SMOOTH RELEVANCE VECTOR MACHINES

Alexander Schmolck

Submitted by Alexander Schmolck to the University of Exeter as a thesis for the degree of Doctor of Philosophy in Computer Science,

May 2008

I certify that all material in this thesis which is not my own work has been identified and that no material is included for which a degree has previously been conferred upon me.

Alexander Schmolck

Abstract

Regression tasks belong to the set of core problems faced in statistics and machine learning and promising approaches can often be generalized to also deal with classification, interpolation or denoising problems. Whereas the most widely used classical statistical techniques place severe a priori constraints on the type of function that can be approximated (e.g. only lines, in the case of linear regression), the successes of sparse kernel learners, such as the SVM (support vector machine) demonstrate that good results may be obtained in a quite general framework by enforcing sparsity. Similarly, even very simple sparsity-based denoising techniques, such as classical wavelet shrinkage, can produce surprisingly good results on a wide variety of different signals, because, unlike noise, most signals of practical interest share vital characteristics (such as *smoothness*, or the ability to be well approximated by piece-wise linear polynomials of a low order) that allow a sparse representation in wavelet space. On the other hand results obtained from SVMs (and classical wavelet-shrinkage) suffer from a certain lack of interpretability, since one cannot straightforwardly attach probabilities to them. By contrast regression, and even more importantly classification, in a Bayesian context always entails a probabilistic measure of confidence in the results, which, provided the model assumptions are reasonably accurate, forms a basis for principled decision-making. The relevance vector machine (RVM) combines these strengths by explicitly encoding the criterion of model sparsity as a (Bayesian) prior over the model weights and offers a single, unified paradigm to efficiently deal with regression as well as classification tasks. However the lack of an explicit prior structure over the weight variances means that the degree of sparsity is to a large extent controlled by the choice of kernel (and kernel parameters). This can lead to severe overfitting or oversmoothing – possibly even both at the same time (e.g. for the multiscale Doppler data). This thesis details an efficient scheme to control sparsity in Bayesian regression by incorporating a flexible noise-dependent smoothness prior into the RVM. The resultant smooth RVM (sRVM) encompasses the original RVM as a special case, but empirical results with a variety of popular data sets show that it can surpass RVM performance in terms of goodness of fit and achieved sparsity as well as computational performance in many cases. As the smoothness prior effectively makes it possible to use (highly efficient) wavelet kernels in an RVM setting this work also unveils a strong connection between Bayesian wavelet shrinkage and RVM regression and effectively further extends the applicability of the RVM to denoising tasks for up to millions of datapoints. We further discuss its applicability to classification tasks.

Contents

List of Figures

List of Tables

Acknowledgments

I wish to thank (in order of appearance): my parents, my long-suffering supervisor Richard Everson, as well as Mika Enslin and Michiru Sekiguchi. They all have, in different ways, supported me far more and for far longer than I could reasonably have asked (or they could have reasonably have wished) for and I want to express my heartfelt gratitude and debt to them.

Chapter 1

Introduction

Life is short and theses are many and mostly long. To do our part in optimizing the allocation of life and theses (parts), we will start out with an attempt to put the tentative reader into a position to quickly determine whether they bring the necessary background, if (some of) the material in this thesis holds interest to him or her and, if so, how to best extract it.

1.1 What, where, why

As to the required background: a good grounding in linear algebra (SVD, eigenvalue decomposition etc.) and multivariate calculus is essential but other than that this thesis tries to be mostly self-contained (we briefly review regression, Bayesian statistics, wavelet shrinkage, kernel methods and classification). Nonetheless some familiarity with regression and statistics/machine learning is undoubtedly beneficial.

To the extent the question of interest to the reader has not been settled by the abstract (even when combined with a peek ahead to the "novel contributions" on page 18): Our smooth relevance vector machine (sRVM) can be used for regression, classification and denoising, but we hope someone familiar with the field will be able to glean a good idea about the central aspects of this work from the very condensed overview of the sRVM from the perspective of denoising that can be found in the next section. We choose denoising (i.e. the task of removing noise-contamination from a signal) partly because it is more easily visualized and involves less conceptual and notational overhead than either regression or classification. We hope it will thus be more accessible for readers unfamiliar with sparse Bayesian learning. The other motivation for our choice of denoising is that the benefits of the smoothness prior for the RVM (which lies at the heart of this thesis) are particularly striking in this application. We then briefly sketch the transition to regression and classification and say a few words about the type and size of data sets that can be handled in each of these three domains, leaving the *why* of the (s)RVM – its main attractions – till after we have explained the *how*. It is our hope that this 10 000 feet overview in combination and figure 1.1 also provide a convenient and concise overview of the overall model structure and notation that can be referred back to as required. This, in combination with the cross-referenced Glossary and overview of notation on page 107, addresses the last of the aforementioned points by (hopefully) making it easier to browse and skim parts of the thesis.

1.2 The sRVM for denoising: a 10 000 feet overview

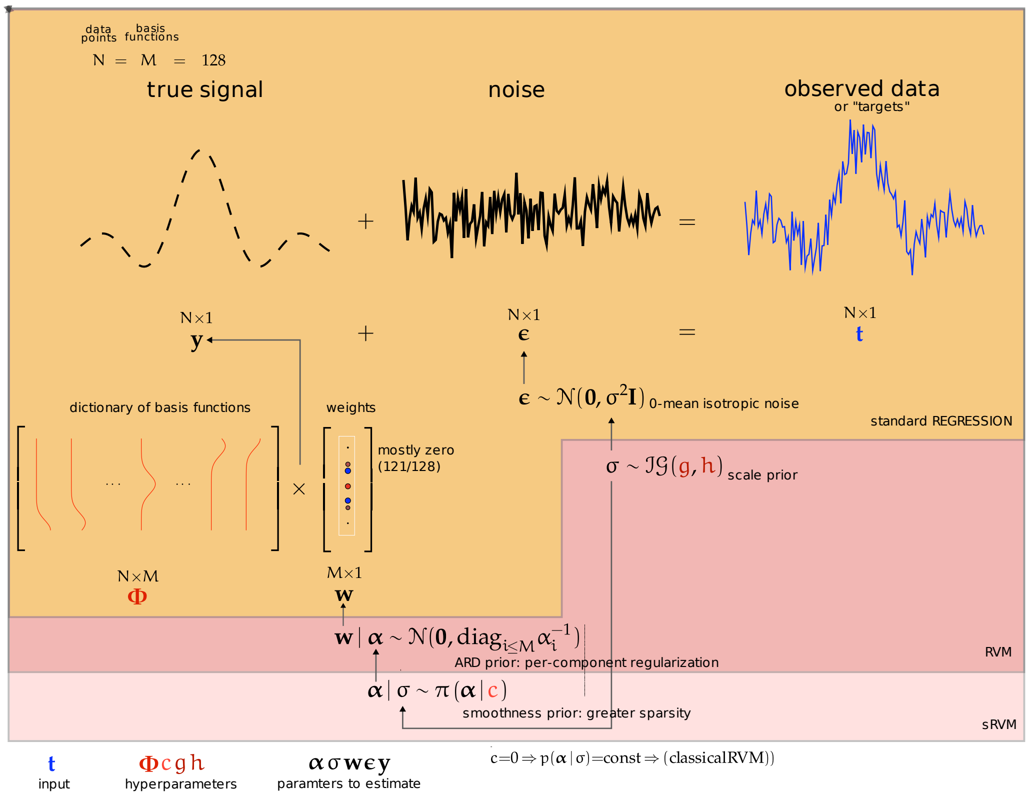

data points basis functions
N = M = 128
true signal
noise
observed data
or "targets"
+
=
N×1
y
+
N×1
ϵ
=
N×1
t
ϵ ∼ N(0, σ²I) 0-mean isotropic noise
standard REGRESSION
dictionary of basis functions
weights
mostly zero (121/128)
σ ∼ IG(g, h) scale prior
×
N×M
Φ
M×1
w
w | α ∼ N(0, diag_{i≤M} α_i^{-1})
ARD prior: per-component regularization
RVM
α | σ ∼ π(α | c)
smoothness prior: greater sparsity
sRVM
t
input
Φ c g h
hyperparameters
α σ w ϵ y
paramters to estimate
c=0 ⇒ p(α | σ)=const ⇒ (classical RVM))

Figure 1.1: 10 000 feet overview of the sRVM. The smoothness prior $\pi(\boldsymbol{\alpha}\,|\,c) \sim \exp(-c\mathrm{DF})$ (2.59) is what sets the sRVM apart from the RVM. It penalizes complex models (models with high degrees of freedom DF, where DF is defined in (2.57)) and can be adjusted in its severity by the user-determined hyperparameter c. Particular values for c can be related to classical model-choice criteria (see section 2.5.1).

Denoising postulates the following scenario: there exists an underlying signal of interest $\mathbf{y}$ which cannot be directly observed; what is actually observed are N values collected in a vector of targets $\mathbf{t}$, which is thought to equal $\mathbf{y}$ plus some, again unknown, added noise $\boldsymbol{\epsilon}$ – thus we have $\mathbf{t} = \mathbf{y} + \boldsymbol{\epsilon}$. In the simplest case, to which we will restrict ourselves, this noise will be zero-mean stationary Gaussian noise, i.e. $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma\mathbf{I})$.

The aim then, is to “undo” the process of noise pollution and infer an estimate of $\mathbf{y}$, $\hat{\mathbf{y}}$. This requires further constraints so one assumes that $\mathbf{y} = f(\mathbf{w})$, where f is a restricted function chosen a-priori and parametrized by a vector $\mathbf{w}$ that one seeks to find. In the simplest and most common case, $\mathbf{y}$ is taken to be a line parametrized by *intercept* w_0 and slope w_1 and the estimate $\hat{\mathbf{w}} = [\hat{w}_0 \quad \hat{w}_1]^T$ chosen to minimize $\|\mathbf{t} - f(\mathbf{w})\|$, which is equivalent to finding the maximum likelihood (ML) estimate $\hat{\mathbf{w}} = \mathrm{argmax}_{\mathbf{w}}\, p(\mathbf{t}\,|\,\mathbf{w})$. A simple generalization of plain linear regression that allows dealing with a much wider class of signals $\mathbf{y}$, whilst retaining linearity in $\mathbf{w}$, is to use a dictionary of basis functions $\boldsymbol{\Phi}$ and let $f(\mathbf{w}) = \boldsymbol{\Phi}\mathbf{w}$. Unfortunately, if $\boldsymbol{\Phi}$ is full-rank, the ML estimate just degrades to $\hat{\mathbf{y}} = \mathbf{t}$.

To overcome this problem, sparse regression further adds the assumption that the “correct” weight vector $\mathbf{w}$ will be mostly zero, because the signal of interest $\mathbf{y}$ belongs to a class of functions that can be sparsely represented in a given dictionary $\boldsymbol{\Phi}$ (so each w_i can be seen to correspond to a particular dictionary basis function or component $\boldsymbol{\phi}_i$ and most are expected to be zero as a result of the sparsity property). Many methods to enforce sparsity of $\mathbf{w}$ have been considered from within a variety of theoretical frameworks; SVMs within statistical learning theory (see section 2.2), LASSO and wavelet shrinkage approaches within frequentist statistics (see 2.1.3) and last but not least, the Relevance Vector Machine (Tipping (2001); shown in orange in Figure 1.1 and discussed in depth in section I) which has long held a prominent position within the (semi-)Bayesian framework. As can be seen in the diagram, the RVM extends the standard regression model by placing hierarchical priors over the noise deviation σ and, crucially, the weights $\mathbf{w}$ given $\boldsymbol{\alpha}$, where each individual α_i can be thought of as a regularization parameter for the corresponding w_i. This contrasts with well known classical approaches like ridge regression, where a single regularization parameter for all weights is used to combat overfitting – an approach that is evidently at odds with the assumption of sparse representability, as sparse representability implies that most components just carry noise (and should thus be turned off completely) whereas the signal is carried by a select few components (which should thus largely remain unpenalized). The hope with a one-parameter-per-weight prior structure therefore is that components can be assessed on their individual merits (and thus each either largely left alone or turned off completely). This prior structure, which aims to elucidate the relevance of individual weights has therefore been come to be known as automatic relevance determination (ARD) prior (MacKay, 1995). This model structure has been implemented by an efficient type II maximum likelihood scheme in Tipping's original RVM Tipping (2000), Tipping and Faul (2003).

Whilst this formulation of the RVM already delivers some measure of sparsity, empirical results demonstrate that the enforced level of sparsity is generally insufficient to use certain types of very flexible and computationally efficient basis functions for $\boldsymbol{\Phi}$, in particular wavelets (Mallat, 1999). We show that extending the RVM formulation by an additional, Bayesian wavelet-shrinkage[1] inspired, hyperprior on $\boldsymbol{\alpha}|\sigma$ yielding the smooth Relevance Vector Machine successfully addresses this problem.

This noise dependent prior is of the form $\exp(-c\mathrm{DF})$, where DF is an approximation of the degrees of freedoms of the smoothing matrix (and thus the amount of sparsity of the corresponding $\mathbf{w}$) that maps the targets $\mathbf{t}$ to the signal-estimate $\hat{\mathbf{y}}$ (see (2.55)).

1.3 The broader perspective: sRVM regression and classification

The denoising scenario can be generalized by no longer considering the signal as a fixed, finite length vector $\mathbf{y}$ but as a function over (possibly multidimensional) data points $\mathbf{x}_i$ so that $y_i = g(\mathbf{x}_i)$, for some unknown function g. The hope then is to predict, $y^\star = g(\mathbf{x}^\star)$ where $\mathbf{x}^\star$ is a novel data point for which the corresponding target $t^\star$ is unknown[2], from the previously encountered inputs and targets $\{(t_i, \mathbf{x}_i)\}_{i=1}^N$. However it is often notationally convenient to make the above functional dependence between x_i and y_i implicit by considering a fixed set of target, data-point tuples $\mathcal{D} = \{(t_i, \mathbf{x}_i)\}_{i=1}^N$ (the training data) and work with $\mathbf{y} = [g(\mathbf{x}_1) \quad \dots \quad g(\mathbf{x}_N)]^T$. Thus we will use the explicit functional form $y_i = g(\mathbf{x}_i)$ only when required (e.g. when considering novel predictions). Furthermore, it is often advantageous to transform the data points by a function $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$ that maps from D dimensional *input space* to M dimensional *feature space*[3], because an appropriate mapping will often allow us to reframe a problem that is difficult to deal with directly in data space into one that can be well modeled by a simple linear (in the weights) model. Again considering a fixed set of training data, we can just write

[1]See section 2.1.4.

[2]To be precise one, often wants to predict $t^\star$ and not $y^\star$, but since we concern ourselves with noise that is stationary – i.e. not a function of $\mathbf{x}_\star$ and zero-mean the difference is immaterial, since $\mathbf{y}^\star$ will be the best possible estimate for $\mathbf{t}^\star$

[3]In other words input space is simply the subspace of $\mathbb{R}^D$ in which we know every thinkable (un-preprocessed) input data point will lie. It is the domain of some pre-processing function of our choosing $\boldsymbol{\phi}$ whose image is the feature space, a subspace of $\mathbb{R}^M$.

$$\mathbf{y} = \begin{bmatrix} g(\mathbf{x}_1) \\ \vdots \\ g(\mathbf{x}_N) \end{bmatrix} = \begin{bmatrix} \boldsymbol{\phi}(\mathbf{x}_1)\mathbf{w} \\ \vdots \\ \boldsymbol{\phi}(\mathbf{x}_N)\mathbf{w} \end{bmatrix} \equiv \boldsymbol{\Phi}^{N\times M}\mathbf{w} = f(\mathbf{w}) \tag{1.1}$$

which is the form familiar from the denoising model.

In regression $\hat{\mathbf{w}}$ is often of more direct interest than it is in denoising, firstly because it tells us something about the extent by which different predictor dimensions relate to the observed signal[4] and secondly because it is precisely $\hat{\mathbf{w}}$ that allows us to make predictions for novel data points, which is what regression is all about. Again the Maximum Likelihood (ML) estimate will be unsatisfactory in many cases: when $\boldsymbol{\Phi}$ is full-rank, all $\hat{w}_m$ will be non-zero, making it hard to tell which dimensions in feature space are relevant and which merely carry noise. On the other hand with badly conditioned $\boldsymbol{\Phi}$, minute perturbations in $\mathbf{t}$ can result in dramatically different $\hat{\mathbf{w}}$. Apart from negatively affecting interpretability, the generalization ability from the training data to novel data points is of course also likely to be quite poor in these cases. A similar point applies to the classification scenario (which is essentially regression with discrete y, and requires further model additions) where generalization ability is of the essence. Sparsity enforcement (in a suitable feature space) can again offer a way to address both issues as the low number of dimensions singled out for prediction by corresponding non-zero w_i translates into better interpretability, robustness and generalization all at significantly lower computational cost when compared to ML estimation.

1.3.1 Kernel space vs feature space

A word of warning: rather than working in feature space (mapped to by a basis function $\mathbf{x} \to \boldsymbol{\phi}(\mathbf{x})$), it is also possible to work in kernel space (mapped to by a kernel function $(\mathbf{x}_a, \mathbf{x}_b) \to k(\mathbf{x}_a, \mathbf{x}_b)$ and again obtain a model that follows the same linear form (1.1), but the two are conceptually different: kernel functions are in fact defined by inner products of basis functions ($k(\mathbf{x}_j, \mathbf{x}) = \boldsymbol{\phi}(\mathbf{x}_j)^T\boldsymbol{\phi}(\mathbf{x})$) and there is a 1-to-1 mapping between the kernel- and feature space but the two spaces will typically have very different characteristics (e.g. a finite dimensional kernel space solution may often have an infinite dimensional feature space solution). To make things even more confusing $\boldsymbol{\Phi}$ can stand both for a kernel and a dictionary of basis functions in later chapters and the terms "kernel", "dictionary" and "design matrix" will be used loosely interchangeably. The reason for this is however simple: unlike the Support Vector Machine which is intrinsically tied to kernel space by its mode of operation, the RVM is essentially agnostic to the question of whether it works with a dictionary of basis functions or a kernel – once the input data has been transformed the difference becomes immaterial.

1.3.2 What makes the (s)RVM attractive?

That it offers a unified, flexible, efficient, effective and critically, probabilistic framework to deal with the main tasks found in supervised machine learning: regression and classification and, with qualifications the unsupervised task of signal denoising! The magic lies in the mix of the above, but, as we will argue, probabilistic interpretability is the key ingredient (and the one that puts the RVM ahead of the more well known SVM, whose outputs can only be interpreted probabilistically in the most tenuous fashion). In real life being (nearly or nearly always) right does not matter –

[4]Assuming standardized input data, a large positive (negative) value means that the given dimension is strongly correlated (anti-correlated) with the signal.

reaching (nearly or nearly always) the best decision does. The reason that, particularly in classification, even being right in say 99% of cases often is not sufficient to make good decisions is that in many, if not most, contexts the costs associated with different errors are highly asymmetrical. Indeed this is why some people use condoms.

Our extension which encompasses the original RVM puts RVM-based denoising firmly on the map and greatly adds to both flexibility and efficiency by allowing entirely new dictionary types to be used – importantly wavelets, but other orthogonal kernel types tailored for very sparse representations of a wide variety of signals should equally profit.

However, specifying (possibly hierarchical) priors over hyper-parameters and working with resultant joint and posterior distributions, as both RVM and sRVM do, is only half of what makes up a Bayesian approach. The reason for referring to the (s)RVM as (semi)-Bayesian a few paragraphs earlier, is that, like all methods discussed so far, it just delivers point-estimates for the parameters of interest – giving rise to an estimate $\hat{\mathbf{y}}$ that corresponds to one single model parametrization. But the other defining characteristic of a truly Bayesian approach is to reject finding a single "best" (e.g. most likely a posteriori) set of model parameters and to instead seek to discover the full posterior distributions. Consequently, in order to arrive at a fully Bayesian estimate for $\hat{\mathbf{y}}$ one would perform a weighted average over the predictions from all possible model parametrization, where the weighting is determined by the posterior pdf of the parameters. This approach properly accounts for the uncertainty in the parameters and thus allows for better estimates of the uncertainties involved in prediction and the ability to sample from the posterior also allows to address a wider range of questions than a simple point estimate. We therefore discuss a Markov Chain Monte Carlo (MCMC)[5] implementation attempt for the sRVM to embody such averaging in section 2.7.

1.4 Research Overview

The introductory section briefly reviews related work in frequentist sparse regression (LASSO and derivative schemes), their relation to wavelet shrinkage, and finally sparse kernel regression, most famously embodied by the SVM before introducing and reviewing Bayesian (and in particular RVM) regression.

The first part of the thesis examines our contribution to sparse Bayesian regression by providing a hierarchical extension to the RVM to increase result sparsity and demonstrating the benefits this increase entails. It also discusses our (not very successful) attempts at pursuing Metropolis-Hastings based MCMC sampling for the RVM as a less greedy and probabilistically more principled implementation strategy.

The second part is devoted to sparse classification. Adapting the fRVM scheme for classification to the smoothness prior is less straightforward than for regression, but we derive an efficient component-wise update formula and discuss its implementation and performance. We also explore an evolutionary optimization scheme that yields an approximately Pareto-optimal ROC curve of sparse kernel classifiers.

1.5 Novel Contributions

RVM sparsity control via prior structure rather than (just) kernel choice The RVM critically depends on propitious kernel choice in order to obtain good results as the sparsity of the result (and hence the degree of overfitting/oversmoothing) is largely controlled by the kernel. The

[5]MCMC algorithms allow one to sample from a wide variety of unwieldy probability distributions by constructing a Markov Chain that has the target distribution as its equilibrium distribution; skip to section 2.7.1 for a brief introduction.

sRVM effectively addresses the problem by dealing with sparsity control as part of the probabilistic model thus making a much wider choice of kernels available for a given problem. In particular it becomes feasible to obtain good results on a wide variety of signals with any one particular (flexible) kernel – something that would invariable result in overfitting or oversmoothing with the RVM.

Linking RVM regression and wavelet shrinkage As the effective sparsity control makes it feasible to use wavelet kernels (without drastic overfitting as in the classical RVM), the sRVM can also be thought of as a super-set of a certain form of Bayesian wavelet shrinkage (indeed the smoothness prior was first proposed in that context) – a connection that to our knowledge has not previously been made.

Proofs concerning the sparsity properties of the smoothness prior It is demonstrated that the smoothness prior is

1. scale invariant for constant signal-to-noise ratio
2. still gives rise to an unique local maximum under a Tipping and Faul (2003) style MAP scheme
3. always increases sparsity compared to the original RVM, unlike other choices for prior-augmentation (such as the Gamma prior).

Some possible causes for confusion are also cleared up. For example, although reading Tipping (2001) can convey the impression that a Gamma prior on α_i is used, in fact a uniform prior is employed both in the classical RVM (Tipping, 2000) and the fRVM (Tipping and Faul, 2003). Specifically, although Tipping (2001) discusses the attractions of the Jeffrey's prior (a degenerate Gamma), Appendix A.6 shows that it is impossible to use a Jeffrey's prior in conjunction with the Tipping and Faul (2003) MAP scheme.

An efficient scheme to find the stationary points of a non-orthogonal smooth w.r.t. a single component In classification problems the design matrix will essentially never be even approximately orthogonal and additional terms enter into the smooth **S**. It therefore becomes non-obvious how to isolate the contribution of a single component to the overall posterior $\hat{\mathcal{L}}$ in order to apply the fast RVM scheme. In Appendix A.8 we give a derivation that allows us to do just that.

Examining the effects of smoothness priors on classification performance Above scheme is utilized to explore the benefits of the smoothness prior in a classification setting.

Assessing the potential of MCMC-based RVM implementations Whilst fairly greedy MAP schemes such as Tipping and Faul (2003) can give good results on many tasks for a very affordable cost; they can suffer from three problems: 1. getting bogged down in a local minima (possibly a quite different one on each run) and 2. giving inaccurate estimates of the posterior distribution since the MAP solution, even if successfully identified can be a poor representative for a highly multimodal posterior and finally 3. the inability to incorporate additional criteria, such as risk minimization.

Given these shortcomings and the popularity of RVM and similar techniques, their amenability to a simple MCMC-based search space exploration is of some interest. We give empirical results concerning the properties of the search space and the results of Metropolis-Hastings based sampling.

Evolving sparse classifiers via TPR/FPR/Lasso[6] multi-objective optimization We give a brief description of a scheme to evolve a Pareto-optimal ROC curve of spare kernel classifiers.

Chapter 2

Background

2.1 The Mechanics of Regression

Regression: A method for fitting a curve (not necessarily a straight line) through a set of points using some goodness-of-fit criterion. – `http://mathworld.wolfram.com/Regression.html`

Regression is a natural starting point for both statistics and machine learning. The task is to reconstruct a signal of interest $\mathbf{y} = [y_1 \quad y_2 \quad \cdots y_N]^T$ given N (approximately) noise-free predictors x_n and corresponding, noisy observations $t_n = y_n + \epsilon_n$, with $(y_n, t_n, \epsilon_n) \in \mathbb{R}^3$. Typically $\boldsymbol{\epsilon} = [\epsilon_1 \cdots \epsilon_N]^T$ is taken to be zero-mean, stationary white noise with variance σ^2 – i.e. $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}_N)$, but for the moment I will put aside any probabilistic interpretation.

In the simplest case, and the one most frequently invoked in practice in statistics (and almost never in practical machine learning) one assumes $\mathbf{y}$ to be a straight line,

$$y_i = m_0 + m_1 x_i \tag{2.1}$$

and finds estimates $\hat{m}_o, \hat{m}_1$ (and thus indirectly $\hat{\mathbf{y}}$) that minimize:

$$E = \sum_{i=1}^{N} (t_i - m_0 - m_1 x_i)^2 \tag{2.2}$$

The approach naturally extends to an arbitrary number of dimensions D for the individual predictors $\mathbf{x}_i^{D\times 1}$. For D dimensional predictors (2.1) becomes

$$y_i = m_o + \sum_{d=1}^{D} m_d x_{id} = [\ 1 \quad \mathbf{x}_i^T\ ]\mathbf{m} \tag{2.3}$$

Changing fully to linear algebra notation, and, introducing the *design matrix* $\mathbf{X}$ in addition to the *weights* $\mathbf{m}$ we write...

$$\mathbf{X} \equiv \overbrace{\begin{bmatrix} 1 & x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \cdots & x_1^{(D)} \\ 1 & x_2^{(1)} & x_2^{(2)} & x_2^{(3)} & \cdots & x_2^{(D)} \\ \vdots & \vdots & \ddots & & & \\ 1 & x_N^{(1)} & x_N^{(2)} & x_N^{(3)} & \cdots & x_N^{(D)} \end{bmatrix}}^{\text{row = bias and D-dim. predictor}} = \left.\begin{bmatrix} 1 & \mathbf{x}_1^T \\ 1 & \mathbf{x}_2^T \\ \vdots & \vdots \\ 1 & \mathbf{x}_N^T \end{bmatrix}\right\} \text{N samples,} \tag{2.4}$$

$$\mathbf{m} \equiv \begin{bmatrix} m_0 & m_1 & m_2 & m_3 & \cdots & m_D \end{bmatrix}^T \tag{2.5}$$

$$\mathbf{y} = \mathbf{Xm} = \left.\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}\right\} \text{N predictions,} \tag{2.6}$$

$$\tag{2.7}$$

... to obtain a more concise rendition of (2.3). This formulation makes it also very obvious that E just represents the squared Euclidean distance between two vectors:

$$E = \|\mathbf{Xm} - \mathbf{t}\|_2^2 = \|\mathbf{y} - \mathbf{t}\|_2^2 = \|\varepsilon\|_2^2 \tag{2.8}$$

So clearly E is minimized by minimizing the distance between $\mathbf{y}$ and $\mathbf{t}$. Assume $\hat{\mathbf{y}}$ is the value for $\mathbf{y}$ that minimizes E, and that $\hat{\varepsilon}$ is the corresponding noise estimate. The above observation then implies that $\hat{\varepsilon}$ must be orthogonal to the column space of $\mathbf{X}$ in which $\hat{\mathbf{y}}$ lies. Therefore $\hat{\varepsilon}$ must lie in the null space of $\mathbf{X}^T$:

$$\mathbf{X}^T\hat{\varepsilon} = \mathbf{0} \tag{2.9}$$

$$\Rightarrow \qquad \mathbf{X}^T(\mathbf{X}\hat{\mathbf{m}} - \mathbf{t}) = \mathbf{0} \tag{2.10}$$

$$\Rightarrow \qquad \hat{\mathbf{m}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{t} = \mathbf{X}^\dagger\mathbf{t} \tag{2.11}$$

In other words,

$$\hat{\mathbf{y}} \equiv \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{t} \equiv \mathbf{Ht} \tag{2.12}$$

is the *orthogonal projection* of the N dimensional vector $\mathbf{t}$ into the $(D+1)$-dimensional subspace spanned by the columns of $\mathbf{X}^{N\times(D+1)}$ (see Figure 2.1 for a visualization with for $\mathbf{x} = [1 \quad 3 \quad 4]^T$ and $\mathbf{t} = [-1 \quad 2 \quad 2]^T$; naturally this result can also be obtained by differentiating (2.8)) wrt $\hat{\mathbf{m}}$. Written as a set of individual equations of m_i, (2.11) is known as the *normal equations*. $\mathbf{X}^\dagger$ is the *pseudo-inverse*[1] of $\mathbf{X}$ and is a generalization of the matrix-inverse to non-square matrices that satisfies the identity $\mathbf{XX}^\dagger\mathbf{X} = \mathbf{X}$ for any $N \times (D+1)$-matrix $\mathbf{X}$ and $\mathbf{H}$ is the *projection matrix* that projects $\mathbf{t}$ onto the subspace spanned by the columns of $\mathbf{X}$. Since $\mathbf{H}^n\mathbf{v} = \mathbf{Hv}$, for any $\mathbf{v}$, $\mathbf{H}$ is also refered to as a *idempotent matrix*. It can be seen that all eigenvalues λ_i of $\mathbf{H}$ must be 1 or 0, so that $\operatorname{tr}\mathbf{H} = \sum \lambda_i$ gives the dimensionality of the subspace $\mathbf{H}$ projects into.

Of course, most data of interest in a machine learning context are not well approximated by hyperplanes, so limiting oneself to $\mathbf{f}(\mathbf{X}, \mathbf{m}) = \mathbf{Xm}$ is too restrictive. On the other hand allowing

[1]More precisely: $\mathbf{X}^\dagger$ as defined in (2.11) is the *Moore-Penrose Pseudoinverse* for $\operatorname{rank}(\mathbf{X}) = \min(N, (D+1))$, there are other generalized inverses that satisfy $\mathbf{XX}^\dagger\mathbf{X} = \mathbf{X}$. The general form of the Moore-Penrose Pseudoinverse, defined for any matrix $\mathbf{X}$, can be expressed it in terms of the SVD of $\mathbf{X} = \mathbf{U}\operatorname{diag}(\boldsymbol{\sigma}, N \times (D+1))\mathbf{V}^T$, and yields both a numerically robust way to compute the pseudo-inverse as well as conceptual insight. For example writing $\mathbf{X}^\dagger = \mathbf{V}(\operatorname{diag}\boldsymbol{\sigma})^\dagger\mathbf{U}^T = \mathbf{V}\operatorname{diag}([\sigma_i^{-1} \text{ if } \sigma_i \neq 0 \text{ else } 0]_{1\leqslant i\leqslant \min(N,(D+1))}, N \times (D+1))\mathbf{U}^T$ shows that the Moore-Penrose Pseudoinverse solution will be quite badly behaved for ill-conditioned $\mathbf{X}$, i.e. $\mathbf{X}$ with some non-zero but very small σ_i. This is because small errors ϵ_i in $(\mathbf{U}^T\mathbf{t})_i = (\mathbf{U}^T(\mathbf{y} + \boldsymbol{\epsilon}))_i$ will be magnified by σ_i^{-1}. A possible remedy is truncated SVD, where σ_i below a certain threshold are simply zeroed.

near arbitrary $\mathbf{f}(\mathbf{X}, \mathbf{m})$ makes parameter estimation impossible, and although iterative minimization methods starting out from an initial guess, such as Levenberg-Marquardt (Press et al., 1992, ch. 15) can be used for a wide variety of $\mathbf{f}$, they will generally fail to converge unless the initial guess is sufficiently good.

