymbol{\varphi}\rangle+b
\end{aligned}
$$

is called decision function.
For instance, this setting is present if a feature map $\varphi: \mathcal{X} \rightarrow \mathcal{H}$ is given and input data points $\boldsymbol{x}_{i}$ are mapped to $\boldsymbol{\varphi}_{i}:=\boldsymbol{\varphi}\left(\boldsymbol{x}_{i}\right)$ for $i \in\{1, \ldots, n\}$. Then, we consider real-valued decisions of the form

$$
t_{i}:=[\boldsymbol{T} \boldsymbol{\omega}]_{i}=d\left(\boldsymbol{\varphi}\left(\boldsymbol{x}_{i}\right)\right)=\left\langle\boldsymbol{w}, \boldsymbol{\varphi}\left(\boldsymbol{x}_{i}\right)\right\rangle+b \quad \text { for } \quad i \in\{1, \ldots, n\} .
$$

### 2.4.4 Standard Regularization Functional

As we have seen in Section 2.2 (cf. problems (2.2.4) and (2.2.5), in particular) the margin maximization in SVMs is equivalent to the minimization of the term $\frac{1}{2}\|\boldsymbol{w}\|^{2}$. Because this is a common choice for the regularization functional, we summarize it in the following definition.

Definition 2.4.4 (standard regularization functional)
Suppose that $\mathcal{F}$ and $\mathcal{F}_{0}$ are real Hilbert spaces. Let $\mathcal{H}:=\mathcal{F} \times \mathcal{F}_{0}$ and let $\lambda>0$. Then, the functional

$$
\begin{align*}
\mathcal{R}_{\lambda}: \mathcal{F} \times \mathcal{F}_{0} & \rightarrow \mathbb{R} \\
\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right) & \mapsto \frac{\lambda}{2}\|\boldsymbol{w}\|^{2} \tag{2.4.2}
\end{align*}
$$

is called standard regularization functional. The coefficient $\lambda$ is called regularization parameter.

The presence of the regularization parameter in the definition of the regularization functional makes it possible to model some training problems in a more descriptive way. On contrast, the usual approach is based on appending a weighting coefficient $C$ to the loss term, which is indeed equivalent for $C=\lambda^{-1}$. However, the scaling of the loss functional leads to another problem formulation in which the interpretation of optimality conditions seems a bit less natural.

### 2.4.5 Standard Training Problem

In general, it is possible to define the loss and regularization functional independently of each other. However, a common approach in applications is to combine the two
special choices presented above. This idea leads to a standard training problem as defined subsequently.

Definition 2.4.5 (standard training problem)
Consider the training problem (2.4.1). Suppose that

- $\mathcal{H}=\mathcal{F} \times \mathbb{R}$,
- $\mathcal{L}$ is a separable loss functional (see Definition 2.4.2),
- $\boldsymbol{T}$ is the standard decision operator (see Definition 2.4.3), and,
- $\mathcal{R}=\mathcal{R}_{\lambda}$ is the standard regularization functional with $\lambda>0$ (see Definition 2.4.4).

Then, problem (2.4.1) is called standard training problem. In this case, the problem has the form

$$
\begin{equation*}
\min _{\substack{\boldsymbol{w} \in \mathcal{F} \\ b \in \mathbb{R}}} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right) . \tag{2.4.3}
\end{equation*}
$$

## 3 Application of Convex Duality Theory

The aim of this chapter is to employ convex duality theory to deduce characteristic properties of the training problem and its solution. In order to do this, we derive a dual training problem that is associated with the optimization problem (2.4.1). To emphasize the notion of duality, the original training problem (2.4.1) is called primal training problem below.

For reference, the basic notations and results of duality theory are briefly summarized in Appendix A. Note that the application of Fenchel's duality theory to particular training problems is not a novelty. For instance, in [RL07] a comparable approach is used to derive the dual problem for a training problem with a more specific structure and several practical applications of the result are presented. Moreover, [BH14] also use particular formulations of the primal training problem to investigate the corresponding dual problem. As a final example, we note that the authors of [GU17] show that a certain interpretation of the resulting problems is possible for particular choices of the regularization and loss terms.

The proposed framework is indeed more general than what is known from the literature. It is not only applicable to well-known problems, as summarized in Chapter 4, but also to a broader class of practical problems, some of which are considered in Chapter 5.

Subsequently, we apply Fenchel's duality theory to the general training problem in order to obtain a formulation of the dual problem and corresponding optimality conditions (Section 3.1). The general form is gradually specialized to separable loss functionals (Section 3.2), to the standard decision operator (Section 3.3), and to the standard training problem (Section 3.4). As a straightforward consequence of the optimality conditions we summarize a formulation of the Representer theorem (Section 3.5), the notion of support vectors, and particular dual optimality conditions (Section 3.6). Finally, we explore the existence (Section 3.7) and uniqueness (Section 3.8) of solutions in more detail.

### 3.1 Dual Training Problem and Optimality Conditions

By construction, the definition of the training problem (2.4.1) has the form of the primal optimization problem given by (A.3.2). As a basis for building the connection between those two problems, we identify $f=\mathcal{R}, g=\mathcal{L}$, and, $\boldsymbol{A}=\boldsymbol{T}$. In order to focus the investigation later on and to apply the convex duality theory, we restrict ourselves to convex training problems. The following assumption summarizes the problem setting which we want to consider subsequently.

Assumption 3.1.1 (convex training problem)
Suppose that a training problem as in Definition 2.4.1 is given. We assume that

- the regularization functional $\mathcal{R}$ is convex and lower semi-continuous,
- the loss functional $\mathcal{L}$ is convex and lower semi-continuous, and,
- the operator $\boldsymbol{T}$ is bounded.

Moreover, we assume a certain regularity condition which is satisfied by virtually all practical problems. This assumption is needed in order to satisfy the conditions of the basic duality theorem stated in Theorem A.3.2.

Assumption 3.1.2 (regularity of training problem)
We assume that there exists a vector $\boldsymbol{\omega} \in \operatorname{dom}(\mathcal{R})$ such that $\mathcal{L}$ is continuous at $\boldsymbol{T} \boldsymbol{\omega}$.
This assumption is not too restrictive because the regularization functional and the loss functional are usually finite-valued in most applications. If we consider a loss functional which can attain infinite values, the assumption forces that the combination of the regularization functional and the loss functional does not lead to difficulties. For instance, it is satisfied if there exists a point which is strictly feasible in the sense that $\mathcal{R} \boldsymbol{\omega}$ is finite and $\boldsymbol{T} \boldsymbol{\omega}$ lies in the interior of $\operatorname{dom}(\mathcal{L})$. Note that this condition is strongly related to the Slater condition, see [BL10].

With these preliminaries, we reconsider the training problem in (2.4.1). Then, we can apply Theorem A.3.2 to obtain the dual training problem as follows.

Corollary 3.1.3 (duality for the general training problem)
Suppose that Assumption 3.1.1 and Assumption 3.1.2 are satisfied.
Then, the dual training problem associated with (2.4.1) is

$$
\begin{equation*}
\min _{\boldsymbol{\alpha} \in \mathcal{D}^{\star}} \psi_{\mathrm{d}}(\boldsymbol{\alpha}) \tag{3.1.1}
\end{equation*}
$$

with the dual objective function defined by

$$
\begin{aligned}
\psi_{\mathrm{d}}: & \mathcal{D}^{\star} \rightarrow \mathbb{R} \cup\{+\infty\} \\
& \boldsymbol{\alpha} \mapsto \psi_{\mathrm{d}}(\boldsymbol{\alpha}):=\mathcal{R}^{\star}\left(\boldsymbol{T}^{\star} \boldsymbol{\alpha}\right)+\mathcal{L}^{\star}(-\boldsymbol{\alpha}) .
\end{aligned}
$$

In particular,
(a) the dual training problem (3.1.1) has a solution, and
(b) the inequality

$$
\begin{equation*}
\Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \boldsymbol{\alpha}):=\psi_{\mathrm{p}}(\boldsymbol{\omega})+\psi_{\mathrm{d}}(\boldsymbol{\alpha}) \geq 0 \tag{3.1.2}
\end{equation*}
$$

is satisfied for all $\boldsymbol{\omega} \in \mathcal{H}$ and $\boldsymbol{\alpha} \in \mathcal{D}^{\star}$ with equality if and only if $\boldsymbol{\omega}$ and $\boldsymbol{\alpha}$ are solutions of the primal and dual training problem, respectively.

Proof. We apply Theorem A.3.1 and Theorem A.3.2 with $f:=\mathcal{R}, g:=\mathcal{L}$, and, $\boldsymbol{A}:=\boldsymbol{T}$. In the convex analysis literature, the dual problem is usually stated as a maximization problem. In order to get a convex minimization optimization problem in the dual, we swap the sign of the objective here. This implies that the weak duality inequality (A.3.1) has also to be rewritten appropriately.

Finally, let us state the optimality conditions from Theorem A.3.2 in terms of the general training problem. Those conditions will be used extensively to state optimality conditions for particular training problems subsequently.

Theorem 3.1.4 (optimality conditions for the general training problem)
Suppose that Assumption 3.1.1 and Assumption 3.1.2 are satisfied.
Then, a vector $\boldsymbol{\omega} \in \mathcal{H}$ is a solution of the primal training problem (2.4.1) if and only if there exists $\boldsymbol{\alpha} \in \mathcal{D}^{\star}$ such that

$$
\begin{equation*}
\boldsymbol{T}^{\star} \boldsymbol{\alpha} \in \partial \mathcal{R}(\boldsymbol{\omega}) \quad \text { and } \quad-\boldsymbol{\alpha} \in \partial \mathcal{L}(\boldsymbol{T} \boldsymbol{\omega}) . \tag{3.1.3}
\end{equation*}
$$

In this case, $\boldsymbol{\alpha}$ is a solution of the dual training problem (3.1.1). Moreover, these conditions are equivalent to

$$
\boldsymbol{\omega} \in \partial \mathcal{R}^{\star}\left(\boldsymbol{T}^{\star} \boldsymbol{\alpha}\right) \quad \text { and } \quad \boldsymbol{T} \boldsymbol{\omega} \in \partial \mathcal{L}^{\star}(-\boldsymbol{\alpha})
$$

In the following subsections our aim is to compute the terms, that are used in the definition of the dual problem and the corresponding optimality conditions, for particular special cases.

### 3.2 Separable Loss Functionals

Since most of the training problems are stated using separable loss functionals, we compute the subdifferential and the convex conjugate for this particular class of functionals first.

Proposition 3.2.1 (subdifferential and conjugate of separable loss functionals) Let $\mathcal{L}$ be a separable loss functional as in Definition 2.4.2.
Then, its subdifferential is given by

$$
\begin{equation*}
\partial \mathcal{L}\left(\boldsymbol{t}_{1}, \ldots, \boldsymbol{t}_{n}\right)=\partial \ell_{1}\left(\boldsymbol{t}_{1}\right) \times \cdots \times \partial \ell_{n}\left(\boldsymbol{t}_{n}\right) . \tag{3.2.1}
\end{equation*}
$$

If the loss functions $\ell_{i}$ for $i \in\{1, \ldots, n\}$ are proper, the convex conjugate has the form

$$
\begin{equation*}
\mathcal{L}^{\star}(\boldsymbol{\alpha})=\sum_{i=1}^{n} \ell_{i}^{\star}\left(\boldsymbol{\alpha}_{i}\right) \tag{3.2.2}
\end{equation*}
$$

for all $\boldsymbol{\alpha} \in \mathcal{D}^{\star}$. Moreover, the subdifferential of $\mathcal{L}^{\star}$ is given by

$$
\begin{equation*}
\partial \mathcal{L}^{\star}(\boldsymbol{\alpha})=\partial \ell_{1}^{\star}\left(\boldsymbol{\alpha}_{1}\right) \times \cdots \times \partial \ell_{n}^{\star}\left(\boldsymbol{\alpha}_{n}\right) \tag{3.2.3}
\end{equation*}
$$

Proof. Let $\boldsymbol{t}=\left(\boldsymbol{t}_{1}, \ldots, \boldsymbol{t}_{n}\right) \in \mathcal{D}$. If $\boldsymbol{t} \notin \operatorname{dom}(\mathcal{L})$, there exists some index $i \in\{1, \ldots, n\}$ such that $\boldsymbol{t}_{i} \notin \operatorname{dom}\left(\ell_{i}\right)$. This means that both sides of the equation (3.2.1) are equal to the empty set.

Now, suppose that $\boldsymbol{t} \in \operatorname{dom}(\mathcal{L})$. Then, $\boldsymbol{t}_{i} \in \operatorname{dom}\left(\ell_{i}\right)$ for all $i \in\{1, \ldots, n\}$. Since the space $\mathcal{D}$ is the product of the spaces $\mathcal{D}_{1}, \ldots, \mathcal{D}_{n}$, its dual space can be represented as the product of the dual spaces of the factors, i.e., $\mathcal{D}^{\star}=\mathcal{D}_{1}^{\star} \times \cdots \times \mathcal{D}_{n}^{\star}$. By means of the sum rule for the convex subdifferential it follows that

$$
\partial \mathcal{L}(\boldsymbol{t})=\partial \sum_{i=1}^{n} \ell_{i}\left(\boldsymbol{t}_{i}\right)=\sum_{i=1}^{n} \partial_{t} \ell_{i}\left(\boldsymbol{t}_{i}\right),
$$

where $\partial_{\boldsymbol{t}}$ denotes that the subdifferential is taken with respect to the variable $\boldsymbol{t}$ (instead of a single factor $\boldsymbol{t}_{i}$ only). Observe that each term on the right-hand side of this equation can be written as

$$
\begin{aligned}
\partial_{t} \ell_{j}\left(\boldsymbol{t}_{j}\right) & =\left\{\boldsymbol{\alpha} \in \mathcal{D}^{\star} \mid \ell_{j}\left(\boldsymbol{s}_{j}\right)-\ell_{j}\left(\boldsymbol{t}_{j}\right) \geq\langle\boldsymbol{\alpha}, \boldsymbol{s}-\boldsymbol{t}\rangle \text { for all } \boldsymbol{s} \in \mathcal{D}\right\} \\
& =\left\{\boldsymbol{\alpha} \in \mathcal{D}^{\star} \mid \ell_{j}\left(\boldsymbol{s}_{j}\right)-\ell_{j}\left(\boldsymbol{t}_{j}\right) \geq \sum_{i=1}^{n}\left\langle\boldsymbol{\alpha}_{i}, \boldsymbol{s}_{i}-\boldsymbol{t}_{i}\right\rangle \text { for all } \boldsymbol{s} \in \mathcal{D}\right\} \\
& =\{\mathbf{0}\} \times \cdots \times\{\mathbf{0}\} \times \partial \ell_{j}\left(\boldsymbol{t}_{j}\right) \times\{\mathbf{0}\} \times \cdots \times\{\mathbf{0}\} .
\end{aligned}
$$

Then, the equation (3.2.1) follows immediately. Finally, a direct calculation shows that the convex conjugate of the loss functional is given by

$$
\begin{aligned}
\mathcal{L}^{\star}(\boldsymbol{\alpha}) & =\sup _{\boldsymbol{t} \in \mathcal{D}}\{\langle\boldsymbol{\alpha}, \boldsymbol{t}\rangle-\mathcal{L}(\boldsymbol{t})\} \\
& =\sup _{\boldsymbol{t} \in \mathcal{D}}\left\{\sum_{i=1}^{n}\left(\left\langle\boldsymbol{\alpha}_{i}, \boldsymbol{t}_{i}\right\rangle-\ell_{i}\left(\boldsymbol{t}_{i}\right)\right)\right\} \\
& =\sum_{i=1}^{n} \sup _{\boldsymbol{t}_{i} \in \mathcal{D}_{i}}\left\{\left\langle\boldsymbol{\alpha}_{i}, \boldsymbol{t}_{i}\right\rangle-\ell_{i}\left(\boldsymbol{t}_{i}\right)\right\}=\sum_{i=1}^{n} \ell_{i}^{\star}\left(\boldsymbol{\alpha}_{i}\right),
\end{aligned}
$$

where the properness of the loss functions is used to exchange the supremum and the sum to obtain the next-to-last equality. This shows that equality (3.2.2) holds. Finally, equation (3.2.3) follows directly from (3.2.2).

This means that the subdifferential and the convex conjugate function of a separable loss functional can be easily computed if the corresponding terms are known for the defining loss functions. Note that the latter are usually much easier to compute in practice.

### 3.3 Regularization with Standard Decisions

In this section we investigate the common regularization approach for the case of standard decision as in Definition 2.4.3 and Definition 2.4.4. First, observe that the adjoint operator of the standard decision operator can be computed directly as follows.

Proposition 3.3.1 (adjoint of standard decision operator)
Let $\boldsymbol{T}: \mathcal{F} \times \mathbb{R} \rightarrow \mathbb{R}^{n}$ be the standard decision operator as in Definition 2.4.3.
Then, the adjoint operator is given by

$$
\begin{aligned}
\boldsymbol{T}^{\star}: \mathbb{R}^{n} & \rightarrow \mathcal{F} \times \mathbb{R}, \\
\boldsymbol{\alpha} & \mapsto\left(\sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}, \mathbf{1}^{\top} \boldsymbol{\alpha}\right) .
\end{aligned}
$$

Proof. The form of the adjoint operator can be directly verified as the equations

$$
\begin{aligned}
\langle\boldsymbol{\alpha}, \boldsymbol{T}(\boldsymbol{w}, b)\rangle=\sum_{i=1}^{n} \alpha_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right) & =\left\langle\boldsymbol{w}, \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}\right\rangle+b \sum_{i=1}^{n} \alpha_{i} \\
& =\left\langle(\boldsymbol{w}, b),\left(\sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}, \mathbf{1}^{\top} \boldsymbol{\alpha}\right)\right\rangle
\end{aligned}
$$

hold for all $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$.
Moreover, it is possible to compute the convex conjugate of the standard regularization functional as follows.

Proposition 3.3.2 (conjugate function of standard regularization functional)
Let $\mathcal{R}: \mathcal{F} \times \mathcal{F}_{0} \rightarrow \mathbb{R}$ be the standard regularization functional as in Definition 2.4.4.
Then, the subdifferential can be computed as

$$
\begin{equation*}
\partial \mathcal{R}\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right)=\{\lambda \boldsymbol{w}\} \times\{\mathbf{0}\} \tag{3.3.1}
\end{equation*}
$$

and the dual conjugate of $\mathcal{R}$ is given by

$$
\mathcal{R}^{\star}\left(\boldsymbol{w}^{\star}, \boldsymbol{w}_{0}^{\star}\right)= \begin{cases}\frac{1}{2 \lambda}\left\|\boldsymbol{w}^{\star}\right\|^{2} & \text { if } \boldsymbol{w}_{0}^{\star}=\mathbf{0} \\ \infty & \text { otherwise } .\end{cases}
$$

Moreover, the subdifferential of $\mathcal{R}^{\star}$ is given by

$$
\partial \mathcal{R}^{\star}\left(\boldsymbol{w}^{\star}, \boldsymbol{w}_{0}^{\star}\right)= \begin{cases}\frac{1}{\lambda} \boldsymbol{w}^{\star}, & \text { if } \boldsymbol{w}_{0}^{\star}=\mathbf{0},  \tag{3.3.2}\\ \varnothing, & \text { otherwise } .\end{cases}
$$

Proof. Because the function $\mathcal{R}$ is continuously differentiable, its subdifferential is a singleton that only contains the derivative due to Proposition A.1.3. Thus, the equation (3.3.1) can be easily verified.

By definition of the convex conjugate (see Definition A.2.1) we obtain

$$
\begin{aligned}
\mathcal{R}^{\star}\left(\boldsymbol{w}^{\star}, \boldsymbol{w}_{0}^{\star}\right) & =\sup _{\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right) \in \mathcal{F} \times \mathcal{F}_{0}}\left\{\left\langle\left(\boldsymbol{w}^{\star}, \boldsymbol{w}_{0}^{\star}\right),\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right)\right\rangle-\mathcal{R}\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right)\right\} \\
& =\sup _{\boldsymbol{w} \in \mathcal{F}}\left\{\left\langle\boldsymbol{w}^{\star}, \boldsymbol{w}\right\rangle-\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}\right\}+\sup _{\boldsymbol{w}_{0} \in \mathcal{F}_{0}}\left\{\left\langle\boldsymbol{w}_{0}^{\star}, \boldsymbol{w}_{0}\right\rangle\right\} .
\end{aligned}
$$

Note that the second term is infinite whenever $\boldsymbol{w}_{0}^{\star} \neq \mathbf{0}$. Moreover, we can rewrite the first term as

$$
\sup _{\boldsymbol{w} \in \mathcal{F}}\left\{\left\langle\boldsymbol{w}^{\star}, \boldsymbol{w}\right\rangle-\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}\right\}=-\inf _{\boldsymbol{w} \in \mathcal{F}}\left\{\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}-\left\langle\boldsymbol{w}^{\star}, \boldsymbol{w}\right\rangle\right\} .
$$

Taking the necessary and sufficient optimality conditions for convex optimization problems into account, the infimum on the right-hand side is attained at the point $\boldsymbol{w} \in \mathcal{F}$ with $\lambda \boldsymbol{w}=\boldsymbol{w}^{\star}$. Thus, the term is equal to $\frac{1}{2 \lambda}\left\|\boldsymbol{w}^{\star}\right\|^{2}$.

In order to compute the subdifferential of $\mathcal{R}^{\star}$, we note that $\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right) \in \partial \mathcal{R}^{\star}\left(\boldsymbol{w}^{\star}, \boldsymbol{w}_{0}^{\star}\right)$ if and only if $\left(\boldsymbol{w}^{\star}, \boldsymbol{w}_{0}^{\star}\right) \in \partial \mathcal{R}\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right)=\{\lambda \boldsymbol{w}\} \times\{\mathbf{0}\}$. This proves equation (3.3.2).

A special case which is very frequently used in applications is the combination of standard decisions and the standard regularization functional. For reference, we write down the terms for this combination in the following proposition. In order to simplify the notation, we introduce the notion of the kernel matrix first.

Definition 3.3.3 (kernel matrix)
Let $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ be a set of feature vectors as in Definition 2.4.3. Then, the matrix

$$
\boldsymbol{K} \in \mathbb{R}^{n \times n} \quad \text { with } \quad K_{i j}=\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}_{j}\right\rangle \text { for } \quad i, j \in\{1, \ldots, n\}
$$

is called kernel matrix.

Then, we can compute the conjugate of the regularization functional for standard decisions as follows.

Proposition 3.3.4 (conjugate regularization functional for standard decisions) Let $\boldsymbol{T}: \mathcal{F} \times \mathbb{R} \rightarrow \mathbb{R}^{n}$ be the standard decision operator as in Definition 2.4.3 and let $\mathcal{R}: \mathcal{F} \times \mathbb{R} \rightarrow \mathbb{R}$ be the standard regularization functional (2.4.2) with $\lambda>0$. Then,

$$
\mathcal{R}^{\star}\left(\boldsymbol{T}^{\star} \boldsymbol{\alpha}\right)= \begin{cases}\frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}, & \text { if } \mathbf{1}^{\top} \boldsymbol{\alpha}=0  \tag{3.3.3}\\ \infty, & \text { otherwise }\end{cases}
$$

Proof. Equation (3.3.3) is an immediate consequence of Proposition 3.3.1 and Proposition 3.3.2, when it is used that

$$
\left\|\sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}\right\|^{2}=\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{i} \alpha_{j}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}_{j}\right\rangle=\boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha} .
$$

Whenever the general training problem is considered, it is not immediately clear whether it has a unique solution. This is because the ordinary regularization functional (2.4.2) does not yield a strictly convex optimization problem in general. To improve this situation, one could alternatively use the term

$$
\begin{equation*}
\mathcal{R}_{\lambda, \mu}(\boldsymbol{w}, b):=\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\frac{\mu}{2} b^{2} \tag{3.3.4}
\end{equation*}
$$

with regularization parameters $\lambda>0$ and $\mu>0$. Then, the functional $\mathcal{R}_{\lambda, \mu}$ is uniformly convex and a training of the form (2.4.1) with this regularization has a unique solution. For reference, the following proposition summarizes the basic properties of the functional $\mathcal{R}_{\lambda, \mu}$.

Proposition 3.3.5 (conjugate function of uniformly convex regularization functional) Let $\mathcal{R}_{\lambda, \mu}$ be the regularization functional defined in (3.3.4) with $\lambda>0$ and $\mu \geq 0$. If $\mu=0$, then $\mathcal{R}_{\lambda, \mu}=\mathcal{R}_{\lambda}$ with $\mathcal{R}_{\lambda}$ as defined in (2.4.2) and the subdifferential and the conjugate function is given by Proposition 3.3.2. If $\mu>0$, then it follows that

$$
\begin{equation*}
\partial \mathcal{R}_{\lambda, \mu}(\boldsymbol{w}, b)=\{\lambda \boldsymbol{w}\} \times\{\mu b\} \tag{3.3.5}
\end{equation*}
$$

and

$$
\begin{equation*}
\mathcal{R}_{\lambda, \mu}^{\star}\left(\boldsymbol{w}^{\star}, b^{\star}\right)=\frac{1}{2 \lambda}\left\|\boldsymbol{w}^{\star}\right\|^{2}+\frac{1}{2 \mu}\left(b^{\star}\right)^{2} . \tag{3.3.6}
\end{equation*}
$$

Proof. Because the function $\mathcal{R}_{\lambda, \mu}$ is continuously differentiable, its subdifferential is a singleton which only contains the derivative due to Proposition A.1.3. This means that (3.3.5) follows immediately. Moreover, by definition of the convex conjugate we obtain

$$
\begin{aligned}
\mathcal{R}_{\lambda, \mu}^{\star}\left(\boldsymbol{w}^{\star}, b^{\star}\right) & =\sup _{(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}}\left\{\left\langle\left(\boldsymbol{w}^{\star}, b^{\star}\right),(\boldsymbol{w}, b)\right\rangle-\mathcal{R}(\boldsymbol{w}, b)\right\} \\
& =\sup _{\boldsymbol{w} \in \mathcal{F}}\left\{\left\langle\boldsymbol{w}^{\star}, \boldsymbol{w}\right\rangle-\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}\right\}+\sup _{b \in \mathbb{R}}\left\{b^{\star} b-\frac{\mu}{2} b^{2}\right\} .
\end{aligned}
$$

However, this implies (3.3.6) because the suprema are attained exactly at the points $\boldsymbol{w}=\lambda^{-1} \boldsymbol{w}^{\star}$ and $b=\mu^{-1} b^{\star}$, respectively.

### 3.4 Duality for the Standard Training Problem

Because the standard training problem is a particular interesting setting which is suitable to represent many practical learning problems, we provide a specialization of the general duality theorem here for reference. Taking the findings from the previous subsections together, we can conclude the following assertion.

Corollary 3.4.1 (dual problem for standard training problems)
Consider a standard training problem as in Definition 2.4.5 and suppose that Assumption 3.1.2 holds.
Then, the dual training problem associated with (2.4.3) is

$$
\begin{equation*}
\min _{\alpha \in \mathbb{R}^{n}} \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}+\sum_{i=1}^{n} \ell_{i}^{\star}\left(-\alpha_{i}\right) \quad \text { s.t. } \quad \mathbf{1}^{\top} \boldsymbol{\alpha}=0 . \tag{3.4.1}
\end{equation*}
$$

In particular, this problem is always solvable.

Proof. The assertion follows directly from Corollary 3.1.3 using Proposition 3.2.1 and Proposition 3.3.4.

It is also possible to obtain a particular characterization of the dual solution if one considers standard training problems only. This leads to the following specialization of the optimality conditions.

Corollary 3.4.2 (optimality conditions for standard training problems)
Consider a standard training problem as in Definition 2.4.5 and suppose that Assumption 3.1.2 holds.
Then, a vector $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ is a solution of the primal training problem (2.4.3) if and only if there exists some vector $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ such that

$$
\begin{gather*}
\mathbf{1}^{\top} \boldsymbol{\alpha}=0,  \tag{3.4.2}\\
\boldsymbol{w}=\frac{1}{\lambda} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}, \tag{3.4.3}
\end{gather*}
$$

and

$$
\begin{equation*}
-\alpha_{i} \in \partial \ell_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right) \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{3.4.4}
\end{equation*}
$$

In particular, $\boldsymbol{\alpha}$ is a solution of the dual training problem (3.4.1) in this case.

Proof. The assertion follows directly from Theorem 3.1.4 using Proposition 3.2.1 and Proposition 3.3.4.

In the same way, the dual conditions of Theorem 3.1.4 can be used to derive optimality conditions for the dual training problem as follows.

Corollary 3.4.3 (dual optimality conditions for standard training problems)
Consider a standard training problem as in Definition 2.4.5 and suppose that Assumption 3.1.2 holds.
Then, a vector $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ is a solution of the dual training problem (3.4.1) if and only if there exists $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ such that the conditions (3.4.2), (3.4.3) and (3.4.4) are satisfied. In particular, the pair $(\boldsymbol{w}, b)$ is a solution of the primal training problem (2.4.3) in this case.

### 3.5 Finite-dimensional Representation of the Solution

One of the theoretical questions associated with the general training problem (2.4.1) is whether its solution can be characterized by means of the given input data. In particular, this is important because the definition of the training problem in an infinitedimensional space prevents the direct application of numerical methods. Using a suitable representation of the solution set, we are able to derive an equivalent finitedimensional optimization problem which can be treated numerically.

A basic theorem that paves the way for a finite-dimensional representation of the solution is the so-called Representer Theorem which dates back to an article about spline interpolation, see [KW70]. Another formulation of the theorem in [SHSO1] attracted attention later on because it enabled a method for solving nonlinear Support Vector Machines practically. Over the years many other theoretical investigations associated with the Representer Theorem in different problem settings have been conducted. For instance, in a recent paper [BRG19] the authors derive an application of the theory to deep kernel learning problems.

Using our problem setting, a first formulation of the finite-dimensional representation property of the solution is the following theorem which is a direct consequence of the optimality conditions summarized in Corollary 3.4.2.

Corollary 3.5.1 (Representer Theorem)
Consider a standard training problem as in Definition 2.4.5 and suppose that Assumption 3.1.2 holds.
Then, every solution ( $\boldsymbol{w}, b$ ) of the primal training problem (2.4.3) satisfies

$$
\begin{equation*}
\boldsymbol{w}=\frac{1}{\lambda} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}, \tag{3.5.1}
\end{equation*}
$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ is a solution of the dual training problem (3.4.1). In particular, the optimal decision function (see Definition 2.4.3) has the form

$$
\begin{equation*}
d(\boldsymbol{\varphi})=\frac{1}{\lambda} \sum_{i=1}^{n} \alpha_{i}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}\right\rangle+b . \tag{3.5.2}
\end{equation*}
$$

Proof. The assertion follows immediately from Corollary 3.4.2.
As a conclusion of this subsection, we also want to present a more general version of the Representer Theorem which follows directly from the general optimality conditions given in Theorem 3.1.4. It is very similar to the Nonparametric Representer Theorem stated in [SHS01, Theorem 1].

Corollary 3.5.2 (Representer Theorem, general version)
Consider a training problem as in Definition 2.4.1. Suppose that Assumption 3.1.1 and Assumption 3.1.2 hold. Let $\boldsymbol{T}$ be the standard decision operator (see Definition 2.4.3) and let $\mathcal{R}=h \circ \mathcal{R}_{\lambda}$ with

- $\mathcal{R}_{\lambda}$ defined by (2.4.2) with $\lambda>0$ and
- a convex, lower-semicontinuous, strictly increasing function $h: \mathbb{R}_{+} \rightarrow \mathbb{R}$.

Then, any solution ( $\boldsymbol{w}, b$ ) of the primal training problem (2.4.1) satisfies

$$
\begin{equation*}
\boldsymbol{w} \in \operatorname{span}\left\{\boldsymbol{\varphi}_{1}, \ldots, \boldsymbol{\varphi}_{n}\right\} \tag{3.5.3}
\end{equation*}
$$

Proof. Under the present assumptions one can apply a particular version of the chain rule for the convex subdifferential (see [BC11, Corollary 16.72]) to obtain

$$
\begin{aligned}
\partial \mathcal{R}(\boldsymbol{w}, b)=\partial\left(h \circ \mathcal{R}_{\lambda}\right)(\boldsymbol{w}, b) & =\left\{c \boldsymbol{\omega}^{\star} \mid\left(c, \boldsymbol{\omega}^{\star}\right) \in \partial h\left(\mathcal{R}_{\lambda}(\boldsymbol{w}, b)\right) \times \partial \mathcal{R}_{\lambda}(\boldsymbol{w}, b)\right\} \\
& =\left\{c \cdot\left(\boldsymbol{w}^{\star}, b^{\star}\right) \mid c \in \partial h\left(\mathcal{R}_{\lambda}(\boldsymbol{w}, b)\right),\left(\boldsymbol{w}^{\star}, b^{\star}\right) \in \partial \mathcal{R}_{\lambda}(\boldsymbol{w}, b)\right\}
\end{aligned}
$$

for any $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$. On the other hand, the subdifferential of the regularization functional is given by $\partial \mathcal{R}_{\lambda}(\boldsymbol{w}, b)=\{(\lambda \boldsymbol{w}, 0)\}$, see Proposition 3.3.2. This means that

$$
\begin{aligned}
\partial \mathcal{R}(\boldsymbol{w}, b) & =\left\{c \cdot\left(\boldsymbol{w}^{\star}, b^{\star}\right) \mid c \in \partial h\left(\mathcal{R}_{\lambda}(\boldsymbol{w}, b)\right), \boldsymbol{w}^{\star}=\lambda \boldsymbol{w}, b^{\star}=0\right\} \\
& =\left\{(c \lambda \boldsymbol{w}, 0) \mid c \in \partial h\left(\mathcal{R}_{\lambda}(\boldsymbol{w}, b)\right)\right\} .
\end{aligned}
$$

Because the function $h$ is strictly increasing, it follows that $c>0$ for all $c \in \partial h\left(\mathcal{R}_{\lambda}(\boldsymbol{w}, b)\right)$. Using Proposition 3.3.1 and the left-hand part of (3.1.3), we obtain

$$
\boldsymbol{w}=(c \lambda)^{-1} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i} \quad \text { and } \quad \mathbf{1}^{\top} \boldsymbol{\alpha}=0
$$

Hence, (3.5.3) is true for any solution ( $\boldsymbol{w}, b$ ) of (2.4.1).
A particular consequence of the Representer Theorem is that one can replace the possibly infinite-dimensional variable $\boldsymbol{w} \in \mathcal{F}$ in the training problem (2.4.1) by means
of

$$
\boldsymbol{w}=\sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}
$$

with some other finite-dimensional variable $\boldsymbol{\alpha} \in \mathbb{R}^{n}$. This means that the training problem can be transformed into a finite-dimensional optimization problem under the assumptions of Corollary 3.5.2. This approach will be discussed in more detail later on in Chapter 6, where we especially investigate the relationship between the primal training problem and the corresponding finite-dimensional formulation.

### 3.6 Interpretation of Optimality Conditions

Whereas we used the first part of the optimality conditions in (3.1.4) in Section 3.5 to obtain a particular finite-dimensional representation of the solution, the second part also leads to interesting observations. The evaluation and interpretation of the resulting properties is the aim of this section. We start this approach by writing down the yet unused conditions for the particular setting.

Corollary 3.6.1 (characterization of loss derivative)
Consider a general training problem as in Definition 2.4.1 with a separable loss functional (see Definition 2.4.2). Suppose that Assumption 3.1.1 and Assumption 3.1.2 hold. Then, for any solution $\boldsymbol{\omega} \in \mathcal{H}$ of the primal training problem (2.4.1) and any solution $\boldsymbol{\alpha} \in \mathcal{D}^{\star}$ of the dual training problem (3.1.1) it follows that

$$
\begin{equation*}
-\boldsymbol{\alpha}_{i} \in \partial \ell_{i}\left([\boldsymbol{T} \boldsymbol{\omega}]_{i}\right) \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{3.6.1}
\end{equation*}
$$

Proof. Under the present assumptions, we can apply the right-hand part of (3.1.3) together with (3.2.1) to obtain (3.6.1).

The previous corollary has a rather descriptive interpretation. It means that each component $\boldsymbol{\alpha}_{i}$ of the dual variable vector (which is associated with exactly one loss function) has a value which is equal to the negative slope of the corresponding loss function at the optimal decision value. In practice, each loss function is often associated with exactly one training sample. This means that the value $\boldsymbol{\alpha}_{i}$ can be interpreted as a sensitivity of the training problem with respect to the $i$ th training sample.

A particular consequence of this observation leads to an idea which is closely related to the classical notion of support vectors. One possible definition of the term support vector is the following.

Definition 3.6.2 (support vector)
Consider a training problem with standard decisions according to Definition 2.4.3. Suppose that Assumption 3.1.1 and Assumption 3.1.2 hold. Then, the vector $\varphi_{i} \in \mathcal{F}$ is called support vector if there exists a solution $\boldsymbol{\alpha}$ of the dual training problem (3.1.1)
such that $\boldsymbol{\alpha}_{i} \neq \mathbf{0}$.

Note that this definition is not strictly equivalent to the definition given in the Subsections 2.2.1 and 2.2.2, where geometric features of the optimization problems were used. Though, in Section 4.3 we will elaborate on the relationship after the introduction of the necessary loss functions.

We also have to point out that the preceding definition of support vectors does not have the form which is typically used in the literature. Typically, one would simply say that $\boldsymbol{\varphi}_{i}$ is a support vector if $\boldsymbol{\alpha}_{i} \neq \mathbf{0}$ for some fixed solution $\boldsymbol{\alpha}$ of the dual problem. Strictly speaking the definition is not accurate if the dual solution is not uniquely defined. In this situation it might happen that there are different solutions with zero and non-zero coefficients $\boldsymbol{\alpha}_{i}$ for some fixed $i \in\{1, \ldots, n\}$, and it is not obvious whether the belonging vector $\boldsymbol{\varphi}_{i}$ should be called a support vector or not. This is why we use the more precise definition above.

Keeping in mind that the optimal decision function has the form (3.5.2), we can see that only support vectors contribute to the generated decision. Moreover, from the practical point of view, the number of nonzero coefficients in the decision function has an outstanding importance because it generally is proportional to the computational effort which is needed to evaluate the decision function. Formally, the number of terms in the decision function is equal to the number of loss functions (or training samples) which can be quite high in practice. This means that it is desirable to determine formulations of the training problem which tend to have sparse solutions and still capture the important parts of the learning problem. Here, by sparse we mean that only a relatively small portion of the training samples leads to nonzero dual coefficients in the solution, i.e., the number of support vectors is limited in a suitable way.

In the formulation of the necessary optimality conditions in Corollary 3.6.1 it was essential to know a solution of the primal training problem. In contrast, the following statement presents necessary and sufficient optimality conditions in terms of the dual variables only.

Corollary 3.6.3 (dual optimality conditions)
Consider a standard training problem as in Definition 2.4.5. Suppose that Assumption 3.1.1 and Assumption 3.1.2 hold.
Then, a vector $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ is a solution of the dual training problem (3.4.1) if and only if

$$
\begin{equation*}
\mathbf{1}^{\top} \boldsymbol{\alpha}=0 \tag{3.6.2}
\end{equation*}
$$

and there exists $b \in \mathbb{R}$ such that

$$
\begin{equation*}
-\alpha_{i} \in \partial \ell_{i}\left(\frac{1}{\lambda} \sum_{j=1}^{n} \alpha_{j}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}_{j}\right\rangle+b\right) \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{3.6.3}
\end{equation*}
$$

Proof. If the primal training problem (2.4.1) is a standard training problem, we can proceed as in the proof of the Representer Corollary 3.5.1 to see that the left part of (3.1.3) is equivalent to

$$
\boldsymbol{w}=\frac{1}{\lambda} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i} \quad \text { and } \quad \mathbf{1}^{\top} \boldsymbol{\alpha}=0
$$

Using the first equation and the representation for the subdifferential of the loss functional in Proposition 3.2.1, we can see that the right part of (3.1.3) is equivalent to (3.6.3). This proves that the conditions (3.6.2) and (3.6.3) are necessary and sufficient for the optimality of $\boldsymbol{\alpha}$ in the dual training problem (3.1.1).

Remark 3.6.4 (rewritten dual optimality conditions)
It is easy to see that (3.6.3) can be written equivalently as

$$
\begin{equation*}
b \in \bigcap_{i=1}^{n}\left(\partial \ell_{i}^{\star}\left(-\alpha_{i}\right)-\frac{1}{\lambda} \sum_{j=1}^{n} \alpha_{j}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}_{j}\right\rangle\right) \tag{3.6.4}
\end{equation*}
$$

because the subdifferential can be inverted in the sense that $-\alpha_{i} \in \partial \ell_{i}\left(t_{i}\right)$ it and only if $t_{i} \in \partial \ell_{i}^{\star}\left(-\alpha_{i}\right)$.

Note that the assertion of Corollary 3.6.3 provides a unification of the different formulation of dual optimality conditions and determination strategies for $b$ in

- [BGV92, Section 2.2] for Support Vector Classification,
- [SS04, Section 1.4] for $\varepsilon$-insensitive Support Vector Regression, and
- [CL11, Subsection 4.1.5] for a general SVM formulation.

Certainly, the above list is not exhaustive and many other variants of dual optimality conditions (depending on the particular training problem) exist in the literature.

### 3.7 Notes on the Existence of Solutions

One natural question concerning the training problem is whether it has a solution at all. In order to answer the question certainly, it is necessary to require additional assumptions in general to rule out particular edge cases. Two of such edge cases are presented in the following examples.

The first example shows that it is possible to have an unbounded objective function in the general context.

Example 3.7.1 (unbounded training problem)
Consider a standard training problem with $\mathcal{F}:=\mathbb{R}$ and a single training sample $\boldsymbol{\varphi}_{1}:=0$
with loss function $\ell_{1}(t):=t$. Then, the training problem (2.4.3) is given by

$$
\min _{w, b \in \mathbb{R}} \frac{\lambda}{2} w^{2}+b
$$

This problem has no solution because the objective function is unbounded for $b \rightarrow-\infty$.

A second example shows that even the boundedness of the objective function does not guarantee the solvability of the training problem.

Example 3.7.2 (training problem with unattained infimum value)
Consider a standard training problem with $\mathcal{F}:=\mathbb{R}$ and a single training sample $\boldsymbol{\varphi}_{1}:=0$ with loss function $\ell_{1}(t):=\log (1+\exp (t))$. Then, the training problem (2.4.3) is given by

$$
\min _{w, b \in \mathbb{R}} \frac{\lambda}{2} w^{2}+\log (1+\exp (b))
$$

This problem has no solution because

$$
\inf _{w, b \in \mathbb{R}}\left\{\frac{\lambda}{2} w^{2}+\log (1+\exp (b))\right\}=0
$$

for $w=0$ and $b \rightarrow-\infty$, but the value is not attained at any $(w, b) \in \mathbb{R}^{2}$.

The common issue in both examples is that there exists a direction for the variable $b$, in which the loss term is not increasing. Of course, this case is not typical for practical applications. With this observation in mind, we can formulate a sufficient condition which implies solvability of the training problem in practice, for instance, the following.

Definition 3.7.3 (coercive function)
A function $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{+\infty\}$ is called coercive, iffor every sequence $\left\{\boldsymbol{x}^{k}\right\}_{k \in \mathbb{N}} \subseteq \mathcal{H}$ with $\lim _{k \rightarrow \infty}\left\|\boldsymbol{x}^{k}\right\|=\infty$ it follows that $\lim _{k \rightarrow \infty} f\left(\boldsymbol{x}^{k}\right)=\infty$.

The following result is derived in [Ale19] and shows a rather general setting, in which coercivity is sufficient for solvability of an optimization problem.

Corollary 3.7.4 (minimizers of coercive functions, [Ale19, Corollary 5.6])
Let $\mathcal{H}$ be a real Hilbert space and let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{+\infty\}$ be lower semi-continuous and coercive. Then, the function $f$ is bounded from below and attains a minimizer.

In the special context of a standard training problem, the following proposition provides a sufficient condition for the coercivity of the objective function.

Proposition 3.7 .5 (sufficient condition for coercivity)
Consider a standard training problem as in Definition 2.4.5. Suppose that Assumption 3.1.1 holds and let $\operatorname{dom}\left(\psi_{\mathrm{p}}\right) \neq \varnothing$. Moreover, suppose that all loss functions are bounded from below, and let there be $i^{+}, i^{-} \in\{1, \ldots, n\}$ such that

$$
\lim _{t \rightarrow \infty} \ell_{i^{+}}(t)=\infty \quad \text { and } \quad \lim _{t \rightarrow-\infty} \ell_{i^{-}}(t)=\infty
$$

Then, the objective function of the primal training problem (2.4.3) is coercive. In particular, this problem has a solution.

Proof. Let $\left\{\left(\boldsymbol{w}^{k}, b^{k}\right)\right\}_{k \in \mathbb{N}} \subseteq \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ be an unbounded sequence. Subsequently, we consider two cases.
Case 1: The sequence $\left\{\boldsymbol{w}^{k}\right\}_{k \in \mathbb{N}}$ is unbounded.
Since we assumed that the loss functions are bounded from below, there exist constants $B_{i} \in \mathbb{R}$ such that

$$
\frac{\lambda}{2}\left\|\boldsymbol{w}^{k}\right\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\left\langle\boldsymbol{w}^{k}, \boldsymbol{\varphi}_{i}\right\rangle+b^{k}\right) \geq \frac{\lambda}{2}\left\|\boldsymbol{w}^{k}\right\|^{2}+\sum_{i=1}^{n} B_{i}
$$

This implies that the objective function is unbounded for $k \rightarrow \infty$.
Case 2: The sequence $\left\{\boldsymbol{w}^{k}\right\}_{k \in \mathbb{N}}$ is bounded.
Then, the sequence $\left\{b^{k}\right\}_{k \in \mathbb{N}}$ must be unbounded, i.e., either $b^{k} \rightarrow \infty$ or $b^{k} \rightarrow-\infty$. In the first case, we obtain

$$
\frac{\lambda}{2}\left\|\boldsymbol{w}^{k}\right\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\left\langle\boldsymbol{w}^{k}, \boldsymbol{\varphi}_{i}\right\rangle+b^{k}\right) \geq \sum_{i \in\{1, \ldots, n\} \backslash\left\{i^{+}\right\}}^{n} B_{i}+\ell_{i^{+}}\left(\left\langle\boldsymbol{w}^{k}, \boldsymbol{\varphi}_{i^{+}}\right\rangle+b^{k}\right) .
$$

Because the sequence $\left\{\boldsymbol{w}^{k}\right\}_{k \in \mathbb{N}}$ is bounded and $\lim _{t \rightarrow \infty} \ell_{i^{+}}(t)=\infty$, the right-hand side is unbounded. The same follows for the case $b^{k} \rightarrow-\infty$ if one considers the loss function $\ell_{i^{-}}$.
Taking both parts together, we showed that the objective function is coercive. The existence of a solution for the training problem directly follows from Corollary 3.7.4.

Note that the stated requirements on the loss functions are virtually always satisfied for practical problems. In particular the classical examples for training problems that we consider in Chapter 4 naturally have the required properties.

### 3.8 Distance Estimation and Uniqueness of Solutions

From the mathematical point of view, another interesting question associated with the general training problem (2.4.1) is under which conditions it has a unique solution. In order to approach this question, we want to apply the distance estimation result of Theorem A.4.2 subsequently.

Corollary 3.8.1 (distance estimation using regularization-induced measure)
Consider the general training problem (2.4.1). Suppose that Assumption 3.1.1 and Assumption 3.1.2 hold. Let the regularization operator be defined by $\mathcal{R}:=\mathcal{R}_{\lambda}$ with $\lambda>0$. Suppose that $\overline{\boldsymbol{\omega}}=\left(\overline{\boldsymbol{w}}, \overline{\boldsymbol{w}}_{0}\right) \in \mathcal{F} \times \mathcal{F}_{0}$ and $\overline{\boldsymbol{\alpha}} \in \mathcal{D}^{\star}$ are solutions of (2.4.1) and (3.1.1), respectively. Moreover, let $\boldsymbol{\alpha} \in \mathcal{D}^{\star}$ be some arbitrary dual point.
Then,

$$
\begin{equation*}
\|\boldsymbol{w}-\overline{\boldsymbol{w}}\| \leq \sqrt{\frac{2}{\lambda} \Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \overline{\boldsymbol{\alpha}})} \leq \sqrt{\frac{2}{\lambda} \Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \boldsymbol{\alpha})} \tag{3.8.1}
\end{equation*}
$$

for any $\boldsymbol{\omega}=\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right) \in \mathcal{F} \times \mathcal{F}_{0}$, where $\Delta_{\mathrm{pd}}$ denotes the primal-dual optimality gap defined in (3.1.2). In particular, the first component of the solution of problem (2.4.1) is unique in the sense that for any solution $\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right) \in \mathcal{F} \times \mathcal{F}_{0}$ of (2.4.1) it holds that $\boldsymbol{w}=\overline{\boldsymbol{w}}$.

Proof. Let $\overline{\boldsymbol{\omega}}=\left(\overline{\boldsymbol{w}}, \overline{\boldsymbol{w}}_{0}\right) \in \mathcal{F} \times \mathcal{F}_{0}$ be a solution of the training problem (2.4.1) and suppose that $\boldsymbol{\omega}=\left(\boldsymbol{w}, \boldsymbol{w}_{0}\right) \in \operatorname{dom}(\mathcal{R})$. Then, we can compute the directional derivative of $\mathcal{R}_{\lambda}$ as

$$
\mathcal{R}_{\lambda}^{\circ}\left(\overline{\boldsymbol{\omega}},\left(\boldsymbol{d}, \boldsymbol{d}_{0}\right)\right)=\lambda\langle\overline{\boldsymbol{w}}, \boldsymbol{d}\rangle .
$$

This implies that the Bregman distance between $\boldsymbol{\omega}$ and $\overline{\boldsymbol{\omega}}$ is given by

$$
D_{\mathcal{R}_{\lambda}}(\boldsymbol{\omega}, \overline{\boldsymbol{\omega}})=\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}-\frac{\lambda}{2}\|\overline{\boldsymbol{w}}\|^{2}-\lambda\langle\overline{\boldsymbol{w}}, \boldsymbol{w}-\overline{\boldsymbol{w}}\rangle=\frac{\lambda}{2}\|\boldsymbol{w}-\overline{\boldsymbol{w}}\|^{2} .
$$

Note that $\boldsymbol{T}^{\star} \overline{\boldsymbol{\alpha}} \in \partial \mathcal{R}(\overline{\boldsymbol{\omega}})$ is satisfied because of the optimality conditions of Theorem 3.1.4. By means of Theorem A.4.2, this implies that

$$
\begin{equation*}
\frac{\lambda}{2}\|\boldsymbol{w}-\overline{\boldsymbol{w}}\|^{2}=D_{\mathcal{R}_{\lambda}}(\boldsymbol{\omega}, \overline{\boldsymbol{\omega}}) \leq \Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \overline{\boldsymbol{\alpha}}) \tag{3.8.2}
\end{equation*}
$$

Thus, the first inequality in (3.8.1) holds. To see that the second inequality is also true, note that $\psi_{\mathrm{d}}(\overline{\boldsymbol{\alpha}}) \leq \psi_{\mathrm{d}}(\boldsymbol{\alpha})$ since $\overline{\boldsymbol{\alpha}}$ is a solution of the dual training problem. This implies

$$
\Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \overline{\boldsymbol{\alpha}})=\psi_{\mathrm{p}}(\boldsymbol{\omega})+\psi_{\mathrm{d}}(\overline{\boldsymbol{\alpha}}) \leq \psi_{\mathrm{p}}(\boldsymbol{\omega})+\psi_{\mathrm{d}}(\boldsymbol{\alpha})=\Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \boldsymbol{\alpha}),
$$

such that (3.8.1) follows directly.
Finally, suppose that $\boldsymbol{\omega} \in \mathcal{F} \times \mathcal{F}_{0}$ is also a solution of the training problem. Then, $\Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \overline{\boldsymbol{\alpha}})=0$, and by means of (3.8.2) it follows that $\frac{\lambda}{2}\|\boldsymbol{w}-\overline{\boldsymbol{w}}\|^{2}=0$, i.e., $\boldsymbol{w}=\overline{\boldsymbol{w}}$. This proves the uniqueness of the first component of the solution.

The corollary also shows a way to obtain an estimate for the distance of the first component of the primal variables to the solution (which is unique in with respect to that component). This could be used to evaluate the quality of the corresponding decision function. Note that we cannot obtain any information about the second component of the variable vector (which corresponds to the bias term). However, this is a natural property because the second component of the solution is not necessarily unique.

Subsequently, we want to apply Theorem A.4.2 again to derive distance estimates which are based on the property of the loss functions. As a preliminary result, we compute the Bregman distance for separable loss functionals in the following lemma.

Lemma 3.8.2 (Bregman distance for separable loss functionals)
Let $\mathcal{L}$ be a separable loss functional. Then,

$$
D_{\mathcal{L}}(\boldsymbol{t}, \overline{\boldsymbol{t}})=\sum_{i=1}^{n} D_{\ell_{i}}\left(\boldsymbol{t}_{i}, \overline{\boldsymbol{t}}_{i}\right) \quad \text { for all } \quad \boldsymbol{t}, \overline{\boldsymbol{t}} \in \operatorname{dom}(\mathcal{L}) .
$$

Proof. Let $\boldsymbol{t}, \overline{\boldsymbol{t}} \in \operatorname{dom}(\mathcal{L})$. Because $\mathcal{D}=\mathcal{D}_{1} \times \cdots \times \mathcal{D}_{n}$, we can write $\boldsymbol{t}=\left(\boldsymbol{t}_{1}, \ldots, \boldsymbol{t}_{n}\right) \in \mathcal{D}$. The same notation can be used for $\overline{\boldsymbol{t}}$ and $\boldsymbol{z} \in \mathcal{D}$. Then, by definition of the directional derivative, we obtain

$$
\begin{aligned}
\mathcal{L}^{\circ}(\overline{\boldsymbol{t}}, \boldsymbol{z}) & =\lim _{h \searrow 0} \frac{1}{h}(\mathcal{L}(\overline{\boldsymbol{t}}+h \boldsymbol{z})-\mathcal{L}(\overline{\boldsymbol{t}})) \\
& =\lim _{h \searrow 0} \frac{1}{h}\left(\sum_{i=1}^{n} \ell_{i}\left(\overline{\boldsymbol{t}}_{i}+h \boldsymbol{z}_{i}\right)-\sum_{i=1}^{n} \ell_{i}\left(\overline{\boldsymbol{t}}_{i}\right)\right) \\
& =\sum_{i=1}^{n} \lim _{h \searrow 0} \frac{1}{h}\left(\ell_{i}\left(\overline{\boldsymbol{t}}_{i}+h \boldsymbol{z}_{i}\right)-\ell_{i}\left(\overline{\boldsymbol{t}}_{i}\right)\right)=\sum_{i=1}^{n} \ell_{i}^{\circ}\left(\overline{\boldsymbol{t}}_{i}, \boldsymbol{z}_{i}\right)
\end{aligned}
$$

for any $\boldsymbol{z} \in \mathcal{D}$. Then, the Bregman distance can be computed as

$$
\begin{aligned}
D_{\mathcal{L}}(\boldsymbol{t}, \overline{\boldsymbol{t}}) & =\mathcal{L}(\boldsymbol{t})-\mathcal{L}(\overline{\boldsymbol{t}})-\mathcal{L}^{\circ}(\overline{\boldsymbol{t}}, \boldsymbol{t}-\overline{\boldsymbol{t}}) \\
& =\sum_{i=1}^{n} \ell_{i}\left(\boldsymbol{t}_{i}\right)-\ell_{i}\left(\overline{\boldsymbol{t}}_{i}\right)-\ell_{i}^{\circ}\left(\overline{\boldsymbol{t}}_{i}, \boldsymbol{t}_{i}-\overline{\boldsymbol{t}}_{i}\right)=\sum_{i=1}^{n} D_{\ell_{i}}\left(\boldsymbol{t}_{i}, \overline{\boldsymbol{t}}_{i}\right) .
\end{aligned}
$$

This proves the assertion of the lemma.
With these particular terms for the Bregman distance induced from the loss functions, a distance estimation in the spirit of Corollary 3.8.1 is possible.

Corollary 3.8.3 (distance estimation using a loss-induced measure)
Consider the general training problem (2.4.1). Suppose that Assumption 3.1.1 and Assumption 3.1.2 hold. Let $\mathcal{L}$ be a separable loss functional. Suppose that $\overline{\boldsymbol{\omega}} \in \mathcal{H}$ and $\overline{\boldsymbol{\alpha}} \in \mathcal{D}^{\star}$ are solutions of (2.4.1) and (3.1.1), respectively. Moreover, let $\boldsymbol{\omega} \in \mathcal{H}$ be some arbitrary point satisfying $\boldsymbol{T} \boldsymbol{\omega} \in \operatorname{dom}(\mathcal{L})$.
Then, for all $\boldsymbol{\alpha} \in \mathcal{D}^{\star}$ it follows that

$$
\sum_{i=1}^{n} D_{\ell_{i}}(\boldsymbol{T} \boldsymbol{\omega}, \boldsymbol{T} \overline{\boldsymbol{\omega}}) \leq \Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \overline{\boldsymbol{\alpha}}) \leq \Delta_{\mathrm{pd}}(\boldsymbol{\omega}, \boldsymbol{\alpha})
$$

Proof. Because the proof follows the same lines as that of Corollary 3.8.1 but uses Lemma 3.8.2 to rewrite the left-hand side, we omit the detailed steps here.

## 4 Examples of Classification and Regression Problems

Up to now, we did not assume too much structure on the functions which make up the general training problem. This section aims to apply the previous general results to some special training problems. Note that for most of the considered problems the dual formulations are already known in the literature to some extent. However, we summarize those formulations here for reference and provide some slight generalizations where appropriate.

All the exemplary problems of this chapter follow the structure of a standard training problem according to Definition 2.4.5. In particular, we consider different formulations of classification (Section 4.1) and regression problems (Section 4.2). Afterwards, the optimality conditions derived in the previous chapter are applied to some particular training problems (Section 4.3). In order to prepare for the application of derivativebased training methods, a principle for the construction of smooth loss function is introduced (Section 4.4). Finally, we briefly show that a smoothing of the KKT system for a dual Support Vector Classification problem results in a particularly smoothed loss function in the primal training problem (Section 4.5).

### 4.1 Binary Classification Problems

In the classical version of Support Vector Machines (see also Subsection 2.2.2) the socalled hinge loss function is used for each training sample. As mentioned in the introduction of the standard regularization term in Definition 2.4.4, we omit the weighting parameter $C$ in front of the loss term. Instead, we introduce the weighting coefficient as part of the regularization term.

### 4.1.1 Classical Support Vector Classification

Taking all ingredients together, we obtain the dual training problem for (2.2.5) with parameter $C:=\lambda^{-1}$ as follows. A visualization of the loss functions and the associated conjugate functions and subdifferentials is given in Figure 4.1.1a.

Proposition 4.1.1 (Support Vector Classification)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in\{-1,1\}^{n}$ be a vector of labels and $\boldsymbol{c} \in \mathbb{R}_{>0}^{n}$ a vector of weights for a binary classification problem. Consider the sample-weighted Support Vector Classification problem

$$
\begin{equation*}
\min _{\boldsymbol{w}, b} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} c_{i} \max \left\{0,1-y_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right)\right\} \tag{4.1.1}
\end{equation*}
$$

with a parameter $\lambda>0$. The dual optimization problem associated with (4.1.1) is given by

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha}} & \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}^{\top} \boldsymbol{\alpha} \\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0, \\
& \alpha_{i} \in c_{i} y_{i}[0,1] \text { for all } i \in\{1, \ldots, n\}, \tag{4.1.2c}
\end{array}
$$

where the term $y_{i}[0,1]$ denotes the interval $\left[y_{i}, 0\right]$ if $y_{i}<0$.

Proof. The problem is a standard training problem. This means that Corollary 3.4.1 is applicable with

$$
\ell_{i}(t):=c_{i} \max \left\{0,1-y_{i} t\right\}
$$

for $i \in\{1, \ldots, n\}$ as loss functions. Now, consider a single index $i \in\{1, \ldots, n\}$. Because the loss function is differentiable at all points except for $t=y_{i}$, the subdifferential can be easily computed as

$$
\partial \ell_{i}(t)= \begin{cases}\{0\} & \text { if } y_{i} t>1 \\ -c_{i} y_{i}[0,1] & \text { if } y_{i} t=1 \\ \left\{-c_{i} y_{i}\right\} & \text { if } y_{i} t<1\end{cases}
$$

This means that, by definition of the conjugate function, we obtain

$$
\ell_{i}^{\star}(\alpha)=\sup _{t \in \mathbb{R}}\left\{\alpha t-c_{i} \max \left\{0,1-y_{i} t\right\}\right\}=\sup _{t \in \mathbb{R}}\left\{\min \left\{\alpha t,\left(\alpha+c_{i} y_{i}\right) t-c_{i}\right\}\right\}
$$

This term is infinite if the linear functions in both terms of the minimum are either both decreasing ( $\alpha<0$ and $\alpha<-c_{i} y_{i}$ ) or increasing ( $\alpha>0$ and $\alpha>-c_{i} y_{i}$ ), i.e., if $\alpha \notin\left[\min \left\{0,-c_{i} y_{i}\right\}, \max \left\{0,-c_{i} y_{i}\right\}\right]$. Because $y_{i} \in\{-1,1\}$, this condition can also be written as $\alpha \notin-c_{i} y_{i}[0,1]$.

On the other hand, if $\alpha \in-c_{i} y_{i}[0,1]$, the supremum is attained at the intersection of the linear functions. This is exactly the point $t=y_{i}^{-1}=y_{i}$. Hence, the supremum has the value $y_{i} \alpha$ in this case. Taking both cases together, it follows that

$$
\ell_{i}^{\star}(\alpha)= \begin{cases}y_{i} \alpha, & \text { if } \alpha \in-c_{i} y_{i}[0,1] \\ \infty, & \text { otherwise }\end{cases}
$$

Taking everything together, we obtain the dual objective function, where the regular-
ization term is given by (3.3.3), and

$$
\mathcal{L}^{\star}(-\boldsymbol{\alpha})=\sum_{i=1}^{n} \ell_{i}^{\star}\left(-\alpha_{i}\right)= \begin{cases}-\sum_{i=1}^{n} y_{i} \alpha_{i}, & \text { if } \alpha_{i} \in c_{i} y_{i}[0,1] \text { for all } i \in\{1, \ldots, n\} \\ \infty, & \text { otherwise }\end{cases}
$$

Finally, note that for the minimization problem only points with finite objective function values need to be considered. For this reason, constraints in the formulation of (4.1.1) are added. The constraints (4.1.2c) ensure that the loss functional is finite. In the same way, the constraint (4.1.2b) implies a finite value of the regularization functional.

Note that the dual problem (4.1.2) is not strictly identical to the dual problem (2.2.6) proposed in the introduction. First, this is due to the weighting, which is not present in the original formulation. By choosing $c_{i}=1$ for all $i \in\{1, \ldots, n\}$ one can obtain the classical problem with uniform weights. Now, we want to mention briefly, how to transform one representation of one dual problem to the other equivalently. If the variables $\alpha_{i}$ in (4.1.2) are replaced by $y_{i} \lambda \alpha_{i}$, the resulting factor $\lambda$ in the objective function does not change the problem and can be omitted. Then, is easy to see that the objective function and the constraints of both problems are equal when the relation $C=\lambda^{-1}$ is used.

### 4.1.2 Maximal Margin Classifier

In the same way as above, the dual training problem for the maximal margin classifier (see Subsection 2.2.1) can be derived. The result is stated in the following proposition. The employed loss functions are shown in Figure 4.1.1b.

Proposition 4.1.2 (dual problem for maximal margin classifier)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in\{-1,1\}^{n}$ be a vector of labels for a binary classification problem. The dual problem of the maximal margin classifier

$$
\min _{\boldsymbol{w}, b} \frac{1}{2}\|\boldsymbol{w}\|^{2} \quad \text { s.t. } \quad y_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right) \geq 1 \quad \text { for all } \quad i \in\{1, \ldots, n\}
$$

is given by

$$
\begin{array}{cl}
\min _{\boldsymbol{\alpha}} & \frac{1}{2} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}^{\top} \boldsymbol{\alpha} \\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0, \\
& y_{i} \alpha_{i} \geq 0 \quad \text { for all } \quad i \in\{1, \ldots, n\} .
\end{array}
$$

Proof. The problem has the form of a standard training problem with $\lambda=1$ and loss functions

$$
\ell_{i}(t):= \begin{cases}0, & \text { if } y_{i} t \geq 1 \\ \infty, & \text { otherwise }\end{cases}
$$

For these loss functions the conjugate functions can be computed as

$$
\ell_{i}^{\star}(-\alpha)=\sup _{t \in \mathbb{R}}\left\{-\alpha t-\ell_{i}(t)\right\}=-\inf _{t \in \mathbb{R}: y_{i} t \geq 1}\{\alpha t\}=-\inf _{t \in \mathbb{R}: t \geq 1}\left\{\alpha y_{i} t\right\}=- \begin{cases}y_{i} \alpha, & \text { if } y_{i} \alpha \geq 0 \\ \infty, & \text { otherwise }\end{cases}
$$

The remaining arguments to show that the dual problem has the particular form are the same as in the proof of Proposition 4.1.1.

