Multi-task Learning with Gaussian Processes

Kian Ming Adam Chai

Doctor of Philosophy

Institute for Adaptive and Neural Computation

School of Informatics

University of Edinburgh

2010

Abstract

Multi-task learning refers to learning multiple tasks *simultaneously*, in order to avoid *tabula rasa* learning and to share information between similar tasks during learning. We consider a multi-task Gaussian process regression model that learns related functions by inducing correlations between tasks *directly*. Using this model as a reference for three other multi-task models, we provide a broad unifying view of multi-task learning. This is possible because, unlike the other models, the multi-task Gaussian process model encodes task relatedness explicitly.

Each multi-task learning model generally assumes that learning multiple tasks together is beneficial. We analyze how and the extent to which multi-task learning helps improve the generalization of supervised learning. Our analysis is conducted for the average-case on the multi-task Gaussian process model, and we concentrate mainly on the case of two tasks, called the primary task and the secondary task. The main parameters are the degree of relatedness ρ between the two tasks, and π_S, the fraction of the total training observations from the secondary task. Among other results, we show that asymmetric multi-task learning, where the secondary task is to help the learning of the primary task, can decrease a lower bound on the average generalization error by a factor of up to $\rho^2\pi_S$. When there are no observations for the primary task, there is also an intrinsic limit to which observations for the secondary task can help the primary task. For symmetric multi-task learning, where the two tasks are to help each other to learn, we find the learning to be characterized by the term $\pi_S(1-\pi_S)(1-\rho^2)$. As far as we are aware, our analysis contributes to an understanding of multi-task learning that is orthogonal to the existing PAC-based results on multi-task learning. For more than two tasks, we provide an understanding of the multi-task Gaussian process model through structures in the predictive means and variances given certain configurations of training observations. These results generalize existing ones in the geostatistics literature, and may have practical applications in that domain.

We evaluate the multi-task Gaussian process model on the inverse dynamics problem for a robot manipulator. The inverse dynamics problem is to compute the torques needed at the joints to drive the manipulator along a given trajectory, and there are advantages to learning this function for adaptive control. A robot manipulator will often need to be controlled while holding different loads in its end effector, giving rise to a multi-context or multi-load learning problem, and we treat predicting the inverse dynamics for a context/load as a task. We view the learning of the inverse dynamics as a function approximation problem and place Gaussian process priors over the space of functions. We first show that this is effective for learning the inverse dynamics for a single context. Then, by placing independent Gaussian process priors over the latent functions of the inverse dynamics, we obtain a multi-task Gaussian process prior for handling multiple loads, where the inter-context similarity depends on the underlying inertial parameters of the manipulator. Experiments demonstrate that this multi-task formulation is effective in sharing information among the various loads, and generally improves performance over either learning only on single contexts or pooling the data over all contexts. In addition to the experimental results, one of the contributions of this study is showing that the multi-task Gaussian process model follows naturally from the physics of the inverse dynamics.

Acknowledgements

It is my fortunate to have Chris Williams as my principal supervisor, and I thank him for his constant guidance and support. His knowledge and eye for details have helped me avoided a number of pitfalls, and his sharp intuition has saved me a number of needless paths.

I thank Edwin Bonilla for the many interesting and inspiring discussions that we have had, and for sharing his experience with multi-task Gaussian processes. I am grateful to Sethu Vijayakumar for proposing the multiple context inverse dynamics problem, and to Stefan Klanke for the experiments and the detailed discussions on this problem. I have also learnt from Amos Storkey during the weekly brainstorm meetings, which has encouraged me to think out of the box. Thanks to Iain Murray and Manfred Opper for their reviews and comments on this work.

It has been a pleasure be among the members of the machine learning group, including Felix Agakov, John Quinn, Lawrence Murray, Nicolas Heess, Andrew Dai and Athina Spiliopoulou. I have benefited from discussions with and comments by them.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Kian Ming Adam Chai*)

In memory of Grandma and Ninn

Contents

List of Figures

List of Tables

Notation

In general, capital letters denote matrices and lower case bold letters denote vectors. All vectors are column vectors.

Symbol	Meaning
Spaces and Sets	
$\{a_i\}_{i=1}^n$	the set $\{a_1, \dots, a_n\}$; the subscript and superscript may be suppressed
$\mathbb{N}_0$	the set of natural numbers, starting from 0; $\{0, 1, 2, \dots\}$
$\mathbb{N}_1$	the set of natural numbers, starting from 1; $\{1, 2, 3, \dots\}$
$\mathbb{R}^n$	space of n-dimensional real vectors
$\mathbb{R}^{n \times m}$	space of n-by-m real matrices
Δ^n	standard n-simplex; $\Delta^n \overset{\text{def}}{=} \big\{(t_0, \cdots, t_n) \in \mathbb{R}^{n+1} \mid \sum_i t_i = 1$ and $t_i \geq 0\ \forall i\big\}$.
Matrices and Vectors	
$\operatorname{tr}(A)$	trace of matrix A
$\operatorname{rank}(A)$	rank of matrix A
$\lvert A \rvert$	determinant of matrix A
A^{-1}	matrix inverse of A
A^{+}	the Moore-Penrose pseudo-inverse of A
$\boldsymbol{a}^{\mathrm{T}}$	transpose of vector $\boldsymbol{a}$
A^{T}	transpose of matrix A
$A \otimes B$	the Kronecker product of matrices A and B
$A \odot B$	the Hadamard product of matrices A and B
$\mathbf{0}$	vector of zeros
$I_{n \times n}, I_n$	n-by-n identity matrix ; subscripts may be suppressed
$0_{n \times n}, 0_n$	n-by-n matrix of zeros; subscripts may be suppressed
$1_{n \times n}, 1_n$	n-by-n matrix of ones; subscripts may be suppressed
Probability and Statistics	
$\langle \cdot \mid y \rangle_x, \mathbb{E}_x(\cdot \mid y)$	expectation with respect to a distribution on x given y; random variable x is usually clear from context and suppressed
$\mathbb{C}(\cdot, \cdot \vert y)$	covariance conditioned on y
$\mathcal{N}(\boldsymbol{\mu}, \Sigma)$	multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance Σ.
$\mathcal{L}aplace(\mu, v)$	the Laplace distribution with location μ and scale v
$\mathcal{M}ultinomial(n, \boldsymbol{\pi})$	multinomial distribution with n trials and probability mass given by $\boldsymbol{\pi}$

Symbol	Meaning
$\mathcal{B}eta(\alpha, \beta)$	the Beta distribution with parameters α and β
$\mathcal{D}ir(\alpha^1, \ldots, \alpha^P)$	the Dirichlet distribution with parameters $\alpha^1, \ldots, \alpha^P$
$a \sim p(\cdot \mid \cdots)$	a is a random variable, vector or matrix with distribution $p(\cdots)$
Functions and Variables	
δ_{ij}	the Kronecker delta function, 1 if $i = j$ and 0 otherwise
$k^{\mathrm{x}}(\cdot, \cdot)$	covariance function for the input space
$k^{\mathrm{f}}(\cdot, \cdot)$	covariance function for the task space
K^{f}	covariance matrix for the task space
ρ	correlation between two tasks
σ_{n}^2	observation noise variance
M	number of tasks
P	number of components or clusters in a mixture model; in general, the number of parts into which the system can be decomposed.
m, m'	indices that iterate over tasks
p, p'	indices that iterate over components, clusters or parts
Order Relations	
$f(n) \sim g(n)$	f is equal to g asymptotically; $\forall \varepsilon > 0 \; \exists n_0 \; \forall n > n_0 \; \lvert f(n)/g(n) - 1 \rvert < \varepsilon$
$f(n) \in O(g(n))$	f is bounded above by g (up to constant factor) asymptotically; $\exists k > 0, n_0 \; \forall n > n_0 \; \lvert f(n) \rvert \leqslant \lvert g(n) \cdot k \rvert$
$f(n) \in \Omega(g(n))$	f is bounded below by g (up to constant factor) asymptotically; $\exists k > 0, n_0 \; \forall n > n_0 \; \lvert g(n) \cdot k \rvert \leqslant \lvert f(n) \rvert$
$f(n) \in \Theta(g(n))$	f is bounded both above and below by g asymptotically; $\exists k_1, k_2 > 0, n_0 \; \forall n > n_0 \; \lvert g(n) \cdot k_1 \rvert < \lvert f(n) \rvert < \lvert g(n) \cdot k_2 \rvert$
$f(n) \in o(g(n))$	f is dominated by g asymptotically; $\forall \varepsilon > 0 \; \exists n_0 \; \forall n > n_0 \; \lvert f(n) \rvert \leq \lvert g(n) \cdot \varepsilon \rvert$

Chapter 1

Introduction

Machine learning has been applied in many areas to learn automatically from data in order to reduce the need for hand-coded rules. Typically, and this is the case here, a machine learning task is of a *supervised* nature, where a mapping $f : \mathcal{X} \mapsto \mathcal{Y}$ is learned from examples $\{(\boldsymbol{x}_i, y_i) \mid \boldsymbol{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$. A successful application to learn f requires imposing an appropriate *bias* in the hypothesis space [Mitchell, 1991] through selecting the learning algorithm, the parameters for that algorithm and the effective features for the domain under consideration. When embedded within an environment of multiple tasks, it may be beneficial to avoid *tabula rasa* learning by having the bias include information present in similar tasks and experiences from learning them. This is motivated by one fundamental aspect of human and animal learning: that knowledge and experiences gained from solving previously encountered tasks are used to guide the approach to a new task.

This is the basis for exploring how learning can occur across tasks in additional to across examples, and a variety of terms describe its research: *meta-learning*, *multi-task learning*, *life-long learning*, *learning to learn*, *inductive bias learning* etc.; for overviews, see Gordon and desJardins [1995]; Thrun and Pratt [1998, chapter 1]; Vilalta and Drissi [2002]; Vilalta et al. [2005]; Giraud-Carrier et al. [2004a]; Anderson and Oates [2007]. As acknowledged by Vilalta and Drissi [2002], different research groups interpret the various terms differently; in addition, these terms are sometimes used interchangeably [Anderson and Oates, 2007]. For the purpose of this thesis, it will be useful to clarify what some of these mean to us:

Meta-learning is an overarching term that encompasses the entire field of exploring and understanding how learning itself can be flexible by taking into account the multitude of tasks in the environment wherein a learning agent is immersed. In particular, in addition to improving the learning of a selected predictive model over many tasks, it aims to also address the practical and important application to automatically select a predictive model for a given task [Gordon and desJardins, 1995]; a brief review is given in section 2.7.1.

Multi-task learning refers to learning multiple tasks *simultaneously*. The transfer of information between tasks is usually eased by learning the tasks with similar machine learning models or algorithms under common input and output representations. The aim may be to increase the performance on all tasks, or it may be to increase the performance on one task of primary interest given other tasks of secondary interests. To differentiate between these two cases of multi-task learning, we borrow from Xue et al. [2007b] and call them *symmetric* and *asymmetric* respectively. In the machine learning literature, it is more common to find symmetric multi-task learning. In contrast, asymmetric multi-task learning is more prevalent in the geostatistics community, where the secondary tasks pertain to information more readily available than the primary task. For example, the primary task may be to predict the concentration of a precious metal, while the secondary tasks provide the concentrations of common metals.

Learning to learn refers to the property that the performance of an algorithm on each task improves with both the number of task-specific training examples and the number tasks [Thrun and Pratt, 1998, chapter 1]. The distinction between this and meta-learning is that it is usually limited to one machine learning algorithm instead of an array of algorithms. It is more general than multi-task learning in that it also includes the case where a bias is learnt from multiple tasks with the aim of using it to obtain good solutions to novel tasks. One form of such bias may be invariances within the domain [Thrun and Mitchell, 1995]. When the bias is learnt using multi-task learning, this is also called *asymmetric multi-task learning* by Xue et al. [2007b].

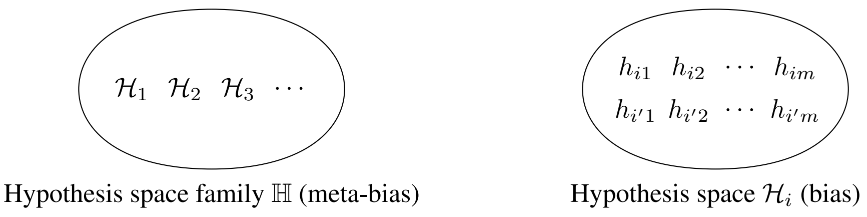
The focus of this thesis is to understand, implement and evaluate an effective multi-task learning model. This will be further elaborated in section 1.3. For the moment, let us broadly survey some existing views on why multi-task learning can be desirable.

1.1 Theoretical and Empirical Benefits

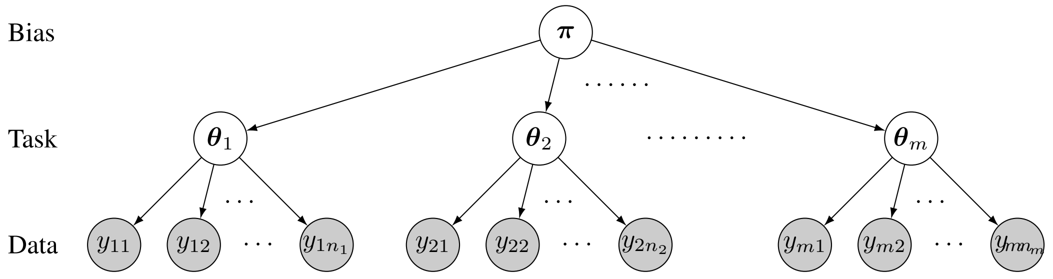
Vilalta and Drissi [2002] note that meta-learning cannot escape the need for some form of bias, since the various no-free lunch theorems, for example, Schaffer [1994] and Wolpert [2001], are also applicable at the “meta” level. One aspect of this “meta-bias” is to determine which information will be shared among or transferred across tasks and which will be specified to each task. It is this aspect that most differentiates between the different models of meta-learning, and it is associated to task relatedness, which will be addressed in section 1.2.

It may feel like going in circles because the need for a bias has been replaced by the need for a “meta-bias” or even a “meta-meta-bias” [Vilalta and Drissi, 2001; Baxter, 1997, 2000]. Nevertheless it has been shown theoretically and empirically that there are certain advantages in multi-task learning.

Theoretical aspects of multi-task learning have been explored by Baxter [1997, 2000] who considers a model for learning the appropriate bias through sampling multiple (say M) independently and identically distributed (i.i.d.) tasks. For *p*robably *a*pproximately *c*orrect (PAC) learning [Valiant, 1984], this means choosing a hypothesis space within a hypothesis space family. Each learning task will then select its own hypothesis from within the chosen hypothesis space (see Figure 1.1a). In the Bayesian framework, this is a hierarchical model where each task is sampled independently from the same prior. Multi-task learning is then placing a distribution and performing probabilistic inference on this prior (see Figure 1.1b). For PAC learning, Baxter [2000] shows that, to achieve a fixed task-averaged generalization error (i.e., the case of symmetric multi-task learning) with high probability, the upper bound on the number of examples needed per task *at best* is inversely proportional to M and *at worst* remains the


$\mathcal{H}_1$ $\mathcal{H}_2$ $\mathcal{H}_3$ $\cdots$
h_{i1} h_{i2} $\cdots$ h_{im}
$h_{i'1}$ $h_{i'2}$ $\cdots$ $h_{i'm}$
Hypothesis space family $\mathbb{H}$ (meta-bias)
Hypothesis space $\mathcal{H}_i$ (bias)

(a) PAC learning model using hypothesis spaces. Data from task j may be explained by hypothesis h_{ij}, which is an element of the hypothesis space $\mathcal{H}_i$, which in turn belongs to the hypothesis space family $\mathbb{H}$. Multi-task learning involves choosing a $\mathcal{H}_i$ from $\mathbb{H}$ so as to minimize the empirical error measure.


Bias
Task
Data
$\boldsymbol{\pi}$
$\boldsymbol{\theta}_1$
$\boldsymbol{\theta}_2$
$\boldsymbol{\theta}_m$
y_{11} y_{12} $\cdots$ y_{1n_1}
y_{21} y_{22} $\cdots$ y_{2n_2}
y_{m1} y_{m2} $\cdots$ y_{mn_m}

(b) Graphical representation of the hierarchical Bayes model. Task i is determined by $\boldsymbol{\theta}_i$, and generates data according to $p(y_{ij}|\boldsymbol{\theta}_i)$. The $\boldsymbol{\theta}_i$ for task i is in turn sampled from the task-prior $p(\boldsymbol{\theta}_i|\boldsymbol{\pi})$. Multi-task learning involves a hyper-prior $p(\boldsymbol{\pi})$ and performing probabilistic inference on $\boldsymbol{\pi}$. The meta-bias is implicit through the distribution on $\boldsymbol{\pi}$.

Figure 1.1: The (a) PAC and (b) Bayesian learning of multiple independent and identically sampled tasks.

same. For the hierarchical Bayes model, similar conclusions hold for the expected amount of information required to achieve good generalization [Baxter, 1997]. An intuitive interpretation for these results is the following: in the best case when all the tasks are closely related, then for any given task, the examples for the other tasks are highly relevant, so each task has in effect roughly M times the number of its examples, and the number of examples needed per task decreases as $O(1/M)$; in the worst case when the tasks are unrelated, then the effective number of examples for each task is the same as when learnt in isolation.

In addition to results similar to those of Baxter's, Ando and Zhang [2004] also argue that learning with multiple tasks can improve statistical estimates of task parameters over learning each task separately. Essentially, this is a statement of Stein's phenomenon [Stein, 1956], which says that the combined estimator of three or more parameters is more accurate than any method that handles the parameters separately. More recent theoretical results of similar nature are given by Maurer [2005, 2006]. Ben-David and Schuller Borbely [2008][1] later strengthen Baxter's PAC learning result using a specific notion of multi-task learning (see section 1.2) so that it also holds for the generalization error for each task, which is the error measure relevant to asymmetric multi-task learning. In chapter 3, we will provide an analysis of multi-task learning that is different from these mentioned here.

These theoretical results suggest that learning with multiple tasks is especially useful in domains when

[1] A preliminary version is Ben-David and Schuller [2003].

the number of examples per task is small (or the examples are sparse within the entire input space), but when the number of tasks can be large.

Empirically in many areas, learning with multiple tasks has been shown to perform better than learning with single tasks. Examples include credit card fraud detection [Stolfo et al., 1997], newspaper sales prediction [Bakker and Heskes, 2003], verb-argument classification [Lee et al., 2007], object recognition [Caruana, 1997; Thrun and Mitchell, 1995], robotic control [Thrun, 1995] and compiler optimization performance predictions [Bonilla et al., 2007]. In chapter 4, we will apply multi-task learning on the inverse dynamics mappings for robotic control under multiple contexts.

1.2 Task Relatedness

A crucial ingredient in the application of meta-learning is the notion of *relatedness* between tasks. Unfortunately it is not clear how this may be defined. Caruana [1997] notes that just because learning two tasks together has better performance than learning each individually does not mean that they are related in the normal sense of the word. Nevertheless, for the lack of a better definition, we shall say two tasks are related to each other when they *benefit mutually* under meta-learning. This utilitarian opinion has also been echoed by Thrun and O'Sullivan [1998]: "The more the performance in task n improves through knowledge transferred from task m, the more related they are". Silver [2000, chapters 2, 3] and Eaton et al. [2008] share a similar view, while Ghosn and Bengio [2003] hold an alternative view that "task similarity should be decided based on the feedback of experts knowledgeable in the field being studied". The view advocated by Ghosn and Bengio is probably a better reflection of the very semantic and innate nature of task relatedness, since the similarity or dissimilarity between any two entities is highly dependent on the selected properties, attributes or features [Edelman, 1995]. However, the need for experts may be rather impractical when one wants to endow an autonomous artificial agent with learning to learn abilities, or when different experts have differing views on task relatedness, or when experts have difficulty elucidating task relatedness in terms concrete enough to be programmed or encoded mathematically.

In general, it will not be known from the onset of (meta-)learning whether two tasks are related or the extent of their relatedness. As mentioned by Baxter [2000] and supported empirically by Caruana [1997], *assuming relatedness in a set of tasks and simply learning them together can be detrimental.*[2] It is therefore important to have multi-task learning models that will generally benefit related tasks and will not hurt performance when these tasks are unrelated.

A common way to ameliorate this is to have *task descriptors* or *task features* that are correlated to relatedness under a certain distance measure that may be learnt parametrically [Bakker and Heskes, 2003; Bonilla et al., 2007; Yu et al., 2007]. However, although task descriptors *can* be used to improve models of task-relatedness, experiments by Evgeniou et al. [2005] throw doubts on whether task descriptors *should* be used in this way. Evgeniou et al. have shown on the ILEA school examination score prediction problem[3] that a multi-task learning model that incorporates task-descriptors-based relatedness is

[2] Although theoretical results by Baxter [2000] and others hint that there is no disadvantage in using multi-task learning, these results are based on upper bounds on the generalization errors, which may deviate from the true generalization errors.

[3] The data comes from the Inner London Education Authority (ILEA), and it consists of examination records from 139 schools.

less effective than a single-task learning model that treats the task descriptors as extra input dimensions. Hence, ironically, it may not be fruitful to consider multi-task learning models for a problem already endowed with task descriptors. Furthermore, in many real-life scenarios, task descriptors are neither available nor easy to define correctly and sufficiently to model complex task relatedness. Hence task relatedness based on task descriptors is not entirely satisfactory.

One may define task relatedness under very general conditions and constraints. Several possible measures of relatedness are discussed and evaluated by Silver [2000, chapter 3]. He distinguishes between (i) static measures, which predetermine relatedness based on statistics between tasks data sets and do not change during learning; (ii) dynamic measures, which change during learning based on the similarity between the current task hypotheses; and (iii) hybrid measures, which combine the first two. For problems lacking task descriptors, Bonilla et al. [2007] construct task descriptors via task responses to a common fixed set of canonical inputs. In the context of meta-learning, Bensusan [1999, §4.5] describes a task using the set of properties extracted from a decision tree trained for that task.

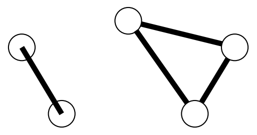
A more general approach is simply for the task descriptor of a task to be the *extent of its relatedness* to all the other tasks.[4] Although apparently circular, this suggests that we model task relatedness directly as an inherent property without artificially introducing task descriptors. To carry on our view that two tasks are related to each other if they benefit mutually when learnt together, task relatedness must have some bearing on predictive performance, and vice versa. In line with this, Thrun and O'Sullivan [1998] compute relatedness, which they call the *task transfer matrix*, within the k-nearest neighbour setting using the expected generalization accuracy when the distance metric learnt for one task is applied to another task. A similar approach is adopted by Eaton et al. [2008]. It is also within this spirit that one may estimate task relatedness by maximizing the marginal likelihood of task-relatedness given the data for all the tasks. This is because the marginal likelihood is defined as the marginal probability of the data, which may be viewed as a predictive score of the model [Kass and Raftery, 1995, §3.2]. We will follow this approach in chapter 4 when we use a multi-task Gaussian process model to learn the inverse dynamics of a robotic manipulator.

Below, we list some common notions of relatedness and discuss what they mean to multi-task learning.

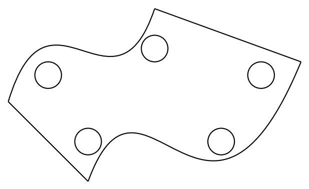
- If relatedness is a transitive binary predicate, then the natural multi-task learning approach that arises is task clustering, where each task strictly belongs to one of many clusters. This partitions the set of tasks into disjoint clusters, so that tasks within a cluster share information among themselves, and there is no transfer of information between clusters. Figure 1.2a illustrates this case. Works of this nature includes Xue et al. [2007b] and Roy and Kaelbling [2007].
- If we know from the onset that all the tasks are related, for example in a learning system specialized for a family of tasks, then a natural assumption is that their hypotheses, instead of varying freely, belong to the same family of hypotheses that (almost) reside on a system of low-dimensional manifolds representing the shared characteristics of the tasks. This stems from the basic principle that hypotheses for related tasks must be similar, and in this case the system of manifolds is used as a vehicle for information transfer between the tasks. It is important to con-

Each task is for a specific school, and is to predict the exam score of a student belonging to the school. Here the task descriptors are four school-dependent features.

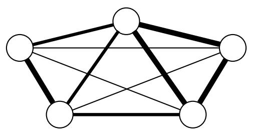
[4] I thank A. Spiliopoulou for suggesting this view.



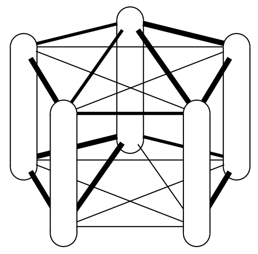
(a) Clusters of tasks



(b) Tasks lying on the same manifold



(c) Tasks having varying degrees of relatedness



(d) Component-dependent relatedness between tasks

Figure 1.2: Different notions of task relatedness in multi-task learning. The figures show multi-task learning for 5 tasks represented by the nodes/cylinders. In (c) and (d), the thicker a line, the more related are the two tasks connected by the line. In (d), a task is a cylinder to represent the additional component-dimension, and the relatedness between two tasks depends on the component.

strain the volume or capacity of the system of manifolds because if the manifolds otherwise span the entire space then the hypotheses are effectively free varying. This is directly related to the intrinsic dimension of the manifold, which is usually either fixed or determined by model selection. This is illustrated in Figure 1.2b, and Omohundro [1996], Ghosn and Bengio [2003] and Ando and Zhang [2005] are examples.

- We may use the *degree of relatedness* to measure the amount of sharing between tasks that will be beneficial for learning, which is the view of Silver [2000, chapter 2] and of Thrun and O'Sullivan [1998]. In this case, we may ask about the correlation between any two tasks: the more the two tasks are correlated, the more the transfer of information between them. Figure 1.2c illustrates this case, and work in this area includes Bonilla et al. [2007] and Yu et al. [2007].

- When each task can be naturally divided into components, we may have component-dependent relatedness between tasks. For example, suppose we have a set of binary classification tasks on object images, and we analyse each image in terms of shapes and appearances using the probabilistic "constellation" model [Burl et al., 1998]; then it may be the case that two tasks share much information between their shapes, but share little information between their appearances. Figure 1.2d illustrates this case. This has been investigated by Teh et al. [2005] and Xue et al. [2007a].

Finally, we emphasis that the above four notions are not mutually exclusive. For example, when tasks have degrees of relatedness, it may just happen that these degrees are either zero or one in a consistent manner, so we effectively have clusters of tasks. Nevertheless, every notion of task relatedness does suggest a different meta-bias or hyper-prior for multi-task learning, and this may result in a different

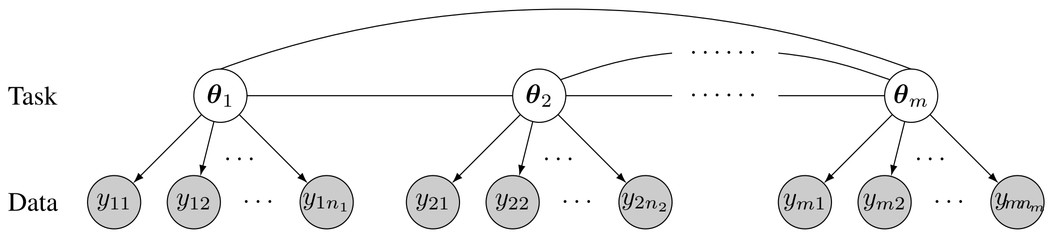


Figure 1.3: Explicit modelling of task relatedness in the graphical model. This is obtained by integrating out the $\boldsymbol{\pi}$ in Figure 1.1b. The undirected lines represent how the tasks are related.

learning algorithm.

1.2.1 Implicit and Explicit Modelling of Task Relatedness

How are the various notions of task relatedness manifested in the i.i.d. task-sampling of multi-task learning considered by, for example, Baxter [1997, 2000], as discussed in section 1.1? In the PAC learning case, this is manifested *implicitly* or *indirectly* through selecting the correct hypothesis space [Ben-David and Schuller, 2003]; in the Bayesian case, this is through a common distribution on the tasks. The specifics of the hypothesis spaces and the distributions determine the notion of task-relatedness involved. For example, a mixture model on the task priors in Bayesian learning will lead to learning clusters of tasks.

In addition to the implicit modelling approach, one may wish for an *explicit* or *direct* means of modelling task relatedness to make multi-task learning easier to understand and employ. In the case of the Bayesian framework, this is readily obtained when the hyper-prior on the tasks are integrated out to obtain a probabilistic model directly on the relations between tasks; see Figure 1.3.[5] In section 2.5.5, we will understand how a multi-task Gaussian process model may be obtained through this approach. The approach by Ben-David and Schuller Borbely [2008] for PAC learning achieves this by modeling explicitly the mappings between the input sampling distributions of the tasks. These mappings partition the hypothesis space into equivalent classes of hypotheses, where each equivalent class may be understood as a cluster of tasks related via mappings on their input spaces.

1.3 Overview and Contributions

We have introduced and motivated the research on multi-task learning (and on meta-learning). We have also discussed task relatedness, which is one key aspect of successful multi-task learning. The focus of this thesis is on a *multi-task Gaussian process model* that explicitly represents task-relatedness in its (cross-)covariance function in an interpretable manner. The model does not rely on task descriptors. There are various reasons for the choice of multi-task Gaussian process model:

[5] This view is presented by Williams and Chai at the Statistical and Machine Learning Interface Meeting at the University of Manchester, 23rd-24th July 2009.

Probabilistic Machine learning with Gaussian processes [Rasmussen and Williams, 2006] is a probabilistic framework that is easily understood and extensible using the tools and literature in statistics and probability theory. Compared to regularization, it gives the variances of predictions in addition to the means, which may be important in decision making and in integration with other tools.

Kernel-based A Gaussian process model can use any Mercer kernel as its covariance function. This means that it can be easily applied to many machine learning domains in which the inputs are complex objects but for which meaningful kernels can be defined. For example, it can use the pyramid kernel, which has been shown to be effective for image classification [Grauman and Darrell, 2006].

Relation to regularization When a concept is more easily expressed mathematically in terms of regularizers than in terms of probability distributions, the close relation that Gaussian processes have with L_2 regularizers enables us to use that concept probabilistically.

Analyzing task relatedness Extracting task-relations from learnt models will allow experts of the field to carry out task-centred analysis. The extraction is straightforward when task-relatedness is represented explicitly, as is the case for the multi-task Gaussian process model. For example, this made possible the analysis of multi-task learning in chapter 3.

Large data sets The use of large data sets can be made tractable for learning Gaussian processes by utilising existing approximation methods in the literature; see e.g., Quiñonero-Candela et al. [2007].

Specifically, we will consider the following multi-task Gaussian process regression model that learns M related functions $\{f_m\}_{m=1}^M$ by placing a zero mean Gaussian process prior that directly induces correlations between tasks. Let y_m be an observation of the $m\text{th}$ function at $\boldsymbol{x}$. Then the model is given by (see e.g., Bonilla et al. 2007)

$$\langle f_m(\boldsymbol{x}) f_{m'}(\boldsymbol{x}')\rangle \stackrel{\text{def}}{=} K^{\text{f}}_{mm'} k^{\text{x}}(\boldsymbol{x}, \boldsymbol{x}') \qquad y_m \sim \mathcal{N}(f_m(\boldsymbol{x}), \sigma_m^2), \tag{1.1}$$

where k^{x} is a covariance function over inputs, and K^{f} is a positive semi-definite matrix of inter-task similarities or relatedness, and σ_m^2 is the noise variance for the $m\text{th}$ task. This model on the functions $\{f_m\}$ is known as the *intrinsic correlation model* [Wackernagel, 1998, chapter 22] or the *proportional covariance model* [Chilès and Delfiner, 1999, §5.4.2] in the geostatistics community within the context of co-kriging. A generalization of this model will be considered in relation to three other multi-task models in **chapter 2**: the *hierarchical Bayesian latent source model* [Zhang, 2006], the *multi-task regularization networks model* [Evgeniou et al., 2005] and the *multi-task Bayesian neural network model* [Bakker and Heskes, 2003]. These multi-task learning models will be presented mainly from the perspective of the different notions of task-relatedness discussed in section 1.2. We will also provide a broad unifying view of multi-task learning using the Gaussian process model. This is possible because, unlike the other models, the multi-task Gaussian process model explicitly encodes task-relatedness (up to second order). In addition, chapter 2 will briefly survey other paradigms for meta-learning, discuss multi-task learning as known in other research areas and compare multi-task learning to other machine learning paradigms.

Each multi-task learning model generally assumes that learning multiple tasks together is beneficial. As discussed in section 1.2, the amount of benefit that can be reaped under multi-task learning is a common utilitarian understanding of task relatedness. In **chapter 3**, we will understand how and the extent to which multi-task learning helps improve the generalization of supervised learning. Our analysis is conducted on the multi-task Gaussian process model since, in contrast to other models, not only does it encode model assumptions for regression and task-relatedness in a transparent way, it is also amenable to the average-case analysis that we are after. The analysis contributes to an understanding of multi-task learning that is orthogonal to the existing PAC-based results outlined in section 1.1:

- With the exception of Baxter [1997], the existing theoretical results on multi-task learning operate within the PAC learning model, and they depend on the uniform convergence of frequencies to their probabilities. As pointed out by Vapnik [2006, Part 2, §1.1.2], "...if uniform convergence does not take place then any algorithm that does not use additional prior information and picks up one function from the set of admissible functions cannot generalize." Vapnik explains that this "leaves an opportunity to use averaging algorithms that possess a priori information about the set of admissible functions. In other words VC theory does not intersect with Bayesian theory." Our analysis precisely fills the void left by existing results, since it uses Gaussian processes, a Bayesian theory of functions.

- All the existing results do not depend explicitly on the extent of relatedness among the tasks; instead, their dependence on relatedness is only indirect, through the hypothesis space in the case of PAC learning and the distribution on the task priors in the case of Baxter [1997]'s Bayesian analysis. In contrast, the results in chapter 3 will depend directly on the correlation or relatedness between functions under the multi-task Gaussian process model.

- In existing analysis on multi-task learning, the number of examples is assumed to be the same for each task. In contrast, our analysis will also include the possibility that the full training set may not be evenly distributed among the tasks. It is important to allow uneven distributions, since one of the motivations of using multi-task learning is to allow a secondary task, for which we have abundant data, to help a primary task, for which we have few data.

Among others, our analysis will uncover an intrinsic limitation of the multi-task Gaussian process model of equation 1.1. Further, section 3.9 of chapter 3 will provide an understanding of the multi-task Gaussian process model through structures in the predictive means and variances given certain configurations of training observations. The results there generalize those of Chilès and Delfiner [1999], and may have practical applications in geostatistics.

In **chapter 4**, we will evaluate the multi-task Gaussian process model of equation 1.1 on the inverse dynamics problem for a robot manipulator. The inverse dynamics problem is to compute the torques needed at the joints to drive the robot manipulator along a given trajectory, and there are advantages to learning this function for adaptive control. The learning of the inverse dynamics can be treated as a function approximation problem. A robot manipulator will often need to be controlled while holding different loads in its end effector, giving rise to a multi-context or multi-load learning problem, and we treat predicting the inverse dynamics for a context/load as a task. We will follow a Bayesian approach and place Gaussian process priors over the space of functions, and we will show that this is effective for learning the inverse dynamics for a single context. Then, by placing independent Gaussian process priors over the latent functions of the inverse dynamics, we obtain the multi-task Gaussian process prior of equation 1.1 for handling multiple loads, where the inter-context similarity depends on the underlying inertial parameters of the robot manipulator. Experiments will demonstrate that this multi-task formulation is effective in sharing information among the various loads, and generally improves performance over either learning only on single contexts or pooling the data over all contexts. In addition to the experimental results, one of the contributions of this chapter is showing that the multi-task Gaussian process model follows naturally from the physics of the inverse dynamics.

Finally, **chapter 5** concludes this thesis by summarizing the results and providing a discussion of possible future directions.

As far as possible, each of chapters 2, 3 and 4 is written in a self-contained manner, so that it may be read independently of the others. Since this thesis applies Gaussian processes within the context of machine learning, the reader may find it helpful to read Rasmussen and Williams [2006], although a brief introduction is given in section 4.3.2.1 on page 108 in the context of probabilistic function approximation. For the purpose of chapter 2, Bishop [2006] and Evgeniou et al. [2000] are useful references for graphical models and regularization networks respectively.

1.4 List of Relevant Published Papers

Below is a list of published peer-reviewed papers. For each paper, we give the authors, the URL and briefly indicate how the paper related to the thesis. The complete reference of each paper can be found in the bibliography.

1. Bonilla, Chai, and Williams [2008], available at `http://books.nips.cc/papers/files/nips20/NIPS2007_0431.pdf`. This paper introduces to the machine learning community the multi-task Gaussian process learning model in the manner of equation 1.1. To the best of our knowledge, although the formulation given by equation 1.1 has been known in the geostatistics community, the machine learning community has initially known it under different disguises: the tensor Gaussian process models of Yu et al. [2007] for link prediction and the semiparametric latent factor models of Teh et al. [2005].

2. Chai, Williams, Klanke, and Vijayakumar [2009], available at `http://books.nips.cc/papers/files/nips21/NIPS2008_0441.pdf`. Chapter 4 is an extension of this paper.

3. Chai [2009], available at `http://books.nips.cc/papers/files/nips22/NIPS2009_0076.pdf`. Chapter 3 is an extension of this paper.

Chapter 2

Models for Multi-task Learning

Generalizing from a set of examples involves more than memorizing them. An abstraction or a model is needed in order to interpolate between the examples and, perhaps, extrapolate to novel regions of the observation space. Likewise, multi-task learning requires a channel between the tasks along which tasks share information. This can be achieved by sharing selected aspects of the learning model for each task. In section 2.1 we consider a generalized linear model (GLM) for each task and see how tasks may share either their model parameters or their features/structures mappings. There are various ways of implementing this manner of sharing between tasks. Three examples will be given: the *hierarchical Bayesian latent source model* of Zhang [2006] in section 2.2, the *multi-task regularization networks model* of Evgeniou et al. [2005] in section 2.3 and the *multi-task Bayesian neural network model* of Bakker and Heskes [2003] in section 2.4.

A different way to think about multi-task learning is in terms of how tasks are related to each other. Common examples of task relatedness are

task clustering, where tasks in the same cluster are related, and tasks in different clusters are not;

manifold of tasks, where tasks reside near a system of low-dimensional manifolds representing the shared characteristics of the tasks;

varying degrees of relatedness between the tasks that allows different extents of information sharing between them;

component-dependent relatedness, where each task has more than one component, and the relatedness between the tasks varies across the components.

These notions of task relatedness have been elaborated in section 1.2. The multi-task learning models in sections 2.2, 2.3 and 2.4 will be discussed primarily with reference to these different notions of task relatedness.

In section 2.5 we will introduce multi-task learning with Gaussian processes (GPs, Rasmussen and Williams [2006]). We will show how the different notions of task relatedness can be achieved within the Gaussian process framework by considering just the covariance function between the different tasks. This will used to understand the other models for multi-task learning. Section 2.5.8 will outline how a multi-task Gaussian process model can be developed for the case where two tasks have different input domains $\mathcal{X}_1$ and $\mathcal{X}_2$.

A summary will be given in section 2.6. We will then proceed to discuss general meta-learning in section 2.7 and related machine learning paradigms in section 2.8.

2.1 Introduction

One of the more commonly used and analyzed paradigms for machine learning is supervised learning. Its goal is to produce a mapping $\mathcal{X} \mapsto \mathcal{Y}$ based on n examples $\mathcal{D} \overset{\text{def}}{=} \{(\boldsymbol{x}_i, y_i) | \boldsymbol{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}_{i=1}^{n}$. Having learned the mapping, the task is to predict a response y given any input $\boldsymbol{x}$. Typically the output space $\mathcal{Y}$ is scalar in nature, such as $\mathbb{R}$ for regression or $\{0, 1\}$ for classification.[1] A general class of models for this purpose is *generalized linear models* (GLMs), which can be expressed as

$$y \sim p(\cdot \mid f(\boldsymbol{x}), \sigma) \qquad y \in \mathcal{Y},\ \sigma \in \mathbb{R} \qquad \text{noise model} \tag{2.1a}$$

$$f(\boldsymbol{x}) = \boldsymbol{u}^{\mathrm{T}} \boldsymbol{\phi}(\boldsymbol{x}) \qquad \boldsymbol{x} \in \mathcal{X},\ \boldsymbol{\phi} : \mathcal{X} \mapsto \mathbb{R}^{D^{\Phi}},\ \boldsymbol{u} \in \mathbb{R}^{D^{\Phi}} \qquad \text{linear model,} \tag{2.1b}$$

where $\boldsymbol{\phi}$ is a mapping from the input space $\mathcal{X}$ onto a D^{Φ}-dimensional Euclidean vector space, which we may also call the *feature* space, and $\boldsymbol{u}$ is the *model parameter vector*. The function f is known as the *latent function*, since it is not observed directly. The noise model given by equation 2.1a will vary with the nature of the tasks, e.g., Gaussian for regression tasks and logistic (or probit) for classification tasks; see e.g., Nelder and Wedderburn [1972]; Hardin and Hilbe [2007]. It is parametrized by σ, which is usually related to the variance of the noise model.

Given M predictive tasks, each may be learnt with a GLM that is different for every task, i.e.,

$$y_m \sim p_m(\cdot \mid f_m(\boldsymbol{x}_m), \sigma_m) \qquad y_m \in \mathcal{Y}_m,\ \sigma_m \in \mathbb{R} \tag{2.2a}$$

$$f_m(\boldsymbol{x}_m) = \boldsymbol{u}_m^{\mathrm{T}} \boldsymbol{\phi}_m(\boldsymbol{x}_m) \qquad \boldsymbol{x}_m \in \mathcal{X}_m,\ \boldsymbol{\phi}_m : \mathcal{X}_m \mapsto \mathbb{R}^{D_m^{\Phi}},\ \boldsymbol{u}_m \in \mathbb{R}^{D_m^{\Phi}}, \tag{2.2b}$$

where $m = 1 \ldots M$ indexes over the tasks. A general approach to multi-task learning is to share information among the tasks through the constituents of the GLMs. This can be achieved by placing shared constraints and priors on the models. A common constraint is that the tasks have the same input and output representations, so that all the tasks are of the same nature, i.e., to produce mappings $\mathcal{X} \mapsto \mathcal{Y}$. To facilitate transfer of information between the tasks, it is also useful for them to have the same feature space. Thus, a reasonable starting point is

$$y_m \sim p(\cdot \mid f_m(\boldsymbol{x}), \sigma_m) \qquad y_m \in \mathcal{Y},\ \sigma_m \in \mathbb{R} \tag{2.3a}$$

$$f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\mathrm{T}} \boldsymbol{\phi}_m(\boldsymbol{x}) \qquad \boldsymbol{x} \in \mathcal{X},\ \boldsymbol{\phi}_m : \mathcal{X} \mapsto \mathbb{R}^{D^{\Phi}},\ \boldsymbol{u}_m \in \mathbb{R}^{D^{\Phi}}. \tag{2.3b}$$

Next, we can obtain different models of multi-task learning by placing different priors on the model parameters $\boldsymbol{u}_m$s and the feature mappings $\boldsymbol{\phi}_m$s. The result is the following general hierarchical Bayesian

[1] The case where $\mathcal{Y}$ is a vector space will be discussed in section 2.7.2; it can be seen to be related to multi-task learning.

model for multi-task learning:

$$y_m \sim p(\cdot \mid f_m(\boldsymbol{x}), \sigma_m) \tag{2.4a}$$

$$f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\mathrm{T}} \boldsymbol{\phi}_m(\boldsymbol{x}) \tag{2.4b}$$

$$\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M \sim p(\cdot \mid \Upsilon^{\mathrm{u}}) \qquad \boldsymbol{u}_m \in \mathbb{R}^{D^\Phi} \tag{2.4c}$$

$$\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_M \sim p(\cdot \mid \Omega^\Phi) \qquad \boldsymbol{\phi}_m : \mathcal{X} \mapsto \mathbb{R}^{D^\Phi}, \tag{2.4d}$$

where Υ^{u} and Ω^Φ are hyper-parameters. It is not necessary for the $\boldsymbol{u}_m$s to be independent, and similarly for the $\boldsymbol{\phi}_m$s.

As it is, the above model for sharing among multiple tasks depends heavily on the hyper-priors on $\{\boldsymbol{u}_m\}$ and $\{\boldsymbol{\phi}_m\}$. On the one hand, if these hyper-priors factorize into independent and identical distributions (i.i.d.) for each $\boldsymbol{u}_m$ and each $\boldsymbol{\phi}_m$, then the sharing between the tasks is rather weak: the tasks are related only through their common hyper-priors. On the other hand, in absence of the i.i.d. assumption it may be unnecessarily involved to specify both hyper-priors in order to achieve a desired notion of task relatedness. One way around this dilemma is for the different tasks to share a common feature mapping $\boldsymbol{\phi}$ and to allow each task its own model parameter $\boldsymbol{u}_m$, i.e.,

$$f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\mathrm{T}} \boldsymbol{\phi}(\boldsymbol{x}). \tag{2.5}$$

The desired notion of task relatedness is now primarily achieved through an appropriate hyper-prior on $\{\boldsymbol{u}_m\}$. If the feature mapping $\boldsymbol{\phi}(\boldsymbol{x})$ is implemented by the hidden layer of a neural network, then we have the *multi-task Bayesian neural network model* of Bakker and Heskes [2003]. This will be discussed in section 2.4.

When $\boldsymbol{x}$ can be conveniently expressed as a D-dimensional vector, we can restrict the common $\boldsymbol{\phi}$ to be a linear mapping specified by a D^Φ-by-D matrix Φ. Equation 2.5 can now be replaced by

$$f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\mathrm{T}} \Phi \boldsymbol{x} \qquad \boldsymbol{x} \in \mathbb{R}^D,\ \boldsymbol{u}_m \in \mathbb{R}^{D^\Phi},\ \Phi \in \mathbb{R}^{D^\Phi \times D}. \tag{2.6}$$

In the sense that Φ will create linear structures of $\boldsymbol{x}$, we will call Φ the *structure matrix*. This gives the *hierarchical Bayesian latent source model* of Zhang [2006] that we will discuss in section 2.2.

Instead of having a shared structure matrix Φ, the M tasks can share a common model parameter vector $\boldsymbol{u}$. Having a shared structure matrix is equivalent to having a shared model parameter vector, in the sense that will now be explained. The function for the $m\mathrm{th}$ task as expressed in (2.6) can be written as

$$f_m(\boldsymbol{x}) = \boldsymbol{w}_m^{\mathrm{T}} \boldsymbol{x}, \tag{2.7}$$

where $\boldsymbol{w}_m \overset{\mathrm{def}}{=} \boldsymbol{u}_m^{\mathrm{T}} \Phi$ may be called the *effective parameter vector*. Since $\boldsymbol{w}_m$ can be expressed as many different products of vectors and matrices, we can traverse between having a shared structure matrix and having a shared model parameter vector via suitable substitutions. For a given choice of shared model parameter vector $\boldsymbol{u} \in \mathbb{R}^{D^\Phi}$, since $\boldsymbol{u}^+ \boldsymbol{u} = 1$, we may write

$$\left.\begin{array}{r}\text{common}\\ \text{structure matrix}\end{array}\right\} \boldsymbol{u}_m^{\mathrm{T}} \Phi = (\boldsymbol{u}^+ \boldsymbol{u})^{\mathrm{T}} \boldsymbol{u}_m^{\mathrm{T}} \Phi = \boldsymbol{u}^{\mathrm{T}} (\boldsymbol{u}_m \boldsymbol{u}^+)^{\mathrm{T}} \Phi = \boldsymbol{u}^{\mathrm{T}} \Phi_m \left\{\begin{array}{l}\text{common}\\ \text{model parameter}\end{array}\right. \tag{2.8}$$

where $(\boldsymbol{u}_m \boldsymbol{u}^+)^{\mathrm{T}} \Phi$ is substituted with Φ_m in the last equality. Thus, it may seem arbitrary to distinguish between having a shared structure matrix and having a shared model parameter vector. Nevertheless, a conceptual difference still exists between the two, which leads to different multi-task learning models and algorithms. For example, the *multi-task regularization networks model* of Evgeniou et al. [2005] uses a shared model parameter $\boldsymbol{u}$ and learns by regularizing the norm $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}$. More details can be found in section 2.3.

For M tasks, each with n_m examples $\mathcal{D}_m \overset{\text{def}}{=} \{(\boldsymbol{x}_{mi}, y_{mi}) | \boldsymbol{x}_{mi} \in \mathcal{X}, y_{mi} \in \mathcal{Y}\}_{i=1}^{n_m}$, let $\mathcal{D}_m^{\mathrm{x}}$ be the set of $\boldsymbol{x}$'s in $\mathcal{D}_m$ and $\mathcal{D}_m^{\mathrm{y}}$ be the set of y's. A line of reasoning by MacKay [2003, section 45.1] can be applied readily to multi-task learning: the representation of the latent functions $f_m(\boldsymbol{x})$ does not matter at least from the point of prediction, since the quantities of interest $p(f_m(\boldsymbol{x}) \mid \{\mathcal{D}_m\}_{m=1}^M, \boldsymbol{x})$ (for prediction) and $p(\{\mathcal{D}_m^{\mathrm{y}}\}_{m=1}^M \mid \{\mathcal{D}_m^{\mathrm{x}}\}_{m=1}^M)$ (for model selection or comparison) make reference to neither the model parameter vectors $\boldsymbol{u}_m$s nor the feature mappings $\boldsymbol{\phi}_m$s. Therefore, if we can deal directly and efficiently with the functions $f_1, \ldots, f_M$, there is no reason why we should not. What remains is to find a suitable class of probability distributions on these functions. One suitable and popular candidate in machine learning is the *Gaussian process* (GP) prior on functions [Rasmussen and Williams, 2006], and we will introduce multi-task learning models based on Gaussian Processes in section 2.5. In that section, we will also see how we may understand the latent source model, the neural network model and the regularization networks model using Gaussian processes. The Gaussian process viewpoint will allow us to extract salient features or notions of multi-task learning that are expressed explicitly in terms of task-relatedness between the M functions.

2.2 Hierarchical Bayesian Latent Source Model

In this section, we will look at multi-task learning models in which tasks have different parameters $\boldsymbol{u}_m$ but share the same structure Φ:

$$f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\mathrm{T}}\Phi\boldsymbol{x} \qquad \boldsymbol{u}_m \in \mathbb{R}^{D^\Phi},\ \Phi \in \mathbb{R}^{D^\Phi \times D},\ \boldsymbol{x} \in \mathbb{R}^D. \tag{2.9}$$

One such structure is to retain the original vector space of the inputs together with a linear transformation of this space, i.e.

$$\Phi = \begin{pmatrix} I_{D\times D} \\ A^{\mathrm{T}} \end{pmatrix} \qquad A \in \mathbb{R}^{D\times D^{\mathrm{s}}}, \tag{2.10}$$

where $D^\Phi = D^{\mathrm{s}} + D$, and A^{T} is a linear transformation from the original D-dimensional input space onto a D^{s}-dimensional feature space. The converse view is that A maps a D^{s}-dimensional latent space onto the effective parameter space of dimension D. Unlike conventional factor analysis or independent component analysis, this latent space is the latent space of task parameters rather than the latent space of the input $\boldsymbol{x}$. To see this, let us breakdown the parameter vector for task m as

$$\boldsymbol{u}_m = \begin{pmatrix} \boldsymbol{\mu}_{\mathrm{w}} + \boldsymbol{\xi}_m \\ \boldsymbol{s}_m \end{pmatrix} \tag{2.11}$$

so that the $m\mathrm{th}$ effective parameter may be expressed as

$$\boldsymbol{w}_m = \begin{pmatrix} I_{D\times D} & A \end{pmatrix} \begin{pmatrix} \boldsymbol{\mu}_{\mathrm{w}} + \boldsymbol{\xi}_m \\ \boldsymbol{s}_m \end{pmatrix} = \boldsymbol{\mu}_{\mathrm{w}} + A\boldsymbol{s}_m + \boldsymbol{\xi}_m, \tag{2.12}$$

which consists of (1) an offset $\boldsymbol{\mu}_{\mathrm{w}}$, (2) a mixing A of the *latent source* $\boldsymbol{s}_m$, and (3) a noise term $\boldsymbol{\xi}_m$; if the latent source has zero mean, then $\boldsymbol{\mu}_{\mathrm{w}}$ can be seen as the mean parameter vector of the tasks. Typically $D^{\mathrm{s}} \leqslant D$, though section 2.2.4 gives an example where $D^{\mathrm{s}} \geqslant D$. The above setup leads to the *latent source model* (LSM) of Zhang [2006, chapter 3] for multi-task learning:

$$y_m \sim p(\cdot \mid f_m(\boldsymbol{x}), \sigma_m^2) \tag{2.13a}$$

$$f_m(\boldsymbol{x}) = \boldsymbol{w}_m^{\mathrm{T}} \boldsymbol{x} \tag{2.13b}$$

$$\boldsymbol{w}_m \sim \mathcal{N}(\cdot \mid \boldsymbol{\mu}_{\mathrm{w}} + A\boldsymbol{s}_m, \Sigma^{\xi}) \qquad \boldsymbol{w}_m \in \mathbb{R}^D,\ \boldsymbol{\mu}_{\mathrm{w}} \in \mathbb{R}^D \tag{2.13c}$$

$$\boldsymbol{s}_1, \ldots, \boldsymbol{s}_M \sim p(\cdot \mid \Upsilon^{\mathrm{s}}) \qquad \boldsymbol{s}_m \in \mathbb{R}^{D^{\mathrm{s}}} \tag{2.13d}$$

$$A \sim p(\cdot \mid \Omega^{\mathrm{A}}). \tag{2.13e}$$

where Σ^{ξ} is the noise (co)variance of $\boldsymbol{\xi}$, i.e., $\boldsymbol{\xi} \sim \mathcal{N}(\cdot \mid 0, \Sigma^{\xi})$, and is usually a diagonal matrix for independent noise; and Υ^{s} and Ω^{A} are the hyperparameters of the distributions on $\{\boldsymbol{s}_m\}$ and A. In general, there may be inter-dependencies among the latent sources. The graphical model is given in Figure 2.1a on the following page.

Let us call each of the D^{s} dimensions of $\boldsymbol{s}_m$ a *latent component*. An alternative to understanding the matrix product $A\boldsymbol{s}_m$ as a mixing of the latent components is to see it as a location in the column space of matrix A, where the location is determined by $\boldsymbol{s}_m$. Within this view, matrix A determines the structure of the predictor space from where $\boldsymbol{s}_m$ selects a predictor for task m. In section 2.2.2, the selection is rather free, spanning the whole of the column space; this will result in a manifold of predictors. In section 2.2.3, the selection is confined to a countable set of locations; this leads to tasks clustering at these locations. However, before delving into the task manifold and task clustering, we first look at two very general kinds of (in)dependencies on $\{\boldsymbol{s}_m\}$.

2.2.1 General (In)Dependencies between Latent Sources

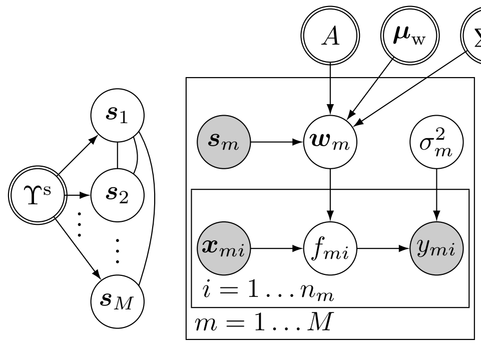
The practicability and usefulness of the model described by (2.13) depends very much on the distributions on the latent sources $\{\boldsymbol{s}_m\}$ and the mixing matrix A. We shall now describe two very general kinds of dependencies for the latent sources, and postpone specific examples on task relatedness to later sections.

2.2.1.1 Independent Sources

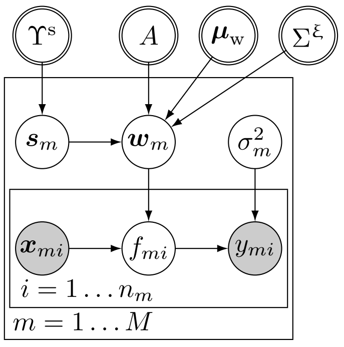
All the specific models considered in [Zhang, 2006, chapters 3, 4, 5 and 7] assume conditional independence between the sources so that

$$p(\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_M \mid \Upsilon^{\mathrm{s}}) = \prod_{m=1}^{M} p(\boldsymbol{s}_m \mid \Upsilon^{\mathrm{s}}) \qquad \text{independent latent sources.} \tag{2.14}$$

The graphical model is given by Figure 2.1b on the next page. Such models — henceforth called *independent latent source models* (ILSM) — will lead to conditional independence between the effective task parameters $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M$. This means that we have M independent meta-samples (i.e., the $\boldsymbol{w}_m$s) for learning the hyper-parameters. In section 2.2.2 and 2.2.3 we shall see how ILSM may be used for learning task manifolds and clustering tasks, two notions of task-relatedness discussed in section 1.2.



(a) Inter-dependent tasks



(b) Conditionally independent tasks

Figure 2.1: Hierarchical Bayesian multi-task learning models using latent sources [Zhang, 2006]. Figure (a) is for the model given by equation 2.13, where general inter-dependencies between the latent sources $\boldsymbol{s}_m$s may be present. Figure (b) is for the model given by equation 2.14, where there is conditional independence between the latent sources given Υ^{s}. The graphical notation follows that of Buntine [1994].

Of particular interest is when, in addition to independence between the latent sources and a diagonal matrix Σ^{ξ}, there is independence between the latent components of each source:

$$p(\boldsymbol{s}_m \mid \Upsilon^{\mathrm{s}}) = \prod_{d=1}^{D^{\mathrm{s}}} p(s_{md} \mid \Upsilon_d^{\mathrm{s}}). \tag{2.15}$$

If the distributions are Gaussian, for example, $\boldsymbol{s}_m \overset{\text{iid}}{\sim} \mathcal{N}(\cdot \mid \mathbf{0}, I)$, then we have the *factor analysis* model on the effective parameter vector $\boldsymbol{w}_m$. If, instead, the distribution for each latent component is non-Gaussian, then we have an *independent component analysis* (ICA) model on $\boldsymbol{w}_m$.[2]

2.2.1.2 Separable Dependencies[3]

Instead of fully independent sources, we may want to retain some relations between the M latent sources and between the components within a latent source. Given that for learning a general $p(\{\boldsymbol{s}_m\} \mid \Upsilon^{\mathrm{s}})$ there is effectively only one meta-sample, which is the set $\{\boldsymbol{w}_m\}$, we must be careful not to introduce too much freedom into the model. This suggests an orthogonal parametrization that separately models the task relations and the latent-component relations. For this purpose, it is useful to introduce the matrix

$$S \overset{\text{def}}{=} (\boldsymbol{s}_1 \mid \boldsymbol{s}_2 \mid \cdots \mid \boldsymbol{s}_M) \tag{2.16}$$

and consider a matrix-variate distribution on S. One commonly used distribution is the matrix normal distribution [Dawid, 1981], which may be expressed as a normal distribution on $\operatorname{vec} S$:

$$\operatorname{vec} S \sim \mathcal{N}(\cdot \mid \boldsymbol{\mu}_{\mathrm{S}}, \Sigma^{\mathrm{f}} \otimes \Sigma^{\mathrm{x}}), \tag{2.17}$$

where $\otimes$ denotes the Kronecker product, and Σ^{f} models variations across tasks while Σ^{x} models variations across latent-components.

[2] The original name given by Zhang et al. [2006] for ILSM is latent ICA. This may be misleading, since ILSM only specifies that the sources are independent, but ICA also requires that the components within $\boldsymbol{s}_m$ are independent. This is later corrected in [Zhang, 2006, section 5.2].

[3] Such a model has not been explored by Zhang [2006].

There are other strategies for separating into task relations and latent-component relations. An example is the *matrix stick-breaking process* of Xue et al. [2007a], which will be described in section 2.2.4.

2.2.2 Manifold of Predictors

The aim in manifold learning is to project the parameters of the predictors onto a low-dimensional manifold without drastic loss in predictive performance. Although most of the published literature may discuss general manifolds, linear manifolds are usually used eventually due to the ease of understanding and efficiency of computations. For linear manifolds, a natural decomposition is that of equation 2.12

$$\boldsymbol{w}_m = \boldsymbol{\mu}_{\mathrm{w}} + A\boldsymbol{s}_m + \boldsymbol{\xi}_m \tag{2.18}$$

when the dimension of the latent sources $\boldsymbol{s}_m$ is smaller than the dimension of the task parameters $\boldsymbol{w}_m$, i.e., $D^{\mathrm{s}} \leqslant D$. The manifold is the column space of A offset by $\boldsymbol{\mu}_{\mathrm{w}}$, and the intrinsic dimension of the manifold is D^{s}, the rank of A. For task m, its latent source $\boldsymbol{s}_m$ determines its location on the manifold; however, the effective parameter vector $\boldsymbol{w}_m$ does not lie exactly on the manifold due to the additive noise component $\boldsymbol{\xi}_m$.

The ILSM model can be used for learning these manifolds (see [Zhang, 2006, sections 3.2.4, 3.2.7]), since, for manifold learning, the tasks are mutually independent *given* the linear manifold. For example, Zhang [2006, equation 5.2, algorithm 3] places zero mean Laplace priors on the mixing matrix A

$$a_{ij} \sim \mathcal{L}aplace(\cdot|0, v) \qquad i = 1 \ldots D,\ j = 1 \ldots D^{\mathrm{s}}, \tag{2.19}$$

where a_{ij} is the $(i,j)\mathrm{th}$ entry of A, and v is the spread parameter of the Laplace distribution. In this case, learning involves finding the maximum likelihood estimates for the independent latent sources $\boldsymbol{s}_m$s and the maximum a posterior estimates of A. The Laplace prior is a sparse prior and placing it on the entries of A will allow, or in fact prefer, certain rows of A to be zero vectors, so that the eventual intrinsic dimension of the manifold may be less than D^{s}.

Instead of a sparse prior on A, Ando and Zhang [2005] constrain A to be orthonormal, i.e., $A^{\mathrm{T}}A = I_{D^{\mathrm{s}}}$, in order to obtain a learning algorithm that is easy to implement. This orthonormal constraint has the side effect of demanding the dimension of the manifold to be fixed to D^{s}, since A must be of rank D^{s} for the constraint to hold. This is in contrast to Zhang's Laplace prior which may allow the dimension to be less than D^{s}.

Let L be the loss function corresponding to the noise model given by equation 2.13a of LSM (see Evgeniou et al. 2000). Ando and Zhang [2005] use the objective function

$$\operatorname*{arg\,min}_{A,\{\boldsymbol{\xi}_\cdot\},\{\boldsymbol{s}_\cdot\}} \sum_{m=1}^{M} \left(\frac{1}{n_m} \sum_{i=1}^{n_m} L\big(f_m(\boldsymbol{x}_{mi}), y_{mi}\big) + \lambda_m \boldsymbol{\xi}_m^{\mathrm{T}} \boldsymbol{\xi}_m \right) \quad \text{s.t. } A^{\mathrm{T}}A = I \tag{2.20a}$$

$$f_m(\boldsymbol{x}) = \boldsymbol{w}_m^{\mathrm{T}} \boldsymbol{x} \qquad\qquad \boldsymbol{w}_m = A\boldsymbol{s}_m + \boldsymbol{\xi}_m, \tag{2.20b}$$

where λ_m is the regularization parameter for the $m\mathrm{th}$ task, the orthonormal constraint is imposed on A, and the regulariser on $\boldsymbol{\xi}_m$ is equivalent to a Gaussian prior with zero mean and variance $1/2\lambda_m$. For the special case where the regularization parameters λ_ms are the same, say λ, we can use equation

2.13c and set $\Sigma^{\xi} = (1/2\lambda)I$. The solution to (2.20) can be obtained by Ando and Zhang's singular-value-decomposition alternating-structure-optimization (SVD-ASO) algorithm, which is an iterative algorithm. Let $\boldsymbol{w}_m = \boldsymbol{\xi}_m + A\boldsymbol{s}_m$. Each iteration consists of two steps:

1. Fix A and $\{\boldsymbol{s}_m\}$, and optimize with respect to $\{\boldsymbol{w}_m\}$ (or equivalently $\{\boldsymbol{\xi}_m\}$) using any convex optimization algorithm.
2. Fix $\{\boldsymbol{w}_m\}$, and optimize with respect to A and $\{\boldsymbol{s}_m\}$ using singular value decomposition.

We give a brief explanation on why singular value decomposition (SVD) can be used for step (2), and refer the reader to Ando and Zhang [2005] for the exact details. Consider the definition of $\boldsymbol{w}_m$ and the constraints given in (2.20a). The isotropic Gaussian prior on the $\boldsymbol{\xi}_m$s imposed by the regularizers indicates that A and $\{\boldsymbol{s}_m\}$ should be fitted to $\boldsymbol{w}_m$ in the *least squares* sense, up to the scaling factors λ_ms. The orthonormal constraint on A will then lead to the SVD algorithm. This is a form of *partial least squares* (see section 2.7.2) in the predictor (or parameter) space.

2.2.3 Task Clustering

Task clustering may be achieved with the ILSM model via a mixture model [Zhang, 2006, sections 3.2.3 and 7.2.1.2], using an appropriate prior on the latent sources $\boldsymbol{s}_m$s. For $D^{\mathrm{s}} = P$ clusters, let Δ^{P-1} be the space of $(P-1)$-simplexes, and use the multinomial distribution (with one trial)[4]

$$\boldsymbol{s}_m \sim \mathcal{M}ultinomial(\cdot|1, \boldsymbol{\pi}) \qquad\qquad \boldsymbol{\pi} \in \Delta^{P-1}. \tag{2.21}$$

Under this distribution, only one element in $\boldsymbol{s}_m$ is one, and the rest are zero. In this way, the P-vector $\boldsymbol{s}_m$ acts as a vector of indicator variables $(z_m^1, z_m^2, \ldots, z_m^P)^{\mathrm{T}}$ that selects one of the P clusters for task m. If A is decomposed as

$$A \stackrel{\text{def}}{=} \left(\boldsymbol{a}^1 \mid \ldots \mid \boldsymbol{a}^P\right) \in \mathbb{R}^{D\times P},$$

then we may rewrite the effective parameter the $m\mathrm{th}$ task as

$$\boldsymbol{w}_m = \boldsymbol{\mu}_{\mathrm{w}} + A\boldsymbol{s}_m + \boldsymbol{\xi}_m = \boldsymbol{\mu}_{\mathrm{w}} + \sum_{p=1}^{P} z_m^p \boldsymbol{a}^p + \boldsymbol{\xi}_m. \tag{2.22}$$

In probabilistic terms, this is a mixture of Gaussians for each $\boldsymbol{w}_m$ if $\boldsymbol{\xi}_m \sim \mathcal{N}(0, \Sigma^{\xi})$:

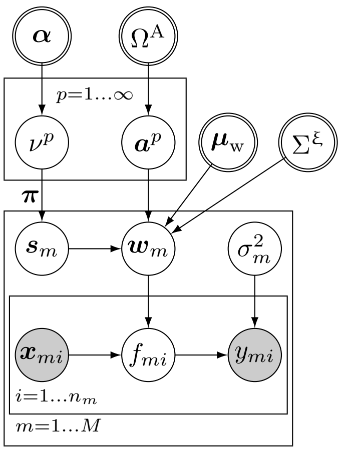
$$p(\boldsymbol{w}_m) = \sum_{p=1}^{P} \pi^p \mathcal{N}(\boldsymbol{w}_m \mid \boldsymbol{\mu}_{\mathrm{w}} + \boldsymbol{a}^p, \Sigma^{\xi}), \tag{2.23}$$

where π^p is the $p\mathrm{th}$ element of $\boldsymbol{\pi}$. All the tasks belonging to the $p\mathrm{th}$ cluster will have $z_m^p = 1$, so they have a shared mean parameter of $\boldsymbol{\mu}_{\mathrm{w}} + \boldsymbol{a}^p$, and $\boldsymbol{\xi}_m$ is the noise term.

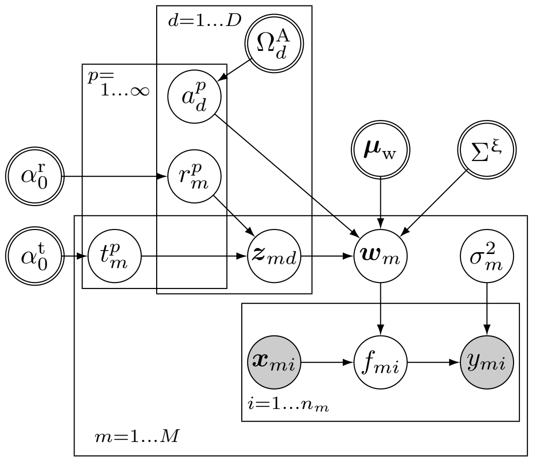
A way to extend the multinomial model of the latent sources given by (2.21) is to place a Dirichlet prior on $\boldsymbol{\pi}$

$$\boldsymbol{\pi} \sim \mathcal{D}ir(\cdot \mid \alpha^1, \ldots, \alpha^P) \qquad\qquad \alpha^p > 0. \tag{2.24}$$

[4] The alternative name is a *discrete* distribution on $\boldsymbol{s}_m$.



(a) Dirichlet Process Model



(b) MSBP Model

Figure 2.2: Multi-task learning under LSM using task clustering. Figure (a) is the same as Figure 2.1b, but with the Υ^{s} and the A there replaced by the Dirichlet process here. Figure (b) is the matrix stick-breaking process which allows separate clustering for different components of $\boldsymbol{w}_m$. Here, the concatenated vector $(\boldsymbol{z}_{m1}, \boldsymbol{z}_{m2}, \ldots, \boldsymbol{z}_{mD})^{\mathrm{T}}$ is the latent source $\boldsymbol{s}_m$. Note that the original MSBP by Xue et al. [2007a] does not involve modelling the noise on $\boldsymbol{w}_m$ due to Σ^{ξ}.

In order not to be constrained to a fixed number of clusters, one can potentially allow the number of clusters, P, to approach infinity; though, in reality, the number of clusters is upper-bounded by the number of tasks. If the distribution on $\boldsymbol{\pi}$ in equation 2.24 is a symmetric Dirichlet distribution with $\alpha_i = \alpha_0/P$ ($\alpha_0 > 0$), then this can be achieved by letting $P \to \infty$. The result is a *Dirichlet Process* (DP) prior on the $\boldsymbol{w}_m$s with precision parameter α_0 [Ferguson, 1973]. The stick-breaking construction view of this process is [Sethuraman, 1994]

$$\nu^p \sim \mathcal{B}eta(\cdot \mid 1, \alpha_0) \qquad \pi^p = \nu^p \prod_{p'=1}^{p-1} \left(1 - \nu^{p'}\right) \tag{2.25a}$$

$$\boldsymbol{a}^p \sim p(\cdot \mid \Omega^{\mathrm{A}}). \tag{2.25b}$$

The resulting multi-task model is depicted in Figure 2.2a. The graphical model is that of Figure 2.1b, but with Υ^{s} and A replaced by a subgraph specified by the stick-breaking construction of the DP. This emphasizes that this Dirichlet process multi-task clustering model is an instance of the ILSM. Also, in contrast to the models with finite task clusters, the DP model demands a generative model for A, so as to generate a $\boldsymbol{a}^p$ for every cluster possible. This is the DP model given by Xue et al. [2007b], except for the addition of $\boldsymbol{\mu}_{\mathrm{w}}$ and $\boldsymbol{\xi}_m$ here.

Such a DP construction is commonly used in the statistics literature to model *multilevel* data (see section 2.7.3 for a brief overview). In multi-task learning, Roy and Kaelbling [2007] have also used a similar DP construction, but within the setting of a generative model for the inputs $\boldsymbol{x}$, i.e., a latent label y generates $\boldsymbol{x}$ instead of the discriminative model discussed here. A more elaborate DP has been proposed by Rodríguez et al. [2008], where the atoms of the DP are in turn drawn from a DP. This has been applied in multiple related hidden Markov models (HMM) by Ni et al. [2007]. A stick-breaking process that models feature-component dependent clustering of tasks is also possible, and this is our next topic.

2.2.4 Feature-component Dependent Clustering

Xue et al. [2007a] propose the matrix stick-breaking process (MSBP) that partitions the components of the task parameters and allows every partition to cluster in a different way. This enables the relatedness of any two tasks to be component dependent. Their motivation is that different components may be on different scales, so that assigning a task to a single cluster is unrealistic. We now give a view of the MSBP that is consistent with the LSM.

First, let us clarify what we mean by a component in this context. From the clustering of the effective parameter $\boldsymbol{w}_m$ given in equation 2.22, the latent function is

$$f_m(\boldsymbol{x}) = \boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{x} = (\boldsymbol{\mu}_{\mathrm{w}})^{\mathrm{T}}\boldsymbol{x} + \sum_{p=1}^{P} z_m^p (\boldsymbol{a}^p)^{\mathrm{T}}\boldsymbol{x} + \boldsymbol{\xi}_m^{\mathrm{T}}\boldsymbol{x} = (\boldsymbol{\mu}_{\mathrm{w}})^{\mathrm{T}}\boldsymbol{x} + \sum_{p=1}^{P} z_m^p \sum_{d=1}^{D} a_d^p x_d + \boldsymbol{\xi}_m^{\mathrm{T}}\boldsymbol{x}, \tag{2.26}$$

where we have expanded the dot product $(\boldsymbol{a}^p)^{\mathrm{T}}\boldsymbol{x}$ into a summation. For feature-component dependent clustering, the indicator variables z_m^ps are given an extra dimension to select each of the D feature components of $\boldsymbol{x}$ separately. That is,

$$f_m(\boldsymbol{x}) = (\boldsymbol{\mu}_{\mathrm{w}})^{\mathrm{T}}\boldsymbol{x} + \sum_{d=1}^{D}\sum_{p=1}^{P} z_{md}^p\, a_d^p\, x_d + \boldsymbol{\xi}_m^{\mathrm{T}}\boldsymbol{x}, \tag{2.27}$$

where, for a given m and d, only one of z_{md}^p $(p = 1 \ldots P)$ is one, and the rest are zeros.

A straightforward correspondence to the LSM is obtained when we identify the summation $\sum_p z_{md}^p a_d^p$ in equation 2.27 as $(A\boldsymbol{s}_m)_d$ and introduce $\boldsymbol{a}_d \stackrel{\text{def}}{=} (a_d^1, a_d^2, \ldots, a_d^P)^{\mathrm{T}}$ and $\boldsymbol{z}_{md} \stackrel{\text{def}}{=} (z_{md}^1, z_{md}^2, \ldots, z_{md}^P)^{\mathrm{T}}$:

$$A\boldsymbol{s}_m = \begin{pmatrix} \sum_{p=1}^P z_{m1}^p\, a_1^p \\ \sum_{p=1}^P z_{m2}^p\, a_2^p \\ \vdots \\ \sum_{p=1}^P z_{mD}^p\, a_D^p \end{pmatrix} = \begin{pmatrix} \boldsymbol{a}_1^{\mathrm{T}}\boldsymbol{z}_{m1} \\ \boldsymbol{a}_2^{\mathrm{T}}\boldsymbol{z}_{m2} \\ \vdots \\ \boldsymbol{a}_D^{\mathrm{T}}\boldsymbol{z}_{mD} \end{pmatrix} = \begin{pmatrix} \boldsymbol{a}_1^{\mathrm{T}} & 0 & \cdots & 0 \\ 0 & \boldsymbol{a}_2^{\mathrm{T}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \boldsymbol{a}_D^{\mathrm{T}} \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_{m1} \\ \boldsymbol{z}_{m2} \\ \vdots \\ \boldsymbol{z}_{mD} \end{pmatrix}. \tag{2.28}$$

Hence A is the block diagonal matrix of D-by-D P-dimensional row vectors, and $\boldsymbol{s}_m$ is the block vector of D indicator variables $\boldsymbol{z}_{md}$s. The constraint on $\boldsymbol{z}_{md}$ demands that $\boldsymbol{z}_{md} \sim \mathcal{M}ultinomial(\cdot \mid 1, \boldsymbol{\pi}_{md})$, where $\boldsymbol{\pi}_{md} \in \Delta^{P-1}$. The key aspect of MSBP is the following stick breaking prior over $\boldsymbol{\pi}_{md}$ that couples task relatedness with the feature components:

$$t_m^p \sim \mathcal{B}eta(\cdot \mid 1, \alpha_0^{\mathrm{t}}) \qquad r_d^p \sim \mathcal{B}eta(\cdot \mid 1, \alpha_0^{\mathrm{r}}) \qquad \nu_{md}^p \stackrel{\text{def}}{=} t_m^p r_d^p \tag{2.29a}$$

$$\pi_{md}^p = \nu_{md}^p \prod_{p'=1}^{p-1}\left(1 - \nu_{md}^{p'}\right). \tag{2.29b}$$

The effective stick-breaking weight ν_{md}^p for the $p\mathrm{th}$ cluster couples together the weight t_m^p for the $m\mathrm{th}$ task and the weight r_d^p for $d\mathrm{th}$ feature component, so the tendency for the parameter of the $d\mathrm{th}$ component for task m to belong to cluster p depends directly on the tendencies of the parameter and the task to belong to that cluster. In this way, feature-component dependent clustering is achieved. The graphical model for MSBP is shown in Figure 2.2b.

Perhaps the easiest way to understand the MSBP model is through its clustering properties given by Xue et al. [2007a, Theorem 4]:

$$P(\boldsymbol{z}_{md} = \boldsymbol{z}_{m'd}) = \frac{1}{(\alpha_0^{\mathrm{t}} + 1)(\alpha_0^{\mathrm{r}} + 2) - 1} \tag{2.30a}$$

$$P(\boldsymbol{z}_{md} = \boldsymbol{z}_{m'd}) < P(\boldsymbol{z}_{md} = \boldsymbol{z}_{m'd} | \boldsymbol{z}_{md'} = \boldsymbol{z}_{m'd'}). \tag{2.30b}$$

The first equation directly relates the hyper-parameters α_0^{t} and α_0^{r} to the (marginal) probability that both task m and task m' have the same parameter for the $d\mathrm{th}$ feature component, up to noise corrections due to $\boldsymbol{\xi}_m$. The second equation says that this probability increases when we know that the tasks also share their parameters for the $d'\mathrm{th}$ component. Hence the clustering of the $d\mathrm{th}$ component borrow information from the clustering of the $d'\mathrm{th}$ component, thereby encouraging task clusterings that are consistent across the components.

2.3 Multi-task Regularization Networks Model

In the previous section, we have looked at how multi-task learning may be achieved using shared structures Φ. We now look at how it can be achieved using shared parameters $\boldsymbol{u}$ but with structures that differ between the tasks. Although this may seem peculiar, we remind ourselves that for linear structures there is actually no clear distinction between parameters and structures (see equation 2.8). The key idea is that, under the L_2-norm, these structures will induce a *reproducing kernel Hilbert space* (RKHS), and that different notions of task relatedness correspond to different reproducing kernels. This view is put forward by Evgeniou, Micchelli, and Pontil [2005], and we shall call it the multi-task regularization networks model (MRNM).

Evgeniou et al. start from the regularization framework

$$\operatorname*{arg\,min}_{\{\boldsymbol{w}_m\}} \frac{1}{Mn} \sum_{m=1}^{M} \sum_{i=1}^{n} L\big(y_{mi}\,,\, \boldsymbol{w}_m^{\mathrm{T}} \boldsymbol{x}_{mi}\big) + \lambda \underline{\boldsymbol{w}}^{\mathrm{T}} E \underline{\boldsymbol{w}}, \tag{2.31}$$

where L is the loss function that depends on the nature of the task (see Evgeniou et al. 2000),

$$\underline{\boldsymbol{w}} \overset{\mathrm{def}}{=} (\boldsymbol{w}_1^{\mathrm{T}}, \ldots, \boldsymbol{w}_M^{\mathrm{T}})^{\mathrm{T}}, \tag{2.32}$$

and E is an MD-by-MD matrix capturing the relations between the tasks. Hence the tasks are independent in the loss function but coupled via the joint regularizer $\underline{\boldsymbol{w}}^{\mathrm{T}} E \underline{\boldsymbol{w}}$. To make the connection to regularization networks (see Evgeniou et al. 2000), they use the linear mapping

$$\boldsymbol{w}_m = \Phi_m^{\mathrm{T}} \boldsymbol{u} \qquad \text{where } \Phi_m \in \mathbb{R}^{D^{\Phi} \times D},\ \boldsymbol{u} \in \mathbb{R}^{D^{\Phi}}, \tag{2.33a}$$

so as to introduce the $m\mathrm{th}$ latent function as

$$f_m(\boldsymbol{x}) = \boldsymbol{w}_m^{\mathrm{T}} \boldsymbol{x} = \boldsymbol{u}^{\mathrm{T}} \Phi_m \boldsymbol{x}. \tag{2.33b}$$

Instead of having M functions, each acting on one task, Evgeniou et al. [2005] take the view of extending the original data space by a task indicator and having a single function on the product space $\mathbb{N}_1 \times \mathcal{X}$, where $\mathbb{N}_1 \stackrel{\text{def}}{=} \{1, 2, \ldots\}$ is the set of task indicators:[5]

$$\underline{f}(m, \boldsymbol{x}) = \boldsymbol{u}^{\mathrm{T}} \Phi_m \boldsymbol{x}. \tag{2.34}$$

The Hilbert space $\mathcal{H}$ of such functions on $\mathbb{N}_1 \times \mathcal{X}$ may now be considered. If the squared norm of all such functions $\|\underline{f}\|^2_{\mathcal{H}}$ is $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}$, then we have the following reproducing kernel

$$k((m, \boldsymbol{x}), (m', \boldsymbol{x}')) = \boldsymbol{x}^{\mathrm{T}} \Phi_m^{\mathrm{T}} \Phi_{m'} \boldsymbol{x}' \tag{2.35}$$

for the Hilbert space of $\underline{f}$; see appendix A.1.1. Evgeniou et al. call this the *linear multi-task kernel*. We may now write the regularization network problem given by (2.31) in its classical form

$$\arg\min_{\underline{f} \in \mathcal{H}} \frac{1}{Mn} \sum_{m=1}^{M} \sum_{i=1}^{n} L\big(y_{mi}\,,\, \underline{f}(m, \boldsymbol{x}_{mi})\big) + \lambda \|\underline{f}\|^2_{\mathcal{H}}. \tag{2.36}$$

More explicitly, this is

$$\arg\min_{\boldsymbol{u}} \frac{1}{Mn} \sum_{m=1}^{M} \sum_{i=1}^{n} L\big(y_{mi}\,,\, \boldsymbol{u}^{\mathrm{T}} \Phi_m \boldsymbol{x}_{mi}\big) + \lambda \boldsymbol{u}^{\mathrm{T}} \boldsymbol{u}. \tag{2.37}$$

The optimizations given by (2.31) and (2.37) are equivalent in the sense that we can convert one to the other by constructing a suitable E (given the Φ_ms) or suitable Φ_ms (given the E). This equivalence is now stated.

Proposition 2.1. *(cf. Evgeniou et al. 2005, Proposition 1)*

(a) Define $\underline{\Phi}$ to be the D^{Φ}-by-MD block matrix obtained by concatenating the matrices Φ_m, i.e.,

$$\underline{\Phi} \stackrel{\text{def}}{=} (\Phi_1 \mid \Phi_2 \mid \cdots \mid \Phi_M)\,.$$

Assume $D^{\Phi} \leqslant MD$, and that $\underline{\Phi}$ has full rank D^{Φ}. Then, for $E = \big(\underline{\Phi}^{\mathrm{T}}\underline{\Phi}\big)^{+}$, the objective functions in (2.31) *and* (2.37) *are equivalent.*

(b) Conversely, for any MD-by-MD positive semi-definite matrix E of rank $D^{\Phi} \leqslant MD$ in (2.31)*, there exists at least one $\underline{\Phi}$ such that objective functions in* (2.31) *and* (2.37) *are equivalent. For example, we may have $\underline{\Phi} = C^{\mathrm{T}} E^{+}$, where C is a matrix square root of E, such as the incomplete Cholesky decomposition of E.*

The proof is in appendix A.1.2, and it depends on showing that $\underline{\Phi} E \underline{\Phi}^{\mathrm{T}}$ is $I_{D^{\Phi}}$, the D^{Φ}-by-D^{Φ} identity matrix. This condition, $\underline{\Phi} E \underline{\Phi}^{\mathrm{T}} = I$, is one of the cornerstones in formulating the MRNM, and hence deserves further discussion. Given a fixed E which is positive semi-definite, there may be multiple $\underline{\Phi}$s that will satisfy $\underline{\Phi} E \underline{\Phi}^{\mathrm{T}}$, and the example given in Proposition 2.1b is but one; appendix A.1.2.2 gives a construction for the different $\underline{\Phi}$s. Since each setting of $\underline{\Phi}$ corresponds to a Hilbert space $\mathcal{H}$, in reality the optimization in (2.36) has to search over the possible $\mathcal{H}$, or, equivalently, the optimization in (2.37) has to search within the set of $\underline{\Phi}_m$s satisfying $\underline{\Phi} E \underline{\Phi}^{\mathrm{T}} = I$.

The discussion in the preceding paragraph is only necessary because E may not have full rank, or $D^{\Phi} \leqslant MD$, which we shall now address. Since the size of E is fixed to MD-by-MD, that of $\underline{\Phi}$ to D^{Φ}-by-MD, and matrix multiplication can never increase rank of matrices, $\underline{\Phi} E \underline{\Phi}^{\mathrm{T}}$ can be of rank at

[5] To be mathematically precise, $\mathbb{N}_1$ is isomorphic to the set of task indicators.

most $\max(MD, D^\Phi)$. However, the identity matrix I_{D^Φ} is of full rank D^Φ, which means that we require $\underline{\Phi}$ and E to be of rank at least D^Φ. The consequence is that $D^\Phi \leqslant MD$ for, if it was to be otherwise, the rank of $\underline{\Phi}$ would have been limited by MD.

This stands in contrast with the view by Evgeniou et al. [2005, section 3] that $D^\Phi \geqslant MD$. While their view makes their definition of $E = \left(\underline{\Phi}^{\mathrm{T}}\underline{\Phi}\right)^{-1}$ valid, it will not lead to $\underline{\Phi}E\underline{\Phi}^{\mathrm{T}} = I_{D^\Phi}$ (unless $D^\Phi = MD$), on which the proof of their Proposition 1 depends.

In sections 2.3.2, 2.3.3 and 2.3.4 we will look at the different notions of task-relatedness that may be implemented by specifying E, following Evgeniou et al. [2005]. But we will first see how separable dependencies may be modelled via this approach.

2.3.1 Separable Multi-task Kernels

Learning the full E matrix requires $O(MDD^\Phi)$ parameters which may be too large for the robustness of statistical estimation. Similar to the separable dependency case that we have discussed in section 2.2.1.2 for the hierarchical Bayesian latent source model, we may simplify E by modelling separately the task relations and the parameter-component relations. This is used by Evgeniou et al. [2005] to base examples of their general model. As we will see in later sections, most examples of multi-task kernels are based on such separability.

The separability is as follows. With an M-by-M matrix E^{f} modelling pair-wise task relations, and a D-by-D matrix E^{x} modelling pair-wise parameter-component relations, we can parametrize matrix E in equation 2.31 as

$$E = E^{\mathrm{f}} \otimes E^{\mathrm{x}}, \tag{2.38}$$

where $\otimes$ denotes the Kronecker product. Thus we have

$$\underline{\boldsymbol{w}}^{\mathrm{T}} E \underline{\boldsymbol{w}} = \sum_{m=1}^{M} \sum_{m'=1}^{M} E^{\mathrm{f}}_{mm'} \boldsymbol{w}_m^{\mathrm{T}} E^{\mathrm{x}} \boldsymbol{w}_m, \tag{2.39}$$

with $E^{\mathrm{f}}_{mm'}$ the $(m, m')\mathrm{th}$ entry of matrix E^{f}. By using the property that the (pseudo-)inverse of a Kronecker product is the Kronecker product of the (pseudo-)inverses, the multi-task kernel that corresponds to this is

$$k((\boldsymbol{x}, m), (\boldsymbol{x}', m')) = K^{\mathrm{f}}_{mm'} \boldsymbol{x}^{\mathrm{T}} (E^{\mathrm{x}})^{+} \boldsymbol{x}', \tag{2.40}$$

with $K^{\mathrm{f}}_{mm'}$ the $(m, m')\mathrm{th}$ entry of matrix $(E^{\mathrm{f}})^{+}$; see appendix A.1.3 on page 137 for the derivation.

Equations 2.39 and 2.40 above are analogous to the equations 20 and 19 of Evgeniou et al. [2005], except that they only consider the case where $E^{\mathrm{x}} = I_D$ and where E^{f} has full rank.

2.3.2 Manifold of Predictors[6]

We now briefly describe how manifold learning may be implemented using multi-task kernels and refer the reader to appendix A.1.4 on page 138 for details. Let $\boldsymbol{\mu}$ be the offset of the manifold from the

[6] To the best of our knowledge, the analysis in this section is novel.

origin, and let the column vectors of the D-by-D^{s} matrix A be orthonormal vectors spanning the linear subspace of the manifold. The pair $(\boldsymbol{\mu}, A)$ gives a D^{s} dimensional linear manifold residing within a D dimensional space. Let $\boldsymbol{d}_m \stackrel{\mathrm{def}}{=} (\boldsymbol{w}_m - \boldsymbol{\mu})$ be the displacement of the effective parameter $\boldsymbol{w}_m$ for task m from $\boldsymbol{u}$. The distance between $\boldsymbol{w}_m$ and its projection onto the manifold is given by $\|\boldsymbol{d}_m - AA^{\mathrm{T}}\boldsymbol{d}_m\|_2$, and the distance of the manifold from the origin is $\|\boldsymbol{u} - AA^{\mathrm{T}}\boldsymbol{u}\|_2$. The regularizer

$$\min_{\boldsymbol{\mu}} \left[\sum_{m=1}^{M} \|\boldsymbol{d}_m - AA^{\mathrm{T}}\boldsymbol{d}_m\|_2^2 + \lambda_\mu \|\boldsymbol{\mu} - AA^{\mathrm{T}}\boldsymbol{\mu}\|_2^2 \right] \tag{2.41a}$$

for learning the manifold of predictors will constrain (a) the parameters of the tasks to lie close to the manifold, and (b) the manifold to lie close to the origin. The minimization with respect to $\boldsymbol{\mu}$ can be solved analytically, and equation 2.41a can be written in the separable form $\underline{\boldsymbol{w}}^{\mathrm{T}} \left(E^{\mathrm{f}} \otimes E^{\mathrm{x}}\right) \underline{\boldsymbol{w}}$ discussed in section 2.3.1, setting

$$E^{\mathrm{f}} = I_{M\times M} - \frac{1}{\lambda_\mu + M} 1_{M\times M} \qquad\qquad E^{\mathrm{x}} = \left(I_{D\times D} - AA^{\mathrm{T}}\right)^2, \tag{2.41b}$$

where $I_{M\times M}$ is the M-by-M identity matrix and $1_{M\times M}$ is the M-by-M matrix of ones. Inverting E^{f} and E^{x} gives the following multi-task kernel for learning a low-dimensional manifold

$$k((\boldsymbol{x}, m), (\boldsymbol{x}', m')) = \left(\delta_{mm'} + \frac{1}{\lambda_{\mu}}\right) \boldsymbol{x}^{\mathrm{T}} A_\perp A_\perp^{\mathrm{T}} \boldsymbol{x}', \tag{2.41c}$$

where $A_\perp$ is a D-by-$(D - D^{\mathrm{s}})$ matrix whose column space is the orthogonal complement to that of A.