The most popular compromise between expressivness and tractability is thus to choose a function that remains linear in the weights but apply a non-linear (possibly) dimension-changing transform $\boldsymbol{\phi} : \mathbb{R}^D \to \mathbb{R}^M$ to the predictors:

$$y_i \equiv f(\boldsymbol{\phi}(\mathbf{x}_i), \mathbf{w}) \equiv \boldsymbol{\phi}(\mathbf{x}_i)^T \mathbf{w} \tag{2.13}$$

which we can see to be directly analogous to (2.3), only that $[1 \quad \mathbf{x}_i] \in \mathbb{R}^{1\times(D+1)}$ has been replaced by $\boldsymbol{\phi}(\mathbf{x}_i)^T \in \mathbb{R}^{1\times M}$ and $\mathbf{m} \in \mathbb{R}^{D+1}$ by $\mathbf{w} \in \mathbb{R}^M$.

Since this model remains amenable to exactly the same (linear) solution procedure as (2.1), it is referred to as generalized linear regression; if we further write $\boldsymbol{\phi}(\mathbf{x}_i) \equiv [\phi_1(\mathbf{x}_i) \quad \phi_2(\mathbf{x}_i) \cdots \phi_M(\mathbf{x}_i)]^T$, Eq (2.4) becomes

$$\boldsymbol{\Phi} \equiv \overbrace{\left.\begin{bmatrix} \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \phi_3(\mathbf{x}_1) & \cdots & \phi_M(\mathbf{x}_1) \\ \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \phi_3(\mathbf{x}_2) & \cdots & \phi_M(\mathbf{x}_2) \\ \vdots & \ddots & & & \\ \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \phi_3(\mathbf{x}_N) & \cdots & \phi_M(\mathbf{x}_N) \end{bmatrix}\right\}}^{\text{M basis-functions, broken up into scalar components } \phi_m : \mathbb{R}^D \to \mathbb{R}} \quad \text{N-samples} \tag{2.14}$$

$$\mathbf{y} = \boldsymbol{\Phi}\mathbf{w} \tag{2.15}$$

The most obvious, if not necessarily attractive, extension is fitting a polynomial, for the univariate case this is achieved[2] by setting $\phi_m(x_n) = x_n^{m-1}$ which for vector-valued $\mathbf{x}_n$ becomes $\boldsymbol{\phi}_m(\mathbf{x}_n) = [(x_n^{(1)})^{m-1} \cdots (x_n^{(D)})^{m-1}]^T$. A corollary is that the form above implicitly entails an optional leading $\mathbf{1}$-column for the bias, as one can always set $\boldsymbol{\phi}_1(\mathbf{x}) = 1$. There is no reason the number of basis functions, M, would need to equal the dimension of an individual predictor, D, and indeed one of the main uses of basis functions is to transform the input data into a higher dimensional feature space where it can be tackled with linear methods. Since $\mathbf{X}$ essentially disappears as an entity of its own right once $\boldsymbol{\Phi}^{N\times M}$ (which represents it indirectly) is defined, we will mostly concern ourselves with the latter from now on.

2.1.1 Regularization

Unfortunately, the generalization of plain linear regression also introduces a new potential problem: overfitting. Specifically if, $M = N$ and $\boldsymbol{\Phi}$ is not rank-deficient, $\hat{\mathbf{y}} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\dagger\mathbf{t} = \boldsymbol{\Phi}\boldsymbol{\Phi}^{-1}\mathbf{t} = \mathbf{t}$, i.e. the regression model always just returns the targets as predictions, which is clearly not desired, indeed as fig. 2.1.1 demonstrates, overfitting is even likely to occur well before $M = N$.

[2]This form of basis function runs into numerical difficulties very quickly as M increases, but much greater robustness can be achieved by using orthonormal polynomials, such as the Laguerre polynomials, as basis functions.

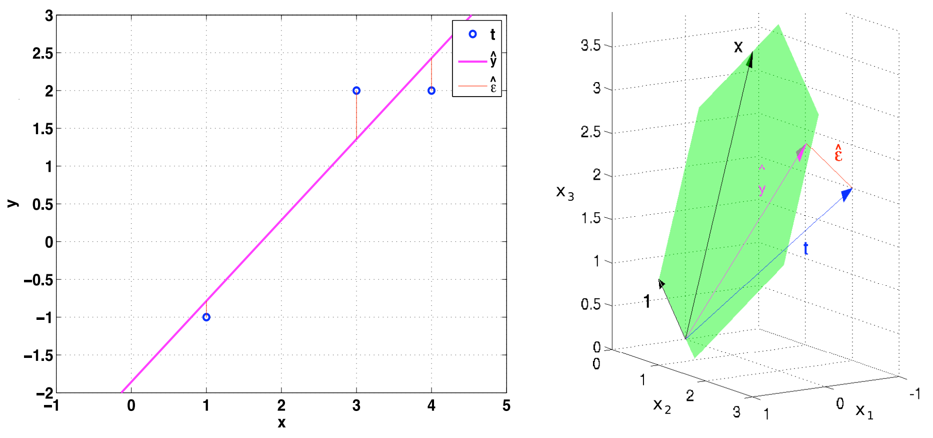


Figure 2.1: Two ways to visualize the least squares fit $\hat{\mathbf{y}}$ for the inputs $\mathbf{x} = [1\ 3\ 4]$ and the targets $\mathbf{t} = [-1\ 2\ 2]^T$. *Left panel:* The familiar form in which the input/target pairs (x_i, t_i) (represented as blue dots) are taken to be points in 2-dimensional space and $\hat{y}_i$ is chosen so that the pairs $(x_i, \hat{y}_i)$ lie on the line (shown in magenta) the minimizes the square of the vertical offset ϵ_i (shown in red) between (x_i, t_i) and the corresponding point on the line at x_i. *Right panel:* The equivalent, but less familiar, linear algebra interpretation. Here the inputs $\mathbf{x}$ are represented by a single vector in $(N = 3)$-dimensional space, as are the targets $\mathbf{t}$, the estimate $\hat{\mathbf{y}}$ and the noise estimate $\hat{\varepsilon}$. The least squares estimate $\hat{\mathbf{y}}$ is the orthogonal projection of $\mathbf{t}$ into the $(D = 2)$-dimensional subspace (shown in green) spanned by $\mathbf{x}$ and the bias vector $\mathbf{1} = [1 \quad 1 \quad 1]^T$; the noise estimate $\hat{\varepsilon}$ is the distance between $\hat{\mathbf{y}}$ and $\mathbf{t}$.

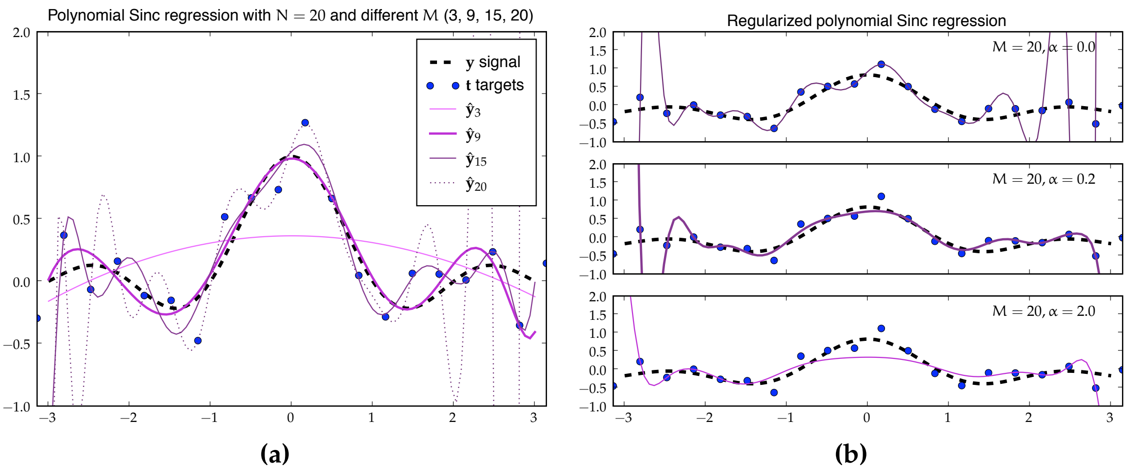


Figure 2.2: **(a)** Sinc regression with polynomials of different degree M. Overfitting starts to occur well before $M = N = 20$. On the other hand $M = 3$ clearly lacks the flexibility required to fit the data and oversmoothes. $M = 9$ gives the best fit. **(b)** Same as pannel on the left with $M = 20$, but increasing level of regularization from top ($\alpha = 0$) to bottom ($\alpha = 2$). The middle panel ($\alpha = 0.23$) gives the lowest error.

Rather than pruning one's design matrix $\mathbf{\Phi}$ in an extremely careful fashion a-priori to make sure it offers just enough flexibility to fit the underlying signal but not the noise (not easy – there are 2^M possible subsets of the M basis functions – and although there exists a clever scheme ("Leaps and Bounds"; Furnival and Robert W. Wilson, 1974) to find the best subset of size S, in terms of MSE for the quadratic error function, even it can only deal with M upto around 40 (Hastie et al., 2001)), a more promising and fine-grained approach involves additionally constraining the weight vector $\mathbf{w}$, which can be justified by observing that overfitting often is associated with extremely large weights.

One way to accomplish this is to introduce and additional weight penalty term $\alpha E_{\mathbf{w}} = \|\mathbf{w}\|_q$ to the Error function E, so that the total error now consists of a weighted sum of the residual E_D

and the a complexity penalty $E_{\mathbf{w}}$:

$$E = E_D + \alpha E_{\mathbf{w}} = \|\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}\|_2^2 + \alpha\|\mathbf{w}\|_q \tag{2.16}$$

$\|\mathbf{w}\|_q \equiv \sqrt[q]{\sum_m^M |w_i|^q}$ denotes the q-norm, with $q = 2$, the familiar Euclidian (or ℓ_2) norm, being the most common choice and thus the one we will therefore consider first. Taking $q = 2$ and setting the derivative of E w.r.t. $\mathbf{w}$ to 0 yields the estimate:

$$\hat{\mathbf{w}} = (\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \alpha\mathbf{I})^{-1}\boldsymbol{\Phi}^T \tag{2.17}$$

This approach is known as ridge-regression (Hoerl and Kennard, 1970).

It is clear[3] that the condition number of $(\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \alpha\mathbf{I})^{-1}$ goes to 1 as $\alpha \to \infty$, which explains why this method was first used in the context of numerically solving badly behaved approximate equations, where it is known as *Tikhonov regularization*(Tikhonov, 1943)[4].

Further insight into the role of α may again be gained via SVD by substituting $\boldsymbol{\Phi} = \mathbf{U}\,\mathrm{diag}(\boldsymbol{\sigma})\mathbf{V}^T$ and $\alpha\mathbf{I} = \mathbf{V}\alpha\mathbf{I}\mathbf{V}^T$ into (2.17) and simplifying to obtain $\hat{\mathbf{w}}_{RR} = \mathbf{V}\,\mathrm{diag}([\frac{\sigma_i}{\sigma_i^2+\alpha}]_i)\mathbf{U}^T\mathbf{t}$, which can be seen to be related to the pseudo-inverse weights $\hat{\mathbf{w}}_{LS} = \boldsymbol{\Phi}^\dagger\mathbf{t} = \mathbf{V}\,\mathrm{diag}([\frac{1}{\sigma_i}]_i)\mathbf{U}^T\mathbf{t}$ by a scaling of the diagonal by $\frac{\sigma_i^2}{\sigma_i^2+\alpha}_i$, for $\sigma_i n \neq 0$. Since $0 < \frac{\sigma_i^2}{\sigma_i^2+\alpha}_i < 1$, the ridge-regression weights are seen to be *"shrunk"* versions of the least squares weights, i.e. $\|\hat{\mathbf{w}}_{RR}\| \leqslant \|\hat{\mathbf{w}}_{LS}\|$ which places ridge regression amongst the class *shrinkage methods*. Specifically, $\mathbf{t}$ can be seen to be shrunk in principal component space, with decreasing amount of shrinkage in the direction of the more important principal components (those with large σ_i^2).

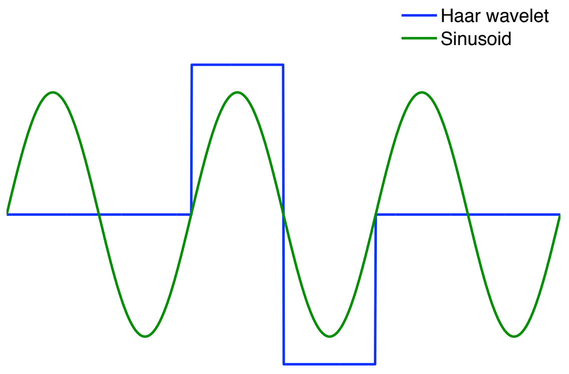


Figure 2.3: Haar wavelet vs sinusoid. The Haar mother wavelet is given by the simple function $\phi_H(x) = -1$ if $0 \leqslant x < 1/2$ else 1 if $1/2 \leqslant x < 1$ else 0. Whilst a sinusoid is continuous and periodic, the Haar wavelet has finite support and is discontinuous. It is however possible to construct an orthogonal basis from shifted and scaled copies of either of these two functions, $\phi_H(x)$ or $\cos(x)$. Because the resulting bases are orthogonal, a signal expressed in either basis can be transferred into the other basis by means of a simple rotation (effected by pre-multiplying the signal with a matrix that spans the respective basis).

So $q = 2$ can clearly addresses the ill-conditioning problem mentioned above – what about our desire for sparsity? One reason to be suspicious about the Euclidian norms' sparsity enforc-

[3]Intuitive, non-rigorous argument: as α becomes very large, the $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$ term becomes negligible and any scalar multiple of the identify matrix has condition number 1. Rigorous argument: the Levy-Desplanques theorem applies since the matrix $(\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \alpha\mathbf{I})$ is diagonally dominant for large enough values of α (i.e. $\forall i : |(\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \alpha\mathbf{I})_{ii}| > \sum_{j\neq i}(\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \alpha\mathbf{I})_{ij}$).

[4]Less obvious, but still interesting: the Moore-Penrose pseudoinverse can equally be defined as the limit of Tikhonov regularization with $\alpha \to 0$.

ing credentials is the fact that it is obviously invariant under rotation. From elementary linear algebra, a change of basis from one space spanned by an orthogonal matrix to another is equivalent to a rotation of the coordinate axis and is simply effected by pre-multiplying the signal to be transformed by the orthogonal matrix that spans the desired space; just as the reverse transform (and inverse rotation) is done by pre-multiplying with the transpose of that matrix.

This means that for example solving with orthogonal Haar basis functions and rotating into a space spanned by orthogonal sinusoids (see Figure 2.3 for a comparison and further explanation) will give exactly the same result as working in sinusoid space all along; even if the signal of interest is for example a three-note chord – intuitively an ideal match for sinusoids and a much worse match for Haar wavelets!

Whilst ridge regression has the convenient property that the amount of shrinking can be easily quantified by α, it has some undesirable properties. Firstly ridge regression is not *scale-invariant* – multiplying a basis function $\boldsymbol{\phi}_i$ by scalar s does not just result in the same weight solution $\hat{w}_i$ only scaled by the same s – however this problem is typically addressed by standardizing the data (Hastie et al., 2001). Secondly, and more problematically, ridge regression penalizes relevant and irrelevant basis functions indiscriminately. This conflicts with the assumption of sparse representability. Thirdly, and lastly, empirically ridge regression indeed generally turns out not to be sufficiently sparsity enforcing. This accords with intuition, considering that a sparsity enforcing prior would need to penalize spreading a fixed amount of "weight-mass" over different w_i whereas $q = 2$ actually encourages such spreading. This becomes quite obvious when considering the extreme case where the design matrix consists of two identical columns $\boldsymbol{\Phi} = [\boldsymbol{\phi} \quad \boldsymbol{\phi}]$ so that for any choice of $\hat{\mathbf{w}} = [tc \quad (1-t)c]^T$ with c fixed and any choice of $t \in [0, 1]$ we obtain the same $\hat{\mathbf{y}} = \boldsymbol{\Phi}\hat{\mathbf{w}}$. Obviously, with sparsity in mind, we would desire either the solution with $t = 0$ or with $t = 1$. Ridge regression, however, gives us the sparsity-wise pessimal solution $t = 1/2$ (as this minimizes $\|\hat{\mathbf{w}}\|_2 = \|[tc \quad (1-t)c]\|_2$, which can be seen by setting $\frac{d\|\mathbf{w}\|_2}{dt} = 2c^2t - 2c^2(1-t) = 0$ and solving for t), completely obscuring the redundancy of the two columns[5].

[5]Although these intuitive arguments are sufficient to see the problems with ridge regression in a sparse regression context, it is also possible to show that ridge regression cannot profit from smoothness in $\mathbf{y}$ beyond $\mathbf{y}$ being twice differentiable, see e.g. Neumaier (1998).

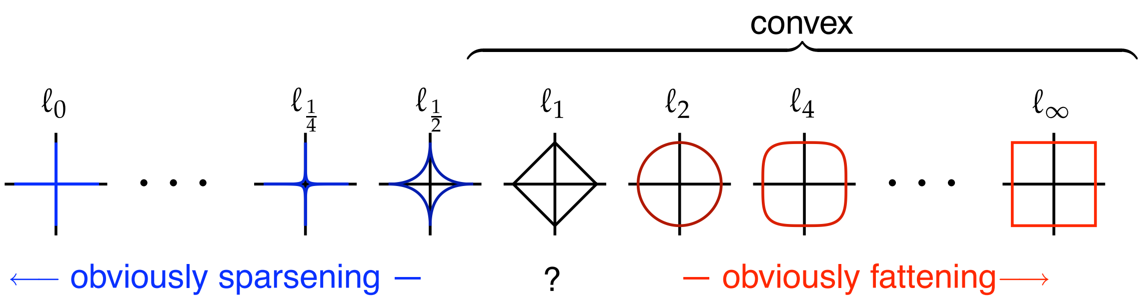


Figure 2.4: The effect of different ℓ_q-norms over $\mathbf{w}$ on sparsity, visualized with isocontour plots for $\mathbf{w} = [w_1 \quad w_2]$ and $q = 2^{-\infty}, 2^{-2}, 2^{-1}, 2^1, 2^2, 2^\infty$. Three values of q are of particular interest: ℓ_2 is the familiar Euclidian norm (and can be seen to be the only rotationally invariant norm), ℓ_1 is also known as Manhattan distance (and from $\ell_q(\mathbf{w}) = \|\mathbf{w}\|_q = \sqrt[q]{\sum_i |w_i|^q}$ it is easy to see that it is the only norm that is invariant under reallocation of total weight mass $m_w = \sum_i^M |w_i|$). Finally ℓ_0, is a pseudo-norm that corresponds to the number of selected basis functions $\boldsymbol{\phi}_i$ (i.e. nonzero w_i). $q = 1$ is a natural middle point – smaller values of q favour concentrating the absolute total weight-mass in a single w_i, thus encouraging sparsity whilst larger values favor spreading it equally over all w_i, thus encouraging "fatness" or inverse-sparsity. Note that ℓ_1 takes a special place as the lowest (and only integer-valued-q) ℓ-norm that is still convex (which can considerably ease optimization) but not obviously fattening. Actually, although ℓ_1 is invariant w.r.t. to the allocation of total weight mass, it still turns out to be sparsening because when used as a weight penalty term in (2.16) it will generally have the effect of setting several $\hat{w}_i$ to exactly 0 (*see text for an explanation*).

It is thus natural to consider other values for q (see Figure 2.4), in particular $q = 1$, as it is the most sparsity enforcing value that still gives rise to *convex* isolines which considerably eases the process of devising an efficient optimization scheme (more on this aspect shortly).

Indeed $q = 1$ is not just somewhat more sparsity enforcing than $q = 2$, it is vastly more likely to produce some w_i which are exactly zero. One way to look at this is by noting that optimizing (2.16) is equivalent to Lagrange optimization with the constraint term $\|\mathbf{w}\|_q$. The optimum has to be either located at a discontinuity in $E_\mathbf{w}$ or at a point where the gradient of this term is parallel to E_D, because otherwise a tiny move along in one direction of the isoline of the constraint surface would further reduce E_D. But with the discontinuities for $q \leqslant 1$ coinciding with $w_i = 0$, we have additional "edges" which afford numerous opportunities for E_D to get "caught on". We can make this intuition more precise – at the optimum we have (for values of w_i for which the partial derivative $\frac{\partial E_\mathbf{w}}{\partial w_i}$ on the RHS is defined):

$$\frac{\partial E}{\partial w_i} = 0 \Rightarrow \frac{\partial E_D}{\partial w_i} = -\alpha \frac{\partial E_\mathbf{w}}{\partial w_i} \tag{2.18}$$

$$= -\alpha \operatorname{sign}(w_i)|w_i|^{q-1} \times \left(\sum_m |w_m|^q\right)^{1/q-1} \tag{2.19}$$

$$= \begin{cases} -\alpha w_i \times \|\mathbf{w}\|_2^{-1} & q = 2 \\ -\alpha \operatorname{sign}(w_i) \times 1 & q = 1 \end{cases} \tag{2.20}$$

For $q = 2$ the partial derivative $\frac{\partial E_\mathbf{w}}{\partial w_i}$ is defined everywhere apart from $\mathbf{w} = \mathbf{0}$ and thus the optimal w_i will be zero if and only if $\frac{\partial E_D}{\partial w_i}|_{w_i}$ is already zero – in other words adding a $q = 2$ regularization term for a *single* w_i has little effect on sparsity as measured in terms of zero components! By contrast, above equation has no solutions in the $q = 1$ case, if $|\frac{\partial E_D}{\partial w_i}| \neq \alpha$ unless $w_i = 0$ which is the only value for which above conditions do not apply. Now let us consider what happens if we apply a penalty term to all weights, rather than just a single $\mathbf{w}_i$. Although $\frac{\partial}{\partial w_i} E_D = \frac{\partial}{\partial w_i} \frac{1}{2} \|\boldsymbol{\Phi}\mathbf{w} - \mathbf{t}\|^2 = \boldsymbol{\phi}_i^T(\boldsymbol{\Phi}\mathbf{w} - \mathbf{t})$ depends not just on w_i but all w_m (for $1 \leqslant m \leqslant M$), a little reflection shows that above conclusion still stands: w_i can only be zero if $\boldsymbol{\phi}_i^T(\boldsymbol{\Phi}\mathbf{w} - \mathbf{t}) = 0$. However, for this to happen either $\boldsymbol{\phi}_i^T$ has to be orthogonal to the residual $(\boldsymbol{\Phi}\mathbf{w} - \mathbf{t})$ and thus redundant, or the residual itself has to be $\mathbf{0}$. For the former to be true for multiple $\boldsymbol{\phi}_i$ without the latter holding, all $\boldsymbol{\phi}_i$ have to be scalar multiples of each other because $\boldsymbol{\Phi}\mathbf{w} - \mathbf{t}$ uniquely determines their direction. The latter case in turn can only occur when $\mathbf{t} = \mathbf{0}$. Otherwise, there would have to be some nonzero w_i so that $\boldsymbol{\Phi}\mathbf{w} = \mathbf{t}$. However for each of these w_i, we would have, for $q = 2$:

$$w_i = \frac{\partial}{\partial w_i} E_{\mathbf{w}} = -\alpha^{-1} \frac{\partial}{\partial w_i} E_D = -\alpha^{-1} \boldsymbol{\phi}_i^T (\boldsymbol{\Phi}\mathbf{w} - \mathbf{t}) = -\alpha^{-1} \boldsymbol{\phi}_i^T \mathbf{0} \tag{2.21}$$

$$= 0 \qquad \text{Contradiction!} \tag{2.22}$$

Thus the above discussion should make it clear that we can expect $q = 1$ to give much sparser results than $q > 1$ (see also Tibshirani (1996)).

Back to the attractiveness, as far as ease of implementation of an effective optimization scheme is concerned, of choosing a value of q which gives rise to a convex E_D.

This is partly because convex optimization is such a well established field, see e.g. (Boyd and Vandenberghe, 2004), with a huge body of knowledge and widely available high-quality off-the-shelf software solutions; but one property that will be seen to be particularly relevant in our context is that E as a weighted sum of two convex error functions (E_D and $E_{\hat{\mathbf{w}}}$) is again convex and thus has a unique minimum. In a statistical context this would be equivalent to noting that the combination of a log-concave likelihood and a log-concave prior gives a log-concave posterior and therefore a unique MAP (maximum a posteriori) solution(Krishnapuram et al., 2004). Furthermore it turns out the the ℓ_1 norm has even more attractive sparsity enforcing properties – for example it can be shown that for several types of dictionaries, including overcomplete ones, ℓ_1 minimization is equivialent to ℓ_0 minimziation and thus offers an effective way to select the optimal subset in certain problem domains (Donoho and Elad, 2002). Also, Ng (2004) shows that in logistic regression ℓ_1 can cope with exponentially as many training samples as irrelevant features, in contrast to ℓ_2 regularization (and all other rotationally invariant regularizations) where the relationship is at least linear.

Before moving on to discuss LASSO, the most widely used method of the form (2.16) with $q = 1$, it is time to say a few words about regression and regularization from a probabilistic perspective, or rather from *different* probabilistic perspectives.

2.1.2 Regularization from a probabilistic perspective

Prior to considering regularization within the context of proper probabilistic models, it is illustrative to discuss an approach that can be used to determine regularization parameters even in the absence of such models – namely *cross-validation* (consult Hastie et al., 2001, for a more detailed introduction to cross-validation from a frequentist perspective).

Cross Validation

In k-fold cross-validation, the available data is partitioned into k sets and an average error rate is computed as follows: for each partition, a classifier is trained on all the training data *except* that partition, which is held back as test set. Compared to just using the whole data for training, a fraction of $1 - 1/k$ of the data is “wasted” (e.g. 20% for $k = 5$) and the training becomes (roughly) $s \times k$ times as expensive, where s is the number of different parameter values to be tried – *in the general case*. The latter qualification is important, because for some important classifiers it is in fact possible to obtain very efficient implementations of certain types of cross-validation; e.g. leave-one-out cross-validation for knn classifiers (Ripley, 1996).

Apart from suffering from the waste of data and a certain ad hocness (for example: what exact type of and parameters for cross-validation should one choose in a particular problem context? How should continuous parameters as our α be discretized? – there are no "right" answers to these questions), the most significant shortcoming of cross-validation is that it scales very badly. Thus whereas even fairly hard-core Bayesians would acknowledge the usefulness of cross-validation – for example for basic sanity checks (e.g. MacKay, 1992) or the determination of a single parameter (such as kernel width) in an otherwise Bayesian model (Tipping, 2001), determining, for example, a separate regularization parameter for each of the M weights w_m via cross validation would be completely out of the question. It is however quite feasible by Bayesian methods as we shall see shortly. But probabilistic modeling is desirable for more fundamental reasons as well:

The appeal of probabilistic modelling

The principal attraction of adopting a probabilistic perspective is simple: the ultimate aim of regression and classification techniques is to support taking useful actions and decisions and this requires associating a measure of confidence in one's results. For example, in certain cases it might be better not to take any action at all, if the model is indicating very low confidence in its own predictions. As another example, particularly pertinent to classification, but also to regression: not all errors are equally costly.

Thus the principled way to reach a decision is to minimize the *expected loss*, i.e. the overall cost of all possible errors the model can make, weighted by their respective likelihood. Thus in the case where we seek to chose a single best model $\mathcal{M}^\star$, we should pick:

$$\mathcal{M}^\star = \underset{\mathcal{M}_i}{\operatorname{argmin}}\, \mathbb{E}(L(\mathbf{t}, \mathbf{y})p(\mathbf{y} \mid \mathcal{M}_i)) \tag{2.23}$$

Bayesianism and Frequentism: the main competitors

In the Bayesian approach not only the observed data but all model quantities of interest are (as far as is possible) modeled by random variables; thus for example rather than just modeling the observed data **t** as normally distributed with unknown mean and variance, we would also place prior or conditional distributions over the model parameters that reflect our prior beliefs and confidence about the likely values of these parameters.

It is easy to equate the error term E (2.16) with a probabilistic interpretation by noting that it only differs by a constant from a log-posterior over the parameters $-\log\left[p(\mathbf{t} \mid \mathbf{w}, \alpha, \sigma^2)p(\mathbf{w} \mid \alpha, \sigma^2)/p(\mathbf{t})\right]$ if the distributions involved belong to the exponential family. In particular if we place a zero-mean isotropic normal prior over $\mathbf{w} \mid \alpha, \sigma^2$, $q = 2$ and we end up with ridge regression. Similarly, for a Laplacian prior $\mathbf{w} \mid \alpha \sim \frac{\alpha}{2}\exp(-\alpha|\mathbf{w}|)$, $q = 1$ (see Figure 2.5). In order to derive an estimate for $\mathbf{y} = \mathbf{\Phi w}$, we *marginalize* over (i.e. integrate out; either analytically for simple distributions like Gaussians, or more often, by using approximate methods like MCMC) the so-called nuisance parameters – those model parameters that are not of direct interest to us ($\alpha, \mathbf{w}$ and σ^2in this case), in order to obtain a posterior distribution over only those quantities that interest us ($\mathbf{y}$).

Frequentists by contrast would reject the idea of placing priors over model parameters – and instead of integrating these parameters out to obtain a posterior distribution, they would seek to find a point estimate for the model quantities using the technique of *maximum likelihood* (ML) – i.e. choosing the model parameters in such a way that they maximize the likelihood of observing the data that actually has been observed. However this approach does not take prior beliefs or uncertainty in the parameters into account and also often yields biased estimates, or for corner cases, pathological results (Bishop, 2006). A similar technique is also used by Bayesians, as a "shortcut" where marginalization is too challenging: Maximum A Posteriori estimates are analogous to ML estimates, with the difference that the function to be optimized is the posterior, not the likelihood.

There is considerable friction between these two schools, and we will avoid getting into details, other than noting that we are firmly in the Bayesian camp [6].

Another contender: Statistical Learning Theory

The claim to fame of Statistical (also: Computational) learning theory (Vapnik, 1998) and PAC learning is the extremely popular support vector machine (SVM; see e.g. Schölkopf and Smola, 2002), which we will consider shortly.

Imagine the following game: you get to chose a set of training points; but we get to assign their classes. You then attempt to configure a given classifier so that all training points are classified according to the class labels we specified. If it succeeds you win, otherwise we win. The *Vapnik-Chervonenkis dimension (VC dimension)* of that classifier is then the highest number of training points for which you can *always* win by chosing training points at the right positions, no matter what class assignment we chose for them. For example, it is clear that classifier that classifies points according to which side of a line to the origin they lie has a VC dimension of at least 3 (and some further reflection shows that it is in fact exactly 3).

The complexity of a particular learning machine can be expressed by its VC dimension, which is the maximum number of points that can be arranged in a manner that allows the machine to correctly classify all possible labellings of these points. The significance of the concept of VC dimension is as follows: given the VC dimension of a model (and a few quite general assumptions) it is possible to derive a lower bound on its generalization ability of the following form: given the VC dimension of the model is below x, the likelihood that the test error exceeds the training error by more than δ is bounded from above by p – this is known as a PAC bound.

There are however some problems with this approach (Ripley, 1996):

- Contrary to a popular misunderstanding(Ripley, 1996), the PAC bound does *not* specify a upper bound on the probability p that given a certain observed training error e we can assume that the test error will lie no further than δ from e. Unfortunately, it would be precisely this claim that would be of the greater practical interest.

- Accurately estimating the VC dimension of a model may be quite hard.

- Finally, PAC bounds tend to be extremely loose: "As a consequence of the lack of any assumptions about the form of the distribution, the PAC bounds are very conservative, in other words they strongly over-estimate the size of data sets required to achieve a given generalization performance. *For this reason, PAC bounds have found few, if any practical applications*"(Bishop, 2006, emphasis mine).

Thus, although Statistical Learning Theory has historically been an important inspiration in the development of the hugely successful SVM, it is in our estimate not a serious competitor to the Bayesian approach as a practically useful modeling framework.