### 4.1.3 Logistic Regression Support Vector Classification

Besides the previous two classical formulations of SVMs for classification problems, one could also think of a combination of the logistic regression approach with the standard SVM regularization term. The resulting problem is presented in the following proposition together with its dual problem. See also Figure 4.1.1c for a plot of the loss functions and the corresponding conjugate functions. For a more detailed discussion of the logistic regression problem we refer to [Bis06, Section 4.3].

Proposition 4.1.3 (logistic regression)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in\{-1,1\}^{n}$ and $\boldsymbol{c} \in \mathbb{R}_{>0}^{n}$ be vectors of labels and weights for a binary classification problem. The dual problem of the logistic regression SVM

$$
\min _{\boldsymbol{w}, b} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} c_{i} \log \left(1+\exp \left(-y_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right)\right)\right)
$$

is given by

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha}} & \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}+\sum_{i=1}^{n}\left(q\left(1-\frac{y_{i}}{c_{i}} \alpha_{i}\right)+q\left(\frac{y_{i}}{c_{i}} \alpha_{i}\right)\right) \\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0, \\
& \alpha_{i} \in c_{i} y_{i}[0,1] \quad \text { for all } \quad i \in\{1, \ldots, n\},
\end{array}
$$

where $q: \mathbb{R} \rightarrow \mathbb{R} \cup\{+\infty\}$ denotes the function

$$
q(x)= \begin{cases}x \log (x) & \text { if } x>0  \tag{4.1.3}\\ 0, & \text { if } x=0 \\ \infty & \text { if } x<0\end{cases}
$$

Proof. As we have seen in the previous propositions, the problem has the form of a standard training problem. Here, the loss functions have the form

$$
\ell_{i}(t):=c_{i} \log \left(1+\exp \left(-y_{i} t\right)\right)
$$

for $i \in\{1, \ldots, n\}$. We first compute the first derivative

$$
\ell_{i}^{\prime}(t)=-c_{i} y_{i} \frac{\exp \left(-y_{i} t\right)}{1+\exp \left(-y_{i} t\right)}=-c_{i} y_{i} \frac{1}{1+\exp \left(y_{i} t\right)}
$$

and the second derivative

$$
\ell_{i}^{\prime \prime}(t)=c_{i} \frac{\exp \left(y_{i} t\right)}{\left(1+\exp \left(y_{i} t\right)\right)^{2}} .
$$

From this, we can see that $\ell_{i}^{\prime \prime}(t)>0$ for all $t \in \mathbb{R}$, i.e., the loss functions are strictly convex. Then, we can compute the conjugate loss functions by means of the definition as

$$
\begin{equation*}
\ell_{i}^{\star}(-\alpha)=\sup _{t \in \mathbb{R}}\left\{-\alpha t-\ell_{i}(t)\right\}=-\inf _{t \in \mathbb{R}}\left\{\alpha t+\ell_{i}(t)\right\} . \tag{4.1.4}
\end{equation*}
$$

Since the function $\ell_{i}$ is continuously differentiable, the derivative of the term with respect to $t$ must vanish at the point where the infimum is attained. This is the case if

$$
\alpha=-\ell_{i}^{\prime}(t)=c_{i} y_{i} \frac{1}{1+\exp \left(y_{i} t\right)} .
$$

The right-hand side of this equation can only have values in $c_{i} y_{i}(0,1)$, i.e., the infimum can only be attained for $\alpha \in c_{i} y_{i}(0,1)$. Moreover, solving this equation for $t$, it follows that

$$
\exp \left(y_{i} t\right)=\frac{c_{i} y_{i}}{\alpha}-1 \quad \text { and } \quad t=y_{i} \log \left(\frac{c_{i} y_{i}}{\alpha}-1\right) .
$$

Then, we get

$$
\begin{aligned}
\ell_{i}^{\star}(-\alpha) & =-y_{i} \alpha \log \left(\frac{c_{i} y_{i}}{\alpha}-1\right)-\log \left(1+\left(\frac{c_{i} y_{i}}{\alpha}-1\right)^{-1}\right) \\
& =-y_{i} \alpha\left(\log \left(1-\frac{y_{i}}{c_{i}} \alpha\right)-\log \left(\frac{y_{i}}{c_{i}} \alpha\right)\right)+\log \left(1-\frac{y_{i}}{c_{i}} \alpha\right) \\
& =\left(1-\frac{y_{i}}{c_{i}} \alpha\right) \log \left(1-\frac{y_{i}}{c_{i}} \alpha\right)+\frac{y_{i}}{c_{i}} \alpha \log \left(\frac{y_{i}}{c_{i}} \alpha\right) \\
& =q\left(1-\frac{y_{i}}{c_{i}} \alpha\right)+q\left(\frac{y_{i}}{c_{i}} \alpha\right)
\end{aligned}
$$

for $\alpha \in c_{i} y_{i}(0,1)$. The edge cases $\alpha=0$ and $\alpha=c_{i} y_{i}$ need to be considered separately. In both cases one can easily verify by means of (4.1.4), that $\ell_{i}^{\star}(-\alpha)=0$. This shows that $\ell_{i}^{\star}(-\alpha)=q\left(1-\frac{y_{i}}{c_{i}} \alpha\right)+q\left(\frac{y_{i}}{c_{i}} \alpha\right)$ for all $\alpha \in c_{i} y_{i}[0,1]$. Finally, note that $\ell_{i}^{\star}(-\alpha)=\infty$ if $\alpha \notin c_{i} y_{i}(0,1)$ is easy to see using (4.1.4). The form of the dual problem is then given by Corollary 3.4.1.

### 4.2 Regression Problems

For regression problems real-valued labels $y_{i}$ for $i \in\{1, \ldots, n\}$ are given. Then, the value of the decision function for the $i$ th data point should be as close as possible
to the value of $y_{i}$ for all $i \in\{1, \ldots, n\}$. However, the measure of closeness which determines the loss function depends on the application at hand. Note that some loss functions can be more appealing than others from a practical point of view. We will give some more detailed explanation in Subsection 8.4.4. Subsequently, we consider three particular examples of regression problems.

### 4.2.1 Least-Squares Support Vector Machine

The first example uses the squared error as a loss function. With this choice, the resulting problem is called Least-Squares Support Vector Machine (LS-SVM). We summarize the problem together with the associated dual training problem in the following proposition. An exemplary loss function and the corresponding derivative and conjugate function are visualized in Figure 4.2.1a.

Proposition 4.2.1 (Least-Squares Support Vector Machine)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in \mathbb{R}^{n}$ and $\boldsymbol{c} \in \mathbb{R}_{>0}^{n}$ be vectors of labels and weights for a regression problem. The dual problem of the sampleweighted Least-Squares Support Vector Machine

$$
\min _{\boldsymbol{w}, b} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\frac{1}{2} \sum_{i=1}^{n} c_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b-y_{i}\right)^{2}
$$

is given by

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha}} & \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}+\sum_{i=1}^{n} c_{i} \alpha_{i}^{2}-\boldsymbol{y}^{\top} \boldsymbol{\alpha} \\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0 .
\end{array}
$$

Proof. Note that the problem is a standard training problem with uniformly convex loss functions

$$
\ell_{i}(t):=\frac{c_{i}}{2}\left(t-y_{i}\right)^{2} \quad \text { for } \quad i \in\{1, \ldots, n\}
$$

In order to compute the conjugate loss function we employ (4.1.4) to get

$$
\ell_{i}^{\star}(-\alpha)=-\inf _{t \in \mathbb{R}}\left\{\alpha t+\frac{c_{i}}{2}\left(t-y_{i}\right)^{2}\right\}=\frac{1}{2 c_{i}} \alpha^{2}-\alpha y_{i},
$$

where we used that the infimum is attained at the point $t=y_{i}-\frac{\alpha}{c_{i}}$. The form of the dual problem follows directly from Corollary 3.4.1.

### 4.2.2 $\varepsilon$-Support Vector Regression

A second example, which we consider, is the so-called $\varepsilon$-Support Vector Regression ( $\varepsilon$-SVR). It results from the definition of loss functions that are zero for values within a distance of at most $\varepsilon$ from the given labels and increase linearly away from this range.

The particular training problem and its dual form is summarized in the subsequent proposition. An $\varepsilon$-insensitive loss function and its conjugate function are visualized in Figure 4.2.1 b.

Proposition 4.2.2 ( $\varepsilon$-Support Vector Regression)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in \mathbb{R}^{n}$ and $\boldsymbol{c} \in \mathbb{R}_{>0}^{n}$ be vectors of labels and weights for a regression problem. The dual problem of the sampleweighted $\varepsilon$-Support Vector Regression problem

$$
\begin{equation*}
\min _{\boldsymbol{w}, b} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} c_{i} \max \left\{0,\left|\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b-y_{i}\right|-\varepsilon\right\} \tag{4.2.1}
\end{equation*}
$$

is given by

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha}} & \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}^{\top} \boldsymbol{\alpha}+\varepsilon \sum_{i=1}^{n}\left|\alpha_{i}\right| \\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0, \\
& \alpha_{i} \in c_{i}[-1,1] \text { for } \quad i \in\{1, \ldots, n\} .
\end{array}
$$

Proof. Again, the problem is a standard training problem and Corollary 3.4.1 can be applied. Subsequently, we compute the conjugate of the loss functions

$$
\ell_{i}(t):=c_{i} \max \left\{0,\left|t-y_{i}\right|-\varepsilon\right\} .
$$

By definition, it follows that

$$
\ell_{i}^{\star}(-\alpha)=\sup _{t \in \mathbb{R}}\left\{-\alpha t-c_{i} \max \left\{0,\left|t-y_{i}\right|-\varepsilon\right\}\right\} .
$$

One can easily see that $\ell_{i}^{\star}(-\alpha)=\infty$ whenever $\alpha \notin c_{i}[-1,1]$. Moreover, we have

$$
-\alpha t-c_{i} \max \left\{0,\left|t-y_{i}\right|-\varepsilon\right\} \leq-\alpha t
$$

for all $t \in \mathbb{R}$ and equality holds for all $t \in\left[y_{i}-\varepsilon, y_{i}+\varepsilon\right]$. This shows that the supremum in the definition of the conjugate loss is attained for $t=y_{i}+\varepsilon$ if $\alpha<0$ and for $t=y_{i}-\varepsilon$ if $\alpha \geq 0$, i.e., at $t=y_{i}-\operatorname{sign}(\alpha)$ for $\alpha \in c_{i}[-1,1]$. Thus,

$$
\ell_{i}^{\star}(-\alpha)= \begin{cases}-y_{i} \alpha+\varepsilon|\alpha| & \text { if } \alpha \in c_{i}[-1,1] \\ \infty, & \text { otherwise }\end{cases}
$$

The form of the dual problem then follows from Corollary 3.4.1.

### 4.2.3 Hard $\varepsilon$-Support Vector Regression

As a third example, we consider a formulation of a regression problem where the model function is required to match the given labels up to some sample-dependent tolerance $\varepsilon_{i}>0$.

Proposition 4.2.3 (hard $\varepsilon$-Support Vector Regression)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in \mathbb{R}^{n}$ and $\varepsilon \in \mathbb{R}_{>0}^{n}$ be vectors of labels and tolerances for a regression problem. Assume that there exists some $(\overline{\boldsymbol{w}}, \bar{b}) \in \mathcal{F} \times \mathbb{R}$ such that

$$
\begin{equation*}
\left|\left\langle\overline{\boldsymbol{w}}, \boldsymbol{\varphi}_{i}\right\rangle+\bar{b}-y_{i}\right|<\varepsilon_{i} \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{4.2.2}
\end{equation*}
$$

Consider the hard $\varepsilon$-Support Vector Regression problem

$$
\begin{equation*}
\min _{\boldsymbol{w}, b} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} \chi_{\mathbb{R}_{+}}\left(\varepsilon_{i}-\left|\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b-y_{i}\right|\right), \tag{4.2.3}
\end{equation*}
$$

where

$$
\begin{aligned}
\chi_{\mathbb{R}_{+}}: \mathbb{R} & \rightarrow \mathbb{R} \cup\{+\infty\}, \\
t & \mapsto \chi_{\mathbb{R}_{+}}(t):= \begin{cases}0, & \text { if } t \geq 0, \\
\infty, & \text { otherwise },\end{cases}
\end{aligned}
$$

denotes the characteristic function of the set $\mathbb{R}_{+}$. Then, the dual problem of (4.2.3) is given by

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha}} & \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}^{\top} \boldsymbol{\alpha}+\sum_{i=1}^{n} \varepsilon_{i}\left|\alpha_{i}\right|  \tag{4.2.4}\\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0 .
\end{array}
$$

Proof. First, note that the existence of $(\overline{\boldsymbol{w}}, \bar{b}) \in \mathcal{F} \times \mathbb{R}$ with (4.2.2) guarantees that Assumption 3.1.2 is satisfies. Then, we consider loss functions

$$
\ell_{i}(t)=\chi_{\mathbb{R}_{+}}\left(\varepsilon_{i}-\left|t-y_{i}\right|\right)= \begin{cases}0, & \text { if }\left|t-y_{i}\right| \leq \varepsilon_{i} \\ \infty, & \text { otherwise }\end{cases}
$$

for $i \in\{1, \ldots, n\}$. It is easy to see that the conjugate loss function can be computed as

$$
\ell_{i}^{\star}(-\alpha)=\sup _{t \in \mathbb{R}}\left\{-\alpha t-\ell_{i}(t)\right\}=\sup _{t:\left|t-y_{i}\right| \leq \varepsilon_{i}}\{-\alpha t\}=\varepsilon_{i}|\alpha|-y_{i} \alpha .
$$

Hence, the form of the dual training problem (4.2.4) can be verified by means of Corollary 3.4.1.

### 4.3 Interpretation of Optimality Conditions Revisited

With some practical training problems at hand, we can now revisit the results of Section 3.5 and Section 3.6 to gain some insight into the properties of the solution. In the following subsections, we consider the typical examples for classification and regression problems, and give a more practical interpretation of the conditions (4.3.2) for those examples. For simplicity, we focus on uniformly weighted training problems
and note that the subsequent observations can be easily transferred to the weighted formulations.

Recall that, in this context, the decision function can be represented by

$$
\begin{equation*}
d(\boldsymbol{\varphi})=\frac{1}{\lambda} \sum_{i=1}^{n} \alpha_{i}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}\right\rangle+b \tag{4.3.1}
\end{equation*}
$$

due to Corollary 3.5.1 with $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ being a solution of the dual training problem. Moreover, following Corollary 3.6.3, the vector $\boldsymbol{\alpha}$ satisfies

$$
\begin{equation*}
-\alpha_{i} \in \partial \ell_{i}\left(d\left(\boldsymbol{\varphi}_{i}\right)\right) \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{4.3.2}
\end{equation*}
$$

This observation has a very natural interpretation considering the structure of the training problem. First, the influence of one loss function $\ell_{i}$ (or training sample) on the decision function is equal to the value of the corresponding dual variable $\alpha_{i}$. Second, this value is equal to the negative slope of the loss function at the optimal decision. Of course, it is in general not possible to use this observation to argue for each loss function independently, because the optimal decision function usually depends on the collection of all loss functions. However, the characterization is still quite representative if one considers the final values depending on the optimal solution of the training problem.

### 4.3.1 ... for Support Vector Classification

Recall that, in the typical formulation of a binary classification problem, loss functions of the form

$$
\ell_{i}(t)=\max \left\{0,1-y_{i} t\right\} \quad \text { for } \quad i \in\{1, \ldots, n\}
$$

are used, and the corresponding subdifferential is given by

$$
\partial \ell_{i}(t)= \begin{cases}\{0\} & \text { if } y_{i} t>1  \tag{4.3.3}\\ -y_{i}[0,1] & \text { if } y_{i} t=1 \\ \left\{-y_{i}\right\} & \text { if } y_{i} t<1\end{cases}
$$

for $i \in\{1, \ldots, n\}$, see Proposition 4.1.1. In the following, we discuss the necessary conditions (4.3.2) for this special case.

First, note that $y_{i} d\left(\boldsymbol{\varphi}_{i}\right)>1$ implies $\alpha_{i}=0$. This means that any training sample for which the decision function has a magnitude of more than one in the correct direction does not contribute to the decision function itself. Associated with this observation there are two important consequences.

On the one hand, one can conclude that adding and removing of such "non-critical" training samples from the training problem does not change the decision function at all. We want to summarize this observation in the following proposition.

Proposition 4.3.1 (adding and removing non-support vectors for SVC) Consider a Support Vector Classification problem as given in Proposition 4.1.1 with uniform weights $\boldsymbol{c}=\mathbf{1}$. Suppose that $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ is a solution of the primal training problem (4.1.1), and let d denote the generated decision function.
Then, the problem can be modified in two possible ways without changing the solution:
(a) for any $i \in\{1, \ldots, n\}$ with $y_{i} d\left(\boldsymbol{\varphi}_{i}\right)>1$, the $i$ th loss function can be removed from the problem, and
(b) any new training sample $\left(\boldsymbol{\varphi}_{j}, y_{j}\right) \in \mathcal{F} \times\{-1,1\}$ satisfying $y_{j} d\left(\boldsymbol{\varphi}_{j}\right)>1$ can be added.

Proof. Suppose that $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ and $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ are solutions of the problems (4.1.1) and (4.1.2), respectively.

First, let $i \in\{1, \ldots, n\}$ with $y_{i} d\left(\boldsymbol{\varphi}_{i}\right)>1$. Because of (4.3.2) and (4.3.3), it follows that $\alpha_{i}=0$. This means that the $i$ th term can be removed from the original decision function without changing it. Now, consider the training problem without the $i$ th loss function and the $i$ th component of the decision operator. Then, it is easy to see (due to Corollary 3.6.3) that the dual point resulting from removing the $i$ th component from the vector $\boldsymbol{\alpha}$ is still optimal for the new dual problem. In particular, the optimality conditions from Corollary 3.4.2 are still satisfied, which means that ( $\boldsymbol{w}, b$ ) is also optimal for the reduced problem.

On the other hand, let $\left(\boldsymbol{\varphi}_{j}, y_{j}\right) \in \mathcal{F} \times\{-1,1\}$ with $y_{j} d\left(\boldsymbol{\varphi}_{j}\right)>1$ be given and consider the training problem with this sample added. Then, one zero component can be appended to the dual point $\boldsymbol{\alpha}$. However, the resulting dual point is feasible for the augmented training problem and satisfies the optimality conditions from Corollary 3.4.2. Thus, $(\boldsymbol{w}, b)$ is again optimal for the new training problem.

The idea of Proposition 4.3.1 could be used to construct suitable strategies for incremental learning problems (or online learning), where only a specific subset of the training set is considered at a particular time. We do not want to elaborate on this idea here and refer to [CPOO] for a promising approach in this direction.

The second consequence of the observation above is that for support vectors it follows that $y_{i} d\left(\boldsymbol{\varphi}_{i}\right) \leq 1$. This has a close connection to the geometrical motivation of SVMs as introduced in Section 2.2. Support vectors (in the sense of Definition 3.6.2) either fall into or lie at the boundary of the space between the two separating hyperplanes as discussed in Subsection 2.2.2. Note that this condition is only necessary and not sufficient for being a support vector. Hence, it might happen that a support vector $\boldsymbol{\varphi}_{i}$ in the sense of Subsection 2.2.2 (lying in or at the boundary of the space between the separating hyperplanes) does not satisfy $\alpha_{i} \neq 0$, i.e., is not a support vector in the sense of Definition 3.6.2.

### 4.3.2 ... for $\varepsilon$-insensitive Support Vector Regression

Similar to the previous investigation, we can proceed to gain insight in the special case of regression problems. As a first example, we consider the $\varepsilon$-insensitive loss function

$$
\ell_{i}(t)=\max \left\{0,\left|t-y_{i}\right|-\varepsilon\right\}
$$

used in Proposition 4.2.2. For this function a simple computation shows that the subdifferential is given by

$$
\partial \ell_{i}(t)= \begin{cases}\{-1\}, & \text { if } t<y_{i}-\varepsilon \\ {[-1,0],} & \text { if } t=y_{i}-\varepsilon, \\ \{0\}, & \text { if } y_{i}-\varepsilon<t<y_{i}+\varepsilon, \\ {[0,1],} & \text { if } t=y_{i}+\varepsilon \\ \{1\}, & \text { if } t>y_{i}+\varepsilon\end{cases}
$$

Considering again the necessary optimality condition (4.3.2) we can observe that $\alpha_{i}=0$ whenever $d\left(\boldsymbol{\varphi}_{i}\right) \in\left(y_{i}-\varepsilon, y_{i}+\varepsilon\right)$ holds. In other words, support vectors correspond to certain "outliers" in the sense that they satisfy $\left|d\left(\boldsymbol{\varphi}_{i}\right)-y_{i}\right| \geq \varepsilon$. In the same fashion as for classification problems, we can formulate a proposition which states that non-support vectors do not contribute to the optimal decision function.

Proposition 4.3.2 (adding and removing non-support vectors for $\varepsilon$-SVR)
Consider an $\varepsilon$-SVR problem as given in Proposition 4.2.2 with uniform weights $\boldsymbol{c}=\mathbf{1}$.
Suppose that $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ is a solution of the training problem (4.2.1) and let $d$ denote the generated decision function.
Then, the problem can be modified in two possible ways without changing the solution:
(a) for any $i \in\{1, \ldots, n\}$ with $\left|d\left(\boldsymbol{\varphi}_{i}\right)-y_{i}\right|<\varepsilon$, the ith loss function can be removed from the problem, and
(b) any new training sample $\left(\boldsymbol{\varphi}_{j}, y_{j}\right) \in \mathcal{F} \times \mathbb{R}$ satisfying $\left|d\left(\boldsymbol{\varphi}_{i}\right)-y_{i}\right|<\varepsilon$ can be added.

Proof. The proof is basically identical to that of Proposition 4.3.1.
As a secondary result, we can see that the choice of the tube parameter $\varepsilon \geq 0$ also regulates the number of support vector. The larger, the parameter is chosen, the more training samples tend to fall into the $\varepsilon$-insensitve tube of the loss function. Of course, this argumentation only works intuitively, and we have not derived any theoretical guarantees here.

Furthermore, one can retrospectively estimate the influence of a single training sample on the decision function by means of the previous discussion and the shape of the decision function given by (4.3.1). Whereas training samples with predictions inside the $\varepsilon$-insensitive range of the loss functions do not contribute to the decision function, there are two other possible cases to consider: samples with a prediction error of exactly $\varepsilon$ and samples with an error greater than $\varepsilon$.

The absolute value of the weight $\alpha_{i}$ in the decision function associated with the former training samples is somewhere between zero and one. For the latter, the absolute value of the corresponding weight is equal to (but not greater than) one. This shows that the influence of outliers is fundamentally bounded when the $\varepsilon$-insensitive loss function is used in the problem definition. Finally, note that the sign of the weight is equal to the sign of the negative error value, i.e., the weight tries to level out the error in some sense.

### 4.3.3 ... for Least-Squares Support Vector Machine

As a second class of regression problems, we want to investigate a Least-Squares Support Vector Machine. For a given training dataset, the loss functions have the form

$$
\ell_{i}(t)=\frac{1}{2}\left(t-y_{i}\right)^{2},
$$

see Proposition 4.2.1. Because the loss function is continuously differentiable, the subdifferential at a given point is equal to the singleton set containing the derivative only, i.e.,

$$
\partial \ell_{i}(t)=\left\{\ell_{i}^{\prime}(t)\right\}=\left\{t-y_{i}\right\} .
$$

From the necessary optimality condition (4.3.2) it immediately follows that $\alpha_{i}=y_{i}-t$.
At this point, we cannot make straightforward observations as in the previous subsections. This is due to the fact that the term $y_{i}-t$ (which is equal to the regression error) will typically be nonzero. As a consequence, in the solution of LS-SVM virtually all training samples will generate support vectors. Strictly speaking, one could argue that the term "Support Vector Machine" is not really applicable for this kind of problem.

Having said that, it is still possible to gain some insight from the optimality conditions. The conditions say that each term $\alpha_{i}$ is proportional to the negative error of the associated training sample. Taking into account the representation of the decision function in (4.3.1), this means that the contribution of a single training sample to the decision function is proportional to the error made by the decision function. In particular, training samples which are true outliers tend to perturb the optimal decision function badly.

### 4.4 Differentiable Approximations of Loss Functions

Up to now, we considered loss functions that were mainly motivated from a theoretical point of view. The aim was to construct training problems which are able to generate decision functions for a particular application. In particular, there was no special focus on the practicability, i.e., on solving the resulting training problems. For instance, derivative-based solution methods require a differentiable objective function. Furthermore, for the application of Newton-type methods one usually requires the objective function to be at least twice differentiable.

Of course, we have already discussed some examples of smooth loss functions, for instance in Proposition 4.1.3 for binary classification problems and in Proposition 4.2.1 for regression problems. However, these particular examples have non-zero derivative everywhere and thus lead to a high number of support vectors as we have seen in the previous sections. To overcome this drawback it is helpful to consider smooth approximations of loss functions which have zero derivative on some appropriate subset. Later on, in Section 8.4, we will see that it is also desirable to consider loss functions for which the second derivative vanishes on a substantially large subset.

### 4.4.1 A Construction Principle

Subsequently, we want to introduce a simple approach for the construction of twice differentiable loss functions. We restrict our investigation to the max-function

$$
\begin{aligned}
m: & \mathbb{R} \\
x & \mapsto m(x):=\operatorname{Rax}\{x, 0\},
\end{aligned}
$$

because it is the common building block for many loss functions that are frequently used for the formulation of classification and regression problems. The following proposition outlines an approach for the construction of smooth approximations for the maximum function.

Proposition 4.4.1 (smooth approximation of the max-function)
Let $h: \mathbb{R} \rightarrow \mathbb{R}_{+}$be an integrable function satisfying
(a) $\int_{-\infty}^{\infty} h(x) \mathrm{d} x=1$ and
(b) $h(x)=h(-x)$ for all $x \in \mathbb{R}$.

Then, the function $\widehat{m}: \mathbb{R} \rightarrow \mathbb{R}_{+}$defined by

$$
\widehat{m}(x)=\int_{-\infty}^{x} \int_{-\infty}^{t} h(s) \mathrm{d} s \mathrm{~d} t
$$

is continuously differentiable with

$$
\begin{equation*}
\widehat{m}^{\prime}(x)=\int_{-\infty}^{x} h(s) \mathrm{d} s \quad \text { for all } \quad x \in \mathbb{R} . \tag{4.4.1}
\end{equation*}
$$

If $h$ is continuous, the function $\widehat{m}$ is twice continuously differentiable with

$$
\begin{equation*}
\widehat{m}^{\prime \prime}(x)=h(x) \quad \text { for all } \quad x \in \mathbb{R} \tag{4.4.2}
\end{equation*}
$$

In any case, $\widehat{m}$ approximates $m$ from above in the sense that

$$
\begin{equation*}
\widehat{m}(x) \geq m(x) \quad \text { for all } \quad x \in \mathbb{R}, \tag{4.4.3}
\end{equation*}
$$

$$
\begin{equation*}
\lim _{x \rightarrow-\infty} \widehat{m}(x)-m(x)=0, \quad \lim _{x \rightarrow \infty} \widehat{m}(x)-m(x)=0 \tag{4.4.4}
\end{equation*}
$$

and,

$$
\begin{equation*}
\max _{x \in \mathbb{R}}|\widehat{m}(x)-m(x)|=\widehat{m}(0) . \tag{4.4.5}
\end{equation*}
$$

If furthermore,
(c) there exists some $\delta>0$ such that $h(x)=0$ for $x \in \mathbb{R} \backslash(-\delta, \delta)$,
the approximation is exact outside the interval $(-\delta, \delta)$, i.e.,

$$
\begin{equation*}
\widehat{m}(x)=m(x) \quad \text { for all } \quad x \in \mathbb{R} \backslash(-\delta, \delta) \tag{4.4.6}
\end{equation*}
$$

In addition, the approximation error is bounded by

$$
\begin{equation*}
\max _{x \in \mathbb{R}}|\widehat{m}(x)-m(x)| \leq \frac{\delta}{2} \tag{4.4.7}
\end{equation*}
$$

Proof. By means of the fundamental theorem of calculus we obtain that $\widehat{m}$ is continuously differentiable, and its derivative has the form given in (4.4.1). In the same way it follows that the function is twice continuously differentiable if $h$ is continuous and the second derivative is given by (4.4.2). Moreover, by definition of $\widehat{m}$, we directly get

$$
\widehat{m}(x) \geq 0 \quad \text { for all } \quad x \in \mathbb{R}
$$

and

$$
\lim _{x \rightarrow-\infty} \widehat{m}(x)-m(x)=\lim _{x \rightarrow-\infty} \widehat{m}(x)=0 .
$$

Then, the symmetry of $h$ can be used to obtain

$$
\int_{-\infty}^{0} \int_{-\infty}^{t} h(s) \mathrm{d} s \mathrm{~d} t=\int_{0}^{\infty} \int_{-\infty}^{-t} h(s) \mathrm{d} s \mathrm{~d} t=\int_{0}^{\infty} \int_{t}^{\infty} h(s) \mathrm{d} s \mathrm{~d} t .
$$

This implies

$$
\begin{align*}
\widehat{m}(x) & =\int_{-\infty}^{x} \int_{-\infty}^{t} h(s) \mathrm{d} s \mathrm{~d} t \\
& =\int_{-\infty}^{0} \int_{-\infty}^{t} h(s) \mathrm{d} s \mathrm{~d} t+\int_{0}^{x} \int_{-\infty}^{t} h(s) \mathrm{d} s \mathrm{~d} t  \tag{4.4.8}\\
& =\int_{0}^{\infty} \int_{t}^{\infty} h(s) \mathrm{d} s \mathrm{~d} t+\int_{0}^{x}\left(1-\int_{t}^{\infty} h(s) \mathrm{d} s\right) \mathrm{d} t \\
& =\int_{x}^{\infty} \int_{t}^{\infty} h(s) \mathrm{d} s \mathrm{~d} t+x .
\end{align*}
$$

Hence, it follows that

$$
\widehat{m}(x) \geq x \quad \text { for all } \quad x \in \mathbb{R}
$$

and

$$
\lim _{x \rightarrow \infty} \widehat{m}(x)-m(x)=\lim _{x \rightarrow \infty} \widehat{m}(x)-x=0
$$

Taking both parts together, one can see that (4.4.3) and (4.4.4) hold.

Now, we can estimate the value of $|\widehat{m}(x)-m(x)|$ by considering two cases. For $x \leq 0$ the difference is given by

$$
\widehat{m}(x)-m(x)=\widehat{m}(x)=\int_{-\infty}^{x} \int_{-\infty}^{t} h(s) \mathrm{d} s \mathrm{~d} t \geq 0,
$$

which is increasing in $x$. For $x \geq 0$ equality (4.4.8) shows

$$
\widehat{m}(x)-m(x)=\widehat{m}(x)-x=\int_{x}^{\infty} \int_{t}^{\infty} h(s) \mathrm{d} s \mathrm{~d} t \geq 0
$$

which implies that the difference is decreasing in $x$. Thus, the largest difference is attained at the point $x=0$. This proves (4.4.5).

Lastly, if the function $h$ satisfies property (c), we directly obtain from the definition of $\widehat{m}$ that $\widehat{m}(x)=0=m(x)$ for $x \leq-\delta$ and from (4.4.8) that $\widehat{m}(x)=x=m(x)$ for $x \geq \delta$. Hence, (4.4.6) is true. The estimate on the approximation error is equal to

$$
\widehat{m}(0)=\int_{0}^{\infty} \int_{t}^{\infty} h(s) \mathrm{d} s \mathrm{~d} t=\int_{0}^{\delta} \int_{t}^{\delta} h(s) \mathrm{d} s \mathrm{~d} t \leq \int_{0}^{\delta} \frac{1}{2} \mathrm{~d} t=\frac{\delta}{2},
$$

which shows (4.4.7).

### 4.4.2 Examples of Smooth Approximation of the Maximum Function

Building on the idea of the previous proposition, we list three different examples for differentiable approximations of the maximum function. All the subsequent examples are visualized in Figure 4.4.1. We start with the simplest possible example for a twice differentiable approximation, namely a piecewise cubic function.

Example 4.4.2 (approximation using a piecewise polynomial of degree 3)
Let $h: \mathbb{R} \rightarrow \mathbb{R}_{+}$be a piecewise linear function defined by $h(x)=\max \{0,1-|x|\}$. Then, the function $\widehat{m}$ defined by

$$
\widehat{m}(x)= \begin{cases}0, & \text { if } x<-1, \\ \frac{1}{6}(1+x)^{3}, & \text { if }-1 \leq x<0, \\ x+\frac{1}{6}(1-x)^{3}, & \text { if } 0 \leq x<1, \\ x, & \text { if } x \geq 1\end{cases}
$$

is an approximation for $m$ with continuous second derivative.

In some applications it can be necessary to have an approximation of the maximum function with differentiable second derivative. The following function is a possible example for this case.

Example 4.4.3 (approximation using a piecewise polynomial of degree 6) Let $h: \mathbb{R} \rightarrow \mathbb{R}_{+}$be a piecewise polynomial function defined by

$$
h(x)= \begin{cases}\frac{15}{16}\left(x^{2}-1\right)^{2}, & \text { if }-1 \leq x \leq 1 \\ 0, & \text { otherwise }\end{cases}
$$

Then, the function $\widehat{m}$ defined by

$$
\widehat{m}(x)= \begin{cases}0, & \text { if } x<-1 \\ \frac{1}{32}\left(x^{6}-5 x^{4}+15 x^{2}+16 x+5\right), & \text { if }-1 \leq x<1 \\ x, & \text { if } x \geq 1\end{cases}
$$

is an approximation for $m$ with differentiable second derivative.

Finally, we propose a third example for an approximation of the maximum function which is rather easy to compute but does not possess a continuous second derivative. The latter fact implies that its applicability for practical methods may be restricted. Nevertheless, its simple form is very attractive from a computational point of view.

Example 4.4.4 (approximation using a piecewise polynomial of degree 2) Let $h: \mathbb{R} \rightarrow \mathbb{R}_{+}$be a piecewise constant function defined by

$$
h(x)= \begin{cases}1, & \text { if }-1 \leq x \leq 1 \\ 0, & \text { otherwise }\end{cases}
$$

Then, the function $\widehat{m}$ defined by

$$
\widehat{m}(x)= \begin{cases}0, & \text { if } x<-1 \\ \frac{1}{4}(1+x)^{2}, & \text { if }-1 \leq x<1 \\ x, & \text { if } x \geq 1\end{cases}
$$

is an approximation for $m$ which is continuously differentiable but not twice differentiable.

### 4.4.3 Convex Conjugate of Approximations

The basic property of all proposed approximations for the maximum function is that they are differentiable. This means that they are also essentially smooth (see Definition A.5.2). Thus, Theorem A.5.3 implies that the corresponding convex conjugate functions are essentially strictly convex.

Consequently, the dual training problem has a unique solution if all loss functions in the primal training problem are defined with the help of differentiable approximations
of the maximum function. This property might be helpful for practical applications, in particular in the construction of dual training methods. Additionally, the special definition of the approximation makes it possible to compute the convex conjugate as follows.

Proposition 4.4.5 (convex conjugate of approximations of the maximum function) Let $\widehat{m}$ be an approximation of the maximum function according to Proposition 4.4.1 and suppose that $h$ is strictly positive on the interval $(-\delta, \delta)$ and zero everywhere else. Then, the convex conjugate of this function is given by

$$
\widehat{m}^{\star}(\alpha)= \begin{cases}\alpha\left(\widehat{m}^{\prime}\right)^{-1}(\alpha)-\widehat{m}\left(\left(\widehat{m}^{\prime}\right)^{-1}(\alpha)\right), & \text { if } \alpha \in[0,1]  \tag{4.4.9}\\ \infty, & \text { otherwise } .\end{cases}
$$

Proof. First, note that for $\alpha \notin[0,1]$ the conjugate function is infinite because from the equations in (4.4.4) it follows that

$$
\widehat{m}^{\star}(\alpha)=\sup _{x \in \mathbb{R}}\{\alpha x-\widehat{m}(x)\}=\lim _{x \rightarrow-\infty}(\alpha x-\widehat{m}(x))=\lim _{x \rightarrow-\infty} \alpha x-\lim _{x \rightarrow-\infty} \widehat{m}(x)=\infty
$$

if $\alpha<0$, and

$$
\widehat{m}^{\star}(\alpha)=\sup _{x \in \mathbb{R}}\{\alpha x-\widehat{m}(x)\}=\lim _{x \rightarrow \infty}(\alpha x-\widehat{m}(x))=\lim _{x \rightarrow \infty}(\alpha-1) x+\lim _{x \rightarrow \infty}(x-\widehat{m}(x))=\infty
$$

if $\alpha>1$. At the boundary points of the interval (i.e., for $\alpha=0$ and $\alpha=1$ ) using (4.4.4) again we obtain

$$
\widehat{m}^{\star}(0)=\sup _{x \in \mathbb{R}}\{-\widehat{m}(x)\}=\lim _{x \rightarrow-\infty}(-\widehat{m}(x))=0
$$

and

$$
\widehat{m}^{\star}(1)=\sup _{x \in \mathbb{R}}\{x-\widehat{m}(x)\}=\lim _{x \rightarrow \infty}(x-\widehat{m}(x))=0 .
$$

For $\alpha \in(0,1)$, we can use that the supremum in the definition is attained at a point $\bar{x} \in[-\delta, \delta]$ which satisfies $\alpha \in \partial \widehat{m}(\bar{x})$. Since we consider differentiable functions, this is equivalent to $\alpha=\widehat{m}^{\prime}(\bar{x})$.

In the particular case when the defining function $h$ is strictly positive on the interval $(-\delta, \delta)$, the function $\widehat{m}^{\prime}$ is strictly increasing there and hence can be inverted. Then, the point $\bar{x}$ is given by $\bar{x}=\left(\widehat{m}^{\prime}\right)^{-1}(\alpha)$. Taking all parts together, we see that the conjugate function is equal to (4.4.9).

For sake of completeness, we use the previous result to obtain the convex conjugate function for the two easier examples introduced in the previous subsection. Note that the results are only listed briefly here and any longer calculation is omitted for brevity. We start with the simpler approximation, which uses a piecewise quadratic function.

Example 4.4.6 (continuation of Example 4.4.4)
Let the function $\widehat{m}$ be defined according to Example 4.4.4.
Then, the derivative is given by

$$
\widehat{m}^{\prime}(x)= \begin{cases}0, & \text { if } x<-1 \\ \frac{1}{2}(1+x), & \text { if }-1 \leq x<1, \\ 1, & \text { if } x \geq 1\end{cases}
$$

and its convex conjugate can be computed by means of (4.4.9) as

$$
\widehat{m}^{\star}(\alpha)= \begin{cases}\alpha(\alpha-1), & \text { if } \alpha \in[0,1] \\ \infty, & \text { if } \alpha \notin[0,1]\end{cases}
$$

It should be emphasized that the form of the conjugate function in the previous example is convenient for the formulation of the dual problem since it is simply quadratic on its domain, which is rather easy to handle in practice. Especially, this means that the general structure of the dual problem does not change if the loss functions in the primal training problem are approximated in this way and well-known approaches for the solution of the dual problem are still applicable.

For the following example using a piecewise cubic function for the approximation of the maximum function the calculations are a bit more elaborate but nevertheless straightforward.

Example 4.4.7 (continuation of Example 4.4.2)
Let the function $\widehat{m}$ be defined according to Example 4.4.2.
Then, the derivative is given by

$$
\widehat{m}^{\prime}(x)= \begin{cases}0, & \text { if } x<-1 \\ \frac{1}{2}(1+x)^{2}, & \text { if }-1 \leq x<0 \\ 1-\frac{1}{2}(1-x)^{2}, & \text { if } 0 \leq x<1 \\ 1, & \text { if } x \geq 1\end{cases}
$$

and its convex conjugate can be computed by means of (4.4.9) as

$$
\widehat{m}^{\star}(\alpha)= \begin{cases}\alpha\left(\frac{2 \sqrt{2}}{3} \sqrt{\alpha}-1\right) & \text { if } \alpha \in\left[0, \frac{1}{2}\right] \\ (1-\alpha)\left(\frac{2 \sqrt{2}}{3} \sqrt{1-\alpha}-1\right) & \text { if } \alpha \in\left(\frac{1}{2}, 1\right] \\ \infty, & \text { if } \alpha \notin[0,1]\end{cases}
$$

Finally, we note briefly that a scaling of the approximated maximum function leads to a particular scaling of its conjugate. Of course, this observation is also true for general convex function.

Observation 4.4.8 (scaling of approximate maximum functions)
Let $\widehat{m}(x)$ be some approximation of the maximum function and $\delta>0$. Then,

$$
\widehat{m}_{\delta}(x):=\delta \widehat{m}\left(\delta^{-1} x\right)
$$

is a scaled version of this approximation with

$$
\widehat{m}_{\delta}^{\star}(\alpha)=\sup _{x \in \mathbb{R}}\left\{\alpha x-\widehat{m}_{\delta}(x)\right\}=\sup _{x \in \mathbb{R}}\left\{\alpha x-\delta \widehat{m}\left(\delta^{-1} x\right)\right\}=\sup _{y \in \mathbb{R}}\{\delta \alpha y-\delta \widehat{m}(y)\}=\delta \widehat{m}^{\star}(\alpha) .
$$

For the functions considered in the previous examples, the scaling by some parameter $\delta \in(0,1)$ leads to a reduction of the area, where the function is nonlinear while keeping its characteristic shape. In particular, the original maximum function is obtained in the limit $\delta \rightarrow 0$.

### 4.5 Dual View on Smoothed Optimality Conditions

Instead of the change of the loss functions in the primal problem, it is also possible to consider a modification of the dual problem in order to realize certain properties of the solution. One example is the regularization approach which we discuss in the following. For illustration purposes, we focus on a single practical training problem but mention that the same idea can equally be applied to other problems. We consider the dual Support Vector Classification problem (4.1.1) in the slightly modified formulation

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha}} & \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}^{\top} \boldsymbol{\alpha}  \tag{4.5.1}\\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0 \quad \text { and } \quad \mathbf{0} \leq \boldsymbol{Y} \boldsymbol{\alpha} \leq \mathbf{1}
\end{array}
$$

where $\boldsymbol{Y}:=\operatorname{diag}\left(y_{1}, \ldots, y_{n}\right)$ denotes the diagonal matrix consisting of the labels. For this problem, the necessary and sufficient optimality conditions can be written as

$$
\begin{align*}
\frac{1}{\lambda} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}+b \mathbf{1} & =\boldsymbol{u}-\boldsymbol{v}, \\
\mathbf{1}^{\top} \boldsymbol{\alpha} & =0,  \tag{4.5.2}\\
\mathbf{0} \leq \boldsymbol{Y} \boldsymbol{\alpha} \perp \boldsymbol{u} & \geq \mathbf{0}, \\
\mathbf{0} \leq \mathbf{1}-\boldsymbol{Y} \boldsymbol{\alpha} \perp \boldsymbol{v} & \geq \mathbf{0} .
\end{align*}
$$

On the other hand, the dual training problem (4.5.1) can also be treated by means of barrier function approach, which leads to a problem of the form

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha}} & \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}^{\top} \boldsymbol{\alpha}-\delta \sum_{i=1}^{n}\left(\log \left(y_{i} \alpha_{i}\right)+\log \left(1-y_{i} \alpha_{i}\right)\right)  \tag{4.5.3}\\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0 \quad \text { and } \quad \mathbf{0}<\boldsymbol{Y} \boldsymbol{\alpha}<\mathbf{1}
\end{array}
$$

with some parameter $\delta>0$. This problem is again convex such that necessary and sufficient conditions for optimality can be stated as

$$
\begin{align*}
\frac{1}{\lambda} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}+b \mathbf{1} & =\boldsymbol{u}-\boldsymbol{v} \\
\mathbf{1}^{\top} \boldsymbol{\alpha} & =0  \tag{4.5.4}\\
\boldsymbol{Y} \boldsymbol{\alpha}>\mathbf{0}, \boldsymbol{u}>\mathbf{0}, \boldsymbol{u} \circ \boldsymbol{Y} \boldsymbol{\alpha} & =\delta \mathbf{1}, \\
\mathbf{1}-\boldsymbol{Y} \boldsymbol{\alpha}>\mathbf{0}, \boldsymbol{v}>\mathbf{0}, \boldsymbol{v} \circ(\mathbf{1}-\boldsymbol{Y} \boldsymbol{\alpha}) & =\delta \mathbf{1},
\end{align*}
$$

where o denotes the component-wise product, and we introduced artificial variables $u_{i}=\delta\left(y_{i} \alpha_{i}\right)^{-1}$ and $v_{i}=\delta\left(1-y_{i} \alpha_{i}\right)^{-1}$ for $i \in\{1, \ldots, n\}$ to represent some terms occurring in the gradient of the objective function.

In this form it is also easily noticeable that the optimality system (4.5.4) of the problem with barrier function can be interpreted as a smoothed version of the associated system (4.5.2) for the original problem (4.5.1), where the complementarity constraint is replaced by an approximated complementarity condition. Of course, this correspondence is well-known in the context of barrier function approaches, cf. [NW06, Chapter 19].

At this point, we can take a step back and note that this problem can be seen as a dual formulation of a standard training problem as given in Corollary 3.4.1. The structure of the corresponding primal problem is summarized in the following proposition.

Proposition 4.5.1 (dual smoothing for Support Vector Classification)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in\{-1,1\}^{n}$ be a vector of labels for a binary classification problem. Let $\delta>0$ and $\lambda>0$ be fixed parameters. Consider the Support Vector Classification problem

$$
\begin{equation*}
\min _{\boldsymbol{w}, b} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} \widehat{m}_{\delta}\left(1-y_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right)\right) \tag{4.5.5}
\end{equation*}
$$

with $\widehat{m}_{\delta}(x):=\delta \widetilde{m}\left(\frac{x}{\delta}\right)$ and $\widetilde{m}: \mathbb{R} \rightarrow \mathbb{R}$ defined by

$$
\begin{equation*}
\widetilde{m}(x):=\frac{x}{2}\left(1+\frac{x}{2} \frac{1}{1+\sqrt{1+\left(\frac{x}{2}\right)^{2}}}\right)-\log \left(2\left(1+\sqrt{1+\left(\frac{x}{2}\right)^{2}}\right)\right) \tag{4.5.6}
\end{equation*}
$$

Then, the dual optimization problem associated with (4.5.5) is given by (4.5.3).

Proof. Instead of verifying the statement directly, we start from the smoothed dual problem (4.5.3) and take some steps to derive the associated primal problem. The derivation in this direction is much more insightful as it shows how the approximation of the loss function is obtained.

First, we observe that the problem (4.5.3) has the form of the dual problem for a
standard training problem (see Corollary 3.4.1) with

$$
\ell_{i}^{\star}(\alpha)= \begin{cases}\alpha y_{i}-\delta\left(\log \left(-y_{i} \alpha\right)+\log \left(1+y_{i} \alpha\right)\right), & \text { if } y_{i} \alpha \in(-1,0)  \tag{4.5.7}\\ \infty, & \text { otherwise }\end{cases}
$$

In accordance with Section 4.1 and Section 4.4, we expect that the loss functions have the form $\ell_{i}(t)=\widehat{m}_{\delta}\left(1-y_{i} t\right)$, where $\widehat{m}_{\delta}$ is some approximation of the maximum function $m(x)=\max \{0, x\}$ depending on the parameter $\delta$. A simple computation shows that

$$
\begin{aligned}
\ell_{i}^{\star}(\alpha) & =\sup _{t \in \mathbb{R}}\left\{\alpha t-\widehat{m}_{\delta}\left(1-y_{i} t\right)\right\} \\
& =\sup _{r \in \mathbb{R}}\left\{\alpha y_{i}(1-r)-\widehat{m}_{\delta}(r)\right\} \\
& =\alpha y_{i}+\sup _{r \in \mathbb{R}}\left\{r\left(-y_{i} \alpha\right)-\widehat{m}_{\delta}(r)\right\} \\
& =\alpha y_{i}+\widehat{m}_{\delta}^{\star}\left(-y_{i} \alpha\right) .
\end{aligned}
$$

This, together with equation (4.5.7), leads to

$$
\widehat{m}_{\delta}^{\star}(\beta)= \begin{cases}-\delta(\log (\beta)+\log (1-\beta)), & \text { if } \beta \in(0,1) \\ \infty, & \text { otherwise }\end{cases}
$$

Then, we can compute the function $\widehat{m}_{\delta}$ from

$$
\begin{equation*}
\widehat{m}_{\delta}(x)=\widehat{m}_{\delta}^{* *}(x)=\sup _{\beta \in(0,1)}\{\beta x+\delta(\log (\beta)+\log (1-\beta))\} . \tag{4.5.8}
\end{equation*}
$$

Because the term in the supremum tends to minus infinity for $\beta \searrow 0$ and $\beta \nearrow$ 1, the supremum is attained at the unique point $\beta$ where the derivative vanishes, i.e., the point satisfying

$$
\frac{x}{\delta}+\frac{1}{\beta}-\frac{1}{1-\beta}=0
$$

For $x=0$, the solution is given by $\beta=\frac{1}{2}$. Otherwise, the equation is equivalent to

$$
\begin{aligned}
& \beta(\beta-1)+\frac{\delta}{x}(2 \beta-1) \\
= & \beta^{2}-2 \beta\left(\frac{1}{2}-\frac{\delta}{x}\right)-\frac{\delta}{x} \\
= & \left(\beta-\left(\frac{1}{2}-\frac{\delta}{x}\right)\right)^{2}-\frac{\delta}{x}-\left(\frac{1}{2}-\frac{\delta}{x}\right)^{2}=0 .
\end{aligned}
$$

One can check that the unique solution of this equation in the interval $(0,1)$ is equal
to

$$
\begin{aligned}
\beta & =\frac{1}{2}-\frac{\delta}{x}+\operatorname{sign}(x) \sqrt{\left(\frac{1}{2}-\frac{\delta}{x}\right)^{2}+\frac{\delta}{x}} \\
& =\frac{1}{2}\left(1-\frac{2 \delta}{x}+\operatorname{sign}(x) \sqrt{1+\left(\frac{2 \delta}{x}\right)^{2}}\right) \\
& =\frac{1}{2}\left(1+\frac{1}{x}\left(\sqrt{x^{2}+(2 \delta)^{2}}-2 \delta\right)\right) \\
& =\frac{1}{2}\left(1+\sigma\left(\frac{x}{2 \delta}\right)\right)
\end{aligned}
$$

with

$$
\sigma(x):=x\left(\sqrt{1+x^{2}}-1\right)=\frac{x}{1+\sqrt{1+x^{2}}}
$$

Moreover,

$$
1-\sigma^{2}(x)=\frac{1+2 \sqrt{1+x^{2}}+1+x^{2}-x^{2}}{\left(1+\sqrt{1+x^{2}}\right)^{2}}=\frac{2}{1+\sqrt{1+x^{2}}} .
$$

This implies that

$$
\beta(1-\beta)=\frac{1}{4}\left(1+\sigma\left(\frac{x}{2 \delta}\right)\right)\left(1-\sigma\left(\frac{x}{2 \delta}\right)\right)=\frac{1}{4}\left(1-\sigma^{2}\left(\frac{x}{2 \delta}\right)\right)=\frac{1}{2} \cdot \frac{1}{1+\sqrt{1+\left(\frac{x}{2 \delta}\right)^{2}}} .
$$

Plugging this term into (4.5.8), we obtain $\widehat{m}_{\delta}(x)=\delta \widetilde{m}\left(\frac{x}{\delta}\right)$ with $\widetilde{m}$ as defined in (4.5.6).

A visual presentation of the resulting loss function is given in Figure 4.5.1. This example shows that the application of a barrier approach (or equivalently a smoothing of the KKT system) can actually be interpreted as a certain smoothing of the loss functions in the primal training problem.


Figure 4.1.1: Examples of loss functions for classification problems together with their conjugate functions and subdifferentials: $y_{i}=+1$ in blue and $y_{i}=-1$ in red


Figure 4.2.1: Examples of loss functions for regression problems together with their conjugate functions and subdifferentials with labels $y_{i}=1$


Figure 4.4.1: Approximations for the maximum function $m$





Figure 4.5.1: Regularized classification loss functions for $\tau=\frac{1}{10}$ together with the conjugate function and subdifferential: $y_{i}=+1$ in blue and $y_{i}=-1$ in red, cf. Figure 4.1.1a

## 5 Examples of Non-Standard Training Problems

The examples of the previous chapter followed the common structure of a standard training problem. In the present chapter, we aim to apply the duality framework to problems which are not as typical. In particular, three different classes of training problems are investigated which deviate from the standard training problem either because of a different loss structure or a change in the regularization term.

We start with the derivation of dual formulations for multi-class classification problems (Section 5.1) which goes beyond what is known from the literature. Afterwards, we use duality theory to gain insight into the structure of $\nu$-SVMs (Section 5.2). Finally, the same theory is applied to Subspace SVMs in order to derive a relationship to classical SVMs (Section 5.3).

### 5.1 Multi-Class Classification Problems

In the case of multi-class classification the output value $y_{i}$ assigned with a given input point $\boldsymbol{x}_{i}$ can have one of several values $v_{j} \in \mathcal{Y}:=\left\{v_{1}, \ldots, v_{r}\right\}$ which have no specific order in general. Hence, the loss functions for binary classification and regression problem cannot be applied directly without further adaptation. Of course, as mentioned briefly in Section 2.1, multi-class classification problems can be reduced to a set of binary classification problem. However, this is only practical if the number is classes is not too high.

A fundamentally different way to handle multi-class classification problems is to define a problem specific loss function. An idea that is typically used for the construction of artificial neural networks is based on the soft-max loss function [Bis06, Section 4.2]. It is a generalization of the logistic regression loss function which is also frequently applied for modeling binary classification problems. This loss function can be translated to fit into the general training problem structure as follows.

First, we assume without loss of generality that $\mathcal{Y}=\{1, \ldots, r\}$ for some $r \in \mathbb{N}$. Then, we consider a separable loss functional with $\mathcal{D}_{i}:=\mathbb{R}^{r}$ for $i \in\{1, \ldots, n\}$, and we suppose that $\boldsymbol{T}$ is defined in a way that $\boldsymbol{t}_{i}=[\boldsymbol{T} \boldsymbol{\omega}]_{i}$ is a real-valued vector. Each component of the vector $\boldsymbol{t}_{i}$ corresponds to exactly one class in $\mathcal{Y}$. This means that the
training problem consists of finding a vector $\boldsymbol{\omega} \in \mathcal{H}$ such that the largest value in the vector $\boldsymbol{t}_{i}$ is at the position which corresponds to the class of the output value $y_{i}$ for each $i \in\{1, \ldots, n\}$. For instance, the value of

$$
\frac{\exp \left(t_{i, y_{i}}\right)}{\sum_{k=1}^{r} \exp \left(t_{i, k}\right)}
$$

can be interpreted as the probability that the $i$ th input point belongs to the class $y_{i}$. Thus, a suitable training problem would be to find a decision function that maximizes the likelihood of the correct class prediction. Equivalently, the logarithm of the likelihood can be maximized. The resulting term is also called categorical cross-entropy.

Suppose that a feature vector $\boldsymbol{\varphi}_{i} \in \mathcal{F}$ is given for each data point. Then, a natural choice of a decision operator would be

$$
\begin{align*}
\boldsymbol{T}: & (\mathcal{F} \times \mathbb{R})^{r} \rightarrow  \tag{5.1.1}\\
& \left(\mathbb{R}^{r}\right)^{n}, \\
\quad\left(\boldsymbol{w}_{k}, b_{k}\right)_{k=1}^{r} & \mapsto\left(\left(\left\langle\boldsymbol{w}_{k}, \boldsymbol{\varphi}_{i}\right\rangle+b_{k}\right)_{k=1}^{r}\right)_{i=1}^{n},
\end{align*}
$$

which is basically a concatenation of $r$ standard decision operators. Then, the notion of standard regularization can be naturally extended for this operator. Hence, we obtain a formulation of a multi-class Support Vector Machine as follows.

Proposition 5.1.1 (multi-class SVM)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in\{1, \ldots, r\}^{n}$ be a vector of labels for a multi-class classification problem. The dual problem of the multi-class Support Vector Machine

$$
\min _{\left(\boldsymbol{w}_{k}, b_{k}\right)_{k=1}^{r}} \frac{\lambda}{2} \sum_{k=1}^{r}\left\|\boldsymbol{w}_{k}\right\|^{2}+\sum_{i=1}^{n} \log \left(\sum_{k=1}^{r} \exp \left(\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{w}_{k}\right\rangle+b_{k}\right)\right)-\left(\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{w}_{y_{i}}\right\rangle+b_{y_{i}}\right)
$$

is given by

$$
\begin{aligned}
\min _{\boldsymbol{\alpha} \in\left(\mathbb{R}^{n}\right)^{r}} & \frac{1}{2 \lambda} \sum_{k=1}^{r} \boldsymbol{\alpha}_{k}^{\top} \boldsymbol{K} \boldsymbol{\alpha}_{k}+\sum_{i=1}^{n} \sum_{k=1}^{r} q\left(\delta_{k y_{i}}-\left[\boldsymbol{\alpha}_{k}\right]_{i}\right) \\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}_{k}=0 \text { for all } k \in\{1, \ldots, r\}, \\
& \mathbf{1}^{\top} \boldsymbol{\alpha}^{i}=0, \boldsymbol{\alpha}^{i}<\boldsymbol{e}_{y_{i}} \quad \text { for all } i \in\{1, \ldots, n\},
\end{aligned}
$$

where the function $q$ is defined according to (4.1.3).

Proof. In order to state the dual problem using Corollary 3.1.3, we need to compute the adjoint of the decision operator (5.1.1) and the conjugate functions of the regularization functional

$$
\mathcal{R}(\boldsymbol{\omega})=\frac{\lambda}{2} \sum_{k=1}^{r}\left\|\boldsymbol{w}_{k}\right\|^{2}
$$

and of the loss functions

$$
\ell_{i}(\boldsymbol{t}):=\log \left(\sum_{k=1}^{r} \exp \left(t_{k}\right)\right)-t_{y_{i}}
$$

which define the separable loss functional.
Note that the dual space is $\mathcal{D}^{\star}=\left(\left(\mathbb{R}^{r}\right)^{n}\right)^{\star}=\left(\left(\mathbb{R}^{r}\right)^{\star}\right)^{n}=\left(\mathbb{R}^{r}\right)^{n}$, which can also be identified with $\left(\mathbb{R}^{n}\right)^{r}$. Hence, for some element $\boldsymbol{\alpha} \in \mathcal{D}^{\star}$ we write $\boldsymbol{\alpha}=\left(\boldsymbol{\alpha}_{k}\right)_{k=1}^{r} \in\left(\mathbb{R}^{n}\right)^{r}$ or $\boldsymbol{\alpha}=\left(\boldsymbol{\alpha}^{i}\right)_{i=1}^{n} \in\left(\mathbb{R}^{r}\right)^{n}$ depending on which representation makes the notation easier.

The conjugate function of the regularization functional can be computed easily, because it decomposes into known parts. The dual space of the product space $\mathcal{H}=$ $(\mathcal{F} \times \mathbb{R})^{r}$ can be identified with the product of the corresponding dual spaces. Moreover, the product spaces of the Hilbert spaces $\mathcal{F}$ and $\mathbb{R}$ can be identified with their selves, i.e., $\mathcal{H}^{\star}=(\mathcal{F} \times \mathbb{R})^{r}$. This means, we can write $\boldsymbol{\omega}^{\star}=\left(\boldsymbol{w}_{k}^{\star}, b_{k}^{\star}\right)_{k=1}^{r} \in \mathcal{H}^{\star}$. It follows that

$$
\begin{align*}
\mathcal{R}^{\star}\left(\boldsymbol{\omega}^{\star}\right) & =\sup _{\boldsymbol{\omega} \in \mathcal{H}}\left\{\left\langle\boldsymbol{\omega}^{\star}, \boldsymbol{\omega}\right\rangle-\mathcal{R}(\boldsymbol{\omega})\right\} \\
& =\sup _{\boldsymbol{\omega} \in \mathcal{H}}\left\{\sum_{k=1}^{r}\left(\left\langle\boldsymbol{w}_{k}^{\star}, \boldsymbol{w}_{k}\right\rangle+b_{k}^{\star} b_{k}\right)-\frac{\lambda}{2} \sum_{k=1}^{r}\left\|\boldsymbol{w}_{k}\right\|^{2}\right\} \\
& =\sum_{k=1}^{r} \sup _{\left(\boldsymbol{w}_{k}, b_{k}\right) \in \mathcal{F} \times \mathbb{R}}\left\{\left\langle\boldsymbol{w}_{k}^{\star}, \boldsymbol{w}_{k}\right\rangle+b_{k}^{\star} b_{k}-\frac{\lambda}{2}\left\|\boldsymbol{w}_{k}\right\|^{2}\right\}  \tag{5.1.2}\\
& = \begin{cases}\frac{1}{2 \lambda} \sum_{k=1}^{r}\left\|\boldsymbol{w}_{k}\right\|^{2}, & \text { if } b_{1}^{\star}=\cdots=b_{r}^{\star}=0, \\
\infty, & \text { otherwise, }\end{cases}
\end{align*}
$$

where the last equality follows as in the proof of Proposition 3.3.2.
In order to compute the adjoint of $\boldsymbol{T}$, we identify the dual space of $\mathcal{D}=\left(\mathbb{R}^{r}\right)^{n}$ with the space $\left(\mathbb{R}^{n}\right)^{r}$. Then, we can write

$$
\begin{align*}
& \boldsymbol{T}^{\star}:\left(\mathbb{R}^{n}\right)^{r} \rightarrow(\mathcal{F} \times \mathbb{R})^{r}, \\
& \boldsymbol{\alpha} \mapsto \boldsymbol{T}^{\star} \boldsymbol{\alpha}=\left(\sum_{i=1}^{n}\left[\boldsymbol{\alpha}_{k}\right]_{i} \boldsymbol{\varphi}_{i}, \mathbf{1}^{\top} \boldsymbol{\alpha}_{k}\right)_{k=1}^{r} \tag{5.1.3}
\end{align*}
$$

using the same computation as in the proof of Proposition 3.3.1.
The final part is the computation of $\ell_{i}^{\star}$ in order to obtain $\mathcal{L}^{\star}$ by means of Proposition 3.2.1. Since the loss function is continuously differentiable, we can compute the gradient

$$
\nabla \ell_{i}(\boldsymbol{t})=\left(\frac{\exp \left(t_{l}\right)}{\sum_{k=1}^{r} \exp \left(t_{k}\right)}\right)_{l=1}^{r}-\boldsymbol{e}_{y_{i}}=\left(\sum_{k=1}^{r} \exp \left(t_{k}\right)\right)^{-1}\left(\exp \left(t_{l}\right)\right)_{l=1}^{r}-\boldsymbol{e}_{y_{i}}
$$

Then, the supremum in the definition

$$
\ell_{i}^{\star}(\boldsymbol{\beta})=\sup _{\boldsymbol{t} \in \mathbb{R}^{r}}\left\{\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle-\ell_{i}(\boldsymbol{t})\right\}
$$

is attained at a point $\boldsymbol{t}$ if and only if $\boldsymbol{\beta}=\nabla \ell_{i}(\boldsymbol{t})$. By summing up the entries of $\boldsymbol{\beta}$ we obtain the necessary condition

$$
\mathbf{1}^{\top} \boldsymbol{\beta}=\mathbf{1}^{\top} \nabla \ell_{i}(\boldsymbol{t})=\left(\sum_{k=1}^{r} \exp \left(t_{k}\right)\right)^{-1} \sum_{l=1}^{r} \exp \left(t_{l}\right)-1=0 .
$$

Moreover, for each $l \in\{1, \ldots, r\}$ it must hold that $\beta_{l}>-\delta_{l y_{i}}$ and

$$
\begin{equation*}
\log \left(\beta_{l}+\delta_{l y_{i}}\right)=t_{l}-\log \left(\sum_{k=1}^{r} \exp \left(t_{k}\right)\right) \tag{5.1.4}
\end{equation*}
$$

in this case. Hence, we get

$$
\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle=\sum_{l=1}^{r} \beta_{l} \log \left(\beta_{l}+\delta_{l y_{i}}\right)+\log \left(\sum_{k=1}^{r} \exp \left(t_{k}\right)\right) \mathbf{1}^{\top} \boldsymbol{\beta}=\sum_{l=1}^{r} \beta_{l} \log \left(\beta_{l}+\delta_{l y_{i}}\right) .
$$

Finally, note that by means of (5.1.4) for $l=y_{i}$ it follows that

$$
\log \left(\beta_{y_{i}}+1\right)=t_{y_{i}}-\log \left(\sum_{k=1}^{r} \exp \left(t_{k}\right)\right)=-\ell_{i}(\boldsymbol{t})
$$

This shows that

$$
\begin{aligned}
\ell_{i}^{\star}(\boldsymbol{\beta}) & =\sum_{k=1}^{r} \beta_{k} \log \left(\beta_{k}+\delta_{k y_{i}}\right)+\log \left(\beta_{y_{i}}+1\right) \\
& =\sum_{k=1}^{r}\left(\beta_{k} \log \left(\beta_{k}+\delta_{k y_{i}}\right)+\delta_{k y_{i}} \log \left(\beta_{k}+\delta_{k y_{i}}\right)\right) \\
& =\sum_{k=1}^{r}\left(\beta_{k}+\delta_{k y_{i}}\right) \log \left(\beta_{k}+\delta_{k y_{i}}\right) \\
& =\sum_{k=1}^{r} q\left(\beta_{k}+\delta_{k y_{i}}\right)
\end{aligned}
$$

for $\boldsymbol{\beta} \in \mathbb{R}^{r}$ with $\mathbf{1}^{\top} \boldsymbol{\beta}=0$ and $\boldsymbol{\beta}>-\boldsymbol{e}_{y_{i}}$ for all $k \in\{1, \ldots, r\}$. If $\boldsymbol{\beta}=-\boldsymbol{e}_{y_{i}}$,

$$
\ell_{i}^{\star}(\boldsymbol{\beta})=\sup _{t \in \mathbb{R}^{r}}\left\{-t_{y_{i}}-\ell_{i}(\boldsymbol{t})\right\}=\sup _{t \in \mathbb{R}^{r}}\left\{-\log \left(\sum_{k=1}^{r} \exp \left(t_{k}\right)\right)\right\}=\infty .
$$

Finally, the dual problem is given by means of Corollary 3.1.3.
In the following, we consider another approach to treat multi-class classification problems using Support Vector Machines. Suppose that the vector of decisions is normalized in the sense that $\mathbf{1}^{\top} \boldsymbol{t}_{i}=1$ and $0 \leq t_{i, k} \leq 1$ for all $k \in\{1, \ldots, r\}$ and all $i \in\{1, \ldots, n\}$. Then, the values can be interpreted as probabilities and the probability of right predictions is equal to

$$
\prod_{i=1}^{n} t_{i, y_{i}} .
$$

This leads to an alternative formulation of a multi-class Support Vector Machines which is presented in the following proposition.