It will be instructive to construct the Φ_m-matrices that correspond to this manifold of predictors. Given $E = E^{\mathrm{f}} \otimes E^{\mathrm{x}}$ and using Proposition 2.1b on page 22, we may construct a $\underline{\Phi}$, from which we extract for the mth task

$$\Phi_m^{\mathrm{T}} = \big(\underbrace{\lambda''_{\mu} A_\perp \mid \cdots \mid \lambda''_{\mu} A_\perp}_{\text{repeat } m-1 \text{ times}} \mid \underbrace{(1+\lambda''_{\mu}) A_\perp}_{m\text{th}} \mid \underbrace{\lambda''_{\mu} A_\perp \mid \cdots \mid \lambda''_{\mu} A_\perp}_{\text{repeat } M-m \text{ times}}\big) \in \mathbb{R}^{D\times M(D-D^{\mathrm{s}})}, \tag{2.42}$$

where $\lambda''_{\mu} \geqslant 0$ is a constant that depends on M and λ_{μ}. One can also verify that, for $(D - D^{\mathrm{s}})$-by-D matrices B_m, $m = 1 \ldots M$, such that $B_m E^{\mathrm{x}} B_{m'}^{\mathrm{T}} = I_{(D-D^{\mathrm{s}})}$, the following generalization of Φ_m

$$\Phi_m^{\mathrm{T}} = \left(\lambda''_{\mu} B_1^{\mathrm{T}} \mid \cdots \mid \lambda''_{\mu} B_{m-1}^{\mathrm{T}} \mid (1+\lambda''_{\mu}) B_m^{\mathrm{T}} \mid \lambda''_{\mu} B_{m+1}^{\mathrm{T}} \mid \cdots \mid \lambda''_{\mu} B_M^{\mathrm{T}}\right). \tag{2.43}$$

is also valid; that is, $\underline{\Phi} E \underline{\Phi}^{\mathrm{T}} = I$. To satisfy $B_m E^{\mathrm{x}} B_{m'}^{\mathrm{T}} = I$, we define D-by-$(D - D^{\mathrm{s}})$ matrices F_m, $m = 1 \ldots M$, such that their column vectors are in the null space of $A_\perp^{\mathrm{T}}$, i.e., $A_\perp^{\mathrm{T}} F_m = 0$, and let

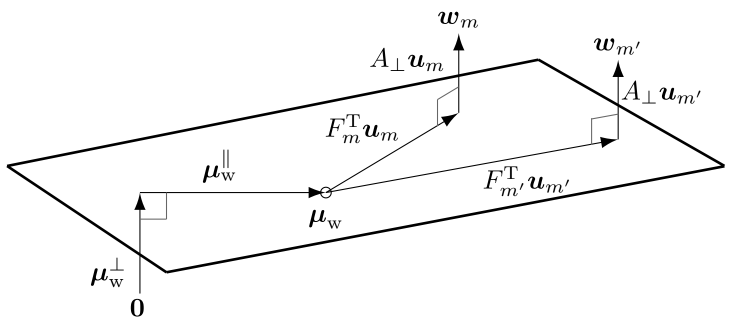
$$B_m^{\mathrm{T}} \stackrel{\mathrm{def}}{=} A_\perp + F_m. \tag{2.44}$$

If the $\boldsymbol{u}_m$s are partitions of $\boldsymbol{u}$, i.e.,

$$\boldsymbol{u}^{\mathrm{T}} = \left(\boldsymbol{u}_1^{\mathrm{T}} \mid \boldsymbol{u}_2^{\mathrm{T}} \mid \cdots \mid \boldsymbol{u}_m^{\mathrm{T}} \mid \cdots \mid \boldsymbol{u}_M^{\mathrm{T}}\right) \qquad \text{where } \boldsymbol{u}_m \in \mathbb{R}^{D-D^{\mathrm{s}}},\ \boldsymbol{u} \in \mathbb{R}^{M(D-D^{\mathrm{s}})}, \tag{2.45}$$

then the effective parameter $\boldsymbol{w}_m$ for the mth latent function is

$$\boldsymbol{w}_m = \Phi_m^{\mathrm{T}} \boldsymbol{u} = B_m^{\mathrm{T}} \boldsymbol{u}_m + \lambda''_{\mu} \sum_{m'=1}^{M} B_{m'}^{\mathrm{T}} \boldsymbol{u}_{m'} = \boldsymbol{\mu}_{\mathrm{w}} + F_m^{\mathrm{T}} \boldsymbol{u}_m + A_\perp \boldsymbol{u}_m, \tag{2.46a}$$



$\boldsymbol{w}_m$
$A_\perp \boldsymbol{u}_m$
$\boldsymbol{w}_{m'}$
$A_\perp \boldsymbol{u}_{m'}$
$F_m^{\mathrm{T}} \boldsymbol{u}_m$
$\boldsymbol{\mu}_{\mathrm{w}}^{\|}$
$F_{m'}^{\mathrm{T}} \boldsymbol{u}_{m'}$
$\boldsymbol{\mu}_{\mathrm{w}}$
$\boldsymbol{\mu}_{\mathrm{w}}^{\perp}$
$\mathbf{0}$

Figure 2.3: The constituents of $\boldsymbol{w}_m$ if tasks reside on a manifold under the multi-task regularization networks model, as given by equation 2.46. The point of reference or mean of the manifold, $\boldsymbol{\mu}_{\mathrm{w}}$, is marked by a hollow circle. The diagram shows how the parameters for tasks m and m' are generated.

where

$$\boldsymbol{\mu}_{\mathrm{w}} \stackrel{\mathrm{def}}{=} \boldsymbol{\mu}_{\mathrm{w}}^{\|} + \boldsymbol{\mu}_{\mathrm{w}}^{\perp} \qquad \boldsymbol{\mu}_{\mathrm{w}}^{\|} \stackrel{\mathrm{def}}{=} \lambda_{\mu}'' \sum_{m'=1}^{M} F_{m'}^{\mathrm{T}} \boldsymbol{u}_{m'} \qquad \boldsymbol{\mu}_{\mathrm{w}}^{\perp} \stackrel{\mathrm{def}}{=} \lambda_{\mu}'' A_\perp \sum_{m'=1}^{M} \boldsymbol{u}_{m'}. \tag{2.46b}$$

Let us interpret the above set of equations. For each F_m, its column vectors are in the null space of $A_\perp^T$, so they must be in the column space of A. Hence the vector $F_m^{\mathrm{T}} \boldsymbol{u}_m$ must be parallel to the manifold. In contrast, the vector $A_\perp \boldsymbol{u}_m$ is perpendicular to the manifold. Thus $\boldsymbol{\mu}_{\mathrm{w}}^{\perp}$ is the displacement of the manifold from the origin. Adding $\boldsymbol{\mu}_{\mathrm{w}}^{\|}$ gives $\boldsymbol{\mu}_{\mathrm{w}}$, the common point of reference on the manifold from which the parameter for each of the M tasks differs. We may regard $\boldsymbol{\mu}_{\mathrm{w}}$ as the mean of the manifold. The projection of the parameter for the $m\mathrm{th}$ task onto the manifold is at $F_m^{\mathrm{T}} \boldsymbol{u}_m$ from the $\boldsymbol{\mu}_{\mathrm{w}}$, and the parameter is displaced from the manifold by $A_\perp \boldsymbol{u}_m$. As an illustration, this process is shown in Figure 2.3 for the effective parameters for tasks m and m'.

2.3.2.1 Comparison to the ILSM approach

We now use the manifold of predictors as an example to compare the multi-task regularization networks model (MRNM) with the latent source model (LSM). Recall that the LSM model for the manifold of predictors in section 2.2.2 uses a shared effective structure A and an offset $\boldsymbol{\mu}_{\mathrm{w}}$ that together determine the manifold. The effective parameter vector for task m in the LSM model is

$$\boldsymbol{w}_m = \boldsymbol{\mu}_{\mathrm{w}} + A\boldsymbol{s}_m + \boldsymbol{\xi}_m, \tag{2.47a}$$

where $\boldsymbol{s}_m$ is the latent source that determines the location of $\boldsymbol{w}_m$ on the manifold, and $\boldsymbol{\xi}_m$ is the noise. For the MRNM model, equation 2.46a says

$$\boldsymbol{w}_m = \Phi_m^{\mathrm{T}} \boldsymbol{u} = B_m^{\mathrm{T}} \boldsymbol{u}_m + \lambda_{\mu}'' \sum_{m'=1}^{M} B_{m'}^{\mathrm{T}} \boldsymbol{u}_{m'} = \boldsymbol{\mu}_{\mathrm{w}} + F_m^{\mathrm{T}} \boldsymbol{u}_m + A_\perp \boldsymbol{u}_m. \tag{2.47b}$$

Within the LSM model, $\boldsymbol{s}_m$ and $\boldsymbol{\xi}_m$ are specific to each tasks, as indicated by the subscripts in (2.47a). In contrast for MRNM the parameter vector $\boldsymbol{u}$, which is partitioned into subvectors $\boldsymbol{u}_m$, is shared across tasks. However, the manner in which the subvectors $\boldsymbol{u}_m$s are used varies for different tasks. This is due to the different structures Φ_m for different tasks. Specifically, according to (2.43) for the manifold of predictors case, Φ_m for task m has an additional B_m at the $m\text{th}$ block, while $\Phi_{m'}$ for task m' has an additional $B_{m'}$ at the $m'\text{th}$ block. These additional B_m matrices allow the effective parameter vector $\boldsymbol{w}_m$ to vary between tasks.

Another difference between the two multi-task models is in how the offset $\boldsymbol{\mu}_{\text{w}}$ of the manifold is determined. For the LSM, the offset is generated independently of A, the $\boldsymbol{s}_m$s and the $\boldsymbol{\xi}_m$s, but, for the MRNM, the offset is determined completely by $\boldsymbol{u}$ and the Φ_ms, since

$$\boldsymbol{\mu}_{\text{w}} = \frac{1}{M + 1/\lambda''_{\mu}} \sum_{m=1}^{M} \boldsymbol{w}_m = \sum_{m=1}^{M} \Phi_m^{\text{T}} \boldsymbol{u}, \tag{2.48}$$

where the first equality can be obtained by re-arrangement after summing (2.47b) on both sides.

2.3.3 Task Clustering

Evgeniou et al. [2005, section 3.1.2] have formulated the notion of task-clustering using multi-task kernels. We briefly describe the approach here and defer the detailed derivation to appendix A.1.5.

For P clusters, let the mean of the parameters for the $p\text{th}$ cluster be denoted by $\boldsymbol{\mu}_p$. Task clustering can then be easily expressed in terms regularizing each $\boldsymbol{w}_m$ to be near its cluster centre, and regularizing each cluster centre. This is idea is expressed by the following regularizer

$$\min_{\{\boldsymbol{\mu}_p\}} \sum_{p=1}^{P} \left(\sum_{m=1}^{M} z_m^p \left(\boldsymbol{w}_m - \boldsymbol{\mu}_p\right)^{\text{T}} E_p^{\text{x}} \left(\boldsymbol{w}_m - \boldsymbol{\mu}_p\right) + \boldsymbol{\mu}_p^{\text{T}} E^{\text{x}} \boldsymbol{\mu}_p \right), \tag{2.49}$$

where z_m^p is an indicator variable that is one if the $m\text{th}$ task belongs to the $p\text{th}$ cluster and zero otherwise, E_p^{x} defines the Mahalanobis metric of the $p\text{th}$ cluster, and E^{x} defines the Mahalanobis metric of the entire parameter space. This regularizer generalizes equation 26 of Evgeniou et al. with the Mahalanobis metrics. If we assume a common metric up to uniform expansions and contractions in the parameter space, i.e., $E_p^{\text{x}} = \lambda^p E^{\text{x}}$, then the regularizer simplifies to

$$\sum_{m=1}^{M} \sum_{m'=1}^{M} \left(\sum_{p=1}^{P} z_m^p z_{m'}^p \left(\delta_{mm'} \lambda^p - \frac{(\lambda^p)^2}{1 + z^p \lambda^p} \right) \right) \boldsymbol{w}_m^{\text{T}} E^{\text{x}} \boldsymbol{w}_{m'}^{\text{T}}, \tag{2.50}$$

where $z^p \overset{\text{def}}{=} \sum_{l=m}^{M} z_m^p$ is the number of tasks in cluster p. This is a separable model in the manner described in section 2.3.1. The corresponding multi-task kernel is

$$k((\boldsymbol{x}, m), (\boldsymbol{x}', m')) = \sum_{p=1}^{P} z_m^p z_{m'}^p \left(\delta_{mm'} \frac{1}{\lambda^p} + 1 \right) \boldsymbol{x}^{\text{T}} (E^{\text{x}})^{-1} \boldsymbol{x}'. \tag{2.51}$$

2.3.4 Correlated Tasks

Given an M-by-M symmetric matrix G in which the $(m, m')\text{th}$ element $g_{mm'} \in [0, 1]$ gives a measure of relatedness between the $m\text{th}$ and $m'\text{th}$ task under a Mahalanobis metric E^{x}, we can have the following regularizer for learning the task hypotheses

$$\frac{1}{2} \sum_{m=1}^{M} \sum_{m'=1}^{M} g_{mm'} (\boldsymbol{w}_m - \boldsymbol{w}_{m'})^{\text{T}} E^{\text{x}} (\boldsymbol{w}_m - \boldsymbol{w}_{m'}). \tag{2.52}$$

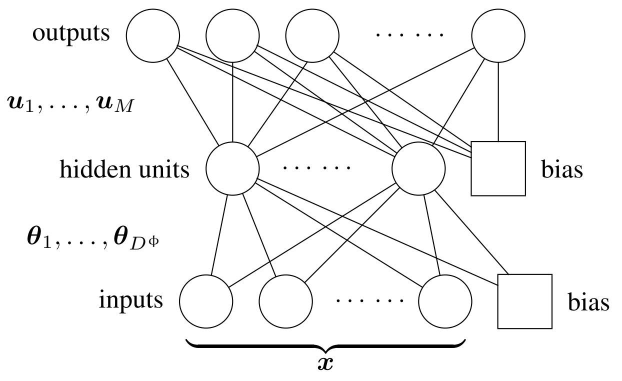

outputs
$\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M$
hidden units
bias
$\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{D^\Phi}$
inputs
bias
$\boldsymbol{x}$

Figure 2.4: Bayesian multi-task neural network model (adapted from Bakker and Heskes [2003, Figure 1]). There are D^Φ hidden units and M outputs (tasks); $\boldsymbol{\theta}_i$ is the fan-in weights for the $i\text{th}$ hidden unit, and $\boldsymbol{u}_m$ is the fan-in weights for the $m\text{th}$ output. The common transfer function g for the hidden units in the neural network is a sigmoid function, for example.

This is a simple generalization of the regularizer given by Evgeniou et al. [2005, section 3.1.3], where they have $E^{\text{x}} = I$. This can be expressed as the separable form $\underline{\boldsymbol{w}}^{\text{T}} \left(E^{\text{f}} \otimes E^{\text{x}}\right) \underline{\boldsymbol{w}}$, with $E^{\text{f}} = D - G$, where D is an M-by-M diagonal matrix with the $m\text{th}$ diagonal element $\sum_{m'=1}^{M} g_{mm'}$. As noted by Evgeniou et al., E^{f} is the graph Laplacian of G.

The multi-task kernel can be directly derived by taking the inverses of E^{f} and E^{x}, though for E^{f} one actually has to take its pseudo-inverse, since a graph Laplacian is guaranteed to be rank deficient. However, we note that this causes no problem with our Proposition 2.1 on page 22.

2.4 Bayesian Multi-task Learning with Neural Networks

So far the models we have encountered are those for which the structures $\{\phi_m(\boldsymbol{x})\}$ are linear transformations. For the LSM, these structures are shared among tasks, while for the MRNM they differ. In this section, we will discuss the work of Bakker and Heskes [2003] in which the shared structure is a neural network. Their model is given by a single hidden-layer neural network with multiple outputs:

$$f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\text{T}} \boldsymbol{\phi}(\boldsymbol{x}) \qquad \boldsymbol{u}_m \in \mathbb{R}^{D^\Phi},\ \boldsymbol{\phi} : \mathbb{R}^D \mapsto \mathbb{R}^{D^\Phi} \tag{2.53a}$$

$$\phi_i(\boldsymbol{x}) = g(\boldsymbol{\theta}_i^{\text{T}} \boldsymbol{x}) \qquad \boldsymbol{\theta} \in \mathbb{R}^D, \tag{2.53b}$$

where $\phi_i(\cdot)$ is the $i\text{th}$ function of the vector function $\boldsymbol{\phi}(\cdot)$, and it depends on a common transfer function g and its vector of parameters $\boldsymbol{\theta}_i$. The neural network view of this is shown by Figure 2.4, where $\boldsymbol{\theta}_i$ is the vector of fan-in weights for the $i\text{th}$ hidden unit, and g is the common transfer function in the neural network. For feedforward neural networks, a common setting for g is the hyperbolic tangent function. Although the figure includes the bias terms (represented by squares) on the hidden layer and the output layer, we have omitted them in (2.53) to simplify our notation. It is straightforward to generalize our following discussion by including the bias as a constant input feature (for the hidden unit) or as a constant hidden unit (for the output layer).

Bakker and Heskes [2003] model different notions of task-relatedness by placing different priors on the $\boldsymbol{u}_m$s, which are the fan-in weights for the multiple outputs or tasks, and this is the Bayesian multi-task neural networks model (BMNNM). This model is similar to the latent source model (LSM) of Zhang [2006]: both are generative models where priors are placed over the model parameter vector $\boldsymbol{u}_m$. For the LSM, the latent function is given by equation 2.13b on page 15, which we recall below:

$$f_m(\boldsymbol{x}) = \boldsymbol{w}_m^{\mathrm{T}} \boldsymbol{x}. \tag{2.54}$$

Comparing the above equation with equation 2.53a, the correspondence is clear: the $\boldsymbol{u}_m$ in BMNNM is to the $\boldsymbol{w}_m$ in LSM, as the the $\boldsymbol{\phi}(\boldsymbol{x})$ in BMNNM is to the $\boldsymbol{x}$ in LSM. The difference is that the BMNNM consists of an additional layer of indirection $\boldsymbol{\phi}(\cdot)$ from the input $\boldsymbol{x}$ that has to be determined by learning from data.

The similarity between BMNNM and LSM means that both model task relatedness in similar ways. Indeed, our subsequent discussion of task relatedness in BMNNM will often refer to the LSM. However, before we discuss the specific examples, we first highlight that two specific models proposed by Bakker and Heskes [2003, §4.1 and §4.3] make use of task descriptors or features, which we shall denote by $\boldsymbol{t}_m$ for the $m\mathrm{th}$ task. They have investigated their methods for the ILEA school data set and the Dutch newspaper sales data set, both of which have task descriptors for each task. Our opinion, as argued in section 1.2, is that using task descriptors in a multi-task setting may not be fruitful, as demonstrated by Evgeniou et al. [2005] on the ILEA data. In addition, we maintain that task-descriptor features can be either unavailable or difficult to define correctly for many real-life scenarios. Keeping this opinion of task descriptors in mind, we now discuss the three specific models of Bakker and Heskes.

2.4.1 Task-dependent Prior Mean and Manifold of Predictors

Bakker and Heskes [2003, §4.1] have proposed placing priors with different means for each task:

$$\boldsymbol{u}_m \sim \mathcal{N}(\cdot \mid A\boldsymbol{t}_m, \Sigma^{\xi}), \tag{2.55}$$

where $\boldsymbol{t}_m$ is the task descriptor for the $m\mathrm{th}$ task, and A is the shared matrix for mapping task descriptors to prior means. The above may be expressed as

$$\boldsymbol{u}_m = A\boldsymbol{t}_m + \boldsymbol{\xi}_m \qquad\qquad \boldsymbol{\xi}_m \sim \mathcal{N}(\cdot \mid \boldsymbol{0}, \Sigma^{\xi}). \tag{2.56}$$

If the dimension of the task descriptors is less than D^{Φ}, then this is a manifold of predictors model: the manifold is the column space of A centred at the origin, and the task descriptor $\boldsymbol{t}_m$ gives the location of $\boldsymbol{u}_m$ on the manifold modulo noise $\boldsymbol{\xi}_m$. In the latent source model, the decomposition of $\boldsymbol{w}_m$ is similar (see equation 2.18 on page 17); the role of the latent source $\boldsymbol{s}_m$ there is played by the task descriptor $\boldsymbol{t}_m$ here.

2.4.2 Task Clustering and Gating

Task clustering can be implemented with a Gaussian mixture model (GMM) for the task parameters [Bakker and Heskes, 2003, §4.2]:

$$\boldsymbol{u}_m \sim \sum_{p=1}^{P} \pi^p \mathcal{N}(\cdot|\boldsymbol{\mu}_\mathrm{u}^p, (\Sigma^\xi)^p) \qquad \boldsymbol{\pi} \overset{\text{def}}{=} (\pi^1, \pi^2, \ldots, \pi^P) \in \Delta^{P-1}, \tag{2.57}$$

where P is the number of mixture components, the π^ps are the mixture proportions, and $\boldsymbol{\mu}_\mathrm{u}^p$ and $(\Sigma^\xi)^p$ are the mean and covariance of the $p\mathrm{th}$ component. Again, this is similar to the GMM on $\boldsymbol{w}_m$ for the latent source model, given by equation 2.23 on page 18.

This model does not *a priori* distinguish among tasks: before observing the data, all tasks have the same probability of being in each cluster. Therefore, with few examples per task, this is the less preferred model compared with the following gating model [Bakker and Heskes, 2003, §4.2, §5.4].

In the gating model, the mixture proportions in equation 2.57 vary with the task:

$$\boldsymbol{u}_m \sim \sum_{p=1}^{P} \pi_m^p \mathcal{N}(\cdot|\boldsymbol{\mu}_\mathrm{u}^p, (\Sigma^\xi)^p) \qquad \boldsymbol{\pi}_m \overset{\text{def}}{=} (\pi_m^1, \pi_m^2, \ldots, \pi_m^P) \in \Delta^{P-1}. \tag{2.58}$$

Bakker and Heskes [2003, §4.3] parameterize $\boldsymbol{\pi}_m$ by task-descriptors using soft-max [Bridle, 1990], i.e.,

$$\pi_m^p = \frac{\exp A^p \boldsymbol{t}_m}{\sum_{p'}^{P} \exp A^{p'} \boldsymbol{t}_m}, \tag{2.59}$$

and learning the matrices A^ps shared among the tasks. In this way, the *a priori* probability of a task being in each cluster is now dependent on its task descriptor. As pointed out by Bakker and Heskes, this is a mixture of experts model [Jacobs et al., 1991] for each task, but with the task descriptors as inputs into the gating network given by equation 2.59.

In the absence of task descriptors, we may place a common prior over the $\boldsymbol{\pi}_m$s:

$$\boldsymbol{\pi}_m \sim \mathcal{Dir}(\alpha^1, \ldots, \alpha^P) \qquad \alpha^p > 0. \tag{2.60}$$

In this way, there are also task specific mixture proportions, and we can ensure that these proportions do not "run-away" if the Dirichlet distribution is unimodal, i.e., if $\sum_{p=1}^{P} \alpha^p > 1$.

2.5 Gaussian Processes for Multi-task Learning

We will now look at Gaussian Processes (GPs, see e.g., Rasmussen and Williams 2006) for multi-task learning. In section 2.5.1, we will first introduce a simple model of multi-task learning, which is called the intrinsic correlation model (ICM), that we have found to be effective for machine learning tasks. A generalization of the ICM model by summing a number of them will be given in section 2.5.2; this is known as the linear model of coregionalization (LMC). Both the ICM and LMC are models known in the geostatistics community. In section 2.5.3, we will review selected literature that also use GPs for multi-task learning. In section 2.5.4, we will see how the LMC model can be understood in terms of latent processes. Sections 2.5.5, 2.5.6 and 2.5.7 will review the latent source model, the regularization network model and the neural network model from the viewpoint of Gaussian processes. Lastly, multi-task learning with GPs across two different input domains $\mathcal{X}_1$ and $\mathcal{X}_2$ will be outlined in section 2.5.8.

2.5.1 A Multi-task GP Model: the Intrinsic Correlation Model

Gaussian Processes is a convenient way to think in terms of functions and their relations in multi-task learning by directly defining mean functions $\langle f_m(\boldsymbol{x})\rangle$ and covariance functions $\mathbb{C}(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}'))$. We use $\mathbb{C}(a, b)$ to denote the covariance between two random variables a and b, while we use $\langle ab\rangle$ to denote the second moment between a and b; the two are the same when the either or both a and b have zero means, since $\mathbb{C}(a, b) = \langle ab\rangle - \langle a\rangle \langle b\rangle$.

One form of covariance function that can be defined with relative ease separates the covariance due to tasks and the covariance due to inputs, i.e.,

$$\langle f_m(\boldsymbol{x})\rangle = \mu_m(\boldsymbol{x}) \qquad \mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')\right) = k^{\mathrm{f}}(m, m')k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}'), \tag{2.61a}$$

where k^{f} and k^{x} are positive definite functions that explain the task relations and input relations respectively. This is also the *separability* assumption that is widely discussed in the context of time-space processes.

For subsequent discussion, we shall call this the *intrinsic correlation model* (ICM). It is also useful to consider the following case. Given M tasks, we have M functions, and function values

$$\boldsymbol{f}(X) \stackrel{\text{def}}{=} (f_1(\boldsymbol{x}_1), \ldots, f_1(\boldsymbol{x}_n), \ldots, f_M(\boldsymbol{x}_1), \ldots, f_M(\boldsymbol{x}_n))^{\mathrm{T}} \in \mathbb{R}^{Mn} \tag{2.61b}$$

at a common set of n inputs $X \stackrel{\text{def}}{=} \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. Let K be the covariance matrix of $\boldsymbol{f}(X)$, and let K^{f} (resp. K^{x}) be the covariance matrix of the functions (resp. inputs) given by the covariance function k^{f} (resp. k^{x}). Then

$$K = K^{\mathrm{f}} \otimes K^{\mathrm{x}}, \tag{2.61c}$$

where $\otimes$ is the Kronecker product operator.

Typically, the covariance function for inputs $k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$ is parameterized, and its parameters are learnt from data by maximizing the marginal likelihood of the model given the data (see Rasmussen and Williams [2006, chapter 5] and chapter 4 of this thesis). If the $m\mathrm{th}$ task has task descriptor $\boldsymbol{t}_m$, the same may be done for the covariance function for tasks, since we may write $k^{\mathrm{f}}(m, m') \equiv k^{\mathrm{f}}(\boldsymbol{t}_m, \boldsymbol{t}_{m'})$. However, if task descriptors are not available, then the structure of the covariance matrix K^{f} between the tasks has to be determined in some other ways. We list some below.

- The most general approach is to only constrain K^{f} to be positive semi-definite, so that it is a valid covariance matrix (though it may be singular).
- For the purpose of understanding task-relatedness, it makes sense to fix the relatedness of a task to itself to, say, one. In line with this, we normalize K^{f} to be a correlation matrix, so that it has ones along its diagonal. Then the extent of relatedness between any two tasks, task m and task m', is measured by their correlation $K^{\mathrm{f}}_{mm'}$, the $(m, m')\mathrm{th}$ entry in K^{f}. If there is reason to believe all pairs of tasks to be similarly related, then we can let K^{f} be an equi-correlated matrix, where all the inter-task correlations are the same.

- If K^{f} is block diagonal with P blocks, then we have task clustering with P clusters, where tasks in the same block are from the same cluster. Explained in terms of probabilistic independence, this says that tasks not in the same block are marginally independent. To relate to the other multi-task models later, it is useful to consider a construction of K^{f} that makes use of indicator variables z_m^ps. Let $z_m^p = 1$ if and only if the $m\mathrm{th}$ task belongs to the $p\mathrm{th}$ cluster, and otherwise it is zero; thus, for each m, only one of $z_m^1, \ldots, z_m^P$ is 1, since a task can only belong to one cluster. We assume for now that tasks in the same block have perfect positive correlation. Then

$$K^{\mathrm{f}}_{mm'} = \sum_{p=1}^{P} z_m^p z_{m'}^p, \tag{2.62a}$$

or, equivalently,

$$K^{\mathrm{f}} = Z^{\mathrm{T}} Z, \tag{2.62b}$$

where $Z \overset{\mathrm{def}}{=} (\boldsymbol{z}_1 \mid \boldsymbol{z}_2 \mid \ldots \mid \boldsymbol{z}_M)$ is a P-by-M matrix, and $\boldsymbol{z}_m \overset{\mathrm{def}}{=} (z_m^1, z_m^2, \ldots, z_m^P)^{\mathrm{T}}$ is the vector of indicator variables for task M. A simple way to allow tasks in the same block to have correlation other than one is to entry-wise product the above with an M-by-M correlation matrix R, i.e.,

$$K^{\mathrm{f}} = (Z^{\mathrm{T}} Z) \odot R, \tag{2.62c}$$

where $\odot$ is the Hadamard product operator. This construction is useful when it is not visually obvious that K^{f} is block diagonal.

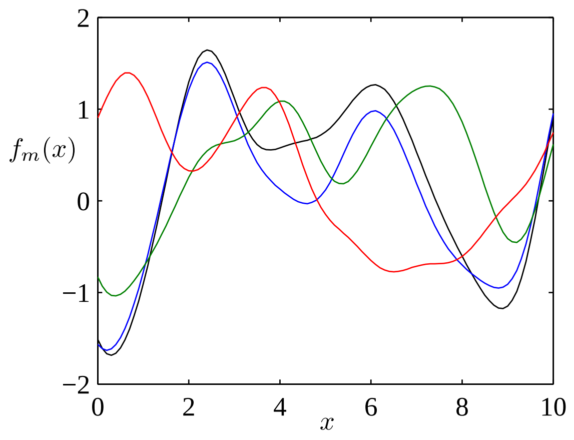
- If K^{f} is of rank $r < M$, then the functions for the tasks lie on a r dimensional manifold in the task space. We have used this approach in [Bonilla et al., 2008] for predicting speed-ups of compiled programs under different code transformations and examination scores of students in different schools. This is also the model used in chapter 4 where the rank r is selected using the *Akaike information criterion with corrections* (AICc, Hurvich and Tsai [1989]). The rank constrained K^{f} will be discussed further in section 2.5.4.

- If K^{f} is a diagonal matrix, then all the functions are independent, though they are identically distributed because they share the same covariance function k^{x}.

Henceforth, unless otherwise stated, we shall assume that task descriptors are not available, and K^{f} is modelled directly using one of the ways listed above. Figure 2.5 on the following page gives three examples implementing the various notions of task relatedness using the ICM: (a) the varying degree of relatedness, (b) task clustering, and (c) 2-dimensional manifold of functions. In the example for the manifold of functions, which is Figure 2.5c, the functions for the black and the blue tasks are independent, so we may call them the independent components of the manifold. The functions for the green and the red tasks are convex combinations of the functions for the black and the blue tasks, so they lie on the manifold defined by the black and the blue tasks. To be precise, we have set

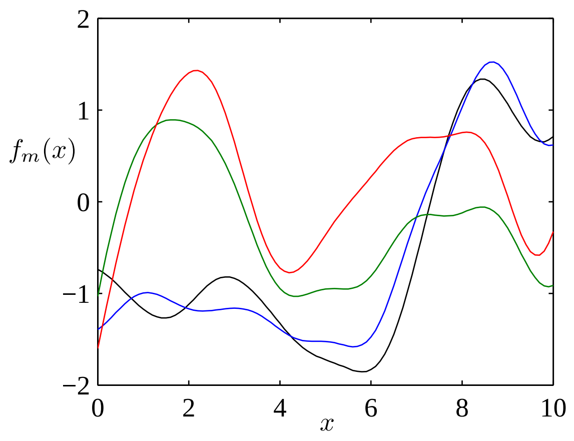
$$f_{\mathrm{green}} = 0.8 * f_{\mathrm{black}} + 0.2 * f_{\mathrm{blue}} \tag{2.63a}$$

$$f_{\mathrm{red}} = 0.2 * f_{\mathrm{black}} + 0.8 * f_{\mathrm{blue}} \tag{2.63b}$$



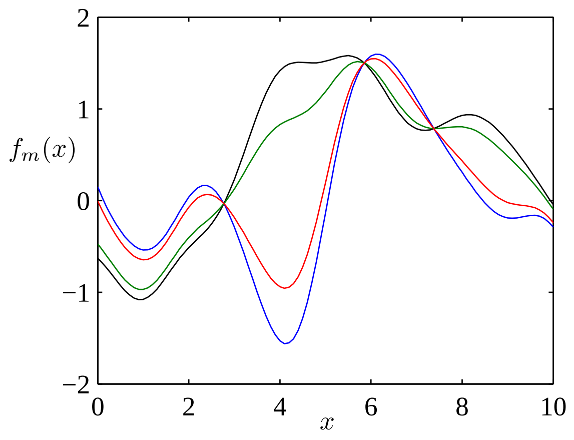
$$K^{\mathrm{f}} = \begin{array}{c|cccc} & \bullet & \bullet & \bullet & \bullet \\ \hline \bullet & 1.0 & 0.9 & 0.7 & 0.1 \\ \bullet & 0.9 & 1.0 & 0.4 & 0.2 \\ \bullet & 0.7 & 0.4 & 1.0 & 0.1 \\ \bullet & 0.1 & 0.2 & 0.1 & 1.0 \end{array}$$

(a) Tasks have varying degrees of correlation or relatedness. The black and the blue tasks are highly correlated with 0.9, so their sample functions are similar. The next highly correlated are the black and the green tasks with 0.7 correlation. Hence we see that the sample function for the green is also similar to that for the black, but not as much as the blue is.



$$K^{\mathrm{f}} = \begin{array}{c|cccc} & \bullet & \bullet & \bullet & \bullet \\ \hline \bullet & 1.0 & 0.8 & 0 & 0 \\ \bullet & 0.8 & 1.0 & 0 & 0 \\ \bullet & 0 & 0 & 1.0 & 0.8 \\ \bullet & 0 & 0 & 0.8 & 1.0 \end{array}$$

(b) Task clustering. Both the sample functions and the K^{f} matrix show that the black and the blue tasks form one cluster while the green and the red tasks form another cluster. Tasks within each cluster can differ, since their correlations are less than one.



$$K^{\mathrm{f}} = \begin{array}{c|cccc} & \bullet & \bullet & \bullet & \bullet \\ \hline \bullet & 1.0 & 0 & 0.8 & 0.2 \\ \bullet & 0 & 1.0 & 0.2 & 0.8 \\ \bullet & 0.8 & 0.2 & 0.68 & 0.32 \\ \bullet & 0.2 & 0.8 & 0.32 & 0.68 \end{array}$$

(c) Tasks on a linear manifold. The matrix K^{f} is of rank two, so we expect the tasks to lie on a two-dimensional linear manifold. The black and the blue tasks are independent, as indicated by the top right 2-by-2 sub-matrix of K^{f}, while the green and the red tasks are linear combinations of the black and the blue tasks. In fact, the linear combinations are convex combinations, so the sample functions for the red and the green tasks are enveloped by the sample functions for the black and the blue tasks.

Figure 2.5: Sample functions from the ICM implementing different notions of task-relatedness for four tasks, which are color-coded with black (●), blue (●), green (●) and red (●). The sample paths are shown on the left, while the corresponding K^{f}s are shown on the right.

so that

$$K^{\mathrm{f}} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0.8 & 0.2 \\ 0.2 & 0.8 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1.0 & 0 & 0.8 & 0.2 \\ 0 & 1.0 & 0.2 & 0.8 \\ 0.8 & 0.2 & 0.68 & 0.32 \\ 0.2 & 0.8 & 0.32 & 0.68 \end{pmatrix}. \tag{2.63c}$$

2.5.2 Linear Model of Coregionalization

We can extend the intrinsic correlation model (ICM) in the previous section by using a sum of P separable covariances. This can be understood as using P groups of M functions that have identical and independent distributions given by the ICM:

$$\langle f_m^p(\boldsymbol{x})\rangle = \mu_m^p(\boldsymbol{x}) \qquad \mathbb{C}\left(f_m^p(\boldsymbol{x}), f_{m'}^p(\boldsymbol{x'})\right) = k^{\mathrm{f}p}(m,m')k^{\mathrm{x}p}(\boldsymbol{x},\boldsymbol{x}') \qquad p = 1\ldots P. \tag{2.64a}$$

If we let $f_m(\boldsymbol{x}) \overset{\text{def}}{=} \sum_{p=1}^{P} f_m^p(\boldsymbol{x})$, then

$$\langle f_m(\boldsymbol{x})\rangle = \sum_{p=1}^{P} \mu_m^p(\boldsymbol{x}) \qquad \mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x'})\right) = \sum_{p=1}^{P} k^{\mathrm{f}p}(m,m')k^{\mathrm{x}p}(\boldsymbol{x},\boldsymbol{x}'). \tag{2.64b}$$

This the *linear model of coregionalization* (LMC). For a common set of n inputs $X \overset{\text{def}}{=} \{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n\}$, let K be the covariance matrix of the Mn vector of responses given by the M functions, and let $K^{\mathrm{f}p}$ (resp. $K^{\mathrm{x}p}$) be the covariance matrix of the functions (resp. inputs) given by the covariance function $k^{\mathrm{f}p}$ (resp. $k^{\mathrm{x}p}$). Then

$$K = \sum_{p=1}^{P} K^{\mathrm{f}p} \otimes K^{\mathrm{x}p}, \tag{2.64c}$$

which is a sum of Kronecker products. This can be used for modelling different task relatedness for different input covariance functions, since each $k^{\mathrm{x}p}$ can be different, and $k^{\mathrm{f}p}$ can model the relation between tasks under that input covariance function. For instance, relatedness under dot product covariance functions may be different from under squared-exponential covariance functions. An interesting case of this is when each $k^{\mathrm{x}p}$ only accounts for a particular component or strata of the input space, so that we model component-dependent task relatedness (see section 2.2.4 for a different model to achieve the same aim).

Although the LMC is more flexible and more powerful than the ICM, in reality it comes with the additional burden of having to either specify appropriate priors or have sufficient data in order to effectively learn the all parameters for the P covariance functions. Hence unless there are sufficient reasons to believe that there are components or strata within the input space that have properties different enough to warrant the flexibility of the LMC, it may well be better to use the simpler ICM. This principle of preferring simpler models to complex models is known as the Ockham's razor: plurality should not be posited without necessity. This principle is often used in machine learning; see, for example, MacKay [1991, chapter 2] and Rasmussen and Ghahramani [2001]. However, it must be admitted that some, Diggle and Ribeiro Jr. [2006, §3.12] for instance, do find the preference for the ICM rather unnatural.

2.5.3 Related Literature

We have called a Gaussian process model with covariance of the form $K^{\mathrm{f}}_{mm'}k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$ the *intrinsic correlation model* (ICM). In geostatistics, the term usually refers to a more specific model for *cokriging* where $k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}) = 1$, i.e., a correlation function [Wackernagel, 1998, chapter 22]. Conversely, if K^{f} is a correlation matrix, then we have the *proportional covariance model* [Chilès and Delfiner, 1999, §5.4.2]. In either case, if we have P of these and sum them in the manner described in section 2.5.2, we obtain the *linear model of coregionalization* (LMC) used in geostatistics. Section 2.7.2 contains further relevant discussion of multivariate geostatistics.

In the context of multi-task learning, models of the form given by the ICM

$$\langle f_m(\boldsymbol{x}) \rangle = \mu_m(\boldsymbol{x}) \qquad\qquad \mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')\right) = k^{\mathrm{f}}(m, m')k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') \tag{2.65}$$

have also been investigated. One simple model is to restrict k^{f} to a delta function or, equivalently, K^{f} to an identity matrix so that K is block diagonal with M n-by-n blocks. This model implies that the tasks are drawn independently from a common Gaussian process prior with covariance function k^{x}. It has been explored by the following authors:

1. Minka and Picard [1997] propose that the functions $\{f_m\}$ have independent and identical Gaussian process distributions. Using this i.i.d. model, Lawrence and Platt [2004] estimate the parameters of the covariance function k^{x} by maximizing the joint likelihood of the parameters using data from all tasks simultaneously.

2. Yu et al. [2005, 2006] place an inverse-Wishart prior on the kernel matrix K^{x} and learn the maximum a posterior (MAP) estimate of K^{x}, using the joint likelihood that includes data from all tasks. Their inverse-Wishart prior has a base covariance matrix induced by a fixed covariance function k^{x} on all observed input locations.

3. The approach of Schwaighofer et al. [2005] is similar to the above, except that their base covariance matrix for the inverse-Wishart prior is not induced by a covariance function — it is simply a positive-definite matrix. To generalize to new input locations, they use a kernel smoother on the scaled eigenvectors of the learnt K^{x} to give scaled eigenfunctions. These scaled eigenfunctions are later recombined to generate a covariance function.

A slightly different model is the Bayesian hierarchical Gaussian process model proposed by Menzefricke [2000], where K is also block diagonal, but each block may be different, unlike the methods listed above. In his model, the function for the $m\mathrm{th}$ task is drawn from a Gaussian process with covariance function $k^{\mathrm{x}}(\cdot, \cdot; \boldsymbol{\theta}_m)$, where $\boldsymbol{\theta}_m$ is the hyper-parameters for the $m\mathrm{th}$ task. Hence, each task is associated with a covariance function with the same parametrization but with different parameters $\boldsymbol{\theta}_m$. Each $\boldsymbol{\theta}_m$ is drawn independently from a common prior, and Menzefricke finds the posterior distributions over the $\boldsymbol{\theta}_m$s given observed data.

If the tasks are a priori related, then K^{f} will differ from the identity matrix. Such models have also been explored in the literature:

4. If task descriptors $\boldsymbol{t}_m$s are available, then one may use any covariance function for k^{f} on the space of task descriptors. For example, Bonilla et al. [2007] use the squared-exponential covariance

function for k^{f}. This can be rather restrictive in the following sense: if the chosen covariance function is unsuitable, or if the task descriptors are not defined correctly, then one may not be able to obtain optimal results. Indeed, this has been observed by us in [Bonilla et al., 2008] for predicting the speed-ups of eleven complied programs using different code transformations: using a k^{f} that is only constrained to be positive semi-definite outperforms using the squared exponential covariance function.

5. In the context of relational link prediction, Yu et al. [2007] use the general form $K = K^{\mathrm{f}} \otimes K^{\mathrm{x}}$, and place inverse-Wishart priors on K^{f} and K^{x}. Similar to [Yu et al., 2005, 2006] discussed in item 2 above, the base covariance matrix of each inverse-Wishart prior is induced by a fixed covariance function on all observed data.

6. In the context of emulating multi-output computer codes, Conti and O'Hagan [2010] also use the model $K = K^{\mathrm{f}} \otimes K^{\mathrm{x}}$. However, their matrix K^{x} is induced directly by a covariance function, instead of going through an inverse-Wishart prior. On K^{f}, they place a vague prior that is inversely proportional to $|K^{\mathrm{f}}|^{(M+1)/2}$; Geisser [1965] has given a comprehensive discussion of this prior.

7. More recently, Yu et al. [2009] use a non-parametric random effects model with constant mean and covariance

$$K = (K^{\mathrm{f}} + \tau I) \otimes K^{\mathrm{x}}, \tag{2.66}$$

and place an inverse-Wishart prior on K^{x} based on a covariance function k^{x}. For the purpose of large-scale implementation, they constrain K^{f} to have low rank.

8. By linearly mixing functions drawn from P latent Gaussian processes, the *semiparametric latent factor model* (SLFM) of Teh et al. [2005] is a LMC with

$$K = (A \otimes I_{N\times N}) \left(\sum_{p=1}^{P} E^{pp} \otimes K^{\mathrm{x}p} \right) (A \otimes I_{N\times N})^{\mathrm{T}} = \sum_{p=1}^{P} \left(A E^{pp} A^{\mathrm{T}} \right) \otimes K^{\mathrm{x}p}, \tag{2.67}$$

where $K^{\mathrm{x}p}$ is the covariance matrix for the $p\mathrm{th}$ latent process, A is the M-by-P mixing matrix, and E^{pp} is a P-by-P matrix with one at its $(p,p)\mathrm{th}$ entry and zero elsewhere. Thus, each $K^{\mathrm{f}p} = A E^{pp} A^{\mathrm{T}}$ is a rank one matrix in Teh et al.'s model. We shall touch on this again in the next section.

2.5.4 Linear Combination of Latent Processes

The ICM and LMC multi-task models may be understood using a shared feature space. If we set $\boldsymbol{\psi}(\cdot) \overset{\mathrm{def}}{=} \boldsymbol{\phi}_1(\cdot) = \ldots \boldsymbol{\phi}_m(\cdot)$ in equation 2.4b, then

$$f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\mathrm{T}} \boldsymbol{\psi}(\boldsymbol{x}) = \sum_{p=1}^{P} u_m^p \psi^p(\boldsymbol{x}), \tag{2.68}$$

where $P = D^{\Phi}$ is the dimension of the feature space, u_m^p is the $p\mathrm{th}$ entry in $\boldsymbol{u}_m$, and ψ^p is the $p\mathrm{th}$ function of the vector function $\boldsymbol{\psi}$. Thus each task is a different linear combination of the same set of P latent functions. If the latent functions are independent and the $p\mathrm{th}$ latent function has GP prior with

$\mathbb{C}(\phi^p(\boldsymbol{x}), \phi^p(\boldsymbol{x})) = k^{\mathrm{x}p}(\boldsymbol{x}, \boldsymbol{x}')$, and if the $\boldsymbol{u}_m$s are treated as fixed parameters, then

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x'})\right) = \sum_{p=1}^{P} u_m^p u_{m'}^p k^{\mathrm{x}p}(\boldsymbol{x}, \boldsymbol{x}'). \tag{2.69}$$

This is the SLFM of Teh et al. [2005] discussed in item 8 of the preceding section. Here the $p\mathrm{th}$ coefficients $u_m^p u_{m'}^p$ can be viewed as elements from the rank one matrices $\boldsymbol{u}^p(\boldsymbol{u}^p)^{\mathrm{T}}$, where

$$\boldsymbol{u}^p \overset{\mathrm{def}}{=} (u_1^p, \ldots, u_M^p)^{\mathrm{T}}. \tag{2.70}$$

If the latent functions have identical GP priors, i.e., $k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') = k^{\mathrm{x}1}(\boldsymbol{x}, \boldsymbol{x}') = \ldots = k^{\mathrm{x}P}(\boldsymbol{x}, \boldsymbol{x}')$, then we have the ICM with $\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x'})\right) = K^{\mathrm{f}}_{mm'} k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$, where $K^{\mathrm{f}}_{mm'}$ is the $(m, m')\mathrm{th}$ entry of

$$K^{\mathrm{f}} = \sum_{p=1}^{P} \boldsymbol{u}^p(\boldsymbol{u}^p)^{\mathrm{T}}. \tag{2.71}$$

Hence the rank of K^{f} is at most P. This construction will be used in section 4.4 for modelling the inverse dynamics of a robot manipulator handling multiple objects, where the rank P will be chosen using the *Akaike information criterion with corrections* (AICc). If $P < M$, then all the M functions lie on a manifold of predictors that is the P-dimensional linear span of the latent functions. An example of this has been shown in Figure 2.5c on page 32.

2.5.5 Latent Source Model from the Gaussian Process Viewpoint

The latent source model (LSM) discussed in section 2.2 gives a distribution over the functions $f_m(\boldsymbol{x})$ defined as

$$f_m(\boldsymbol{x}) = (\boldsymbol{\mu}_{\mathrm{w}} + A\boldsymbol{s}_m + \boldsymbol{\xi}_m)^{\mathrm{T}}\boldsymbol{x}. \tag{2.72}$$

In general, the LSM's distributions on the functions are not Gaussian processes. Nevertheless, it is worthwhile to know the two cases when they are.

1. If A is a fixed parameter, and $\boldsymbol{\mu}_{\mathrm{w}}$, the $\boldsymbol{s}_m$s and the $\boldsymbol{\xi}_m$s are normally distributed random variables, then $\{f_m\}$ has a Gaussian process prior.
2. Alternatively, if the $\boldsymbol{s}_m$s are fixed parameters, and the entries in A are normally distributed random variables, then $\{f_m\}$ also has a Gaussian process prior.

Our present purpose is to understand multi-task learning with the LSM from the Gaussian process viewpoint directly over the latent functions. We will show that it is possible to capture the key notions of task relatedness in the LSM by just considering its second order statistics. To this end, it is not necessary for the distributions on the constituents of f_m given in equation 2.72 to be normally distributed.

Consider the first case where we fix A, place an i.i.d. prior (not necessarily normal) with zero mean and covariance Σ^{ξ} on the noise $\boldsymbol{\xi}_m$s, and also place a prior (also not necessarily normal) with zero mean and covariance Σ^{μ} on the common mean $\boldsymbol{\mu}_{\mathrm{w}}$. The distribution on $\{\boldsymbol{s}_m\}$, parameterized by Υ^{s}, will be specific later. Our interest is in the conditional distribution $p(\{f_m(\cdot)\}_{m=1}^M \mid \Theta)$ on the latent functions given the parameters $\Theta \overset{\mathrm{def}}{=} \{A, \Sigma^{\xi}, \Sigma^{\mu}, \Upsilon^{\mathrm{s}}\}$. Obtaining this distribution is intractable in general, thus we approximate it with a simpler $q(\{f_m(\cdot)\}_{m=1}^M \mid \Theta)$ so that the Kullback-Leibler divergence from

p to q is minimized. This is the assumed density filtering (ADF) approximation, and *if q is from the exponential family* then the minimization can be achieved by matching the expected sufficient statistics of q to that of p. If we restrict q to be Gaussian modelling only mean and covariance, we simply require

$$\langle f_m(\boldsymbol{x}) \rangle = \langle \boldsymbol{s}_m \rangle^{\mathrm{T}} A^{\mathrm{T}} \boldsymbol{x} \tag{2.73a}$$

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x'})\right) = \boldsymbol{x}^{\mathrm{T}} \Sigma^{\mu} \boldsymbol{x}' + \delta_{mm'} \boldsymbol{x}^{\mathrm{T}} \Sigma^{\xi} \boldsymbol{x}' + \boldsymbol{x}^{\mathrm{T}} A \big(\langle \boldsymbol{s}_m \boldsymbol{s}_{m'}^{\mathrm{T}} \rangle - \langle \boldsymbol{s}_m \rangle \langle \boldsymbol{s}_{m'}^{\mathrm{T}} \rangle \big) A^{\mathrm{T}} \boldsymbol{x}' \tag{2.73b}$$

where the expectations over $\boldsymbol{s}_m$ and $\boldsymbol{s}_m \boldsymbol{s}_{m'}^{\mathrm{T}}$ are with respect to the unspecified distribution on the $\{\boldsymbol{s}_m\}$.

Independence or correlation in the latent sources will induce the corresponding property in the covariance. As an example, we take the separable dependencies given by equation 2.17 on page 16, where $(\boldsymbol{s}_1^{\mathrm{T}}, \ldots, \boldsymbol{s}_M^{\mathrm{T}})^{\mathrm{T}}$ is normally distributed with covariance $\Sigma^{\mathrm{f}} \otimes \Sigma^{\mathrm{x}}$. Denote the $(m, m')\mathrm{th}$ entry of Σ^{f} by $\Sigma^{\mathrm{f}}_{mm'}$. Then the distribution on the latent functions has covariance

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x'})\right) = \boldsymbol{x}^{\mathrm{T}} \Sigma^{\mu} \boldsymbol{x}' + \delta_{mm'} \, \boldsymbol{x}^{\mathrm{T}} \Sigma^{\xi} \boldsymbol{x}' + \Sigma^{\mathrm{f}}_{mm'} \, \boldsymbol{x}^{\mathrm{T}} A \Sigma^{\mathrm{x}} A^{\mathrm{T}} \boldsymbol{x}'. \tag{2.74}$$

As another example for comparison, we use the linear manifold modeling described in section 2.2.2, where the latent sources $\boldsymbol{s}_m$s are independent among the tasks. Suppose the $\boldsymbol{s}_m$s are identically distributed with covariance Σ^{s}. Then the distribution on the latent functions has covariance

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x'})\right) = \boldsymbol{x}^{\mathrm{T}} \Sigma^{\mu} \boldsymbol{x}' + \delta_{mm'} \boldsymbol{x}^{\mathrm{T}} \Sigma^{\xi} \boldsymbol{x}' + \delta_{mm'} \boldsymbol{x}^{\mathrm{T}} A \Sigma^{\mathrm{s}} A^{\mathrm{T}} \boldsymbol{x}'. \tag{2.75}$$

Both this and equation 2.74 are instances of the LMC introduced in section 2.5.2. Comparing the covariances given by (2.74) and (2.75), we see that they both have the same “mean” and “noise” component. The “mean” component with covariance $\boldsymbol{x}^{\mathrm{T}} \Sigma^{\mu} \boldsymbol{x}'$ is shared by all the functions, while the “noise” component with covariance $\boldsymbol{x}^{\mathrm{T}} \Sigma^{\xi} \boldsymbol{x}'$ is independent for the M tasks. The correlated latent sources model given by equation 2.74 has an additional component that gives covariance $\Sigma^{\mathrm{t}}_{mm'}$ between task m and task m'. In contrast, the additional component for the linear manifold model is independent between the tasks. We can also see that although the linear manifold is for the effective parameters, its eventual effect is to map $\boldsymbol{x}$ of dimension D onto a lower D^{s}-dimensional space, since A is D-by-D^{s} with $D^{\mathrm{s}} \leqslant D$.

The approximation of the LSM by Gaussian processes using equation 2.73 is not effective for the task-clustering that we have seen in section 2.2.3 because we now have a mixture model. However, we may condition upon the latent sources $\boldsymbol{s}_m$s instead of marginalizing them out. Moreover, we can place a distribution on A and marginalize out A. This is thus the second case mentioned at the beginning of this section, where the $\boldsymbol{s}_m$s are fixed parameters, and A is random. Although it may seem that we have arbitrarily chosen which variable to fix as parameters and which to marginalize out as nuisance random variables, we shall see that the choice serves its purpose of extracting key notions of task relatedness from the LSM.

Let $P = D^{\mathrm{s}}$, and let each column $\boldsymbol{a}^p \in \mathbb{R}^D$ of A, $p = 1 \ldots P$, be independently and identically drawn from a prior with covariance Σ^{a}. If we express $\boldsymbol{s}_m$ as a vector of indicator variables $(z_m^1, \ldots, z_m^P)$, then $A\boldsymbol{s}_m = \sum_{p=1}^{P} z_m^p a^p$, and the effective prior over the latent functions (given the $\boldsymbol{s}_m$s) has covariance

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x'}) \mid \boldsymbol{s}_m, \boldsymbol{s}_{m'}\right) = \boldsymbol{x}^{\mathrm{T}} \Sigma^{\mu} \boldsymbol{x}' + \delta_{mm'} \boldsymbol{x}^{\mathrm{T}} \Sigma^{\xi} \boldsymbol{x}' + \left(\sum_{p=1}^{P} z_m^p z_{m'}^p \right) \boldsymbol{x}^{\mathrm{T}} \Sigma^{\mathrm{a}} \boldsymbol{x}', \tag{2.76}$$

Again, here we see the shared mean component with covariance $\boldsymbol{x}^{\mathrm{T}}\Sigma^{\mu}\boldsymbol{x}'$ and the independent (among tasks) component with covariance $\boldsymbol{x}^{\mathrm{T}}\Sigma^{\xi}\boldsymbol{x}'$. The clustering property is given by the last covariance component, which says that if (and only if) task m and task m' are in the same cluster, the $p\mathrm{th}$ cluster say, then $z_m^p z_{m'}^p = 1$, and the two tasks additionally have the same covariance $\boldsymbol{x}^{\mathrm{T}}\Sigma^{\mathrm{a}}\boldsymbol{x}'$. In this way, the mean of cluster p is modelled by $\boldsymbol{x}^{\mathrm{T}}(\Sigma^{\mu}+\Sigma^{\mathrm{a}})\boldsymbol{x}'$, while the deviation of each task from its cluster's mean is modelled by $\boldsymbol{x}^{\mathrm{T}}\Sigma^{\xi}\boldsymbol{x}'$. Finally, note that $\sum_{p=1}^{P} z_m^p z_{m'}^p$ gives a block diagonal K^{f}; see equation 2.62a on page 31.

The above exercise of approximating LSM with Gaussian processes highlights that by just considering the second moment properties of the latent functions under the LSM, the effect of different priors on task-relatedness can be directly seen. In the next section, we shall use the ICM multi-task Gaussian process model to understand the MRNM.

2.5.6 Multi-task Regularization Networks and Multi-task Gaussian Processes

Bayesian learning with priors are closely connected to learning with regularization. In fact, learning regularization networks can be viewed as learning the *maximum a posteriori* (MAP) function under Gaussian process priors and likelihoods; see, for example, Evgeniou et al. [2000, section 7], Opper and Winther [2000] and Rasmussen and Williams [2006, section 6.2]. Therefore, the regularizer $\lambda\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}$ in (2.37) can be seen as a zero mean isotopic Gaussian prior on $\boldsymbol{u}$, so that a probabilistic view give the zero mean and covariance

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')\right) = \boldsymbol{x}^{\mathrm{T}}\Phi_m \left\langle \boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}\right\rangle \Phi_{m'}^{\mathrm{T}}\boldsymbol{x}' = \frac{1}{2\lambda}\boldsymbol{x}^{\mathrm{T}}\Phi_m\Phi_{m'}^{\mathrm{T}}\boldsymbol{x}'. \tag{2.77a}$$

Up to the scaling factor $1/2\lambda$, this recovers the linear multi-task kernel given by (2.35). For the separable multi-task kernel in section 2.3.1, we have

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')\right) = \frac{1}{2\lambda}K^{\mathrm{f}}_{m,m'}\boldsymbol{x}^{\mathrm{T}}(E^{\mathrm{x}})^{+}\boldsymbol{x}', \tag{2.77b}$$

which is the ICM given in section 2.5.1 up to the scaling factor. In fact, it is also possible to see the above directly from the regularizer $\lambda\underline{\boldsymbol{w}}^{\mathrm{T}}E\underline{\boldsymbol{w}}$ in (2.3) when $E = E^{\mathrm{f}} \otimes E^{\mathrm{x}}$ by considering the regularizer as defining a multivariate Gaussian distribution with zero mean and (perhaps degenerate) covariance $E^{+}/2\lambda$:

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')\right) = \boldsymbol{x}^{\mathrm{T}}\left\langle \boldsymbol{w}_m\boldsymbol{w}_{m'}^{\mathrm{T}}\right\rangle \boldsymbol{x}' = \frac{1}{2\lambda}K^{\mathrm{f}}_{m,m'}\boldsymbol{x}^{\mathrm{T}}(E^{\mathrm{x}})^{+}\boldsymbol{x}'. \tag{2.77c}$$

where $K^{\mathrm{f}}_{m,m'}$ is the $(m, m')\mathrm{th}$ entry of $(E^{\mathrm{f}})^{+}$.

Let us consider in greater detail the notion of task-clustering — other notions of task-relatedness may be treated similarly. Let z_m^p equals one if the $m\mathrm{th}$ task belongs to the p^{th} cluster and zero otherwise. Using the multi-task kernel given by equation 2.51, we have a multi-task GP prior with zero mean and covariance

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')\right) = \delta_{mm'}\frac{1}{2\lambda\lambda^q}\boldsymbol{x}^{\mathrm{T}}(E^{\mathrm{x}})^{-1}\boldsymbol{x}' + \left(\sum_{p=1}^{P} z_m^p z_{m'}^p\right)\frac{1}{2\lambda}\boldsymbol{x}^{\mathrm{T}}(E^{\mathrm{x}})^{-1}\boldsymbol{x}', \tag{2.78}$$

where q is the cluster for the $m\mathrm{th}$ function; the derivation is given in appendix A.2.1 on page 142. This, of course, is an instance of the linear model of coregionalization model introduced in section 2.5.2. The second term of the covariance function has the factor $\left(\sum_{p=1}^{P} z_m^p z_{m'}^p\right)$ that is also present in the second moment of the LSM for task clustering, i.e., the third term in equation 2.76 on page 37. This term will lead to a block diagonal K^{f}; see equation 2.62a on page 31. As discussed after equation 2.76, this term says that two functions from the same cluster share a Gaussian process component with covariance function $\boldsymbol{x}^{\mathrm{T}}(E^{\mathrm{x}})^{-1}\boldsymbol{x}'/2\lambda$. Moreover, there is an additive "noise" component given by the first term in equation 2.78 that allows each function to deviate independently from the mean of its cluster, say the $q\mathrm{th}$ cluster, through a Gaussian process with covariance function $\boldsymbol{x}^{\mathrm{T}}(E^{\mathrm{x}})^{-1}\boldsymbol{x}'/2\lambda\lambda^q$.

The above is an example of how multi-task regularization can be understood using Gaussian processes.

2.5.7 Gaussian Processes from Infinite Neural Networks

Gaussian processes were made known to the machine learning community through the evolution of neural networks to Bayesian neural networks [MacKay, 1991; Neal, 1996] and then to Gaussian processes in the limit of infinite number of hidden units [Neal, 1994]. Therefore, one natural way to understand the Bayesian multi-task neural network model (BMNNM) discussed in section 2.4 is to consider it in relation to the multi-task GP model. To make the discussion more concrete, we shall consider the case where the common transfer function of the hidden units is the error function, i.e.,

$$g(z) = \operatorname{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z \exp(-t^2)\mathrm{d}t. \tag{2.79}$$

Under this setting, if the fan-in weights $\boldsymbol{\theta}_i$s of the hidden units are i.i.d. $\mathcal{N}(\boldsymbol{0}, \Sigma^{\theta})$ then each hidden unit, say the $i\mathrm{th}$ unit, produce a response $\phi_i(\boldsymbol{x}) = g(\boldsymbol{\theta}_i^{\mathrm{T}}\boldsymbol{x})$ that is independent of the others and has zero mean and covariance

$$\begin{aligned} k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}') \overset{\mathrm{def}}{=} \left\langle \phi_i(\boldsymbol{x})\phi_i(\boldsymbol{x}') \right\rangle_{\boldsymbol{\theta}_i} &= \left\langle g(\boldsymbol{\theta}_i^{\mathrm{T}}\boldsymbol{x}) g(\boldsymbol{\theta}_i^{\mathrm{T}}\boldsymbol{x}') \right\rangle_{\boldsymbol{\theta}_i} \\ &= \frac{2}{\pi} \arcsin\left(\frac{2\boldsymbol{x}^{\mathrm{T}}\Sigma^{\theta}\boldsymbol{x}'}{\sqrt{1 + 2\boldsymbol{x}^{\mathrm{T}}\Sigma^{\theta}\boldsymbol{x}}\sqrt{1 + 2\boldsymbol{x}'^{\mathrm{T}}\Sigma^{\theta}\boldsymbol{x}'}} \right); \end{aligned} \tag{2.80}$$

a derivation for this covariance function can be found in Williams [1998]. Though we use this as an example, we keep in mind that we just require $g(z)$ to be a bounded function so that its second moment (under random $\boldsymbol{\theta}_i$s) is bounded.

As an example, consider the case where the fan-in weights $\boldsymbol{u}_m$ for the $m\mathrm{th}$ output of the neural network have the task dependent prior mean discussed in section 2.4.1, i.e., $\boldsymbol{u}_m \sim \mathcal{N}(\cdot \mid A\boldsymbol{t}_m, \Sigma^{\xi})$, where $\boldsymbol{t}_m$ is the task descriptor for the $m\mathrm{th}$ task. For simplicity, let these weights have an isotropic prior, setting $\Sigma^{\xi} = \sigma_{\xi}^2$. Given D^{Φ} hidden units, the functions given by $f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})$ have zero mean and covariance

$$\mathbb{C}(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')) = \delta_{mm'}(\sigma_{\xi}^2 D^{\Phi})\, k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}') + \boldsymbol{t}_m^{\mathrm{T}} A^{\mathrm{T}} A \boldsymbol{t}_{m'}\, k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}'). \tag{2.81}$$

The derivation is in appendix A.2.2. Suppose we wish to let the number of hidden units D^{Φ} approach infinity so that the central limit theorem may be applied to argue that each $f_m(\boldsymbol{x})$ is exactly a Gaussian process in distribution. To keep the covariance function bounded, we must set $\sigma_{\xi}^2 = (\sigma_{\xi}')^2/D^{\Phi}$ for some fixed positive constant σ_{ξ}'. Similarly, since $(A^{\mathrm{T}}A)_{ij} = \sum_{k=1}^{D^{\Phi}} a_{ki}a_{kh}$, the entries in A must scale inversely with D^{Φ}. Alternatively, since the interest is in the D^{t}-by-D^{t} matrix $A^{\mathrm{T}}A$, where D^{t} is the dimension of the task descriptors, we may fix $A^{\mathrm{T}}A = (A')^{\mathrm{T}}A'$ for some matrix A' that has a finite number of rows. With these settings, the functions have Gaussian process priors with zero mean and covariance

$$\mathbb{C}(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')) = \delta_{mm'}{\sigma'_\xi}^2\, k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}') + \boldsymbol{t}_m^{\mathrm{T}} A'^{\mathrm{T}} A' \boldsymbol{t}_{m'}\, k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}'). \tag{2.82}$$

We may write the above as $K^{\mathrm{f}}_{mm'} k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}')$, where $K^{\mathrm{f}}_{mm'}$ is the $(m, m')\mathrm{th}$ entry in

$$K^{\mathrm{f}} = \left({\sigma'_\xi}^2 I + (A'T)^{\mathrm{T}}(A'T)\right) \tag{2.83}$$

and $T \overset{\mathrm{def}}{=} (\boldsymbol{t}_1 \mid \ldots \mid \boldsymbol{t}_M)$. This gives the non-parametric effects model of Yu et al. [2009] discussed in section 2.5.3, and given by equation 2.66, except that here the task descriptors $\boldsymbol{t}_m$s are used.

As noted in our introduction to BMNNM in section 2.4, the neural network model is similar to the LSM except for an additional indirection via the hidden units $\boldsymbol{\phi}(\boldsymbol{x})$. Translated to Gaussian processes, this indirection is given by, say, the k_{NN} covariance function in (2.82). This covariance function is parameterized by Σ^{θ} and may be optimized to best explain the training data.

2.5.8 Input Domain Transformation

The notion of task relatedness that we have investigated until now relates the outputs or responses of the functions f_ms given the input locations $\boldsymbol{x}$s from the same space $\mathcal{X}$. However, there may be circumstances where it makes sense to relate the M functions when the input spaces are different. For Gaussian processes, which are determined up to second order statistics, this is achieved by suitable cross-covariances between the functions. This can be done by introducing a common latent process $g(\boldsymbol{x})$, typically a white noise process, and then letting each function f_m be the response of g under convolution with a filter $h_m(\boldsymbol{x}, \boldsymbol{x}')$ that may differ among the functions. We refer the reader to Ver Hoef and Barry [1998], Higdon [2002], and Boyle and Frean [2005] for this approach that introduces the auxiliary latent process g. Here, we present a *transformation* approach that appeals more directly to how tasks are related.

The idea comes from the inter-domain Gaussian process introduced by Lázaro-Gredilla and Figueiras-Vidal [2009] for sparse and efficient learning of Gaussian process models. Here, we wish to use it for multi-task learning. The approach is to model the transformation in the input space using convolution directly on the output function. Consider two functions $f_1 : \mathcal{X}_1 \mapsto \mathbb{R}$ and $f_2 : \mathcal{X}_2 \mapsto \mathbb{R}$ on possibly different domains $\mathcal{X}_1$ and $\mathcal{X}_2$. Given some deterministic function $g : \mathcal{X}_1 \times \mathcal{X}_2 \mapsto \mathbb{R}$, let f_2 be directly related to the f_1 via the transformation

$$f_2(\boldsymbol{z}) = \int_{\mathcal{X}_1} f_1(\boldsymbol{x}) g(\boldsymbol{x}, \boldsymbol{z}) \mathrm{d}\boldsymbol{x}, \qquad \text{where } \boldsymbol{z} \in \mathcal{X}_2. \tag{2.84}$$

Since the transformation is linear, a Gaussian process prior on f_1 will also lead to a Gaussian process prior on f_2. Let the Gaussian process prior on f_1 be such that

$$\langle f_1(\boldsymbol{x}) \rangle = \mu(\boldsymbol{x}) \qquad \mathbb{C}(f_1(\boldsymbol{x}), f_1(\boldsymbol{x}')) = k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}'). \tag{2.85}$$

Then

$$\langle f_2(\boldsymbol{z})\rangle = \int_{\mathcal{X}_1} \mu(\boldsymbol{x}) g(\boldsymbol{x},\boldsymbol{z})\mathrm{d}\boldsymbol{x} \tag{2.86a}$$

$$\mathbb{C}(f_1(\boldsymbol{x}), f_2(\boldsymbol{z})) = \int_{\mathcal{X}_1} k(\boldsymbol{x},\boldsymbol{x}') g(\boldsymbol{x}',\boldsymbol{z})\mathrm{d}\boldsymbol{x}' \tag{2.86b}$$

$$\mathbb{C}(f_2(\boldsymbol{z}), f_2(\boldsymbol{z}')) = \int_{\mathcal{X}_1}\int_{\mathcal{X}_1} k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x}') g(\boldsymbol{x},\boldsymbol{z}) g(\boldsymbol{x}',\boldsymbol{z}')\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}' \tag{2.86c}$$

$$= \int_{\mathcal{X}_1} \mathbb{C}(f_1(\boldsymbol{x}), f_2(\boldsymbol{z}')) g(\boldsymbol{x},\boldsymbol{z})\mathrm{d}\boldsymbol{x}. \tag{2.86d}$$

Thus, by directly specifying how $\boldsymbol{z} \in \mathcal{X}_2$ is related to $\boldsymbol{x} \in \mathcal{X}_1$ through g, we are able to specify fully the joint Gaussian process involving both f_1 and f_2. In addition, this transformation approach allows $\mathcal{X}_2 \neq \mathcal{X}_1$, which may be important in some applications. The disadvantage of this approach is that the covariance function for f_2 cannot be directly specified in general; instead it is fixed entirely by the covariance function k^{x} for f_1 and the transformation g between the two domains.

In certain cases, we will be able to derive analytical cross-covariance functions between f_1 and f_2. As a worked example, let us set $\mathcal{X}_1 \equiv \mathcal{X}_2 \equiv \mathbb{R}^D$,

$$g(\boldsymbol{x},\boldsymbol{z}) \stackrel{\mathrm{def}}{=} \frac{1}{(2\pi)^{D/2}\sqrt{|C_{\mathrm{g}}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{z}-\boldsymbol{\mu})^{\mathrm{T}} C_{\mathrm{g}}^{-1}(\boldsymbol{x}-\boldsymbol{z}-\boldsymbol{\mu})\right), \tag{2.87a}$$

and also set the GP prior on f_1 to have zero mean and covariance function

$$k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x}') = \frac{v_1^2}{\sqrt{|C|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}')^{\mathrm{T}} C^{-1}(\boldsymbol{x}-\boldsymbol{x}')\right) \tag{2.87b}$$

for symmetric positive definite matrices C and C_{g}. Then we have

$$\langle f_2(\boldsymbol{z})\rangle = 0 \tag{2.88a}$$

$$\mathbb{C}(f_1(\boldsymbol{x}), f_2(\boldsymbol{z})) = \frac{v_1^2}{\sqrt{|C+C_{\mathrm{g}}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{z}-\boldsymbol{\mu})^{\mathrm{T}}(C+C_{\mathrm{g}})^{-1}(\boldsymbol{x}-\boldsymbol{z}-\boldsymbol{\mu})\right) \tag{2.88b}$$

$$\mathbb{C}(f_2(\boldsymbol{z}), f_2(\boldsymbol{z}')) = \frac{v_1^2}{\sqrt{|C+2C_{\mathrm{g}}|}} \exp\left(-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{z}')^{\mathrm{T}}(C+2C_{\mathrm{g}})^{-1}(\boldsymbol{z}-\boldsymbol{z}')\right). \tag{2.88c}$$

The translation $\boldsymbol{\mu}$ is present in the cross-covariance $\mathbb{C}(f_1(\boldsymbol{x}), f_2(\boldsymbol{z}))$, but it is absent in the covariance $\mathbb{C}(f_2(\boldsymbol{z}), f_2(\boldsymbol{z}'))$. This means that translation is only required when the inference is across the functions f_1 and f_2, but not within the functions.

The important element in learning across domains (or transformations within the same domain) is the transformation function $g(\boldsymbol{x},\boldsymbol{z})$ that relates a point $\boldsymbol{x} \in \mathcal{X}_1$ to another point $\boldsymbol{z} \in \mathcal{X}_2$. Lázaro-Gredilla and Figueiras-Vidal [2009] discuss particular choices that allow learning from a different scale, and learning from the frequency domain. An important area for future research is to broaden the space for $g(\boldsymbol{x},\boldsymbol{z})$ and to allow the learning of this transformation function from data.

2.6 Interlude

Until this point, we have looked at four different models for multi-task learning and seen how they may be used for implementing common notions of task relatedness in multi-task learning, such as task-clustering and manifold of related predictors. We have also proposed and used Gaussian processes as a vehicle for understanding task relatedness directly via correlations between the tasks. There are certainly other models for multi-task learning in addition to the four we have reviewed in this chapter. In particular, we have omitted the pioneering works that are mostly based on artificial neural networks. We remedy this by referring the reader to excellent reviews by Gordon and desJardins [1995], Thrun and Pratt [1998, chap 1], Silver [2000, chap 2], Vilalta and Drissi [2002], Giraud-Carrier et al. [2004a], and Anderson and Oates [2007].

Multi-task learning is part of the greater effort in meta-learning to allow machine learning itself to be flexible. In section 2.7 we will briefly survey the current approaches of meta-learning to automatically select the best algorithms from an array of available ones. In addition, multi-task learning is called by other names in the statistics and applied-statistics communities, and we will survey their approaches also in section 2.7. The important issue of consolidating knowledge from multiple tasks for subsequent use in learning to learn is also addressed in section 2.7

Lastly, in section 2.8 we discuss some areas of machine learning research that have frequent cross-fertilization of ideas to research in multi-task learning.

2.7 Meta-learning from Other Perspectives

In section 2.7.1 we will review how algorithms and data sets may be characterized so as to alleviate the burden of manually selecting the best algorithm or machine learning model for a given problem. We will survey how multi-task learning is modelled in the statistics and applied-statistics communities in sections 2.7.2 and 2.7.3. In section 2.7.2 we will also survey some recent work in the very related areas of multi-class classification and multi-label prediction in machine learning.

In learning to learn, one of the aims is to consolidate experiences in order to use them efficiently and effectively when encountering novel tasks. One way to consolidate knowledge is to learn the intrinsic knowledge representations inherent to the tasks. The difference between this and learning predictor-manifolds is that the former usually encodes semantic meaning while the latter deals with "merely parameters". A good knowledge representation will allow human experts to peer into the system to evaluate it based on human understanding instead of simply relying on predictive performance. In section 2.7.4 we will give a brief survey of research in this area.

2.7.1 Data Sets and Algorithms Characterizations

When encountered with a predictive task and data set, one may have to choose an algorithm from an array of possible learning algorithms, such as decision trees, linear classifiers (or their kernelized versions), nearest neighbour classifiers and multilayer artificial neural networks. Research in alleviating the system engineering and knowledge engineering burden on humans are the motivation of two European projects: the StatLog project [Michie et al., 1994] and the MetaL project [e.g. Bensusan and Giraud-Carrier, 2000; Pfahringer et al., 2000]. A recent empirical comparison by Caruana and Niculescu-Mizil [2006] also brings out the importance of choosing the suitable algorithm(s) for a given task: "Although some methods clearly perform better or worse than other methods *on average*, there is significant variability across problems and metrics. Even the best models sometimes perform poorly, and models with poor average performance occasionally perform exceptionally well." Hence, the goal of meta-learning is *not* to find the best method on average; rather, it is to select the best method for a *given* problem.

In general, there are two steps in the selection of an algorithm. The first and more crucial step is to define and compute the morphological characteristics of the data sets and the learning algorithms, and we will look at examples of these later in the section. This characterization defines the space on which relatedness between tasks is defined.

The second step is to use these characteristics to predict the suitability of an algorithm for a given task and data set. These predictions can be used to select a single best algorithm. Alternatively, they can be used for selecting or ranking a group of algorithms for further exploration. The suitability criteria of an algorithm may include the performance (e.g. accuracy) of the algorithm, the time required for training and testing, and the interpretability of the model. Most work focuses on the performance of the algorithm.

2.7.1.1 Characterising data sets

The StatLog project [Michie et al., 1994] compares about 20 learning algorithms on 20 classification tasks, and uses C4.5 to relate the performance of the algorithms to data characteristics such as simple measures (e.g., the number of attributes), statistical measures and information-theoretic measures. An extensive list of possible data characteristics is given by Lindner and Studer [1999]. Due to the fact that some measures are non-applicable in certain data sets (e.g., mean for categorical variables), care must be taken when using these. The data characteristics must not only be correlated with performance of an algorithm, they must also be computationally cheap to calculate; otherwise, one may simply run all the algorithms and use performance on a validation set for selecting the "best" algorithm. Castiello et al. [2005] give some considerations in choosing such data characteristics so that they reflect the actual task instead of the particular sample of data set, and, to this end, they also propose certain transformations of data characteristics.

An alternative approach is to use the learning complexity on a data set for characterization [Bensusan, 1999; Peng et al., 2002]. This, for a given data set, can be approximated with the structure and size of a decision tree — called the *baseline learner* in this context — trained on that data set. Selected properties of the decision tree, such as its maximum depth, are then used in predicting the performance of various other learning algorithms on the data set. Instead of using just selected properties of the tree, Bensusan et al. [2000] further propose using the entire tree structure as input to the meta-learning module.

Instead of using just one baseline learner, the methodology of *landmarking* [Bensusan and Giraud-Carrier, 2000] uses a pool of simple and computationally efficient learners, called *landmarkers* in this context. Each landmarker uncovers a specific nature of the data; for example, the linear discriminant is used measure the linear separability of the data via its error rate on the data.

Though not directly relevant to algorithm selection, it is worth mentioning the approach of Bonilla et al. [2007] to characterizing tasks. Theirs is a multi-task learning approach to a set of related predictive tasks for which obtaining actual responses is expensive but possible. First, they sample a fixed set of input locations. Next, the set of actual responses for each task on these locations is computed and then used to characterize or describe the task, i.e., it is the task descriptor for that task. The task descriptors, together with a learnt metric, then approximate task relatedness.

2.7.1.2 Characterising learning algorithms

Just as data sets can be characterized by one or many baseline learners, learning algorithms can be characterized by a selection of data sets. Kalousis et al. [2004] have explored this idea by clustering classification algorithms into groups that have similar error patterns on the data sets.

2.7.2 Vector-valued Functions

Vector-valued functions are those with ranges that are vector fields instead of the usual scalar fields. Learning a vector-valued function is also known as *multiple response/output regression* or *multivariate regression*. If we treat prediction in each scalar dimension of the vector-valued function as a task, then we have M tasks for an M-dimensional vector field. Learning these tasks jointly can then be viewed as an example of multi-task learning.

The technique of *partial least squares* (PLS, see Rosipal and Krämer [2006] for a recent review), which is widely used in the field of chemometrics, has been applied to multivariate regression problems. Here, the covariance structure between the input variables and output responses is jointly extracted, and there is transfer between the different output dimensions due to this joint extraction. An alternative approach is the “curds and whey” procedure proposed by Breiman and Friedman [1997], though it has been pointed out by many that this is inferior to PLS (see separate discussions by Wold, Bur et al., and Helland in [Breiman and Friedman, 1997]). Other methods in statistics also include *joint continuum regression* [Brooks and Stone, 1994] and *multivariate ridge regression* (MRR) [Brown and Zidek, 1980; Haitovsky, 1987]. A statistical framework, called the *latent variable multivariate regression model*, has been proposed by Burnham et al. [1999] in an attempt to unify these methods and some others. These models can be viewed as multi-task learning for the specific case where the set of responses are completely given for all the inputs (or covariates); as pointed out below, the technical term for this in the geostatistics community is *isotopic* data. In the general case for multi-task learning, the complete set of responses may not be given or may not be sensible, because the domain of the tasks may be different.

In econometrics the *seemingly unrelated regression* (SUR) model introduced by Zellner [1962] has been used to analyze a system of multiple equations with cross-equation parameter restrictions and correlated error terms.[7] In our context, each equation can be seen as a task, and the simultaneous learning of the parameters facilitates the transfer of information between the tasks. A key element that influence the transfer is the covariance between the error terms, and this is usually estimated by first solving each equation separately, and then taking the covariance between the residuals. In machine learning, the errors (or noise) are usually taken to be uncorrelated; therefore, the SUR model may not be so applicable

[7] MRR is a special case of the SUR model [Haitovsky, 1987].

in this case. Nevertheless, a correlated error model using Gaussian processes has been proposed by Silva et al. [2008] for modelling relational data.

Cokriging is used in multivariate geostatistics [Wackernagel, 1998] when there are multiple spatial variables at any one location, and we would like predictions to use information from other variables in addition to information at other locations (note that this is also a regression procedure; see Stein and Corsten [1991]). In this subject area, data may come in one of the following three forms [Wackernagel, 1998]:

entirely heterotopic, where the variables have no sample locations in common;

partially heterotopic, where the variables share some sample locations; and

isotopic, where, for all the sampling locations, data for each variable is available.

A common and general model for cokriging is the *linear model of coregionalization* (LMC) that we have seen in section 2.5. The estimation of the parameters in LMC has been somewhat *ad-hoc* until the recent work of Zhang [2007] and Zhu et al. [2005]. In our work, we have found maximizing the marginal likelihood by gradient methods to be effective in estimating the parameters; see section 4.3.3 and appendix C.3. If the prediction of each variable is seen as a task, then the LMC can be used as a general model for multi-task learning, especially when applied to heterotopic data. In geostatistics, the LMC is used for *factorial kriging analysis*, where the interest is in extracting factors of different spatial scale. The LMC has also been used to approximate a general covariance matrix K by minimizing the Frobenius norm between K and $\sum_p K^{\mathrm{f}p} \otimes K^{\mathrm{x}p}$ [Genton, 2007].

Closely related is the work of Micchelli and Pontil [2005] defining *reproducing kernel Hilbert spaces* of vector-valued functions in the field of learning theory. This is later applied to the learning of multiple tasks by Evgeniou et al. [2005], which we have described in section 2.3.

A related area that is frequently explored in machine learning is *multi-class classification*, and simultaneous learning of a model for the classes is a kind of transfer between the classes. Although the relation to multi-task learning is clear when we consider each of the single-class classifications as a task, there are distinctions between multi-class classification and general multi-task learning — the former is heavily constrained to exactly one class for any given input, and its input set is shared between all the classes.

For multi-class classification, transfer across classes can be achieved by extracting within-class structures that are shared between the classes. Amit et al. [2007] motivate trace-norm regularization of the weight vector of a linear classifier by considering a shared linear transformation of the input space for the classes. Torralba et al. [2004] use a joint boosting algorithm on weak-learners, some of which are shared between the classes; transfer is via the shared weak-learners. Similarly, Fink et al. [2006] introduce a mixing framework that combines perceptrons, each of which predicts membership in a given subset of the classes; transfer between classes is achieved by perceptrons that predict for non-singleton subsets.

Related to multi-class classification is *multi-label prediction*, where now each object may be labelled with more than one label; see Tsoumakas and Katakis [2007] for an overview. For a given object, we may expect that it having one label is related to it having another label. For instance, for labelling a scene image, the labels *beach* and *sunset* may be positively correlated, while the labels *beach* and *mountains* may be negatively correlated. A recent reduction of multi-label prediction with a multitude of labels to binary regression problems using the notion of compressed sensing [Hsu et al., 2009] makes multi-label prediction more relevant to multi-task learning: by compressing the high dimensional sparse vector of labels $\boldsymbol{l} \in \mathbb{R}^d$ to a vector $\boldsymbol{y} \in \mathbb{R}^c$ of a much lower dimension using a linear map $\boldsymbol{y} = A\boldsymbol{l}$, where $c \ll d$ and A is a c-by-d matrix, one may expect correlations to exist between the different dimensions of $\boldsymbol{y}$ due to its construction, and this makes multi-task learning for $\boldsymbol{y}$ attractive.

2.7.3 Multilevel Models

Multilevel models [Goldstein, 2003] are commonly used in statistics for analyzing social data which are hierarchical or clustered due to the natural groupings of social processes. For example, students are grouped within schools which may be further grouped into private or public. Multilevel models use common and group-specific factors to respectively account for the homogeneity and heterogeneity of the data. Such models are also known by various other terms such as Bayesian hierarchical models, mixed models, hierarchical linear models, random effects models, random coefficient models, subject specific models, variance component models, variance heterogeneity[8] and seemingly unrelated regression. The most common methods are linear models, though nonlinear extensions have also been proposed [Goldstein, 1991; Müller et al., 2007]. The framework by Zhang [2006], or indeed the general formulation given in equation 2.4 on page 13, is also a multilevel model.

Commonly in the multilevel models literature, a level 1 unit is an individual sample, a level 2 unit is a group of samples, and a level 3 unit is the entire set of samples;[9] for example, a level 1 unit would be a student and a level 2, a school. The direct correspondence to machine learning is this: a level 2 unit is a prediction task, while a level 3 unit is a set of tasks over which multi-task learning is performed. Hence, multilevel models can be directly applied to multi-task learning for modelling differences and shared similarities among the tasks. In fact, many of the recent work in multi-task learning, such as that of Zhang [2006], are along these lines, but extend the approaches to deal with non-linearity.

Finally, multilevel models can also be used as predictors for vector-valued functions (see section 2.7.2) when we view each dimension as a group.

2.7.4 Learning Knowledge Representation

Knowledge representations, instead of being pre-encoded, may be acquired through learning, and then used in providing a strong inductive bias for solving new tasks [Kemp et al., 2004]. Stracuzzi [2006] have argued that having the acquired knowledge organized into discrete concepts will allow more tractable transfer of knowledge onto new tasks. Recently, Niculescu-Mizil and Caruana [2007] give a greedy algorithm to search for a common Bayesian network structure between related tasks.

For a given representation, there will be interactions between its layers. If learning each layer can be naturally defined as a sub-task, then one has the choice to learn and/or apply these sub-tasks jointly,

[8] Taken from Centre for Multilevel Modelling website `http://www.cmm.bristol.ac.uk/` in Dec 2009.

[9] More levels are possible of course, but this is the most common setup in the literature.

hence transferring information from layer to layer. This has been investigated by Sutton and McCallum [2005] in the context of linear-chain *conditional random fields*.

A common kind of knowledge representation is the state-space of a Markov Decision Processes (MDPs). Walsh et al. [2006] and Wolfe and Barto [2006] have proposed using homomorphisms between MDPs to facilitate transfer between similar learning environments.

2.8 Related Research Areas

We now describe some areas of active research in machine learning that, although are not models for multi-task learning or meta-learning, are nevertheless closely intertwined with meta-learning. Although we highlight differences between each area and multi-task learning, there exists a common theme among them and multi-task learning — the existence of some form of information transfer beyond those between individual samples. Hence ideas from these areas may serve as inspirations for new multi-task learning models. The following areas are neither mutually exclusive nor altogether exhaustive.

2.8.1 Semi-supervised Learning

Semi-supervised learning aims to augment the use of scarce amount of labelled data with the use of more abundant unlabelled data in order to increase the performance of machine learning algorithms. Although one may say that there is some notion of transfer of information from the unlabelled data, there are some differences with learning from multiple tasks. In learning from multiple tasks, the tasks (and sometimes their data) are different, although they may be from the same general domain. For semi-supervised learning, there is only one task, and the augmenting data is actually for the same task but without any labels. See Zhu and Goldberg [2009] for an introduction, Zhu [2005] for a survey, and Chawla and Karakoulas [2005] for an empirical study.

A particular semi-supervised learning strategy that makes use of more than one classifier is *co-training* [Blum and Mitchell, 1998] or versions and generalisations of it [Zhou and Goldman, 2004; Zhou and Li, 2005]. In co-training, two classifiers are first trained on different views on a common set of labelled training data. Each classifier is then "taught" by the other classifier through the labels assigned by the other on unlabelled data. Here, there is some transfer of information from one classifier to the other via the labelling of unlabelled data. However, unlike learning from multiple tasks, the classifiers here are for the same task.

The strategy of Ando and Zhang [2004, 2005] on semi-supervised learning makes use of learning to learn. Multiple auxiliary prediction tasks are *created* from unlabelled data, and the underlying structure of the domain is extracted via learning these tasks jointly using the SVD-ASO algorithm described in section 2.2.2. This structure is then used for learning the original task *à la* learning-to-learn, treating the original task as the novel task. However, unlike normal learning to learn where the multiple tasks arise *naturally* from the environment, their auxiliary tasks are *artificial* and may not have any direct practical use.