[6]For a highly partisan, but illuminating account of Bayesian statistics and comparison with Frequentism consult (Jaynes, 2003), maybe the prime contender for an survey of machine learning from a frequentist perspective is (Hastie et al., 2001)

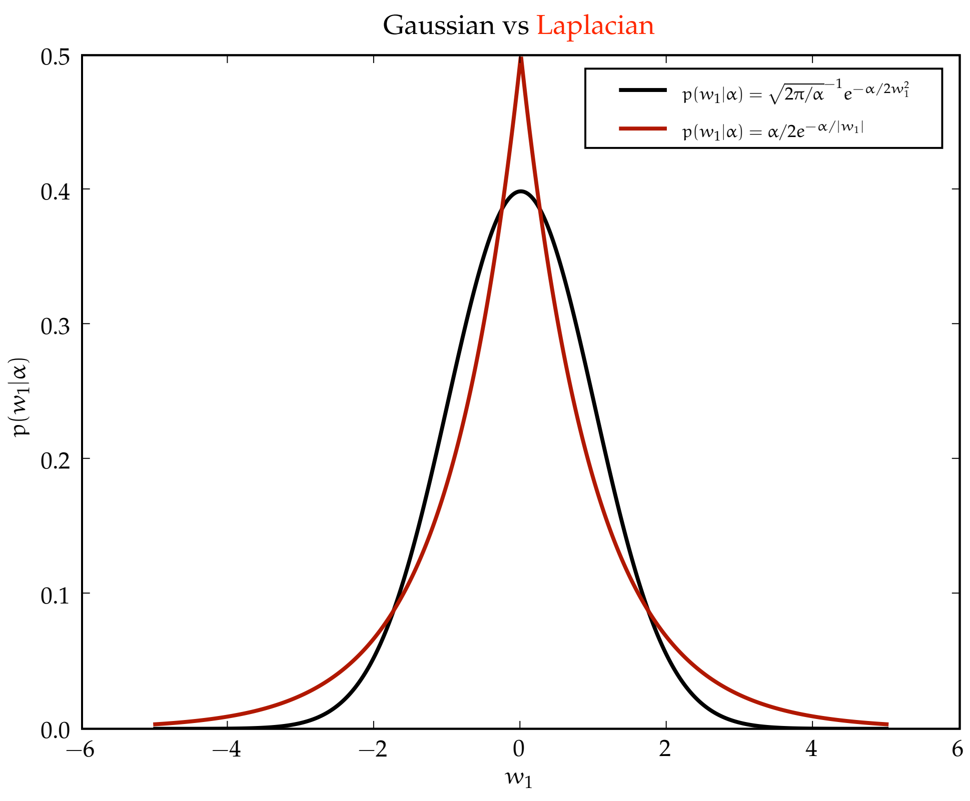


Figure 2.5: The zero-mean Laplace distribution (solid red) assigns more probability mass to the region immediately around zero and to the tails than the zero mean Gaussian (solid black), thus favouring sparser results.

2.1.3 The LASSO

Tibshirani (1996) popularized the regularization term $\alpha\|\mathbf{w}\|_1$ in a frequentist regression context (but also see Williams, 1994). The paper proposes choosing:

$$\tilde{\mathbf{w}} = \underset{\{w_m\}_{m=1}^{M}}{\operatorname{argmin}} \; (\mathbf{t} - \boldsymbol{\Phi}\mathbf{w})^2 , \qquad \text{subject to } \|\mathbf{w}\|_1 - |w_0| \leqslant \alpha \tag{2.24}$$

(*NB.* $\mathbf{w}$'s intercept coefficient w_0 is unconstrained). The solution can be found with quadratic programming methods, but a more recent and efficient scheme is based on the greedy least-angle-regression approach given in (Efron et al., 2002, 2004).

One difficulty with the lasso is obtaining a measure of confidence in the accuracy of the result. Tibshirani (1996) notes that the analytically inconvenient form of (2.24) complicates finding a good estimate of standard error and propose bootstrapping or an approximation via ridge regression; however the latter method unattractively assigns a variance of 0 to all $w_m = 0$.

There are different methods to choose the regularization parameter α – cross-validation is an obvious, if data-hungry and inconvenient choice; generalized cross validation Wahaba (1984) and Stein's unbiased estimator of risk (SURE; Stein, 1981) are other possibilities.

As a constraint of the form $\|\mathbf{w}\| \leqslant$ const may be rewritten as an additional penalty term of the form $\lambda\|\mathbf{w}\|$, optimizing (2.24) is equivalent to finding the MAP solution after placing a Laplacian prior over $\mathbf{w}$: $p(\mathbf{w}_i \,|\, \lambda) = \frac{1}{\lambda}\exp(\lambda|w_m|)$. Thus, the lasso can also be seen in a (somewhat) Bayesian light. However, since the choice of λ will generally be non-obvious a-priori and crucially affects the result, a more fully Bayesian scheme not using cross-validation as a crutch would also need to place a prior over λ.

It is often illuminating to consider the simple case of orthogonal predictors. In particular it can be shown that the solutions to (2.24) then are just given by (momentarily using $\hat{\mathbf{w}}$ to denote the least-square estimate):

$$\hat{w}_m = \text{sign}(\hat{w}_m)(|\hat{w}_m| - \gamma)_+ \tag{2.25}$$

where $(x)_+ = x$ if $x > 0$ else 0 and γ is a positive constant that determined by the constraint $\|\mathbf{w}\|_1 - |w_0| \leqslant \alpha$. This above equation is precisely equivalent to the soft-threshold rule popular in wavelet shrinkage that we shall encounter now.

2.1.4 Wavelet Shrinkage

As is well known the Fast Fourier Transform (FFT) efficiently (in $O(N \log N)$ steps) decomposes a signal into a series of orthogonal sinusoids, corresponding to a transformation from time to frequency space. Thus, it is possible to represent a chord of three pure notes with only three non-zero Fourier coefficients, each corresponding to a note. However, any signal that is completely localized in frequency space is necessarily periodic, whereas in reality most signals will be composed of parts that are approximately localized in both time and space and are not naturally seen as forever repeating. For example, we do not encounter chords as eternal and platonic sounds on their own but as part of a score, which, when viewed in Fourier space would appear as a smear of frequencies.

Such a decomposition of a signal into components localized in both frequency and time is provided by the Discrete Wavelet Transform (DWT) and in only $O(N)$ steps, by means of clever filter bank designs. Algebraically, however, the DWT of a vector $\mathbf{t}$ is simply a length-preserving linear operator and as such equivalent to $\mathbf{\Phi}^T\mathbf{t}$, where $\mathbf{\Phi}$ is a particular orthogonal matrix. However, unlike the Fourier transform, which uses a single fixed set of basis functions (scaled and shifted sinusoids), there are various different families wavelets that each constitute a complete and orthogonal set of basis functions(see also Ogden, 1997).

As is the case with the Fourier transform, a simple way[7] to extend the one-dimensional wavelet transform to multiple dimensions is to perform consecutive 1D transforms over each axis of the multidimensional array. For 2D arrays such as image (denoted by the matrix $\mathbf{X}$ below) this boils simply down to first wavelet transforming the columns followed by row-wise transform on the obtained coefficient vectors:

$$\mathbf{\Theta} = \mathbf{W}^T\mathbf{X}\mathbf{W} = (\mathbf{W}^T\mathbf{X})\mathbf{W} = \mathbf{W}^T(\mathbf{X}\mathbf{W}) \tag{2.26}$$

or vice-versa – as emphasized by the bracketing, the order in which the conversions are performed is immaterial.

Since the discrete wavelet transform is an orthogonal linear operator, it is easily verified that it maps stationary white noise[8] on the targets to stationary white noise of the same amplitude on the wavelet coefficients. On the other hand, for "reasonable" noise-free curves ("reasonable" in this context means signals that can be well approximated by piecewise polynomials of a small degree; something that is true) and a suitable choice of wavelet, wavelet transforms are said to be *decorrelating*. In other words whilst noise enters equally into all wavelet coefficients, the true signal carried by the targets will be mostly concentrated in but a few.

This suggests the following template for wavelet-based denoising: transform to wavelet space

[7]There are also more complicated constructions for multidimensional wavelets, see e.g. (Kovačević and Vetterli, 1991) as well as several newer wavelet-like transforms such as ridgelets and curvelets (see e.g. Do and Vetterli (2003), Starck et al. (2002))that are specifically aimed at addressing weaknesses of wavelets for higher dimensional data, such as lack of multi-orientation in addition to multi-scale resolution.

[8]Noise consisting of independent variates drawn from $\mathcal{N}(0, \sigma^2)$ for some fixed σ^2

($\mathbf{w_t} = \mathbf{\Phi}^T \mathbf{t}$), somehow cull or *shrink* those coefficients that contain largely noise leaving the signal carrying ones mostly intact and transform back ($\hat{\mathbf{y}} = \mathbf{W}\,\mathrm{shrink}(\mathbf{w_t})$).

But $\mathbf{W}\,\mathrm{shrink}(\mathbf{w_t})$ is really just a regularized projection from (2.55) – with regularization coefficients $\sigma^2\boldsymbol{\alpha}$ controlling the amount of shrinkage and the added bonus that the covariance matrix $\mathbf{\Phi}^T\mathbf{\Phi}$ becomes the identity matrix $\mathbf{I}$ and multiplications by $\mathbf{\Phi}$ can be carried out in linear time. This implies that wavelet shrinkage is just a particularly attractive special case of sparse regression and, as will be seen in section 2.5, a method that works well for wavelets but does not inherently rely on orthogonality or wavelet specific features (multi-resolution) can also be generalized to other types of dictionary $\mathbf{\Phi}$.

A number of approaches to shrinking the wavelet coefficients have been devised. A straightforward idea is to just set to zero those coefficients whose absolute values remains below a certain threshold λ, i.e. set $\alpha_i = \infty$ for all $|w_i| < \lambda$ (*hard thresholding*). Additionally reducing all the other coefficients towards zero by said threshold λ is another, often preferable, alternative (*soft thresholding*; inter alia it gives a continuous shrinkage curve which is analytically more convenient) (Donoho and Johnstone, 1994). A simple and popular threshold choice is the *universal threshold* proposed by Donoho et al. (1995):

$$\lambda = \sigma\sqrt{2\log N} \tag{2.27}$$

Where σ is the (true) noise standard deviation. Since σ is not known in most applications, Donoho et al. (1995) suggest using the median absolute deviation estimator to derive a robust estimate:

$$\hat{\sigma} \equiv 1.483 median(|\tilde{w}_{J-1,1}| \dots |\tilde{w}_{J-1,2^{J-1}}|) \tag{2.28}$$

The universal threshold can be shown to zero out noisy components with high probability (asymptotically 1 as $N \to \infty$) [9]

Undoubtedly the reader will also be pleased to learn that both hard and soft-thresholding are asymptotically optimal for a wide variety of function spaces Donoho and Johnstone (1994).

Of course a very natural approach to consider for threshold selection is cross validation. However wavelet-shrinkage poses a difficulty here: there is no straighforward way of efficiently carrying out wavelet transforms on arbitrary subsets of the data, so that there is no obvious way of applying common CV schemes like V-fold or 1-fold CV, because only uniformly spaced samples of dyadic length can be handled with ease. Nason (1996) tackled this hurdle by essentially using odd-valued coefficients (shrunk to a certain threshold λ) to predict the even-valued targets and vice versa (thus cleverly creating two wavelet-transform-friendly, uniformly sampled data sets of dyadic length for validation purposes); his optimal threshold takes the form:

$$\lambda^\star = \frac{1}{\sqrt{1-\frac{\log 2}{\log n}}} \operatorname*{argmax}_{\lambda} \sum_{n=1}^{N/2} (\hat{y}^E_{\lambda,n} - t^E_{2n+1})^2 + (\hat{y}^O_{\lambda,n} - t^O_{2n})^2 \tag{2.29}$$

where $\{\hat{y}^E_{\lambda,n}\}_{n=}^{N/2}$ ($\{\hat{y}^O_{\lambda,n}\}_{n=}^{N/2}$) are the even-valued (odd-valued) coefficients shrunk to the threshold λ, and $\{t^E_{2n+1}\}_{n=1}^{N/2}$ ($\{t^O_{2n}\}_{n=1}^{N/2}$) are the odd-valued (even-valued) targets. It can be seen that without the leading adjustment factor, the selected λ would simply be the one that gave the lowest squared prediction error averaged between predictions from even to odd and vice versa. The first factor is a heuristic adjustment for the sample size ($N/2$ instead of N) derived from Donoho and Johnstone (1994)'s aforementioned universal threshold, $\lambda^\star = \sqrt{2\log N\sigma^2}$.

A related method, *Generalized Cross Validation* (GCV), was first introduced by Wahaba (1984)

[9]The universal threshold can be shown to be the expected maximum of N iid Gaussian variables. Since as discussed above, the noise in wavelet space is of this form $\tilde{\boldsymbol{\epsilon}} \sim \mathcal{N}(\mathbf{0}, \sigma\mathbf{I})$ it follows that as $N \to \infty$, $\Pr(\mathbb{E}[\max\{\epsilon_n\}_n^N] \leqslant \sigma\sqrt{2\log N}) \to 1$.

in the context of spline approximations but has also been used for wavelets by several authors (see references in Vidakovic, 1999):

$$\mathrm{GCV}(\lambda) = \frac{\frac{1}{N}\|\mathbf{t} - \hat{\mathbf{y}}_\lambda\|^2}{\left(\frac{1}{N}\operatorname{tr}(\mathbf{I} - \mathbf{A}_\lambda)\right)^2} \tag{2.30}$$

$$\mathbf{A}_\lambda = \mathbf{W}\operatorname{diag}_n(\mathrm{I}[|w_n| > \lambda])\mathbf{W}^\mathrm{T} \tag{2.31}$$

$$\hat{\mathbf{y}}_\lambda = \mathbf{A}_\lambda \mathbf{t} \tag{2.32}$$

$$\lambda^\star = \operatorname*{argmin}_\lambda \mathbb{E}[\mathrm{GCV}(\lambda)] \simeq \operatorname*{argmin}_\lambda \mathrm{GCV}(\lambda) \tag{2.33}$$

Whilst a full discussion of GCV would lead to far afield, I will give a brief rationale for the above formulas. $\mathrm{GCV}(\lambda)$ is essentially an penalty based on the badness of fit of the estimate $\hat{\mathbf{y}}_\lambda$ to the targets $\mathbf{t}$, tempered by an inverse measure of the complexity of said estimate (respectively numerator and denominator of (2.30)). The estimate itself is derived by projecting $\mathbf{t}$ into a rank-reduced space via the matrix $\mathbf{A}_\lambda$ (2.31), and the severity of the rank reduction is controlled by λ. Finally the value for λ that achieves the lowest GCV penalty is selected (2.33), (and its a corresponding estimate $\hat{\mathbf{y}}_\lambda$) One attraction of the method is that $\mathrm{GCV}(\lambda)$ can be computed (or well approximated) with comparatively little computational effort for many models.

For an overview of various other traditional threshold selection methods (including induced percetile, Lorentz curve and block thresholding estimators) see (Vidakovic, 1999, ch. 6). Hard thresholding and soft-thresholding suffer from different defects, the former results is liable to result in spike artifacts the latter can so severly dampen large scale, but mostly noise-free components that $\hat{\mathbf{y}}$ is an inferior estimate than the unshrunk targets $\mathbf{t}$! (Denison et al., 2002). Bayesian methods are a natural choice to effect a more subtle calibration between those two extremes and a number of such schemes have been advocated.

Clyde et al. (1998) suggest:

$$\theta_{rd}\hat{w}_{rd} = w_{rd} + \tau_{rd} \tag{2.34}$$

$$p(w_{rd} \mid \theta_{rd}) = \mathcal{N}(0, \theta_{rd}\sigma^2\nu_r), \theta_{rd} \sim \mathrm{Br}(\gamma_r) \tag{2.35}$$

where $\mathrm{Br}(\theta_{rd} \mid \gamma_r) = \gamma_r^{\theta_{rd}}(1-\gamma_r)^{1-\theta_{rd}}$ is the Bernoulli distribution and $\{\gamma_r\}_r$, $\{\nu_r\}_r$; are fixed, resolution level dependent hyperparameters. Unfortunately, as the authors note, "all hyperparameters play an important role and should be elicited carefully". They suggest Bayesian variants of classical model choice criteria as a possible recourse.

In a similar vein, Chipman et al. (1997) propose:

$$p(w_{ji} \mid \theta_{ji}) = \theta_{ji}\mathcal{N}(0, \nu_j) + (1 - \theta_{ji})\mathcal{N}(0, c_j\nu_j) \tag{2.36}$$

where c_j is close to zero and the hyperparameters are determined via empirical Bayes.

Last but certainly not least there is Holmes and Denisons' (1999) smoothness prior which we will cover in considerable detail in the following chapter since, unlike most popular wavelet shrinkage priors, it is not dependent on the wavelet length scale or level and hence also suitable for non-wavelet dictionaries. Holmes and Denison reject such level dependence as inconsistent with the knowledge that noise enters additively across all components.

More in-depth treatments of wavelets in general can be found in Mallat (1999) whereas wavelet shrinkage is extensively reviewed in Jansen (2001); Vidakovic (1998); Denison et al. (2002, sec. 3.4).

2.1.5 Beyond wavelets: pursuit methods

Closely related to wavelet shrinkage are various *pursuit* methods, which seek an "optimal" representation of a signal in an overcomplete dictionary of basis functions. The definition of optimal varies between method, but typically the problem is NP-hard so that a (typically greedy) heuristic method has to be adopted. What sets time-frequency methods like *Basis pursuit* (Chen et al., 1999) and *Matching pursuit* (Mallat and Zhang, 1993) apart from older more general greedy regression techniques such as backfitting (Hastie et al., 2001, see, e.g.) is their extreme efficiency (often linear!) and sparsity which essentially also makes them suitable for blind source separation (for a very modest and non-representative toy example of achieving something similar with the sRVM see section 2.6.5).

2.1.6 Shortcomings of wavelets

Wavelets have been seen to boast a combination of extraordinary strengths: they are analytically and numerically convenient due to their orthogonality, decorrelate signal and noise, due to a mixture of orthogonality (again) and the ability to sparsely represent a wide range of signals of interest and, best of all, achieve all that at a ridiculously low computational cost (in terms of algorithmic complexity $O(N)$ is just as good as it gets for a method that needs to consider N data points). So where's the catch? The most striking limitation is that the DWT only really works well with uniformly sampled data of dyadic length (i.e. $N = 2^J, J \in \mathbb{N}$). Neither of these points are strict requirements – there are various proposals for dealing with non-uniformly spaced data, and as in is the case for the FFT, padding the signal to dyadic length is always an option. However in practice, equal spacing and signals of dyadic length are near ubiquitous in both literature and readily available implementations[10], non-equal spacing moreover necessarily comes at the price of non-orthogonality. So whilst the Wavelets are very well suited to periodically sampled signals as one often finds in time series analysis and signal processing (Vidakovic, 1999), they are much less well suited for e.g. regression tasks in the social sciences.

Nonetheless, on balance wavelets are more widely applicable than maybe any other set of orthogonal basis functions (such as orthogonal polynomials, natural splines and, of course, sinusoids) which is why we almost exclusively concentrate on them here.

2.2 Classification and Kernels

Basis functions have already been mentioned as a means to increase model expressiveness whilst retaining most of the benefits of plain linear models. Kernels are in a way a step further in the same direction, and although they are equally applicable to regression, they are maybe most naturally illustrated in the context where they had most of their initial impact: classification.

In classification, which can be regarded as discrete regression, a continuous variable $\mathbf{x}$ is to be mapped to one of a discrete number, most often 2, of classes: i.e. $t \in 0, 1$.

As figure 2.2 illustrates, many problems become amenable to linear classifiers after applying a transformation $\phi : M \to M'$ to map the $M-$dimensional input space in of the data point $\mathbf{x}_n$ into a higher, M'-dimensional feature space.

1011

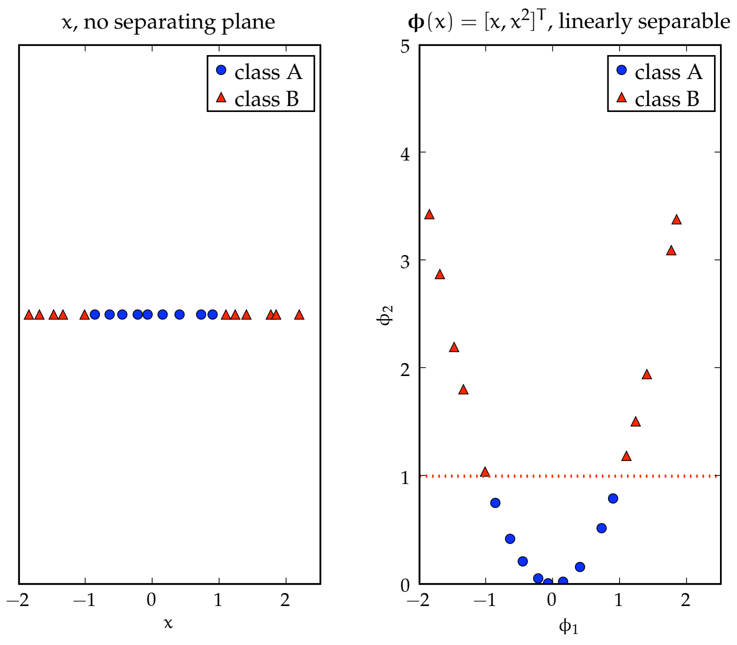


Figure 2.6: Achieving linear separability through mapping to a higher dimensional feature space. By mapping the one dimensional data set $\{x_n\}_n^N$ into two dimensions; $\{\boldsymbol{\phi} x_n\}_n^N$ with an appropriate basis function (here $\boldsymbol{\phi}(x) = [x \quad x^2]^T$), it becomes linearly separable.

This much can achieved within the framework of basis functions already considered.

Kernels, however, go one better on basis functions as follows: they allow working in *very* high, possibly infinite-dimensional, feature spaces. This works as follows: it turns out that many pattern recognition methods can be reformulated so as not to use $\boldsymbol{\Phi}$ directly, instead working in terms $K = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$.

To give a concrete example, by substituting $\hat{\mathbf{a}} = \boldsymbol{\Phi}^T\hat{\mathbf{w}}$ in the ridge-regression error function (2.16) and setting the derivative with respect to $\hat{\mathbf{a}}$ to zero, one obtains the *dual* (i.e. equivalent) kernel space solution (cf 2.17)

$$\hat{\mathbf{a}} = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{t} \tag{2.37}$$

One may then express the regression estimate $\hat{\mathbf{y}} = \boldsymbol{\Phi}\hat{\mathbf{w}}$ solely in terms of $\mathbf{K}$, without resorting to $\boldsymbol{\Phi}$:

$$\hat{\mathbf{y}} = \boldsymbol{\Phi}\hat{\mathbf{w}} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T\hat{\mathbf{a}} = \mathbf{K}\hat{\mathbf{a}} \tag{2.38}$$

Writing $K_{nm} = k(\mathbf{x}_n, \mathbf{x}_m) \equiv \boldsymbol{\phi}(\mathbf{x}_m)^T\boldsymbol{\phi}(\mathbf{x}_n)$ makes it plain that the basis functions now only occur, indirectly, in inner products. Since any direct mention of M' has disappeared along with $\boldsymbol{\Phi}$ (NB: whereas $\hat{\mathbf{w}}$ in (2.17) was found by inverting a $M' \times M'$ matrix, (2.37) involves inverting a $N \times N$ matrix!) and since it can be shown that many simple choices for $k(\mathbf{x}, \mathbf{y})$ correspond to a basis function of the type $\phi : \mathbb{R}^M \to \mathbb{R}^\infty$, we are thus free to work in ∞-dimensional feature space!

A prime instance of such a kernel function, and one that will be occur frequently in this thesis is the humble Gaussian[12]: $k(\mathbf{x}, \mathbf{y}) = \exp(-\frac{1}{2}\sigma^{-2}\|\mathbf{x} - \mathbf{y}\|^2) = \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{y})$, where $\boldsymbol{\phi} : \mathbb{R}^M \to \mathbb{R}^\infty$. Although the second equality is not immediately obvious, it can be easily demonstrated by expanding into $\exp(-\frac{1}{2}\sigma^{-2}\mathbf{x}^T\mathbf{x})\exp(\sigma^{-2}\mathbf{x}^T\mathbf{y})\exp(-\frac{1}{2}\sigma^{-2}\mathbf{y}^T\mathbf{y})$. The outer factors pose no problem, because given a suitable $\tilde{\boldsymbol{\phi}}$ such that $\tilde{\boldsymbol{\phi}}(x)^T\tilde{\boldsymbol{\phi}}(y) = exp(-\sigma^{-2}\mathbf{x}^T\mathbf{y})$ one can just set $\phi(\mathbf{x}) =$

[12]No normalization constant is needed in the context of kernels and basis functions.

$\exp(-\frac{1}{2}\sigma^{-2}\mathbf{x}^T\mathbf{x})\tilde{\boldsymbol{\phi}}(\mathbf{x})$. The middle term itself can be expanded using the power-series expansion for e^x, as below. I restrict myself to 2-dimensional vectors $\mathbf{x}$ and $\mathbf{y}$, to avoid notational clutter but the generalization from the binomial to the multinomial theorem is obvious.

$$\begin{aligned}
\exp(\sigma^{-2}\mathbf{x}^T\mathbf{y}) &= \sigma^{-2}\sum_{n=0}^{\infty}\frac{(\mathbf{x}^T\mathbf{y})^n}{n!} && (2.39)\\
&= \sigma^{-2}\sum_{n=0}^{\infty}\frac{(x_1y_1+x_2y_2)^n}{n!} && (2.40)\\
&= \sigma^{-2}\sum_{n=0}^{\infty}\frac{\sum_{i=0}^{n}\binom{n}{i}(x_1y_1)^i(x_2y_2)^{n-i}}{n!} && (2.41)\\
&= \sum_{n=0}^{\infty}\frac{\sigma^{-2}}{n!}\sum_{i=0}^{n}\binom{n}{i}x_1^ix_2^{n-i}y_1^iy_2^{n-i} && (2.42)\\
&= \left[\sqrt{\frac{1}{\sigma^2 n!}\binom{n}{i}}x_1^ix_2^{n-i}\right]^T_{0\leqslant n<\infty,0\leqslant i\leqslant n}\left[\sqrt{\frac{1}{\sigma^2 n!}\binom{n}{i}}y_1^iy_2^{n-i}\right]_{0\leqslant n<\infty,0\leqslant i\leqslant n} && (2.43)\\
&= \tilde{\boldsymbol{\phi}}(\mathbf{x})^T\tilde{\boldsymbol{\phi}}(\mathbf{y}), \qquad \tilde{\boldsymbol{\phi}}:\mathbb{R}^M\to\mathbb{R}^\infty && (2.44)
\end{aligned}$$

This shows that a Gaussian kernel function indeed corresponds to using a basis function $\tilde{\boldsymbol{\phi}}$ that maps to an infinite dimensional feature space.

The Gaussian belongs to the popular class of distance based kernel functions of the type $k(\mathbf{x},\mathbf{y}) = f(\|\mathbf{x}-\mathbf{y}\|)$, which in turn is subsumed by the class of *stationary*[13] kernels of the form $k(\mathbf{x},\mathbf{y}) = f(\mathbf{x}-\mathbf{y})$. Other popular kernel choices include polynomial kernels as well as different types of spline kernels.

A further important attraction of kernels is that they provide a powerful way to deal with symbolic inputs, for example, $k(A,B) = 2^{|A\cap B|}$, the size of the powerset of the intersection of two sets A and B, is a valid kernel function Bishop (2006).

Finally, in order to construct a valid kernel, it is not necessary to demonstrate its equivalence to some particular basis function inner product – an important result states that fulfilling *Mercer's condition* is a necessary and sufficient condition for k to constitute a valid kernel function (in linear algebra terms, k constitutes a valid kernel function iff for any choice of $\{\mathbf{x}^{(n)}\}_n$, $\forall\mathbf{v}:\mathbf{v}^T\mathbf{K}\mathbf{v}\geqslant 0$, in other words iff $\mathbf{K}$ is positive-semidefinite) and any book on kernel methods will provide conditions for easily deriving new kernel functions from existing kernel functions.

2.2.1 Gaussian processes

Loosely speaking, Gaussian Processes (Rasmussen and Williams, 2006, see, e.g.) are to the normal distribution as square integrable functions are to vectors: both are extensions of finite-dimensional concepts to the infinite dimensional case.

Assuming a fixed set of basis functions $\boldsymbol{\phi}_m$, the Bayesian linear regression model considered earlier, which places a Gaussian prior over the weights $\mathbf{w}\sim\mathcal{N}(\mathbf{0},\alpha^{-1}\mathbf{I})$, also indirectly specifies a prior over the signal $\mathbf{y}=\boldsymbol{\Phi}\mathbf{w}$, and this prior, too, is a Gaussian, since $\mathbf{y}$ is a linear combination of normally distributed variables. The key characteristic of a Gaussian Process (GP) is the covariance

[13]So called because they are invariant to translations in input space, $k(\mathbf{x}-\boldsymbol{\Delta},\mathbf{y}-\boldsymbol{\Delta}) = k(\mathbf{x},\mathbf{y})$.

function that defines it[14]. In it we again encounter a kernel :

$$\mathrm{cov}(\mathbf{y}) = \mathbb{E}(\mathbf{y}^T\mathbf{y}) = \mathbf{\Phi}\mathbb{E}(\mathbf{ww})\mathbf{\Phi}^T = \mathbf{\Phi}\,\mathrm{cov}(\mathbf{w})\mathbf{\Phi}^T = \alpha^{-1}\mathbf{\Phi}\mathbf{\Phi}^T \equiv \mathbf{K} \tag{2.45}$$

$$p(\mathbf{t}) = \int p(\mathbf{t}\,|\,\mathbf{y})p(\mathbf{y})d\mathbf{y} \equiv \mathcal{N}(\mathbf{t}\,|\,\mathbf{0},\mathbf{C}) \tag{2.46}$$

As usual for kernel models, **K** can also be specified directly without the intermediate step of basis functions and we again write $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$; a popular choice is $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\rho|\mathbf{x}_i - \mathbf{x}_j|^2)$, where ρ is a scale parameter that can be learned as part of the model.

Inference in Gaussian processes leverages convenient partitioning identities available for Gaussians – thus the mean and covariance for the conditional prediction $p(t_{N+1}\,|\,t_1,\ldots,t_N)$ can be expressed as linear combinations of $\mathbf{C}^{-1}$, the inverse noise variance β, the targets **t** and the kernel function of $\mathbf{x}_{N+1}$.

Unfortunately Gaussian processes, although powerful, tend to be rather computationally demanding, and as Quinonero-Candela et al. (2007) conclude in a recent survey of approximation methods for Gaussian Processes:

> Probably the question that is most on the mind of the practitioner is "what [approximation] method should I use on my problem?" Unfortunately this is not an easy question to answer, as it will depend on various aspects of the problem such as the complexity of the underlying target function, the dimensionality of the inputs, the amount of noise on the targets, etc. There has been some empirical work on the comparison of approximate methods, for example in Schwaighofer and Tresp (2003) and Rasmussen and Williams (2006, chapter 8) but more needs to be done.

2.2.2 Support Vector Machines

Support Vector Machines (see e.g. Schölkopf and Smola, 2002) are a very successful and popular paradigm for classification (also extensible to regression) that leverages this kernel trick. The general idea is to use effective interior point methods to fit a hyperplane in kernel space between the different classes maximizing the margin between hyperplane and the closest targets to it for both classes. These targets, which "prop up" the hyperplane are the eponymous support vectors. It is easy to appreciate the positive effect of the maximum margin criterion on generalization performance: the wider the margin between the two classes, the less likely new test-points on the fringes of either class are likely to find themselves on the wrong side of the boundary. However, since noise in the targets almost invariably means that models powerful enough to achieve perfect linear separation on the training data overfit drastically, a cross-validation determined slack parameter is introduced. For the purpose of fitting the hyperplane, misclassified points whose added distance from the (putative) separating hyperplane, relative to the margin-width, lies below a certain total, the slack parameter, are simply ignored.

The ideas behind SVM classification can also be adapted to regression. The error criterion that is minimized is split into a data and regularization term. The data error is measures as the total absolute difference between individual training points and predictions, ignoring the cases in which this difference is below a pre-chosen slack parameter and the regularization term is a constant times the quadratic norm of the weights (apart from the intercept) that parametrize the model. Thus there are two hyperparameters, the slack parameter and a "classical" regularization term. Whilst the former is often set a priori, the latter typically has to be found via cross validation.