Proposition 5.1.2 (simplified multi-class SVM)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors and let $\boldsymbol{y} \in\{1, \ldots, r\}^{n}$ be a vector of labels for a multi-class classification problem. The dual problem of the simplified multi-class Support Vector Machine

$$
\begin{aligned}
\min _{\left(\boldsymbol{w}_{k}, b_{k}\right)_{k=1}^{r}} & \frac{\lambda}{2} \sum_{k=1}^{r}\left\|\boldsymbol{w}_{k}\right\|^{2}-\sum_{i=1}^{n} \log \left(\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{w}_{y_{i}}\right\rangle+b_{y_{i}}\right) \\
\text { s.t. } & \sum_{k=1}^{r}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{w}_{k}\right\rangle+b_{k}=1 \quad \text { for all } \quad i \in\{1, \ldots, n\}, \\
& \left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{w}_{k}\right\rangle+b_{k} \geq 0 \quad \text { for all } \quad k \in\{1, \ldots, r\} \text { and } i \in\{1, \ldots, n\}
\end{aligned}
$$

is given by

$$
\begin{equation*}
\min _{\boldsymbol{\alpha} \in\left(\mathbb{R}^{n}\right)^{r}} \frac{1}{2 \lambda} \sum_{k=1}^{r} \boldsymbol{\alpha}_{k}^{\top} \boldsymbol{K} \boldsymbol{\alpha}_{k}+\sum_{i=1}^{n} \sum_{k=1}^{r} w\left(\left[\boldsymbol{\alpha}^{i}\right]_{y_{i}}-\min \boldsymbol{\alpha}^{i}\right)-\left[\boldsymbol{\alpha}^{i}\right]_{y_{i}}, \tag{5.1.5}
\end{equation*}
$$

where $w(x):=x-(1+\log (x))$ for $x \geq 1$ and $w(x):=0$ otherwise.
Proof. The computation of the conjugate function associated with the regularization term has already been conducted in the proof of Proposition 5.1.1. Thus, we only need to compute the conjugate of the loss functions

$$
\ell_{i}(\boldsymbol{t}):=-\log \left(t_{y_{i}}\right)+ \begin{cases}0, & \text { if } \mathbf{1}^{\top} \boldsymbol{t}=1 \text { and } \boldsymbol{t} \geq 0 \\ \infty, & \text { otherwise }\end{cases}
$$

of the separable loss functional. By definition, the conjugate of $\ell_{i}$ is given by

$$
\ell_{i}^{\star}(\boldsymbol{\beta})=\sup _{t \in \mathbb{R}^{r}}\left\{\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle-\ell_{i}(\boldsymbol{t})\right\}=\sup _{\substack{t \geq 0 \\ \mathbf{1}^{\top} \boldsymbol{t}=1}}\left\{\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle+\log \left(t_{y_{i}}\right)\right\}
$$

This means that we need to compute the optimal value of the optimization problem

$$
\begin{aligned}
& \max _{t \in \mathbb{R}^{r}}\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle+\log \left(t_{y_{i}}\right) \\
& \text { s.t. } \quad \mathbf{1}^{\top} \boldsymbol{t}=1 \text { and } \boldsymbol{t} \geq 0
\end{aligned}
$$

to obtain the value of the conjugate loss function. Note that the problem has a continuous objective function and that the feasible set is non-empty and compact. This means that there exists always a solution of this problem. Moreover, the constraints

$$
\mathbf{1}^{\top} \boldsymbol{t}=t_{y_{i}}+\sum_{k \neq y_{i}} t_{k}=1
$$

and $\boldsymbol{t} \geq 0$ imply that $t_{y_{i}} \leq 1$ and hence $\log \left(t_{y_{i}}\right) \leq 0$. Thus,

$$
\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle+\log \left(t_{y_{i}}\right) \leq\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle \leq \max \left\{\beta_{k} \mid k \in\{1, \ldots, r\}\right\}=: \beta_{\max }
$$

for all feasible $\boldsymbol{t} \in \mathbb{R}^{r}$. Note that the maximum on the right-hand side of the inequality can be obtained if it is equal to $\beta_{y_{i}}$. Otherwise, $\delta:=\beta_{\max }-\beta_{y_{i}}>0$ and we can estimate the value of the objective function by

$$
\begin{align*}
\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle+\log \left(t_{y_{i}}\right) & =\sum_{k \neq y_{i}} \beta_{k} t_{k}+\beta_{y_{i}} t_{y_{i}}+\log \left(t_{y_{i}}\right) \\
& \leq \beta_{\max } \sum_{k \neq y_{i}} t_{k}+\left(\beta_{\max }-\delta\right) t_{y_{i}}+\log \left(t_{y_{i}}\right)  \tag{5.1.6}\\
& =\beta_{\max }-\delta t_{y_{i}}+\log \left(t_{y_{i}}\right) .
\end{align*}
$$

Subsequently, we consider two cases:
(a) If $\delta \leq 1$, it follows that $1-\delta \geq 0$ and hence

$$
\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle+\log \left(t_{y_{i}}\right) \leq \beta_{\max }+(1-\delta) t_{y_{i}}-1 \leq \beta_{\max }-\delta=\beta_{y_{i}}
$$

holds by means of (5.1.6), $\log \left(t_{y_{i}}\right) \leq t_{y_{i}}-1$, and $t_{y_{i}} \leq 1$. It is easy to see that this upper bound is attained at the point $\overline{\boldsymbol{t}}:=\boldsymbol{e}_{y_{i}}$.
(b) If $\delta>1$, we can use (5.1.6) again to derive

$$
\langle\boldsymbol{\beta}, \boldsymbol{t}\rangle+\log \left(t_{y_{i}}\right) \leq \beta_{\max }-1+\log \left(\frac{1}{\delta}\right) .
$$

Now, let $k \in \operatorname{argmax}\left\{\beta_{k} \mid k \in\{1, \ldots, r\}\right\}$. Then, the value on the right-hand side is attained at the feasible point $\overline{\boldsymbol{t}}:=\frac{1}{\delta} \boldsymbol{e}_{y_{i}}+\left(1-\frac{1}{\delta}\right) \boldsymbol{e}_{k}$ since

$$
\langle\boldsymbol{\beta}, \overline{\boldsymbol{t}}\rangle+\log \left(\bar{t}_{y_{i}}\right)=\beta_{k}\left(1-\frac{1}{\delta}\right)+\beta_{y_{i}} \cdot \frac{1}{\delta}+\log \left(\frac{1}{\delta}\right)=\beta_{\max }-1+\log \left(\frac{1}{\delta}\right) .
$$

Taking both cases together, it can be seen that

$$
\ell_{i}^{\star}(\boldsymbol{\beta})=\beta_{y_{i}}+ \begin{cases}0, & \text { if } \beta_{y_{i}} \geq \max \boldsymbol{\beta}-1 \\ \max \boldsymbol{\beta}-\beta_{y_{i}}-1-\log \left(\max \boldsymbol{\beta}-\beta_{y_{i}}\right), & \text { if } \beta_{y_{i}}<\max \boldsymbol{\beta}-1 .\end{cases}
$$

Finally, the dual problem stated in the proposition can be obtained by means of Proposition 3.2.1, equations (5.1.2) and (5.1.3), and, the definition in Corollary 3.1.3.

At first sight, the dual problem (5.1.5) does not seem to be tractable easily in practice. However, by means of the substitution $\min \boldsymbol{\alpha}^{i}=\eta_{i}$, it can be written equivalently as a constrained optimization problem

$$
\begin{aligned}
\min _{\left.\boldsymbol{\alpha} \in \mathbb{R}^{n}\right)^{r}} & \frac{1}{2 \lambda} \sum_{k=1}^{r} \boldsymbol{\alpha}_{k}^{\top} \boldsymbol{K} \boldsymbol{\alpha}_{k}+\sum_{i=1}^{n} \sum_{k=1}^{r} w\left(\left[\boldsymbol{\alpha}^{i}\right]_{y_{i}}-\eta_{i}\right)-\left[\boldsymbol{\alpha}^{i}\right]_{y_{i}} \\
\text { s.t. } & \boldsymbol{\alpha}^{i} \geq \eta_{i} \mathbf{1} \text { for all } i \in\{1, \ldots, n\}
\end{aligned}
$$

with a differentiable objective function.

## $5.2 \nu$-Support Vector Machines

Considering the practical importance of limiting the number of support vectors (as discussed in Section 3.6) and the corresponding behavior of some typical training problems presented in the previous chapter, one could try to incorporate an additional restriction into the training problem from the beginning. This is the aim of the present section and leads to the idea of $\nu$-SVMs, see [CL01, CL02]. The optimization problem which we derive here is not new, but the motivation and procedure of derivation differs significantly from that of the literature. In particular, we aim to embed the underlying idea into the primal-dual framework.

For this reason, reconsider the dual problem (3.4.1) associated with a standard training problem. In order to force the dual solution to be sparse, it would be reasonable to add the constraint $\|\boldsymbol{\alpha}\|_{0} \leq \nu n$ to the dual problem, where $\|\boldsymbol{\alpha}\|_{0}$ denotes the number of nonzero components of $\boldsymbol{\alpha}$ and $\nu \in[0,1]$ is the maximum portion of support vectors. However, this constraint is not convex and hence would not help if we want to apply convex duality theory. To circumvent this problem, a common approach is to replace the term $\|\cdot\|_{0}$ by an appropriate approximation. The closest convex approximation (in a particular sense) for this function is the absolute norm $\|\cdot\|_{1}$. Thus, we follow up the lines and examine what happens if the constraint $\|\boldsymbol{\alpha}\|_{1} \leq \nu n$ is added to the dual training problem.

Subsequently, we want to explain the details of possible primal problems coming from the introduction of the additional constraint rather than just proposing a particular problem. For this reason, our aim is to identify each part of the particular dual training problem with a suitable term in the general dual training problem (3.1.1). We start by noting that the conjugate of the loss functional already matches that of a separable loss functional given by Proposition 3.2.1, which means that it is not necessary to modify the loss functional in the primal training problem.

On the other hand, both constraints, $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ and $\|\boldsymbol{\alpha}\|_{1} \leq \nu n$, can be incorporated into the regularization functional if the decision operator is defined in the right way. To see this, consider the dual regularization term of the form

$$
\mathcal{R}^{\star}\left(\boldsymbol{T}^{\star} \boldsymbol{\alpha}\right)= \begin{cases}\frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}, & \text { if } \mathbf{1}^{\top} \boldsymbol{\alpha}=0 \text { and }\|\boldsymbol{\alpha}\|_{1} \leq \nu n,  \tag{5.2.1}\\ \infty, & \text { otherwise } .\end{cases}
$$

One possible choice of parts in the primal training problem that match this definition is summarized in the following proposition.

Proposition 5.2.1 (first characterization of $\nu$-SVM)
Suppose that $\left\{\boldsymbol{\varphi}_{i}\right\}_{i=1}^{n} \subseteq \mathcal{F}$ are given feature vectors. Let $\mathcal{H}=\mathcal{F} \times \mathbb{R}^{n}$. Suppose that the decision operator is defined by

$$
\begin{equation*}
\boldsymbol{T}(\boldsymbol{w}, \boldsymbol{\beta}):=\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+\beta_{i}\right)_{i=1}^{n} \tag{5.2.2}
\end{equation*}
$$

and the regularization operator is defined by

$$
\begin{equation*}
\mathcal{R}(\boldsymbol{w}, \boldsymbol{\beta}):=\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\frac{\nu n}{2}\left(\max _{i=1}^{n} \beta_{i}-\min _{i=1}^{n} \beta_{i}\right) . \tag{5.2.3}
\end{equation*}
$$

Then, equation (5.2.1) is satisfied.
If $\mathcal{L}$ is a separable loss functional, a primal formulation of the $\nu$-SVM is given by

$$
\begin{equation*}
\min _{\boldsymbol{w}, \boldsymbol{\beta}} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+\beta_{i}\right)+\frac{\nu n}{2}\left(\max _{i=1}^{n} \beta_{i}-\min _{i=1}^{n} \beta_{i}\right) . \tag{5.2.4}
\end{equation*}
$$

Proof. First, we easily observe that

$$
\left\langle(\boldsymbol{w}, \boldsymbol{\beta}), \boldsymbol{T}^{\star} \boldsymbol{\alpha}\right\rangle=\sum_{i=1}^{n} \alpha_{i}\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+\langle\boldsymbol{\beta}, \boldsymbol{\alpha}\rangle=\left\langle\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle\right\rangle_{i=1}^{n}+\boldsymbol{\beta}, \boldsymbol{\alpha}\right\rangle=\langle\boldsymbol{T}(\boldsymbol{w}, \boldsymbol{\beta}), \boldsymbol{\alpha}\rangle,
$$

i.e., the adjoint decision operator is given by

$$
\begin{equation*}
\boldsymbol{T}^{\star} \boldsymbol{\alpha}=\left(\sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}, \boldsymbol{\alpha}\right) . \tag{5.2.5}
\end{equation*}
$$

In the next step, we show that the conjugate of the proposed regularization operator equal to

$$
\widetilde{\mathcal{R}}\left(\boldsymbol{w}^{\star}, \boldsymbol{\beta}^{\star}\right):= \begin{cases}\frac{1}{2 \lambda}\left\|\boldsymbol{w}^{\star}\right\|^{2}, & \text { if } \mathbf{1}^{\top} \boldsymbol{\beta}^{\star}=0 \text { and }\left\|\boldsymbol{\beta}^{\star}\right\|_{1} \leq \nu n  \tag{5.2.6}\\ \infty, & \text { otherwise }\end{cases}
$$

For this reason, note that the $\mathcal{R}$ is convex and lower semicontinuous. This means that $\mathcal{R}=\mathcal{R}^{* *}$, i.e., it is sufficient to show that the conjugate of $\widetilde{\mathcal{R}}$ is equal to $\mathcal{R}$ in order to prove that $\widetilde{\mathcal{R}}$ is actually the conjugate of $\mathcal{R}$. A straightforward calculation shows that

$$
\begin{aligned}
\widetilde{\mathcal{R}}^{\star}(\boldsymbol{w}, \boldsymbol{\beta}) & =\sup _{\boldsymbol{w}^{\star}, \boldsymbol{\beta}^{\star}}\left\{\left\langle(\boldsymbol{w}, \boldsymbol{\beta}),\left(\boldsymbol{w}^{\star}, \boldsymbol{\beta}^{\star}\right)\right\rangle-\widetilde{\mathcal{R}}\left(\boldsymbol{w}^{\star}, \boldsymbol{\beta}^{\star}\right)\right\} \\
& =\sup _{\boldsymbol{w}^{\star}}\left\{\left\langle\boldsymbol{w}, \boldsymbol{w}^{\star}\right\rangle-\frac{1}{2 \lambda}\left\|\boldsymbol{w}^{\star}\right\|^{2}\right\}+\sup _{\boldsymbol{\beta}^{\star}: 1^{\top} \boldsymbol{\beta}^{\star}=0,\left\|\boldsymbol{\beta}^{\star}\right\|_{1} \leq \nu n}\left\{\left\langle\boldsymbol{\beta}, \boldsymbol{\beta}^{\star}\right\rangle\right\} .
\end{aligned}
$$

Here, the first term is equal to $\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}$, and it remains to consider the second term. We can see that

$$
\left\langle\boldsymbol{\beta}, \boldsymbol{\beta}^{\star}\right\rangle=\sum_{i: \beta_{i}^{\star} \geq 0} \beta_{i} \beta_{i}^{\star}+\sum_{i: \beta_{i}^{\star}<0} \beta_{i} \beta_{i}^{\star} \leq \max _{i=1}^{n} \beta_{i} \sum_{i: \beta_{i}^{\star} \geq 0} \beta_{i}^{\star}+\min _{i=1}^{n} \beta_{i} \sum_{i: \beta_{i}^{\star}<0} \beta_{i}^{\star} .
$$

Using the fact that

$$
\sum_{i: \beta_{i}^{\star} \geq 0} \beta_{i}^{\star}=\frac{1}{2}\left(\sum_{i: \beta_{i}^{\star} \geq 0} \beta_{i}^{\star}-\sum_{i: \beta_{i}^{\star}<0} \beta_{i}^{\star}\right)=\frac{1}{2}\left\|\boldsymbol{\beta}^{\star}\right\|_{1} \quad \text { and } \quad \sum_{i: \beta_{i}^{\star}<0} \beta_{i}^{\star}=-\frac{1}{2}\left\|\boldsymbol{\beta}^{\star}\right\|_{1}
$$

for all $\boldsymbol{\beta}^{\star}$ with $\mathbf{1}^{\top} \boldsymbol{\beta}^{\star}=0$ together with the inequality $\left\|\boldsymbol{\beta}^{\star}\right\|_{1} \leq \nu n$ we obtain

$$
\left\langle\boldsymbol{\beta}, \boldsymbol{\beta}^{\star}\right\rangle \leq\left(\max _{i=1}^{n} \beta_{i}-\min _{i=1}^{n} \beta_{i}\right) \frac{1}{2}\left\|\boldsymbol{\beta}^{\star}\right\|_{1} \leq \frac{\nu n}{2}\left(\max _{i=1}^{n} \beta_{i}-\min _{i=1}^{n} \beta_{i}\right) .
$$

On the other hand, the right-hand side of this inequality is attained at the point

$$
\widetilde{\boldsymbol{\beta}}^{\star}=\frac{\nu n}{2}\left(\boldsymbol{e}_{i}-\boldsymbol{e}_{j}\right) \quad \text { with } \quad i \in \operatorname{argmax}_{i=1}^{n} \beta_{i} \text { and } j \in \operatorname{argmin}_{i=1}^{n} \beta_{i} \text {. }
$$

Since this point also satisfies $\mathbf{1}^{\top} \widetilde{\boldsymbol{\beta}}^{\star}=0$ and $\left\|\widetilde{\boldsymbol{\beta}}^{\star}\right\|_{1} \leq \nu n$, it follows that

$$
\sup _{\boldsymbol{\beta}^{\star}: \mathbf{1}^{\top} \boldsymbol{\beta}^{\star}=0,\left\|\boldsymbol{\beta}^{\star}\right\|_{1} \leq \nu n}\left\{\left\langle\boldsymbol{\beta}, \boldsymbol{\beta}^{\star}\right\rangle\right\}=\frac{\nu n}{2}\left(\max _{i=1}^{n} \beta_{i}-\min _{i=1}^{n} \beta_{i}\right) .
$$

This shows that the conjugate of $\widetilde{\mathcal{R}}$ is actually the function $\mathcal{R}$ defined according (5.2.3), i.e., $\mathcal{R}^{\star}=\widetilde{\mathcal{R}}$. By taking together (5.2.5) and (5.2.6) we can see that the equality (5.2.1) holds indeed.

Formally, a primal training problem can be defined by using the definitions of $\boldsymbol{T}$ and $\mathcal{R}$ from Proposition 5.2 .1 together with a suitable separable loss functional. However, the interpretation of the decision operator in (5.2.2) is a bit problematic because the decision has a sample-dependent offset $\beta_{i}$ instead of the usual bias term present in other formulations. In particular, this means that it is not obvious how to derive an appropriate decision function from the solution of the training problem.

In a next step, we want to illustrate the differences between the standard training problem (2.4.3) and the corresponding $\nu$-SVM problem (5.2.4). Comparing these two problems, we can see that there are two basic differences. On the one hand, there is sample-dependent bias term in each loss function of (5.2.4) instead of the independent bias in (2.4.3). On the other hand, the minimal and the maximal value of these bias terms are used in another term of the regularization functional. Because of the additional term

$$
\frac{\nu n}{2}\left(\max _{i=1}^{n} \beta_{i}-\min _{i=1}^{n} \beta_{i}\right),
$$

in the objective function, the spread of the individual bias terms cannot be too large in a solution of the training problem. Moreover, the new regularization term does not dependent on the mean value of the individual offsets which means that these offsets can actually replace an overall bias as it is the case in the standard training problem.

One could also argue that a natural choice for the definition of a sample-independent bias term in a decision function would be the mean value between the minimum and the maximum, i.e., to define a decision function

$$
d(\boldsymbol{\varphi}):=\langle\boldsymbol{w}, \boldsymbol{\varphi}\rangle+b \quad \text { with } \quad b:=\frac{1}{2}\left(\min _{i=1}^{n} \beta_{i}+\max _{i=1}^{n} \beta_{i}\right) .
$$

Using this idea together with an artificial variable

$$
c=\frac{1}{2}\left(\max _{i=1}^{n} \beta_{i}-\min _{i=1}^{n} \beta_{i}\right)
$$

measuring the maximal distance from the bias, we can write the $\nu$-SVM training problem equivalently as follows.

Proposition 5.2.2 (second characterization of $\nu$-SVM)
The problem (5.2.4) is equivalent to

$$
\begin{equation*}
\min _{\boldsymbol{w}, b, c, \boldsymbol{\delta}} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b+\delta_{i}\right)+\nu n c \quad \text { s.t. } \quad-c \mathbf{1} \leq \boldsymbol{\delta} \leq c \mathbf{1} . \tag{5.2.7}
\end{equation*}
$$

This formulation of the $\nu$-SVM training problem sheds some light on the behavior of the resulting decision function. Whereas in the standard formulation of the training problem, the decision function is chosen in a way to minimize the loss directly, the $\nu$ SVM formulation allows additionally for the compensation of particular mispredictions. The parameter $\nu$ controls how much cost has to be paid for these compensations. Especially, together with the observations from Section 4.3, this means that the decision offsets are aligned such that the portion of outliers (or almost falsely predicted samples) is potentially reduced in order to meet the dual constraint $\|\boldsymbol{\alpha}\|_{1} \leq \nu n$.

In a last step, our aim is to get rid of the variables $\delta_{i}$ in the formulation of the primal training problem. This is possible by restricting our investigation to monotone loss functions. For classification problems this is no true restriction because there loss functions are usually monotone. In the case of regression problems each loss function can be easily decomposed into the sum of a non-increasing and a non-decreasing part. We summarize an equivalent formulation of the $\nu$-SVM training problem in the subsequent proposition.

Proposition 5.2.3 (third characterization of $\nu$-SVM)
Suppose that each loss function $\ell_{i}$ is either non-increasing or non-decreasing and let

$$
s_{i}:=\left\{\begin{array}{ll}
-1, & \text { if } \ell_{i} \text { is non-decreasing, } \\
+1, & \text { if } \ell_{i} \text { is non-increasing }
\end{array} \quad \text { for } i \in\{1, \ldots, n\} .\right.
$$

Then, problem (5.2.7) is equivalent to

$$
\begin{equation*}
\min _{\boldsymbol{w}, b, c} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b+s_{i} c\right)+\nu n c \quad \text { s.t. } \quad c \geq 0 \tag{5.2.8}
\end{equation*}
$$

Proof. We show that (5.2.8) is equivalent to (5.2.7) and refer to Proposition 5.2.2 for the equivalence to (5.2.4). Since the variables $\delta_{i}$ in (5.2.7) only occur in the loss terms and the loss functions are monotone, it is possible to determine the optimal values of each variable. Hence, we consider the $i$ th loss term and distinguish two cases:

- If the $\ell_{i}$ is non-decreasing, the variable $\delta_{i}$ should be chosen as small as possible in order to minimize the loss term. Due to the constraint $-c \leq \delta_{i}$, the minimal value is equal to $-c=s_{i} c$.
- If the $\ell_{i}$ is non-increasing, the variable $\delta_{i}$ should be chosen as large as possible in order to minimize the loss term. Due to the constraint $\delta_{i} \leq c$, the maximal value
is equal to $c=s_{i} c$.
Taking both cases together, it follows that there exists a solution of (5.2.7) (which is not necessarily unique) such that $\boldsymbol{\delta}=c \boldsymbol{s}$. Thus, $\boldsymbol{\delta}$ in (5.2.7) can be replaced by $c \boldsymbol{s}$ leading to the equivalent problem (5.2.8).

Finally, we come back to the dual formulation of the training problem. This consideration is especially important in order to obtain the corresponding optimality conditions.

Proposition 5.2.4 (dual $\nu$-SVM training problem)
Suppose that the assumptions of Proposition 5.2.3 are satisfied. The dual problem associated with (5.2.8) is given by

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha} \in \mathbb{R}^{n}} & \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}+\sum_{i=1}^{n} \ell_{i}^{\star}\left(-\alpha_{i}\right)  \tag{5.2.9}\\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0 \quad \text { and } \quad\|\boldsymbol{\alpha}\|_{1} \leq \nu n .
\end{array}
$$

Proof. Strictly speaking, a proof that the dual problem has actually the proposed form is not necessary because the primal problem is equivalent to (5.2.4) and that problem was constructed to be the dual of (5.2.9). However, we present a full derivation here to make sure that everything is working as we expected.

First, we formulate problem (5.2.8) as a general training problem by using $\mathcal{H}=\mathcal{F} \times$ $\mathbb{R} \times \mathbb{R}$ and defining

$$
\boldsymbol{T}(\boldsymbol{w}, b, c):=\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b+s_{i} c\right)_{i=1}^{n} \quad \text { and } \quad \mathcal{R}(\boldsymbol{w}, b, c):=\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\nu n c .
$$

Then, using a straightforward computation, we obtain

$$
\boldsymbol{T}^{\star} \boldsymbol{\alpha}=\left(\sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}, \mathbf{1}^{\top} \boldsymbol{\alpha}, \boldsymbol{s}^{\top} \boldsymbol{\alpha}\right)
$$

and

$$
\mathcal{R}^{\star}\left(\boldsymbol{w}^{\star}, b^{\star}, c^{\star}\right)= \begin{cases}\frac{1}{2 \lambda}\left\|\boldsymbol{w}^{\star}\right\|^{2}, & \text { if } b^{\star}=0 \text { and } c^{\star} \leq \nu n, \\ 0, & \text { otherwise. }\end{cases}
$$

Finally, note that by Theorem 3.1.4 and Proposition 3.2.1 it follows that

$$
-\alpha_{i} \in \partial \ell_{i}(\boldsymbol{T}(\boldsymbol{w}, b, c)) \quad \text { for all } \quad i \in\{1, \ldots, n\} .
$$

Because of the definition of $s_{i}$ and the fact that the subdifferential for non-decreasing functions can only contain non-negative values, we get that $\left|\alpha_{i}\right|=s_{i} \alpha_{i} \geq 0$ for all $i \in\{1, \ldots, n\}$. Thus,

$$
\boldsymbol{s}^{\top} \boldsymbol{\alpha}=\sum_{i=1}^{n} s_{i} \alpha_{i}=\sum_{i=1}^{n}\left|\alpha_{i}\right|=\|\boldsymbol{\alpha}\|_{1} .
$$

This means that (5.2.9) is actually the dual problem associated with (5.2.8) due to Corollary 3.1.3.

Of course, the presence of the constraint $\|\boldsymbol{\alpha}\|_{1} \leq \nu n$ in the dual problem does not guarantee that the solution (and hence the decision function) is sparse. In particular, the level of sparseness also depends on the concrete choice of the decision functions. In the extreme case, on could think of the definition of the training problem using loss functions not having zeros in the subdifferential. One example for this is the logistic regression loss function used in Proposition 4.1.3. Here, the additional constraint in the dual $\nu$-SVM formulation cannot lead to a sparsification of the dual solution.

In this view, one could expect that for Least-Squares Support Vector Machines a similar observation is true because of the discussion in Subsection 4.3.3. However, in order to apply Proposition 5.2.3 it is necessary to split the loss functions into a nonincreasing and a non-decreasing part. In this context, a canonical choice would be to consider the decomposition

$$
\ell_{i}(t)=\frac{1}{2}\left(t-y_{i}\right)^{2}=\frac{1}{2} \max \left\{t-y_{i}, 0\right\}^{2}+\frac{1}{2} \max \left\{y_{i}-t, 0\right\}^{2} .
$$

Then, each one-sided loss function can actually have zeros in the subdifferential and the addition of the constraint $\|\boldsymbol{\alpha}\|_{1} \leq \nu n$ will generally lead to a sparser solution. This is because the resulting loss function (using the sample-dependent offset $s_{i} c$ in (5.2.8)) is equivalent to some $\varepsilon$-insensitive version of the squared loss function. Note that the amount of insensitivity (usually defined by the parameter $\varepsilon$ ) is equal to the optimal value of $c$. This value is not known beforehand and depends on the choice of the sparseness parameter $\nu$.

### 5.3 Subspace Support Vector Machines

Another particularly interesting approach to obtain a sparse solution is the idea of Reduced-Set Support Vector Machines as introduced in [Bur96, BS96]. In its original form, a standard training problem is solved and after that another optimization problem is considered to approximate the found solution by means of a suitable reduced set of expansion vectors.

Subsequently, we follow the basic idea of restricting the solution to some pre-defined subset. Of course, in its usual presentation, the motivation behind these methods is not to introduce regularization to the training problem but to simplify the structure of the resulting decision model function. However, it is clear that any restriction of the variables describing the model function implies a certain regularization. Moreover, using convex duality theory, the interpretation of the subspace method as regularization mechanism gives rise to some insight.

We start by defining the regularization functional associated with a Subspace Support Vector Machine as follows.

Definition 5.3.1 (subspace regularization functional)
Let $\mathcal{F}_{s} \subseteq \mathcal{F}$ be a closed subspace of $\mathcal{F}$ and fix $\lambda>0$.

Then, the subspace regularization functional is defined by

$$
\begin{aligned}
\mathcal{R}_{\mathcal{F}_{s}}: \mathcal{F} \times \mathbb{R} & \rightarrow \mathbb{R} \cup\{+\infty\} \\
(\boldsymbol{w}, b) & \mapsto \begin{cases}\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}, & \text { if } \boldsymbol{w} \in \mathcal{F}_{s} \\
\infty, & \text { otherwise }\end{cases}
\end{aligned}
$$

Subsequently, we compute the subdifferential and convex conjugate of the subspace regularization functional.

Proposition 5.3.2 (properties of the subspace regularization functional)
Let $\mathcal{F}_{s} \subseteq \mathcal{F}$ be a closed subspace of $\mathcal{F}$.
Then, the subdifferential of $\mathcal{R}_{\mathcal{F}_{s}}$ is given by

$$
\partial \mathcal{R}_{\mathcal{F}_{s}}(\boldsymbol{w}, b)= \begin{cases}\left\{\left(\boldsymbol{w}^{\star}, b^{\star}\right) \in \mathcal{F} \times \mathbb{R} \mid P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}=\lambda \boldsymbol{w}, b^{\star}=0\right\}, & \text { if } \boldsymbol{w} \in \mathcal{F}_{s}  \tag{5.3.1}\\ \varnothing, & \text { otherwise }\end{cases}
$$

and its conjugate is equal to

$$
\mathcal{R}_{\mathcal{F}_{s}}^{\star}\left(\boldsymbol{w}^{\star}, b^{\star}\right)= \begin{cases}\frac{1}{2 \lambda}\left\|P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}\right\|^{2}, & \text { if } b^{\star}=0  \tag{5.3.2}\\ \infty, & \text { otherwise }\end{cases}
$$

where $P_{\mathcal{F}_{s}}$ denotes the projection operator onto the subspace $\mathcal{F}_{s}$.

Proof. We consider the defining inequality

$$
\begin{equation*}
\mathcal{R}_{\mathcal{F}_{s}}(\tilde{\boldsymbol{w}}, \tilde{b})-\mathcal{R}_{\mathcal{F}_{s}}(\boldsymbol{w}, b) \geq\left\langle\left(\boldsymbol{w}^{\star}, b^{\star}\right),(\tilde{\boldsymbol{w}}, \tilde{b})-(\boldsymbol{w}, b)\right\rangle \tag{5.3.3}
\end{equation*}
$$

for the subdifferential of $\mathcal{R}_{\mathcal{F}_{s}}$ at a fixed point $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ according to Definition A.1.2. First, note that this inequality cannot hold if ( $\boldsymbol{w}, b) \notin \mathcal{F}_{s} \times \mathbb{R}$ because in this case $\mathcal{R}_{\mathcal{F}_{s}}(\boldsymbol{w}, b)=\infty$ by definition. On the other hand, the inequality is trivially satisfied if $(\tilde{\boldsymbol{w}}, \tilde{b}) \notin \mathcal{F}_{s} \times \mathbb{R}$. Hence, let $(\boldsymbol{w}, b),(\tilde{\boldsymbol{w}}, \tilde{b}) \in \mathcal{F}_{s} \times \mathbb{R}$. Then, the left-hand side of (5.3.3) can be written as

$$
\mathcal{R}_{\mathcal{F}_{s}}(\tilde{\boldsymbol{w}}, \tilde{b})-\mathcal{R}_{\mathcal{F}_{s}}(\boldsymbol{w}, b)=\frac{\lambda}{2}\left(\|\tilde{\boldsymbol{w}}\|^{2}-\|\boldsymbol{w}\|^{2}\right)
$$

Because the inequality (5.3.3) has to hold for arbitrary $\tilde{b} \in \mathbb{R}$ and the left-hand side is independent of $\tilde{b}$, it follows that $b^{\star}=0$. This means that the right-hand side of (5.3.3) is equal to

$$
\left\langle\left(\boldsymbol{w}^{\star}, b^{\star}\right),(\tilde{\boldsymbol{w}}, \tilde{b})-(\boldsymbol{w}, b)\right\rangle=\left\langle\boldsymbol{w}^{\star}, \tilde{\boldsymbol{w}}-\boldsymbol{w}\right\rangle=\left\langle P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}, \tilde{\boldsymbol{w}}-\boldsymbol{w}\right\rangle .
$$

Note that the introduction of the projection operator $P_{\mathcal{F}_{s}}$ does not change the value because $\boldsymbol{w}, \tilde{\boldsymbol{w}} \in \mathcal{F}_{s}$. By taking both parts together, we can observe that the inequality (5.3.3) is equivalent to

$$
\frac{\lambda}{2}\|\tilde{\boldsymbol{w}}\|^{2}-\left\langle P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}, \tilde{\boldsymbol{w}}\right\rangle \geq \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}-\left\langle P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}, \boldsymbol{w}\right\rangle
$$

Recall that this inequality must hold for all $\tilde{\boldsymbol{w}} \in \mathcal{F}_{s}$. It is easy to see that the term on the left-hand side attains its minimal value for $\tilde{\boldsymbol{w}}=\lambda^{-1} P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}$. Finally, in order for (5.3.3) to hold for all $(\tilde{\boldsymbol{w}}, \tilde{b}) \in \mathcal{F} \times \mathbb{R}$, it is necessary that

$$
0 \geq \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}-\left\langle P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}, \boldsymbol{w}\right\rangle+\frac{1}{2 \lambda}\left\|P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}\right\|^{2}=\frac{1}{2 \lambda}\left\|\lambda \boldsymbol{w}-P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}\right\|^{2}
$$

i.e., $P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}=\lambda \boldsymbol{w}$. This shows that the subdifferential of $\mathcal{R}_{\mathcal{F}_{s}}$ is given by (5.3.1).

In order to compute the conjugate of $\mathcal{R}_{\mathcal{F}_{s}}$ we use the definition from Definition A.2.1 to obtain

$$
\begin{aligned}
\mathcal{R}_{\mathcal{F}_{s}}^{\star}\left(\boldsymbol{w}^{\star}, b^{\star}\right) & =\sup _{(\boldsymbol{w}, b) \in \mathcal{F} \times b}\left\{\left\langle\left(\boldsymbol{w}^{\star}, b^{\star}\right),(\boldsymbol{w}, b)\right\rangle-\mathcal{R}_{\mathcal{F}_{s}}(\boldsymbol{w}, b)\right\} \\
& =\sup _{(\boldsymbol{w}, b) \in \mathcal{F}_{s} \times b}\left\{\left\langle\left(P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}, b^{\star}\right),(\boldsymbol{w}, b)\right\rangle-\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}\right\} \\
& = \begin{cases}\sup _{\boldsymbol{w} \in \mathcal{F}_{s}}\left\{\left\langle P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}, \boldsymbol{w}\right\rangle-\frac{\lambda}{2}\|\boldsymbol{w}\|^{2}\right\}, & \text { if } b^{\star}=0, \\
\infty, & \text { otherwise. }\end{cases}
\end{aligned}
$$

The supremum in the last term is attained for $\boldsymbol{w}=\lambda^{-1} P_{\mathcal{F}_{s}} \boldsymbol{w}^{\star}$, which proves (5.3.2).
Having the subdifferential and the convex conjugate of the subspace regularization operator at hand, we can proceed to derive the dual optimization problem and optimality conditions for a general formulation of a Subspace Support Vector Machine.

Corollary 5.3.3 (dual problem for Subspace SVM)
Let $\mathcal{F}_{s} \subseteq \mathcal{F}$ be a closed subspace of $\mathcal{F}$. Suppose that $\boldsymbol{T}$ is the standard decision operator and $\mathcal{L}$ is a separable loss functional such that Assumption 3.1.2 is satisfied with $\mathcal{R}=$ $\mathcal{R}_{\mathcal{F}_{s}}$.
Then, the dual training problem associated with

$$
\begin{equation*}
\min _{\substack{\boldsymbol{w} \in \mathcal{F}_{s} \\ b \in \mathbb{R}}} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right) \tag{5.3.4}
\end{equation*}
$$

is given by

$$
\begin{equation*}
\min _{\boldsymbol{\alpha} \in \mathbb{R}^{n}} \frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K}_{s} \boldsymbol{\alpha}+\sum_{i=1}^{n} \ell_{i}^{\star}\left(-\alpha_{i}\right) \quad \text { s.t. } \quad \mathbf{1}^{\top} \boldsymbol{\alpha}=0, \tag{5.3.5}
\end{equation*}
$$

where $\boldsymbol{K}_{s} \in \mathbb{R}^{n \times n}$ is the projected kernel matrix defined by

$$
\left[\boldsymbol{K}_{s}\right]_{i j}=\left\langle P_{\mathcal{F}_{s}} \boldsymbol{\varphi}_{i}, P_{\mathcal{F}_{s}} \boldsymbol{\varphi}_{j}\right\rangle \quad \text { for } \quad i, j \in\{1, \ldots, n\}
$$

Proof. By taking together the results of Proposition 3.3.1 and Proposition 5.3.2, we get

$$
\mathcal{R}_{\mathcal{F}_{s}}^{\star}\left(\boldsymbol{T}^{\star} \boldsymbol{\alpha}\right)= \begin{cases}\frac{1}{2 \lambda}\left\|P_{\mathcal{F}_{s}} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}\right\|^{2}, & \text { if } \mathbf{1}^{\top} \boldsymbol{\alpha}=0 \\ \infty, & \text { otherwise }\end{cases}
$$

Then, the term can be simplified using

$$
\left\|P_{\mathcal{F}_{s}} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}\right\|^{2}=\left\|\sum_{i=1}^{n} \alpha_{i} P_{\mathcal{F}_{s}} \boldsymbol{\varphi}_{i}\right\|^{2}=\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{i} \alpha_{j}\left\langle P_{\mathcal{F}_{s}} \boldsymbol{\varphi}_{i}, P_{\mathcal{F}_{s}} \boldsymbol{\varphi}_{j}\right\rangle=\boldsymbol{\alpha}^{\top} \boldsymbol{K}_{s} \boldsymbol{\alpha} .
$$

Finally, we apply Corollary 3.1.3 together with Proposition 3.2.1 to obtain the dual problem as stated in (5.3.5).

For sake of completeness, we also provide a detailed formulation of the optimality conditions for the Subspace SVM as follows.

Corollary 5.3.4 (optimality conditions for Subspace SVM)
Let the assumptions of Corollary 5.3.3 be satisfied.
Then, $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ is a solution of (5.3.4) if and only if there exists $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ such that

$$
\begin{gather*}
\mathbf{1}^{\top} \boldsymbol{\alpha}=0  \tag{5.3.6}\\
\lambda \boldsymbol{w}=\sum_{i=1}^{n} \alpha_{i} P_{\mathcal{F}_{s}} \boldsymbol{\varphi}_{i}, \tag{5.3.7}
\end{gather*}
$$

and

$$
\begin{equation*}
-\alpha_{i} \in \partial \ell_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}\right\rangle+b\right) \quad \text { for all } \quad i \in\{1, \ldots, n\} . \tag{5.3.8}
\end{equation*}
$$

In this case, $\boldsymbol{\alpha}$ is also a solution of the dual problem (5.3.5).

Proof. The optimality conditions follow directly from Theorem 3.1.4, Proposition 3.2.1, and Proposition 5.3.2.

Now, we can compare the dual and the optimality conditions of the Subspace SVM with that of the standard SVM given in Corollary 3.4.1 and Corollary 3.4.2. First, we can see that the only difference between the dual problems (3.4.1) and (5.3.5) is in the different choice of the kernel matrix, namely $\boldsymbol{K}$ and $\boldsymbol{K}_{s}$, respectively. Furthermore, the new optimality conditions (5.3.6) and (5.3.8) are identical to the conditions (3.4.2) and (3.4.4) of the standard training problem. The only difference in the optimality conditions lies in (5.3.7), where an additional projection is present opposed to the case of (3.4.3).

## 6 Reformulations of the Primal Training Problem

In this chapter, we consider some particular aspects connected to the training of Support Vector Machines using finite-dimensional problems. In particular, we present different formulations of the primal training problem and investigate their relationships. Moreover, the relation to a dual optimality system is considered. This preliminary work substantially extends what is known from the literature and lays the foundation for the application of derivative-based primal training methods in Chapter 8.

Initially, we state a simple finite-dimensional formulation of the training problem (Section 6.1) which is then refined by means of suitably defined bijections (Section 6.2). The applied idea immediately leads to another finite-dimensional problem formulation (Section 6.3) retaining the uniformly convex part in the objective function of the original training problem. Finally, we discuss the relation between the finite-dimensional formulations of the primal training problem and the associated dual problem (Section 6.4). By means of these investigations we are able to define a system of nonlinear equations which is equivalent to the training problem under certain assumptions (Section 6.5).

### 6.1 First Finite-Dimensional Formulation

Recall that, due to the Representer Theorem (stated in Corollary 3.5.1), for any solution $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ of the primal training problem, it follows that

$$
\begin{equation*}
\boldsymbol{w}=\frac{1}{\lambda} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i}, \tag{6.1.1}
\end{equation*}
$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ is a solution of the associated dual training problem. In the following, we focus on the primal formulation of the training problem only and ignore the fact that $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ is closely related to a dual solution for a moment.

A first finite-dimensional formulation of the training problem is based solely on the previous observation. Plugging the representation of $\boldsymbol{w}$ of (6.1.1) into the original train-
ing problem (2.4.3), we obtain the problem

$$
\begin{equation*}
\min _{\substack{\boldsymbol{\alpha} \in \mathbb{R}^{n} \\ b \in \mathbb{R}}} \psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b) \tag{6.1.2}
\end{equation*}
$$

with the objective function defined by

$$
\begin{equation*}
\psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b):=\frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}+\sum_{i=1}^{n} \ell_{i}\left(\frac{1}{\lambda}[\boldsymbol{K} \boldsymbol{\alpha}]_{i}+b\right) \tag{6.1.3}
\end{equation*}
$$

where the kernel matrix $\boldsymbol{K}$ is defined according to Definition 3.3.3. We call this problem the finite-dimensional primal training problem. The equivalence between the primal training problem (2.4.3) and problem (6.1.2) can be formulated as follows.

Proposition 6.1.1 (equivalent primal training problem)
Let (2.4.3) be a standard training problem according to Definition 2.4.5. Then, the optimization problems (2.4.3) and (6.1.2) are equivalent in the sense that

- if $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ is a solution of the primal training problem (2.4.3), then there exists some $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ such that (6.1.1) holds and $(\boldsymbol{\alpha}, b)$ is a solution of the finitedimensional primal training problem (6.1.2), and
- if $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R}$ is a solution of the finite-dimensional primal training problem (6.1.2), then $(\boldsymbol{w}, b)$ with $\boldsymbol{w}$ defined according to (6.1.1) is a solution of the primal training problem (2.4.3).

As we have seen in Section 3.8 (particularly, in Corollary 3.8.1), the $\boldsymbol{w}$-part of the solution is unique if the standard regularization operator is used. This is exactly the setting that we consider here. However, the substitution of the vector $\boldsymbol{w}$ by its expansion can destroy this uniqueness as it may introduce additional ambiguity.

In the formulation of the training problem in (6.1.2) this can be seen from the fact that any offset $\boldsymbol{\delta} \in \operatorname{ker}(\boldsymbol{K})$ can be added to the variable $\boldsymbol{\alpha}$ without changing the objective function at all. When training methods are under consideration, this degeneration property can lead to additional difficulties. We will discuss the resulting issues and how to handle them later on in the subsequent sections.

## Remark 6.1 .2 (application of $\nu$-SVMs)

Note that the assertion of Proposition 6.1.1 is also true for the $\nu$-SVM training problem introduced in (5.2.8) if the additional variable $c \in \mathbb{R}$ is added. This means that most of the observations and computations below can be transferred to this problem, which can have beneficial properties for practical application. We do not present every step for that problem here in order to keep the presentation simple.

Subsequently, the finite-dimensional training problem (6.1.2) forms the basis of our investigation. Of course, the main motivation for considering (6.1.2) instead of (2.4.3) is that it is written in terms of a finite number of variables. This means that it is quite straightforward to treat it practically using standard optimization methods.

While considering the primal formulation of the training problem, we want to study the application of gradient-based optimization methods and second-order methods. For this reason, we present the particular form of the gradient of the objective function $\psi_{\mathrm{pf}}$ in the following proposition.

Proposition 6.1.3 (gradient of the training objective function)
Let (2.4.3) be a standard training problem according to Definition 2.4.5 and let $\psi_{\mathrm{pf}}$ be defined according to (6.1.3). If the loss functions $\ell_{i}(i \in\{1, \ldots, n\})$ are differentiable, the gradient of $\psi_{\mathrm{pf}}$ is given by

$$
\begin{equation*}
\nabla \psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b)=\binom{\frac{1}{\lambda} \boldsymbol{K}(\boldsymbol{\alpha}+\boldsymbol{g}(\boldsymbol{\alpha}, b))}{\mathbf{1}^{\top} \boldsymbol{g}(\boldsymbol{\alpha}, b)} \tag{6.1.4}
\end{equation*}
$$

with

$$
\begin{equation*}
g_{i}(\boldsymbol{\alpha}, b):=\ell_{i}^{\prime}\left(\frac{1}{\lambda}[\boldsymbol{K} \boldsymbol{\alpha}]_{i}+b\right) \quad \text { for } \quad i \in\{1, \ldots, n\} . \tag{6.1.5}
\end{equation*}
$$

Moreover, for the application of second-order methods it is necessary to compute the Hessian of the objective function. This is done in the following proposition.

Proposition 6.1.4 (Hessian of the training objective function)
Let $\psi_{\mathrm{pf}}$ be defined according to (6.1.2). If the loss functions $\ell_{i}(i \in\{1, \ldots, n\})$ are twice differentiable, the Hessian of $\psi_{\text {pf }}$ is given by

$$
\nabla^{2} \psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b)=\frac{1}{\lambda}\left(\begin{array}{cc}
\boldsymbol{K}+\frac{1}{\lambda} \boldsymbol{K} \boldsymbol{H} \boldsymbol{K} & \boldsymbol{K} \boldsymbol{H} \mathbf{1} \\
\mathbf{1}^{\top} \boldsymbol{H} \boldsymbol{K} & \lambda \mathbf{1}^{\top} \boldsymbol{H} \mathbf{1}
\end{array}\right)
$$

with

$$
h_{i}(\boldsymbol{\alpha}, b):=\ell_{i}^{\prime \prime}\left(\frac{1}{\lambda}[\boldsymbol{K} \boldsymbol{\alpha}]_{i}+b\right) \quad \text { for } \quad i \in\{1, \ldots, n\}
$$

and the diagonal matrix $\boldsymbol{H}:=\operatorname{diag}(\boldsymbol{h}(\boldsymbol{\alpha}, b))$.

The previous observations will be used in Chapter 8 for the construction of efficient primal training methods.

### 6.2 Equivalent Formulation using a Bijection

Now, we come back to the consideration of the equivalence relation between the problems (2.4.3) and (6.1.2) given in Proposition 6.1.1 and its implications to the application of training methods. For the moment we postpone the discussion of problems associated with the possible non-uniqueness of the variable $b$ in the solution and concentrate
on the relation between the variables $\boldsymbol{w}$ and $\boldsymbol{\alpha}$.
As we have argued above, the finite-dimensional problem (6.1.2) has no unique solution if the kernel matrix has non-trivial null space. In order to construct an equivalent training problem without losing the partial uniqueness, we consider two particular bijective linear operators subsequently.

Proposition 6.2 .1 (bijections between $\mathcal{F}_{n}$ and $\operatorname{ker}(\boldsymbol{K})^{\perp}$ )
Let $\mathcal{F}_{n}:=\operatorname{span}\left\{\boldsymbol{\varphi}_{1}, \ldots, \boldsymbol{\varphi}_{n}\right\}$ be the finite-dimensional subspace of $\mathcal{F}$ generated by the feature vectors $\varphi_{1}, \ldots, \varphi_{n} \in \mathcal{F}$. Denote the associated kernel matrix according to Definition 3.3.3 by $\boldsymbol{K}$. Let $\boldsymbol{K}^{\frac{1}{2}}$ denote the square root (i.e., the unique symmetric positive semi-definite matrix satisfying $\left(\boldsymbol{K}^{\frac{1}{2}}\right)^{2}=\boldsymbol{K}$ ) of the positive semi-definite matrix $\boldsymbol{K}$.
Then, we have the following two bijections.
(a) The linear operator

$$
\begin{align*}
\boldsymbol{\Phi}: \operatorname{ker}(\boldsymbol{K})^{\perp} & \rightarrow \mathcal{F}_{n}, \\
\boldsymbol{\alpha} \mapsto \boldsymbol{\Phi} \boldsymbol{\alpha}: & =\frac{1}{\lambda} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i} \tag{6.2.1}
\end{align*}
$$

is bijective.
(b) It holds that $\operatorname{ker}(\boldsymbol{K})=\operatorname{ker}\left(\boldsymbol{K}^{\frac{1}{2}}\right)$ and the linear operator

$$
\boldsymbol{K}^{\frac{1}{2}}: \operatorname{ker}(\boldsymbol{K})^{\perp} \rightarrow \operatorname{ker}(\boldsymbol{K})^{\perp}
$$

is bijective.

Proof. Suppose that $\boldsymbol{\alpha}, \overline{\boldsymbol{\alpha}} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$ satisfy $\boldsymbol{\Phi} \boldsymbol{\alpha}=\boldsymbol{\Phi} \overline{\boldsymbol{\alpha}}$. Then, multiplication of the equation by $\boldsymbol{\varphi}_{i} \in \mathcal{F}_{n}$ for $i \in\{1, \ldots, n\}$ yields the equation $\boldsymbol{K} \boldsymbol{\alpha}=\boldsymbol{K} \overline{\boldsymbol{\alpha}}$. This means that $\boldsymbol{\alpha}-\overline{\boldsymbol{\alpha}} \in \operatorname{ker}(\boldsymbol{K})$. But we also have that $\boldsymbol{\alpha}-\overline{\boldsymbol{\alpha}} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$ by definition. Thus, the difference lies in $\operatorname{ker}(\boldsymbol{K}) \cap \operatorname{ker}(\boldsymbol{K})^{\perp}=\{\mathbf{0}\}$, i.e., $\boldsymbol{\alpha}=\overline{\boldsymbol{\alpha}}$, which proves part (a).

In order to check the facts about $\boldsymbol{K}^{\frac{1}{2}}$ in part (b), we first note that the square root of the positive semi-definite matrix $\boldsymbol{K}$ is uniquely defined, cf. [HJ12, Theorem 7.2.6]. Suppose that $\boldsymbol{K}=\boldsymbol{V} \boldsymbol{\Sigma} \boldsymbol{V}^{\top}$ is the spectral decomposition of $\boldsymbol{K}$ with an orthogonal matrix $\boldsymbol{V}$ and a non-negative diagonal matrix $\boldsymbol{\Sigma}$. Then, the square root of $\boldsymbol{K}$ is given by $\boldsymbol{K}^{\frac{1}{2}}=\boldsymbol{V} \boldsymbol{\Sigma}^{\frac{1}{2}} \boldsymbol{V}^{\top}$. Using this representation it is easy to see that

$$
\operatorname{ker}(\boldsymbol{K})=\operatorname{ker}\left(\boldsymbol{V} \boldsymbol{\Sigma} \boldsymbol{V}^{\top}\right)=\operatorname{ker}\left(\boldsymbol{\Sigma} \boldsymbol{V}^{\top}\right)=\operatorname{ker}\left(\boldsymbol{\Sigma}^{\frac{1}{2}} \boldsymbol{V}^{\top}\right)=\operatorname{ker}\left(\boldsymbol{V} \boldsymbol{\Sigma}^{\frac{1}{2}} \boldsymbol{V}^{\top}\right)=\operatorname{ker}\left(\boldsymbol{K}^{\frac{1}{2}}\right)
$$

Moreover, the symmetry of $\boldsymbol{K}^{\frac{1}{2}}$ implies that $\operatorname{im}\left(\boldsymbol{K}^{\frac{1}{2}}\right)=\operatorname{ker}\left(\left(\boldsymbol{K}^{\frac{1}{2}}\right)^{\top}\right)^{\perp}=\operatorname{ker}\left(\boldsymbol{K}^{\frac{1}{2}}\right)^{\perp}$. Hence, $\boldsymbol{K}^{\frac{1}{2}}$ (seen as a linear operator on $\operatorname{ker}(\boldsymbol{K})^{\perp}$ ) clearly is a bijection.

As a first consequence of this proposition, we obtain an optimization problem which is equivalent to the original training problem in the sense that there is a bijection between feasible points. This means that the uniqueness of solutions is preserved naturally.

Proposition 6.2 .2 (equivalent primal training problem)
Let (2.4.3) be a standard training problem according to Definition 2.4.5.
Then, the optimization problem (2.4.3) is equivalent to

$$
\begin{equation*}
\min _{\substack{\alpha \in \mathbb{R}^{n} \\ b \in \mathbb{R}}} \psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b) \quad \text { s.t. } \quad \boldsymbol{\alpha} \in \operatorname{ker}(\boldsymbol{K})^{\perp} \tag{6.2.2}
\end{equation*}
$$

in the sense that $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ is a solution of the (2.4.3) if and only if $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R}$ is a solution of (6.2.2) and $\boldsymbol{\Phi} \boldsymbol{\alpha}=\boldsymbol{w}$.

Note that problem (6.2.2) is not suitable for applications because the constraint $\boldsymbol{\alpha} \in$ $\operatorname{ker}(\boldsymbol{K})^{\perp}$ cannot be handled in practice due to a large size of the kernel matrix $\boldsymbol{K}$. Nevertheless, it provides a convenient basis for further theoretical investigations.

### 6.3 Second Finite-Dimensional Formulation

Aside from the additional constraint $\boldsymbol{\alpha} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$ in (6.2.2), there is another possible downside associated with the finite-dimensional formulations in (6.1.2) and (6.2.2). After the transformation of $\boldsymbol{w}$ using the linear operator $\boldsymbol{\Phi}$, the resulting objective function has not a uniformly convex part anymore. On the one hand, this possibly results in a loss of uniqueness of the solution as we have already mentioned above. On the other hand, it can prevent the application of Newton's methods because of zero eigenvalues of the Hessian matrix. To circumvent this problem, we apply the second bijective operator of Proposition 6.2.1 that makes it possible to recover the original structure of the regularization term as follows.

Proposition 6.3.1 (another equivalent primal training problem)
Let (2.4.3) be a standard training problem according to Definition 2.4.5.
Then, the optimization problem (2.4.3) is equivalent to

$$
\begin{equation*}
\min _{\substack{\tilde{\alpha} \in \mathbb{R}^{n} \\ b \in \mathbb{R}}} \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b) \tag{6.3.1}
\end{equation*}
$$

with

$$
\begin{equation*}
\tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b):=\frac{1}{2 \lambda}\|\tilde{\boldsymbol{\alpha}}\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\frac{1}{\lambda}\left[\boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{\alpha}}\right]_{i}+b\right) \tag{6.3.2}
\end{equation*}
$$

in the sense that $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ is a solution of (2.4.3) if and only if $(\tilde{\boldsymbol{\alpha}}, b) \in \mathbb{R}^{n} \times \mathbb{R}$ is a solution of (6.3.1) and $\boldsymbol{\Phi} \boldsymbol{K}^{-\frac{1}{2}} \tilde{\boldsymbol{\alpha}}=\boldsymbol{w}$. In particular, the solution satisfies $\tilde{\boldsymbol{\alpha}} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$ such that the term $\boldsymbol{K}^{-\frac{1}{2}} \tilde{\boldsymbol{\alpha}}$ is well-defined.

Proof. To see that the problems are equivalent, we use the two bijections $\boldsymbol{\Phi}$ and $\boldsymbol{K}^{\frac{1}{2}}$ proposed in Proposition 6.2.1. First, the bijective transformation $\boldsymbol{w}=\boldsymbol{\Phi} \boldsymbol{\alpha}$ between
$\boldsymbol{w} \in \mathcal{F}_{n}$ and $\boldsymbol{\alpha} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$ yields that problem (2.4.3) is equivalent to (6.1.2) if the constraint $\boldsymbol{\alpha} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$ is added. Second, in the same way, the problem (6.1.2) is equivalent to (6.3.1) with $\tilde{\boldsymbol{\alpha}}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}$ under the additional constraints $\boldsymbol{\alpha} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$ and $\tilde{\boldsymbol{\alpha}} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$, respectively. Taking both transformations together, the equivalence between (6.1.2) and (6.3.1) follows, again with the additional constraint $\tilde{\boldsymbol{\alpha}} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$.

Finally, we show that it is not necessary to add this constraint because any solution of the problem satisfies it anyway. For this purpose, let $(\tilde{\boldsymbol{\alpha}}, b) \in \mathbb{R}^{n} \times \mathbb{R}$ and suppose that $\tilde{\boldsymbol{\alpha}}=\tilde{\boldsymbol{\alpha}}^{0}+\tilde{\boldsymbol{\alpha}}^{1}$ with $\tilde{\boldsymbol{\alpha}}^{0} \in \operatorname{ker}(\boldsymbol{K})$ and $\tilde{\boldsymbol{\alpha}}^{1} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$. Then, $\boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{\alpha}}=\boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{\alpha}}^{1}$ and we obtain

$$
\tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b)=\frac{1}{2 \lambda}\left\|\tilde{\boldsymbol{\alpha}}^{0}\right\|^{2}+\frac{1}{2 \lambda}\left\|\tilde{\boldsymbol{\alpha}}^{1}\right\|^{2}+\sum_{i=1}^{n} \ell_{i}\left(\frac{1}{\lambda}\left[\boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{\alpha}}^{1}\right]_{i}+b\right)=\frac{1}{2 \lambda}\left\|\tilde{\boldsymbol{\alpha}}^{0}\right\|^{2}+\tilde{\psi}_{\mathrm{pf}}\left(\tilde{\boldsymbol{\alpha}}^{1}, b\right)
$$

This means that the point $(\tilde{\boldsymbol{\alpha}}, b)$ can only be a solution of problem (6.3.1) if $\tilde{\boldsymbol{\alpha}}^{0}=\mathbf{0}$, i.e., the condition $\tilde{\boldsymbol{\alpha}} \in \operatorname{ker}(\boldsymbol{K})^{\perp}$ is necessary for optimality and the constraint would be redundant in (6.3.1).

In order to prepare for the application of derivative-based methods in Chapter 8, we state the form of the gradient and the Hessian of the objective function $\tilde{\psi}_{\mathrm{pf}}$ below. The given formulas can be verified by means of simple computations.

Proposition 6.3.2 (gradient and Hessian of the second training objective function) Let (2.4.3) be a standard training problem according to Definition 2.4.5 and let $\tilde{\psi}_{\mathrm{pf}}$ be defined according to (6.3.2).
Then, we obtain the following representations for the first and second-order derivatives of $\tilde{\psi}_{\mathrm{pf}}$.
(a) If the loss functions $\ell_{i}$ for $i \in\{1, \ldots, n\}$ are differentiable, the gradient of $\tilde{\psi}_{\mathrm{pf}}$ is given by

$$
\begin{equation*}
\nabla \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b)=\binom{\frac{1}{\lambda}\left(\tilde{\boldsymbol{\alpha}}+\boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{g}}(\tilde{\boldsymbol{\alpha}}, b)\right)}{\mathbf{1}^{\top} \tilde{\boldsymbol{g}}(\tilde{\boldsymbol{\alpha}}, b)} \tag{6.3.3}
\end{equation*}
$$

with $\tilde{g}_{i}(\tilde{\boldsymbol{\alpha}}, b):=\ell_{i}^{\prime}\left(\frac{1}{\lambda}\left[\boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{\alpha}}\right]_{i}+b\right)$ for $i \in\{1, \ldots, n\}$.
(b) Moreover, if the loss functions $\ell_{i}$ for $i \in\{1, \ldots, n\}$ are twice differentiable, the Hessian of $\tilde{\psi}_{\mathrm{pf}}$ is given by

$$
\nabla^{2} \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b)=\frac{1}{\lambda}\left(\begin{array}{cc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{H}} \boldsymbol{K}^{\frac{1}{2}} & \boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{H}} \mathbf{1} \\
\mathbf{1}^{\top} \tilde{\boldsymbol{H}} \boldsymbol{K}^{\frac{1}{2}} & \lambda \mathbf{1}^{\top} \tilde{\boldsymbol{H}} \mathbf{1}
\end{array}\right)
$$

with $\tilde{h}_{i}(\tilde{\boldsymbol{\alpha}}, b):=\ell_{i}^{\prime \prime}\left(\frac{1}{\lambda}\left[\boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{\alpha}}\right]_{i}+b\right)$ for $i \in\{1, \ldots, n\}$ and $\tilde{\boldsymbol{H}}:=\operatorname{diag}(\tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b))$.

### 6.4 Review of Dual Optimality Conditions

In the remaining part of this chapter, we focus on the connection between the dual training problem (3.4.1) and the finite-dimensional primal training problem (6.1.2). For the derivation of the latter problem we already used the Representer Theorem (cf. Corollary 3.5.1) which indicates that the two problems must be related in some sense. However, there is no simple one-to-one relation between thos problems in general. This claim is illustrated by the following example.

Example 6.4.1 (difference between dual and finite-dimensional primal problem) Consider a standard training problem given by the training samples

$$
\begin{aligned}
& \boldsymbol{\varphi}_{1}:=\mathbf{0}, \quad \ell_{1}(t)=\frac{1}{2}(t-1)^{2}, \\
& \boldsymbol{\varphi}_{2}:=\mathbf{0}, \quad \ell_{2}(t)=\frac{1}{2}(t+1)^{2} .
\end{aligned}
$$

Then, $\boldsymbol{K}=\boldsymbol{O} \in \mathbb{R}^{2 \times 2}$, and the conjugate loss functions are given by

$$
\begin{aligned}
& \ell_{1}^{\star}\left(-\alpha_{1}\right)=\frac{1}{2} \alpha_{1}^{2}-\alpha_{1}=\frac{1}{2}\left(\alpha_{1}-1\right)^{2}-\frac{1}{2}, \\
& \ell_{2}^{\star}\left(-\alpha_{2}\right)=\frac{1}{2} \alpha_{2}^{2}+\alpha_{2}=\frac{1}{2}\left(\alpha_{2}+1\right)^{2}-\frac{1}{2} .
\end{aligned}
$$

For the different formulation of the training problem, we obtain the following:
(a) The finite-dimensional primal training problem in (6.1.2) reads as

$$
\min _{\alpha_{1}, \alpha_{2}, b \in \mathbb{R}} \frac{1}{2}(b-1)^{2}+\frac{1}{2}(b+1)^{2} .
$$

Thus, any point $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{2} \times \mathbb{R}$ with $b=0$ is a solution of this problem.
(b) The finite-dimensional primal training problem with additional constraint given by (6.2.2) has the unique solution $(\boldsymbol{\alpha}, b)=(\mathbf{0}, 0) \in \mathbb{R}^{2} \times \mathbb{R}$, because $\operatorname{ker}(\boldsymbol{K})=\mathbb{R}^{2}$ implies $\operatorname{ker}(\boldsymbol{K})^{\perp}=\{\mathbf{0}\}$.
(c) The dual problem is given by

$$
\min _{\alpha \in \mathbb{R}^{2}} \frac{1}{2}\left(\alpha_{1}-1\right)^{2}+\frac{1}{2}\left(\alpha_{2}+1\right)^{2}-1 \quad \text { s.t. } \quad \alpha_{1}+\alpha_{2}=0 .
$$

This problem has the unique solution $\boldsymbol{\alpha}=(1,-1) \in \mathbb{R}^{2}$.
This shows that the solution of the dual problem is generally not equal to the $\boldsymbol{\alpha}$-part of the solution of the problems (6.1.2) or (6.2.2).