2.8.2 Domain Adaptation

Domain adaptation[10] is aimed at supervised machine learning of tasks when the labelled training data of the relevant domains is scarce (or non-existent) but there are plentiful labelled data in related domains. In order to increase the performance of the classifiers on the relevant domain, a domain adaptation strategy tries to make use of data from the related domains. Domain adaptation differs from multi-task learning in that, for domain adaptation, there is *a single task defined on two (or more) domains* [Ben-David et al., 2007] and the aim is to optimize the performance on the target domain. It is instructive to consider the linear regression case given by Daumé III [2009]: for domain adaptation, one may have a single vector of regression parameters shared among the different domains, and optimize this vector for the single task across the different domains; in contrast, for multi-task learning one usually assumes each task has its own vector of parameters, but the within-vector co-variations are common among the tasks. However, despite the subtle differences, research on how to transfer information between domains may serve as an inspiration for that between tasks. In addition, similar to multi-task learning, there can also be a notion of relatedness between domains [Daumé III and Marcu, 2006].

Research in this area is important for natural language processing where one would like to make use of annotated (or labelled) texts from an "out-of-domain" genre when learning an "in-domain" genre, as investigated by Chan and Ng [2005], Blitzer et al. [2006] and Daumé III and Marcu [2006]. Chan and Ng transfer knowledge between the two domains in the naïve Bayes formulation by assuming that the domains share common within-class probabilities. Blitzer et al. use the structural learning framework of Ando and Zhang [2005], described in section 2.2.2, to find a common representation for features in both domains; this representation is then used to transfer knowledge between the two domains. Daumé III and Marcu view training data from both domains as sharing a common mixture component in a hierarchical Bayes model; in this case, the transfer of knowledge occurs through joint learning of the parameters for the common component.

Raina et al. [2007] formalize a similar but related problem where the data from the related domains are unlabelled instead of labelled and call this *self-taught learning*. In general, the case where the training and test distributions differ is called the *sample selection bias* in econometrics [Heckman, 1979]. In fact, domain adaptation, self-taught learning, sample selection bias and multi-task learning may be broadly called *transfer learning*, and a recent survey is given by Pan and Yang [to appear].

2.8.3 Composite Learners

To increase predictive performance, simple learners, such as linear classifiers, can be combined in various ways using, e.g., *stacked generalization* [Wolpert, 1992], *bagging* [Breiman, 1996], *boosting* [Freund and Schapire, 1996] or *arbitration* [Ortega et al., 2001]. For example, stacked generalization consists of layers of learners, each applied to meta-data derived from the learner of the layer directly below.

Vilalta and Drissi [2001] consider stacked generalization to be a meta-learning algorithm; Anderson and Oates [2007] consider stacked generalization and boosting; Giraud-Carrier et al. [2004a] consider

[10] In this context, a *domain* is a distribution on the data, which differs when different portions of the population are sampled.

stacked generalization and arbitration. However, we note that these algorithms are designed for learning a single task and not multiple tasks.

2.8.4 Multi-resolution Features

Multi-resolution methods, recently popular in image processing, extract features at different resolutions, scales or quantization levels of the data; see, for example, Adelson et al. [1991], Hadjidemetriou et al. [2004], Grauman and Darrell [2006] and Kivinen et al. [2007]. Although there may be transfer of information between the predictors for the different resolutions in the ensemble,[11] this is only for single tasks. However, transfer of information between tasks at the various resolution levels has been proposed in [Eaton, 2006], [Eaton and desJardins, 2006] and [Eaton et al., 2007].

[11] The *pyramid match kernels* of Grauman and Darrell [2006] can be viewed as inducing a set of classifiers at different levels of the pyramid.

Chapter 3

Generalization Errors and Learning Curves for Regression with Multi-task Gaussian Processes, and Their applications

3.1 Introduction

In the previous chapter, we have reviewed a number of models for multi-task learning. In the design of each model, it is generally assumed that learning multiple tasks together is beneficial. As discussed in section 1.2, the common utilitarian understanding of task relatedness is closely connected to the amount of benefit that can be reaped under multi-task learning. However, other than *p*robably *a*pproximately *c*orrect (PAC) theoretical analysis [Baxter, 2000; Maurer, 2006; Ben-David and Schuller Borbely, 2008], we are not aware of any work that quantifies such benefits.[1] Following the tradition of the theoretical works on Gaussian processes in machine learning, our goal in this chapter is to quantify the benefits using average-case analysis.

Our analysis is conducted on the multi-task Gaussian process (GP) model since, in contrast to other models, not only does it encode model assumptions for regression in a transparent way [Sollich and Halees, 2002], it is also amenable to the average-case analysis that we are after. We concentrate mainly on the asymmetric two-tasks case, where the secondary task is to help the learning of the primary task, although we will also give results for the symmetric two-tasks case, where the two tasks are to help each other to learn. Within this setting, the main parameters are (1) the degree of "relatedness" ρ between the two tasks, and (2) π_S, the fraction of the total training observations from the secondary task. While higher $|\rho|$ and lower π_S is clearly more beneficial to the primary task in the asymmetric learning case,

[1] Although Baxter [1997]'s analysis is on the Bayesian hierarchical model, he deals with the amount of information learnt for the prior rather than the predictive errors of the learnt model, which relates more directly to "the amount of benefit reaped".

the extent and manner that this is so has not been clear. To address this, we measure the benefits using generalization error, the learning curve and optimal error, and investigate the influence of ρ and π_S on these quantities.

We will give non-trivial lower and upper bounds on the generalization error and the learning curve. Both types of bounds are important in providing assurance on the quality of predictions: an upper bound provides an estimate of the amount of training data needed to attain a minimum performance level[2], while a lower bound provides an understanding of the limitations of the model [Williams and Vivarelli, 2000]. For a one-dimensional input-space under optimal sampling with data only for the secondary task, we show the limit to which error for the primary task can be reduced. This dispels any misconception that abundant data for the secondary task can remedy having no data for the primary task.

Our main approach relates the posterior variances, generalization errors and learning curves of multi-task GPs to that of single-task GPs. This is preferable to calculating the learning curves for multi-task GPs from scratch. First, known bounds and approximations for the case of single-task GPs can be utilized. Secondly, the relations between multi-task GPs and single-task GPs are made explicit; this allows for intuitive understandings of multi-task GPs. This second point is exploited by applying the learning curves to understand (1) the effects of multi-task learning over single-task learning in terms of increase and decrease in the average error; and (2) the effective number of observations for the primary task contributed by multi-task learning with a secondary task.

We outline the background concepts and state the problem precisely in the next section. Section 3.3 considers generalization error, and highlights the limitations of multi-task GP learning. In addition, lower and upper bounds on the generalization error are developed. In section 3.4, we give the optimal generalization error for the degenerate case when training data is provided only for the secondary task. Section 3.5 gives theoretical bounds on the learning curves, and provides simulation results for the learning curve and its bounds. Sections 3.6 and 3.7 use the theoretical bounds on the learning curves to provide further insights into multi-task learning, in terms of how the learning curve changes from the single-task setting, and the effective number of additional observations contributed by multi-task learning. Selected theoretical results are further evaluated in section 3.8 using an empirical data set. In section 3.9, we aim to understand the structure of multi-task GP prediction for more than one secondary tasks by focusing on noise-free observations on locations arranged in specific configurations. We conclude this chapter in section 3.10. Figure 3.1 diagrammatically lays out selected symbols of interest in this chapter.

3.1.1 Main Results

Before we delve into the details, we give the main results of this chapter.

1. If there is no data for the primary task, the generalization error for the primary task *cannot* be less than $(1-\rho^2)\sum\kappa_i$, where $\sum\kappa_i$ may be understood as the generalization error before the secondary data is given. See Propositions 3.3 and 3.11.

2. A rather tight lower bound to the generalization error for the primary task can be achieved by first providing additional observations for the primary task and then giving randomized predictions by

[2] This is related to the sample complexity in PAC models. See section 3.7.

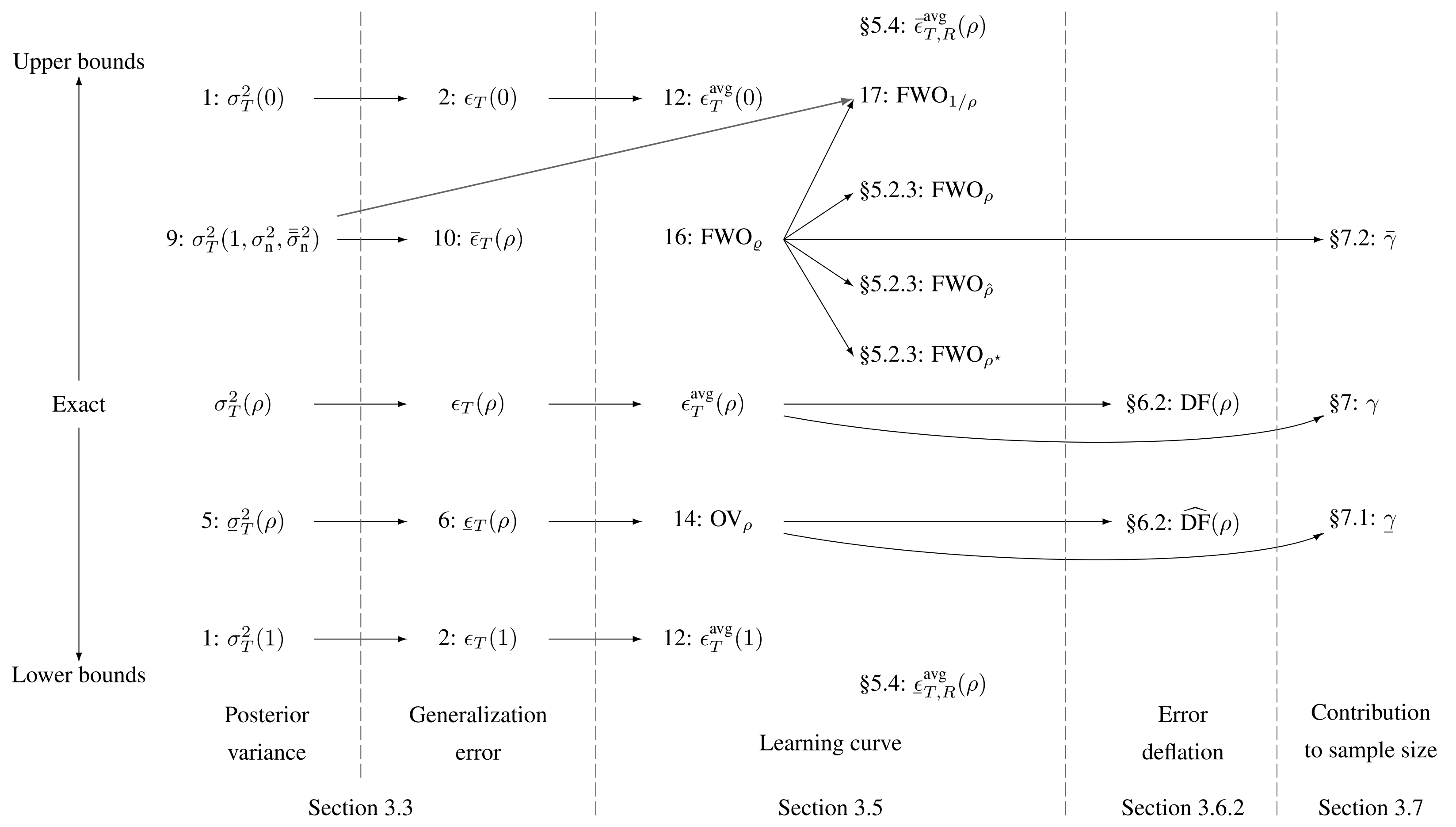

Upper bounds
Exact
Lower bounds
1: $\sigma_T^2(0)$
2: $\epsilon_T(0)$
12: $\epsilon_T^{\text{avg}}(0)$
§5.4: $\bar{\epsilon}_{T,R}^{\text{avg}}(\rho)$
17: $\text{FWO}_{1/\rho}$
9: $\sigma_T^2(1, \sigma_{\text{n}}^2, \bar{\bar{\sigma}}_{\text{n}}^2)$
10: $\bar{\epsilon}_T(\rho)$
16: FWO_ϱ
§5.2.3: FWO_ρ
§5.2.3: $\text{FWO}_{\hat{\rho}}$
§5.2.3: $\text{FWO}_{\rho^\star}$
§7.2: $\bar{\gamma}$
$\sigma_T^2(\rho)$
$\epsilon_T(\rho)$
$\epsilon_T^{\text{avg}}(\rho)$
§6.2: $\text{DF}(\rho)$
§7: γ
5: $\underline{\sigma}_T^2(\rho)$
6: $\underline{\epsilon}_T(\rho)$
14: OV_ρ
§6.2: $\widehat{\text{DF}}(\rho)$
§7.1: $\underline{\gamma}$
1: $\sigma_T^2(1)$
2: $\epsilon_T(1)$
12: $\epsilon_T^{\text{avg}}(1)$
§5.4: $\underline{\epsilon}_{T,R}^{\text{avg}}(\rho)$
Posterior variance
Generalization error
Learning curve
Error deflation
Contribution to sample size
Section 3.3
Section 3.5
Section 3.6.2
Section 3.7

Figure 3.1: Relations between selected symbols of interest in this chapter. Those placed horizontally along the center line are the exact ones, while those above and below are the corresponding upper and lower bounds (though $\underline{\gamma}$ and $\bar{\gamma}$ do not bound γ; and $\widehat{\text{DF}}$ does not bound DF). We place from left the right the symbols relating to posterior variance, generalization error, learning curve, error deflation factor and contribution of the secondary task to the effective sample size of the primary task. Each symbol is given with the proposition/corollary number or the section number (if the number is prefixed with '§') that introduces it. The chapter number is omitted to minimize clutter.

discarding this extra information ρ^2 of the time. For the Sarcos data, we show that this approach can provide an approximate lower bound on the empirical error of the multi-task Gaussian process. See Propositions 3.7 and section 3.8.2.

3. In terms of the lower bound on the learning curve for the primary task, multi-task learning can decrease the error by a factor of up to $\rho^2\pi_S$. See discussion of Proposition 3.14, and see sections 3.6.2 and 3.8.4.

4. For symmetric multi-task learning, where the interest is in the error averaged across the two tasks, the governing factor for performance is $\omega \overset{\text{def}}{=} \pi_S(1-\pi_S)(1-\rho^2)$. We may say that symmetric multi-task learning can be characterised by ω. Section 3.6.1 verifies this by providing a comparison between theoretical and simulated learning curves.

The above list of results are based on the lower bound on the generalization error and the learning curve for multi-task learning. Admittedly, the lower bound is easier to understand and perhaps more useful than the upper bound because of its simplicity. Nevertheless, the upper bound is still useful, not only in providing additional insights to multi-task Gaussian process learning, but also in the following ways.

1. For small sample sizes, the theoretical upper bound on the learning curve (called the FWO$_{\hat{\varrho}}$ bound) is much tighter than the theoretical lower bound (called the OV$_\rho$ bound). In fact, also for small sample sizes, the FWO$_{\hat{\varrho}}$ theoretical upper bound can be tighter than an empirical trivial upper bound that ignores the data from the secondary task. See section 3.5.5.

2. Using the FWO$_{\hat{\varrho}}$ bound, we obtain the secondary task's contribution to the average-case sample complexity of the primary task. See section 3.7.2.

Although theoretical bounds on the learning curve can be rather loose, they are useful if they vary *meaningfully* with quantities of interest. Indeed, this is the case for the OV$_\rho$ and FWO$_{\hat{\varrho}}$ bounds developed in this chapter, and their use in sections 3.6 and 3.7 reveals the effects of ρ and π_S on multi-task learning.

3.2 Preliminaries and Problem Statement

3.2.1 Multi-task GP Regression Model and The Setup for Analysis

We consider the following multi-task Gaussian process regression model that learns M related functions $\{f_m\}_{m=1}^M$ by placing a zero mean GP prior which directly induces correlations between tasks. Let y_m be an observation of the m^{th} function at $\boldsymbol{x}$. Then the model is given by

$$\langle f_m(\boldsymbol{x}) f_{m'}(\boldsymbol{x}')\rangle \overset{\text{def}}{=} K^{\text{f}}_{mm'} k^{\text{x}}(\boldsymbol{x},\boldsymbol{x}') \qquad y_m \sim \mathcal{N}(f_m(\boldsymbol{x}), \sigma_m^2), \tag{3.1}$$

where k^{x} is a covariance function over inputs, and K^{f} is a positive semi-definite matrix of inter-task similarities, and σ_m^2 is the noise variance for the m^{th} task.

The current focus is on the two tasks case, where the secondary task S is to help improve the performance of the primary task T; this is the *asymmetric multi-task learning* defined in chapter 1 on page 1. We fix K^{f} to be a correlation matrix, and let the variance be explained fully by k^{x}; the converse has been done in Bonilla et al. [2008]. Thus K^{f} is fully specified by the correlation $\rho \in [-1, 1]$ between the two

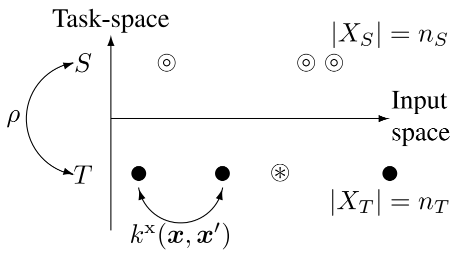


Figure 3.2: Multi-task GP model and the setup for analysis. The two tasks S and T have task correlation ρ. The data set X_T (resp. X_S) for task T (resp. S) consists of the ●s (resp. ◎s). The test location $\boldsymbol{x}_*$ for task T is denoted by ⊛.

tasks. We further fix the noise variances of the two tasks to be the same, say σ_{n}^2. For the training data, there are n_T (resp. n_S) observations $\boldsymbol{y}_T$ (resp. $\boldsymbol{y}_S$) at locations X_T (resp. X_S) for task T (resp. S). We use $n \stackrel{\text{def}}{=} n_T + n_S$ for the total number of observations, $\pi_S \stackrel{\text{def}}{=} n_S/n$ for the proportion of observations for task S, and also $X \stackrel{\text{def}}{=} X_T \cup X_S$. The aim is to infer the noise-free response f_{T*} for task T at $\boldsymbol{x}_*$. See Figure 3.2 for a summary.

The covariance matrix of the noisy training data is $K(\rho) + \sigma_{\mathrm{n}}^2 I$, where

$$K(\rho) \stackrel{\text{def}}{=} \begin{pmatrix} K_{TT}^{\mathrm{x}} & \rho K_{TS}^{\mathrm{x}} \\ \rho K_{ST}^{\mathrm{x}} & K_{SS}^{\mathrm{x}} \end{pmatrix}; \tag{3.2}$$

and K_{TT}^{x} (resp. K_{SS}^{x}) is the matrix of covariances (due to k^{x}) between locations in X_T (resp. X_S); K_{TS}^{x} is the matrix of cross-covariances from locations in X_T to locations in X_S; and K_{ST}^{x} is K_{TS}^{x} transposed. The posterior variance at $\boldsymbol{x}_*$ for task T is

$$\sigma_T^2(\boldsymbol{x}_*, \rho, \sigma_{\mathrm{n}}^2, X_T, X_S) = k_{**} - \boldsymbol{k}_*^{\mathrm{T}}(K(\rho) + \sigma_{\mathrm{n}}^2 I)^{-1}\boldsymbol{k}_*, \quad \text{where } \boldsymbol{k}_*^{\mathrm{T}} \stackrel{\text{def}}{=} \left((\boldsymbol{k}_{T*}^{\mathrm{x}})^{\mathrm{T}} \; \rho(\boldsymbol{k}_{S*}^{\mathrm{x}})^{\mathrm{T}}\right); \tag{3.3}$$

and k_{**} is the prior variance at $\boldsymbol{x}_*$, and $\boldsymbol{k}_{T*}^{\mathrm{x}}$ (resp. $\boldsymbol{k}_{S*}^{\mathrm{x}}$) is the vector of covariances (due to k^{x}) between locations in X_T (resp. X_S) and $\boldsymbol{x}_*$. Where appropriate and clear from context, we will suppress some of the parameters in $\sigma_T^2(\boldsymbol{x}_*, \rho, \sigma_{\mathrm{n}}^2, X_T, X_S)$, or use X for (X_T, X_S). Note that $\sigma_T^2(\rho) = \sigma_T^2(-\rho)$, so that $\sigma_T^2(1)$ is the same as $\sigma_T^2(-1)$; for brevity, we only write the former.

If the GP prior is correctly specified,[3] then it is straightforward to show that the posterior variance (3.3) is also the generalization error at $\boldsymbol{x}_*$ [Rasmussen and Williams, 2006, §7.3]. The latter is defined as

$$\left\langle (f_T^\star(\boldsymbol{x}_*) - \bar{f}_T(\boldsymbol{x}_*))^2 \right\rangle_{f_T^\star}, \tag{3.4}$$

where the predictor

$$\bar{f}_T(\boldsymbol{x}_*) = \boldsymbol{k}_*^{\mathrm{T}}(K(\rho) + \sigma_{\mathrm{n}}^2 I)^{-1} \left(\boldsymbol{y}_T^{\mathrm{T}} \; \boldsymbol{y}_S^{\mathrm{T}}\right)^{\mathrm{T}} \tag{3.5}$$

is the posterior mean at $\boldsymbol{x}_*$ for task T, and the expectation in (3.4) is taken over the distribution from which the true function $f_T^\star$ is drawn. In this chapter, in order to distinguish succinctly from the generalization error introduced in the next section, we use posterior variance to mean the generalization error at $\boldsymbol{x}_*$. Note that the actual y-values observed at X do not affect the posterior variance at any test location.

[3] As far as we are aware, other works on generalization errors and learning curves for GPs also rely on this assumption, with the following exceptions: Vivarelli [1998] relates the incorrectly specified case to the correctly specified case; for low dimensional problems, Sollich [2005] shows that optimized hyperparameters can lead to optimal learning rates; and Stein [1999b] investigates how mismatch in the spectral densities of the true and assumed covariance functions affects convergence to optimality.

Problem statement Given the above setting, the aim is to investigate how training observations for task S can benefit the predictions for task T. We measure the benefits using generalization error, learning curve and optimal error, and investigate how these quantities vary with ρ and π_S.

3.2.2 Generalization Errors, Learning Curves and Optimal Errors

We outline the general approach to obtain the generalization error and the learning curve [Rasmussen and Williams, 2006, §7.3] under our setting, where we have two tasks and are concerned with the primary task T. Let $p(\boldsymbol{x})$ be the probability density, common to both tasks, from which test and training locations are drawn, and assume that the GP prior is correctly specified. The generalization error for task T is obtained by averaging the posterior variance (i.e., generalization error at a test location) for task T over $\boldsymbol{x}_*$, and the learning curve for task T is obtained by averaging the generalization error over training sets X of a cardinality n:

$$\text{generalization error:} \qquad \epsilon_T(\rho, \sigma_{\mathrm{n}}^2, X_T, X_S) \overset{\text{def}}{=} \int \sigma_T^2(\boldsymbol{x}_*, \rho, \sigma_{\mathrm{n}}^2, X_T, X_S) p(\boldsymbol{x}_*) \mathrm{d}\boldsymbol{x}_* \tag{3.6}$$

$$\text{learning curve:} \qquad \epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) \overset{\text{def}}{=} \int \epsilon_T(\rho, \sigma_{\mathrm{n}}^2, X_T, X_S) p(X) \mathrm{d}X, \tag{3.7}$$

where the training locations in X are drawn i.i.d, that is, $p(X)$ factorizes completely into a product of $p(\boldsymbol{x})$'s. The learning curve (3.7) relates the model performance to the amount of training data, and thus indicates how "fast" the model learns. Besides averaging ϵ_T to obtain the learning curve, one may also use the optimal experimental design methodology and minimize ϵ_T over X to find the optimal generalization error [Ritter, 2000, chap. II]:

$$\text{optimal error:} \qquad \epsilon_T^{\mathrm{opt}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) \overset{\text{def}}{=} \min_X \epsilon_T(\rho, \sigma_{\mathrm{n}}^2, X_T, X_S). \tag{3.8}$$

Since the posterior variance does not depend on the actual y-values observed at X, neither do the generalization error, the learning curve and the optimal error.

Both $\epsilon_T(0, \sigma_{\mathrm{n}}^2, X_T, X_S)$ and $\epsilon_T(1, \sigma_{\mathrm{n}}^2, X_T, X_S)$ reduce to single-task GP cases; the former discards training observations at X_S, while the latter includes them. Similar analogues to single-task GP cases for $\epsilon_T^{\mathrm{avg}}(0, \sigma_{\mathrm{n}}^2, \pi_S, n)$ and $\epsilon_T^{\mathrm{avg}}(1, \sigma_{\mathrm{n}}^2, \pi_S, n)$, and $\epsilon_T^{\mathrm{opt}}(0, \sigma_{\mathrm{n}}^2, \pi_S, n)$ and $\epsilon_T^{\mathrm{opt}}(1, \sigma_{\mathrm{n}}^2, \pi_S, n)$ can be obtained. Note that $\epsilon_T^{\mathrm{avg}}$ and $\epsilon_T^{\mathrm{opt}}$ are well-defined since $\pi_S n = n_S \in \mathbb{N}_0$ by the definition of π_S.

We can relate the asymmetric multi-task case to the symmetric case, where the effect on task S by task T is included. Let $\epsilon_S^{\mathrm{avg}}$ be defined analogously to $\epsilon_T^{\mathrm{avg}}$, and assume that the distribution over the two tasks is the same during both training and testing. Then the learning curve averaged across both task S and task T is ϵ^{avg} given by

$$\epsilon^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) = \pi_S \epsilon_S^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) + (1 - \pi_S) \epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n). \tag{3.9}$$

By symmetry, $\epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, 1 - \pi_S, n) = \epsilon_S^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n)$, so that

$$\epsilon^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) = \pi_S \epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, 1 - \pi_S, n) + (1 - \pi_S) \epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n). \tag{3.10}$$

This symmetry is reflected by the lack of subscript T or S for ϵ^{avg}. From the above equation, it is also clear that $\epsilon^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) = \epsilon^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, 1 - \pi_S, n)$.

3.2.3 Eigen-analysis

The following results on eigen-analysis will be used in this chapter. Let $\bar{\kappa} \overset{\text{def}}{=} \kappa_1 > \kappa_2 > \kappa_3 > \ldots$ and $\phi_1(\cdot), \phi_2(\cdot), \phi_3(\cdot), \ldots$ be the eigenvalues and eigenfunctions of the covariance function k^{x} under the measure $p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$. That is, they satisfy the integral equation

$$\int k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')\phi_i(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \kappa_i\phi_i(\boldsymbol{x}'). \tag{3.11a}$$

These κ_is and $\phi_i(\cdot)$s are known as the *process eigenvalues and eigenfunctions* of the Gaussian process with covariance function k^{x}. Throughout, the eigenfunctions are assumed to be normalized, i.e.,

$$\int \phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \delta_{ij}. \tag{3.11b}$$

Mercer [1909] has given a construction of $k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$ using these eigenvalues and eigenfunctions:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sum_{i=1}^{\infty} \kappa_i\phi_i(\boldsymbol{x})\phi_i(\boldsymbol{x}'). \tag{3.11c}$$

Using Mercer's theorem, we can show that mean of the prior variance is the sum of the eigenvalues:

$$\int k(\boldsymbol{x}, \boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \sum_{i=1}^{\infty} \kappa_i \int \phi_i(\boldsymbol{x})\phi_i(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \sum_{i=1}^{\infty} \kappa_i. \tag{3.11d}$$

The eigenvalues and eigenfunctions can have analytical expressions for certain choices of measures $p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$ and covariance functions k^{x}. For example, if the measure is uniform on a one-dimensional unit interval and the covariance function is that of the unit variance stationary Ornstein-Uhlenbeck process

$$k^{\mathrm{x}}_{\mathrm{OU}}(x, x') \overset{\text{def}}{=} \exp\left(-|x - x'|/l\right), \tag{3.12}$$

then the eigenvalues and eigenfunctions are

$$\kappa_i = \frac{2}{lt_i^2 + l^{-1}} \qquad \phi_i(x) = \sqrt{\frac{2}{l^2t_i^2 + 2l + 1}}\Big(lt_i \cos t_i x + \sin t_i x\Big), \tag{3.13}$$

where the t_is are given implicitly by $\tan t_i = 2lt_i/[(lt_i)^2 - 1]$. This is derived in appendix B.6.2, following the outline by Hawkins [1989]. Another example is when the measure is a centred Gaussian with variance σ_{x}^2 on the real-line and the covariance function is squared exponential, i.e.,

$$k^{\mathrm{x}}_{\mathrm{SE}}(x, x') \overset{\text{def}}{=} \exp\left(-(x - x')^2/2l^2\right), \tag{3.14}$$

in which case Zhu et al. [1998] and Rasmussen and Williams [2006, §4.3.1] give

$$\kappa_i = \sqrt{\frac{2a}{A}}(b/A)^i \qquad \phi_i(x) = \frac{\sqrt[4]{c/a}}{\sqrt{2^i i!}}\exp(-(c - a)x^2)H_i(\sqrt{2c}x), \tag{3.15}$$

where H_i is the ith order Hermite polynomial, $a^{-1} = 4\sigma_{\mathrm{x}}^2$, $b^{-1} = 2l^2$, $c = \sqrt{a^2 + 2ab}$, $A = a + b + c$.

For symmetric multi-task learning, one may also consider a Gaussian process on the product space of tasks and input locations, that is, where the task indicator or descriptor t and input locations $\boldsymbol{x}$ are used jointly as inputs $(t, \boldsymbol{x})$. In the case of our two tasks setting, the integral equation for this product space will give the eigenvalues

$$\left(\frac{1}{2} \pm \sqrt{\frac{1}{4} - \omega}\right)\kappa_i \qquad \omega \overset{\text{def}}{=} \pi_S(1 - \pi_S)(1 - \rho^2), \tag{3.16}$$

where κ_i are the eigenvalues solely for $k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$; see appendix B.6.1 for the details. We shall see in section 3.6 that ω is crucial in understanding symmetric multi-task learning.

The Gaussian process prior is a probability distribution over functions, and the smoothness or continuity of the functions is closely related to the decay of the tail-end eigenvalues, as we shall now explain. In one-dimension with uniform $p(\boldsymbol{x})$ on the unit interval, Sacks and Ylvisaker [1966, 1968, 1970] give the regularity conditions on k^{x} for characterising the smoothness of the functions. Broadly speaking, if k^{x} satisfies the Sacks-Ylvisaker conditions of order s, then the Gaussian process is exactly s-times mean square differentiable, and its sample paths are s-times continuously differentiable almost surely (see Ritter et al. 1995, §3; Adler 1981, Theorem 3.4.1). Moreover, the Sacks-Ylvisaker conditions is related to the eigenvalues in this way: for Sacks-Ylvisaker conditions of order s, $\kappa_i \propto (\pi i)^{-2s-2}$ in the limit $i \to \infty$; see Ritter [2000, Proposition IV.10, Remark IV.2], and appendix B.8.1 on page 163. For example, the stationary Ornstein-Uhlenbeck process is of order $s = 0$, and its eigenvalues (3.13) satisfy $\lim_{i\to\infty} \kappa_i = 2(i\pi)^{-2}/l$ (see appendix B.6.2).

It will also be useful to relate the process eigenvalues to the eigenvalues of the covariance matrix K^{x}_{SS} (say) when the locations in X_S are sampled from $p(\boldsymbol{x})$. Let $\bar{\lambda} \overset{\text{def}}{=} \lambda_1 > \lambda_2 > \ldots > \lambda_{n_S} \overset{\text{def}}{=} \underline{\lambda}$ be the eigenvalues of K^{x}_{SS}; then $\kappa_i = \lim_{n_S\to\infty} \lambda_i/n_S$, $i = 1 \ldots n_S$. See e.g., Rasmussen and Williams [2006, §4.3.2] and Baker [1977, Theorem 3.4]. For finite n_S used in practice, the estimate λ_i/n_S for κ_i is better for the larger eigenvalues than for the smaller ones.

3.3 Generalization Error

In this section, we derive expressions for the generalization error (and the bounds thereon) for the two-tasks case in terms of the single-task case. To illustrate and further motivate the problem, Figure 3.3 on the facing page plots the posterior variance $\sigma^2_T(\boldsymbol{x}_*, \rho)$ as a function of $\boldsymbol{x}_*$ given two observations for task T and three observations for task S. We roughly follow Sollich and Halees [2002, Figure 2], and use squared exponential covariance function $k(x, x') = \exp(-(x - x')^2/2l^2)$ with length-scale $l = 0.11$ and noise variance $\sigma^2_{\mathrm{n}} = 0.05$. Six solid curves are plotted, corresponding to $\rho^2 = 0, 1/8, 1/4$, $1/2$, $3/4$ and 1 from top to bottom. The two dashed curves enveloping each solid curve are the lower and upper bounds derived in this section; the dashed curves are hardly visible because the bounds are rather tight. The horizontal dotted line at 0.05 is the prior noise variance.

Similar to the case of single-task learning, each training point creates a depression on the $\sigma^2_T(\boldsymbol{x}_*, \rho)$ surface [Williams and Vivarelli, 2000; Sollich and Halees, 2002]. However, while each training point for task T creates a “full” depression that reaches the prior noise variance, the depression created by each training point for task S depends on ρ, “deeper” depressions for larger ρ^2. From the figure, it is clear that the following trivial bounds on $\sigma^2_T(\boldsymbol{x}_*, \rho)$ hold:

Proposition 3.1. *For all $\boldsymbol{x}_*$, $\sigma^2_T(\boldsymbol{x}_*, 1) \leqslant \sigma^2_T(\boldsymbol{x}_*, \rho) \leqslant \sigma^2_T(\boldsymbol{x}_*, 0)$.*

This result can be derived from the information theoretic intuition used by Williams and Vivarelli [2000]: conditioning on additional information decreases the entropy, and a normally distributed random variable with variance v^2 has entropy $\log\sqrt{2\pi v^2 e}$, which increases monotonically with v^2. Since larger $|\rho|$

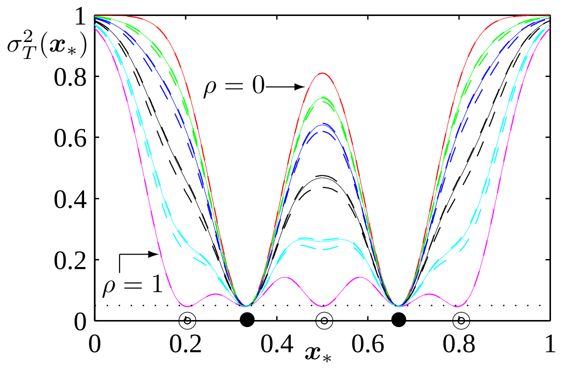


Figure 3.3: The effect of primary and secondary data on posterior variance. The curves are the posterior variances of each test location within $[0, 1]$ given data ●s at $1/3$ and $2/3$ for task T, and ⊚s at $1/5$, $1/2$ and $4/5$ for task S. The six solid curves correspond, from top to bottom, to $\rho^2 = 0, 1/8, 1/4, 1/2, 3/4$ and 1. The two dashed curves enveloping each solid curve are the lower and upper bounds given in Propositions 3.5 and 3.9.

means more information for task T, the above proposition is implied. Appendix B.2 gives a different proof that makes use of Proposition 3.5, which will be stated in section 3.3.2.

Integrating with respect to $\boldsymbol{x}_*$ then gives the following corollary:

Corollary 3.2. $\epsilon_T(1, \sigma_n^2, X_T, X_S) \leqslant \epsilon_T(\rho, \sigma_n^2, X_T, X_S) \leqslant \epsilon_T(0, \sigma_n^2, X_T, X_S)$.

Sections 3.3.2 and 3.3.3 derive lower and upper bounds that are tighter than the above trivial bounds. Prior to the bounds, we consider a degenerate case to illustrate the limitations of multi-task learning.

3.3.1 The Case of No Training Data for the Primary Task

It is clear that if there is no training data for the secondary task, that is, if $X_S = \emptyset$, then $\sigma_T^2(\boldsymbol{x}_* 1) = \sigma_T^2(\boldsymbol{x}_*, \rho) = \sigma_T^2(\boldsymbol{x}_* 0)$ for all $\boldsymbol{x}_*$ and ρ. In the converse case where there is no training data for the primary task, that is, $X_T = \emptyset$, we instead have the following proposition:

Proposition 3.3. *For all* $\boldsymbol{x}_*$, $\sigma_T^2(\boldsymbol{x}_*, \rho, \emptyset, X_S) = \rho^2 \sigma_T^2(\boldsymbol{x}_*, 1, \emptyset, X_S) + (1 - \rho^2) k_{**}$.

Proof.

$$\begin{aligned}\sigma_T^2(\boldsymbol{x}_*, \rho, \emptyset, X_S) &= k_{**} - \rho^2 (\boldsymbol{k}_{S*}^{\mathrm{x}})^{\mathrm{T}} (K_{SS}^{\mathrm{x}} + \sigma_n^2 I)^{-1} \boldsymbol{k}_{S*}^{\mathrm{x}} \\ &= (1 - \rho^2) k_{**} + \rho^2 \left[k_{**} - (\boldsymbol{k}_{S*}^{\mathrm{x}})^{\mathrm{T}} (K_{SS}^{\mathrm{x}} + \sigma_n^2 I)^{-1} \boldsymbol{k}_{S*}^{\mathrm{x}} \right] \\ &= (1 - \rho^2) k_{**} + \rho^2 \sigma_T^2(\boldsymbol{x}_*, 1, \emptyset, X_S). \end{aligned}$$ □

Hence the posterior variance is a weighted average of the prior variance k_{**} and the posterior variance at perfect correlation. When the cardinality of X_S increases under infill asymptotics [Cressie, 1993, §3.3],

$$\lim_{n_S \to \infty} \sigma_T^2(\boldsymbol{x}_*, 1, \emptyset, X_S) = 0 \qquad \Longrightarrow \qquad \lim_{n_S \to \infty} \sigma_T^2(\boldsymbol{x}_*, \rho, \emptyset, X_S) = (1 - \rho^2) k_{**}. \tag{3.17}$$

This is the limit for the posterior variance at *any* test location for task T, if one has training data only for the secondary task S. This is because a correlation of ρ between the tasks prevents any training location for task S from having correlation higher than ρ with a test location for task T. We can also understand this in terms of perceived distances between the locations. Suppose correlations in the input-space are given by an isotropic covariance function $k^{\mathrm{x}}(|\boldsymbol{x} - \boldsymbol{x}'|)$. If we translate correlations into distances between data locations, then any training location from task S is beyond a certain radius from any test location for task T. In contrast, a training location from task T may lay arbitrarily close to a test location for task T, subject to the constraints of noise.[4]

We obtain the generalization error in this degenerate case, by integrating Proposition 3.3 with respect to $p(\boldsymbol{x}_*)\mathrm{d}\boldsymbol{x}_*$ and using the fact that the mean prior variance is given by the sum of the process eigenvalues, i.e., $\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \sum \kappa_i$.

Corollary 3.4. $\epsilon_T(\rho,\sigma_{\mathrm{n}}^2,\emptyset,X_S) = \rho^2\epsilon_T(1,\sigma_{\mathrm{n}}^2,\emptyset,X_S) + (1-\rho^2)\sum_{i=1}^{\infty}\kappa_i$.

A directly corollary of the above result is that one cannot expect to do better than $(1-\rho^2)\sum\kappa_i$ on the average. Since the use of an incorrectly specified Gaussian process prior increases the generalization error [Vivarelli, 1998, appendix A.8], a lower bound for the incorrectly specified case is also $(1-\rho^2)\sum\kappa_i$.

3.3.2 A Lower Bound

When $X_T \neq \emptyset$, the correlations between locations in X_T and locations in X_S complicate the situation. However, since $\sigma_T^2(\rho)$ is a continuous and monotonically decreasing function of ρ, there exists an $\alpha \in [0,1]$, which depends on ρ, $\boldsymbol{x}_*$ and X, such that $\sigma_T^2(\rho) = \alpha\sigma_T^2(1) + (1-\alpha)\sigma_T^2(0)$. That α depends on $\boldsymbol{x}_*$ obstructs further analysis. The next proposition gives a lower bound $\underline{\sigma}_T^2(\rho)$ of the same form satisfying $\sigma_T^2(1) \leqslant \underline{\sigma}_T^2(\rho) \leqslant \sigma_T^2(\rho)$, where the mixing proportion is independent of $\boldsymbol{x}_*$.

Proposition 3.5. *Let* $\underline{\sigma}_T^2(\boldsymbol{x}_*,\rho) \overset{\mathrm{def}}{=} \rho^2\sigma_T^2(\boldsymbol{x}_*,1) + (1-\rho^2)\sigma_T^2(\boldsymbol{x}_*,0)$. *Then for all* $\boldsymbol{x}_*$*:*

(a) $\underline{\sigma}_T^2(\boldsymbol{x}_*,\rho) \leqslant \sigma_T^2(\boldsymbol{x}_*,\rho)$

(b) $\sigma_T^2(\boldsymbol{x}_*,\rho) - \underline{\sigma}_T^2(\boldsymbol{x}_*,\rho) \leqslant \rho^2(\sigma_T^2(\boldsymbol{x}_*,0) - \sigma_T^2(\boldsymbol{x}_*,1))$

(c) $\arg\max_{\rho^2}\left[\sigma_T^2(\boldsymbol{x}_*,\rho) - \underline{\sigma}_T^2(\boldsymbol{x}_*,\rho)\right] \geqslant 1/2$.

The proofs are in appendix B.1 on page 145. The lower bound $\underline{\sigma}_T^2(\rho)$ depends explicitly on ρ^2. It depends implicitly on π_S, which is the proportion of observations for task S, through the gap between $\sigma_T^2(1)$ and $\sigma_T^2(0)$. If there is no training data for the primary task, i.e., if $\pi_S = 1$, the bound reduces to Proposition 3.3, and becomes exact for all values of ρ. If $\pi_S = 0$, the bound also becomes exact. For $\pi_S \notin \{0,1\}$, the bound becomes exact when $\rho \in \{-1,0,1\}$. As from Figure 3.3 and later from our simulation results in section 3.5.5, this bound is rather tight. Part (b) of the proposition states the tightness of the bound: it is no more than factor ρ^2 of the gap between the trivial bounds $\sigma_T^2(0)$ and $\sigma_T^2(1)$. Part (c) of the proposition says that the bound is least tight for a value of ρ^2 greater than $1/2$. The lower bound $\underline{\sigma}_T^2(\rho)$ is plotted with the lower dashed lines in Figure 3.3.

The next corollary is obtained from Proposition 3.5 by integrating with respect to $p(\boldsymbol{x}_*)\mathrm{d}\boldsymbol{x}_*$. This is possible because ρ is independent of $\boldsymbol{x}_*$.

Corollary 3.6. *Let* $\underline{\epsilon}_T(\rho,\sigma_{\mathrm{n}}^2,X_T,X_S) \overset{\mathrm{def}}{=} \rho^2\epsilon_T(1,\sigma_{\mathrm{n}}^2,X_T,X_S) + (1-\rho^2)\epsilon_T(0,\sigma_{\mathrm{n}}^2,X_T,X_S)$. *Then*

(a) $\underline{\epsilon}_T(\rho,\sigma_{\mathrm{n}}^2,X_T,X_S) \leqslant \epsilon_T(\rho,\sigma_{\mathrm{n}}^2,X_T,X_S)$*, and*

(b) $\epsilon_T(\rho,\sigma_{\mathrm{n}}^2,X_T,X_S) - \underline{\epsilon}_T(\rho,\sigma_{\mathrm{n}}^2,X_T,X_S) \leqslant \rho^2\left(\epsilon_T(0,\sigma_{\mathrm{n}}^2,X_T,X_S) - \epsilon_T(1,\sigma_{\mathrm{n}}^2,X_T,X_S)\right)$.

[4] Noise also has the effect of decreasing the correlations between the data locations.

Of interest is the following question: instead of predicting the posterior mean of task T at $\boldsymbol{x}_*$ using the multi-task model given by equation 3.1, is there an alternative so that the generalization error at $\boldsymbol{x}_*$ is $\underline{\sigma}_T^2(\boldsymbol{x}_*, \rho)$? Since $\underline{\sigma}_T^2(\boldsymbol{x}_*, \rho)$ is better than $\sigma_T^2(\boldsymbol{x}_*, \rho)$ in terms of error, the alternative must involve more information about task T than the multi-task model. A direct way to incorporate this extra information is through the (noisy) observations $\boldsymbol{y}_T^S$ for task T at the locations X_S. The following proposition gives two ways in which $\boldsymbol{y}_T^S$ may be used, each giving a different error.

Proposition 3.7. *Given data $(\boldsymbol{y}_T, X_T)$ and $(\boldsymbol{y}_T^S, X_S)$, let $\bar{f}_1$ (resp. $\bar{f}_0$) be the posterior mean function of the single-task GP when $\rho = 1$ (resp. $\rho = 0$), i.e.,*

$$\bar{f}_1(\boldsymbol{x}_*) \stackrel{\text{def}}{=} \begin{pmatrix} \boldsymbol{k}_{T*}^{\text{x}} \\ \boldsymbol{k}_{S*}^{\text{x}} \end{pmatrix}^{\text{T}} \left(\begin{pmatrix} K_{TT}^{\text{x}} & K_{TS}^{\text{x}} \\ K_{ST}^{\text{x}} & K_{SS}^{\text{x}} \end{pmatrix} + \sigma_{\text{n}}^2 I \right)^{-1} \begin{pmatrix} \boldsymbol{y}_T \\ \boldsymbol{y}_T^S \end{pmatrix} \tag{3.18}$$

$$\bar{f}_0(\boldsymbol{x}_*) \stackrel{\text{def}}{=} (\boldsymbol{k}_{T*}^{\text{x}})^{\text{T}} (K_{TT}^{\text{x}} + \sigma_{\text{n}}^2 I)^{-1} \boldsymbol{y}_T. \tag{3.19}$$

Then the generalization error at $\boldsymbol{x}_*$ of predicting

(a) $\bar{f}_1(\boldsymbol{x}_*)$ with probability ρ^2 and $\bar{f}_0(\boldsymbol{x}_*)$ with probability $(1-\rho^2)$ is exactly $\underline{\sigma}_T^2(\boldsymbol{x}_*, \rho)$; and

(b) $\bar{f}_{lc}(\boldsymbol{x}_*) \stackrel{\text{def}}{=} \rho^2 \bar{f}_1(\boldsymbol{x}_*) + (1-\rho^2)\bar{f}_0(\boldsymbol{x}_*)$ is not more than $\underline{\sigma}_T^2(\boldsymbol{x}_*, \rho)$.

The proof for (a) is straightforward, and the proof for (b) is given in appendix B.3. Always predicting $\bar{f}_1(\boldsymbol{x}_*)$ leads to the trivial lower bound $\sigma_T^2(\boldsymbol{x}_*, 1)$. The predictors (a) and (b) above give lower bounds tighter than $\sigma_T^2(\boldsymbol{x}_*, 1)$ by "throwing away" information, leading to more errors. In particular, the randomized predictor (a) throws away more information than the linear combination predictor (b), so it gives a tighter lower bound on $\sigma_T^2(\boldsymbol{x}_*, \rho)$. Related to $\bar{f}_1(\boldsymbol{x}_*)$ is the predictor obtained when the observations $\boldsymbol{y}_T^S$ in (3.18) are replaced by the expectation of $\boldsymbol{y}_T^S$ given $\boldsymbol{y}_S$. This predictor actually produces more errors and gives an upper bound on the learning curve, as discussed later in section 3.5.2.2.

3.3.3 An Upper Bound via Equivalent Isotropic Noise at Secondary Locations

The following question motivates our upper bound: if the training locations in X_S had been observed for task T rather than for task S, *what is the variance $\tilde{\sigma}_{\text{n}}^2$ of the equivalent isotropic noise at X_S* so that the posterior variance remains the same? To answer this question, we first refine the definition of $\sigma_T^2(\cdot)$ to include a different noise variance parameter s^2 for the X_S observations:

$$\sigma_T^2(\boldsymbol{x}_*, \rho, \sigma_{\text{n}}^2, s^2, X_T, X_S) \stackrel{\text{def}}{=} k_{**} - \boldsymbol{k}_*^{\text{T}} \left[K(\rho) + \begin{pmatrix} \sigma_{\text{n}}^2 I & 0 \\ 0 & s^2 I \end{pmatrix} \right]^{-1} \boldsymbol{k}_*; \tag{3.20}$$

cf. (3.3). We may suppress the parameters $\boldsymbol{x}_*$, X_T and X_S when writing $\sigma_T^2(\cdot)$. The variance $\tilde{\sigma}_{\text{n}}^2$ of the equivalent isotropic noise is a function of $\boldsymbol{x}_*$ defined by the equation

$$\sigma_T^2(\boldsymbol{x}_*, 1, \sigma_{\text{n}}^2, \tilde{\sigma}_{\text{n}}^2) = \sigma_T^2(\boldsymbol{x}_*, \rho, \sigma_{\text{n}}^2, \sigma_{\text{n}}^2). \tag{3.21}$$

For any $\boldsymbol{x}_*$ there is always a $\tilde{\sigma}_{\text{n}}^2$ that satisfies the equation because the difference

$$\Delta(\rho, \sigma_{\text{n}}^2, s^2) \stackrel{\text{def}}{=} \sigma_T^2(\rho, \sigma_{\text{n}}^2, \sigma_{\text{n}}^2) - \sigma_T^2(1, \sigma_{\text{n}}^2, s^2) \tag{3.22}$$

is a continuous and monotonically decreasing function of s^2. To make progress, we seek an upper bound $\bar{\sigma}_{\text{n}}^2$ for $\tilde{\sigma}_{\text{n}}^2$ that is independent of the choice of $\boldsymbol{x}_*$; that is, $\Delta(\rho, \sigma_{\text{n}}^2, \bar{\sigma}_{\text{n}}^2) \leqslant 0$ for all test locations. Of interest is the tight upper bound $\bar{\bar{\sigma}}_{\text{n}}^2$, which is the minimum possible $\bar{\sigma}_{\text{n}}^2$, given in the next proposition. First, we need the following lemma.

Lemma 3.8. *Let $\beta \overset{\text{def}}{=} \rho^{-2} - 1$. In the terms of $\sigma_T^2(\rho, \sigma_\text{n}^2, \sigma_\text{n}^2)$, having data X_S for task S is equivalent to an additional correlated noise $\beta(K_{SS}^\text{x} + \sigma_\text{n}^2 I)$ at these observations for task T.*

Proof. Matrix $K(\rho)$ may be factorized as

$$K(\rho) = \begin{pmatrix} I & 0 \\ 0 & \rho I \end{pmatrix} \begin{pmatrix} K_{TT}^\text{x} & K_{TS}^\text{x} \\ K_{ST}^\text{x} & \rho^{-2} K_{SS}^\text{x} \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & \rho I \end{pmatrix}. \tag{3.23}$$

By using this factorization in the posterior variance (3.20) and taking out the $\left(\begin{smallmatrix} I & 0 \\ 0 & \rho I \end{smallmatrix}\right)$ factors, we obtain

$$\sigma_T^2(\rho, \sigma_\text{n}^2, s^2) = k_{**} - (\boldsymbol{k}_*^\text{x})^\text{T} [\Sigma(\rho, \sigma_\text{n}^2, s^2)]^{-1} \boldsymbol{k}_*^\text{x}, \tag{3.24}$$

where $(\boldsymbol{k}_*^\text{x})^\text{T} \overset{\text{def}}{=} \left((\boldsymbol{k}_{T*}^\text{x})^\text{T}, (\boldsymbol{k}_{S*}^\text{x})^\text{T}\right)$ and

$$\Sigma(\rho, \sigma_\text{n}^2, s^2) \overset{\text{def}}{=} \begin{pmatrix} K_{TT}^\text{x} & K_{TS}^\text{x} \\ K_{ST}^\text{x} & \rho^{-2} K_{SS}^\text{x} \end{pmatrix} + \begin{pmatrix} \sigma_\text{n}^2 I & 0 \\ 0 & \rho^{-2} s^2 I \end{pmatrix} = \Sigma(1, \sigma_\text{n}^2, s^2) + \beta \begin{pmatrix} 0 & 0 \\ 0 & K_{SS}^\text{x} + s^2 I \end{pmatrix}.$$

The second expression for Σ makes clear that, in the terms of $\sigma_T^2(\rho, \sigma_\text{n}^2, \sigma_\text{n}^2)$, having data X_S for task S is equivalent to an additional correlated noise $\beta(K_{SS}^\text{x} + \sigma_\text{n}^2 I)$ at these observations for task T. □

The above lemma motivates the question that began this section. Note that $\rho^{-2} \geqslant 1$, and hence $\beta \geqslant 0$. We are now ready to give the upper bound on the posterior variance of the predictions.

Proposition 3.9. *Let $\bar\lambda$ be the maximum eigenvalue of K_{SS}^x, $\beta \overset{\text{def}}{=} \rho^{-2} - 1$ and $\bar{\bar\sigma}_\text{n}^2 \overset{\text{def}}{=} \beta(\bar\lambda + \sigma_\text{n}^2) + \sigma_\text{n}^2$. Then*

$$\forall \boldsymbol{x}_* \, \sigma_T^2(\boldsymbol{x}_*, \rho, \sigma_\text{n}^2, \sigma_\text{n}^2) \leqslant \sigma_T^2(\boldsymbol{x}_*, 1, \sigma_\text{n}^2, \bar{\bar\sigma}_\text{n}^2).$$

The bound is tight in this sense: for any $\bar\sigma_\text{n}^2$,

$$\forall \boldsymbol{x}_* \, \sigma_T^2(\boldsymbol{x}_*, \rho, \sigma_\text{n}^2, \sigma_\text{n}^2) \leqslant \sigma_T^2(\boldsymbol{x}_*, 1, \sigma_\text{n}^2, \bar\sigma_\text{n}^2) \implies \forall \boldsymbol{x}_* \, \sigma_T^2(\boldsymbol{x}_*, \rho, \sigma_\text{n}^2, \bar{\bar\sigma}_\text{n}^2) \leqslant \sigma_T^2(\boldsymbol{x}_*, 1, \sigma_\text{n}^2, \bar\sigma_\text{n}^2).$$

Proof sketch. The increase in posterior variance due to having X_S at task S with noise variance σ_n^2 rather than having them at task T with noise variance s^2 is given by $\Delta(\rho, \sigma_\text{n}^2, s^2)$, which we may write as

$$\Delta(\rho, \sigma_\text{n}^2, s^2) = (\boldsymbol{k}_*^\text{x})^\text{T} \left[(\Sigma(1, \sigma_\text{n}^2, s^2))^{-1} - (\Sigma(\rho, \sigma_\text{n}^2, \sigma_\text{n}^2))^{-1}\right] \boldsymbol{k}_*^\text{x} \tag{3.25}$$

using equations 3.22 and 3.24. Recall that we seek an upper bound $\bar\sigma_\text{n}^2$ for $\tilde\sigma_\text{n}^2$ such that $\Delta(\rho, \sigma_\text{n}^2, \bar\sigma_\text{n}^2) \leqslant 0$ for all test locations. In general, this requires $\bar\sigma_\text{n}^2 \geqslant \bar{\bar\sigma}_\text{n}^2 \overset{\text{def}}{=} \beta(\bar\lambda + \sigma_\text{n}^2) + \sigma_\text{n}^2$; details can be found in section B.4. The tightness $\bar{\bar\sigma}_\text{n}^2$ is evident from the construction. □

The above upper bound is plotted with upper dashed lines in Figure 3.3. Intuitively, $\sigma_T^2(\boldsymbol{x}_*, 1, \sigma_\text{n}^2, \bar{\bar\sigma}_\text{n}^2)$ is the tight upper bound because it inflates the noise covariance at X_S to just sufficient isotropic noise, from $(\beta K_{SS}^\text{x} + \sigma_\text{n}^2 I/\rho^2)$ to $\bar{\bar\sigma}_\text{n}^2 I$. Analogous to Proposition 3.9, the tight lower bound on $\tilde\sigma_\text{n}^2$ is given by $\underline{\underline{\sigma}}_\text{n}^2 \overset{\text{def}}{=} \beta(\underline{\lambda} + \sigma_\text{n}^2) + \sigma_\text{n}^2$, where $\underline{\lambda}$ is the smallest eigenvalue of K_{SS}^x. In summary,

$$\rho^{-2}\sigma_\text{n}^2 \leqslant \underline{\underline{\sigma}}_\text{n}^2 \leqslant \tilde\sigma_\text{n}^2 \leqslant \bar{\bar\sigma}_\text{n}^2 \leqslant \bar\sigma_\text{n}^2,$$

where the leftmost inequality is obtained by substituting zero for $\underline{\lambda}$ in $\underline{\underline{\sigma}}_\text{n}^2$. Hence observing X_S at S is at most as "noisy" as an additional $\beta(\bar\lambda + \sigma_\text{n}^2)$ noise variance, and at least as "noisy" as an additional $\beta(\underline{\lambda} + \sigma_\text{n}^2)$ noise variance. Since β decreases with $|\rho|$, the additional noise variances are smaller when $|\rho|$ is larger, i.e., when the task S is more correlated with task T.

Finally, if we refine ϵ_T as we have done for σ_T^2 using equation 3.20, we obtain the following corollary by integrating with respect to $p(\boldsymbol{x}_*)\mathrm{d}\boldsymbol{x}_*$.

Corollary 3.10. *Let* $\bar{\epsilon}_T(\rho, \sigma_{\mathrm{n}}^2, \sigma_{\mathrm{n}}^2, X_T, X_S) \overset{\mathrm{def}}{=} \epsilon_T(1, \sigma_{\mathrm{n}}^2, \bar{\bar{\sigma}}_{\mathrm{n}}^2, X_T, X_S)$*. Then*

$$\bar{\epsilon}_T(\rho, \sigma_{\mathrm{n}}^2, \sigma_{\mathrm{n}}^2, X_T, X_S) \geqslant \epsilon_T(\rho, \sigma_{\mathrm{n}}^2, \sigma_{\mathrm{n}}^2, X_T, X_S).$$

Scaling with n_S We give a description of how the above bounds scale with n_S, using the results stated in section 3.2.3. For large enough n_S, we may write $\bar{\lambda} \approx n_S \bar{\kappa}$ and $\underline{\lambda} \approx n_S \kappa_{n_S}$. Furthermore, for uniformly distributed inputs in the one-dimension unit interval, if the covariance function satisfies Sacks-Ylvisaker conditions of order s, then $\kappa_{n_S} = \Theta\left((\pi n_S)^{-2s-2}\right)$, so that $\underline{\lambda} = \Theta\left((\pi n_S)^{-2s-1}\right)$. Since $\bar{\bar{\sigma}}_{\mathrm{n}}^2$ and $\underline{\underline{\sigma}}_{\mathrm{n}}^2$ are linear in $\bar{\lambda}$ and $\underline{\lambda}$, we have $\bar{\bar{\sigma}}_{\mathrm{n}}^2 = \rho^{-2}\sigma_{\mathrm{n}}^2 + \beta\,\Theta(n_S)$ and $\underline{\underline{\sigma}}_{\mathrm{n}}^2 = \rho^{-2}\sigma_{\mathrm{n}}^2 + \beta\,\Theta\left(n_S^{-2s-1}\right)$. For the upper bound, note that although $\bar{\bar{\sigma}}_{\mathrm{n}}^2$ scales linearly with n_S, the eigenvalues of $K(1)$ scale with n, so $\sigma_T^2(1, \sigma_{\mathrm{n}}^2, \bar{\bar{\sigma}}_{\mathrm{n}}^2)$ depends on $\pi_S \overset{\mathrm{def}}{=} n_S/n$. In contrast the lower bound $\underline{\underline{\sigma}}_{\mathrm{n}}^2$ is dominated by $\rho^{-2}\sigma_{\mathrm{n}}^2$, so $\sigma_T^2(1, \sigma_{\mathrm{n}}^2, \underline{\underline{\sigma}}_{\mathrm{n}}^2)$ does not depend on π_S even for moderate sizes n_S. This lack of dependence on π_S means that the lower bound is not as useful as the upper bound for understanding multi-task learning.

3.3.4 Exact Computation of Generalization Error

The factorization of σ_T^2 expressed by equation 3.24 in Lemma 3.8 allows the generalization error to be computed exactly in certain cases. We replace the quadratic form in (3.24) by the matrix trace $\mathrm{tr}(\Sigma^{-1}\boldsymbol{k}_*^{\mathrm{x}}(\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}})$ and then integrate out $\boldsymbol{x}_*$ under $p(\boldsymbol{x}_*)$ to give

$$\epsilon_T(\rho, \sigma_{\mathrm{n}}^2, X_T, X_S) = \langle k_{**} \rangle - \mathrm{tr}\left(\Sigma^{-1}\langle \boldsymbol{k}_*^{\mathrm{x}}(\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} \rangle\right) = \textstyle\sum_{i=1}^{\infty} \kappa_i - \mathrm{tr}\left(\Sigma^{-1}M\right),$$

where Σ denotes $\Sigma(\rho, \sigma_{\mathrm{n}}^2, \sigma_{\mathrm{n}}^2)$, the expectations are taken over $\boldsymbol{x}_*$, and M is an n-by-n matrix with the $(p,q)\mathrm{th}$ entry

$$M_{pq} \overset{\mathrm{def}}{=} \int k^{\mathrm{x}}(\boldsymbol{x}_p, \boldsymbol{x}_*)\, k^{\mathrm{x}}(\boldsymbol{x}_q, \boldsymbol{x}_*)\, p(\boldsymbol{x}_*)\mathrm{d}\boldsymbol{x}_* = \sum_{i=1}^{\infty} \kappa_i^2 \phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q), \tag{3.26}$$

where $\boldsymbol{x}_p, \boldsymbol{x}_q \in X$ are the $p\mathrm{th}$ and $q\mathrm{th}$ observed locations. When the eigenfunctions $\phi_i(\cdot)$s are not bounded, the infinite-summation expression for M_{pq} is often difficult to use. Nevertheless, analytical results for M_{pq} are still possible in some cases using the integral expression. An example is the case of the squared exponential covariance function with normally distributed $\boldsymbol{x}$, when the integrand is a product of three Gaussians. Another example is the case of the covariance function of the stationary Ornstein-Uhlenbeck process with uniformly distributed $\boldsymbol{x}$. Expressions for both examples are given in section B.7.2.

3.4 Optimal Error when the Primary Task has no Training Data

If training examples are provided only for task S, then the optimal performance of task T is governed by the following proposition.

Proposition 3.11. *Under optimal sampling on a 1-d space, if the covariance function satisfies Sacks-Ylvisaker conditions of order s, then* $\epsilon_T^{opt}(\rho, \sigma^2, 1, n) = \Theta(n_S^{-(2s+1)/(2s+2)}) + (1-\rho^2)\sum_{i=1}^{\infty} \kappa_i$.

Proof. We obtain $\epsilon_T^{\text{opt}}(\rho, \sigma^2, 1, n) = \rho^2 \epsilon_T^{\text{opt}}(1, \sigma_{\text{n}}^2, 1, n) + (1-\rho^2)\sum_{i=1}^{\infty}\kappa_i$ by minimizing Corollary 3.4 wrt X_S. Under the same conditions as the proposition, the optimal generalization error of the single-task GP decays with training set size n as $\Theta(n^{-(2s+1)/(2s+2)})$ [Ritter 2000, Proposition V.3; Ritter 1996, Remark 5]. Thus $\rho^2\epsilon_T^{\text{opt}}(1, \sigma_{\text{n}}^2, 1, n) = \rho^2\Theta(n_S^{-(2s+1)/(2s+2)}) = \Theta(n_S^{-(2s+1)/(2s+2)})$. □

3.5 Theoretical Bounds on the Learning Curve

Using the results from section 3.3, lower and upper bounds on the learning curve may be computed by averaging over the choice of X using Monte Carlo approximation.[5] For example, using Corollary 3.2 and integrating with respect to $p(X)\mathrm{d}X$ gives the following trivial bounds on the learning curve:

Corollary 3.12. $\epsilon_T^{avg}(1, \sigma_{\text{n}}^2, \pi_S, n) \leqslant \epsilon_T^{avg}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \leqslant \epsilon_T^{avg}(0, \sigma_{\text{n}}^2, \pi_S, n)$.

The gap between the trivial bounds can be analyzed as follows. Recall that $\pi_S n \in \mathbb{N}_0$ by definition. Observe that $\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, \pi_S, (1-\pi_S)n) = \epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, n)$, since both effectively give $(1-\pi_S)n$ examples for task T. Therefore $\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, \pi_S, n)$ is equivalent to $\epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, n)$ scaled along the n-axis by the factor $(1-\pi_S) \in [0,1]$, and hence the gap between the trivial bounds becomes wider with π_S.

For future reference, we also give the following result, which is obtained by using Corollary 3.4, integrating with respect to $p(X)\mathrm{d}X$ and noting that $\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 1, n) = \epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 0, n)$.

Corollary 3.13.
$$\begin{aligned}\epsilon_T^{avg}(\rho, \sigma_{\text{n}}^2, 1, n) &= \rho^2\epsilon_T^{avg}(1, \sigma_{\text{n}}^2, 1, n) + (1-\rho^2)\textstyle\sum_{i=1}^{\infty}\kappa_i \\ &= \rho^2\epsilon_T^{avg}(1, \sigma_{\text{n}}^2, 0, n) + (1-\rho^2)\textstyle\sum_{i=1}^{\infty}\kappa_i.\end{aligned}$$

In the rest of this section, we derive non-trivial theoretical bounds on the learning curve before providing simulation results. Theoretical bounds are particularly attractive for high-dimensional input-spaces, on which Monte Carlo approximation is harder.

3.5.1 OV-type Lower Bounds

For the single-task Gaussian process, Opper and Vivarelli [1999] have shown that its learning curve is bounded from below by $\sigma_{\text{n}}^2\sum_{i=1}^{\infty}\kappa_i/(\sigma_{\text{n}}^2 + n\kappa_i)$, which becomes exact in the asymptotic limit $n \to \infty$. We shall call this the single-task Opper-Vivarelli (OV) bound. This lower bound can be readily combined with Corollary 3.6a to give

Proposition 3.14. *The following three lower bounds on $\epsilon_T^{avg}(\rho, \sigma_{\text{n}}^2, \pi_S, n)$ are equivalent.*

(a) $$\epsilon_T^{avg}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \geqslant \rho^2\sigma_{\text{n}}^2\sum_{i=1}^{\infty}\frac{\kappa_i}{\sigma_{\text{n}}^2 + n\kappa_i} + (1-\rho^2)\sigma_{\text{n}}^2\sum_{i=1}^{\infty}\frac{\kappa_i}{\sigma_{\text{n}}^2 + (1-\pi_S)n\kappa_i},$$

(b) $$\epsilon_T^{avg}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \geqslant \sigma_{\text{n}}^2\sum_{i=1}^{\infty}\frac{b_i^1\kappa_i}{\sigma_{\text{n}}^2 + n\kappa_i}, \qquad \textit{where } b_i^1 \overset{\text{def}}{=} \frac{\sigma_{\text{n}}^2 + (1-\rho^2\pi_S)n\kappa_i}{\sigma_{\text{n}}^2 + (1-\pi_S)n\kappa_i}, \textit{ and}$$

(c) $$\epsilon_T^{avg}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \geqslant \sigma_{\text{n}}^2\sum_{i=1}^{\infty}\frac{b_i^0\kappa_i}{\sigma_{\text{n}}^2 + (1-\pi_S)n\kappa_i}, \textit{ where } b_i^0 \overset{\text{def}}{=} \frac{\sigma_{\text{n}}^2 + (1-\rho^2\pi_S)n\kappa_i}{\sigma_{\text{n}}^2 + n\kappa_i}.$$

[5] Other than using Monte Carlo approximation, we can also obtain approximate lower bounds by combining Corollary 3.6a and the approximate learning curves for single-task GPs (see e.g., Sollich and Halees [2002]).

Proof sketch. To obtain (a), we integrate Corollary 3.6a with respect to $p(X)\mathrm{d}X$, and apply the single-task OV bound twice. For (b), its ith summand is obtained by combining the corresponding pair of ith summands in (a). Inequality (c) is obtained from (b) by swapping the denominator of b_i^1 with that of $\kappa_i/(\sigma_\mathrm{n}^2 + n\kappa_i)$ for every i. $\square$

For fixed σ_n^2, π_S and n, denote the above bound by OV_ρ. Then

$$\mathrm{OV}_0 = \sigma_\mathrm{n}^2 \sum_{i=1}^{\infty} \frac{\kappa_i}{\sigma_\mathrm{n}^2 + (1-\pi_S)n\kappa_i} \qquad\qquad \mathrm{OV}_1 = \sigma_\mathrm{n}^2 \sum_{i=1}^{\infty} \frac{\kappa_i}{\sigma_\mathrm{n}^2 + n\kappa_i}$$

are both single-task bounds. In particular, from Corollary 3.12, we have that the OV_1 is a lower bound on $\epsilon_T^{\mathrm{avg}}(\rho, \sigma_\mathrm{n}^2, \pi_S, n)$, since it is a lower bound on $\epsilon_T^{\mathrm{avg}}(1, \sigma_\mathrm{n}^2, \pi_S, n)$. How does the lower bound OV_1 compare with the lower bound OV_ρ? From Proposition 3.14a, it is clear from the "mixture" nature of the bound that the two-tasks bound OV_ρ is always better than OV_1. As ρ^2 decreases, the two-tasks bound moves towards the OV_0; and as π_S increases, the gap between OV_0 and OV_1 increases. In addition, the gap is also larger for rougher processes, which are harder to learn.[6] Therefore, the relative tightness of OV_ρ over OV_1 is more noticeable for lower ρ^2, higher π_S and rougher processes.

Proposition 3.14b is useful for comparing the OV_ρ bound with the OV_1 bound. Each summand for the two-tasks case is a factor b_i^1 of the corresponding summand for the single-task case. By considering the limits $n = 0$ and $n \to \infty$, we obtain $b_i^1 \in [1, (1-\rho^2\pi_S)/(1-\pi_S)[$. Hence OV_ρ is larger than OV_1 by at most $(1-\rho^2\pi_S)/(1-\pi_S)$ times. Similarly, Proposition 3.14c is useful for comparing with OV_0: now each summand for the the two-tasks case is a factor $b_i^0 \in](1-\rho^2\pi_S), 1]$ of the corresponding single-task one. Hence OV_ρ is less than OV_0 by up to $\rho^2\pi_S$ times. In terms of the lower bound on the learning curve, this is the limit to which asymmetric multi-task learning can outperform the single-task learning that ignores the secondary task. We shall encounter a factor similar to $(1-\rho^2\pi_S)$ when we examine error deflation factors in section 3.6.2.

It is also worthwhile to see how multi-task learning is affected by the amount of noise in the observations. To this end, we fix the κ_is, ρ^2, π_S and n, and see b_i^1 and b_i^2 as functions of the observation noise variance σ_n^2. Then clearly

$$\sigma_\mathrm{n}^2 < (\sigma_\mathrm{n}^2)' \qquad \Longrightarrow \qquad b_i^1(\sigma_\mathrm{n}^2) > b_i^1((\sigma_\mathrm{n}^2)') \qquad \text{and} \qquad b_i^0(\sigma_\mathrm{n}^2) < b_i^0((\sigma_\mathrm{n}^2)'), \tag{3.27}$$

so b_i^1 and b_i^0 become closer to one as the noise increases. The means that the effect of multi-task learning on the OV_ρ lower bound decreases.

3.5.2 FWO-type Upper Bounds

In this section, we give upper bounds on the learning curve for the primary task T. These upper bounds are based on the variational approach of Ferrari Trecate, Williams, and Opper [1999]. To obtain the bounds, we require a variant of the Lemma 4 from Ferrari Trecate et al.. First, some additional notations are necessary. Let $\boldsymbol{y} \overset{\mathrm{def}}{=} (y_1 \dots y_n)^\mathrm{T}$ be the n values observed at the set of data locations

[6] The less times a process is mean-square differentiable, the rougher it is. For the modified Bessel covariance function of orders one to three, Williams and Vivarelli [2000, §6] have verified that a rougher process has a learning curve that decays slower. Intuitively, a rougher process is more "wriggly" and requires more examples to "pin" down. In our context, this means the effect of π_S is more pronounced in the single-task OV bound for $\rho = 0$ since it determines the number of examples for task T.

$X \overset{\text{def}}{=} \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ generated via a model $\mathcal{M}$ that will be specified later. Recall that $\phi_1, \phi_2, \ldots$ are the eigenfunctions of the covariance function k^{x}; and $\int \phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \delta_{ij}$. We introduce an (infinite) matrix Φ with entries $\Phi_{ij} \overset{\text{def}}{=} \phi_j(\boldsymbol{x}_i)$, an (infinite) vector function $\boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) \overset{\text{def}}{=} (\phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{x}), \ldots)^{\mathrm{T}}$ and an (infinite) diagonal matrix Λ_κ with κ_i on the diagonals.[7] Given the data, we have the space of functions

$$\mathcal{H}_\star \overset{\text{def}}{=} \{\boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x})L\boldsymbol{y} \mid L \in \mathbb{R}^{\infty \times n}\}.$$

The aim is to use a function g from $\mathcal{H}_\star$ to estimate the true function $f^\star$ drawn from the Gaussian process with zero mean and covariance function $k^{\mathrm{x}}(\cdot,\cdot)$. The quality of this estimation can be evaluated using the following lemma.

Lemma 3.15. *(cf. Ferrari Trecate et al. [1999, Lemma 4]) The generalization error of a function $g \in \mathcal{H}_\star$ is*

$$\begin{aligned}\epsilon(g \in \mathcal{H}_\star, X, \mathcal{M}) &\overset{\text{def}}{=} \left\langle (f^\star(\boldsymbol{x}) - g(\boldsymbol{x}))^2 \right\rangle_{f^\star, \boldsymbol{y}, \boldsymbol{x}} \\ &= \sum_{i=1}^{\infty} \kappa_i + \operatorname{tr}\left(L \left\langle \boldsymbol{y}\boldsymbol{y}^{\mathrm{T}} \right\rangle_{\boldsymbol{y}} L^{\mathrm{T}}\right) - 2\operatorname{tr}\left(L \left\langle \langle \boldsymbol{y} f^\star(\boldsymbol{x})\rangle_{f^\star,\boldsymbol{y}} \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) \right\rangle_{\boldsymbol{x}}\right)\end{aligned}$$

The proof is given in appendix B.5.1.

In order to proceed, it is necessary to specify how $\boldsymbol{y}$ is obtained. For the single-task GP with isotropic noise, and under correct prior specification, each entry in $\boldsymbol{y}$ is generated via

$$\mathcal{M}_{\text{iso}}: \quad y(\boldsymbol{x}) \sim \mathcal{N}(f^\star(\boldsymbol{x}), \sigma_{\mathrm{n}}^2). \tag{3.28}$$

Within this setting, it can be shown that minimizing the generalization error with respect to L leads to the GP mean predictor [Ferrari Trecate et al., 1999, Theorem 5]; see also Kimeldorf and Wahba [1970], and Ritter [2000, Proposition V.1]. The single-task Ferrari-Williams-Opper (FWO) bound on the learning curve of the GP is obtained by minimizing $\langle \epsilon(g \in \mathcal{H}_\star, X, \mathcal{M}_{\text{iso}})\rangle_X$ with respect to g within only the sub-space of functions

$$\mathcal{H}_1 \overset{\text{def}}{=} \{\boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x})D\Phi^{\mathrm{T}}\boldsymbol{y} \mid D \text{ is a diagonal matrix}\}.$$

This results in an upper bound on the learning curve. This is because $\mathcal{H}_1 \subseteq \mathcal{H}_\star$, so that minimizing $\langle \epsilon(g \in \mathcal{H}_1, X, \mathcal{M}_{\text{iso}})\rangle_X$ naturally gives predictors that cannot outperform the GP mean predictor. The form of functions in $\mathcal{H}_1$ is motivated by Projected Bayes Regression [Ferrari Trecate et al., 1999, Definition 1], wherein the L in $\mathcal{H}_\star$ is constrained to be $M\Phi^{\mathrm{T}}$ for a square matrix M.

To obtain the upper bound on the learning curve for the primary task T when it is assisted in learning by a secondary task S, we use the asymmetric two-tasks setup described in section 3.2.1 instead of the $\mathcal{M}_{\text{iso}}$ model (3.28). Our interest is in the primary task T, so we identify $f^\star$ with the function for task T, i.e., $f^\star$ and f_T are synonymous. As before, let f_S be the function for the secondary task S. Under the multi-task Gaussian process model for two tasks, we have $\langle f^\star(\boldsymbol{x}) f_S(\boldsymbol{x}')\rangle = \rho k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$, and $\boldsymbol{y}$ is generated via

$$\mathcal{M}_{\text{mt}}: \quad y(\boldsymbol{x}) \sim \begin{cases} \mathcal{N}\left(f^\star(\boldsymbol{x}), \sigma_{\mathrm{n}}^2\right) & \text{if } \boldsymbol{x} \in X_T \\ \mathcal{N}\left(f_S(\boldsymbol{x}), \sigma_{\mathrm{n}}^2\right) & \text{if } \boldsymbol{x} \in X_S. \end{cases} \tag{3.29}$$

[7] In Ferrari Trecate et al. [1999], where the focus is on finite-dimensional approximating Gaussian processes, only the m major eigenvalues and eigenfunctions are used. Since our focus is to obtain an upper bound on the learning curve, we follow Sollich and Halees [2002] and let $m \to \infty$.

Let $\boldsymbol{y}$ and X be ordered such that the first $n_T \stackrel{\text{def}}{=} |X_T|$ elements are the observations for task T. Applying Lemma 3.15 to the above data generation model gives

$$\epsilon(g \in \mathcal{H}_\star, X, \mathcal{M}_{\text{mt}}) = \sum_{i=1}^{\infty} \kappa_i + \operatorname{tr}\left(L(K(\rho) + \sigma_{\text{n}}^2 I)L^{\text{T}}\right) - 2\operatorname{tr}\left(LI(\rho)\Phi\Lambda_\kappa\right), \tag{3.30}$$

where $K(\rho)$ is defined by equation 3.2 on page 55, and

$$I(s) \stackrel{\text{def}}{=} \begin{pmatrix} I_{n_T \times n_T} & 0 \\ 0 & sI_{n_S \times n_S} \end{pmatrix}. \tag{3.31}$$

This is shown in appendix B.5.3. In addition, we shall constrain g to be from the space of functions

$$\mathcal{H}_\varrho \stackrel{\text{def}}{=} \{\boldsymbol{\phi}^{\text{T}}(\boldsymbol{x}) D \Phi^{\text{T}} I(\varrho)\boldsymbol{y} \mid D \text{ is a diagonal matrix}\}$$

parameterized by ϱ, which may be fixed or optimized for. Let d_i be the ith entry of matrix D, and define

$$G_\varrho \stackrel{\text{def}}{=} \begin{pmatrix} K_{TT}^{\text{x}} & \rho\varrho K_{TS}^{\text{x}} \\ \rho\varrho K_{ST}^{\text{x}} & \varrho^2 K_{SS}^{\text{x}} \end{pmatrix} + \sigma_{\text{n}}^2 I(\varrho^2). \tag{3.32a}$$

Then from equation 3.30 and the factorization in $\mathcal{H}_\varrho$, which gives $L = D\Phi^{\text{T}} I(\varrho)$, we have

$$\begin{aligned} \epsilon(g \in \mathcal{H}_\varrho, X, \mathcal{M}_{\text{mt}}) &= \sum_{i=1}^{\infty} \kappa_i + \operatorname{tr}\left(D\Phi^{\text{T}} G_\varrho \Phi D\right) - 2\operatorname{tr}\left(D\Phi^{\text{T}} I(\rho\varrho)\Phi\Lambda_\kappa\right) \\ &= \sum_{i=1}^{\infty} \kappa_i + \sum_{i=1}^{\infty} d_i^2 \left(\Phi^{\text{T}} G_\varrho \Phi\right)_{ii} - 2\sum_{i=1}^{\infty} d_i \left(\Phi^{\text{T}} I(\rho\varrho)\Phi\Lambda_\kappa\right)_{ii}. \end{aligned} \tag{3.32b}$$

Since $\mathcal{H}_\varrho \subseteq \mathcal{H}_\star$, minimizing $\langle \epsilon(g \in \mathcal{H}_\varrho, X, \mathcal{M}_{\text{mt}}) \rangle_X$ will give an upper bound on the learning curve for task T. Following the derivation in appendix B.5.2, we have the next proposition.

Proposition 3.16. *Let* $\alpha(x) \stackrel{\text{def}}{=} (1 - \pi_S) + \pi_S x$ *and*

$$\begin{aligned} c_i(\varrho) = &\left\{ \frac{1}{\alpha(\varrho^2)} \left[(1-\pi_S)^2 + \varrho^2\pi_S^2 + 2\rho\varrho\pi_S(1-\pi_S)\right] n - 1 \right\} \kappa_i \\ &+ \int k^{\text{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} + \sigma_{\text{n}}^2. \end{aligned}$$

Then, for all ϱ,

$$\epsilon_T^{avg}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \leqslant \sum_{i=1}^{\infty} \kappa_i - n \frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)} \sum_{i=1}^{\infty} \frac{\kappa_i^2}{c_i(\varrho)}. \tag{3.33}$$

We shall call this the FWO$_\varrho$ upper bound. The single-task FWO upper bound is recovered with $\rho = \varrho = \pm 1$ or $\pi_S = 0$.

The tightness of the FWO$_\varrho$ upper bound depends on ϱ. One trivial setting is $\varrho = 0$, giving the FWO$_0$ bound. It is clear from equation 3.32 that $\varrho = 0$ will "zero-out" the observations for task S, so that we are effectively only left with the observations for task T. The FWO$_0$ bound is therefore also the single-task FWO upper bound on the learning curve $\epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, n)$, which in turn is an upper bound on $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n)$ according to Corollary 3.12.

For the rest of this section, we shall look at three other settings of ϱ. The first is $\varrho = 1/\rho$; this is related to the equivalent noise construction of section 3.3.3. The second is $\varrho = \rho$, and this has a natural interpretation of a two-step prediction process. Finally, the third is $\varrho = \hat{\varrho}$, where $\hat{\varrho}$ is chosen to approximately minimize FWO$_{\hat{\varrho}}$.

3.5.2.1 Equivalent Noise and the FWO$_{1/\rho}$ Bound

An upper bound on the learning curve for task T may be obtained by using $\varrho = 1/\rho$ in Proposition 3.16. This upper bound, called the FWO$_{1/\rho}$ bound, may be understood in an intuitive way, based on the equivalent noise construction of section 3.3.3. Instead of proceeding from the upper bound $\sigma_T^2(1, \sigma_n^2, \bar{\bar{\sigma}}_n^2)$, we proceed directly from the exact posterior variance given by equation 3.24. Suppose that the data is generated via a single-task GP with correlated observation noise

$$\mathcal{M}_{\text{co}}: \quad y(\boldsymbol{x}) \sim \mathcal{GP}(f^\star(\boldsymbol{x}), \gamma^2(\boldsymbol{x}, \boldsymbol{x}')), \tag{3.34}$$

where the observation noise (co)variance is

$$\gamma^2(\boldsymbol{x}_i, \boldsymbol{x}_j) \stackrel{\text{def}}{=} \delta(\boldsymbol{x}_i \in X_T)\delta(\boldsymbol{x}_j \in X_T)\,\delta_{ij}\sigma_n^2 \;+\; \delta(\boldsymbol{x}_i \in X_S)\delta(\boldsymbol{x}_j \in X_S)\;\big[\beta k^{\text{x}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \rho^{-2}\delta_{ij}\sigma_n^2\big].$$

By Lemma 3.8 on page 62, this data model gives the same posterior variance for task T as the multi-task data model $\mathcal{M}_{\text{mt}}$. Together with the constraint that g belongs to $\mathcal{H}_1$, we have the following generalization error using Lemma 3.15:

$$\epsilon(g \in \mathcal{H}_1, X, \mathcal{M}_{\text{co}}) = \sum_{i=1}^{\infty} \kappa_i + \operatorname{tr}\left(D\Phi^{\text{T}} G_{1/\rho}\Phi D\right) - 2\operatorname{tr}\left(D\Phi^{\text{T}}\Phi\Lambda_\kappa\right), \tag{3.35}$$

where $G_{1/\rho}$ is G_ϱ with $\varrho = 1/\rho$, and G_ϱ is defined by (3.32a). The derivation is in appendix B.5.4. Compared with equation 3.32, it is clear that

$$\epsilon(g \in \mathcal{H}_1, X, \mathcal{M}_{\text{co}}) = \epsilon(g \in \mathcal{H}_{1/\rho}, X, \mathcal{M}_{\text{mt}}).$$

These are the two ways in which the upper bound FWO$_{1/\rho}$ may be obtained. The equivalent (correlated) noise construction is useful for comparing to the single-task FWO bound since both constrain g to be from the same hypothesis space $\mathcal{H}_1$. Of independent interest is that this construction provides an approach to upper bound the learning curve of single-task GP with observations corrupted by correlated noise, where the correlated noise needs not arise from the multi-task setting considered here. The following variant of Theorem 6 from Ferrari Trecate et al. [1999] states this upper bound.

Theorem 3.17. *(Ferrari Trecate et al. [1999], modified second part of Theorem 6) Consider a zero-mean GP with covariance function $k^{\text{x}}(\cdot,\cdot)$, and eigenvalues κ_i and eigenfunctions $\phi_i(\cdot)$ under the measure $p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$; and suppose that the noise (co)variances of the observations are given by $\gamma^2(\cdot,\cdot)$. For n observations $\{\boldsymbol{x}_i\}_{i=1}^n$, let H and Φ be matrices such that $H_{ij} \stackrel{\text{def}}{=} k^{\text{x}}(\boldsymbol{x}_i, \boldsymbol{x}_j) + \gamma^2(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $\Phi_{ij} \stackrel{\text{def}}{=} \phi_j(\boldsymbol{x}_i)$. Then the learning curve at n is upper-bounded by*

$$\sum_{i=1}^{\infty} \kappa_i - n\sum_{i=1}^{\infty} \kappa_i^2 / c_i, \qquad \text{where } c_i \stackrel{\text{def}}{=} \left\langle (\Phi^{\text{T}} H \Phi)_{ii} \right\rangle / n,$$

and the expectation in c_i is taken over the set of n input locations drawn independently from $p(\boldsymbol{x})$.

In the next section, we shall look at the FWO$_\rho$ bound which is provably better than the FWO$_{1/\rho}$ bound.

3.5.2.2 Transformation of Secondary Observations and the FWO$_\rho$ Bound

We now turn our attention to the FWO$_\rho$ bound, which is the FWO$_\varrho$ bound with $\varrho = \rho$. This is constructed by restricting g to the sub-space of functions

$$\mathcal{H}_\rho = \{\boldsymbol{\phi}^{\text{T}}(\boldsymbol{x}) D\Phi^{\text{T}} I(\rho)\boldsymbol{y} \mid D \text{ is a diagonal matrix}\}.$$

Let $\boldsymbol{y}$ be partitioned into $\boldsymbol{y}_T$ and $\boldsymbol{y}_S$ for observations at X_T and X_S. Under $\mathcal{H}_\rho$, the data vector $\boldsymbol{y}$ is linearly transformed into $(\boldsymbol{y}_T, \rho\boldsymbol{y}_S)^{\mathrm{T}}$ before applying a function from $\mathcal{H}_1$. As noted previously, the choice of $\mathcal{H}_1$ is motivated by Projected Bayes Regression. We now provide a statistical understanding to the transformation of $\boldsymbol{y}$, specifically that of the sub-vector $\boldsymbol{y}_S$. Consider the multi-task data model $\mathcal{M}_{\text{mt}}$ given by (3.29). Let $\boldsymbol{y}_T^S$ be the vector of noisy observations for task T at X_S. Then random vector $(\boldsymbol{y}_T^S, \boldsymbol{y}_S)^{\mathrm{T}}$ has zero mean and covariance

$$\mathbb{C}\left(\begin{pmatrix}\boldsymbol{y}_T^S\\ \boldsymbol{y}_S\end{pmatrix}\right) = \begin{pmatrix} K^{\mathrm{x}}_{SS} + \sigma_{\mathrm{n}}^2 I & \rho K^{\mathrm{x}}_{SS} \\ \rho K^{\mathrm{x}}_{SS} & K^{\mathrm{x}}_{SS} + \sigma_{\mathrm{n}}^2 I \end{pmatrix}.$$

Since our observations are $\boldsymbol{y}_S$, we condition $\boldsymbol{y}_T^S$ on $\boldsymbol{y}_S$ to obtain the predictive mean

$$\langle \boldsymbol{y}_T^S \mid \boldsymbol{y}_S \rangle = \rho K^{\mathrm{x}}_{SS}(K^{\mathrm{x}}_{SS} + \sigma_{\mathrm{n}}^2 I)^{-1}\boldsymbol{y}_S. \tag{3.36}$$

In the limit of negligible noise, i.e., $\sigma_{\mathrm{n}}^2 \to 0$, we may approximate the right of the equation by $\rho\boldsymbol{y}_S$. Thus, ρ is the approximate common slope of the regression line from each element in $\boldsymbol{y}_T^S$ on the corresponding element in $\boldsymbol{y}_S$ [Rodgers and Nicewander, 1988, §3]. Therefore, the transformation $I(\rho)$ in $\mathcal{H}_\rho$ converts the data vector $\boldsymbol{y}$, which is generated by both tasks T and S, to the data vector $(\boldsymbol{y}_T, \rho\boldsymbol{y}_S)^{\mathrm{T}}$, which is approximately generated by only the primary task T.

Comparison with the FWO$_{1/\rho}$ bound The following proposition is useful for excluding the FWO$_{1/\rho}$ bound from further analysis.

Proposition 3.18. *Then, for all ρ, σ_{n}^2, π_S and n, the FWO$_\rho$ bound is not more than the FWO$_{1/\rho}$ bound, i.e. the FWO$_\rho$ bound is as least as tight as the FWO$_{1/\rho}$ bound.*

Proof sketch. Using Proposition 3.16, a sufficient condition for the current proposition to hold is

$$\frac{\alpha(1/\rho^2)}{[\alpha(1)]^2} c_i(1/\rho) \geqslant \frac{\alpha(\rho^2)}{[\alpha(\rho^2)]^2} c_i(\rho),$$

where $c_i(\varrho)$ and $\alpha(\cdot)$ are as defined in Proposition 3.16. This condition is proved in appendix B.5.5. □

Although the FWO$_{1/\rho}$ is not as useful as FWO$_\rho$ for bounding the learning curve of task T from above, we emphasize that the FWO$_{1/\rho}$ bound is still useful for intuitive comparison with the single-task FWO bound, via the equivalent noise construction described in section 3.5.2.1.