As a result of the popularity and maturity of the SVM framework much effort has been ex-

[14]To be precise, this is just a particularly common instance of a GP with zero mean, since $\mathbb{E}(\mathbf{y}) = \mathbf{0}$; in the general case we also need to know the mean in addition to the covariance **K**

panded in creating highly tuned libraries for SVM classification, such as LIBSVM[15] and making SVMs perform well on very large data sets continues to be an area with much active research (Bottou et al., 2007).

It must also be noted that asymptotic complexity results can be misleading, due to large constant factors that turn out to matter a lot in practice. For example, although SVMs scale cubically in the number of selected components, S and up to cubically in N (when the margin parameter C gets large), in practice the asymptotically comparatively innocuous looking $O(N^2)$ time (and space) complexity of computing the N^2 kernel matrix entries $\mathbf{K}_{[i,j]} = k(\mathbf{x}_i, \mathbf{x}_j)$ turns out to be so expensive in practice that all competitive SVM implementations incorporate techniques to avoid (re)computation of unnecessary kernel entries that range from caching to much more sophisticated schemes[16](Bottou and Lin, 2007).

However, despite the great success that SVMs have enjoyed in the machine learning and data mining community for over a decade and the continued manpower dedicated to research and implementation efforts, the SVM has several shortcomings. Firstly, it only handles dichotomization[17] naturally and extensions to the multi-class case tend to be somewhat awkwardBishop (2006). However, the most severe shortcoming to our mind is the difficulty to obtain proper probabilistic outputs and confidences, whose crucial role in the decision making process has already been touched upon. Thus we will now consider an alternative to SVMs that affords such probabilistic interpretation: the Relevance Vector Machine.

[15]`http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

[16]This is one of the reasons that early expectations for leveraging state-of-the art Quadratic Programming Solvers for efficient SVM implementations did not come to pass: these solvers generally assume a complete **K** in memory.

[17]Two-class problems.

Part I

Regression and the smooth Relevance Vector Machine

Our starting point for sparse Bayesian regression is the basic linear regression model with iid Gaussian noise $\boldsymbol{\epsilon}^{N\times 1}$, signal $\mathbf{y}^{N\times 1}$ and targets $\mathbf{t}^{N\times 1}$ such that:

$$\mathbf{t} = \mathbf{y} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{m}0, \sigma^2 \mathbf{I}_N) \tag{2.47}$$

Whilst in non-Bayesian schemes, such as SVM regression, a desirable level of sparsity has often to be brought about indirectly by determining an error or margin parameter via a cross-validation scheme, the Bayesian formulation of the regression problem in the relevance vector machine (RVM) (Tipping, 2000, 2001; Faul and Tipping, 2002; Tipping and Faul, 2003) allows for a prior structure that explicitly encodes the desirability of sparse representations.

This is done by complementing the standard likelihood function (which follows directly from the above assumptions):

$$p(\mathbf{t} \,|\, \mathbf{w}, \sigma^2) = (2\pi\sigma^2)^{-\frac{N}{2}} \exp\left(-\frac{\|(\mathbf{t} - \boldsymbol{\Phi}\mathbf{w})\|^2}{2\sigma^2}\right) \tag{2.48}$$

with an "automatic relevance determination" prior (MacKay, 1992) over the weights $\mathbf{w}^{M\times 1}$:

$$p(\mathbf{w} \,|\, \boldsymbol{\alpha}) = (2\pi)^{-\frac{M}{2}} \prod_{m=1}^{M} \alpha_m^{\frac{1}{2}} \exp\left(-\frac{1}{2}\alpha_m w_m^2\right) \tag{2.49}$$

that has the effect of "switching off" basis functions for which there is no evidence in the data (more on this in sec. 2.5).

Whilst $p(\boldsymbol{\alpha})$ is effectively uniform[18], a standard inverse gamma prior is placed over the noise variance σ^2:

$$p(\sigma^2) = \mathcal{IG}(\sigma^2 \,|\, g, h) = \frac{h^g}{\Gamma(g)} \sigma^{-2(g+1)} e^{-h/\sigma^2} \tag{2.50}$$

where g and h are fixed hyperparameters, usually set to some uninformative value (e.g., $g = h = 10^{-4}$).

It should be stressed that in this scheme g and h are the only "true" hyperparameters, in the sense that unlike everything else that is introduced in extending the standard regression model (2.48) by a hierarchical prior (2.49), (2.50), they are additional parameters that require specification by the user (and even in the somewhat unlikely absence of any prior information about $p(\sigma|g, h)$ just setting them to some uninformative default will work fine). All the other variables ($\boldsymbol{\alpha}$ etc.) are just nuisance parameters that can be integrated out or determined by the Bayesian approach.

Thus everything else, including ultimately $\hat{\mathbf{y}}$, the mean posterior prediction that we wish to obtain is determined by the values of $\boldsymbol{\Phi}$, $\mathbf{t}$, g and h.

This is the beauty of the Bayesian paradigm – it allows one to reap the benefits of a probabilistic approach, without burdening the model with additional externally-determined parameters (unlike the SVM, there is no need to expensively determine a regularization parameter via cross-validation and furthermore confidence intervals, likelihood values and posterior probabilities for the solution can easily be obtained).

The learning of the model parameters proceeds by a type II likelihood maximization scheme in which values of $\boldsymbol{\alpha}$ and σ that maximize the log marginal likelihood $\mathcal{L}(\boldsymbol{\alpha}) = \log p(\mathbf{t} \,|\, \boldsymbol{\alpha}, \sigma^2)$ are found iteratively (Faul and Tipping, 2002; Tipping and Faul, 2003). Weights w_m for which the learned precision α_m is large are effectively switched off because w_m is constrained to be close to, or, more often, exactly zero. Although the importance of this will become clear soon, it is useful to mention even at this point that the above scheme is a "second generation" RVM implementation for which turned-off weights have crucial performance (and numerical robustness)

[18]A Jeffreys' prior is apparently advocated in (Tipping, 2001), but (Tipping, 2000; Faul and Tipping, 2002; Tipping and Faul, 2003) and the published implementation use a uniform prior. This is discussed further in see sec. 2.8.1.

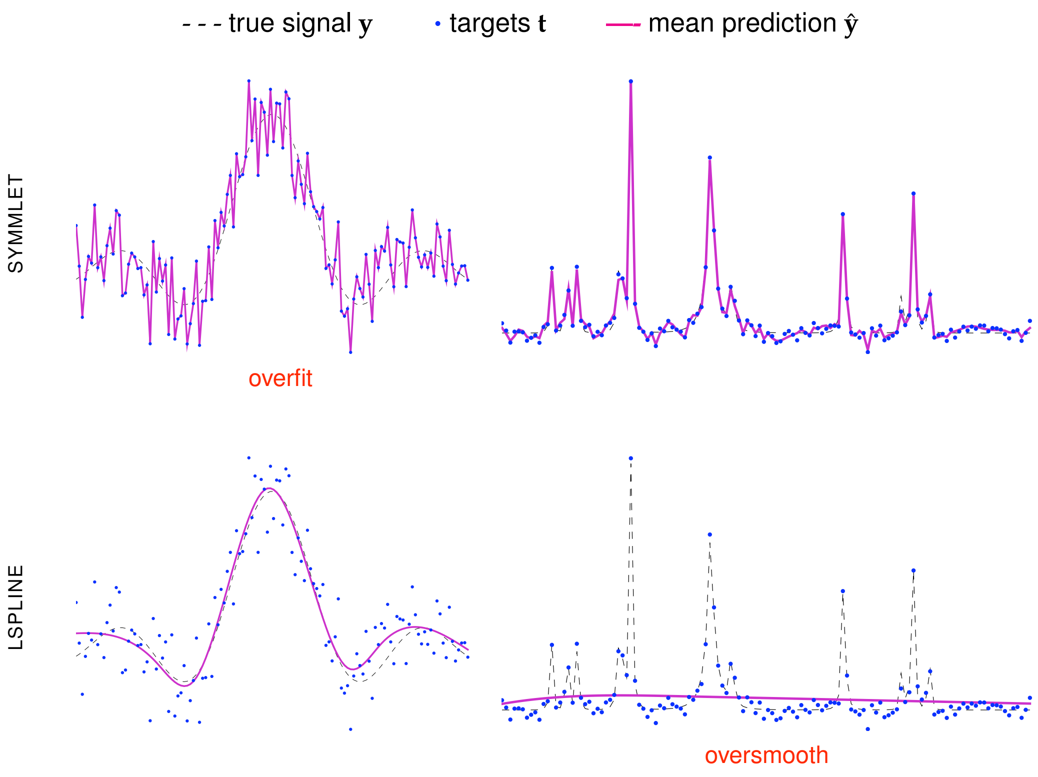


Figure 2.7: Classical RVM. The effect of dictionary choice on the smoothness of the regression result (Sinc data *left*, Bumps data *right*) when there is no prior over $\boldsymbol{\alpha}$. Choosing a flexible symmlet-wavelet dictionary (*top row*) results in drastic overfitting for the Sinc data set (*top left*; N=128, SNR=2.0). To obtain the appropriate level of smoothing for the Sinc data one has to resort to a different dictionary type, such as lspline (*bottom left*). However an lspline dictionary cannot resolve the Bumps data (*bottom right*; N=128, SNR=7.0) at all.

benefits. Although the original RVM implementation (Tipping, 2000, 2001) also operated by iteratively maximizing the log marginal likelihood $\mathcal{L}(\boldsymbol{\alpha})$, its cost was largely dominated by cubic (in M) operations on $M \times M$ matrices, where M is the *total* number of weights (viz. basis functions) in the model. By contrast the new implementation (Tipping and Faul, 2003), which we will discuss in some detail (as we base our extension of the probabilistic model of the RVM on an adapted version of it), still needs to perform expensive cubic operations at each iteration, only this time they are on matrices of size $S \times S$, where S is the number of currently non-zero weights w_m (viz. *active* basis functions), and since typically $S \ll M$, performance is often orders of magnitude better (see also the benchmarks in Tipping and Faul, 2003).

2.3 Shortcomings of the classical RVM

Although the RVM carries the benefits of a probabilistic formulation, unfortunately it still does not go far enough in its Bayesian encoding of the sparsity constraint — in practice one finds that in spite of (2.49), the choice of highly resolving dictionaries for data which do not need the many degrees of freedom offered by these dictionaries will still result in severe overfitting. This overfitting is illustrated in Figure 2.7 by using a symmlet wavelet basis for regression to the Sinc data set ($N = 128$, $\text{SNR} = 2$) (Tipping, 2001). As the top-left plot shows, the multi-scale nature of the symmlet dictionary results in drastic overfitting. Employing, for example, Gaussian or linear spline (lspline; pictured) basis functions[19] results in a good fit (bottom-left) and an apparently sparse solution: only 7 of the 128 available lspline basis functions are not switched off (have

[19]See sec. A.1 for details of the kernel functions; the lspline examples use $r = 3.0$.

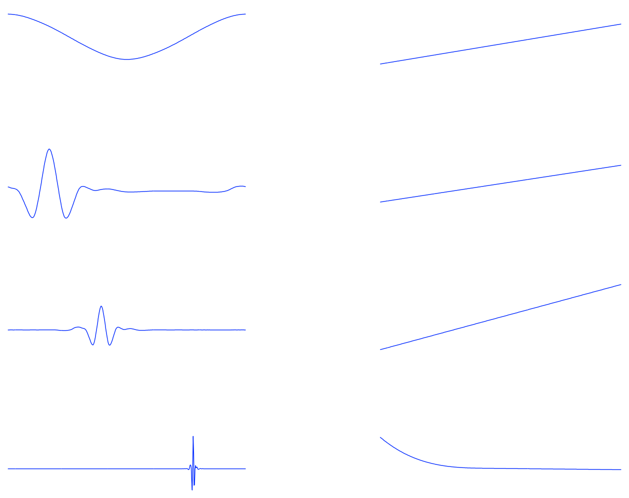


Figure 2.8: Basis functions 1, 10, 23, 230 from $N = 512$ symmlet (left) and lspline dictionaries (right). Whereas the symmlet dictionary contains components at all frequencies, the lspline dictionary only offers low-frequency components. This explains why the classical RVM's relatively weak sparseness enforcement suffices for lspline kernels, but not symmlets.

$\alpha_m < \infty$) compared with 127 symmlet basis functions.

Closer examination, however, reveals that the Gaussian and lspline bases, which do not contain high frequency basis functions, simply have difficulty fitting the noise. In the case of the Gaussian kernel (which with a kernel width that gives good results for the Sinc data yields a very ill-conditioned design matrix[20]), this is already apparent in the least squares estimate: choosing coefficients w_m to minimize $E = \|\mathbf{t} - \sum_m w_m \boldsymbol{\phi}_m\|$ yields a substantial error that is in fact larger than $\|\mathbf{y} - \sum_m w_m \boldsymbol{\phi}_m\|$. The case is a bit more subtle for the lspline kernel, because unlike the Gaussian kernel, a lspline kernel with a kernel width that gives good results for (classical) RVM regression can still exactly represent $\mathbf{t}$, just as the symmlet dictionary.

However, as illustrated by Figure 2.8, on comparing individual basis functions from both dictionaries, it becomes clear that whereas the symmlet basis offers multi-scale resolution and hence can fit a large proportion of the noise with relatively few components, a much greater number of the exclusively low-frequency basis functions in the lspline kernel are needed to fit a comparable proportion of the noise.

Consequently the relatively mild enforcement of sparsity of the classical RVM scheme, which proves insufficient for symmlet dictionaries, is already enough to prevent overfitting for lspline and Gaussian kernels.

The sparse regression provided by the RVM with lspline kernels thus partially depends on a propitious choice of kernel. Although the aforementioned gauss or lspline kernels are sufficiently resolving for the Sinc data, they cannot resolve data with genuine high frequencies such as the Bumps data (Donoho and Johnstone, 1994) which is therefore severely oversmoothed (bottom-right of Figure 2.7), although it poses no problems for the symmlet basis (top-right).

In other words, it is apparent that a crucial aspect of sparsity control (kernel choice) remains outside the principled probabilistic framework. Choosing kernel type (and in some cases width parameters) via cross validation-schemes is not just cumbersome and wasteful on data and computational resources, it also typically offers only a crude approach to sparsity control, making

[20]Condition number $\kappa = 6 \times 10^{18}$ for $N = 128$, $r = 3.0$. To see why the numeric behavior of Gaussian kernels with high kernel width r (which work well for low frequency signals as the Sinc data set) tends to be bad, consider the following: as the individual kernel rows, which consist of Gaussians centered on the diagonal, grow narrower (smaller kernel width r), the kernel matrix approaches const$\mathbf{I}$ (and thus condition number 1). Conversely, as the kernel width grows beyond a certain threshold, the kernel becomes numerically very badly behaved; it is clear that this has to happen eventually because with growing radius the increasingly stretched out Gaussians in each row become ever more similar.

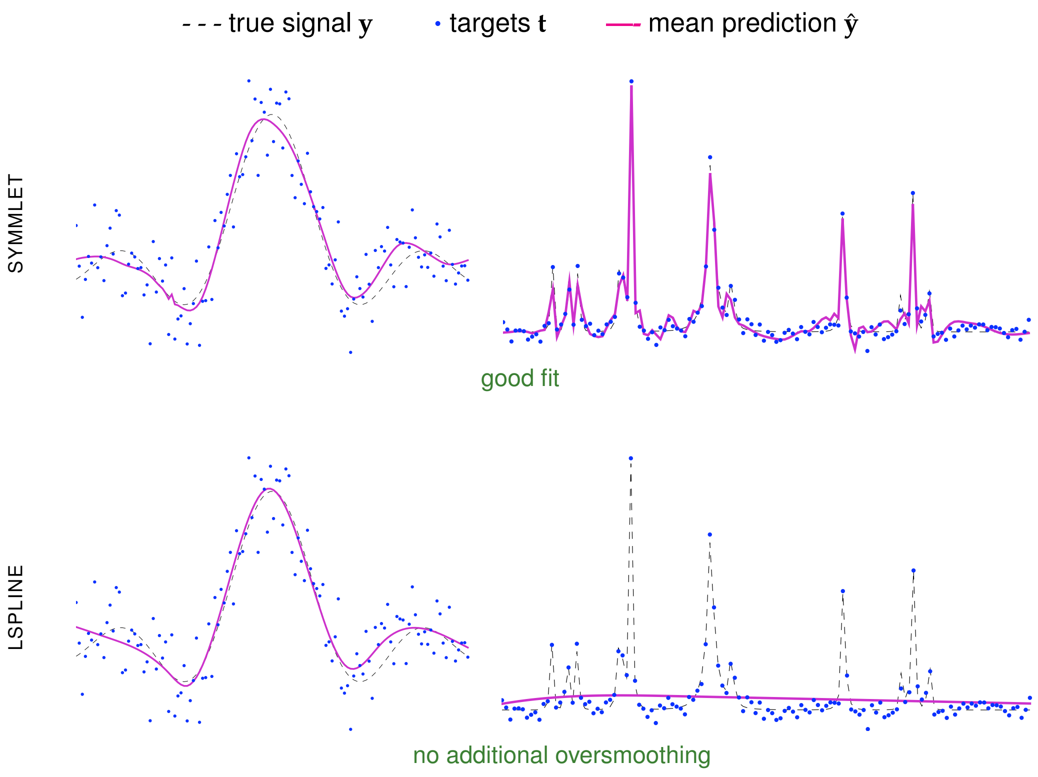


Figure 2.9: sRVM. The smoothness prior means that enforcing sparsity is no longer mostly relegated to the choice of kernel. A symmlet kernel (*top row*) no longer results in drastic overfitting for the Sinc data set (*on the left*). The bottom row shows that the smoothness prior typically has no adverse effect when smoothing is already mandated by the kernel. The data sets are identical to Figure 2.7.

it for example difficult or impossible to obtain good results with standard kernels for multi-resolution data (for a good example see Figure 2.12 later).

2.4 Amending the RVM; outlook and overview

Fortunately, a strength of Bayesian models is their inherent extensibility by means of additional prior structure; here we examine how to incorporate a wavelet-shrinkage inspired, noise-dependent *smoothness prior* for RVM models without degrading the efficiency of Tipping and Faul's (2003) fast RVM scheme (in fact performance can in many cases be *significantly* improved due to increased sparsity and the gained ability to obtain good results with wavelet dictionaries which allow efficient ($O(N)$) implementations of operations which are cubic in the general case). In brief, our prior is of the form, $p(\alpha_m \mid \sigma^2) \propto e^{-c/(1+\sigma^2\alpha_m)}$ (where c is a constant that controls the level of smoothing) and as is visible from inspection of Figure 2.9 (c.f. 2.7), or indeed the formula itself, it greatly promotes sparsity. The reason is that sparsity (i.e. small or more strictly zero weights w_m) is associated with large (or more strictly infinite) α_m and the smoothness prior can be seen to be monotonically increasing in α_m

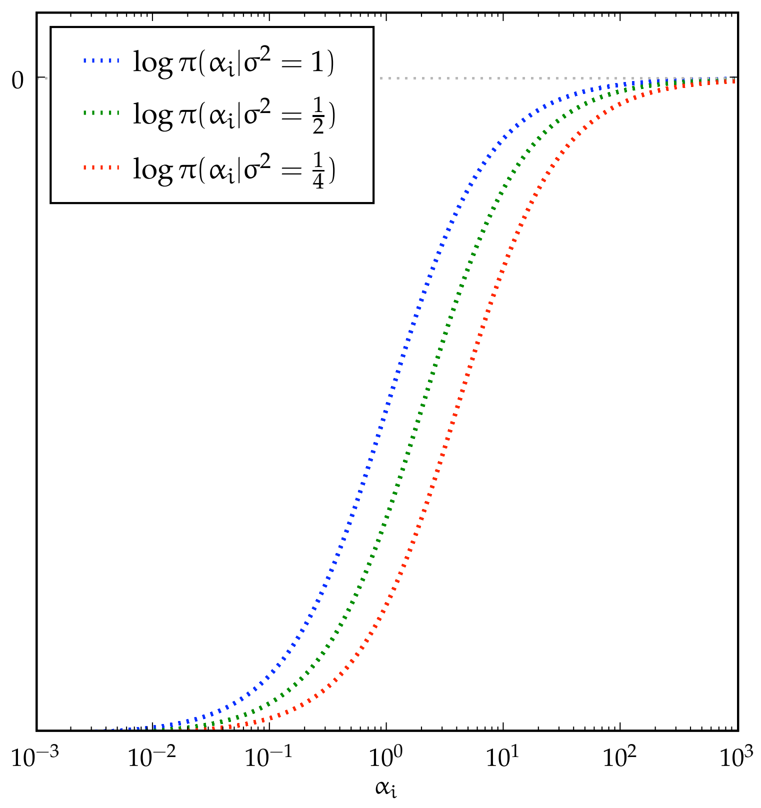


Figure 2.10: The smoothness prior in logspace: $\log \pi(\alpha_i \mid \sigma^2) = -c/(1 + \sigma^2\alpha_i)$. As it increases monotonically with α_i the prior encourages sparsity (associated with large/infinite values of α_i). The noise-dependence of the prior is illustrated by showing plots for different values of $\sigma^2(1/4, 1/2, 1$; $c = 1$ for all figures); higher noise is associated with more severe sparsity enforcement. This is as it should be, since (the unknown) noise $\boldsymbol{\epsilon}$ and (the unknown) underlying signal $\mathbf{y}$ effectively offer “competing explanations” for the observed data $\mathbf{t}$ and the smoothness prior helps to mediate the trade-off (Consider the two extremes: if there is no noise, no smoothing of $\mathbf{t}$ should take place in the computation of the posterior estimate $\hat{\mathbf{y}}$, which in that case should just be identical to $\mathbf{t}$; conversely if the observed data is all noise, the result should ideally be completely sparse, i.e. a flat line).

Having outlined the motivation for better prior-controlled sparsity control and briefly introduced our proposed prior in this section, we discuss our smoothness prior in more detail in section 2.5. In section 2.6 we show how the efficient scheme for learning the parameters in (Faul and Tipping, 2002) may be simply adapted to incorporate the smoothness prior. Results on a variety of standard data[21] sets (as well as some real-world EEG data) showing that it effectively controls sparsity are provided in section 2.6.5, followed by a discussion of alternative priors and the summary and conclusion in 2.8. More detailed theoretical discussions and proofs are relegated to the appendix. A preliminary report on this work appeared in (Schmolck and Everson, 2005) and (Schmolck and Everson, 2007) provides a more recent overview of the denoising and regression aspects.

[21]See also Appendix A.2.

2.5 The smoothness prior

Gaussians are not heavy-tailed, so on its own (2.49) does not appear to strongly favour sparsity (refer back to Figure 2.5), but of course the overall effect on the weights depends on the prior assigned to $\boldsymbol{\alpha}$. Here we consider only priors of the form $p(\boldsymbol{\alpha}, \sigma^2) = \prod_{m=1}^{M} p(\alpha_m \,|\, \sigma^2) p(\sigma^2)$. Then, the effective prior on w_m is found from:

$$p(w_m \,|\, \sigma^2) = \int p(w_m \,|\, \alpha_m) p(\alpha_m \,|\, \sigma^2) \, d\alpha_m \tag{2.51}$$

When the prior is a Gamma density, $p(\alpha_m \,|\, \sigma^2) = p(\alpha_m) = \Gamma(\alpha_m)^{-1} b^a \alpha_m^{a-1} e^{-b\alpha_m}$ with hyper-parameters a and b, then $p(w_m)$ is a Student-t density. Tipping (2001) presents a nice graphical illustration that the joint distribution $p(w_1, w_2)$ of two Student-t densities concentrates probability mass close to zero values of w_1 and w_2 rather than in regions where both w_1 and w_2 are non-zero, thus encouraging sparse solutions. In fact, the product $p(w_1)p(w_2)$ of any two super-Gaussian[22] prior densities in combination with a Gaussian noise model favours posterior solutions for which one or the other or both of w_1, w_2 are close to zero. This may be seen by noting that the log likelihood (2.48) is quadratic in $\mathbf{w}$ so that if $\log p(w_1, w_2) = \log p(w_1) + \log p(w_2) \approx w_1^q + w_2^q$ with $q < 2$ (and thus tails that decay slower than a Gaussian's), then as either coefficient moves away from the coordinate axis the log likelihood decays more rapidly than the log prior, thus encouraging a sparse posterior solution.

The expression (2.51) with the ARD prior (2.49) shows that $p(w_m)$ is a scale mixture of Gaussians and so under quite general conditions has positive kurtosis (Clarkson and Barrett, 2001; Lam and Goodman, 2000). It appears, therefore, since almost any prior on $\boldsymbol{\alpha} \,|\, \sigma^2$ will favour sparsity to some extent, that there is considerable freedom in its choice.

As it is empirically clear that the $p(\mathbf{w})$ resulting from a uniform $p(\boldsymbol{\alpha} \,|\, \sigma^2)$ (henceforward **None** prior) does not enforce sparsity strongly enough for flexible dictionary types (Figure 2.7), a well-founded, sparser prior over $\boldsymbol{\alpha} \,|\, \sigma^2$ is desirable. Since the question of existing and proposed prior types for the RVM is somewhat convoluted, we postpone a more extensive discussion till section 2.8.1, and concentrate for now on a smoothness prior.

As our desire for sparse $\mathbf{w}$ is ultimately grounded in beliefs about the complexity and structure of the signal $\mathbf{y}$, it is in a way natural to work one's way backwards, viz to fashion the prior $p(\boldsymbol{\alpha} \,|\, \sigma^2)$ so that the mean posterior prediction $\hat{\mathbf{y}}$ reflects these beliefs.

Given the posterior over the weights

$$\begin{aligned} p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) &= \frac{p(\mathbf{w}|\mathbf{t}, \sigma^2) p(\mathbf{w}|\boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)} \\ &= \mathcal{N}(\mathbf{w} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \end{aligned} \tag{2.52}$$

with

$$\boldsymbol{\Sigma} = (\sigma^{-2} \boldsymbol{\Phi}^T \boldsymbol{\Phi} + \mathrm{diag}(\boldsymbol{\alpha}))^{-1} \tag{2.53}$$

$$\boldsymbol{\mu} = \boldsymbol{\Sigma} \boldsymbol{\Phi}^T \sigma^{-2} \mathbf{t} \tag{2.54}$$

we obtain

$$\hat{\mathbf{y}} = \boldsymbol{\Phi} \boldsymbol{\mu} = (\boldsymbol{\Phi} \boldsymbol{\Sigma} \boldsymbol{\Phi}^T \sigma^{-2}) \mathbf{t} \equiv \mathbf{S} \mathbf{t} \tag{2.55}$$

where $\mathbf{S}$ is known as the *smoothing matrix* (Hastie and Tibshirani, 1990). Note that without the term $\mathrm{diag}(\boldsymbol{\alpha})$, which can be regarded as a regularization term, $\mathbf{St}$ would just be the projection of $\mathbf{t}$

[22] Densities whose tails decay more slowly than Gaussians.

into the column space of $\mathbf{\Phi}$, or equivalently, the least squares estimate

$$\hat{\mathbf{y}}_{\text{LSQ}} = \mathbf{\Phi}(\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t} = \mathbf{\Phi}\mathbf{\Phi}^{\dagger}\mathbf{t} \tag{2.56}$$

Thus $\mathbf{S}$ computes the *regularized* or smoothed projection of $\mathbf{t}$. Furthermore the special case where all α_i are identical is equivalent to ridge-regression (Hoerl and Kennard, 1970) with regularization parameter $\lambda = \sigma^2\alpha$ (the larger λ, the smoother the estimate, the more all w_i are *shrunk* towards zero compared to the least squares estimate). The "ridge-regression prior" $p(\mathbf{w}|\alpha) = \mathcal{N}(0, \alpha^{-1}\mathbf{I})$ is naturally also just a special case of the ARD prior (2.49). Of course indiscriminately shrinking coefficients for relevant as well as irrelevant basis functions is unattractive, but ridge regression has the convenient property that the amount of shrinking can be easily quantified.

However, even with an ARD prior, it is possible to quantify the degree of smoothing imposed by the model by a single number: the degrees of freedom[23] of $\mathbf{S}$, given by its trace:

$$\text{DF} = \operatorname{tr}\mathbf{S} \tag{2.57}$$

It is helpful to consider the case of orthonormal basis functions (such as wavelets) so that $\mathbf{\Phi}^T\mathbf{\Phi} = \mathbf{I}_M$ and the trace of the smoothing matrix is seen to be (using the identity $\operatorname{tr}\mathbf{ABC} = \operatorname{tr}\mathbf{CAB}$, see Appendix B.1):

$$\text{DF} = \operatorname{tr}\mathbf{S} = \sum_{m=1}^{M}(1 + \sigma^2\alpha_m)^{-1} \tag{2.58}$$

In (2.58) it is evident that basis functions with $\alpha_m \to \infty$ make no contribution to the degrees of freedom, whereas functions with $\alpha_m = 0$ contribute fully; DF thus counts the number of active basis functions, where the extent to which they are active is measured relative to the noise magnitude.

These equations make the related roles of $\boldsymbol{\alpha}$ and $\operatorname{tr}\mathbf{S}$ for sparsity control and smoothing apparent. As all $\alpha_m \to 0$ we approach the least squares estimate (2.56). In this case $\operatorname{tr}\mathbf{S} = N$, there is no smoothing and the model interpolates the data (indeed for orthonormal or invertible $\mathbf{\Phi}$, the least squares estimate is just $\mathbf{t}$). Conversely, as $\alpha_m \to \infty$ the corresponding component $\boldsymbol{\phi}_m$ is turned off ($w_m = 0$). Here $\operatorname{tr}\mathbf{S} = 0$ and the mean posterior estimate is zero.

As the model typically has roughly as many (for square $N \times M$ design matrix $\mathbf{\Phi}$, $M = N$) or more parameters (for overcomplete $\mathbf{\Phi}$, $N < M$) as training examples, the least squares estimate (all $\alpha_m \to 0$) will almost always result in drastic overfitting (never mind severe computational headaches[24]). Conversely all $\alpha_m \to \infty$ will just yield a constant prediction as all $w_m = 0$. But since the RVM associates an unique hyperparameter α_m with each weight w_m a suitable prior over $\boldsymbol{\alpha} \,|\, \sigma^2$ will bring about just the right amount of smoothing for each individual component when we maximize the posterior probability over the weights $p(\mathbf{w} \,|\, \mathbf{t}, \boldsymbol{\alpha}, \sigma^2)$. We expect most components to be turned off, hence most α_m to be ∞ and thus their corresponding weights w_m to be 0, but the few relevant components will have finite α_m and $w_m \neq 0$.

[23]More motivating details on the significance of the trace of $\mathbf{S}$ can be found in the literature, but the less diligent reader may be satisfied by considering the analogy to the projection matrix $\mathbf{H}$ in (2.12).

[24]Apart from numerical issues, the asymptotic complexity of all standard direct solutions are cubic in M (see, e.g. (Golub and van Loan, 1996)). However, sparse greedy matrix approximations might be used to enhance the convergence rates (Smola and Schmölkopf, 2000), and iterative schemes with improved convergence properties have also been developed; in particular backfitting (Hastie and Tibshirani, 1990) which D'Souza et al. (2004) adapted to a Bayesian EM framework to obtain an $O(MN)$ complexity RVM implementation.

2.5.1 Finding a suitable prior over α or wavelet shrinkage to the rescue

Similar observations lead Holmes and Denison (1999) to choose the following prior structure for encoding sparsity beliefs for the related problem of wavelet shrinkage:

$$p(\boldsymbol{\alpha} \,|\, \sigma^2) \propto e^{-c\mathrm{DF}} \tag{2.59}$$

Since DF may be regarded as the effective number of parameters in the regression problem with $\boldsymbol{\Phi}$ fixed, different choices for the hyperparameter c may be related to different classical model choice criteria (Holmes and Denison, 1999):

$$c = \begin{cases} 0 & \textbf{None}\text{, Bayes factor (so the classical RVM is just a special case)} \\ 1 & \textbf{AIC}\text{, Akaike}^{25}\text{ information criterion (Akaike, 1974)} \\ \log(N)/2 & \textbf{BIC}\text{, Bayesian information criterion (Schwarz, 1978)} \\ \log(N) & \textbf{RIC}\text{, Risk inflation criterion (Foster and Goerg, 1994)} \end{cases}$$

Thus we are left with 4 different weight variance priors, from least smoothing to most smoothing as follows: **None, AIC, BIC, RIC**.