Because of the previous observations, we cannot expect to obtain the dual solution directly by solving problem (6.1.2). However, in view of the optimality conditions given
in Corollary 3.6.3 one can indeed expect that it is possible to fix this issue. We start the discussion by refining the previous statement about dual optimality conditions.

Corollary 6.4.2 (system of conditions for the dual training problem)
Consider a standard training problem as in Definition 2.4.5 and suppose that Assumption 3.1.2 holds. Consider the system of conditions

$$
\begin{align*}
& -\alpha_{i} \in \partial \ell_{i}\left(\frac{1}{\lambda} \sum_{j=1}^{n} \alpha_{j}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}_{j}\right\rangle+b\right) \quad \text { for all } \quad i \in\{1, \ldots, n\},  \tag{6.4.1}\\
& \mathbf{1}^{\top} \boldsymbol{\alpha}=0 .
\end{align*}
$$

(a) A point $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ is a solution of the dual training problem (3.4.1) if and only if there exists some $b \in \mathbb{R}$ such that the system (6.4.1) is satisfied.
(b) For any solution $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R}$ of (6.4.1), the point $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ with $\boldsymbol{w}=\boldsymbol{\Phi} \boldsymbol{\alpha}$ is a solution of the primal training problem (2.4.3).
(c) Any solution of the system (6.4.1) is also a solution of the finite-dimensional primal training problem (6.1.2).

Proof. Assertion (a) is already given in Corollary 3.6.3 and assertion (b) follows from Corollary 3.4.2. Hence, it remains to show that assertion (c) is correct. For this reason, let $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R}$ be a solution of (6.4.1). Of course, $(\boldsymbol{\alpha}, b)$ is also a feasible point for (6.1.2). Then, because $(\boldsymbol{w}, b)$ with $\boldsymbol{w}=\boldsymbol{\Phi} \boldsymbol{\alpha}$ is optimal for (2.4.3), the point $(\boldsymbol{\alpha}, b)$ must also be a solution of (6.1.2), cf. Proposition 6.1.1.

Now there is still one open question, namely whether it is possible to obtain a solution of the dual training problem from a solution of (6.1.2). In the following, we show that one cannot expect that a simple procedure solves this problem in general. The example follows a similar structure as the previous Example 6.4.1 and reinforces the observation of the latter.

Example 6.4.3 (choice of a dual solution) Consider a standard training problem given by the training samples

$$
\boldsymbol{\varphi}_{i}:=\mathbf{0} \text { and } \ell_{i}(t):=|t| \quad \text { for } i \in\{1,2,3\} .
$$

Then, $\boldsymbol{K}=\boldsymbol{O} \in \mathbb{R}^{3 \times 3}$, and the conjugate loss functions are given by

$$
\ell_{i}^{\star}(-\alpha)= \begin{cases}0, & \text { if } \alpha \in[-1,1] \\ \infty, & \text { otherwise }\end{cases}
$$

for $i \in\{1,2,3\}$.
(a) The primal training problem is given by

$$
\min _{\substack{\boldsymbol{w} \in \mathcal{F} \\ b \in \mathbb{R}}} \frac{\lambda}{2}\|\boldsymbol{w}\|^{2}+3|b|
$$

and has the unique solution $(\boldsymbol{w}, b)=(\mathbf{0}, 0) \in \mathcal{F} \times \mathbb{R}$.
(b) The dual training problem is given by

$$
\min _{\alpha \in \mathbb{R}^{3}} 0 \quad \text { s.t. } \quad \alpha_{1}+\alpha_{2}+\alpha_{3}=0, \boldsymbol{\alpha} \in[-1,1]^{3} .
$$

Any feasible point of this problem is also a solution.
(c) The finite-dimensional formulation in (6.1.2) can be written as

$$
\min _{\substack{\alpha \in \mathbb{R}^{3} \\ b \in \mathbb{R}}} 3|b|
$$

and has solutions $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{3} \times \mathbb{R}$ with arbitrary $\boldsymbol{\alpha} \in \mathbb{R}^{3}$ and $b=0$.
This shows again that the solution sets of the problems (3.4.1) and (6.1.2) are in general not equal. Note that any solution of the second problem nevertheless leads to the same primal solution under the transformation $\boldsymbol{w}=\boldsymbol{\Phi} \boldsymbol{\alpha}$.
Finally, we consider the optimality conditions given in Corollary 3.4.3. It is easy to see that the equation (3.4.3) is always satisfied because $\boldsymbol{w}=\mathbf{0}$ follows from $\boldsymbol{\varphi}_{i}=\mathbf{0}$. Moreover, from (3.4.4) it follows that

$$
-\alpha_{i} \in[-1,1] \text { for } i \in\{1,2,3\}
$$

and the equation $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ must additionally be satisfied for a dual solution according to (3.4.2).
Notably, this example shows that for the definition of a dual solution it is not sufficient to consider the conditions (3.4.4) alone. For instance, choosing $\alpha_{1}=\alpha_{2}=1$ first makes it impossible to find a dual solution because the two remaining conditions $\alpha_{3} \in[-1,1]$ and $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ are then incompatible.

### 6.5 Formulation as a System of Nonlinear Equations

A possible approach to circumvent the previously described problem is to make sure that the dual solution is unique. Under appropriate conditions (for instance Assumption 3.1.2), it follows that the dual training problem is solvable due to Corollary 3.1.3. For a standard training problem this assertion is also given in Corollary 3.4.1. In order to state a sufficient condition for the uniqueness of the solution, we can use the relationship between essential strict convexity and essential smoothness summarized in Theorem A.5.3. This observation leads directly to the following assertion.

Proposition 6.5.1 (unique dual solution)
Consider a standard training problem as in Definition 2.4.5 with essentially differentiable loss functions and suppose that Assumption 3.1.2 holds.
Then, the dual training problem (3.4.1) has a unique solution.

Proof. Due to Corollary 3.4.1, the dual training problem has a solution. Moreover, because the loss functions are essentially smooth, the conjugate loss functions $\ell_{i}^{\star}$ are essentially strictly convex for all $i \in\{1, \ldots, n\}$ due to Theorem A.5.3. This implies that the objective function of the dual training problem (3.4.1) is also essentially strictly convex. Thus, the solution of this problem must be unique.

For differentiable loss functions, the assertion of Corollary 6.4.2 can be further refined as follows. Note that we use differentiable instead of only essentially smooth functions to simplify the presentation here. In an analog way, the system can be written down in a more general context. However, then it is necessary to add conditions which guarantee that the derivative of the loss functions exists at the considered points.

Corollary 6.5 .2 (system of nonlinear equations for the training problem)
Consider a standard training problem as in Definition 2.4.5 with differentiable loss functions and suppose that Assumption 3.1.2 holds. Consider the system of nonlinear equations

$$
\begin{equation*}
\boldsymbol{F}(\boldsymbol{\alpha}, b):=\binom{\boldsymbol{\alpha}+\boldsymbol{g}(\boldsymbol{\alpha}, b)}{\mathbf{1}^{\top} \boldsymbol{\alpha}}=\binom{0}{0} \tag{6.5.1}
\end{equation*}
$$

with $\boldsymbol{g}$ defined in (6.1.5).
(a) A point $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ is a solution of the dual training problem (3.4.1) if and only if there exists some $b \in \mathbb{R}$ such that the system (6.5.1) is satisfied.
(b) For any solution $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R}$ of (6.5.1), the point $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ with $\boldsymbol{w}=\boldsymbol{\Phi} \boldsymbol{\alpha}$ is a solution of the primal training problem (2.4.3).
(c) Any solution of the system (6.5.1) is also a solution of the finite-dimensional primal training problem (6.1.2).

In a next step, we want to make a connection between the primal training problem and the associated system of nonlinear equations for the case that the solution of the former is unique.

Proposition 6.5.3 (system of nonlinear equations for the training problem)
Consider a standard training problem as in Definition 2.4.5 with differentiable loss functions and suppose that Assumption 3.1.2 holds. Suppose that the primal training problem (2.4.3) has a unique solution $\left(\boldsymbol{w}^{\star}, b^{\star}\right) \in \mathcal{F} \times \mathbb{R}$. Then, the following assertions are valid.
(a) The system (6.5.1) has a unique solution ( $\boldsymbol{\alpha}^{\star \star}, b^{\star \star}$ ).
(b) The point $\boldsymbol{\alpha}^{\star \star}$ is equal to the unique solution of the dual training problem (3.4.1) and $b^{\star \star}$ satisfies (3.6.4).
(c) The point $\left(\boldsymbol{\alpha}^{\star \star}, b^{\star \star}\right)$ is a solution of the finite-dimensional primal training problem (6.1.2).
(d) It holds that $\left(\boldsymbol{w}^{\star}, b^{\star}\right)=\left(\mathbf{\Phi} \boldsymbol{\alpha}^{\star \star}, b^{\star \star}\right)$ for $\left(\boldsymbol{w}^{\star}, b^{\star}\right)$ with $\boldsymbol{\Phi}$ as defined in Proposition 6.2.1.

Proof. We first show, that system (6.5.1) and problem (3.4.1) are solvable. For this purpose, let $\left(\boldsymbol{w}^{\star}, b^{\star}\right) \in \mathcal{F} \times \mathbb{R}$ denote the unique solution of problem (2.4.3). Then, it is easy to see that the point $\boldsymbol{\alpha}^{\star \star} \in \mathbb{R}^{n}$ defined by

$$
\begin{equation*}
\alpha_{i}^{\star \star}=-\ell_{i}^{\prime}\left(\left\langle\boldsymbol{w}^{\star}, \boldsymbol{\varphi}_{i}\right\rangle+b^{\star}\right) \quad \text { for } \quad i \in\{1, \ldots, n\} \tag{6.5.2}
\end{equation*}
$$

is a solution of (3.4.1) because it satisfies the optimality conditions given in Corollary 3.4.3. This means that $\left(\boldsymbol{w}^{\star}, b^{\star}\right)$ and $\boldsymbol{\alpha}^{\star \star}$ are solutions of the primal and dual training problem, respectively, and $\boldsymbol{w}^{\star}=\boldsymbol{\Phi} \boldsymbol{\alpha}^{\star \star}$ follows. Moreover, it follows that the point $\left(\boldsymbol{\alpha}^{\star \star}, b^{\star \star}\right)$ with $b^{\star \star}:=b^{\star}$ is a solution of system (6.5.1). This shows the existence of solutions for (3.4.1) and (6.5.1).

Now, let $\left(\boldsymbol{\alpha}^{\star \star}, b^{\star \star}\right) \in \mathbb{R}^{n} \times \mathbb{R}$ be a solution of (6.5.1). It follows that $\boldsymbol{\alpha}^{\star \star}$ is a solution of the finite-dimensional primal training problem (6.1.2) because $\nabla \psi_{\mathrm{pf}}\left(\boldsymbol{\alpha}^{\star \star}, b^{\star \star}\right)=\mathbf{0}$, cf. Proposition 6.1.3. Due to Proposition 6.1.1, the point $\left(\boldsymbol{w}, b^{\star \star}\right)$ with $\boldsymbol{w}=\boldsymbol{\Phi} \boldsymbol{\alpha}^{\star \star}$ is a solution of (2.4.3). But since the solution of this problem is unique, $\boldsymbol{w}=\boldsymbol{w}^{\star}$ and $b^{\star \star}=b^{\star}$. This means that, (6.5.2) is a part of the system (6.5.1) and hence ( $\boldsymbol{\alpha}^{\star \star}, b^{\star \star}$ ) is the unique solution of this system. Additionally, assertion (b) follows by means of Proposition 6.5.1. Moreover, (c) follows from $\nabla \psi_{\text {pf }}\left(\boldsymbol{\alpha}^{\star \star}, b^{\star \star}\right)=\mathbf{0}$.

## 7 Sequential Minimal Optimization for the Dual Training Problem

After the extensive theoretical investigation of the primal and dual training problem, we want to focus on solution methods subsequently. A well known and state-of-theart method for training SVMs is the so-called Sequential Minimal Optimization (SMO) method, which is the subject of the present chapter.

We start by explaining the basic idea of the approach (Section 7.1). Then, we derive a more detailed theoretical overview concerning the formulation of dual optimality conditions (Section 7.2) and the relationship between so-called violating pairs and the directional derivative of the objective function (Section 7.3). Motivated by the previous findings, we propose a convergence framework (Section 7.4) using a particularly defined first-order optimality measure which can help to prove convergence for general iterative methods for convex optimization problems. In particular, this framework paves the way for a novel approach providing a simple convergence proof for a broad class of SMO-type methods (Section 7.8). As intermediate steps, a connection between violating pairs and the first-order optimality measure is considered (Section 7.5), a new optimality measure is introduced (Section 7.6), and working set selection rules are recalled and extended (Section 7.7). Afterwards, some ideas for improving the practical performance of this type of solution method are presented (Section 7.9). Finally, the well-known shrinking approach is justified with further theoretical results (Section 7.10) extending what is known in the literature.

### 7.1 Basic Idea of Decomposition Methods

As initial point for deriving a solution method for the dual training problem, we summarize the necessary assertions from Section 3.1. To keep the notation simple, we restrict the subsequent investigation to the standard training problem. Note that the major part of the following considerations can be transferred to similar training problem (for instance the $\nu$-SVM and its dual problem in Proposition 5.2.4) without too many modifications.

Recall that the dual problem for a standard training problem can be written as

$$
\begin{equation*}
\min _{\boldsymbol{\alpha} \in \mathbb{R}^{n}} \psi_{\mathrm{d}}(\boldsymbol{\alpha}) \quad \text { s.t. } \quad \mathbf{1}^{\top} \boldsymbol{\alpha}=0 \tag{7.1.1}
\end{equation*}
$$

with an objective function $\psi_{\text {d }}$ defined by

$$
\psi_{\mathrm{d}}(\boldsymbol{\alpha}):=\frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}+\sum_{i=1}^{n} \ell_{i}^{\star}\left(-\alpha_{i}\right)
$$

according to Corollary 3.4.1. By observing the structure of this problem, one can see that there is an obvious linear constraint $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$, which has to be handled by a solution method in a certain way. Moreover, there are some implicit constraints which are implied by the domain of the conjugate loss functions. By this we mean that the constraint $-\alpha_{i} \in \operatorname{dom}\left(\ell_{i}^{\star}\right)$ can be added to the problem for each $i \in\{1, \ldots, n\}$. These bound constraints can be written as $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ for short. The combination of both kinds of constraints must be handled suitably in the derivation of solution methods.

Recall that Assumption 3.1.2 ensures that there always exists a solution of this problem due to Corollary 3.4.1. In particular, there exists a feasible point

$$
\boldsymbol{\alpha}^{0} \in\left\{\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right) \mid \mathbf{1}^{\top} \boldsymbol{\alpha}=0\right\} .
$$

Now, the basic idea of decomposition methods is to start from such a feasible point and update a restricted set of variables in each step of the method. The potential in realizing this kind of strategy is that the considered subproblems are far easier to solve than the original problem. Moreover, a deliberate implementation of the approach makes it is possible to reduce the memory demands by not computing the full kernel matrix at once.

Because of the equality constraint $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ which is present in problem (7.1.1) it is necessary to update at least two variables in order to improve the objective function value while preserving feasibility of the iterates. This leads directly to the idea of Sequential Minimal Optimization (SMO) in which a subproblem with two variables is solved exactly at each step. Later on we will see that it is already sufficient to restrict the update to two variables in order to obtain convergence of a decomposition method if the choice of the variables is done suitably and the update leads to a sufficient improvement.

### 7.2 Optimality Conditions Revisited

Subsequently, we consider the dual optimality conditions again from a practical point of view. For this purpose, let $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ satisfying $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ be chosen arbitrarily. Following Remark 3.6.4, the vector $\boldsymbol{\alpha}$ is optimal if and only if there exists some $b \in \mathbb{R}$ satisfying

$$
\begin{equation*}
b \in \bigcap_{i=1}^{n}\left(\partial \ell_{i}^{\star}\left(-\alpha_{i}\right)-\frac{1}{\lambda} \sum_{j=1}^{n} \alpha_{j}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}_{j}\right\rangle\right) . \tag{7.2.1}
\end{equation*}
$$

Because the subdifferential of the conjugate loss function (depending on a single variable only) is equal to a closed interval, this condition can also be written as

$$
\begin{equation*}
b \in \bigcap_{i=1}^{n}\left[b_{i}^{-}(\boldsymbol{\alpha}), b_{i}^{+}(\boldsymbol{\alpha})\right] \tag{7.2.2}
\end{equation*}
$$

with

$$
\begin{equation*}
b_{i}^{-}(\boldsymbol{\alpha}):=\inf \partial \ell_{i}^{\star}\left(-\alpha_{i}\right)-\tilde{d}_{i}(\boldsymbol{\alpha}) \quad \text { and } \quad b_{i}^{+}(\boldsymbol{\alpha}):=\sup \partial \ell_{i}^{\star}\left(-\alpha_{i}\right)-\tilde{d}_{i}(\boldsymbol{\alpha}), \tag{7.2.3}
\end{equation*}
$$

where we used the abbreviation

$$
\tilde{d}_{i}(\boldsymbol{\alpha}):=\frac{1}{\lambda} \sum_{j=1}^{n} \alpha_{j}\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}_{j}\right\rangle .
$$

Note that the vector containing the entries $\tilde{d}_{i}(\boldsymbol{\alpha})$ for $i \in\{1, \ldots, n\}$ can be written shortly as $\tilde{\boldsymbol{d}}(\boldsymbol{\alpha})=\frac{1}{\lambda} \boldsymbol{K} \boldsymbol{\alpha}$. Using this term, the values of $b_{i}^{-}(\boldsymbol{\alpha})$ and $b_{i}^{+}(\boldsymbol{\alpha})$ can be easily obtained if the conjugate loss functions are given in a closed form. In particular, this is the common case for the practical applications which we consider. In order to avoid further issues coming from infinite values of $b_{i}^{-}$or $b_{i}^{+}$, we suppose that the following condition is satisfied.

Assumption 7.2.1 (non-degenerate dual problem)
Consider a dual training problem of the form (7.1.1). We assume that for each $\boldsymbol{\alpha} \in$ $\operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ with $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ there exist indices $i^{+}, i^{-} \in\{1, \ldots, n\}$ such that

$$
b_{i^{-}}^{-}(\boldsymbol{\alpha})>-\infty \quad \text { and } \quad b_{i^{+}}^{+}(\boldsymbol{\alpha})<\infty
$$

Note that this assumption is not restrictive in practice because it basically demands that a given point cannot lay either on the lower or on the upper bound of the domain of each conjugate loss function at the same time.

An immediate observation from the optimality condition in (7.2.1) is the following equivalent formulation.

Observation 7.2 .2 (practical optimality condition for the dual training problem)
A vector $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ satisfying $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ is optimal for the dual training problem (7.1.1) if and only if

$$
\begin{equation*}
r(\boldsymbol{\alpha}):=\max _{l=1}^{n} b_{l}^{-}(\boldsymbol{\alpha})-\min _{l=1}^{n} b_{l}^{+}(\boldsymbol{\alpha}) \leq 0 \tag{7.2.4}
\end{equation*}
$$

with $b_{l}^{-}(\boldsymbol{\alpha})$ and $b_{l}^{+}(\boldsymbol{\alpha})$ as defined in (7.2.3). We will call the function $r$ defined above violation measure for the dual training problem.

Indeed, this is true because the inequality (7.2.4) is satisfied if and only if the intersection of all intervals in (7.2.2) is non-empty. Then, any

$$
\begin{equation*}
b^{\star} \in\left[\max _{l=1}^{n} b_{l}^{-}(\boldsymbol{\alpha}), \min _{l=1}^{n} b_{l}^{+}(\boldsymbol{\alpha})\right] \tag{7.2.5}
\end{equation*}
$$

satisfies (7.2.2) and the equivalent condition (7.2.1).
For practical purposes it is convenient to make a note of the following fact.

Observation 7.2 .3 (computational effort for the violation measure)
By definition of r it directly follows that the evaluation of the violation measure is possible in $\mathcal{O}(n)$ steps.

Recall that the $\boldsymbol{w}$-part of the solution of the primal training problem is uniquely determined due to Corollary 3.8.1 and can be determined from a solution of the dual problem using the Representer Theorem, see Corollary 3.5.1. Now, by means of (7.2.5) it is also possible to find optimal values of the variable $b$ in the primal training problem. This idea gives rise to the definition of another optimality measure which is based on the primal-dual gap as follows.

Observation 7.2 .4 (induced primal-dual gap)
For any dual feasible point $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$, the point $\boldsymbol{\omega}(\boldsymbol{\alpha}):=(\boldsymbol{w}(\boldsymbol{\alpha}), b(\boldsymbol{\alpha})) \in \mathcal{F} \times \mathbb{R}$ defined by

$$
\boldsymbol{w}(\boldsymbol{\alpha}):=\frac{1}{\lambda} \sum_{i=1}^{n} \alpha_{i} \boldsymbol{\varphi}_{i} \quad \text { and } \quad b(\boldsymbol{\alpha}):=\frac{1}{2}\left(\max _{l=1}^{n} b_{l}^{-}(\boldsymbol{\alpha})+\min _{l=1}^{n} b_{l}^{+}(\boldsymbol{\alpha})\right)
$$

is a feasible estimate for a primal solution based on $\boldsymbol{\alpha}$. Moreover, the induced primaldual gap

$$
\Delta^{\star}(\boldsymbol{\alpha}):=\Delta_{\mathrm{pd}}(\boldsymbol{\omega}(\boldsymbol{\alpha}), \boldsymbol{\alpha})
$$

is an optimality measure, i.e., $\Delta^{\star}(\boldsymbol{\alpha}) \geq 0$ for all $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ and $\Delta^{\star}(\boldsymbol{\alpha})=0$ if and only if $\boldsymbol{\alpha}$ is a solution of (7.1.1).

An important theoretical question is whether the optimality measure can be used to estimate the distance of a given point to the solution set of the optimization problem. This is also a desirable property in practice because it enables the definition of stopping criteria in iterative methods which lead to approximate solutions with certain guaranteed properties. In the subsequent example we show that we cannot expect the violation measure to be useful for this purpose in general.

Example 7.2 .5 (piecewise constant violation measure)
Consider a dual training problem (7.1.1) with $n=2$ samples and conjugate loss functions

$$
\ell_{1}^{\star}\left(-\alpha_{1}\right):=\left\{\begin{array}{ll}
\alpha_{1}, & \text { if } \alpha_{1} \geq 0, \\
\infty, & \text { otherwise, }
\end{array} \quad \text { and } \quad \ell_{2}^{\star}\left(-\alpha_{2}\right):= \begin{cases}0, & \text { if } \alpha_{2} \leq 0 \\
\infty, & \text { otherwise } .\end{cases}\right.
$$

Let $\boldsymbol{K}:=\boldsymbol{O} \in \mathbb{R}^{2 \times 2}$. Then, $\operatorname{dom}\left(\psi_{\mathrm{d}}\right)=\left\{\boldsymbol{\alpha} \in \mathbb{R}^{2} \mid \alpha_{1} \geq 0\right.$ and $\left.\alpha_{2} \leq 0\right\}$, and we can
compute

$$
\begin{aligned}
& b_{1}^{-}(\boldsymbol{\alpha})=\inf \partial \ell_{1}^{\star}\left(-\alpha_{1}\right) \quad=-1, \quad b_{1}^{+}(\boldsymbol{\alpha})=\sup \partial \ell_{1}^{\star}\left(-\alpha_{1}\right) \quad= \begin{cases}-1, & \text { if } \alpha_{1}>0, \\
\infty, & \text { if } \alpha_{1}=0,\end{cases} \\
& b_{2}^{+}(\boldsymbol{\alpha})=\sup \partial \ell_{2}^{\star}\left(-\alpha_{2}\right) \quad=0, \quad b_{2}^{-}(\boldsymbol{\alpha})=\inf \partial \ell_{2}^{\star}\left(-\alpha_{2}\right) \quad= \begin{cases}0, & \text { if } \alpha_{2}<0, \\
-\infty, & \text { if } \alpha_{2}=0,\end{cases}
\end{aligned}
$$

For any $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ with $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$, it follows that

$$
\max _{l=1}^{n} b_{l}^{-}(\boldsymbol{\alpha})=\left\{\begin{array}{ll}
-1, & \text { if } \alpha_{2}=0, \\
0, & \text { if } \alpha_{2}<0,
\end{array} \quad \text { and } \quad \min _{l=1}^{n} b_{l}^{+}(\boldsymbol{\alpha})= \begin{cases}0, & \text { if } \alpha_{1}=0 \\
-1, & \text { if } \alpha_{1}>0\end{cases}\right.
$$

which yields

$$
r(\boldsymbol{\alpha})= \begin{cases}-1, & \text { if } \alpha_{1}=0 \\ 1, & \text { if } \alpha_{1}>0\end{cases}
$$

due to the definition in (7.2.4). Now, it can be seen that the unique solution of the problem is given by $\boldsymbol{\alpha}^{\star}=(0,0)$ because of Observation 7.2.2. On the other hand, $r(\boldsymbol{\alpha})=1$ for all non-optimal $\boldsymbol{\alpha}$, which means that the value of $r$ cannot be used to derive any information about the distance between $\boldsymbol{\alpha}$ and the solution.

Note that the definition of the problem is chosen to result in a rather simple structure, which can be analyzed easily by hand. However, the resulting problem usually also occurs for practical problems that are not constructed in that particular way.

### 7.3 Maximal Violating Pair and Directional Derivative

As we have seen in Observation 7.2.2 the difference between the maximal value in $\left\{b_{l}^{-}(\boldsymbol{\alpha})\right\}_{l=1}^{n}$ and the minimal value in $\left\{b_{l}^{+}(\boldsymbol{\alpha})\right\}_{l=1}^{n}$ plays an important role for the conclusion of optimality. This immediately leads to the idea that variables associated with pairs realizing a high value for this difference should be chosen to be optimized in a decomposition method. Subsequently, we argue that the resulting strategy can be easily motivated theoretically because it corresponds to the use of the steepest descent direction. Note that the choice of violating pairs for the selection of the working sets dates back to [KSBM01], see also [CL11] and [FCL05]. It can be formalized as follows.

Definition 7.3 .1 (maximal violating pair)
Let $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ with $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ be given. A pair $(i, j) \in\{1, \ldots, n\}^{2}$ of indices with

$$
\begin{equation*}
i \in \underset{l \in\{1, \ldots, n\}}{\operatorname{argmax}} b_{l}^{-}(\boldsymbol{\alpha}) \quad \text { and } \quad j \in \underset{l \in\{1, \ldots, n\}}{\operatorname{argmin}} b_{l}^{+}(\boldsymbol{\alpha}) \tag{7.3.1}
\end{equation*}
$$

is called maximal violating pair (MVP).

Observe that a maximal violating pair can be determined in $\mathcal{O}(n)$ step according to Observation 7.2.3.

In order to see that the definition of the MVP is closely connected to the direction of the steepest descent (in a certain sense which we have make clear later), we use the following observation.

Proposition 7.3 .2 (violation and directional derivative)
The directional derivatives of $\psi_{\mathrm{d}}$ (cf. Definition A.4.1) in the directions $\pm \boldsymbol{e}^{i}$ are given by

$$
\begin{equation*}
\psi_{\mathrm{d}}^{\circ}\left(\boldsymbol{\alpha} ; \boldsymbol{e}^{i}\right)=-b_{i}^{-}(\boldsymbol{\alpha}) \quad \text { and } \quad \psi_{\mathrm{d}}^{\circ}\left(\boldsymbol{\alpha} ;-\boldsymbol{e}^{i}\right)=b_{i}^{+}(\boldsymbol{\alpha}) \tag{7.3.2}
\end{equation*}
$$

for all $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ and all $i \in\{1, \ldots, n\}$. In particular,

$$
\begin{equation*}
\psi_{\mathrm{d}}^{\circ}\left(\boldsymbol{\alpha} ; \boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right)=b_{j}^{+}(\boldsymbol{\alpha})-b_{i}^{-}(\boldsymbol{\alpha}) \quad \text { for all } \quad i, j \in\{1, \ldots, n\} \tag{7.3.3}
\end{equation*}
$$

and

$$
\begin{equation*}
r(\boldsymbol{\alpha})=\max _{(i, j) \in\{1, \ldots, n\}^{2}}\left\{-\psi_{\mathrm{d}}^{\circ}\left(\boldsymbol{\alpha} ; \boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right)\right\} \tag{7.3.4}
\end{equation*}
$$

for all $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$.

Proof. It is well-known that the directional derivative of a convex scalar function can be characterized by means of its subdifferential, see [BL10, Theorem 3.1.8], for instance. A short calculation yields that the subdifferential of $\psi_{\mathrm{d}}^{(i)}(\tau):=\psi_{\mathrm{d}}\left(\boldsymbol{\alpha}+\tau \boldsymbol{e}^{i}\right)$ is given by

$$
\partial \psi_{\mathrm{d}}^{(i)}(\tau)=\frac{1}{\lambda} K_{i i}+\frac{1}{\lambda}[\boldsymbol{K} \boldsymbol{\alpha}]_{i}-\partial \ell_{i}^{\star}\left(-\left(\alpha_{i}+\tau\right)\right) .=\frac{1}{\lambda} K_{i i}+\tilde{d}_{i}(\boldsymbol{\alpha})-\partial \ell_{i}^{\star}\left(-\left(\alpha_{i}+\tau\right)\right) .
$$

In particular, we obtain

$$
\psi_{\mathrm{d}}^{\circ}\left(\boldsymbol{\alpha} ; \boldsymbol{e}^{i}\right)=\left(\psi_{\mathrm{d}}^{(i)}\right)^{\circ}(0 ; 1)=\sup \partial \psi_{\mathrm{d}}^{(i)}(0)=\tilde{d}(\boldsymbol{\alpha})+\sup -\partial \ell_{i}^{\star}\left(-\alpha_{i}\right)=-b_{i}^{-}(\boldsymbol{\alpha})
$$

In the same way it follows that

$$
\psi_{\mathrm{d}}^{\circ}\left(\boldsymbol{\alpha} ;-\boldsymbol{e}^{i}\right)=\left(\psi_{\mathrm{d}}^{(i)}\right)^{\circ}(0 ;-1)=\sup -\partial \psi_{\mathrm{d}}^{(i)}(0)=\sup \partial \ell_{i}^{\star}\left(-\alpha_{i}\right)-\tilde{d}(\boldsymbol{\alpha})=b_{i}^{+}(\boldsymbol{\alpha}) .
$$

This shows that (7.3.2) is indeed correct. In the same way, one can see that (7.3.3) holds. Finally, observe that (7.3.4) follows directly from the definition of $r$ in (7.4.1) and (7.3.3).

The relation given in (7.3.4) shows that the optimality condition which we derived in Observation 7.2.2 is directly connected to a particular descent direction. Namely, we see that whenever $r(\boldsymbol{\alpha})>0$ (i.e., the point $\boldsymbol{\alpha}$ is not optimal due to Observation 7.2.2) there exists a feasible direction of the form $\boldsymbol{e}^{i}-\boldsymbol{e}^{j}$ with a pair of indices $(i, j)$ that is a descent direction for the dual objective function $\psi_{\mathrm{d}}$. Here, feasibility of the direction means that
(a) the objective function is finite for all points in that direction which are sufficiently close to $\boldsymbol{\alpha}$, and
(b) the linear constraint of problem (7.1.1) remains satisfied by construction because

$$
\mathbf{1}^{\top}\left(\boldsymbol{\alpha}+\tau\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right)\right)=\mathbf{1}^{\top} \boldsymbol{\alpha}=0 .
$$

The directional derivative is only a local measure of the ascent (or descent) of objective function. In order to incorporate information about the possible step size in a particular direction, we additionally consider the following terms.

Definition 7.3 .3 (bounds on $\boldsymbol{\alpha}$ and maximal feasible step size)
We define

$$
\bar{\alpha}_{i}:=\sup \operatorname{dom}\left(\ell_{i}^{\star}(-\cdot)\right) \quad \text { and } \quad \underline{\alpha}_{j}:=\inf \operatorname{dom}\left(\ell_{j}^{\star}(-\cdot)\right)
$$

and denote by

$$
\begin{equation*}
\epsilon^{i j}(\boldsymbol{\alpha}):=\min \left\{\bar{\alpha}_{i}-\alpha_{i}, \alpha_{j}-\underline{\alpha}_{j}\right\} \tag{7.3.5}
\end{equation*}
$$

the maximal feasible step size (or almost feasible step size, if the domain of a conjugate loss function is open) for the pair $(i, j)$ at the point $\boldsymbol{\alpha}$.

### 7.4 A General Convergence Framework

In order to prove convergence of the class of decomposition methods which will be introduced below, we first provide some more general convergence framework. For this purpose, we suppose subsequently that the considered objective function is differentiable on its domain in the following sense.

Assumption 7.4.1 (differentiability of convex function on its domain)
Let $\psi: \mathbb{R}^{n} \rightarrow \mathbb{R} \cup\{+\infty\}$ be a convex function and suppose that $\psi_{\text {ext }}: D \rightarrow \mathbb{R}$ is a convex and continuously differentiable function defined on an open set $D \supseteq \operatorname{dom}(\psi)$ such that $\psi_{\text {ext }}(\boldsymbol{\alpha})=\psi(\boldsymbol{\alpha})$ for all $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$.

### 7.4.1 First-Order Optimality Measure

In the setting of Assumption 7.4.1, we can use the differentiability of $\psi_{\text {ext }}$ to define a particular optimality measure as follows.

Theorem 7.4.2 (first-order optimality measure)
Let Assumption 7.4.1 be satisfied and define $R: \operatorname{dom}(\psi) \times \mathbb{R}_{+} \rightarrow \mathbb{R} \cup\{+\infty\}$ by

$$
\begin{equation*}
R(\boldsymbol{\alpha}, \rho):=\sup \left\{-\nabla \psi_{\mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) \mid \tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi) \cap \mathcal{B}(\boldsymbol{\alpha}, \rho)\right\} \tag{7.4.1}
\end{equation*}
$$

Then, for all $\rho>0$ it holds that
(a) $R(\boldsymbol{\alpha}, \rho) \geq 0$ for all $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$
(b) $R(\boldsymbol{\alpha}, \rho)=0$ if and only if $\boldsymbol{\alpha}$ is a minimizer of $\psi$, and,
(c) the function $\boldsymbol{\alpha} \mapsto R(\boldsymbol{\alpha}, \rho)$ is continuous on $\operatorname{dom}(\psi)$.

Proof. First, note that for all $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ and all $\rho>0$, it follows that $R(\boldsymbol{\alpha}, \rho) \geq 0$ because $\boldsymbol{\alpha} \in \operatorname{dom}(\psi) \cap \mathcal{B}(\boldsymbol{\alpha}, \rho)$, i.e., assertion (a) is certainly true.
Moreover, since the function $\psi_{\text {ext }}$ is continuously differentiable on an open set containing $\operatorname{dom}(\psi)$, a point $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ is a minimizer of $\psi$ if and only if

$$
\begin{equation*}
\nabla \psi_{\mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) \geq 0 \quad \text { for all } \quad \tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi) \tag{7.4.2}
\end{equation*}
$$

In particular, this implies that $R(\boldsymbol{\alpha}, \rho)=0$ for all $\rho>0$, i.e., any minimizer $\boldsymbol{\alpha}$ of $\psi$ satisfies $R(\boldsymbol{\alpha}, \rho)=0$.
On the other hand, let $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ with $R(\boldsymbol{\alpha}, \rho)=0$ for some $\rho>0$. In the following, we show that (7.4.2) is satisfied. For this purpose, let $\tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi)$ be chosen arbitrarily. If $\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\| \leq \rho$, we directly obtain

$$
-\nabla \psi_{\mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) \leq R(\boldsymbol{\alpha}, \rho)=0
$$

If $\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\|>\rho$, consider the point

$$
\widehat{\boldsymbol{\alpha}}:=\boldsymbol{\alpha}+\rho \frac{\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}}{\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\|}=\left(1-\frac{\rho}{\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\|}\right) \boldsymbol{\alpha}+\frac{\rho}{\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\|} \tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi) .
$$

Then, it follows that

$$
-\nabla \psi_{\mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha})=-\nabla \psi_{\mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\widehat{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) \frac{\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\|}{\rho} \leq R(\boldsymbol{\alpha}, \rho) \frac{\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\|}{\rho}=0
$$

because $\|\widehat{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\|=\rho$. Taking both cases together, we see that (7.4.2) holds, i.e., $\boldsymbol{\alpha}$ is a minimizer of $\psi$. This proves part (b).

In order to prove part (c), let $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in \mathbb{N}} \subseteq \operatorname{dom}(\psi)$ be some sequence with $\boldsymbol{\alpha}^{\star}:=$ $\lim _{k \rightarrow \infty} \boldsymbol{\alpha}^{k} \in \operatorname{dom}(\psi)$. Because $\psi$ is continuously differentiable on $\operatorname{dom}(\psi)$, it follows that $\lim _{k \rightarrow \infty} \nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{k}\right)=\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{\star}\right)$. Let $\tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi)$ be chosen arbitrarily. If $\left\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right\| \leq \rho$ it follows that

$$
-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right) \leq R\left(\boldsymbol{\alpha}^{k}, \rho\right) .
$$

Otherwise, considering the point $\widehat{\boldsymbol{\alpha}}^{k}:=\boldsymbol{\alpha}^{k}+\rho \frac{\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}}{\left\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right\|} \in \operatorname{dom}(\psi) \cap \mathcal{B}\left(\boldsymbol{\alpha}^{k}, \rho\right)$ we get $-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right)=-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\widehat{\boldsymbol{\alpha}}^{k}-\boldsymbol{\alpha}^{k}\right) \rho^{-1}\left\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right\| \leq R\left(\boldsymbol{\alpha}^{k}, \rho\right) \rho^{-1}\left\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right\|$.

Then, for $\tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi) \cap \mathcal{B}\left(\boldsymbol{\alpha}^{\star}, \rho\right)$ using the triangle inequality to estimate $\left\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right\|$ this implies

$$
\begin{aligned}
-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right) & \leq R\left(\boldsymbol{\alpha}^{k}, \rho\right) \rho^{-1}\left(\left\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{\star}\right\|+\left\|\boldsymbol{\alpha}^{\star}-\boldsymbol{\alpha}^{k}\right\|\right) \\
& \leq R\left(\boldsymbol{\alpha}^{k}, \rho\right)\left(1+\rho^{-1}\left\|\boldsymbol{\alpha}^{\star}-\boldsymbol{\alpha}^{k}\right\|\right) .
\end{aligned}
$$

Taking both cases together, it is shown that

$$
-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right) \leq R\left(\boldsymbol{\alpha}^{k}, \rho\right)\left(1+\rho^{-1}\left\|\boldsymbol{\alpha}^{\star}-\boldsymbol{\alpha}^{k}\right\|\right) .
$$

Hence, we obtain

$$
\begin{aligned}
-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{\star}\right)= & -\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right) \\
& -\left(\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right) \\
& -\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top}\left(\boldsymbol{\alpha}^{k}-\boldsymbol{\alpha}^{\star}\right) \\
\leq & R\left(\boldsymbol{\alpha}^{k}, \rho\right)\left(1+\rho^{-1}\left\|\boldsymbol{\alpha}^{\star}-\boldsymbol{\alpha}^{k}\right\|\right) \\
& +\left\|\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)-\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{k}\right)\right\| \cdot\left\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{k}\right\| \\
& +\left\|\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{\star}\right)\right\| \cdot\left\|\boldsymbol{\alpha}^{k}-\boldsymbol{\alpha}^{\star}\right\| .
\end{aligned}
$$

Taking the limit $k \rightarrow \infty$, this implies

$$
\begin{align*}
R\left(\boldsymbol{\alpha}^{\star}, \rho\right) & =\sup \left\{-\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{\star}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}^{\star}\right) \mid \tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi) \cap \mathcal{B}\left(\boldsymbol{\alpha}^{\star}, \rho\right)\right\} \\
& \leq \liminf _{k \rightarrow \infty} R\left(\boldsymbol{\alpha}^{k}, \rho\right) . \tag{7.4.3}
\end{align*}
$$

In the same way as above, we get

$$
-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}^{k}-\boldsymbol{\alpha}^{\star}\right) \leq R\left(\boldsymbol{\alpha}^{\star}, \rho\right)\left(1+\rho^{-1}\left\|\boldsymbol{\alpha}^{k}-\boldsymbol{\alpha}^{\star}\right\|\right),
$$

for $\tilde{\boldsymbol{\alpha}}^{k} \in \operatorname{dom}(\psi) \cap \mathcal{B}\left(\boldsymbol{\alpha}^{k}, \rho\right)$, which can be used to obtain

$$
\begin{aligned}
-\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}^{k}-\boldsymbol{\alpha}^{k}\right)= & -\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{\star}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}^{k}-\boldsymbol{\alpha}^{\star}\right) \\
& -\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{\star}\right)^{\top}\left(\boldsymbol{\alpha}^{\star}-\boldsymbol{\alpha}^{k}\right) \\
& -\left(\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{k}\right)-\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{\star}\right)\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}^{k}-\boldsymbol{\alpha}^{k}\right) \\
\leq & R\left(\boldsymbol{\alpha}^{\star}, \rho\right)\left(1+\rho^{-1}\left\|\boldsymbol{\alpha}^{k}-\boldsymbol{\alpha}^{\star}\right\|\right) \\
& +\left\|\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{\star}\right)\right\| \cdot\left\|\boldsymbol{\alpha}^{\star}-\boldsymbol{\alpha}^{k}\right\| \\
& +\left\|\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{k}\right)-\nabla \psi_{\text {ext }}\left(\boldsymbol{\alpha}^{\star}\right)\right\| \cdot\left\|\tilde{\boldsymbol{\alpha}}^{k}-\boldsymbol{\alpha}^{k}\right\|
\end{aligned}
$$

Rearranging the terms shows that

$$
\begin{aligned}
\left(1+\rho^{-1}\left\|\boldsymbol{\alpha}^{k}-\boldsymbol{\alpha}^{\star}\right\|\right)^{-1}( & -\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\tilde{\boldsymbol{\alpha}}^{k}-\boldsymbol{\alpha}^{k}\right) \\
& -\left\|\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)\right\| \cdot\left\|\boldsymbol{\alpha}^{\star}-\boldsymbol{\alpha}^{k}\right\| \\
& \left.-\left\|\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)\right\| \cdot\left\|\tilde{\boldsymbol{\alpha}}^{k}-\boldsymbol{\alpha}^{k}\right\|\right) \leq R\left(\boldsymbol{\alpha}^{\star}, \rho\right) .
\end{aligned}
$$

Now, taking the supremum over all $\tilde{\boldsymbol{\alpha}}^{k} \in \operatorname{dom}(\psi) \cap \mathcal{B}\left(\boldsymbol{\alpha}^{k}, \rho\right)$ and considering the limit $k \rightarrow \infty$ shows that

$$
\begin{equation*}
\limsup _{k \rightarrow \infty} R\left(\boldsymbol{\alpha}^{k}, \rho\right) \leq R\left(\boldsymbol{\alpha}^{\star}, \rho\right) . \tag{7.4.4}
\end{equation*}
$$

By taking together the inequalities (7.4.3) and (7.4.4) it is shown that $\boldsymbol{\alpha} \mapsto R(\boldsymbol{\alpha}, \rho)$ is continuous on $\operatorname{dom}(\psi)$.

Remark 7.4.3 (about the norm in the optimality measure)
Note that the properties of the function $R$ do not depend on the particular choice of the norm which defines the neighborhood $\mathcal{B}(\boldsymbol{\alpha}, \rho)$ in the definition of $R$. Most notably, one could also use the maximum norm there, which can make the evaluation of $R$ easier in certain cases.

### 7.4.2 Convergence under Sufficient Decrease Condition

As a tool to formulate appropriate assumptions for proving convergence to minimizers, we consider the following notion of a forcing function which was introduced and applied in [Dan70].

Definition 7.4.4 (forcing function, cf. [Dan70, Definition 3.1])
A function $c: \mathbb{R}_{+} \rightarrow \mathbb{R}_{+}$satisfying

$$
\lim _{k \rightarrow \infty} c\left(r_{k}\right)=0 \Longrightarrow \lim _{k \rightarrow \infty} r_{k}=0
$$

for any sequence $\left\{r_{k}\right\}_{k \in \mathbb{N}} \subseteq \mathbb{R}_{+}$is called forcing function.

Using this definition and the first-order optimality measure defined in Theorem 7.4.2, we can formulate the following general convergence framework for methods solving convex optimization problems. In order to guarantee convergence to a global optimizer, we only require a sufficient decrease of the objective function value in an infinite number of iterations and the convergence of the corresponding subsequence of iterates.

Theorem 7.4.5 (convergence under sufficient decrease condition)
Suppose that the assumptions of Theorem 7.4 .2 are satisfied. Let $c: \mathbb{R}_{+} \rightarrow \mathbb{R}_{+}$be a forcing function and let $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in \mathbb{N}} \subseteq \operatorname{dom}(\psi)$ be a sequence of points such that the sequence $\left\{\psi\left(\boldsymbol{\alpha}^{k}\right)\right\}_{k \in \mathbb{N}}$ is non-increasing and bounded from below. Suppose that there exists a convergent subsequence $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in K}$ of $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in \mathbb{N}}$ such that

$$
\begin{equation*}
\psi\left(\boldsymbol{\alpha}^{k}\right)-\psi\left(\boldsymbol{\alpha}^{k+1}\right) \geq c\left(R\left(\boldsymbol{\alpha}^{k}, \rho\right)\right) \tag{7.4.5}
\end{equation*}
$$

holds for some $\rho>0$. Then,

$$
\lim _{k \rightarrow \infty} \psi\left(\boldsymbol{\alpha}^{k}\right)=\inf _{\boldsymbol{\alpha} \in \mathbb{R}^{n}} \psi(\boldsymbol{\alpha})
$$

and any accumulation point of $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in \mathbb{N}}$ is a minimizer of $\psi$.

Proof. Because the sequence of function values is non-increasing and bounded from below, the limit $\psi^{\star}:=\lim _{k \rightarrow \infty} \psi\left(\boldsymbol{\alpha}^{k}\right)$ exists. Let $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in K}$ with $\tilde{\boldsymbol{\alpha}}:=\lim _{k \in K} \boldsymbol{\alpha}^{k}$ be a
subsequence of $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in \mathbb{N}}$ such that (7.4.5) holds for all $k \in K$. In particular, we obtain

$$
\lim _{k \in K} c\left(R\left(\boldsymbol{\alpha}^{k}, \rho\right)\right)=0
$$

which implies $\lim _{k \rightarrow \infty} R\left(\boldsymbol{\alpha}^{k}, \rho\right)=0$.
Because the function $\boldsymbol{\alpha} \mapsto R(\boldsymbol{\alpha}, \rho)$ is continuous due to Theorem 7.4.2(c), it follows that $R(\tilde{\boldsymbol{\alpha}}, \rho)=0$. Using Theorem 7.4.2(a) we obtain that $\tilde{\boldsymbol{\alpha}}$ is a minimizer of $\psi$. Finally, since the sequence of function values is non-increasing, we get

$$
\psi^{\star}=\lim _{k \rightarrow \infty} \psi\left(\boldsymbol{\alpha}^{k}\right)=\inf _{\boldsymbol{\alpha} \in \mathbb{R}^{n}} \psi(\boldsymbol{\alpha})=\psi(\tilde{\boldsymbol{\alpha}}),
$$

which also shows that any accumulation point of the sequence $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in \mathbb{N}}$ must be a minimizer of $\psi$.

Note that the assertions of Theorem 7.4.2 and Theorem 7.4.5 do not depend on the particular structure of the dual training problem (7.1.1). In particular, those results could also be applied to other optimization problems.

Moreover, a condition comparable to (7.4.5) also occurs in the literature for the definition of efficient step sizes in the context of unconstrained optimization problems, see, for instance, [GK99, Definition 4.5]. There, a particular direction is considered, and the condition restricts the choice of the step size. In contrary, the condition (7.4.5) leaves more room because it does not explicitly assume a line search approach.

In the following, we apply the previous result to a particular iterative optimization framework for solving convex optimization problems, which is still general enough to capture some typical applications. For this reason, we consider two possible line search approaches for which it can be shown that a sufficient decrease is possible under assumptions that are not too restrictive.

### 7.4.3 Exact Line Search

We start with a setting using an exact line search as follows.

Lemma 7.4.6 (decrease with exact line search)
Let Assumption 7.4.1 hold and suppose that $\nabla \psi_{\text {ext }}$ is Lipschitz continuous on the level set $L_{\psi_{\text {ext }}}\left(\psi\left(\boldsymbol{\alpha}^{0}\right)\right)$ for some $\boldsymbol{\alpha}^{0} \in \operatorname{dom}(\psi)$. Let $\boldsymbol{\alpha}^{k} \in L_{\psi}\left(\psi\left(\boldsymbol{\alpha}^{0}\right)\right)$ and $\boldsymbol{d}^{k} \in \mathbb{R}^{n}$ be given such that

$$
\begin{equation*}
\boldsymbol{\alpha}^{k}+\boldsymbol{d}^{k} \in \operatorname{dom}(\psi) \tag{7.4.6}
\end{equation*}
$$

and

$$
\begin{equation*}
r_{k}:=-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top} \boldsymbol{d}^{k}>0 . \tag{7.4.7}
\end{equation*}
$$

Then, any point $\boldsymbol{\alpha}^{k+1}:=\boldsymbol{\alpha}^{k}+\tau_{k} \boldsymbol{d}^{k}$ with

$$
\begin{equation*}
\tau_{k} \in \underset{\tau \in[0,1]}{\operatorname{argmin}} \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}+\tau \boldsymbol{d}^{k}\right) \tag{7.4.8}
\end{equation*}
$$

satisfies

$$
\begin{equation*}
\psi\left(\boldsymbol{\alpha}^{k}\right)-\psi\left(\boldsymbol{\alpha}^{k+1}\right) \geq \min \left\{\frac{r_{k}}{2}, \frac{r_{k}^{2}}{2 L\left\|\boldsymbol{d}^{k}\right\|^{2}}\right\} . \tag{7.4.9}
\end{equation*}
$$

Proof. Let $\boldsymbol{\alpha}^{k} \in L_{\psi}\left(\psi\left(\boldsymbol{\alpha}^{0}\right)\right)$ and $\boldsymbol{d}^{k} \in \mathbb{R}^{n}$ satisfying (7.4.6) be given. Observe that the Lipschitz continuity of $\nabla \psi_{\text {ext }}$ implies that the function value can be estimated from above by a quadratic function (see, for instance, [Zho18]), which yields

$$
\begin{equation*}
\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}+\tau \boldsymbol{d}^{k}\right)-\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right) \leq \tau \nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top} \boldsymbol{d}^{k}+\frac{L}{2} \tau^{2}\left\|\boldsymbol{d}^{k}\right\|^{2}=-\tau r_{k}+\frac{L}{2} \tau^{2}\left\|\boldsymbol{d}^{k}\right\|^{2} \tag{7.4.10}
\end{equation*}
$$

for all $\tau \in[0,1]$ with $L>0$ denoting the Lipschitz constant of $\nabla \psi_{\text {ext }}$. In the following, we determine the minimal value of the right-hand side over $\tau \in[0,1]$, which results in an upper bound for the actual descent because

$$
\begin{equation*}
\psi\left(\boldsymbol{\alpha}^{k}\right)-\psi\left(\boldsymbol{\alpha}^{k+1}\right)=\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)-\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}+\tau_{k} \boldsymbol{d}^{k}\right) \geq \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)-\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}+\tau \boldsymbol{d}^{k}\right) \tag{7.4.11}
\end{equation*}
$$

for all $\tau \in[0,1]$ due to the step size rule (7.4.8).
Let us consider the unconstrained minimum of the right-hand side in (7.4.10). For this reason, note that condition (7.4.7) implies $\boldsymbol{d}^{k} \neq 0$. This means that the minimal value is attained at

$$
\begin{equation*}
\tau_{k}^{\star}:=\frac{r_{k}}{L\left\|\boldsymbol{d}^{k}\right\|^{2}} \tag{7.4.12}
\end{equation*}
$$

We now consider two cases:

- If $\tau_{k}^{\star} \leq 1$, the inequalities (7.4.10) and (7.4.11) imply that

$$
\psi\left(\boldsymbol{\alpha}^{k}\right)-\psi\left(\boldsymbol{\alpha}^{k+1}\right) \geq \frac{r_{k}^{2}}{2 L\left\|\boldsymbol{d}^{k}\right\|^{2}} .
$$

- Otherwise, $\tau_{k}^{\star}>1$, which is equivalent to $r_{k}>L\left\|\boldsymbol{d}^{k}\right\|^{2}$. Hence, (7.4.10) and (7.4.11) can be used with $\tau=1$ to obtain

$$
\psi\left(\boldsymbol{\alpha}^{k}\right)-\psi\left(\boldsymbol{\alpha}^{k+1}\right) \geq r_{k}-\frac{L}{2}\left\|\boldsymbol{d}^{k}\right\|^{2}>\frac{r_{k}}{2} .
$$

Because at least one of these two cases must occur, we conclude that (7.4.9) holds.
Note that the basic question for the application of the previous result is whether it is possible to construct a direction satisfying (7.4.6) and (7.4.7). Subsequently, we will see that this is rather easy in the context of decomposition methods for the dual training problem.

### 7.4.4 Armijo-Type Step Size Selection

On the other hand, one could ask whether the computation of the step size according to the exact line search rule (7.4.8) is actually necessary. In order to provide an answer to this question, we show that a similar descent result also holds if an approximate line search rule is established instead.

Lemma 7.4.7 (decrease with Armijo-type step size)
Let Assumption 7.4.1 hold and suppose that $\nabla \psi_{\text {ext }}$ is Lipschitz continuous on the level set $L_{\psi_{\text {ext }}}\left(\psi\left(\boldsymbol{\alpha}^{0}\right)\right)$ for some $\boldsymbol{\alpha}^{0} \in \operatorname{dom}(\psi)$. Choose constants $\eta \in(0,1)$ and $\sigma \in(0,1)$. Let $\boldsymbol{\alpha}^{k} \in L_{\psi}\left(\psi\left(\boldsymbol{\alpha}^{0}\right)\right)$ and $\boldsymbol{d}^{k} \in \mathbb{R}^{n}$ be given such that (7.4.6) and (7.4.7) holds. Define $\boldsymbol{\alpha}^{k+1}:=\boldsymbol{\alpha}^{k}+\tau_{k} \boldsymbol{d}^{k}$ with $\tau_{k}$ being the largest value in the set $\left\{\eta^{\nu} \mid \nu \in\{0,1, \ldots\}\right\}$ satisfying the Armijo condition

$$
\begin{equation*}
\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}+\tau_{k} \boldsymbol{d}^{k}\right) \leq \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)+\sigma \tau_{k} \nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top} \boldsymbol{d}^{k} \tag{7.4.13}
\end{equation*}
$$

Then, the point $\boldsymbol{\alpha}^{k+1}$ satisfies

$$
\begin{equation*}
\psi\left(\boldsymbol{\alpha}^{k}\right)-\psi\left(\boldsymbol{\alpha}^{k+1}\right) \geq \min \left\{\sigma r_{k}, 2 \eta \sigma(1-\sigma) L^{-1} \frac{r_{k}^{2}}{\left\|\boldsymbol{d}^{k}\right\|^{2}}\right\} . \tag{7.4.14}
\end{equation*}
$$

Proof. As in the proof of Lemma 7.4.6, we consider two cases.

- If the condition (7.4.13) is satisfied for $\tau_{k}=1$, it can be used together with (7.4.7) to obtain

$$
\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)-\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k+1}\right)=\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)-\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}+\boldsymbol{d}^{k}\right) \geq-\sigma \nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top} \boldsymbol{d}^{k}=\sigma r_{k} .
$$

- Otherwise, the condition (7.4.13) must be violated for $\eta^{-1} \tau_{k}$, which reads as

$$
\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}+\eta^{-1} \tau_{k} \boldsymbol{d}^{k}\right)>\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)+\sigma \eta^{-1} \tau_{k} \nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top} \boldsymbol{d}^{k}=\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)-\sigma \eta^{-1} \tau_{k} r_{k} .
$$

On the other hand, we can use the fact that $\nabla \psi_{\text {ext }}$ is Lipschitz as in (7.4.10) with $\tau:=\eta^{-1} \tau_{k}$ to obtain

$$
\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}+\eta^{-1} \tau_{k} \boldsymbol{d}^{k}\right) \leq \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)-\eta^{-1} \tau_{k} r_{k}+\frac{L}{2} \frac{\tau_{k}^{2}}{\eta^{2}}\left\|\boldsymbol{d}^{k}\right\|^{2}
$$

with $L>0$ denoting the Lipschitz constant of $\nabla \psi_{\text {ext }}$. By taking both inequalities together and rearranging the terms, we get

$$
\frac{\tau_{k}}{\eta^{2}}\left(-\eta(1-\sigma) r_{k}+\frac{L}{2} \tau_{k}\left\|\boldsymbol{d}^{k}\right\|^{2}\right)>0
$$

Note that $\boldsymbol{d}^{k} \neq 0$ because of (7.4.7). Hence, from (7.4.6) it follows that

$$
\tau_{k}>2 \eta(1-\sigma) L^{-1} \frac{r_{k}}{\left\|\boldsymbol{d}^{k}\right\|^{2}}
$$

Finally, the combination of this inequality with (7.4.13) and (7.4.7) shows that

$$
\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)-\psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k+1}\right) \geq \sigma \tau_{k} r_{k} \geq 2 \eta \sigma(1-\sigma) L^{-1} \frac{r_{k}^{2}}{\left\|\boldsymbol{d}^{k}\right\|^{2}}
$$

Since $\boldsymbol{\alpha}^{k} \in L_{\psi}\left(\psi\left(\boldsymbol{\alpha}^{0}\right)\right) \subseteq \operatorname{dom}(\psi)$ by assumption and $\boldsymbol{\alpha}^{k+1}=\boldsymbol{\alpha}^{k}+\tau_{k} \boldsymbol{d}^{k} \in \operatorname{dom}(\psi)$ due to (7.4.6) and the convexity of $\psi$, it follows that $\psi\left(\boldsymbol{\alpha}^{k}\right)-\psi\left(\boldsymbol{\alpha}^{k+1}\right)=\psi_{\text {ext }}\left(\boldsymbol{\alpha}^{k}\right)-\psi_{\text {ext }}\left(\boldsymbol{\alpha}^{k+1}\right)$. Thus, we conclude from the two cases that (7.4.14) holds.

Now, there is still an open question which is whether we are able to find a feasible direction (satisfying (7.4.6)) such that the decrease of the objective function (according to (7.4.9) or (7.4.14), respectively) is large enough for the application of Theorem 7.4.5. Particularly, it would be helpful to determine directions such that the value of the directional derivative $r_{k}$ defined in (7.4.7) can be estimated by $R\left(\boldsymbol{\alpha}^{k}, \rho\right)$.

### 7.5 Violating Pairs and First-Order Optimality

Note that the evaluation of the function $R$ for given point $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ and radius $\rho>0$ is not trivial in general. In particular, one cannot expect the problem to be solvable in linear time (opposed to what we observed about the violation measure $r$ in Observation 7.2.3). For this reason, we want to derive some relationships between $R$ and $r$ subsequently, which sheds some light on the application of the previous convergence result. In order to use the definition of $R$, suppose that the dual objective function meets the condition in Assumption 7.4.1 as follows. Simultaneously, we make some assumptions preventing certain edge cases resulting from degenerate variables.

Assumption 7.5 .1 (non-degenerated dual problem with smooth objective function) Consider the dual training problem (7.1.1) and let $\underline{\boldsymbol{\alpha}}$ and $\overline{\boldsymbol{\alpha}}$ be defined according to Definition 7.3.3.
(a) Let $\underline{\alpha}_{i}<\bar{\alpha}_{i}$ for all $i \in\{1, \ldots, n\}$.
(b) Let $\mathbf{1}^{\top} \underline{\boldsymbol{\alpha}}<0<\mathbf{1}^{\top} \overline{\boldsymbol{\alpha}}$.
(c) Suppose that there exists some continuously differentiable function $\psi_{\mathrm{d}, \mathrm{ext}}: D \rightarrow \mathbb{R}$ defined on an open set $D \supseteq \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ such that $\psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})=\psi_{\mathrm{d}}(\boldsymbol{\alpha})$ for all $\boldsymbol{\alpha} \in$ $\operatorname{dom}\left(\psi_{\mathrm{d}}\right)$

Note that part (a) of the assumption prevents cases in which some variables are either fixed to some particular value (if their bounds are equal) or induce infeasibility of the dual problem (if the associated conjugate loss function is nowhere finite-valued). Similarly, part (b) avoids cases where it is not possible to find a feasible point because of the linear constraint or there is only one feasible point (in which all variables are fixed to their lower or upper bound). Finally, part (c) ensures that the gradient of the objective function is available (but has to be used in the right way if some variable lays at its lower or upper bound). In the setting of Assumption 7.5.1 it is possible to define the optimality measure $R$ from Theorem 7.4.2 as follows.

Definition 7.5 .2 (optimality measure for dual problem)
Suppose that Assumption 7.5 .1 holds. For the dual training problem, we define $R$ ac-
cording to (7.4.1) with

$$
\psi(\boldsymbol{\alpha}):= \begin{cases}\psi_{\mathrm{d}}(\boldsymbol{\alpha}), & \text { if } \boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right) \text { and } \mathbf{1}^{\top} \boldsymbol{\alpha}=0 \\ +\infty, & \text { otherwise }\end{cases}
$$

and $\psi_{\text {ext }}=\psi_{\mathrm{d}, \text { ext }}$.

Note that if the objective function of the dual problem is differentiable as proposed above, the directional derivatives can be computed easily and some parts of the previously introduced notation simplifies as follows.

Remark 7.5 .3 (partial derivatives of dual objective function)
If the dual objective function is differentiable in the sense of Assumption 7.5.1, it follows by means of Proposition 7.3.2 that

$$
b_{i}^{-}(\boldsymbol{\alpha})= \begin{cases}-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}, & \text { if } \alpha_{i}<\bar{\alpha}_{i}, \\ -\infty, & \text { if } \alpha_{i}=\bar{\alpha}_{i}\end{cases}
$$

and

$$
b_{i}^{+}(\boldsymbol{\alpha})= \begin{cases}-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}, & \text { if } \alpha_{i}>\underline{\alpha}_{i}, \\ \infty, & \text { if } \alpha_{i}=\underline{\alpha}_{i}\end{cases}
$$

for all $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$.

Additionally, a first result shows that the violation term provides an upper bound for the optimality measure. This observation is consistent with the fact that $r(\boldsymbol{\alpha}) \leq 0$ whenever $\boldsymbol{\alpha}$ is a solution of the dual training problem, cf. Observation 7.2.2 and Theorem 7.4.2.

Lemma 7.5.4 (connection between violation and optimality measure - first part) Let Assumption 7.5.1 hold. Then,

$$
\begin{equation*}
R(\boldsymbol{\alpha}, \rho) \leq \rho \sqrt{n} r(\boldsymbol{\alpha}) \tag{7.5.1}
\end{equation*}
$$

for all $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ with $r(\boldsymbol{\alpha})>0$.

Proof. Suppose that $\boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi)$ with $r(\boldsymbol{\alpha})>0$ are given. Note that $\boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi)$ and $\tilde{\alpha}_{i}>\alpha_{i}$ implies $\alpha_{i}<\bar{\alpha}_{i}$ for $i \in\{1, \ldots, n\}$. In the same way we get $\alpha_{i}>\underline{\alpha}_{i}$ for $i \in\{1, \ldots, n\}$ with $\tilde{\alpha}_{i}<\alpha_{i}$. Then, by means of Remark 7.5.3 and the definition of $r$
in (7.2.4) we obtain

$$
\begin{aligned}
-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) & =\sum_{i=1}^{n}-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}\left(\tilde{\alpha}_{i}-\alpha_{i}\right) \\
& =\sum_{i: \tilde{\alpha}_{i}>\alpha_{i}} b_{i}^{-}(\boldsymbol{\alpha})\left(\tilde{\alpha}_{i}-\alpha_{i}\right)-\sum_{j: \tilde{\alpha}_{j}<\alpha_{j}} b_{i}^{+}(\boldsymbol{\alpha})\left(\alpha_{j}-\tilde{\alpha}_{j}\right) \\
& \leq \max _{l=1}^{n} b_{l}^{-}(\boldsymbol{\alpha}) \sum_{i: \tilde{\alpha}_{i}>\alpha_{i}}\left(\tilde{\alpha}_{i}-\alpha_{i}\right)-\min _{l=1}^{n} b_{l}^{+}(\boldsymbol{\alpha}) \sum_{j: \tilde{\alpha}_{j}<\alpha_{j}}\left(\alpha_{j}-\tilde{\alpha}_{j}\right) .
\end{aligned}
$$

By definition of $\psi$ we have $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ and $\mathbf{1}^{\top} \tilde{\boldsymbol{\alpha}}=0$ such that it follows that

$$
\sum_{j: \tilde{\alpha}_{j}<\alpha_{j}}\left(\alpha_{j}-\tilde{\alpha}_{j}\right)=-\sum_{j: \tilde{\alpha}_{j}>\alpha_{j}}\left(\alpha_{j}-\tilde{\alpha}_{j}\right)=\sum_{j: \tilde{\alpha}_{j}>\alpha_{j}}\left(\tilde{\alpha}_{j}-\alpha_{j}\right),
$$

i.e., the previous equation implies

$$
\begin{aligned}
-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) & \leq\left(\max _{l=1}^{n} b_{l}^{-}(\boldsymbol{\alpha})-\min _{l=1}^{n} b_{l}^{+}(\boldsymbol{\alpha})\right) \sum_{i: \tilde{\alpha}_{i}>\alpha_{i}}\left(\tilde{\alpha}_{i}-\alpha_{i}\right) \\
& =r(\boldsymbol{\alpha}) \sum_{i: \tilde{\alpha}_{i}>\alpha_{i}}\left(\tilde{\alpha}_{i}-\alpha_{i}\right) \\
& \leq r(\boldsymbol{\alpha})\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\|_{1} \\
& \leq r(\boldsymbol{\alpha}) \sqrt{n}\|\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}\| .
\end{aligned}
$$

Finally, taking the supremum over $\tilde{\boldsymbol{\alpha}} \in \operatorname{dom}(\psi) \cap \mathcal{B}(\boldsymbol{\alpha}, \rho)$, this shows that (7.5.1) holds.