Comparison with trivial single-task FWO upper bound, FWO$_0$ We now compare the FWO$_\rho$ bound with the FWO$_0$ bound, which is also the FWO bound on the single-task learning curve $\epsilon_T^{\text{avg}}(0, \sigma_{\mathrm{n}}^2, \pi_S, n)$. We wish to learn when the FWO$_\rho$ bound is tighter than the FWO$_0$ bound. For a stationary covariance function k^{x}, the following condition is sufficient for the FWO$_\rho$ bound to be tighter:

$$n < \frac{\sum_{i=2}^{\infty}\kappa_i + \sigma_{\mathrm{n}}^2}{\kappa_1} \frac{1 - \pi_S + \pi_S\rho^2}{(1-\rho^2)(1-\pi_S)\pi_S}, \tag{3.37}$$

where, we recall, κ_1 is the largest eigenvalue of k^{x} under $p(\boldsymbol{x})$. This expression is derived in appendix B.5.6. Consider the second factor on the right of the inequality. The higher the value of this factor, the larger the range of n over which the FWO$_\rho$ bound is guaranteed to be a tighter bound. Figure

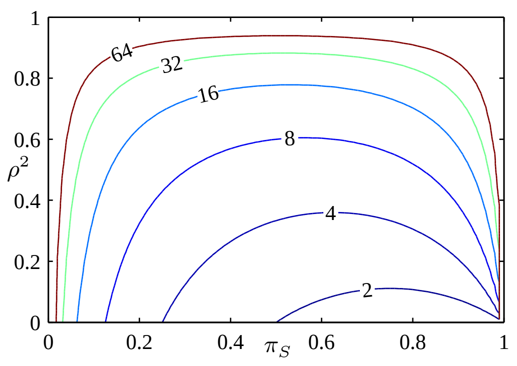


Figure 3.4: Contour plot of the factor $(1-\pi_S+\pi_S\rho^2)/(1-\rho^2)(1-\pi_S)\pi_S$ in (3.37). The higher the value of this factor, the larger the range of n over which the FWO$_\rho$ bound is surely tighter than FWO$_0$.

3.4 provides a contour plot of this factor. From the plot, we can deduce that a higher value of ρ^2 and/or a more extreme value of π_S give a tighter FWO$_\rho$ over a larger range of n.

Now, consider the first factor on the right of (3.37). Dividing the numerator and denominator by $\sum_{i=1}^{\infty}\kappa_i$, which is the variance of the noise-free stationary process, we see that this factor decreases with the signal-to-noise ratio $\sum_{i=1}^{\infty}\kappa_i/\sigma_{\mathrm{n}}^2$. This first factor also clearly depends on the ratio $\sum_{i=2}^{\infty}\kappa_i/\kappa_1$, so it is worthwhile to investigate this relation. Without any loss of generality, we limit attention to the case of unit variance for the noise-free Gaussian process, so that $\sum_{i=1}^{\infty}\kappa_i=1$, $\kappa_1<1$ and the ratio can be expressed as $1/\kappa_1-1$. For a Gaussian process on a d-dimensional input space, let subscript (i) denotes the i^{th} dimension, and use $\bar{\kappa}$ for the largest process eigenvalue. If $k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x}')=\prod_i k^{\mathrm{x}}_{(i)}(x_{(i)},x'_{(i)})$, then $\bar{\kappa}=\prod_i\bar{\kappa}_{(i)}$, where $\bar{\kappa}_{(i)}<1$ for all i. Thus the largest process eigenvalue $\bar{\kappa}$ decays with increasing dimension d, so that $\sum_{i=2}^{\infty}\kappa_i/\kappa_1$ grows with d.

Summarizing the analysis of the preceding two paragraphs, we can say that the FWO$_\rho$ bound is surely tighter than the FWO$_0$ bound over a larger range of n for higher $|\rho|$, more extreme π_S, higher dimension d, and lower signal-to-noise ratio $\sum_{i=1}^{\infty}\kappa_i/\sigma_{\mathrm{n}}^2$.

3.5.2.3 Optimal FWO$_\varrho$ Upper Bounds: the FWO$_{\varrho^\star}$ and FWO$_{\hat{\varrho}}$ Bounds

The comparison that concludes the preceding section also has this interpretation: FWO$_\rho$ is the tighter bound for small n, and FWO$_0$ is the tighter bound for large n. We now give a FWO$_\varrho$-type bound that transits naturally from the FWO$_\rho$ bound to the FWO$_0$ bound as n increases. We begin by differentiating the FWO$_\varrho$ bound (equation 3.33 on page 67) with respect to ϱ and setting this derivative to zero. Thus, the best ϱ is $\varrho^\star$ such that the gradient

$$\frac{\mathrm{d}}{\mathrm{d}\varrho}\left\{-\frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)}\sum_{i=1}^{\infty}\frac{\kappa_i^2}{c_i(\varrho)}\right\}=2\pi(1-\pi)\frac{\alpha(\rho\varrho)}{[\alpha(\varrho^2)]^2}\sum_{i=1}^{\infty}\frac{\kappa_i^2}{[c_i(\varrho)]^2}\tilde{c}_i(\varrho) \tag{3.38}$$

is zero at $\varrho=\varrho^\star$, where

$$\tilde{c}_i(\varrho)\overset{\text{def}}{=}\pi\varrho(1-\rho^2)n\kappa_i+(\varrho-\rho)\left(\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}+\sigma_{\mathrm{n}}^2-\kappa_i\right). \tag{3.39}$$

An analytical solution for $\varrho^\star$ seems implausible due to the coupling between ϱ and the eigenvalues κ_is. Nevertheless we may perform a computation search for $\varrho^\star$ for every n, using the gradient (3.38).

However, we prefer to avoid the search for every n, and for this we shall seek a solution that captures the desired behaviour qualitatively.

First, observe that the derivative is a continuous function of ϱ. Moreover, for $\rho > 0$, it is clear that $\tilde{c}_i(0) < 0$ and $\tilde{c}_i(\rho) > 0$,[8] so that the gradient for the FWO$_\varrho$ bound is negative at $\varrho = 0$ and positive at $\varrho = \rho$. The converse holds for $\rho < 0$. Therefore, the optimal value $\varrho^\star$ is between 0 and ρ where the gradient is zero; it is also clear that this optimal value depends on n. If we may approximate $\varrho^\star$ with a value between 0 and ρ, the value may as well be obtained by setting $\tilde{c}_1(\varrho) = 0$, giving

$$\hat{\varrho} \stackrel{\text{def}}{=} \frac{\rho}{1 + \pi_S(1-\rho^2)n\left\{\left(\int k^{\text{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_1(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + \sigma_{\text{n}}^2\right)/\kappa_1 - 1\right\}^{-1}}. \tag{3.40}$$

The choice of $\tilde{c}_1(\varrho) = 0$ is to optimize for the contribution by the largest eigenvalue to the FWO$_\varrho$ bound. Using $\varrho = \hat{\varrho}$, we have a natural transition from FWO$_\rho$ to FWO$_0$ as n increases, since $\hat{\varrho} = \rho$ when $n = 0$, and $\hat{\varrho} = 0$ when $n \to \infty$. For a stationary covariance function k^{x}, the expression within the curly brackets in (3.40) can be written as $(\sum_{i=2}^{\infty} \kappa_i + \sigma_{\text{n}}^2)/\kappa_1$, which we have encountered previously in (3.37), and the higher the value of expression, the higher the value of $\hat{\rho}$. By arguments similar to those for (3.37), $\hat{\rho}$ grows with the dimension of the input space and with lower signal-to-noise ratio $\sum_{i=1}^{\infty} \kappa_i/\sigma_{\text{n}}^2$.

The gradient argument can be modified straightforwardly into an alternative proof for Proposition 3.18, by observing, for positive ρ, that $\rho \leqslant 1/\rho$ and that the gradient (3.38) is positive for $\varrho \geqslant \rho$.

3.5.3 OV-type and FWO-type Bounds with No Training Data for the Primary Task

We now return to the case when there is no training data for the primary task, i.e., when $X_T = \emptyset$ and $\pi_S = 1$. According to Corollary 3.13,

$$\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 1, n) = \rho^2 \epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 1, n) + (1-\rho^2)\sum_{i=1}^{\infty} \kappa_i. \tag{3.41}$$

Notice that $\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 1, n)$ is the learning curve for a single-task GP. Hence, when no training data is available for the primary task, any bound on the single-task GP may be used for $\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 1, n)$ in the right of (3.41) to obtain a bound for the asymmetric two-tasks case. In particular, when the single-task OV lower bound is used, we recover the OV$_\rho$ bound (Proposition 3.14) with $\pi_S = 1$. Similarly, when the single-task FWO upper bound is used, we recover the FWO$_\varrho$ bound (Proposition 3.16) with $\pi_S = 1$. In the sense that no extra slack is involved in the recovery, we say that the OV$_\rho$ and FWO$_\varrho$ bounds obey Corollary 3.13, and that they are tight at $\pi_S = 1$.

3.5.4 Reversing Upper and Lower Bounds using Symmetric Multi-task Curves

Any bound for the asymmetric two-tasks case may be reversed by making use of the relation given by equation 3.10 between the symmetric and asymmetric scenarios. Denote the lower bounds on the learning curve for the symmetric and asymmetric case by $\underline{\epsilon}^{\text{avg}}$ and $\underline{\epsilon}_T^{\text{avg}}$, and the respective upper bounds

[8] This observation depends on $\int k^{\text{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \geqslant \kappa_i$, which is proved in appendix B.5.5.

by $\bar{\epsilon}^{\mathrm{avg}}$ and $\bar{\epsilon}_T^{\mathrm{avg}}$. To obtain the $\underline{\epsilon}^{\mathrm{avg}}$ and $\bar{\epsilon}^{\mathrm{avg}}$, the eigenvalues (3.16) for the symmetric two-tasks case are used in, say, the single-task OV and FWO bounds. Then, using equation 3.10, we have

$$\begin{aligned}
\underline{\epsilon}^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) &\leqslant \pi_S \epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, 1-\pi_S, n) + (1-\pi_S)\epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) \\
&\leqslant \pi_S \bar{\epsilon}_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, 1-\pi_S, n) + (1-\pi_S)\epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) \qquad (3.42) \\
\Longrightarrow \epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) &\geqslant (1-\pi_S)^{-1}\left[\underline{\epsilon}^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) - \pi_S \bar{\epsilon}_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, 1-\pi_S, n)\right]; \qquad (3.43)
\end{aligned}$$

and similarly,

$$\epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) \leqslant (1-\pi_S)^{-1}\left[\bar{\epsilon}^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) - \pi_S \underline{\epsilon}_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, 1-\pi_S, n)\right]. \qquad (3.44)$$

We will not consider these reversed bounds any further since we find them to be extremely loose in the simulations.

3.5.5 Comparing Bounds to Simulations of the Learning Curve

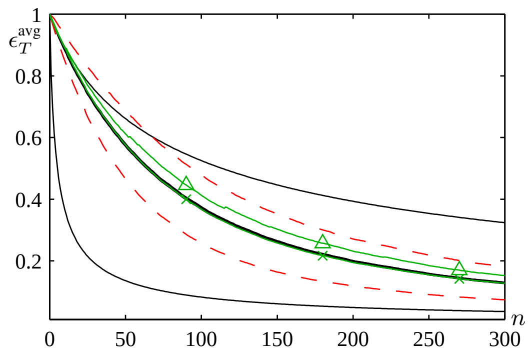
We compare our bounds with simulated learning curves. We follow two setups from Sollich and Halees [2002], namely their third scenario and Figure 6 therefor, and their first scenario and Figure 3(top) therefor. For the first setup, the input space is one dimensional with Gaussian distribution $\mathcal{N}(0, 1/12)$, and the covariance function is the unit variance squared exponential (SE) $k^{\mathrm{x}}(x, x') = \exp[-(x-x')^2/(2l^2)]$. For the second setup, the input space is one dimensional with uniform density on the unit interval $[0, 1]$, and the covariance function is that of the unit variance stationary Ornstein-Uhlenbeck process (OU) $k^{\mathrm{x}}(x, x') = \exp[-|x-x'|/l]$. In either case, the length-scale is $l = 0.01$, the observation noise variance is $\sigma_{\mathrm{n}}^2 = 0.05$, and the learning curves are computed for up to $n = 300$ training data points. The length-scale is chosen by Sollich and Halees so that the learning curves decay reasonably for $n = 1 \ldots 300$.[9] When required, the average over $\boldsymbol{x}_*$ is computed analytically, as per section 3.3.4. The empirical average over $X \stackrel{\mathrm{def}}{=} X_T \cup X_S$, denoted by $\langle\!\langle \cdot \rangle\!\rangle$, is computed over 100 randomly sampled training sets. The process eigenvalues κ_is needed to compute the theoretical bounds for both the SE and OU covariance functions[10] are given in section 3.2.3. Appendix B.7 gives additional details on the simulations. These two setups will be used in later sections again.

Learning curves for pairwise combinations of $\rho^2 \in \{1/8, 1/4, 1/2, 3/4\}$ and $\pi_S \in \{1/4, 1/2, 3/4\}$ are computed. We compare the following curves:

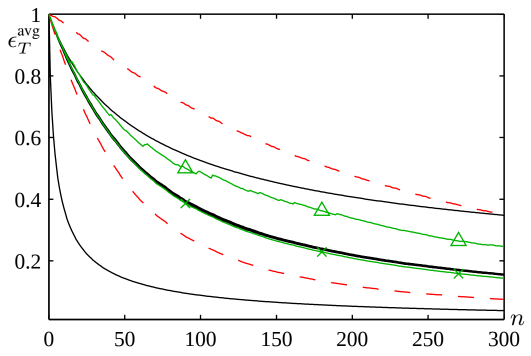
1. the "true" multi-task learning curve $\langle\!\langle \epsilon_T \rangle\!\rangle$ obtained by averaging $\sigma_T^2(\rho, \pi_S)$ over $\boldsymbol{x}_*$ and X;
2. the theoretical bounds OV_ρ and $\mathrm{FWO}_{\hat{\varrho}}$ from sections 3.5.1 and 3.5.2.3;

[9] The expected number of zero crossings (up or down) per unit interval for a stationary Gaussian process with covariance function $k(x, x') = k(x - x')$ is given by $\sqrt{-k''(0)/k(0)}/\pi$ [Itô, 1964; Ylvisaker, 1965]. For the SE covariance function with $l = 0.01$, this is $1/l\pi \approx 32$. With 32 expected zero crossings per interval, the Gaussian process prior favours "wriggly" functions, so that a rather large number of samples are required to learn the true function. The decay of the learning curve is therefore not too rapid for $n = 1 \ldots 300$, and we mostly stay in the "interesting" region of the learning curve.

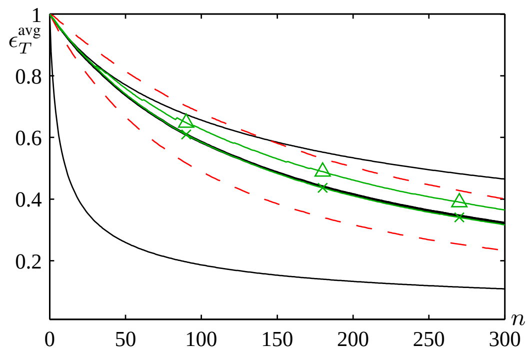
[10] In fact, Sollich and Halees [2002] use a periodic modification of the OU covariance function in order to obtain the process eigenvalues by Fourier transformation. We, however, prefer and choose to use the OU covariance function as it is, since this is what is typically used in practice. Nevertheless, Sollich and Halees [2002] explain that one can expect similar learning curves in either case for the choice of input domain and process length-scale. Appendix B.6.2 shows that the asymptotic eigenvalues of the periodic OU covariance differs from those of the conventional by a constant multiplicative factor of 4.



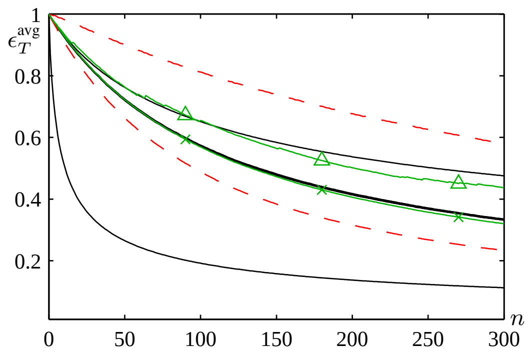
(a) $k^{\text{x}} = \text{SE}, \rho^2 = 1/2, \pi_S = 1/2$



(b) $k^{\text{x}} = \text{SE}, \rho^2 = 3/4, \pi_S = 3/4$



(c) $k^{\text{x}} = \text{OU}, \rho^2 = 1/2, \pi_S = 1/2$



(d) $k^{\text{x}} = \text{OU}, \rho^2 = 3/4, \pi_S = 3/4$

Figure 3.5: Comparison of various bounds for two settings of (ρ, π_S) with the SE and OU covariance functions. Each graph plots ϵ_T^{avg} against n and consists of the multi-task learning curve (middle —), the theoretical bounds OV$_\rho$ and FWO$_{\hat{\varrho}}$ using Propositions 3.14/3.16 (lower/upper —), the experimental trivial lower/upper bounds using Corollary 3.12 (lower/upper - -), and the experimental lower/upper bounds using Corollaries 3.6a/3.10 (✳ / △). The reversed bounds are omitted because they are rather loose for the given (ρ^2, π_S) values. The thickness of the multi-task curve reflects 95% confidence interval.

3. the single-task learning curves $\langle\!\langle \epsilon_T(0) \rangle\!\rangle$ and $\langle\!\langle \epsilon_T(1) \rangle\!\rangle$ — obtained by averaging $\sigma_T^2(0)$ and $\sigma_T^2(1)$ — that trivially bound the multi-task learning curve from above and below;

4. the empirical or experimental lower bound $\langle\!\langle \underline{\epsilon_T} \rangle\!\rangle$ and upper bound $\langle\!\langle \bar{\epsilon}_T \rangle\!\rangle$ obtained by averaging Corollaries 3.6a and 3.10 over X; and

We do not include the theoretical upper bounds FWO$_{1/\rho}$ and FWO$_\rho$ since they are both looser than FWO$_{\hat{\varrho}}$. Figure 3.5 provides some indicative plots of the curves.

We summarize with the following observations that apply to both the SE and OU covariance functions: (a) The gap between the trivial bounds $\langle\!\langle \epsilon_T(0) \rangle\!\rangle$ and $\langle\!\langle \epsilon_T(1) \rangle\!\rangle$ increases with π_S, as described at the start of section 3.5. (b) We find the lower bound $\langle\!\langle \underline{\epsilon_T}(\rho) \rangle\!\rangle$ a rather close approximation to the "true" multi-task learning curve $\langle\!\langle \epsilon_T(\rho) \rangle\!\rangle$, as evidenced by the overlap between the ✳ lines and the middle — lines in Figure 3.5. (c) The curve for the experimental upper bound $\langle\!\langle \bar{\epsilon}_T(\rho) \rangle\!\rangle$ using the equivalent noise method has jumps, e.g., the △ lines in Figure 3.5, because the equivalent noise variance $\bar{\bar{\sigma}}_{\text{n}}^2$ increases whenever a datum for X_S is sampled. (d) The theoretical FWO$_{\hat{\rho}}$ upper bound is even tighter than the

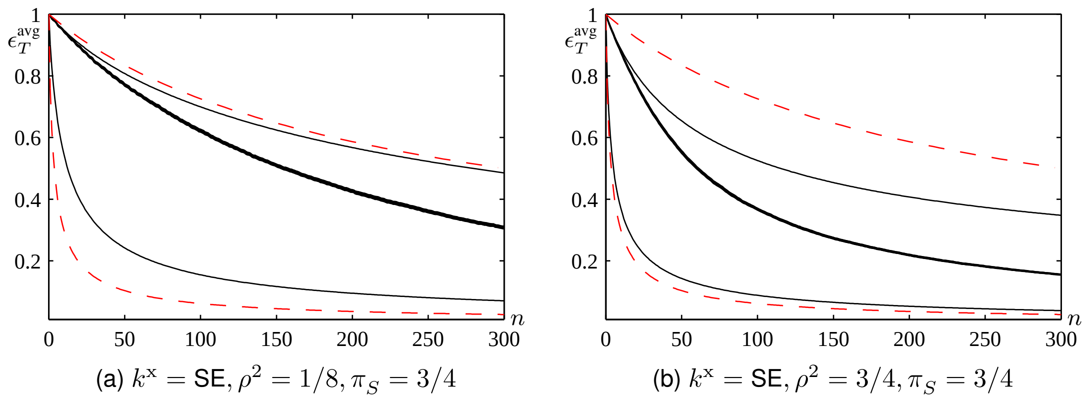


Figure 3.6: Comparison to the trivial bounds for two settings of (ρ, π_S) with the SE covariance function. Each graph plots ϵ_T^{avg} against n and consists of the "true" multi-task learning curve (middle **—**), the theoretical lower/upper bounds using Propositions 3.14/3.16 (lower/upper —), and the OV and FWO single-task bounds for $\rho = 1$ and $\rho = 0$ (lower/upper - -). The thickness of the "true" multi-task learning curve reflects 95% confidence interval. The graphs for the OU covariance function are similar except that the gaps between the lower/upper — and lower/upper - - lines are larger; see point (e) in text.

experimental trivial upper bound $\langle\!\langle \epsilon_T(0) \rangle\!\rangle$ for small n; however, we find the theoretical OV$_\rho$ lower bound to be consistently looser than the experimental trivial lower bound $\langle\!\langle \epsilon_T(1) \rangle\!\rangle$. (e) For small n, $\langle\!\langle \epsilon_T(\rho) \rangle\!\rangle$ is closer to FWO$_{\hat{\rho}}$, but becomes closer to OV$_\rho$ as n increases, as shown by the unmarked solid lines in Figure 3.5. This is because the theoretical lower bound OV$_\rho$ is based on the asymptotically exact single-task OV bound and the $\underline{\epsilon}_T(\rho)$ bound, which has been observed to approximate the multi-task learning curve $\langle\!\langle \epsilon_T \rangle\!\rangle$ rather closely; see point (b).

Comparison with trivial theoretical bounds Corollary 3.12 states that the multi-task learning curve $\langle\!\langle \epsilon_T \rangle\!\rangle$ is bounded from below and above by the $\rho = 1$ and $\rho = 0$ single-task learning curves. We compare the single-task OV lower bound for $\rho = 1$ against OV$_\rho$, and the single-task FWO upper bound for $\rho = 0$ against FWO$_{\hat{\varrho}}$. Two examples are given in Figure 3.6. In summary, we find that (a) the asymmetric two-tasks bounds OV$_\rho$ and FWO$_{\hat{\varrho}}$ are always tighter than the trivial single-task OV and FWO bounds, as illustrated in Figure 3.6 where the - - lines envelope the — lines; (b) the lower the value of π_S, the more similar the single-task bounds are to their respective two-tasks bounds. This is intuitive since the smaller the training set for task S, the lesser the effect of multi-task learning for any ρ; (c) the relative tightness of the OV$_\rho$ bound is more noticeable for smaller ρ^2, in agreement with the discussion in section 3.5.1; (d) the trivial single-task FWO bound and the FWO$_{\hat{\varrho}}$ bound converges for large n, since $\hat{\varrho} \to 0$ as $n \to \infty$; and (e) the difference between the single-task bounds and the two-tasks bounds is more noticeable for the OU covariance function than for the SE covariance function. This agrees with analyses in sections 3.5.1 and 3.5.2.3, since a process with a SE covariance function is smoother than one with an OU covariance function, and the largest eigenvalue of the SE covariance function, $\bar{\kappa} = 0.034$, is larger than that of the OU covariance function, $\bar{\kappa} = 0.020$.

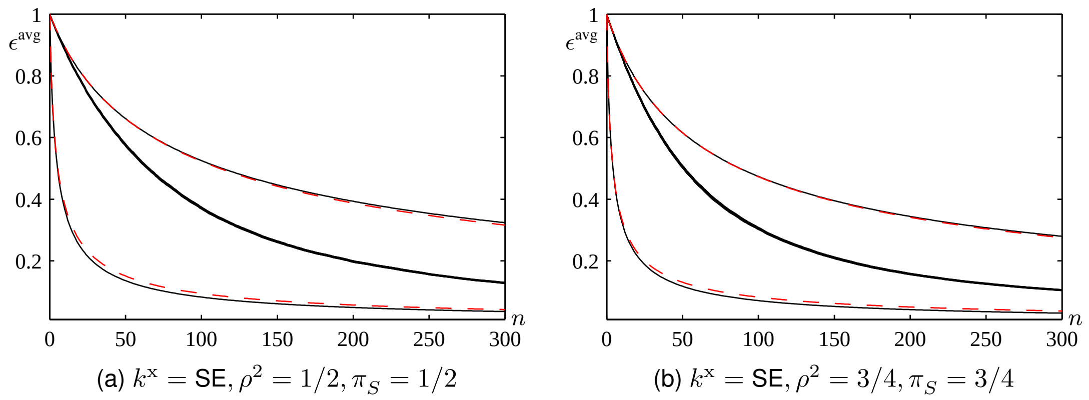


Figure 3.7: Bounds on learning curves in the symmetric multi-task case, for two settings of (ρ, π_S) with the SE covariance function. Each graph plots ϵ^{avg} against n and consists of the "true" multi-task learning curve (middle —), the theoretical lower/upper bounds from bounds in the asymmetric case using equation 3.45 (lower/upper —), and the lower/upper bounds using the analytical eigenvalues (lower/upper - -) with the single-task OV and FWO bounds. The thickness of the "true" multi-task learning curve reflects 95% confidence interval. The graphs for the OU covariance function are qualitatively similar.

Symmetric case For completeness, we also compute and compare the bounds on the symmetric two-tasks case. We compare the theoretical OV lower bound $\underline{\epsilon}^{\text{avg}}$ and FWO upper bound $\bar{\epsilon}^{\text{avg}}$ using the eigenvalues given by equation 3.16 to the lower and upper bounds obtained using equation 3.10

$$\epsilon^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \geqslant \pi_S \underline{\epsilon}_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 1-\pi_S, n) + (1-\pi_S)\underline{\epsilon}_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \tag{3.45a}$$

$$\epsilon^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \leqslant \pi_S \bar{\epsilon}_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 1-\pi_S, n) + (1-\pi_S)\bar{\epsilon}_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n) \tag{3.45b}$$

that involve pairs of asymmetric two-tasks bounds. For $\underline{\epsilon}_T^{\text{avg}}$ and $\bar{\epsilon}_T^{\text{avg}}$ we use OV_ρ and $\text{FWO}_{\hat{\varrho}}$ and denote the corresponding symmetric two-tasks bounds by $\underline{\epsilon}_{\text{OV}}^{\text{avg}}$ and $\bar{\epsilon}_{\text{FWO}}^{\text{avg}}$. Figure 3.7 gives plots for two chosen values of (ρ^2, π_S) and the SE covariance function. We find $\underline{\epsilon}^{\text{avg}}$ (resp. $\bar{\epsilon}^{\text{avg}}$) to be very similar curves to $\underline{\epsilon}_{\text{OV}}^{\text{avg}}$ (resp. $\bar{\epsilon}_{\text{FWO}}^{\text{avg}}$), with the former slightly tighter. This simulation result suggests that our asymmetric two-tasks bounds are as tight as can be achieved using OV-type and FWO-type bounds.

3.6 The Effects of ρ^2 and π_S on Multi-task Learning

For the multi-task model described in section 3.2.1, multi-task learning is characterized by two parameters: (1) the degree of "relatedness" ρ between the two tasks, and (2) the ratio π_S of total training data for the secondary task. This section will look at how ρ^2 and π_S affect multi-task learning; in particular, we will see how they interact with the smoothness of the function that is to be learnt. We measure the effects on multi-task learning with the ratio of the learning curve for multi-task GP learning to that for an appropriate baseline single-task case. By approximating this ratio, we obtain analytic expressions so as to further our insights into multi-task learning. The approximation replaces the true learning curve with the asymptotic orders of the OV lower bound for multi-task learning, which are derived directly from those for the single-task case.

We shall focus on and give examples using the squared exponential (SE) covariance function and the covariance function of the stationary Ornstein-Uhlenbeck process (OU), both on a one-dimensional input space. These two are at the extreme ends of smoothness of random functions — the SE covariance function gives an infinitely mean square differentiable function while the OU process is not mean square differentiable — and will serve to bring out how smoothness bears on multi-task learning. One conclusion in this section is that multi-task learning, whether symmetric or asymmetric, generally has more influence over the learning of a smooth process than a rough process.

For a Gaussian process with covariance function k^{x} under a given probability measure, we consider the following decays of the process eigenvalues κ_is of k^{x}:

$$\kappa_i \sim \begin{cases} \eta_0 (bi)^{-r} & \text{for power-law decay} \\ \eta_0 b^{-i} & \text{for exponential decay,} \end{cases} \tag{3.46}$$

for some constants $\eta_0 > 0$, and $b > 0$ (for power-law decay) or $b > 1$ (for exponential decay). For example, the OU process with length-scale l has eigenvalues decaying with power-law with $r = 2$, $b = \pi$ and $\eta_0 = 2/l$; and the SE covariance function in section 3.5.5 has exponentially decaying eigenvalues with $\eta_0 = 0.034$, which is obtained by using $\sigma_x^2 = 1/12$ and $l = 0.01$ in equation 3.15 on page 57. To avoid distraction from our present purpose, we postpone the discussion on the generality of equation 3.46 to appendix B.8.1. It suffices to bear in mind that a Gaussian process with faster decaying eigenvalues is smoother in the mean-square and almost-sure sense; see section 3.2.3.

We now state the asymptotic orders for the OV bound on the learning curve of single-task GP learning. It is shown in appendix B.8 that the OV lower bound on the learning curve for the single-task case at n noisy observations is

$$\sigma_{\mathrm{n}}^2 \sum_{i=1}^{\infty} \frac{\kappa_i}{\sigma_{\mathrm{n}}^2 + n\kappa_i} \in \begin{cases} \Theta\left((\eta_0 \tilde{n})^{1/r}/\tilde{n}\right) & \text{for power-law decay} \\ \Theta\left(\log(1 + \eta_0 \tilde{n})/\tilde{n}\right) & \text{for exponential decay,} \end{cases} \tag{3.47}$$

where $\tilde{n} \overset{\text{def}}{=} n/\sigma_{\mathrm{n}}^2$ and σ_{n}^2 is the common noise variance of the observations. That n and σ_{n}^2 occur in equation 3.47 only through $\tilde{n}$ shows the duality between n and σ_{n}^2: large sample size is the same as small noise variance within the asymptotics of the OV lower bound. In additional, since the sum of the eigenvalues is the average prior variance of a Gaussian process, the prior variance of the process enters the order relations through η_0, where higher prior variance gives higher η_0. The constants implicit within the Θ operators are given in appendix B.8; for our current purpose, it suffices to treat the above order relations as equalities.[11]

Bibliographic note The two asymptotic orders (3.47) agree with the experimental results of Williams and Vivarelli [2000, §6.2]. The order for the case of eigenvalues decaying with power-law agrees with the order for the *lower continuous* approximation to the true learning curve [Sollich and Halees, 2002, §2], and with the order for the optimal error (see section 3.4), while the order for the case of exponentially decaying eigenvalues has been stated by Opper and Vivarelli [1999]. The merit of (3.47) over the above cited literature is that it is more exact: it also includes dependence on σ_{n}^2 and η_0. This exactness is required for analyzing multi-task learning with the SE covariance function in the rest of this section.

[11] It is possible to state the conditions under which the constants cancel out when considering the ratios of the learning curves. This is not done at present in order not to obscure the main subject of this section.

3.6.1 Symmetric Multi-task Learning and Error Inflation Factors

Symmetric multi-task learning may be compared with single-task learning using the ratio, at n training samples, of the learning curve of multi-task learning to the learning curve of single-task learning:

$$\mathrm{IF}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) \overset{\mathrm{def}}{=} \frac{\epsilon^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n)}{\epsilon^{\mathrm{avg}}(1, \sigma_{\mathrm{n}}^2, 1, n)}. \tag{3.48}$$

This ratio is to the single-task learning when $\rho = 1$, i.e., when task T and task S are indistinguishable. The ratio is certainly greater than one, so it is an *inflation factor* that measures the increase in error when tasks T and S are other than perfectly correlated.

An approximation replaces each true learning curve value in (3.48) with the asymptotics of the OV lower bound on that value. In the symmetric multi-task case, the asymptotics can be derived using the process eigenvalues $(1/2 \pm \sqrt{1/4 - \omega})\kappa_i$, where $\omega \overset{\mathrm{def}}{=} \pi_S(1 - \pi_S)(1 - \rho^2)$ (see section 3.2.3), which leads to the following order relations for the Ornstein-Uhlenbeck process (OU) and the squared exponential (SE) covariance function on one-dimensional input spaces.

$$\text{Ornstein-Uhlenbeck (OU)}: \quad \Theta\left(\frac{1}{\tilde{n}}(\eta_0 \tilde{n})^{1/2}\sqrt{1 + 2\sqrt{\omega}}\right) \tag{3.49}$$

$$\text{Squared Exponential (SE)}: \quad \Theta\left(\frac{1}{\tilde{n}} \log\left[1 + \eta_0 \tilde{n} + \omega(\eta_0 \tilde{n})^2\right]\right). \tag{3.50}$$

The derivations are in appendix B.8.3. Using these order relations in place of the true learning curves in (3.48) gives approximate inflation factors for the OU and SE covariance functions:

$$\widehat{\mathrm{IF}}^{\mathrm{OU}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) = \sqrt{1 + 2\sqrt{\omega}} \tag{3.51a}$$

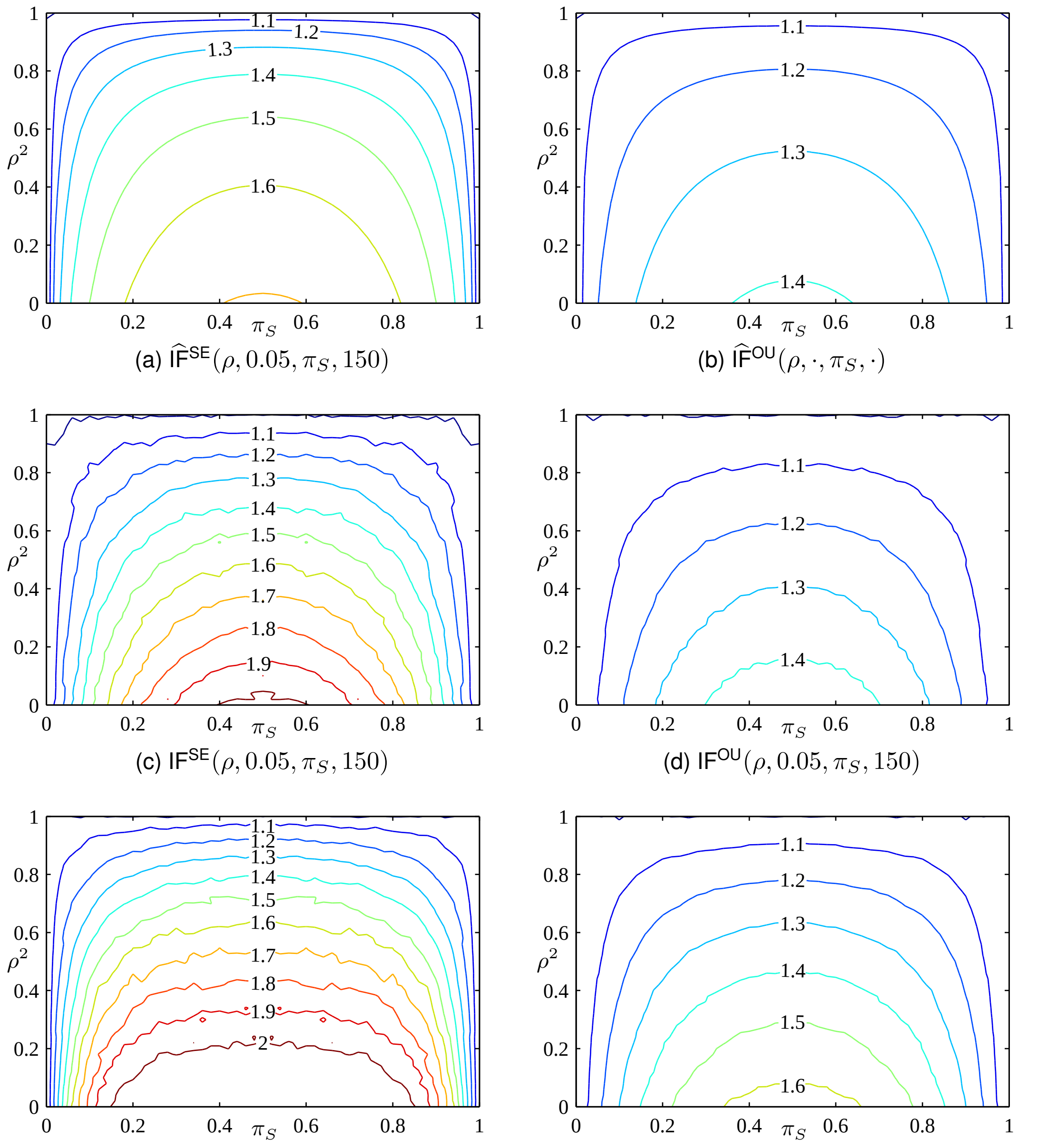
$$\widehat{\mathrm{IF}}^{\mathrm{SE}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) = \frac{\log\left[1 + \eta_0 \tilde{n} + \omega(\eta_0 \tilde{n})^2\right]}{\log\left[1 + \eta_0 \tilde{n}\right]} \approx 1 + \frac{\log\left[1 + \omega \eta_0 \tilde{n}\right]}{\log\left[1 + \eta_0 \tilde{n}\right]}, \tag{3.51b}$$

where the approximation on $\widehat{\mathrm{IF}}^{\mathrm{SE}}$ is for large $\eta_0 \tilde{n}$.

Figure 3.8 on the next page uses the setup in section 3.5.5 and plots the contours of the inflation factors (and their approximations) for the OU and SE covariance functions. Comparing Figure 3.8a to Figure 3.8c for the SE covariance function, and Figure 3.8b to Figure 3.8d and Figure 3.8f for the OU covariance function, we see that the contours for the approximations to the inflation factors agree rather well with the empirical ones, in both scale and shape. This suggests that it is worthwhile to use the approximations (3.51) to understand symmetric multi-task learning. We make the following observations.

1. The quantity ω, which also appears in the process eigenvalues, is intrinsic to symmetric multi-task learning. For a given k^{x} and $p(\boldsymbol{x})$, similar learning curves may be obtained for two multi-task settings if they have the same value for ω. For example, we may expect a multi-task setup with $(\pi_S, \varrho^2) = (0.5, 0.5)$ to have a learning curve similar to that of a setup with $(\pi_S, \varrho^2) = (0.3, 0.4)$, since the values for ω in these two setups are 0.125 and 0.126. This observation is supported empirically: referring to Figure 3.8e, which plots the empirical inflation factor contours at $n = 300$ for the SE covariance function, we see that these two multi-task setups both lie close to the contour valued 1.7.

2. The contours in Figure 3.8 are steeper for the SE covariance function than for the OU covariance function. This means that multi-task learning has more influence on the learning curve for the SE



(a) $\widehat{\mathsf{IF}}^{\mathsf{SE}}(\rho, 0.05, \pi_S, 150)$ (b) $\widehat{\mathsf{IF}}^{\mathsf{OU}}(\rho, \cdot, \pi_S, \cdot)$

(c) $\mathsf{IF}^{\mathsf{SE}}(\rho, 0.05, \pi_S, 150)$ (d) $\mathsf{IF}^{\mathsf{OU}}(\rho, 0.05, \pi_S, 150)$

(e) $\mathsf{IF}^{\mathsf{SE}}(\rho, 0.05, \pi_S, 300)$ (f) $\mathsf{IF}^{\mathsf{OU}}(\rho, 0.05, \pi_S, 300)$

Figure 3.8: Contour plots of the error inflation factors for the symmetric multi-task case at $n = 150$ and $n = 300$ (with $\sigma_n^2 = 0.05$), and using the setup in section 3.5.5. The left plots are for the SE covariance function, and the right plots are for the OU covariance function. The top plots are the approximations to the inflation factors using the asymptotics to the OV lower bound, while the middle and bottom plots are obtained using the empirical learning curves. The contour plot (b) is independent of n and σ_n^2; see (3.51a).

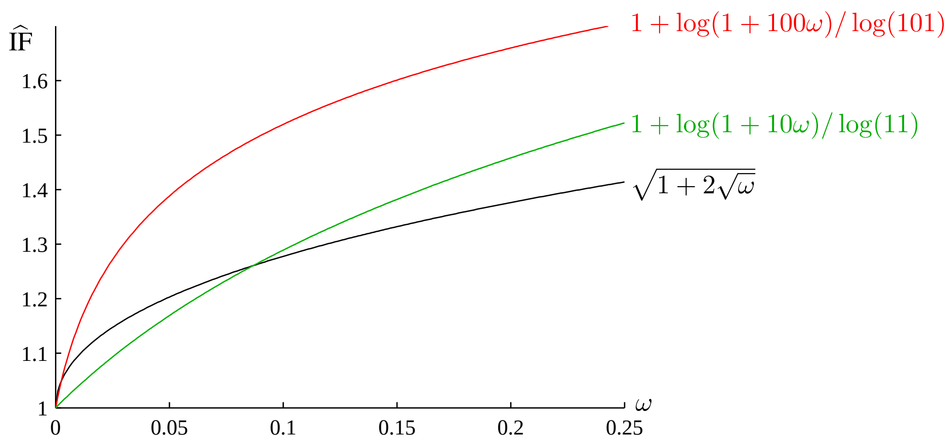


Figure 3.9: Curves summarizing the effects of $\omega \overset{\text{def}}{=} \pi_S(1-\pi_S)(1-\rho^2)$ on symmetric multi-task learning. The black line is for the OU covariance function using (3.51a), while the green and red lines are for the SE case using (3.51b) with $\eta_0\tilde{n}$ taking values 10 and 100. The red and green lines range to a larger value, showing that symmetric multi-task learning affects the GP with SE covariance more than it does the OU process.

than for the OU. Figure 3.9 plots the approximations (3.51) as functions of ω, which ranges from 0 to $1/4$. In the SE case, two functions are plotted, one with $\eta_0\tilde{n} = 10$ (green curve) and another with $\eta_0\tilde{n} = 100$ (red curve), which is approximately the value of $\eta_0\tilde{n}$ for Figure 3.8a. The figure shows that, even for moderate values of $\eta_0\tilde{n}$, the functions of ω in the SE case range to a higher value than the function in the OU case. Therefore multi-task learning has more effect in the SE case than in the OU case.

3. The parameters n, σ_{n}^2 and η_0 do not appear in (3.51a); this suggests that multi-task learning is less affected by variations in these three parameters for the OU covariance function than for the SE. This is illustrated empirically for parameter n when we compare Figure 3.8c to Figure 3.8e for the SE covariance function, and Figure 3.8d to Figure 3.8f for the OU covariance function. We observe that the contour plots for the SE case have more differences between $n = 150$ and $n = 300$ than the contour plots for the OU case. This shows that n influences multi-task learning with the SE covariance function more than it influences learning with the OU.

3.6.2 Asymmetric Multi-task Learning and Error Deflation Factors

We now look at the influence of ρ^2 and π_S on asymmetric multi-task learning. Instead of the inflation factor used the symmetric multi-task case, we consider the *deflation factor*

$$\mathrm{DF}(\rho,\sigma_{\mathrm{n}}^2,\pi_S,n) \overset{\text{def}}{=} \frac{\epsilon_T^{\text{avg}}(\rho,\sigma_{\mathrm{n}}^2,\pi_S,n)}{\epsilon_T^{\text{avg}}(0,\sigma_{\mathrm{n}}^2,\pi_S,n)} \tag{3.52}$$

that measures the reduction in error when the secondary task S is used to assist the primary task T in learning. As before, an approximation to the factor replaces each value of the true learning curve by the asymptotic value of its OV_ρ bound, which is obtained by applying the asymptotics of the single-task OV bound twice within the mixture formation of the OV_ρ bound (Proposition 3.14a). In this case, it is relatively straightforward to consider the more general class of GPs with eigenvalues decaying with an order-r power-law instead of restricting to just the Ornstein-Uhlenbeck process (OU) that has quadratic-decaying eigenvalues. The approximations to the error deflation factors for power-law decaying and exponentially decaying process eigenvalues are

$$\begin{aligned}
\widehat{\mathrm{DF}}^{(r)}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) &= \frac{\rho^2(\eta_0\tilde{n})^{1/r}/\tilde{n} + (1-\rho^2)\left(\eta_0(1-\pi_S)\tilde{n}\right)^{1/r} / ((1-\pi_S)\tilde{n})}{\left(\eta_0(1-\pi_S)\tilde{n}\right)^{1/r} / ((1-\pi_S)\tilde{n})} \\
&= 1 - \rho^2\left(1 - (1-\pi_S)^{(r-1)/r}\right) && (3.53a) \\
\widehat{\mathrm{DF}}^{\mathrm{SE}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) &= \frac{\rho^2 \log(1+\eta_0\tilde{n})/\tilde{n} + (1-\rho^2)\log\left(1+\eta_0(1-\pi_S)\tilde{n}\right) / ((1-\pi_S)\tilde{n})}{\log\left(1+\eta_0(1-\pi_S)\tilde{n}\right) / ((1-\pi_S)\tilde{n})} \\
&= 1 - \rho^2\left(1 - (1-\pi_S)\frac{\log\left(1+\eta_0\tilde{n}\right)}{\log\left(1+\eta_0(1-\pi_S)\tilde{n}\right)}\right). && (3.53b)
\end{aligned}$$

The $\widehat{\mathrm{DF}}^{\mathrm{SE}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n)$, which is for exponentially decaying eigenvalues, applies to the squared exponential covariance function (SE). Similar to the approximate error inflation factors for symmetric multi-task learning, the approximate error deflation factor for power-law decay (3.53a) does not depend on n, σ_{n}^2 and η_0, while that for exponential decay does.

Figure 3.10 uses the setup in section 3.5.5 and plots the contours of the error deflation factors (and their approximations) for the OU ($r = 2$) and the SE covariance functions. As for the symmetric multi-task case, we compare Figure 3.8a to Figure 3.8c for the SE covariance function, and Figure 3.8b to Figure 3.8d for the OU covariance function. The contours for the approximations to the deflation factors agree rather well with the empirical ones, in both scale and shape. Thus the approximations (3.53) may be further analyzed to give insights to asymmetric multi-task learning. This is the subject henceforth.

Unlike symmetric multi-task learning that seems to be intrinsically governed by $\pi_S(1-\pi_S)(1-\rho^2)$, this governing property does not seem to feature in asymmetric multi-task learning. However, if we may generalize, the approximate error deflation factors share the common form

$$\widehat{\mathrm{DF}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) = 1 - \rho^2\hat{\pi}_S, \tag{3.54}$$

where we call $\hat{\pi}_S$ the effective proportion of observations for the secondary task S. This is related to Proposition 3.14c, the third formulation for the OV_ρ lower bound, since there the b_i^0 factors equal $1 - \rho^2\pi_S$ in the limit of large n. Matching (3.53) to (3.54), the effective proportions for order-r power law decay and exponential decay of the process eigenvalues are

$$\hat{\pi}_S^{(r)} = 1 - (1-\pi_S)^{(r-1)/r} \qquad \hat{\pi}_S^{\mathrm{SE}} = 1 - (1-\pi_S)\frac{\log\left(1+\eta_0\tilde{n}\right)}{\log\left(1+\eta_0(1-\pi_S)\tilde{n}\right)}. \tag{3.55}$$

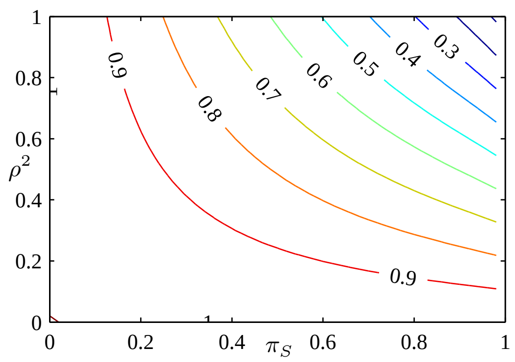
For a fixed π_S and in the limit of large r and large $\tilde{n}$

$$\lim_{r\to\infty} \hat{\pi}_S^{(r)} = \pi_S \qquad \lim_{\tilde{n}\to\infty} \hat{\pi}_S^{\mathrm{SE}} = \pi_S, \tag{3.56}$$

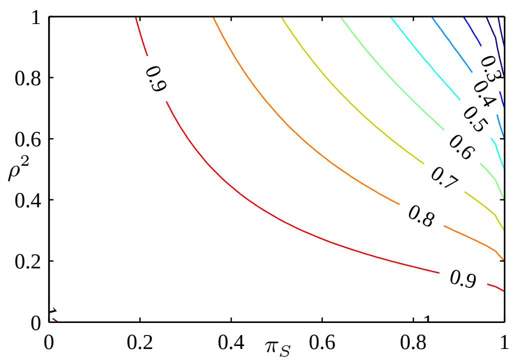
where the limit to π_S approaches from below in both cases. Thus

$$\widehat{\mathrm{DF}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n) > 1 - \rho^2\pi_S. \tag{3.57}$$

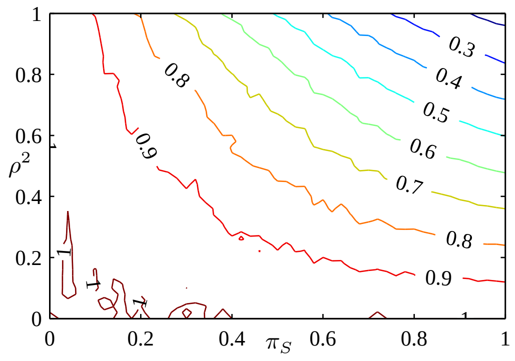
So, in terms of the asymptotics of the OV lower bound, asymmetric multi-task learning decreases the average error by a factor of at most $\rho^2\pi_S$. This is also a direct consequence of Proposition 3.14c,



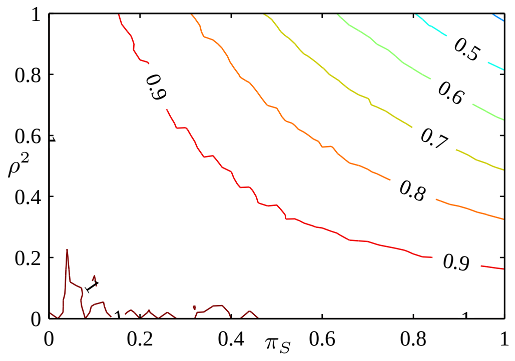
(a) $\widehat{\text{DF}}^{\text{SE}}(\rho, 0.05, \pi_S, 150)$



(b) $\widehat{\text{DF}}^{\text{OU}}(\rho, \cdot, \pi_S, \cdot)$



(c) $\text{DF}^{\text{SE}}(\rho, 0.05, \pi_S, 150)$



(d) $\text{DF}^{\text{OU}}(\rho, 0.05, \pi_S, 150)$

Figure 3.10: Contour plots of the error deflation factors for the asymmetric multi-task case at $n = 150$ (with $\sigma_{\text{n}}^2 = 0.05$) using the setup in section 3.5.5. The left plots are for the SE covariance function, and the right plots are for the OU covariance function ($r = 2$). The top plots are the approximations (3.53) to the deflation factors using the asymptotics to the OV lower bound, while the bottom plots are obtained using the empirical learning curves. The contour plot in (b) is independent of n and σ_{n}^2; see (3.53a).

since $b_i^0 \in](1-\rho^2\pi_S), 1]$ there. Based on the agreement between the contour plots in Figure 3.10, we may also use this factor to quantify the maximum reduction in the true learning curves. This will be investigated in section 3.8.4 on an empirical data set.

We investigate how the smoothness of the primary task affects multi-task learning. Observe that

$$r > r' \qquad \iff \qquad \hat{\pi}_S^{(r)} > \hat{\pi}_S^{(r')} \qquad \iff \qquad \widehat{\mathrm{DF}}^{(r)} < \widehat{\mathrm{DF}}^{(r')}. \tag{3.58}$$

So, in terms of the asymptotics of the OV lower bound, asymmetric multi-task learning benefits a smooth process more than it does a rough process. This is supported empirically by comparing the contour plot in Figure 3.10c for the SE covariance function to the contour plot in Figure 3.10d for the OU covariance function: the contours in Figure 3.10c are steeper than in Figure 3.10d, and, for the same (ρ^2, π_S), the contour value in Figure 3.10c is smaller than in Figure 3.10d.

Remark It is worthwhile to highlight that $\hat{\pi}_S^{(r)}$ is an overestimate, so $\widehat{\mathrm{DF}}^{(r)}$ is an underestimate, at least when $\pi_S = 1$. To see this, note that at $\pi_S = 1$, $\hat{\pi}_S^{(r)} = 1$, which gives $\widehat{\mathrm{DF}}^{(r)} = 1 - \rho^2$. Compare this with the exact error deflation factor that can be computed using Corollary 3.13:

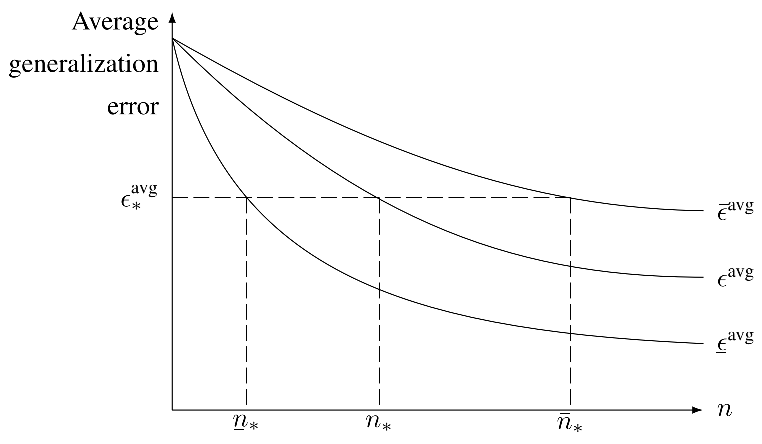
$$\mathrm{DF}^{(r)}(\rho, \sigma_{\mathrm{n}}^2, 1, n) = \frac{\rho^2 \epsilon_T^{\mathrm{avg}}(1, \sigma_{\mathrm{n}}^2, 0, n) + (1-\rho^2)\sum_{i=1}^{\infty}\kappa_i}{\sum_{i=1}^{\infty}\kappa_i} = 1 - \rho^2\left(1 - \frac{\epsilon_T^{\mathrm{avg}}(1, \sigma_{\mathrm{n}}^2, 0, n)}{\sum_{i=1}^{\infty}\kappa_i}\right). \tag{3.59}$$

The fraction within the rightmost expression is clearly smaller than one; in fact, for process eigenvalues decaying with order-r power-law, which is the case here, this fraction decays as $\Theta((\eta_0\tilde{n})^{1/r}/\tilde{n})$, if we may use the asymptotics for the single-task OV bound (3.47). Thus, at least when $\pi_S = 1$, $\hat{\pi}_S^{(r)}$ is an overestimate that becomes worse as $\tilde{n}$ increases. One reason for this is that the $\hat{\pi}_S^{(r)}$ is computed using the ratios of the asymptotics of the OV$_\rho$ bounds and discarding the possibly different constant factors within the asymptotics. In contrast, it can be shown that $\hat{\pi}_S^{\mathrm{SE}}$ obeys (3.59) qualitatively.

3.7 Effective Number of Additional Data Points for the Primary Task

The learning curve may be inverted by fixing the error to a desired value, say $\epsilon_*^{\mathrm{avg}}$, in order to obtain the required number of observations n_* to attain that fixed error. A lower bound on the learning curve provides a limit on the error beyond which the learning algorithm cannot achieve, so inverting it gives a lower bound $\underline{n}_*$ on n_*. Similarly, an upper bound $\bar{n}_*$ on n_* can be obtained using an upper bound on the learning curve in the same way. Figure 3.11 illustrates this procedure for obtaining $\underline{n}_*$ and $\bar{n}_*$. We may understand $\underline{n}_*$ as a *necessary but not sufficient sample size* to attain $\epsilon_*^{\mathrm{avg}}$ since at least $\underline{n}_*$ examples are needed. Similarly, $\bar{n}_*$ is a *sufficient but not necessary sample size* because having $\bar{n}_*$ examples should more than guarantee an error of not more than $\epsilon_*^{\mathrm{avg}}$. This application of the upper bound on the learning curve to obtain $\bar{n}_*$ has previously been noted by Williams and Vivarelli [2000].

The upper bound $\bar{n}_*$ is related to the concept of *sample complexity* in the PAC-based analysis of learning algorithms, while the lower bound $\underline{n}_*$ is related to the lower bound on the sample complexity [Kearns, 1990, chapter 6]. However, the complexity there is worse case complexity. To keep our exposition concise, we shall borrow the term *sample complexity* for $\bar{n}_*$, but keeping in mind that we mean this in the *average case* sense. Similarly, $\underline{n}_*$ is a lower bound on the (average case) sample complexity.



Average generalization error
ϵ_*^{avg}
$\bar{\epsilon}^{\text{avg}}$
ϵ^{avg}
$\underline{\epsilon}^{\text{avg}}$
n
$\underline{n}_*$
n_*
$\bar{n}_*$

Figure 3.11: Converting bounds on the learning curve to bounds on the required number of samples n_* to attain an average performance of ϵ_*^{avg}. A lower bound $\underline{\epsilon}^{\text{avg}}$ on the learning curve leads to a lower bound $\underline{n}_*$ on n_*, while an upper bound $\bar{\epsilon}^{\text{avg}}$ on the learning curve leads to an upper bound $\bar{n}_*$ on n_*.

In this section, we investigate how asymmetric multi-task learning affects n_* through its bounds. For a multi-task setup with $n(1 - \pi_S)$ observations for the primary task T, we define the factor γ such that single-task learning with $\gamma n(1 - \pi_S)$ observations will have the same learning curve value as the learning curve for the multi-task one, i.e.,

$$\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n) = \epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, \gamma n) \tag{3.60a}$$

or, equivalently,

$$\frac{\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n)}{\epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, \gamma n)} = 1. \tag{3.60b}$$

We expect $\gamma > 1$ so that multi-task learning effectively contributes additional $(\gamma - 1)(1 - \pi_S)n$ observations for task T in the single-task learning setting. We seek analytical solutions to gain insights to multi-task learning; unfortunately, (3.60) cannot be solved analytically, except for the following two cases.

1. If $\rho = 0$, then $\gamma = 1$ by matching the parameters to ϵ_T^{avg} on the left and right of (3.60a). This means that the observations for the secondary task S do not contribute at all to the learning of the primary task T.

2. If $\rho = \pm 1$, then $\gamma = 1/(1 - \pi_S)$. This is because, when $\rho = \pm 1$, there are effectively n observations for task T. Having $n/(1 - \pi_S)$ total observations when the two tasks were to be uncorrelated would then mean that task T had $(1 - \pi_S)n/(1 - \pi_S) = n$ observations; see the discussion of Corollary 3.12 on page 64.

To make progress for the other values of ρ, we shall follow the approach of section 3.6 and replace the true learning curves in (3.60) by theoretical bounds on the learning curves.

3.7.1 Contribution to Lower Bound on the Average-case Sample Complexity

In section 3.6, we have seen that ratios of the asymptotics of the OV_ρ lower bounds are indicative of the ratios of the true learning curve. Building upon this, we shall replace each true learning curve in

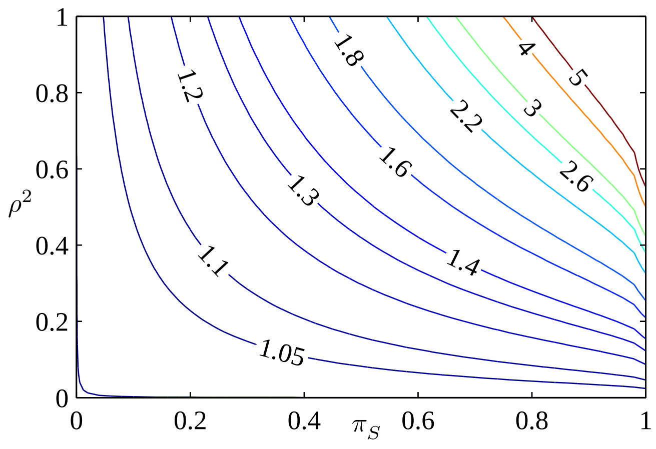


Figure 3.12: Contour plot of $\underline{\gamma}^{(r)}$, according to equation 3.62 with $r = 2$ for the Ornstein-Uhlenbeck process. The contour intervals are not uniform.

(3.60b) by the asymptotics of its OV$_\rho$ lower bounds. As in section 3.6.2, the asymptotics of the OV$_\rho$ lower bound is obtained by applying the asymptotic value of the single-task OV bound twice within the mixture formation of the OV$_\rho$ bound (Proposition 3.14a). In the case of a GP with process eigenvalues decaying as an order-r power law, we use equation 3.60b, replacing γ by $\underline{\gamma}^{(r)}$, to obtain

$$\begin{aligned}1 &= \frac{\rho^2(\eta_0\tilde{n})^{1/r}/\tilde{n} + (1-\rho^2)\left(\eta_0(1-\pi_S)\tilde{n}\right)^{1/r}/\left((1-\pi_S)\tilde{n}\right)}{\left(\eta_0(1-\pi_S)\underline{\gamma}\tilde{n}\right)^{1/r}/\left((1-\pi_S)\underline{\gamma}\tilde{n}\right)} \\ &= \left(\underline{\gamma}^{(r)}\right)^{(r-1)/r}\frac{\rho^2(\eta_0\tilde{n})^{1/r}/\tilde{n} + (1-\rho^2)\left(\eta_0(1-\pi_S)\tilde{n}\right)^{1/r}/\left((1-\pi_S)\tilde{n}\right)}{\left(\eta_0(1-\pi_S)\tilde{n}\right)^{1/r}/\left((1-\pi_S)\tilde{n}\right)} \qquad (*) \\ &= \left(\underline{\gamma}^{(r)}\right)^{(r-1)/r}\left\{\widehat{\mathrm{DF}}^{(r)}(\rho,\sigma_{\mathrm{n}}^2,\pi_S,n)\right\} \qquad (3.61)\end{aligned}$$

where we have recognized the fraction in $(*)$ as the approximate error deflation factor $\widehat{\mathrm{DF}}^{(r)}$ given by equation 3.53a. The η_0 and $\tilde{n}$ above follows those defined in section 3.6. Substituting in $\widehat{\mathrm{DF}}^{(r)}$ given by equation 3.53a, we get

$$\underline{\gamma}^{(r)} = \left[1 - \rho^2\left(1 - (1-\pi_S)^{(r-1)/r}\right)\right]^{-r/(r-1)}, \qquad (3.62)$$

which is at least one. Since $\underline{\gamma}^{(r)}$ is obtained by using the lower bounds for the true learning curves, it measures how asymmetric multi-task learning affects the necessary sample size for learning task T. Factor $\underline{\gamma}^{(r)}$ becomes the factor γ for the exact sample sizes when $\rho^2 = 0$ and $\rho^2 = 1$, for which $\underline{\gamma}^{(r)} = 1$ and $\underline{\gamma}^{(r)} = 1/(1-\pi_S)$ respectively.

Figure 3.12 gives the contour plot $\underline{\gamma}^{(r)}$ for $r = 2$, which is for the Ornstein-Uhlenbeck process (OU), on the π_S-ρ^2 space. The contour plot, together with equation 3.62, confirms the intuition that multi-task learning with higher ρ^2 and higher π_S contributes more to the lower bound on sample complexity. For example, multi-task learning for the OU process with $\pi_S = \rho^2 = 0.5$ effectively contributes 37% additional examples to the lower bound on sample complexity; with $\pi_S = \rho^2 = 0.75$, the contribution is 156%. Similar contour plots are obtained for other values of r.

3.7.2 Contribution to Average-case Sample Complexity

The preceding section approximates the ratios of the true learning curves by the ratios of the asymptotics of the OV$_\rho$ bounds. Instead of using the OV$_\rho$ bound, we may also choose to use the FWO$_\varrho$ upper bound proposed in Proposition 3.16. In this way, we can quantify how asymmetric multi-task learning affects the sample complexity for learning task T.

For a stationary Gaussian process with eigenvalues decaying as the order-r power-law, i.e., $\kappa_i \sim \eta_0 (bi)^{-r}$ for some constant $\eta_0 > 0$, the FWO$_\varrho$ bound may be shown to be asymptotically bounded from above by

$$\beta_1 \sum_{i=1}^{\infty} \kappa_i + 2(1-\beta_1)\eta_1 \eta_0^{1/r} (\beta_2 \underset{\sim}{n})^{-(r-1)/r}, \tag{3.63a}$$

where

$$\underset{\sim}{n} \overset{\text{def}}{=} \frac{n}{\sum_{i=1}^{\infty} \kappa_i + \sigma_\mathrm{n}^2} \qquad \eta_1 = \frac{b}{\pi r \sin(\pi/r)} \tag{3.63b}$$

$$\beta_1 = \frac{\pi_S^2 \varrho^2 (1-\rho^2)}{(1-\pi_S)^2 + 2\pi_S(1-\pi_S)\rho\varrho + \pi_S^2\varrho^2} \qquad \beta_2 = \frac{(1-\pi_S)^2 + 2\pi_S(1-\pi_S)\rho\varrho + \pi_S^2\varrho^2}{1-\pi_S+\pi_S\varrho^2}. \tag{3.63c}$$

This is derived in appendix B.8.5. It is straightforward to verify that $\beta_1 \in [0,1]$, so one may view (3.63a) as a weighted average between the average prior variance and a term that depends on sample size. Using (3.63a) for the learning curves and $\overline{\gamma}^{(r)}$ for the γ in (3.60), and then solving for $\overline{\gamma}^{(r)}$ gives

$$\overline{\gamma}^{(r)} = \frac{\beta_2}{1-\pi_S} \left[(1-\beta_1) + \beta_1 \frac{\sum_{i=1}^{\infty} \kappa_i}{2\eta_1 \eta_0^{1/r}} \left(\beta_2 \underset{\sim}{n}\right)^{(r-1)/r} \right]^{-r/(r-1)}. \tag{3.64}$$

The above formula for $\overline{\gamma}^{(r)}$ depends on the choice of ϱ; presently, we shall use $\varrho = \hat{\varrho}$ is given by equation 3.40 on page 71:

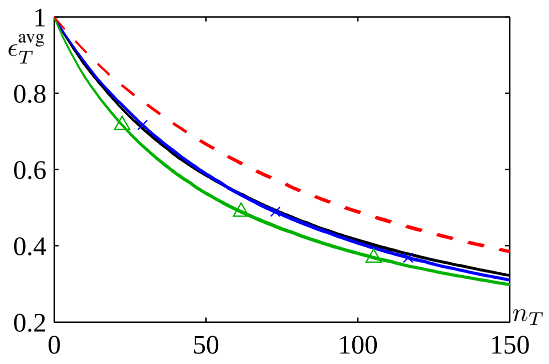
$$\hat{\varrho} \overset{\text{def}}{=} \frac{\rho}{1 + \pi_S(1-\rho^2) n \left\{ \left(\sum_{i=1}^{\infty} \kappa_i + \sigma_\mathrm{n}^2 \right) / \bar{\kappa} - 1 \right\}^{-1}}, \tag{3.65}$$

where $\bar{\kappa} = \kappa_1$ is the largest process eigenvalue, and we have used $\int k^\mathrm{x}(\boldsymbol{x},\boldsymbol{x})[\phi_1(\boldsymbol{x})]^2 p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} = \sum \kappa_i$ for stationary GPs. Under this setting, the factor γ for the true sample sizes is recovered at zero and perfect correlations: if $\rho = 0$, then $\hat{\varrho} = 0$, $\beta_1 = 0$ and $\beta_2 = 1 - \pi_S$, so that $\overline{\gamma}^{(r)} = 1$; if $\rho = \pm 1$, then $\hat{\varrho} = \pm 1$, $\beta_1 = 0$ and $\beta_2 = 1$, so that $\overline{\gamma}^{(r)} = 1/(1-\pi_S)$.

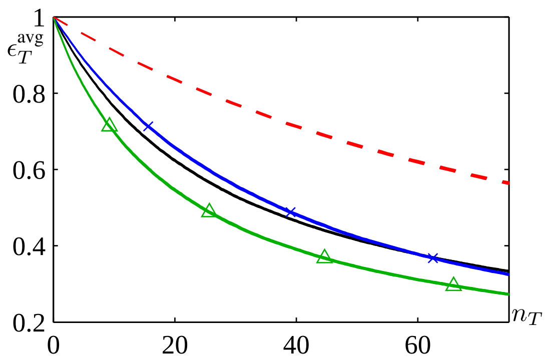
3.7.3 Example: Multi-task Ornstein-Uhlenbeck Process Learning

The preceding two sections have proposed the factor $\underline{\gamma}^{(r)}$ for approximating contribution of the secondary task to the lower bound on sample complexity for the primary task, and the factor $\overline{\gamma}^{(r)}$ for approximating the contribution to the sample complexity. In this section, we examine how well these factors approximate or bound the contribution to the effective number of observations for the primary task. This will be done using the simulated data set for the Ornstein-Uhlenbeck process described in section 3.5.5, where the length-scale is $l = 0.01$, and we use $\varrho = \hat{\varrho}$ in $\overline{\gamma}^{(r)}$. Within this setting,

$$r = 2 \qquad b = \pi \qquad \sigma_\mathrm{n}^2 = 0.05 \qquad \sum_{i=1}^{\infty} \kappa_i = 1 \qquad \bar{\kappa} = 0.020 \qquad \eta_0 = 2/l = 200. \tag{3.66}$$



(a) $k^{\text{x}} = \text{OU}, \rho^2 = 1/2, \pi_S = 1/2$



(b) $k^{\text{x}} = \text{OU}, \rho^2 = 3/4, \pi_S = 3/4$

Figure 3.13: Comparison for the effective increase in sample size for the target task, for two settings of (ρ^2, π_S) for multi-task learning with the Ornstein-Uhlenbeck process $(r = 2)$. Each graph plots ϵ_T^{avg} against n_T and consists of: the single-task learning curve $\epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, n)$ (- -), the multi-task learning curve $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n)$ (—), the single-task learning curve with $(\underline{\gamma}^{(r)} - 1)n_T$ additional observations (✕) and the single-task learning curve with $(\overline{\gamma}^{(r)} - 1)n_T$ additional observations (△). The thickness of the learning curves reflect 95% confidence intervals.

In Figure 3.13, we plot the baseline single-task learning curve $\epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, n)$, the multi-task learning curve $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n)$, the single-task learning curve with $(\underline{\gamma}^{(r)} - 1)n_T$ additional observations and the single-task learning curve with $(\overline{\gamma}^{(r)} - 1)n_T$ additional observations. In contrast to the other graphs of learning curves in this chapter, which are plotted against n, the graphs in Figure 3.13 are plotted against $n_T = (1 - \pi_S)n$ in order to emphasize that we are currently using the single-task learning $\rho = 0$ as the baseline. Since $\underline{\gamma}^{(r)}$ is for the necessary (but not sufficient) sample size, we expect the single-task learning curve with this factor to lie above the multi-task learning curve, since it does not give enough additional samples to equate the performance of multi-task learning. Similarly, we expect the single-task learning curve for $\overline{\gamma}^{(r)}$ to lie below the multi-task learning curve, since the factor $\overline{\gamma}^{(r)}$ gives more than enough samples for single-task learning to equate multi-task learning.

The proximity between the $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n)$ curves (—) and the $\epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, \underline{\gamma}^{(r)}n)$ curves (✕) shows that $\underline{\gamma}^{(r)}$ reflects rather accurately the increase in the effective observations due to asymmetric multi-task learning. However for larger n_T, $\epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, \underline{\gamma}^{(r)}n)$ lies below $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n)$, so that $\underline{\gamma}^{(r)}$ is an overestimate. One reason for this has been remarked upon in section 3.6.2, and it is that $\hat{\pi}_S^{(r)} \stackrel{\text{def}}{=} 1 - (1 - \pi_S)^{(r-1)/r}$ is an overestimate that becomes worse as $\tilde{n}$ increases, so

$$\underline{\gamma}^{(r)} = \left(1 - \rho^2 \hat{\pi}_S^{(r)}\right)^{-r/(r-1)} \tag{3.67}$$

is also an overestimate. In contrast, the $\epsilon_T^{\text{avg}}(0, \sigma_{\text{n}}^2, \pi_S, \overline{\gamma}^{(r)}n)$ curves (△) lie consistently below the multi-task $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, n)$ curves (—) in the figures. Thus $\overline{\gamma}^{(r)}$ seems to be a reasonable estimate for the factor of increase contributed by multi-task learning to the sample complexity.

3.8 Empirical Evaluation with Sarcos Data

The results developed so far in this chapter have been illustrated using the idealized one-dimensional input distributions described in section 3.5.5. We now revisit some of these results and re-evaluate them on the Sarcos data, which is a high-dimensional data set based on a real application. In particular, we shall look at the lower bound to the posterior variance

$$\underline{\sigma}_T^2(\boldsymbol{x}_*, \rho, \sigma_{\mathrm{n}}^2, X_T, X_S) \overset{\mathrm{def}}{=} \rho^2 \sigma_T^2(\boldsymbol{x}_*, 1, \sigma_{\mathrm{n}}^2, X_T, X_S) + (1-\rho^2)\sigma_T^2(\boldsymbol{x}_*, 0, \sigma_{\mathrm{n}}^2, X_T, X_S)$$

given by Proposition 3.5a. The simulation in section 3.5.5 has shown that this bound is rather tight when averaged over test locations $\boldsymbol{x}_*$s and data sets X_Ts and X_Ss; see, for example, the $\times$ lines in Figure 3.5. Proposition 3.7a gives a predictor that achieves the generalization error $\underline{\sigma}_T^2(\boldsymbol{x}_*, \rho)$ exactly; in section 3.8.2 we shall use this predictor to approximate the empirical errors of the multi-task GP predictor. Then, in section 3.8.3, we shall evaluate the tightness of the lower bound $\underline{\sigma}_T^2$ by examining the learning curves for the Sarcos data under the assumption of correct prior specification. Finally, section 3.8.4 illustrates the utility of the optimal error deflation factor $(1 - \rho^2\pi_S)$ in estimating the potential gain from a multi-task setup. We begin with a description of the setup below.

3.8.1 The Sarcos Data and Hyperparameter Estimation

We use data from a Sarcos Dextrous Arm$^{\mathrm{TM}}$, a manipulator with seven degrees-of-freedom.[12] Following Rasmussen and Williams [2006, §2.5], GP regression is used to learn and predict the inverse dynamics of this manipulator, i.e., to map the seven 3-tuples of joint positions, joint velocities and joint accelerations to the corresponding seven joint torques. Thus, we have a 21-dimensional input space and a 7-dimensional output space. The training data size is 44,484, and the test data size, 4,449.

The primary task T is to predict the first of the seven torques. Each of the other torques is used separately as a secondary task to task T. If the i^{th} torque is used, we denote the secondary task by S_i, $i = 2 \ldots 7$. The model follows that described in section 3.2.1, where K^{f} is a two-by-two correlation matrix fully parameterized by ρ. The covariance function k^{x} on the input space is the squared-exponential covariance function parameterized for automatic relevance determination (ARD), i.e.,

$$k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left[-(\boldsymbol{x} - \boldsymbol{x}')^{\mathrm{T}} D^{-1} (\boldsymbol{x} - \boldsymbol{x}')/2\right]$$

with a diagonal matrix D of squared length-scales and signal variance of σ_f^2. Observations for tasks T and S_i have a common isotropic noise variance σ_{n}^2.

To proceed further, we have to determine the model by fixing the hyperparameters ρ, σ_f^2, D and σ_{n}^2. This is achieved using the subset of data approximation to the full-sample GP [Rasmussen and Williams, 2006, §8.3.3]. For a total of $n = 4096$ observations, we sample $n/2$ input locations from the training set for task T (i.e., the torque for the first joint) and $n/2$ input locations for task S_i. We standardize the inputs and outputs of the sampled data set to zero mean and unit variance for each dimension.[13] Then

[12] `http://www.sarcos.com`. The data and its description available from `http://www.gaussianprocess.org/gpml/data/`. The data is provided by Sethu Vijayakumar.

[13] It is important to standardize the output to a common variance, since our model have common variance parameters for both task T and task S_i.

Table 3.1: nMSEs for predicting the first torque (task T) and the correlation (ρ) after optimization. The first two rows are single-task GP results, while the rest are multi-task GP results using secondary task S_i. The nMSEs for the multi-task GP predictor and the randomized predictor are given in the last two columns. Results are given with one standard deviation around the mean.

Setup			Empirical Results		Derived Results
Secondary	n	π_S	ρ	nMSE$_T$ (Multi-task)	nMSE$_T$ (Randomized)
None	2048			0.0289 ± 0.0014	
None	4096			0.0198 ± 0.0009	
S_2	4096	0.5	0.373 ± 0.060	0.0277 ± 0.0015	0.0276 ± 0.0020
S_3	4096	0.5	-0.499 ± 0.041	0.0279 ± 0.0017	0.0266 ± 0.0018
S_4	4096	0.5	0.596 ± 0.041	0.0269 ± 0.0016	0.0257 ± 0.0020
S_5	4096	0.5	-0.076 ± 0.061	0.0284 ± 0.0014	0.0288 ± 0.0014
S_6	4096	0.5	-0.363 ± 0.056	0.0275 ± 0.0014	0.0277 ± 0.0019
S_7	4096	0.5	0.458 ± 0.045	0.0271 ± 0.0016	0.0270 ± 0.0018

the set of hyperparameters in each multi-task setting is determined using L-BFGS [Liu and Nocedal, 1989] to optimize its marginal likelihood on the sampled data. We evaluate the model on the full test set, and summarize the quality of the hyperparameter using the normalized mean-square error (nMSE) on task T.[14] This is repeated ten times.

Table 3.1 summarizes the results of learning the hyperparameters. The first two rows of the table are the single-task results, and they are consistent with those from Rasmussen and Williams [2006, Table 8.1]. The column headed by ρ gives the estimated correlations between task T and tasks S_i, while the column headed by *nMSE$_T$ (Multi-task)* gives the nMSEs of predicting with the multi-task GP model (3.1). The last column in Table 3.1 gives the nMSEs when using the randomized predictor. This is discussed next.

3.8.2 Using the Randomized Predictor to Estimate Multi-task Empirical Errors

Proposition 3.7 provides an understanding of the lower bound $\underline{\sigma}_T^2(\boldsymbol{x}_*, \rho)$ in terms of a randomized predictor. To recap, let $\bar{f}_1$ be the single-task GP mean predictor that uses the observations for the primary task T at all training locations X_T and X_S, and let $\bar{f}_0$ be the single-task GP mean predictor only uses the observations for task T at training locations X_T. The randomized predictor is that which predicts $\bar{f}_1$ with probability ρ^2 and $\bar{f}_0$ with probability $(1-\rho^2)$.

For a given observation set size $n = |X|$, if π_S is the proportion of training data for the secondary task S_i, then $\bar{f}_1$ predicts using n observations while $\bar{f}_0$ predicts using $(1-\pi_S)n$ observations. Let nMSE$_T(n)$ be the nMSE for task T of the single-task GP with n training samples. Then the nMSE for task T of the randomized predictor is given by

$$\rho^2 \mathrm{nMSE}_T(n) + (1-\rho^2)\mathrm{nMSE}_T((1-\pi_S)n). \tag{3.68}$$

[14] This is done by factoring back the standard deviation and adding back the mean to the predictions, computing the mean-square error with respect to the test targets, and then dividing it by the variance of the test targets.

We have seen that the generalization error of the randomized predictor is a tight lower bound on the generalization error of the multi-task GP predictor. However, in terms of the empirical error, how does the randomized predictor compare with the multi-task GP predictor? We perform a post-hoc analysis for this comparison and compute (3.68) for each T-S_i multi-task setup in Table 3.1. This gives the last column in the table, where the standard deviations of (3.68) are obtained by propagating those of the constituents; see Bevington and Robinson 2002, section 3. Comparing the last two columns in the table, we find the empirical error of the randomized predictor to be an estimate of the empirical nMSE$_T$. We also find that the empirical error of the randomized predictor is not a lower bound on the empirical error of the multi-task GP in general. This apparent discrepancy can be reconciled by realising that (a) the true model is not known so that the learning model is probably not correctly specified; and (b) the empirical error nMSE$_T$ is with respect to *one fixed function* in the test data, and not averaged over possible functions specified by the prior.

3.8.3 The $\underline{\sigma}_T^2$ Lower Bound

Presently, we examine the lower bound $\underline{\sigma}_T^2$ given by Proposition 3.5a. For each T-S_i multi-task setup, we choose, from among the ten repetitions, the set of hyperparameters that gives the lowest nMSE on the full test set. Using the chosen hyperparameters — together with the corresponding mean and variance for standardizing data — for each setup, we generate the learning curve assuming that the true function is also a Gaussian process with the same covariance function, i.e., the idealized case where we need only to consider the posterior variance, and where the observed torque values are not involved. Each learning curve is computed by averaging the posterior variance over test locations $\boldsymbol{x}_*$ in the test set of 4,449 samples, and over 50 samples of X drawn at random from the full training set of 44,484 examples. Each sampled training set X is partitioned randomly into X_T and X_S based on the proportion π_S. Values $1/4$, $1/2$ and $3/4$ are used for π_S. Below, the average over $\boldsymbol{x}_*$ is denoted by $\langle\cdot\rangle$, and the average over X by $\langle\!\langle\cdot\rangle\!\rangle$.

Figure 3.14 on the following page shows the learning curves with S_4 and S_7 as the secondary tasks, and when $\pi_S = 3/4$. For each multi-task setting, the set of optimal hyperparameters includes a correlation $\hat{\rho}$ between the two tasks that is most likely for the data. The "true" multi-task learning curve $\langle\!\langle\langle\sigma_T(\hat{\rho})\rangle\rangle\!\rangle$ is computed using $\hat{\rho}$. This gives the upper black lines in Figure 3.14. The experimental trivial lower bound on the multi-task curve is the curve when $\rho = 1$, and this gives the lower dashed lines in the figure. The experimental trivial upper bound is when $\rho = 0$, and this gives the upper dashed lines in the figure; see Proposition 3.1. We obtain the experimental trivial upper bound by rescaling the trivial lower bound on the n-axis by $1/(1-\pi_S)$; see the discussion on Corollary 3.12. For the trivial upper bound at $n = 1$ when the rescaling cannot be applied, we instead use the following equation obtained from Corollary 3.13:

$$\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, 1) = \pi_S(1-\rho^2)\sum_{i=1}^{\infty}\kappa_i + [1 - \pi_S(1-\rho^2)]\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 1, 1), \tag{3.69}$$

where $\sum \kappa_i = \sigma_f^2$ in this case. This is derived in appendix B.9. Using Proposition 3.5a, a tighter lower bound on the multi-task curve is the weighted average of the trivial lower and upper bounds, with weights $\hat{\rho}^2$ and $(1-\hat{\rho}^2)$. This tighter lower bound $\langle\!\langle\langle\underline{\sigma}_T(\hat{\rho})\rangle\rangle\!\rangle$ is the blue solid lines in Figure 3.14.

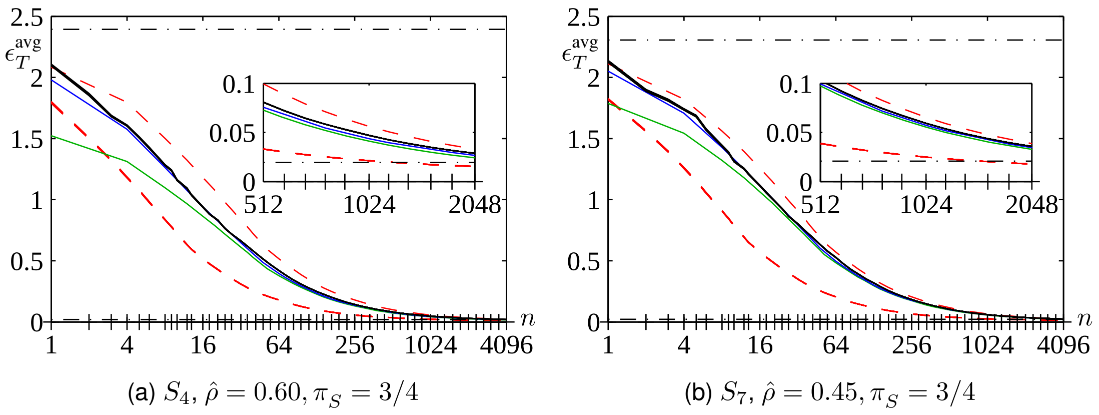


Figure 3.14: Learning curves for multi-task learning the inverse dynamics for the first joint (task T) with the fourth joint (task S_4) and with the seventh joint (S_7), using the Sarcos data. Each graph plots ϵ_T^{avg} against n, and consists of: the "true" multi-task learning curve (upper —), the experimental trivial lower/upper bounds using Corollary 3.12 (lower/upper - -), the experimental lower bound using Proposition 3.5a (middle —), and an estimate of the multi-task learning curve using the optimal error deflation factor (lower —). Other than the lower — , which is described in section 3.8.4, the rest of the curves are described in section 3.8.3. The prior and noise variances are plotted with upper and lower horizontal dash-dotted lines. The horizontal axis is on the $\log_2$ scale, and the ticks on this axis represent values of n for which the learning curves are computed by data averaging. The inset magnifies the curves for $n \in [512, 2048]$. The thickness of the multi-task learning curve reflects 95% confidence interval; similarly for the experimental trivial lower bound curve.

The proximity between the black and blue solid lines in Figure 3.14 demonstrates that the multi-task learning curve $\langle\!\langle \langle \sigma_T(\hat{\rho}) \rangle \rangle\!\rangle$ is well approximated by the tighter lower bound $\langle\!\langle \langle \underline{\sigma}_T(\hat{\rho}) \rangle \rangle\!\rangle$. The same conclusion can be drawn from the graphs for the other S_is and π_Ss. Bearing in mind that this data set is an empirical data set, that the input space is 21-dimensional, and that the set of hyperparameters is optimized for its marginal likelihood on the training data, it hints strongly that $\langle\!\langle \langle \underline{\sigma}_T(\hat{\rho}) \rangle \rangle\!\rangle$ may generally be used as a proxy for the multi-task learning curve. It is also pleasing that $\langle\!\langle \langle \underline{\sigma}_T(\hat{\rho}) \rangle \rangle\!\rangle$ can be obtained with a procedure that only requires data averaging for the experimental trivial lower bound, so that the learning curves for different ρs and π_Ss can be approximated relatively cheaply.

The experimental results in this section provide confidence that the multi-task OV_ρ lower bound given in Proposition 3.14, which is based on Proposition 3.5a, may provide insights into multi-task learning. It is with this confidence that we have approached sections 3.6 and 3.7 by using the OV_ρ bound as a proxy for the multi-task learning curve.

3.8.4 Error Deflation Factor

In section 3.6.2, we have defined error deflation factors in asymmetric multi-task learning as that which quantifies the reduction in error of the primary task T. One of the conclusions there is that the factor $\rho^2\pi_S$ may be used to quantify the maximum reduction in the learning curve. We investigate this suggestion by plotting

$$(1 - \rho^2 \pi_S) \, \langle\!\langle \langle \sigma_T(0) \rangle \rangle\!\rangle \tag{3.70}$$

in Figure 3.14, where, we recall, $\langle\!\langle \langle \sigma_T(0) \rangle \rangle\!\rangle$ is the learning curve for the single-task learning solely with observations for task T. In the figure, (3.70) gives the lower green solid line, $\langle\!\langle \langle \sigma_T(0) \rangle \rangle\!\rangle$ is the upper dashed line, and the multi-task learning curve $\langle\!\langle \langle \sigma_T(\hat\rho) \rangle \rangle\!\rangle$ is the upper black solid line. Comparing the curves, we see that (3.70) bounds $\langle\!\langle \langle \sigma_T(\hat\rho) \rangle \rangle\!\rangle$ from below, so that error reduction in this case does not exceed the factor $\rho^2 \pi_S$. This result bears well with the analysis in section 3.6.2. Another observation is that (3.70) actually approximates the multi-task learning curve rather well from moderate to large n, although not as well as $\langle\!\langle \langle \underline{\sigma}_T(\hat\rho) \rangle \rangle\!\rangle$ does.