Using (2.58) to compute DF even in the non-orthogonal case yields a convenient prior expression for an individual α_i that does not depend on any of the other $\alpha_{j\neq i}$ and we adopt this form throughout:

$$p(\boldsymbol{\alpha} \,|\, \sigma^2) \propto e^{-c\sum_{m=1}^{M}(1+\sigma^2\alpha_m)^{-1}} \simeq e^{-c\mathrm{DF}} \tag{2.60}$$

This approximation finds justification beyond computational and analytical expediency. Firstly, we have obtained good empirical results (low MSEs) with this prior even when orthonormality is not present (e.g. with various spline kernels and even with overcomplete dictionaries with $M = 2N$; see e.g. Figure 2.14). Secondly, we also ran some tests where, during model runs, we simultaneously computed the true DF expression and compared it to the approximation above and have obtained very similar results. Thirdly our prior structure exerts strong pressure to exclude[26] redundant components, hence although the basis functions might not be orthogonal the eventually *included* basis functions can be expected to be typically near-orthogonal.

A noteworthy and distinguishing characteristic of the smoothness prior is its noise dependency. The degrees of freedom amounts to a count of the number of active basis functions. As the noise increases, the DF decrease, i.e. $\mathbf{S}$ becomes more strongly smoothing. This is what one would intuitively expect to happen: everything else staying fixed, if there is no noise ($\sigma^2 = 0$) then the observations $\mathbf{t}$ ought to equal the true signal $\mathbf{y}$ and so should the posterior estimate $\hat{\mathbf{y}}$, thus $\mathbf{S}$ must be the identity. However as the level of noise increases, more and more of the targets $\mathbf{t}$ has to be explained by the noise and hence $\mathbf{S}$ should become more smoothing as the noise level increases.

Note that this means that basis functions with smaller α_m have greater prior support when the noise is larger, which may appear counter-intuitive at first.

Although (2.60) is an improper prior[27], a proper prior may be obtained by restricting α_m to a

[26] i.e. to set the corresponding $\alpha_i = \infty$, peek ahead to sec. 2.6.1 (2.59, 2.60) to see that the corresponding $\boldsymbol{\phi}_i$ play indeed no role

[27] Meaning that the integral diverges and hence cannot be normalized to obtain a valid pdf that integrates to 1. This is easy to see because (taking $M = 1$ for simplicity), $e^{-c(1+\sigma^2\alpha_1)^{-1}}$ is positive ($\geqslant e^{-c}$ to be precise) and monotonically increasing in α_1 for any choice of $c, \sigma^2 \in \mathbb{R}_+$ and $\int_0^U e^{-1}\,d\alpha_1$ diverges as $U \to \infty$.

finite range $[L, H]$. In this case we may write:

$$p(\boldsymbol{\alpha} \mid \sigma^2) = \begin{cases} Ze^{-c\sum_{m=1}^{M}(1+\sigma^2\alpha_m)^{-1}} & L \leqslant \alpha_m \leqslant H \\ 0 & \text{otherwise} \end{cases} \tag{2.61}$$

and the normalization constant Z is given by

$$\begin{aligned} Z =& (\sigma^{-2} + H)\exp(-c/(1+\sigma^2 H)) - (\sigma^{-2} + L)\exp(-c/(1+\sigma^2 L)) \\ &+ c\sigma^{-2}[\mathrm{Ei}(c/(1+\sigma^2 L)) - \mathrm{Ei}(c/(1+\sigma^2 H))] \end{aligned} \tag{2.62}$$

where $\mathrm{Ei}(x) = \int_{-\infty}^{x} \frac{e^t}{t}\, dt$ denotes the exponential integral function (Arfken, 1985). In the work reported here we choose $L = 10^{-10}$, $H = 10^{10}$. This expression is however only needed when we would like to obtain the posterior probability of the result (see A.7 for how to do so efficiently). Furthermore we note that when $\sigma^2 H/c \gg 1$ then $Z \to H$, that for a large range of σ^2 values Z stays essentially constant which makes numerically finding the MAP value for σ^2 easier (see section 2.6.3).

To summarize: The smoothness prior clearly favours large α_m, thus encoding a belief that the weight w_m should be close to zero and consequently a sparse solution. The prior has a desirable dependency on the noise. The constant c controls the smoothness prior's severity (with the least severe prior $c = 0$ reducing to the classical RVM's uniform prior). It is also easy to prove that this prior results in a scale invariant posterior (i.e. our model will give the same answers if we rescale $\mathbf{t} \to k\mathbf{t}$ and simultaneously $\sigma \to k\sigma$, see A.3).

Apart from the observations $\mathbf{t}$ and the choice of design matrix $\boldsymbol{\Phi}$, the hyperparameters g, h and c are the only parameters to be externally specified by the user. Moreover we find that $c = \log(N)/2$, the value for **BIC** makes a good default for c, while we can generally, in the absence of any prior belief about the likely shape of $p(\sigma)$, just set g, h to 10^{-4} or some other uninformative value.

How to effectively learn the other parameter values and estimates is the topic of the next section.

2.6 An Implementation based on Tipping and Faul's fRVM

For clarity we first recapitulate the decomposition of the log marginal likelihood used by Tipping and Faul's original "fast RVM" scheme (fRVM) (Tipping and Faul, 2003) before we describe the modifications needed to incorporate the smoothness prior. The following subscripts will be used: $_S$ denotes the value of a variable with only the S selected components included whereas $_{-i}$ denotes the value of a variable with the ith component removed.

With the **None** prior and uniform $p(\boldsymbol{\alpha} \mid \sigma)$ maximization of $\log p(\boldsymbol{\alpha} \mid \sigma, \mathbf{t})$ is equivalent to maximizing the log marginal likelihood $\mathcal{L}(\boldsymbol{\alpha}) = \log p(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma)$.

2.6.1 Maximizing the marginal likelihood

The key idea is to write

$$\mathcal{L}(\boldsymbol{\alpha}) = \mathcal{L}(\boldsymbol{\alpha}_{-i}) + \ell(\alpha_i) \tag{2.63}$$

in order to separate out the contribution of the ith basis function $\boldsymbol{\phi}_i$ into the term $\ell(\alpha_i)$ which depends solely on α_i and a term $\mathcal{L}(\boldsymbol{\alpha}_{-i})$ that is independent of α_i. Maximization of $\mathcal{L}(\boldsymbol{\alpha})$ then proceeds by successive maximizations of $\ell(\alpha_i)$ for a sequence of components.

The log marginal likelihood

$$\begin{aligned}\mathcal{L}(\boldsymbol{\alpha}) &= \log p(\mathbf{t} \,|\, \boldsymbol{\alpha}, \sigma^2) && (2.64)\\ &= \log \int p(\mathbf{t} \,|\, \mathbf{w}, \sigma^2) p(\mathbf{w} \,|\, \boldsymbol{\alpha}) d\mathbf{w} && (2.65)\\ &= \log \int \mathcal{N}(\mathbf{t} \,|\, \boldsymbol{\Phi}\mathbf{w}, \sigma^2) \mathcal{N}(\mathbf{w} \,|\, \mathbf{0}, \operatorname{diag} \boldsymbol{\alpha}^{-1}) d\mathbf{w} && (2.66)\\ &= \log \mathcal{N}(\mathbf{t} \,|\, \mathbf{0}, \mathbf{C}) && (2.67)\\ &= -\frac{1}{2}\left[N \log 2\pi + \log |\mathbf{C}| + \mathbf{t}^T \mathbf{C}^{-1} \mathbf{t}\right] && (2.68)\end{aligned}$$

where

$$\mathbf{C} = \sigma^2 \mathbf{I} + \boldsymbol{\Phi} \operatorname{diag}(\boldsymbol{\alpha}^{-1}) \boldsymbol{\Phi}^T \qquad (2.69)$$

can be decomposed to isolate the terms involving a particular α_i. Writing

$$\begin{aligned}\mathbf{C} &= \sigma^2 \mathbf{I} + \sum_{m \neq i} \alpha_m^{-1} \boldsymbol{\phi}_m \boldsymbol{\phi}_m^T + \alpha_i^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T && (2.70)\\ &\equiv \mathbf{C}_{-i} + \alpha_i^{-1} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^T && (2.71)\end{aligned}$$

the log likelihood may be reformulated, using standard matrix identities (see Appendix B.1) for inverse and determinant of $\mathbf{C}$, as (2.63) where:

$$\mathcal{L}(\boldsymbol{\alpha}_{-i}) = -\frac{1}{2}\left[N \log 2\pi + \log |\mathbf{C}_{-i}| + \mathbf{t}^T \mathbf{C}_{-i}^{-1} \mathbf{t}\right] \qquad (2.72)$$

and

$$\ell(\alpha_i) = \frac{1}{2}\left[\log \alpha_i - \log(\alpha_i + s_i) + \frac{q_i^2}{\alpha_i + s_i}\right] \qquad (2.73)$$

The quantities

$$s_i \equiv \boldsymbol{\phi}_i^T \mathbf{C}_{-i}^{-1} \boldsymbol{\phi}_i \qquad (2.74)$$

and

$$q_i \equiv \boldsymbol{\phi}_i^T \mathbf{C}_{-i}^{-1} \mathbf{t} \qquad (2.75)$$

respectively measure the degree to which $\boldsymbol{\phi}_i$ overlaps other basis functions in the solution (its "sparsity") and its "quality," namely its correlation with the model error with $\boldsymbol{\phi}_i$ excluded: $q_i = \sigma^{-2} \boldsymbol{\phi}_i^T (\mathbf{t} - \hat{\mathbf{y}}_{-i})$.

Now it is easy to maximize $\mathcal{L}(\boldsymbol{\alpha})$ with respect to α_i. Faul and Tipping (2002) show that $\ell(\alpha_i)$ has a single unique maximum at

$$\alpha_i^\star = \begin{cases} \frac{s_i^2}{q_i^2 - s_i} & \text{if } q_i^2 > s_i \\ \infty & \text{otherwise} \end{cases} \qquad (2.76)$$

Kernels for which $q_i^2 < s_i$ are effectively excluded from the model and the elegance of the Tipping and Faul (2003) fast RVM scheme derives from the fact that s_i and q_i can be calculated from quantities involving only the $S \ll M$ included components.

2.6.2 Overview of MAP implementation properties and strategy

The efficient calculation of MAP point estimates for the model parameters that we are about to detail is based on the "fast RVM" scheme and rests mostly on three facts:

1. For all data sets we experimented upon, our sparsity prior ensured that the posterior peaks in regions with mostly infinite α_m and as discussed an infinite value for α_m is equivalent to the exclusion of ith component from the model, so only $S \ll M$ of the coefficients w_m were nonzero. We expect this finding to generalize to other data sets, provided the smoothness prior is used in conjunction with a dictionary that allows sparse representation of the underlying signal in question.

2. Although the introduction of this sparsity prior structure means that some expressions no longer have convenient closed form solutions, the solutions are still easily and efficiently found by simple numerical methods in all instances and all the important desirable properties of the fast RVM that are detailed in (Faul and Tipping, 2002) remain unaffected by the inclusion of the smoothness prior.

3. In particular it is still possible to determine the relevance of a basis function not currently included in the model (so that components can be included one by one, starting with an empty model) and to derive expressions for all quantities of interest that only depend on S and not M. Consequently, the computational complexity scales cubically with the number of included components S, rather than the number of basis functions M.

As we have seen, for the original RVM, which corresponds to the **None** prior, maximization of the posterior is equivalent to maximizing the likelihood, which can be efficiently effected by the type II maximum likelihood scheme by Tipping and Faul (2003) which we just described.

In this case the maximizing $\alpha_i^\star = \operatorname{argmax}_{\alpha_i} \ell(\alpha_i)$ can be found particularly simply and cheaply, because there is an analytic solution. However in our case, the addition of the smoothness prior means that rather than the log likelihood, we seek to maximize the log posterior:

$$\hat{\mathcal{L}}(\boldsymbol{\alpha}) \equiv \log p(\mathbf{t} \mid \boldsymbol{\alpha}, \sigma^2) + \log p(\boldsymbol{\alpha} \mid \sigma^2) \tag{2.77}$$

$$= \mathcal{L}(\boldsymbol{\alpha}) + \log p(\boldsymbol{\alpha} \mid \sigma^2) \tag{2.78}$$

Due to the multiplicative prior structure, $p(\boldsymbol{\alpha} \mid \sigma^2) = \prod_i p(\alpha_i \mid \sigma^2)$ the dependence of $\hat{\mathcal{L}}(\boldsymbol{\alpha})$ on α_i can still be isolated, although the additional term requires that the optimal

$$\hat{\alpha}_i = \underset{\alpha_i}{\operatorname{argmax}}\, \hat{\ell}(\alpha_i) = \underset{\alpha_i}{\operatorname{argmax}}\, [\ell(\alpha_i) + \log p(\alpha_i \mid \sigma)] \tag{2.79}$$

is found numerically, rather than analytically as in Tipping and Faul (2003). Before we proceed to further discuss both the original, analytical, maximization scheme and our numerical extension for the smoothness prior, we would like to remark that our extension is straightforward and has desirable properties. In particular:

1. There still is at most one local maximum for $\ell(\alpha_i)$.

2. $\hat{\alpha}_i \geqslant \alpha_i^\star$, in other words the sRVM or smoothness prior MAP solution is always at least as sparse as the RVM or ML solution (see appendix A.4), but typically is much sparser for flexible kernels.

3. It adds virtually no computational overhead, but allows enormous computational savings in many cases – this is because the complexity of a single step in the model is essentially $O(S^3)$ and the sparsity prior will always produce at least as small an S as the **None** prior,

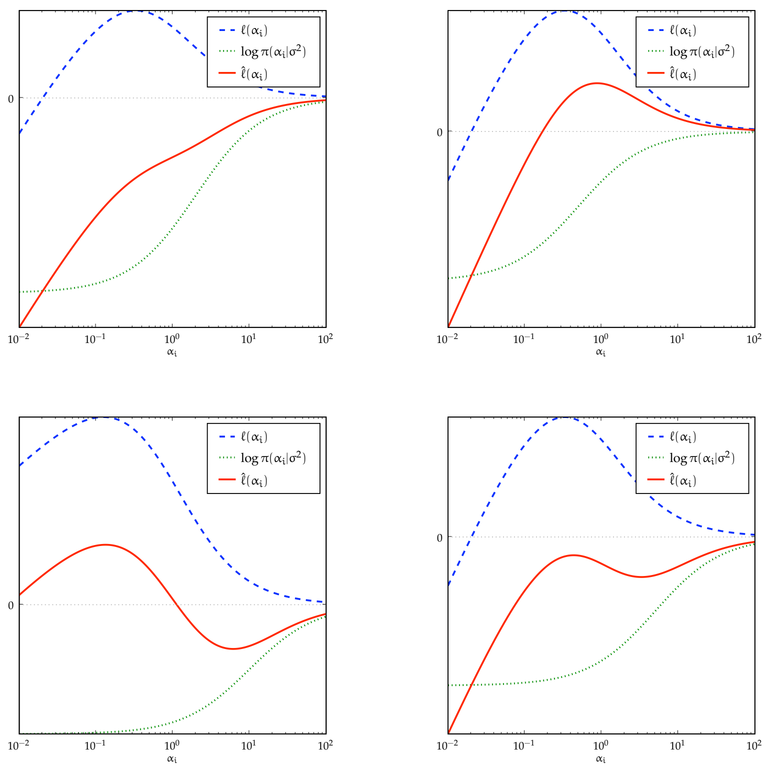


Figure 2.11: Log posteriors $\hat{\ell}(\alpha_i)$ (solid), log likelihoods $\ell(\alpha_i)$ (dashed), and log priors $-c(1+\sigma^2\alpha_i)^{-1}$ (dotted) plotted versus $\log \alpha_i$ for the four possible scenarios of how the addition of the smoothness prior affects the pre-existing mode of $\ell(\alpha_i)$. Firstly since the (s)RVM is based on MAP maximization and, in particular, in the case of the fast RVM implementation (Tipping and Faul, 2003), on the maximization of $\hat{\ell}(\alpha_i)$ wrt α_i, one would hope to see that the introduction of the smoothness prior does not introduce additional (finite) modes (which could hamper optimization or even introduce ambiguities). Secondly, since the aim of the smoothness prior is to more stringently enforce sparsity, and since sparsity is associated with large (or more strictly, infinite) α_i, one would further hope to find that the mode of $\hat{\ell}(\alpha_i)$ (viz the mode of the sparsity-prior-enhanced version of $\ell(\alpha_i)$) is always located on the right of $\ell(\alpha_i)$. This is indeed the case, and the possible scenarios are illustrated in the above four panels: *Top-left:* Prior nulls maximum in posterior: the addition of the smoothness prior removes the finite maximum, so that the optimal α_i is now ∞. *Top-right:* Single turning point with $\hat{\alpha}_i$ finite: the addition of the smoothness prior still gives a finite mode, but the regression result will still a little bit smoother since the mode is shifted to the right. *Bottom-left and bottom-right:* although two turning points can occur in the posterior, at most one mode will be finite: the single finite mode can either be larger than $0 = \lim_{\alpha\to\infty} \hat{\ell}(\alpha_i)$ (*bottom-left*) or smaller (*bottom-right*).

but often only a fraction. The increased sparsity also makes it feasible to use wavelet design matrices for many tasks without fear of overfitting which further reduces the complexity per step to $O(N)$, because all matrix multiplications disappear[28]; we have empirically verified that approximately linear in N per-step behaviour obtains in our implementation for $4096 \leqslant N \leqslant 524288$.

We now give further details of the maximization scheme, although proofs are relegated to the appendix.

Maximization of the log marginal posterior with respect to a single α_i can be achieved by maximizing

$$\hat{\ell}(\alpha_i) = \ell(\alpha_i) - \frac{c}{1+\sigma^2\alpha_i}. \tag{2.80}$$

[28]The covariance matrix becomes the identity due to orthonormality, $\boldsymbol{\Sigma}$ becomes diagonal and other multiplications by $\boldsymbol{\Phi}$ is can be replaced by the equivalent, but much more efficient, discrete wavelet transform.

The derivative of $\hat{\ell}(\alpha_i)$ wrt. α_i is

$$\hat{\ell}'(\alpha_i) = \frac{1}{2}\left[\frac{1}{\alpha_i} - \frac{1}{\alpha_i + s_i} - \frac{q_i^2}{(\alpha_i + s_i)^2}\right] + \frac{c\sigma^2}{(1+\sigma^2\alpha_i)^2} \tag{2.81}$$

$$= \frac{P(\alpha_i)}{2\alpha_i(\alpha_i + s_i)^2(\alpha_i + \sigma^{-2})^2} \tag{2.82}$$

where $P(\alpha_i)$ is cubic in α_i and given by :

$$\begin{aligned} P(\alpha_i) =& \alpha_i^2 s_i^2 \sigma^4 + \alpha_i^3 s_i \sigma^4 - \alpha_i^3 q_i^2 \sigma^4 + 2\alpha_i c s_i^2 \sigma^2 + 2\alpha_i s_i^2 \sigma^2 + 4\alpha_i^2 c s_i \sigma^2 + 2\alpha_i^2 s_i \sigma^2 \\ & - 2\alpha_i^2 q_i^2 \sigma^2 + 2\alpha_i^3 c \sigma^2 + s_i^2 + \alpha_i s_i - \alpha_i q_i^2 \end{aligned} \tag{2.83}$$

Simple closed form solutions to the roots of $P(\alpha_i) = 0$ are not available, however it is simple and computationally cheap to numerically find the roots of P. Since $\lim_{\alpha_i\to\infty} \hat{\ell}(\alpha_i) = 0$ the maximum of $\hat{\ell}(\alpha_i)$ may occur at infinite α_i, corresponding to the ith basis function being "switched off". A basis function ϕ_i is active if the maximum occurs for $\alpha_i < \infty$. With the smoothness prior the posterior may have more than one turning point but, as shown in Appendix A.4, there can be at most one maximum $\hat{\alpha}_i < \infty$ and $\alpha_i^\star < \hat{\alpha}_i$ showing that the smoothness prior always has the effect of making the solution sparser. Figure 2.11 shows the four possible cases that may arise. Most severely, (*top-left*) the prior can null the maximum in the likelihood. Alternatively, (*top-right*) the posterior has a, possibly local, maximum at finite $\hat{\alpha}_i > \alpha_i^\star$; when there is a single turning point $\hat{\alpha}_i > \lim_{\alpha_i\to\infty} \hat{\ell}(\alpha_i) = 0$ and the basis function is active; if there are two turning points in $\hat{\ell}(\hat{\alpha}_i)$ the global maximum may be at the turning point (*bottom-left*) or at infinite α_i. It is straightforward during learning to distinguish between these last two cases by evaluating $\hat{\ell}(\hat{\alpha}_i)$.

In brief, maximization of $\hat{\mathcal{L}}(\boldsymbol{\alpha})$ therefore proceeds by successively choosing (at random) a component i to include in the model, maximizing $\hat{\ell}(\alpha_i)$ with respect to α_i and reestimating the parameters $\boldsymbol{\Sigma}_S$, $\boldsymbol{\mu}_S$, $\mathbf{s}$ and $\mathbf{q}$ which depend upon α_i ($\mathbf{s}$ and $\mathbf{q}$ are M vectors of the sparsity and quality indices q_i and s_i of the corresponding α_i). Since the posterior is increased by maximization of each individual $\hat{\ell}(\alpha_i)$ such a sequence of maximizations terminates when a, possibly local, maximum of the posterior is located at which the inclusion or deletion of any single component can only decrease the $\hat{\mathcal{L}}(\boldsymbol{\alpha})$. Tipping and Faul (2003) use this computationally efficient sequential maximization as the basis of the *fast* RVM. Although in (Faul and Tipping, 2002) an attempt is made to prove that "sequential optimization of individual α_i cannot lead to a stationary point from which a joint maximization over all $\boldsymbol{\alpha}$ may have escaped", it appears that the proof is flawed and this desirable property may indeed often *not* hold. However, by examining the Hessian of $\hat{\mathcal{L}}(\boldsymbol{\alpha})$ at the maximum it is straightforward to show that this property does hold for orthonormal basis functions (such as wavelet design matrices) regardless of the imposition of a smoothness prior (Appendix A.5).

2.6.3 Noise reestimation

Again, whereas the classical RVM can employ an analytical update rule (Tipping, 2001, eq 46) for σ^2 (by setting $\frac{\partial\mathcal{L}(\boldsymbol{\alpha},\sigma^2)}{\partial\sigma^{-2}} = 0$, and solving for σ^2), the introduction of the smoothness prior term means we have to resort to a numerical scheme. As noted above, provided that $\sigma^2 H/c \gg 1$ the normalisation term in the prior Z (2.62) is effectively constant, and we therefore numerically

Algorithm 1 The sRVM algorithm. The differences to the plain RVM algorithm are blue.

```
1:  $\sigma^2 \leftarrow 0.1 \times \mathrm{var}(\mathbf{t})$                          initialization for $\sigma^2$
2:  $\boldsymbol{\alpha} \leftarrow [\infty \cdots \infty]^T$                                  start with the empty model
3:  $i \leftarrow \mathrm{argmax}\,(\|\boldsymbol{\phi}_m^T \mathbf{t}\| / \|\boldsymbol{\phi}_m\|)$                pick an i that stands a good chance of being relevant
4:  $\mathcal{S} \leftarrow \{i\}$                                               include it in the set of included components
5:  $\mathrm{update}_{\mathrm{sRVM}}(\alpha_i, \boldsymbol{\Sigma}_S, \boldsymbol{\mu}_S, \mathbf{s}, \mathbf{q})$             compute initial values for all model parameters
6:  $R \leftarrow 10$                                                reestimate noise every $R$ steps
7:  step $\leftarrow 1$                                              already made the first step
8:  until converged()
9:      $i \leftarrow \mathrm{randint}(M)$                                    pick a random component $i$
10:     DID-NOTHING $\leftarrow$ False
11:     if ( $q_i^2 - s_i > 0$ and ...
          has-real-positive-root(numerator($\hat{\ell}'(\alpha_i)$)))   if component $i$ is relevant
12:         unless $\alpha_i < \infty$                                 unless it is already included
13:             $\mathcal{S} \leftarrow \mathcal{S} \cup \{i\}$                             add it
14:     else
15:         if $\alpha_i < \infty$                                    the component is irrelevant but currently included
16:             $\mathcal{S} \leftarrow \mathcal{S} \setminus \{i\}$                             delete it
17:         else                                             component is and was irrelevant
18:             DID-NOTHING $\leftarrow$ True                       no need for action
19:     unless DID-NOTHING                                   otherwise update everything
20:         step $\leftarrow$ step + 1
21:         $\mathrm{update}_{\mathrm{sRVM}}(\alpha_i, \boldsymbol{\Sigma}_S, \boldsymbol{\mu}_S, \mathbf{s}, \mathbf{q})$         update the model parameters
22:         if step mod $R = 0$
23:             $\mathrm{reestimate}_{\mathrm{sRVM}}(\sigma^2)$
```

solve[29] :

$$\frac{\partial \hat{\mathcal{L}}(\boldsymbol{\alpha}, \sigma^2)}{\partial \sigma^{-2}} \approx \frac{1}{2}\left[N\sigma^2 - \|\mathbf{t} - \boldsymbol{\Phi}\boldsymbol{\mu}\|^2 - \sigma^2 \sum_m (1 - \alpha_m \boldsymbol{\Sigma}_{[m,m]})\right] - c \sum_m \frac{\alpha_m}{(\sigma^{-2} + \alpha_m)^2} + (g-1)\sigma^2 - h = 0 \quad (2.84)$$

Good initial guesses for starting the numerical solution are either the $c = 0$ linear solution or the previous value of σ^2, and we find that convergence is rapid. A graphical analysis of (2.84) shows that the smoothness prior term serves always to increase the effective noise variance as may be expected because a sparser solution requires more of the error to be explained by noise. However, for even moderate amounts of data, we find that the estimate of σ^2 is dominated by the marginal likelihood terms, being insensitive to the noise prior $p(\sigma^2|g, h)$ and hence the choices for g and h.

2.6.4 The algorithm

The outline of the algorithm, based on the fRVM algorithm, is as follows (c.f. pseudocode in Alg. 1).

We start out with a model that includes only a single component (a good default choice is the component that has the largest projection onto the target $\mathbf{t}$) (lines 1-7). Then, until the model has converged (line 8), at each iteration a candidate component is picked at random (line 9). If upon testing this component turns out to be neither currently included (i.e. $\alpha_i = \infty$) nor relevant (i.e. inclusion would not increase $\hat{\ell}(\alpha_i)$ or equivalently the overall posterior $\hat{\mathcal{L}}(\boldsymbol{\alpha})$) nothing is done (line 18f). In all other cases the model is updated: α_i and all other relevant parameters are reestimated (line 21); only the noise estimation is not updated on each step (lines 22f) to prevent spurious oscillation.

There are thus 4 possible cases that can occur after the relevance of some component i given

[29]Note that we use explicit subscripting, $\boldsymbol{\Sigma}_{[m,m]}$ to denote the element at $\boldsymbol{\Sigma}$'s m-th row and m-th column, in favour of following our usual convention σ_{mm}. This is in order to avoid confusion with the noise standard deviation σ.

the current state of the model has been established:

- The component is currently not included but is still deemed irrelevant, so nothing happens.
- The component is currently not included, but since inclusion would increase $\hat{\ell}$, it is included (line 13).
- The component is currently included and α_i is updated to reflect the current state of the model which can either mean:
  - deletion: α_i is set to ∞ if doing so does not reduce $\hat{\ell}$ (line 16);
  - (mere) reestimation: the value of α_i is set to some finite value, possibly the same that it already had (line 21).

It should be noted that whereas only a single α_i of all the $\boldsymbol{\alpha}$ is updated on each step, *all* the other parameters, i.e. all the M q_m and s_m as well as the S elements of $\boldsymbol{\mu}_S$ and $S \times S$ elements of $\boldsymbol{\Sigma}_S$ are completely recalculated. The following efficient formulas are used to calculate $\mathbf{q}$ and $\mathbf{s}$ (Tipping and Faul, 2003):

$$s_m = \frac{\alpha_m S_m}{\alpha_m - S_m} \tag{2.85}$$

$$q_m = \frac{\alpha_m Q_m}{\alpha_m - S_m} \boldsymbol{\Sigma}_S \boldsymbol{\Phi}_S^T \boldsymbol{\phi}_m \tag{2.86}$$

$$S_m \equiv \sigma^{-2} \boldsymbol{\phi}_m^T \boldsymbol{\phi}_m - \sigma^{-4} \boldsymbol{\phi}_m^T \boldsymbol{\Phi}_S \tag{2.87}$$

$$Q_m \equiv \sigma^{-2} \boldsymbol{\phi}_m^T \mathbf{t} - \sigma^{-4} \boldsymbol{\phi}_m^T \boldsymbol{\Phi}_S \boldsymbol{\Sigma}_S \boldsymbol{\Phi}_S^T \mathbf{t} \tag{2.88}$$

The convergence criterion

Due to the greedy nature and itemwise update of the algorithm finding, a good convergence criterion requires a bit of tweaking to prevent premature convergence while at the same time avoiding endless iterations close to the solution.

We follow Tipping and Faul (2003) in requiring that the differences between successful values for any α_i in logarithmic space must be less than 10^{-6} and that all components currently deemed to be relevant are actually included in the model.

Whilst this suffices in many cases, we have found it useful to add some additional requirements and as a consequence in practice the scheme is a bit more complex than that depicted in Alg. 1. Most importantly, before we declare convergence we ensure that all component α_m are re-evaluated and that the noise reestimation remains stable over several past estimates. We also test that $\hat{\mathcal{L}}$ no longer increases noticeably. Full details may be found in the implementation which is available from `<http://www.dcs.ex.ac.uk/~reverson/sRVM>`.

2.6.5 Results

Simple data

As Figure 2.9 shows, we find that use of the smoothness prior typically yields substantial improvements for tasks where overfitting is a problem due to the multi-scale resolution of the design matrix, while it generally has no appreciable negative impact when overfitting is not an issue.