Note that it is not possible to estimate the value of $R(\boldsymbol{\alpha}, \rho)$ from below by means of $r(\boldsymbol{\alpha})$ in general. This is due to the fact that in the definition of $r$ a unit step size is implicitly used which may be much larger than the maximal feasible step which is enforced in the definition of $R$. However, one can obtain an estimate by scaling the value of $r$ appropriately as proposed in the following statement.

Lemma 7.5.5 (connection between violation and optimality measure - second part) Let Assumption 7.5.1 hold. Let $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ with $r(\boldsymbol{\alpha})>0$ be given and let $i, j \in$ $\{1, \ldots, n\}$ satisfy

$$
\begin{equation*}
b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha}) \geq \omega r(\boldsymbol{\alpha}) \tag{7.5.2}
\end{equation*}
$$

Then,

$$
\begin{equation*}
\omega \min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\} r(\boldsymbol{\alpha}) \leq R(\boldsymbol{\alpha}, \sqrt{2} \rho) \tag{7.5.3}
\end{equation*}
$$

Proof. Given $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ and $\epsilon^{i j}(\boldsymbol{\alpha})$ defined in (7.3.5), we find that

$$
\tilde{\boldsymbol{\alpha}}(t):=\boldsymbol{\alpha}+t\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right) \in \operatorname{dom}(\psi)
$$

for all $t<\epsilon^{i j}(\boldsymbol{\alpha})$. Moreover, $\|\boldsymbol{\alpha}-\tilde{\boldsymbol{\alpha}}(t)\|=t\left\|\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right\|=\sqrt{2} t$. Using the definition of $R$, Remark 7.5.3 and condition (7.5.2) it follows that

$$
\begin{equation*}
R(\boldsymbol{\alpha}, \sqrt{2} \rho) \geq-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}(t)-\boldsymbol{\alpha})=t\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right) \geq t \omega r(\boldsymbol{\alpha}) \tag{7.5.4}
\end{equation*}
$$

for all $t<\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}$. Finally, taking the supremum over all $t<\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}$ we obtain (7.5.3).

Of course, the inequality (7.5.3) is not always sufficient to obtain a strong relation between $r$ and $R$ because the factor on the left-hand side can be arbitrarily small in general. On the other hand, we did not prove that this inequality cannot be improved. However, the steps in the proof suggest that there are actually cases in which the weak bound is attained (at least up to some constant factor). This means that the selection of a maximal violating pair (according to Definition 7.3.1) or some approximate violating pair (satisfying (7.5.2)) is not sufficient to guarantee convergence by means of Theorem 7.4.5 in general.

There are at least two possibilities to close this theoretical issue. One could try to employ another approach to prove convergence, which is certainly feasible and was done for instance in [CFLO6]. Closely connected to this approach is the idea to incorporate a certain procedure into the decomposition method preventing degeneration over the steps, see for instance [PS05]. A second possible remedy to close the theoretical gap is to restrict the working set selection by means of a certain sufficient condition as we will see below.

### 7.6 Definition and Properties of a Bound-Aware Violation Measure

The assertion of the previous Lemma 7.5.5 together with Proposition 7.3.2 motivates the following definition of a bound-aware violation measure, which is the main subject of the present section.

Definition 7.6.1 (bound-aware violation measure)
The value

$$
r^{\diamond}(\boldsymbol{\alpha}, \rho):=\max _{i, j}\left\{\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\right\}
$$

for $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ and $\rho>0$ (where a term of the form $0 \cdot(-\infty)$ is interpreted as 0 ) is called bound-aware violation measure.

To start with an investigation of the introduced measure, we are interested in the computational effort which is needed to compute the bound-aware violation measure. We observed in Observation 7.2.3 that the computation of the violation measure $r$ is possible in $\mathcal{O}(n)$ runtime. At first sight, one would expect the computation of $r^{\diamond}$ to take $\mathcal{O}\left(n^{2}\right)$ operations. However, using a certain transformation of the term, one can get a lower computational effort as follows.

Remark 7.6.2 (computational effort for the bound-aware violation measure) Denote by $\delta_{i}^{+}:=\min \left\{\bar{\alpha}_{i}-\alpha_{i}, \rho\right\}$ and $\delta_{j}^{-}:=\min \left\{\alpha_{j}-\underline{\alpha}_{j}, \rho\right\}$ the maximal step size for $\alpha_{i}$
(in positive direction) and $\alpha_{j}$ (in negative direction). Then, the bound-aware violation can be computed by means of

$$
\begin{aligned}
& r^{\diamond}(\boldsymbol{\alpha}, \rho)= \max _{i, j}\left\{\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\right\} \\
&= \max \left\{\max _{j} \max _{i: \delta_{i}^{+} \geq \delta_{j}^{-}}\left\{\delta_{j}^{-}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\right\},\right. \\
&\left.\max _{i} \max _{j: \delta_{i}^{+} \leq \delta_{j}^{-}}\left\{\delta_{i}^{+}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\right\}\right\} \\
&= \max \left\{\max _{j}\left\{\delta_{j}^{-}\left(\max _{i: \delta_{i}^{+} \geq \delta_{j}^{-}} b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\right\}\right. \\
&\left.\max _{i}\left\{\delta_{i}^{+}\left(b_{i}^{-}(\boldsymbol{\alpha})-\min _{j: \delta_{i}^{+} \leq \delta_{j}^{-}} b_{j}^{+}(\boldsymbol{\alpha})\right)\right\}\right\} .
\end{aligned}
$$

In particular, using sorted lists for the evaluation of the inner terms, it is possible to compute the value in $\mathcal{O}(n \log n)$ steps. Moreover, a particular violating pair can be determined during this computation.

Similar to what we have seen in Example 7.2.5, the bound-aware violation measure can also be discontinuous which we discuss in the following example. However, the key point in the construction of this example is the fact that the conjugate loss function is not smooth enough.

Example 7.6 .3 (piecewise constant bound-aware violation measure)
Consider a dual training problem (7.1.1) with $n=2$ samples and conjugate loss functions

$$
\ell_{1}^{\star}\left(\alpha_{1}\right):=\left|\alpha_{1}\right| \quad \text { and } \quad \ell_{2}^{\star}\left(\alpha_{2}\right):=0 .
$$

Let $\boldsymbol{K}:=\boldsymbol{O} \in \mathbb{R}^{2 \times 2}$. Then, $\operatorname{dom}\left(\ell_{i}^{\star}\right)=\mathbb{R}$, which implies that $\underline{\alpha}_{i}=-\infty$ and $\bar{\alpha}_{i}=\infty$ for $i \in\{1,2\}$. This means that $\epsilon^{i j}(\boldsymbol{\alpha})=\infty$ for $i, j \in\{1,2\}$ and hence $\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}=\rho$ for all $\boldsymbol{\alpha} \in \mathbb{R}^{2}$. Moreover, we can compute

$$
\partial \ell_{1}^{\star}\left(\alpha_{1}\right)= \begin{cases}\{-1\}, & \text { if } \alpha_{1}<0 \\ {[-1,1],} & \text { if } \alpha_{1}=0 \\ \{1\}, & \text { if } \alpha_{1}>0\end{cases}
$$

which yields

$$
b_{1}^{-}(\boldsymbol{\alpha})=\left\{\begin{array}{ll}
1, & \text { if } \alpha_{1}<0, \\
-1, & \text { if } \alpha_{1} \geq 0
\end{array} \quad \text { and } \quad b_{1}^{+}(\boldsymbol{\alpha})= \begin{cases}1, & \text { if } \alpha_{1} \leq 0 \\
-1, & \text { if } \alpha_{1}>0\end{cases}\right.
$$

On the other hand, $\partial \ell_{2}^{\star}\left(\alpha_{2}\right)=\{0\}$ and $b_{2}^{-}(\boldsymbol{\alpha})=b_{2}^{+}(\boldsymbol{\alpha})=0$ for all $\boldsymbol{\alpha} \in \mathbb{R}^{2}$. Finally, it
follows immediately from the definition that

$$
r^{\diamond}(\boldsymbol{\alpha}, \rho)= \begin{cases}0, & \text { if } \boldsymbol{\alpha}=\mathbf{0} \\ \rho, & \text { otherwise }\end{cases}
$$

Even though there is a possibility for $r^{\circ}$ to be discontinuous, one can expect it to be continuous if the conjugate loss functions defining the problem are continuously differentiable on an open set containing their domain. This observation is summarized in the following proposition.

Proposition 7.6 .4 (continuous bound-aware violation measure)
Let Assumption 7.5.1 hold and $\rho>0$. Then, $\boldsymbol{\alpha} \mapsto r^{\diamond}(\boldsymbol{\alpha}, \rho)$ is continuous.

Proof. First, consider an arbitrary point $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ and a pair $(i, j) \in\{1, \ldots, n\}^{2}$. If $\alpha_{i}=\bar{\alpha}_{i}$ or $\alpha_{j}=\underline{\alpha}_{j}$, it follows that $\epsilon^{i j}(\boldsymbol{\alpha})=0$ from the definition of $\epsilon^{i j}$ in (7.3.5). In this case,

$$
\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}\left(\boldsymbol{\alpha}^{k}\right)\right)=0
$$

(using the convention that $0 \cdot(-\infty)=0$ according to Definition 7.6.1). Otherwise, following Remark 7.5.3 the conditions $\alpha_{i}<\bar{\alpha}_{i}$ and $\alpha_{j}>\underline{\alpha}_{j}$ imply that

$$
\begin{equation*}
\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}\left(\boldsymbol{\alpha}^{k}\right)\right)=-\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\} \nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top}\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right) \tag{7.6.1}
\end{equation*}
$$

Note that this equation is true in both cases, which is the fundamental idea of the remaining part of the proof.

Let a sequence $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in \mathbb{N}} \subseteq \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ with $\boldsymbol{\alpha}^{\star}=\lim _{k \rightarrow \infty} \boldsymbol{\alpha}^{k}$ be given. By Assumption 7.5.1, $\lim _{k \rightarrow \infty} \nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)=\nabla \psi_{\mathrm{d} \text {,ext }}\left(\boldsymbol{\alpha}^{\star}\right)$. Let $\left(i_{k}, j_{k}\right)$ denote a pair of indices such that

$$
\begin{aligned}
r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \rho\right) & =\min \left\{\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right), \rho\right\}\left(b_{i_{k}}^{-}\left(\boldsymbol{\alpha}^{k}\right)-b_{j_{k}}^{+}\left(\boldsymbol{\alpha}^{k}\right)\right) \\
& =-\min \left\{\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right), \rho\right\} \nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\boldsymbol{e}^{i_{k}}-\boldsymbol{e}^{j_{k}}\right)
\end{aligned}
$$

for all $k \in \mathbb{N}$, where the second equation follows by means of (7.6.1). The sequence can be decomposed into subsequences with index sets

$$
K_{i j}:=\left\{k \in \mathbb{N} \mid i_{k}=i \text { and } j_{k}=j\right\}
$$

for $(i, j) \in\{1, \ldots, n\}^{2}$. In order to determine the value of $\lim _{k \rightarrow \infty} r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \rho\right)$ it is sufficient to consider only those pairs $(i, j)$ for which the set $K_{i j}$ contains infinitely many items. Hence, it follows that

$$
\begin{aligned}
\limsup _{k \rightarrow \infty} r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \rho\right) & =\max _{(i, j):\left|K_{i j}\right|=\infty} \limsup _{k \in K_{i j}}\left\{\min \left\{\epsilon^{i j_{k}}\left(\boldsymbol{\alpha}^{k}\right), \rho\right\}\left(b_{i_{k}}^{-}\left(\boldsymbol{\alpha}^{k}\right)-b_{j_{k}}^{+}\left(\boldsymbol{\alpha}^{k}\right)\right)\right\} \\
& =\max _{(i, j):: K_{i j} \mid=\infty} \lim _{k \in K_{i j}}\left\{-\min \left\{\epsilon^{i j}\left(\boldsymbol{\alpha}^{k}\right), \rho\right\} \nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right)\right\} \\
& =\max _{(i, j):\left|K_{i j}\right|=\infty}\left\{-\min \left\{\epsilon^{i j}\left(\boldsymbol{\alpha}^{\star}\right), \rho\right\} \nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top}\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right)\right\} \\
& =\max _{(i, j):\left|K_{i j}\right|=\infty}\left\{\min \left\{\epsilon^{i j}\left(\boldsymbol{\alpha}^{\star}\right), \rho\right\}\left(b_{i}^{-}\left(\boldsymbol{\alpha}^{\star}\right)-b_{j}^{+}\left(\boldsymbol{\alpha}^{\star}\right)\right)\right\} \\
& \leq r^{\diamond\left(\boldsymbol{\alpha}^{\star}, \rho\right) .}
\end{aligned}
$$

On the other hand,

$$
\begin{aligned}
r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \rho\right) & =-\min \left\{\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right), \rho\right\} \nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\boldsymbol{e}^{i_{k}}-\boldsymbol{e}^{j_{k}}\right) \\
& \geq-\min \left\{\epsilon^{i j}\left(\boldsymbol{\alpha}^{k}\right), \rho\right\} \nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top}\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right)
\end{aligned}
$$

for any $(i, j) \in\{1, \ldots, n\}$. Considering the limit $k \rightarrow \infty$ on the right-hand side and maximizing it with respect to $(i, j)$, it also follows that

$$
\liminf _{k \rightarrow \infty} r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \rho\right) \geq r^{\diamond}\left(\boldsymbol{\alpha}^{\star}, \rho\right)
$$

Taking the previous considerations together, it follows that

$$
\lim _{k \rightarrow \infty} r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \rho\right)=r^{\diamond}\left(\boldsymbol{\alpha}^{\star}, \rho\right)
$$

i.e., the function $\boldsymbol{\alpha} \mapsto r^{\diamond}(\boldsymbol{\alpha}, \rho)$ is indeed continuous.

Subsequently, we also summarize fundamental properties of the bound-aware vioIation measure, which connect its value to the value of the first-order optimality measure introduced in Theorem 7.4.2. The following two lemmas are the natural translations of Lemma 7.5.4 and Lemma 7.5.5 to the present setting.

Lemma 7.6 .5 (bound-aware violation and optimality measure - first part) Let Assumption 7.5.1 hold. Then,

$$
\begin{equation*}
R(\boldsymbol{\alpha}, \rho) \leq n r^{\diamond}(\boldsymbol{\alpha}, \rho) \tag{7.6.2}
\end{equation*}
$$

for all $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ with $r^{\diamond}(\boldsymbol{\alpha}, \rho)>0$.

Proof. Subsequently, we follow the lines of the proof of Lemma 7.5.4. First, we use Remark 7.5.3 and $\mathbf{1}^{\top} \boldsymbol{\alpha}=\mathbf{1}^{\top} \tilde{\boldsymbol{\alpha}}=0$ to get

$$
\begin{aligned}
-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha})= & \sum_{i: \alpha_{i}<\bar{\alpha}_{i}} b_{i}^{-}(\boldsymbol{\alpha})\left(\tilde{\alpha}_{i}-\alpha_{i}\right)+\sum_{i: \alpha_{i}=\bar{\alpha}_{i}} b_{i}^{+}(\boldsymbol{\alpha})\left(\tilde{\alpha}_{i}-\alpha_{i}\right) \\
= & \sum_{i: \alpha_{i}<\bar{\alpha}_{i}}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\left(\tilde{\alpha}_{i}-\alpha_{i}\right) \\
& +\sum_{i: \alpha_{i}=\bar{\alpha}_{i}}\left(b_{i}^{+}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\left(\tilde{\alpha}_{i}-\alpha_{i}\right)
\end{aligned}
$$

for any $j \in\{1, \ldots, n\}$. If $j \in\{1, \ldots, n\}$ is chosen such that $b_{j}^{+}(\boldsymbol{\alpha})=\min _{l=1}^{n} b_{l}^{+}(\boldsymbol{\alpha})$, the second sum on the right-hand side is non-positive. Thus, we obtain

$$
\begin{align*}
-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) & \leq \sum_{i: \alpha_{i}<\bar{\alpha}_{i}}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\left(\tilde{\alpha}_{i}-\alpha_{i}\right) \\
& \leq \max _{j} \sum_{i: \alpha_{i}<\bar{\alpha}_{i}}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\left(\tilde{\alpha}_{i}-\alpha_{i}\right) \\
& \leq n \max _{j} \max _{i: \alpha_{i}<\bar{\alpha}_{i}}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)\left(\tilde{\alpha}_{i}-\alpha_{i}\right)  \tag{7.6.3}\\
& \leq n \max _{j} \max _{i: \alpha_{i}<\bar{\alpha}_{i}}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right) \min \left\{\bar{\alpha}_{i}-\alpha_{i}, \rho\right\} \\
& \leq n \max _{i, j}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right) \min \left\{\bar{\alpha}_{i}-\alpha_{i}, \rho\right\} .
\end{align*}
$$

In the same way (and by changing the roles of $i$ and $j$ ) we can show that

$$
-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) \leq \sum_{j: \alpha_{j}>\underline{\alpha}_{j}}\left(b_{j}^{+}(\boldsymbol{\alpha})-b_{i}^{-}(\boldsymbol{\alpha})\right)\left(\tilde{\alpha}_{j}-\alpha_{j}\right)
$$

for $i \in\{1, \ldots, n\}$ with $b_{i}^{-}(\boldsymbol{\alpha})=\max _{l=1}^{n} b_{l}^{-}(\boldsymbol{\alpha})$. Hence, it also follows that

$$
\begin{equation*}
-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top}(\tilde{\boldsymbol{\alpha}}-\boldsymbol{\alpha}) \leq n \max _{i, j}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right) \min \left\{\alpha_{j}-\underline{\alpha}_{j}, \rho\right\} \tag{7.6.4}
\end{equation*}
$$

Finally, taking together (7.6.3) and (7.6.4) proves (7.6.2).

Lemma 7.6.6 (bound-aware violation and optimality measure - second part) Let Assumption 7.5.1 hold. Then,

$$
\begin{equation*}
r^{\diamond}(\boldsymbol{\alpha}, \rho) \leq R(\boldsymbol{\alpha}, \sqrt{2} \rho) \tag{7.6.5}
\end{equation*}
$$

for all $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ and $\rho>0$.
Proof. Subsequently, we follow the lines of the proof of Lemma 7.5.5. Let $(i, j) \in$ $\{1, \ldots, n\}^{2}$ be chosen such that

$$
\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)=r^{\diamond}(\boldsymbol{\alpha}, \rho) .
$$

By definition, $\tilde{\boldsymbol{\alpha}}:=\boldsymbol{\alpha}+t\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right) \in \operatorname{dom}(\psi) \cap \mathcal{B}(\boldsymbol{\alpha}, \sqrt{2} \rho)$ for all $t \in(0, \rho)$. Using the same computation as in inequality (7.5.4) we get

$$
R(\boldsymbol{\alpha}, \sqrt{2} \rho) \geq t\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right) \geq \min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho\right\}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right)=r^{\diamond}(\boldsymbol{\alpha}, \rho)
$$

This proves (7.6.5).
The combination of the previous two lemmas shows that the value of the boundaware violation measure $r^{\diamond}$ is actually comparable to the value of the first-order optimality measure $R$. This makes it easily possible to apply the convergence result of Theorem 7.4.5 together with Lemma 7.4.6 or Lemma 7.4.7 to some particular realization of a decomposition method as we will see below.

### 7.7 Working Set Selection Rules

In the history of the development of the SMO method several possibilities for choosing the working set were proposed. The aim of each of the approaches is to improve the convergence speed of the overall method while keeping the computational effort for the selection relatively low.

The first approaches are based solely on the idea of violating pairs, i.e., pairs of variables violating the optimality conditions of the dual training problem. If the corresponding variables are updated appropriately, a certain amount of improvement in the objective function is realized, cf. [OFG97, Pla99].

Note that [Pla99] claims that this is already sufficient to guarantee convergence, which is not true in general because the improvement of the objective function value could be vanishing over time if unsuitable pairs are chosen.

A particularly popular choice is the selection of a maximal violating pair, cf. Definition 7.3.1. In general, determining a violating pair takes $\mathcal{O}(n)$ operations.

In the more recent development of SMO methods, the idea of second-order approximations for the working set selection has been established. This approach aims to approximate the expected improvement of the objective function value based on its curvature information which is easily accessible for quadratic functions, see [FCL05, CL11].

A pair of variables is then chosen by first fixing one variable according to the firstorder rule (7.3.1) and afterwards a second one that is feasible (i.e., can be altered in the given direction) and maximizes the predicted descent. The latter is computed using the unconstrained minimum of the remaining scalar quadratic function, for which a closed-form solution exists. This selection rule can be formally described as follows.

Definition 7.7.1 (maximal unbounded decrease pair)
Let $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ be given and let $\left(i_{1}, j_{2}\right) \in\{1, \ldots, n\}^{2}$ be a maximal violating pair. Determine

$$
j_{1} \in \underset{j \in\{1, \ldots, n\}}{\operatorname{argmin}} \min _{t \in \mathbb{R}} \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}+t\left(\boldsymbol{e}^{i_{1}}-\boldsymbol{e}^{j}\right)\right)
$$

and

$$
i_{2} \in \underset{i \in\{1, \ldots, n\}}{\operatorname{argmin}} \min _{t \in \mathbb{R}} \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}+t\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j_{2}}\right)\right) .
$$

A pair $\left(i_{1}, j_{1}\right)$ or $\left(i_{2}, j_{2}\right)$ yielding the smaller objective function value above is called maximal unbounded decrease pair.

This definition is an extension of the working set selection using second order information proposed in [FCL05] because the quadratic approximation (using Taylor's theorem) of $\psi_{\text {d,ext }}$ is exact for quadratic objective functions.

The overall process of checking all possible candidates (for fixed first variable) can be conducted in $n$ steps. This means that the determination of a working set based on second-order information as described above is not significantly more expensive than computing a maximal violating pair. Because such a selection improves the convergence speed of the SMO method generally, it is preferred in practice.

Finally, note that it is also possible to compute the actual decrease of the objective function value when selecting a second variable in a second-order selection procedure. Since this does not result in too much additional computational effort (at least for quadratic objective functions), we would argue that this strategy is preferable in general. Hence, we summarize this idea in the following.

Definition 7.7.2 (maximal decrease pair)
Let $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ with $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ be given and let $\left(i_{1}, j_{2}\right) \in\{1, \ldots, n\}^{2}$ be a maximal violating pair. Determine

$$
j_{1} \in \underset{j \in\{1, \ldots, n\}}{\operatorname{argmin}} \min _{t \in \mathbb{R}} \psi_{\mathrm{d}}\left(\boldsymbol{\alpha}+t\left(\boldsymbol{e}^{i_{1}}-\boldsymbol{e}^{j}\right)\right)
$$

and

$$
i_{2} \in \underset{i \in\{1, \ldots, n\}}{\operatorname{argmin}} \min _{\in \in \mathbb{R}} \psi_{\mathrm{d}}\left(\boldsymbol{\alpha}+t\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j_{2}}\right)\right) .
$$

A pair $\left(i_{1}, j_{1}\right)$ or $\left(i_{2}, j_{2}\right)$ yielding the smaller objective function value above is called maximal decrease pair.

It should be pointed out that a maximum decrease pair as proposed above maximizes the decrease only partially because it fixes one index by means of the maximal violating pair strategy first. In order to obtain a pair which actually maximizes the decrease, an exhaustive search over all pairs is necessary in general. For completeness, we introduce this working set selection strategy subsequently.

Definition 7.7.3 (total maximal decrease pair)
Let $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ with $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$ be given. We call a pair of indices

$$
(i, j) \in \underset{(i, j) \in\{1, \ldots, n\}^{2}}{\operatorname{argmin}} \min _{t \in \mathbb{R}} \psi_{\mathrm{d}}\left(\boldsymbol{\alpha}+t\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right)\right)
$$

a total maximal decrease pair.

### 7.8 Convergence of SMO

In the following, we briefly discuss the proposed strategies from a theoretical point of view with the focus on convergence guarantees. As a general restriction on the working set selection rule, the following definition is helpful. It will be the basis of a convergence of the concrete realization of the SMO method.

Definition 7.8.1 ( $\omega$-violating pair)
Let $\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right)$ with $\mathbf{1}^{\top} \boldsymbol{\alpha}=0$. A pair of indices $(i, j) \in\{1, \ldots, n\}^{2}$ satisfying

$$
\begin{equation*}
\min \left\{\epsilon^{i j}(\boldsymbol{\alpha}), \rho_{1}\right\}\left(b_{i}^{-}(\boldsymbol{\alpha})-b_{j}^{+}(\boldsymbol{\alpha})\right) \geq \omega r^{\diamond}\left(\boldsymbol{\alpha}, \rho_{1}\right) \tag{7.8.1}
\end{equation*}
$$

is called (bound-aware) $\omega$-violating at $\boldsymbol{\alpha} \in \mathbb{R}^{n}$ for $\rho_{1}>0$.

Using the notion of $\omega$-violating pairs, it is possible to define a rather general algorithm, for which convergence can be proven as follows.

```
Algorithm 7.8.1: Sequential Minimal Optimization method
    Input:
    starting point \(\boldsymbol{\alpha}^{0} \in\left\{\boldsymbol{\alpha} \in \operatorname{dom}\left(\psi_{\mathrm{d}}\right) \mid \mathbf{1}^{\top} \boldsymbol{\alpha}=0\right\}\),
    parameters \(\omega>0, \rho_{0}>0\) and \(\rho_{1} \in\left(0, \rho_{0}\right.\) ]
    for \(k=0,1,2, \ldots\) do
        if \(r\left(\boldsymbol{\alpha}^{k}\right) \leq 0\) then
            Stop: \(\boldsymbol{\alpha}^{k}\) is optimal due to Observation 7.2.2.
        Determine an \(\omega\)-violating pair \(\left(i_{k}, j_{k}\right) \in\{1, \ldots, n\}^{2}\) for \(\rho_{1}\).
        Define the direction \(\boldsymbol{d}^{k}:=\min \left\{\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right), \rho_{0}\right\}\left(\boldsymbol{e}^{i_{k}}-\boldsymbol{e}^{j_{k}}\right)\).
        Compute a step size \(\tau_{k}\) such that (7.4.8) or (7.4.13) holds.
        Update \(\boldsymbol{\alpha}^{k+1}:=\boldsymbol{\alpha}^{k}+\tau_{k} \boldsymbol{d}^{k}\).
```

Now, a convergence result for this method can be derived from the general convergence framework given by Theorem 7.4.5 as follows.

Corollary 7.8.2 (convergence of SMO method with $\omega$-violating pairs)
Let Assumption 7.5.1 hold and suppose that $\nabla \psi_{\text {d,ext }}$ is Lipschitz continuous on the level set $L_{\psi_{d, \mathrm{ext}}}\left(\psi\left(\boldsymbol{\alpha}^{0}\right)\right)$. Let $\rho_{0}>0$. Then, for any direction of the form

$$
\boldsymbol{d}^{k}=\min \left\{\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right), \rho_{0}\right\}\left(\boldsymbol{e}^{i_{k}}-\boldsymbol{e}^{j_{k}}\right)
$$

with $\left(i_{k}, j_{k}\right) \in\{1, \ldots, n\}^{2}$ being $\omega$-violating pairs at $\boldsymbol{\alpha}^{k}$ for some $\rho_{1} \in\left(0, \rho_{0}\right]$ the conditions (7.4.6) and (7.4.7) are satisfied. In particular, any accumulation point of the resulting iterative scheme

$$
\boldsymbol{\alpha}^{k+1}:=\boldsymbol{\alpha}^{k}+\tau_{k} \boldsymbol{d}^{k}
$$

with exact line search (7.4.8) or Armijo-type line search (7.4.13) for $\tau_{k}$ is a solution of (7.1.1).

Proof. It is easy to see that $\mathbf{1}^{\top}\left(\boldsymbol{\alpha}^{k}+\boldsymbol{d}^{k}\right)=0$, i.e., condition (7.4.6) follows directly. Moreover, because $\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right)>0$, the definition of $\boldsymbol{d}^{k}$ and Remark 7.5.3 implies

$$
r_{k}=-\nabla \psi_{\mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top} \boldsymbol{d}^{k}=\min \left\{\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right), \rho_{0}\right\}\left(b_{i_{k}}^{-}\left(\boldsymbol{\alpha}^{k}\right)-b_{j_{k}}^{+}\left(\boldsymbol{\alpha}_{k}\right)\right) .
$$

Then, due to the definition of $\omega$-violation in (7.8.1) with $\rho_{1} \leq \rho_{0}$ and Lemma 7.6.5, it follows that

$$
r_{k} \geq \omega r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \rho_{1}\right) \geq \frac{\omega}{n} R\left(\boldsymbol{\alpha}^{k}, \rho_{1}\right) .
$$

In a final step, we apply the result of Lemma 7.4.6 or Lemma 7.4.7, respectively, and use that $\left\|\boldsymbol{d}^{k}\right\| \leq \sqrt{2} \rho_{0}$. In the case of an exact line search, condition (7.4.9) implies that (7.4.5) holds for $\rho:=\rho_{1}$ with

$$
c(r):=\min \left\{\frac{\omega}{2 n} r, \frac{\omega^{2}}{4 L \rho_{0}^{2} n^{2}} r^{2}\right\}
$$

which is indeed a forcing function in the sense of Definition 7.4.4. Similarly, for the Armijo-type line search rule (7.4.13), condition (7.4.5) with $\rho:=\rho_{1}$ is satisfied with the
forcing function

$$
c(r):=\min \left\{\frac{\sigma \omega}{n} r, \frac{\eta \sigma(1-\sigma) \omega^{2}}{L \rho_{0}^{2} n^{2}} r^{2}\right\}
$$

because of (7.4.14). This means that Theorem 7.4.5 can be applied to conclude convergence (in the sense that any accumulation point of the generated sequence is a solution of the dual training problem).

Observe that there are two different step size parameters $\rho_{0}$ and $\rho_{1}$ to choose in the implementation of Algorithm 7.8.1. The first one determines the maximum length of the direction vector $\boldsymbol{d}^{k}$ and is needed generally for theoretical purposes only, cf. Corollary 7.8.2. If all variables have bound-constraints, the value of $\rho_{0}$ can be chosen large enough to ensure that the minimum in the definition of $\boldsymbol{d}^{k}$ is not capped artificially. In particular this makes sense if the step size is selected by means of an exact line search because otherwise the steps would be unnecessarily small. On the other hand, the parameter $\rho_{1}$ controls the working set selection and should be chosen in a way to reflect the expected step size such that the first-order descent prediction is reasonable.

### 7.9 Implementation and Complexity Analysis

In this section, we extend on the practical realization of the general SMO method presented in Algorithm 7.8.1. First, an investigation of the per-step complexity of the algorithm is conducted. In a second subsection, an argumentation concerning the convergence speed is proposed.

### 7.9.1 Discussion of Complexity

Subsequently, we consider the overall complexity of a single step of Algorithm 7.8.1 and approaches to keep the computational effort as low as possible. In accordance with the common argumentation for the application of decomposition methods, we assume that it is not feasible to compute and save the kernel matrix efficiently. This is the usual assumption for large datasets and leads to the situation that the entries of the matrix $\boldsymbol{K}$ must be computed on demand. We will also briefly discuss the possibility of caching which enables some sort of reuse of the computed data below.

To start with an investigation of the overall effort, observe that the SMO method consists of three major parts:

- the working set selection (or determination of search direction) in lines 2 to 5,
- the step size determination in line 6, and
- the update of variables in line 7.

During the selection of the working set, it is theoretically important to determine an $\omega$-violating pair according to Definition 7.8.1. At least this is necessary in an infinite number of steps to obtain convergence from Corollary 7.8.2. Given that the value of
the kernel expansion $\left[\boldsymbol{K} \boldsymbol{\alpha}^{k}\right]_{i}$ is available for all $i \in\{1, \ldots, n\}$, it is possible to compute such a pair in $\mathcal{O}(n \log n)$ steps as we have seen in Remark 7.6.2. Practically, however, it is also sufficient to select working sets related to the maximal violating pair. In particular, the determination of a maximal violating pair (or some appropriate approximation) is sufficient if the sequence of associated maximal feasible step sizes $\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right)$ remains bounded from below. Only if the value of $\epsilon^{i_{k} j_{k}}\left(\boldsymbol{\alpha}^{k}\right)$ gets too small, it would be necessary to determine a (bound-aware) $\omega$-violating pair in order to guarantee convergence by means of Corollary 7.8.2. This reduces the effort to $\mathcal{O}(n)$ steps in practice.

In the same way, the determination of a maximal decrease pair (as proposed in Definition 7.7.2), which helps to increase the convergence speed in practice, can also be done quite efficiently. In this case it is also necessary to have second-order information at hand, which means that some columns of the kernel matrix must be computed. However, since some of these values are needed for the step size computation and the update later on anyway, the additional effort is not significant. Of course, it is theoretically necessary to ensure that the selected pair is $\omega$-violating, which would lead to $\mathcal{O}(n \log n)$ operations. However, this issue is not relevant practically because maximal decrease pairs usually yield a sufficient descent.

Although the update step in line 7 of the algorithm is trivial because only two variables are touched, we want to point out that the effort is larger in the overall context as it is also helpful to consider the update of the kernel expansion here. Note that this update can be conducted by means of the relation $\boldsymbol{K} \boldsymbol{\alpha}^{k+1}:=\boldsymbol{K} \boldsymbol{\alpha}^{k}+\tau_{k} \boldsymbol{K}\left(\boldsymbol{e}^{i_{k}}-\boldsymbol{e}^{j_{k}}\right)$. This means that the update involves the computation of two columns of the kernel matrix $\boldsymbol{K}$ and their weighted sum, which takes again $\mathcal{O}(n)$ operations. This reduces the computational effort of the subsequent step drastically because it is than not necessary to compute the value from scratch (which would take $\mathcal{O}\left(n^{2}\right)$ operations in the worst case to compute all values of the matrix $\boldsymbol{K}$ ).

Finally, consider the effort of the step size selection rule. We focus on the exact step size (7.4.8) here. For this problem, the objective function (depending on the step size $\tau$ only) can be computed as

$$
\begin{align*}
\psi_{\mathrm{d}}\left(\boldsymbol{\alpha}^{k}+\tau\left(\boldsymbol{e}^{i}-\boldsymbol{e}^{j}\right)\right)= & \psi_{\mathrm{d}}\left(\boldsymbol{\alpha}^{k}\right)-\ell_{i}^{\star}\left(-\alpha_{i}^{k}\right)-\ell_{j}^{\star}\left(-\alpha_{j}^{k}\right)  \tag{7.9.1a}\\
& +\frac{\tau^{2}}{2 \lambda}\left(K_{i i}+K_{j j}-2 K_{i j}\right)+\frac{\tau}{\lambda}\left(\left[\boldsymbol{K} \boldsymbol{\alpha}^{k}\right]_{i}-\left[\boldsymbol{K} \boldsymbol{\alpha}^{k}\right]_{j}\right)  \tag{7.9.1b}\\
& +\ell_{i}^{\star}\left(-\alpha_{i}^{k}-\tau\right)+\ell_{j}^{\star}\left(-\alpha_{j}^{k}+\tau\right) \tag{7.9.1c}
\end{align*}
$$

The structure of the objective function is the following:

- The term on the right-hand side of (7.9.1a) does not depend on $\tau$ and hence it can be ignored when solving the subproblem (7.4.8).
- The term in (7.9.1 b) consists of a quadratic part and a linear part. By definition of the kernel matrix, $K_{i i}+K_{j j}-2 K_{i j} \geq 0$ and the factor is strictly positive if and only if $\boldsymbol{\varphi}_{i} \neq \boldsymbol{\varphi}_{j}$. The factor $\left[\boldsymbol{K} \boldsymbol{\alpha}^{k}\right]_{i}-\left[\boldsymbol{K} \boldsymbol{\alpha}^{k}\right]_{j}$, which defines the linear part in (7.9.1b) is easily computable if the vector $\boldsymbol{K} \boldsymbol{\alpha}^{k}$ is known. However, this is the case because this term is typically updated iteratively in the SMO method.
- Finally, the structure of the last term in objective function (7.9.1c) depends on the particular choice of the loss functions. For instance, for the classical Support Vector Classification training problem considered in Proposition 4.1.1 the conjugate loss function is affine linear on a closed interval and infinite otherwise. This means that the terms in (7.9.1c) introduce a linear term and simple bounds on the variable $\tau$.
Similarly, the training problem for $\varepsilon$-Support Vector Regression (see Proposition 4.2.2) can be rewritten in a way that the conjugate loss functions have the same structure (by doubling the number of variables in the dual problem).
Note that an approximation of the maximum function in these problems by the piecewise quadratic approximation proposed in Example 4.4.4 leads to an additional quadratic term in the objective function, which does not change the fundamental structure of the subproblem. However, many other loss functions lead to non-quadratic terms in (7.9.1c), which are not as easy to handle in the subproblem as the quadratic terms which we observed above.

These considerations show that the subproblem can be solved very easily for commonly considered training problems and certain extensions thereof. Basically, the solution of problem (7.4.8) can be obtained by computing the root of a quadratic function and projecting it onto a fixed interval. This operation can be implemented efficiently.

This leads to an overall effort of one step of the SMO method of $\mathcal{O}(n)$ operation (at least in most of the steps). Hence, the method is applicable to treat the dual training problem also for a large number of training samples.

As we mentioned above, the computational effort of one step could be lowered if a suitable strategy for caching of the computed parts of the kernel matrix is employed. The idea which makes it possible to use caching is the fact that only a rather small portion of variables is updated during successive steps. In particular, those variables which are at their lower or upper bound are not subject to change in the later phase of the optimization process. For instance, in the state-of-the-art implementation of the SMO method of LIBSVM a simple least-recently-used caching strategy is employed, cf. [CL11, Subsection 5.2]. However, a more extensive discussion of possible caching strategies is out of scope of the present work.

Besides the caching of kernel function values, there is another prominent technique that helps to improve the performance of the SMO method. An approach called shrinking aims to reduce the effort of $\mathcal{O}(n)$ operations per step to some lower number. The basic idea is that a large portion of the variables will be at their bound, a fact which can possibly be detected rather early in the optimization process. If a variable is expected to stay at its bound, it is not necessary to consider it in the working set selection. Moreover, the corresponding value for the kernel expansion is not needed in the subsequent steps of the SMO method in this case. This means that the number variables (and hence the computational effort of each step) can be reduced to a certain set of active variables. We come back to this idea below in Section 7.10.

### 7.9.2 Notes on the Convergence Speed

We want to briefly argue that the step size and descent can be estimated asymptotically by means of known measures. For this reason, recall that an exact line search procedure generates a step size in the order of

$$
\tau_{k} \sim \frac{r_{k}}{L\left\|\boldsymbol{d}^{k}\right\|^{2}}
$$

see (7.4.12). Here, the only factor which is independent of $k$ and hence could be used as a guide for the value of $\rho_{1}$ is $L^{-1}$, where $L$ is the Lipschitz constant of $\nabla \psi_{\text {d,ext }}$. The previous approximation can also be used to obtain that

$$
\begin{equation*}
\left\|\boldsymbol{\alpha}^{k+1}-\boldsymbol{\alpha}^{k}\right\|=\tau_{k}\left\|\boldsymbol{d}^{k}\right\| \sim \frac{r_{k}}{L\left\|\boldsymbol{d}^{k}\right\|}, \tag{7.9.2}
\end{equation*}
$$

which gives an approximation of the distance between subsequent points.
On the other hand, the value of the constant $L$ is also important in the estimation of the descent. Especially, because of (7.4.9) and since $r_{k}$ is asymptotically small, one can expect the descent of the objective function in each step to be in the order of

$$
\begin{equation*}
\psi\left(\boldsymbol{\alpha}^{k}\right)-\psi\left(\boldsymbol{\alpha}^{k+1}\right) \sim \frac{r_{k}^{2}}{2 L\left\|\boldsymbol{d}^{k}\right\|^{2}} . \tag{7.9.3}
\end{equation*}
$$

For particular problem classes, the Lipschitz constant can be estimated as the following example shows.

Example 7.9.1 (Lipschitz continuous gradient for classical dual SVM problem)
For the dual problem of the classical SVM problem presented in Proposition 4.1.1, it can be used that

$$
\psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})=\frac{1}{2 \lambda} \boldsymbol{\alpha}^{\top} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}^{\top} \boldsymbol{\alpha} .
$$

In particular, we can compute

$$
\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})=\lambda^{-1} \boldsymbol{K} \boldsymbol{\alpha}-\boldsymbol{y}
$$

which yields the estimate

$$
\left\|\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\tilde{\boldsymbol{\alpha}})\right\|=\lambda^{-1}\|\boldsymbol{K}(\boldsymbol{\alpha}-\tilde{\boldsymbol{\alpha}})\| \leq \lambda^{-1}\|\boldsymbol{K}\| \cdot\|\boldsymbol{\alpha}-\tilde{\boldsymbol{\alpha}}\|
$$

Thus, $\nabla \psi_{\mathrm{d}, \mathrm{ext}}$ is Lipschitz continuous with a constant $L:=\lambda^{-1}\|\boldsymbol{K}\|$.
Note that similar computations are possible for other training problems. In doing so, one can observe that the order of the Lipschitz constant in the estimated is governed by the value $\lambda^{-1}\|\boldsymbol{K}\|$ for any standard training problem. This observation is important practically because the value of $L$ determines the convergence speed of the SMO method crucially.

### 7.10 Detection of Fixed Variables for Shrinking

As we have seen in Subsection 7.9.1, it is practically very important to determine the set of fixed variables (i.e., variables at their bound for some solution) in order to apply a shrinking procedure. The determination of fixed variables is closely connected to the identification of active constraints. The latter problem is, for instance, considered in [FFK98], where the notion of an identification function is introduced and applied to KKT systems of nonlinear programs with inequality constraints. This approach was also used in [LL10] for special versions of SVMs.

In the following, we propose a similar approach which makes use of the optimality measure $R$ defined in (7.4.1) instead and exploits the particular structure of the dual training problem. To start with, let us consider the following estimates on the optimal value of the bias variable in the primal training problem.

Proposition 7.10 .1 (estimate on the optimal bias value)
Let Assumption 7.5.1 hold and let $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ be given. Suppose that the minimizing set $A$ of $\psi$ is not empty and that $\nabla \psi_{\mathrm{d}, \mathrm{ext}}$ is Lipschitz continuous on $L_{\psi}(\psi(\boldsymbol{\alpha}))$. Then, for any solution $\left(\boldsymbol{w}^{\star}, b^{\star}\right) \in \mathcal{F} \times \mathbb{R}$ of the primal training problem (2.4.3) it follows that

$$
\begin{equation*}
\sup _{i: \alpha_{i}<\bar{\alpha}_{i}-\operatorname{dist}[\boldsymbol{\alpha}, A]} b_{i}^{-}(\boldsymbol{\alpha})-L \operatorname{dist}[\boldsymbol{\alpha}, A] \leq b^{\star} \leq \inf _{i: \alpha_{i}>\underline{\alpha}_{i}+\operatorname{dist}[\boldsymbol{\alpha}, A]} b_{i}^{+}(\boldsymbol{\alpha})+L \operatorname{dist}[\boldsymbol{\alpha}, A] . \tag{7.10.1}
\end{equation*}
$$

Proof. Let $\boldsymbol{\alpha}^{\star} \in A$ be any point with $\left\|\boldsymbol{\alpha}^{\star}-\boldsymbol{\alpha}\right\|=\operatorname{dist}[\boldsymbol{\alpha}, A]$. Because $\nabla \psi_{\mathrm{d}, \mathrm{ext}}$ is assumed to be Lipschitz continuous, it follows that

$$
\begin{align*}
\left|\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}-\nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top} \boldsymbol{e}^{i}\right| & \leq\left\|\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})-\nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)\right\| \\
& \leq L\left\|\boldsymbol{\alpha}-\boldsymbol{\alpha}^{\star}\right\|  \tag{7.10.2}\\
& =L \operatorname{dist}[\boldsymbol{\alpha}, A]
\end{align*}
$$

for all $i \in\{1, \ldots, n\}$.
If no index $i \in\{1, \ldots, n\}$ with $\alpha_{i}>\underline{\alpha}_{i}+\operatorname{dist}[\boldsymbol{\alpha}, A]$ exists, the infimum in (7.10.1) evaluates to infinity, which means that the inequality holds trivially. Otherwise, let $i \in$ $\{1, \ldots, n\}$ with $\alpha_{i}>\underline{\alpha}_{i}+\operatorname{dist}[\boldsymbol{\alpha}, A]$ be chosen arbitrarily. Then, it follows directly from (7.3.2) that

$$
\begin{equation*}
b_{i}^{+}(\boldsymbol{\alpha})=\psi_{\mathrm{d}}^{\circ}\left(\boldsymbol{\alpha} ;-\boldsymbol{e}^{i}\right)=-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i} \tag{7.10.3}
\end{equation*}
$$

Additionally, we obtain that

$$
\left|\alpha_{i}-\alpha_{i}^{\star}\right| \leq\left\|\boldsymbol{\alpha}-\boldsymbol{\alpha}^{\star}\right\|=\operatorname{dist}[\boldsymbol{\alpha}, A],
$$

which implies $\alpha_{i}^{\star}+\operatorname{dist}[\boldsymbol{\alpha}, A] \geq \alpha_{i}$. This means that $\alpha_{i}^{\star}>\underline{\alpha}_{i}$ because of the assumption that $\alpha_{i}>\underline{\alpha}_{i}+\operatorname{dist}[\boldsymbol{\alpha}, A]$. However, if $\alpha_{i}^{\star}>\underline{\alpha}_{i}$, following (7.10.3) we also get $b_{i}^{+}\left(\boldsymbol{\alpha}^{\star}\right)=$ $-\nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top} \boldsymbol{e}^{i}$. Now, the optimality of $\boldsymbol{\alpha}^{\star}$ together with (7.10.2) implies
$b^{\star} \leq b_{i}^{+}\left(\boldsymbol{\alpha}^{\star}\right)=-\nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top} \boldsymbol{e}^{i} \leq-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}+L \operatorname{dist}[\boldsymbol{\alpha}, A]=b_{i}^{+}(\boldsymbol{\alpha})+L \operatorname{dist}[\boldsymbol{\alpha}, A]$, cf. condition (7.2.5). Thus, the upper bound in (7.10.1) is proven. In order to see that the lower bound is also valid, the same argumentation can be used.

Next, we show how a bound on the optimal bias value of the type (7.10.1) can be utilized to derive conditions which imply that certain variables attain their lower or upper bound at the optimum.

Proposition 7.10 .2 (detection of fixed variables)
Let Assumption 7.5.1 hold and let $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ be given. Suppose that the minimizing set $A$ of $\psi$ is not empty and that $\nabla \psi_{\text {d,ext }}$ is Lipschitz continuous on $L_{\psi}(\psi(\boldsymbol{\alpha}))$. If $b^{\star} \in$ $\left[b^{-}, b^{+}\right]$holds for any solution $\left(\boldsymbol{w}^{\star}, b^{\star}\right) \in \mathcal{F} \times \mathbb{R}$ of the primal training problem (2.4.3), there exists a solution $\boldsymbol{\alpha}^{\star} \in \mathbb{R}^{n}$ of the dual training problem (7.1.1) such that

$$
\begin{equation*}
\alpha_{i}^{\star}=\underline{\alpha}_{i} \quad \text { for all } \quad i \in\{1, \ldots, n\} \quad \text { with } \quad-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}<b^{-}-L \operatorname{dist}[\boldsymbol{\alpha}, A] \tag{7.10.4a}
\end{equation*}
$$

and

$$
\begin{equation*}
\alpha_{i}^{\star}=\bar{\alpha}_{i} \quad \text { for all } \quad i \in\{1, \ldots, n\} \quad \text { with } \quad-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}>b^{+}+L \operatorname{dist}[\boldsymbol{\alpha}, A] . \tag{7.10.4b}
\end{equation*}
$$

Proof. Let $\boldsymbol{\alpha}^{\star}$ be a solution of (7.1.1) with $\left\|\boldsymbol{\alpha}-\boldsymbol{\alpha}^{\star}\right\|=\operatorname{dist}[\boldsymbol{\alpha}, A]$ and let $i \in\{1, \ldots, n\}$ with $-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}<b^{-}-L \operatorname{dist}[\boldsymbol{\alpha}, A]$ be given. Then, (7.10.2) and (7.2.5) yield that

$$
-\nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top} \boldsymbol{e}^{i} \leq-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}+L \operatorname{dist}[\boldsymbol{\alpha}, A]<b^{-} \leq b^{\star} \leq b_{i}^{+}\left(\boldsymbol{\alpha}^{\star}\right) .
$$

If $\alpha_{i}^{\star}>\underline{\alpha}_{i}$, Proposition 7.3.2 implies $-\nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{\star}\right)^{\top} \boldsymbol{e}^{i}=b_{i}^{+}\left(\boldsymbol{\alpha}^{\star}\right)$, which would contradict the above inequality chain, i.e., $\alpha_{i}^{\star}=\underline{\alpha}_{i}$ must hold. This proves (7.10.4a). Condition (7.10.4b) follows in the same way.

Remark 7.10 .3 (detection of fixed variables in practice)
At this point, the application of the previous propositions does not seem to be practical because the conditions are formulated by means of the distance of a given point $\boldsymbol{\alpha} \in$ $\operatorname{dom}(\psi)$ to the solution set of the optimization problem. However, for practical applications it would be sufficient to estimate the value of dist[ $[\boldsymbol{\alpha}, A]$ from above in the following sense. Let $q_{1}, q_{2}, q_{3}: \mathbb{R}^{n} \rightarrow \mathbb{R}_{+}$be functions, which satisfy

$$
\begin{equation*}
\max \{L, 1\} \operatorname{dist}[\boldsymbol{\alpha}, A] \leq q_{j}(\boldsymbol{\alpha}) \quad \text { for } \quad j \in\{1,2,3\} \tag{7.10.5}
\end{equation*}
$$

for all $\boldsymbol{\alpha}$ in a suitable subset of dom $(\psi)$ and consider some arbitrary vector $\boldsymbol{\alpha}$ satisfying this condition. Then, under the assumptions of Proposition 7.10.1, (7.10.1) holds, which implies that $b^{-} \leq b^{\star} \leq b^{+}$for any solution $\left(\boldsymbol{w}^{\star}, b^{\star}\right) \in \mathcal{F} \times \mathbb{R}$ of (2.4.3) with

$$
\begin{equation*}
b^{-}:=b^{-}(\boldsymbol{\alpha}):=\sup _{i: \alpha_{i}<\bar{\alpha}_{i}-q_{1}(\boldsymbol{\alpha})} b_{i}^{-}(\boldsymbol{\alpha})-q_{2}(\boldsymbol{\alpha}) \tag{7.10.6a}
\end{equation*}
$$

and

$$
\begin{equation*}
b^{+}:=b^{+}(\boldsymbol{\alpha}):=\inf _{i: \alpha_{i}>\underline{\alpha}_{i}+q_{1}(\boldsymbol{\alpha})} b_{i}^{+}(\boldsymbol{\alpha})+q_{2}(\boldsymbol{\alpha}) . \tag{7.10.6b}
\end{equation*}
$$

Furthermore, any $i \in\{1, \ldots, n\}$ with

$$
-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}<b^{-}-q_{3}(\boldsymbol{\alpha})
$$

also satisfies the condition in (7.10.4a). Hence, we can conclude that there exists a soIution $\boldsymbol{\alpha}^{\star} \in \mathbb{R}^{n}$ of (7.1.1) with $\alpha_{i}^{\star}=\underline{\alpha}_{i}$. Accordingly, condition (7.10.4b) can be used to detect variables which can be fixed to their upper bound.

In order to make the practical application of the previous ideas possible we assume that the optimality measure $R$ provides a certain error bound. A similar condition is considered in [LL10]. In that paper the authors investigate some special formulations of the training problem and choose the natural residual for the corresponding linear complementarity problem as optimality measure.

Assumption 7.10 .4 (error bound condition)
Let Assumption 7.5.1 hold and denote by $A$ the solution set of the dual training problem (7.1.1). Fix $\rho>0$. Suppose that there exists a constant $K>0$ such that

$$
\begin{equation*}
\operatorname{dist}[\boldsymbol{\alpha}, A] \leq K R(\boldsymbol{\alpha}, \rho) \tag{7.10.7}
\end{equation*}
$$

for all $A \in \operatorname{dom}(\psi)$.

With the help of the previous assumption we formulate a framework in which it is possible to derive an upper bound of the type (7.10.5). This paves the way for the application in SMO optimization methods. In particular, we show how to construct a computable estimate for the set of variables which are fixed in an optimal solution and prove that this estimate is asymptotically correct.

Corollary 7.10 .5 (practical estimation of fixed variables)
Let $\boldsymbol{\alpha} \in \operatorname{dom}(\psi)$ be given and suppose that Assumption 7.10.4 is satisfied. Moreover, suppose that the minimizing set $A$ of $\psi$ is not empty and that $\nabla \psi_{\mathrm{d}, \mathrm{ext}}$ is Lipschitz continuous on $L_{\psi}(\psi(\boldsymbol{\alpha}))$. Define the index sets

$$
\begin{equation*}
\underline{I}(\boldsymbol{\alpha}):=\left\{i \in\{1, \ldots, n\} \mid-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}<b^{-}(\boldsymbol{\alpha})-q_{3}(\boldsymbol{\alpha})\right\} \tag{7.10.8a}
\end{equation*}
$$

and

$$
\begin{equation*}
\bar{I}(\boldsymbol{\alpha}):=\left\{i \in\{1, \ldots, n\} \mid-\nabla \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha})^{\top} \boldsymbol{e}^{i}>b^{+}(\boldsymbol{\alpha})+q_{3}(\boldsymbol{\alpha})\right\} \tag{7.10.8b}
\end{equation*}
$$

where the estimates $b^{-}(\boldsymbol{\alpha})$ and $b^{+}(\boldsymbol{\alpha})$ are defined according to (7.10.6) with

$$
\begin{equation*}
q_{j}(\boldsymbol{\alpha}):=r^{\diamond}(\boldsymbol{\alpha}, \rho)^{\eta} \quad \text { for } \quad j \in\{1,2,3\} \tag{7.10.9}
\end{equation*}
$$

for some $\rho>0$ and $\eta \in(0,1)$. Let $\boldsymbol{\alpha}$ satisfy

$$
\begin{equation*}
\operatorname{dist}[\boldsymbol{\alpha}, A] \leq\left(\max \{L, 1\}(K n)^{\eta}\right)^{-\frac{1}{1-\eta}} \tag{7.10.10}
\end{equation*}
$$

where $L$ denotes the Lipschitz constant of $\nabla \psi_{\mathrm{d}, \mathrm{ext}}$ and $K$ is the constant from Assumption 7.10.4. Then, there exists a solution $\boldsymbol{\alpha}^{\star} \in A$ of the dual training problem (7.1.1) such that

$$
\alpha_{i}^{\star}=\underline{\alpha}_{i} \text { for all } \quad i \in \underline{I}(\boldsymbol{\alpha}) \quad \text { and } \quad \alpha_{i}^{\star}=\bar{\alpha}_{i} \quad \text { for all } \quad i \in \bar{I}(\boldsymbol{\alpha}) .
$$

Proof. The main idea in the following is to combine Assumption 7.10 .4 with the results of Proposition 7.10.1 and Proposition 7.10.2. By assumption, all preconditions of these propositions are satisfied. Moreover, by means of (7.10.7) and Lemma 7.6 .5 we obtain

$$
\operatorname{dist}[\boldsymbol{\alpha}, A] \leq K R(\boldsymbol{\alpha}, \rho) \leq K n r^{\diamond}(\boldsymbol{\alpha}, \rho)
$$

This inequality together with (7.10.9) and (7.10.10) implies that

$$
\begin{aligned}
\operatorname{dist}[\boldsymbol{\alpha}, A] & =\operatorname{dist}[\boldsymbol{\alpha}, A]^{1-\eta} \operatorname{dist}[\boldsymbol{\alpha}, A]^{\eta} \\
& \leq \operatorname{dist}[\boldsymbol{\alpha}, A]^{1-\eta}(K n)^{\eta} r^{\diamond}(\boldsymbol{\alpha}, \rho)^{\eta} \\
& =\operatorname{dist}[\boldsymbol{\alpha}, A]^{1-\eta}(K n)^{\eta} q_{j}(\boldsymbol{\alpha}) \\
& \leq \frac{1}{\max \{L, 1\}} q_{j}(\boldsymbol{\alpha}),
\end{aligned}
$$

for all $j \in\{1,2,3\}$, i.e., (7.10.5) holds. Finally, the assertion of the corollary follows according to Remark 7.10.3.

Note that it is also possible to use the classical violation measure $r$ for the definition of $q_{j}$ if the inequality (7.10.10). However, because $\left\{r\left(\boldsymbol{\alpha}^{k}\right)\right\}_{k \in \mathbb{N}}$ does not necessarily converge to zero if $\left\{\boldsymbol{\alpha}^{k}\right\}_{k \in \mathbb{N}}$ converges to some solution of the dual training problem (see Example 7.2.5), the value of $q_{j}$ may not be meaningful in this case. Hence, the resulting index sets possibly contain only few elements and the prediction is not very expressive. In contrast, the definition using the bound-aware measure $r^{\diamond}$ works because of its continuity (due to Proposition 7.6.4).

Furthermore, it should be noted that the assertion of Corollary 7.10.5 is only a theoretical justification of the shrinking approach because it is in general not possible to determine whether the inequality (7.10.10) is satisfied. This means that practically, the functions $q_{j}(j \in\{1,2,3\})$ can be chosen in any suitable way such that a relation of the type (7.10.5) can be expected (at least asymptotically). For the application of the framework for detecting fixed variables this is relevant because the values of $b^{-}(\boldsymbol{\alpha})$ and $b^{+}(\boldsymbol{\alpha})$ can be infinite if $q_{1}(\boldsymbol{\alpha})$ is too large since the index set considered in the supremum and infimum, respectively, can be empty. In order to obtain any estimation, it is thus necessary to keep $q_{1}$ relatively small while maintaining (7.10.5) asymptotically. A reasonable approach would be to use

$$
q_{1}(\boldsymbol{\alpha}):=c_{1} r^{\diamond}\left(\boldsymbol{\alpha}, \rho_{1}\right)^{\eta_{1}}
$$

with $\rho_{1}>0, \eta_{1} \in(0,1)$ and $c_{1}>0$.
One can also see that it is not necessary to distinguish between $q_{2}$ and $q_{3}$ from a practical point of view because only their sum is relevant for the definition of the index sets in (7.10.8). Hence, only the sum has to be defined, which could have the form

$$
q_{2}(\boldsymbol{\alpha})+q_{3}(\boldsymbol{\alpha}):=c_{2} r^{\diamond}\left(\boldsymbol{\alpha}, \rho_{2}\right)^{\eta_{2}}
$$

with $\rho_{2}>0, \eta_{1} \in(0,1)$ and $c_{2}>0$. Of course, the choice of the parameters is an open question here and should be considered in the practical application, for instance for shrinking.

Finally, we want to emphasize that the correct prediction of fixed variables is not only helpful in the context of the SMO method. It can also be used to obtain a simpler optimization problem, which is equivalent to the dual training problem and can be solved exactly in some particular settings as follows.

Remark 7.10 .6 (equivalent training problem without bound constraints)
Suppose that Assumption 7.5.1 holds. Assume that there exists a solution $\boldsymbol{\alpha}^{\star}$ of (7.1.1) and an index set $I^{\star} \subseteq\{1, \ldots, n\}$ such that

$$
\begin{equation*}
\alpha_{i}=\alpha_{i}^{\star} \quad \text { for all } \quad i \in I^{\star} \tag{7.10.11}
\end{equation*}
$$

and

$$
\begin{equation*}
\alpha_{i} \in\left(\underline{\alpha}_{i}, \bar{\alpha}_{i}\right) \text { for all } \quad i \in\{1, \ldots, n\} \backslash I^{\star} . \tag{7.10.12}
\end{equation*}
$$

Then, any solution of the reduced problem

$$
\begin{array}{ll}
\min _{\boldsymbol{\alpha} \in \mathbb{R}^{n}} \psi_{\mathrm{d}, \mathrm{ext}}(\boldsymbol{\alpha}) \\
\text { s.t. } & \mathbf{1}^{\top} \boldsymbol{\alpha}=0,  \tag{7.10.13}\\
& \alpha_{i}=\alpha_{i}^{\star} \quad \text { for all } \quad i \in I^{\star}
\end{array}
$$

is also a solution of the dual training problem (7.1.1).
The detection of variables satisfying (7.10.11) is possible by means of Corollary 7.10.5. On the other hand, the set of free variables in (7.10.12) can also be detected asymptotically by means of a function of the form (7.10.9) if an error bound condition holds. Also note that one can easily verify whether a particular solution of the reduced problem (7.10.13) is also a solution of the original dual training problem by checking appropriate optimality conditions.

Finally, it should be pointed out that the solution of (7.10.13) can be obtained from a system of $n-\left|I^{\star}\right|+1$ linear equations if the objective function $\psi_{\mathrm{d}, \mathrm{ext}}$ is quadratic. This is in particular the case for the classical SVM training problems.

# 8 Optimization Methods for the Solution of the Primal Training Problem 


#### Abstract

The aim of this chapter is to introduce some particular methods for the solution of the primal training problem and discuss their practical applicability. For ease of presentation, we restrict the investigation again to the standard training problem as defined in Definition 2.4.5. However, note that many of the observations made below are also true or can be adapted in a suitable manner for other problem formulations. Throughout the chapter we assume that the loss functions defining the training problem are convex and sufficiently smooth. For the investigation of practical methods, we consider the finite-dimensional problem formulations introduced in Chapter 6.

We start this chapter with a brief summary of descent-based optimization methods (Section 8.1). Afterwards, we consider two first-order methods, namely the simple gradient descent method (Section 8.2) and a preconditioned one (Section 8.3). For the latter it is argued that sparse updates could occur theoretically and would lead to a reduction of the computational effort. This idea is then extended and transferred to a second-order method (Section 8.4). The particular formulation of a globalized Newton method is argued to be a new promising approach for the solution of the primal training problem. Because the structure of the Newton system is very similar to the linearization of the system of nonlinear equations introduced in Section 6.5, we briefly investigate the latter at the end of this chapter (Section 8.5).


### 8.1 Basic Idea of Descent-Based Methods

Before the practical application of optimization methods makes sense, it should be assured that the training problem has a solution in the first place. For this reason, we suppose that the objective function of (2.4.3) is coercive. Recall that Proposition 3.7.5 provides sufficiently weak conditions which guarantee coercivity. Note that the coercivity of the original objective function $\psi_{\mathrm{p}}$ does not necessarily imply that the objective function $\psi_{\text {pf }}$ of the finite-dimensional training problem (6.1.3) is also coercive. However, solvability of problem (6.1.3) is still guaranteed.

Because the training problem is supposed to be convex, several classes of optimization methods will generate sequences converging to a global solution of the problem in general. So the main question is not whether some algorithm is able to find a solution but how fast (and under which additional assumptions) it is able to find it.

Of course, the precise definition of the term 'fast' depends on the considered application. On the one hand, the theoretical convergence rate can be a suitable measure. On the other hand, one also has to consider the computational complexity and the memory consumption of a particular method. In order to assess the overall performance of a particular training algorithm, both parts are important to consider.

In the following, we restrict our investigation to iterative optimization methods which yield a descent of the objective function at each iteration. Such an iterative method can be characterized by a sequence of points

$$
\begin{equation*}
\boldsymbol{z}^{k+1}:=\boldsymbol{z}^{k}+\tau_{k} \boldsymbol{d}^{k} \quad \text { for } k \in\{0,1, \ldots\} \quad \text { with } \quad \psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k+1}\right) \leq \psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}\right), \tag{8.1.1}
\end{equation*}
$$

where $\boldsymbol{z}^{0}:=\left(\boldsymbol{\alpha}^{0}, b^{0}\right) \in \mathbb{R}^{n+1}$ denotes a suitable starting point and $\tau_{k} \in \mathbb{R}$ and $\boldsymbol{d}^{k} \in \mathbb{R}^{n+1}$ denote the step size and the direction at the $k$ th step, respectively.

If the loss functions are continuously differentiable, the training problems (6.1.2) and (6.3.1) are finite-dimensional convex optimization problems with continuously differentiable objective functions. Hence, classical derivative-based optimization methods are applicable. In the general case of non-smooth convex loss functions, the application of subgradient methods is still possible. However, we do not follow this approach here because it would go beyond the scope of this work.