3.9 Asymmetric Multi-task Learning with Noise-free Data

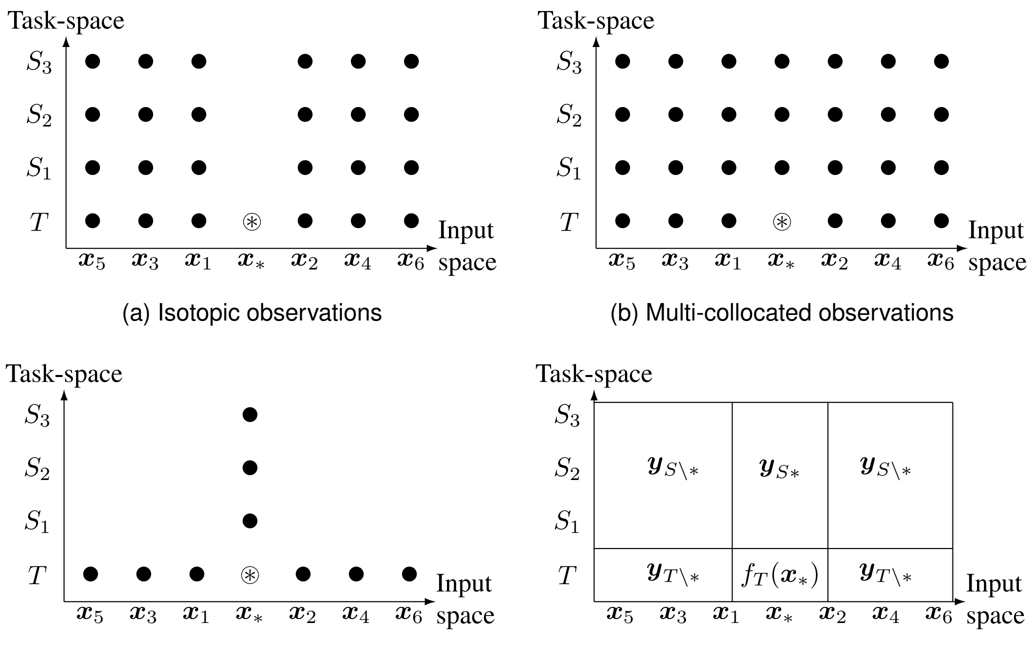
So far, we have looked at multi-task learning where there are two tasks. This has been a convenient choice for analysis since the relation between the two tasks can be given completely by the single parameter $\rho \in [-1, 1]$, which measures the correlation or relatedness between the two tasks. For $M > 2$ tasks, however, we have to involve $\Theta(M^2)$ parameters in the task correlation matrix K^{f} for describing task relations. With these many parameters, we may well need to limit to selected scenarios in order to have any results meaningful enough to provide additional insights to multi-task learning over the two-tasks case. Indeed, it will prove helpful to limit ourselves to noise-free observations and specific placements of training observations. This is the subject of this section. Our analysis here is for the posterior mean and variance of the multi-task GP for the primary task T at any test location $\boldsymbol{x}_*$. The posterior mean is useful for understanding structure in predicting with a multi-task GP, while, as we recall, the posterior variance is the generalization error for task T at $\boldsymbol{x}_*$ when the GP prior is correctly specified.

To be clear, the multi-task learning model under consideration is that given by (3.1) in section 3.2.1, but with perhaps more than two tasks and with observation noise variance $\sigma_{\mathrm{n}}^2 = 0$. To make the notation cleaner, we consider $M + 1$ tasks and denote the primary task by T and the secondary tasks by S_i, $i = 1 \ldots M$. For the symmetric multi-task case, the learning behaviour will depend on the eigenvalues of K^{f}, as has been in the two-tasks case; see sections 3.2.3 and 3.6.1. Presently, our focus is the asymmetric multi-task case, where we wish to predict the response of f_T at $\boldsymbol{x}_*$ given observations at X_T for task T and X_i for task S_i. For a given covariance function $k^{\mathrm{x}}(\cdot, \cdot)$ on the input space, we let k^{x}_{**} be the prior variance $k^{\mathrm{x}}(\boldsymbol{x}_*, \boldsymbol{x}_*)$, let K^{x}_{TT} (resp. K^{x}_{ii}) be the matrix of covariances (due to k^{x}) between locations in X_T (resp. X_i), and let $\boldsymbol{k}^{\mathrm{x}}_{T*}$ (resp. $\boldsymbol{k}^{\mathrm{x}}_{i*}$) be the vector of covariances (due to k^{x}) between the locations in X_T (resp. X_i) and the test location $\boldsymbol{x}_*$. Finally, the task correlation matrix K^{f} is partitioned as

$$K^{\mathrm{f}} \overset{\mathrm{def}}{=} \begin{pmatrix} 1 & \boldsymbol{\rho}^{\mathrm{T}} \\ \boldsymbol{\rho} & K^{\mathrm{f}}_S \end{pmatrix}, \tag{3.71}$$

where K^{f}_S is the task correlation matrix between the secondary tasks S_is and $\boldsymbol{\rho}$ is the vector of task correlations between the secondary tasks S_is and the primary task T.

Section 3.9.1 reviews a property of the multi-task GP model that is of importance to modelling relations between tasks: when the noise-free observations are at the same locations for all tasks, there is no


Task-space
S_3
S_2
S_1
T
Input space
$\boldsymbol{x}_5$ $\boldsymbol{x}_3$ $\boldsymbol{x}_1$ $\boldsymbol{x}_*$ $\boldsymbol{x}_2$ $\boldsymbol{x}_4$ $\boldsymbol{x}_6$
$\boldsymbol{y}_{S\backslash *}$
$\boldsymbol{y}_{S*}$
$\boldsymbol{y}_{T\backslash *}$
$f_T(\boldsymbol{x}_*)$

(a) Isotopic observations

(b) Multi-collocated observations

(c) Collocated observations

(d) Pictorial view of the symbols used in the text

Figure 3.15: Given a test location $\circledast$ for task T, Figures (a)-(c) show three layouts of the noise-free observations ●s on the task-input product space. Figure (d) places the symbols for the observations pictorially on the task-input space.

transfer of information between the tasks during inference. While this property has been known in the geostatistics and statistics literature, its implications in machine learning, particularly with respect to multi-task learning, may not have been known. In sections 3.9.2 and 3.9.3, we give expressions for the posterior distribution that allow us to understand multi-task learning with multi-collocated and collocated data. As far as we are aware, these expressions have been known only for the case of only one secondary task; here, the expressions are given for the case of an arbitrary number of secondary tasks. Figure 3.15 summarizes the scenarios considered in this section.

3.9.1 Isotopic Observations and Non-transference[15]

One particularly interesting case to consider is observations at the same locations for all tasks, i.e., $X_T \equiv X_1 \equiv \cdots \equiv X_M$, so $K^{\mathrm{x}}_{\backslash *} \stackrel{\mathrm{def}}{=} K^{\mathrm{x}}_{TT} \equiv K^{\mathrm{x}}_{11} \equiv \cdots K^{\mathrm{x}}_{MM}$. This is known as the isotopic case in the geostatistics literature [Wackernagel, 1998, chapter 25] and is illustrated in Figure 3.15a. In machine learning, one may expect such data in the case of learning vector-valued functions (see section 2.7.2).

If the isotopic observations are noise-free, there will be no transfer or sharing of information between the tasks. To see this, consider making predictions at a new location $\boldsymbol{x}_*$ common for all tasks. Let $\boldsymbol{y}_{T\backslash *}$ (resp. $\boldsymbol{y}_{i\backslash *}$) be the noise-free observations at X_T (resp. X_i) for task T (resp S_i), and let $\boldsymbol{y}^{\mathrm{T}}_{\backslash *} \stackrel{\mathrm{def}}{=} \left(\boldsymbol{y}^{\mathrm{T}}_{T\backslash *}, \boldsymbol{y}^{\mathrm{T}}_{1\backslash *}, \ldots, \boldsymbol{y}^{\mathrm{T}}_{M\backslash *}\right)$ be the full sequence of observations. Also, let $\boldsymbol{k}^{\mathrm{x}}_{*} \stackrel{\mathrm{def}}{=} \boldsymbol{k}^{\mathrm{x}}_{T*} \equiv \boldsymbol{k}^{\mathrm{x}}_{1*} \equiv \cdots \equiv \boldsymbol{k}^{\mathrm{x}}_{M*}$. Using the mixed-product property of Kronecker products, the posterior means at $\boldsymbol{x}$ for all

[15] This section is included for completeness; the results therein are not novel.

tasks is

$$\begin{aligned}
\mathbb{E}\left((f_T(\boldsymbol{x}_*), f_1(\boldsymbol{x}_*), \ldots, f_M(\boldsymbol{x}_*))^{\mathrm{T}} \mid \boldsymbol{y}_{\setminus *} \right) &= \left(K^{\mathrm{f}} \otimes \boldsymbol{k}_*^{\mathrm{x}}\right)^{\mathrm{T}} \left(K^{\mathrm{f}} \otimes K^{\mathrm{x}}_{\setminus *}\right)^{-1} \boldsymbol{y}_{\setminus *} \\
&= \left((K^{\mathrm{f}})^{\mathrm{T}} \otimes (\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}}\right) \left((K^{\mathrm{f}})^{-1} \otimes (K^{\mathrm{x}}_{\setminus *})^{-1}\right) \boldsymbol{y}_{\setminus *} \\
&= \left[\left(K^{\mathrm{f}} (K^{\mathrm{f}})^{-1}\right) \otimes \left((\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} (K^{\mathrm{x}}_{\setminus *})^{-1}\right)\right] \boldsymbol{y}_{\setminus *} \\
&= \left[I \otimes \left((\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} (K^{\mathrm{x}}_{\setminus *})^{-1}\right)\right] \boldsymbol{y}_{\setminus *} \\
&= \begin{pmatrix} \hat{y}_{T*} \\ \hat{\boldsymbol{y}}_{S*} \end{pmatrix}, \qquad (3.72a)
\end{aligned}$$

where

$$\hat{y}_{T*} \stackrel{\mathrm{def}}{=} (\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} (K^{\mathrm{x}}_{\setminus *} x)^{-1} \boldsymbol{y}_{T \setminus *} \qquad (3.72b)$$

$$(\hat{\boldsymbol{y}}_{S*})_i = \hat{y}_{i*} \stackrel{\mathrm{def}}{=} (\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} (K^{\mathrm{x}}_{\setminus *})^{-1} \boldsymbol{y}_{i \setminus *}. \qquad (3.72c)$$

Similarly for the posterior covariance:

$$\begin{aligned}
\mathbb{C}\left((f_T(\boldsymbol{x}_*), f_1(\boldsymbol{x}_*), \ldots, f_M(\boldsymbol{x}_*))^{\mathrm{T}} \mid \boldsymbol{y}_{\setminus *} \right) &= K^{\mathrm{f}} k^{\mathrm{x}}_{**} - \left(K^{\mathrm{f}} \otimes \boldsymbol{k}_*^{\mathrm{x}}\right)^{\mathrm{T}} \left(K^{\mathrm{f}} \otimes K^{\mathrm{x}}_{\setminus *}\right)^{-1} \left(K^{\mathrm{f}} \otimes \boldsymbol{k}_*^{\mathrm{x}}\right) \\
&= K^{\mathrm{f}} k^{\mathrm{x}}_{**} - \left(K^{\mathrm{f}} (K^{\mathrm{f}})^{-1} K^{\mathrm{f}}\right) \otimes \left((\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} (K^{\mathrm{x}}_{\setminus *})^{-1} \boldsymbol{k}_*^{\mathrm{x}}\right) \\
&= K^{\mathrm{f}} \left(k^{\mathrm{x}}_{**} - (\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} (K^{\mathrm{x}}_{\setminus *})^{-1} \boldsymbol{k}_*^{\mathrm{x}}\right) \\
&= K^{\mathrm{f}} \sigma_*^2, \qquad (3.72d)
\end{aligned}$$

where

$$\sigma_*^2 \stackrel{\mathrm{def}}{=} k^{\mathrm{x}}_{**} - (\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} (K^{\mathrm{x}}_{\setminus *})^{-1} \boldsymbol{k}_*^{\mathrm{x}}. \qquad (3.72e)$$

Thus, given noise-free isotopic observations, the predictions $\hat{y}_{T*}$ (resp. $\hat{y}_{i*}$) for task T (resp. S_i) depend only on the targets $\boldsymbol{y}_T$ (resp. $\boldsymbol{y}_i$), and the variances of the predictions take the same value σ_*^2 (since K^{f} is defined to have ones on its diagonal). In other words, there is no transfer or sharing of information between the tasks. This result is known as *self-* or *auto-krigeability* in the geostatistics literature [Wackernagel, 1998, chapter 25]. It is also related to the *Markov property* of covariance functions [Xu et al., 1992; Almeida and Journel, 1994; Goovaerts, 1997; O'Hagan, 1998]; the Markovian nature of the model will be further explained in section 3.9.3. Williams, Chai, and Bonilla [2007] have generalized the above result to multidimensional tensor product covariance functions and grids.

This non-transference for a given task does still hold even if the isotopic observations are only sparsely present on the other tasks. The intuition is that, since the observations from the other tasks are not used in the prediction, their presence or absence is irrelevant to the given task; see Helterbrand and Cressie [1994], Chilès and Delfiner [1999, §5.6.4] and Williams et al. [2007, Proposition 2]. However, if the observations are noisy, or if the observations are not isotopic, then this result on the cancellation of transfer will not hold. For example, this cancellation is not present in the simple violation of isotopicity considered in the next section.

3.9.2 Multi-collocated Observations

We now consider noise-free multi-collocated data [Chilès and Delfiner, 1999, §5.4.3], which is perhaps best explained using Figure 3.15b. In this case, in addition to the observations for isotopic data (Figure

3.15a), we have observations for the secondary tasks at $\boldsymbol{x}_*$, i.e., $X_1 \equiv \cdots \equiv X_M \equiv X_T \cup \{\boldsymbol{x}_*\}$. We may partition K^{x}_{ii} in the manner of

$$K^{\mathrm{x}} \overset{\mathrm{def}}{=} K^{\mathrm{x}}_{11} = \cdots = K^{\mathrm{x}}_{MM} = \begin{pmatrix} k^{\mathrm{x}}_{**} & (\boldsymbol{k}^{\mathrm{x}}_*)^{\mathrm{T}} \\ \boldsymbol{k}^{\mathrm{x}}_* & K^{\mathrm{x}}_{\setminus *} \end{pmatrix} \overset{\mathrm{def}}{=} \begin{pmatrix} k^{\mathrm{x}}_{**} & (\boldsymbol{k}^{\mathrm{x}}_{T*})^{\mathrm{T}} \\ \boldsymbol{k}^{\mathrm{x}}_{T*} & K^{\mathrm{x}}_{TT} \end{pmatrix}, \tag{3.73}$$

in which we have also defined the shorthands K^{x}, $\boldsymbol{k}^{\mathrm{x}}_{T*}$ and $K^{\mathrm{x}}_{\setminus *}$. For the observations, we denote those at X_T for task T with $\boldsymbol{y}_{T\setminus *}$, those at the $X_i \setminus \{\boldsymbol{x}_*\}$ for task S_i with $\boldsymbol{y}_{i\setminus *}$ and those at $\boldsymbol{x}_*$ for task S_i with y_{i*}. The full sequence of observations at all the secondary tasks S_is is $\boldsymbol{y}_S$ defined using

$$\boldsymbol{y}_{S*} \overset{\mathrm{def}}{=} \begin{pmatrix} y_{1*} & \ldots & y_{M*} \end{pmatrix}^{\mathrm{T}} \tag{3.74a}$$

$$\boldsymbol{y}_{S\setminus *} \overset{\mathrm{def}}{=} \begin{pmatrix} \boldsymbol{y}^{\mathrm{T}}_{1\setminus *} & \cdots & \boldsymbol{y}^{\mathrm{T}}_{M\setminus *} \end{pmatrix}^{\mathrm{T}} \tag{3.74b}$$

$$\boldsymbol{y}_{S} \overset{\mathrm{def}}{=} \begin{pmatrix} y_{S_1 *} & \boldsymbol{y}^{\mathrm{T}}_{1\setminus *} & \cdots & y_{M*} & \boldsymbol{y}^{\mathrm{T}}_{M\setminus *} \end{pmatrix}^{\mathrm{T}}. \tag{3.74c}$$

This provides an input-major ordering of observations for the secondary tasks. These definitions are consistent with those from the preceding section. The posterior mean and covariance of the multi-collocated case may be obtained by (a) conditioning on all observations except for those at $\boldsymbol{x}_*$ for the secondary tasks, and then (b) conditioning on those at $\boldsymbol{x}_*$ for the secondary tasks. Step (a) is equivalent to using isotopic observations, and, for this, the posterior distribution at $\boldsymbol{x}_*$ for all tasks is Gaussian with mean and covariance given by equation 3.72. Step (b) will use this intermediate posterior distribution and further condition on observations at $\boldsymbol{x}_*$ for the secondary tasks. Thus, we begin from equation 3.72 for the isotopic observations case, which is

$$\begin{pmatrix} f_T(\boldsymbol{x}_*) \\ \boldsymbol{f}_S(\boldsymbol{x}_*) \end{pmatrix} \Bigg| \boldsymbol{y}_{T\setminus *}, \boldsymbol{y}_{S\setminus *} \sim \mathcal{N}\left(\begin{pmatrix} \hat{y}_{T*} \\ \hat{\boldsymbol{y}}_{S*} \end{pmatrix}, \begin{pmatrix} 1 & \boldsymbol{\rho}^{\mathrm{T}} \\ \boldsymbol{\rho} & K^{\mathrm{f}}_S \end{pmatrix} \sigma^2_* \right), \tag{3.75}$$

where we have introduced the vector function for secondary tasks $\boldsymbol{f}^{\mathrm{T}}_S(\cdot) \overset{\mathrm{def}}{=} (f_1(\cdot), \ldots, f_M(\cdot))$, and substituted (3.71) for K^{f}. Since the observations are noise-free, we condition the above distribution on $\boldsymbol{f}_S(\boldsymbol{x}_*) = \boldsymbol{y}_{S*}$, giving

$$\mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\setminus *}, \boldsymbol{y}_S) = \hat{y}_{T*} + \boldsymbol{\rho}^{\mathrm{T}}(K^{\mathrm{f}}_S)^{-1} (\boldsymbol{y}_{S*} - \hat{\boldsymbol{y}}_{S*}) \tag{3.76a}$$

$$\mathbb{V}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\setminus *}, \boldsymbol{y}_S) = \upsilon^2_T \sigma^2_*, \tag{3.76b}$$

where

$$\upsilon^2_T \overset{\mathrm{def}}{=} 1 - \boldsymbol{\varrho}^{\mathrm{T}}(K^{\mathrm{f}}_S)^{-1}\boldsymbol{\varrho} \in [0, 1], \tag{3.77a}$$

and σ^2_* is given by (3.72e). The above expressions give an understanding of asymmetric multi-task learning by way of correction factors. The posterior mean (3.76a) depends on $(\hat{\boldsymbol{y}}_{S*})$, of which the ith entry $\hat{y}_{i*}$ is the prediction by task S_i at $\boldsymbol{x}_*$ based on its observations at $X_i \setminus \{\boldsymbol{x}_*\}$. The difference $\boldsymbol{y}_{S*} - \hat{\boldsymbol{y}}_{S*}$ is thus the vector of predictive errors at $\boldsymbol{x}_*$ made *independently* by the secondary tasks. The posterior mean at $\boldsymbol{x}_*$ for task T expressed as (3.76a) is $\hat{y}_{T*}$, which is the posterior mean by primary task T made independently of the secondary tasks (i.e., conditioned only on $\boldsymbol{y}_{T\setminus *}$), corrected with the predictive errors made by the secondary tasks, where the correction $\boldsymbol{\rho}^{\mathrm{T}}(K^{\mathrm{f}}_S)^{-1} (\boldsymbol{y}_{S*} - \hat{\boldsymbol{y}}_{S*})$ can be understood as the posterior mean of the predictive errors. This idea has been expressed succinctly by Chilès and Delfiner [1999, §5.4.3] as “cokriging ‘learns’ from kriging errors”.

The posterior variance (3.76b) is the product of (a) v_T^2, which is a Schur complement of K^{f} and may be interpreted as the variance of the primary task T conditioned on the secondary tasks S_is; and (b) σ_*^2, which for any single task, is the variance at $\boldsymbol{x}_*$ conditioned on observations at X_T or $X_i \setminus \{\boldsymbol{x}_*\}$. The following mnemonic may be useful:

$$\begin{aligned}\text{posterior variance} &= \text{posterior variance due to task correlation}\\ &\qquad\qquad\qquad\qquad \times\ \text{posterior variance due to input covariance}\end{aligned}$$

It is also clear that $v_T^2\sigma_*^2 \leqslant \sigma_*^2$, i.e., posterior variance is reduced when observations at other tasks are available.

The above result generalizes the co-kriging example of Chilès and Delfiner [1999, §5.4.2]. That example is for two tasks and two observed locations arranged in a multi-collocated manner. Here, we have generalized to multiple observed locations and multiple tasks. To the best of our knowledge, this generalization is novel.

The next section will look at a practical simplification of multi-collocated data.

3.9.3 Collocated Observations

The use of multi-collocated (noise-free) observations typically causes matrix instability in implementation. To avoid this, Xu et al. [1992] have suggested using only collocated observations, which are observations at $\boldsymbol{x}_*$ for the secondary tasks along with any observations for the primary task T. This is illustrated in Figure 3.15c. We shall use the Markovian nature of the multi-task model to obtain the posterior Gaussian distribution of $f_T(\boldsymbol{x}_*)$ given collocated observations.

To be consistent with the definitions for the case of isotopic and multi-collocated observations in the previous two sections, we use $\boldsymbol{y}_{T\setminus *}$ for the training observations at X_T for the primary task T and $\boldsymbol{y}_{S*}$ for the training observations at $X_1 \equiv \cdots X_M \equiv \{\boldsymbol{x}_*\}$ for all the secondary tasks. If $\boldsymbol{y}_{T\setminus *}$ and $\boldsymbol{y}_{S*}$ were the observations, then by the autokrigeability of the model discussed in section 3.9.1, we have

$$p(\boldsymbol{y}_{T\setminus *} \mid f_T(\boldsymbol{x}_*), \boldsymbol{y}_{S*}) = p(\boldsymbol{y}_{T\setminus *} \mid f_T(\boldsymbol{x}_*)). \tag{3.78}$$

One may say that $f_T(\boldsymbol{x}_*)$ screens the influence of $\boldsymbol{y}_{S*}$ on $\boldsymbol{y}_{T\setminus *}$. The above probability statement is equivalent to

$$p(\boldsymbol{y}_{T\setminus *}, \boldsymbol{y}_{S*} \mid f_T(\boldsymbol{x}_*)) = p(\boldsymbol{y}_{T\setminus *} \mid f_T(\boldsymbol{x}_*))\, p(\boldsymbol{y}_{S*} \mid f_T(\boldsymbol{x}_*)). \tag{3.79}$$

That is, $\boldsymbol{y}_{T\setminus *}$ and $\boldsymbol{y}_{S*}$ are independent given $f_T(\boldsymbol{x}_*)$, i.e., a Markov property. This property is known to be a characteristic of the separable covariance function $K^{\mathrm{f}}k^{\mathrm{x}}(\cdot,\cdot)$ [Xu et al., 1992; Almeida and Journel, 1994; Goovaerts, 1997; O'Hagan, 1998].

Using Bayes' theorem to invert (3.79) gives

$$p(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\setminus *}, \boldsymbol{y}_{S*}) \propto p(\boldsymbol{y}_{T\setminus *} \mid f_T(\boldsymbol{x}_*))\, p(\boldsymbol{y}_{S*} \mid f_T(\boldsymbol{x}_*))\, p(f_T(\boldsymbol{x}_*)) \tag{3.80}$$

The left of the equation is our goal, while the right is a product of three Gaussian distributions. By completing the squares in the exponent, we obtain the following posterior mean and variance for $f_T(\boldsymbol{x}_*)$:

$$\mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\backslash *}, \boldsymbol{y}_{S*}) = \sigma_{\mathrm{c}}^2 \left\{ \frac{1}{\sigma_*^2} \mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\backslash *}) + \frac{1}{k_{**}^{\mathrm{x}} \upsilon_T^2} \mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{S*}) \right\} \tag{3.81a}$$

$$\mathbb{V}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\backslash *}, \boldsymbol{y}_{S*}) = \sigma_{\mathrm{c}}^2 \tag{3.81b}$$

where

$$\mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\backslash *}) = (\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} (K_{\backslash *}^{\mathrm{x}})^{-1} \boldsymbol{y}_{T\backslash *} \tag{3.81c}$$

$$\mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{S*}) = \boldsymbol{\rho}^{\mathrm{T}} (K_S^{\mathrm{f}})^{-1} \boldsymbol{y}_{S*} \tag{3.81d}$$

$$\frac{1}{\sigma_{\mathrm{c}}^2} = \frac{1}{\sigma_*^2} + \frac{1}{k_{**}^{\mathrm{x}} \upsilon_T^2} - \frac{1}{k_{**}^{\mathrm{x}}} \tag{3.81e}$$

where, using definitions consistent with the preceding two sections, $K_{\backslash *}^{\mathrm{x}} \stackrel{\mathrm{def}}{=} K_{TT}^{\mathrm{x}}$ and $\boldsymbol{k}_*^{\mathrm{x}} \stackrel{\mathrm{def}}{=} \boldsymbol{k}_{T*}^{\mathrm{x}}$, and σ_*^2 and υ_T^2 are given by (3.72e) and (3.77a) respectively. The derivation is in appendix B.10. Chilès and Delfiner [1999, §5.4.3], have given similar expressions for the mean and variance, though only for the two-task and two-observed-locations case and not in its entire generality as we do here.

One can easily verify that $\upsilon_T^2 \sigma_*^2 / \sigma_{\mathrm{c}}^2 \leqslant 1$, so the multi-collocated case has lower variance than the collocated one. Hence, the additional observations in the multi-collocated case do provide additional information to predicting $f_T(\boldsymbol{x}_*)$, and one may say that $\boldsymbol{y}_{T\backslash *}$ and $\boldsymbol{y}_{S*}$ *do not* screen $\boldsymbol{y}_{S\backslash *}$ from $f_T(\boldsymbol{x}_*)$, where $\boldsymbol{y}_{S\backslash *}$ defined by (3.74c) forms part of the observations in the multi-collocated case. This clarifies that the Markov nature of the multi-task GP model differs from that of a Gaussian Markov random field [Rue and Held, 2005].

As noted by Chilès and Delfiner [1999, §5.4.3], the posterior mean (3.81a) can be seen as the variance-weighted linear combination of the predictions $\mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\backslash *})$ and $\mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{S*})$, since

$$\mathbb{V}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\backslash *}) = \sigma_*^2 \qquad\qquad \mathbb{V}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{S*}) = k_{**}^{\mathrm{x}} \upsilon_T^2. \tag{3.82}$$

It is also easy to verify that

$$1 \leqslant (1 + \sigma_{\mathrm{c}}^2 / k_{**}^{\mathrm{x}}) \leqslant \sigma_{\mathrm{c}}^2 \left(1/\sigma_*^2 + 1/k_{**}^{\mathrm{x}} \upsilon_T^2\right) \leqslant 2. \tag{3.83}$$

Thus, perhaps surprisingly, the posterior mean is *not* a convex combination of $\mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\backslash *})$ and $\mathbb{E}(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{S*})$.

3.10 Conclusions

In this chapter, we have measured the influence of the secondary task on the primary task using the generalization error and the learning curve, parameterizing these with the correlation ρ between the two tasks, and the proportion π_S of observations for the secondary task. We have provided bounds on the generalization error and learning curves, and these bounds highlight the effects of ρ and π_S. In particular, we have found the lower bound $\underline{\sigma}_T^2$ on the posterior variance rather tight and useful for subsequent analysis. The development of the FWO$_{\hat{\varrho}}$ upper bound on the learning curve has provided further insights into multi-task learning. We have also explored the structure in the multi-task GP predictions by using isotopic, multi-collocated and collocated noiseless observations, generalizing two known results from the geostatistics literature.

Our analysis on the degenerate case of no training data for the primary task has uncovered an intrinsic limitation of multi-task GP: the error cannot be lower than $(1-\rho^2)\sum\kappa_i$ if one learns with only observations from the secondary task. For symmetric multi-task learning, the term $\pi_S(1-\pi_S)(1-\rho^2)$ is demonstrated to have a governing role. For asymmetric multi-task learning, we have shown that the lower bound on the learning curve cannot be less than $(1-\rho^2\pi_S)$ of that from single-task learning; this is found to be satisfied for the learning curve in the empirical evaluation with the Sarcos data. For both symmetric and asymmetric multi-task learning, we have shown theoretically and experimentally that the secondary task has more influence on the primary task for smoother processes rather than for rougher processes. The effective number of observations that the secondary task contributes to the primary task has also been investigated.

This chapter is a step towards understanding the role of the matrix K^{f} of inter-task similarities in multi-task GPs with more than two tasks. We believe it has contributed to an understanding of multi-task learning that is orthogonal to the existing PAC-based results in the literature.

Chapter 4

Multi-task Gaussian Process Learning of Robot Inverse Dynamics

4.1 Overview

In this chapter we are concerned with the problem of controlling a robot manipulator (i.e., a multi-jointed robot arm) to follow a given trajectory; this is known as the *inverse dynamics* problem. We consider a robot manipulator with ℓ revolute joints. Denote the joint angles by $\boldsymbol{q} \stackrel{\text{def}}{=} (q_1, \ldots, q_\ell)^{\mathrm{T}}$, and, similarly, the joint velocities and accelerations by $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$. Let $\boldsymbol{x} \stackrel{\text{def}}{=} (\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})^{\mathrm{T}} \in \mathbb{R}^{3\ell}$. The aim is to learn (or estimate) the inverse dynamics of the robot from data; that is, to learn the ℓ torque functions $\boldsymbol{\tau}(\boldsymbol{x})$, $\boldsymbol{\tau} : \mathbb{R}^{3\ell} \mapsto \mathbb{R}^{\ell}$.

It may seem unnecessary to estimate $\boldsymbol{\tau}(\boldsymbol{x})$, given knowledge of the physics of the robot. Indeed, for a simple and highly structured robot manipulator, it is often possible to find an analytical form for the input/output mapping that is needed to compute the torques, for example using inverse models based on rigid body dynamics derived from the Newton-Euler algorithm [Featherstone, 1987]. These models are parameterized in terms of kinematic and dynamic parameters. The latter, which include the mass, centre of mass and moments of inertia of each link, are usually unknown even to the manufacturers of the robots [An et al., 1988]. The calibration of these dynamic parameters is neither trivial nor robust; for example, Armstrong et al. [1986] estimated them for a PUMA 560 arm by disassembling it and measuring the properties of the individual links using a set of rather elaborate procedures, and Corke and Armstrong-Hélouvry [1994] have noted 200% to 400% variations in the parameters of PUMA 560 arm reported in the literature. Some dynamic parameters, such as those for friction, may also vary with time, rendering previously estimated values less useful.

In addition to difficulties caused by unknown physical parameters, there are difficulties in modelling interference with cables, joint elasticity, friction and contact forces, so that analytical predictions can be infeasible. This is particularly the case for robots with designs that make analytical modelling hard, for example the control of compliant, lightweight humanoid and personal robots, which deviate significantly from idealized rigid body dynamics.

To overcome the above limitations of analytical models, there is an increasing emphasis on adaptive controllers that enable the automatic calibration of robots based directly on input-output data, ideally with minimal human involvement [An et al., 1988]. In general, there are two classes of adaptive controllers: parametric adaptive controllers, which use analytical models but change the dynamic parameters to match observed data; and non-parametric adaptive controllers, which learn the mappings directly through function approximation [Burdet and Codourey, 1998]. In this chapter, we follow the latter route and place Gaussian process (GP) priors directly on the torque functions. This approach gives a model abstract enough to be generally applicable and not limited to a particular robot manipulator.

The adaptation or learning of the inverse dynamics is especially necessary for robot manipulators, which, unlike traditional machine tools, are required to have high dexterity, handle a variety of loads, and perform a number of different tasks. In general, each different circumstance or *context*, such as changing the load, will correspond to a different inverse dynamics. If the inverse dynamics changes abruptly and/or frequently when, say, manipulating a set of tools, then it is appropriate to use different inverse dynamics mappings, one for each of the many contexts. A classic adaptive controller that continuously changes its inverse dynamics mapping to match observed dynamics is inadequate in this case, since it has only a single mapping at any one time [Petkos et al., 2006]. For learning under such multiple contexts, we propose the multi-task Gaussian process model described in section 2.5.1 on page 30, which is capable of handling multiple inverse dynamics mappings. In addition, being fully probabilistic, it can leverage the multiple inverse dynamics mappings by "sharing statistical strength" among them.

Although it is desirable, it is not necessary for the estimated inverse dynamics functions to be fully accurate, since they will typically be used as the feedforward component within a *composite controller* [An et al., 1988, §1.3.2], as illustrated in Figure 4.1. In this controller, the vector of actuation torques $\boldsymbol{\tau}(\boldsymbol{x})$ is a sum of feedforward torques $\boldsymbol{\tau}_{\mathrm{ff}}(\boldsymbol{x})$ and a corrective feedback component $\boldsymbol{\tau}_{\mathrm{fb}}(\boldsymbol{x})$. The latter is required for disturbance suppression and for dealing with unforeseen deviations from the model. Thus we have $\boldsymbol{\tau}(\boldsymbol{x}) = \boldsymbol{\tau}_{\mathrm{ff}}(\boldsymbol{x}) + \boldsymbol{\tau}_{\mathrm{fb}}(\boldsymbol{x}) = \boldsymbol{\tau}_{\mathrm{ff}}(\boldsymbol{x}) + (K_{\mathrm{p}}\boldsymbol{e} + K_{\mathrm{d}}\dot{\boldsymbol{e}})$, where $K_{\mathrm{p}},\ K_{\mathrm{d}}$ are feedback gains for a proportional-derivative (PD) controller, and $\boldsymbol{e}$ is the deviation from the desired trajectory. If $\boldsymbol{\tau}_{\mathrm{ff}}$ is accurate there will be little feedback correction required, allowing us to use low feedback gains and yet achieving fast, accurate movements. Low gains imply high compliance, that is if the movement is suddenly obstructed, the system will not generate unduly large and dangerous corrective commands. This is desirable for robots meant to operate amongst humans.

In summary, we present in this chapter a probabilistic model based on multi-task Gaussian processes (see Bonilla et al. [2008] and section 2.5.1 on page 30) for learning the feedforward torques across multiple contexts. Using a nonparametric approach, we obtain an abstract model that does not (from the outset) model the specifics of an individual robot, but is generally applicable to families of robots. The proposed model is capable of exploiting the commonalities between different contexts in order to give improved performance for a given context, compared to learning only on data from that context. Through coupling inference and learning across multiple contexts, the model propagates uncertainties automatically between the different contexts. This is particularly advantageous when the data for each context explores rather different portions of $\boldsymbol{x}$-space.

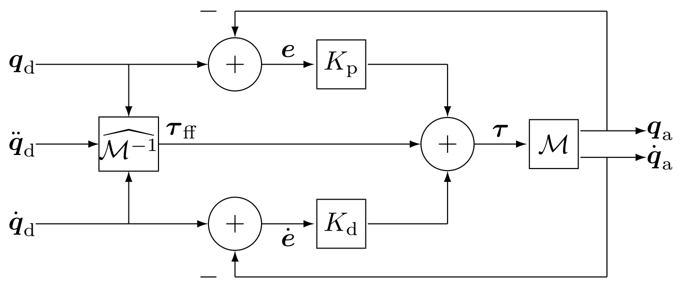

$\boldsymbol{q}_{\mathrm{d}}$
$\ddot{\boldsymbol{q}}_{\mathrm{d}}$
$\dot{\boldsymbol{q}}_{\mathrm{d}}$
$-$
$\boldsymbol{e}$
K_{p}
$\widehat{\mathcal{M}^{-1}}$
$\boldsymbol{\tau}_{\mathrm{ff}}$
$\boldsymbol{\tau}$
$\mathcal{M}$
$\boldsymbol{q}_{\mathrm{a}}$
$\dot{\boldsymbol{q}}_{\mathrm{a}}$
$\dot{\boldsymbol{e}}$
K_{d}
$-$

Figure 4.1: Feedforward composite controller, adapted from An et al. [1988, Figure 1.5]. The triplet $(\boldsymbol{q}_{\mathrm{d}}, \dot{\boldsymbol{q}}_{\mathrm{d}}, \ddot{\boldsymbol{q}}_{\mathrm{d}})$ gives the desired trajectory, while $(\boldsymbol{q}_{\mathrm{a}}, \dot{\boldsymbol{q}}_{\mathrm{a}})$ is the current state of the actual trajectory. The actual dynamics model of the system is denoted by $\mathcal{M}$, while the estimated inverse dynamics model is denoted by $\widehat{\mathcal{M}^{-1}}$. The vector of torques $\boldsymbol{\tau}_{\mathrm{ff}}$ is computed by the feedforward component using $\widehat{\mathcal{M}^{-1}}$, and it is corrected with the PD feedback to give the vector of actuation torques $\boldsymbol{\tau}$.

4.1.1 Outline

The rest of this chapter is structured as follows. In section 4.2 we introduce the inverse dynamics data used for building and assessing our models, and study some characteristics of the data. In particular, we highlight the presence of discontinuities in the inverse dynamics function that will influence our modelling later.

Our model is based on placing Gaussian process priors on the torque functions. Section 4.3 introduces and motivates our particular GP model for a single context, and compares it with three simpler models. In section 4.4, the GP model is extended to the multi-task GP model to learn from multiple contexts, using the property that the inverse dynamics function is linear with respect to dynamic parameters. Experimental results illustrating the benefits of the extended model are provided. Lastly, section 4.5 summarizes our findings and provides some discussions.

4.2 Data Collection and Exploration

In this section, we first describe the data used, then explore some characteristics of the data, and end with how the data is sampled for the later experiments.

4.2.1 Data

We use a realistic simulation of the industrial grade revolute robot PUMA 560 (Programmable Universal Machine for Assembly). This robot is used widely in industry, and has been well studied in the robotics literature. It has $\ell = 6$ degrees of freedom, nominal payload of 4kg, maximum speed of 1.0m/s and extended limb length of 0.878m. This robot is a “general purpose robot” because it has six degrees of freedom, the minimum required for manipulation in three-dimensional space.

The simulation models both viscous and asymmetric-Coulomb frictional forces, and uses realistic iner-

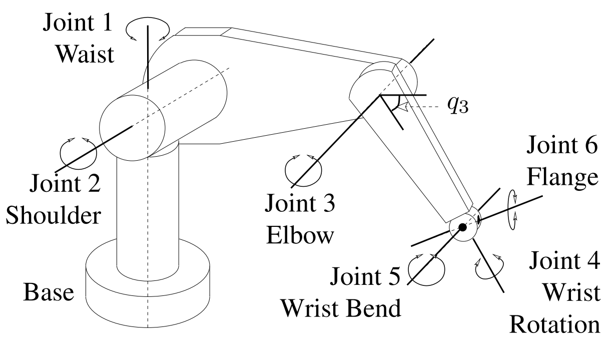


Figure 4.2: Schematic of the PUMA 560 without the end-effector (to be connected to joint 6).

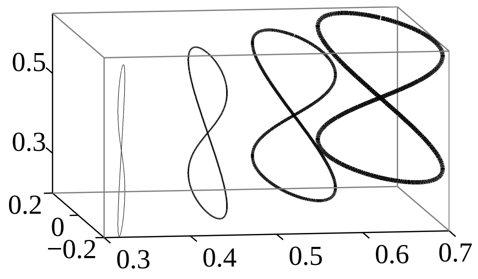


Figure 4.3: Paths p_1 to p_4 for the data set. The robot base is located at $(0, 0, 0)$.

Table 4.1: Speeds of trajectories

Speeds	Time to complete path/s
s_1	20
s_2	15
s_3	10
s_4	5

Table 4.2: Paths of trajectories

	Centre		Rotation
Paths	along x-axis/m	along z-axis/m	about z-axis/°
p_1	0.35	0.36	-10
p_2	0.45	0.40	0
p_3	0.55	0.44	10
p_4	0.65	0.48	20

tial parameters [Corke, 1996; Corke and Armstrong-Hélouvry, 1994].[1] Our aim is to model the inverse dynamics for a single robot for use in a composite controller, hence we do not introduce any noise or parameter uncertainty in the simulation.

Robot manipulators are used for different tasks, handling different loads. We explore this circumstance by simulating the handling of $M = 15$ different loads at the end-effector through 4 different figure-of-eight paths at 4 different speeds. Figure 4.3 shows the paths $p_1, \dots, p_4$, which are placed at 0.35m to 0.65m along the x-axis, at 0.36m to 0.48m along the z-axis, and rotated about the z-axis by $-10°$ to $20°$. There are four speeds, denoted by $s_1, \dots, s_4$, with s_1 completing a path in 20s, and s_4 in 5s. Details are given in Tables 4.1 and 4.2.

In general, loads can have very different physical characteristics; in our case, this is done by representing each load as a cuboid with differing dimensions and masses, and attaching each load rigidly to a random point at the end-effector. The masses used are given in Table 4.3; the other parameters are omitted.

For each path-speed (trajectory) combination $(p_\cdot, s_\cdot)$, and for each load c_m, 4000 data points are sampled at regular intervals along the path. Each sample is the pair $(\boldsymbol{x}, \boldsymbol{t})$, where $\boldsymbol{t} \in \mathbb{R}^\ell$ are the observed torques at the joints, and $\boldsymbol{x} \in \mathbb{R}^{3\ell}$ are the joint angles, velocities and accelerations.

Table 4.3: Mass/kg of the contexts or loads

c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}
0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0	2.2	2.4	2.6	2.8	3.0

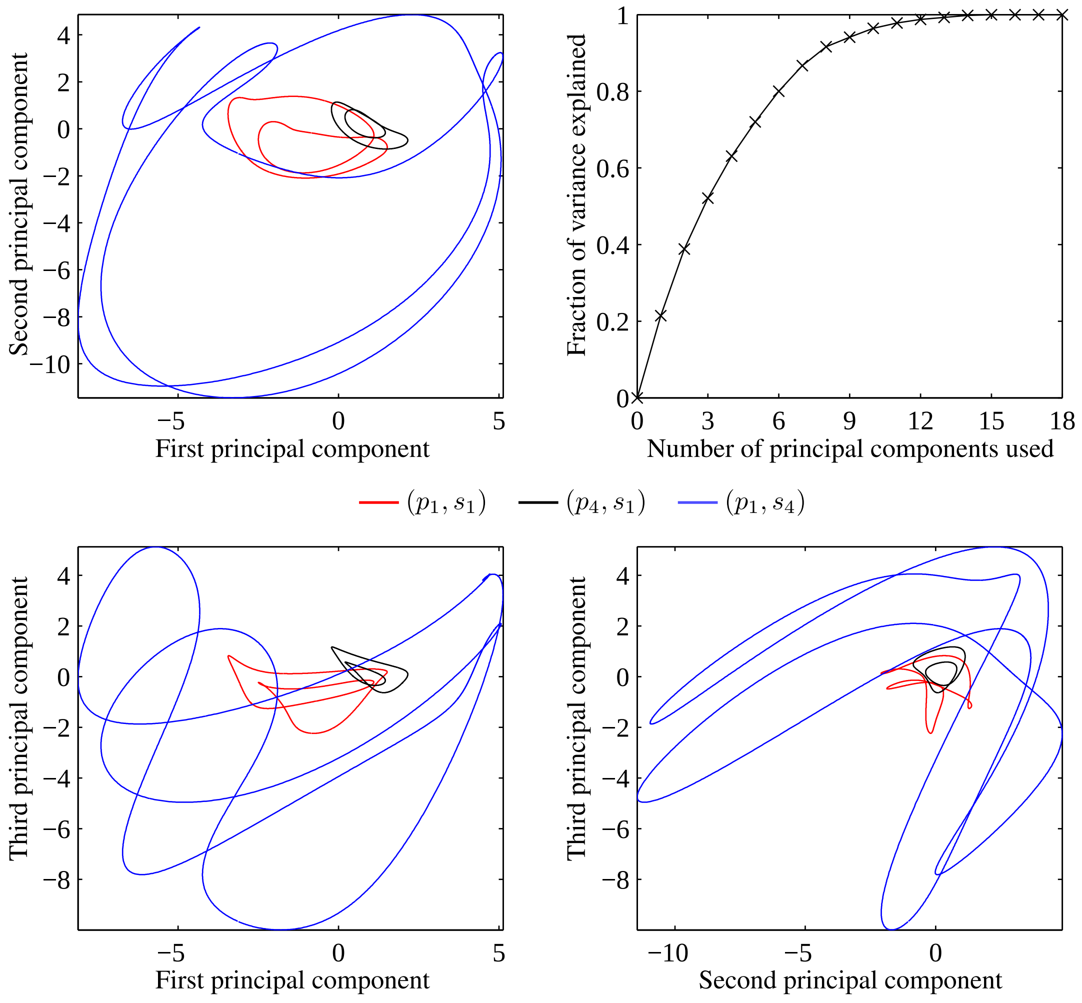
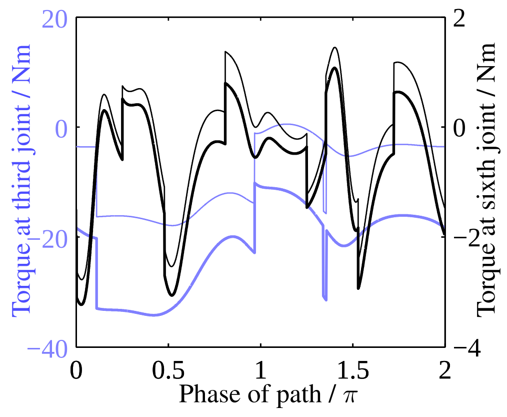
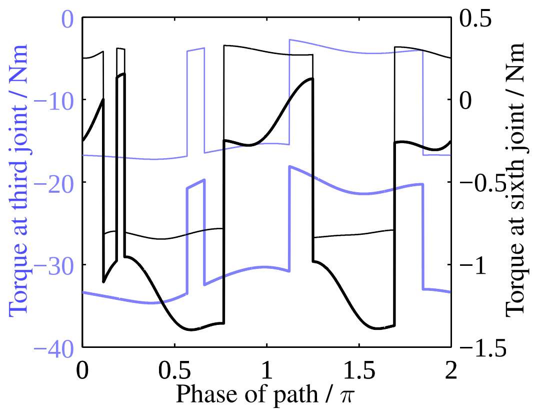


Figure 4.4: Plots of the three trajectories projected onto the first 3 principal components, and plot of the proportion of variance explained against the number of principal components.



(a) Torque versus phase plot for (p_1, s_4)



(b) Torque versus phase plot for (p_4, s_1)

Figure 4.5: Example plots of torques versus the phase of a path. The thin lines are for load c_1, and the thick ones for c_{15}; the blue lines are for the torques at the third joint, and the black ones for the torques at the sixth joint.

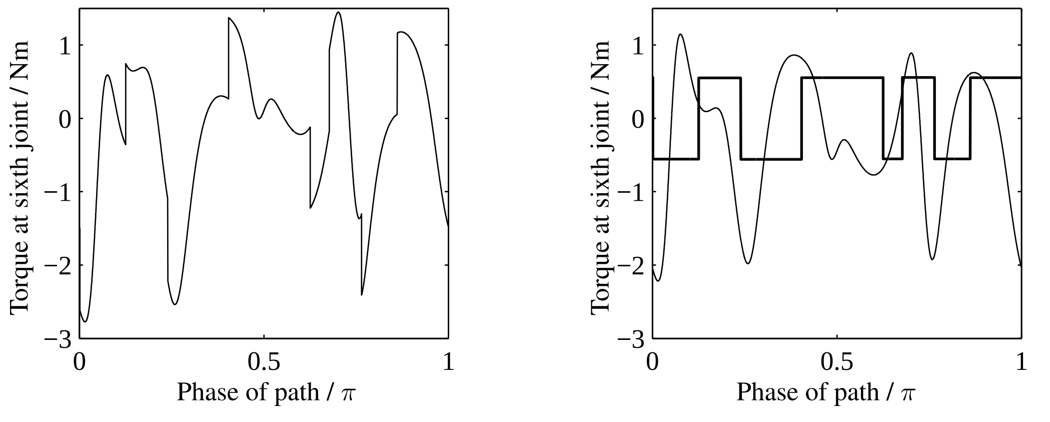
4.2.2 Data Analysis

To help visualize the multidimensional data, principal component analysis (PCA) is performed on all the sampled inputs $\{\boldsymbol{x}\}$. Figure 4.4 plots for selected path-speed combinations the inputs projected onto the first three principal components, which account for about 50% of the variance. It shows that p_1 explores larger variations in the possible space more than p_4. This is because p_1 is closer to the robot base than p_4, and therefore the arm requires larger joint movements to cover similar areas in the Cartesian space. Also, s_4 explores larger variations than s_1 because of the larger changes in velocities and accelerations needed to complete a path faster. For example, if (p_1, s_1) traces $(\boldsymbol{q}(\phi), \dot{\boldsymbol{q}}(\phi), \ddot{\boldsymbol{q}}(\phi))$ in the $\boldsymbol{x}$-space, $\phi \in [0, 2\pi]$ being the phase of the path, then (p_1, s_4) traces $(\boldsymbol{q}(\phi), 4\dot{\boldsymbol{q}}(\phi), 16\ddot{\boldsymbol{q}}(\phi))$, since s_4 completes the path 4 times faster than s_1.

Figure 4.4 makes clear that the inputs have structure reflecting the trajectories, and that different trajectories will explore different parts of the $\boldsymbol{x}$-space, although there is a certain degree of overlap. This suggests that a model of inverse dynamics learnt for one path and one speed may not fit the inverse dynamics for another path and/or another speed. In this case, it should be advantageous to combine data for different trajectories using, e.g., the multi-task GP proposed in section 4.4. Note that performing PCA on simple transformed spaces, such as trigonometric functions of the angles or sigmoid transformations of the velocities, will give the same qualitative observations, although the plots may be less contorted.

Figure 4.5 gives examples of how the torques at the joints vary with phase ϕ as the robot arm traces a figure-of-eight path. Within the same trajectory, we see that the torques for the different loads are similar, the most noticeable difference being a vertical offset along the torque axes. This suggests that it will be beneficial to use a multi-task model for the torques for different loads. In contrast, the torques for different joints are quite dissimilar, so that modelling these with multi-task models may not be as

[1] The simulation package is available from `http://www.petercorke.com/Robotics%20Toolbox.html`.



(a) Torque versus phase plot

(b) Torque components versus phase plot

Figure 4.6: Example of the decomposition of a torque into its components. (a) Torque versus phase plot for trajectory (p_1, s_4), load c_1 and τ_6; (b) The same torque is decomposed into a square wave and a smooth function.

beneficial.

The torques in Figure 4.5 display discontinuities, due to Coulomb friction. This is a constant force opposing the current direction of movement, so each time a joint velocity crosses zero, the corresponding torque jumps.[2] The visibly large effects of the Coulomb friction on the torques suggest it will be important to model the friction. Figure 4.6 shows that a torque function $\tau_i(\boldsymbol{x})$ can be effectively modelled using the sum of a square wave and a smooth function. We shall show in section 4.3.2 that $\tau_i(\boldsymbol{x})$ can be readily modelled by the sum of two GPs.

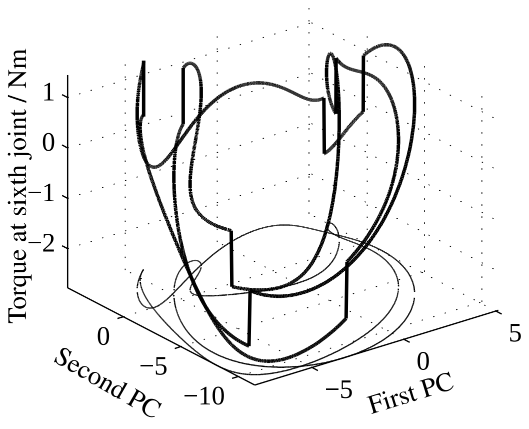
Figure 4.7 gives example plots of the torque as a function of the first two principal components, making it clear that the discontinuities occur also in the torque-$\boldsymbol{x}$ space. An artifact of the projection used in the plots may suggest that a given $\boldsymbol{x}$ can produce two different torques. However, the inverse dynamics is really a function, not a relation.

4.2.3 Data Sampling for Experiments

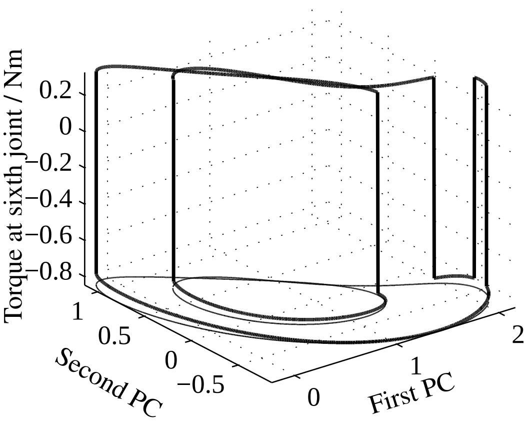
It may be too expensive to acquire data combinatorially, sampling data for every possible load-trajectory pair. However, one may imagine that data for the handling of a load can be obtained along a fixed reference trajectory for calibration purposes, and also along a trajectory typical for that load. We explore such a scenario here, selecting the training samples in the manner described below.

For each load, 2000 training samples are acquired at a common reference trajectory T_r, which is (p_2, s_3). In addition to this, 2000 training samples are acquired at a trajectory unique to each load, denoted by T_m for the $m\mathrm{th}$ load. Table 4.4 gives the combinations. Hence each load has a training set of 4000 samples, but acquired only on two different trajectories. These training samples are random subsamples of the data described in section 4.2.1.

[2] Since the data is generated by calculating inverse dynamics from a smooth trajectory, the jumps in the torque sequence are more visible than for e.g. PD control, where the discontinuity would be “shared” among torques and accelerations.



(a) Torque versus first two principal components for (p_1, s_4)



(b) Torque versus first two principal components for (p_4, s_1)

Figure 4.7: Example plots of torques versus the first two principal components (PCs) of the inputs $\boldsymbol{x}$. The thick line is the torque of the sixth joint, and thin line is the projection of this onto the plane of the first two principal components.

Table 4.4: The path-speed combinations at which training samples for each load are acquired, marked with ■ or □. For the multiple-contexts setting, c_{15}, and hence T_{15}, is not used for training.

	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}
$T_1 : p_1, s_1$	■														
$T_6 : p_2, s_1$						■									
$T_{11} : p_3, s_1$											■				
$T_2 : p_4, s_1$		■													
$T_7 : p_1, s_2$							■								
$T_{12} : p_2, s_2$												■			
$T_3 : p_3, s_2$			■												
$T_8 : p_4, s_2$								■							
$T_{13} : p_1, s_3$													■		
$T_\mathrm{r} : p_2, s_3$	■	■	■	■	■	■	■	■	■	■	■	■	■	■	□
$T_4 : p_3, s_3$				■											
$T_9 : p_4, s_3$									■						
$T_{14} : p_1, s_4$														■	
$T_5 : p_2, s_4$					■										
$T_{10} : p_3, s_4$										■					
$T_{15} : p_4, s_4$															□

Following Burdet and Codourey [1998], we use two kinds of test sets to assess our models for (a) control along a repeated trajectory (which is of practical interest in industry), and (b) control along arbitrary trajectories (which is of general interest to roboticists). The test for (a) assesses the accuracy of torque predictions for staying *within* the trajectories that were used for training. In this case, the test set for load c_m, denoted by *interp*$_m$ for *interpolation*, consists of the rest of the samples from T_r and T_m that were not used for training. The test for (b) assesses the accuracy also for *extrapolation* to trajectories not sampled for training. The test set for this, denoted by *extrap*$_m$, consists of all the samples (collated from all trajectories) that were not training samples for c_m.

In addition, we consider a data-poor scenario, and investigate the quality of the models using randomly selected subsets of the training data. The size of these subsets ranges from 20 to 1858. For each size, 5 subsets are sampled, and the experiments are replicated over these 5 subsets.

Remark Our data is obtained by sampling at regularly spaced intervals along the paths for all the speeds. Since each of our paths is a closed loop, this approximates the realistic case where the sampling is at a fixed rate for a real robot tracing the same path periodically.

Remark Recall that our inputs $\boldsymbol{x}$ for the inverse dynamics model are from the joint space, with $\boldsymbol{x} \stackrel{\mathrm{def}}{=} (\boldsymbol{q}^\mathrm{T}, \dot{\boldsymbol{q}}^\mathrm{T}, \ddot{\boldsymbol{q}}^\mathrm{T})^\mathrm{T}$. Rather than specifying inputs in the joint space directly, the $\boldsymbol{x}$s in our data are obtained by specifying the path of the end-effector positions in Cartesian coordinates of the operational space, and applying the inverse kinematics to obtain the joint variables. This end-effector control of manipulators is the natural approach for adaptive and compliance control in the work place, even though it precludes the use of techniques for designing trajectories that cover the entire range of dynamics in the joint space [Armstrong, 1987; Swevers et al., 1997].

4.3 Inverse Dynamics Model for a Single Context

In this section, we describe how inverse dynamics models in the joint space may be obtained under a single context, when the dynamics does not change. We first review the classic approach in robotics for obtaining dynamic models of a robot manipulator. This approach produces parametric formulae for computing the torques, in which the parameters are known as the dynamic parameters. Next, we discuss how we may use Gaussian process priors for learning the torque functions, incorporating different amounts of information into the priors. Our choice of prior based on general function approximation principles is detailed in section 4.3.2.1. In section 4.3.3, we outline how parameterized Gaussian process priors may be optimized to give better predictions. The merits of our prior are explored experimentally in section 4.3.4, together with three other simpler models. Finally, we give some related work for learning the inverse dynamics in a single context in section 4.3.5.

4.3.1 Analytical Models: Lagrangian Formulation

The classic approach to obtain a model for the dynamics of the robot is based on a Lagrangian formulation [Sciavicco and Siciliano, 2000, §4.1]

$$\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathcal{T}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \mathcal{U}(\boldsymbol{q}) \qquad\qquad \xi_i = \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i}, \tag{4.1}$$

where $\mathcal{L}$ is the Lagrangian of the system, $\mathcal{T}$ and $\mathcal{U}$ are the total kinetic energy and potential energy of the system respectively, and ξ_i is the torque for the $i\text{th}$ joint due the Lagrangian. The kinetic energy can be further expressed as $\mathcal{T} = \dot{\boldsymbol{q}}^{\mathrm{T}} B(\boldsymbol{q})\dot{\boldsymbol{q}}/2$, where $B(\boldsymbol{q})$ is the ℓ-by-ℓ *inertial matrix* of the system, and is positive definite. Denote by $\boldsymbol{b}_i(\boldsymbol{q})$ the $i\text{th}$ column of $B(\boldsymbol{q})$, and by $b_{ij}(\boldsymbol{q})$ the $(i,j)\text{th}$ entry of $B(\boldsymbol{q})$. Using a simplified friction model of only the symmetric-Coulomb and viscous frictions (see, for example, Makkar et al. [2005] for more detailed friction models), and assuming no contact by the end effector on the environment, we may write the actuation torque for the $i\text{th}$ joint as

$$\tau_i(\boldsymbol{x}) = \underbrace{\boldsymbol{b}_i^{\mathrm{T}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^{\mathrm{T}} H_i(\boldsymbol{q})\dot{\boldsymbol{q}}}_{\text{kinetic}} \quad + \quad \underbrace{g_i(\boldsymbol{q})}_{\text{potential}} \quad + \quad \underbrace{f_i^{\mathrm{visc}}\dot{q}_i + f_i^{\mathrm{clmb}}\,\mathrm{sgn}(\dot{q}_i)}_{\text{viscous and Coulomb frictions}}, \tag{4.2}$$

where sgn is the signum function; f_i^{clmb} and f^{visc} are Coulomb and viscous friction coefficients; $g_i(\boldsymbol{q}) \overset{\text{def}}{=} \partial\mathcal{U}(\boldsymbol{q})/\partial q_i$; and $H_i(\boldsymbol{q})$ is a ℓ-by-ℓ matrix of Coriolis and centrifugal effects, with the $(j,k)\text{th}$ entry given by $h_{ijk}(\boldsymbol{q}) \overset{\text{def}}{=} \partial b_{ij}(\boldsymbol{q})/\partial q_k - \frac{1}{2}\partial b_{jk}(\boldsymbol{q})/\partial q_i$. The required torque is therefore a sum of contributions due to kinetic energy, potential energy and frictional forces.

The analytical model (4.2) is parameterized by *inertial parameters* and *friction parameters*, collectively called *dynamic parameters*. There are 10 inertial parameters for each link, describing its mass ($\mathbb{R}$), centre of mass ($\mathbb{R}^3$) and moments of inertia ($\mathbb{R}^6$). Accurate estimation of these parameters is very important in analytical models. It is, however, rather involved; see a recent review by Schedlinski and Link [2001].

4.3.2 Gaussian Process Prior Models

Bayesian approaches to learning the inverse dynamics begin by specifying a prior over the torque functions $\boldsymbol{\tau} : \mathbb{R}^{3\ell} \mapsto \mathbb{R}^{\ell}$, either directly or through parameters. After conditioning on observed data, we obtain the posterior over the functions, which is then used subsequently for inference. For the $i\text{th}$ joint, let $\mathcal{D}_i \overset{\text{def}}{=} \{(\boldsymbol{x}^{(j)}, t_i^{(j)})\}_{j=1}^{n}$ be the observed set of n input-torque pairs, X be the set of inputs in $\mathcal{D}_i$, and $\boldsymbol{x}^*$ be the point for which inference is required. We write $\boldsymbol{t} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ to mean that the random vector $\boldsymbol{t}$ has multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance Σ. Then for a zero-mean Gaussian process prior $\tau_i \sim GP(0, k_i)$, where k_i is the covariance function, we have according to the prior the following multivariate-normal distribution

$$\begin{pmatrix} \boldsymbol{t}_i \\ \tau_i^* \end{pmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{pmatrix} K_i + \sigma_i^2 I & \boldsymbol{k}_i \\ \boldsymbol{k}_i^{\mathrm{T}} & k_i(\boldsymbol{x}^*, \boldsymbol{x}^*) \end{pmatrix}\right), \tag{4.3}$$

where τ_i^* is the torque at $\boldsymbol{x}^*$, $\boldsymbol{t}_i \overset{\text{def}}{=} (t_i^{(1)}, \ldots, t_i^{(n)})$ are observations with additive normal noise of variance σ_i^2, K_i is the matrix of covariances for all pairs in X, and $\boldsymbol{k}_i$ is the vector of covariances between $\boldsymbol{x}^*$ and X. Conditioned on the observation data $\mathcal{D}_i$, the posterior is

$$\tau_i^* | \boldsymbol{x}^*, \mathcal{D}_i \sim \mathcal{N}\left(\boldsymbol{k}_i^{\mathrm{T}}(K_i + \sigma_i^2)^{-1}\boldsymbol{t}_i,\; k_i(\boldsymbol{x}^*, \boldsymbol{x}^*) - \boldsymbol{k}_i^{\mathrm{T}}(K_i + \sigma_i^2)^{-1}\boldsymbol{k}_i\right). \tag{4.4}$$

The above expression for the GP posterior can be found in standard references for GPs, e.g., [O'Hagan, 1978; Rasmussen and Williams, 2006].

The prior on $\tau_i(\cdot)$ encodes beliefs about the function *before* observing the data. For zero-mean GPs this is specified by the covariance function $k_i(\cdot, \cdot)$. In the absence of strong evidence for otherwise, it is common to use vague priors to prevent a-priori ruling out reasonable explanations of the data [Rasmussen and Williams, 2006, §5.2]. The natural question is thus: how much information in a GP prior over torque functions is necessary for good predictive performance? One highly informative GP prior can be derived based on the detailed analysis in (4.2). For a given robot manipulator, explicit parameterized expressions for B, H_i and g_i can be obtained using, for example, computer algebra [Corke, 1998]. The resultant torque functions will be linear with respect to the dynamic parameters (see section 4.4.1). Placing Gaussian priors over these parameters will give a GP prior over the torque functions. This is a Bayesian linear-in-the-parameters model in which the regressors are functions of $\boldsymbol{x}$ instead of simply $\boldsymbol{x}$, and it is a model for parametric adaptive control. It is specialized for the given manipulator, and may be grossly mismatched when used for another. Given the non-generality of this approach, we shall shy away from such a prior.

A more general GP prior on $\tau_i(\boldsymbol{q})$ that still retains much of the information present in (4.2) involves placing GP priors over the functions $\boldsymbol{b}_i$ and g_i, and Gaussian priors over parameters f_i^{visc} and f_i^{clmb}.[3] There are two main difficulties with such an approach. The first is that it is non-trivial to model the positive definite matrix function $B(\boldsymbol{q})$. Similar to the Wishart distribution, one may define a prior over $B(\boldsymbol{q})$ by letting $B(\boldsymbol{q}) \overset{\mathrm{def}}{=} A(\boldsymbol{q})A^{\mathrm{T}}(\boldsymbol{q})$ and placing a matrix-variate GP prior on $A(\boldsymbol{q})$. Unfortunately, such a prior inherits all the restrictions of the Wishart distribution and at the same time complicates the model through its non-normality. The second difficulty is because $H_i(\boldsymbol{q})$, being derived from $B(\boldsymbol{q})$, has non-trivial correlation with $\boldsymbol{b}_i(\boldsymbol{q})$. This coupling makes the covariance function of $\tau_i(\boldsymbol{q})$ rather complicated. We have actually attempted priors over $\tau_i(\boldsymbol{q})$ similar to what is described here[4], and preliminary results (along the lines of the experiments described in section 4.3.4) show that they give rise to inferior performance relative to the more general prior described in the next paragraph. In essence, this approach is similar to the *latent force model* of Alvarez et al. [2009], which marries purely mechanistic models of physics with the data driven paradigm of machine learning.

A yet more general GP prior is one that models only the analytic properties of the torque functions. This is the prior that we favour, and that has produced predictive results better than the other GP models we have tried. This is described in the following section.

4.3.2.1 Gaussian process priors for function approximation

Our preferred choice of GP prior is based on the tenet that a GP prior is a prior over the space of functions, so that we can directly view learning the inverse dynamics as a *function approximation* problem. Within this view, we wish to express in the prior the analytic properties of the function to be estimated. This is done by first modelling the torque function without the Coulomb friction, and then the torque

[3] The prior on H_is follows directly from the prior on $\boldsymbol{b}_i$s, so there is no need to specify its prior.

[4] Examples of priors we have attempted are: (a) approximating $B(\boldsymbol{q})$ with a GP by matching first and second moments; and (b) assuming independence between $H_i(\boldsymbol{q})$ and $B(\boldsymbol{q})$ in addition to the approximation in (a).

caused by the Coulomb friction.

Consider the torque function $\tau_i^{\backslash\mathrm{clmb}}$ given by (4.2) except the last term, which models the Coulomb friction. By construction, $\tau_i^{\backslash\mathrm{clmb}}$ is analytic[5] with respect to $\boldsymbol{x}$, so that a GP prior over $\tau_i^{\backslash\mathrm{clmb}}$ should be *infinitely mean square differentiable* [Belyaev, 1959]. An additional property is the *nonstationarity* of the GP prior, due (at least) to the kinetic and viscous friction terms in (4.2). Under these considerations, we let $\tau_i^{\backslash\mathrm{clmb}} \sim GP(0, k_i^{\backslash\mathrm{clmb}})$, with the covariance between $\tau_i^{\backslash\mathrm{clmb}}(\boldsymbol{x}^{(\alpha)})$ and $\tau_i^{\backslash\mathrm{clmb}}(\boldsymbol{x}^{(\beta)})$ given by

$$k_i^{\backslash\mathrm{clmb}}(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)}) \stackrel{\mathrm{def}}{=} b_i^2 + \sum_{j=1}^{3\ell} u_{ij}^2 x_j^{(\alpha)} x_j^{(\beta)} + v_i^2 \exp\left(-\frac{1}{2} \sum_{j=1}^{3\ell} \left(\frac{x_j^{(\alpha)} - x_j^{(\beta)}}{l_{ij}} \right)^2 \right) + \sigma_i^2 \delta_{\alpha,\beta}, \quad (4.5)$$

where δ is the Kronecker delta function, and b_i, the u_{ij}s, v_i, the l_{ij}s and σ_i are unknown hyperparameters. The hyperparameters are subscripted with i so that the covariance function is unique to each joint. This is a GP prior for approximating general smooth functions. It has been used before, and has been found to work well for a number of problems; see Neal [1997], Williams and Rasmussen [1996] and Rasmussen [1997].[6]

The covariance function $k_i^{\backslash\mathrm{clmb}}$ consists of a constant offset term for modelling the mean of the posterior, a term for modelling any trends linear in the inputs, a squared-exponential term for modelling fluctuations around the linear trends, and a jitter term for better matrix conditioning and to account for model inadequacies. With the exception of the jitter term, $k_i^{\backslash\mathrm{clmb}}$ gives a GP that is infinitely mean square differentiable. Also, nonstationarity is present due to the linear term.

A covariance function parameterized in the manner of (4.5) implements automatic relevance determination [ARD, Neal, 1996], since the length scales l_{ij} and the weights u_{ij} determine the influence of the respective input dimensions in the covariance function $k_i^{\backslash\mathrm{clmb}}$ [Rasmussen and Williams, 2006, §5.1]. For example, $u_{i1} = 0$ means that x_1 will not have any contribution to the linear trend of the process for predicting the $i\mathrm{th}$ torque. Because the posterior process is highly dependent on the covariance function, it is usually desirable to have such a general parameterization to allow observed data to influence the choice of the hyperparameters (see section 4.3.3 later). Hence, we take the approach of using ARD to parameterize covariance functions wherever possible.

Alone, a smooth process based on $k_i^{\backslash\mathrm{clmb}}$ is unable to model the discontinuities due to the Coulomb friction that is present in the torque function τ_i. Since, under (4.2), the Coulomb friction is added to a smooth function to generate the torques, τ_i can be modelled by adding a relevant stochastic process prior τ_i^{clmb} for the Coulomb friction; see also the discussion in section 4.2.2 on Figure 4.6. The prior we use is $\tau_i^{\mathrm{clmb}} \sim GP(0, k_i^{\mathrm{clmb}})$, with the covariance between $\tau_i^{\mathrm{clmb}}(\boldsymbol{x}^{(\alpha)})$ and $\tau_i^{\mathrm{clmb}}(\boldsymbol{x}^{(\beta)})$ given by

$$k_i^{\mathrm{clmb}}(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)}) \stackrel{\mathrm{def}}{=} \sum_{j=1}^{\ell} w_{ij}^2 \operatorname{sgn}\left(\dot{q}_j^{(\alpha)}\right) \operatorname{sgn}\left(\dot{q}_j^{(\beta)}\right), \quad (4.6)$$

where the ARD parameterization is used, and the w_{ij}s are the unknown hyperparameters. This GP prior corresponds to modelling $\sum_j f_{ij}^{\mathrm{clmb}} \operatorname{sgn}(\dot{q}_j)$ using the prior $f_{ij}^{\mathrm{clmb}} \sim \mathcal{N}(0, w_{ij}^2)$.

[5] Any analytic function is infinitely differentiable.

[6] Neal [1997] allowed the exponent within the exponential to vary within $[0, 2]$; Williams and Rasmussen [1996] and Rasmussen [1997] constrained $u_{i1} = u_{i2} \ldots = u_{i\ell}$.

Hence, our prior for $\tau_i = \tau_i^{\setminus\mathrm{clmb}} + \tau_i^{\mathrm{clmb}}$ is $\tau_i \sim GP(0, k_i)$, with

$$k_i(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)}) \overset{\mathrm{def}}{=} k_i^{\setminus\mathrm{clmb}}(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)}) + k_i^{\mathrm{clmb}}(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)}). \tag{4.7}$$

4.3.3 Estimating Hyperparameters by Optimizing Marginal Likelihood

The covariance function k_i of the GP prior on τ_i belongs to the parametric class indexed by

$$\boldsymbol{\theta}_i^{\mathrm{x}} \overset{\mathrm{def}}{=} (b_i, v_i, \sigma_i, \boldsymbol{u}_i, \boldsymbol{l}_i, \boldsymbol{w}_i)^{\mathrm{T}} \in \mathbb{R}^{7\ell+3}, \tag{4.8}$$

where $\boldsymbol{u}_i$ (resp. $\boldsymbol{l}_i$, resp. $\boldsymbol{w}_i$) are the vectors of u_{ij}'s (resp. l_{ij}'s, resp. w_{ij}'s). These are called the hyperparameters of the model. Given observed data $\mathcal{D}_i \overset{\mathrm{def}}{=} \{(\boldsymbol{x}^{(j)}, t_i^{(j)})\}_{j=1}^n$ for the ith joint, the log marginal likelihood of $\boldsymbol{\theta}_i^{\mathrm{x}}$ is[7]

$$\log p(\boldsymbol{t}_i | X, \boldsymbol{\theta}_i^{\mathrm{x}}) = -\frac{1}{2}\boldsymbol{t}_i^{\mathrm{T}} K_i^{-1} \boldsymbol{t}_i - \frac{1}{2}\log|K_i| - \frac{n}{2}\log 2\pi, \tag{4.9}$$

where $X \overset{\mathrm{def}}{=} \{\boldsymbol{x}^{(j)}\}_{j=1}^n$, $\boldsymbol{t}_i \overset{\mathrm{def}}{=} (t_i^{(1)}, \ldots, t_i^{(n)})^{\mathrm{T}}$, and K_i is the matrix of covariances for all pairs in X given $\boldsymbol{\theta}_i^{\mathrm{x}}$. A fully Bayesian approach can be used in principle by introducing hyperpriors $p(\boldsymbol{\theta}_i^{\mathrm{x}})$ and inferring the posterior $p(\boldsymbol{\theta}_i^{\mathrm{x}}|\mathcal{D}_i)$. For the PUMA 560 robot manipulator with $\ell = 6$ degrees of freedom, this will involve a 45-dimensional numerical integral, and requires the use of Monte Carlo Markov Chain methods [Neal, 1997; Rasmussen, 1997], which are computationally intensive.

In this chapter, we shall take the more pragmatic and less computationally expensive approach, and instead treat the hyperparameters as fixed but unknown. Their values are estimated by maximizing the marginal likelihood of the model (see, for example, Mardia and Marshall [1984]). This is widely done in practice, for example Kennedy and O'Hagan [2001], and often suffices [Rasmussen and Williams, 2006, §5.2].

The marginal likelihood will usually have multiple local maxima, and a typical numerical local optimizer will only find the one that is in the basin of attraction of the initial guess of the hyperparameters. In our case where the ARD parameterization of the covariance function is used, we initialize by giving each covariate equal weights, and let the data break the symmetry to inform us which input dimensions are irrelevant. With this consideration, we optimize starting from values of the hyperparameters as calculated in appendix C.1. To prevent matrix conditioning problems, we furthermore constrain $\sigma_i^2 \geqslant 10^{-4}$, and $u_{ij} \in (0, 100]$. Optimization is done using the L-BFGS algorithm [Liu and Nocedal, 1989][8].

4.3.4 A Comparison of Four Models

We compare the GP prior given in the previous section with three other simpler models, to see if any simpler model is sufficient to provide an acceptable predictive performance. One example of a simpler model would be to ignore the presence of the discontinuities and omit the term k_i^{clmb}. Another example of a simpler model is linear regression (LR) on the same space of inputs

$$\tilde{\boldsymbol{x}} \overset{\mathrm{def}}{=} (\boldsymbol{x}^{\mathrm{T}}, \mathrm{sgn}(\dot{q}_1), \ldots, \mathrm{sgn}(\dot{q}_\ell), 1)^{\mathrm{T}},$$

[7] In the statistics community, this is more commonly called the likelihood instead. We follow the tradition in the machine learning community and call it the marginal likelihood, see [Rasmussen and Williams, 2006, §5.4.1].

[8] Software available from `http://www.ece.northwestern.edu/~nocedal/lbfgs.html`.

Table 4.5: The average nMSEs of the predictions by models LRc and GPc. Results for joints 1, 3 and 6, and for both kinds of test sets are shown. Training set sizes given in the second row. The nMSEs are averaged over loads c_1 to c_{15}. The mean (over 5 replications) of the average nMSEs are shown; the 90% confidence interval for the mean is suppressed to reduce clutter, but it is at least an order of magnitude smaller than the mean.

Setup		average nMSEs for *interp*$_m$			average nMSEs for *extrap*$_m$		
Joint	Method	$n=52$	546	1004	$n=52$	546	1004
1	LRc	1.5×10^{-3}	6.1×10^{-6}	5.9×10^{-6}	3.1×10^{-3}	8.8×10^{-4}	8.9×10^{-4}
	GPc	1.5×10^{-3}	1.4×10^{-7}	1.0×10^{-9}	2.6×10^{-3}	4.9×10^{-4}	3.2×10^{-4}
3	LRc	3.0×10^{-3}	6.0×10^{-4}	5.9×10^{-4}	2.7×10^{-1}	2.1×10^{-1}	2.1×10^{-1}
	GPc	2.8×10^{-5}	3.5×10^{-8}	1.7×10^{-8}	4.0×10^{-2}	1.5×10^{-2}	5.2×10^{-3}
6	LRc	9.2×10^{-4}	1.5×10^{-4}	1.5×10^{-4}	4.2×10^{-2}	3.0×10^{-2}	2.9×10^{-2}
	GPc	2.7×10^{-4}	4.4×10^{-6}	2.6×10^{-6}	1.3×10^{-2}	1.7×10^{-2}	1.5×10^{-2}

which can be motivated using a linearization of (4.2) under certain conditions [Schedlinski and Link, 2001, §2.2]. We use the Bayesian linear model (see, e.g., Rasmussen and Williams [2006, §2.1.1])

$$\tau_i(\boldsymbol{x}) = \tilde{\boldsymbol{x}}^{\mathrm{T}}\boldsymbol{\beta}_i \qquad t_i \sim \mathcal{N}(\tau_i(\boldsymbol{x}), 10^{-4}) \qquad \boldsymbol{\beta}_i \sim \mathcal{N}(\mathbf{0}, 10^4 I), \tag{4.10}$$

where $\boldsymbol{\beta}_i \in \mathbb{R}^{4\ell+1}$ is the vector of regression coefficients, and the variances are chosen to coincide with the bounds $\sigma_i^2 \geqslant 10^{-4}$, $u_{ij} \in (0, 100]$ placed on the hyperparameters of the GP prior. These settings correspond to a vague prior over the regression coefficients and a small noise variance over observations.

We refer to the models as follows: if a GP or LR model includes the Coulomb terms, we augment it with c, and with c̄ otherwise, e.g. GPc. The quality of the torques predicted by LRc̄, LRc, GPc̄ and GPc are compared in the following way. For each load c_m ($m = 1 \dots 15$), a model is learnt using the training data sampled from (T_{r}, T_m), and predictions are then made on the test sets *interp*$_m$ and *extrap*$_m$. Next, the normalized mean square errors (nMSEs) of the predictions are computed, by dividing the mean square errors by the variances of the test data. These nMSEs are then averaged over the 15 loads. The average nMSEs are computed for different sizes of the training sets, and for each of the six torques. The mean of the average nMSEs are then taken over 5 independent replications of the training sets. In general, we will use *average* to denote the average over the 15 loads, and *mean* to denote the mean over the 5 replications.

The average nMSEs take values within a wide range, from 5×10^{-10} to 2×10^4. For ease of comparison, we display the base 10 logarithm of the average nMSEs using the Hinton diagram in Figure 4.8. The Hinton diagram provides a qualitative display of the scale of the average nMSEs by representing each value with a square whose size is proportional to the value. To facilitate more quantitative comparisons, Table 4.5 gives the average nMSEs for LRc and GPc on joints 1, 3 and 6.

Comparing results for *interp*$_m$ and *extrap*$_m$ in Figure 4.8 and Table 4.5,we find that the latter is roughly at least an order of magnitude harder than the former. This is to be expected from section 4.2.2: different trajectories explore different parts of the $\boldsymbol{x}$-space, so that the training sets and the *extrap* test sets may

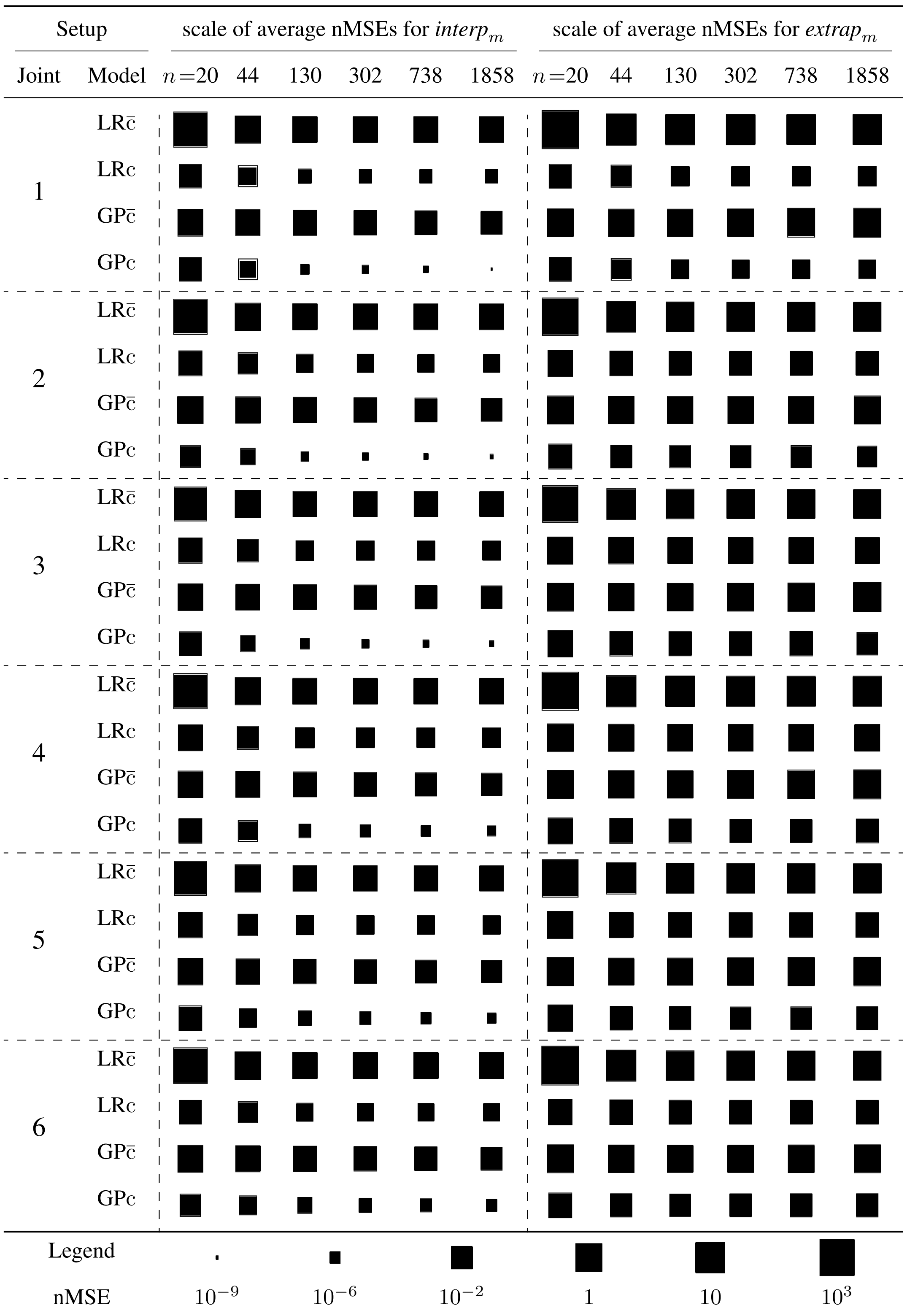


Figure 4.8: Comparing models LR$\bar{\mathrm{c}}$, LRc, GP$\bar{\mathrm{c}}$ and GPc for different joints and test scenarios using $\log_{10}$ of average nMSEs of the predictions. Training set sizes are given in the second row. The nMSEs are averaged over loads c_1 to c_{15}. The size of the filled squares represents the mean of the scale of the average nMSEs, taken over 5 replications, while the size of the enveloping hollow squares, if visible, represents the 95% confidence interval of the mean.

have little overlap in the $\boldsymbol{x}$-space. For theoretical perspectives of why extrapolation is less robust to model misspecifications than interpolation, see [Stein, 1999a, §3.6]. In section 4.4 we introduce a model that reduces the difficulty of extrapolation by sharing data and "borrowing statistical strength" across the different contexts.

From Figure 4.8 and Table 4.5, we have the following order of the models by decreasing predictive performance: GPc, LRc GP$\bar{\text{c}}$, LR$\bar{\text{c}}$. Even though full Bayesian inference is not used for the GP models in this case, we postulate that this conclusion is robust since the results are averaged over 15 contexts, and meaned over 5 replications. It is evident that modelling the discontinuities will produce much better predictions, and that GP will generally produce predictions that are better than LR.

4.3.5 Related Work

To get around the limitations of analytical models, learning the inverse dynamics of robot manipulators using non-parametric methods has previously been investigated. Some examples are locally weighted projection regression [LWPR, Vijayakumar and Schaal, 2000; Schaal et al., 2000], least squares support vector machines [de Kruif and de Vries, 2002], and Gaussian processes [Nguyen-Tuong et al., 2008; Rasmussen and Williams, 2006, §2.5]. The prior works on Gaussian processes are closely related to ours, since they are based on the same underlying statistical method. It is interesting to highlight that Nguyen-Tuong et al. [2008] have found linear regression using an analytical model to perform poorly; one possible reason is that their analytical model does not account for discontinuities due to friction.

4.4 A Gaussian Process Prior for Inverse Dynamics Modelling of Multiple Contexts

We have seen in section 4.2.2 how the torque functions will change depending on the load held at the end-effector of the manipulator. We have also seen in section 4.3.4 how an estimated torque function may produce relatively poor predictive results when used on trajectories previously unseen during training. In this section, we introduce a model that allows the sharing of information between torque functions trained on different contexts.

We first describe the relationship of inverse dynamics among contexts in section 4.4.1, and briefly review a multi-task GP regression model in section 4.4.2. In section 4.4.3 we describe how the multi-task GP (mGP) model can be used for learning across multiple contexts. Hyperparameter estimation and model selection are addressed in sections 4.4.4 and 4.4.5. In section 4.4.6 we show that sharing data across multiple contexts using the mGP model leads to more accurate torque predictions for the data set in section 4.2. Finally, related work is discussed in section 4.4.7.

4.4.1 Linear Relationship of Inverse Dynamics between Contexts

In this section, we introduce the linear relation between inverse dynamics and dynamic parameters. We will then exploit this relation to give a linear model for learning across multiple contexts.

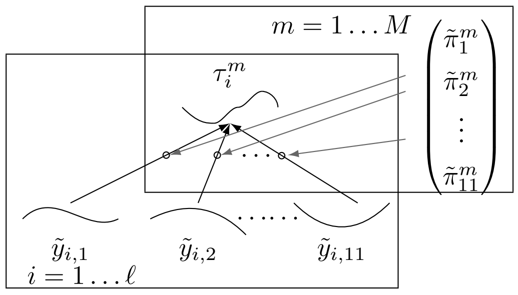


Figure 4.9: A schematic diagram on how the different functions are related. A rectangular frame, called a *plate*, repeats its contents over the specified range.

It is known, for example in An et al. [1988] and Sciavicco and Siciliano [2000], that the analytical model given by (4.2) may be re-written as

$$\boldsymbol{\tau}(\boldsymbol{x}) = Y(\boldsymbol{x})\boldsymbol{\pi}, \tag{4.11}$$

where $Y : \mathbb{R}^{3\ell} \mapsto \mathbb{R}^{\ell \times 12\ell}$ is a matrix of functions parameterized by the kinematic parameters only, and $\boldsymbol{\pi} \in \mathbb{R}^{12\ell}$ is the vector of dynamic parameters of all the links of the robot manipulator.[9] When, as in our case, the loads are rigidly attached to the end effector, each load may be considered as part of the last link, and thus modifies the inertial parameters for the last link only [Petkos and Vijayakumar, 2007]. The dynamic parameters for the other links remain unchanged since the parameters are local to the links and their frames. Let $\boldsymbol{\pi}_{\ell,\text{inertia}} \in \mathbb{R}^{10}$ be the vector of inertial parameters of the last link to which the end-effector is connected, and let $\tilde{\boldsymbol{\pi}} \overset{\text{def}}{=} (1, \boldsymbol{\pi}_{\ell,\text{inertia}}^{\mathrm{T}})^{\mathrm{T}} \in \mathbb{R}^{11}$. Then we may re-write (4.11) as

$$\boldsymbol{\tau}(\boldsymbol{x}) = \tilde{Y}(\boldsymbol{x})\tilde{\boldsymbol{\pi}} \tag{4.12}$$

where $\tilde{Y} : \mathbb{R}^{3\ell} \mapsto \mathbb{R}^{\ell \times 11}$ is now also parameterized by dynamic parameters $\boldsymbol{\pi} \setminus \boldsymbol{\pi}_{\ell,\text{inertia}}$. Introducing superscript m for the mth load or context, gives

$$\boldsymbol{\tau}^m(\boldsymbol{x}) = \tilde{Y}(\boldsymbol{x})\tilde{\boldsymbol{\pi}}^m. \tag{4.13}$$

Denote the ith row of $\tilde{Y}(\boldsymbol{x})$ by the vector function $\tilde{\boldsymbol{y}}_i(\boldsymbol{x})$. Then the ith torque is $\tau_i^m(\boldsymbol{x}) = \tilde{\boldsymbol{y}}_i^{\mathrm{T}}(\boldsymbol{x})\tilde{\boldsymbol{\pi}}^m$. Note that the $\tilde{\boldsymbol{y}}_i$s are shared among the contexts while the $\tilde{\boldsymbol{\pi}}^m$s are shared among the ℓ links, as illustrated in Figure 4.9. This decomposition is not unique, since given a non-singular square 11-by-11 matrix A_i, setting $\boldsymbol{z}_i(\boldsymbol{x}) \overset{\text{def}}{=} A_i^{-\mathrm{T}}\tilde{\boldsymbol{y}}_i(\boldsymbol{x})$ and $\boldsymbol{\rho}_i^m \overset{\text{def}}{=} A_i\tilde{\boldsymbol{\pi}}^m$, we also have

$$\tau_i^m(\boldsymbol{x}) = \tilde{\boldsymbol{y}}_i^{\mathrm{T}}(\boldsymbol{x})A_i^{-1}A_i\tilde{\boldsymbol{\pi}}^m = \boldsymbol{z}_i(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{\rho}_i^m. \tag{4.14}$$

Hence the vector of parameters $\tilde{\boldsymbol{\pi}}^m$ is identifiable only up to a linear combination. In general the matrix A_i may vary across the joints.

[9] We have used (4.2) as our basis, so that the dynamic parameters for each link consist of 12 parameters, 10 of which are the inertial parameters, and 2 of which are the friction coefficients. Others may have used different number of dynamic parameters, depending on whether they have included or excluded more factors, such as motor inertia or other models of friction. This, however, will not affect our subsequent analysis since only the inertial parameters vary here.