Multiscale data

In Figure 2.12 we can clearly see the advantages of smoothness control via prior structure as opposed to kernel choice: with the sRVM a multiscale signal can receive just the right level of

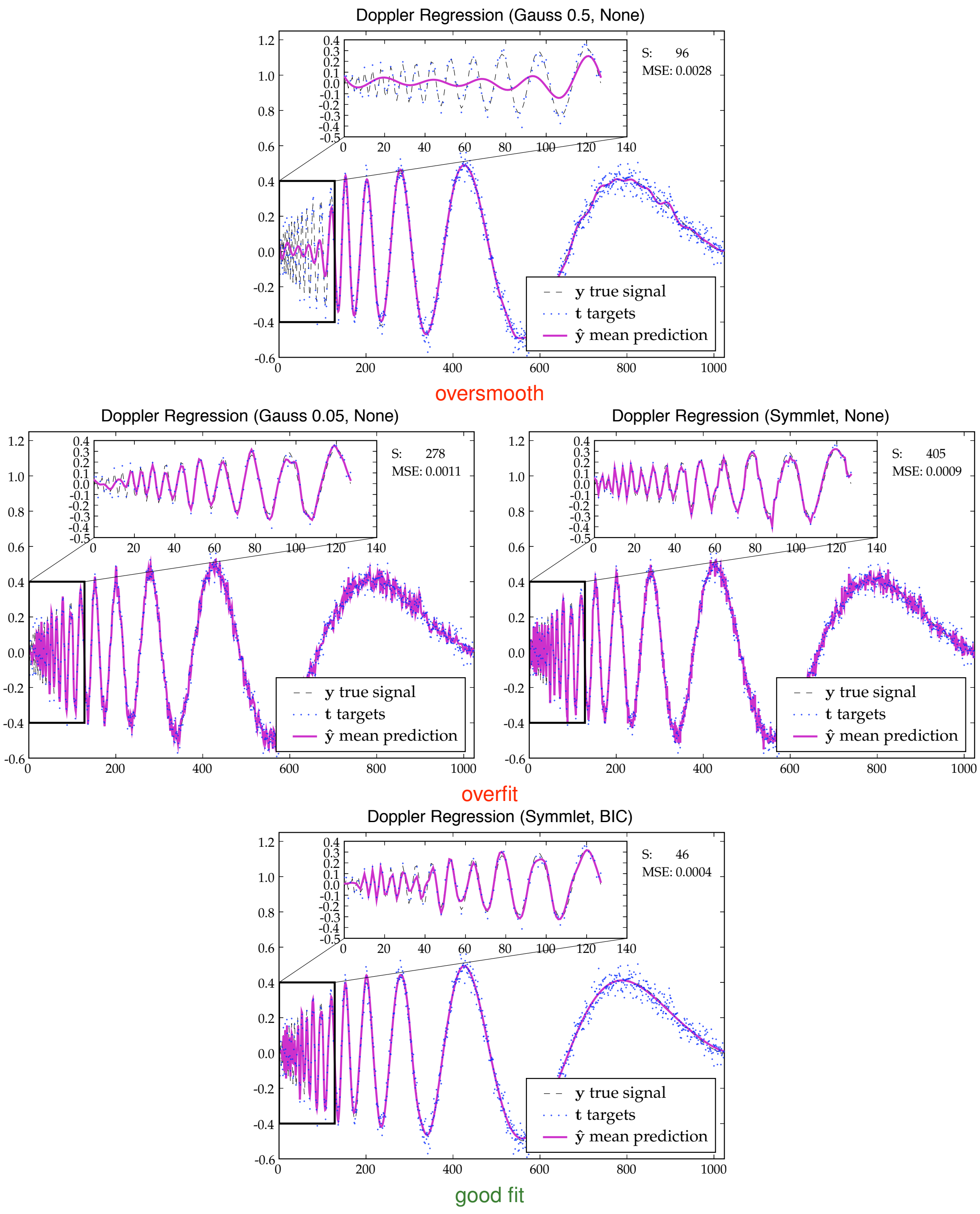


Figure 2.12: The shortcomings of smoothness control via kernel choice (RVM, *top, middle*) in comparison to smoothness control via prior choice (sRVM, *bottom*), illustrated by denoising data with multiscale resolution (identical data set in all panels: Doppler data, $N = 1024$, $SNR = 7.0$). The Doppler data is characteristically multiscale: it goes from very high frequency on the very left side of the picture to mid and low frequencies towards the right end. In order to fully appreciate the result of a particular kernel/prior combination, the effects on both the high frequency part (the first 128 of the N data points – shown magnified in the inset at the top of each figure) and the mid/low frequency part (the remainder) should be studied: ideally both should show a good fit. The three panels in the *top* and *middle* rows were created with the classical RVM (i.e. None prior) and differ by kernel choice alone (Gaussian kernel with kernel width $r = 0.5$ in the *top* panel, and $r = 0.05$ in the *left middle* panel; symmlet in the *right middle* panel). They all show a bad fit: the *top* panel shows oversmoothing (visible in the inset) whereas the two *middle* panels show overfitting (visible particularly towards the right of each panel). The problem is that smoothness control via kernel width or type acts globally, whereas only part of the signal is respectively fine scale/large scale. Thus even though overfitting already starts to become apparent in the *top* panel, the fine scale information on the left side is still severely oversmoothed (as a glance at the inset reveals). Decreasing kernel width (*middle*) to improve resolution sufficiently to fit the fine scale details on the left is seen to be tied to drastic overfitting in the right part of the plot. By contrast, the *bottom* panel was created with the sRVM (BIC prior, symmlet) and shows that a smoothness prior in combination with a multi-resolution design matrix achieves an adaptive level of smoothing: there is neither overfitting in the inset nor oversmoothing towards the right of the figure.

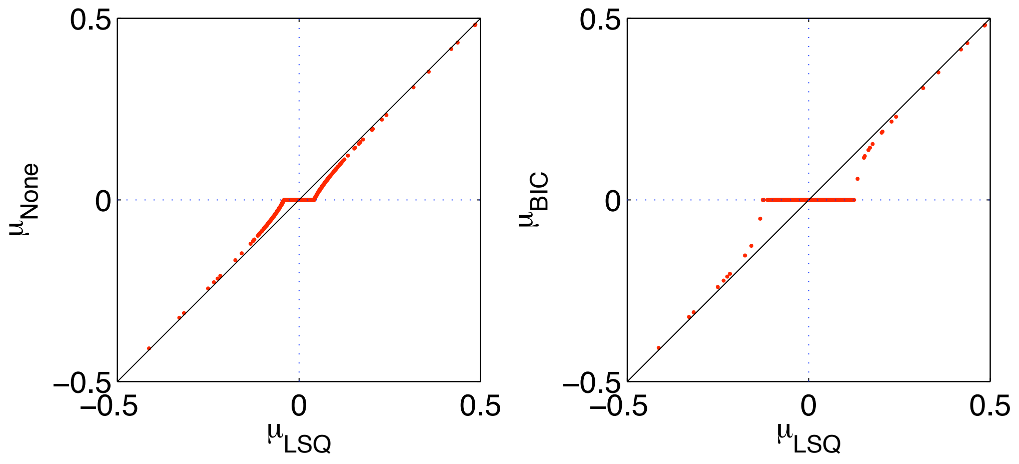


Figure 2.13: Shrinkage plots. Plotting the least squares estimate of the weights μ_{LSQ} against the posterior weight estimates obtained with **None** (*left*) and **BIC** (*right*) priors clearly shows that the **BIC** smoothness prior is much more effective at weeding out small, irrelevant components by setting them to 0. These plots correspond to the middle right and bottom panel, respectively, of Figure 2.12) and are clipped to $|\mu_m| \leqslant 0.5$ (the few larger components are essentially unaffected by shrinkage and thus lie on the diagonal).

smoothing to fit the signal, but not the noise, at each scale (Figure 2.12 *bottom*), whereas the RVM's dependence on kernel choice for sparsity control and thus smoothing means choosing between the evils of oversmoothing the high-frequency structure (Figure 2.12 *top*) or overfitting the low-frequency structure (Figure 2.12 *middle*).

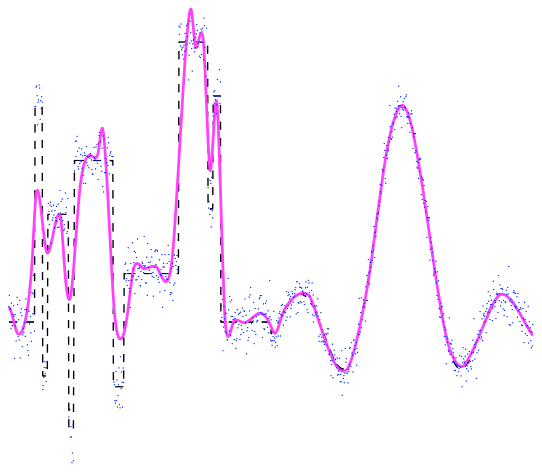
The effect of the smoothness prior on sparsity is most clearly visualized by comparing "shrinkage plots" for **None** and **BIC** (Figure 2.13) for the Doppler data.

Heterogeneous data and overcomplete dictionaries

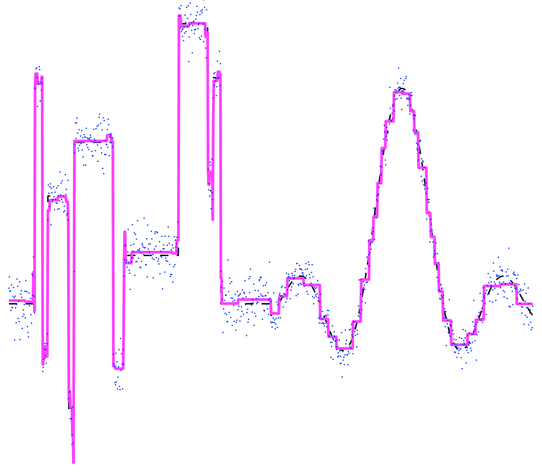
Another attractive ability of the sRVM is to automatically choose the right locally fitting components from an overcomplete dictionary. This can be used to obtain very good results for signals that are heterogeneous to the extent that particular regions can be well represented sparsely by a particular (non-custom) kernel whilst another standard kernel (or combination of kernels) would yield much better results for other regions. Figure 2.14 presents an illustrative toy example in which an overcomplete dictionary of thin-plate spline (tpspline) kernels and Haar wavelets is shown to provide an effective sparse representation of the concatenation of the smooth Sinc data and the step-like Blocks data (Donoho and Johnstone, 1994). Likewise, data such as the HeaviSine data set (ibid.), with small discontinuity regions but mostly smooth and continuous overall can also profit from a similar overcomplete dictionaries.

Overcomplete dictionaries constructed from morphologically diverse kernels have also found applications to blind source separation problems, where the morphological differences in individual signal components allow such components to be largely represented by morphologically similar parts of the overcomplete dictionary which can be leveraged to effect the separation (Bobin et al., 2005). Although the sRVM is clearly not designed with that task in mind, in this specific, simple example we obtain near-perfect separation of the Blocks and Sinc signal components by discarding the tpspline and Haar contribution respectively (see Figure 2.6.5 *right*). By contrast, the plain RVM cannot achieve this separation (see Figure 2.6.5 *left*), apart from needing 419 instead of 88 components and achieving only a MSE of 0.024 instead of 0.009.

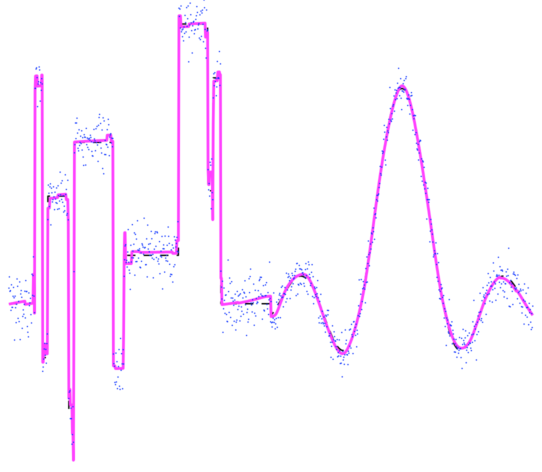
It has to be noted, however, that although with symmlet or spline kernels we generally achieve near-identical results for $\hat{\mathbf{y}}$, regardless of the way component inclusion proceeds, overcomplete



(a) sRVM 3.0 tpspline kernel (MSE: 0.379, S: 72)
left half (Blocks) of signal cannot be properly resolved



(b) sRVM Haar design matrix (MSE: 0.020, S: 75)
now the right half (Sinc) shows staircase artifacts



(c) sRVM overcomplete haar+tpspline dictionary (MSE: 0.009, S: 88)
the sRVM automatically finds the right basis functions for each region of the signal

Figure 2.14: The sRVM (here with **RIC** prior) makes it possible to obtain very good results by using overcomplete dictionaries. The example data ("BlocksSinc") is constructed by concatenating two signals with very different characteristics: Blocks and Sinc and adding Gaussian noise (SNR: 7.0). Whilst no standard kernel will give ideal results for this combination, thin-plate splines (tpsplines) are well suited for smooth, continuous curves such as Sinc (a), whilst the step-like nature of Haar wavelets makes them the ideal candidate for the Blocks subset (b). However, thanks to the smoothness prior, the sRVM can do a remarkably good job at automatically picking the appropriate components for each part of the signal from an overcomplete dictionary obtained by concatenating both these two sets of basis functions together (also see Figure 2.6.5).

.

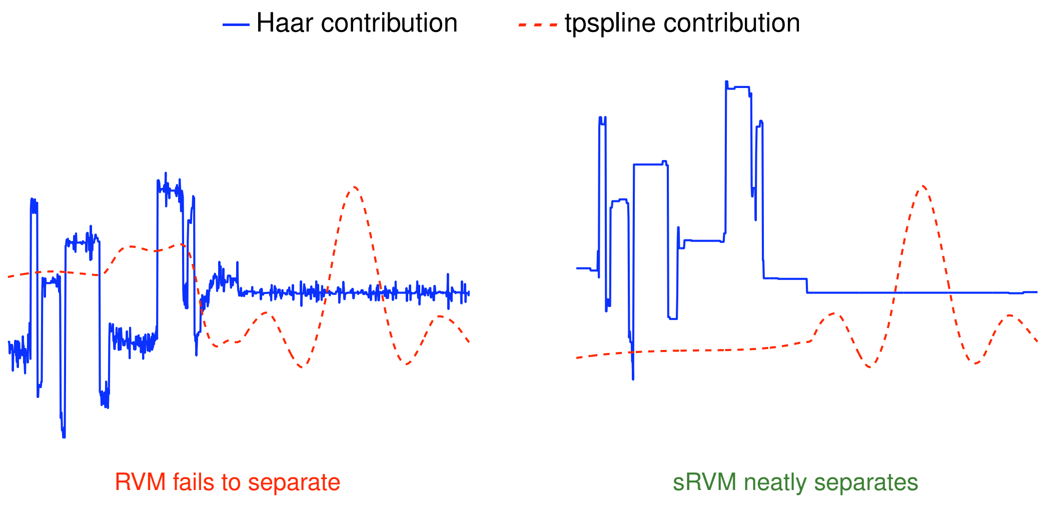


Figure 2.15: sRVM regression as ad hoc blind-source separation (same toy BlocksSinc data set and overcomplete haar+tspline dictionary as the previous figure, 2.14; but this time the contributions of each dictionary part to the final result are shown as separate, differently coloured curves (Haar, tpspline)). *Left panel:* The RVM's posterior mean prediction $\hat{\mathbf{y}}$ for the above setup: although the RVM achieves only a limited degree of separation (the Blocks part of the left half of the signal is largely, but not exclusively, reconstructed out of Haar basis functions and the Sinc part on the right is mostly made up out of tpspline components). It also fits a significant part of the noise (mostly with the Haar part of the overcomplete dictionary, in both halves of the signal). *Right panel:* The sRVM achieves near-perfect denoising and separation of the two signal components (Blocks and Sinc) into the respective dictionary parts (Haar and tpspline) in its posterior mean prediction $\hat{\mathbf{y}}$ (note that this is just a different visualization of the result displayed in the bottom panel of Figure 2.14).

dictionaries appear to expose some limitations of the fRVM scheme on which we build. Apart from numerical issues caused by the overcompleteness, getting trapped in different local maxima starts to become a problem, so that we see more variability in the results than we do under simpler scenarios. The question of local minima will be explored in more detail in section 2.6.6 and we discuss the use of Markov Chain Monte Carlo (MCMC) sampling in section 2.7.

Component separation in Real data

Although the (very nice) BlocksSinc data results we presented in Figures 2.14 and 2.6.5 have to be taken with more than a grain of salt in light of the fact that the data set has been purposefully grafted out of components with markedly different characteristics, precisely for the purpose of being well-separable into the two different parts of the over-complete dictionary we used, they nevertheless make it tempting to ask: is there a possibility to obtain a similar, albeit likely much more modest success in "ad-hoc blind source separation" for a real data set? To answer this question we carried out some experiments with real data sets, of which we will report one in the following section.

Spindles in sleep EEG The data set (partly depicted in the top panel of Figure 2.16) in question is a single channel sleep EEG recording with $N = 4097$ data-points (Rezek, 2008, we removed the last data point to make the number dyadic). The data set contains sporadic sleep spindles, and the intervals in which they occur (marked in red in Figures 2.16ff) have been marked by an expert. We were interested in finding out if the sRVM could be used to separate the spindle sleep regions from the non-spindle sleep regions. We tried various combinations of overcomplete dictionaries composed symmlet, gauss and tspline kernels for different kernel widths but did not see promising results. By visual inspection the most obvious distinction between spindle regions and non-spindle regions is the a difference in frequency, so we also attempted overcomplete dictionaries with a discrete cosine transform base (just a matrix of orthogonal shifted and scaled cosines, see Ahmed et al. (1974) and section A.1). Since the DCT dictionary part, in the overcomplete dictionary experiments we did, usurped most of the signal (both spindles and non-spindles), we eventually opted for using it exclusively (which also brings an enormous speed improvement, because the DCT can be much more efficiently implemented than via matrix multiplication (e.g. via the FFT) – as a $O(N \log N)$ operation (almost as good as wavelets). Also, like wavelets, it is very well-behaved numerically and analytically convenient due to orthogonality. From Figure 2.17 we can also see that the EEG data occupies distinct lumps or "blobs" in DCT space, and that those "blobs" only become clearly separated after sRVM denoising (we incidentally also attempted a RVM run for comparison, but the aborted when the model reached close to 2000 active components).

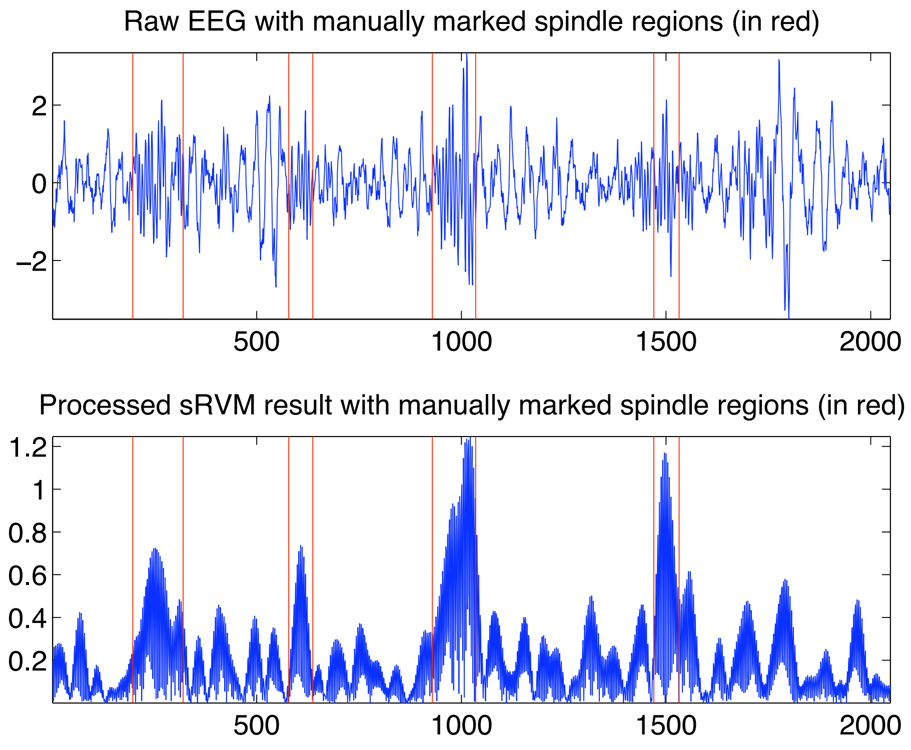


Figure 2.16: *Top panel:* Raw sleep EEG data (only the first half of $N = 4096$ data points is displayed for clarity) with human expert marked spindle regions (delimited by red lines). Note that whilst it is possible to make out some distinguishing characteristics in the spindle regions (more high frequency components and an increased amplitude, compared to the surrounding data), it is difficult to correctly identify the relevant regions by eye for a non-expert and neither criterion seems sufficient on its own. *Bottom panel:* The post-processed RIC sRVM denoised version of the same data as above. After denoising the data with the RVM (RIC prior), we removed all components outside the "second blob" in DCT- (viz. weight-) space, located around abscissa coordinates 880-1150 in Figure 2.17. This was done simply by setting the w_i in this range to zero and pre-multiplying $\mathbf{\Phi}$ to yield a modified posterior estimate $\hat{\mathbf{y}}_{\text{pruned}}$. Subsequently we took the absolute value of the result in data-space, but this was done just to make it slightly easier to see how well the red lines of expert-determined spindle regions align with the peaks. We can see that the spindle regions are much easier to visually discern in the bottom graph – in particular for the displayed excerpt we could now detect the existence of a region by amplitude alone (in this case an ordinate threshold of 0.65 would work)

Whilst we didn't obtain a perfect separation within different parts of just the DCT dictionary – see Figures 2.16 (bottom) and 2.18 – the latter of the two figures, in particular, suggests that we isolated at least a significant aspect of the spindle data. Of course the fact that the data is not artificial (and thus does not come with a known right answer) makes it harder to judge the quality of the result, especially for a non-EEG expert.

A more satisfying separation experiment on real data would likely require more carefully constructed dictionaries that reflect deeper domain knowledge about the signals they were intended to sparsely represent.

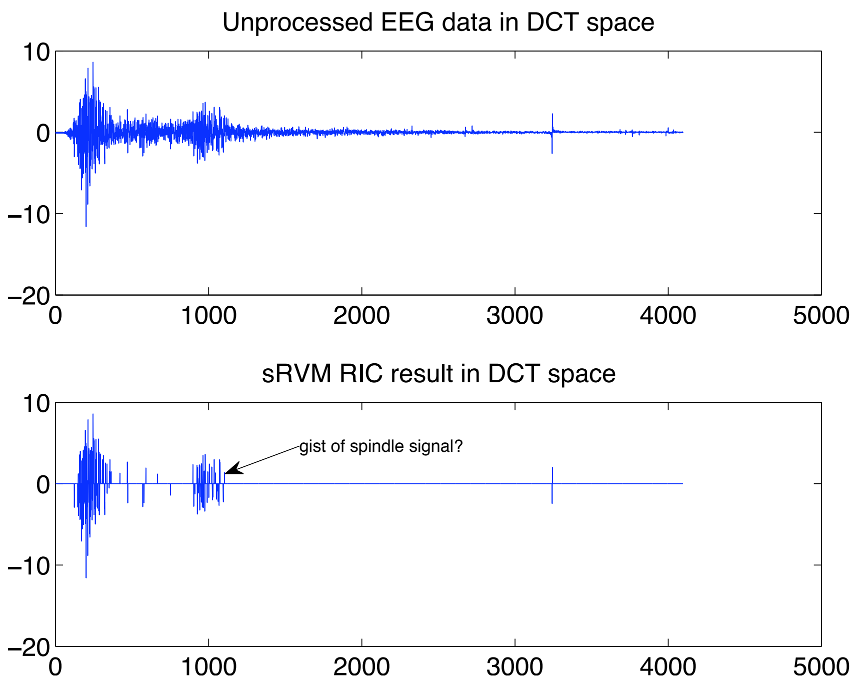


Figure 2.17: Unprocessed EEG data (*top*) and the sRVM denoised EEG data in DCT (discrete cosine transform) space (*bottom*). Low frequency components are on the right and high frequency components on the right. Note that whilst although even before denoising one can easily discern two, three distinct "blobs" as well as a lonely spike (in the right half); no obvious boundary between them is apparent till after denoising. Both visual inspection of the original signal and further experimentation suggests that the spindle waveforms are mostly located in the frequencies occupied by the middle blob (marked in the lower subplot). It should maybe also be explicitly stated that since we are using a $\mathbf{\Phi}$ that implements the DCT, DCT-space is simply equivalent to weight space. In other words the second plot simply shows the posterior mean weight estimate $\boldsymbol{\mu}$ and the first plot is identical to the least-squares estimate of the vector $\mathbf{m}$ that solves $\mathbf{t} = \mathbf{\Phi m}$.

Summary statistics

Table 2.1 shows for a number of standard data sets the sparsity, measured by the number of included components S, and the MSE between the mean prediction $\hat{\mathbf{y}}$ and the true signal $\mathbf{y}$. Clearly the **None** prior is generally insufficiently severe to control the sparsity for multiresolution dictionaries, while the smoothness priors provide sufficient smoothing and thus permit σ^2 to be correctly estimated. On the other hand it is evident from the gauss 3.0 (see A.1 for definitions of all used kernels) examples that the smoothness priors do not result in missestimation when smoothing is already enforced by the kernel. Further, the Doppler data in the second half of the table demonstrates that even if the true value for σ^2 is given so that incorrect noise estimation is not an issue, the **None** prior is still too weak to bring about the desired level of sparsity.[30]

Lastly, although **BIC** rarely obtains the best answer, it is typically not too far off which recommends it as the default choice.

[30]With this exception all results in this thesis, including those in Figure 2.12, were obtained using noise estimation.

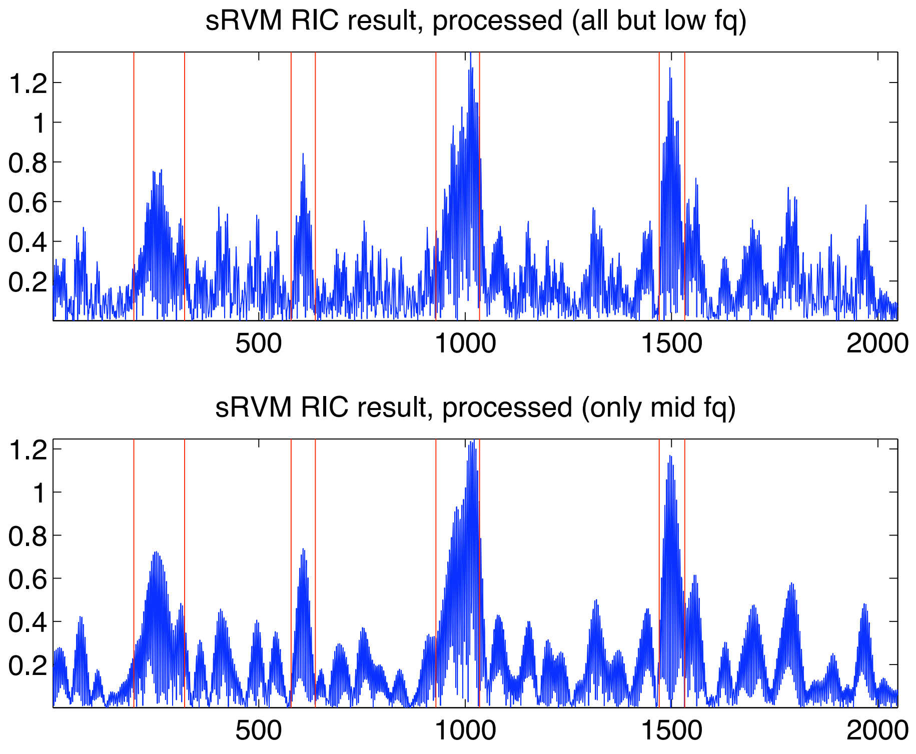


Figure 2.18: EEG spindle data: comparison between *including* only the second big blob in DCT space (centered around abscissa coordinate 1000 in the bottom of Figure 2.17) in the reconstruction (*bottom*, same as Figure 2.16 *bottom*) and *excluding* only the first blob in DCT space (centered at around 100 in the bottom of 2.17) from reconstruction. Since both plots looks rather similar, it appears the most salient aspects of the spindle regions (again marked by red bars) are indeed to be found in the second blob.

σ^2 is estimated

Bumps SNR=2.0 ($\sigma^2 = 0.119$, $N = 128$)

Kernel	Prior	S	MSE	σ^2_{MAP}
symmlet	None	127.0±0.0	0.119±0.001	0.000±0.000
symmlet	**AIC**	36.3±7.8	**0.088±0.008**	0.121±0.037
symmlet	BIC	11.9±2.5	0.153±0.034	0.262±0.038
symmlet	RIC	2.6±1.4	0.320±0.051	0.450±0.068

Bumps SNR=7.0 ($\sigma^2 = 0.010$, $N = 128$)

Kernel	Prior	S	MSE	σ^2_{MAP}
symmlet	None	127.0±0.0	0.010±0.000	0.000±0.000
symmlet	**AIC**	61.9±6.0	**0.009±0.001**	0.010±0.004
symmlet	BIC	19.2±4.9	0.081±0.024	0.106±0.029
symmlet	RIC	6.4±1.3	0.203±0.024	0.238±0.018

Sinc SNR=2.0 ($\sigma^2 = 0.031$, $N = 128$)

Kernel	Prior	S	MSE	σ^2_{MAP}
gauss 3.0	None	5.7±0.7	0.004±0.001	0.032±0.001
gauss 3.0	**AIC**	5.4±1.1	**0.004±0.001**	0.034±0.001
gauss 3.0	BIC	5.2±0.6	0.005±0.001	0.035±0.002
gauss 3.0	RIC	4.9±1.0	0.005±0.001	0.036±0.002
symmlet	None	127.0±0.0	0.031±0.000	0.000±0.000
symmlet	AIC	28.9±5.9	0.012±0.003	0.020±0.004
symmlet	BIC	9.1±1.9	0.006±0.002	0.032±0.003
symmlet	**RIC**	6.2±0.6	**0.006±0.001**	0.036±0.002

σ^2 is given

Doppler SNR=2.0 ($\sigma^2 = 0.031$, $N = 1024$)

Kernel	Prior	S	MSE
symmlet	None	367.3±10.5	0.00067±0.00002
symmlet	AIC	130.5±6.9	0.00041±0.00002
symmlet	**BIC**	56.6±2.8	**0.00026±0.00002**
symmlet	RIC	42.2±1.5	0.00036±0.00002

Table 2.1: Empirical comparisons of different priors on standard data sets. Results are averaged over 10 runs (with different noise ϵ on each run). Results with lowest MSE appear in **bold**.

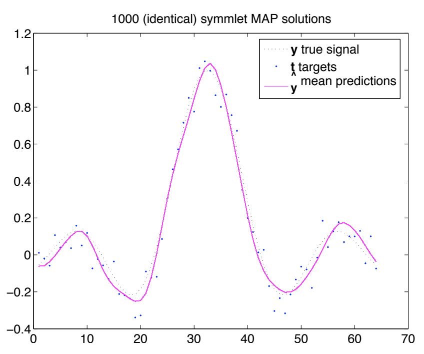
2.6.6 Exploring local minima

As we have noted, for overcomplete dictionaries local minima appear to be an issue, whereas the $\hat{\mathbf{y}}$ results we have observed for wavelet dictionaries and also spline kernels appear very uniform. In order to understand the issue more deeply we decided on a more detailed exploration of the search space for the simple Sinc regression.

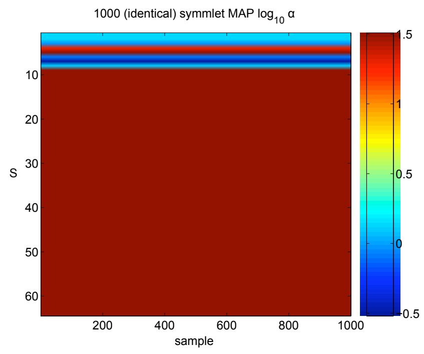
A detailed investigation of $N = 64$, SNR $= 5$ Sinc

To gain insights about the robustness of sRVM solutions and the nature of the solution landscape, we compared the results of 1000 runs on an identical data set (Sinc, $N = 64$, SNR $= 5$) differing only by the random seed for the component selector, both with orthogonal symmlet wavelets and with lsplines (as have been used by Tipping for such data).

Symmlet results As Figure 2.19 shows, a thousand differently seeded runs with symmlets produce essentially identical results. The true MSE per point lies at 0.0017, the data MSE is 0.0055; this is achieved with $S = 8$ selected components.



The $\hat{\mathbf{y}}$ results from 1000 BIC sRVM runs on the same data (Sinc, SNR=5, N=64) with symmlets – they can be seen to be visually absolutely identical (the largest difference of corresponding entries is about 10^{-10})

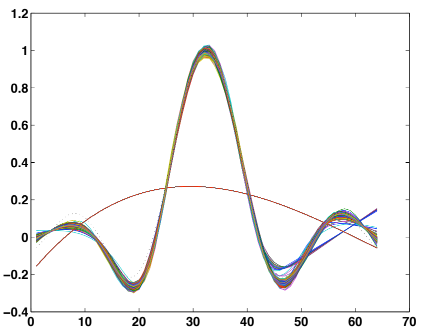


The corresponding $\boldsymbol{\alpha}$ values, which are again virtually identical (the maximal log difference is about 10^{-8}) There are 8 non-∞ components (although the colorbar seems to suggest otherwise, the darkest red is really ∞).