Obviously, loss functions are generally not differentiable in many of the example training problems introduced in Chapter 4. If the non-smoothness comes from infinite values of the objective function (which model constraints in some sense), an approximate reformulation of the training problem is not obvious and the solution methods below might not be appropriate. Then, it could be suitable to apply methods for constrained optimization method which we only mention here for completeness.

For other examples, assuming smooth loss functions is not too restrictive because is it possible to use smooth approximations in many cases as proposed in Section 4.4. From the practical point of view, an approximation of an originally non-smooth problem (or a smooth formulation of it in the first place) is sufficient in most cases as long as certain fundamental properties of the loss functions are retained.

### 8.2 Simple Gradient Descent Method

Probably the simplest approach to solve the training problem is to use the negative gradient as search direction and define a particular line search strategy for the determination of the step size. In the general iterative scheme (8.1.1) this means that the direction is chosen as $\boldsymbol{d}^{k}:=-\nabla \psi\left(\boldsymbol{z}^{k}\right)$. For the step size it is sufficient to use Armijo's step size rule to obtain convergence, see [GK99, Satz 8.9]. In the reference it is shown that each accumulation point of the generated sequence $\left\{\boldsymbol{z}^{k}\right\}_{k \in \mathbb{N}}$ is a stationary point of the optimization problem. However, because the training problem is convex, each
stationary point is a solution of the problem. The overall algorithm is presented in Algorithm 8.2.1.

```
Algorithm 8.2.1: Gradient Descent Method
    Input: \(\boldsymbol{z}^{0} \in \mathbb{R}^{n+1}\), parameters \(\eta_{0}>0, \eta \in(0,1)\) and \(\sigma \in(0,1)\)
    for \(k=0,1,2, \ldots\) do
        if \(\nabla \psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}\right)=\mathbf{0}\) then
            Stop: \(\boldsymbol{z}^{k}\) is a solution of (6.1.2).
        Compute \(\boldsymbol{d}^{k}:=-\nabla \psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}\right)\).
        Let \(\tau_{k} \in\left\{\eta_{0} \cdot \eta^{\ell} \mid \ell \in\{0,1,2, \ldots\}\right\}\) be the largest number satisfying
                    \(\psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}+\tau_{k} \boldsymbol{d}^{k}\right) \leq \psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}\right)+\sigma \tau_{k} \nabla \psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}\right)^{\top} \boldsymbol{d}^{k}\).
        Update \(\boldsymbol{z}^{k+1}:=\boldsymbol{z}^{k}+\tau_{k} \boldsymbol{d}^{k}\).
```

Observe that the computationally most expensive part of the method is in the evaluation of the gradient in step 4. According to the particular formula given in (6.1.4), for the computation of the gradient it is necessary to evaluate entries of the vector $\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)$ for the current point $\boldsymbol{z}^{k}=\left(\boldsymbol{\alpha}^{k}, b^{k}\right)$. For this reason, the values of the decision function, i.e.,

$$
\begin{equation*}
t_{i}^{k}=\frac{1}{\lambda}\left[\boldsymbol{K} \boldsymbol{\alpha}^{k}\right]_{i}+b^{k} \quad \text { for } \quad i \in\{1, \ldots, n\} \tag{8.2.1}
\end{equation*}
$$

are needed. In a naive implementation, the computation of the values requires $n^{2}$ operations because of the matrix-vector product. Moreover, the whole kernel matrix is needed and has to be computed if it is not already known.

The computational effort can be reduced if only non-zero entries of the vector $\boldsymbol{\alpha}^{k}$ are considered. Obviously, vanishing entries do not contribute to the decision values. This leads to a reduced complexity of $n \cdot n_{\mathrm{NZ}}^{k}$ (where $n_{\mathrm{NZ}}^{k}$ denotes the number of nonzero entries in $\boldsymbol{\alpha}^{k}$ ) and for the computation only those columns of the kernel matrix associated with non-zero entries in $\boldsymbol{\alpha}^{k}$ are actually needed.
When $\boldsymbol{\alpha}^{k}$ is close to a solution of the training problem, the entries of the vector which correspond to non-support vectors of the problem are close to zero. However, this observation does not help to reduce the computational effort in practice, because one cannot expect entries of $\boldsymbol{\alpha}^{k}$ to be exactly equal to zero in general. This means that $n_{\mathrm{N} Z}^{k}$ will not be significantly smaller than $n$ and one still needs about $n^{2}$ operations for the computation of the gradient.

Furthermore, given the values of $\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)$, it is necessary to compute another matrixvector product with the kernel matrix in $\nabla \psi_{\text {pf }}\left(\boldsymbol{z}^{k}\right)$, cf. (6.1.4). In this second step, the corresponding vector in the product is

$$
\begin{equation*}
\boldsymbol{r}^{k}:=\boldsymbol{\alpha}^{k}+\boldsymbol{g}\left(\boldsymbol{z}^{k}\right) . \tag{8.2.2}
\end{equation*}
$$

Again, only those columns of $\boldsymbol{K}$ with non-zero entry in the vector $\boldsymbol{r}^{k}$ are needed. Note that due to (3.6.3), $\boldsymbol{r}^{k}=\mathbf{0}$ if $\boldsymbol{\alpha}^{k}$ is a solution of the dual problem and $b^{k}$ is chosen
suitably. Unfortunately, as we have seen in Section 6.4 the solution of the training problem (6.1.2) is not necessarily a dual solution. However, as this difference only happens in certain degenerate cases, one can expect $\boldsymbol{r}^{k}$ to be close to zero usually. Of course, it is not clear whether some entries of $\boldsymbol{r}^{k}$ are exactly zero. In general, for the gradient descent method one cannot expect to obtain exactly vanishing components. This means that the computation of the gradient usually requires $n^{2}$ operations and the evaluation of the whole kernel matrix.

This observation and the fact that the gradient descent method has a low convergence rate makes the practical application of the method not very attractive. Hence, we consider alternative approaches in the following.

### 8.3 Preconditioned Gradient Descent Method

While the finite-dimensional formulation in (6.1.2) is quite straight-forward, the second formulation (6.3.1) seems impractical at the first sight because the square root of the kernel matrix is needed therein. Subsequently, we will show it is possible to bypass the explicit computation of $\boldsymbol{K}^{\frac{1}{2}}$.

### 8.3.1 General Idea and Derivation

In the same way as in the original gradient descent approach (8.1.1), we consider an iterative scheme of the form

$$
\begin{equation*}
\tilde{\boldsymbol{z}}^{k+1}:=\tilde{\boldsymbol{z}}^{k}+\tau_{k} \tilde{\boldsymbol{d}}^{k} \quad \text { for } k \in\{0,1, \ldots\}, \tag{8.3.1}
\end{equation*}
$$

where $\tilde{\boldsymbol{z}}^{0}:=\left(\tilde{\boldsymbol{\alpha}}^{0}, b^{0}\right) \in \mathbb{R}^{n+1}$ denotes a suitable starting point and $\tau_{k} \in \mathbb{R}$ and $\tilde{\boldsymbol{d}}^{k} \in$ $\mathbb{R}^{n+1}$ denote the step size and the direction at the $k$ th step, respectively. As direction, we choose the componentwisely scaled negative gradient of the function $\tilde{\psi}_{\mathrm{pf}}$ at the current iterate, i.e.,

$$
\begin{equation*}
\tilde{\boldsymbol{d}}^{k}:=-\boldsymbol{W} \nabla \tilde{\psi}_{\mathrm{pf}}\left(\tilde{\boldsymbol{z}}^{k}\right) \tag{8.3.2}
\end{equation*}
$$

with a diagonal matrix

$$
\boldsymbol{W}:=\left(\begin{array}{cc}
\boldsymbol{I} & \mathbf{0} \\
\mathbf{0}^{\top} & \lambda^{-2}
\end{array}\right)
$$

The factor $\lambda^{-2}$ for the scaling of $b$ is motivated by practical experiments where it can be observed that the bias term converges relatively slow without that correction. Partially this can already be seen from the structure of the problem (6.3.1) because there the variable $\tilde{\boldsymbol{\alpha}}$ is scaled by the factor $\lambda^{-1}$ in the loss term but $b$ is not.

Because we used the relation $\tilde{\boldsymbol{\alpha}}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}$ to derive problem (6.3.1) in Section 6.3 we could also identify the generated sequence $\left\{\tilde{\boldsymbol{z}}^{k}\right\}_{k \in \mathbb{N}}:=\left\{\left(\tilde{\boldsymbol{\alpha}}^{k}, b^{k}\right)\right\}_{k \in \mathbb{N}}$ with a sequence $\left\{\boldsymbol{z}^{k}\right\}_{k \in \mathbb{N}}:=\left\{\left(\boldsymbol{\alpha}^{k}, b^{k}\right)\right\}_{k \in \mathbb{N}}$ using the relation $\tilde{\boldsymbol{\alpha}}^{k}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}^{k}$. This means that the gradient
given in (6.3.3) can be written as

$$
\nabla \tilde{\psi}_{\mathrm{pf}}\left(\tilde{\boldsymbol{z}}^{k}\right)=\binom{\frac{1}{\lambda}\left(\tilde{\boldsymbol{\alpha}}^{k}+\boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{g}}\left(\tilde{\boldsymbol{z}}^{k}\right)\right)}{\mathbf{1}^{\top} \tilde{\boldsymbol{g}}\left(\tilde{\boldsymbol{z}}^{k}\right)}=\left(\begin{array}{cc}
\boldsymbol{K}^{\frac{1}{2}} & \mathbf{0}  \tag{8.3.3}\\
\mathbf{0}^{\top} & 1
\end{array}\right)\binom{\frac{1}{\lambda}\left(\boldsymbol{\alpha}^{k}+\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\right)}{\mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)}
$$

Hence, we can write the update rule in (8.3.1) as

$$
\left(\begin{array}{cc}
\boldsymbol{K}^{\frac{1}{2}} & \mathbf{0} \\
\mathbf{0}^{\top} & 1
\end{array}\right) \boldsymbol{z}^{k+1}=\left(\begin{array}{cc}
\boldsymbol{K}^{\frac{1}{2}} & \mathbf{0} \\
\mathbf{0}^{\top} & 1
\end{array}\right) \boldsymbol{z}^{k}-\tau_{k}\left(\begin{array}{cc}
\boldsymbol{K}^{\frac{1}{2}} & \mathbf{0} \\
\mathbf{0}^{\top} & 1
\end{array}\right) \boldsymbol{W}\binom{\frac{1}{\lambda}\left(\boldsymbol{\alpha}^{k}+\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\right)}{\mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)}
$$

Note that each term in this equation is pre-multiplied by the same matrix. Consequently, we can define an explicit iterative scheme for the sequence $\left\{\boldsymbol{z}^{k}\right\}_{k \in \mathbb{N}}$, namely

$$
\begin{equation*}
\boldsymbol{z}^{k+1}:=\boldsymbol{z}^{k}-\frac{\tau_{k}}{\lambda}\binom{\boldsymbol{\alpha}^{k}+\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)}{\frac{1}{\lambda} \mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)} . \tag{8.3.4}
\end{equation*}
$$

This shows that the gradient descent method for the second formulation of the training problem given in (6.3.1) can also be interpreted as an iterative optimization of the form (8.1.1) with the direction

$$
\begin{equation*}
\boldsymbol{d}^{k}:=\frac{1}{\lambda}\binom{\boldsymbol{\alpha}^{k}+\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)}{\frac{1}{\lambda} \mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)} . \tag{8.3.5}
\end{equation*}
$$

Note that this method can also be seen as a preconditioned version of the simple gradient descent method. According to the argumentation given in Section 6.3, the sequence defined by (8.3.4) is not necessarily the only one which leads to (8.3.1) under the equation $\tilde{\boldsymbol{\alpha}}^{k}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}^{k}$. In theory, the value of $\boldsymbol{\alpha}^{k}$ is only uniquely determined if the kernel matrix is invertible. However, the sequence defined in (8.3.4) seems to be a natural choice even if the kernel matrix is not invertible because of its simple form.

### 8.3.2 Practical Implementation

In order to proceed with the step size selection in the resulting gradient descent method, we also need to derive an explicit term for the descent estimate $\nabla \tilde{\psi}_{\mathrm{pf}}\left(\tilde{\boldsymbol{z}}^{k}\right)^{\top} \tilde{\boldsymbol{d}}^{k}$. By definition of the direction $\tilde{\boldsymbol{d}}^{k}$ and using (8.3.3), we obtain

$$
-\nabla \tilde{\psi}_{\mathrm{pf}}\left(\tilde{\boldsymbol{z}}^{k}\right)^{\top} \tilde{\boldsymbol{d}}^{k}=\frac{1}{\lambda^{2}}\left(\left(\boldsymbol{r}^{k}\right)^{\top} \boldsymbol{K} \boldsymbol{r}^{k}+\left(\mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\right)^{2}\right)
$$

with $\boldsymbol{r}^{k}$ as defined in (8.2.2). Again, the term can be computed explicitly using the iterates $\boldsymbol{z}^{k}=\left(\boldsymbol{\alpha}^{k}, b^{k}\right)$ only. Thus, the explicit knowledge of the iterates $\tilde{\boldsymbol{z}}^{k}=\left(\tilde{\boldsymbol{\alpha}}^{k}, b^{k}\right)$ and the computation of the matrix $\boldsymbol{K}^{\frac{1}{2}}$ is not necessary at all. We summarize the resulting training method in Algorithm 8.3.1.

```
Algorithm 8.3.1: Preconditioned Gradient Descent Method
    Input: \(\boldsymbol{z}^{0} \in \mathbb{R}^{n+1}\), parameters \(\eta_{0}>0, \eta \in(0,1)\) and \(\sigma \in(0,1)\)
    for \(k=0,1,2, \ldots\) do
        Compute \(\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\) and \(\boldsymbol{r}^{k}:=\boldsymbol{\alpha}^{k}+\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\).
        if \(\boldsymbol{r}^{k}=\mathbf{0}\) and \(\mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)=0\) then
            Stop: \(\boldsymbol{z}^{k}\) is a solution of (6.1.2).
        Compute \(\boldsymbol{d}^{k}\) according to (8.3.5).
        Let \(\tau_{k} \in\left\{\eta_{0} \cdot \eta^{\ell} \mid \ell \in\{0,1,2, \ldots\}\right\}\) be the largest number satisfying
            \(\psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}+\tau_{k} \boldsymbol{d}^{k}\right) \leq \psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}\right)-\frac{\sigma \tau_{k}}{\lambda^{2}}\left(\left(\boldsymbol{r}^{k}\right)^{\top} \boldsymbol{K} \boldsymbol{r}^{k}+\left(\mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\right)^{2}\right)\).
        Update \(\boldsymbol{z}^{k+1}:=\boldsymbol{z}^{k}+\tau_{k} \boldsymbol{d}^{k}\).
```

Observe that the convergence of Algorithm 8.3.1 to a solution of the primal training follows by means of the well-known convergence properties of the gradient method (cf. [GK99, Satz 8.9]) because the method is a realization of the gradient descent method for the problem (6.3.1) and the variables are translated appropriately.

### 8.3.3 Discussion of the Computational Effort

With these preliminary considerations, we want to focus on the practical application of the preconditioned gradient descent method defined in Algorithm 8.3.1. In the same way as in the previous section, we note that the realization of one gradient step according to (8.3.4) involves the computation of the vectors $\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)$ and $\boldsymbol{r}^{k}$. Again, for the evaluation of the value of $\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)$, the values of the decision functions according to (8.2.1) are needed. This means that the computational effort for the computation of $\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)$ and $\boldsymbol{r}^{k}$ is proportional to $n \cdot n_{\mathrm{NZ}}^{k}$.

At first sight, the situation is not significantly different from what we have observed before because the number of non-zero entries in $\boldsymbol{\alpha}^{k}$ can be high, too. However, considering the update rule (8.3.4) there is hope to accomplish zero entries. This is because the training problem is often constructed in way that leads to a limited number of support vectors at optimum. In particular, the loss functions usually have vanishing derivative in a (typically infinitely) large interval. Moreover, a large portion of the optimal decision values fall into the corresponding interval which leads to a high number of non-support vectors.

In addition, typically chosen loss functions are not only zero in some interval but also affine linear in a large range. In particular this is true for the training problems considered in Proposition 4.1.1, Proposition 4.2.2, and the corresponding approximate formulations using smooth loss functions as introduced in Section 4.4. This means that $g_{i}(\boldsymbol{z})$ is often constant for $\boldsymbol{z}$ close to $\boldsymbol{z}^{k}$. If the sequence $\left\{\boldsymbol{z}^{k}\right\}_{k \in \mathbb{N}}$ converges to some point $\boldsymbol{z}^{\star}$, this fact can be used practically. We summarize the idea in the following observation.

Observation 8.3.1 (loss function with affine linear part)
Consider a standard training problem with continuously differentiable loss functions. Suppose that $\boldsymbol{z}^{\star}=\left(\boldsymbol{\alpha}^{\star}, b^{\star}\right)$ is a solution of (6.1.2). Let

$$
\varrho_{i}\left(t_{i}\right):=\inf \left\{\left|t-t_{i}\right| \mid t \in \mathbb{R}, \ell^{\prime}(t) \neq \ell^{\prime}\left(t_{i}\right)\right\}
$$

for $i \in\{1, \ldots, n\}$ denote the distance between $t_{i}$ and the closest point where the derivative of $\ell_{i}$ changes its value. Then, for all $i \in\{1, \ldots, n\}$ with

$$
\begin{equation*}
\varrho_{i}\left(\frac{1}{\lambda}\left[\boldsymbol{K} \boldsymbol{\alpha}^{\star}\right]_{i}+b^{\star}\right)>0 \tag{8.3.7}
\end{equation*}
$$

there exists an $\epsilon_{i}>0$ such that

$$
g_{i}(\boldsymbol{z})=g_{i}\left(\boldsymbol{z}^{\star}\right)
$$

for all $\boldsymbol{z} \in \mathcal{B}\left(\boldsymbol{z}^{\star}, \epsilon_{i}\right)$.

In order to discuss the practical implication of this observation, let $\boldsymbol{z}^{\star}:=\lim _{k \rightarrow \infty} \boldsymbol{z}^{k}$ denote the limit point of the generated sequence. Then, $g_{i}\left(\boldsymbol{z}^{k}\right)=g_{i}\left(\boldsymbol{z}^{\star}\right)$ whenever $\varrho_{i}\left(t_{i}^{\star}\right)>0$ and $k$ is large enough. Since the vector $\boldsymbol{g}\left(\boldsymbol{z}^{\star}\right)$ has zero components for all non-support vectors, this will also be true for $\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)$ if $\varrho_{i}\left(t_{i}^{\star}\right)>0$. Then, the upper part of (8.3.4) yields

$$
\begin{equation*}
\alpha_{i}^{k+1}:=\alpha_{i}^{k}-\frac{\tau_{k}}{\lambda}\left(\alpha_{i}^{k}+g_{i}\left(\boldsymbol{z}^{k}\right)\right) \tag{8.3.8}
\end{equation*}
$$

for $i \in\{1, \ldots, n\}$. This implies that $\alpha_{i}^{k+1}=-g_{i}\left(\boldsymbol{z}^{k}\right)$ if the step size is chosen as $\tau_{k}=\lambda$. In view of the Armijo step size rule according to (8.3.6), $\tau_{k}=\lambda$ follows if the initial step size is chosen as $\eta_{0}:=\lambda$ and no further reduction is needed. Indeed, the selection of the initial step size of $\lambda$ is quite natural when the training problem in (6.3.1) is under consideration because of the quadratic regularization term $\frac{1}{\lambda}\|\tilde{\boldsymbol{\alpha}}\|^{2}$, which would be minimized in one gradient descent step with this step size.

Due to (8.3.8) a step size of $\lambda$ generally leads to $\alpha_{i}^{k+1}=-g_{i}\left(\boldsymbol{z}^{k}\right)$. In the case that $g_{i}\left(\boldsymbol{z}^{k}\right)=g_{i}\left(\boldsymbol{z}^{\star}\right)$ this implies that $\alpha_{i}^{k+1}$ is updated to a value such that the $i$ th component of the optimality conditions (6.5.1) is exactly satisfied. Moreover, no further update happens at the subsequent step if $g_{i}\left(\boldsymbol{z}^{k+1}\right)=g_{i}\left(\boldsymbol{z}^{k}\right)$ because $r_{i}^{k+1}=0$. It follows that the update usually affects only a relatively small subset of the training set, namely the set of those training samples for which the decision function value falls into the nonlinear region of the loss function.

Subsequently, we want to note that only a smaller part of the kernel matrix is actually needed for the realization of one update step. First, observe that the decision function values can be updated iteratively as

$$
\begin{align*}
\boldsymbol{t}^{k+1} & =\frac{1}{\lambda} \boldsymbol{K} \boldsymbol{\alpha}^{k+1}+b^{k+1} \mathbf{1} \\
& =\boldsymbol{t}^{k}+\frac{1}{\lambda} \boldsymbol{K}\left(\boldsymbol{\alpha}^{k+1}-\boldsymbol{\alpha}^{k}\right)+\left(b^{k+1}-b^{k}\right) \mathbf{1}  \tag{8.3.9}\\
& =\boldsymbol{t}^{k}-\frac{\tau_{k}}{\lambda^{2}} \boldsymbol{K} \boldsymbol{r}^{k}+\left(b^{k+1}-b^{k}\right) \mathbf{1}
\end{align*}
$$

Hence, only the columns of $\boldsymbol{K}$ associated with non-zero entries of $\boldsymbol{r}^{k}$ are needed. Following the previous discussion, it can be assumed that these are not too many. Furthermore, the set of non-zero entries in $\boldsymbol{r}^{k}$ can be assumed to change not too drastically, i.e., it makes sense to use a caching procedure if the columns of the kernel matrix are computed on demand.

Using the update rule in (8.3.9), the computation of $\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)$ in step 2 of Algorithm 8.3.1 involves only $n$ operations. Moreover, a similar update procedure can be used for the evaluation of the objective function in (8.3.6). Then, the overall computational effort for one step in Algorithm 8.3.1 is proportional to $n \cdot n_{N Z(r)}^{k}$, where $n_{N Z(r)}^{k}$ denotes the number of non-zero entries of $\boldsymbol{r}^{k}$.

### 8.3.4 Shrinking of the Training Problem

Finally, we want to mention an approach which makes it possible to reduce the computational effort even more. The idea is basically the same as in the shrinking procedure which is applied commonly in implementations of the SMO method as discussed in Subsection 7.9.1.

Above we noted that the update rule (8.3.8) leads to exact updates (in the sense that $\left.\alpha_{i}^{k+1}=-g_{i}\left(\boldsymbol{z}^{k+1}\right)\right)$ in certain cases. Whenever the value of the $i$ th decision function stays in the same linear region of the $i$ th loss function, no further update is happening to the variable $\alpha_{i}$ in the subsequent steps. In practice this means that the variable can actually be ignored in the update routine. Moreover, it is not necessary to update the $i$ th decision function value anymore because it is not relevant for the following steps of the optimization method.

The preceding consideration leads to the idea of shrinking. After a predefined number of steps of the optimization routine, all variables which qualify for shrinking (because they satisfy a particular condition) are removed from the set of active variables. For those variables we omit the evaluation of the decision values and the loss terms entirely. Possible shrinking rules can be derived following the idea of Observation 8.3.1. Instead of the original training problem (6.1.2) a reduced problem of the form

$$
\min _{\substack{\boldsymbol{\alpha}_{I} \in \mathbb{R}^{I} \\ b \in \mathbb{R}^{I}}} \frac{1}{2 \lambda} \boldsymbol{\alpha}_{I}^{\top}\left(\boldsymbol{K}_{I I} \boldsymbol{\alpha}_{I}+\boldsymbol{K}_{I \bar{I}} \boldsymbol{\alpha}_{\bar{I}}\right)+\sum_{i \in I} \ell_{i}\left(\frac{1}{\lambda}[\boldsymbol{K} \boldsymbol{\alpha}]_{i}+b\right)
$$

with active set $I \subseteq\{1, \ldots, n\}$ and fixed set $\bar{I}:=\{1, \ldots, n\} \backslash I$ is considered. Note that the objective function is only used for the estimation of the descent condition in (8.3.6), and it is necessary to compute the term $\boldsymbol{K}_{I \bar{I}} \boldsymbol{\alpha}_{\bar{I}}$ (which occurs in the first and the second part of the objective function) only once because $\boldsymbol{\alpha}_{\bar{I}}$ is kept fixed.

Of course, we cannot be certain that the selection of the estimated set of fixed variables is correct. So, in general it is necessary to check whether the shrunk index set is still feasible for which it is essential to re-evaluate the $i$ th decision value. Since this evaluation is computationally expensive, one has to aim for a tradeoff between shrinking (which makes each subsequent step less expensive) and conservative selection of the shrunk index set (which prevents excessive re-evaluation of the decision function values).

### 8.3.5 Potential Application of Error Bound Conditions

In order to apply the shrinking idea practically, one could hope to obtain an error bound condition (EBC) for the solution set of the training problem by using a suitable set of first-order optimality conditions, for instance those defined in Corollary 3.4.2 or (for smooth loss functions) by the system (6.5.1). For the latter system an error bound condition holds if there exists some constant $K>0$ such that

$$
\begin{equation*}
\operatorname{dist}\left[(\boldsymbol{\alpha}, b), \mathcal{Z}^{\star}\right] \leq K\|\boldsymbol{F}(\boldsymbol{\alpha}, b)\| \quad \text { for all } \quad(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R}, \tag{8.3.10}
\end{equation*}
$$

where

$$
\mathcal{Z}^{\star}:=\left\{(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R} \mid \boldsymbol{F}(\boldsymbol{\alpha}, b)=\mathbf{0}\right\}
$$

denotes the set of solutions of the system. Note that it is also sufficient that (8.3.10) holds locally (i.e., for all $\boldsymbol{\alpha}$ in some neighborhood of $\mathcal{Z}^{\star}$ ) for a practical application. Indeed, a condition of the form (8.3.10) could help to derive an asymptotically exact estimate for the index set satisfying (8.3.7).

In the differentiable setting, in which the system (6.5.1) is defined, one can expect that the condition (8.3.10) is satisfied at least locally. However, such a condition is not available in the general case as we can see in the following example.

Example 8.3.2 (violated error bound condition)
Consider the problem

$$
\begin{equation*}
\min |b| \tag{8.3.11}
\end{equation*}
$$

which can be seen as a part of a training problem. Then, the unique solution of the problem is given by $b^{\star}=0$ and hence any feasible point $b \in \mathbb{R}$ has a distance of $|b|>0$ to the solution set. On the other hand, the ordinary first-order optimality condition is

$$
0 \in \partial|b|= \begin{cases}-1, & \text { if } b<0 \\ {[-1,1],} & \text { if } b=0 \\ 1, & \text { if } b>0\end{cases}
$$

In particular, the value of the subgradient contains no information about the distance of a point $b \neq 0$ to the solution set. Thus, the first-order optimality condition cannot be used to form an error bound condition in this example.

Note that the structure of the previous problem also occurs if the $\boldsymbol{w}$-part of the solution is fixed to its optimal value (which is unique in many applications due to Corollary 3.8.1). Then, the remaining problem (i.e., optimizing $b$ only) has a piecewise linear objective function similar to that of (8.3.11) in many practical applications in particular for the classical SVM in (4.1.1) and the $\varepsilon$-SVR in (4.2.1).

### 8.4 Application of Newton's Method

Building on the investigation of equivalent problems for the training problem in Chapter 6 we aim to derive a particular realization of Newton's method in the following. The resulting solution method basically follows the lines of the preconditioned gradient descent method presented in Algorithm 8.3.1 but uses another choice for the definition of the direction, namely the Newton direction, in each step.

Note that a similar approach was also proposed in [LM01], where the primal problem is treated in its original form (comparable to the standard training problem (2.4.3)) and smoothed versions of the classification loss functions are considered. In contrast to this, we describe the Newton method in a more general framework, which enables the application to a broader class of training problems and allows for a more efficient computation in certain cases. Furthermore, similar ideas are used in the application of weighted Least Squares Support Vector Machines [SDBLV02], the Iterative Re-Weighted Least Square procedure [PCNVADAR00] and the Lagrangian Support Vector Machines [FM03]. However, we do not pursue an extensive comparison to our approach here. In a nutshell, the most important drawback of these methods is that the application to large-scale is not feasible in practice. In contrast, our method aims at tackling exactly this problem.

Subsequently, we start with a version of a globalized Newton method based on a descent approach for a particular objective function. It is justified that this method converges to some solution of the training problem under rather weak assumptions. Moreover, we motivate that each step of the method can be conducted with a relatively low computational effort, if the starting point is not too far from the solution of the problem.

### 8.4.1 Analysis of the Newton System

For the application of Newton's method to the training problem (6.1.2) one could consider the system of linear equations

$$
\begin{equation*}
\nabla^{2} \psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b)\binom{\boldsymbol{\delta}}{\delta_{0}}=\nabla \psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b) \tag{8.4.1}
\end{equation*}
$$

However, the Hessian matrix of the objective function $\psi_{\mathrm{pf}}$ is not guaranteed to be invertible. In particular, if the kernel matrix is singular, the matrix $\nabla^{2} \psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b)$ is also singular for all $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R}$, which can be easily observed from the form provided by Proposition 6.1.4. Thus, a restriction to the subspace $\operatorname{ker}(\boldsymbol{K})^{\perp}$ as proposed in Proposition 6.2.2 seems a reasonable way to handle this issue. This leads again to the use of the second finite-dimensional formulation of the primal training problem given by Proposition 6.3.1. As in the derivation of the preconditioned gradient descent method above, we will see, that it is actually not necessary to compute $\boldsymbol{K}^{\frac{1}{2}}$ explicitly in order to apply Newton's method because all algorithmic steps can be transferred to the variables of the first finite-dimensional training problem.

To start with an explanation of this observation, we examine the gradient and the Hessian of the objective function $\tilde{\psi}_{\text {pf }}$. Using the representation of the derivatives in Proposition 6.3.2, we consider the Newton system for problem (6.3.1), which is given by

$$
\begin{equation*}
\nabla^{2} \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b)\binom{\tilde{\boldsymbol{\delta}}}{\delta_{0}}=\nabla \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b) \tag{8.4.2}
\end{equation*}
$$

As the function $\tilde{\psi}_{\text {pf }}$ is convex its Hessian is at least positive semi-definite. Moreover, since $\tilde{\psi}_{\mathrm{pf}}$ is also uniformly convex with respect to $\tilde{\boldsymbol{\alpha}}$ one would expect that the Hessian is positive definite in most cases. In the following proposition we identify under which conditions this is the case.

Proposition 8.4.1 (positive definiteness of $\nabla^{2} \tilde{\psi}_{\mathrm{pf}}$ )
Consider a standard training problem according to Definition 2.4 .5 with twice continuously differentiable loss functions. Let $\tilde{\psi}_{\mathrm{pf}}$ be defined according to (6.3.2).
Then, the Hessian matrix $\nabla^{2} \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b)$ is positive definite if and only if $\mathbf{1}^{\top} \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)>0$.
Proof. First, note that $\mathbf{1}^{\top} \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)=0$ if and only if $\tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)=\mathbf{0}$ because the loss functions are convex and the vector $\tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)$ consists of the second derivatives. Then, it is easy to see that the matrix

$$
\lambda \nabla^{2} \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b)=\left(\begin{array}{cc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{H}} \boldsymbol{K}^{\frac{1}{2}} & \boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{H}} \mathbf{1} \\
\mathbf{1}^{\top} \tilde{\boldsymbol{H}} \boldsymbol{K}^{\frac{1}{2}} & \lambda \mathbf{1}^{\top} \tilde{\boldsymbol{H}} \mathbf{1}
\end{array}\right)
$$

is not invertible if $\tilde{\boldsymbol{H}}=\boldsymbol{O}$, i.e., if $\mathbf{1}^{\top} \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)=0$.
Now, suppose that $\mathbf{1}^{\top} \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)>0$. A well-known fact is that a matrix is positive definite if and only if its Schur complement is positive definite, cf. [H]12, Theorem 7.7.7]. For the block matrix $\lambda \nabla^{2} \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b)$ there are two possibilities to form a Schur complement. Subsequently, we consider the form

$$
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{H}} \boldsymbol{K}^{\frac{1}{2}}-\frac{1}{\lambda \mathbf{1}^{\top} \tilde{\boldsymbol{H}} \mathbf{1}} \boldsymbol{K}^{\frac{1}{2}} \tilde{\boldsymbol{H}} \mathbf{1} \mathbf{1}^{\top} \tilde{\boldsymbol{H}} \boldsymbol{K}^{\frac{1}{2}}=\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{K}^{\frac{1}{2}}\left(\tilde{\boldsymbol{H}}-\frac{1}{\mathbf{1}^{\top} \tilde{\boldsymbol{H}} \mathbf{1}} \tilde{\boldsymbol{H}} \mathbf{1} \mathbf{1}^{\top} \tilde{\boldsymbol{H}}\right) \boldsymbol{K}^{\frac{1}{2}} .
$$

It remains to show that this matrix is positive definite. Fur this purpose, let $\boldsymbol{\gamma} \in \mathbb{R}^{n}$ be some arbitrary vector. Then, by means of the Cauchy-Schwarz inequality we obtain

$$
\gamma^{\top} \tilde{\boldsymbol{H}} 11^{\top} \tilde{\boldsymbol{H}} \gamma=\left(\gamma^{\top} \tilde{\boldsymbol{H}} 1\right)^{2}=\left(\left(\tilde{\boldsymbol{H}}^{\frac{1}{2}} \gamma\right)^{\top} \tilde{\boldsymbol{H}}^{\frac{1}{2}} \mathbf{1}\right)^{2} \leq\left\|\tilde{\boldsymbol{H}}^{\frac{1}{2}} \gamma\right\|^{2} \cdot\left\|\tilde{\boldsymbol{H}}^{\frac{1}{2}} 1\right\|^{2}=\gamma^{\top} \tilde{\boldsymbol{H}} \gamma \cdot \mathbf{1}^{\top} \tilde{\boldsymbol{H}} \mathbf{1} .
$$

This implies that

$$
\gamma^{\top}\left(\tilde{\boldsymbol{H}}-\frac{1}{\mathbf{1}^{\top} \tilde{\boldsymbol{H}} \mathbf{1}} \tilde{\boldsymbol{H}} 11^{\top} \tilde{\boldsymbol{H}}\right) \gamma \geq 0,
$$

i.e., the Schur complement of $\left.\lambda \nabla^{2} \tilde{\psi}_{\text {pf }} \tilde{\boldsymbol{\alpha}}, b\right)$ is positive definite. Hence, the Hessian matrix $\nabla^{2} \tilde{\psi}_{\text {pf }}(\tilde{\boldsymbol{\alpha}}, b)$ is also positive definite if $\mathbf{1}^{\top} \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)>0$.

### 8.4.2 Efficient Solution of the Newton System

As the theory indicates that Newton's method can be used to solve the training problem, the question arises whether an efficient implementation is possible in practice. Following the assertion of Proposition 8.4.1, we suppose that $\mathbf{1}^{\top} \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)>0$ holds when we consider the Newton system such that it is solvable by assumption. Then, taking into account the relations $\tilde{\boldsymbol{\alpha}}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}$ and $\tilde{\boldsymbol{\delta}}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\delta}$, the Newton system (8.4.2) can be written as

$$
\frac{1}{\lambda}\left(\begin{array}{cc}
\boldsymbol{K}^{\frac{1}{2}} & \mathbf{0}  \tag{8.4.3}\\
\mathbf{0}^{\top} & 1
\end{array}\right)\left(\begin{array}{cc}
\boldsymbol{I}+\frac{1}{\lambda} \tilde{\boldsymbol{H}} \boldsymbol{K} & \tilde{\boldsymbol{H}} \mathbf{1} \\
\mathbf{1}^{\top} \tilde{\boldsymbol{H}} \boldsymbol{K} & \lambda \mathbf{1}^{\top} \tilde{\boldsymbol{H}} \mathbf{1}
\end{array}\right)\binom{\boldsymbol{\delta}}{\delta_{0}}=\left(\begin{array}{cc}
\boldsymbol{K}^{\frac{1}{2}} & \mathbf{0} \\
\mathbf{0}^{\top} & 1
\end{array}\right)\binom{\frac{1}{\lambda}(\boldsymbol{\alpha}+\boldsymbol{g}(\boldsymbol{\alpha}, b))}{\mathbf{1}^{\top} \boldsymbol{g}(\boldsymbol{\alpha}, b)}
$$

Because of the special structure of the Hessian matrix, we can now observe that it is possible to compute a solution of the Newton system rather efficiently. The basic procedure to do this is summarized in Algorithm 8.4.1 and justified in the subsequent proposition.

```
Algorithm 8.4.1: Solution of the Newton system (8.4.1)
    1 Let \(J:=\left\{i \in\{1, \ldots, n\} \mid h_{i}(\boldsymbol{\alpha}, b)>0\right\}\) and \(\bar{J}:=\{1, \ldots, n\} \backslash J\).
    2 Compute
\[
\begin{equation*}
\boldsymbol{\delta}_{\bar{J}}:=\boldsymbol{\alpha}_{\bar{J}}+\boldsymbol{g}_{\bar{J}}(\boldsymbol{\alpha}, b) . \tag{8.4.4}
\end{equation*}
\]
```

3 Compute

$$
\begin{equation*}
\boldsymbol{a}_{J}:=\boldsymbol{H}_{J J}^{-1}\left(\boldsymbol{\alpha}_{J}+\boldsymbol{g}_{J}(\boldsymbol{\alpha}, b)\right)-\frac{1}{\lambda} \boldsymbol{K}_{J \bar{J}} \boldsymbol{\delta}_{\bar{J}} \quad \text { and } \quad a_{0}:=\mathbf{1}^{\top} \boldsymbol{\alpha}_{J}-\mathbf{1}^{\top} \boldsymbol{g}_{\bar{J}}(\boldsymbol{\alpha}, b) \text {. } \tag{8.4.5}
\end{equation*}
$$

4 With $\boldsymbol{Q}:=\boldsymbol{H}_{J J}^{-1}+\frac{1}{\lambda} \boldsymbol{K}_{J J}$ compute

$$
\begin{equation*}
\boldsymbol{p}_{J}:=\boldsymbol{Q}^{-1} \boldsymbol{a}_{J} \quad \text { and } \quad \boldsymbol{q}_{J}:=\boldsymbol{Q}^{-1} \mathbf{1} \tag{8.4.6}
\end{equation*}
$$

5 Compute

$$
\begin{equation*}
\delta_{0}:=\frac{\mathbf{1}^{\top} \boldsymbol{p}_{J}-a_{0}}{\mathbf{1}^{\top} \boldsymbol{q}_{J}} \quad \text { and } \quad \boldsymbol{\delta}_{J}:=\boldsymbol{p}_{J}-\delta_{0} \boldsymbol{q}_{J} . \tag{8.4.7}
\end{equation*}
$$

The major advantage of the application of Algorithm 8.4.1 for the solution of the Newton system is that a part of the solution (namely all variables with indices belonging to the set $\bar{J}$ ) can be determined very easily due to (8.4.4). The remaining part of the variables is then defined by the solutions of two systems of linear equations with the same system matrix which can be rather small (depending on the size of the index set $J$ ), cf. (8.4.6) and (8.4.7).

Of course, the sizes of the sets $J$ and $\bar{J}$ depend on the current point and the particular choice of the loss functions. We will discuss this issue again later on in Subsection 8.4.4.

In the following proposition we show that Algorithm 8.4.1 actually yields a solution
of the Newton system (8.4.1). Furthermore, we state the relation between the Newton system and another related system of linear equations.

Proposition 8.4.2 (solution of the Newton system)
Consider a standard training problem according to Definition 2.4.5 with twice continuously differentiable loss functions. Let $(\boldsymbol{\alpha}, b) \in \mathbb{R}^{n} \times \mathbb{R}$ satisfy $\mathbf{1}^{\top} \boldsymbol{h}(\boldsymbol{\alpha}, b)>0$.
Then, the system

$$
\left(\begin{array}{cc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{H} \boldsymbol{K} & \boldsymbol{H} \mathbf{1}  \tag{8.4.8}\\
\mathbf{1}^{\top} \boldsymbol{H} \boldsymbol{K} & \lambda \mathbf{1}^{\top} \boldsymbol{H} \mathbf{1}
\end{array}\right)\binom{\boldsymbol{\delta}}{\delta_{0}}=\binom{\boldsymbol{\alpha}+\boldsymbol{g}(\boldsymbol{\alpha}, b)}{\lambda \mathbf{1}^{\top} \boldsymbol{g}(\boldsymbol{\alpha}, b)}
$$

has a unique solution which is computed by means of Algorithm 8.4.1. Moreover, this solution is also a solution of the Newton system (8.4.1) and of the system (8.4.2) with $\tilde{\boldsymbol{\delta}}=$ $\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\delta}$ and $\tilde{\boldsymbol{\alpha}}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}$. In particular, $\left(\boldsymbol{\delta}, \delta_{0}\right)$ is an ascent direction for $\psi_{\mathrm{pf}}$ at $(\boldsymbol{\alpha}, b)$ if and only if $(\boldsymbol{\alpha}, b)$ is not a solution of the finite-dimensional primal training problem (6.1.2).

Proof. Because the loss functions are convex, the entries of the vector $\boldsymbol{h}(\boldsymbol{\alpha}, b)$ are nonnegative. Let $J:=\left\{i \in\{1, \ldots, n\} \mid h_{i}(\boldsymbol{\alpha}, b)>0\right\}$ and $\bar{J}:=\{1, \ldots, n\} \backslash J$. Then, the system (8.4.8) can be written in blocks as

$$
\left(\begin{array}{ccc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{H}_{J J} \boldsymbol{K}_{J J} & \frac{1}{\lambda} \boldsymbol{H}_{J J} \boldsymbol{K}_{J \bar{J}} & \boldsymbol{H}_{J J} \mathbf{1} \\
\boldsymbol{O} & \boldsymbol{I} & \mathbf{0} \\
\mathbf{1}^{\top} \boldsymbol{H}_{J J} \boldsymbol{K}_{J J} & \mathbf{1}^{\top} \boldsymbol{H}_{J J} \boldsymbol{K}_{J \bar{J}} & \lambda \mathbf{1}^{\top} \boldsymbol{H}_{J J} \mathbf{1}
\end{array}\right)\left(\begin{array}{c}
\boldsymbol{\delta}_{J} \\
\boldsymbol{\delta}_{\bar{J}} \\
\delta_{0}
\end{array}\right)=\left(\begin{array}{c}
\boldsymbol{\alpha}_{J}+\boldsymbol{g}_{J}(\boldsymbol{\alpha}, b) \\
\boldsymbol{\alpha}_{\bar{J}}+\boldsymbol{g}_{\bar{J}}(\boldsymbol{\alpha}, b) \\
\lambda \mathbf{1}^{\top} \boldsymbol{g}(\boldsymbol{\alpha}, b)
\end{array}\right) .
$$

In this representation, it is easy to see that $\boldsymbol{\delta}_{\bar{J}}$ given by (8.4.4) is uniquely determined. It remains to consider the reduced system

$$
\left(\begin{array}{cc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{H}_{J J} \boldsymbol{K}_{J J} & \boldsymbol{H}_{J J} \mathbf{1}  \tag{8.4.9}\\
\mathbf{1}^{\top} \boldsymbol{H}_{J J} \boldsymbol{K}_{J J} & \lambda \mathbf{1}^{\top} \boldsymbol{H}_{J J} \mathbf{1}
\end{array}\right)\binom{\boldsymbol{\delta}_{J}}{\delta_{0}}=\binom{\tilde{\boldsymbol{a}}_{J}}{\tilde{a}_{0}}
$$

with right-hand side

$$
\binom{\tilde{\boldsymbol{a}}_{J}}{\tilde{a}_{0}}:=\binom{\boldsymbol{\alpha}_{J}+\boldsymbol{g}_{J}(\boldsymbol{\alpha}, b)}{\lambda \mathbf{1}^{\top} \boldsymbol{g}(\boldsymbol{\alpha}, b)}-\binom{\frac{1}{\lambda} \boldsymbol{I}}{\mathbf{1}^{\top}} \boldsymbol{H}_{J J} \boldsymbol{K}_{J \bar{J}} \boldsymbol{\delta}_{\bar{J}} .
$$

By summing up the first block of equations in (8.4.9), we obtain

$$
\mathbf{1}^{\top} \boldsymbol{\delta}_{J}+\frac{1}{\lambda} \mathbf{1}^{\top} \boldsymbol{H}_{J J} \boldsymbol{K}_{J J} \boldsymbol{\delta}_{J}+\mathbf{1}^{\top} \boldsymbol{H}_{J J} \mathbf{1} \delta_{0}=\mathbf{1}^{\top} \tilde{\boldsymbol{a}}_{J}
$$

Now, subtracting $\lambda^{-1}$ times the second block of (8.4.9) from this equation, it follows that the reduced system is equivalent to

$$
\left(\begin{array}{cc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{H}_{J J} \boldsymbol{K}_{J J} & \boldsymbol{H}_{J J} \mathbf{1} \\
\mathbf{1}^{\top} & 0
\end{array}\right)\binom{\boldsymbol{\delta}_{J}}{\delta_{0}}=\binom{\tilde{\boldsymbol{a}}_{J}}{\mathbf{1}^{\top} \tilde{\boldsymbol{a}}_{J}-\frac{1}{\lambda} \tilde{a}_{0}} .
$$

Note that

$$
\mathbf{1}^{\top} \tilde{\boldsymbol{a}}_{J}=\mathbf{1}^{\top}\left(\boldsymbol{\alpha} \boldsymbol{\alpha}_{J}+\boldsymbol{g}_{J}(\boldsymbol{\alpha}, b)\right)-\frac{1}{\lambda} \mathbf{1}^{\top} \boldsymbol{H}_{J J} \boldsymbol{K}_{J \bar{J}} \boldsymbol{\delta}_{\bar{J}}=\mathbf{1}^{\top} \boldsymbol{\alpha}_{J}-\mathbf{1}^{\top} \boldsymbol{g}_{\bar{J}}(\boldsymbol{\alpha}, b)+\frac{1}{\lambda} \tilde{a}_{0} .
$$

This means that the second term of the right-hand side is equal to $a_{0}:=\mathbf{1}^{\top} \boldsymbol{\alpha}_{J}-$ $\mathbf{1}^{\top} \boldsymbol{g}_{\bar{J}}(\boldsymbol{\alpha}, b)$ as defined in (8.4.5). Lastly, because $\boldsymbol{H}_{J J}$ is invertible (as a diagonal matrix with only positive entries), the system is equivalent to

$$
\left(\begin{array}{cc}
\boldsymbol{H}_{J J}^{-1}+\frac{1}{\lambda} \boldsymbol{K}_{J J} & \mathbf{1} \\
\mathbf{1}^{\top} & 0
\end{array}\right)\binom{\boldsymbol{\delta}_{J}}{\delta_{0}}=\binom{\boldsymbol{a}_{J}}{a_{0}}
$$

with $\boldsymbol{a}_{J}:=\boldsymbol{H}_{J J}^{-1} \tilde{\boldsymbol{a}}_{J}$ as defined in (8.4.5). Because the kernel matrix $\boldsymbol{K}_{J J}$ is positive semidefinite, it follows that the matrix $\boldsymbol{Q}:=\boldsymbol{H}_{J J}^{-1}+\frac{1}{\lambda} \boldsymbol{K}_{J J}$ is positive definite. In particular, it is invertible and the term $\mathbf{1}^{\top} \boldsymbol{Q}^{-1} \mathbf{1}$ is positive. This means that the upper block of the previous system can further be rewritten as

$$
\begin{equation*}
\boldsymbol{\delta}_{J}+\boldsymbol{Q}^{-1} \mathbf{1} \delta_{0}=\boldsymbol{Q}^{-1} \boldsymbol{a}_{J} \tag{8.4.10}
\end{equation*}
$$

Moreover, summing up both sides of the equation yields

$$
\delta_{0}=\frac{\mathbf{1}^{\top} \boldsymbol{Q}^{-1} \boldsymbol{a}_{J}-a_{0}}{\mathbf{1}^{\top} \boldsymbol{Q}^{-1} \mathbf{1}}
$$

This together with (8.4.10) and the equation for $\boldsymbol{\delta}_{\bar{J}}$ in (8.4.4) shows that the solution of system (8.4.8) is uniquely determined. Note that the definitions in (8.4.7) of Algorithm 8.4.1 are equal to the terms above with the vectors $\boldsymbol{p}_{J}$ and $\boldsymbol{q}_{J}$ previously computed in (8.4.6).

It is easy to see that the solution of (8.4.8) also satisfies the Newton system (8.4.1) because the latter equation is implied by the former when the first block of equations is multiplied by $\boldsymbol{K}$. It also directly follows that $\left(\tilde{\boldsymbol{\delta}}, \delta_{0}\right)$ with $\tilde{\boldsymbol{\delta}}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\delta}$ is a solution of the system (8.4.2) if one uses that $\tilde{\boldsymbol{\alpha}}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}$ and (8.4.3).

Finally, we use the equivalence between the problems (6.1.2) and (6.3.1) to show that the direction $\left(\boldsymbol{\delta}, \delta_{0}\right)$ is an ascent direction for $\psi_{\mathrm{pf}}$ at $(\boldsymbol{\alpha}, b)$ if and only if $(\boldsymbol{\alpha}, b)$ is not a solution of (6.1.2). If ( $\boldsymbol{\alpha}, b$ ) solves problem (6.1.2), it follows that $\nabla \psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b)=\mathbf{0}$ and there exists no ascent direction. Otherwise, the point $(\tilde{\boldsymbol{\alpha}}, b)$ with $\tilde{\boldsymbol{\alpha}}=\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}$ is not optimal for (6.3.1). Because (under the presumption that $\mathbf{1}^{\top} \boldsymbol{h}(\boldsymbol{\alpha}, b)=\mathbf{1}^{\top} \tilde{\boldsymbol{h}}(\tilde{\boldsymbol{\alpha}}, b)>0$ ) the Hessian $\nabla_{\sim}^{2} \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{\alpha}}, b)$ is positive definite, i.e., the Newton direction $\left(\tilde{\boldsymbol{\delta}}, \delta_{0}\right)$ is an ascent direction for $\tilde{\psi}_{\text {pf }}$ at ( $\left.\tilde{\boldsymbol{\alpha}}, b\right)$. Finally, by construction, $\left(\boldsymbol{\delta}, \delta_{0}\right)$ is also an ascent direction for $\psi_{\text {pf }}$ at $(\boldsymbol{\alpha}, b)$ because $\psi_{\mathrm{pf}}(\boldsymbol{\alpha}, b)=\tilde{\psi}_{\mathrm{pf}}\left(\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\alpha}, b\right)$.

### 8.4.3 Formulation of Newton's Method

In conclusion of the previous observations, we want to formulate a particular realization of a globalized Newton method, which makes it possible to infer guaranteed
convergence. In particular, we introduce a line search procedure to ensure a sufficient decrease of the objective function in each step. Additionally, in order to avoid degenerated directions, a fallback to the search direction from the preconditioned gradient descent method is applied if necessary. The resulting method is summarized in Algorithm 8.4.2. In the definition of the algorithm we use the terms

$$
\begin{equation*}
\nabla \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{z}})^{\top} \tilde{\boldsymbol{d}}=\binom{\frac{1}{\lambda} \boldsymbol{K}^{\frac{1}{2}}(\boldsymbol{\alpha}+\boldsymbol{g}(\boldsymbol{\alpha}, b))}{\mathbf{1}^{\top} \boldsymbol{g}(\boldsymbol{\alpha}, b)}^{\top}\binom{\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\delta}}{\delta_{0}}=\frac{1}{\lambda} \boldsymbol{r}^{\top} \boldsymbol{K} \boldsymbol{\delta}+\mathbf{1}^{\top} \boldsymbol{g}(\boldsymbol{z}) \delta_{0} \tag{8.4.11}
\end{equation*}
$$

see (6.3.3), and

$$
\begin{equation*}
\|\tilde{\boldsymbol{d}}\|=\sqrt{\boldsymbol{\delta}^{\top} \boldsymbol{K} \boldsymbol{\delta}+\delta_{0}^{2}} \tag{8.4.12}
\end{equation*}
$$

with the identification $\tilde{\boldsymbol{d}}=\left(\boldsymbol{K}^{\frac{1}{2}} \boldsymbol{\delta}, \delta_{0}\right)$.

```
Algorithm 8.4.2: Newton's method for the primal training problem
    Input: \(\boldsymbol{z}^{0} \in \mathbb{R}^{n+1}\), parameters \(\mu>0, p>2, \eta \in(0,1)\) and \(\sigma \in\left(0, \frac{1}{2}\right)\)
    for \(k=0,1,2, \ldots\) do
        Compute \(\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\) and \(\boldsymbol{r}^{k}:=\boldsymbol{\alpha}^{k}+\boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\).
        if \(\boldsymbol{r}^{k}=\mathbf{0}\) and \(\mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)=0\) then
            Stop: \(\boldsymbol{z}^{k}\) is a solution of (6.1.2).
        if \(\mathbf{1}^{\top} \boldsymbol{h}\left(\boldsymbol{z}^{k}\right)>0\) then
            Compute \(\boldsymbol{d}^{k, N}\) as solution of (8.4.8) with \((\boldsymbol{\alpha}, b):=\boldsymbol{z}^{k}\).
            Compute \(R_{k}:=-\nabla \tilde{\psi}_{\mathrm{pf}}\left(\tilde{\boldsymbol{z}}^{k}\right)^{\top} \tilde{\boldsymbol{d}}^{k, N}\) and \(\left\|\tilde{\boldsymbol{d}}^{k, N}\right\|\) according to (8.4.11) and
            (8.4.12).
        if \(\boldsymbol{d}^{k, N}\) is available and \(R_{k} \leq-\mu\left\|\tilde{\boldsymbol{d}}^{k, N}\right\|^{p}\) then
            Define \(\boldsymbol{d}^{k}:=\boldsymbol{d}^{k, N}\).
        else
            Compute \(\boldsymbol{d}^{k}\) according to (8.3.5).
            Compute \(R_{k}:=\frac{1}{\lambda^{2}}\left(\boldsymbol{r}^{k}\right)^{\top} \boldsymbol{K} \boldsymbol{r}^{k}+\left(\mathbf{1}^{\top} \boldsymbol{g}\left(\boldsymbol{z}^{k}\right)\right)^{2}\).
        Let \(\tau_{k} \in\left\{\eta^{\ell} \mid \ell \in\{0,1,2, \ldots\}\right\}\) be the largest number satisfying
                    \(\psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}+\tau_{k} \boldsymbol{d}^{k}\right) \leq \psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}\right)-\sigma \tau_{k} R_{k}\).
        Update \(\boldsymbol{z}^{k+1}:=\boldsymbol{z}^{k}+\tau_{k} \boldsymbol{d}^{k}\).
```

In the same way as for the convergence of the preconditioned gradient descent method in Section 8.3 it is also possible to obtain convergence for Algorithm 8.4.2. To do this, we interpret the method as a realization of Newton's method for the objective function $\tilde{\psi}_{\mathrm{pf}}$ as defined in (6.3.1). With this approach the convergence of the algorithm to solutions of the training problem follows immediately by means of [GK99, Satz 9.5].

### 8.4.4 Practical Remarks

It should be noted that the efficiency of the proposed implementation of Newton's method highly depends on the fact that only a small proportion of the kernel matrix is needed in each step. In Algorithm 8.4.1 for the solution of the Newton system this is already reflected partially. However, the full realization of Newton's method according to Algorithm 8.4.2 requires more steps, in which certain columns of $\boldsymbol{K}$ are potentially needed. This is especially the case for the line search step (line 13) which requires the value of $\nabla \tilde{\psi}_{\mathrm{pf}}(\tilde{\boldsymbol{z}})^{\top} \tilde{\boldsymbol{d}}$ given by (8.4.11). To estimate the effort for its computation, the sparseness structure of the step vector $\boldsymbol{d}^{k}$ can be used as follows.

Similarly to the argumentation for the preconditioned gradient descent method in Subsection 8.3.3, it is also possible to find out that one step of Newton's method can have a remarkable effect on particular training samples. To do this, we consider a fixed step $k$ in the following. Let $i \in\{1, \ldots, n\}$ be some index with $h_{i}\left(\boldsymbol{\alpha}^{k}, b^{k}\right)=0$. Then, the $i$ th component of the Newton direction is equal to

$$
\delta_{i}^{k}=\alpha_{i}^{k}+g_{i}\left(\boldsymbol{\alpha}^{k}, b^{k}\right) .
$$

When the Newton method is considered, one usually expects the step size to be equal to one. In particular, close to a non-degenerated solution of the problem, the unit step size is guaranteed in theory, cf. [GK99, Lemma 9.9]. If the full step length $\tau_{k}=1$ is accepted, the update yields for the $i$ th component that

$$
\begin{equation*}
\alpha_{i}^{k+1}=\alpha_{i}^{k}-\tau_{k} \delta_{i}^{k}=-g_{i}\left(\boldsymbol{\alpha}^{k}, b^{k}\right) . \tag{8.4.13}
\end{equation*}
$$

This property is especially useful if the derivate of the $i$ th loss function is not changed by the update, i.e., if $g_{i}\left(\boldsymbol{\alpha}^{k+1}, b^{k+1}\right)=g_{i}\left(\boldsymbol{\alpha}^{k}, b^{k}\right)$. Note that this can happen quite often if the loss function is defined piecewise and is affine linear for a large range, cf. Observation 8.3.1. In this case, the update (8.4.13) implies that the $i$ th component of the optimality conditions is satisfied exactly at the next step because

$$
\alpha_{i}^{k+1}+g_{i}\left(\boldsymbol{\alpha}^{k+1}, b^{k+1}\right)=\alpha_{i}^{k+1}+g_{i}\left(\boldsymbol{\alpha}^{k}, b^{k}\right)=0 .
$$

This means especially that the $i$ th variables does not need any update in the next step of the method if $h_{i}\left(\boldsymbol{\alpha}^{k+1}, b^{k+1}\right)=0$. However, this condition can be expected to be satisfied very frequently for certain classes of loss functions.

Observation 8.4.3 (partial exactness of the Newton update)
Consider a standard training problem as in Definition 2.4.5 with twice differentiable loss functions.
Then, the ith variable is not updated in the subsequent step of Algorithm 8.4.2 if

- the step size of $\tau_{k}=1$ is accepted,
- $h_{i}\left(\boldsymbol{\alpha}^{k}, b^{k}\right)=h_{i}\left(\boldsymbol{\alpha}^{k+1}, b^{k+1}\right)=0$, and
- $g_{i}\left(\boldsymbol{\alpha}^{k+1}, b^{k+1}\right)=g_{i}\left(\boldsymbol{\alpha}^{k}, b^{k}\right)$.

It should be emphasized again that, as argued above, all the required conditions are not implausible in practical applications. In particular, loss functions derived by means of some smooth approximation of the maximum function (according to Section 4.4) are suitable to promote these conditions.

On the other hand, the conditions of Observation 8.4.3 are not very useful if a step is taken starting from a point far from the solution of the problem. This is because one cannot expect the values of the first and second derivative of the loss functions to stay constant when a significant update of the variables (and hence of the decision function values) is necessary. Thus, it is desirable to start from an appropriate approximation of the solution, which can be obtained, for instance, by means of the SMO method.

### 8.5 Properties of the Jacobian

In this section, we want to analyze the Jacobian of the function $\boldsymbol{F}$ defining system (6.5.1) in detail. By construction, the Jacobian of $\boldsymbol{F}$ is closely connected to the Newton systems (8.4.1) and (8.4.2). In particular, its structure is very similar to that of the system matrix which occurs in the reduced Newton system (8.4.8). Moreover, it is needed later on in Section 9.2 for the application of the implicit function theorem to obtain sensitivity information for the solution. The subsequent proposition summarizes basic observations about the structure of the Jacobian matrix and the associated Newton system.

Proposition 8.5 .1 (invertibility of the Jacobian matrix)
Consider a standard training problem as in Definition 2.4.5 with twice continuously differentiable loss functions. Let $\boldsymbol{F}$ be defined according to (6.5.1). Consider the Jacobian matrix

$$
\boldsymbol{F}^{\prime}(\boldsymbol{\alpha}, b)=\left(\begin{array}{cc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{H}(\boldsymbol{\alpha}, b) \boldsymbol{K} & \boldsymbol{H}(\boldsymbol{\alpha}, b) \mathbf{1} \\
\mathbf{1}^{\top} & 0
\end{array}\right)
$$

(a) The matrix $\boldsymbol{F}^{\prime}(\boldsymbol{\alpha}, b)$ is invertible if and only if $\mathbf{1}^{\top} \boldsymbol{h}(\boldsymbol{\alpha}, b)>0$.
(b) If $\boldsymbol{F}^{\prime}(\boldsymbol{\alpha}, b)$ is invertible, the solution of the system of linear equations

$$
\begin{equation*}
\boldsymbol{F}^{\prime}(\boldsymbol{\alpha}, b)\binom{\boldsymbol{u}}{u_{0}}=\binom{\boldsymbol{v}}{v_{0}} \tag{8.5.1}
\end{equation*}
$$

can be computed by means of Algorithm 8.5.1.
(c) Any solution of (8.5.1) satisfies $u_{i}=v_{i}$ for all $i \in\{1, \ldots, n\}$ with $h_{i}(\boldsymbol{\alpha}, b)=0$.

Proof. The basic idea of the proof follows the lines of Proposition 8.4.2. Let $J:=\{i \in$ $\left.\{1, \ldots, n\} \mid h_{i}(\boldsymbol{\alpha}, b)>0\right\}$ and $\bar{J}:=\{1, \ldots, n\} \backslash J$. Then, system (8.5.1) can be written in
blocks as

$$
\left(\begin{array}{ccc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{H}_{J J} \boldsymbol{K}_{J J} & \frac{1}{\lambda} \boldsymbol{H}_{J J} \boldsymbol{K}_{J \bar{J}} & \boldsymbol{H}_{J J} \mathbf{1} \\
\boldsymbol{O} & \boldsymbol{I} & \mathbf{0} \\
\mathbf{1}^{\top} & \mathbf{1}^{\top} & 0
\end{array}\right)\left(\begin{array}{c}
\boldsymbol{u}_{J} \\
\boldsymbol{u}_{\bar{J}} \\
u_{0}
\end{array}\right)=\left(\begin{array}{c}
\boldsymbol{v}_{J} \\
\boldsymbol{v}_{\bar{J}} \\
v_{0}
\end{array}\right) .
$$

Thus, it immediately follows that $\boldsymbol{u}_{\bar{J}}:=\boldsymbol{v}_{\bar{J}}$ is uniquely determined, i.e., assertion (c) is verified.

It remains to consider the reduced system

$$
\left(\begin{array}{cc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{H}_{J J} \boldsymbol{K}_{J J} & \boldsymbol{H}_{J J} \mathbf{1} \\
\mathbf{1}^{\top} & 0
\end{array}\right)\binom{\boldsymbol{u}_{J}}{u_{0}}=\binom{\boldsymbol{v}_{J}-\frac{1}{\lambda} \boldsymbol{H}_{J J} \boldsymbol{K}_{J \bar{J}} \boldsymbol{v}_{\bar{J}}}{v_{0}-\mathbf{1}^{\top} \boldsymbol{v}_{\bar{J}}},
$$

which can also be written as

$$
\left(\begin{array}{cc}
\boldsymbol{H}_{J J}^{-1}+\frac{1}{\lambda} \boldsymbol{K}_{J J} & \mathbf{1}  \tag{8.5.2}\\
\mathbf{1}^{\top} & 0
\end{array}\right)\binom{\boldsymbol{u}_{J}}{u_{0}}=\binom{\boldsymbol{H}_{J J}^{-1} \boldsymbol{v}_{J}-\frac{1}{\lambda} \boldsymbol{K}_{J \bar{J}} \boldsymbol{v}_{\bar{J}}}{v_{0}-\mathbf{1}^{\top} \boldsymbol{v}_{\bar{J}}}=:\binom{\boldsymbol{a}_{J}}{a_{0}}
$$

because the matrix $\boldsymbol{H}$ is positive definite.
Moreover, the matrix $\boldsymbol{Q}:=\boldsymbol{H}_{J J}^{-1}+\frac{1}{\lambda} \boldsymbol{K}_{J J}$ is also positive definite. This means that the solution of the remaining system can be obtained by solving two smaller systems of linear equations according to (8.5.3). Finally, computing the solution by means of (8.5.4) is possible. Hence, the matrix $\boldsymbol{F}^{\prime}(\boldsymbol{\alpha}, b)$ is invertible.