4.4.2 A Multi-task GP Regression Model

We briefly summarize the multi-task Gaussian process (mGP) regression model described by Bonilla et al. [2008] and discussed in section 2.5.1. This model learns M related functions $\{f^m\}_{m=1}^M$ by placing a zero mean GP prior that directly induces correlations between tasks. Let t^m be the observation of the $m\mathrm{th}$ function at $\boldsymbol{x}$. Then the model is given by[10]

$$\mathbb{C}\left(f^m(\boldsymbol{x}^{(\alpha)}), f^{m'}(\boldsymbol{x}^{(\beta)})\right) \stackrel{\text{def}}{=} K^{\mathrm{f}}_{mm'}\, k^{\mathrm{x}}(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)}) \qquad t^m \sim \mathcal{N}(f^m(\boldsymbol{x}), \sigma_m^2), \tag{4.15}$$

where $k^{\mathrm{x}}(\cdot,\cdot)$ is a covariance function over inputs, K^{f} is a positive semi-definite matrix of inter-task similarities, and σ_m^2 is the noise variance for the $m\mathrm{th}$ task. More detailed discussion of this and related models is given in section 4.4.7.

4.4.3 A Multi-task GP Model for Multiple Contexts

The multi-task GP model described above can be used for inferring inverse dynamics for multiple contexts. We begin from (4.14) and place independent zero mean GP priors on all the component functions of $\boldsymbol{z}_1(\cdot), \ldots, \boldsymbol{z}_\ell(\cdot)$. Let μ and ν index into the vector function $\boldsymbol{z}_i(\cdot)$. Our prior is

$$\mathbb{C}\left(z_{i\mu}(\boldsymbol{x}^{(\alpha)}), z_{j\nu}(\boldsymbol{x}^{(\beta)})\right) = \delta_{ij}\delta_{\mu\nu}\, k_i^{\mathrm{x}}(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)}), \tag{4.16}$$

where k_i^{x} is the same as the k_i defined in section 4.3.2.1 except for the jitter term. In addition to the independencies specified by the Kronecker delta function δ, this model also imposes the constraint that all component functions for a given joint i share the same covariance function $k_i^{\mathrm{x}}(\cdot,\cdot)$. With this prior over the $\boldsymbol{z}_i$s, the Gaussian process prior for $\tau_i^m(\cdot)$ has covariance

$$\mathbb{C}\left(\tau_i^m(\boldsymbol{x}^{(\alpha)}), \tau_j^{m'}(\boldsymbol{x}^{(\beta)})\right) = \delta_{ij}\,(K_i^{\mathrm{f}})_{mm'}\, k_i^{\mathrm{x}}(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)}), \tag{4.17}$$

where we have set $\mathcal{P}_i \stackrel{\text{def}}{=} (\boldsymbol{\rho}_i^1|\cdots|\boldsymbol{\rho}_i^M)$ and $K_i^{\mathrm{f}} \stackrel{\text{def}}{=} \mathcal{P}_i{}^{\mathrm{T}}\mathcal{P}_i$, so that $(\boldsymbol{\rho}_i^m)^{\mathrm{T}}\boldsymbol{\rho}_i^m = (K_i^{\mathrm{f}})_{mm'}$, which is the the $(m,m')\mathrm{th}$ entry of the positive semi-definite matrix K_i^{f}. The similarity between different contexts is defined by K_i^{f}. The rank of K_i^{f} is the rank of $\mathcal{P}_i$, and is upper bounded by $\min(M, 11)$, reflecting the fact that there are at most 11 underlying latent functions (see Figure 4.9 and section 4.4.1).

Due to the presence of noise and inadequacies in the model, the observations $t_i^m(\boldsymbol{x})$ will deviate from $\tau_i^m(\boldsymbol{x})$. This may be modelled with $t_i^m(\boldsymbol{x}) \sim \mathcal{N}(\tau_i^m(\boldsymbol{x}), (\sigma_i^m)^2)$, though in practice we share the variance parameters among the contexts, setting $\sigma_i \stackrel{\text{def}}{=} \sigma_i^1 \equiv \sigma_i^2 \ldots \equiv \sigma_i^M$. This completes the correspondence with the multi-task GP model in (4.15). However, in this case there are ℓ multi-task GP models, one for each joint.

This model is a simple and convenient one where the prior, likelihood and posterior factorize over joints. Hence inference and hyperparameter estimation can be done separately for each joint.

[10] Here and thereafter, for a GP prior on a function f, we shall give the covariance function directly in terms of the covariance $\mathbb{C}\left(f(\boldsymbol{x}^{(\alpha)}), f(\boldsymbol{x}^{(\beta)})\right)$ rather than in terms of the covariance function $k(\boldsymbol{x}^{(\alpha)}, \boldsymbol{x}^{(\beta)})$. This makes clear the covariance between the different torque functions for joints and contexts, and allows us to express independence between torque functions via Kronecker delta functions δ_{ij} concisely.

4.4.3.1 Predictions

Inference in the multiple context model can be done by using the standard GP formulae for the mean and variance of the predictive distribution with the covariance function given in equation 4.17 together with the normal noise model. For the mth context and the ith joint, denote the set of n^m observed data by $\mathcal{D}_i^m \overset{\text{def}}{=} \{(\boldsymbol{x}^{m\,(j)}, t_i^{m\,(j)})\}_{j=1}^{n^m}$. We lay out all the data for the ith joint in the order of the context using $\mathcal{D}_i \overset{\text{def}}{=} \big((\boldsymbol{x}^{1\,(j)}, t_i^{1\,(j)})_{j=1}^{n^1}, \ldots, (\boldsymbol{x}^{M\,(j)}, t_i^{M\,(j)})_{j=1}^{n^M}\big)$. Suppose we wish to predict the torque for the mth context and the ith joint given a novel input $\boldsymbol{x}^{(*)}$. Let K_i be the matrix of covariances for all pairs of inputs in $\mathcal{D}_i$ due to equation 4.17; let $\boldsymbol{k}_i^m$ be the vector of covariances between $\boldsymbol{x}^{(*)}$ under the mth context and inputs in $\mathcal{D}_i$; let D_i be a diagonal matrix with diagonal entries $\big((\sigma_i^1)^2$ (repeats n^1 times), $\ldots, (\sigma_i^M)^2$ (repeats n^M times)$\big)$; and let

$$\boldsymbol{t}_i \overset{\text{def}}{=} \big(t_i^{1\,(1)}, \ldots, t_i^{1\,(n^1)}, \ldots, t_i^{M\,(1)}, \ldots, t_i^{M\,(n^M)}\big).$$

That is, if the pth and p'th element in $\mathcal{D}_i$ are $(\boldsymbol{x}^{q\,(j)}, t_i^{q\,(j)})$ and $(\boldsymbol{x}^{q'\,(j')}, t_i^{q'\,(j')})$, then the (p,p')th entry of K_i is $\mathbb{C}\left(\tau_i^q(\boldsymbol{x}^{q\,(j)}), \tau_i^{q'}(\boldsymbol{x}^{q'\,(j')})\right)$, and the pth entry in $\boldsymbol{k}_i^m$ is $\mathbb{C}\left(\tau_i^p(\boldsymbol{x}^{p\,(j)}), \tau_i^m(\boldsymbol{x}^{(*)})\right)$. With these defined, the posterior mean of the required torque is given by

$$\bar{\tau}_i^m(\boldsymbol{x}^{(*)}) = (\boldsymbol{k}_i^m)^{\mathrm{T}}(K_i + D_i)^{-1}\boldsymbol{t}_i. \tag{4.18}$$

A clearer notation using the Kronecker products $\otimes$ of matrices can be used in the case of a *complete block design* or *isotopic data*, where the n^ms are the same, say $n^\bullet$, and the same inputs are used in all contexts, i.e. $\boldsymbol{x}^{\bullet(j)} \overset{\text{def}}{=} \boldsymbol{x}^{1\,(j)} \equiv \boldsymbol{x}^{2\,(j)} \equiv \ldots \equiv \boldsymbol{x}^{M\,(j)}$, for all $j = 1 \ldots n^\bullet$. In this case, let K_i^{x} be the matrix of covariances between the $\boldsymbol{x}^{\bullet(j)}$s due to k_i^{x}, and similarly for $\boldsymbol{k}_i^{\mathrm{x}}$, the vector of covariances between $\boldsymbol{x}^{(*)}$ and the $\boldsymbol{x}^{\bullet(j)}$s. Also let D_i^{f} be a diagonal matrix with diagonal entries $((\sigma_i^1)^2, \ldots, (\sigma_i^M)^2)$. Then

$$\bar{\tau}_i^m(\boldsymbol{x}^{(*)}) = \left((K_i^{\mathrm{f}})_m \otimes \boldsymbol{k}_i^{\mathrm{x}}\right)^{\mathrm{T}} \left(K_i^{\mathrm{f}} \otimes K_i^{\mathrm{x}} + D_i^{\mathrm{f}} \otimes I_{n^\bullet \times n^\bullet}\right)^{-1} \boldsymbol{t}_i, \tag{4.19}$$

where $(K_i^{\mathrm{f}})_m$ is the mth column of K_i^{f}.

Prediction $\bar{\tau}_i^m(\boldsymbol{x}^{(*)})$ for context m makes use of data $\boldsymbol{t}_i$ from all the contexts, as evident from (4.18) and (4.19). Thus "sharing of statistical strength" is achieved in general. It is of interest to understand the particular cases when such sharing does not occur, and this is discussed next.

4.4.3.2 Noiseless observations and cancellation of transfer during prediction

Consider the case of noise-free observations for a complete block design. Then maximizing the marginal likelihood $p(\boldsymbol{t}_i \mid \{\boldsymbol{x}^{\bullet(j)}\}_{j=1}^{n^\bullet}, \boldsymbol{\theta}_i^{\mathrm{x}}, K_i^{\mathrm{f}}, \sigma_i)$ with respect to the parameters $\boldsymbol{\theta}_i^{\mathrm{x}}$ of k_i^{x} reduces to maximizing $-M \log |K_i^{\mathrm{x}}| - n^\bullet \log |T_i^{\mathrm{T}}(K_i^{\mathrm{x}})^{-1}T_i|$, where T_i is an $n^\bullet$-by-M matrix such that $\operatorname{vec} T_i \overset{\text{def}}{=} \boldsymbol{t}_i$ (see appendix C.2 for details). This objective function is convenient in that it does not depend on K_i^{f}. After convergence we can obtain K_i^{f} as $T_i^{\mathrm{T}}(K_i^{\mathrm{x}})^{-1}T_i/n^\bullet$. The intuition behind is this: The responses T_i are correlated via K_i^{f} and K_i^{x}. We can learn K_i^{f} by de-correlating T_i with $(K_i^{\mathrm{x}})^{-1}$ first so that only the correlations due to K_i^{f} are left. Then K_i^{f} is simply the sample covariance of the de-correlated T_i.

Unfortunately, in this case there is effectively no transfer of information across the contexts (given the covariance function k_i^{x}). To see this, we continue from (4.19) for torques of all M contexts and setting

D_i^{f} to be the matrix of zeros:

$$\begin{aligned}\left(\bar{\tau}_i^1(\boldsymbol{x}^{(*)}) \cdots \bar{\tau}_i^M(\boldsymbol{x}^{(*)})\right)^{\mathrm{T}} &= \left(K_i^{\mathrm{f}} \otimes \boldsymbol{k}_i^{\mathrm{x}}\right)^{\mathrm{T}} \left(K_i^{\mathrm{f}} \otimes K_i^{\mathrm{x}}\right)^{-1} \boldsymbol{t}_i \\ &= \left\{(K_i^{\mathrm{f}})^{\mathrm{T}} \otimes (\boldsymbol{k}_i^{\mathrm{x}})^{\mathrm{T}}\right\} \left\{(K_i^{\mathrm{f}})^{-1} \otimes (K_i^{\mathrm{x}})^{-1}\right\} \boldsymbol{t}_i \\ &= \left[\left\{(K_i^{\mathrm{f}})^{\mathrm{T}}(K_i^{\mathrm{f}})^{-1}\right\} \otimes \left\{(\boldsymbol{k}_i^{\mathrm{x}})^{\mathrm{T}}(K_i^{\mathrm{x}})^{-1}\right\}\right] \boldsymbol{t}_i \\ &= \left[I_{M\times M} \otimes \left\{(\boldsymbol{k}_i^{\mathrm{x}})^{\mathrm{T}}(K_i^{\mathrm{x}})^{-1}\right\}\right] \boldsymbol{t}_i \\ &= \left((\boldsymbol{k}_i^{\mathrm{x}})^{\mathrm{T}}(K_i^{\mathrm{x}})^{-1}\boldsymbol{t}_i^1 \quad \cdots \quad (\boldsymbol{k}_i^{\mathrm{x}})^{\mathrm{T}}(K_i^{\mathrm{x}})^{-1}\boldsymbol{t}_i^M\right)^{\mathrm{T}}, \end{aligned} \tag{4.20}$$

where $\boldsymbol{t}_i^m \overset{\text{def}}{=} (t_i^{m\,(1)}, \ldots, t_i^{m\,(n)})$, $m = 1 \ldots M$, which is the sub-vector in $\boldsymbol{t}_i$ corresponding to the $m\mathrm{th}$ context. A similar result holds for the covariances. Thus, in the noiseless case with a complete block design, the predictions for the $m\mathrm{th}$ context depend only on the targets $\boldsymbol{t}_i^m$. In other words, there is no transfer of information among the contexts. One can in fact generalize this result to show that the cancellation of transfer for the $m\mathrm{th}$ context does still hold even if the observations are only sparsely observed at locations $\{\boldsymbol{x}^{\bullet(1)}, \ldots, \boldsymbol{x}^{\bullet(n)}\}$ on the other contexts. This is known as *auto-krigeability* in the geostatistics literature [Wackernagel, 1998], and is also related to the *symmetric Markov property* of covariance functions formulated by O'Hagan [1998]. This result can also be generalized to multidimensional tensor product covariance functions and grids [Williams et al., 2007]. More details can be found in section 3.9.1 on page 92.

4.4.3.3 The relationship among task similarity matrices

Let $\tilde{\Pi} \overset{\text{def}}{=} (\tilde{\boldsymbol{\pi}}^1|\cdots|\tilde{\boldsymbol{\pi}}^M)$. Recall that $\tilde{\boldsymbol{\pi}}^m$ is an 11 dimensional vector, so $\tilde{\Pi}$ is an 11-by-M matrix. However, if the different loads in the end effector do not explore the full $\mathbb{R}^{11}$ space (e.g. if some of the inertial parameters are constant over all loads), then it can happen that $s \overset{\text{def}}{=} \operatorname{rank}(\tilde{\Pi}) \leqslant \min(M, 11)$. Indeed, this is what we will find in our experiments in section 4.4.6.

It is worthwhile to investigate the relationship between K_i^{f} and K_j^{f}, $i \neq j$. Recall from (4.14) that $\boldsymbol{\rho}_i^m \overset{\text{def}}{=} A_i\tilde{\boldsymbol{\pi}}^m$, where A_i is a full-rank square matrix. This gives $\mathcal{P}_i = A_i\tilde{\Pi}$ and $K_i^{\mathrm{f}} = \tilde{\Pi}^{\mathrm{T}}{A_i}^{\mathrm{T}}A_i\tilde{\Pi}$, so that $\operatorname{rank}(K_i^{\mathrm{f}}) = \operatorname{rank}(\tilde{\Pi})$. Therefore the K_i^{f}s have the same rank for all joints, although their exact values may differ. This observation will be useful for model selection in section 4.4.5.

4.4.4 Estimating Hyperparameters by Optimizing Marginal Likelihood

Similar to the single context, we estimate the hyperparameters $\boldsymbol{\theta}_i^{\mathrm{x}}$ (of k_i^{x}), K_i^{f} and σ_i of the multi-task GP model for each joint i by maximizing the marginal likelihood $p(\boldsymbol{t}_i \mid X, \boldsymbol{\theta}_i^{\mathrm{x}}, K_i^{\mathrm{f}}, \sigma_i)$, where X is the set of inputs in $\mathcal{D}_i$. As pointed out us in [Bonilla et al., 2008], one may approach this using either general gradient-based optimization, or expectation-maximization [Zhang, 2007]. In this chapter we use the L-BFGS algorithm for gradient-based optimization, as has been done for the single context model. The positive semi-definiteness of K_i^{f} is ensured during optimization by using the incomplete Cholesky decomposition parameterization $K_i^{\mathrm{f}} \overset{\text{def}}{=} \Lambda_i\Lambda_i^{\mathrm{T}}$, where Λ_i is an $M \times r$ truncated lower triangular matrix and $r \leqslant M$. The issue of multiple local maxima in the objective function is dealt with by localizing the search to regions of preferred interpretations. This is detailed in appendix C.3.

Different values of r for rank-r constrained K_i^{f} are tried, each giving a different model. The next section gives the model selection procedure we use.

4.4.5 Model Selection

The choice of the rank r of K_i^{f} in the model is important, since it reflects on the underlying dimensionality s of $\tilde{\boldsymbol{\pi}}^m$. It is not a fair comparison to select r by comparing the optimized marginal likelihoods for different values of r, since a larger r has more free parameters (see equation 4.21 below) and leads to a better optimized marginal likelihood.[11] Thus to infer its value we rely on an information criterion, which penalizes free parameters, to select the most parsimonious correct model. Here we use the Akaike information criterion (AIC) with a second order correction for small sample sizes (AICc); see Hurvich and Tsai [1989] and Burnham and Anderson [2002, §2.4].

As discussed in section 4.4.3.3, the rank r is the same for all joints by the construction of the matrices K_i^{f}s. We would like the model selection to reflect this. Let L_{ir} be the likelihood for each joint at optimized hyperparameters $\boldsymbol{\theta}_i^{\mathrm{x}}$, K_i^{f}, and σ_i^2, when K_i^{f} is constrained to have rank r; let n_i^m be the number of observations for ith joint and the mth context, and $n_{\mathrm{grand}} \overset{\mathrm{def}}{\equiv} \sum_{i,m} n_i^m$ be the total number of observations; and let d_i be the dimensionality of $\boldsymbol{\theta}_i^{\mathrm{x}}$.[12] Since the number of parameters needed to define an incomplete Cholesky decomposition of rank r for an M-by-M matrix is $r(2M+1-r)/2$, the number of parameters in a rank-r mGP model for each joint is

$$\mathrm{nparam}_i(r) = d_i + r(2M+1-r)/2 + 1, \tag{4.21}$$

where the noise variance parameter is also included. Thus, there are $\mathrm{nparam}(r) \overset{\mathrm{def}}{\equiv} \sum_{i=1}^{\ell} \mathrm{nparam}_i(r)$ parameters altogether for a rank-r mGP model for all joints. Since the likelihood of the model factorizes over joints, the criterion is

$$\mathrm{AICc}(r) = -2\sum_{i=1}^{\ell} \log L_{ir} + 2\,\mathrm{nparam}(r) + \frac{2\,\mathrm{nparam}(r)\,[\,\mathrm{nparam}(r)+1\,]}{n_{\mathrm{grand}} - \mathrm{nparam}(r) - 1} \tag{4.22}$$

To select the appropriate rank of the K_i^{f}s, we compute and compare AICc(r) for different values of r, and choose the r that gives the lowest AICc(r).

Remark The use of the Bayesian information criterion (BIC) or the Hannan-Quinn criterion is similar to the use of AICc. Burnham and Anderson [2002, §2.4] advocate the use of AICc over AIC when the number of samples is small compared with number of parameters, since AICc gives better results with little extra effort. Comparing AIC and BIC, the predictive distribution usually converges to the true distribution at the optimal rate for AIC [Shitbata, 1983], while the BIC only achieves the optimal rate within a logarithm factor [Rissanen et al., 1992]. An explanation on why BIC sometimes converges slowly is given by van Erven et al. [2008]; see also Yang [2005]. Across the replications of our experiments, AICc gives more consistent predictive results while BIC selects more consistent ranks.

[11] This is not always the case due to our optimization procedure.

[12] In fact, for our data and model, we have $n_1^m \equiv \ldots \equiv n_\ell^m$ and $d_1 \equiv \ldots \equiv d_\ell$.

4.4.6 Experiments and Results for Torque Prediction

We compare the predictive accuracy of the multi-task GP (mGP) model against three simpler alternatives. Using the AICc, a mGP model is selected from 14 mGP models with ranks r of the K_i^{f}s taking values 1 to 14, and we denote the selected model by mGP-AICc. As discussed in section 4.4.5, the same rank is chosen for all the joints. The alternatives are described below, and are applied separately for each joint. The first alternative, denoted by sGP, is a single GP for each context, as per the GPc of section 4.3.4. The second alternative, iGP, is a collection of independent GPs for the contexts, but sharing hyperparameters $(\boldsymbol{\theta}_i^{\mathrm{x}}, \sigma_i)$ among the contexts. The third alternative, pGP, pools together data from all the contexts, and models the torques from all the contexts with a single GP by assuming the variations among torques from different contexts are due to noise. The iGP and pGP models can be seen as restrictions of the multi-task GP model, restricting K_i^{f} to the identity matrix $I_{M\times M}$ and the matrix of ones $1_{M\times M}$ respectively.[13]

The models are learnt and their hyperparameters optimized for loads c_1 to c_{14} using the training data given in Table 4.4. For subsequent reference, we use n to denote the number of observed torques for each joint totalled across all $M = 14$ contexts, i.e., $n \stackrel{\mathrm{def}}{=} \sum_{m=1}^{M} n_i^m$, which is the same for all joints. Predictions are then made on the test sets *interp*$_m$ and *extrap*$_m$, $m = 1 \ldots 14$. Similar to section 4.3.4, the normalized mean square errors (nMSEs) of the predictions are computed separately for each context and each joint. Data from c_{15} is not part of the training data, and hence trajectory T_{15} is entirely unobserved during learning. Nonetheless, this trajectory is present in the *extrap*$_m$ test sets, so that predictive performance on a previously unobserved trajectory is also measured. The experiment is replicated five times.

As discussed in appendix C.3, the hyperparameters for the mGPs are initialized to either those of pGP or those of iGP during optimization, choosing the one with the higher marginal likelihood. For our data, we find that the choice is mostly iGP; pGP is only chosen for the case of joint 1 and $n < 30$, and, even so, only 28% of the time. In addition, the chosen ranks of the K_i^{f}s based on the AICc are $r = 4$ for 52.5% of the cases, $r = 5$ for 38.8% of the cases, $r = 6$ for 7.5% of the cases, and $r = 7$ for 1.2% of the cases, aggregated across sixteen different training set sizes and the five replications. Therefore the inverse dynamics in our setup lie on a low dimensional manifold in a 14 dimensional space.

Figures 4.10 and 4.11 give the nMSEs averaged over the 14 contexts for sGP, iGP, pGP and mGP-AICc. In Figure 4.10, the means (over the five replications) of the average nMSEs are plotted, together with an error bar representing the 90% confidence interval for each mean. Figure 4.10 plots the medians of the average nMSEs. The means and medians are plotted against the number of observations on a logarithmic scale. For Figure 4.12, we calculate the difference in the average nMSEs between mGP-AICc and the best alternative for each of the 5 replications, and plot the differences against the number of observations using a scatter plot. The mean of the differences are also plotted in Figure 4.12.

In each figure, the top plots give the results for the interpolation task, while the bottom plots give the results for the extrapolation task, and leftmost plots are for joint 1, followed by plots for joint 2, etc. Specified values taken by n are represented by the ticks on the x-axis. Note that the vertical scales of the plots vary across the joints, and between the interpolation and extrapolation tasks. If a value lies

[13] Some of these experiments are done on the Edinburgh Compute and Data Facility [Richards and Baker, 2008].

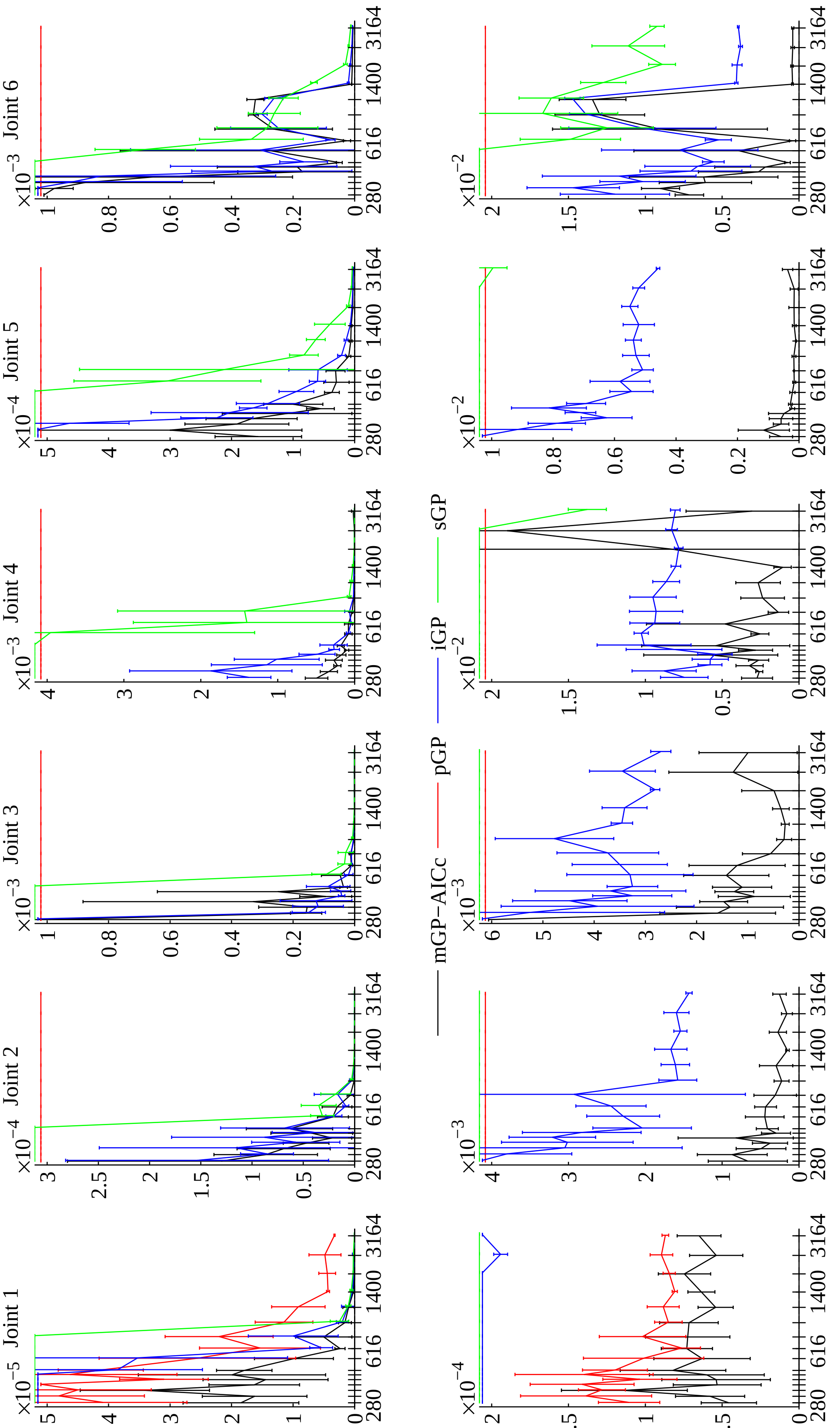


Figure 4.10: Plots of mean of average nMSEs (with 90% confidence interval) against total number of observations (on a logarithmic scale). If a value lies above the upper limit of the vertical range, it is given a nominal value near the upper limit.

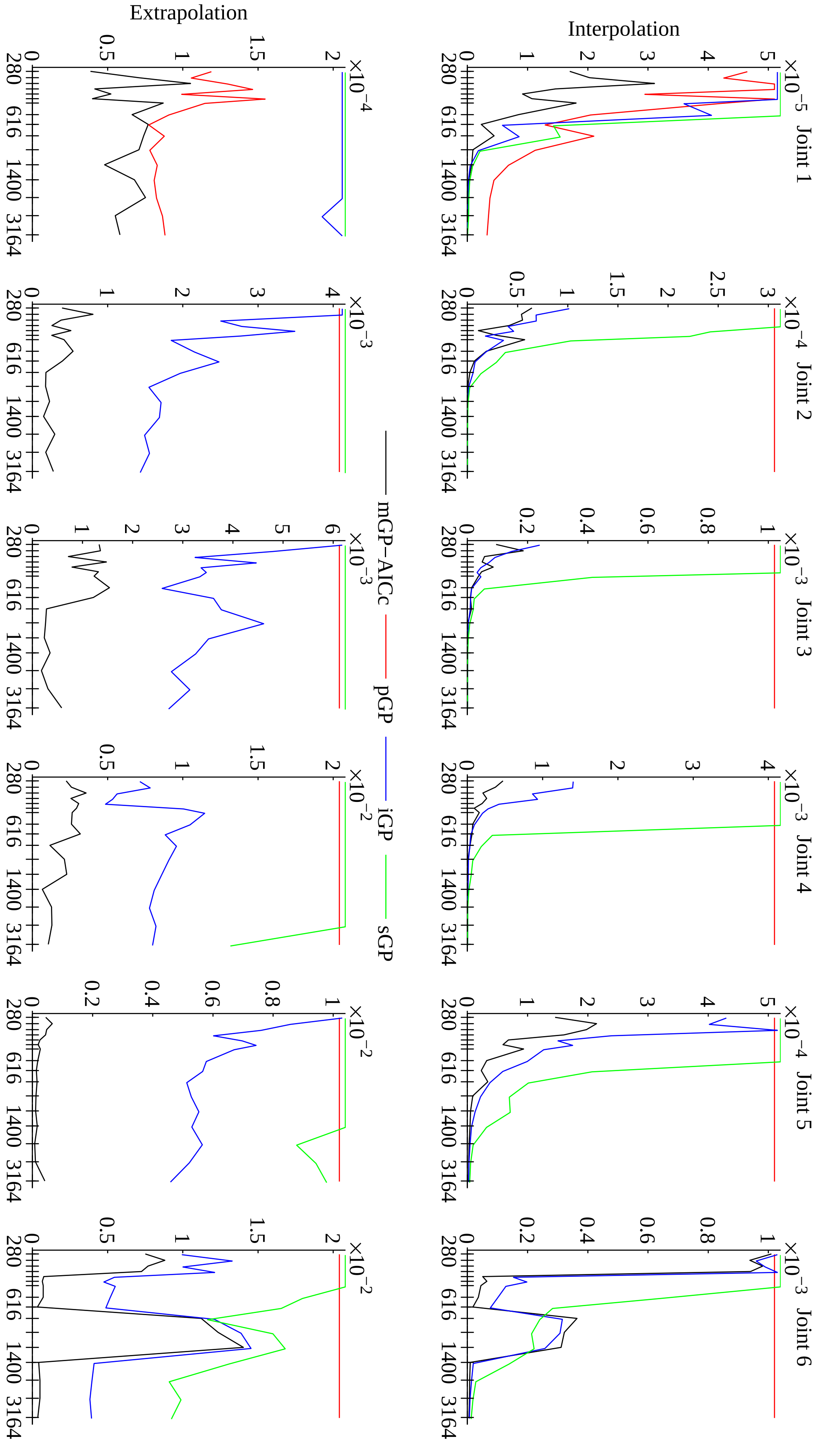


Figure 4.11: Plots of median of average nMSEs against total number of observations (on a logarithmic scale). If a value lies above the upper limit of the vertical range, it is given a nominal value near the upper limit.

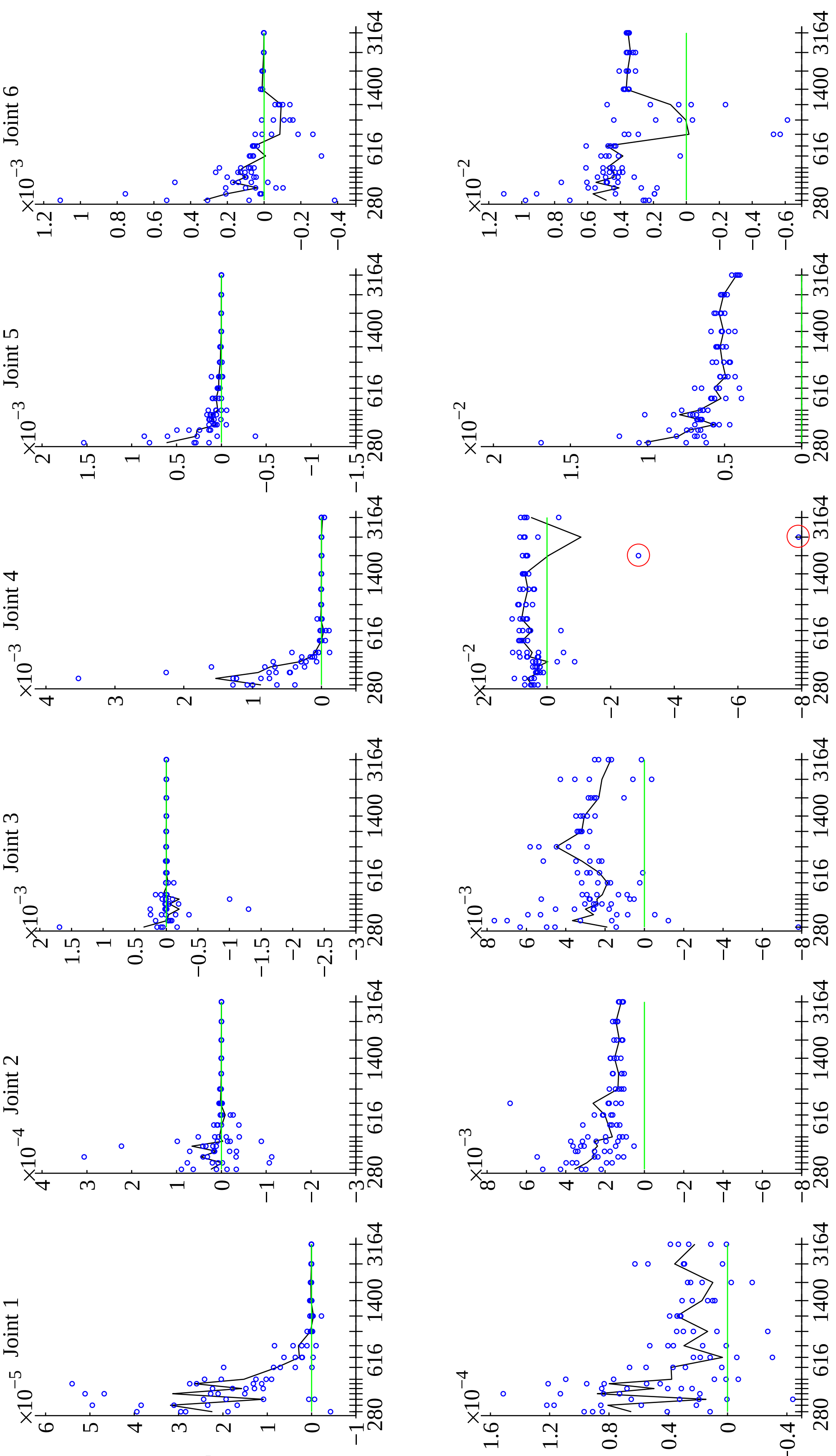


Figure 4.12: Scatter plots of paired differences of average nMSEs between mGP-AICc and the best alternative against total number of observations (on a logarithmic scale). The mean curve of the differences is also plotted.

above the upper limit of the vertical range, it is given a nominal value near the upper limit.

Since the training data are subsets selected independently for the different values of n, the plots reflect the underlying variability in sampling. Nevertheless, we can see that mGP-AICc performs favorably in almost all the cases, and especially so for the extrapolation task. In the extrapolation task for joint 4, we find a decrease in the mean predictive performance of mGP-AICc at $n = 1820$ and $n = 2380$; see Figure 4.10. However, if we look at the corresponding median plot in Figure 4.11, we find that mGP-AICc still does better in general. The discrepancy between the mean and median plots can be explained by examining the paired difference plot in Figure 4.12, where we see that the decrease in mean performance of the mGP-AICc is due to one particular replication each for $n = 1820$ and $n = 2380$, circled red in the figure. For joint 1, we see a close match between the predictive performances of mGP-AICc and pGP. This is due to the limited variation among observed torques for this joint across the different contexts for the range of end-effector movements investigated here, as can be seen intuitively from Figures 4.2 and 4.3. Therefore it is not surprising that pGP produces good predictions for joint 1. For the other joints, iGP is usually the next best after mGP-AICc. In particular, iGP is better than sGP, showing that (in this case) it is better to use all the data to estimate the parameters of a single common covariance function rather than splitting the data to estimate the parameters of 14 covariance functions.

4.4.7 Related Work

We consider related work first with regard to the inverse dynamics problem, then to multi-task learning with Gaussian processes, and finally to tensor product covariance functions in other problems.

Inverse dynamics for multiple contexts The linearity of torque functions in the dynamic parameters as expressed by (4.11) has been previously exploited for learning the inverse dynamics of multiple contexts, in the *multiple model switching and tuning* (MMST) model of Cılız and Narendra [1996] and the model of Petkos and Vijayakumar [2007] based on *locally weighted projection regression* (LWPR).

MMST uses an inverse dynamics model and a controller per context, switching among the models by selecting the one that predicts the torque most accurately. Here, the structure of the models and the number of discrete models are assumed to be known. Referring to (4.13), MMST assumes the non-linear regressor matrix $\tilde{Y}(\boldsymbol{x})$ is known, and estimates only the inertial parameters $\tilde{\boldsymbol{\pi}}^m$ for each context. Hence MMST is a linear-in-the-parameters model, and involves very little dynamics learning. In contrast, the multi-task GP model presented in this chapter does not assume a known $\tilde{Y}(\boldsymbol{x})$.

The LWPR-based method of Petkos and Vijayakumar estimates the torque function $\boldsymbol{\tau}^m(\cdot)$ of each context individually using LWPR. If the parameters $\tilde{\boldsymbol{\pi}}^m$s at the last links are known for at least 11 contexts, then the estimated torque functions can be used to estimate the underlying $\tilde{Y}(\boldsymbol{x})$ using linear regression

$$\tilde{Y}(\boldsymbol{x}) = \left(\begin{array}{c|c|c} \boldsymbol{\tau}^1 & \cdots & \boldsymbol{\tau}^M \end{array} \right) \tilde{\Pi}^{\mathrm{T}} \left(\tilde{\Pi}\tilde{\Pi}^{\mathrm{T}}\right)^{-1} \qquad \tilde{\Pi} \overset{\text{def}}{=} \left(\begin{array}{c|c|c} \tilde{\pi}^1 & \cdots & \tilde{\pi}^M \end{array} \right) \qquad \text{for } M \geqslant 11, \tag{4.23}$$

and prediction in a novel context (with limited training data) will depend on estimating the inertial parameters for that context. Assuming the original estimated torque functions are imperfect, having more than 11 models for distinct known inertial parameters will improve load estimation. If the parameters are unknown, the novel torque function can still be represented as a linear combination of a set of 11 linearly independent torque functions, and so one can estimate the inverse dynamics in a novel context by linear regression on those estimated functions. In contrast to the known case, however, no more than 11 models can be used [Petkos and Vijayakumar, 2007, §V]. Another difference between known and unknown inertial parameters is that in the former case the resulting $\tilde{\boldsymbol{\pi}}^m$s are interpretable, while in the latter case there is ambiguity due to the matrix A_i in (4.14).

Comparing our approach with that of Petkos and Vijayakumar, we note that: (a) their approach does not exploit the knowledge that the torque functions for the different contexts are known to share latent functions as in equation 4.13, and thus it may be useful to learn the M inverse dynamics models *jointly*. This is expected to be particularly advantageous when the data for each task explores rather different portions of $\boldsymbol{x}$-space; (b) rather than relying on least-squares methods (which assume equal error variances everywhere), our fully probabilistic model will propagate uncertainties (co-variances for jointly Gaussian models) automatically; and (c) equation 4.17 shows that we do not need to be limited to exactly 11 reference contexts, either fewer or more than 11 can be used — and indeed the experiment in section 4.4.7 uses 14 contexts. On the other hand, using the LWPR methods will generally give rise to better computational scaling for large data sets (although see approximate GP methods in Quiñonero-Candela et al. [2007]), and are perhaps less complex than the method in this chapter.

A model that uses the linear-in-dynamic-parameters property of (4.13) only indirectly is the *modular neural network architecture* (MNN) of Jacobs and Jordan [1993]. Similar to MMST, they have assumed a known $\tilde{Y}(\boldsymbol{x})$. For M loads, each of the M "expert networks" is parameterized by $\tilde{\boldsymbol{\pi}}^m$ and computes a torque using the linear model. The resultant torque used for control is a convex combination of these M torques, using mixing proportions produced by a "gating network" that is a function of load/context identities.[14] In contrast to multiple context models motivated by the linearity in dynamic parameters, their model is set up to discover decompositions of inverse dynamics models for the different contexts using the competitive learning nature of mixture models. Nevertheless, they have noted that the expert networks can be seen as "basis functions", and a weighted combination of these can predict torques for novel contexts.

A model that does not make use of the linearity in dynamic parameters is the biologically plausible *modular selection and identification for control* (MOSAIC) model of Haruno et al. [2001] [see also Wolpert and Kawato, 1998], which uses an inverse dynamics model and a forward dynamics model[15] for each context. MOSAIC uses the gating network idea of MNN, but now the vector of mixing proportions is a function of prediction errors of the forward dynamics models. The inverse dynamics models of MOSAIC neither assume a known $\tilde{Y}(\boldsymbol{x})$ nor make use of the known factorization of (4.13), but instead are learnt *de novo*.

Multi-task learning If we view the inverse dynamics for each context as a task to be learnt, then learning inverse dynamics for multiple context can be naturally viewed as an application of multi-task learning (MTL). Early references to general multi-task learning are Caruana [1997] and Thrun and Pratt

[14] Jacobs and Jordan [1993] have also proposed two variants that include a shared network which is not gated.

[15] Forward dynamics models are used for computing joint accelerations $\ddot{\boldsymbol{q}}$ as a function of joint angles $\boldsymbol{q}$, joint velocities $\dot{\boldsymbol{q}}$ and applied torques $\boldsymbol{\tau}$.

[1998]. There has been a lot of work in recent years on MTL with e.g. neural networks, Dirichlet processes, Gaussian processes and support vector machines. A more thorough review of MTL models is given in chapter 2.

For Gaussian processes, one important related work is the semiparametric latent factor model of Teh et al. [2005]. This model has a number of latent processes that are linearly combined to produce observable functions as in (4.14), and each latent process may have its own covariance function. However, all the latent functions in our model share a common covariance function, which reduces the number of free parameters and should thus help to reduce over-fitting. Also we note that the regression experiments by Teh et al. [2005, §4] used a forward dynamics problem on a four-jointed robot arm for a single context, with an artificial linear mixing of the four target joint accelerations to produce six response variables. In contrast, the model in this chapter exploits the linear mixing that arises naturally in a multi-context inverse dynamics situation. In relation to work by Bonilla et al. [2008] described in section 4.4.2, we note that the factorization between inter-task similarity K^{f} and a common covariance function c^{x} is an *assumption* there, while we have shown that such decomposition is *inherent* in our application.

Tensor product covariance functions in other problems In the geostatistics literature, the prior model for the f^ms given in (4.15) is known as the *intrinsic correlation model* [Wackernagel, 1998], a specific case of *co-kriging*. A similar model is also used by Conti and O'Hagan [2010] for emulating multi-output simulators. An extension to multiple factors has been explored by Wang et al. [2007] for modelling time-series motion capture data. Interpreted in terms of (4.15), each element of K^{f} in Wang et al.'s model is a product of covariance functions over the subject identity (represented by a vector $\boldsymbol{s}$) and the gait of locomotion (vector $\boldsymbol{g}$), while k^{x} is the covariance function over motion state. However, in that model K^{f} is parameterized by $\boldsymbol{s}$ and $\boldsymbol{g}$ and thus constrained accordingly, in contrast to the general positive semi-definite matrix in our model, the intrinsic correlation model and the model of Conti and O'Hagan. In addition, Wang et al. have applied their multi-factor model for learning latent spaces using the Gaussian process latent variable model [Lawrence, 2005] and not for regression tasks as explored in this chapter.

4.5 Summary and Further Discussions

This chapter focuses on learning inverse dynamics models for controlling a robot manipulator to follow a given trajectory, particularly for operating under multiple contexts when different loads are attached to the end-effector of the robot. Instead of modelling the specifics of individual robots, we have taken a general probabilistic function approximation approach based on Gaussian Processes, which is abstract enough to be applicable to broad families of robots. In section 4.2.2 we have looked at some characteristics of the torques, and have subsequently proposed covariance functions for modelling these. The functional forms of the proposed covariance function are also motivated by analytical robotics models, as discussed in sections 4.3.1 and 4.4.1. In particular, the linear-in-dynamic-parameters property of inverse dynamics in section 4.4.1 justifies the use of the multi-task GP regression model (mGP), which would otherwise be just an assumption. The results in section 4.3.4 have shown the importance of modelling the Coulomb frictions, while the results in section 4.4.6 have demonstrated the merits of using the mGP model to allow flexible sharing of data among the multiple contexts. Even though our experiments are on simulated data, it is nevertheless a realistic simulation based on estimated parameters of actual robots. We believe that the flexibility of GP will allow modelling of actual robots, based on the broad principles discussed in this chapter.

Below, we briefly discuss some possible extensions.

Coupling over joints The multi-context model of section 4.4.3 shares data among the different contexts to better predictive performance. The model is convenient to use because the prior, likelihood and posterior factorize over joints, so that inference and hyperparameter estimation can be done separately for each joint. Since inference is not coupled over joints, this allows *independent joint control* of the robot manipulator [Sciavicco and Siciliano, 2000, §6.3], which is useful for simplifying controller implementation.

However, models that also share data among the joints are possible if one wishes to make fuller use of limited data. The sharing of information among the ℓ inverse dynamics models, one for each joint, can be achieved through the common ℓ-by-ℓ inertial matrix $B(\boldsymbol{q})$, which is a matrix of functions on $\boldsymbol{q}$ (see section 4.3.1), in a manner similar to the *latent force model* of Alvarez et al. [2009]. However, because $B(\boldsymbol{q})$ is constrained to be positive definite, one cannot simply use a matrix of GP priors. One also needs to be aware that the Coriolis and centrifugal effects $H(\boldsymbol{q})$ depend on $B(\boldsymbol{q})$ in a non-trivial way, so that the eventual modelling of the torque functions can be rather involved. See the discussion in section 4.3.2 for more details.

A simpler model for sharing data among joints uses (4.14) with all the transformations A_is constrained to be the same. In this model, the inter-context correlations are common among joints, i.e. $K^{\mathrm{f}}_{\bullet} \stackrel{\mathrm{def}}{=} K^{\mathrm{f}}_1 \equiv \ldots \equiv K^{\mathrm{f}}_{\ell}$. Hyperparameter estimation needs to be done for the joints together, while inference can still be done separately for each joint.

Context estimation Suppose we have already estimated the hyperparameters of the multi-task GP model for M contexts using data $\mathcal{D}$. Then, given a new data $\mathcal{D}^* \stackrel{\mathrm{def}}{=} \{(\boldsymbol{x}^{(j)}, \boldsymbol{t}^{(j)})\}_{j=1}^{n^*}$, where $\boldsymbol{x}^{(j)} \in \mathbb{R}^{3\ell}$ and $\boldsymbol{t}^{(j)} \in \mathbb{R}^{\ell}$, assumed to be from one of the known contexts, one may wish to infer its identity. This problem is known as context estimation. We outline how this may be done below.

Let the observed torques in $\mathcal{D}^*$ for joint i be collected into $\boldsymbol{t}^*_i \stackrel{\mathrm{def}}{=} (t_i^{(1)}, \ldots, t_i^{(n^*)})^{\mathrm{T}}$, and also define $X^* \stackrel{\mathrm{def}}{=} \{\boldsymbol{x}^{(j)}\}_{j=1}^{n^*}$, the set of inputs in $\mathcal{D}^*$. Then, using the mGP model for joint i learnt from $\mathcal{D}$, we write $p(\boldsymbol{t}^*_i \mid X^*, \mathcal{D}, c_m)$ for the predictive probability that the vector of torques for the $i\mathrm{th}$ joint at X^* in the $m\mathrm{th}$ context equals to $\boldsymbol{t}^*_i$. Viewed as a function of c_m, this is the likelihood that the torques $\boldsymbol{t}^*_i$ at X^* are for the robot operating in context c_m. With these defined, we can follow Petkos et al. [2006, §3.1] and place a prior over contexts, and obtain the following posterior given $\mathcal{D}^*$:

$$P(c_m \mid \mathcal{D}, \mathcal{D}^*) \propto P(c_m) \prod_{i=1}^{\ell} p\left(\boldsymbol{t}^*_i \mid X^*, \mathcal{D}, c_m\right) \qquad m = 1 \ldots M. \tag{4.24}$$

One can use the c_m that gives the highest posterior as the estimated context. For on-line control of robots operating in an environment involving multiple contexts, the prior $P(c_m)$ can be time dependent in a Markovian way; see Petkos et al. [2006, §3.1].

Incorporating novel contexts It may be that the data $\mathcal{D}^*$ defined above is from a previously unobserved context c_*. Such cases can be handled in the following way. Having previously estimated the hyperparameters of the multi-task GP model for the M known contexts using data $\mathcal{D}$, we fix these hyperparameters while extending K^{f} by an extra row and column for the new context c_*. The entries in this new border are to be estimated using $\mathcal{D}^*$ by optimizing the marginal likelihood. Note that as K^{f} is positive semi-definite this means learning only at most M new parameters, fewer if we exploit the rank-constraint property of K^{f}.

Priors over hyperparameters In our treatment we have used point estimates for K^{f} and $\boldsymbol{\theta}^{\mathrm{x}}$. It would be possible to model uncertainty with respect to these parameters. For example it would be possible to place a prior over K^{f} as has been done in Conti and O'Hagan [2010]; they used the non-informative Jeffrey's prior $p(K^{\mathrm{f}}) \propto |K^{\mathrm{f}}|^{-(M+1)/2}$, but an inverse-Wishart prior would also be possible. In the case of noise free observations this prior can be integrated out analytically to give an M-variate Student's-t process. However, with noisy observations the justification for this is more problematic, as the noise and signal get entangled by common scaling factors; see, e.g., the discussion in Rasmussen and Williams [2006, §9.9].

Of course one could specify priors on K^{f} and $\boldsymbol{\theta}^{\mathrm{x}}$ and then resort to Markov chain Monte Carlo methods to sample from the posterior distribution. For example, Williams and Rasmussen (1996) have used the hybrid Monte Carlo method of Duane et al. [1987]. With s posterior samples of $\boldsymbol{\theta}^{\mathrm{x}}$, each prediction in this case is a weighted average of posterior means of s different Gaussian processes. However, for s large enough to accurately represent the posterior distribution, the computational demand may not be realistic to provide real-time feedforward torques needed for control.

Chapter 5

Conclusions and Future Work

Multi-task learning is an important paradigm in machine learning that pools knowledge from multiple tasks together in order to improve predictive performance. The intuition that multi-task learning is beneficial comes from the observation that we do this all the time: when encountered with multiple related tasks, we constantly use knowledge gained from one task to help another.

The key aspect of this paradigm is understanding *relatedness* between the tasks. Common notions of task relatedness are: (a) task clustering, where tasks within each cluster are related but tasks in different clusters are not; (b) a low-dimensional manifold on which tasks reside; (c) tasks have varying degrees of relatedness that influence the amount of information shared between tasks; and (d) different components of the tasks being related in different ways. These notions are introduced in chapter 1, where we have also distinguished between implicit and explicit modelling of task relatedness. In chapter 2 we have reviewed the hierarchical Bayesian latent source model, the multi-task regularization networks model and the Bayesian multi-task neural networks model. In these models, task relatedness is not directly specified — the notions of task relatedness are implicit through the sharing of selected aspects of the learning model for each task. Gaussian process multi-task learning models, which explicitly model task-relatedness via the cross-covariance between the latent functions for the tasks, are introduced in section 2.5. This explicit modelling of task-relatedness allows an average case analysis of multi-task learning using the intrinsic correlation model (ICM) for two tasks in chapter 3. There, we have related the correlation between the two tasks to the average predictive accuracy of the primary task of interest. In chapter 4, we have shown that learning the inverse dynamics of a robot manipulator for moving multiple loads along given trajectories falls naturally into the ICM multi-task GP model. Here, moving each load corresponds naturally to a task or context. Our experiments have demonstrated that this multi-task formulation is effective.

The next section gives a more detailed summary of our contributions. Some future research directions will be suggested in section 5.2.

5.1 Contributions

One attraction in using Gaussian processes for multi-task learning is that task-relatedness can be explicitly and transparently encoded in the cross-covariance function between the tasks. Therefore, we can use Gaussian processes to understand other models of multi-task learning, by making explicit the task-relatedness in existing models. This realization seeds the main contribution of **chapter 2**: using the linear model of coregionalization, which is a multi-task Gaussian process model, we have given a unified view over the hierarchical Bayesian latent source model, the multi-task regularization networks model, and the multi-task Bayesian neural network model. In addition, section 2.3 gives a correction to the Proposition 1 of Evgeniou et al. [2005].

In **chapter 3**, we have analyzed the multi-task Gaussian process model to understand how and the extent to which multi-task learning helps improve the generalization of supervised learning. Our analysis is for the average-case, in contrast and orthogonal to existing works, which deal with worst-case analysis using uniform convergence bounds. As far as we are aware, this is the first average-case analysis of multi-task learning that directly addresses predictive errors. We focus on the learning of two tasks under a specific Gaussian process multi-task model: the intrinsic correlation model (ICM) with observation noise. In addition, we concentrate mainly on asymmetric multi-task learning, in which case the two tasks can be referred to as the primary and the second task, where the secondary task is to help the learning of the primary task. Within this setting, the main parameters are (1) the degree of "relatedness" ρ between the two tasks, and (2) π_S, the fraction of the total training observations from the secondary task. Our results are expressed explicitly terms of ρ and π_S. Again, this is in contrast to existing results, which (1) are not expressed explicitly in terms of the relatedness among the tasks; and (2) only deal with the case where each task has the same number of samples. We believe that it is important to analyze multi-task learning under uneven distributions, since one of the motivations of using multi-task learning is to allow a secondary task, for which there is abundant data, to help a primary task, for which there is few data.

We have provided bounds on the generalization errors and the learning curves in chapter 3, and these bounds highlight the effects of ρ and π_S. For the case of no training data for the primary task, we have shown that the error cannot be lower than $(1-\rho^2)\sum \kappa_i$ if one learns with only observations from the secondary task. We have also given the lower bound $\underline{\sigma}_T^2$ (Proposition 3.5 on page 60) on the posterior variance that is tight, simple and useful for subsequent analysis. The term $\pi_S(1-\pi_S)(1-\rho^2)$ has been demonstrated to have a governing role for symmetric multi-task learning, where the predictive accuracy of the secondary task is also considered. For asymmetric multi-task learning, we have shown that the lower bound on the learning curve cannot be less than $(1-\rho^2\pi_S)$ of that from single-task learning. For both symmetric and asymmetric multi-task learning, we have shown that the influence of the secondary task on the primary task is larger for smoother processes than for rougher processes. We have also investigated the effective number of observations contributed by the secondary task to the primary task.

Another contribution in chapter 3 is the generalization of two known results from the geostatistics literature concerning multi-collocated and collocated noiseless observations. This contribution extends Chilès and Delfiner [1999]'s results from two tasks to more than two tasks, and provides an understanding of the ICM multi-task Gaussian process model through the structures in the predictive means and variances.

In **chapter 4**, we have evaluated multi-task Gaussian process on learning inverse dynamics models for controlling a robot manipulator to follow a given trajectory operating under multiple contexts, when different loads are attached to the end-effector of the robot. Instead of modelling the specifics of individual robots, we have taken a general probabilistic function approximation approach and have placed Gaussian process priors over the space of functions. We believe this is abstract enough to be applicable to broad families of robots. Our covariance function for the inverse dynamics within a single context is motivated by analyzing the dynamics and understanding how the torques are generated. We have shown that the Gaussian process with the chosen covariance function is effective for learning the inverse dynamics for a single context.

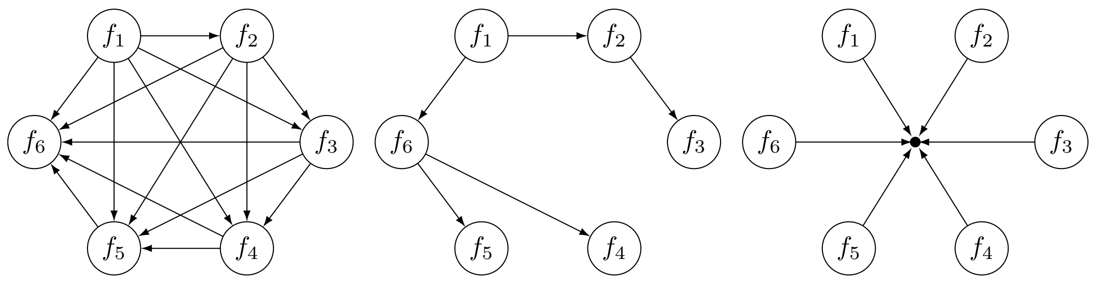
For multiple contexts, we have used the linear-in-dynamic-parameters property of inverse dynamics to motivate multi-task Gaussian process regression, specifically the ICM with noisy observations. We have shown that the inter-context similarity depends on the underlying inertial parameters of the robot manipulator. We have also used the Akaike information criterion (with a second order correction) for selecting the underlying dimensionality of the multi-task Gaussian process, pooling together information from all joints. Our experiments have demonstrated that this multi-task formulation is effective in sharing information among the various loads, and generally improves performance over either learning only on single contexts or pooling the data over all contexts.

5.2 Future Work

5.2.1 Inter-domain Multi-task Learning

Much of the research in multi-task learning is for tasks acting on the same input space, and it is not immediately clear how one may model the case where the input spaces differ. Perhaps the most direct way is to use transformations from one input space into another. A theory of multi-task learning that takes this view is provided by Ben-David and Schuller Borbely [2008], and in section 2.5.8 we have provided a multi-task Gaussian process construction that makes use of such transformations for learning inter-domain tasks. A naïve implementation of this form of multi-task learning requires $M(M-1)/2$ pairwise transformations for M tasks each acting on a different input space. One may also arrange the transformations to follow a hierarchy so that only $(M-1)$ transformations are needed. Alternatively, Evgeniou et al. [2005] suggest considering a kernel defined on the product space of all the input spaces, and they suggest implementing this by using linear transformations to project the different input spaces onto a common linear subspace. This will give M transformations to the common space. The above three topologies are illustrated in Figure 5.1. One possible future research direction is to investigate these three topologies, and perhaps also other topologies, and to make learning the transformations feasible and robust.

A related but simpler situation is when the outputs of the tasks are different, for example, if one is a regression task and the other a classification task. For this, we can still couple the latent functions using multi-task learning, but use different link functions (as in generalized linear models) for each task; see



(a) $M(M-1)/2$ pairwise transformations

(b) $M-1$ transformations in a hierarchy

(c) M transformations to a common space

Figure 5.1: Multi-task learning for six functions, $f_1, \ldots, f_6$, each acting on a different input space. Three possible topologies are illustrated above, where an arrow represents a transformation of the input space. For (c), the solid circle in the centre represents the common space onto which all other input spaces map.

Dunson [2000]; Gueorguieva and Agresti [2001]; Yang et al. [2009].

5.2.2 Further Analysis on Multi-task Learning

In chapter 3, we have analyzed multi-task learning where there are two tasks. This has been a convenient choice for analysis since the relation between the two tasks can be given completely by the single parameter $\rho \in [-1, 1]$ that measures the correlation or relatedness between the two tasks. A more thorough theory of multi-task learning is one for $M > 2$ tasks. This will involve the $\Theta(M^2)$ parameters in the task correlation matrix K^{f} for describing task relations. For symmetric multi-task learning, which can be regarded as single-task learning on an expanded input space that includes the task indicators, the performance of all the tasks will depend on the M eigenvalues of K^{f}. However, the analysis is not so straightforward for asymmetric multi-task learning, where we are interested mainly in the performance of one of the M tasks, called the primary task. Perhaps here it is worthwhile to limit the analysis to particular cases, for example, when K^{f} is rank-constrained, when K^{f} is an equi-correlated matrix (i.e., all the inter-task correlations are the same), or when the correlations in K^{f} are bounded within a sub-range of $[-1, 1]$. The challenge is for the results to make sense in terms of intuitive quantities such as the Schur complement $\upsilon_T^2 \overset{\mathrm{def}}{=} 1 - \boldsymbol{\varrho}^{\mathrm{T}}(K_S^{\mathrm{f}})^{-1}\boldsymbol{\varrho}$ given in equation 3.77a on page 94. This will be an interesting area for future research.

Orthogonal to the above, it is also useful to consider the case where the training set is not partitioned exclusively among the tasks. For example, for two tasks, called task T and task S, one may wish to partition the training set locations into fraction π_T for task T, fraction π_S for task S, and fraction $(1 - \pi_T - \pi_S)$ that is shared between task T and task S.[1] A similar setup has been used in chapter 4, where there are 15 contexts sharing training locations sampled from a reference trajectory.

Another area for future research is to analyze when the input spaces for the tasks are different. Again, it will prove useful to draw inspiration from particular cases. For two tasks, suppose there is already a deterministic mapping from each task to a common real vector space $\mathbb{R}^D$, for some D, so that we

[1] This was suggested by E. Bonilla when I presented the preliminary results of chapter 3 at the Statistical and Machine Learning Interface Meeting in the University of Manchester, 23rd-24th July 2009.

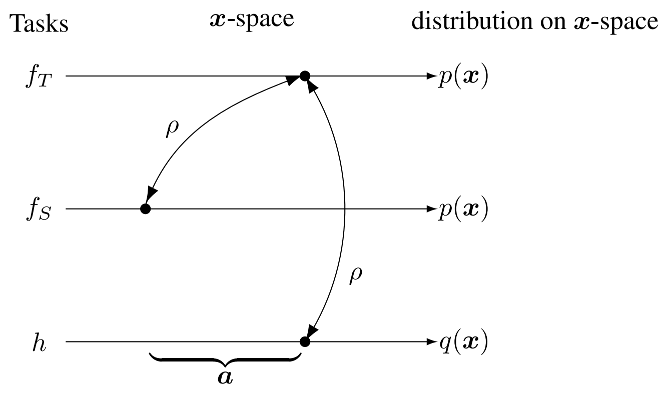


Figure 5.2: A case of inter-domain transfer between f_T and f_S. Function h is a translation of f_S in the $\boldsymbol{x}$-space by $\boldsymbol{a}$, and f_T and h are related via the ICM multi-task Gaussian process model, with inter-task correlation ρ. The distribution on the input $\boldsymbol{x}$ is $p(\boldsymbol{x})$ for both f_T and f_S, but h has a different input distribution $q(\boldsymbol{x})$.

can restrict our attention to two functions f_T and f_S on $\mathbb{R}^D$. We would like to analyze the multi-task learning of f_T and f_S. If there is a function $h(\boldsymbol{x}) \stackrel{\text{def}}{=} f_S(\boldsymbol{x} - \boldsymbol{a})$ such that f_T and h jointly have the multi-task Gaussian process prior as per chapter 3, with an isotropic covariance function k^{x} on the input space, then f_T and f_S are related by an inter-domain Gaussian process in the manner section 2.5.8 on page 40. In this case, the transformation g is a translation in the $\boldsymbol{x}$-space by $\boldsymbol{a}$. This is illustrated in Figure 5.2.

A direct analysis of the multi-task learning between f_T and f_S is to translate f_S into h, and apply the generalization errors results in chapter 3 on f_T and h. For the learning curves, we will need to specify the input distributions for f_T and f_S. If both of f_T and f_S have the same input distributions $p(\boldsymbol{x})$, then f_T and h will not have the same distribution — the distribution $q(\boldsymbol{x})$ for h is a translation of that for f_T, i.e., $q(\boldsymbol{x}) = p(\boldsymbol{x} - \boldsymbol{a})$. Analysis may still be possible but could be rather limited.

5.2.3 Task Identity Determination and Learning to Learn

A future direction is to try the multi-task Gaussian process learning model on task identity determination and learning to learn. These have already been pointed out in section 4.5 on page 126 for learning the inverse dynamics of a robotic manipulator for multiple contexts. We reiterate these here in more generality; further details can be found in section 4.5.

Task identity determination Suppose we have already learnt a multi-task Gaussian process model for M tasks. Then given a set of new data $\mathcal{D}^* \stackrel{\text{def}}{=} \{(\boldsymbol{x}^{(j)}, \boldsymbol{y}^{(j)})\}_{j=1}^{n^*}$, the aim is to infer which of the M tasks most likely generated $\mathcal{D}^*$.

Learning to learn It may be that the data $\mathcal{D}^*$ defined above is from a previously unobserved task. We would like to know if this is indeed the case. If $\mathcal{D}^*$ is from a novel task, then we would like to learn this task by incorporating $\mathcal{D}^*$ into the multi-task model that we have already learnt for the existing M tasks. One way to achieve this in the ICM Gaussian process model is to extend K^{f} by an extra row and column for the new task, and estimate the entries in this new border by optimizing the marginal likelihood.

Appendix A

Appendix to Chapter 2

In this appendix, we provide material supplementary to chapter 2.

A.1 Derivations for Multi-task Regularization Networks Model

This section contains derivations for the multi-task regularization networks model of Evgeniou, Micchelli, and Pontil [2005] discussed in section 2.3.

A.1.1 Derivation for Linear Multi-task Kernel

Consider two functions $\underline{f}$ and $\underline{f}'$ which belong to the Hilbert space $\mathcal{H}$:

$$\underline{f}(m, \boldsymbol{x}) = \boldsymbol{u}^{\mathrm{T}} \Phi_m \boldsymbol{x} \qquad\qquad \underline{f}'(m, \boldsymbol{x}) = (\boldsymbol{u}')^{\mathrm{T}} \Phi_m \boldsymbol{x}.$$

If we assert that the squared norm of these functions is $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}$, that is,

$$\|\underline{f}\|^2_{\mathcal{H}} = \boldsymbol{u}^{\mathrm{T}}\boldsymbol{u},$$

then we have the inner product

$$\langle \underline{f}, \underline{f}' \rangle = \boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}' \tag{$*$}$$

in $\mathcal{H}$ so that $\|\underline{f}\|^2_{\mathcal{H}} \overset{\text{def}}{=} \langle \underline{f}, \underline{f} \rangle$ will give $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}$. Let the reproducing kernel of $\mathcal{H}$ be $k(\cdot, \cdot)$. Then

$$\langle f(\cdot), k(\cdot, (m', \boldsymbol{x}')) \rangle = f(m', \boldsymbol{x}') = \boldsymbol{u}^{\mathrm{T}} \Phi_{m'} \boldsymbol{x}' \tag{$\dagger$}$$

by the reproducing property. Since $k(\cdot, (m', \boldsymbol{x}')) \in \mathcal{H}$, we may choose $k(\cdot, (m', \boldsymbol{x}')) = \underline{f}'(\cdot)$ without any lost of generality so that we can identify $\Phi_{m'}\boldsymbol{x}'$ with $\boldsymbol{u}'$ by matching equations $*$ and $\dagger$. Then by evaluating $\underline{f}'$ at $(m, \boldsymbol{x})$,

$$k((m, \boldsymbol{x}), (m', \boldsymbol{x}')) = \underline{f}'(m, \boldsymbol{x}) = \left(\Phi_{m'}\boldsymbol{x}'\right)^{\mathrm{T}} \Phi_m \boldsymbol{x} = \boldsymbol{x}^{\mathrm{T}} \Phi_m^{\mathrm{T}} \Phi_{m'} \boldsymbol{x}'. \qquad \square$$

A.1.2 Proof of Proposition 2.1

We wish to show the equivalence of the following two objective functions

$$\frac{1}{MN}\sum_{m=1}^{M}\sum_{i=1}^{N} L\big(y_{mi}\,,\, \boldsymbol{w}_m^{\mathrm{T}}\boldsymbol{x}_{mi}\big) + \lambda \underline{\boldsymbol{w}}^{\mathrm{T}} E \underline{\boldsymbol{w}}, \tag{A.1a}$$

$$\frac{1}{MN}\sum_{m=1}^{M}\sum_{i=1}^{N} L\big(y_{mi}\,,\, \boldsymbol{u}^{\mathrm{T}}\Phi_m\boldsymbol{x}_{mi}\big) + \lambda \boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}, \tag{A.1b}$$

where $\boldsymbol{w}_m \in \mathbb{R}^D$, $\boldsymbol{u} \in \mathbb{R}^{D^\Phi}$, $\Phi_m \in \mathbb{R}^{D^\Phi \times D}$, $E \in \mathbb{R}^{MD\times MD}$ and

$$\boldsymbol{w}_m = \Phi_m^{\mathrm{T}}\boldsymbol{u} \qquad\qquad \underline{\boldsymbol{w}} \overset{\text{def}}{=} (\boldsymbol{w}_1^{\mathrm{T}}, \ldots, \boldsymbol{w}_M^{\mathrm{T}})^{\mathrm{T}}. \tag{A.2}$$

The equivalence will be shown in two directions in the following two sections.

A.1.2.1 Proof of Proposition 2.1a

In this part of the proof, we show that objective function (A.1b) implies objective function (A.1a). That is, given any set of $\{\Phi_m\}$, we can construct an E such that the two objective functions have the same value. The equivalence is to be shown under the conditions that $D^\Phi \leqslant MD$. Let

$$\underline{\Phi} \overset{\text{def}}{=} (\Phi_1 \mid \Phi_2 \mid \cdots \mid \Phi_M) \in \mathbb{R}^{D^\Phi \times DM} \qquad\qquad E = \left(\underline{\Phi}^{\mathrm{T}}\underline{\Phi}\right)^{+}. \tag{A.3}$$

The construction of E is the left equation above. Note that $\underline{\Phi}$ has rank D^Φ. We can stack $\boldsymbol{w}_m = \Phi_m^{\mathrm{T}}\boldsymbol{u}$ (given by equation A.2) vertically and write

$$\underline{\boldsymbol{w}} = \underline{\Phi}^{\mathrm{T}}\boldsymbol{u}. \tag{A.4}$$

The equivalence of the loss function terms in the objective functions follows directly from substituting in (A.2). For the regularizer, the following lemma is required.

Lemma A.1. *[Barnet, 1990; Petersen and Pedersen, 2008]. For an n-by-m matrix A of rank r, let $A = CD$, where C is an n-by-r matrix and D is a r-by-m matrix, and both matrices are of rank r. Then*

$$A^{+} = D^{\mathrm{T}}\left(DD^{\mathrm{T}}\right)^{-1}\left(C^{\mathrm{T}}C\right)^{-1}C^{\mathrm{T}}.$$

Since $\underline{\Phi}$ is of rank D^Φ, we may apply the lemma on E to give

$$E = \left(\underline{\Phi}^{\mathrm{T}}\underline{\Phi}\right)^{+} = \underline{\Phi}^{\mathrm{T}}\left(\underline{\Phi}\,\underline{\Phi}^{\mathrm{T}}\right)^{-1}\left(\underline{\Phi}\,\underline{\Phi}^{\mathrm{T}}\right)^{-1}\underline{\Phi}, \tag{A.5}$$

so

$$\underline{\Phi}E\underline{\Phi}^{\mathrm{T}} = \underline{\Phi}\left[\underline{\Phi}^{\mathrm{T}}\left(\underline{\Phi}\,\underline{\Phi}^{\mathrm{T}}\right)^{-1}\left(\underline{\Phi}\,\underline{\Phi}^{\mathrm{T}}\right)^{-1}\underline{\Phi}\right]\underline{\Phi}^{\mathrm{T}} = \left(\underline{\Phi}\underline{\Phi}^{\mathrm{T}}\right)\left(\underline{\Phi}\,\underline{\Phi}^{\mathrm{T}}\right)^{-1}\left(\underline{\Phi}\,\underline{\Phi}^{\mathrm{T}}\right)^{-1}\left(\underline{\Phi}\underline{\Phi}^{\mathrm{T}}\right) = I, \tag{A.6}$$

where the identity matrix is of size D^Φ-by-D^Φ. Then, using $\underline{\boldsymbol{w}} = \underline{\Phi}^{\mathrm{T}}\boldsymbol{u}$ (equation A.4), we have

$$\underline{\boldsymbol{w}}^{\mathrm{T}}E\underline{\boldsymbol{w}} = \left(\boldsymbol{u}^{\mathrm{T}}\underline{\Phi}\right)E\left(\underline{\Phi}^{\mathrm{T}}\boldsymbol{u}\right) = \boldsymbol{u}^{\mathrm{T}}\left(\underline{\Phi}E\underline{\Phi}^{\mathrm{T}}\right)\boldsymbol{u} = \boldsymbol{u}^{\mathrm{T}}I\boldsymbol{u} = \boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}, \tag{A.7}$$

which completes the proof. □

A.1.2.2 Proof of Proposition 2.1b

Given any MD-by-MD positive semi-definite matrix E of rank $D^{\Phi} \leqslant MD$, we show how to construct $\underline{\Phi}$ for which the objective functions in (A.1) are equivalent. This is done using the following lemma.

Lemma A.2. *Let A be an n-by-n positive semi-definite matrix of rank r. Then we can construct a r-by-n matrix B such that $BAB^{\mathrm{T}} = I_{r \times r}$.*

Proof. By construction. Matrix A is positive semi-definite, and so it can be expressed as CC^{T}, where C is a n-by-r matrix of rank r. The matrix C is called a matrix square root of A. For example, C can be the incomplete Cholesky decomposition of A, or a its columns vectors can be the first r principal eigenvectors of A, each scaled by the reciprocal of the square-root of its eigenvalues. Then for any r-by-r unitary matrix U, $B = UC^{\mathrm{T}}A^{+}$ will satisfy $BAB^{\mathrm{T}} = I_{r \times r}$. This is evident when we expand A^{+} using Lemma A.1 on the facing page. □

Using the construction in the proof above, we let C be any matrix square root of E. It is necessary that C is a full rank MD-by-D^{Φ} matrix. To satisfy $\underline{\Phi}E\underline{\Phi}^{\mathrm{T}} = I_{D^{\Phi}}$, we may set

$$\underline{\Phi} = C^{\mathrm{T}}E^{+}. \tag{A.8}$$

Having determined a $\underline{\Phi}$ such as the above, we may use it to generate other suitable $\underline{\Phi}$s in the following way. Let X be an MD-by-D^{Φ} matrix whose column vectors are in the null space of C^{T}, i.e., $C^{\mathrm{T}}X = 0$. Define

$$\underline{\Phi}' = \underline{\Phi} + X^{\mathrm{T}}. \tag{A.9}$$

Then,

$$\underline{\Phi}'E(\underline{\Phi}')^{\mathrm{T}} = \left(\underline{\Phi} + X^{\mathrm{T}}\right)CC^{\mathrm{T}}\left(\underline{\Phi} + X^{\mathrm{T}}\right)^{\mathrm{T}} = \left(\underline{\Phi}C + X^{\mathrm{T}}C\right)\left(C^{\mathrm{T}}\underline{\Phi}^{\mathrm{T}} + C^{\mathrm{T}}X\right) = \underline{\Phi}CC^{\mathrm{T}}\underline{\Phi}^{\mathrm{T}} = I_{D^{\Phi}}.$$

Thus, using $\underline{\Phi}'$ will also make the objective functions in equation A.1 equivalent. □

A.1.3 Derivation of Equation 2.40

To prove equation 2.40, which is

$$k((\boldsymbol{x}, m), (\boldsymbol{x}', m')) = K^{\mathrm{f}}_{mm'}\boldsymbol{x}^{\mathrm{T}}(E^{\mathrm{x}})^{+}\boldsymbol{x}',$$

we first introduce a corollary of lemma A.1 on the preceding page.

Corollary A.3. *Kronecker Product of Moore-Penrose Pseudo-inverses. The pseudo-inverse of a Kronecker product is the Kronecker product of the pseudo-inverses.*

Proof. Let A be an n-by-m matrix of rank r, and B be a p-by-q matrix of rank s. Then there exist an n-by-r matrix C, a r-by-m matrix D, a p-by-s matrix E and a s-by-q matrix F, all of full rank, such that $A = CD$ and $B = EF$. Using Lemma A.1 on the facing page, the mix-product property of Kronecker products, and the distributivity of transposition over Kronecker products, we have

$$\begin{aligned}
(A \otimes B)^+ &= (CD \otimes EF)^+ \\
&= ((C \otimes E)(D \otimes F))^+ \\
&= (D \otimes F)^{\mathrm{T}} \left[(D \otimes F)(D \otimes F)^{\mathrm{T}}\right]^{-1} \left[(C \otimes E)^{\mathrm{T}}(C \otimes E)\right]^{-1} (C \otimes E) \\
&= \left(D^{\mathrm{T}} \otimes F^{\mathrm{T}}\right) \left[DD^{\mathrm{T}} \otimes FF^{\mathrm{T}}\right]^{-1} \left[C^{\mathrm{T}}C \otimes E^{\mathrm{T}}E\right]^{-1} (C \otimes E) \\
&= \left(D^{\mathrm{T}} \otimes F^{\mathrm{T}}\right) \left[(DD^{\mathrm{T}})^{-1} \otimes (FF^{\mathrm{T}})^{-1}\right] \left[(C^{\mathrm{T}}C)^{-1} \otimes (E^{\mathrm{T}}E)^{-1}\right] (C \otimes E) \\
&= \left(D^{\mathrm{T}} \left(DD^{\mathrm{T}}\right)^{-1} \left(C^{\mathrm{T}}C\right)^{-1} C^{\mathrm{T}}\right) \otimes \left(F^{\mathrm{T}} \left(FF^{\mathrm{T}}\right)^{-1} \left(E^{\mathrm{T}}E\right)^{-1} E^{\mathrm{T}}\right) \\
&= A^+ \otimes B^+
\end{aligned}$$

□

By Proposition 2.1b on page 22, we can let $\underline{\Phi} = CC^{\mathrm{T}}E^+$, where $CC^{\mathrm{T}} = E$. Then

$$\begin{aligned}
\underline{\Phi}^{\mathrm{T}}\underline{\Phi} &= E^{+\mathrm{T}}CC^{\mathrm{T}}E^+ \\
&= E^+EE^+ && \text{because } CC^{\mathrm{T}} = E \text{ and } E \text{ is symmetric} \\
&= E^+ && \text{using Lemma A.1} \\
&= (E^{\mathrm{f}} \otimes E^{\mathrm{x}})^+ \\
&= (E^{\mathrm{f}})^+ \otimes (E^{\mathrm{x}})^+ && \text{using Corollary A.3}
\end{aligned}$$

By the definition of $\underline{\Phi} \overset{\mathrm{def}}{=} (\Phi_1 \mid \cdots \mid \Phi_M)$ and equation 2.35 on page 22, which gives

$$k((m, \boldsymbol{x}), (m', \boldsymbol{x}')) = \boldsymbol{x}^{\mathrm{T}}\Phi_m^{\mathrm{T}}\Phi_{m'}\boldsymbol{x}',$$

we have

$$k((m, \boldsymbol{x}), (m', \boldsymbol{x}')) = \boldsymbol{x}^{\mathrm{T}}(E^{\mathrm{f}})^+_{mm'}(E^{\mathrm{x}})^+\boldsymbol{x}'.$$ □

A.1.4 Manifold Learning

We now describe how manifold learning may be achieved using multi-task kernels. We shall start from equation 2.31 on page 21, where the hypothesis for the $m\mathrm{th}$ task is identified by the vector $\boldsymbol{w}_m$. Let the linear subspace of the D^{s} dimensional manifold be given by the column space of a D-by-D^{s} matrix A located at $\boldsymbol{\mu}$. Without loss of generality, let the column vectors of A be orthonormal vectors spanning the subspace. For the $m\mathrm{th}$ task, the projection $\mathbb{P}\boldsymbol{w}_m$ of $\boldsymbol{w}_m$ on the manifold is given by

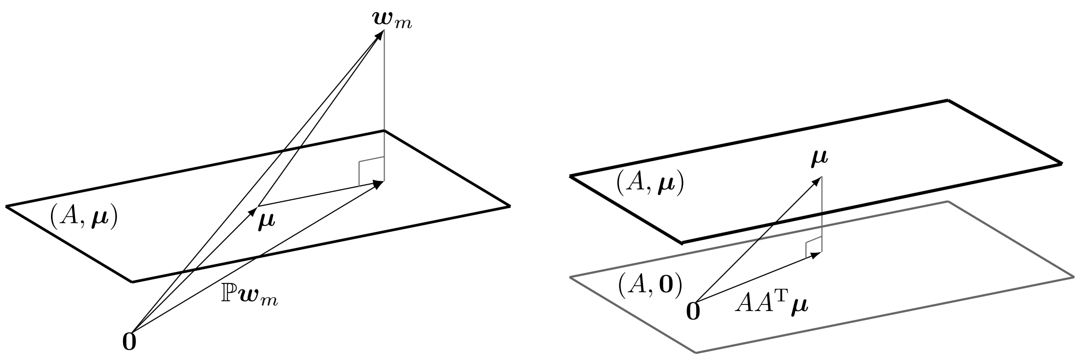
$$\mathbb{P}\boldsymbol{w}_m = AA^{\mathrm{T}}(\boldsymbol{w}_m - \boldsymbol{\mu}) + \boldsymbol{\mu} \qquad A \in \mathbb{R}^{D \times D^{\mathrm{s}}} \tag{A.10a}$$

This is depicted in Figure A.1a. The L_2 distance between $\boldsymbol{w}_m$ and its projection is

$$\|\boldsymbol{w}_m - \mathbb{P}\boldsymbol{w}_m\|_2 = \sqrt{(\boldsymbol{w}_m - \mathbb{P}\boldsymbol{w}_m)^{\mathrm{T}}(\boldsymbol{w}_m - \mathbb{P}\boldsymbol{w}_m)}. \tag{A.10b}$$

This may be simplified using

$$\boldsymbol{w}_m - \mathbb{P}\boldsymbol{w}_m = \boldsymbol{w}_m - (AA^{\mathrm{T}}(\boldsymbol{w}_m - \boldsymbol{\mu}) + \boldsymbol{\mu}) = I(\boldsymbol{w}_m - \boldsymbol{\mu}) - AA^{\mathrm{T}}(\boldsymbol{w}_m - \boldsymbol{\mu}) = (I - AA^{\mathrm{T}})(\boldsymbol{w}_m - \boldsymbol{\mu})$$



(a) The projection of $\boldsymbol{w}_m$ onto the manifold $(A,\boldsymbol{\mu})$ (b) Distance of the manifold $(A,\boldsymbol{\mu})$ from the origin

Figure A.1: Manifold of predictors using multi-task kernels.

so that

$$\|\boldsymbol{w}_m - \mathbb{P}\boldsymbol{w}_m\|_2 = \sqrt{(\boldsymbol{w}_m-\boldsymbol{\mu})^{\mathrm{T}}(I-AA^{\mathrm{T}})^2(\boldsymbol{w}_m-\boldsymbol{\mu})} \tag{A.10c}$$

The aim of manifold learning is to minimize the L_2 distances between all the task vectors and their projection. In addition, it is important to regularize the offset of the manifold, and we shall see why later. As depicted in Figure A.1b, the perpendicular vector from the origin to the manifold is $\boldsymbol{\mu} - AA^{\mathrm{T}}\boldsymbol{\mu}$, so the distance from the manifold to the origin is

$$\|\boldsymbol{\mu} - AA^{\mathrm{T}}\boldsymbol{\mu}\|_2 = \sqrt{(\boldsymbol{\mu} - AA^{\mathrm{T}}\boldsymbol{\mu})^{\mathrm{T}}(\boldsymbol{\mu} - AA^{\mathrm{T}}\boldsymbol{\mu})} = \sqrt{\boldsymbol{\mu}^{\mathrm{T}}\left(I-AA^{\mathrm{T}}\right)^2\boldsymbol{\mu}}. \tag{A.10d}$$

Taking these into consideration, we have the following regularizer

$$\min_{\boldsymbol{\mu}}\left[\sum_{m=1}^{M}(\boldsymbol{w}_m-\boldsymbol{\mu})^{\mathrm{T}}(I-AA^{\mathrm{T}})^2(\boldsymbol{w}_m-\boldsymbol{\mu}) + \lambda_{\mu}\boldsymbol{\mu}^{\mathrm{T}}(I-AA^{\mathrm{T}})^2\boldsymbol{\mu}\right], \tag{A.10e}$$

where λ_{μ} is parameter controlling the extend of regularizing the offset of the manifold. The minimization can be achieved by setting the gradient of the objective with respect to $\boldsymbol{\mu}$ to zero:

$$-2\sum_{m=1}^{M}(I-AA^{\mathrm{T}})^2(\boldsymbol{w}_m-\boldsymbol{\mu}) + 2\lambda_{\mu}(I-AA^{\mathrm{T}})^2\boldsymbol{\mu} = \boldsymbol{0} \quad\Longrightarrow\quad -\sum_{m=1}^{M}(\boldsymbol{w}_m-\boldsymbol{\mu}) + \lambda_{\mu}\boldsymbol{\mu} = \boldsymbol{0}$$

$$\therefore \boldsymbol{\mu} = \frac{1}{\lambda_{\mu}+M}\boldsymbol{s}, \tag{A.11}$$

where $\boldsymbol{s} \overset{\text{def}}{=} \sum_{m=1}^{M}\boldsymbol{w}_m$. Hence the offset of the manifold from the origin is a weighted sum of the task parameters $\boldsymbol{w}_m$s. The weighted sum is less than the average, reflecting the preference for the manifold to be near the origin. Substituting (A.11) into the objective in (A.10e) and expanding gives:

$$\begin{aligned}
&\sum_{m=1}^{M}\left(\boldsymbol{w}_m - \frac{1}{\lambda_{\mu}+M}\boldsymbol{s}\right)^{\mathrm{T}}\left(I-AA^{\mathrm{T}}\right)^2\left(\boldsymbol{w}_m - \frac{1}{\lambda_{\mu}+M}\boldsymbol{s}\right) + \frac{\lambda_{\mu}}{(\lambda_{\mu}+M)^2}\boldsymbol{s}^{\mathrm{T}}\left(I-AA^{\mathrm{T}}\right)^2\boldsymbol{s} \\
&= \sum_{m=1}^{M}\boldsymbol{w}_m^{\mathrm{T}}(I-AA^{\mathrm{T}})^2\boldsymbol{w}_m - \frac{2}{\lambda_{\mu}+M}\underbrace{\textstyle\sum_{m=1}^{M}\boldsymbol{w}_m^{\mathrm{T}}}_{\boldsymbol{s}^{\mathrm{T}}}\left(I-AA^{\mathrm{T}}\right)^2\boldsymbol{s} \\
&\qquad + \frac{M}{(\lambda_{\mu}+M)^2}\boldsymbol{s}^{\mathrm{T}}\left(I-AA^{\mathrm{T}}\right)^2\boldsymbol{s} + \frac{\lambda_{\mu}}{(\lambda_{\mu}+M)^2}\boldsymbol{s}^{\mathrm{T}}\left(I-AA^{\mathrm{T}}\right)^2\boldsymbol{s} \\
&= \sum_{m=1}^{M}\boldsymbol{w}_m^{\mathrm{T}}(I-AA^{\mathrm{T}})^2\boldsymbol{w}_m - \frac{1}{\lambda_{\mu}+M}\boldsymbol{s}^{\mathrm{T}}\left(I-AA^{\mathrm{T}}\right)^2\boldsymbol{s}.
\end{aligned}$$

Writing the expression in terms of $\underline{\boldsymbol{w}} \overset{\text{def}}{=} (\boldsymbol{w}_1^{\mathrm{T}}, \dots, \boldsymbol{w}_M^{\mathrm{T}})^{\mathrm{T}}$ gives

$$\begin{aligned}\underline{\boldsymbol{w}}^{\mathrm{T}} \left(I \otimes (I - AA^{\mathrm{T}})^2\right) \underline{\boldsymbol{w}} - \frac{1}{\lambda_\mu + M} \underline{\boldsymbol{w}}^{\mathrm{T}} \left(1_{M\times M} \otimes (I - AA^{\mathrm{T}})^2\right) \underline{\boldsymbol{w}} \\ = \underline{\boldsymbol{w}}^{\mathrm{T}} \left[\left(I - \frac{1}{\lambda_\mu + M} 1_{M\times M}\right) \otimes (I - AA^{\mathrm{T}})^2\right] \underline{\boldsymbol{w}},\end{aligned}$$

where $1_{M\times M}$ is the M-by-M matrix of ones. Therefore the regularizer (A.10e) can be written in the separable form $\underline{\boldsymbol{w}}^{\mathrm{T}} \left(E^{\mathrm{f}} \otimes E^{\mathrm{x}}\right) \underline{\boldsymbol{w}}$ discussed in section 2.3.1, setting

$$E^{\mathrm{f}} = I_{M\times M} - \frac{1}{\lambda_\mu + M} 1_{M\times M} \qquad E^{\mathrm{x}} = (I_{D\times D} - AA^{\mathrm{T}})^2. \tag{A.12a}$$

Define the D-by-$(D - D^{\mathrm{s}})$ matrix $A_\perp$ of which the column vectors are orthonormal vectors spanning the complement of the subspace of the manifold. Then the concatenation $(A \mid A_\perp)$ is unitary, so

$$E^{\mathrm{x}} = \left[\begin{pmatrix} A & A_\perp \end{pmatrix} \begin{pmatrix} A & A_\perp \end{pmatrix}^{\mathrm{T}} - AA^{\mathrm{T}}\right]^2 = \left(A_\perp A_\perp^{\mathrm{T}}\right)^2 = A_\perp A_\perp^{\mathrm{T}}, \tag{A.12b}$$

To obtain the multi-task kernel, we need to invert E^{f} and E^{x}. Using the Sherman-Morrison formula to invert E^{f}, we have

$$\left(E^{\mathrm{f}}\right)^{-1} = I_{M\times M} + \frac{1}{\lambda_\mu} 1_{M\times M}. \tag{A.12c}$$

We can now see the importance of regularizing the distance of the manifold to the origin: if regularization is absent, then $\lambda_\mu = 0$, so E^{f} is singular.