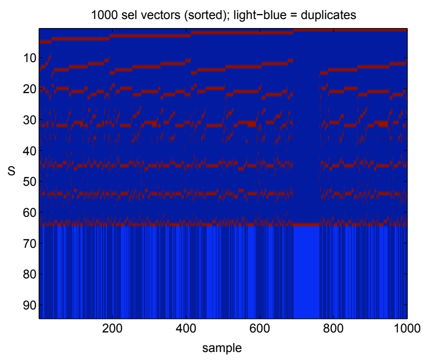
Figure 2.19: The 1000 BIC symmlet results

Lspline results Using lsplines (with scale = 3.0) on the same data gives much more complex results; whilst the posterior mean predictions (the $\hat{\mathbf{y}}$s) in Figure 2.20 are mostly similar there. In particular there is a noticeably outlier in form of the red concave line that corresponds to less than 5% of the results. Although the minimum MSE per point lies at only 0.0009 (corresponding data mse, c.d.m: 0.0043), the average MSE is 0.0083, due largely to the 0.1006 maximum MSE (c.d.m.: 0.1142), which is obviously associated with the drastically oversmoothed concave prediction displayed in red in Figure 2.20 a). Removing the 40 drastically oversmoothed results yields a mean MSE of 0.0013. These oversmoothed results correspond to the rightmost σ^2 histogram bar in Figure 2.20 c), as can be seen the other noise estimation results are remarkably robust and close to the truth.

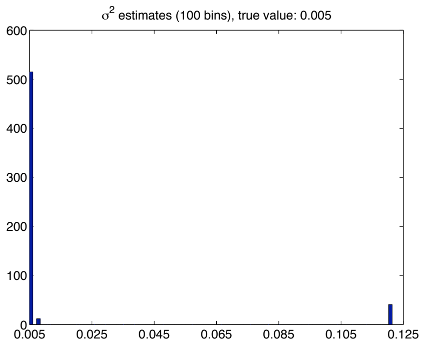
The variety of solutions is reflected by 568 unique $\mathcal{S}$ vectors in Figure 2.20 b), which is indicative of a large number of local modes – the simplicity of the data notwithstanding.



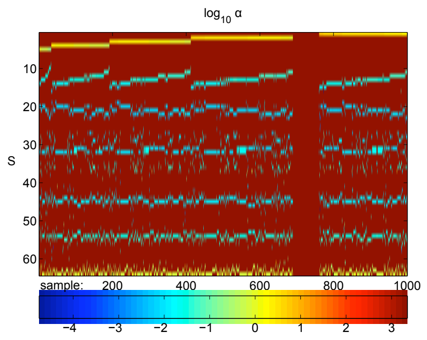
The $\hat{\mathbf{y}}$ results from 1000 BIC sRVM runs on the same data (Sinc, SNR=5, N=64) with an 3.0 lspline kernel. Note that oversmoothed red line corresponds to $\sim$ 40 identical samples in the figure on the right.



The selection vectors $\mathcal{S}$ as columns (🟥: selected 🟦: unselected), sorted (arbitrarily) to show (the 432) duplicates (indicated by light-blue vertical bars). The homogeneous patch with only two active components located roughly between samples 740 and 780 corresponds to the oversmoothed red line in the panel to the left, suggesting a local minimum that is difficult to escape from.



Apart from the small fraction corresponding to the severely oversmoothed results (rightmost bin), the noise estimation is very accurate and robust (despite the tiny bin-distance of 0.0012, there are only 3 visible bins; including duplicates has no appreciable effect on the histogram).



The $\log_{10} \alpha$ values sorted as above; note that that in the two-components-only results in the region around 730 have quite large $\log_{10} \alpha$ at around 3.3 (infinities are "invisible", because Matlab$^{\text{TM}}$by default colors them just like the maximum value).

Figure 2.20: The 1000 BIC 3.0 lspline results corresponding to Figure 2.19.

Oversmoothed runs in the Sinc lspline experiment The question what gives rise to the drastic oversmoothing is of course of special interest; closer inspection of actual runs and region 1 in the top graph of Figure 2.6.6 shows that these results are not simply due to premature convergence: the longest numbers of steps before convergence was in fact taken by the runs that resulted in oversmoothing (with one exception around 330). What seems to be happening is that the sRVM gets stuck in a local optimum with extremely overestimated noise and just 2 active components; due to the poor noise estimate the addition of a further component cannot bring enough of an improvement to be justified which in turn thwarts attempts to reestimate the noise properly, however there is still enough "wiggle-room" for the small ratio-differences in successive values of the σ^2 estimates ($< 10^{-4}$) to delay convergence. Still, the local maximum seems to be a genuine one –

forcing a larger minimum number of steps before convergence has no effect; the local maximum cannot be escaped.

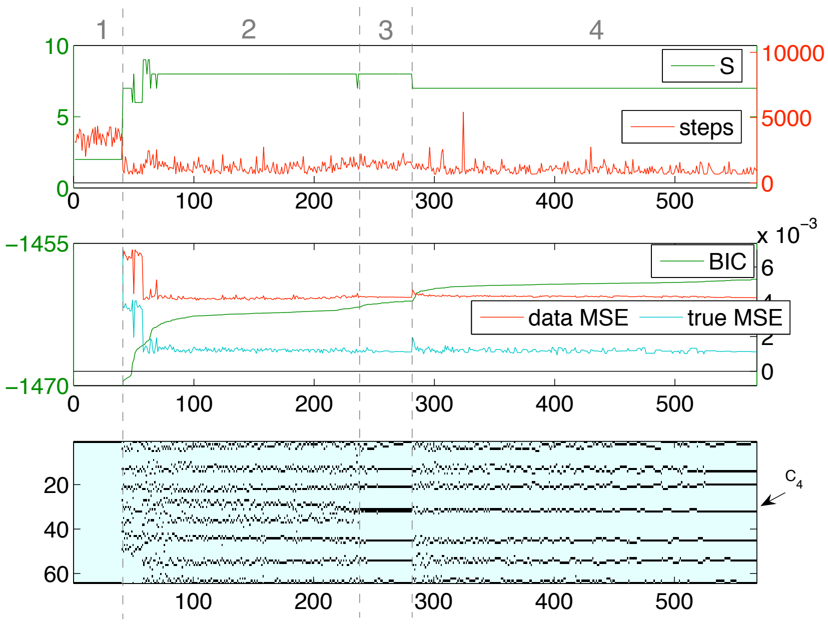
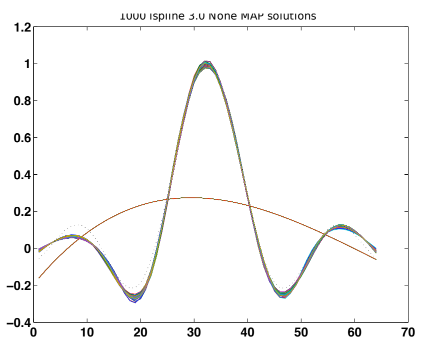
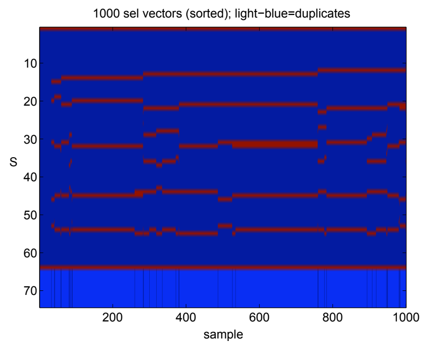


Figure 2.21: A comprehensive inspection of the Sinc 3.0 lspline results. The results are sorted in by $\hat{\mathcal{L}}_{\text{BIC}}(\boldsymbol{\alpha})$ for and duplicates (as determined by $\mathcal{S}$) are removed; all graphs are divided horizontally into 4 zones of interest. The first graph shows the number of steps that a run took to converge and the number of selected components. The second graph shows the posterior likelihood (BIC) and the MSE of the predictions $\hat{\mathbf{y}}$ to the underlying data $\mathbf{t}$ as well as the true signal $\mathbf{y}$. Note that the left and right abscissas have been truncated to remove the very low BIC results in partition 1 (which lie at around -1500) and the very large MSEs (around 0.1). The third and final graph displays the active components after each run, the development of what ends up as component 4 (marked as C_4) is of special interest.

Another question of interest is whether the few occurring instances of oversmoothing can be blamed on the smoothness (BIC) prior. Although the None prior of the original RVM does an even better and more consistent job for this simple data set (producing only 38 unique solutions) (see Figure 2.6.6), interestingly the radically oversmoothed results are still present and still occupy a similar proportion (32/1000 instead of 40/1000). The minimum true MSE is 0.0010 (corresponding data MSE: 0.0043) and the average with the oversmoothed examples discarded is almost the same; 0.0043 if they are included.



The $\hat{\mathbf{y}}$ results from 1000 None sRVM runs on the same data (Sinc, SNR=5, N=64) with an 3.0 lspline kernel. Note that the over-smoothed red curve appears again, although no smoothness prior is used (which proves that this spurious local minimum is not an artifact introduced by the smoothness prior).



The selection vectors $\mathcal{S}$ as columns (🟥: selected 🟦: unselected), sorted (arbitrarily) to show (the 962) duplicates (indicated by light-blue vertical bars).

Figure 2.22: The 1000 3.0 lspline results for a None prior.

Other patterns in the lspline Sinc experiments Removing duplicates (as determined by $\mathcal{S}$ and not α values) and sorting by the posterior probability (or BIC), a number of interesting patterns can be observed in Figure 2.6.6. In region 1 we are looking at the drastically oversmoothed estimates discussed in the previous section. Going from region 2 to 3 and finally 4 we can see how a cluster of possible adjacent component pairs is eventually unified into a single component between them labelled as C_4 (see bottom graph, right), resulting in a sparser model with higher BIC. The reasons that such a pattern can be observed are two-fold:

1. locality; for lsplines (and gauss basis functions), basis function are fairly localized and adjacent basis functions have a high degree of redundancy – this makes it possible to replace two nearby basis functions with an intermediary.

2. greediness; the (s)RVM operates on a greedy, one-basis-function-a-time scheme – this makes it difficult to replace two basis functions with a single one, since the first step (removal of the two basis functions) would generally result in a significant, albeit temporary, reduction of the $\hat{\ell}$ viz the posterior $\hat{\mathcal{L}}$ it is unlikely to be undertaken. Hence the stability of the local optima of region 2.

Conclusions from Sinc lspline experiment Although the smoothness prior seems to complicate the fitness landscape, it is not responsible for the presence of the local drastically oversmoothed mode, that has a very low probability compared to all the other found solutions.

The results in Figure 2.6.6 also suggest an inability to escape local modes that would require several RVM-steps to amortize.

In summary the greediness of the RVM scheme combined with its inclusion or exclusion of one component at a time certainly does create a practical problem with local minima that cannot be easily escaped, particularly taking into account the simplicity of the data set and basis functions. After all, the Sinc data set in combination with lspline basis functions does not harbour much potential for multiple truly alternative competing explanations – unlike the overcomplete BlocksSinc data where two complete sets of basis functions compete and several quite different reconstructions with low to moderate reconstruction error (in terms of MSE) but highly discernible visual characteristics and basis-function make-up are achievable. This of course makes BlocksSinc with an overcomplete Haar/spline dictionary a data set of much greater interest with respect to local modes and we will examine it next.

2.6.7 A not quite so detailed investigation of BlocksSinc

Ideally one would hope that the blocks part and the Sinc part should each exclusively use the Haar and spline dictionary respectively, but clearly other divisions of labor are thinkable so that one expects a sizable number of modes many of which – unlike the Sinc case – give rise to $\hat{\mathbf{y}}$ with quite different characteristics. Unfortunately the ideal reconstruction seems to require fairly high resolution; for computational considerations we restricted ourselves to doubling the resolution compared to that of the Sinc experiments to $\mathbf{\Phi}^{256\times 512}$, although cursory further examination suggested that even quadrupling the resolution to $N = 512$ appears to be insufficient to obtain *really* good results with $\mathrm{SNR} = 7$.

BlocksSinc with None prior Using a None prior resulted in an exceptionally large percentage of convergence failures so that 1000 RVM runs took around 5 days as only 81/1000 trials converged within the limit of 10^6 steps[31]. For this reason we will not further discuss the None case, apart from saying the MSEs for the converged trials were fairly close – within $[4.4054, 4.4102]$.

BlocksSinc with BIC prior With the BIC prior all trials converged and gave rise to different $\mathcal{S}$. It is immediately apparent that this time noticeable differences in the $\boldsymbol{\alpha}$ values translate into noticeable differences in the posterior mean estimates $\hat{\mathbf{y}}$ (see Figure 2.6.7).

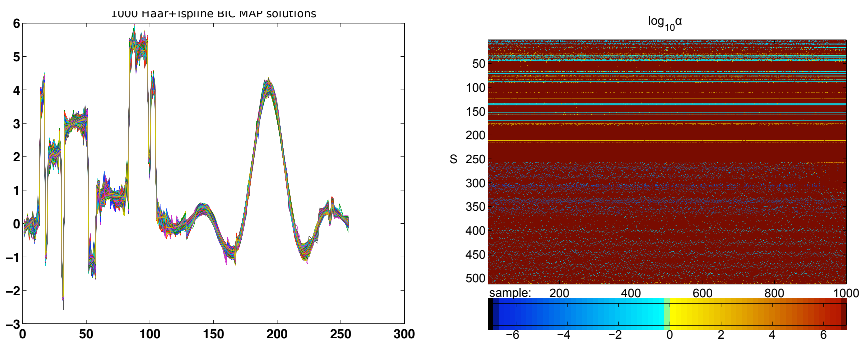


Figure 2.23: 1000 runs of BlocksSinc (SNR=7.0, N=256, M=512) with a BIC prior and mixed Haar/lspline dictionary, left the $\hat{\mathbf{y}}$ values, right the corresponding $\boldsymbol{\alpha}$. Note that in contrast to the spline experiments, here different $\boldsymbol{\alpha}$ values really result in noticeably different predictions $\hat{\mathbf{y}}$.

Exploring local minima: summary and conclusion Although the preceding experiments were rather limited in scope, they certainly seem to suggest that the getting trapped in local minima has not received adequate attention in the RVM literature. Whilst it is true that convergence detection is fiddly, which means that there always is a possibility that these results to some extent reflect quality of implementation issues more than shortcomings of the scheme as such, there is at least some corroborating evidence pointing in a similar direction in the existing literature: Quiñonero-Candela's 2004 thesis. It contains an experiment that is very similar in spirit, but with 100 runs

[31]To be precise the maximum number of steps for any trial that converged was around 10^5 and the minimum was around 7000

of an old-style (and hence not directly comparable) implementation of the RVM. It also has to be admitted that there are some differences; for example he observes some degenerate results due to severe *underestimation* of the noise, something we have not seen crop up as a problem.

Despite these caveats, we are tempted to conclude that the relative lack of attention local minima have seen in the RVM literature lies less in the insignificance of the problem as such and more in the prevalent use of spline kernels. These kernels mask the issue to some extent by mapping a variety of quite distinct $\boldsymbol{\alpha}$ to very similar $\hat{\mathbf{y}}$, because the spline basis functions tend to be somewhat redundant.

Thus whilst we observed mostly near identical mean predictions $\hat{\mathbf{y}}$ on multiple runs of the same experiment, the underlying $\boldsymbol{\alpha}$ values often differed substantially for lsplines and closer inspection also suggested that the greediness of the fRVM scheme, which takes one step at a time makes it quite difficult to replace two good complementary components with a better third one (Figure 2.6.6, C_4).

Moreover, although we found that wavelets gave essentially identical results on multiple runs, down to the actual values of $\boldsymbol{\alpha}$ (no doubt because the orthogonality of wavelets results in a substantially simpler search space), the results on the overcomplete BlocksSinc data set show that differing $\boldsymbol{\alpha}$ predictions for re-runs of the same experiment are not always benign in the sense that they map to very similar $\hat{\mathbf{y}}$; because here both ($\hat{\mathbf{y}}$ and $\boldsymbol{\alpha}$) turn out quite different from one run to the next.

In summary, the results motivate a search for less greedy implementation strategies than fRVM.

2.7 An attempt at a Markov Chain Monte Carlo implementation

Although empirically the greedy nature of the fRVM type II MAP scheme does not seem to be much of in an issue under many scenarios (including the spline kernel examples that have dominated the RVM literature and our use of wavelet dictionaries), our explorations in the previous section in fact suggest that even the standard Sinc spline example is subject to convergence to different local maxima in $\boldsymbol{\alpha}$ which happen to yield similar $\hat{\mathbf{y}}$ but that under the stress-test of using overcomplete dictionaries local maxima can start to become a real and obvious problem.

Whilst we have found it useful to increase exploration by ad hoc measures[32], the added flexibility of the sRVM for kernel choice appears to call for a principled way to deal with local maxima. Maybe the most versatile and theoretically principled non-greedy scheme is Markov Chain Monte Carlo sampling, and we will now consider its application to the sRVM. Further significant advantages of an MCMC scheme include obtainment of full posterior distributions instead of point estimates as well as proper reflection of the uncertainty in intermediate parameters that, in a MAP scheme, are just represented via an estimate of the mode (most importantly $\alpha^\star$).

The great advantage of MCMC is thus that it offers a fully Bayesian treatment and, provided one has some confidence that the sampler converged[33], it gives more reliable and powerful results than a simple MAP or ML point-estimate obtained via greedy maximization. The reasons for this are threefold: Firstly, whereas the fRVM only finds a single a-posteriori estimate for $\boldsymbol{\alpha}$ and σ, and thus fails to take the uncertainty in these parameters properly into account, possibly resulting in over-confident estimates, MCMC can sample over them and is thus effectively able to integrate them out. Secondly these samples hold some interest in themselves, because they describe the problem space. Thirdly, by virtue of sampling uniformly over all possible values of these parameters, the problem of getting stuck in local minima disappears.

It should however be stressed from the outset that there is a certain tension between MCMC and the (s)RVM and, more generally, sparsity which is fundamentally at odds with sampling. The RVM uses sparsity to great computational advantage (both in terms of speed and in terms of numerical robustness), but at the price of employing an overly greedy search that can be trapped in local minima and even using a probabilistic model that must be considered as somewhat degenerate.

On the other hand at a fundamental level a MCMC scheme will be non-sparse because it samples over all possible values for $\boldsymbol{\alpha}$ so that the posterior mean estimate $\hat{\mathbf{y}}$ will no longer correspond to a single value for $\boldsymbol{\alpha}$ or σ [34]. However all is not lost: firstly as far as the computational advantage is concerned, we can still avail ourselves of sparsity at each individual sampling step[35], secondly as far as the interpretability advantages of sparsity is concerned, we can still examine which basis functions find frequent inclusion, i.e. often have non-infinite corresponding α_i. Indeed, if anything MCMC allows us to a deeper understanding of the role of different basis functions; because it allows us to better judge the importance of a single basis function by checking, for example, what percentage of samples contain it or to test if there are different competing sets (or subsets) of basis functions that form local modes in the posterior. Generally having (an approximation of) the posterior distribution available allows to answer a richer set of questions – even if we are just interested in the MAP solution the fRVM tries to find, we could use MCMC with the advantage that because of the sampling we can have higher confidence in its being the true global mode

[32]e.g. by initially also including α_i for which $\hat{\ell}(\alpha_i) \leqslant \hat{\ell}(\infty)$.

[33]This is more than a minor caveat because it is precisely the fact that sampler convergence is often not only difficult to achieve but also difficult to reliably diagnose once achieved, that has earned MCMC the reputation of still being something of an black art.

[34]To be more precise: unless we chose to base our estimate on the MAP value of $\boldsymbol{\alpha}$, but there is no good reason to do so if we can integrate $\boldsymbol{\alpha}$ out instead

[35]Or to be pedantic: most sampling steps, because as we sample from the full posterior, non-sparse (even full) $\boldsymbol{\alpha}$ will also find inclusion in theory if we sample long enough. Of course we expect the relative likelihood of such steps compared to the sparse ones to be exceedingly low, so that we will only rarely if ever observe them.

rather than a local maximum than we could have in the max a posteriori solution obtained by the fRVM's greedy scheme.

2.7.1 An introduction to MCMC

In this section will we very briefly give a quick introduction/refresher to two of the most widely used MCMC algorithms: Gibbs and Metropolis-Hastings sampling (a good and thorough introduction can be found in Neal (1993)).

There is a variety of methods to sample indirectly from a distribution p for which it is at the minimum possible to evaluate $p^\star(x) \propto p(x)$ for any x (not having to know the constant of proportionality is often important, since finding it might in itself require a complicated integration).

Rejection sampling One of the simplest and oldest is rejection sampling, which works as follows: assume there is a distribution q that is easy to sample from and that $p^\star$ is scaled such that $\forall x : p^\star(x) < q(x)$. Then the following procedure will sample from p: draw a sample x' from q and a then a second uniformly distributed sample $u \sim U[0, q(x')]$. If $u \leqslant p^\star(x')$ accept the sample; otherwise repeat (see Figure 2.24 for a graphical illustration that can also serve as an informal proof).

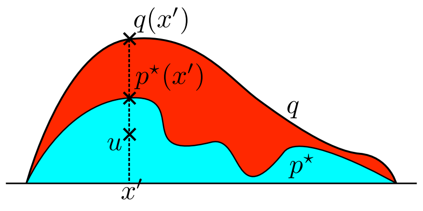


Figure 2.24: Rejection sampling: to sample from awkward distribution p, use a proposal distribution q and a function $p^\star$ such that $\forall x : p^\star(x) \propto p(x) \wedge p^\star(x) < q(x)$. Then generate a proposal $x' \sim q$ and a uniform sample $u \sim U[0, q(x')]$. Accept x' as a sample from p if u falls in the blue area of the graph ($u \leqslant p^\star(x')$), else repeat. Informally, this procedure works because the likelihood of a sample lying in an infinitesimal small region around x', namely $q(x')$ multiplied by its probability of then being accepted, namely $p^\star(x')/q(x')$, simply yields $p^\star(x') \propto p(x')$ which shows that (an accepted) x' is an unbiased sample from p.

Of course the rejection rate and hence efficiency of this method depends critically on how tightly q bounds $p^\star$, and it is easily seen (see, e.g. Bishop, 2006, p. 531) that since the rejection rate will generally grow exponentially with the number of dimensions, a sufficiently tight bound for higher-dimensional cases will generally be unobtainable, which is why sampling mostly finds application in the univariate case.

Metropolis-Hastings sampling Metropolis-Hastings (MH) sampling can be regarded as an extension of rejection sampling that is more impervious to the curse of dimensionality and thus can be used to sample in very high dimensions.

To explain MH sampling it is first necessary to introduce the concept of a Markov Chain: consider a probabilistic model where the likelihood of witnessing a certain state $\mathbf{x}^{(t)}$ at a discrete time step t is governed by the states at previous time steps, i.e. $p(\mathbf{x}^{(t)} \,|\, \mathbf{x}^{(t-1)}, \ldots, \mathbf{x}^{(0)})$ – a *Markov Chain* is the special case of such a model in which the current state only depends on its direct predecessor, i.e.

$$p(\mathbf{x}^{(t)} \,|\, \mathbf{x}^{(t-1)}, \ldots, \mathbf{x}^{(0)}) = p(\mathbf{x}^{(t)} \,|\, \mathbf{x}^{(t-1)}) \tag{2.89}$$

If in addition the value of t itself is immaterial we have a (time) *homogeneous Markov Chain*, and we will use the notation $T(\mathbf{x}^{(t-1)} \to \mathbf{x}^{(t)}) \equiv p(\mathbf{x}^{(t)} \,|\, \mathbf{x}^{(t-1)})$ to denote such transition probabilities in a more intuitive and distinct fashion.

The basic idea behind MH sampling is to set up a homogeneous Markov Chain that in the limit $t \to \infty$ converges to the distribution of interest $p(x)$. It can be shown that (subject to some weak conditions, see Neal, 1993) this will occur if the target distribution p is *invariant* with respect to T, i.e.:

i

$$p(\mathbf{x}') = \sum_{\mathbf{x}} p(\mathbf{x})T(\mathbf{x} \to \mathbf{x}') \tag{2.90}$$

As some quick algebra will show, a sufficient condition for invariance is *detailed balance*:

$$p(\mathbf{x})T(\mathbf{x} \to \mathbf{x}') = p(\mathbf{x}')T(\mathbf{x}' \to \mathbf{x}) \tag{2.91}$$

The recipe for Metropolis Hastings sampling is to take samples from an easily sampled proposal distribution $q(\mathbf{x}' \mid \mathbf{x})$ and tweak the acceptance process to obtain a Markov Chain $T(\mathbf{x} \to \mathbf{x}')$ that is invariant wrt. $p(\mathbf{x})$. To that effect we accept a proposal $\mathbf{x}'$ as sample $\mathbf{x}^{(t+1)}$ with the following probability:

$$A(\mathbf{x}^{(t)} \to \mathbf{x}') = \min\left(1, \frac{p^\star(\mathbf{x}')q(\mathbf{x}^{(t)} \mid \mathbf{x}')}{p^\star(\mathbf{x}^{(t)})q(\mathbf{x}' \mid \mathbf{x}^{(t)})}\right) \tag{2.92}$$

Expanding shows that $T(\mathbf{x}^{(t)} \to \mathbf{x}') = A(\mathbf{x}^{(t)} \to \mathbf{x}')q(x' \mid \mathbf{x}^{(t)})$ satisfies the detailed balance property for $p(\mathbf{x})$:

$$\begin{aligned}
p(\mathbf{x})T(\mathbf{x}^{(t)} \to \mathbf{x}') &= p(\mathbf{x})A(\mathbf{x}^{(t)} \to \mathbf{x}')q(\mathbf{x}' \mid \mathbf{x}^{(t)}) && (2.93)\\
&= \min(p(\mathbf{x}^{(t)})q(\mathbf{x}' \mid \mathbf{x}^{(t)}), p(\mathbf{x}')q(\mathbf{x}^{(t)} \mid \mathbf{x}')) && (2.94)\\
&= \min(p(\mathbf{x}')q(\mathbf{x}' \mid \mathbf{x}), p(\mathbf{x}^{(t)})q(\mathbf{x}' \mid \mathbf{x}^{(t)})) && (2.95)\\
&= p(\mathbf{x}')T(\mathbf{x}' \to \mathbf{x}^{(t)}) && (2.96)
\end{aligned}$$

Some further notes: unlike plain rejection sampling, a rejection does *not* result in a repeated sampling attempt in order to determine $\mathbf{x}^{(t+1)}$, instead $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}$. Furthermore, in order to obtain (approximately) independent samples it is necessary to discard the first b steps the sampler takes to reach convergence (the so called *burn-in* period, which has to be determined empirically) and then to collect only samples separated by k steps (i.e. $\mathbf{x}^{(ki)}, \mathbf{x}^{(k(i+1))}$), where k is problem-dependent.

Gibbs Sampling Gibbs sampling is sampling from the joint distribution $p(\mathbf{x}) = p(x_1, \dots, x_n)$ by cyclically sampling from the conditional distributions as follows:

$$\begin{aligned}
x_1^{(t+1)} &\sim p(x_1 \mid x_2^{(t)}, \dots, x_n^{(t)}) && (2.97)\\
x_2^{(t+1)} &\sim p(x_2 \mid x_1^{(t+1)}, x_3^{(t)}, \dots, x_n^{(t)}) && (2.98)\\
&\vdots && (2.99)\\
x_n^{(t+1)} &\sim p(x_n \mid x_1^{(t+1)}, x_2^{(t+1)}, \dots x_{n-1}^{(t+1)}) && (2.100)
\end{aligned}$$

and is just a special case of MH sampling in which the proposal distribution $q(\mathbf{x}' \mid \mathbf{x}^t)$ for a step is the conditional $p(x_n \mid \{x_j\}_{j \neq n})$, and the acceptance probability A simplifies to 1. It can be used whenever although sampling from the joint distribution is difficult or intractable, but sampling from the conditional distributions is (comparatively) easy.

2.7.2 The details of the MCMC scheme

The random variables of interest are $\mathbf{w}, \boldsymbol{\alpha}, \sigma$. Since $p(\mathbf{w} \mid \boldsymbol{\alpha}, \sigma)$ is just normal, we can easily draw samples for $\mathbf{w}$ once we have samples for $\boldsymbol{\alpha}$ and $\boldsymbol{\sigma}$ hence we will only focus on the latter, using a two-pronged approach: first we will consider the noise variance σ^2 fixed and known and describe how we can sample from $p(\boldsymbol{\alpha} \mid \sigma^2, \mathbf{t}, g, h)$ using Reversible Jump Metropolis Hastings sampling. Then we will show how to extend this scheme with an additional Gibbs sampling step to also estimate the noise variance σ^2.

Reversible Jump Metropolis Hastings sampling of α

Since basis functions in the (s)RVM can be turned on or off (corresponding to the relevant α_i being finite or $+\infty$), we need a sampler that can change the model dimension. In the most general case, for such a *Reversible Jump* Metropolis Hastings sampler (Green, 1995) the acceptance ratio for a change from a model $\mathcal{M}_i$ (with dim n_i) to a model $\mathcal{M}_j$ (with corresponding dimensionalities $n_i < n_j$ and parameter vectors $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$) is given by (Ehlers and Brooks, tted):

$$A_{i,j}(\boldsymbol{\theta}_i \to \boldsymbol{\theta}_j) = \frac{p(\mathcal{M}_j, \boldsymbol{\theta}_j) r_j(i)}{p(\mathcal{M}_i, \boldsymbol{\theta}_i) r_i(j) q_{n_j - n_i}(\mathbf{u})} \left| \frac{\partial h_{i,j}(\boldsymbol{\theta}_i, \mathbf{u})}{\partial(\boldsymbol{\theta}_i, \mathbf{u})} \right| \tag{2.101}$$

where $r_j(i)$ is the probability of attempting a jump from $\mathcal{M}_j$ to $\mathcal{M}_i$, q is the the proposal distribution to, h maps $\boldsymbol{\theta}_i, \mathbf{u}$ to $\boldsymbol{\theta}_j$, the final Jacobian term arises because of this change in variables[36]. We use a simple scheme similar to (Denison et al., 2002, p. 53ff), for which the above expression simplifies considerably, as we will see. At each step t we make one of the following proposals:

BIRTH Propose to add a currently inactive basis function with probability b_t.

DEATH Propose to remove a currently active basis function with probability d_t.

MOVE Propose to swap a currently active basis function for one that it is inactive with probability m_t.

Since these three possibilities are exhaustive and mutually exclusive $b_t + d_t + m_t = 1$. We also see that the dimensionality of the model either stays constant or changes by +1 or -1, so $|n_{t+1} - n_t| = 1$. Also, b_t and d_t must clearly depend on n_t, the number of active basis functions at the step, because it is impossible to remove a basis function from an empty model or to add a basis function to a full one. Furthermore because a model with no basis functions is essentially useless, we require always at least one active basis function. Another choice we make is that for non-boundary numbers of components, a MOVE step proposal is as likely as a DEATH or BIRTH step proposal. Together with the requirement of detailed balance this already fixes everything and we obtain the following probabilities:

$$r_i(j) = b_i = \begin{cases} \frac{1}{2} & n_j = 1 \\ \frac{1}{3} & 1 < n_j < S_{\text{max}} \\ 0 & n_j = S_{\text{max}} \end{cases} \qquad r_j(i) = d_j = \begin{cases} 0 & n_j = 1 \\ \frac{1}{3} & 1 < n_j < S_{\text{max}} \\ 1 & n_j = S_{\text{max}} \end{cases}. \tag{2.102}$$

where we use $S_{\text{max}} \leqslant M$ denotes the maximum number of basis functions that we wish to allow in a model. Since exactly one of the three proposals gets made at each step, the move proposal probability simply is $m_i = 1 - d_i - b_i$.

We will further consider three alternative schemes to give birth to a new (i.e. non-∞) α_j:

[36]The Jacobian (determinant) occurs in the multi-dimensional generalization of the simple substitution rule for scalar integrals (in place of the derivative $\frac{dx}{dt}$ for the substitution $x = g(t)$). Changing variables in pdfs is analogous which can be seen by considering that integrating both pdfs (before and after the change of variable) over the analogous bounds must yield the same area; in particular integrating over the whole domain must yield 1.

1. sampling the new α_j, and since we can follow Holmes and Denison (1999) and choose q to be our prior (i.e. exp $\frac{-c}{1+\sigma^2\alpha_j}$), $h = \text{id}$.
2. using a fixed value for α_j, 10^{-5} – following Holmes and Denison (1999) again (*select* scheme), in which case $q(\alpha_j = 10^{-5}) = 1$ and again $h = \text{id}$.
3. using a similar proposal ratio as for the MH case; a zero-centered log-normal with proposal variance s.

Due to the detailed balance constraint we just need to consider two cases when calculating $A_{i,j}$: BIRTH/DEATH (which must be symmetrical) on one hand and MOVE on the other.