Note that the whole solution procedure is based on the existence of $\boldsymbol{H}_{J J}^{-1}$, i.e., on the fact that $J$ contains at least one index. This implies assertion (a).

```
Algorithm 8.5.1: Solution of the system (8.5.1)
    1 Let \(J:=\left\{i \in\{1, \ldots, n\} \mid h_{i}(\boldsymbol{\alpha}, b)>0\right\}\) and \(\bar{J}:=\{1, \ldots, n\} \backslash J\).
    2 Compute \(\boldsymbol{u}_{\bar{J}}:=\boldsymbol{v}_{\bar{J}}\).
    3 Compute \(\boldsymbol{a}_{J}:=\boldsymbol{H}_{J J}^{-1} \boldsymbol{v}_{J}-\frac{1}{\lambda} \boldsymbol{K}_{J J} \boldsymbol{v}_{J}\).
    4 Compute \(a_{0}:=v_{0}-\mathbf{1}^{\top} \boldsymbol{v}_{\bar{J}}\).
    5 With \(\boldsymbol{Q}:=\boldsymbol{H}_{J J}^{-1}+\frac{1}{\lambda} \boldsymbol{K}_{J J}\) compute
```

$$
\begin{equation*}
\boldsymbol{p}_{J}:=\boldsymbol{Q}^{-1} \boldsymbol{a}_{J} \quad \text { and } \quad \boldsymbol{q}_{J}:=\boldsymbol{Q}^{-1} \mathbf{1} \tag{8.5.3}
\end{equation*}
$$

6 Compute

$$
\begin{equation*}
u_{0}:=\frac{\mathbf{1}^{\top} \boldsymbol{p}_{J}-a_{0}}{\mathbf{1}^{\top} \boldsymbol{q}_{J}} \quad \text { and } \quad \boldsymbol{u}_{J}:=\boldsymbol{p}_{J}-u_{0} \boldsymbol{q}_{J} . \tag{8.5.4}
\end{equation*}
$$

One interesting observation concerning the solution of (8.5.1) using Algorithm 8.5.1 is that only the part of the kernel matrix associated with the index set $J$ is needed in the computations. That part can be seen as a kind of active set. We follow this argumentation later on in Subsection 9.2.2.

In the following, we also present an adapted solution method for a system of linear equations which has the transposed Jacobian as system matrix. Again, this will be helpful in Section 9.2.

Proposition 8.5.2 (solution of the system with transposed Jacobian matrix)
Under the assumptions of Proposition 8.5.1, the system with the transposed Jacobian

$$
\begin{equation*}
\boldsymbol{F}^{\prime}(\boldsymbol{\alpha}, b)^{\top}\binom{\boldsymbol{u}}{u_{0}}=\binom{\boldsymbol{v}}{v_{0}} \tag{8.5.5}
\end{equation*}
$$

can be solved by means of Algorithm 8.5.2 for all $(\boldsymbol{\alpha}, b)$ with $\mathbf{1}^{\top} \boldsymbol{h}(\boldsymbol{\alpha}, b)>0$.

Proof. With $J:=\left\{i \in\{1, \ldots, n\} \mid h_{i}(\boldsymbol{\alpha}, b)>0\right\}$ and $\bar{J}:=\{1, \ldots, n\} \backslash J$ the system (8.5.5) can be written in block form as

$$
\left(\begin{array}{ccc}
\boldsymbol{I}+\frac{1}{\lambda} \boldsymbol{K}_{J J} \boldsymbol{H}_{J J} & \boldsymbol{O} & \mathbf{1} \\
\frac{1}{\lambda} \boldsymbol{K}_{\bar{J} J} \boldsymbol{H}_{J J} & \boldsymbol{I} & \mathbf{1} \\
\mathbf{1}^{\top} \boldsymbol{H}_{J J} & \boldsymbol{0}^{\top} & 0
\end{array}\right)\left(\begin{array}{l}
\boldsymbol{u}_{J} \\
\boldsymbol{u}_{\bar{J}} \\
u_{0}
\end{array}\right)=\left(\begin{array}{c}
\boldsymbol{v}_{J} \\
\boldsymbol{v}_{\bar{J}} \\
v_{0}
\end{array}\right) .
$$

Motivated by the computations in the proof of the previous proposition, we consider the part

$$
\left(\begin{array}{cc}
\boldsymbol{H}_{J J}^{-1}+\frac{1}{\lambda} \boldsymbol{K}_{J J} & \mathbf{1} \\
\mathbf{1}^{\top} & 0
\end{array}\right)\binom{\boldsymbol{H}_{J J} \boldsymbol{u}_{J}}{u_{0}}=\binom{\boldsymbol{v}_{J}}{v_{0}}
$$

first and remember that $\boldsymbol{u}_{\bar{J}}=\boldsymbol{v}_{\bar{J}}-\frac{1}{\lambda} \boldsymbol{K}_{\bar{J} J} \boldsymbol{H}_{J J} \boldsymbol{u}_{J}-\mathbf{1} u_{0}$. This system has the same matrix as the system (8.5.2) in the previous proof. Hence, the same solution technique is applicable. We do not go into the details here and refer to Algorithm 8.5.2 for the remaining steps.

Algorithm 8.5.2: Solution of the system (8.5.5)
1 Let $J:=\left\{i \in\{1, \ldots, n\} \mid h_{i}(\boldsymbol{\alpha}, b)>0\right\}$ and $\bar{J}:=\{1, \ldots, n\} \backslash J$.
2 With $\boldsymbol{Q}:=\boldsymbol{H}_{J J}^{-1}+\frac{1}{\lambda} \boldsymbol{K}_{J J}$ compute

$$
\boldsymbol{p}_{J}:=\boldsymbol{Q}^{-1} \boldsymbol{v}_{J} \quad \text { and } \quad \boldsymbol{q}_{J}:=\boldsymbol{Q}^{-1} \mathbf{1}
$$

3 Compute

$$
u_{0}:=\frac{\mathbf{1}^{\top} \boldsymbol{p}_{J}-v_{0}}{\mathbf{1}^{\top} \boldsymbol{q}_{J}} \quad \text { and } \quad \boldsymbol{u}_{J}:=\boldsymbol{H}_{J J}^{-1}\left(\boldsymbol{p}_{J}-u_{0} \boldsymbol{q}_{J}\right) .
$$

4 Compute $\boldsymbol{u}_{\bar{J}}:=\boldsymbol{v}_{\bar{J}}-\frac{1}{\lambda} \boldsymbol{K}_{\bar{J} J}\left(\boldsymbol{p}_{J}-u_{0} \boldsymbol{q}_{J}\right)-\mathbf{1} u_{0}$.

## 9 Hyperparameter Optimization for Support Vector Machines


#### Abstract

In general, training problems depend on a set of hyperparameters. An obvious example for such a hyperparameter is the regularization parameter $\lambda$ in the definition of the standard training problem, see Definition 2.4.5. But there are many possible other sources of hyperparameters, which will be discussed later on in Subsection 9.1.3.


Clearly, the values of hyperparameters influence the characteristics and hence the performance of the resulting decision function. This implies immediately that a suitable choice for those values is crucial for practical applications, cf. [Bur98]. In a general context, the problem arising from this requirement is called model selection problem. If this problem is formulated as an optimization problem, it is also referred to as hyperparameter optimization problem.

The aim of the present chapter is to discuss the latter problem (Section 9.1) and to formulate a particular solution approach which is based on the formulation of sufficiently smooth training problems and the application of the implicit function theorem (Section 9.2). The proposed idea is closely based on the preliminary work in [FLL+15] but extends the setting to more general training problems. In particular, we are able to exploit the sparsity of the solutions of the training problems (which is naturally not present in the LS-SVM considered earlier) to make the approach applicable to even larger problems in practice.

### 9.1 Formulation of a Bilevel Optimization Problem

The problem of hyperparameter optimization occurs naturally after the definition of a particular training problem which depends on parameters that are fixed and not determined by the solution of the problem. In the following, we consider the formulation of the resulting problem in terms of a bilevel optimization problem. Before starting with this formulation, we summarize related approaches known from the literature.

### 9.1.1 Summary of Approaches in the Literature

A simple and well-established idea for the determination of suitable hyperparameter values is to apply a grid search procedure. In this approach, the training problem is solved for a fixed set of hyperparameter values (which are usually chosen to lie on a grid) and the resulting decision function is validated by computing the performance on some part of the dataset which was left out from the training. In a more elaborated approach, the performance of particular selections of hyperparameter values is measured by means of cross-validation, i.e., the training and validation is conducted for multiple splits of the given dataset.

Since the number of grid points scales exponentially with the number of hyperparameters, the grid search idea is only feasible for a rather small number of hyperparameters. Furthermore, this approach only considers discrete parameter values such that only a coarse selection is performed.

A general idea to reduce the practical effort of the grid search procedure is to apply some general purpose optimization routine that does not exploit the special structure of the underlying objective function. For instance, one could use a pattern search method as proposed in [MB02] or a genetic algorithm following [ÜMOB05]. Although these ideas may work for practical problems, an improvement could be expected if one takes the structural properties of the problem into account.

For instance, if classical Support Vector Machines are used for classification problems, particular performance estimates (or performance bounds) can be computed from the solution of the training problem, cf. [CV99, CVBM02]. Similar measures can also be computed for regression problems, see [CL05]. These measures can then be used to determine suitable values for the hyperparameters and the overall approach is shown to yield good results in practice. Unfortunately, there is no guarantee that the used performance bounds correspond to the true performance of the generated decision function. Moreover, it cannot be assumed that the resulting objective function which is considered for the optimization is differentiable. Due to these possible difficulties and the restriction to a particular problem structure, we do not consider this approach subsequently.

One of the first approaches that uses the notion of bilevel optimization explicitly to model the hyperparameter optimization problem is given in [BHJ+06, KBHPO8] and was extended in [MBB11]. The authors propose to consider a bilevel optimization problem in which the upper level problem has an objective function measuring the validation error while the lower level problem consists of one or more training problems.

We also discuss and formalize this basic idea later on in Subsection 9.1.2. In the previously mentioned papers, the bilevel optimization problem is transformed into a nonlinear optimization problem by replacing the lower level problem with its optimality conditions. The resulting problem is treated practically by means of general-purpose optimization tools. Due to the reformulation, a lot of the problems' structure is lost such that the proposed method is restricted to relatively small datasets. See also [LLZ22] for a more recent approach using a similar idea.

Instead of considering the bilevel optimization problem in its entirety, it is also possible to follow an implicit function approach. To use this idea, it is usually assumed that the solution of the lower level problem is unique and that there is some optimality condition which can be used to compute derivatives of the solution with respect to the hyperparameters. This has the advantage that the size of the overall problem is tremendously reduced compared to the bilevel reformulation approach because the lower and upper level problems are treated alternately and not in parallel.

Some authors propose to consider optimality which yield differentiability for almost all values of hyperparameters, see, for instance, [Sch05, KSC06, JS20]. In other approaches, the loss term in the training problem is chosen in a way that differentiability is satisfied without further assumptions, see [DFN07] for log-linear models and [FLL+15] for Least Squares Support Vector Regression. Subsequently, we follow the same basic idea and argue for the use of smoothed training problems for which it can be shown that the solution is differentiable with respect to hyperparameters.

### 9.1.2 Formulation of the General Idea

In order to derive a bilevel optimization model for the hyperparameter selection problem, we consider each part of the problem sequentially. We start with the lower level problem consisting of one or more training problems. To keep the notation simple, only a single problem is considered initially and a possible extension to the setting of multiple independent training problems is discussed later on in Subsection 9.1.4.

Let $\boldsymbol{\vartheta} \in \Theta$ be a vector of hyperparameters which are present in the training problem. Then, a parameter-dependent training problem following the original structure of the general training problem (2.4.1) can be formulated as

$$
\begin{equation*}
\min _{\boldsymbol{\omega} \in \mathcal{H}} \mathcal{R}(\boldsymbol{\omega}, \boldsymbol{\vartheta})+\mathcal{L}(\boldsymbol{T} \boldsymbol{\omega}, \boldsymbol{\vartheta}) \tag{9.1.1}
\end{equation*}
$$

where $\mathcal{R}(\boldsymbol{\omega}, \boldsymbol{\vartheta})$ and $\mathcal{L}(\boldsymbol{T} \boldsymbol{\omega}, \boldsymbol{\vartheta})$ denote the parameter-dependent regularization and loss terms, respectively. Note that the value of $\boldsymbol{\vartheta}$ is fixed for the training problem and determines the characteristics of the problem. In particular, this means that a different choice of $\boldsymbol{\vartheta}$ will in general result in a different solution, which leads to a different decision function. For short, we let

$$
\mathcal{Z}(\boldsymbol{\vartheta}):=\{\boldsymbol{\omega} \in \mathcal{H} \mid \boldsymbol{\omega} \text { solves (9.1.1) with parameter } \boldsymbol{\vartheta}\}
$$

denote the solution set of the training problem for $\boldsymbol{\vartheta} \in \Theta$. Then, problem (9.1.1) constitutes the lower level problem of the bilevel problem which is introduced in the following.

In order to define the upper level problem, we let $E: \mathcal{H} \times \Theta \rightarrow \mathbb{R}$ be a function measuring the performance of a vector $\boldsymbol{\omega}$ for a fixed parameter $\boldsymbol{\vartheta} \in \Theta$. Since a vector $\boldsymbol{\omega} \in \mathcal{H}$ is usually identified with a particular decision function, the value $E(\boldsymbol{\omega}, \boldsymbol{\vartheta})$ can also be interpreted as the performance of the underlying decision function. In the most common formulations of the hyperparameter optimization problem, the function $E$ is composed of a particular error measure evaluated by means of a dataset
which is not used in the training problem, such that it represents a validation error. Of course, it is also possible to consider certain other performance estimates as discussed in the previous subsection. The performance measure $E$ is then used as objective function in the upper level problem as follows.

Taking together the training problem defined in (9.1.1) and the performance measure given by $E$, the hyperparameter optimization problem can be stated as

$$
\begin{equation*}
\min _{\substack{\boldsymbol{\omega} \in \mathcal{H} \\ \boldsymbol{\vartheta} \in \Theta}} E(\boldsymbol{\omega}, \boldsymbol{\vartheta}) \quad \text { s.t. } \quad \boldsymbol{\omega} \in \mathcal{Z}(\boldsymbol{\vartheta}) \text {. } \tag{9.1.2}
\end{equation*}
$$

If the solution set of the lower level problem is not unique, one commonly distinguishes between two different variants of the bilevel optimization problem. In this setting, the problem (9.1.2) is an example for a bilevel optimization problem in an optimistic formulation. Here, optimistic means that the solution of the lower level problem can be chosen in favor of the upper level objective. In contrast, a pessimistic formulation of the hyperparameter optimization problem could be stated as

$$
\min _{\boldsymbol{\vartheta} \in \Theta} \max _{\boldsymbol{\omega} \in \mathcal{Z}(\boldsymbol{\vartheta})} E(\boldsymbol{\omega}, \boldsymbol{\vartheta})
$$

In this formulation the solution of the lower level problem is chosen against the upper level objective. Since we are targeting a setting in which the solution of the lower level problem is unique for each hyperparameter choice, we do not discuss the implications of the two different formulations further and use (9.1.2) as a basis for the subsequent considerations.

### 9.1.3 Parameter-Dependent Standard Training Problem

Of course, it is possible to consider the hyperparameter optimization problem in its general form as stated above. However, in order to make the problem better understandable and easier to apply in practice, we restrict our further investigation to standard training problems in the sense of Definition 2.4.5. Recall that the definition of a standard training problem is generally sufficient to formalize training problems originating from classification and regression problems. Thus, in the following, we consider a parameter-dependent standard training problem of the form

$$
\begin{equation*}
\min _{\substack{\boldsymbol{w} \in \mathcal{F} \\ b \in \mathbb{R}}} \frac{\lambda(\boldsymbol{\vartheta})}{2}\|\boldsymbol{w}\|^{2}+\sum_{i \in \mathcal{N}_{\mathrm{tr}}} \ell_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta})\right\rangle+b, \boldsymbol{\vartheta}\right) . \tag{9.1.3}
\end{equation*}
$$

Note that we used an index set $\mathcal{N}_{\text {tr }}$ instead of the full set $\mathcal{N}:=\{1, \ldots, n\}$ in the definition of the training problem (9.1.3). This notation simplifies the description of the upper level problem in the following.

In the previous training problem the particular choice of the hyperparameter vector $\boldsymbol{\vartheta}$ determines
(a) the value of the regularization parameter $\lambda$,
(b) the values of the feature vectors $\boldsymbol{\varphi}_{i}$
(or equivalently the values in the kernel matrix $\boldsymbol{K}$ ), and
(c) the shape of the loss functions.

Depending on the application, not all of these dependencies are necessarily present in a particular training problem. The presence of the hyperparameter $\lambda$ occurs quite naturally through the definition of the standard training problem. Moreover, the dependency of the feature vectors on hyperparameters can be easily motivated by means of the kernel trick, cf. Subsection 2.2.4.

Example 9.1.1 (kernel parameters and Gaussian kernel function)
Suppose that the feature vectors $\boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta})$ are defined by the transformation of given input points $\boldsymbol{x}_{1}, \ldots, \boldsymbol{x}_{n} \in \mathcal{X}$ under a parameter-dependent feature map $\varphi: \mathcal{X} \times \Theta \rightarrow \mathcal{F}$, i.e., $\boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta}):=\boldsymbol{\varphi}\left(\boldsymbol{x}_{i}, \boldsymbol{\vartheta}\right)$ for $i \in\{1, \ldots, n\}$. Then, such a feature map can also be defined implicitly by

$$
\left\langle\boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta}), \boldsymbol{\varphi}_{j}(\boldsymbol{\vartheta})\right\rangle=\kappa\left(\boldsymbol{x}_{i}, \boldsymbol{x}_{j}, \boldsymbol{\vartheta}\right)
$$

with a parameter-dependent kernel function $\kappa: \mathcal{X} \times \mathcal{X} \times \Theta \rightarrow \mathbb{R}$ having certain properties, see [Bis06, Chapter 6] and [SS02, Chapter 13].
A prominent example for a kernel function is the Gaussian kernel function defined by

$$
\kappa(\boldsymbol{x}, \tilde{\boldsymbol{x}}, \boldsymbol{\vartheta}):=\exp \left(-\gamma\|\boldsymbol{x}-\tilde{\boldsymbol{x}}\|^{2}\right)
$$

for real input vectors $\boldsymbol{x}, \tilde{\boldsymbol{x}} \in \mathbb{R}^{d}$ with a scaling parameter $\gamma>0$, which is generally also part of the vector $\boldsymbol{\vartheta}$. In some applications, the structure of the Gaussian kernel function is used to derive a kernel function with more degrees of freedom. For instance, a common approach is to consider a function of the form

$$
\kappa(\boldsymbol{x}, \tilde{\boldsymbol{x}}, \boldsymbol{\vartheta}):=\exp \left(-\left\|\boldsymbol{\Gamma}(\boldsymbol{\vartheta})\left(\boldsymbol{x}_{i}-\boldsymbol{x}_{j}\right)\right\|^{2}\right)
$$

with a parameter-dependent matrix $\boldsymbol{\Gamma}(\boldsymbol{\vartheta}) \in \mathbb{R}^{d^{\prime} \times d}$. Practically, the latter matrix is often chosen as diagonal matrix which scales each component of the input vectors $\boldsymbol{x}_{i}$ independently.

The third group of hyperparameters is of those that determine the shape of the loss functions or the weighting of training samples (or classes in classification problems). Those hyperparameters are called loss parameters. One obvious example is the parameter $\varepsilon$ in the $\varepsilon$-Support Vector Regression problem (4.2.1). Going a bit further, the parameter $\nu$ in the general $\nu$-Support Vector Machine (5.2.4) is another example of this type, which, however, does not fit into the setting of standard training problems.

For each solution of the standard training problem (9.1.3) it is possible to define the resulting decision function according to Definition 2.4.3. Because the feature vectors are supposed to be dependent on $\boldsymbol{\vartheta}$, the definition has to be modified slightly. For this reason, we consider a parameter dependent feature vector $\boldsymbol{\varphi}(\boldsymbol{\vartheta}) \in \mathcal{F}$ and the
parameter-dependent decision function defined by

$$
d(\boldsymbol{\varphi}(\boldsymbol{\vartheta})):=\langle\boldsymbol{w}, \boldsymbol{\varphi}(\boldsymbol{\vartheta})\rangle+b,
$$

which also depends on the particular choice of the solution $(\boldsymbol{w}, b) \in \mathcal{F} \times \mathbb{R}$ of the training problem (9.1.3).

### 9.1.4 Formulation of the Upper Level Problem

In order to formulate the hyperparameter optimization problem concretely, we consider an approach for which a given dataset is split into parts that are used for training and performance estimation, respectively.

## Performance Estimation from a Validation Set

In the simplest setting, the dataset is split once into two parts identified by the index sets $\mathcal{N}_{\text {tr }}$ and $\mathcal{N}_{\text {val }}$. For each element $i \in \mathcal{N}_{\text {val }}$ a particular error measure $L_{i}: \mathbb{R} \times \boldsymbol{\vartheta} \rightarrow \mathbb{R}$ is used to define the overall performance measure by

$$
E((\boldsymbol{w}, b), \boldsymbol{\vartheta}):=\sum_{i \in \mathcal{N}_{\text {val }}} L_{i}\left(\left\langle\boldsymbol{w}, \boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta})\right\rangle+b, \boldsymbol{\vartheta}\right) .
$$

Then, the resulting hyperparameter optimization problem is stated as

$$
\begin{equation*}
\min _{\boldsymbol{\vartheta} \in \Theta} E((\boldsymbol{w}, b), \boldsymbol{\vartheta}) \tag{9.1.4}
\end{equation*}
$$

The problem associated with this simple approach is that the given dataset is not exploited efficiently in general. On the one hand, the size of the dataset that is used for the training procedure is made smaller. This implies that the generated decision function is usually less precise. On the other hand, the performance measure which is computed from the validation set may not represent the true performance appropriately. This is especially the case if the validation set is small or chosen in an unsuitable way. To circumvent these issues, a common approach is the application of a crossvalidation as discussed next.

## Performance Estimation using Cross-Validation

The basic idea of a cross-validation procedure is to split the given dataset into two or more parts. We refer to the partition by means of the index sets $\mathcal{N}_{1} \cup \cdots \cup \mathcal{N}_{Q}=\mathcal{N}$. For each index set the corresponding part is left out from a training problem and the resulting decision function is evaluated by means of that part. Formally, the resulting hyperparameter optimization problem can be written as

$$
\begin{equation*}
\min _{\boldsymbol{\vartheta} \in \Theta} \sum_{q=1}^{Q} \sum_{i \in \mathcal{N}_{q}} L_{i}\left(\left\langle\boldsymbol{w}^{q}, \boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta})\right\rangle+b^{q}, \boldsymbol{\vartheta}\right) \tag{9.1.5}
\end{equation*}
$$

s.t. $\quad\left(\boldsymbol{w}^{q}, b^{q}\right)$ solves (9.1.3) with $\mathcal{N}_{\text {tr }}:=\mathcal{N} \backslash \mathcal{N}_{q}$ for $q \in\{1, \ldots, Q\}$.

If the considered partition is disjoint, each element of the dataset is used in exactly one term of the objective function and in all except for one of the training problems. This means that the whole dataset is used more efficiently than in the case of a single validation set. Hence, one could expect to obtain better performance estimates for the optimization procedure. On the other hand, it should be emphasized that the effort which is needed to evaluate the performance grows (approximately linearly) with the number of parts in the partition. Thus, it is necessary to make a compromise between the quality of the estimates and the computational effort in practice.

At first sight, it is not obvious that this problem also matches the structure of the general hyperparameter optimization problem introduced in (9.1.2). However, if all training problems are accumulated into a single one of the form

$$
\min _{\boldsymbol{\omega}} \sum_{q=1}^{Q}\left(\frac{\lambda(\boldsymbol{\vartheta})}{2}\left\|\boldsymbol{w}^{q}\right\|^{2}+\sum_{i \in \mathcal{N} \backslash \mathcal{N}_{q}} \ell_{i}\left(\left\langle\boldsymbol{w}^{q}, \boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta})\right\rangle+b^{q}, \boldsymbol{\vartheta}\right)\right)
$$

with variables $\boldsymbol{\omega}:=\left(\left(\boldsymbol{w}^{1}, b^{1}\right), \ldots,\left(\boldsymbol{w}^{Q}, b^{Q}\right)\right) \in \mathcal{H}:=(\mathcal{F} \times \mathbb{R})^{Q}$, and the upper level objective function is defined by

$$
E(\boldsymbol{\omega}, \boldsymbol{\vartheta}):=\sum_{q=1}^{Q} \sum_{i \in \mathcal{N}_{q}} L_{i}\left(\left\langle\boldsymbol{w}^{q}, \boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta})\right\rangle+b^{q}, \boldsymbol{\vartheta}\right),
$$

this is indeed the case. Of course, a rewriting of this manner withdraws the structure of the lower level which actually consists of $Q$ independent optimization problems. This means that it is generally not desirable to use this formulation as a starting point for practical applications, and the explicit problem formulation in (9.1.5) should be preferred.

### 9.2 An Implicit Function Approach

The aim of this section is the application of the implicit function theorem in order to obtain sensitivity information for the solution of the training problem and, particularly, for the resulting decision function. In view of Proposition 6.5.3, we consider the system of nonlinear equations

$$
\begin{equation*}
\boldsymbol{F}(\boldsymbol{\alpha}, b ; \boldsymbol{\vartheta}):=\binom{\boldsymbol{\alpha}+\boldsymbol{g}(\boldsymbol{\alpha}, b ; \boldsymbol{\vartheta})}{\mathbf{1}^{\top} \boldsymbol{\alpha}}=\binom{\mathbf{0}}{0}, \tag{9.2.1}
\end{equation*}
$$

where $\boldsymbol{g}(\boldsymbol{\alpha}, b ; \boldsymbol{\vartheta})$ is defined in the same way as in (6.1.5) but with an additional dependency on the hyperparameter vector $\boldsymbol{\vartheta}$. Recall that, this system is equivalent to the training problem (9.1.3) if the solution of this problem is unique due to Proposition 6.5.3. Moreover, the decision function resulting from the solution of the training problem can easily be computed from the solution of (9.2.1).

Now, the question is under which assumptions a derivative of the solution of the system (9.2.1) with respect to the hyperparameters is available and how it can be computed practically. In order to make the following steps possible, we require a stronger condition than Assumption 3.1.2 to be satisfied.

Assumption 9.2.1 (unique solution of the training problem)
Consider a parameter dependent standard training problem (9.1.3) with twice continuously differentiable loss functions. Let problem (9.1.3) have a unique solution ( $\boldsymbol{w}^{\star}, b^{\star}$ )
and suppose that

$$
\begin{equation*}
\sum_{i \in \mathcal{N}_{\mathrm{tr}}} \ell_{i}^{\prime \prime}\left(\left\langle\boldsymbol{w}^{\star}, \boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta})\right\rangle+b^{\star}, \boldsymbol{\vartheta}\right)>0 \tag{9.2.2}
\end{equation*}
$$

is satisfied for all $\boldsymbol{\vartheta} \in \Theta$.

Note that for problems with twice differentiable loss functions, the condition (9.2.2) is virtually always satisfied. Additionally, it is possible to add an artificial loss term which is uniformly convex in order to force this condition theoretically. This could be, for instance, a quadratic penalty for the bias multiplied with a sufficiently small factor such that the original problem is not perturbed too much.

### 9.2.1 Application of the Implicit Function Theorem

If Assumption 9.2.1 is satisfied, the solution of the system (9.2.1) is unique for each value of $\boldsymbol{\vartheta} \in \Theta$. In this case, let $\boldsymbol{z}: \Theta \rightarrow \mathbb{R}^{\mathcal{N}_{\text {tr }}} \times \mathbb{R}$ with $\boldsymbol{z}(\boldsymbol{\vartheta})=(\boldsymbol{\alpha}(\boldsymbol{\vartheta}), b(\boldsymbol{\vartheta}))$ denote the unique solution of that system.

Using the solution mapping $\boldsymbol{z}$, it is possible to reformulate the hyperparameter optimization problem (9.1.4) equivalently as

$$
\begin{equation*}
\min _{\boldsymbol{\vartheta} \in \Theta} \sum_{i \in \mathcal{N}_{\text {val }}} E_{\mathrm{f}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}) \tag{9.2.3}
\end{equation*}
$$

with the objective function

$$
E_{\mathrm{f}}(\boldsymbol{z}, \boldsymbol{\vartheta}):=E((\boldsymbol{\Phi}(\boldsymbol{\vartheta}) \boldsymbol{\alpha}, b), \boldsymbol{\vartheta}),
$$

where $\boldsymbol{w}=\boldsymbol{\Phi}(\boldsymbol{\vartheta}) \boldsymbol{\alpha}$ and the linear operator $\boldsymbol{\Phi}(\boldsymbol{\vartheta})$ is defined by

$$
\boldsymbol{\Phi}(\boldsymbol{\vartheta}) \boldsymbol{\alpha}:=\frac{1}{\lambda(\boldsymbol{\vartheta})} \sum_{i \in \mathcal{N}_{\mathrm{tr}}} \alpha_{i} \boldsymbol{\varphi}_{i}(\boldsymbol{\vartheta}) .
$$

Note that the latter representation of $\boldsymbol{w}$ is introduced in Corollary 3.5.1 and formally defined in (6.2.1). Here, it is used in the appropriate parameter-dependent formulation. Observe that the same reformulation of the bilevel optimization problem can be used to transform the hyperparameter optimization problem with cross-validation which was introduced in (9.1.5).

Now, the resulting problem (9.2.3) is a nonlinear optimization problem, which depends only on the hyperparameter vector $\boldsymbol{\vartheta}$, whereas the variables of the former lower level problem are only implicitly a part of the problem. Note that the structure of the set $\Theta$ can include certain constraints on the values of $\boldsymbol{\vartheta}$ such that the characteristics of the problem (9.2.3) depends on the choice of $\Theta$.

As mentioned above, due to Proposition 6.5.3 the solution of the finite-dimensional standard training problem associated with (9.1.3) is then uniquely characterized by the equation

$$
\begin{equation*}
\boldsymbol{F}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})=\mathbf{0} \tag{9.2.4}
\end{equation*}
$$

for each fixed hyperparameter $\boldsymbol{\vartheta} \in \Theta$. For the function $\boldsymbol{F}$ defining the system, we let

$$
\boldsymbol{F}^{\prime}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}):=\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{z}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})
$$

denote the partial derivative with respect to the training variables in accordance to the notation of Section 6.5. Then, the implicit function theorem can be used to compute sensitivity information of the solution of (9.2.4) as follows.

Proposition 9.2.2 (derivative of the decision function w.r.t. hyperparameters) Suppose that Assumption 9.2.1 holds.
Then,

$$
\begin{equation*}
\boldsymbol{F}^{\prime}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}) \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{\vartheta})+\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})=\mathbf{0} \tag{9.2.5}
\end{equation*}
$$

i.e.,

$$
\begin{equation*}
\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{\vartheta})=-\left(\boldsymbol{F}^{\prime}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})\right)^{-1} \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}), \tag{9.2.6}
\end{equation*}
$$

With the help of the solution mapping $\boldsymbol{z}$ (which is implicitly defined by the training problem) and its derivative, that is given by (9.2.6), the reduced hyperparameter optimization problem (9.2.3) can be treated by means of derivative-based optimization methods.

This is the fundamental idea of the proposed hyperparameter optimization method. To show that the computation of the derivatives from (9.2.6) can be performed efficiently, we consider some particular aspects subsequently.

### 9.2.2 Sensitivity Analysis for Particular Training Samples

First, we consider the system (9.2.4) together with the derivatives (9.2.6) and Algorithm 8.5.1. Recall that for the realization of Newton's method discussed in Subsection 8.4.4 we found that there are some variables with vanishing entries in the Newton direction. In the same way, we can observe that the derivative of certain variables also vanishes as follows.

Proposition 9.2.3 (sensitivity of particular coefficients)
Suppose that Assumption 9.2.1 holds and let the loss functions be independent of the hyperparameter vector $\boldsymbol{\vartheta}$. Consider a fixed hyperparameter vector $\boldsymbol{\vartheta} \in \Theta$.
Then,

$$
\begin{equation*}
\frac{\partial \alpha_{i}(\boldsymbol{\vartheta})}{\partial \boldsymbol{\vartheta}}=0 \quad \text { for all } \quad i \in \mathcal{N}_{\operatorname{tr}} \text { with } h_{i}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})=0 \tag{9.2.7}
\end{equation*}
$$

Proof. A simple computation shows that

$$
\begin{aligned}
\frac{\partial g_{i}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}, \boldsymbol{\vartheta}) & =\ell_{i}^{\prime \prime}\left(\frac{1}{\lambda(\boldsymbol{\vartheta})}[\boldsymbol{K}(\boldsymbol{\vartheta}) \boldsymbol{\alpha}]_{i}+b, \boldsymbol{\vartheta}\right) \frac{\partial}{\partial \boldsymbol{\vartheta}}\left(\frac{1}{\lambda(\boldsymbol{\vartheta})}[\boldsymbol{K}(\boldsymbol{\vartheta}) \boldsymbol{\alpha}]_{i}\right) \\
& =h_{i}(\boldsymbol{z}, \boldsymbol{\vartheta}) \frac{\partial}{\partial \boldsymbol{\vartheta}}\left(\frac{1}{\lambda(\boldsymbol{\vartheta})}[\boldsymbol{K}(\boldsymbol{\vartheta}) \boldsymbol{\alpha}]_{i}\right)
\end{aligned}
$$

for $i \in \mathcal{N}_{\text {tr }}$. This means that the vector

$$
\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}, \boldsymbol{\vartheta})=\binom{\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}, \boldsymbol{\vartheta})}{\mathbf{0}^{\top}}
$$

occurring in the system (9.2.5) contains zero entries for all $i \in \mathcal{N}_{\text {tr }}$ with $h_{i}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})=0$. Together with (9.2.5) and Proposition 8.5.1 (c) this directly implies (9.2.7).

Note that the variables which are considered in the proposition play also an important role in the application of Newton's method. In particular, we have seen in Observation 8.4.3 that those variables are potentially determined exactly by the method. Moreover, if a smooth approximation of the maximum function is used in the definition of the training problem, those variables can easily be identified with dual variables that lie strictly between their bounds.

This observation also shows a close connection to other hyperparameter optimization approaches, which use dual optimality conditions for the derivation of sensitivity such as [Sch05] and [KSC06].

### 9.2.3 Derivatives of the Validation Error Measure

If we consider the reduced problem (9.2.3) together with the derivative of the solution mapping $\boldsymbol{z}$ given by Proposition 9.2.2, we are of course able to compute the derivative of the validation error measure with respect to the hyperparameters. For the practical computation in a straightforward manner, according to (9.2.5), a single linear system of the form

$$
\boldsymbol{F}^{\prime}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}) \frac{\partial \boldsymbol{z}}{\partial \vartheta_{p}}(\boldsymbol{\vartheta})+\frac{\partial \boldsymbol{F}}{\partial \vartheta_{p}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})=\mathbf{0}
$$

has to be solved for each component $\vartheta_{p}$ of the hyperparameter vector $\boldsymbol{\vartheta}$.
In the following proposition, we summarize an approach which reduces the effort to the solution of a single system of linear equations for all hyperparameters using an elementary transformation.

Proposition 9.2.4 (efficient computation of derivatives of the validation error) Suppose that Assumption 9.2.1 holds. Consider a fixed hyperparameter vector $\boldsymbol{\vartheta} \in \Theta$. Define

$$
\begin{equation*}
\boldsymbol{\gamma}:=\left(\boldsymbol{F}^{\prime}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})^{\top}\right)^{-1}\left(\frac{\partial E_{\mathrm{f}}}{\partial \boldsymbol{z}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})\right)^{\top} . \tag{9.2.8}
\end{equation*}
$$

Then, the derivative of the validation error with respect to the hyperparameter vector is given by

$$
\begin{equation*}
\frac{\partial}{\partial \boldsymbol{\vartheta}} E_{\mathrm{f}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})=\frac{\partial E_{\mathrm{f}}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})-\boldsymbol{\gamma}^{\top} \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}) \tag{9.2.9}
\end{equation*}
$$

Proof. Due to the chain rule and by means of (9.2.6) it follows that

$$
\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\vartheta}} E_{\mathrm{f}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}) & =\frac{\partial E_{\mathrm{f}}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})+\frac{\partial E_{\mathrm{f}}}{\partial \boldsymbol{z}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}) \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{\vartheta}) \\
& =\frac{\partial E_{\mathrm{f}}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})-\frac{\partial E_{\mathrm{f}}}{\partial \boldsymbol{z}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}) \boldsymbol{F}^{\prime}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta})^{-1} \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\vartheta}}(\boldsymbol{z}(\boldsymbol{\vartheta}), \boldsymbol{\vartheta}) .
\end{aligned}
$$

Thus, it is easy to see that (9.2.9) holds with the definition of $\boldsymbol{\gamma}$ in (9.2.8).
Note that the system of linear equations (9.2.8) can be solved efficiently by means of Algorithm 8.5.2. This shows that the effort for computing the derivatives of the objective function of the problem (9.2.3) can be reduced significantly if the number of hyperparameters is high. For a practical investigation of the presented hyperparameter optimization approach we refer to Section 10.4 below.

## 10 Computational Study

In the present chapter, we consider selected methods from the previous chapters in practical applications. After the introduction of an artificial training problem (Section 10.1), we consider the two basic approaches for the solution of the training problem, namely the dual SMO method investigated in Chapter 7 (Section 10.2) and the primal training methods of Chapter 8 (Section 10.3). Finally, we apply the hyperparameter optimization idea discussed in Chapter 9 to some particular practical problems (Section 10.4).

### 10.1 Definition of Dataset and Training Problem

To get an idea of the behavior of the considered methods in practice, we define a particular artificial dataset stating a classification problem which is then used repeatedly. For this purpose, we start with a set of $n$ points $\left\{\boldsymbol{x}_{i}\right\}_{i=1}^{n} \subseteq \mathbb{R}^{2}$, which are sampled from a two-dimensional standard uniform distribution, i.e., a uniform distribution on the square $[0,1]^{2}$. Unless otherwise noted, a dataset of size $n=1,000$ is considered. In order to create a classification problem, each point $\boldsymbol{x}_{i}$ is associated with a label $y_{i} \in\{-1,+1\}$ defined by

$$
y_{i}:= \begin{cases}+1, & \text { if } f\left(\boldsymbol{x}_{i}\right) \geq 0 \\ -1, & \text { if } f\left(\boldsymbol{x}_{i}\right)<0\end{cases}
$$

where $f(\boldsymbol{x}):=\sin \left(4 \pi[\boldsymbol{x}]_{1}\right) \cdot \sin \left(4 \pi[\boldsymbol{x}]_{2}\right)$. With this definition, the dataset forms a checkerboard pattern, which is visualized in Figure 10.1.1, where elements with $y_{i}=1$ are colored in red and elements with $y_{i}=-1$ are blue. Note that a similar dataset was also used in [MM01].

For this dataset, we consider the classical Support Vector Classification problem as introduced in Proposition 4.1.1 with uniform weighting, i.e., $c_{i}=1$ for all $i \in\{1, \ldots, n\}$. With each point $\boldsymbol{x}_{i}$ a feature vector $\boldsymbol{\varphi}_{i} \in \mathcal{F}$ is associated implicitly such that the inner products are the values of the Gaussian kernel function, namely

$$
\left\langle\boldsymbol{\varphi}_{i}, \boldsymbol{\varphi}_{j}\right\rangle=\kappa\left(\boldsymbol{x}_{i}, \boldsymbol{x}_{j}\right)=\exp \left(-\gamma\left\|\boldsymbol{x}_{i}-\boldsymbol{x}_{j}\right\|^{2}\right)
$$

with some scaling parameter $\gamma>0$.


Figure 10.1.1: Training dataset with $n=1,000$ randomly chosen elements following a checkerboard pattern

Depending on the choice of the regularization parameter $\lambda$ and the scaling parameter $\gamma$, different forms of decision functions are realized by the solution of the training problem. In order to observe a representative behavior of the training method, we first identify a set of parameters, which results in a decision function which seems to be reasonable. Here, the evaluation of a particular decision function is not conducted rigorously but with a sense of proportion.

A short trial of different parameter settings showed that choosing $\lambda=10^{-3}$ and $\gamma=30$ yields suitable results. Note that values in a similar range are also obtained from the hyperparameter optimization approach which will be conducted in Section 10.4. In order to support the parameter choice further, the resulting decision functions for different values of the hyperparameters are presented in Figure 10.1.2. Moreover, these differently chosen parameter values are also the basis for the practical investigation of the SMO method below.

In the visualization of the decision functions in Figure 10.1.2 one can observe the well-known behavior that

- small values for the regularization parameter $\lambda$ lead to an overfitting to the underlying dataset, whereas large values result in underfitting, and
- a small value for the scaling parameter $\gamma$ produces smoother decision boundaries (underfitting), whereas a large value leads to rather jagged ones (overfitting).

Another fact about the parameter choice that should be emphasized here (and which is also well-known from practical applications) is that it strongly determines the runtime of the solution method. In particular a choice of a small regularization parameter $\lambda$ (which corresponds to a large value of $C$ in the classical SVM formulation) leads to an increased amount of necessary steps in the optimization procedure. On the other hand, the problem gets computationally easier for large values of the scaling parameter $\gamma$.


Figure 10.1.2: Generated decision boundaries (i.e., zero level sets of the generated decision functions) for different parameter settings

### 10.2 Sequential Minimal Optimization

In this section, we want to substantiate the previously derived theoretical results by means of a particular training problem. For this purpose, we implemented and applied a prototypical algorithm using the Python programming language. The computations were conducted on an AMD Ryzen 9 5900X processor with 64 GB of memory under Python 3.10.8 and NumPy 1.23.5, cf. [HMvdW+20].

Subsequently, different quantities are reported for each step of the optimization method. For instance, we study the values of optimality measures for a given point $\boldsymbol{\alpha}^{k}$, namely

- the violation measure $r\left(\boldsymbol{\alpha}^{k}\right)$ defined in Observation 7.2.2,
- the primal-dual gap $\Delta^{\star}\left(\boldsymbol{\alpha}^{k}\right)$ defined in Observation 7.2.4
- the bound-aware violation measure $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \rho\right)$ with maximal step size $\rho:=\lambda$ defined in Definition 7.6.1,
- the distance of the current point to the solution $D_{k}:=\left\|\boldsymbol{\alpha}^{k}-\boldsymbol{\alpha}^{\star}\right\|$, and,
- the dual objective gap $V_{k}:=\psi_{\mathrm{d}}\left(\boldsymbol{\alpha}^{k}\right)-\psi_{\mathrm{d}}\left(\boldsymbol{\alpha}^{\star}\right)$.

Note that a maximal step size of $\rho=\lambda$ is used for the evaluation of the bound-aware violation measure because the step size is expected to scale with the value $\lambda$, which will be observed later on empirically. Moreover, the solution of the dual training problem is unique for the considered problem such that the value of $D_{k}$ is well-defined.

Additionally, we consider

- the distance between two subsequent points

$$
\tilde{\tau}_{k}:=\left\|\boldsymbol{\alpha}^{k+1}-\boldsymbol{\alpha}^{k}\right\| / \sqrt{2}=\tau_{k}\left\|\boldsymbol{d}^{k}\right\| / \sqrt{2}
$$

- the decrease of the objective function value

$$
\Delta \psi_{\mathrm{d}}^{k}:=\psi_{\mathrm{d}}\left(\boldsymbol{\alpha}^{k+1}\right)-\psi_{\mathrm{d}}\left(\boldsymbol{\alpha}^{k}\right),
$$

and

- the value of the normalized directional derivative

$$
\tilde{r}_{k}:=\sqrt{2} r_{k} /\left\|\boldsymbol{d}^{k}\right\|
$$

with $r_{k}:=-\nabla \psi_{\mathrm{d}, \mathrm{ext}}\left(\boldsymbol{\alpha}^{k}\right)^{\top} \boldsymbol{d}^{k}$ according to (7.4.7).
The latter term plays a major role in the proposed convergence proof of the SMO method. For instance, the step length $\tilde{\tau}_{k}$ is expected to be proportional to $\tilde{r}_{k}$ according to (7.9.2), whereas the decrease is asymptotically in the order of $\tilde{r}_{k}^{2}$ due to (7.9.3). The factor $\sqrt{2}$ in the definition of $\tilde{\tau}_{k}$ and $\tilde{r}_{k}$ is added in order to cancel out the term $\left\|\boldsymbol{e}^{i_{k}}-\boldsymbol{e}^{j_{k}}\right\|$, which is a part of $\left\|\boldsymbol{d}^{k}\right\|$. One effect of this scaling is that the value of $\tilde{r}_{k}$ is then directly connected to $r\left(\boldsymbol{\alpha}^{k}\right)$ for directions corresponding to maximal violating pairs, cf. Definition 7.3.1 and Proposition 7.3.2.

### 10.2.1 Working Set Selection

Subsequently, we examine the behavior of the SMO method depending on the chosen working set selection strategy. In particular, in each step of the SMO method we select

- a maximal violating pair (MVP) according to Definition 7.3.1,
- a bound-aware maximal violating pair (baMVP), i.e., a pair realizing the value of $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right)$ defined in Definition 7.6.1,
- a maximal decrease pair (MDP) according to Definition 7.7.2,
- a maximal unbounded decrease pair (MuDP) according to Definition 7.7.1,
- a total maximal decrease pair (tMDP) according to Definition 7.7.3.

Recall that the MuDP selection rule is used in state-of-the-art implementations of the SMO method, cf. [FCL05, CL11].

To obtain a broad overview, all five selection strategies are compared by means of the development of the optimality measures in Figure 10.2.1. Because the values of the considered measures are fluctuating, only the minimal value obtained so far is visualized for each step. Of course, the particular performance of the selection rule depends on the choice of the parameters in the training problem. This means that small differences in the behavior should not be over-interpreted. However, one could expect that large differences can be pointed down to the concrete selection method.


Figure 10.2.1: Development of the optimality measures for different working set selection rules

From the graphs it is obvious to see that there is a vast difference between the selection methods using only first-order information (MVP and baMVP) and the ones using second-order information (MDP, MuDP and tMDP). Hereby it should be pointed out that the visualization of the measures over the number of steps can be misleading because the computational effort for each step differs. Recall that the realizations of the selection rules MVP, MDP and MuDP take $\mathcal{O}(n)$ operations each. In contrast, for baMVP and tMDP the number of operations is $\mathcal{O}\left(n^{2}\right)$ (or $\mathcal{O}(n \log n)$ using some
additional sorting in the implementation), because all pairs are considered. This fact must be taken into account for the evaluation of the results below.

Considering the first-order selection rules only, one can see that there is virtually no difference in the development of the primal-dual gap and the bound-aware violation measure. On the other hand, the progress of the simple violation measure (see Figure 10.2.1(c)) differs significantly in the beginning. That is due to the fact that with the baMVP rule there is no immediate demand for the handling of variables corresponding to large values of $r\left(\boldsymbol{\alpha}^{k}\right)$. As a byproduct, this observation shows that the value of $r\left(\boldsymbol{\alpha}^{k}\right)$ might not be an appropriate optimality measure because there exist points which are close to optimality (in terms of the other measures) but yield rather high values for the simple violation measure, cf. also Example 7.2.5.

Another observation here is that practically (i.e., for the relevant optimality measures) there is no difference between MVP and baMVP which means that it is not recommendable to use the baMVP rule at all because of its higher runtime per step. On the other hand, recall that the baMVP is still necessary for the application of the convergence theory proposed in Section 7.4.


Figure 10.2.2: Development of the optimality measures for second-order working set selection rules

For the second-order selection rules presented in Figure 10.2.2, the overall picture does not allow for definitive conclusions. One could argue that the tMDP rule always performs best. But as said before, the gain of performance is paid for by a disproportionately high computational effort per step. This means that the application of the tMDP rule is not usable in practice. Nevertheless, it is interesting to observe that the convergence speed (per step) of the conventional rules could be improved.

Comparing the other two rules, we see that they perform very similar in the beginning of the optimization procedure, where selecting the MDP yields a marginally small advantage. Towards the end of the training process, this advantage turns into a slight disadvantage. Unfortunately, it is not clear whether this behavior is characteristic from the evaluation of a single training problem.

### 10.2.2 Comparison of Optimality Measures

In this subsection, we analyze the relations between the previously introduced measures empirically. We start with the setting of the previous subsection and visualize the values of the simple violation measure and the primal-dual optimality gap depending on the associated value of the bound-aware violation measure at each step of the SMO method. To highlight the different behavior of the violation measures, which was already noted in Figure 10.2.1, the values are presented for the MVP and the baMVP selection rule independently in Figure 10.2.3. The presentation of the other selection rules is omitted here, because the observed behavior is not significantly different from that of the MVP rule. Note that the value of the bound-aware violation measure is scaled by the factor $\lambda^{-1}$ in order to match the order of magnitude of the other optimality measures.


Figure 10.2.3: Comparison of optimality measures

It can be seen that the relation between the bound-aware violation measure and the primal-dual gap is almost identical for both selection approach. For the considered realizations of the training method, both optimality measures seem to be in a similar order of magnitude (visualized by the bisecting line).

On the other hand, the simple violation measure behaves differently. In the case of MVP selection, the values of $r\left(\boldsymbol{\alpha}^{k}\right)$ and $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) / \lambda$ are almost equal with a few exceptions which deviate only slightly. This means that the values could be used equally in applications (for instance for the definition of a suitable termination criterion or as reference value in a shrinking approach following Section 7.10). For the baMVP selection rule the image is not as clear because no apparent relation can be observed for larger values. Note that we have already seen this kind of relation in Figure 10.2.1 (c) and discussed this issue in the previous subsection. Nevertheless, we observe in this example that

$$
\begin{equation*}
r\left(\boldsymbol{\alpha}^{k}\right) \approx r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) / \lambda \tag{10.2.1}
\end{equation*}
$$

for small values.

### 10.2.3 Behavior of Directional Derivative

In Figure 10.2 .4 we consider the relation between the bound-aware violation measure and the value of the directional derivative in the direction $\boldsymbol{d}^{k}$ for two different working set selection rules.


Figure 10.2.4: Development of $r_{k}$ compared to $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right)$
Recall from the discussion in Subsection 7.9.2 and the definition of the SMO method in Algorithm 7.8.1 that there is a close connection between both values if the baMVP selection strategy is applied. The difference comes from the idea that the parameters $\rho_{0}$ and $\rho_{1}$ are chosen to allow for large steps in the step size computation on the one hand while keeping the descent estimates suitable for the working set selection on the other hand. For the particular realization of the method the values $\rho_{0}=1$ and $\rho_{1}=\lambda$ are used. This implies that there is at most a factor of $\lambda$ between the values of $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right)$ and $r_{k}$ for baMVP selection. This theoretical bound is visualized as additional line in the figure. Note that this relation is not forced for the simple MVP selection such that $r_{k}$ can be smaller than $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right)$, which is also observed in Figure 10.2.4(a).

Asymptotically, the values of $r_{k}$ are strictly larger than $r^{\circ}\left(\boldsymbol{\alpha}^{k}, \lambda\right)$ and often close to $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) / \lambda$. Using the relation observed in Figure 10.2.3, this means that it can be expected that $r_{k}$ is in order of $r\left(\boldsymbol{\alpha}^{k}\right)$.

In Figure 10.2.5 the values of $r\left(\boldsymbol{\alpha}^{k}\right)$ and $\tilde{r}_{k}$ are compared for two different working set selection rules. Note that the MVP strategy is not considered here, because by construction it holds that $r\left(\boldsymbol{\alpha}^{k}\right)=\tilde{r}_{k}$ in that case. For the MDP rule one can observe that both measures behave similarly, whereas this is only true for smaller values when baMVP selection is applied. But for the latter rule, one can observe that both values are very close asymptotically, i.e.,

$$
\tilde{r}_{k} \approx r\left(\boldsymbol{\alpha}^{k}\right) \approx r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) / \lambda,
$$

using (10.2.1).


Figure 10.2.5: Development of $\tilde{r}_{k}$ compared to $r\left(\boldsymbol{\alpha}^{k}\right)$

Because it is fairly expensive to compute the value of $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right)$ in each step and based on the previous findings that there is a strong connection to the value of $r\left(\boldsymbol{\alpha}^{k}\right)$ for small values, we use the simple violation measure as a baseline instead of the bound-aware violation measure from now on.

### 10.2.4 Relations between the Measures

In another experiment, we consider the relation between the proposed measures for small values. In order to observe asymptotic behavior in a reasonable amount of time, we restrict our investigation to the MDP selection rule and skip the computation of the bound-aware violation measure. The obtained results are visualized for different parameter choices in Figure 10.2.6.

The plots show that there is an apparent relation between the considered quantities. In particular, the simple violation measure $r\left(\boldsymbol{\alpha}^{k}\right)$ and the step size $\tau_{k}$ is always in the same order of magnitude as the primal-dual gap and the distance of the current point to the solution. To obtain the same observation for the decrease of the objective function $\Delta \psi_{\mathrm{d}}^{k}$, the square roots of the values are visualized. This shows that $\Delta \psi_{\mathrm{d}}^{k}$ is in the order of $r\left(\boldsymbol{\alpha}^{k}\right)^{2}$, which supports the decrease condition (7.4.9) if one considers that the values of $\tilde{r}_{k}, r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) / \lambda$ and $r\left(\boldsymbol{\alpha}^{k}\right)$ are all proportional to each other empirically.

### 10.2.5 Parameter Dependency of Relations

Note that the previously found relations are not very sensitive with respect to changes in the scaling parameter $\gamma$. In contrast, a change of the regularization parameter $\lambda$ introduces a shift of the points in the log-log plot, which corresponds to a variation of the relationship by means of some factor. To analyze this behavior empirically, we consider the problem for fixed $\gamma=30$ and varying parameter $\lambda$. For each fixed value of $\lambda$ we assume that the relation between a measure $v_{k}$ and the gap $r\left(\boldsymbol{\alpha}^{k}\right)$ can be approximately described by

$$
\begin{equation*}
v_{k} \approx c(\lambda) \cdot r\left(\boldsymbol{\alpha}^{k}\right) \tag{10.2.2}
\end{equation*}
$$



Figure 10.2.6: Relation between the primal-dual gap and the simple violation measure, the step size and the decrease of the objective function
which has been observed in Figure 10.2.6. The SMO algorithm is conducted with the MDP rule for varying value of $\lambda$ and is stopped after a maximum iteration number of 500,000 or whenever $r\left(\boldsymbol{\alpha}^{k}\right)<10^{-10}$.

Using the obtained results, the factor $c(\lambda)$ is estimated by means of a linear regression over all measured points with $10^{-9} \leq r\left(\boldsymbol{\alpha}^{k}\right) \leq 10^{-3}$, which is the region, where the linear behavior is observed. Note that the linear regression yields a constant term of approximately zero for (10.2.2), such that it makes sense to omit it in the first place. The resulting values are visualized in Figure 10.2.7.

From the visualization of the fitted parameters, one can conclude that there is a linear tendency in the log-log space for all considered measures. Only the values of $D_{k}$


Figure 10.2.7: Development of the factor $c(\lambda)$ for the relation assumed in (10.2.2)
and $V_{k}$ do not show a tight linear relation. Hence, these measures are not considered subsequently. In the following, a linear regression will be applied for all other measures (after taking the logarithm of $\lambda$ and $c(\lambda)$, respectively) to estimate the parameters of the underlying functions. The result of this regression is already shown in Figure 10.2.7 and the resulting relationships can be summarized as

$$
\begin{aligned}
\Delta^{\star}\left(\boldsymbol{\alpha}^{k}\right) & \approx 5.770 \cdot \lambda^{-0.04} \cdot r\left(\boldsymbol{\alpha}^{k}\right), \\
\tilde{\tau}_{k} & \approx 1.648 \cdot \lambda^{0.958} \cdot r\left(\boldsymbol{\alpha}^{k}\right), \\
\Delta \psi_{\mathrm{d}}^{k} & \approx 0.427 \cdot \lambda^{0.989} \cdot r\left(\boldsymbol{\alpha}^{k}\right)^{2}, \\
\tilde{r}_{k} & \approx 0.707 \cdot \lambda^{0.014} \cdot r\left(\boldsymbol{\alpha}^{k}\right) .
\end{aligned}
$$

Together with the observation that $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) \approx r\left(\boldsymbol{\alpha}^{k}\right)$ (which is true unless the baMVP strategy is used) and the very small exponent for $\lambda$ in the first and fourth approximation, we find that

$$
r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) \approx r\left(\boldsymbol{\alpha}^{k}\right) \approx 0.173 \Delta^{\star}\left(\boldsymbol{\alpha}^{k}\right) \approx 1.415 \tilde{r}_{k}
$$

Moreover, the fact that the exponents for $\lambda$ in the other two expressions are approximately equal to one, we end up with the further approximations

$$
\tilde{\tau}_{k} \approx 1.648 \cdot \lambda \cdot r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) \approx 2.331 \cdot \lambda \cdot \tilde{r}_{k}
$$

and

$$
\Delta \psi_{\mathrm{d}}^{k} \approx 0.427 \cdot \lambda \cdot r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right)^{2} \approx 0.855 \cdot \lambda \cdot \tilde{r}_{k}^{2} .
$$

This confirms the decrease condition (7.9.3) and also matches the theoretical step size (7.9.2). The factor $\lambda$ which occurs in both approximation can be explained by means of the Lipschitz constant in the theoretical estimates and the observation that $L^{-1} \sim \lambda\|\boldsymbol{K}\|^{-1}$, cf. Example 7.9.1.

### 10.2.6 Observation for Problems with Smoothed Maximum Function

In order to prepare for the application of derivative-based methods for the solution of the primal training problem, we now change the training problem by using a smooth
approximation $\widehat{m}_{\delta}$ of the maximum function according to Observation 4.4 .8 with $\delta=\frac{1}{4}$ and compare the newly observed results with the previous ones. Because the dual problem is easier to solve for quadratic loss functions, we first consider the function $\widehat{m}$ defined in Example 4.4.4 and exchange it later if more smoothness is needed in the primal training problem.


Figure 10.2.8: Behavior of the optimality measures for a problem smoothed maximum function and MDP working set selection

The visualization of the results in Figure 10.2 .8 shows that the values of the violation measures $r\left(\boldsymbol{\alpha}^{k}\right)$ and $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) / \lambda$ behave in the same way as before, whereas there is a fundamental change in the development of the primal-dual gap. In particular, one can see that the values of $\Delta^{\star}\left(\boldsymbol{\alpha}^{k}\right)$ and $V_{k}$ are only comparable in the smooth setting. Here, we observe empirically that

$$
\Delta^{\star}(\boldsymbol{\alpha}) \sim r\left(\boldsymbol{\alpha}^{k}\right)^{2}
$$

Further computational experiments show that this relationship is also present for smaller values of $\delta$, which means that it is caused by the smoothness of the primal training problem.

As a secondary result we find that

$$
D_{k} \sim r\left(\boldsymbol{\alpha}^{k}\right) \sim r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right) / \lambda,
$$

which means that the error bound condition (7.10.7) is observed in practice.

### 10.2.7 Number of Support Vectors and Active Set

In order to get an idea of the dependency of the method's performance on the regularization parameter we consider the development of the number of support vectors and the number of iterations in Figure 10.2.9. The computational setting is the same as in previous subsection.


Figure 10.2.9: Behavior of the SMO method for different values of the parameter $\lambda$

The increase of the number of support vectors in the upper part of Figure 10.2.9(a) is related to the fitting quality of the decision function. As we have already argued in Section 10.1, for larger values of the regularization parameter $\lambda$, the resulting decision function tends to under-fit. In this case, many training samples have to be misclassified by the solution of training problem, which leads to a high fraction of nonzero variables in the solution. Following Subsection 4.3.1, extremely misclassified training samples lead to variables with $\alpha_{i}=y_{i}$. This means that the associated variables are fixed at their bound in the dual problem. Note that a similar argumentation is possible for other formulations of the training problem, in particular for $\varepsilon$-Support Vector Regression.

The lower part of Figure 10.2.9(a) shows the development of the number of free variables depending on $\lambda$, which cause the major part of the computational effort in the SMO method, cf. Subsection 7.9.1. One can observe that this number is relatively small and does not change significantly for varying $\lambda$. This means that the shrinking approach is very helpful to reduce the computational effort per step if the correct set of active variables can be detected successfully.

### 10.2.8 Application of the Shrinking Approach

In order to evaluate the approach for detecting fixed variables introduced in Section 7.10, we consider the application of Corollary 7.10 .5 for the practical setting. In particular, we investigate the behavior of the index sets predicted by means of (7.10.8) during the optimization process compared to the set of variables attaining the corresponding bound at the solution of the training problem.

Let $\boldsymbol{\alpha}^{\star} \in \mathbb{R}^{n}$ be a solution of the dual training problem. In practice this solution is unique such that the set of variables attaining a bound depends only on the training problem. We consider the sets of indices

$$
\underline{I}^{\star}:=\left\{i \in\{1, \ldots, n\} \mid \alpha_{i}^{\star}=\underline{\alpha}_{i}\right\} \quad \text { and } \quad \bar{I}^{\star}:=\left\{i \in\{1, \ldots, n\} \mid \alpha_{i}^{\star}=\bar{\alpha}_{i}\right\}
$$

associated with variables at the lower and upper bound, respectively. Then,

$$
I_{f}^{\star}:=\{1, \ldots, n\} \backslash\left(\underline{I}^{\star} \cap \bar{I}^{\star}\right)
$$

is the index set of variables, which are strictly between the bounds at the solution. For the evaluation of the prediction, the ratio of false detections (of variables at the bound)

$$
R_{\mathrm{false}}:=\frac{\left|\underline{I}(\alpha) \backslash \underline{I}^{\star}\right|+\left|\bar{I}(\alpha) \backslash \bar{I}^{\star}\right|}{\left|\underline{I}^{\star}\right|+\left|\bar{I}^{\star}\right|}
$$

and the ratio of missing detections (of variables at the bound)

$$
R_{\text {missing }}:=\frac{\left|\underline{I}^{\star} \backslash \underline{I}(\alpha)\right|+\left|\bar{I}^{\star} \backslash \bar{I}(\alpha)\right|}{\left|\underline{I}^{\star}\right|+\left|\bar{I}^{\star}\right|}
$$

are considered.
From a practical point of view it is desirable to realize a detection which has a low number of missing detections because every variable, which is detected to be at the bound, can be removed from the optimization problem according to Proposition 4.3.1 and the argumentation at the end of Subsection 7.9.1, see also Subsection 8.3.4 for the according discussion in the primal training setting. On the other hand, it is necessary not to overestimate the set of fixed variables, i.e., to keep the value of $R_{\text {false }}$ equal to zero or at least close to zero. Otherwise, shrinking would lead to subproblems which are not equivalent to the original one and additional effort has to be made in a subsequent recovery strategy.

During the process of the SMO method, the development of the values of $R_{\text {false }}$ and $R_{\text {missing }}$ can be observed given that the solution of the problem is precomputed. For the subsequent visualization, we compute those values every 10 steps. Note that we use $r\left(\boldsymbol{\alpha}^{k}\right)$ instead of $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right)$ for the determination of the index sets in order to avoid the computation of $r^{\diamond}\left(\boldsymbol{\alpha}^{k}, \lambda\right)$.

The behavior of the detection approach for a fixed training problem and parameter $c_{2}=1$ while varying the parameters $\eta=\eta_{1}=\eta_{2}$ and $c_{1}$ is visualized in Figure 10.2.10. Note that the choice of $\eta=1$ is not justified by the theory but serves as a limiting baseline for the practical evaluation.


Figure 10.2.10: Performance of the detection for different parameters: a vertical line marks the last step at which a nonzero value of $R_{\text {false }}$ and $R_{\text {missing }}$, respectively, occurs

In the graphics, a vertical line is added to highlight the number of steps which are necessary in order to obtain a perfect detection of variables at the bound. For the particular training problem, the ratio of false detection drops relatively fast, whereas much more steps are needed in order to obtain a completely correct detection.

For the choice of the parameters for the detecting functions it is necessary to find a tradeoff between conflicting goals. From the visualization it can be seen that the value of $c_{1}$ mainly determines at which step the first detection occurs. However, there is no significant change in the performance of the estimation from that step on. Moreover, one cannot observe an obvious shift of the vertical lines for varying $c_{1}$, i.e., the precise choice of that parameter is not critical.

On the other hand, the choice of $\eta$ significantly changes the quality of the estimates. As indicated before, there is a tradeoff to make because a smaller value of $\eta$ helps to prevent false detection in the early phase, whereas a larger value leads to a faster reduction of $R_{\text {missing }}$ later on. With regard to the application of the shrinking idea inside the SMO method, a fast decline of false predictions is essential because any false prediction results in additional effort of the necessary post-processing step. In contrast, it is possible that the reduction of the computational effort by means of early shrinking
outweighs this cost. Unfortunately, one cannot expect to obtain a universally valid rule from this kind of argumentation because the overall performance gain introduced by the shrinking heuristic highly depends on the characteristics of the training problem at hand.

To demonstrate the possible effect of shrinking on the runtime, the training problem is considered for $n=10,000$ training samples and the SMO method is applied with and without shrinking for different tolerance values in the termination criterion. In order to make the runtime of the SMO method competitive, the best identified approach so far is implemented in the Nim programming language, which is compiled using the Nim Compiler in version 1.6.8. This implementation is also used in the subsequent subsection.

The SMO method is stopped if the value of $r\left(\boldsymbol{\alpha}^{k}\right)$ falls below a predefined tolerance level and the measured runtimes are visualized in Figure 10.2.11. This figure shows that there is a significant reduction of the runtime if shrinking is applied. Moreover, the application of the shrinking approach makes it particularly possible to reduce the time of the steps in the later phase of the optimization. This can be seen from the fact that the increase of the runtime for a reduced termination tolerance is far less important if shrinking is used. This means that is especially important to implement shrinking if the training problem has to be solved with a higher precision.


Figure 10.2.11: Runtime (measured in seconds) for different choices of the termination criterion

### 10.2.9 Performance for Increasing Problem Size

In a final subsection, we briefly investigate the dependence of the runtime of the SMO method on the size of the dataset and the values of the regularization parameter. Note that, it is practically not reasonable to use a first-order method as SMO with a very strict termination criterion. However, as we have noted previously, the application of shrinking makes it possible to obtain points with a rather small violation measure in reasonable runtimes. Thus, we consider the SMO method which is terminated if $r\left(\boldsymbol{\alpha}^{k}\right) \leq 10^{-8}$. The results are shown in Figure 10.2.12.