The column vectors of $A_\perp$ are orthonormal, so we have $A_\perp^{\mathrm{T}} A_\perp = I$. Then, using Lemma A.1 on page 136, we obtain

$$(E^{\mathrm{x}})^+ = A_\perp (A_\perp^{\mathrm{T}} A_\perp)^{-1} (A_\perp^{\mathrm{T}} A_\perp)^{-1} A_\perp^{\mathrm{T}} = A_\perp A_\perp^{\mathrm{T}} = E^{\mathrm{x}}. \tag{A.12d}$$

Thus the multi-task kernel for learning a low-dimensional manifold is given by

$$k((\boldsymbol{x}, m), (\boldsymbol{x}', m')) = \left(\delta_{mm'} + \frac{1}{\lambda_\mu}\right) \boldsymbol{x}^{\mathrm{T}} A_\perp A_\perp^{\mathrm{T}} \boldsymbol{x}' \tag{A.13}$$

Given $E = E^{\mathrm{f}} \otimes E^{\mathrm{x}}$, we may construct a $\underline{\Phi}$ using Proposition 2.1b on page 22. For this purpose, a matrix square root of E is

$$\left(I_{M\times M} + \lambda'_\mu 1_{M\times M}\right) \otimes A_\perp \qquad \text{where } \lambda'_\mu \overset{\text{def}}{=} \frac{1}{M}\left(-1 \pm \sqrt{\frac{\lambda_\mu}{\lambda_\mu + M}}\right). \tag{A.14}$$

This gives

$$\begin{aligned}\underline{\Phi} &= \left[\left(I_{M\times M} + \lambda'_\mu 1_{M\times M}\right) \otimes A_\perp\right]^{\mathrm{T}} \left[(E^{\mathrm{f}})^{-1} \otimes (E^{\mathrm{x}})^+\right] \\ &= \left[\left(I_{M\times M} + \lambda'_\mu 1_{M\times M}\right)\left(I_{M\times M} + \frac{1}{\lambda_\mu} 1_{M\times M}\right)\right] \otimes \left[A_\perp^{\mathrm{T}} A_\perp A_\perp^{\mathrm{T}}\right] \\ &= \left(I_{M\times M} + \lambda''_\mu 1_{M\times M}\right) \otimes A_\perp^{\mathrm{T}},\end{aligned} \tag{A.15}$$

where $\lambda''_\mu \overset{\text{def}}{=} \lambda'_\mu + 1/\lambda_\mu + \lambda'_\mu M/\lambda_\mu$. It is easy to verify that $\lambda''_\mu \geqslant 0$.

A.1.5 Task Clustering

In this section, we give a more detailed derivation of the multi-task clustering kernel than can be found in [Evgeniou et al., 2005, section 3.1.2]. The regularizer is the optimization

$$\min_{\{\boldsymbol{\mu}_p\}} \sum_{p=1}^{P} \left(\sum_{m=1}^{M} z_m^p \left(\boldsymbol{w}_m - \boldsymbol{\mu}_p\right)^{\mathrm{T}} E_p^{\mathrm{x}} \left(\boldsymbol{w}_m - \boldsymbol{\mu}_p\right) + \boldsymbol{\mu}_p^{\mathrm{T}} E^{\mathrm{x}} \boldsymbol{\mu}_p \right). \tag{A.16}$$

This minimization can be solved by differentiating the objective with respect to $\{\boldsymbol{\mu}_p\}$ and equating to zero. Let $z^p \overset{\text{def}}{=} \sum_{m=1}^{M} z_m^p$ denotes the number of tasks in cluster p. Then

$$\sum_{m=1}^{M} z_m^p (-2) E_p^{\mathrm{x}} (\boldsymbol{w}_m - \boldsymbol{\mu}_p) + 2 E^{\mathrm{x}} \boldsymbol{\mu}_p = \mathbf{0} \quad \Longrightarrow \quad -\sum_{m=1}^{M} z_m^p E_p^{\mathrm{x}} \boldsymbol{w}_m + z^p E_p^{\mathrm{x}} \boldsymbol{\mu}_p + E^{\mathrm{x}} \boldsymbol{\mu}_p = \mathbf{0}$$

$$\therefore \boldsymbol{\mu}_p = (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \boldsymbol{s}_p,$$

where $\boldsymbol{s}_p \overset{\text{def}}{=} \sum_{m=1}^{M} z_m^p \boldsymbol{w}_m$. Substituting the above formula for $\boldsymbol{\mu}_p$ into the expression within the parenthesis in (A.16) and expanding gives

$$\begin{aligned}
&\sum_{m=1}^{M} z_m^p \left(\boldsymbol{w}_m - (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \boldsymbol{s}_p\right)^{\mathrm{T}} E_p^{\mathrm{x}} \left(\boldsymbol{w}_m - (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \boldsymbol{s}_p\right) \\
&\qquad + \boldsymbol{s}_p^{\mathrm{T}} E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \boldsymbol{s}_p \\
&= \sum_{m=1}^{M} z_m^p \boldsymbol{w}_m^{\mathrm{T}} E_p^{\mathrm{x}} \boldsymbol{w}_m - 2 \underbrace{\textstyle\sum_{m=1}^{M} z_m^p \boldsymbol{w}_m^{\mathrm{T}}}_{\boldsymbol{s}_p^{\mathrm{T}}} E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \boldsymbol{s}_p \\
&\qquad + \underbrace{\textstyle\sum_{m=1}^{M} z_m^p}_{z^p} \boldsymbol{s}_p^{\mathrm{T}} E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \boldsymbol{s}_p \\
&\qquad + \boldsymbol{s}_p^{\mathrm{T}} E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \boldsymbol{s}_p \\
&= \sum_{m=1}^{M} z_m^p \boldsymbol{w}_m^{\mathrm{T}} E_p^{\mathrm{x}} \boldsymbol{w}_m - \boldsymbol{s}_p^{\mathrm{T}} E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} \Big[2I \underbrace{- z^p E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} - E^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1}}_{-I} \Big] E_p^{\mathrm{x}} \boldsymbol{s}_p \\
&= \sum_{m=1}^{M} z_m^p \boldsymbol{w}_m^{\mathrm{T}} E_p^{\mathrm{x}} \boldsymbol{w}_m - \boldsymbol{s}_p^{\mathrm{T}} E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \boldsymbol{s}_p
\end{aligned}$$

The regularizer is now obtained from the above expression expanding $\boldsymbol{s}_p$ and summing over p:

$$\sum_{m,m'=1}^{M} \boldsymbol{w}_m^{\mathrm{T}} \left[\sum_{p=1}^{P} z_m^p z_{m'}^p \left(\delta_{mm'} E_p^{\mathrm{x}} - E_p^{\mathrm{x}} (E^{\mathrm{x}} + z^p E_p^{\mathrm{x}})^{-1} E_p^{\mathrm{x}} \right) \right] \boldsymbol{w}_{m'}^{\mathrm{T}}.$$

Let $E_p^{\mathrm{x}} \overset{\text{def}}{=} \lambda^p E^{\mathrm{x}}$ so that there is a common metric up to uniform expansions and contractions in the parameter space. Then

$$\sum_{m=1}^{M} \sum_{m'=1}^{M} \left[\sum_{p=1}^{P} z_m^p z_{m'}^p \left(\delta_{mm'} \lambda^p - \frac{(\lambda^p)^2}{1 + z^p \lambda^p} \right) \right] \boldsymbol{w}_m^{\mathrm{T}} E^{\mathrm{x}} \boldsymbol{w}_{m'}^{\mathrm{T}}, \tag{A.17}$$

which is the separable model in the manner of section 2.3.1.

The reproducing kernel of the Hilbert space that corresponds to this regularizer is obtained in the following way. We need to invert the M-by-M matrix E^{f} consisting of entries given by the expression within the square brackets in equation A.17. If we rearrange E^{f} so that tasks from the same clusters are placed together, then E^{f} is a block diagonal matrix with P blocks. The pth block is the z^p-by-z^p matrix

$$\lambda^p I_{z^p} - \frac{(\lambda^p)^2}{1 + z^p \lambda^p} 1_{z^p \times z^p}, \tag{A.18}$$

which can be readily inverted using the Sherman-Morrison formula to give

$$\frac{1}{\lambda^p} I_{z^p} + 1_{z^p \times z^p}. \tag{A.19}$$

This is similar to the equation 27 of Evgeniou et al. [2005]. The multi-task kernel is

$$k((\boldsymbol{x}, m), (\boldsymbol{x}', m')) = \sum_{p=1}^{P} z_m^p z_{m'}^p \left(\delta_{mm'} \frac{1}{\lambda^p} + 1 \right) \boldsymbol{x}^{\mathrm{T}} (E^{\mathrm{x}})^{-1} \boldsymbol{x}'. \tag{A.20}$$

A.2 Covariances for Multi-task Gaussian Processes

In this section, we give derivations for the covariances of other multi-task models so as to relate them to Gaussian processes.

A.2.1 Derivation of Equation 2.78

To derive equation 2.78 on page 38, we start from the multi-task kernel given by equation 2.51 on page 26:

$$\begin{aligned} \mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')\right) &= \frac{1}{2\lambda} \sum_{p=1}^{P} z_m^p z_{m'}^p \left(\delta_{mm'} \frac{1}{\lambda^p} + 1 \right) \boldsymbol{x}^{\mathrm{T}} (E^{\mathrm{x}})^{-1} \boldsymbol{x}' \\ &= \delta_{mm'} \left(\sum_{p=1}^{P} \frac{z_m^p z_{m'}^p}{\lambda^p} \right) \frac{1}{2\lambda} \boldsymbol{x}^{\mathrm{T}} (E^{\mathrm{x}})^{-1} \boldsymbol{x}' + \left(\sum_{p=1}^{P} z_m^p z_{m'}^p \right) \frac{1}{2\lambda} \boldsymbol{x}^{\mathrm{T}} (E^{\mathrm{x}})^{-1} \boldsymbol{x}' \end{aligned} \tag{A.21}$$

Let the $m\text{th}$ function be in the $q\text{th}$ cluster. Then

$$\delta_{mm'} \sum_{p=1}^{P} \frac{z_m^p z_{m'}^p}{\lambda^p} = \delta_{mm'} \sum_{p=1}^{P} \frac{z_m^p z_m^p}{\lambda^p} = \delta_{mm'} \sum_{p=1}^{P} \frac{z_m^p}{\lambda^p} = \delta_{mm'} \frac{1}{\lambda^q}$$

Substituting the above back into equation (A.21) gives

$$\mathbb{C}\left(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')\right) = \delta_{mm'} \frac{1}{2\lambda\lambda^q} \boldsymbol{x}^{\mathrm{T}} (E^{\mathrm{x}})^{-1} \boldsymbol{x}' + \left(\sum_{p=1}^{P} z_m^p z_{m'}^p \right) \frac{1}{2\lambda} \boldsymbol{x}^{\mathrm{T}} (E^{\mathrm{x}})^{-1} \boldsymbol{x}' \qquad \square$$

A.2.2 Derivation of Equation 2.81

To derive equation 2.81 on page 39, we express $f_m(\boldsymbol{x})$ in terms of summation, i.e.,

$$f_m(\boldsymbol{x}) = \boldsymbol{u}_m^{\mathrm{T}} \boldsymbol{\phi}(\boldsymbol{x}) = \sum_{i=1}^{D^\Phi} u_{mi} \phi_i(\boldsymbol{x}).$$

Since u_{mi} and $\phi_i(\boldsymbol{x})$ are uncorrelated, and $\langle \phi_i(\boldsymbol{x}) \rangle = 0$, we have

$$\langle f_m(\boldsymbol{x}) \rangle = \sum_{i=1}^{D^\Phi} \langle u_{mi} \phi_i(\boldsymbol{x}) \rangle = \sum_{i=1}^{D^\Phi} \langle u_{mi} \rangle \langle \phi_i(\boldsymbol{x}) \rangle = 0.$$

Hence we have a zero mean process. The covariance is

$$\begin{aligned}
\mathbb{C}(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')) &= \langle f_m(\boldsymbol{x}) f_{m'}(\boldsymbol{x}') \rangle - \underbrace{\langle f_m(\boldsymbol{x}) \rangle \langle f_{m'}(\boldsymbol{x}') \rangle}_{0} \\
&= \left\langle \sum_{i=1}^{D^\Phi} u_{mi} \phi_i(\boldsymbol{x}) \sum_{j=1}^{D^\Phi} u_{m'j} \phi_j(\boldsymbol{x}) \right\rangle \\
&= \sum_{i,j=1}^{D^\Phi} \langle u_{mi} u_{m'j} \rangle \langle \phi_i(\boldsymbol{x}) \phi_j(\boldsymbol{x}) \rangle \\
&= \sum_{i,j=1}^{D^\Phi} \langle u_{mi} u_{m'j} \rangle \, \delta_{ij} k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}') \qquad (*) \\
&= k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}') \sum_{i=1}^{D^\Phi} \langle u_{mi} u_{m'i} \rangle , \qquad \text{(A.22)}
\end{aligned}$$

where $(*)$ is because the $\{\phi_i(\boldsymbol{x})\}$s are independent. For the last expectation above, we use

$$\begin{aligned}
& & \mathbb{C}(u_{mi}, u_{m'i}) &= \langle u_{mi} u_{m'i} \rangle - \langle u_{mi} \rangle \langle u_{m'i} \rangle \\
\Longrightarrow & & \delta_{mm'} \sigma_\xi^2 &= \langle u_{mi} u_{m'i} \rangle - (A\boldsymbol{t}_m)_i (A\boldsymbol{t}_{m'})_i \\
\Longrightarrow & & \langle u_{mi} u_{m'i} \rangle &= \delta_{mm'} \sigma_\xi^2 + (A\boldsymbol{t}_m)_i (A\boldsymbol{t}_{m'})_i \\
\Longrightarrow & & \sum_{i=1}^{D^\Phi} \langle u_{mi} u_{m'i} \rangle &= \delta_{mm'} D^\Phi \sigma_\xi^2 + \sum_{i=1}^{D^\Phi} (A\boldsymbol{t}_m)_i (A\boldsymbol{t}_{m'})_i \\
\Longrightarrow & & \sum_{i=1}^{D^\Phi} \langle u_{mi} u_{m'i} \rangle &= \delta_{mm'} D^\Phi \sigma_\xi^2 + \boldsymbol{t}_m^{\mathrm{T}} A^{\mathrm{T}} A \boldsymbol{t}_{m'}
\end{aligned}$$

Substituting back into (A.22) gives the required result

$$\mathbb{C}(f_m(\boldsymbol{x}), f_{m'}(\boldsymbol{x}')) = \delta_{mm'} D^\Phi \sigma_\xi^2 k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}') + \boldsymbol{t}_m^{\mathrm{T}} A^{\mathrm{T}} A \boldsymbol{t}_{m'} k_{\mathrm{NN}}(\boldsymbol{x}, \boldsymbol{x}').$$

Appendix B

Appendix to Chapter 3

In this appendix, we provide material supplementary to chapter 3. Much of this will be detailed proofs for the propositions and corollaries.

B.1 Proof for Proposition 3.5

In this section, we give the proof for Proposition 3.5 in the main text.

B.1.1 Proof for Proposition 3.5a

Recall from (3.3) that $\sigma_T^2(\rho)$ is given by

$$\sigma_T^2(\rho) \overset{\text{def}}{=} k_{**} - \begin{pmatrix} \boldsymbol{k}^{\text{x}}_{T*} \\ \rho \boldsymbol{k}^{\text{x}}_{S*} \end{pmatrix}^{\text{T}} \begin{pmatrix} K^{\text{x}}_{TT} + \sigma_{\text{n}}^2 I & \rho K^{\text{x}}_{TS} \\ \rho K^{\text{x}}_{ST} & K^{\text{x}}_{SS} + \sigma_{\text{n}}^2 I \end{pmatrix}^{-1} \begin{pmatrix} \boldsymbol{k}^{\text{x}}_{T*} \\ \rho \boldsymbol{k}^{\text{x}}_{S*} \end{pmatrix} \tag{B.1}$$

To perform the matrix inverse in the above equation, we use the following formula for inverting block matrices:

Theorem B.1. *Banachiewicz inversion formula (see e.g., Puntanen and Styan [2005]).*

$$\begin{aligned} \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}^{-1} &= \begin{pmatrix} A_{11}^{-1} + A_{11}^{-1} A_{12} C^{-1} A_{21} A_{11}^{-1} & -A_{11}^{-1} A_{12} C^{-1} \\ -C^{-1} A_{21} A_{11}^{-1} & C^{-1} \end{pmatrix} \\ &= \begin{pmatrix} A_{11}^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} -A_{11}^{-1} A_{12} \\ I \end{pmatrix} C^{-1} \begin{pmatrix} -A_{21} A_{11}^{-1} & I \end{pmatrix}, \end{aligned}$$

where $$C \overset{\text{def}}{=} A_{22} - A_{21} A_{11}^{-1} A_{12}.$$

The role of C in the above theorem is played by $A(\rho)$ defined as

$$A(\rho) \overset{\text{def}}{=} K^{\text{x}}_{SS} + \sigma_{\text{n}}^2 I - \rho^2 K^{\text{x}}_{ST} \left(K^{\text{x}}_{TT} + \sigma_{\text{n}}^2 I \right)^{-1} K^{\text{x}}_{TS}. \tag{B.2}$$

In addition, we let

$$\begin{aligned}\boldsymbol{v}(\rho) &\overset{\text{def}}{=} \begin{pmatrix}-\rho K^{\mathrm{x}}_{ST}(K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I)^{-1} & I\end{pmatrix}\begin{pmatrix}\boldsymbol{k}^{\mathrm{x}}_{T*}\\ \rho\boldsymbol{k}^{\mathrm{x}}_{S*}\end{pmatrix}\\ &= -\rho K^{\mathrm{x}}_{ST}(K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I)^{-1}\boldsymbol{k}^{\mathrm{x}}_{T*}+\rho\boldsymbol{k}^{\mathrm{x}}_{S*}\\ &= \rho\left(\boldsymbol{k}^{\mathrm{x}}_{S*}-K^{\mathrm{x}}_{ST}(K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I)^{-1}\boldsymbol{k}^{\mathrm{x}}_{T*}\right). \end{aligned} \tag{B.3}$$

Then

$$\sigma^2_T(\rho) = k_{**} - \begin{pmatrix}\boldsymbol{k}^{\mathrm{x}}_{T*}\\ \rho\boldsymbol{k}^{\mathrm{x}}_{S*}\end{pmatrix}^{\mathrm{T}}\begin{pmatrix}(K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I)^{-1} & 0\\ 0 & 0\end{pmatrix}\begin{pmatrix}\boldsymbol{k}^{\mathrm{x}}_{T*}\\ \rho\boldsymbol{k}^{\mathrm{x}}_{S*}\end{pmatrix} - \boldsymbol{v}(\rho)^{\mathrm{T}}[A(\rho)]^{-1}\boldsymbol{v}(\rho) \tag{B.4}$$

$$= k_{**} - (\boldsymbol{k}^{\mathrm{x}}_{T*})^{\mathrm{T}}\left(K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I\right)^{-1}\boldsymbol{k}^{\mathrm{x}}_{T*} - \boldsymbol{v}(\rho)^{\mathrm{T}}[A(\rho)]^{-1}\boldsymbol{v}(\rho) \tag{B.5}$$

We can identify $\boldsymbol{v}(\rho) = \rho\boldsymbol{v}(1)$. If we also write $\boldsymbol{v}_1$ for $\boldsymbol{v}(1)$, then

$$\sigma^2_T(\rho) = k_{**} - (\boldsymbol{k}^{\mathrm{x}}_{T*})^{\mathrm{T}}\left(K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I\right)^{-1}\boldsymbol{k}^{\mathrm{x}}_{T*} - \rho^2\boldsymbol{v}_1^{\mathrm{T}}\left[A(\rho)\right]^{-1}\boldsymbol{v}_1 \tag{B.6}$$

By substituting 0 for ρ above, the first two terms on the right of the equation can be identified with $\sigma^2_T(0)$. Thus

$$\sigma^2_T(\rho) = \sigma^2_T(0) - \rho^2\boldsymbol{v}_1^{\mathrm{T}}\left[A(\rho)\right]^{-1}\boldsymbol{v}_1. \tag{B.7}$$

Now, write A_1 for $A(1)$, and express $A(\rho)$ as

$$A(\rho) = A_1 + (1-\rho^2)K^{\mathrm{x}}_{ST}(K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I)^{-1}K^{\mathrm{x}}_{TS}. \tag{B.8}$$

Observe that $K^{\mathrm{x}}_{ST}(K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I)^{-1}K^{\mathrm{x}}_{TS}$ is positive semi-definite, since we can factorize it into the form XX^{T} for some matrix X, and that $(1-\rho^2)\geqslant 0$. Hence we can write

$$\begin{aligned} & A(\rho) \succcurlyeq A_1\\ \iff\quad & [A(\rho)]^{-1} \preccurlyeq [A_1]^{-1}\\ \iff\quad & \boldsymbol{v}_1^{\mathrm{T}}[A(\rho)]^{-1}\boldsymbol{v} \leqslant \boldsymbol{v}^{\mathrm{T}}[A_1]^{-1}\boldsymbol{v}\\ \iff\quad & \sigma^2_T(0) - \rho^2\boldsymbol{v}_1^{\mathrm{T}}[A(\rho)]^{-1}\boldsymbol{v} \geqslant \sigma^2_T(0) - \rho^2\boldsymbol{v}^{\mathrm{T}}[A_1]^{-1}\boldsymbol{v}\\ \text{i.e.,}\quad & \sigma^2_T(\rho) \geqslant \sigma^2_T(0) - \rho^2\boldsymbol{v}^{\mathrm{T}}[A_1]^{-1}\boldsymbol{v}. \end{aligned}$$

To complete the proof, we use the identity $\boldsymbol{v}_1^{\mathrm{T}}A_1^{-1}\boldsymbol{v}_1 = \sigma^2_T(0) - \sigma^2_T(1)$, which is obtained by by substituting 1 for ρ into (B.7). Further re-grouping of terms leads to the result. □

Remark The expression for $\sigma^2_T(\rho)$ given by (B.7) can also be obtained by repeated conditioning. Consider the following covariance matrix between the query f^T_*, the noisy observations $\boldsymbol{y}^S_S$ at X_S for task S, and the noisy observations $\boldsymbol{y}^T_T$ at X_T for task T,

$$\mathbb{C}\begin{pmatrix}f^T_*\\ \boldsymbol{y}^S_S\\ \boldsymbol{y}^T_T\end{pmatrix} = \begin{pmatrix}k_{**} & \rho(\boldsymbol{k}^{\mathrm{x}}_{S*})^{\mathrm{T}} & (\boldsymbol{k}^{\mathrm{x}}_{T*})^{\mathrm{T}}\\ \rho\boldsymbol{k}^{\mathrm{x}}_{S*} & K^{\mathrm{x}}_{SS}+\sigma^2_{\mathrm{n}}I & \rho K^{\mathrm{x}}_{ST}\\ \boldsymbol{k}^{\mathrm{x}}_{T*} & \rho K^{\mathrm{x}}_{TS} & K^{\mathrm{x}}_{TT}+\sigma^2_{\mathrm{n}}I\end{pmatrix}. \tag{B.9}$$

By conditioning on $\boldsymbol{y}_T^T$, we obtain

$$\mathbb{C}\left(\begin{pmatrix} f_*^T \\ \boldsymbol{y}_S^S \end{pmatrix} \middle| \boldsymbol{y}_T^T\right) = \begin{pmatrix} \sigma_T(0) & (\boldsymbol{v}(\rho))^{\mathrm{T}} \\ \boldsymbol{v}(\rho) & A(\rho) \end{pmatrix}. \tag{B.10}$$

Conditioning subsequently on $\boldsymbol{y}_S^S$ gives

$$\sigma_T^2(\rho) \overset{\text{def}}{=} \mathbb{C}(f_*^T | \boldsymbol{y}_T^T, \boldsymbol{y}_S^S) = \sigma_T^2(0) - \boldsymbol{v}(\rho)^{\mathrm{T}}[A(\rho)]^{-1}\boldsymbol{v}(\rho). \tag{B.11}$$

To complete, we simply write $\rho\boldsymbol{v}_1$ for $\boldsymbol{v}(\rho)$.

B.1.2 Proof for Proposition 3.5b

Recall that the exact posterior variance is

$$\sigma_T^2(\rho) = \sigma_T^2(0) - \rho^2\boldsymbol{v}_1^{\mathrm{T}}\left[A(\rho)\right]^{-1}\boldsymbol{v}_1, \tag{B.12}$$

i.e., equation B.7. Denote the lower bound by $\underline{\sigma}_T^2(\rho)$. Then from the proof for Proposition 3.5a, we have

$$\underline{\sigma}_T^2(\rho) = \sigma_T^2(0) - \rho^2\boldsymbol{v}_1^{\mathrm{T}}A_1^{-1}\boldsymbol{v}_1. \tag{B.13}$$

Define the gap between the exact posterior variance and its lower bound as

$$\begin{aligned} g(\rho^2) &\overset{\text{def}}{=} \sigma_T^2(\rho) - \underline{\sigma}_T^2(\rho) \\ &= -\rho^2\boldsymbol{v}_1^{\mathrm{T}}[A(\rho)]^{-1}\boldsymbol{v}_1 + \rho^2\boldsymbol{v}_1^{\mathrm{T}}A_1^{-1}\boldsymbol{v}_1. \end{aligned} \tag{B.14}$$

Ignoring the first term, which is negative, gives

$$\begin{aligned} g(\rho^2) &\leqslant \rho^2\boldsymbol{v}_1^{\mathrm{T}}A_1^{-1}\boldsymbol{v}_1 \\ &= \rho^2\left[\sigma_T^2(0) - \sigma_T^2(1)\right]. \end{aligned}$$

□

B.1.3 Proof for Proposition 3.5c

We rewrite the gap (B.14) between the exact posterior variance and its lower bound as

$$g(\rho^2) = \rho^2\boldsymbol{v}_1^{\mathrm{T}}\left[A_1^{-1} - A(\rho)^{-1}\right]\boldsymbol{v}_1. \tag{B.15}$$

Next, express $A(\rho)$ as

$$A(\rho) = A_1 + (1-\rho^2)K_{ST}^{\mathrm{x}}(K_{TT}^{\mathrm{x}} + \sigma_{\mathrm{n}}^2 I)^{-1}K_{TS}^{\mathrm{x}}, \tag{B.16}$$

so that we can use the Woodbury identity to expand its inverse in B.15, and write

$$g(\rho^2) = \boldsymbol{v}_1^{\mathrm{T}}A_1^{-1}K_{ST}^{\mathrm{x}}[B(\rho^2)]^{-1}K_{TS}^{\mathrm{x}}A_1^{-1}\boldsymbol{v}_1, \tag{B.17}$$

where

$$B(\rho^2) \overset{\text{def}}{=} D(\rho^2) + \frac{1}{\rho^2}K_{TS}^{\mathrm{x}}A_1^{-1}K_{ST}^{\mathrm{x}} \tag{B.18}$$

$$D(\rho^2) \overset{\text{def}}{=} \frac{1}{\rho^2(1-\rho^2)}(K_{TT}^{\mathrm{x}} + \sigma_{\mathrm{n}}^2 I). \tag{B.19}$$

Notice that the dependence of $g(\rho^2)$ on ρ^2 is only through $B(\rho^2)$. We differentiate $g(\rho^2)$ with respect to ρ^2:

$$\frac{\mathrm{d}g}{\mathrm{d}\rho^2} = \boldsymbol{v}_1^{\mathrm{T}} A_1^{-1} K_{ST}^{\mathrm{x}} [B(\rho^2)]^{-1} C(\rho^2) [B(\rho^2)]^{-1} K_{TS}^{\mathrm{x}} A_1^{-1} \boldsymbol{v}_1, \tag{B.20}$$

where

$$\begin{aligned} C(\rho^2) &\stackrel{\text{def}}{=} -\frac{\mathrm{d}B(\rho^2)}{\mathrm{d}\rho^2} \\ &= \frac{1-2\rho^2}{\rho^2(1-\rho^2)} D(\rho^2) + \frac{1}{\rho^4} K_{TS}^{\mathrm{x}} A_1^{-1} K_{ST}^{\mathrm{x}} \\ &= \frac{1}{\rho^2} B(\rho^2) - \frac{1}{1-\rho^2} D(\rho^2). \end{aligned} \tag{B.21}$$

Substituting the last expression for $C(\rho^2)$ back into (B.20) gives

$$\frac{\mathrm{d}g}{\mathrm{d}\rho^2} = \frac{1}{\rho^2} g - \frac{1}{1-\rho^2} h, \tag{B.22}$$

where $$h(\rho^2) \stackrel{\text{def}}{=} \boldsymbol{v}_1^{\mathrm{T}} A_1^{-1} K_{ST}^{\mathrm{x}} [B(\rho^2)]^{-1}\, D(\rho^2)\, [B(\rho^2)]^{-1} K_{TS}^{\mathrm{x}} A_1^{-1} \boldsymbol{v}_1. \tag{B.23}$$

From equation B.18, we have $D(\rho^2) \preccurlyeq B(\rho^2)$. Putting this inequality into $h(\rho^2)$ gives

$$\begin{aligned} h(\rho^2) &\leqslant \boldsymbol{v}_1^{\mathrm{T}} A_1^{-1} K_{ST}^{\mathrm{x}} [B(\rho^2)]^{-1}\, B(\rho^2)\, [B(\rho^2)]^{-1} K_{TS}^{\mathrm{x}} A_1^{-1} \boldsymbol{v}_1 \\ &= \boldsymbol{v}_1^{\mathrm{T}} A_1^{-1} K_{ST}^{\mathrm{x}} [B(\rho^2)]^{-1} K_{TS}^{\mathrm{x}} A_1^{-1} \boldsymbol{v}_1 \\ &= g(\rho^2). \end{aligned} \tag{B.24}$$

Putting the above inequality into (B.22) leads to

$$\frac{\mathrm{d}g}{\mathrm{d}\rho^2} \geqslant f(\rho^2)\, g(\rho^2), \qquad \text{where} \quad f(\rho^2) \stackrel{\text{def}}{=} \frac{1}{\rho^2} - \frac{1}{1-\rho^2}. \tag{B.25}$$

Since $\underline{\sigma}_T^2(\rho)$ is a lower bound, $g(\rho^2) \geqslant 0$ (also see the quadratic form in (B.17)). In addition, the multiplicative factor $f(\rho^2)$ is positive for $\rho^2 \in [0, 1/2[$, zero at $\rho^2 = 1/2$, and negative for $\rho^2 \in]1/2, 1]$. Thus $\mathrm{d}g/\mathrm{d}\rho^2 > 0$ for $\rho^2 \in [0, 1/2[$. Therefore g is monotonically increasing within $\rho^2 \in [0, 1/2[$, and its maximum value must be at $\hat{\rho}^2 \geqslant 1/2$. □

B.2 Proof for Proposition 3.1

To proof

$$\sigma_T^2(1) \leqslant \sigma_T^2(\rho) \leqslant \sigma_T^2(0), \tag{B.26}$$

we make use of three relations from Proposition 3.5:

$$\underline{\sigma}_T^2(\rho) \stackrel{\text{def}}{=} \rho^2 \sigma_T^2(1) + (1-\rho^2)\sigma_T^2(0) \tag{B.27a}$$

$$\underline{\sigma}_T^2(\rho) \leqslant \sigma_T^2(\rho) \tag{B.27b}$$

$$\sigma_T^2(\rho) - \underline{\sigma}_T^2(\rho) \leqslant \rho^2(\sigma_T^2(0) - \sigma_T^2(1)). \tag{B.27c}$$

Before we proceed, first note that (B.26) has not been used in the proof of any of the assertions in (B.27). Hence using (B.27) to prove (B.26) will not lead to circular arguments.

Substituting (B.27a) into (B.27c), and then re-arranging readily yields

$$\sigma_T^2(\rho) \leqslant \sigma_T^2(0). \tag{B.28}$$

Also, since $\rho^2 \in [0,1]$, we have that $\underline{\sigma}_T^2(\rho)$ is a convex combination of $\sigma_T^2(0)$ and $\sigma_T^2(1)$, and so

$$\sigma_T^2(1) \leqslant \underline{\sigma}_T^2(\rho) \leqslant \sigma_T^2(0). \tag{B.29}$$

Inserting (B.27b) and (B.28) into the above inequality gives

$$\sigma_T^2(1) \leqslant \underline{\sigma}_T^2(\rho) \leqslant \sigma_T^2(\rho) \leqslant \sigma_T^2(0), \tag{B.30}$$

from which (B.26) can be extracted. □

B.3 Proof for Proposition 3.7b

Recall that $\bar{f}_1$ (resp. $\bar{f}_0$) is defined to be the posterior mean of the single-task GP when $\rho = 1$ (resp. $\rho = 0$). We wish to obtain the error with respect to the true function $f_T^\star$ when using the linear combination predictor

$$\bar{f}_{\mathrm{lc}}(\boldsymbol{x}_*) \overset{\mathrm{def}}{=} \rho^2 \bar{f}_1(\boldsymbol{x}_*) + (1-\rho^2)\bar{f}_0(\boldsymbol{x}_*).$$

For regression, the squared error is typically used. Suppressing the argument $\boldsymbol{x}_*$ to the functions in the derivation below for conciseness, we have

$$\begin{aligned}
(f_T^\star - \bar{f}_{\mathrm{lc}})^2 &= (f_T^\star)^2 - 2\rho^2 f_T^\star \bar{f}_1 - 2(1-\rho^2) f_T^\star \bar{f}_0 + \rho^4 (\bar{f}_1)^2 + 2\rho^2(1-\rho^2)\bar{f}_1\bar{f}_0 + (1-\rho^2)^2(\bar{f}_0)^2 \\
&= \rho^2 (f_T^\star)^2 + (1-\rho^2)(f_T^\star)^2 - 2\rho^2 f_T^\star \bar{f}_1 - 2(1-\rho^2) f_T^\star \bar{f}_0 + \underline{\rho^2(\bar{f}_1)^2 + (1-\rho^2)(\bar{f}_0)^2} \\
&\qquad \underline{-\rho^2(\bar{f}_1)^2 - (1-\rho^2)(\bar{f}_0)^2} + \rho^4(\bar{f}_1)^2 + 2\rho^2(1-\rho^2)\bar{f}_1\bar{f}_0 + (1-\rho^2)^2(\bar{f}_0)^2 \\
&= \rho^2 \left[(f_T^\star)^2 - 2 f_T^\star \bar{f}_1 + (\bar{f}_1)^2\right] + (1-\rho^2)\left[f_T^\star)^2 - 2 f_T^\star \bar{f}_0 + (\bar{f}_0)^2\right] \\
&\qquad - \rho^2(1-\rho^2)\left[(\bar{f}_1)^2 - 2\bar{f}_1\bar{f}_0 + (\bar{f}_0)^2\right] \\
&= \rho^2 (f_T^\star - \bar{f}_1)^2 + (1-\rho^2)(f_T^\star - \bar{f}_0)^2 - \rho^2(1-\rho^2)(\bar{f}_1 - \bar{f}_0)^2 \\
&\leqslant \rho^2 (f_T^\star - \bar{f}_1)^2 + (1-\rho^2)(f_T^\star - \bar{f}_0)^2,
\end{aligned} \tag{B.31}$$

where the underlined terms that are introduced in the derivation cancel to zero. The equality in (B.31) holds only when $\rho^2 \in \{0,1\}$ or $\bar{f}_1(\boldsymbol{x}_*) = \bar{f}_0(\boldsymbol{x}_*)$, which happens when $\pi_S = 0$. Averaging over the possible functions $f_T^\star$ on both sides of inequality, we obtain

$$\left\langle (f_T^\star(\boldsymbol{x}_*) - \bar{f}_{\mathrm{lc}}(\boldsymbol{x}_*))^2 \right\rangle \leqslant \rho^2 \sigma_T^2(\boldsymbol{x}_*, 1) + (1-\rho^2)\sigma_T^2(\boldsymbol{x}_*, 0), \tag{B.32}$$

where the right of the inequality is $\underline{\sigma}_T^2(\boldsymbol{x}_*, \rho)$ as defined in Proposition 3.5. □

B.4 Proof for Proposition 3.9

Recall that we seek an upper bound $\bar{\sigma}_{\mathrm{n}}^2$ for $\tilde{\sigma}_{\mathrm{n}}^2$ such that $\Delta(\rho, \sigma_{\mathrm{n}}^2, \bar{\sigma}_{\mathrm{n}}^2) \leqslant 0$ for all test locations, where

$$\Delta(\rho, \sigma_{\mathrm{n}}^2, s^2) \overset{\mathrm{def}}{=} (\boldsymbol{k}_*^{\mathrm{x}})^{\mathrm{T}} \left[(\Sigma(1, \sigma_{\mathrm{n}}^2, s^2))^{-1} - (\Sigma(\rho, \sigma_{\mathrm{n}}^2, \sigma_{\mathrm{n}}^2))^{-1}\right] \boldsymbol{k}_*^{\mathrm{x}}. \tag{B.33}$$

For the derived bound to be applicable in general, it is necessary that the condition "for all test locations" be equivalent to the condition "for all $\boldsymbol{k}_*^{\mathrm{x}} \in \mathbb{R}^n$", which is easier to handle; we shall remark on this later. Thus, we start from the requirement that $\Delta(\rho, \sigma_{\mathrm{n}}^2, \bar{\sigma}_{\mathrm{n}}^2) \leqslant 0$ for all $\boldsymbol{k}_*^{\mathrm{x}} \in \mathbb{R}^n$:

$$\begin{aligned}
& \Delta(\rho, \sigma_{\mathrm{n}}^2, \bar{\sigma}_{\mathrm{n}}^2) \leqslant 0 \quad \forall \boldsymbol{k}_*^{\mathrm{x}} \in \mathbb{R}^n && \text{(B.34a)} \\
\iff \quad & (\Sigma(1, \sigma_{\mathrm{n}}^2, \bar{\sigma}_{\mathrm{n}}^2))^{-1} - (\Sigma(\rho, \sigma_{\mathrm{n}}^2, \sigma_{\mathrm{n}}^2))^{-1} \preccurlyeq 0 && \text{(B.34b)} \\
\iff \quad & (\Sigma(1, \sigma_{\mathrm{n}}^2, \bar{\sigma}_{\mathrm{n}}^2))^{-1} \preccurlyeq (\Sigma(\rho, \sigma_{\mathrm{n}}^2, \sigma_{\mathrm{n}}^2))^{-1} && \text{(B.34c)} \\
\iff \quad & \Sigma(1, \sigma_{\mathrm{n}}^2, \bar{\sigma}_{\mathrm{n}}^2) \succcurlyeq \Sigma(\rho, \sigma_{\mathrm{n}}^2, \sigma_{\mathrm{n}}^2) && \text{(B.34d)} \\
\iff \quad & \begin{pmatrix} K_{TT}^{\mathrm{x}} & K_{TS}^{\mathrm{x}} \\ K_{ST}^{\mathrm{x}} & K_{SS}^{\mathrm{x}} \end{pmatrix} + \begin{pmatrix} \sigma_{\mathrm{n}}^2 I & 0 \\ 0 & \bar{\sigma}_{\mathrm{n}}^2 I \end{pmatrix} \succcurlyeq \begin{pmatrix} K_{TT}^{\mathrm{x}} & K_{TS}^{\mathrm{x}} \\ K_{ST}^{\mathrm{x}} & \rho^{-2} K_{SS}^{\mathrm{x}} \end{pmatrix} + \begin{pmatrix} \sigma_{\mathrm{n}}^2 I & 0 \\ 0 & \rho^{-2}\sigma_{\mathrm{n}}^2 I \end{pmatrix} && \text{(B.34e)} \\
\iff \quad & \begin{pmatrix} 0 & 0 \\ 0 & \beta K_{SS}^{\mathrm{x}} \end{pmatrix} \preccurlyeq \begin{pmatrix} 0 & 0 \\ 0 & (\bar{\sigma}_{\mathrm{n}}^2 - \rho^{-2}\sigma_{\mathrm{n}}^2) I \end{pmatrix}, \quad \beta \overset{\mathrm{def}}{=} \rho^{-2} - 1 && \text{(B.34f)} \\
\iff \quad & \beta K_{SS}^{\mathrm{x}} \preccurlyeq (\bar{\sigma}_{\mathrm{n}}^2 - \rho^{-2}\sigma_{\mathrm{n}}^2) I && \text{(B.34g)} \\
\iff \quad & K_{SS}^{\mathrm{x}} \preccurlyeq \frac{\bar{\sigma}_{\mathrm{n}}^2 - \rho^{-2}\sigma_{\mathrm{n}}^2}{\beta} I && \text{(B.34h)} \\
\iff \quad & \bar{\lambda} \leqslant \frac{\bar{\sigma}_{\mathrm{n}}^2 - \rho^{-2}\sigma_{\mathrm{n}}^2}{\beta} && \text{(B.34i)} \\
\iff \quad & \bar{\sigma}_{\mathrm{n}}^2 \geqslant \beta\bar{\lambda} + \rho^{-2}\sigma_{\mathrm{n}}^2 && \text{(B.34j)} \\
& \qquad = \beta(\bar{\lambda} + \sigma_{\mathrm{n}}^2) + \sigma_{\mathrm{n}}^2. && \text{(B.34k)}
\end{aligned}$$

Therefore we have the minimum of the upper bound is

$$\bar{\bar{\sigma}}_{\mathrm{n}}^2 \overset{\mathrm{def}}{=} \beta(\bar{\lambda} + \sigma_{\mathrm{n}}^2) + \sigma_{\mathrm{n}}^2. \tag{B.35}$$

The tightness of the bound is evident from the construction of $\bar{\bar{\sigma}}_{\mathrm{n}}^2$. □

Remark For the bound to hold in general, we have claimed in the above proof that the condition "for all test locations" must be equivalent to the condition "for all $\boldsymbol{k}_*^{\mathrm{x}} \in \mathbb{R}^n$". To show this, we give a particular example that demands this equivalence. Let the input domain be the one-dimensional interval $[-1, 1]$, and let $k^{\mathrm{x}}(x, x') = xx'$. We fix x_*. If the observed locations $\{x_1, x_2, \ldots\}$ are densely located on $[-1, 1] \setminus \{x_*\}$, then the entries in

$$\boldsymbol{k}_*^{\mathrm{x}} = \begin{pmatrix} x_1 x_* \\ x_2 x_* \\ \vdots \end{pmatrix}$$

are densely located on $[-x_*, x_*] \setminus \{x_*^2\}$. Since scaling $\boldsymbol{k}_*^{\mathrm{x}}$ does not affect the inequality

$$\Delta(\rho, \sigma_{\mathrm{n}}^2, \bar{\sigma}_{\mathrm{n}}^2) \leqslant 0,$$

we have, equivalently, $\boldsymbol{k}_*^{\mathrm{x}} \in \mathbb{R}^n$.

B.5 FWO$_\varrho$ Upper Bounds

B.5.1 Proof for Lemma 3.15

$$
\begin{aligned}
&\left\langle (f^\star(\boldsymbol{x}) - g(\boldsymbol{x}))^2 \right\rangle_{f^\star,\boldsymbol{y},\boldsymbol{x}} \\
&= \left\langle \left(f^\star(\boldsymbol{x}) - \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) L\boldsymbol{y}\right)^2 \right\rangle_{f^\star,\boldsymbol{y},\boldsymbol{x}} \\
&= \left\langle f^\star(\boldsymbol{x}) f^\star(\boldsymbol{x}) \right\rangle_{f^\star,\boldsymbol{x}} + \left\langle \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) L\boldsymbol{y}\boldsymbol{y}^{\mathrm{T}} L^{\mathrm{T}} \boldsymbol{\phi}(\boldsymbol{x}) \right\rangle_{\boldsymbol{y},\boldsymbol{x}} - 2\left\langle \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) L \left\langle \boldsymbol{y} f^\star(\boldsymbol{x})\right\rangle_{f^\star,\boldsymbol{y}} \right\rangle_{\boldsymbol{x}} \\
&= \left\langle k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x}) \right\rangle_{\boldsymbol{x}} + \operatorname{tr}\left(L \left\langle \boldsymbol{y}\boldsymbol{y}^{\mathrm{T}} \right\rangle_{\boldsymbol{y}} L^{\mathrm{T}} \left\langle \boldsymbol{\phi}(\boldsymbol{x})\boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) \right\rangle_{\boldsymbol{x}} \right) - 2\operatorname{tr}\left(L \left\langle \left\langle \boldsymbol{y} f^\star(\boldsymbol{x}) \right\rangle_{f^\star,\boldsymbol{y}} \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) \right\rangle_{\boldsymbol{x}} \right) \\
&= \sum_{i=1}^{\infty} \kappa_i + \operatorname{tr}\left(L \left\langle \boldsymbol{y}\boldsymbol{y}^{\mathrm{T}} \right\rangle_{\boldsymbol{y}} L^{\mathrm{T}} \right) - 2\operatorname{tr}\left(L \left\langle \left\langle \boldsymbol{y} f^\star(\boldsymbol{x}) \right\rangle_{f^\star,\boldsymbol{y}} \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) \right\rangle_{\boldsymbol{x}} \right)
\end{aligned}
$$

Note that we have used $\left\langle \boldsymbol{\phi}(\boldsymbol{x})\boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}) \right\rangle_{\boldsymbol{x}} = I$ for the last expression. □

B.5.2 Proof for Proposition 3.16

To prove Proposition 3.16, we start from (3.32), which we recall below:

$$
\epsilon(g \in \mathcal{H}_\varrho, X, \mathcal{M}_{\mathrm{mt}}) = \sum_{i=1}^{\infty} \kappa_i + \sum_{i=1}^{\infty} d_i^2 \left(\Phi^{\mathrm{T}} G_\varrho \Phi\right)_{ii} - 2\sum_{i=1}^{\infty} d_i \left(\Phi^{\mathrm{T}} I(\rho\varrho)\Phi\Lambda_\kappa\right)_{ii},
$$

where
$$
G_\varrho \stackrel{\text{def}}{=} \begin{pmatrix} K^{\mathrm{x}}_{TT} & \rho\varrho K^{\mathrm{x}}_{TS} \\ \rho\varrho K^{\mathrm{x}}_{ST} & \varrho^2 K^{\mathrm{x}}_{SS} \end{pmatrix} + \sigma_{\mathrm{n}}^2 I(\varrho^2).
$$

To obtain a learning curve, the expectation of $\epsilon(g \in \mathcal{H}_\varrho, X)$ over all data sets X is taken. This requires the expectations $\left\langle \left(\Phi^{\mathrm{T}} G_\varrho \Phi\right)_{ii} \right\rangle_X$ and $\left\langle \left(\Phi^{\mathrm{T}} I(\rho\varrho)\Phi\Lambda_\kappa\right)_{ii} \right\rangle_X$. In order to present the expressions for these expectations, some notations are required. Recall that the cardinality of these sets are $|X| = n$, $|X_S| = n_S$ and $|X_T| = n_T$, and that $\pi_S \stackrel{\text{def}}{=} n_S/n$. We partition the index set $\mathcal{I} \stackrel{\text{def}}{=} \{1 \dots n\}$ into $\mathcal{I}_T \stackrel{\text{def}}{=} \{1 \dots n_T\}$, and $\mathcal{I}_S \stackrel{\text{def}}{=} \{(n_T + 1) \dots n\}$. We enumerate and order the elements of X so that $X = \{\boldsymbol{x}_i\}_{i=1}^n$, $X_T = \{\boldsymbol{x}_i \mid i \in \mathcal{I}_T\}$, and $X_S = \{\boldsymbol{x}_i \mid i \in \mathcal{I}_S\}$. We shall represent by $\langle \cdots \rangle_X$ the expectation over the set X, and write $p(X)\mathrm{d}X \stackrel{\text{def}}{=} \prod_{i=1}^n p(\boldsymbol{x}_i)\mathrm{d}\boldsymbol{x}_i$, where the distributions over the $\boldsymbol{x}_i$s are identical; expressions $\langle \cdots \rangle_{X_S}$ and $\langle \cdots \rangle_{X_T}$, and $p(X_S)\mathrm{d}X_S$ and $p(X_T)\mathrm{d}X_T$ have similar meanings.

Recall the definition of eigenvalues and eigenfunctions using the integral equation and the orthogonality of eigenfunctions:

$$
\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x}')\phi_i(\boldsymbol{x}')p(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}' = \kappa_i\phi_i(\boldsymbol{x}) \qquad \int \phi_i(\boldsymbol{x})\phi_i(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = 1. \tag{B.36}
$$

Using the these two equalities, we can show that

$$
\begin{aligned}
\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x}')\phi_i(\boldsymbol{x})\phi_i(\boldsymbol{x}')p(\boldsymbol{x})p(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{x}' &= \int \left(\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x}')\phi_i(\boldsymbol{x}')p(\boldsymbol{x}')\mathrm{d}\boldsymbol{x}' \right) \phi_i(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\
&= \int (\kappa_i\phi_i(\boldsymbol{x}))\, \phi_i(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\
&= \kappa_i \int \phi_i(\boldsymbol{x})\phi_i(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\
&= \kappa_i
\end{aligned} \tag{B.37}
$$

The next two equalities will be used in the main part of the proof:

$$\begin{aligned}
\left\langle \sum_{p,q\in\mathcal{I}} \delta_{pq}\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q) \right\rangle_X &= \int \sum_{p=1}^{n} \phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_p)p(X)\mathrm{d}X \\
&= \sum_{p=1}^{n} \int \phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_p)p(\boldsymbol{x}_p)\mathrm{d}\boldsymbol{x}_p \\
&= \sum_{p=1}^{n} 1 \\
&= n \qquad \text{(B.38)}
\end{aligned}$$

$$\begin{aligned}
\left\langle \sum_{p,q\in\mathcal{I}} k^{\mathrm{x}}(\boldsymbol{x}_p,\boldsymbol{x}_q)\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q) \right\rangle_X &= \int \sum_{p,q\in\mathcal{I}} k^{\mathrm{x}}(\boldsymbol{x}_p,\boldsymbol{x}_q)\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q)p(X)\mathrm{d}X \\
&= \sum_{\substack{p,q\in\mathcal{I}\\ p\neq q}} \int k^{\mathrm{x}}(\boldsymbol{x}_p,\boldsymbol{x}_q)\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q)p(\boldsymbol{x}_p)p(\boldsymbol{x}_q)\mathrm{d}\boldsymbol{x}_p\mathrm{d}\boldsymbol{x}_q \\
&\qquad + \sum_{p=1}^{n} \int k^{\mathrm{x}}(\boldsymbol{x}_p,\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_p)p(\boldsymbol{x}_p)\mathrm{d}\boldsymbol{x}_p \\
&= \sum_{\substack{p,q\in\mathcal{I}\\ p\neq q}} \kappa_i + \sum_{p=1}^{n} \int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})\left[\phi_i(\boldsymbol{x})\right]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \qquad (*) \\
&= n(n-1)\kappa_i + n\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})\left[\phi_i(\boldsymbol{x})\right]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}, \qquad \text{(B.39)}
\end{aligned}$$

where equation B.37 is used in getting to $(*)$. By similar arguments we can show equivalent results when the summations and expectations are taken only over data points in X_S and X_T.

We now turn to the main part of the proof, giving expressions for the expectations $\left\langle\left(\Phi^{\mathrm{T}}G_{\varrho}\Phi\right)_{ii}\right\rangle_X$ and $\left\langle\left(\Phi^{\mathrm{T}}I(\rho\varrho)\Phi\Lambda_{\kappa}\right)_{ii}\right\rangle_X$. Define

$$\alpha(x) \overset{\mathrm{def}}{=} (1-\pi_S) + \pi_S x. \qquad \text{(B.40)}$$

The following expressions involve $\alpha(\varrho^2)$ and $\alpha(\rho\varrho)$.

$$\begin{aligned}
\left\langle\left(\Phi^{\mathrm{T}}I(\rho\varrho)\Phi\Lambda_{\kappa}\right)_{ii}\right\rangle_X &= \kappa_i\left\langle \sum_{p,q\in\mathcal{I}_T} \delta_{pq}\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q) \right\rangle_{X_T} + \rho\varrho\kappa_i\left\langle \sum_{p,q\in\mathcal{I}_S} \delta_{pq}\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q) \right\rangle_{X_T} \\
&= \kappa_i(n_T + \rho\varrho n_S) \\
&= \kappa_i n\alpha(\rho\varrho)
\end{aligned}$$

$$\begin{aligned}
\left\langle\left(\Phi^{\mathrm{T}}G_{\varrho}\Phi\right)_{ii}\right\rangle_X = &\left\langle \sum_{p,q\in\mathcal{I}_T} \left(k^{\mathrm{x}}(\boldsymbol{x}_p,\boldsymbol{x}_q) + \delta_{pq}\sigma_{\mathrm{n}}^2\right)\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q) \right\rangle_{X_T} \\
&+ \varrho^2\left\langle \sum_{p,q\in\mathcal{I}_S} \left(k^{\mathrm{x}}(\boldsymbol{x}_p,\boldsymbol{x}_q) + \delta_{pq}\sigma_{\mathrm{n}}^2\right)\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q) \right\rangle_{X_S} \\
&+ 2\rho\varrho\left\langle \sum_{p\in\mathcal{I}_T,q\in\mathcal{I}_S} k^{\mathrm{x}}(\boldsymbol{x}_p,\boldsymbol{x}_q)\phi_i(\boldsymbol{x}_p)\phi_i(\boldsymbol{x}_q) \right\rangle_X \\
= &\; n_T(n_T-1)\kappa_i + n_T\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + n_T\sigma_{\mathrm{n}}^2 \\
&+ \varrho^2\left(n_S(n_S-1)\kappa_i + n_S\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + n_S\sigma_{\mathrm{n}}^2\right) + 2\rho\varrho n_S n_T\kappa_i \\
= &\; n\alpha(\varrho^2)c_i(\varrho)
\end{aligned}$$

where,

$$c_i(\varrho) \overset{\text{def}}{=} \left\{ \frac{1}{\alpha(\varrho^2)} \left[(1-\pi_S)^2 + \varrho^2 \pi_S^2 + 2\rho\varrho\pi_S(1-\pi_S) \right] n - 1 \right\} \kappa_i + \int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} + \sigma_{\mathrm{n}}^2.$$

Thus we have

$$\langle \epsilon(g \in \mathcal{H}_\varrho, X, \mathcal{M}_{\mathrm{mt}}) \rangle_X = \sum_{i=1}^{\infty} \kappa_i + n\alpha(\varrho^2) \sum_{i=1}^{\infty} d_i^2 c_i(\varrho) - 2n\alpha(\rho\varrho) \sum_{i=1}^{\infty} d_i \kappa_i.$$

The minimum learning curve is obtained by minimizing the above quantity with respect to d_i, which on solving gives

$$\min_{\{d_i\}} \langle \epsilon(g \in \mathcal{H}_3, X, \mathcal{M}_{\mathrm{mt}}) \rangle_X = \sum_{i=1}^{\infty} \kappa_i - n\frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)} \sum_{i=1}^{\infty} \frac{\kappa_i^2}{c_i(\varrho)}. \qquad \square$$

Remark. Although we do not have the proof by Ferrari Trecate et al. [1999] for their upper bound on the learning curve for single-task GP with isotropic noise, it is conceivable that some variation of the above proof has been used by them.

B.5.3 Derivation of Equation 3.30

Define

$$K(\rho) \overset{\text{def}}{=} \begin{pmatrix} K_{TT}^{\mathrm{x}} & \rho K_{TS}^{\mathrm{x}} \\ \rho K_{ST}^{\mathrm{x}} & K_{SS}^{\mathrm{x}} \end{pmatrix} \qquad I(s) \overset{\text{def}}{=} \begin{pmatrix} I_{n_T \times n_T} & 0 \\ 0 & sI_{n_S \times n_S} \end{pmatrix}.$$

Then, under the generative model $\mathcal{M}_{\mathrm{mt}}$ given by equation 3.29 on page 66, we have

$$\left\langle \boldsymbol{y}\boldsymbol{y}^{\mathrm{T}} \right\rangle_{\boldsymbol{y}} = K(\rho) + \sigma_{\mathrm{n}}^2 I$$

$$\left\langle \langle \boldsymbol{y} f^\star(\boldsymbol{x}_*) \rangle_{f^\star,\boldsymbol{y}} \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}_*) \right\rangle_{\boldsymbol{x}_*} = \left\langle \begin{pmatrix} \boldsymbol{k}_{T*}^{\mathrm{x}} \\ \rho \boldsymbol{k}_{S*}^{\mathrm{x}} \end{pmatrix} \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}_*) \right\rangle_{\boldsymbol{x}_*} = I(\rho) \left\langle \begin{pmatrix} \boldsymbol{k}_{T*}^{\mathrm{x}} \\ \boldsymbol{k}_{S*}^{\mathrm{x}} \end{pmatrix} \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}_*) \right\rangle_{\boldsymbol{x}_*} = I(\rho)\Phi\Lambda_\kappa.$$

Substituting the above into Lemma 3.15 gives the required expression.

B.5.4 Derivation of Equation 3.35

Define

$$K^{\mathrm{x}} \overset{\text{def}}{=} \begin{pmatrix} K_{TT}^{\mathrm{x}} & K_{TS}^{\mathrm{x}} \\ K_{ST}^{\mathrm{x}} & K_{SS}^{\mathrm{x}} \end{pmatrix} = K(1) \qquad I(s) \overset{\text{def}}{=} \begin{pmatrix} I_{n_T \times n_T} & 0 \\ 0 & sI_{n_S \times n_S} \end{pmatrix}.$$

Then, under the generative model $\mathcal{M}_{\mathrm{co}}$ given by equation 3.34 on page 68, we have

$$\left\langle \boldsymbol{y}\boldsymbol{y}^{\mathrm{T}} \right\rangle_{\boldsymbol{y}} = K^{\mathrm{x}} + \begin{pmatrix} \sigma_{\mathrm{n}}^2 I_{n_T \times n_T} & 0 \\ 0 & \beta K_{SS}^{\mathrm{x}} + \rho^{-2}\sigma_{\mathrm{n}}^2 I_{n_S \times n_S} \end{pmatrix} = \begin{pmatrix} K_{TT}^{\mathrm{x}} & K_{TS}^{\mathrm{x}} \\ K_{ST}^{\mathrm{x}} & \rho^{-2} K_{SS}^{\mathrm{x}} \end{pmatrix} + \sigma_{\mathrm{n}}^2 I(\rho^{-2})$$

$$\left\langle \langle \boldsymbol{y} f^\star(\boldsymbol{x}_*) \rangle_{f^\star,\boldsymbol{y}} \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}_*) \right\rangle_{\boldsymbol{x}_*} = \left\langle \begin{pmatrix} \boldsymbol{k}_{T*}^{\mathrm{x}} \\ \boldsymbol{k}_{S*}^{\mathrm{x}} \end{pmatrix} \boldsymbol{\phi}^{\mathrm{T}}(\boldsymbol{x}_*) \right\rangle_{\boldsymbol{x}_*} = \Phi\Lambda_\kappa.$$

Substituting the above into Lemma 3.15 leads to

$$\epsilon(g \in \mathcal{H}_\star, X, \mathcal{M}_{\mathrm{co}}) = \sum_{i=1}^{\infty} \kappa_i + \operatorname{tr}\left(L\left(\begin{pmatrix} K^{\mathrm{x}}_{TT} & K^{\mathrm{x}}_{TS} \\ K^{\mathrm{x}}_{ST} & \rho^{-2}K^{\mathrm{x}}_{SS}\end{pmatrix} + \sigma_{\mathrm{n}}^2 I(\rho^{-2})\right)L^{\mathrm{T}}\right) - 2\operatorname{tr}\left(L\Phi\Lambda_\kappa\right).$$

To obtain equation 3.35, we restrict g to be from $\mathcal{H}_1$ and let $L = D\Phi^{\mathrm{T}}$.

B.5.5 Proof for Proposition 3.18

Recall from Proposition 3.16 that

$$\begin{aligned} \alpha(x) &\stackrel{\text{def}}{=} (1-\pi_S) + \pi_S x \\ c_i(\varrho) &\stackrel{\text{def}}{=} \left\{\frac{1}{\alpha(\varrho^2)}\left[(1-\pi_S)^2 + \varrho^2\pi_S^2 + 2\rho\varrho\pi_S(1-\pi_S)\right]n - 1\right\}\kappa_i \\ &\qquad + \int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + \sigma_{\mathrm{n}}^2. \end{aligned}$$

By comparing the terms for the i^{th} eigenvalue, a sufficient condition for Proposition 3.18 is

$$\frac{\alpha(1/\rho^2)}{[\alpha(1)]^2}c_i(1/\rho) \geqslant \frac{\alpha(\rho^2)}{[\alpha(\rho^2)]^2}c_i(\rho).$$

By grouping the factors involving n and the factors not involving n, the following two conditions are sufficient *together*:

$$(1-\pi_S)^2 + \rho^{-2}\pi_S^2 + 2\pi_S(1-\pi_S) \geqslant [\alpha(\rho^2)]^{-2}\left[(1-\pi_S)^2 + \rho^2\pi_S^2 + 2\rho^2\pi_S(1-\pi_S)\right] \quad \text{(Sufficient condition } A)$$

$$\alpha(\rho^{-2})\left(\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} - \kappa_i + \sigma_{\mathrm{n}}^2\right) \geqslant \frac{1}{\alpha(\rho^2)}\left(\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} - \kappa_i + \sigma_{\mathrm{n}}^2\right) \quad \text{(Sufficient condition } B)$$

Sufficient condition B is fulfilled by the following two sub-conditions

$$\alpha(\rho^{-2}) \geqslant 1/\alpha(\rho^2) \qquad \text{(Sufficient condition } B1)$$

$$\int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} - \kappa_i \geqslant 0. \qquad \text{(Sufficient condition } B2)$$

Therefore, all that is required is to prove for conditions A, $B1$ and $B2$. Condition A can be shown to hold via the sequence

$$\begin{aligned} & 1 - \pi_S^2 + \rho^{-2}\pi_S^2 \geqslant [\alpha(\rho^2)]^{-2}\left([\alpha(\rho^2)]^2 + \rho^2\pi_S^2 - \rho^4\pi_S^2\right) \\ \Longleftrightarrow\quad & \rho^{-2}\pi_S^2(1-\rho^2) \geqslant [\alpha(\rho^2)]^{-2}\rho^2\pi_S^2\left(1-\rho^2\right) \\ \Longleftrightarrow\quad & [\alpha(\rho^2)]^2 - \rho^4 \geqslant 0 \\ \Longleftrightarrow\quad & [\alpha(\rho^2)+\rho^2][\alpha(\rho^2)-\rho^2] \geqslant 0 \\ \Longleftrightarrow\quad & \alpha(\rho^2) - \rho^2 \geqslant 0 \\ \Longleftrightarrow\quad & (1-\rho^2)(1-\pi_S) \geqslant 0, \end{aligned}$$

which clearly holds. Similarly, the following sequence shows that $B1$ is always true.

$$\begin{aligned} & \alpha(\rho^{-2})\alpha(\rho^2) \geqslant 1 \\ \iff \quad & (1-\pi_S)^2 + \pi_S^2 + (1/\rho^2+\rho^2)\pi_S(1-\pi_S) \geqslant 1 \\ \iff \quad & (1-\rho^2)^2\pi_S(1-\pi_S)/\rho^2 \geqslant 0. \end{aligned}$$

To prove that condition $B2$ holds, first note that

$$\begin{aligned} \delta_{ij} = (\delta_{ij})^2 = \left(\int \phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}\right)^2 &\leqslant \int 1^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \int [\phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\ &= \int [\phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}, \end{aligned} \tag{B.41}$$

where the inequality is due to the Cauchy-Bunyakovsky-Schwarz inequality. Now, making use of the Mercer's theorem [Mercer, 1909] to expand $k(\boldsymbol{x},\boldsymbol{x})$,

$$\begin{aligned} \int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} &= \int \sum_j^\infty \kappa_j [\phi_j(\boldsymbol{x})]^2[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\ &= \sum_j^\infty \kappa_j \int [\phi_i(\boldsymbol{x})\phi_j(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\ &\geqslant \sum_j^\infty \kappa_j \delta_{ij} = \kappa_i, \end{aligned}$$

where the inequality uses (B.41) for each j. Thus condition $B2$ holds. □

B.5.6 Comparing the FWO$_\rho$ Bound to the Trivial Single-task FWO$_0$ Bound

We compare the FWO$_\rho$ upper bound, which is the FWO$_\varrho$ bound given by Proposition 3.16 for $\varrho = \rho$, with the FWO upper bound on the single-task trivial upper bound $\epsilon_T^{\mathrm{avg}}(0,\sigma_{\mathrm{n}}^2,\pi_S,n)$. The latter upper bound, which is also FWO$_0$, is given by

$$\bar{\epsilon}_{\mathrm{FWO}}^{\mathrm{avg}}(\sigma_{\mathrm{n}}^2, n(1-\pi_S)) = \sum_{i=1}^\infty \kappa_i - n(1-\pi_S)\sum_{i=1}^\infty \frac{\kappa_i^2}{(n(1-\pi_S)-1)\,\kappa_i + \iota_i + \sigma_{\mathrm{n}}^2},$$

where $\iota_i \overset{\mathrm{def}}{=} \int k^{\mathrm{x}}(\boldsymbol{x},\boldsymbol{x})[\phi_i(\boldsymbol{x})]^2 p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$. A sufficient condition for the FWO$_\rho$ bound to be the tighter of the two bounds can be obtained by comparing the terms for the i^{th} eigenvalue. This condition is

$$\forall i \qquad (1-\pi_S)c_i(\rho) < \alpha(\rho^2)\left[(n(1-\pi_S)-1)\,\kappa_i + \iota_i + \sigma_{\mathrm{n}}^2\right], \tag{B.42}$$

where $c_i(\rho)$ and $\alpha(\cdot)$ is as defined in Proposition 3.16. The inequality can be simplified to give

$$\forall i \qquad n < \left(\frac{\iota_i + \sigma_{\mathrm{n}}^2}{\kappa_i} - 1\right)\frac{1-\pi_S+\pi_S\rho^2}{(1-\rho^2)(1-\pi_S)\pi_S}. \tag{B.43}$$

For a stationary k^{x}, ι_i is independent of i and is given by $\sum_{i=1}^\infty \kappa_i$. In this case, the sufficient condition can be simplified to

$$n < \left(\frac{\sum_{i=1}^\infty \kappa_i + \sigma_{\mathrm{n}}^2}{\kappa_1} - 1\right)\frac{1-\pi_S+\pi_S\rho^2}{(1-\rho^2)(1-\pi_S)\pi_S} = \frac{\sum_{i=2}^\infty \kappa_i + \sigma_{\mathrm{n}}^2}{\kappa_1}\,\frac{1-\pi_S+\pi_S\rho^2}{(1-\rho^2)(1-\pi_S)\pi_S}, \tag{B.44}$$

where κ_1 is the maximum process eigenvalue of k^{x} under $p(\boldsymbol{x})$.

B.6 Eigen-analysis

In this section, we give the eigenvalues and eigenfunctions for the symmetric two-tasks multi-task learning, and for the OU process on a one-dimensional unit interval with uniform density. Asymptotics of the eigenvalues are treated in section B.8.1 on page 163.

B.6.1 Covariance Function of a Symmetric Two-task GP

For the multi-task Gaussian process model, the eigenvalues and eigenfunctions of its covariance function

$$K^{\mathrm{f}}_{mm'} k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$$

can be obtained through the eigen-analysis of the Kronecker product (see e.g., Brewer [1978]). Let the eigen-equations for K^{f} and k^{x} be

$$\sum_{m'} K^{\mathrm{f}}_{mm'} u_{m'} p_{m'} = \mu u_m \qquad \int k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') \phi(\boldsymbol{x}') p(\boldsymbol{x}') \mathrm{d}\boldsymbol{x}' = \kappa \phi(\boldsymbol{x}), \tag{B.45}$$

where we have introduced the eigenvalue μ and eigenvector (u_1, u_2) of K^{f}, and p_m is the probability of being in task m. To obtain the eigenvalues and eigenfunctions of $K^{\mathrm{f}}_{mm'} k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$, we multiply the two equations above to give

$$\begin{aligned}
&\sum_{m'} \int K^{\mathrm{f}}_{mm'} k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') u_{m'} \phi(\boldsymbol{x}') p_{m'} p(\boldsymbol{x}') \mathrm{d}\boldsymbol{x}' \\
&= \left(\sum_{m'} K^{\mathrm{f}}_{mm'} u_{m'} p_{m'} \right) \left(\int k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') \phi(\boldsymbol{x}') p(\boldsymbol{x}') \mathrm{d}\boldsymbol{x}' \right) \\
&= (\mu u_m)\,(\kappa \phi(\boldsymbol{x})) \\
&= (\mu\kappa)\,(u_m \phi(\boldsymbol{x}))\,.
\end{aligned} \tag{B.46}$$

Therefore $\mu\kappa$ is an eigenvalue $K^{\mathrm{f}}_{mm'} k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}')$, and the corresponding eigenfunction is $u_m \phi(\boldsymbol{x})$.

We now proceed to identify the eigenvalues μ_i of K^{f} for the setup given in section 3.2.1, where there are two tasks S and T, and where K^{f} is constrained to be a correlation matrix. That is,

$$K^{\mathrm{f}} \stackrel{\mathrm{def}}{=} \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}, \qquad \rho \in [-1, 1]. \tag{B.47}$$

In addition, the probability of the two tasks are given by $p_S = \pi_S$ and $p_T = 1 - \pi_S$. For an eigenvalue μ and an eigenvector $\boldsymbol{u} \stackrel{\mathrm{def}}{=} (u_S, u_T)^{\mathrm{T}}$, the left equation of (B.45) means that μ must simultaneously satisfy

$$(1 - \pi_S) u_T + \pi_S \rho u_S = \mu u_T \qquad (1 - \pi_S) \rho u_T + \pi_S u_S = \mu u_S. \tag{B.48}$$

Both u_T and u_S can be eliminated by the method of substitution, resulting in the quadratic equation $\mu^2 - \mu + \pi_S(1 - \pi_S)(1 - \rho^2) = 0$ in μ. The solutions to this quadratic equation are given by

$$\mu = 1/2 \pm \sqrt{1/4 - \omega} \qquad \text{where} \quad \omega \stackrel{\mathrm{def}}{=} \pi_S(1 - \pi_S)(1 - \rho^2). \tag{B.49}$$

Observe that the eigenvalues are real, since $\pi_S(1 - \pi_S) \leqslant 1/4$ and $(1 - \rho^2) \leqslant 1$ so that $\omega \geqslant 0$.

For completeness, we shall also give the eigenvectors. First, we use the constraint that the eigenvectors has unit length,

$$u_S^2\pi_S + u_T^2(1-\pi_S) = 1,$$

into (B.48), and then simplify to give

$$u_S^2 = \frac{2(1-\pi_S)\rho^2}{1-4\omega \pm (1-2\pi_S)\sqrt{1-4\omega}} \qquad u_T^2 = \frac{2\pi_S\rho^2}{1-4\omega \mp (1-2\pi_S)\sqrt{1-4\omega}}. \tag{B.50}$$

Notice the symmetry between u_S^2 and u_T^2 via the substitution $\pi_S \rightsquigarrow (1-\pi_S)$.

We now distinguish the two pairs of eigenvalue and eigenvector, that is, μ_i and $\boldsymbol{u}_i \overset{\text{def}}{=} (u_{iS}, u_{iT})^{\mathrm{T}}$, $i \in \{1,2\}$. Let $(\cdot)_+$ and $(\cdot)_-$ denote selecting either the $+$ or the $-$ case for an expression having a $\pm$ or $\mp$ operator. Then, by using the orthogonality between the eigenvectors, i.e.,

$$u_{1S}\, u_{2S}\, \pi_S + u_{1T}\, u_{2T}\, (1-\pi_S) = 0,$$

we obtain

$$u_{1S} = s_{1S}\sqrt{(u_S^2)_+} \qquad u_{1T} = s_{1T}\sqrt{(u_T^2)_-} \qquad \mu_1 = (\mu)_+ \tag{B.51}$$

$$u_{2S} = s_{2S}\sqrt{(u_S^2)_-} \qquad u_{2T} = s_{2T}\sqrt{(u_T^2)_+} \qquad \mu_2 = (\mu)_- \tag{B.52}$$

with $s_{\bullet\bullet} \in \{-1,1\}$ satisfying $s_{1S}\, s_{2S} = -s_{1T}\, s_{2T}$. Note that μ_1 is the major eigenvalue, and μ_2 is the minor. Substituting back into (B.48) gives $s_{1S}\, s_{1T} = \operatorname{sgn}(\rho)$ and $s_{2S}\, s_{2T} = -\operatorname{sgn}(\rho)$, where sgn is the signum function.

B.6.2 Stationary Ornstein-Uhlenbeck Process on the Uniform Unit Interval

In this section, we give the eigenvalues and eigenfunctions of covariance function of the stationary Ornstein-Uhlenbeck (OU) process on a one-dimensional unit interval with uniform density.

Proposition B.2. *For covariance function* $k(x,x') = \exp(-|x-x'|/l)$ *defined on* $[0,1]^2$ *with uniform density, the eigenvalues and eigenfunctions are given by*

$$\kappa = \frac{2}{lt^2 + l^{-1}} \qquad \phi(x) = \sqrt{\frac{2}{l^2t^2+2l+1}}\,(lt\cos tx + \sin tx)\,,$$

where $\tan t = 2lt/(l^2t^2-1)$.

Proof. Our derivation follows the outline given by Hawkins [1989]. The eigen-equation to be solved is

$$\int_0^1 \exp -\frac{|x-y|}{l}\phi(y)\mathrm{d}y = \kappa\phi(x), \qquad x \in [0,1]. \tag{B.53}$$

Breaking the range of integration into $y \leqslant x$ and $y > x$ gives

$$e^{-x/l}\int_0^x e^{y/l}\phi(y)\mathrm{d}y + e^{x/l}\int_x^1 e^{-y/l}\phi(y)\mathrm{d}y = \kappa\phi(x). \tag{B.54}$$

Differentiate the above equation wrt x and then simplify to give

$$-e^{-x/l}\int_0^x e^{y/l}\phi(y)\mathrm{d}y + e^{x/l}\int_x^1 e^{-y/l}\phi(y)\mathrm{d}y = l\kappa\phi'(x). \tag{B.55}$$

Adding and subtracting the above two equations give, respectively,

$$2e^{x/l}\int_x^1 e^{-y/l}\phi(y)\mathrm{d}y = \kappa\left(\phi(x) + l\phi'(x)\right) \tag{B.56}$$

$$2e^{-x/l}\int_0^x e^{y/l}\phi(y)\mathrm{d}y = \kappa\left(\phi(x) - l\phi'(x)\right) \tag{B.57}$$

Dividing both sides of (B.56) by $e^{x/l}$, differentiating with respect to x, and then simplifying leads to the following second order homogeneous linear ordinary differential equation with constant coefficients

$$\phi''(x) - s\phi(x) = 0 \qquad \text{where } s \overset{\text{def}}{=} l^{-2} - 2(\kappa l)^{-1}. \tag{B.58}$$

The boundary conditions of this equation are obtained by using $x = 0$ in (B.57) and $x = 1$ in (B.56):

$$l\phi'(0) = \phi(0) \qquad l\phi'(1) = -\phi(1). \tag{B.59}$$

The characteristic equation associated with (B.58) is $r^2 - s = 0$, with solutions $r = \pm\sqrt{s}$. Consider the following three cases:

Case $s = 0$ *when* $\kappa = 2l$. In this case, $r = 0$, and the general solution for ϕ is $\phi(x) = c_1 + c_2 x$. The boundary conditions requires $c_1 = c_2 = 0$, so that ϕ is a zero function. Since a zero function is not an eigenfunction by definition, this case leads to no solution.

Case $s > 0$ *when* $\kappa > 2l$. In this case, the general solution for ϕ is $\phi(x) = c_1 e^{\sqrt{s}x} + c_2 e^{-\sqrt{s}x}$. The boundary conditions requires

$$e^{\sqrt{s}} = \pm\left(1 - \frac{2}{l\sqrt{s}+1}\right).$$

Differentiating the above with respect to $\sqrt{s}$, and then equating with the above to eliminate $e^{\sqrt{s}}$ leads to the following quadratic equation after simplification

$$(l\sqrt{s}+1)^2 - 2(l\sqrt{s}+1) - 2l = 0.$$

The solution to this equation is $s = l^{-2} + 2l^{-1}$. Equating this solution to the definition of s in (B.58) gives $\kappa = -1$, which contradicts the initial assumption $\kappa > 2l$ for this case. Thus there is no solution for this case.

Case $s < 0$ *when* $\kappa < 2l$. In this case, $r = \pm ti$, where $t^2 \overset{\text{def}}{=} -s > 0$ and $i^2 = \sqrt{-1}$. Here, the general solution for ϕ is

$$\phi(x) = c_1 \cos tx + c_2 \sin tx \qquad \text{where } t^2 = -s = 2(\kappa l)^{-1} - l^{-2}. \tag{B.60}$$

The boundary conditions requires $c_1 = ltc_2$ and

$$\tan t = \frac{2lt}{(lt)^2 - 1}. \tag{B.61}$$

To normalize the eigenfunctions, we solve for c_2 in

$$\begin{aligned}
1 = \int_0^1 [\phi(x)]^2 \mathrm{d}x &= \int_0^1 (ltc_2 \cos tx + c_2 \sin tx)^2 \,\mathrm{d}x \\
&= c_2^2 \int_0^1 l^2t^2 \cos^2 tx + \sin^2 tx + 2lt \,\cos tx \,\sin tx \,\mathrm{d}x \\
&= c_2^2 \int_0^1 \frac{l^2t^2+1}{2} + \frac{l^2t^2-1}{2} \cos 2tx + lt \sin 2tx \,\mathrm{d}x \qquad (*) \\
&= c_2^2 \left[\frac{l^2t^2+1}{2} x + \frac{l^2t^2-1}{4t} \sin 2tx - \frac{l}{2} \cos 2tx \right]_0^1 \\
&= c_2^2 \left(\frac{l^2t^2+1}{2} + \frac{l^2t^2-1}{4t} \sin 2t - \frac{l}{2} \cos 2t + \frac{l}{2} \right) \\
&= c_2^2 \left(\frac{l^2t^2+l+1}{2} + \frac{l}{2\tan t} \cdot \frac{2\tan t}{1+\tan^2 t} - \frac{l}{2} \cdot \frac{1-\tan^2 t}{1+\tan^2 t} \right) \qquad (\dagger) \\
&= c_2^2 \left(\frac{l^2t^2+l+1}{2} + \frac{l}{2} \right) \\
&= c_2^2 \left(\frac{l^2t^2+2l+1}{2} \right),
\end{aligned}$$

where the double angle trigonometric formulae are used in $(*)$, and the tangent half-angle formulae and (B.61) are used in $(\dagger)$. Thus the eigenvalues and eigenfunctions are given by

$$\kappa = \frac{2}{lt^2 + l^{-1}} \qquad\qquad \phi(x) = \sqrt{\frac{2}{l^2t^2+2l+1}} \, (lt \cos tx + \sin tx) \,,$$

where the expression for κ is obtained from the definition of t, and the ts are given implicitly by (B.61).

$\square$

For the proposition to be of practical use, we need to be able to compute t. Let $u_l(t) \overset{\text{def}}{=} 2lt/(l^2t^2-1)$, i.e., the function on the right of (B.61). First, observe that if t is solution to (B.61), then $-t$ is also a solution, since both $\tan t$ and $u_l(t)$ are odd functions of t. Moreover, the eigenvalues depends on t only through t^2, and the eigenfunctions are odd functions of t. This means that we only need to consider non-negative values for t for recover all eigenvalues and eigenfunctions. In addition, we know that $t \neq 0$, since if that were not the case, then $s = 0$ which has no associated eigenvalue and eigenfunction, as shown by the first case in the proof above. Therefore, only positive values for t need to be considered.

Restricting to $t > 0$, the curve of $u_l(t)$ against t has a vertical asymptote at $t = 1/l$. The position of this vertical asymptote will determine the set of solutions for $\tan t = u_l(t)$. Let $S \overset{\text{def}}{=} \{2/(2k+1)\pi \mid k \in \mathbb{N}_0\}$. We have the following three cases.

Case $l \in]2/\pi, \infty]$. In this case, the vertical asymptote of $u_l(t)$ is at $t < \pi/2$. Figure B.1a provides an example. The (positive) solutions of $\tan t = u_l(t)$ lie in the set $\{t \mid t > 0, \tan(t) > 0\}$, or equivalently,

$$\bigcup_{k \in \mathbb{N}_0} \left] k\pi, \left(k + \frac{1}{2}\right)\pi \right[.$$

Moreover, each interval above has exactly one solution.

Case $l \in S$. In this case, the vertical asymptote of $u_l(t)$ is at $t = (\tilde{k} + 1/2)\pi$, for a fixed $\tilde{k} \in \mathbb{N}_0$ depending on l. Figure B.1b provides an example. For $t < (k + 1/2)\pi$, the solutions are such that

$\tan t < 0$, and for $t > (k + 1/2)\pi$, the solutions are such that $\tan t > 0$. Thus, the (positive) solutions lie in the set

$$\left\{ \bigcup_{k=1}^{\tilde{k}} \left] \left(k - \frac{1}{2}\right)\pi, k\pi \right[\right\} \cup \left\{ \bigcup_{k=\tilde{k}+1}^{\infty} \left] k\pi, \left(k + \frac{1}{2}\right)\pi \right[\right\}, \qquad \text{where } \tilde{k} = \frac{1}{l\pi} - \frac{1}{2}$$

Moreover, each interval above has exactly one solution.

Case $l \in]0, 2/\pi[\,\backslash S$. This is similar to the case above, except that now there is a period of $\tan t$ that provides two solutions. Figure B.1c provides an example. The (positive) solutions lie in the set

$$\left\{ \bigcup_{k=1}^{\tilde{k}} \left] \left(k - \frac{1}{2}\right)\pi, k\pi \right[\right\} \cup \left\{ \bigcup_{k=\tilde{k}}^{\infty} \left] k\pi, \left(k + \frac{1}{2}\right)\pi \right[\right\}, \qquad \text{where } \tilde{k} = \left\lceil \frac{1}{l\pi} - \frac{1}{2} \right\rceil$$

Moreover, each interval above has exactly one solution. Section 3.5.5, which gives the simulations of learning curve for multi-task learning with the OU covariance function, uses $l = 0.01$, and is of this case.

In every case we can provide a set of intervals containing all the solutions, with each interval bracketing exactly one solution. For a given interval, the exact location of the bracketed solution can be computed starting from an initial guess using the Newton-Raphson method, falling back on the golden-section search or bisection search method if the Newton-Raphson method gives an update outside the interval. The size of the interval can be reduced at each iteration, since both $\tan t$ and $u_l(t)$ are monotonic within the interval.

Let us index the solutions for $\tan t = u_l(t)$ by $i \in \mathbb{N}_1$, and similarly for the corresponding eigenvalues given by Proposition B.2. From the graphs in Figure B.1, it is clear that since $\lim_{t\to\infty} u_l(t) = 0$, the solutions in the limit $i \to \infty$ satisfy $\tan t_i = 0$. Thus $t_i = i\pi$ asymptotically.[1] Hence the asymptotic eigenvalues are given by

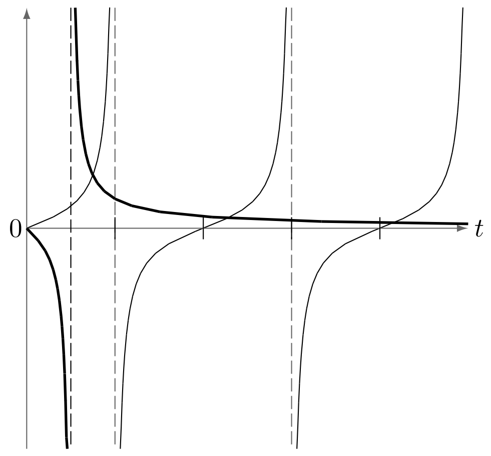
$$\lim_{i\to\infty} \kappa_i = \lim_{i\to\infty} \frac{2}{l(i\pi)^2 + l^{-1}} = \frac{2}{l}(i\pi)^{-2}.$$

This is in agreement with the result of Ritter [2000, Proposition IV.10, Remark IV.2] on the asymptotic eigenvalues of covariance functions satisfying the Sacks-Ylvisaker conditions. The asymptotic eigenvalues of the periodic OU covariance function [Sollich and Halees, 2002, equation 37] on one dimension is $(i\pi)^{-2}/2l$, which is the same as the above modulo the constant multiplicative factor.

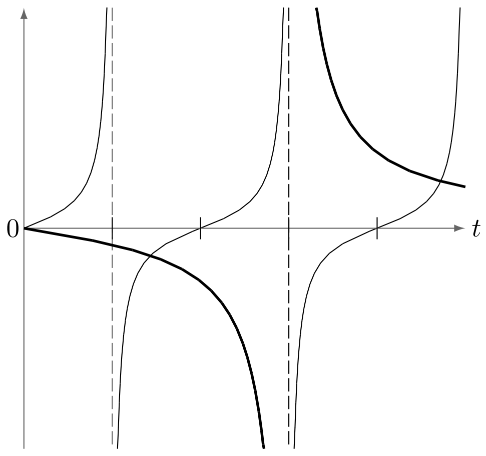
Related work

Hawkins [1989] has given the eigenvalues and eigenfunctions for the OU covariance function with $l = 1$. We have extended his result to general l. Note that the proof for Proposition B.2 can be simplified if we restrict $l = 1$, since this means $\kappa < 1 < 2l$, and only the third case in the proof needs to be considered. In addition to Hawkins, we have also analyzed the solutions to $\tan t = u_l(t)$, and have provided a bracketing interval per solution. This allows for using numerical computations to obtain the solutions.

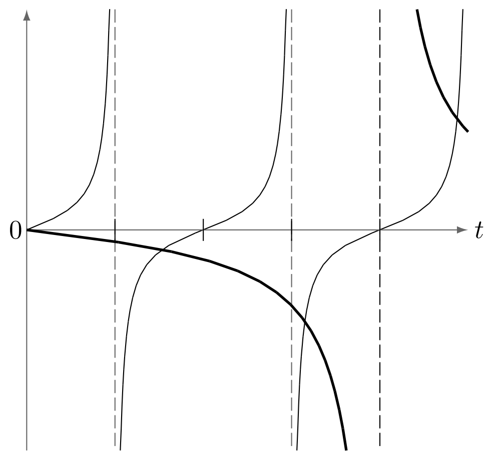
[1] To be exact, if $l \in]0, 2/\pi[\,\backslash S$, then we have $t_i = (i-1)\pi$ instead, since in this case there is a period of $\tan t$ having two solutions. See Figure B.1c.



(a) $l = 4/\pi$



(b) $l = 2/3\pi$



(c) $l = 1/2\pi$

Figure B.1: Plots of $\tan t$ and $u(t)$ against t. Each graph plots $\tan t$, using dark thin lines (—), and $u(t)$, using dark thick lines (—), against t, for $t \in [0, 5\pi/2]$. The vertical asymptote for $u(t)$ is given by the dark dashed line (--). The ticks on the horizontal axis mark intervals of $\pi/2$.

B.7 Simulations of the Learning Curve, Details

In this section, we give additional details for our simulations of the learning curve in section 3.5.5.

B.7.1 Continuation of $\epsilon_T^{\mathrm{avg}}$ in $\pi_S n$

Recall that $\epsilon_T^{\mathrm{avg}}(\rho, \sigma_{\mathrm{n}}^2, \pi_S, n)$ is only defined for values of π_S and n such that $\pi_S n = n_S \in \mathbb{N}_0$. In our simulations, however, we extend the domain to allow $\pi_S n \in \mathbb{R}$, so that smooth curves can be plotted. For the theoretical OV_ρ and $\mathrm{FWO}_{\hat\varrho}$ bounds, this is done by simply using the respective expressions verbatim. For the experimental bounds, which require sampling over $X \overset{\text{def}}{=} X_T \cup X_S$, this is achieved in the manner described next.

For a given π_S, we sample the sizes of the training sets X_T and X_S to satisfy

$$\lfloor \pi_S n \rfloor \leqslant n_S \leqslant \lceil \pi_S n \rceil \qquad \text{and} \qquad \langle n_S \rangle = \pi_S n, \tag{B.62}$$

where the expectation is taken over simulation runs. The first condition ensures that, within each simulation run, the size n_S of X_S is $\pi_S n$ whenever the latter is an integer. The second condition ensures that the ratio n_S/n is consistent with π_S when averaged over multiple simulation runs. For each simulation run, the training set is constructed sequentially by randomly drawing additional training locations. For each new location, we determine its task by using Algorithm 1.

Algorithm 1 Decide the task for a new input

```
Require: Ratio $\pi_S$, required cardinality $n$ of $X$, and and previous cardinality $n_S^{\mathrm{old}}$ of $X_S$.
1: if $n_S^{\mathrm{old}} < \lfloor \pi_S n \rfloor$ then
2:     new input is for task $S$
3: else if $n_S^{\mathrm{old}} = \lceil \pi_S n \rceil$ then
4:     new input is for task $T$
5: else
6:     new input is for task $S$ with probability $(\pi_S n - \lfloor \pi_S n \rfloor)$
       {or, equivalently, for task $T$ with probability $1 - (\pi_S n - \lfloor \pi_S n \rfloor)$}
7: end if
```

B.7.2 Analytical Averaging over Test Locations

The simulation study in section 3.5.5 uses the squared exponential (SE) covariance function with normally distributed inputs, and the covariance function of the stationary Ornstein-Uhlenbeck (OU) process with uniformly distributed inputs. As claimed in section 3.3.4, for either of these cases, the expectation over test locations can be done analytically to obtain the generalization error ϵ_T exactly. In order to do this, we need to be able to compute (see (3.26))

$$M_{pq} \overset{\text{def}}{=} \int k^{\mathrm{x}}(\boldsymbol{x}_p, \boldsymbol{x}_*)\, k^{\mathrm{x}}(\boldsymbol{x}_q, \boldsymbol{x}_*)\, p(\boldsymbol{x}_*) \mathrm{d}\boldsymbol{x}_* = \sum_{i=1}^{\infty} \kappa_i^2 \phi_i(\boldsymbol{x}_p) \phi_i(\boldsymbol{x}_q) \tag{B.63}$$

for a fixed pair of input locations $(\boldsymbol{x}_p, \boldsymbol{x}_q)$. In the one-dimensional case with the SE covariance function and normally distributed inputs, i.e.,

$$k^{\mathrm{x}}(x, x') \overset{\mathrm{def}}{=} \exp -\frac{(x-x')^2}{2l^2} \qquad \text{and} \qquad p(x) \overset{\mathrm{def}}{=} \frac{1}{\sqrt{2\pi}\sigma_x} \exp -\frac{x^2}{2\sigma_x^2}, \tag{B.64}$$

we can use the integral expression for M_{pq} to obtain

$$M_{pq} = \frac{l}{\sqrt{2\sigma_x^2 + l^2}} \exp -\frac{\sigma_x^2 (x_p - x_q)^2 + l^2 (x_p^2 + x_q^2)}{2l^2 (2\sigma_x^2 + l^2)}. \tag{B.65}$$

This can be easily generalized to input spaces of higher-dimensions. Note that the infinite sum expression for M_{pq} is not useful in this case, even though analytic expressions for the eigenfunctions are available Zhu et al. [1998]. This is because the eigenfunctions corresponding to the smaller eigenvalues exhibit larger oscillations around zero in regions where $p(x)$ is low, so that it is hard to determine when to truncate the infinite sum for certain pairs of (x_p, x_q).

For the one-dimensional case with the OU covariance function and uniformly distributed inputs on the unit interval, i.e.,

$$k^{\mathrm{x}}(x, x') \overset{\mathrm{def}}{=} \exp -\frac{|x-x'|}{l} \qquad \text{and} \qquad p(x) \overset{\mathrm{def}}{=} \delta(x \in [0,1]) \tag{B.66}$$

we can again use the integral expression for M_{pq} to obtain

$$M_{pq} = (l + |x_p - x_q|) \exp -\frac{|x_p - x_q|}{l} - \frac{l}{2} \exp -\frac{x_p + x_q}{l} - \frac{l}{2} \exp \frac{x_p + x_q - 2}{l}. \tag{B.67}$$

This can be derived by breaking the range of integration into three intervals given by $[0, \min(x_p, x_q)]$, $[\min(x_p, x_q), \max(x_p, x_q)]$ and $[\max(x_p, x_q), 1]$.

B.8 Asymptotics of the OV and the FWO Bounds on the Learning Curve

In this section, we derive expressions for the asymptotics of the OV lower bound and the FWO upper bound on the learning curve for multi-task GP learning. The asymptotics for the OV bound are in the regime of large number of data points n or small noise variance σ_{n}^2. The expressions given here are more exact then existing ones in the literature. This exactness is necessary in order to obtain results in sections 3.6 and 3.7. First, it is necessary to understand the asymptotic behaviour of the process eigenvalues.

B.8.1 Asymptotic Behaviour of Eigenvalues

The asymptotics of the OV and FWO bounds on the learning curves rests on the premise that

$$\kappa_i = o\left(1/i\right). \tag{B.68}$$

Let us understand why this must be so. First, the covariance function k^{x} is in the trace class, i.e., it has finite trace. Intuitively, this means that the variance must be finite for a Gaussian process. Moreover, Lidskii's Theorem says that $\mathrm{tr}(k^{\mathrm{x}}) = \sum_{i=1}^{\infty} \kappa_i$. Thus $\sum_{i=1}^{\infty} \kappa_i$ must be finite.[2] Applying the Maclaurin-Cauchy test for the convergence of infinite series then leads to (B.68). In this section, our focus is on process eigenvalues of the order of power-law and exponential decays. To be precise, we introduce *idealized* process eigenvalues

$$\acute{\kappa}_i = \begin{cases} \eta_0 (bi)^{-r} & \text{for power-law decay} \\ \eta_0 b^{-i} & \text{for exponential decay} \end{cases} \tag{B.69}$$

for some constants $b > 0$ (for power-law decay) or $b > 1$ (for exponential law decay), and $\eta_0 > 0$. Then, the actual eigenvalues κ_is are equal to the idealized eigenvalues asymptotically, i.e.,

$$\kappa_i \sim \acute{\kappa}_i \qquad \Longleftrightarrow \qquad \forall \varepsilon > 0 \; \exists j \; \forall i > j \; |\kappa_i / \acute{\kappa}_i - 1| < \varepsilon. \tag{B.70}$$

For example, the OU covariance function on the one-dimensional unit uniform interval has power-law decay with $r = 2$, $b = \pi$ and $\eta_0 = 2/l$, where l is the length-scale (see section B.6.2). Throughout this section, we choose a fixed value of $\varepsilon \in [0, 1[$, and let j be the minimum index beyond which the above asymptotics apply for the chosen ε.

Remark on smoothness of functions The case of power-law decay has been treated extensively in the literature. In approximation theory, for Sacks-Ylvisaker conditions of order s, which applies roughly to a Gaussian process on one-dimension that is exactly s-times mean-square differentiable, Ritter [2000, Proposition IV.10, Remark IV.2] has shown that $\kappa_i \propto (\pi i)^{-2s-2}$ in the limit $i \to \infty$. In operator theory, Buescu and Paixão [2007] and others have shown a corresponding result. To understand the intuition, it is useful to restrict our attention to covariance functions on the one-dimensional bounded interval, for which Buescu and Paixão have given the following result.

Lemma B.3. *[Buescu and Paixão, 2007, Lemma 3.4] Let m be a positive integer, and let $k : [0, L]^2 \mapsto \mathbb{C}$ be a positive definite kernel such that for every $m_1 = 0, 1, \ldots m$ and $m_2 = 0, 1, \ldots m$, the partial derivatives $\frac{\partial^{m_1+m_2}}{\partial x^{m_1} \partial y^{m_2}} k(x, y)$ exists and are continuous in $[0, L]^2$. Define the symmetric derivatives*

$$k_m(x, y) \overset{\text{def}}{=} \frac{\partial^{2m}}{\partial x^m \partial y^m} k(x, y).$$

Let $\{\lambda_i(k)\}_{i \in \mathbb{N}_1}$ (resp. $\{\lambda_i(k_m)\}_{i \in \mathbb{N}_1}$) be the sequence of eigenvalues of the integral operator with kernel k (resp. k_m). Then, for every $n \geqslant 2m + 1$,

$$\lambda_{2i}(k) \leqslant L^{2m} \left(\frac{4}{\pi^2} \right) \frac{\lambda_i(k_m)}{(2i - 4m - 1)^{2m}}.$$

In terms of order relations, this is $\lambda_{2i}(k) \in O(i^{-2m} \lambda_i(k_m))$. By choosing $m = 1$ and then nesting the order relation repeatedly, we obtain a result that is consistent with Ritter's: the eigenvalues decays i^{-2} faster for every increase in continuity in the sense of that the derivative $k_{m'}(x, y)$ exists.

For exponential decay of eigenvalues, Raman and Rao [1994] have shown that the decay $o(b^{-i})$ in (B.69) is sufficient for having an infinitely mean square differentiable covariance function on a one-dimensional open interval, but not necessary. In fact, for a one-dimensional closed interval, all that is necessary is for the decay to be $o(b^{-i^{1-\delta}})$, for $\delta > 0$. Nevertheless, we note that the decay $o(b^{-i})$ holds for the conventional non-periodic (see remark below) squared exponential covariance function, which is commonly used in machine learning.

[2] Alternatively, on a finite measure $p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$, Mercer's theorem states that the eigenvalues are absolutely summable.

Remark on dimensionality of the input space The above remark on the continuity of the Gaussian process holds on the one-dimensional input space. For a d-dimensional input space Gaussian process with a stationary covariance function, Widom [1963] relates the process eigenvalues to the spectral density function f:

$$\kappa_i \sim f((2\pi)^d i) \qquad \text{as} \qquad i \to \infty. \tag{B.71}$$

For illustration, consider the isotropic exponential correlation function $k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-a\|\boldsymbol{x} - \boldsymbol{x}'\|)$ that has spectral density function $f(t) \propto (a^2 + t^2)^{-(d+1)/2}$, and the isotropic squared-exponential correlation function $k^{\mathrm{x}}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-a\|\boldsymbol{x} - \boldsymbol{x}'\|^2)$ that has spectral density function $f(t) \propto e^{-t^2/4a}$ [Abrahamsen, 1997; Yaglom, 1987]. We may use (B.71) to obtain, as $i \to \infty$, $\kappa_i \propto i^{-(d+1)}$ for the exponential case and $\kappa_i \propto b^{-i^2}$ for the squared-exponential case. For the squared-exponential, the apparent discrepancy between $\kappa_i \propto b^{-i^2}$ and the $\kappa_i \propto b^{-i}$ stated in the previous remark may be explained by the periodicity of the correlation function as implied by the use of spectral density; see also Sollich and Halees [2002, appendix B].

One implication of the above two remarks is that the power-law decay in (B.69) that we shall consider is rather general, covering the cases of finite differentiability of the Gaussian process. In contrast, the exponential decay in (B.69) is more restrictive; however, as far as we are aware, this is the only analytical Gaussian process case for which useful asymptotic rates for the learning curve bounds can obtained.

B.8.2 OV Bound for Single-task Learning

Recall that the OV bound on the learning curve of a single-task Gaussian Process [Opper and Vivarelli, 1999] is $\sum_{i=1}^{\infty}(1/\kappa_i + n/\sigma_{\mathrm{n}}^2)^{-1}$. We define $\tilde{n} \overset{\text{def}}{=} n/\sigma_{\mathrm{n}}^2$ and $\tilde{n}_{\pm} \overset{\text{def}}{=} (1 \pm \varepsilon)\tilde{n}$, where ε is the fixed chosen value in (B.70). Further, the following condition on $\tilde{n}$ is imposed

$$\frac{j-1}{\kappa_1^{-1} + \tilde{n}} \leqslant (1+\varepsilon)\int_0^{\infty} \frac{1}{\acute{\kappa}_i^{-1} + \tilde{n}_+}\mathrm{d}i; \tag{B.72}$$

later, we discuss when this condition can be satisfied. The OV bound can be upper bounded in the following way.

$$\begin{aligned}
\sum_{i=1}^{\infty} \frac{1}{\kappa_i^{-1} + \tilde{n}} = \sum_{i=1}^{j-1} \frac{1}{\kappa_i^{-1} + \tilde{n}} + \sum_{i=j}^{\infty} \frac{1}{\kappa_i^{-1} + \tilde{n}} &< \sum_{i=1}^{j-1} \frac{1}{\kappa_1^{-1} + \tilde{n}} + \sum_{i=j}^{\infty} \frac{1}{(\acute{\kappa}_i(1+\varepsilon))^{-1} + \tilde{n}} \\
&= \frac{j-1}{\kappa_1^{-1} + \tilde{n}} + (1+\varepsilon)\sum_{i=j}^{\infty} \frac{1}{\acute{\kappa}_i^{-1} + \tilde{n}_+} \\
&\leqslant \frac{j-1}{\kappa_1^{-1} + \tilde{n}} + (1+\varepsilon)\sum_{i=1}^{\infty} \frac{1}{\acute{\kappa}_i^{-1} + \tilde{n}_+} \qquad (*) \\
&< \frac{j-1}{\kappa_1^{-1} + \tilde{n}} + (1+\varepsilon)\int_0^{\infty} \frac{1}{\acute{\kappa}_i^{-1} + \tilde{n}_+}\mathrm{d}i \qquad (\dagger) \\
&\leqslant 2(1+\varepsilon)\int_0^{\infty} \frac{1}{\acute{\kappa}_i^{-1} + \tilde{n}_+}\mathrm{d}i,
\end{aligned} \tag{B.73}$$

where the inequality between $(*)$ and $(\dagger)$ is because the integrand in $(\dagger)$ is a decreasing function of i (this can be easily shown by drawing a sketch), and (B.72) is used in obtaining the final expression.

The integral in (B.72) has the following solutions

$$\int_0^\infty \frac{1}{\acute{\kappa}_i^{-1} + \tilde{n}_+} \mathrm{d}i = \begin{cases} \eta_1 (\eta_0 \tilde{n}_+)^{1/r} / \tilde{n}_+ & \text{if power-law decay,} \\ \eta_1 \log(1 + \eta_0 \tilde{n}_+) \, / \, \tilde{n}_+ & \text{if exponential decay,} \end{cases} \tag{B.74a}$$

where

$$\eta_1^{-1} = \begin{cases} \pi r \sin(\pi / r) / b & \text{if power-law decay,} \\ \log b & \text{if exponential decay;} \end{cases} \tag{B.74b}$$

see Gradshteyn and Ryzhik [2007, §3.222 & §2.313]. For power-law decay, it is necessary that $r > 1$ for the integral to be finite. This is guarantee since (B.68) must be satisfied, as discussed after (B.68). Substituting (B.74) into (B.73) gives

$$\sum_{i=1}^\infty \frac{1}{\kappa_i^{-1} + \tilde{n}} < \begin{cases} 2\eta_1 (\eta_0 \tilde{n}_+)^{1/r} / \tilde{n} & \text{if power-law decay,} \\ 2\eta_1 \log\left(1 + \eta_0 \tilde{n}_+\right) / \tilde{n} & \text{if exponential decay.} \end{cases} \tag{B.75}$$

We now return to discuss (B.72). Using (B.74a), it can be show that inequality (B.72) is valid when

$$j \leqslant \begin{cases} \eta_1 (\eta_0 \tilde{n}_+)^{1/r} & \text{if power-law decay,} \\ \eta_1 \log(\eta_0 \tilde{n}_+) & \text{if exponential decay.} \end{cases} \tag{B.76}$$

In terms of sufficient conditions on $\tilde{n}$, this is $\tilde{n} = \Omega\left(j^r\right)$ and $\tilde{n} = \Omega\left(e^j\right)$ for the respective eigenvalue decays. Recall that j is the minimum index beyond which (B.70) applies, so that j is typically rather large. Thus the condition (B.76) is really a requirement on n being sufficiently large or σ_{n}^2 being sufficiently small.[3]

We are now ready to lower bound the OV bound. First, we choose a i' such that $i' \geqslant j$. Then

$$\sum_{i=1}^\infty \frac{1}{\kappa_i^{-1} + \tilde{n}} > \sum_{i=1}^{i'} \frac{1}{\kappa_i^{-1} + \tilde{n}} > \sum_{i=1}^{i'} \frac{1}{\kappa_{i'}^{-1} + \tilde{n}} = \frac{i'}{\kappa_{i'}^{-1} + \tilde{n}} \geqslant \frac{i'}{(\acute{\kappa}_{i'}(1-\varepsilon))^{-1} + \tilde{n}} = \frac{(1-\varepsilon) i'}{\acute{\kappa}_{i'}^{-1} + \tilde{n}_-}.$$

In particular, if we select i' to have values on the right of the inequality in (B.76) which upper bounds j, then

$$\frac{i'}{\acute{\kappa}_{i'}^{-1} + \tilde{n}_-} = \begin{cases} \eta_2 \eta_1 (\eta_0 \tilde{n}_+)^{1/r} / \tilde{n} & \text{if power-law decay,} \\ \eta_2 \eta_1 \log\left(1 + \eta_0 \tilde{n}_+\right) / \tilde{n} & \text{if exponential decay,} \end{cases} \tag{B.77}$$

where

$$\eta_2 = \begin{cases} (1-\varepsilon) \, / \, [(\pi \eta_1)^r (1+\varepsilon) + (1-\varepsilon)] & \text{if power-law decay,} \\ (1-\varepsilon) \, / \, 2 & \text{if exponential decay.} \end{cases} \tag{B.78}$$

Thus,

$$\sum_{i=1}^\infty \frac{1}{\kappa_i^{-1} + \tilde{n}} > \begin{cases} \eta_2 \eta_1 (\eta_0 \tilde{n}_+)^{1/r} / \tilde{n} & \text{if power-law decay,} \\ \eta_2 \eta_1 \log\left(1 + \eta_0 \tilde{n}_+\right) / \tilde{n} & \text{if exponential decay.} \end{cases} \tag{B.79}$$

[3] The pair of order relations for $\tilde{n}$ does not imply that the single-task OV lower bound enters the asymptotic regime earlier for smaller r rather than bigger r, and for power-law decay rather than exponential decay. Experiments indicate otherwise; cf. the curves for the OU and SE covariance functions in Figure 3.5, and also see Sollich and Halees [2002]. To reconcile, recall that j is fixed given ε, so that the js for the different decays of eigenvalues may be very different.

Putting everything thing together, we have the following: for a chosen ε, we can fix j, and obtain η_1 and η_2 using (B.74b) and (B.78), so that, for all $\tilde{n}$ satisfying (B.76), the single-task OV bound is within factor $]\eta_2\eta_1, 2\eta_1[$ of $(\eta_0\tilde{n}_+)^{1/r}/\tilde{n}$ (for power-law decay), or $\log(1+\eta_0\tilde{n}_+)/\tilde{n}$ (for exponential decay).

We now seek to remove the parameter ε from the asymptotic bounds on the OV bound. It is clear that the lower bound (B.79) still holds if we replace each occurrence of $\tilde{n}_+$ by $\tilde{n}$. For the upper bound (B.75), we may do the following: in the case of power-law decay, $(1+\epsilon)$ is factored out of $\tilde{n}_+$; in the case of the exponential decay, we use $\log(1+\eta_0\tilde{n}_+) < \log((1+\epsilon)(1+\eta_0\tilde{n}))$ to extract an additive term $2\eta_1\log(1+\epsilon)/\tilde{n}$. Thus, we conclude that

$$\sum_{i=1}^{\infty}\frac{1}{\kappa_i^{-1}+\tilde{n}} \in \begin{cases}\Theta\left((\eta_0\tilde{n})^{1/r}/\tilde{n}\right) & \text{if power-law decay,}\\ \Theta\left(\log(1+\eta_0\tilde{n})/\tilde{n}\right) & \text{if exponential decay.}\end{cases} \tag{B.80}$$

B.8.3 OV Bound for Symmetric Multi-task Learning

For symmetric multi-task learning, $(1/2 \pm \sqrt{1/4-\omega})\kappa_i$, where $\omega \overset{\text{def}}{=} \pi_S(1-\pi_S)(1-\rho^2)$, are the eigenvalues; see section B.6.1. The OV lower bound to the learning curve in symmetric multi-task learning is therefore just just the lower bound for the single task, but with these modified eigenvalues. Thus, we may write the OV bound as

$$\sum_{i=1}^{\infty}\sum_{z\in\{-1,1\}}\frac{1}{[(1/2+z\sqrt{1/4-\omega})\kappa_i]^{-1}+\tilde{n}} \tag{B.81}$$

$$= \sum_{z\in\{-1,1\}}\sum_{i=1}^{\infty}\frac{(1/2+z\sqrt{1/4-\omega})}{\kappa_i^{-1}+(1/2+z\sqrt{1/4-\omega})\tilde{n}} \tag{B.82}$$

$$\in\,]\eta_2\eta_1, 2\eta_1[\, \sum_{z\in\{-1,1\}}\begin{cases}(\eta_0(1/2+z\sqrt{1/4-\omega})\tilde{n}_+)^{1/r}\,/\,\tilde{n} & \text{if power-law decay,}\\ \log\left(1+\eta_0(1/2+z\sqrt{1/4-\omega})\tilde{n}_+\right)\,/\,\tilde{n} & \text{if exponential decay.}\end{cases} \tag{B.83}$$

For power-law decay with $r=2$, which is the case for the OU stationary process on the one-dimension, we have

$$]\eta_2\eta_1, 2\eta_1[\ \cdot\ \frac{1}{\tilde{n}}(\eta_0\tilde{n}_+)^{1/2}\underline{\sqrt{1+2\sqrt{\omega}}} \tag{B.84}$$

where the underlined factor can be derived by considering $\sqrt{1/2-\sqrt{1/4-\omega}}+\sqrt{1/2+\sqrt{1/4-\omega}}$ squared. The exponential decay case can also be simplified to give

$$]\eta_2\eta_1, 2\eta_1[\ \cdot\ \frac{1}{\tilde{n}}\log\left[1+\eta_0\tilde{n}_+ + \underline{\omega(\eta_0\tilde{n}_+)^2}\right] \tag{B.85}$$

The above two equations are the same as the corresponding ones for single-task bound, with the exceptions of the underlined terms.

B.8.4 FWO Bound for Single-task Learning

We now examine the asymptotics of the FWO upper bound for the single-task GP with *isotropic noise* and *stationary covariance function*. First, let $\underset{\sim}{n}_i \overset{\text{def}}{=} n/(\sum_{i'\neq i}\kappa_{i'}+\sigma_n^2)$, and $\underset{\sim}{n} \overset{\text{def}}{=} n/(\sum_{i'}\kappa_{i'}+\sigma_n^2)$.

For all i, $\underset{\sim}{n}_i > \underset{\sim}{n}$ clearly. Then the FWO upper bound may be written as follows:

$$\sum_i \kappa_i - n \sum_i \frac{\kappa_i^2}{(n-1)\kappa_i + \sum_{i'} \kappa_{i'} + \sigma_{\mathrm{n}}^2}$$
$$= \sum_i \kappa_i \frac{\sum_{i' \neq i} \kappa_{i'} + \sigma_{\mathrm{n}}^2}{n\kappa_i + \sum_{i' \neq i} \kappa_{i'} + \sigma_{\mathrm{n}}^2} = \sum_i \frac{1}{\underset{\sim}{n}_i + \kappa_i^{-1}} < \sum_i \frac{1}{\underset{\sim}{n} + \kappa_i^{-1}}. \quad \text{(B.86)}$$

Notice that the last expression has the form of the single-task OV lower bound. Using the same arguments as for the single-task OV bound leading to equation B.75, and letting $\underset{\sim}{n}_+ \overset{\mathrm{def}}{=} (1+\varepsilon)\underset{\sim}{n}$, we can say

$$\sum_i \kappa_i - n \sum_i \frac{\kappa_i^2}{(n-1)\kappa_i + \sum_{i'} \kappa_{i'} + \sigma_{\mathrm{n}}^2} < \begin{cases} 2\eta_1 (\eta_0 \underset{\sim}{n}_+)^{1/r} / \underset{\sim}{n} & \text{if power-law decay,} \\ 2\eta_1 \log(1 + \eta_0 \underset{\sim}{n}_+) / \underset{\sim}{n} & \text{if exponential decay,} \end{cases} \quad \text{(B.87)}$$

Thus,

$$\sum_i \kappa_i - n \sum_i \frac{\kappa_i^2}{(n-1)\kappa_i + \sum_{i'} \kappa_{i'} + \sigma_{\mathrm{n}}^2} \in \begin{cases} O\left((\eta_0 \underset{\sim}{n})^{1/r} / \underset{\sim}{n}\right) & \text{if power-law decay,} \\ O\left(\log(1 + \eta_0 \underset{\sim}{n}) / \underset{\sim}{n}\right) & \text{if exponential decay.} \end{cases} \quad \text{(B.88)}$$

This is the asymptotic upper bound to the single-task FWO bound. We now turn to the asymmetric multi-task FWO bound.

B.8.5 FWO Bound for Asymmetric Multi-task Learning

We now derive the asymptotics for the FWO$_\varrho$ upper bound for the case of stationary covariance function k^{x}. of Proposition 3.16 on page 67 Let

$$\beta_1 = \frac{\pi_S^2 \varrho^2 (1-\rho^2)}{(1-\pi_S)^2 + 2\pi_S(1-\pi_S)\rho\varrho + \pi_S^2\varrho^2} \qquad \beta_2 = \frac{(1-\pi_S)^2 + 2\pi_S(1-\pi_S)\rho\varrho + \pi_S^2\varrho^2}{1 - \pi_S + \pi_S \varrho^2}. \quad \text{(B.89)}$$

Then

$$c_i(\varrho) = (\beta_2 n - 1)\kappa_i + \sum_j \kappa_j + \sigma_{\mathrm{n}}^2 = \beta_2 n \kappa_i + \sum_{j \neq i} \kappa_j + \sigma_{\mathrm{n}}^2 = \left(\beta_2 \underset{\sim}{n}_i \kappa_i + 1\right) \left(\textstyle\sum_{j \neq i} \kappa_j + \sigma_{\mathrm{n}}^2\right),$$

where $\underset{\sim}{n}_i \overset{\mathrm{def}}{=} n / (\sum_{i' \neq i} \kappa_{i'} + \sigma_{\mathrm{n}}^2)$ as in the preceding section. Using $\alpha(x) \overset{\mathrm{def}}{=} (1-\pi_S) + \pi_S x$, the FWO$_\varrho$ bound given by the left of equation 3.33 on page 67 can now be expressed as

$$\begin{aligned}
&\sum_{i=1}^{\infty} \kappa_i - n \frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)} \sum_{i=1}^{\infty} \frac{\kappa_i^2}{\left(\beta_2 \underset{\sim}{n}_i \kappa_i + 1\right) \left(\sum_{j \neq i} \kappa_j + \sigma_{\mathrm{n}}^2\right)} \\
&= \sum_{i=1}^{\infty} \kappa_i - \frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)} \sum_{i=1}^{\infty} \frac{\underset{\sim}{n}_i \kappa_i^2}{\beta_2 \underset{\sim}{n}_i \kappa_i + 1} \\
&= \sum_{i=1}^{\infty} \kappa_i - \frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)} \sum_{i=1}^{\infty} \frac{\underset{\sim}{n}_i \kappa_i^2 + \kappa_i - \kappa_i}{\beta_2 \underset{\sim}{n}_i \kappa_i + 1} \\
&= \sum_{i=1}^{\infty} \kappa_i - \frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)} \left(\sum_{i=1}^{\infty} \kappa_i - \sum_{i=1}^{\infty} \frac{1}{\beta_2 \underset{\sim}{n}_i + \kappa_i^{-1}} \right) \\
&= \left(1 - \frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)}\right) \sum_{i=1}^{\infty} \kappa_i + \frac{[\alpha(\rho\varrho)]^2}{\alpha(\varrho^2)} \sum_{i=1}^{\infty} \frac{1}{\beta_2 \underset{\sim}{n}_i + \kappa_i^{-1}}
\end{aligned}$$

By showing $1 - \beta_1 = [\alpha(\rho\varrho)]^2/\alpha(\varrho^2)$, we obtain

$$\beta_1 \sum_{i=1}^{\infty} \kappa_i + (1-\beta_1) \sum_{i=1}^{\infty} \frac{1}{\beta_2 \underset{\sim}{n}_i + \kappa_i^{-1}}. \tag{B.90}$$

We now show that $\beta_1 \in [0,1]$. The lower bound is obtained by setting $\rho = 1$. The upper bound is obtained by observing that the numerator and denominator in β_1 are positive, and that they are less than and greater than $\pi_S^2 \varrho^2$ respectively.

Since $\beta_1 \in [0,1]$, so one may view (B.90) as a weighted average between the average prior variance and a term that depends on sample size. The arguments to obtain the asymptotics for the single-task FWO bound can now be applied on the second summation in (B.90) to obtain the following upper bounds on the asymmetric multi-task FWO bound

$$\begin{array}{ll} \beta_1 \sum_{i=1}^{\infty} \kappa_i + 2(1-\beta_1)\eta_1(\eta_0\beta_2\underset{\sim}{n}_+)^{1/r}/\beta_2\underset{\sim}{n} & \text{if power-law decay,} \\ \beta_1 \sum_{i=1}^{\infty} \kappa_i + 2(1-\beta_1)\eta_1 \log(1+\eta_0\beta_2\underset{\sim}{n}_+)/\beta_2\underset{\sim}{n} & \text{if exponential decay.} \end{array} \tag{B.91}$$

B.9 Proof for Equation 3.69

We wish to compute $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, 1)$, when there is only $n = 1$ sample. With probability π_S, the sample is for task S, and the learning curve is the same as $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 1, 1)$; with probability $1 - \pi_S$, the sample is for task T, and the learning curve is the same as $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 0, 1)$. Thus

$$\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, 1) = \pi_S \epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 1, 1) + (1-\pi_S)\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 0, 1).$$

If $\pi_S = 0$, then ρ has no effect on the learning curve, so $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 0, 1) = \epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 0, 1)$. This gives

$$\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, 1) = \pi_S \epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 1, 1) + (1-\pi_S)\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 0, 1).$$

Using Corollary 3.13 on $\epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, 1, 1)$ gives

$$\begin{aligned} \epsilon_T^{\text{avg}}(\rho, \sigma_{\text{n}}^2, \pi_S, 1) &= \pi_S \left(\rho^2 \epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 0, 1) + (1-\rho^2) \sum_{i=1}^{\infty} \kappa_i \right) + (1-\pi_S)\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 0, 1) \\ &= \pi_S(1-\rho^2) \sum_{i=1}^{\infty} \kappa_i + [1 - \pi_S(1-\rho^2)]\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 0, 1). \end{aligned}$$

Finally, when $\rho = 1$, the value of π_S does not matter, so $\epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 0, 1) = \epsilon_T^{\text{avg}}(1, \sigma_{\text{n}}^2, 1, 1)$. $\square$

B.10 The Posterior Distribution Given Collocated Observations

We follow the notations in section 3.9.3. The posterior distribution given by (3.80) is proportional to the product of three Gaussians:

$$p(f_T(\boldsymbol{x}_*) \mid \boldsymbol{y}_{T\setminus *}, \boldsymbol{y}_{S*}) \propto p(\boldsymbol{y}_{T\setminus *} \mid f_T(\boldsymbol{x}_*))\, p(\boldsymbol{y}_{S*} \mid f_T(\boldsymbol{x}_*))\, p(f_T(\boldsymbol{x}_*)).$$

Thus, the exponent in posterior distribution is given by $-1/2$ times

$$\begin{aligned}\left[\boldsymbol{y}_{T\backslash *} - \boldsymbol{k}^{\mathrm{x}}_{*} f_T(\boldsymbol{x}_*)/k^{\mathrm{x}}_{**}\right]^{\mathrm{T}} \left[K^{\mathrm{x}}_{\backslash *} - \boldsymbol{k}^{\mathrm{x}}_{*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}/k^{\mathrm{x}}_{**}\right]^{-1} \left[\boldsymbol{y}_{T\backslash *} - \boldsymbol{k}^{\mathrm{x}}_{*} f_T(\boldsymbol{x}_*)/k^{\mathrm{x}}_{**}\right] \\ + \left[\boldsymbol{y}_{S*} - (k^{\mathrm{x}}_{**}\boldsymbol{\rho}) f_T(\boldsymbol{x}_*)/k^{\mathrm{x}}_{**}\right]^{\mathrm{T}} \left[k^{\mathrm{x}}_{**}K^{\mathrm{f}}_{S} - (k^{\mathrm{x}}_{**}\boldsymbol{\rho})(k^{\mathrm{x}}_{**}\boldsymbol{\rho})^{\mathrm{T}}/k^{\mathrm{x}}_{**}\right]^{-1} \left[\boldsymbol{y}_{S*} - (k^{\mathrm{x}}_{**}\boldsymbol{\rho}) f_T(\boldsymbol{x}_*)/k^{\mathrm{x}}_{**}\right] \\ + \left[f_T(\boldsymbol{x}_*)\right]^2/k^{\mathrm{x}}_{**}\end{aligned} \tag{B.92}$$

The terms involving $(k^{\mathrm{x}}_{**}\boldsymbol{\rho})$ can be simplified by cancelling the k^{x}_{**} within with the $1/k^{\mathrm{x}}_{**}$s outside. We use the Sherman-Morrison formula

$$(A + \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}})^{-1} = A^{-1} - \frac{A^{-1}\boldsymbol{u}\boldsymbol{v}^{\mathrm{T}}A^{-1}}{1 + \boldsymbol{v}^{\mathrm{T}}A^{-1}\boldsymbol{u}}$$

to simplify the matrix inversions. Hence,

$$\begin{aligned}\left[K^{\mathrm{x}}_{\backslash *} - \boldsymbol{k}^{\mathrm{x}}_{*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}/k^{\mathrm{x}}_{**}\right]^{-1} &= (K^{\mathrm{x}}_{\backslash *})^{-1} + \frac{(K^{\mathrm{x}}_{\backslash *})^{-1}\boldsymbol{k}^{\mathrm{x}}_{*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}/k^{\mathrm{x}}_{**}}{1 - (\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}\boldsymbol{k}^{\mathrm{x}}_{*}/k^{\mathrm{x}}_{**}} \\ &= (K^{\mathrm{x}}_{\backslash *})^{-1} + \frac{(K^{\mathrm{x}}_{\backslash *})^{-1}\boldsymbol{k}^{\mathrm{x}}_{*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}}{\sigma^2_*},\end{aligned} \tag{B.93a}$$

where

$$\sigma^2_* \overset{\mathrm{def}}{=} k^{\mathrm{x}}_{**} - (\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}\boldsymbol{k}^{\mathrm{x}}_{*}, \tag{B.93b}$$

and

$$\left[k^{\mathrm{x}}_{**}K^{\mathrm{f}}_{S} - k^{\mathrm{x}}_{**}\boldsymbol{\rho}\boldsymbol{\rho}^{\mathrm{T}}\right]^{-1} = \frac{1}{k^{\mathrm{x}}_{**}}\left[(K^{\mathrm{f}}_{S})^{-1} + \frac{(K^{\mathrm{f}}_{S})^{-1}\boldsymbol{\rho}\boldsymbol{\rho}^{\mathrm{T}}(K^{\mathrm{f}}_{S})^{-1}}{v^2_T}\right], \tag{B.93c}$$

where

$$v^2_T \overset{\mathrm{def}}{=} 1 - \boldsymbol{\varrho}^{\mathrm{T}}(K^{\mathrm{f}}_{S})^{-1}\boldsymbol{\varrho}. \tag{B.93d}$$

Using the above expressions for the matrix inversions, we obtain

$$\begin{aligned}\frac{1}{k^{\mathrm{x}}_{**}}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}\left[K^{\mathrm{x}}_{\backslash *} - \boldsymbol{k}^{\mathrm{x}}_{*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}/k^{\mathrm{x}}_{**}\right]^{-1} &= \frac{1}{k^{\mathrm{x}}_{**}}\Bigg[(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1} \\ &\qquad + \frac{(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}\boldsymbol{k}^{\mathrm{x}}_{*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}}{\sigma^2_*}\Bigg] \\ &= \frac{1}{k^{\mathrm{x}}_{**}}\left[1 + \frac{k^{\mathrm{x}}_{**} - \sigma^2_*}{\sigma^2_*}\right](\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1} \\ &= \frac{1}{\sigma^2_*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}\end{aligned} \tag{B.94a}$$

$$\begin{aligned}\frac{1}{(k^{\mathrm{x}}_{**})^2}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}\left[K^{\mathrm{x}}_{\backslash *} - \boldsymbol{k}^{\mathrm{x}}_{*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}/k^{\mathrm{x}}_{**}\right]^{-1}\boldsymbol{k}^{\mathrm{x}}_{*} &= \frac{1}{k^{\mathrm{x}}_{**}\sigma^2_*}(\boldsymbol{k}^{\mathrm{x}}_{*})^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}\boldsymbol{k}^{\mathrm{x}}_{*} \\ &= \frac{(k^{\mathrm{x}}_{**} - \sigma^2_*)}{k^{\mathrm{x}}_{**}\sigma^2_*}.\end{aligned} \tag{B.94b}$$

Similarly

$$\boldsymbol{\rho}^{\mathrm{T}}\left[k^{\mathrm{x}}_{**}K^{\mathrm{f}}_{S} - k^{\mathrm{x}}_{**}\boldsymbol{\rho}\boldsymbol{\rho}^{\mathrm{T}}\right]^{-1} = \frac{1}{k^{\mathrm{x}}_{**}v^2_T}\boldsymbol{\rho}^{\mathrm{T}}(K^{\mathrm{f}}_{S})^{-1} \tag{B.94c}$$

$$\boldsymbol{\rho}^{\mathrm{T}}\left[k^{\mathrm{x}}_{**}K^{\mathrm{f}}_{S} - k^{\mathrm{x}}_{**}\boldsymbol{\rho}\boldsymbol{\rho}^{\mathrm{T}}\right]^{-1}\boldsymbol{\rho} = \frac{1 - v^2_T}{k^{\mathrm{x}}_{**}v^2_T} \tag{B.94d}$$

Using the equations in (B.94), we can simplify the expansion of B.92. In the expansion, the coefficient for the term quadratic in $f_T(\boldsymbol{x}_*)$ is

$$\begin{aligned}
&\frac{1}{(k^{\mathrm{x}}_{**})^2}(\boldsymbol{k}^{\mathrm{x}}_*)^{\mathrm{T}}\left[K^{\mathrm{x}}_{\backslash *}-\boldsymbol{k}^{\mathrm{x}}_*(\boldsymbol{k}^{\mathrm{x}}_*)^{\mathrm{T}}/k^{\mathrm{x}}_{**}\right]^{-1}\boldsymbol{k}^{\mathrm{x}}_*+\boldsymbol{\rho}^{\mathrm{T}}\left[k^{\mathrm{x}}_{**}K^{\mathrm{f}}_S-k^{\mathrm{x}}_{**}\boldsymbol{\rho}\boldsymbol{\rho}^{\mathrm{T}}\right]^{-1}\boldsymbol{\rho}+\frac{1}{k^{\mathrm{x}}_{*}*}\\
&=\frac{(k^{\mathrm{x}}_{**}-\sigma^2_*)}{k^{\mathrm{x}}_{**}\sigma^2_*}+\frac{1-v^2_T}{k^{\mathrm{x}}_{**}v^2_T}+\frac{1}{k^{\mathrm{x}}_{*}*}\\
&=\frac{1}{\sigma^2_*}+\frac{1}{k^{\mathrm{x}}_{**}v^2_T}-\frac{1}{k^{\mathrm{x}}_{**}},
\end{aligned} \tag{B.95}$$

where the coefficient for the term linear in $f_T(\boldsymbol{x}_*)$ is

$$\begin{aligned}
&-2(\boldsymbol{k}^{\mathrm{x}}_*/k^{\mathrm{x}}_{**})^{\mathrm{T}}\left[K^{\mathrm{x}}_{\backslash *}-\boldsymbol{k}^{\mathrm{x}}_*(\boldsymbol{k}^{\mathrm{x}}_*)^{\mathrm{T}}/k^{\mathrm{x}}_{**}\right]^{-1}\boldsymbol{y}_{T\backslash *}-2\boldsymbol{\rho}^{\mathrm{T}}\left[k^{\mathrm{x}}_{**}K^{\mathrm{f}}_S-k^{\mathrm{x}}_{**}\boldsymbol{\rho}\boldsymbol{\rho}^{\mathrm{T}}\right]^{-1}\boldsymbol{y}_{S*}\\
&=-2\left\{\frac{1}{\sigma^2_*}(\boldsymbol{k}^{\mathrm{x}}_*)^{\mathrm{T}}(K^{\mathrm{x}}_{\backslash *})^{-1}\boldsymbol{y}_{T\backslash *}+\frac{1}{k^{\mathrm{x}}_{**}v^2_T}\boldsymbol{\rho}^{\mathrm{T}}(K^{\mathrm{f}}_S)^{-1}\boldsymbol{y}_{S*}\right\}.
\end{aligned} \tag{B.96}$$

Matching the above coefficients to those for a Gaussian distribution will give the mean and variance of the prediction for $f_T(\boldsymbol{x}_*)$.

Appendix C

Appendix to Chapter 4

C.1 Initialization for Optimization

As discussed in section 4.3.3, we treat the hyperparameters as fixed but unknown, and find their values by maximizing the likelihood. We start optimizing from the set of values that gives equal weightings among the inputs and among the components of the covariance function (except for the jitter/noise term). Since our Gaussian process prior has zero mean, we distribute the sample second moment of the torques evenly within the covariance function among the constant term, the term linear in inputs, the squared exponential term and the term linear in the $\mathrm{sgn}(\dot{q}_j)$s.

For clarity, we restrict the notation to the case for the torque function of a single joint, and omit the unnecessary subscripts. Let $\overline{t^2}$ be the sample second moment of the observed torques, and $\overline{x_j}$ and $\overline{x_j^2}$ be the sample first and second moments of the $j\mathrm{th}$ covariate respectively. We set the initial values such that

$$b^2 = \frac{1}{4}\overline{t^2} \quad u_j^2\overline{x_j^2} = \frac{1}{3\ell}\frac{1}{4}\overline{t^2} \quad v_j^2 = \frac{1}{4}\overline{t^2} \quad l_j^2 = \overline{x_j^2} - (\overline{x_j})^2 \quad w_j^2 = \frac{1}{\ell}\frac{1}{4}\overline{t^2} \quad \sigma^2 = 0.01\,\overline{t^2}. \tag{C.1}$$

C.2 Marginal Likelihood of Noise-free Multi-task GP in Complete Block Design

In this appendix, we give an expression for the marginal likelihood of multi-task GP model in the case when there is no noise in the observations, and when there is a complete block design. For the sake of brevity and clarity, we drop the subscripts for the joints.

Let the complete block design or isotopic data consists of $n^\bullet$ inputs $X^\bullet \overset{\text{def}}{=} \{\boldsymbol{x}^{\bullet(j)}\}_{j=1}^{n^\bullet}$ on which torques for all M contexts are observed. Denote the observed torques for the $m\mathrm{th}$ context by the vector $\boldsymbol{t}^m \overset{\text{def}}{=} (t^{m\,(1)}, \ldots, t^{m\,(n^\bullet)})^{\mathrm{T}}$, and the vector of all observed torques by $\boldsymbol{t} \overset{\text{def}}{=} \big((\boldsymbol{t}^1)^{\mathrm{T}}, \ldots, (\boldsymbol{t}^M)^{\mathrm{T}}\big)^{\mathrm{T}}$. Using K^{x} for the matrix of covariances for all pairs in $X^\bullet$ due to k^{x}, and $\boldsymbol{\theta}^{\mathrm{x}}$ for the parameters in k^{x}, we can write the log marginal likelihood of noise-free multi-task GP as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}^{\mathrm{x}}, K^{\rho}) &\overset{\text{def}}{=} \log p(\boldsymbol{t} \mid X^{\bullet}, \boldsymbol{\theta}^{\mathrm{x}}, K^{\rho}) \\ &= -\frac{1}{2}\log|K^{\rho} \otimes K^{\mathrm{x}}| - \frac{1}{2}\boldsymbol{t}^{\mathrm{T}}\left(K^{\rho} \otimes K^{\mathrm{x}}\right)^{-1}\boldsymbol{t} - \frac{Mn^{\bullet}}{2}\log 2\pi \qquad \text{(C.2)} \\ &= -\frac{n^{\bullet}}{2}\log|K^{\rho}| - \frac{M}{2}\log|K^{\mathrm{x}}| - \frac{1}{2}\operatorname{tr}\left[T^{\mathrm{T}}(K^{\mathrm{x}})^{-1}T(K^{\rho})^{-1}\right] - \frac{Mn^{\bullet}}{2}\log 2\pi,\end{aligned}$$

where T is the $n^{\bullet}$-by-M matrix satisfying $\operatorname{vec} T \overset{\text{def}}{=} \boldsymbol{t}$. The partial derivative with respect to K^{ρ} is

$$\frac{\partial \mathcal{L}}{\partial K^{\rho}} = -\frac{n^{\bullet}}{2}(K^{\rho})^{-1} + \frac{1}{2}(K^{\rho})^{-1}T^{\mathrm{T}}(K^{\mathrm{x}})^{-1}T(K^{\rho})^{-1}, \qquad \text{(C.3)}$$

so that we may write the optimal K^{ρ} as a function of $\boldsymbol{\theta}^{\mathrm{x}}$:

$$\hat{K}^{\rho}(\boldsymbol{\theta}^{\mathrm{x}}) = \frac{1}{n^{\bullet}}T^{\mathrm{T}}(K^{\mathrm{x}})^{-1}T. \qquad \text{(C.4)}$$

Substituting this into C.2 gives

$$\mathcal{L}(\boldsymbol{\theta}^{\mathrm{x}}, \hat{K}^{\rho}(\boldsymbol{\theta}^{\mathrm{x}})) = -\frac{n^{\bullet}}{2}\log\left|\frac{1}{n^{\bullet}}T^{\mathrm{T}}(K^{\mathrm{x}})^{-1}T\right| - \frac{M}{2}\log|K^{\mathrm{x}}| - \frac{Mn^{\bullet}}{2}(1 + \log 2\pi). \qquad \text{(C.5)}$$

To maximizing the marginal likelihood, we can optimize the above expression with respect to $\boldsymbol{\theta}^{\mathrm{x}}$.

C.3 Learning Hyperparameters of the Multi-task GP Model by Staged Optimization

In this appendix, we give details on how one of the maxima on the likelihood surface is located. This is done in stages, each stage locating a maximum near to an initialized point that corresponds to the desired interpretation. For the sake of brevity and clarity, we drop the joint index i. The following applies separately for each joint.

Let $\boldsymbol{t}^m$ be the vector of n^m observed torques at the joint for context m, and X^m be the corresponding 3ℓ-by-n^m design matrix, i.e. each column of X^m is a vector consisting of ℓ joint angles, ℓ joint velocities and ℓ joint accelerations. Further, let X be the 3ℓ-by-n^{uniq} design matrix of distinct $\boldsymbol{x}$-configurations observed over all M contexts.

Given this data, the marginal likelihood $L(\boldsymbol{\theta}^{\mathrm{x}}, K^{\rho}, \sigma^2) \overset{\text{def}}{=} p(\{\boldsymbol{t}^m\}_{m=1}^{M} \mid X, \boldsymbol{\theta}^{\mathrm{x}}, K^{\rho}, \sigma^2)$, where $\boldsymbol{\theta}^{x}$ collects the parameters of k^{x}, is optimized using a gradient-based method. We propose a staged strategy during optimization to help localize the search region. This is outlined below, with details given in the subsections that follow.

```
Require: Starting positions $\boldsymbol{\theta}_0^{\mathrm{x}}$, $K_0^{\rho}$, $\sigma_0^2$, and rank $r$
    {All $\arg\max$ operations are understood to find only the local maximum.}
1:  Starting from $\boldsymbol{\theta}_0^{\mathrm{x}}$ and $\sigma_0^2$, find $(\boldsymbol{\theta}_1^{\mathrm{x}}, \sigma_1^2) = \arg\max_{\boldsymbol{\theta}^{\mathrm{x}},\sigma^2} L(\boldsymbol{\theta}^{\mathrm{x}}, K_0^{\rho}, \sigma^2)$.
2:  Calculate $K_1^{\rho}$ based on details in section C.3.2
3:  Starting from $\boldsymbol{\theta}_1^{\mathrm{x}}$, $K_1^{\rho}$, and $\sigma_0^2$, find $(\boldsymbol{\theta}_{\text{ans}}^{\mathrm{x}}, K_{\text{ans}}^{\rho}, \sigma_{\text{ans}}^2) = \arg\max_{\boldsymbol{\theta}^{\mathrm{x}},K^{\rho},\sigma^2} L(\boldsymbol{\theta}^{\mathrm{x}}, K^{\rho}, \sigma^2)$.
```

The order of optimization reflects the relative importance we place on the different constituents of the model: k^{x} is the most important, and σ^2 the least.

C.3.1 The Initial Choice of K^ρ

The choice of K_0^ρ is important, since it affects the search very early on. Reasonable values that admit ready interpretations are the matrix of ones $1_{M\times M}$ and the identity matrix $I_{M\times M}$. For $K_0^\rho = 1_{M\times M}$, we initially assume the contexts to be indistinguishable from each other; while for $K_0^\rho = I_{M\times M}$, we initially assume the contexts to be independent given the kernel parameters, which is a multi-task learning model that has been previously explored in, for example, Minka and Picard [1997]. These two are at the opposite extremes in the spectrum of inter-context/task correlation, and we believe the merit of each will be application dependent. Since these two models have the same number of free parameters, we select the one with the higher likelihood as the starting point for the search in step 2. However, we note that in some applications there may be reasons to prefer one over the other.

C.3.2 Computation of K_1^ρ in Step 2

Given estimates $\boldsymbol{\theta}_1^{\mathrm{x}}$ and σ_1^2, we wish to estimate a K_1^ρ from which the likelihood can be optimized in step 3. Here we give the sequence of considerations that leads to a formula for computing K_1^ρ.

Let K_1^{x} be the covariance matrix for all pairs in X, using $\boldsymbol{\theta}_1^{\mathrm{x}}$ for c^{x}. Let T be the n^{uniq}-by-M matrix that corresponds to the true values of the torque function $\tau^m(\boldsymbol{x}^{(j)})$ for $m = 1\dots M$ and $j = 1\dots n^{\mathrm{uniq}}$. Then as per the Expectation-Maximization (EM) step discussed in [Bonilla et al., 2008, eq. 4], we have

$$K_{\mathrm{EM}}^\rho = \frac{1}{n^{\mathrm{uniq}}}\left\langle T^{\mathrm{T}}(K_1^{\mathrm{x}})^{-1}T\right\rangle \simeq \frac{1}{n^{\mathrm{uniq}}}\langle T\rangle^{\mathrm{T}}(K_1^{\mathrm{x}})^{-1}\langle T\rangle\,, \tag{C.6}$$

where the expectations are taken with respect to a GP with parameters $\boldsymbol{\theta}_1^{\mathrm{x}}$, K_0^ρ and σ_1^2, and the $(j,m)\mathrm{th}$ entry of $\langle T\rangle$ is the mean prediction of $\tau^m(\boldsymbol{x}^{(j)})$ using this GP. If all the torques in T are observed so that the expectation is a null operation, then the equality the same as (C.4). The approximation neglects the variance of the GP; this is justifiable since the current aim is only to obtain a starting estimate of K^ρ for a search procedure.

There are two weaknesses with this that we shall address. The first is that the rank of $\langle T\rangle$ is upper bounded by that of K_0^ρ, so that the rank of K_{EM}^ρ is similarly upper bounded.[1] This property is undesirable, particularly when $K_0^\rho = 1_{M\times M}$. We ameliorate this by replacing $\left\langle\tau^m(\boldsymbol{x}^{(j)})\right\rangle$ with the corresponding observed value $t^m(\boldsymbol{x}^{(j)})$ wherever it is available, and call the resultant matrix T_{aug}. The second weakness is this: with the commonly used covariance functions, K_1^{x} will typically have rapidly decaying eigenvalues [Rasmussen and Williams, 2006, §4.3.1]. To overcome this, we regularize its inversion by adding $\eta^2 I$ to the diagonal of K_1^{x} to give

$$K_{\mathrm{aug}}^\rho = \frac{1}{n^{\mathrm{uniq}}}T_{\mathrm{aug}}^{\mathrm{T}}(K_1^{\mathrm{x}} + \eta^2 I)^{-1}T_{\mathrm{aug}}. \tag{C.7}$$

We set η^2 to $\mathrm{tr}(T_{\mathrm{aug}}^{\mathrm{T}}T_{\mathrm{aug}})/(Mn^{\mathrm{uniq}})$, so that $\mathrm{tr}(K_{\mathrm{aug}}^\rho) = M$ if K_1^{x} were to be a zero matrix. Other ways of setting η^2 are also possible.

Finally, the required K_1^ρ is the rank r constrained positive semi-definite matrix of K_{aug}^ρ. This can be achieved either by computing the eigen-decomposition of K_{aug}^ρ and keeping only the top r eigenvectors/values, or by using an incomplete Cholesky decomposition.

[1] This is not due to our approximation; indeed, it can be shown that the rank of K_{EM}^ρ is upper bounded by that of K_0^ρ even if the exact EM update in (C.6) has been used.

Bibliography

P. Abrahamsen. A review of Gaussian random fields and correlation functions. Technical Report 917, Norwegian Computing Center, Oslo, 1997. p. 165

E. H. Adelson, S. E. P., and W. T. Freeman. Pyramids and multiscale representations. In A. Gorea, editor, *Representations of Vision*, pages 3–16. Cambridge University Press, 1991. p. 49

R. J. Adler. *The Geometry of Random Fields*. Wiley, London, 1981. p. 58

A. S. Almeida and A. G. Journel. Joint simulation of multiple variables with a Markov-type coregionalization model. *Mathematical Geology*, 26(5):565–588, July 1994. pp. 93, 95

M. Alvarez, D. Luengo, and N. D. Lawrence. Latent force models. In D. van Dyk and M. Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, Apr. 2009. pp. 109, 127

Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering shared structures in multiclass classification. In Ghahramani [2007]. p. 45

C. H. An, C. G. Atkeson, and J. M. Hollerbach. *Model-based control of a robot manipulator*. MIT Press, Cambridge, MA, USA, 1988. pp. 99, 100, 101, 115

M. L. Anderson and T. Oates. A review on recent research in metareasoning and metalearning. *AI Magazine*, 28(1):7–16, 2007. pp. 1, 42, 48

R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. IBM Research Report RC23462 (W0412-022) Computer Science, IBM Research Division, Thomas J. Watson Research Center, Dec. 2004. pp. 3, 47

R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, Nov. 2005. pp. 6, 17, 18, 47, 48

B. Armstrong. On finding 'exciting' trajectories for identification experiments involving systems with non-linear dynamics. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 4, pages 1131–1139, 1987. p. 107

B. Armstrong, O. Khatib, and J. Burdick. The explicit dynamic model and inertial parameters of the PUMA 560 arm. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, pages 510–518, 1986. p. 99

C. T. H. Baker. *The Numerical Treatment of Integral Equations*. Clarendon Press, 1977. p. 58

B. Bakker and T. Heskes. Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research*, 4:83–99, May 2003. pp. 4, 8, 11, 13, 27, 28, 29

S. Barnet. *Matrices Methods and Applications*. Oxford Applied Mathematics and Computing Sciences Series. Clarendon Press, Oxford, 1990. p. 136

J. Baxter. A Model of Inductive Bias Learning. *Journal of Artificial Intelligence Research*, 12:149–198, Mar. 2000. pp. 2, 3, 4, 7, 51

J. Baxter. A Bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, 28(1):7–39, 1997. pp. 2, 3, 7, 9, 51

Y. K. Belyaev. Analytic random processes. *Theory of Probability and its Applications*, 4(4):402–409, 1959. p. 110

S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. In *Proceedings of the 16th Annual Conference on Learning Theory*, 2003. pp. 3, 7

S. Ben-David and R. Schuller Borbely. A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine Learning*, 73(3):273–287, 2008. pp. 3, 7, 51, 131

S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In Schölkopf et al. [2007], pages 137–144. p. 48

Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors. *Advances in Neural Information Processing Systems*, volume 22, Cambridge, MA, 2009. MIT Press. pp. 179, 182, 188

H. Bensusan. *Automatic Bias Learning: An Inquiry Into the Inductive Basis of Induction*. Ph.D. thesis, University of Sussex, School of Cognitive and Computing Sciences, 1999. pp. 5, 43

H. Bensusan and C. Giraud-Carrier. Discovering task neighbourhoods through landmark learning performances. In D. A. Zighed, J. Komorowski, and J. Żytkow, editors, *Principles of Data Mining and Knowledge Discovery*, volume 1910 of *Lecture Notes in Computer Science*, pages 325–330. Springer, September 2000. pp. 42, 43

H. Bensusan, C. Giraud-Carrier, and C. Kennedy. A higher-order approach to meta-learning. In *Proceedings of the ECML'2000 workshop on Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, pages 109–117. ECML'2000, June 2000. p. 43

P. R. Bevington and D. K. Robinson. *Data Reduction and Error Analysis for the Physical Sciences*. McGraw-Hill, 3rd edition, 2002. p. 89

C. M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006. p. 10

J. Blitzer, R. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia, 2006. p. 48

A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational on Learning Theory*, pages 92–100. Morgan Kaufmann, 1998. p. 47

E. V. Bonilla, F. V. Agakov, and C. K. I. Williams. Kernel multi-task learning using task-specific features. In Meila and Shen [2007]. pp. 4, 5, 6, 8, 34, 43

E. V. Bonilla, K. M. A. Chai, and C. K. I. Williams. Multi-task Gaussian process prediction. In Platt et al. [2008], pages 153–160. pp. 10, 31, 35, 54, 100, 116, 118, 126, 175

P. Boyle and M. Frean. Dependent Gaussian processes. In Saul et al. [2005], pages 217–224. p. 40

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996. p. 48

L. Breiman and J. H. Friedman. Predicting multivariate responses in multiple linear regression (with discussions). *Journal of the Royal Statistical Society. Series B (Methodological)*, 59(1):3–54, 1997. p. 44

J. W. Brewer. Kronecker products and matrix calculus in system theory. *IEEE Tansactions on Circuits and Systems*, 25(9):772–781, Sept. 1978. p. 156

J. S. Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 211–217. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990. p. 29

R. Brooks and M. Stone. Joint continuum regression for multiple predictands. *Journal of the American Statistical Association*, 89(428):1374–1377, Dec. 1994. p. 44

P. J. Brown and J. V. Zidek. Adaptive multivariate ridge regression. *The Annals of Statistics*, 8(1):64–74, Jan. 1980. p. 44

J. Buescu and A. C. Paixão. Eigenvalue distribution of positive definite kernels on unbounded domains. *Integral Equations and Operator Theory*, 57(1):19–41, Jan. 2007. p. 164

W. L. Buntine. Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, 2:159–225, 1994. p. 16

E. Burdet and A. Codourey. Evaluation of parametric and nonparametric nonlinear adaptive controllers. *Robotica*, 16(1):59–73, 1998. ISSN 0263-5747. pp. 100, 107

M. C. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In H. Burkhardt and B. Neumann, editors, *Computer Vision — ECCV'98*, volume 1407 of *Lecture Notes in Computer Science*, pages 628–641. Springer, June 1998. p. 6

A. J. Burnham, J. F. MacGregor, and R. Viveros. A statistical framework for multivariate latent variable regression methods based on maximum likelihood. *Journal of Chemometrics*, 13(1):49–65, 1999. p. 44

K. P. Burnham and D. R. Anderson. *Model Selection and Multi-Model Inference: A Practical Information-Theoretic Approach.* Springer Science+Business Media, New York, second edition, 2002. p. 119

R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997. pp. 4, 125

R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In Cohen and Moore [2006]. p. 42

C. Castiello, G. Castellano, and A. M. Fanelli. Meta-data: Characterization of input features for meta-learning. In V. Torra, Y. Narukawa, and S. Miyamoto, editors, *Modeling Decisions for Artificial Intelligence: Proceedings of the 2nd International Conference on Modeling Decisions for Artificial Intelligence (MDAI 2005)*, volume 3558 of *LNCS*, pages 457–468. Springer-Verlag Berlin Heidelberg, 2005. p. 43

K. M. A. Chai. Generalization errors and learning curves for regression with multi-task Gaussian processes. In Bengio et al. [2009], pages 279–287. p. 10

K. M. A. Chai, C. K. I. Williams, S. Klanke, and S. Vijayakumar. Multi-task Gaussian process learning of robot inverse dynamics. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21, pages 265–272. MIT Press, Cambridge, MA, 2009. p. 10

Y. S. Chan and H. T. Ng. Word sense disambiguation with distribution estimation. In *Proceedings of the International Joint Conference on Artificial Intelligience*, 2005. p. 48

N. V. Chawla and G. Karakoulas. Learning from labeled and unlabeled data: An empirical study across techniques and domains. *Journal of Artificial Intelligence Research*, 23:331–366, Mar. 2005. p. 47

J.-P. Chilès and P. Delfiner. *Geostatistics: Modeling Spatial Uncertainty.* John Wiley, New York, 1999. pp. 8, 9, 34, 93, 94, 95, 96, 130

M. K. Cılız and K. S. Narendra. Adaptive control of robotic manipulators using multiple models and switching. *International Journal of Robotics Research*, 15(6):592–610, 1996. p. 124

W. W. Cohen and A. Moore, editors. *Proceedings of the 23rd International Conference on Machine Learning*, June 2006. ACM. pp. 179, 180, 188

S. Conti and A. O'Hagan. Bayesian emulation of complex multi-output and dynamic computer models. *Journal of Statistical Planning and Inference*, 140(3):640–651, Mar. 2010. pp. 35, 126, 128

P. I. Corke. A robotics toolbox for MATLAB. *IEEE Robotics and Automation Magazine*, 3(1):24–32, 1996. p. 102

P. I. Corke. A symbolic and numeric procedure for manipulator rigid-body dynamic significance analysis and simplification. *Robotica*, 16:589–594, 1998. p. 109

P. I. Corke and B. Armstrong-Hélouvry. A search for consensus among model parameters reported for the PUMA 560 robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 1608–1613, 1994. pp. 99, 102

N. A. Cressie. *Statistics for Spatial Data*. Wiley, New York, 1993. p. 59

H. Daumé III. Bayesian multitask learning with latent hierarchies. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, June 2009. p. 48

H. Daumé III and D. Marcu. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research*, 26:101–126, May 2006. p. 48

A. P. Dawid. Some matrix-variate distribution theory: Notational considerations and a Bayesian application. *Biometrika*, 68(1):265–274, 1981. p. 16

B. J. de Kruif and T. J. A. de Vries. Support-vector-based least squares for learning non-linear dynamics. In *Proceedings of the 41st IEEE Conference on Decision and Control*, volume 2, pages 1343–1348, 2002. p. 114

P. J. Diggle and P. J. Ribeiro Jr. *Model-based Geostatistics*. Springer, New York, 2006. p. 33

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195: 216–222, 1987. p. 128

D. B. Dunson. Bayesian latent variable models for clustered mixed outcomes. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 62(2):355–366, 2000. p. 132

E. Eaton. Multi-resolution learning for knowledge transfer. In *Proceedings of the 21st National Conference on Artificial Intelligence*. AAAI Press, July 2006. p. 49

E. Eaton and M. desJardins. Knowledge transfer with a multiresolution ensemble of classifiers. In *Proceedings of the 23rd International Conference on Machine Learning: Workshop on Structural Knowledge Transfer for Machine Learning*, June 2006. p. 49

E. Eaton, M. desJardins, and J. Stevenson. Using multiresolution learning for transfer in image classification. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, 2007. p. 49

E. Eaton, M. desJardins, and T. Lane. Modeling transfer relationships between learning tasks for improved inductive transfer. In W. Daelemans, B. Goethals, and K. Morik, editors, *Proceedings of the 19th European Conference on Machine Learning and Knowledge Discovery in Databases*, volume 5211 of *Lecture Notes in Computer Science*, pages 317–332, Berlin, Heidelberg, 2008. Springer-Verlag. pp. 4, 5

S. Edelman. Representation, similarity, and the chorus of prototypes. *Minds and Machines*, 5(1):45–68, 1995. p. 4

T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50, 2000. pp. 10, 17, 21, 38

T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–537, Apr. 2005.
pp. 4, 8, 11, 14, 21, 22, 23, 26, 27, 28, 45, 130, 131, 135, 141, 142

R. Featherstone. *Robot Dynamics Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 1987. p. 99

T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2): 209–230, 1973. p. 19

G. Ferrari Trecate, C. K. I. Williams, and M. Opper. Finite-dimensional approximation of Gaussian processes. In Kearns et al. [1999], pages 218–224. pp. 65, 66, 68, 153

M. Fink, S. Shalev-Shwartz, Y. Singer, and S. Ullman. Online multiclass learning by interclass hypothesis sharing. In Cohen and Moore [2006]. p. 45

Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1996. p. 48

S. Geisser. Bayesian estimation in multivariate analysis. *The Annals of Mathematical Statistics*, 36(1): 150–159, 1965. p. 35

M. G. Genton. Separable approximations of space-time covariance matrices. *Environmetrics*, 18:681–695, 2007. Special Issue for METMA3 Workshop: Spatial and Spatio-temporal Modelling. p. 45

Z. Ghahramani, editor. *Proceedings of the 24th International Conference on Machine Learning*, 2007. Omni Press. pp. 177, 182, 183, 184, 188

J. Ghosn and Y. Bengio. Bias learning, knowledge sharing. *IEEE Transactions on Neural Networks*, 14: 748–765, July 2003. pp. 4, 6

C. G. Giraud-Carrier, R. Vilalta, and P. Brazdil. Introduction to the special issue on meta-learning. In *Special Issue on Meta-Learning* [Giraud-Carrier et al., 2004b]. pp. 1, 42, 48

C. G. Giraud-Carrier, R. Vilalta, and P. Brazdil, editors. *Special Issue on Meta-Learning*, volume 54(3) of *Machine Learning*. Springer Netherlands, 2004b. pp. 181, 182

H. Goldstein. *Multilevel Statistical Models*. Kendall's Library of Statistics. Edward Arnold, London, 3rd edition, May 2003. p. 46

H. Goldstein. Nonlinear multilevel models, with an application to discrete response data. *Biometrika*, 78(1):45–51, Mar. 1991. p. 46

P. Goovaerts. *Geostatistics for Natural Resources Evaluation*. Oxford Universiy Press, New York, 1997. pp. 93, 95

D. F. Gordon and M. desJardins. Evaluation and selection of biases in machine learning. *Special Issue on Bias Evaluation and Selection*, 20(1-2):5–22, 1995. pp. 1, 42

I. S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products*. Academic Press, seventh edition, 2007. Alan Jeffrey, and Daniel Zwillinger, editors. p. 166

K. Grauman and T. Darrell. Pyramid match kernels: Discriminative classification with sets of image features (version 2). Technical Report MIT-CSAIL-TR-2006-020, Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory, Mar. 2006. pp. 8, 49

R. V. Gueorguieva and A. Agresti. A correlated probit model for joint modeling of clustered binary and continuous responses. *Journal of the American Statistical Association*, 96:1102–1112, Sept. 2001. p. 132

E. Hadjidemetriou, M. D. Grossberg, and S. K. Nayar. Multiresolution histograms and their use for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):831–847, 2004. p. 49

Y. Haitovsky. On multivariate ridge regression. *Biometrika*, 74(3):563–570, Sept. 1987. p. 44

J. W. Hardin and J. M. Hilbe. *Generalized Linear Models and Extensions*. Stata Press, 2nd edition, Jan. 2007. p. 12

M. Haruno, D. M. Wolpert, and M. Kawato. MOSAIC model for sensorimotor learning and control. *Neural Computation*, 13(10):2201–2220, 2001. p. 125

D. L. Hawkins. Some practical problems in implementing a certain sieve estimator of the Gaussian mean function. *Communications in Statistics — Simulation and Computation*, 18(2):481–500, 1989. pp. 57, 157, 160

J. J. Heckman. Sample selection bias as a specification error. *Econometrica*, 47(1):153–162, Jan. 1979. p. 48

J. D. Helterbrand and N. Cressie. University cokriging under intrinsic coregionalization. *Mathematical Geology*, 26(2):205–226, Feb. 1994. p. 93

D. Higdon. Space and space-time modeling using process convolutions. In C. Anderson, V. Barnett, P. Chatwin, and A. El-Shaarawi, editors, *Quantitative Methods for Current Environmental Issues*, pages 37–56. Springer-Verlag, London, 2002. p. 40

D. Hsu, S. Kakade, J. Langford, and T. Zhang. Multi-label prediction via compressed sensing. In Bengio et al. [2009], pages 772–780. p. 46

C. M. Hurvich and C.-L. Tsai. Regression and time series model selection in small samples. *Biometrika*, 76(2):297–307, 1989. pp. 31, 119

K. Itô. The expected number of zeros of continuous stationary Gaussian processes. *Journal of Mathematics of Kyoto University*, pages 207–216, 1964. p. 72

R. A. Jacobs and M. I. Jordan. Learning piecewise control strategies in a modular neural network architecture. *IEEE Transactions on Systems, Man and Cybernetics*, 23(2):337–345, 1993. p. 125

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991. p. 29

A. Kalousis, J. Gama, and M. Hilario. On data and algorithms: Understanding inductive performance. In *Special Issue on Meta-Learning* [Giraud-Carrier et al., 2004b]. p. 44

R. E. Kass and A. E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430): 773–795, 1995. p. 5

M. J. Kearns. *Computational Complexity of Machine Learning*. ACM Distinguished Dissertation. MIT Press, 1990. p. 82

M. J. Kearns, S. A. Solla, and D. A. Cohn, editors. *Advances in Neural Information Processing Systems*, volume 11, Cambridge, MA, 1999. The MIT Press. pp. 180, 184

C. Kemp, A. Perfors, and J. B. Tenenbaum. Learning domain structures. In *Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society*, 2004. p. 46

M. C. Kennedy and A. O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 63(3):425–464, 2001. p. 111

G. S. Kimeldorf and G. Wahba. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, 41(2):495–502, 1970. p. 66

J. J. Kivinen, E. B. Sudderth, and M. I. Jordan. Learning multiscale representations of natural scenes using dirichlet processes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct. 2007. p. 49

N. D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 2005. p. 126

N. D. Lawrence and J. C. Platt. Learning to learn with the informative vector machine. In R. Greiner and D. Schuurmans, editors, *Proceedings of the 21st International Conference on Machine Learning*. ACM, July 2004. p. 34

M. Lázaro-Gredilla and A. Figueiras-Vidal. Inter-domain Gaussian processes for sparse inference using inducing features. In Bengio et al. [2009], pages 1087–1095. pp. 40, 41

S.-I. Lee, V. Chatalbashev, D. Vickrey, and D. Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In Ghahramani [2007]. p. 4

G. Lindner and R. Studer. AST: Support for algorithm selection with a CBR approach. In *Workshop Proceedings of the 16th International Conference on Machine Learning: Recent Advances in Meta Learning and Future Work*, June 1999. p. 43

D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming B*, 45(3):503–528, 1989. Software available from `http://www.ece.northwestern.edu/~nocedal/lbfgs.html`. pp. 88, 111

D. J. C. MacKay. *Information Theory, Inference, and Learning Algorithms.* Cambridge University Press, 2003. p. 14

D. J. C. MacKay. *Bayesian methods for adaptive models.* Ph.D. Thesis, California Institute of Technology, Pasadena, California, 1991. pp. 33, 39

C. Makkar, W. Dixon, W. Sawyer, and G. Hu. A new continuously differentiable friction model for control systems design. In *Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 600–605, 2005. p. 108

K. V. Mardia and R. J. Marshall. Maximum likelihood estimation of models for residual covariance in spatial regression. *Biometrika*, 71(1):135–146, 1984. p. 111

A. Maurer. Algorithmic stability and meta-learning. *Journal of Machine Learning Research*, 6:967–994, June 2005. p. 3

A. Maurer. Bounds for linear multi-task learning. *Journal of Machine Learning Research*, 7:117–139, Jan. 2006. pp. 3, 51

M. Meila and X. Shen, editors. *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, Mar. 2007. Omni Press. pp. 178, 183

U. Menzefricke. Hierarchical modeling with Gaussian processes. *Communications in Statistics – Simulation and Computation*, 29(4):1089–1108, 2000. p. 34

J. Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A*, 209:415–446, 1909. pp. 57, 155

C. A. Micchelli and M. Pontil. On learning vector-valued functions. *Neural Computation*, 17(1):177–204, 2005. p. 45

D. Michie, D. J. Spiegelhalter, and C. C. Taylor. *Machine Learning, Neural and Statistical Classification.* Ellis Horwood, 1994. pp. 42, 43

T. P. Minka and R. W. Picard. Learning how to learn is learning with point sets, 1997. URL `http://research.microsoft.com/~minka/papers/point-sets.html`. revised 1999. pp. 34, 175

T. M. Mitchell. The need for bias in learning generalizations. In T. Dietterich and J. Shavlik, editors, *Readings in Machine Learning*. Morgan Kaufmann, San Francisco, 1991. p. 1

P. Müller, F. A. Quintana, and G. L. Rosner. Semiparametric Bayesian inference for multilevel repeated measurement data. *Biometrics*, 63(1):280–289, 2007. p. 46

R. M. Neal. *Bayesian Learning for Neural Networks*. Springer-Verlag New York, 1996. pp. 39, 110

R. M. Neal. Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. Technical Report 9702, University of Toronto, 1997. pp. 110, 111

R. M. Neal. Priors for infinite networks. Technical Report CRG-TR-94-1, University of Toronto, Department of Computer Science, 1994. p. 39

J. Nelder and R. W. M. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384, 1972. p. 12

D. Nguyen-Tuong, J. Peters, and M. Seeger. Computed torque control with nonparametric regression models. In *Proceedings of the 2008 American Control Conference*, pages 1–6, 2008. p. 114

K. Ni, L. Carin, and D. Dunson. Multi-task learning for sequential data via iHMMs and the nested Dirichlet process. In Ghahramani [2007]. p. 19

A. Niculescu-Mizil and R. Caruana. Inductive transfer for Bayesian network structure learning. In Meila and Shen [2007]. p. 46

A. O'Hagan. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society. Series B (Methodological)*, 40(1):1–42, 1978. p. 109

A. O'Hagan. A Markov property for covariance structures. Statistics Research Report 98–13, Nottingham University, 1998. pp. 93, 95, 118

S. M. Omohundro. Family discovery. In Touretzky et al. [1996], pages 402–408. p. 6

M. Opper and F. Vivarelli. General bounds on Bayes errors for regression with Gaussian processes. In Kearns et al. [1999], pages 302–308. pp. 64, 76, 165

M. Opper and O. Winther. Gaussian processes and SVM: Mean field and leave-one-out. In A. J. Smola, P. J. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large-Margin Classifiers*, pages 311–326. MIT Press, Oct. 2000. p. 38

J. Ortega, M. Koppel, and S. Argamon. Arbitrating among competing classifiers using learned referees. *Knowledge and Information Systems*, 3(4):470–490, Nov. 2001. p. 48

S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, to appear. p. 48

Y. Peng, P. A. Flach, C. Soares, and P. Brazdil. Improved dataset characterisation for meta-learning. In S. Lange, K. Satoh, and C. H. Smith, editors, *Discovery Science*, volume 2534 of *Lecture Notes in Computer Science*, pages 141–152. Springer, 2002. p. 43

K. B. Petersen and M. S. Pedersen. *The Matrix Cookbook*. November 14, 2008 edition, 2008. URL `http://matrixcookbook.com`. p. 136

G. Petkos and S. Vijayakumar. Load estimation and control using learned dynamics models. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1527–1532, 2007. pp. 115, 124, 125

G. Petkos, M. Toussaint, and S. Vijayakumar. Learning multiple models of non-linear dynamics for control under varying contexts. In S. D. Kollias, A. Stafylopatis, W. Duch, and E. Oja, editors, *ICANN (1)*, volume 4131 of *Lecture Notes in Computer Science*, pages 898–907. Springer, 2006. pp. 100, 127

B. Pfahringer, H. Bensusan, and C. G. Giraud-Carrier. Meta-learning by landmarking various learning algorithms. In P. Langley, editor, *Proceedings of the 17th International Conference on Machine Learning*, pages 743–750. Morgan Kaufmann, June 2000. p. 42

J. Platt, D. Koller, Y. Singer, and S. Roweis, editors. *Advances in Neural Information Processing Systems*, volume 20, Cambridge, MA, 2008. MIT Press. pp. 178, 186, 187

S. Puntanen and G. P. H. Styan. Historical introduction: Issai Schur and the early development of the Schur complement. In F. Zhang, editor, *The Schur complement and its applications*, Numerical Methods and Algorithms, pages 1–16. Springer, 2005. p. 145

J. Quiñonero-Candela, C. E. Rasmussen, and C. K. I. Williams. Approximation methods for Gaussian process regression. In *Large Scale Kernel Machines*. MIT Press, 2007. pp. 8, 125

R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: Transfer learning from unlabelled data. In Ghahramani [2007]. p. 48

S. G. Raman and R. V. Rao. Eigenvalues of integral operators on $L_2(I)$ given by analytic kernels. *Integral Equations and Operator Theory*, 18(1):109–117, Mar. 1994. p. 164

C. E. Rasmussen. *Evaluation of Gaussian processes and other methods for non-linear regression*. PhD thesis, University of Toronto, Graduate Department of Computer Science, 1997. pp. 110, 111

C. E. Rasmussen and Z. Ghahramani. Occam's razor. In V. T. T. K. Leen, T. G. Diettrich, editor, *Advances in Neural Information Processing Systems*, volume 13. MIT Press, Cambridge, MA, 2001. p. 33

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, Massachusetts, 2006. pp. 8, 10, 11, 14, 29, 30, 38, 55, 56, 57, 58, 87, 88, 109, 110, 111, 112, 114, 128, 175

O. Richards and M. Baker. GridPP and the Edinburgh Compute and Data Facility or How a general purpose cluster bore the weight of Atlas on its shoulders. In *Proceedings of the UK All Hands Meeting*, 2008. p. 120

J. Rissanen, T. P. Speed, and B. Yu. Density estimation by stochastic complexity. *IEEE Transactions on Information Theory*, 38(2):315–323, Mar. 1992. p. 119

K. Ritter. *Average-Case Analysis of Numerical Problems*, volume 1733 of *Lecture Notes in Mathematics*. Springer, 2000. pp. 56, 58, 64, 66, 160, 164

K. Ritter. Almost optimal differentiation using noisy data. *Journal of Approximation Theory*, 86(3): 293–309, Sept. 1996. p. 64

K. Ritter, G. W. Wasilkowski, and H. Wozniakowski. Multivariate integration and approximation for random fields satisfying Sacks-Ylvisaker conditions. *The Annals of Applied Probability*, 5(2):518–540, 1995. p. 58

J. L. Rodgers and A. W. Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988. p. 69

A. Rodríguez, D. B. Dunson, and A. E. Gelfand. The nested Dirichlet process. *Journal of the American Statistical Association*, 103(483):1131–1154, 2008. p. 19

R. Rosipal and N. Krämer. Overview and recent advances in partial least squares. In C. Saunders, M. Grobelnik, S. Gunn, and J. Shawe-Taylor, editors, *Subspace, Latent Structure and Feature Selection Techniques*, volume 3940 of *Lecture Notes in Computer Science*, pages 34–51. Springer, 2006. Statistical and Optimization Perspectives Workshop, SLSFS 2005, Bohinj, Slovenia, February 23-25, 2005, Revised Selected Papers. p. 44

D. M. Roy and L. P. Kaelbling. Efficient Bayesian task-level transfer learning. In *International Joint Conference on Artificial Intelligience*, Jan. 2007. pp. 5, 19

H. Rue and L. Held. *Gaussian Markov Random Fields: Theory and Applications*, volume 104 of *Monographs on Statistics and Applied Probability*. Chapman & Hall, London, 2005. p. 96

J. Sacks and D. Ylvisaker. Designs for regression problems with correlated errors. *The Annals of Mathematical Statistics*, 37(1):66–89, 1966. p. 58

J. Sacks and D. Ylvisaker. Designs for regression problems with correlated errors: Many parameters. *The Annals of Mathematical Statistics*, 39(1):49–69, 1968. p. 58

J. Sacks and D. Ylvisaker. Designs for regression problems with correlated errors III. *The Annals of Mathematical Statistics*, 41(6):2057–2074, 1970. p. 58

L. K. Saul, Y. Weiss, and L. Bottou, editors. *Advances in Neural Information Processing Systems*, volume 17, Cambridge, MA, 2005. MIT Press. pp. 178, 185

S. Schaal, C. Atkeson, and S. Vijayakumar. Real-time robot learning with locally weighted statistical learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 288–293, 2000. p. 114

C. Schaffer. A conservation law for generalization performance. In *International Conference on Machine Learning*. Morgan Kaufmann, July 1994. p. 2

C. Schedlinski and M. Link. A survey of current inertia parameter identification methods. *Mechanical Systems and Signal Processing*, 15(1):189–211, 2001. pp. 108, 112

B. Schölkopf, J. Platt, and T. Hofmann, editors. *Advances in Neural Information Processing Systems*, volume 19, Cambridge, MA, 2007. MIT Press. pp. 178, 188

A. Schwaighofer, V. Tresp, and K. Yu. Learning Gaussian process kernels via hierarchical Bayes. In Saul et al. [2005]. p. 34

L. Sciavicco and B. Siciliano. *Modelling and Control of Robot Manipulators*. Advanced Textbooks in Control and Signal Processing. Springer-Verlag, London, 2nd edition, 2000. pp. 108, 115, 127

J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994. p. 19

R. Shitbata. Asymptotic mean efficiency of a selection of regression variables. *Annals of the Institute of Statistical Mathematics*, 35(1):415–423, December 1983. p. 119

R. Silva, W. Chu, and Z. Ghahramani. Hidden common cause relations in relational learning. In Platt et al. [2008], pages 1345–1352. p. 45

D. L. Silver. *Selective Transfer of Neural Network Task Knowledge*. Ph.D. thesis, University of Western Ontario, June 2000. pp. 4, 5, 6, 42

P. Sollich. Can Gaussian process regression be made robust against model mismatch? In N. D. L. J. Winkler and M. Niranjan, editors, *Deterministic and Statistical Methods in Machine Learning*, volume 3635 of *Lecture Notes in Computer Science*, pages 199–210. Springer-Verlag, Berlin / Heidelberg, Sept. 2005. p. 55

P. Sollich and A. Halees. Learning curves for Gaussian process regression: Approximations and bounds. *Neural Computation*, 14(6):1393–1428, 2002. pp. 51, 58, 64, 66, 72, 76, 160, 165, 166

A. Stein and L. C. A. Corsten. Universal kriging and cokriging as a regression procedure. *Biometrics*, 47(2):575–587, June 1991. p. 45

C. Stein. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In J. Neyman, editor, *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, volume 1: Contributions to the Theory of Statistics, pages 197–206. University of California Press, 1956. p. 3

M. L. Stein. *Interpolation of Spatial Data : Some Theory for Kriging*. Springer-Verlag, New York, 1999a. p. 114

M. L. Stein. Predicting random fields with increasing dense observations. *The Annals of Applied Probability*, 9(1):242–273, 1999b. p. 55

S. J. Stolfo, D. W. Fan, W. Lee, A. L. Prodromidis, and P. K. Chan. Credit card fraud detection using meta-learning: Issues and initial results. In *Workshop on AI Approaches to Fraud Detection and Risk Management, 14th National Conference on Artificial Intelligence*. AAAI Press, July 1997. p. 4

D. J. Stracuzzi. *Scalable Knowledge Acquistion through Cumulative Learning and Memory Organisation*. Ph.D. thesis, University of Massachusetts, Amherst, MA, Department of Computer Science, 2006. p. 46

C. Sutton and A. McCallum. Composition of conditional random fields for transfer learning. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 748–754, Vancouver, British Columbia, Canada, October 2005. Association for Computational Linguistics. p. 47

J. Swevers, C. Ganseman, D. Tukel, J. de Schutter, and H. Van Brussel. Optimal robot excitation and identification. *IEEE Transactions on Robotics and Automation*, 13(5):730–740, 1997. p. 107

Y. W. Teh, M. Seeger, and M. I. Jordan. Semiparametric latent factor models. In R. G. Cowell and Z. Ghahramani, editors, *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, pages 333–340. Society for Artificial Intelligence and Statistics, 2005. pp. 6, 10, 35, 36, 126

S. Thrun. A lifelong learning perspective for mobile robot control. In V. Graefe, editor, *IEEE/RSJ/GI Conference on Intelligent Robots and Systems*. Elsevier, 1995. p. 4

S. Thrun and T. M. Mitchell. Learning one more thing. In *International Joint Conference on Artificial Intelligience*, pages 1217–1223. Morgan Kaufman, Aug. 1995. pp. 2, 4

S. Thrun and J. O'Sullivan. Clustering learning tasks and the selective cross-task transfer of knowledge. In Thrun and Pratt [1998]. pp. 4, 5, 6

S. Thrun and L. Pratt, editors. *Learning to Learn*. Kluwer Academic Publishers, 1998. pp. 1, 2, 42, 125, 186

A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages II–762–II–769, Vol.2, 2004. p. 45

D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors. *Advances in Neural Information Processing Systems*, volume 8, 1996. The MIT Press. pp. 184, 187

G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3):1–13, 2007. p. 45

L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984. p. 2

T. van Erven, P. Grünwald, and S. D. Rooij. Catching up faster in Bayesian model selection and model averaging. In Platt et al. [2008]. p. 119

V. Vapnik. *Estimation of Dependences Based on Empirical Data: Empirical Inference Science, Afterword of 2006*. Information Science and Statistics. Springer, second edition, 2006. p. 9

J. M. Ver Hoef and R. P. Barry. Constructing and fitting models for cokriging and multivariable spatial prediction. *Journal of Statistical Planning and Inference*, 69(2):275–294, 1998. p. 40

S. Vijayakumar and S. Schaal. LWPR: An $O(n)$ algorithm for incremental real time learning in high dimensional space. In P. Langley, editor, *Proceedings of the 17th International Conference on Machine Learning*, pages 1079–1086. Morgan Kaufmann, 2000. p. 114

R. Vilalta and Y. Drissi. Research directions in meta-learning. In *Proceedings of the International Conference on Artificial Intelligence, (ICAI01)*, 2001. pp. 2, 48

R. Vilalta and Y. Drissi. A perspective view and survey of meta-learning. *Journal of Artificial Intelligence Review*, 18(2):77–95, 2002. pp. 1, 2, 42

R. Vilalta, C. G. Giraud-Carrier, and P. Brazdil. Meta-learning: Concepts and techniques. In O. Maimon and L. Rokach, editors, *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers*. Springer Publishers, 2005. p. 1

F. Vivarelli. *Studies on generalisation in Gaussian process and Bayesian neural networks*. Ph.D. Thesis, Aston University, Birmingham, United Kingdom, Neural Computing Research Group, 1998. pp. 55, 60

H. Wackernagel. *Multivariate Geostatistics: An Introduction with Applications*. Springer-Verlag, Berlin, second edition, 1998. pp. 8, 34, 45, 92, 93, 118, 126

T. J. Walsh, L. Li, and M. L. Littman. Transferring state abstractions between MDPs. In *Proceedings of the 23rd International Conference on Machine Learning: Workshop on Structural Knowledge Transfer for Machine Learning*, June 2006. p. 47

J. M. Wang, D. J. Fleet, and A. Hertzmann. Multifactor Gaussian process models for style-content separation. In Z. Ghahramani, editor, *Proceedings of the 24th Annual International Conference on Machine Learning*, pages 975–982. Omnipress, 2007. p. 126

H. Widom. Asymptotic behavior of the eigenvalues of certain integral equations. *Transactions of the American Mathematical Society*, 109(2):278–295, 1963. p. 165

C. K. I. Williams. Computation with infinite neural networks. *Neural Computation*, 10(5):1203–1216, 1998. p. 39

C. K. I. Williams and C. E. Rasmussen. Gaussian processes for regression. In Touretzky et al. [1996]. p. 110

C. K. I. Williams and F. Vivarelli. Upper and lower bounds on the learning curve for Gaussian processes. *Machine Learning*, 40(1):77–102, 2000. pp. 52, 58, 65, 76, 82

C. K. I. Williams, K. M. A. Chai, and E. V. Bonilla. A note on noise-free Gaussian process prediction with separable covariance functions and grid designs. Informatics Research Report 1228, University of Edinburgh, School of Informatics, Dec. 2007. pp. 93, 118

A. P. Wolfe and A. G. Barto. Defining object types and options using MDP homomorphisms. In *Proceedings of the 23rd International Conference on Machine Learning: Workshop on Structural Knowledge Transfer for Machine Learning*, June 2006. p. 47

D. H. Wolpert. The supervised learning no-free-lunch theorems. In *Proceedings of the 6th Online World Conference on Soft Computing*, 2001. p. 2

D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992. p. 48

D. M. Wolpert and M. Kawato. Multiple paired forward and inverse models for motor control. *Neural Networks*, 11(7-8):1317–1329, 1998. p. 125

W. Xu, T. T. Tran, and R. M. S. amd A. G. Journel. Integrating seismic data in reservoir modeling: The collocated cokriging alternative. In *67th Annual Technical Conference and Exhibition of the Society of Petroleum Engineers*, pages 833–842, 1992. SPE 24742. pp. 93, 95

Y. Xue, D. Dunson, and L. Carin. The matrix stick-breaking process for flexible multi-task learning. In Ghahramani [2007]. pp. 6, 17, 19, 20, 21

Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with Dirichlet process prior. *Journal of Machine Learning Research*, 8:35–63, Jan. 2007b. pp. 2, 5, 19

A. M. Yaglom. *Correlation theory of stationary and related random functions I: Basic results*. Springer Series in Statistics. Springer-Verlag, New York, 1987. p. 165

X. Yang, S. Kim, and E. Xing. Heterogeneous multitask learning with joint sparsity constraints. In Bengio et al. [2009]. p. 132

Y. Yang. Can the strengths of AIC and BIC be shared? A conflict between model indentification and regression estimation. *Biometrika*, 92(4):937–950, 2005. p. 119

N. D. Ylvisaker. The expected number of zeros of a stationary Gaussian process. *The Annals of Mathematical Statistics*, 36(3):1043–1046, 1965. p. 72

K. Yu, V. Tresp, and A. Schwaighofer. Learning Gaussian processes from multiple tasks. In L. D. Raedt and S. Wrobel, editors, *Proceedings of the 22nd International Conference on Machine Learning*. ACM, Aug. 2005. pp. 34, 35

K. Yu, W. Chu, S. Yu, V. Tresp, and Z. Xu. Stochastic relational models for discriminative link prediction. In Schölkopf et al. [2007]. pp. 4, 6, 10, 35

K. Yu, J. Lafferty, S. Zhu, and Y. Gong. Large-scale collaborative prediction using a non-parametric random effects model. In L. Bottou and M. Littman, editors, *Proceedings of the 26th International Conference on Machine Learning*, Montreal, June 2009. Omnipress. pp. 35, 40

S. Yu, K. Yu, V. Tresp, and H.-P. Kriegel. Collaborative ordinal regression. In Cohen and Moore [2006]. pp. 34, 35

A. Zellner. An efficient method of estimating seemingly unrelated regressions and tests for aggregation bias. *Journal of the American Statistical Association*, 57(298):348–368, June 1962. p. 44

H. Zhang. Maximum-likelihood estimation for multivariate spatial linear coregionalization models. *Environmetrics*, 18(2):125–139, 2007. pp. 45, 118

J. Zhang. *A Probabilistic Framework for Multi-Task Learning*. Ph.D thesis, Carnegie Mellon University, Language Technologies Institute, School of Computer Science, Aug. 2006. CMU-LTI-06-006. pp. 8, 11, 13, 15, 16, 17, 18, 28, 46

J. Zhang, Z. Ghahramani, and Y. Yang. Learning multiple related tasks using latent independent component analysis. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, pages 1585–1592, Cambridge, MA, 2006. MIT Press. p. 16

Y. Zhou and S. Goldman. Democratic co-learning. In *Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'04)*, pages 594–202, Washington, DC, USA, 2004. IEEE Computer Society. p. 47

Z.-H. Zhou and M. Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1529–1541, 2005. p. 47

H. Zhu, C. K. I. Williams, R. Rohwer, and M. Morciniec. Gaussian regression and optimal finite dimensional linear models. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, volume 168 of *NATO ASI Series F: Computer and Systems Sciences*, pages 167–184. Springer-Verlag, Berlin, 1998. pp. 57, 163

J. Zhu, J. C. Eickhoff, and P. Yan. Generalized linear latent variable models for repeated measures of spatially correlated multivariate data. *Biometrics*, 61(3):674–683, Sept. 2005. p. 45

X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, Sept. 2005. URL `http://pages.cs.wisc.edu/~jerryzhu/research/ssl/semireview.html`. Revised July 19 2008. p. 47

X. Zhu and A. B. Goldberg. *Introduction to Semi-Supervised Learning*, volume 6 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan & Claypool, 2009. p. 47