Figure 10.2.12: Development of runtime and number of steps of SMO method for different choices of problem size and regularization parameter

From the visualization in Figure 10.2.12(a) it can be seen that the number of steps which are needed to meet the termination criterion grows approximately in the order of $n^{0.6}$. Practically, this number scales similar to the number of variables which are strictly between their bounds at the solution. The latter number is visualized for increasing problem size in Figure 10.2.13. Of course, the fact that the number of iterations scales with the number of free variables can be expected for first-order methods.


Figure 10.2.13: Number of variables which are strictly between the bounds at the solution for training problem of varying size with $\lambda=10^{-3}$

On the other hand, the runtime scales approximately in the order of $n^{1.6}$. This seems to be inconsistent with the observed number of iterations and the idea that each step has a computational effort in the order of the number of variables in the reduced problem, which is approximately equal to the number of free variables because of the shrinking approach. However, even though the termination tolerance is set relatively low, the major share in runtime is taken by the first phase of the SMO method before the shrinking can actually be applied.

In Figure 10.2.12(a) one can also see that the number of iterations is strongly influenced by the value of the regularization parameter whereas the effect on the runtime is not as strong but still noticeable. A smaller value of $\lambda$ leads to an increased number of steps and a higher runtime. This implies that it is desirable to avoid unnecessarily small values for $\lambda$ in practice in order to keep the training time as small as possible. Unfortunately, this is not always feasible because the value of the regularization parameter also determines the quality of the resulting decision function. Hence, it is practically necessary to find a tradeoff between the runtime of the training method and the desired precision.

### 10.3 Methods for the Primal Training Problem

The focus in this section is on the application of the methods for solving the primal training problems which were introduced in Chapter 8, namely

- the simple gradient descent method according to Algorithm 8.2.1,
- the preconditioned gradient descent method according to Algorithm 8.3.1, and
- the realization of Newton's method according to Algorithm 8.4.2.

We consider the same training problem as in the previous section but with a smooth approximation of the maximum function in the problem formulation. This setting has already been introduced in Subsection 10.2.6.

### 10.3.1 Performance of First-Order Methods

In the first experiments, we only consider the first two methods because the complexity of Newton's method per step is not comparable to that of the other ones. Right from the start, it should be emphasized that the simple gradient method is in general not applicable practically because the full kernel matrix is needed in each step, which leads to a tremendous computational effort per step. Moreover, even though Subsection 8.3.3 suggests that some effort can be reduced for the preconditioned gradient descent method, this approach is not very relevant in practice as we will see in the following. For comparison, the methods were applied each with $1,000,000$ steps and with the parameters $\eta_{0}=\lambda, \eta=\frac{1}{10}$, and $\sigma=\frac{1}{10}$. The development of some problem-specific optimality measures are visualized in Figure 10.3.1.

The direct comparison of the simple and preconditioned gradient descent method demonstrates a drastic difference in the convergence speed. While the performance is almost equal during in the first phase, a faster convergence is noticeable for the preconditioned version later on. In addition, the computational cost per step of the simple gradient method is slightly higher than that of the preconditioned method because an additional matrix-vector multiplication is needed for the computation of the direction. This also reinforces the fact that the preconditioned gradient descent method should be preferred practically.

Note that the preconditioned gradient descent method stops prematurely after 282,527 steps because of numerical instabilities. Those led to insufficient precision


Figure 10.3.1: Development of optimality measures for the application of first-order primal training methods
such that no further descent for the objective function was possible numerically. Of course, the same problem would occur for the simple gradient descent method if the iteration process was proceeded.

Even though the optimality measures are not directly comparable to that of the dual SMO method in the previous section, a rough comparison of Figure 10.3.1 and Figure 10.2.8 implies that the primal methods are outperformed by the SMO method. On the one hand, it is obvious that the dual method takes far fewer steps to obtain low values for the optimality measures. On the other hand, the cost per step is tremendously lower than that of the primal methods. Of course, this observation is based on a single training problem. But there is no compelling reason for assuming that this behavior should be significantly different for other training problems. This means that the first-order primal training methods considered here do not seem to be recommendable for practical applications.

### 10.3.2 Empirical Evaluation of an Error Bound Condition

In order to evaluate the possibility that the optimality system satisfies an error bound condition of the form (8.3.10), we examine the relation between the distance of a given point to the solution and the computed optimality measures. The values, which were recorded in the process of the preconditioned gradient descent method, are visualized in Figure 10.3.2.

For the particular training problem and in the application of the preconditioned gradient descent method we find that empirically the error bound condition holds for the iterates because

$$
\left\|\boldsymbol{F}\left(\boldsymbol{z}^{k}\right)\right\| \sim\left\|\boldsymbol{z}^{k}-\boldsymbol{z}^{\star}\right\|=\operatorname{dist}\left[\boldsymbol{z}^{k}, \mathcal{Z}^{\star}\right]
$$

holds asymptotically. In Figure 10.3.2(a) we can also observe that the norm of the gra-


Figure 10.3.2: Empirical behavior of particular measures compared to $\left\|\boldsymbol{F}\left(\boldsymbol{z}^{k}\right)\right\|$
dients for both considered problems are fluctuating whereas the values of $\left\|\boldsymbol{F}\left(\boldsymbol{z}^{k}\right)\right\|$ and the distance fairly align. This indicates that the norm of the system of nonlinear equations is a far better measure to estimate the distance than the norm of the gradients. We can also see that

$$
\psi_{\mathrm{pf}}\left(\boldsymbol{z}^{k}\right)-\psi_{\mathrm{pf}}\left(\boldsymbol{z}^{\star}\right) \sim\left\|\boldsymbol{F}\left(\boldsymbol{z}^{k}\right)\right\|^{2} \sim\left\|\boldsymbol{z}^{k}-\boldsymbol{z}^{\star}\right\|^{2}
$$

which is of course justified by the fact that the objective function is locally quadratic.
Finally, Figure 10.3.2(b) shows that the distances of both parts of the variables behave quite similar asymptotically. There is a difference between those values, which can be described by a factor asymptotically, because the term $\left\|\boldsymbol{\alpha}^{k}-\boldsymbol{\alpha}^{\star}\right\|$ captures $n$ variables, whereas the other term captures only a single one. However, both parts of the distance converge in the same order of magnitude. This observation supports the preconditioning approach introduced by the scaling matrix $\boldsymbol{W}$ in (8.3.2) because the distance term for the bias dominates practically without this scaling.

### 10.3.3 Comparison of Newton's Method and SMO

In a final experiment in this section, we consider the performance of Newton's method for an increasing number of training samples and different choices of the regularization parameter. The size of the training set is chosen in the range between $n=100$ and $n=40,000$ training samples. In the particular implementation of the method the kernel matrix is precomputed once in order to reduce the overall runtime. Of course, this is only feasible for this relatively small problem size because of memory limits. The results for the application of Newton's method to the training problem are presented in Figure 10.3.3.


Figure 10.3.3: Development of runtime and number of steps of Newton's method for different choices of problem size and regularization parameter

One can observe that the number of steps needed for the convergence of Newton's method is more or less independent of the problem size, but increases for smaller choices of the regularization parameter. On the other hand, the runtime of Newton's method shows empirically a quadratic growth, i.e., the empirical complexity of Newton's method is worse than that of the SMO considered in Subsection 10.2.9.

Note that the difference here is that it makes not a great difference to terminate Newton's method early by means of some inexact termination criterion because of the fast asymptotic convergence rate and the fact that the final steps are relatively cheap due to vanishing second derivatives of many loss functions. Practically, the application of Newton's method generally yields a more precise solution of the training problem, whereas the SMO method has to be stopped with an approximate solution. This means that the comparison of both approaches is not fully appropriate. Nevertheless, the runtime of both methods is presented in Figure 10.3.4 for different choices of the regularization parameter.


Figure 10.3.4: Comparison between Newton's method and SMO method with respect to the runtime (measured in seconds)

One can observe that the runtime of Newton's method is better than that of the SMO method only for medium-sized training problems. For small problems the overall runtime is very low such that the comparison is not quite expressive. This means that a noticeable difference in the performance of the considered methods occurs primarily for problems with more than a few thousand training samples. It should be emphasized that this observation possibly depends on the structure of the particular dataset and that the behavior can actually be different in practical applications.

Moreover, the better performance of the SMO method can also be used to generate sufficiently good starting points for Newton's method. This is particularly useful if the problem size is very large because then the application of Newton's method is practically not feasible due to memory constraints and the large expected runtime. However, by switching from the SMO method to Newton's method it is possible to obtain very precise solutions of the training problem which are needed, for instance, for the application of the derivative-based hyperparameter optimization approach. The latter will be considered in the following subsection.

### 10.4 Practical Application of Hyperparameter Optimization

In this section, we evaluate the hyperparameter optimization approach proposed in Chapter 9 for practical problems. For each of the considered problems, we use the cross-validation error minimization problem proposed in (9.1.5) combined with the implicit function approach discussed in Section 9.2. For the resulting optimization problem, in which the hyperparameter vector $\boldsymbol{\vartheta}$ consists of the hyperparameters $\lambda$ and $\gamma$, the lower level training problems are solved by means of a combination of the SMO method and Newton's method as considered in the previous sections. Instead of optimizing the parameters directly, the terms $\log (\lambda)$ and $\log (\gamma)$ are considered in (9.1.5). This idea is commonly applied in practice (cf., for instance, [KSC06]) and improves the scaling while implicitly enforcing the non-negativity of the parameter values. The method L-BFGS-B in scipy.optimize.minimize of SciPy 1.9.3 (cf. [VGO+20]) is used to determine a locally optimal point of the hyperparameter optimization problem.

### 10.4.1 Consideration of the Artificial Dataset

In a first experiment, we consider the relation between the hyperparameter values determined by the optimization approach and the number of training samples. For this reason, a dataset of the form introduced in Section 10.1 is used with a number of training points varying between $n=100$ and $n=10,000$. The classical Support Vector Classification problem as summarized in Proposition 4.1.1 with a smooth approximation of the maximum function according to Subsection 10.2.6 is used as a training problem in the lower level problem. Moreover, the validation loss functions in the upper level problem are also chosen as a smoothed version of hinge loss, i.e.,

$$
L_{i}(t):=\widehat{m}_{\delta}\left(1-y_{i} t\right)
$$

with $\delta=\frac{1}{4}$ and $\widehat{m}$ defined according to Example 4.4.2. The optimization is started with an initial guess of $\lambda=10^{-2}$ and $\gamma=4$, and the parameters are restricted to satisfy $\lambda \in\left[10^{-4}, 1\right]$ and $\gamma \in\left[10^{-12}, 10^{4}\right]$. Note that the lower bound on $\lambda$ is chosen relatively large in order to avoid unnecessarily high solution times. In Figure 10.4.1 the obtained hyperparameter values are visualized.


Figure 10.4.1: Development of the determined parameter values depending on the problem size: values obtained from the optimization approach and loglog linear fit

The plots in Figure 10.4.1(a) suggest that the determined value for the scaling parameter $\gamma$ is more or less unaffected by the change in the number of training samples. On the other hand, the values for the regularization parameter shown in Figure 10.4.1 (b) tend to become smaller for larger datasets. Indeed, as a result of the log-log linear fit, we obtain that

$$
\gamma^{\star}(n) \approx 22.869 \cdot n^{0.044} \quad \text { and } \quad \lambda^{\star}(n) \approx 12.091 \cdot n^{-1.305}
$$

Note that these relations are also consistent with the parameter values $\gamma=30$ and $\lambda=$ $10^{-3}$ chosen in Section 10.1 for $n=1,000$.

This behavior of the hyperparameter values is understandable because the scaling is not affected by the number of training samples, such that an optimal value of $\gamma$ can be expected to be more or less independent of $n$. On the other hand, if a larger amount of training samples is available, the effect of overfitting is reduced. Hence, the decision function could be adapted more tightly to the given dataset which is only possible for smaller values of the regularization parameter $\lambda$.

### 10.4.2 Application to Real-World Regression Problems

In this subsection, we consider real-world regression problems and define particular training problems according to the $\varepsilon$-Support Vector Regression ( $\varepsilon$-SVR), cf. Proposition 4.2.2. In each problem, the maximum function in the loss functions of (4.2.1)
is replaced by a smooth approximation $\widehat{m}_{\delta}$ according to Observation 4.4 .8 with $\widehat{m}$ as defined in Example 4.4.2 and $\delta=\frac{\varepsilon}{4}$.

Additionally, we employ the Least Squares Support Vector Machine (LS-SVM) as defined in Proposition 4.2.1. Note that it is not necessary to apply the SMO method for this problem because Newton's method terminates with the exact solution after the first step since the objective function of the training problem is quadratic.

The validation loss functions in the upper level problem are chosen as a smoothed version of the absolute error, namely

$$
\begin{equation*}
L_{i}(t):=\widehat{m}_{\delta}\left(t-y_{i}\right)+\widehat{m}_{\delta}\left(y_{i}-t\right) \tag{10.4.1}
\end{equation*}
$$

with $\delta=\frac{1}{10}$ and $\widehat{m}$ defined according to Example 4.4.4. This means that the validation loss is proportional to the squared loss for small errors and equal to the absolute loss for large errors. In the literature this term is known as Huber loss, cf. [Hub64]. For reference, we also consider the squared loss function

$$
\begin{equation*}
L_{i}(t):=\left(t-y_{i}\right)^{2} . \tag{10.4.2}
\end{equation*}
$$

The optimization is started with an initial guess of $\lambda=10^{-2}$ and $\gamma=\log (2) / d$, where $d$ denotes the number of attributes of the input vectors. As already mentioned in the previous examples, a restriction of the parameter range is practically important in order to prevent high training times. Here, we use a slightly larger area of $\lambda \in\left[10^{-6}, 1\right]$ and $\gamma \in\left[10^{-12}, 10^{4}\right]$.

Note that we do not consider the parameter $\varepsilon$ within the hyperparameter optimization but apply the approach for different fixed values. This is because the value of $\varepsilon$ determines the number of support vectors in the final solution, which (depending on the particular application) should be better defined by the user. For most problems, a smaller value of $\varepsilon$ leads to a better performance and an increased number of support vectors.

Subsequently, we mostly adopt the testing procedure of [FLL+15], which can be summarized as follows. Each of the considered datasets is randomly split into two parts for 30 times. In fact, $80 \%$ of the dataset is used for the hyperparameter optimization routine and a subsequent training, whereas the remaining $20 \%$ of the dataset is used to compute error measures. For datasets with more than 1,000 elements, we additionally consider random samplings of 1,000 elements for the whole evaluation procedure.

For each split, the resulting training subset is normalized to have zero mean and unit variance in each component of the input vectors. The same transformation is applied to the left-out test set. In order to simplify the selection of the parameter $\varepsilon$ and the smoothing parameter in the validation loss, the labels of the training set are also normalized to zero mean and unit variance. Afterwards, the generated decision function is modified to undo this transformation for the evaluation on the test set.

The optimization routine is stopped after a time limit of 600 s if it does not reach the default termination criterion earlier. In every case, the best parameter value found
so far is used to obtain the decision function. Once the training is carried out and a decision function $f$ is computed, the test error is evaluated as

$$
\operatorname{MSE}=\frac{1}{\left|\mathcal{N}_{\text {test }}\right|} \sum_{i \in \mathcal{N}_{\text {test }}}\left(f\left(\boldsymbol{x}_{i}\right)-y_{i}\right)^{2} \quad \text { and } \quad \text { MAE }=\frac{1}{\left|\mathcal{N}_{\text {test }}\right|} \sum_{i \in \mathcal{N}_{\text {test }}}\left|f\left(\boldsymbol{x}_{i}\right)-y_{i}\right| .
$$

The results are shown in Table 10.4.1.

| dataset | size | $L_{i}$ | MSE |  |  |  | MAE |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  |  |  | $\varepsilon=0.1$ | $\varepsilon=0.25$ | $\varepsilon=0.5$ | LS-SVM | $\varepsilon=0.1$ | $\varepsilon=0.25$ | $\varepsilon=0.5$ | LS-SVM |
| pyrim | 74 | (10.4.1) | 0.00656 | 0.00625 | 0.00645 | 0.00597 | 0.0532 | 0.0512 | 0.0526 | 0.0491 |
|  |  | (10.4.2) | 0.00780 | 0.00780 | 0.00745 | 0.00814 | 0.0559 | 0.0535 | 0.0548 | 0.0554 |
| triazines | 186 | (10.4.1) | 0.0244 | 0.0239 | 0.0237 | 0.0232 | 0.107 | 0.109 | 0.111 | 0.110 |
|  |  | (10.4.2) | 0.0251 | 0.0246 | 0.0238 | 0.0233 | 0.112 | 0.112 | 0.111 | 0.110 |
| mpg | 392 | (10.4.1) | 6.50 | 6.58 | 7.37 | 6.63 | 1.81 | 1.84 | 1.98 | 1.82 |
|  |  | (10.4.2) | 6.47 | 6.66 | 7.53 | 6.58 | 1.81 | 1.84 | 2.01 | 1.82 |
| housing | 506 | (10.4.1) | 10.8 | 10.9 | 11.9 | 9.54 | 2.10 | 2.17 | 2.43 | 2.07 |
|  |  | (10.4.2) | 10.9 | 10.5 | 12.4 | 9.75 | 2.12 | 2.16 | 2.47 | 2.08 |
| spacega | 1,000 | (10.4.1) | 0.0133 | 0.0127 | 0.0128 | 0.0127 | 0.0849 | 0.0845 | 0.0854 | 0.0845 |
|  |  | (10.4.2) | 0.0130 | 0.0128 | 0.0127 | 0.0125 | 0.0848 | 0.0849 | 0.0852 | 0.0848 |
|  | 3,107 | (10.4.1) | 0.0110 | 0.0109 | 0.0113 | 0.0110 | 0.0760 | 0.0757 | 0.0773 | 0.0755 |
|  |  | (10.4.2) | 0.0109 | 0.0110 | 0.0112 | 0.0110 | 0.0759 | 0.0764 | 0.0779 | 0.0764 |
| abalone | 1,000 | (10.4.1) | 4.72 | 4.61 | 4.56 | 4.53 | 1.50 | 1.50 | 1.52 | 1.53 |
|  |  | (10.4.2) | 4.67 | 4.61 | 4.56 | 4.53 | 1.50 | 1.50 | 1.52 | 1.53 |
|  | 4,177 | (10.4.1) | 4.46 | 4.47 | 4.40 | 4.37 | 1.45 | 1.46 | 1.48 | 1.49 |
|  |  | (10.4.2) | 4.45 | 4.45 | 4.40 | 4.37 | 1.45 | 1.46 | 1.48 | 1.49 |
| cpusmall | 1,000 | (10.4.1) | 19.1 | 20.0 | 32.6 | 18.7 | 2.70 | 2.96 | 3.66 | 2.70 |
|  |  | (10.4.2) | 19.2 | 19.5 | 28.6 | 19.5 | 2.72 | 2.96 | 3.75 | 2.73 |
|  | 8,192 | (10.4.1) | 10.6 | 10.4 | 18.1 | 9.06 | 2.22 | 2.47 | 3.34 | 2.16 |
|  |  | (10.4.2) | 10.7 | 10.2 | 15.8 | 8.89 | 2.22 | 2.47 | 3.22 | 2.17 |

Table 10.4.1: Final test error estimates: the lowest values in each row (and all values which are at most $1 \%$ higher) are written in boldface type

It can be seen that the value of the MSE is in most cases best if the LS-SVM is applied. Of course, this is rather natural because this model is using the squared error already in the objective function of the training problem. The only notable difference occurs for the dataset mpg where the $\varepsilon$-SVR with parameter $\varepsilon=0.1$ performs best with respect
to the MSE. Because of the relatively small number of data points this suggests that the loss term in this training problem is able to handle outliers better than that of the LS-SVM for that particular dataset.

On the other hand, the considered formulation of $\varepsilon$-SVR yields comparable or better results than the LS-SVM in terms of the MAE measure for most of the datasets. This is also explainable because the loss function in the training problem is basically an $\varepsilon$-insensitive absolute error term. The actual performance depends on the choice of the parameter $\varepsilon$, and there is a trend that better values can be observed for smaller values of $\varepsilon$. However, the best values are not always obtained for the smallest choice of $\varepsilon$. This observation can be explained by the idea that the parameter $\varepsilon$ also regulates overfitting such that larger values can actually improve the generalization performance for a particular regression problem.

A last observation which we want to make from Table 10.4.1 is that there is no obvious trend on which upper level loss function yields a better performance. In most cases there is no significant difference between the two proposed measures but only a very slight advantage of (10.4.1). Hence, in order to keep the presentation short, we restrict the further investigations to the application of (10.4.1) only.

For each of the trials, we additionally capture the runtime and the number of objective function evaluations. It should be emphasized that the latter value may not be very meaningful because the evaluation time corresponds to the training time. Hence, it essentially depends on the hyperparameter values, i.e., on the point for which the objective function is evaluated. The obtained values are summarized in Table 10.4.2.

|  |  | runtime [s] |  |  |  |  | number of function evaluations |  |  |  |
| :--- | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | ---: | :---: |
| dataset | size | $\varepsilon=0.1$ | $\varepsilon=0.25$ | $\varepsilon=0.5$ | LS-SVM | $\varepsilon=0.1$ | $\varepsilon=0.25$ | $\varepsilon=0.5$ | LS-SVM |  |
| pyrim | 74 | 1.3 | 1.5 | 2.2 | 0.1 | 32.4 | 33.5 | 46.4 | 14.5 |  |
| triazines | 186 | 6.1 | 3.8 | 3.3 | 0.1 | 42.3 | 34.2 | 32.6 | 18.0 |  |
| mpg | 392 | 6.5 | 3.1 | 1.3 | 0.2 | 22.2 | 20.8 | 19.9 | 14.6 |  |
| housing | 506 | 16.3 | 6.6 | 2.6 | 0.3 | 26.3 | 23.4 | 21.6 | 12.0 |  |
| spacega | 1,000 | 41.5 | 35.6 | 20.7 | 2.2 | 21.8 | 22.7 | 20.4 | 17.2 |  |
|  | 3,107 | 361.1 | 265.8 | 141.4 | 16.2 | 18.5 | 20.3 | 19.2 | 14.4 |  |
| abalone | 1,000 | 26.6 | 19.6 | 13.7 | 1.9 | 23.3 | 22.9 | 22.9 | 17.1 |  |
|  | 4,177 | 300.3 | 224.5 | 139.2 | 41.0 | 18.2 | 18.5 | 18.0 | 15.7 |  |
| cpusmall | 1,000 | 53.2 | 17.1 | 24.8 | 1.7 | 29.1 | 21.7 | 33.5 | 15.4 |  |
|  | 8,192 | 600.0 | 560.5 | 600.0 | 130.8 | 4.2 | 12.5 | 15.6 | 12.1 |  |

Table 10.4.2: Runtime (measured in seconds) and number of function evaluations during the hyperparameter optimization

From these results it can be observed that the runtime of the LS-SVM strictly dominates that of the other models. As mentioned earlier, this can be expected because the
solution of each training problem is obtained after a single step of Newton's method (i.e., by means of the solution of a single system of linear equations). In contrast, the computation of the solution of an $\varepsilon$-SVR training problem usually encompasses the application of the SMO method and some further steps of Newton's method. Of course, one could hope that each of the steps are somewhat easier because of the sparseness induced by the structure of the training problem. Note that it can be seen that the runtime decreases for increasing value of $\varepsilon$ which indicates that sparseness is indeed helpful. However, this effect is not strong enough to compete with the LS-SVM as the considered datasets are relatively small. However, the sparseness property is essential for larger problems because the LS-SVM then leads to large systems of linear equations which cannot be solved practically due to memory limits.

Considering the number of function evaluation presented in Table 10.4.2 one can see that those are usually higher for the $\varepsilon$-SVR compared to the LS-SVM. Here, it should be emphasized that the hyperparameter optimization problem is not always solved successfully within the time limit of $600 s$ for the larger dataset with small values of $\varepsilon$ such that the numbers in these cases are not suitable for a comparison. Nevertheless, the hyperparameter optimization problem seems to be more complicated if the $\varepsilon$ SVR training problems are considered. This could be expected because the shape of the decision function depends less smoothly on the hyperparameters as the set of support vectors can vary quickly.
In a last step, we compare the final objective function value and the fraction of support vectors in the decision function for each of the considered settings, cf. Table 10.4.3.

|  |  | objective function value |  |  |  | fraction of support vectors |  |  |  |
| :--- | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| dataset | size | $\varepsilon=0.1$ | $\varepsilon=0.25$ | $\varepsilon=0.5$ | LS-SVM | $\varepsilon=0.1$ | $\varepsilon=0.25$ | $\varepsilon=0.5$ | LS-SVM |
| pyrim | 74 | 0.402 | 0.404 | 0.425 | 0.391 | 0.84 | 0.64 | 0.50 | 1.00 |
| triazines | 186 | 0.639 | 0.654 | 0.672 | 0.660 | 0.90 | 0.78 | 0.61 | 1.00 |
| mpg | 392 | 0.249 | 0.253 | 0.271 | 0.248 | 0.75 | 0.49 | 0.23 | 1.00 |
| housing | 506 | 0.241 | 0.249 | 0.275 | 0.241 | 0.74 | 0.45 | 0.23 | 1.00 |
| spacega | 1,000 | 0.419 | 0.417 | 0.422 | 0.416 | 0.87 | 0.68 | 0.44 | 1.00 |
|  | 3,107 | 0.388 | 0.387 | 0.394 | 0.386 | 0.85 | 0.65 | 0.40 | 1.00 |
| abalone | 1,000 | 0.469 | 0.469 | 0.476 | 0.478 | 0.87 | 0.69 | 0.46 | 1.00 |
|  | 4,177 | 0.456 | 0.457 | 0.462 | 0.465 | 0.86 | 0.68 | 0.45 | 1.00 |
| cpusmall | 1,000 | 0.154 | 0.167 | 0.203 | 0.153 | 0.61 | 0.30 | 0.11 | 1.00 |
|  | 8,192 | 0.128 | 0.139 | 0.229 | 0.124 | 0.56 | 0.24 | 0.09 | 1.00 |

Table 10.4.3: Obtained objective function values and fraction of support vectors
First, note that the obtained objective function values lie in a comparable range for the $\varepsilon$-SVR with small parameter $\varepsilon$ and the LS-SVM. In contrast, for larger values
of $\varepsilon$, it happens that the $\varepsilon$-SVR model is not able to fit the validation set because of the incorporated insensitivity which results in a perceptibly larger objective function value. This property could help to prevent overfitting if many hyperparameters are to be optimized.

Finally, consider the fraction of support vectors presented in the right part of Table 10.4.3. By definition, this ratio is equal to one for the LS-SVM because it does not induce any sparsity. On the other hand, it is clear that the number of support vectors decreases for increasing value of $\varepsilon$ if the $\varepsilon$-SVR is applied. This fact has already been observed previously in terms of the runtime. Since the labels for all datasets were normalized to unit variance, the relation between the value of $\varepsilon$ and the fraction of support vectors is comparable over all datasets. By comparing these values to the associated objective function values, it can be also observed that less support vector are necessary if the model can made to fit better to the dataset.

## 11 Summary and Outlook

In this thesis we considered the application of Fenchel's duality theory and gradientbased methods for the training and hyperparameter optimization of Support Vector Machines. We could show that the dualization of convex training problems is possible theoretically in a rather general formulation. For training problems following a special structure (for instance, standard training problems) we found that the resulting optimality conditions can be interpreted concretely. This approach immediately led to the well-known notion of support vectors and a formulation of the Representer Theorem. The proposed theory was applied to several examples such that dual formulations of training problems and associated optimality conditions could be derived straightforwardly. Furthermore, we considered different formulations of the primal training problem which are equivalent under certain conditions. We also argued that the relation of the corresponding solutions to the solution of the dual training problem is not always intuitive. Based on the previous findings, we considered the application of customized optimization methods to the primal and dual training problems. A particular realization of Newton's method was derived which could be used to solve the primal training problem accurately. Moreover, we introduced a general convergence framework covering different types of decomposition methods for the solution of the dual training problem. In doing so, we were able to generalize well-known convergence results for the SMO method. Additionally, a discussion of the complexity of the SMO method and a motivation for a shrinking strategy reducing the computational effort is provided. In a last theoretical part, we considered the problem of hyperparameter optimization. We argued that this problem can be handled efficiently by means of gradient-based methods if the training problems are formulated appropriately. Finally, we evaluated the theoretical results concerning the training and hyperparameter optimization approaches practically by means of several example training problems.

Although we considered a multitude of examples and discussed several practical issues concerning the training and hyperparameter optimization for SVMs, there are still open problems. For instance, we could not investigate the application of the nonstandard training problems elaborately. In this context one could ask whether the solution of the proposed multi-class SVM training problems is practically feasible. Probably it is possible to derive a particular SMO-type method following the same strategy as in Chapter 7. Similarly, it would be interesting to consider the $\nu$-SVM training problem following the proposed framework which would lead to sparser solutions in
a certain sense. On the other hand, the efficiency of the proposed training methods could be investigated further. In particular, the theoretical shrinking approach should be evaluated in practice because there are still many open parameters to choose appropriately. Moreover, we could not thoroughly discuss the question when the switch from the dual SMO method to the primal Newton method should be done in order to reduce the training time efficiently. Lastly, the proposed hyperparameter optimization approach is only outlined sketchily. For instance, a generalization to other types of training problems would be possible without too much effort. In addition, it would be interesting to consider approaches which could lower the overall runtime of the procedure. A particular idea in this direction could be to apply a suitable sampling strategy instead of cross-validation. Furthermore, the behavior of the proposed method should be evaluated more extensively by means of other benchmark problems.

## A Definitions and Basic Theory

In this chapter, some basic definitions and assertions from the field of convex analysis are summarized. This summary is primarily here for recapitulation and reference. For a more detailed introduction we refer to the monographs [BZ05] and [BC11]. Throughout the whole chapter, we suppose that $\mathcal{H}$ be a real Banach space and $\mathcal{H}^{\star}$ the topological dual space of $\mathcal{H}$.

## A. 1 Convexity and Convex Subdifferential

We start with the well-known definition of a convex function and associated terms.

Definition A.1.1 (convex function; domain; proper function)
Let $C \subseteq \mathcal{H}$ be a convex set and let $f: C \rightarrow \mathbb{R} \cup\{+\infty\}$.

- The function $f$ is convex if

$$
f(t \boldsymbol{x}+(1-t) \boldsymbol{y}) \leq t f(\boldsymbol{x})+(1-t) f(\boldsymbol{y})
$$

for all $\boldsymbol{x}, \boldsymbol{y} \in C$ and all $t \in(0,1)$.

- The function $f$ is strictly convex if the inequality above is strictly satisfied for all $\boldsymbol{x}, \boldsymbol{y} \in C$ and all $t \in(0,1)$.
- The domain of $f$ is denoted by

$$
\operatorname{dom}(f):=\{\boldsymbol{x} \in C \mid f(\boldsymbol{x})<+\infty\} .
$$

- The function $f$ is proper if $\operatorname{dom}(f) \neq \varnothing$.

Convex functions play an important role in optimization because of many reasons. For instance, the lower level set $L_{f}(c):=\{\boldsymbol{x} \in \mathcal{H} \mid f(\boldsymbol{x}) \leq c\}$ is convex for each value $c \in \mathbb{R}$. This property helps to construct descent methods which are able to solve convex optimization problems globally.

One important object which is commonly associated with convex functions is the so-called convex subdifferential. It is defined as follows.

Definition A.1.2 (convex subdifferential)
Let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{+\infty\}$ be a proper function. We define the convex subdifferential of $f$ at $\boldsymbol{x} \in \mathcal{H}$ by

$$
\partial f(\boldsymbol{x}):=\left\{\boldsymbol{x}^{\star} \in \mathcal{H}^{\star} \mid f(\boldsymbol{y})-f(\boldsymbol{x}) \geq\left\langle\boldsymbol{x}^{\star}, \boldsymbol{y}-\boldsymbol{x}\right\rangle \text { for all } \boldsymbol{y} \in \mathcal{H}\right\} .
$$

The elements of $\partial f(\boldsymbol{x})$ are called subgradients of $f$ at $\boldsymbol{x}$.

Note that the convex subdifferential is also well-defined for non-convex functions. However, we restrict our investigation to convex function below. A consequence of the definition of the subdifferential is that is compatible with the usual notion of differentiability in the following sense. For the precise definition of the set core( $\operatorname{dom} f$ ) and Gâteaux differentiability used in the following proposition we refer to [BZ05, pages 114 and 121].

Proposition A.1.3 (differentiability of convex functions [BZO5, Corollary 4.2.9])
Let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{-\infty,+\infty\}$ be a convex function and let $\overline{\boldsymbol{x}} \in \operatorname{core}(\operatorname{dom} f)$. Then, $f$ is Gâteaux differentiable at $\overline{\boldsymbol{x}}$ if and only if $f$ has a unique subgradient at $\overline{\boldsymbol{x}}$ (in which case this subgradient is the derivative).

In fundamental calculus the notion of continuous functions is very important. If one considers functions which may attain infinite function values, continuity is not a suitable property anymore. Luckily, there is a straightforward generalization of continuity in this case which helps to transfer many helpful properties to extended real-valued functions. We define lower semi-continuity of functions as follows.

Definition A.1.4 (lower semi-continuous function)
The function $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{-\infty,+\infty\}$ is called lower semi-continuous if

$$
\liminf _{\boldsymbol{x} \rightarrow \overline{\boldsymbol{x}}} f(\boldsymbol{x}) \geq f(\overline{\boldsymbol{x}})
$$

for all $\overline{\boldsymbol{x}} \in \mathcal{H}$.

## A. 2 Convex Conjugates

Another fundamental object in convex analysis is the convex conjugate of a function. In particular, it will be used to define the dual optimization problem.

Definition A.2.1 (convex conjugate)
Let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{-\infty,+\infty\}$. The convex conjugate of $f$ is defined by

$$
\begin{aligned}
f^{\star}: \mathcal{H}^{\star} & \rightarrow \mathbb{R} \cup\{-\infty,+\infty\}, \\
\boldsymbol{x}^{\star} & \mapsto f^{\star}\left(\boldsymbol{x}^{\star}\right):=\sup _{\boldsymbol{x} \in \mathcal{H}}\left\{\left\langle\boldsymbol{x}^{\star}, \boldsymbol{x}\right\rangle-f(\boldsymbol{x})\right\} .
\end{aligned}
$$

Recall that the convex conjugate of a convex function is again convex. Furthermore, if the function $f$ is lower semi-continuous, the original function can be reconstructed from its conjugate, see [BL10, Theorem 4.2.1]. Thus, the conjugation is a bijective operator on the set of convex lower semi-continuous functions.

A fundamental property of the subdifferential which follows directly from its definition is the following.

Proposition A.2.2 (Fenchel-Young inequality [BL10, Proposition 3.3.4])
Let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{+\infty\}$. Suppose that $\boldsymbol{x}^{\star} \in \mathcal{H}^{\star}$ and $\boldsymbol{x} \in \operatorname{dom}(f)$. Then,

$$
f(\boldsymbol{x})+f^{\star}\left(\boldsymbol{x}^{\star}\right) \geq\left\langle\boldsymbol{x}^{\star}, \boldsymbol{x}\right\rangle .
$$

Equality holds if and only if $\boldsymbol{x}^{\star} \in \partial f(\boldsymbol{x})$.

As consequence of this inequality one can see that there is a close relationship between the subdifferential and the conjugate function as follows.

Proposition A. 2.3 (inverse of subdifferential, [BL10, Section 4.2, Exercise 7])
Let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{+\infty\}$ and suppose that $\boldsymbol{x} \in \mathcal{H}$ and $\boldsymbol{x}^{\star} \in \mathcal{H}^{\star}$ are points satisfying $\boldsymbol{x}^{\star} \in \partial f(\boldsymbol{x})$. Then, $\boldsymbol{x} \in \partial f^{\star}\left(\boldsymbol{x}^{\star}\right)$. Moreover, if $f$ is convex and lower semi-continuous, it follows that $\boldsymbol{x}^{\star} \in \partial f(\boldsymbol{x})$ if and only if $\boldsymbol{x} \in \partial f^{\star}\left(\boldsymbol{x}^{\star}\right)$.

## A. 3 Convex Duality

An immediate consequence of the Fenchel-Young inequality is the following notation of weak duality which is the starting point of the investigation of the dual optimization problem.

Theorem A.3.1 (weak duality [BZ05, Theorem 4.4.2])
Let $\mathcal{H}$ and $\mathcal{D}$ be Banach spaces, let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{-\infty,+\infty\}$ and $g: \mathcal{D} \rightarrow \mathbb{R} \cup\{-\infty,+\infty\}$ be convex functions and let $\boldsymbol{A}: \mathcal{H} \rightarrow \mathcal{D}$ be a bounded linear map. Then, the weak duality inequality

$$
\begin{equation*}
f(\boldsymbol{x})+g(\boldsymbol{A} \boldsymbol{x}) \geq-f^{\star}\left(\boldsymbol{A}^{\star} \boldsymbol{y}^{\star}\right)-g^{\star}\left(-\boldsymbol{y}^{\star}\right) \tag{A.3.1}
\end{equation*}
$$

is satisfied for all $\boldsymbol{x} \in \mathcal{H}$ and $\boldsymbol{y}^{\star} \in \mathcal{D}^{\star}$.

The following theorem shows how a dual optimization problem can be derived. It also provides necessary and sufficient optimality conditions which relates solutions of the primal and the dual optimization problem.

Theorem A.3.2 (convex duality [BP12, Theorem 3.53])
Let $\mathcal{H}$ and $\mathcal{D}$ be Banach spaces, let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{+\infty\}$ and $g: \mathcal{D} \rightarrow \mathbb{R} \cup\{+\infty\}$ be proper, convex and lower semi-continuous functions and $\boldsymbol{A}: \mathcal{H} \rightarrow \mathcal{D}$ be a bounded linear map. Suppose that there exists $\overline{\boldsymbol{x}} \in \mathcal{H}$ such that $f(\overline{\boldsymbol{x}})<+\infty$ and $g$ is continuous at $\boldsymbol{A} \overline{\boldsymbol{x}}$. Then, the optimal values of

$$
\begin{equation*}
\inf _{\boldsymbol{x} \in \mathcal{H}} f(\boldsymbol{x})+g(\boldsymbol{A} \boldsymbol{x}) \tag{A.3.2}
\end{equation*}
$$

and

$$
\begin{equation*}
\max _{\boldsymbol{y} \in \mathcal{D}^{\star}}-f^{\star}\left(\boldsymbol{A}^{\star} \boldsymbol{y}\right)-g^{\star}(-\boldsymbol{y}) \tag{A.3.3}
\end{equation*}
$$

coincide. In particular, the dual problem (A.3.3) has a solution.
Moreover, the following two properties are equivalent:
(i) $\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right) \in \mathcal{H} \times \mathcal{D}^{\star}$ is a couple of solutions for (A.3.2) and (A.3.3).
(ii) $\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right) \in \mathcal{H} \times \mathcal{D}^{\star}$ satisfies the system

$$
\boldsymbol{A}^{\star} \boldsymbol{y}^{\star} \in \partial f(\boldsymbol{x}) \quad \text { and } \quad-\boldsymbol{y}^{\star} \in \partial g(\boldsymbol{A} \boldsymbol{x}) .
$$

Consequently, a particular optimality measure can be defined as follows.

Definition A.3.3 (primal-dual optimality gap)
The term

$$
\Delta_{\mathrm{pd}}\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right):=f(\boldsymbol{x})+g(\boldsymbol{A} \boldsymbol{x})-\left(-f^{\star}\left(\boldsymbol{A}^{\star} \boldsymbol{y}^{\star}\right)-g^{\star}\left(-\boldsymbol{y}^{\star}\right)\right)
$$

is called the primal-dual optimality gap for the pair $\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right) \in \mathcal{H} \times \mathcal{D}^{\star}$.

By means of the weak duality theorem Theorem A.3.1, it is easy to see that

$$
\Delta_{\mathrm{pd}}\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right) \geq 0
$$

and equality holds if and only if $\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right)$ is a couple of solutions for (A.3.2) and (A.3.3). This means that $\Delta_{\mathrm{pd}}\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right)=0$ is a necessary and sufficient optimality condition. In particular, this fact can be used in the construction and investigation of optimization methods because it is rather easy to evaluate the primal-dual gap. We summarize the previous observation in the following proposition.

Proposition A.3.4 (optimality condition using primal-dual gap)
Suppose that the assumptions of Theorem A.3.2 are satisfied and let $\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right) \in \mathcal{H} \times \mathcal{D}^{\star}$. Then, $\boldsymbol{x}$ is a solution of (A.3.2) and $\boldsymbol{y}^{\star}$ is a solution of (A.3.3) if and only if $\Delta_{\mathrm{pd}}\left(\boldsymbol{x}, \boldsymbol{y}^{\star}\right)=0$.

## A. 4 Bregman Distance

For any proper convex function a distance measure can be defined as follows. Later on in this appendix, we will derive a relationship between this distance measure and the primal-dual optimality gap.

Definition A.4.1 (Bregman distance and directional derivative, c.f. [Res04]) Let $f: \mathcal{H} \rightarrow \mathbb{R} \cup\{+\infty\}$ be a proper convex function. The function

$$
\begin{aligned}
D_{f}: & \operatorname{dom} f \times \operatorname{dom} f \rightarrow \mathbb{R}_{+} \cup\{+\infty\} \\
& \quad(\boldsymbol{y}, \boldsymbol{x}) \mapsto D_{f}(\boldsymbol{y}, \boldsymbol{x}):=f(\boldsymbol{y})-f(\boldsymbol{x})-f^{\circ}(\boldsymbol{x}, \boldsymbol{y}-\boldsymbol{x})
\end{aligned}
$$

is called the Bregman distance (also: Bregman divergence) with respect to $f$. Here, $f^{\circ}(\boldsymbol{x}, \boldsymbol{z})$ denotes the directional derivative of $f$ at $\boldsymbol{x}$ in direction $\boldsymbol{z}$, i.e.,

$$
f^{\circ}(\boldsymbol{x}, \boldsymbol{z}):=\lim _{t \searrow 0} \frac{f(\boldsymbol{x}+t \boldsymbol{z})-f(\boldsymbol{x})}{t} .
$$

It is worth noting that the function $D_{f}$ is not a metric in general because it is not necessarily symmetric and can violate the triangle inequality. Nevertheless, it can be used as a distance measure in a certain way.

In the following theorem we propose a general distance estimate based on the notion of the primal-dual gap and the Bregman distance. In its proof only basic properties of the primal and dual optimization problem are used. Nevertheless, we are not aware of any publication containing this basic assertion.

Theorem A.4.2 (Bregman distance estimation)
Consider the pair of optimization problems (A.3.2) and (A.3.3) under the assumptions of Theorem A.3.2. Then, for all $\boldsymbol{x}, \overline{\boldsymbol{x}} \in \operatorname{dom}(f)$ and $\overline{\boldsymbol{y}} \in \mathcal{D}^{\star}$ satisfying $\boldsymbol{A}^{\star} \overline{\boldsymbol{y}} \in \partial f(\overline{\boldsymbol{x}})$ it holds that

$$
\begin{equation*}
D_{f}(\boldsymbol{x}, \overline{\boldsymbol{x}}) \leq \Delta_{\mathrm{pd}}(\boldsymbol{x}, \overline{\boldsymbol{y}}), \tag{A.4.1}
\end{equation*}
$$

Moreover, for all $\boldsymbol{x}, \overline{\boldsymbol{x}} \in \operatorname{dom}(f)$ and $\overline{\boldsymbol{y}} \in \mathcal{D}^{\star}$ satisfying $\boldsymbol{A} \boldsymbol{x} \in \operatorname{dom}(g)$, $\boldsymbol{A} \overline{\boldsymbol{x}} \in \operatorname{dom}(g)$ and $-\overline{\boldsymbol{y}} \in \partial g(\boldsymbol{A} \overline{\boldsymbol{x}})$ it holds that

$$
\begin{equation*}
D_{g}(\boldsymbol{A} \boldsymbol{x}, \boldsymbol{A} \overline{\boldsymbol{x}}) \leq \Delta_{\mathrm{pd}}(\boldsymbol{x}, \overline{\boldsymbol{y}}) . \tag{A.4.2}
\end{equation*}
$$

Proof. Let $\overline{\boldsymbol{x}} \in \mathcal{H}$ and $\overline{\boldsymbol{y}} \in \mathcal{D}^{\star}$ and suppose that $\boldsymbol{A}^{\star} \overline{\boldsymbol{y}} \in \partial f(\overline{\boldsymbol{x}})$. Then, from Proposition A.2.2 it follows that

$$
f(\overline{\boldsymbol{x}})+f^{\star}\left(\boldsymbol{A}^{\star} \overline{\boldsymbol{y}}\right)=\left\langle\boldsymbol{A}^{\star} \overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\right\rangle .
$$

Moreover, the definition of the subdifferential implies

$$
f^{\circ}(\overline{\boldsymbol{x}}, \boldsymbol{z}) \geq\left\langle\boldsymbol{A}^{\star} \overline{\boldsymbol{y}}, \boldsymbol{z}\right\rangle
$$

for all $\boldsymbol{z} \in \mathcal{H}$. Taking both parts together, we obtain

$$
\begin{aligned}
D_{f}(\boldsymbol{x}, \overline{\boldsymbol{x}}) & =f(\boldsymbol{x})-f(\overline{\boldsymbol{x}})-f^{\circ}(\overline{\boldsymbol{x}}, \boldsymbol{x}-\overline{\boldsymbol{x}}) \\
& \leq f(\boldsymbol{x})-f(\overline{\boldsymbol{x}})+\left\langle-\boldsymbol{A}^{\star} \overline{\boldsymbol{y}}, \boldsymbol{x}-\overline{\boldsymbol{x}}\right\rangle \\
& =f(\boldsymbol{x})+\left\langle-\boldsymbol{A}^{\star} \overline{\boldsymbol{y}}, \boldsymbol{x}\right\rangle-f(\overline{\boldsymbol{x}})+\left\langle\boldsymbol{A}^{\star} \overline{\boldsymbol{y}}, \overline{\boldsymbol{x}}\right\rangle \\
& =f(\boldsymbol{x})+\langle-\overline{\boldsymbol{y}}, \boldsymbol{A} \boldsymbol{x}\rangle+f^{\star}\left(\boldsymbol{A}^{\star} \overline{\boldsymbol{y}}\right)
\end{aligned}
$$

Furthermore, by means of the Fenchel-Young inequality (see Proposition A.2.2) it follows that

$$
\langle-\overline{\boldsymbol{y}}, \boldsymbol{A} \boldsymbol{x}\rangle \leq g(\boldsymbol{A} \boldsymbol{x})+g^{\star}(-\overline{\boldsymbol{y}})
$$

for all $\boldsymbol{x} \in \mathcal{H}$. Together with the inequality above, this shows that (A.4.1) holds. Using similar computations it can be shown that the inequality (A.4.2) is true.

Note that, under the assumptions of Theorem A.4.2 one can also show the corresponding dual assertions. We summarize these statements subsequently for sake of completeness.

Theorem A.4.3 (Bregman distance estimation, dual form)
Consider the pair of optimization problems (A.3.2) and (A.3.3) under the assumptions of Theorem A.3.2. Let $\overline{\boldsymbol{y}} \in \operatorname{dom}\left(g^{\star}\right)$ and $\overline{\boldsymbol{x}} \in \mathcal{H}$. Then, for all $\overline{\boldsymbol{x}} \in \mathcal{H}$ and $\boldsymbol{y}, \overline{\boldsymbol{y}} \in \operatorname{dom}\left(g^{\star}\right)$ satisfying $\boldsymbol{A} \overline{\boldsymbol{x}} \in \partial g^{\star}(-\overline{\boldsymbol{y}})$ it holds that

$$
D_{g^{\star}}(\boldsymbol{y}, \overline{\boldsymbol{y}}) \leq \Delta_{\mathrm{pd}}(\overline{\boldsymbol{x}}, \boldsymbol{y})
$$

Moreover, for all $\overline{\boldsymbol{x}} \in \mathcal{H}$ and $\boldsymbol{y}, \overline{\boldsymbol{y}} \in \operatorname{dom}\left(g^{\star}\right)$ satisfying $\boldsymbol{A}^{\star} \overline{\boldsymbol{y}} \in \operatorname{dom}\left(f^{\star}\right), \boldsymbol{A}^{\star} \boldsymbol{y} \in \operatorname{dom}\left(f^{\star}\right)$ and $\overline{\boldsymbol{x}} \in \partial f^{\star}\left(\boldsymbol{A}^{\star} \overline{\boldsymbol{y}}\right)$ it holds that

$$
D_{f^{\star}}\left(\boldsymbol{A}^{\star} \boldsymbol{y}, \boldsymbol{A}^{\star} \overline{\boldsymbol{y}}\right) \leq \Delta_{\mathrm{pd}}(\overline{\boldsymbol{x}}, \boldsymbol{y}) .
$$

## A. 5 Strict-Smooth Duality

As a last part of this chapter, we want to present an assertion which connects convexity and differentiability of a function and its conjugate, respectively. To be precise, we use the terms of essential strict convexity and essential smoothness which are basically restrictions of well-known properties to the domain of a convex function. To start with, consider essentially strictly convex functions.

Definition A.5.1 (essentially strictly convex, cf. [BL10, page 35])
A convex function $f$ is called essentially strictly convex if it is strictly convex on any convex subset of $\operatorname{dom}(\partial f)$.

Moreover, let essentially smoothness be defined as follows.

Definition A.5.2 (essentially smooth, cf. [BL10, page 37])
A convex function $f$ is called essentially smooth if it is Gâteaux differentiable on $\operatorname{dom}(\partial f)$.

With the terms of essential strict convexity and essential smoothness, the connection between convexity and differentiability of the conjugate functions can be stated as follows.

Theorem A.5.3 (strict-smooth duality, [BL10, Theorem 4.2.5])
A proper closed convex function is essentially strictly convex if and only if its conjugate is essentially smooth.

The basic consequence of the previous assertion is that smoothness of the primal objective function translates into extra convexity of the dual objective function. This property on its part helps to conclude that the dual problem has a unique solution in particular settings. On the other hand, strict convexity in the primal problem immediately implies a certain smoothness of the objective function in the dual problem.

## Bibliography

[Ale19] Alen Alexanderian. Optimization in infinite-dimensional Hilbert spaces https://aalexan3.math.ncsu.edu/articles/hilbert.pdf. North Carolina State University, Raleigh, NC, USA, 2019.
[BC11] Heinz H. Bauschke and Patrick L. Combettes. Convex Analysis and Monotone Operator Theory in Hilbert Spaces. CMS Books in Mathematics. Springer New York, 2011.
[BGV92] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A Training Algorithm for Optimal Margin Classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92, page 144-152, New York, NY, USA, 1992. Association for Computing Machinery.
[BH14] Radu loan Boţ and André Heinrich. Regression tasks in machine learning via Fenchel duality. Annals of Operations Research, 222:197-211, 2014.
$\left[B H J^{+} 06\right] \quad$ Kristin P. Bennett, Jing Hu, Xiaoyun Ji, Gautam Kunapuli, and Jong-Shi Pang. Model Selection via Bilevel Optimization. In The 2006 IEEE International Joint Conference on Neural Network Proceedings, pages 19221929, 2006.
[Bis06] Christopher. M. Bishop. Pattern Recognition and Machine Learning. Information Science and Statistics. Springer New York, 2006.
[BL10] Jonathan Borwein and Adrian Lewis. Convex Analysis and Nonlinear Optimization: Theory and Examples. CMS Books in Mathematics. Springer New York, 2010.
[BM98] Paul S. Bradley and Olvi L. Mangasarian. Feature Selection via Concave Minimization and Support Vector Machines. In Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, page 82-90, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.
[BP12] Viorel Barbu and Teodor Precupanu. Convexity and Optimization in Banach Spaces. Springer Monographs in Mathematics. Springer Dordrecht, 2012.
[BRG19] Bastian Bohn, Christian Rieger, and Michael Griebel. A Representer Theorem for Deep Kernel Learning. Journal of Machine Learning Research, 20(64):1-32, 2019.
[BS96] Christopher J. C. Burges and Bernhard Schölkopf. Improving the Accuracy and Speed of Support Vector Machines. In M.C. Mozer, M. Jordan, and T. Petsche, editors, Advances in Neural Information Processing Systems, volume 9. MIT Press, 1996.
[Bur96] Christopher J. C. Burges. Simplified Support Vector Decision Rules. In Proceedings of the Thirteenth International Conference on International Conference on Machine Learning, ICML'96, page 71-77, San Francisco, CA, USA, 1996. Morgan Kaufmann Publishers Inc.
[Bur98] Christopher J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery, 2(2):121167, 1998.
[BZ05] Jonathan M. Borwein and Qiji J. Zhu. Techniques of Variational Analysis. CMS Books in Mathematics. Springer New York, 2005.
[CFL06] Pai-Hsuen Chen, Rong-En Fan, and Chih-Jen Lin. A Study on SMO-Type Decomposition Methods for Support Vector Machines. IEEE Transactions on Neural Networks, 17(4):893-908, 2006.
[CL01] Chih-Chung Chang and Chih-Jen Lin. Training $\nu$-Support Vector Classifiers: Theory and Algorithms. Neural Computation, 13(9):2119-2147, sep 2001.
[CLO2] Chih-Chung Chang and Chih-Jen Lin. Training $\nu$-Support Vector Regression: Theory and Algorithms. Neural Computation, 14(8):19591977, 2002.
[CL05] Ming-Wei Chang and Chih-Jen Lin. Leave-One-Out Bounds for Support Vector Regression Model Selection. Neural Computation, 17(5):1188-1222, 2005.
[CL11] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A Library for Support Vector Machines. ACM transactions on intelligent systems and technology (TIST), 2(3):1-27, 2011.
[CP00] Gert Cauwenberghs and Tomaso Poggio. Incremental and Decremental Support Vector Machine Learning. In Proceedings of the 13th International Conference on Neural Information Processing Systems, NIPS'00, page 388-394, Cambridge, MA, USA, 2000. MIT Press.
[CV95] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. Machine Learning, 20:273-297, 1995.
[CV99] Olivier Chapelle and Vladimir Vapnik. Model Selection for Support Vector Machines. In S. Solla, T. Leen, and K. Müller, editors, Advances in Neural Information Processing Systems, volume 12. MIT Press, 1999.
[CVBM02] Olivier Chapelle, Vladimir Vapnik, Olivier Bousquet, and Sayan Mukherjee. Choosing Multiple Parameters for Support Vector Machines. Machine Learning, 46(1-3):131-159, 2002.
[Dan70] James W. Daniel. Convergent Step-Sizes for Gradient-Like Feasible Direction Algorithms for Constrained Optimization. In J. B. Rosen, O. L. Mangasarian, and K. Ritter, editors, Nonlinear Programming, pages 245-274. Academic Press, 1970.
[DFN07] Chuong B. Do, Chuan-Sheng Foo, and Andrew Y. Ng. Efficient Multiple Hyperparameter Learning for Log-Linear Models. In Proceedings of the 20th International Conference on Neural Information Processing Systems, NIPS'07, page 377-384, Red Hook, NY, USA, 2007. Curran Associates Inc.
[FCH ${ }^{+}$08] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and ChihJen Lin. LIBLINEAR: A Library for Large Linear Classification. Journal of Machine Learning Research, 9:1871-1874, jun 2008.
[FCL05] Rong-En Fan, Pai-Hsuen Chen, and Chih-Jen Lin. Working Set Selection Using Second Order Information for Training Support Vector Machines. Journal of Machine Learning Research, 6:1889-1918, dec 2005.
[FFK98] Francisco Facchinei, Andreas Fischer, and Christian Kanzow. On the Accurate Identification of Active Constraints. SIAM Journal on Optimization, 9(1):14-32, 1998.
[FLL+15] Andreas Fischer, Gerd Langensiepen, Klaus Luig, Nico Strasdat, and Thorsten Thies. Efficient Optimization of Hyper-Parameters for Least Squares Support Vector Regression. Optimization Methods and Software, 30(6):1095-1108, 2015.
[FM03] Glenn Fung and O. L. Mangasarian. Finite Newton method for Lagrangian support vector machine classification. Neurocomputing, 55(1):39-55, 2003.
[FM04] Glenn M. Fung and Olvi L. Mangasarian. A Feature Selection Newton Method for Support Vector Machine Classification. Computational Optimization and Applications, 28(2):185-202, 2004.
[GK99] Carl Geiger and Christian Kanzow. Numerische Verfahren zur Lösung unrestringierter Optimierungsaufgaben. Springer-Lehrbuch. Springer Berlin Heidelberg, 1999.
[GU17] Jun-ya Gotoh and Stan Uryasev. Support vector machines based on convex risk functions and general norms. Annals of Operations Research, 249:301-328, 2017.
[HJ12] Roger A. Horn and Charles R. Johnson. Matrix Analysis. Cambridge University Press, 2012.
[HMvdW+20] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian TayIor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre GérardMarchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. Nature, 585(7825):357-362, September 2020.
[Hub64] Peter J. Huber. Robust Estimation of a Location Parameter. The Annals of Mathematical Statistics, 35(1):73-101, 1964.
[JS20] Wei Jiang and Sauleh Siddiqui. Hyper-parameter optimization for support vector machines using stochastic gradient descent and dual coordinate descent. EURO Journal on Computational Optimization, 8(1):85-101, 2020.
[KBHP08] Gautam Kunapuli, Kristin P. Bennett, Jing Hu, and Jong-Shi Pang. Bilevel Model Selection for Support Vector Machines. In CRM Proceedings and Lecture Notes, volume 45, 2008.
[KSBM01] S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karuturi Radha Krishna Murthy. Improvements to Platt's SMO algorithm for SVM classifier design. Neural Computation, 13(3):637649, 2001.
[KSC06] Sathiya Keerthi, Vikas Sindhwani, and Olivier Chapelle. An Efficient Method for Gradient-Based Adaptation of Hyperparameters in SVM Models. Advances in Neural Information Processing Systems, 19, 2006.
[KW70] George S. Kimeldorf and Grace Wahba. A Correspondence Between Bayesian Estimation on Stochastic Processes and Smoothing by Splines. The Annals of Mathematical Statistics, 41 (2):495-502, 1970.
[LL10] R. De Leone and C. Lazzari. Error bounds for support vector machines with application to the identification of active constraints. Optimization Methods and Software, 25(2):185-202, 2010.
[LLZ22] Qingna Li, Zhen Li, and Alain Zemkoho. Bilevel hyperparameter optimization for support vector classification: theoretical analysis and a solution method. Mathematical Methods of Operations Research, pages 1-36, 2022.
[LM01] Yuh-Jye Lee and Olvi L Mangasarian. SSVM: A Smooth Support Vector Machine for Classification. Computational Optimization and Applications, 20(1):5-22, 2001.
[Man65] Olvi L. Mangasarian. Linear and Nonlinear Separation of Patterns by Linear Programming. Operations Research, 13(3):444-452, 1965.
[Man68] Olvi L. Mangasarian. Multisurface method of pattern separation. IEEE Transactions on Information Theory, 14(6):801-807, 1968.
[MB02] Michinari Momma and Kristin P. Bennett. A Pattern Search Method for Model Selection of Support Vector Regression. In Proceedings of the 2002 SIAM International Conference on Data Mining, pages 261-274. SIAM, 2002.
[MBB11] Gregory Moore, Charles Bergeron, and Kristin P. Bennett. Model selection for primal SVM. Machine Learning, 85:175-208, 2011.
[MM01] Olvi L. Mangasarian and David R. Musicant. Lagrangian Support Vector Machines. Journal of Machine Learning Research, 1(Mar):161-177, 2001.
[NW06] Jorge Nocedal and Stephen J. Wright. Numerical Optimization. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006.
[OFG97] Edgar Osuna, Robert Freund, and Federico Girosi. An improved training algorithm for support vector machines. In Neural networks for signal processing VII. Proceedings of the 1997 IEEE signal processing society workshop, pages 276-285. IEEE, 1997.
[PCNVADAR00] Fernando Pérez-Cruz, A. Navia-Vázquez, Pedro Luis Alarcón-Diana, and Antonio Artés-Rodríguez. An IRWLS procedure for SVR. In 2000 10th European Signal Processing Conference, pages 1-4. IEEE, 2000.
[Pla99] John C. Platt. Fast Training of Support Vector Machines Using Sequential Minimal Optimization. In Advances in Kernel Methods: Support Vector Learning, page 185-208, Cambridge, MA, USA, 1999. MIT Press.
[PS05] Laura Palagi and Marco Sciandrone. On the Convergence of a Modified Version of SVM Light Algorithm. Optimization Methods and Software, 20(2-3):317-334, 2005.
[Res04] Elena Resmerita. On total convexity, Bregman projections and stability in Banach spaces. Journal of Convex Analysis, 11(1):1-16, 2004.
[RL07] Ryan M. Rifkin and Ross A. Lippert. Value Regularization and Fenchel Duality. The Journal of Machine Learning Research, 8:441-479, dec 2007.
[Sch05] Klaus Schittkowski. Optimal Parameter Selection in Support Vector Machines. Journal of Industrial and Management Optimization, 1(4):465-476, 2005.
[SDBLV02] Johan A. K. Suykens, Jos De Brabanter, Lukas Lukas, and Joos Vandewalle. Weighted Least Squares Support Vector Machines: robustness and sparse approximation. Neurocomputing, 48(1-4):85-105, 2002.
[SGV98] Craig Saunders, Alexander Gammerman, and Volodya Vovk. Ridge Regression Learning Algorithm in Dual Variables. In Proceedings of the Fifteenth International Conference on Machine Learning, ICML '98, page 515-521, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.
[SHS01] Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A Generalized Representer Theorem. In David Helmbold and Bob Williamson, editors, Computational Learning Theory, pages 416-426. Springer Berlin Heidelberg, 2001.
[SS02] Bernhard Schölkopf and Alexander J. Smola. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. Adaptive Computation and Machine Learning. MIT Press, 2002.
[SS04] Alex J. Smola and Bernhard Schölkopf. A tutorial on support vector regression. Statistics and Computing, 14(3):199-222, 2004.
[SSM98] Alex J. Smola, Bernhard Schölkopf, and Klaus-Robert Müller. General cost functions for support vector regression. In Ninth Australian Conference on Neural Networks, pages 79-83, 1998.
[STC04] John Shawe-Taylor and Nello Cristianini. Kernel Methods for Pattern Analysis. Kernel Methods for Pattern Analysis. Cambridge University Press, 2004.
[SV99] Johan A. K. Suykens and Joos Vandewalle. Least Squares Support Vector Machine Classifiers. Neural Processing Letters, 9(3):293-300, 1999.
[SVGDB $\left.{ }^{+} 02\right]$ Johan A. K. Suykens, Tony Van Gestel, Jos De Brabanter, Bart De Moor, and Joos Vandewalle. Least Squares Support Vector Machines. World Scientific, 2002.
[SWS ${ }^{+99] ~ B e r n h a r d ~ S c h o ̈ l k o p f, ~ R o b e r t ~ W i l l i a m s o n, ~ A l e x ~ S m o l a, ~ J o h n ~ S h a w e-~}$ Taylor, and John Platt. Support Vector Method for Novelty Detection. In Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS'99, page 582-588, Cambridge, MA, USA, 1999. MIT Press.
[ÜMOB05] B. Üstün, W.J. Melssen, M. Oudenhuiizen, and L. M. C. Buydens. Determination of optimal support vector regression parameters by genetic
algorithms and simplex optimization. Analytica Chimica Acta, 544(1-2):292-305, 2005.
[Vap63] Vladimir Vapnik. Pattern recognition using generalized portrait method. Automation and Remote Control, 24:774-780, 1963.
[Vap95] Vladimir N. Vapnik. The Nature of Statistical Learning Theory. Information Science and Statistics. Springer New York, 1995.
[Vap98] Vladimir N. Vapnik. Statistical Learning Theory. Adaptive and Cognitive Dynamic Systems: Signal Processing, Learning, Communications and Control. Wiley \& Sons Ltd, 1998.
[Vap06] Vladimir Vapnik. Estimation of Dependences Based on Empirical Data: Empirical Inference Science. Information Science and Statistics. Springer, Berlin, Heidelberg, 2006.
[VGO+20] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, ilhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17:261272, 2020.
[VGS96] Vladimir Vapnik, Steven E. Golowich, and Alex Smola. Support Vector Method for Function Approximation, Regression Estimation and Signal Processing. In Proceedings of the 9th International Conference on Neural Information Processing Systems, NIPS'96, page 281-287, Cambridge, MA, USA, 1996. MIT Press.
[WEST03] Jason Weston, André Elisseeff, Bernhard Schölkopf, and Mike Tipping. Use of the Zero-Norm with Linear Models and Kernel Methods. Journal of Machine Learning Research, 3(Mar):1439-1461, 2003.
[Zho18] Xingyu Zhou. On the Fenchel duality between strong convexity and Lipschitz continuous gradient. arXiv preprint arXiv:1803.06573, 2018.

## Versicherung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Die vorliegende Dissertation habe ich am Institut für Numerische Mathematik der Technischen Universität Dresden unter der wissenschaftlichen Betreuung von Prof. Dr. Andreas Fischer angefertigt.

Es wurden zuvor keine Promotionsvorhaben unternommen.
Ich erkenne die Promotionsordnung des Bereichs Mathematik und Naturwissenschaften der Technischen Universität Dresden vom 23.02.2011, in der geänderten Fassung vom 23.05.2018, an.

Dresden, den

Nico Strasdat