BIRTH/DEATH For case 1 we thus obtain the BIRTH (of j) acceptance ratio

$$\begin{aligned}
\log A_{i,j} &= \hat{\mathcal{L}}(\boldsymbol{\alpha}_j) - \hat{\mathcal{L}}(\boldsymbol{\alpha}_i) - \frac{-c}{1+\sigma^2\alpha_j} + \log\left(\frac{d_j}{b_i}\right) && (2.103)\\
&= \hat{\mathcal{L}}(\boldsymbol{\alpha}_j) - \hat{\mathcal{L}}(\boldsymbol{\alpha}_{j\setminus\{i\}}) - \frac{-c}{1+\sigma^2\alpha_j} + \log\left(\frac{d_j}{b_i}\right) && (2.104)\\
&= \hat{\ell}(\alpha_j|\boldsymbol{\alpha}_{-j}) - \frac{-c}{1+\sigma^2\alpha_j} + \log\left(\frac{d_j}{b_i}\right) && (2.105)\\
&= \ell(\alpha_j|\boldsymbol{\alpha}_{-j}) + \log\left(\frac{d_j}{b_i}\right) && (2.106)
\end{aligned}$$

And for case 2

$$\begin{aligned}
\log A_{i,j} &= \hat{\mathcal{L}}(\boldsymbol{\alpha}_j) - \hat{\mathcal{L}}(\boldsymbol{\alpha}_i) - 0 + \log\left(\frac{d_j}{b_i}\right) && (2.107)\\
&= \hat{\mathcal{L}}(\boldsymbol{\alpha}_j) - \hat{\mathcal{L}}(\boldsymbol{\alpha}_{j\setminus\{i\}}) + \log\left(\frac{d_j}{b_i}\right) && (2.108)\\
&= \hat{\ell}(\alpha_j|\boldsymbol{\alpha}_{-j}) + \log\left(\frac{d_j}{b_i}\right) && (2.109)
\end{aligned}$$

Finally, for case 3 we additionally need (using $I[\cdot]$ for the indicator function and $x_i \equiv \log\alpha_i \in [\log(L), \log(H)]$, where L and H are the bounds needed to obtain a normalized distribution for the smoothness prior, see (2.61))

$$\begin{aligned}
q(\alpha_i) &= q(\alpha_i; 0) = \frac{1}{x_i} q(x_i|0) && (2.110)\\
&= I[\log(L) \leqslant x \leqslant \log(H)] \frac{1}{x_i} Z^{-1} \frac{1}{\sqrt{2\pi s}} \exp\left(-\frac{1}{2} x_i^2/s^2\right) && (2.111)\\
Z &= \frac{1}{2}\left[\operatorname{erf}\left(\frac{\log(H)}{\sqrt{2}s}\right) + \operatorname{erf}\left(\frac{-\log(L)}{\sqrt{2}s}\right)\right] && (2.112)
\end{aligned}$$

thus

$$\begin{aligned}
\log A_{i,j} &= \hat{\mathcal{L}}(\boldsymbol{\alpha}_j) - \hat{\mathcal{L}}(\boldsymbol{\alpha}_i) - \log q(\alpha_i) + \log\left(\frac{d_j}{b_i}\right) && (2.113)\\
&= \hat{\ell}(\alpha_j|\boldsymbol{\alpha}_{-j}) - \log\left(\frac{1}{\alpha_i} q(x_i)\right) + \log\left(\frac{d_j}{b_i}\right) && (2.114)\\
&= \hat{\ell}(\alpha_j|\boldsymbol{\alpha}_{-j}) + x_i - \left[\log Z^{-1} - \frac{1}{2}\left(\log(2\pi s^2) + \frac{x_i^2}{s^2}\right)\right] + \log\left(\frac{d_j}{b_i}\right) && (2.115)\\
&= \hat{\ell}(\alpha_j|\boldsymbol{\alpha}_{-j}) + x_i + \log Z + \frac{1}{2}\left(\log(2\pi s^2) + \frac{x_i^2}{s^2}\right) + \log\left(\frac{d_j}{b_i}\right) && (2.116)
\end{aligned}$$

MOVE If we swap component i for i′, (i.e. kill i and introduce i′) and use non-identical births:

$$\begin{aligned}
\log A_{i,i'} &= \hat{\mathcal{L}}(\boldsymbol{\alpha}_{i'}) - \hat{\mathcal{L}}(\boldsymbol{\alpha}_{i}) - \frac{-c}{1+\sigma^2\alpha_{i'}} + \frac{-c}{1+\sigma^2\alpha_i} + \log\left(\frac{m_{i'}}{m_i}\right) && (2.117)\\
&= \hat{\mathcal{L}}(\boldsymbol{\alpha}_{i'}) - \hat{\mathcal{L}}(\boldsymbol{\alpha}_{i}) - \frac{-c}{1+\sigma^2\alpha_{i'}} + \frac{-c}{1+\sigma^2\alpha_i} + 0 && (2.118)\\
&= (\hat{\mathcal{L}}(\boldsymbol{\alpha}_{\setminus i,i'}) + \hat{\ell}(\alpha_{i'}|\boldsymbol{\alpha}_{\setminus i,i'})) - (\hat{\mathcal{L}}(\boldsymbol{\alpha}_{\setminus i,i'}) + \hat{\ell}(\alpha_{i}|\boldsymbol{\alpha}_{\setminus i,i'})) && (2.119)\\
&\quad - \frac{-c}{1+\sigma^2\alpha_{i'}} + \frac{-c}{1+\sigma^2\alpha_i}\\
&= \hat{\ell}(\alpha_{i'}|\boldsymbol{\alpha}_{\setminus i,i'}) - \hat{\ell}(\alpha_{i}|\boldsymbol{\alpha}_{\setminus i,i'}) - \frac{-c}{1+\sigma^2\alpha_{i'}} + \frac{-c}{1+\sigma^2\alpha_i} && (2.120)\\
&= \ell(\alpha_{i'}|\boldsymbol{\alpha}_{\setminus i,i'}) - \ell(\alpha_{i}|\boldsymbol{\alpha}_{\setminus i,i'}) && (2.121)
\end{aligned}$$

case 2, identical birth, gives:

$$\log A_{i,i'} = \hat{\ell}(\alpha_{i'}|\boldsymbol{\alpha}_{\setminus i,i'}) - \hat{\ell}(\alpha_{i}|\boldsymbol{\alpha}_{\setminus i,i'}) \tag{2.122}$$

finally case 3:

$$\begin{aligned}
\log A_{i,i'} &= \hat{\ell}(\alpha_{i'}|\boldsymbol{\alpha}_{\setminus i,i'}) - \hat{\ell}(\alpha_{i}|\boldsymbol{\alpha}_{\setminus i,i'}) + \log q(\alpha_i) - \log q(\alpha_{i'}) && (2.123)\\
&= \hat{\ell}(\alpha_{i'}|\boldsymbol{\alpha}_{\setminus i,i'}) - \hat{\ell}(\alpha_{i}|\boldsymbol{\alpha}_{\setminus i,i'}) - x_i + x_{i'} + \frac{1}{2}\frac{x_{i'}^2 - x_i^2}{s^2} && (2.124)
\end{aligned}$$

Gibbs sampling the noise

In our concrete case the joint distribution is $p(\boldsymbol{\alpha}, \sigma^2 \mid \mathbf{t})$ and the conditional distributions are $p(\boldsymbol{\alpha} \mid \sigma^2, \mathbf{t})$ and $p(\sigma^2 \mid \boldsymbol{\alpha}, \mathbf{t})$; sampling from the former is taken care of by the MH step above and the latter we need[37] to introduce an additional step for which there exist several options. One simple possibility is to make a draw of $\mathbf{w} \mid \boldsymbol{\alpha}$ and use those coefficients to estimate the noise. In this case the likelihood (2.48) is regarded as an exponential density (again using $\beta \equiv \sigma^{-2}$ for notational convenience):

$$p(\mathbf{t} \mid \beta, \mathbf{w}) = (2\pi)^{-N/2}\beta^{N/2}\exp\left\{-\frac{1}{2}(\mathbf{t}-\boldsymbol{\Phi}\mathbf{w})^{\mathrm{T}}(\mathbf{t}-\boldsymbol{\Phi}\mathbf{w})\beta\right\} \tag{2.125}$$

Then taking the same conjugate prior for β, the posterior is

$$p(\beta \mid \mathbf{t}, \mathbf{w}) = \mathrm{Ga}(\beta \mid g + N/2, h + (\mathbf{t}-\boldsymbol{\Phi}\mathbf{w})^{\mathrm{T}}(\mathbf{t}-\boldsymbol{\Phi}\mathbf{w})/2) \tag{2.126}$$

and sampling from this posterior is straightforward.

2.7.3 The complete MCMC algorithm

We have considered multiple possibilities for the proposal distribution q, but once one is chosen the actual algorithm is simple: we loop for a pre-determined number of sample steps, Gibbs sampling alternatingly $\boldsymbol{\alpha}$ and σ^2 (in practice it is generally better not to reestimate σ^2 at each turn, so we only sample σ^2 only every j steps). For the MH sampling of $\boldsymbol{\alpha}$ we make a BIRTH, DEATH or MOVE proposal at each step, according to the proposal probabilities b_t, d_t, m_t outlined above, then a sample α_i' is generated from the proposal distribution q in the case of a BIRTH or a MOVE. This sample (or the drop in dimensionality for a DEATH) is then accepted with probability $A_{t,t'}$,

[37](Holmes and Denison, 1999) sample $beta = \sigma^{-2}$ directly from $\mathrm{Ga}(\beta, g + N/2, h + \mathbf{t}^{\mathrm{T}}\mathbf{t}/2)$, but this appears likely that this is derived by mistakenly identifying 2.48 with an exponential density and taking a conjugate prior of the form $p(\beta \mid g, h) = \mathrm{Ga}(\beta, g, h)$.

otherwise the previous sample is repeated (i.e. $\boldsymbol{\alpha}^{(t+1)} = \boldsymbol{\alpha}^{(t)}$). We dedicate a certain number of steps during which no samples are discarded to burn-in. After that we store every kth sample.

2.7.4 MCMC: Results and Conclusion

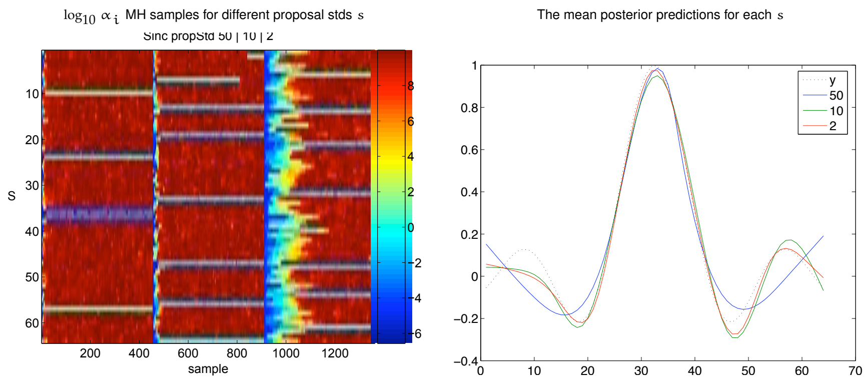


Figure 2.25: *Left panel:* Horizontally concatenated MH samples for $\boldsymbol{\alpha}$ (color-coded in log-scale) with different proposal standard deviations s (the values are 50, 10 and 2) for lspline 3.0 Sinc data (N=M=64, SNR=5). Even with a quite high rejection rate (~ 0.05 for propStd 50), the sampler gets stuck in "furrows" and moreover these furrows are pretty arbitrary as is evinced by the fact that all three sample runs fell into quite different ones. *Right panel:* The corresponding averaged mean posterior predictions $\hat{\mathbf{y}}$.

Unfortunately our repeated attempts to implement a viable MCMC scheme did not meet with success. Despite trying various different variations on the outlined scheme (no RJ, no moves in RJ sampling, different proposal densities for "born" α_i etc. – and of course different values for tuning parameters such as the proposal density), Figure shows a typical outcome: the sampler gets stuck in a local minimum. For this reason

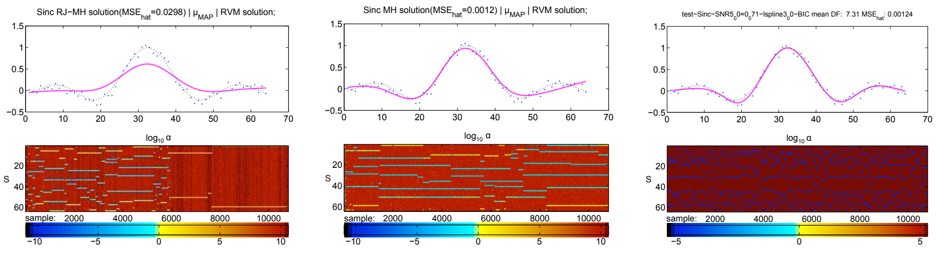


Figure 2.26: Different MCMC sampling schemes tested on the Sinc data using a large number of samples each time. *Left panel:* A long run of Reversible Jump Metropolis Hastings (RJMH) samples of $\boldsymbol{\alpha}$ (again color-coded on a log-scale) and the resultant mean posterior prediction $\hat{\mathbf{y}}$ for Sinc. The mixing is fairly poor and the model suddenly degenerates to a drastic underfit with only two active components in the right half of the figure. *Middle panel:* A long run of MH-only samples, again showing poor mixing.*Right panel:* RJ only with moves disabled – active α_i are always set to 10^{-5}. This approach gave us the best mixing behavior we observed, but somewhat defeats the point of our complex probabilistic model. Furthermore, even the RJ-only approach did not always mix well (see Figure 2.7.4, *right panel*)

The poor mixing is not just an artifact of running the model for a few iterations, the examples in Figure 2.7.4 were run for 4×10^5 steps, retaining every 37th sample.

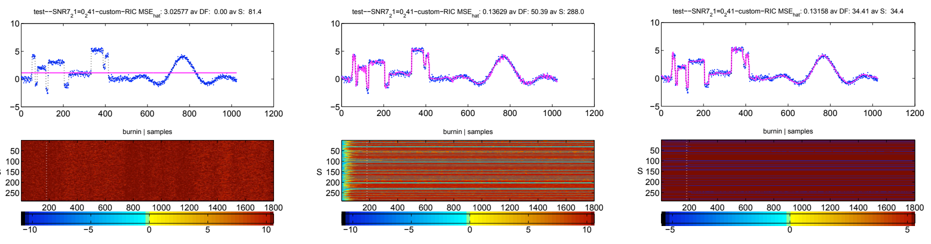


Figure 2.27: BlocksSinc sampling analogous to Figure 2.7.4, using fewer samples and a M-reduced "pseudo-overcomplete" dictionary for reasons of computational expense. *Left panel:* The full RJMH sampling fails completely. *Middle panel:* MH-only sampling produces an acceptable mean prediction $\hat{\mathbf{y}}$ but gets trapped in a single local maxima. *Right panel:* RJ only – active α_i are always set to 10^{-5}, but this time even that approach fails to show any mixing.

There are several possible explanations, not least that tuning MCMC parameters remains somewhat of a black art that can require a considerable amount of trial and error[38]. It is thus quite possible that further time and perseverance would have eventually overcome these difficulties.

However, as has been mentioned in the introduction the RVM is in many ways an uneasy match with MCMC. The principal attraction of MCMC is that it makes faithful modeling of complex probabilistically governed events feasible. The principal attraction of the RVM, on the other hand, consists of retaining a good measure of probabilistic interpretability whilst trading in some Bayesian principledness for good performance and sparsity.

But sparsity is destroyed by the model averaging implicit in MCMC, and so is the good performance (in the case of our implementations there was an difference by several orders of magnitude between fRVM and the MCMC). Conversely the degenerate variance structure of the RVM (see Rasmussen and Quiñonero-Candela, 2005), which is one of the chief reasons for its superior performance to (non-degenerate) Gaussian Process models, results in unreasonable posterior variances which are not ameliorated by MCMC, and that in turn negates much of the value of MCMC from the point of view of principledness.

Notwithstanding these observations, there are not inconsiderable indirect benefits of a well working MCMC scheme for RVM, such as diagnostic value and insight into the search space (e.g. how well does the normal RVM do in finding the global minimum for certain types of problems? How bumpy/multimodal is the search space?).

On the whole, however, our impression after several attempts and variations, is that obtaining a reliably working MCMC scheme for the RVM appears to be quite hard and even if successful would only offer a compromised increase in probabilistic rigour for a severe degradation in performance. We are thus tempted to conclude that the fRVM-implementation of the sRVM occupies a sweet spot in the efficiency/principledness continuum that will be very difficult to match with an MCMC based scheme[39]. If greater probabilistic felicity is desired an efficient implementation of a (non-degenerative) Gaussian Process model might thus be a more promising target.

2.8 Discussion of the sRVM for regression

Whilst we have concentrated on the fRVM framework (Faul and Tipping, 2002), since our implementation is based on it, it is worth mentioning that the fRVM is by no means the only attempt

[38]Assessing convergence in MCMC is generally difficult and the particular methods we considered are fairly sensitive to parameters like step-size (MacKay, 2003).

[39]In fact Bishop and Tipping (2000) made a somewhat similar experience with their variational implementation, which according to its authors, turned out to yield the same results for vastly greater computational expense.

to provide a scheme that is computationally more efficient than the original, "slow" RVM (Tipping, 2000) and might be adapted to incorporate a smoothness prior; we draw attention to the Subspace EM (SSEM) algorithm (Quiñonero-Candela, 2004) and a version based on a Bayesian interpretation of backfitting (D'Souza et al., 2004).

Before examining more closely the issue of different choices for $p(\boldsymbol{\alpha})$, we mention other work pertinent to RVM learning. Wipf and Rao (2004) provide a principled justification for approximating the hyperparameter posterior $p(\boldsymbol{\alpha}, \sigma^2 \,|\, \mathbf{t})$ with the point estimates $\boldsymbol{\alpha}_{\text{MAP}}$ and σ_{MAP}. Figueiredo (2003) provides an illuminating perspective on sparse Bayesian learning and presents an Expectation Maximisation algorithm for learning the coefficients $\mathbf{w}$ directly, treating the $\boldsymbol{\alpha}$ as hidden data. Quiñonero-Candela (2004); Rasmussen and Quiñonero-Candela (2005) offer an augmentation to the RVM at the prediction stage to ameliorate the problem of artificially low predictive variances for test-points that are far off the "centers" of the "relevance vectors" (i.e. the final set of basis functions for well-localized kernels such as gauss) – an issue that may be regarded as an undesirable side-effect of sparsity. Whilst their original scheme is more of expository value because, amongst other things, it increases the cost of predictions dramatically[40], Quiñonero-Candela et al. (2007), have recently proposed a technique of more practical interest that involves adapting the basis functions.

2.8.1 Other prior choices for α

A scheme that shares some conceptual similiarity with ours is (Kroptov and Vetrov, 2007). Whereas we approximate the degrees of freedom by the trace of the smoothing matrix $\mathbf{S}$ (which equals the sum of $\mathbf{S}$'s eigenvalues), (Kroptov and Vetrov, 2007) apply the ARD-style regularization to the eigenvectors of the $\boldsymbol{\Sigma}_{\text{LP}}$, rather than the weights $\mathbf{w}$. Both a standard (Gaussian) ARD-style prior and a Laplacian ARD-style prior are considered, and can be respectively be expressed in $\mathbf{w}$-space by using the eigenvalue decomposition $\boldsymbol{\Sigma}_{\text{LP}} = \mathbf{Q}^{\mathsf{T}}\boldsymbol{\Lambda}\mathbf{Q}$:

$$p(\mathbf{w} \,|\, \boldsymbol{\alpha}) = \frac{\sqrt{|\mathbf{A}|}}{(2\pi)^{M/2}} \exp(-\frac{1}{2}\mathbf{w}\mathbf{Q}^{\mathsf{T}}\mathbf{A}\mathbf{Q}) \qquad \text{(Gaussian prior)} \qquad (2.127)$$

$$p(\mathbf{w} \,|\, \boldsymbol{\alpha}) = \frac{|\mathbf{A}|}{4^M} \exp(-\frac{1}{2}\sum_{i}^{M} |\sum_{j}^{M} q_{ij}w_j|) \qquad \text{(Laplacian prior)} \qquad (2.128)$$

As we noted in section 2.5, any super-Gaussian prior on each $p(w_m)$ will encourage sparseness or shrinkage. A natural prior that has been used to promote sparsity in a variety of contexts is the Laplacian prior, $p(w_m) \propto e^{-|w_m|}$, which leads to the LASSO (least absolute shrinkage and selection operator) scheme (Tibshirani, 1996), although in this context the prior is introduced as the penalty in a penalized likelihood. As Figueiredo (2003) shows, the Laplacian prior on $p(w_m)$ may be obtained via a hierarchical scheme, like ours, in which a $p(w_m)$ arises as a scale mixture of Gaussians with an exponential prior on the precisions: $p(\alpha_m \,|\, \gamma) \propto e^{-\gamma\alpha/2}$, where γ is a hyperparameter. In fact, Figueiredo abandons the exponential/Laplacian scheme in favour of a Jeffreys' prior on α_m, namely $p(\alpha_m) \propto 1/\alpha_m$, which in turn results in a similar very heavy-tailed prior on the coefficients: $p(w_m) \propto 1/|w_m|$. The attractions of the Jeffreys' prior result from the fact that it is a *non-informative* prior (Bernardo and Smith, 1994): first, it is scale invariant and, secondly, there are no (hyper)parameters to adjust. Before examining the Jeffreys' prior in more detail we first discuss the Gamma prior which has the Jeffreys' prior as a limiting case.

[40]"The RVM* is thus [...] not necessarily an interesting algorithm in practice." (Rasmussen and Quiñonero-Candela, 2005)

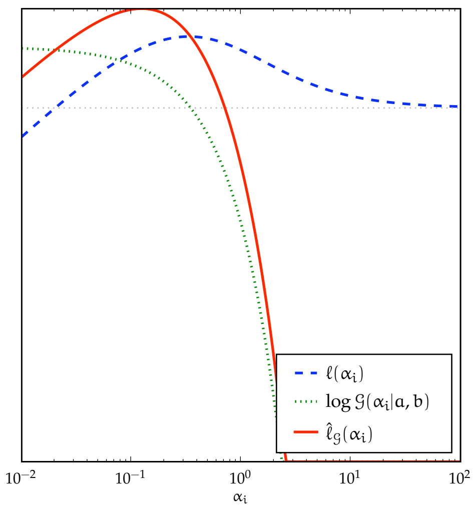


Figure 2.28: Log likelihoods, $\ell(\alpha_i)$ (dashed), and log gamma prior, $\log \mathcal{G}(\alpha_i \,|\, a, b)$ (dotted), and finally the log posteriors for a Gamma prior, $\hat{\ell}_{\mathcal{G}}(\alpha_i)$ (solid), plotted versus $\log \alpha_i$, showing that in the case of a Gamma prior, with $q_i = 1$, $s_i = 2$, $a = 1$ and $b = 2$ the MAP α_i is less sparse than the maximum likelihood solution – something that will never occur with the smoothness prior and the corresponding posterior $\hat{\ell}(\alpha_i)$.

The Gamma prior

$$p(\alpha_m \,|\, a, b) = \mathcal{G}(\alpha_m \,|\, a, b) = \frac{b^a}{\Gamma(a)} \alpha_m^{a-1} e^{-b\alpha_m} \tag{2.129}$$

has two hyperparameters, $a > 0$ and $b > 0$, which respectively control the shape and width of the density. This prior, which leads to a Student-t $p(w_m)$, is considered by (Tipping, 2001; Wipf and Rao, 2005). However it is not clear how the hyperparameters a and b are to be chosen, except by cross-validation, which is a data and time consuming procedure, or via a variational approach, which, in the formulation that corresponds closest to the classical RVM Bishop and Tipping (2000), is neither computationally efficient nor of clear practical value[41]. Furthermore, as illustrated in Figure 2.28, it is possible for the Gamma prior with particular values of a and b to yield $\hat{\alpha}_i < \alpha_i^\star$, that is a MAP α_i that is less sparse than the $\alpha_i^\star$ which maximizes the likelihood alone. By contrast the smoothness prior always results in $\hat{\alpha}_i > \alpha_i^\star$ and we point out that the smoothness prior is strictly increasing and so always assigns increasing weight to increasing precisions (c.f. Figure 2.11).

The Jeffreys' prior, a uniform density on the logarithmic scale, is obtained in the limit $a, b \to 0$, which (Tipping, 2001) appears to advocate for the RVM although the fRVM (Tipping and Faul, 2003) clearly uses a uniform prior in "un-logged space"—called the **None** prior here. In this limit the Student-t density for $p(w_i)$ becomes $1/|w_i|$. However, the analogous component-wise maximization scheme leads to models in which all components are active when $a < 1/2$ because $\hat{\ell}(\alpha_i)$ is maximized at $\alpha_i = 0$ regardless of the values of s_i, q_i and b (see Appendix A.6). Approaching the Jeffreys' prior by $p(\alpha_i) \propto \alpha_i^\zeta$ as $\zeta \to -1$ leads to the same conclusion: for $\zeta < -1/2$ every component is active because $\hat{\ell}(\alpha_i)$ is maximized at $\alpha_i = 0$ (Appendix A.6).

Thus although the scale invariance and the absence of hyperparameters of Jeffrey's prior is appealing, the type II MAP solution sought here does not accommodate it. However, the smoothness prior, which is noise-dependent and therefore confers scale invariance with invariant SNR, has a single hyperparameter and is readily interpretable in terms of the solution sparseness, the

[41]See (Tipping, 2001, footnote 6) or `<http:www.miketipping.com/index.php?page=rvm>`: "Note that the 'variational' relevance vector machine is pretty much identical to the non-variational version, but is a lot slower to train."

degrees of freedom in the smoothing matrix.

As has already been noted in the introduction, the Holmes and Denison (1999) smoothness prior is also suitable for non-wavelet design matrices because, unlike most popular wavelet shrinkage priors, it is not dependent on the wavelet length scale or level. But there is, in principle, no reason not to incorporate priors in the RVM that only work in conjunction with certain kernel types.

Finally we note that a further alternative hierarchical prior to address the under-determination of the $\mathbf{w}$ is explored in (Fokoué et al., 2004), while Girolami and Rogers (2005) (and references therein) pursue a completely different avenue: a Bayesian treatment of kernel construction itself.

2.8.2 Summary and Conclusion

We have presented a straightforward extension to the RVM that imposes a more stringent prior on the variance of the weights in nonlinear regression, and we have described an efficient algorithm for maximizing the marginal posterior probability of the model.

From a theoretical perspective we have seen that unlike other proposed prior types (such as the implicit uniform prior in the original RVM implementation, or a Gamma prior) the smoothness prior we presented is noise-dependent in a principled fashion (data/kernel rescaling whilst keeping the SNR fixed does not change the result and, as one would expect, setting $\hat{\sigma}^2$ to a multiple or fraction of the real σ^2 in experiments results respectively in a sparser or less sparse regression).

Further, our results indicate that symmlets with a smoothness prior make an attractive default choice for RVM regression tasks: the combination is flexible enough to be suitable for a large variety of signals, requires no additional kernel parameters to be determined by cross-validation (e.g. scale for Gaussian kernels). The hyperparameter c could be optimised by cross-validation, but our experiments show that the BIC choice works well for a wide range of problems. The sRVM has attractive computational characteristics resulting from the properties of wavelets. In particular the matrix-multiplication by dictionary columns can be carried out by the mathematically equivalent but much more efficient discrete wavelet transform ($O(N)!$); this implies that no $N \times M$ design matrix needs to be constructed and held in memory and that the per-step time complexity drops from cubic in S to linear in N. Furthermore numerical robustness also tends to be better than for many other dictionaries.

This might seem to beg the question “why not just use wavelet shrinkage to start with?” – of course there are limitations of wavelets that other types of dictionary do not share (the data must be equally spaced) and although symmlets perform well across a wide range of signals one might find in practice, it is difficult to beat the performance of less general kernels for tasks for which they are particularly well suited (e.g. lsplines for Sinc-like data).

But the deeper point is that the RVM updated with a smoothness prior (sRVM) can be profitably regarded as a *generalization* of wavelet shrinkage.

Figure 2.14 demonstrates that we can even obtain the best of both worlds in *one and the same experiment* by using an overcomplete dictionary composed of different dictionary types (such as Haar wavelets and thin-plate splines) that each capture certain aspects of the overall signal particularly well and then rely on the sRVM to automatically select a sparse representation from this overcomplete dictionary.

In other words a chief attraction of the sRVM for regression is that spans a bridge between the RVM and related methods on the one hand and wavelet shrinkage on the other, yielding a powerful synthesis.

Part II

Classification and the smooth Relevance Vector Machine

2.9 Extending the RVM for regression for classification

To deal with binary[42] categorical variables Tipping (2001) uses the standard sigmoid link function in order to derive a classifier version from the regression RVM:

$$\sigma(y) = \frac{1}{1+e^{-y}} \tag{2.130}$$

Consequently $y = \boldsymbol{\phi}(\mathbf{x})^T\mathbf{w}$ in the regression context becomes

$$y = \sigma(\boldsymbol{\phi}(\mathbf{x})^T\mathbf{w}) \tag{2.131}$$

(Note that the use of σ to denote the sigmoid in classification context is completely unrelated to the use of σ^2 to denote the variance in the regression context – this confusing notational choice is solely motivated by a desire to conform with established convention).

Unfortunately, the introduction of the sigmoid to the model results in several complications. As far as the the classical RVM (sans smoothness prior) is concerned, it means that the weights, **w** can no longer be marginalized, necessitating an approximation scheme (we follow Tipping (2001) who uses a Laplace approximation). As for the smoothness prior, we find that optimizing the DF w.r.t. α_i becomes much harder because treating the resultant smoothing matrix **S** as approximately orthogonal is out of the question. As we will see this is because an awkward diagonal matrix **B** will take the place of the simple scalar σ^{-2} in all expressions. After recapitulating the Laplace approximation, we will detail our approach for dealing with this issue.

2.10 IRLS and the Laplace approximation

The Laplace approximation consists of approximating an unwieldy, possibly unnormalized, distribution $Q(\theta)$ by fitting a Gaussian at its (Q's) mode **m**, and using this Gaussian in lieu of Q. This can be seen to be equivalent, up to an additive constant, to a 2nd order Taylor expansion of the (unnormalized) log-likelihood

$$\ln Q(\boldsymbol{\theta}) = \ln Q(\mathbf{m}) - \nabla(\ln Q)(\mathbf{m})(\boldsymbol{\theta}-\mathbf{m}) - \frac{1}{2}(\boldsymbol{\theta}-\mathbf{m})^T\nabla\nabla(\ln Q)(\mathbf{m})(\boldsymbol{\theta}-\mathbf{m}) + \dots,$$

because the log-Gaussian is a quadratic function of $\boldsymbol{\theta}$ and the mode **m** is an extremal point, so that the linear term vanishes: $\nabla(Q)(\mathbf{m}) = \mathbf{0}$.

Consequently we have:

$$\ln Q(\boldsymbol{\theta}) \simeq \ln \mathcal{N}(\boldsymbol{\theta} \,|\, \mathbf{m}, (\nabla\nabla(\ln Q)(\mathbf{m}))^{-1}) + \text{const} \tag{2.132}$$

$$\text{const} = \ln Q(\mathbf{m}) - \ln \frac{\det(\nabla\nabla(\ln Q)(\mathbf{m}))^{1/2}}{2\pi^{M/2}} \tag{2.133}$$

For a two class-model in which we assume individual targets t_n to be independent, we expect them to follow a Bernoulli distribution: $p(t_n|\mathbf{w}) = y_n$ if $(t_n = 1)$ else $1 - y_n = y_n^{t_n}(1-y_n)^{(1-t_n)}$,

[42]We shall limit ourselves to dichotmization, but the fairly straightforward of the RVM to $K > 2$-class problems is discussed in Tipping (2001). Although the naturalness with which the RVM can deal with multi-class problems in a principled fashion contrasts favorably with the SVM, it comes at the price of an additional K^3 factor in the training complexity.

$y_n = \sigma((\mathbf{\Phi}_{[n,:]]})^T \mathbf{w})$ and because of independence

$$p(\mathbf{t} \,|\, \mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{(1-t_n)} \tag{2.134}$$

$$\tag{2.135}$$

so (writing again $\mathbf{A} \equiv \operatorname{diag} \boldsymbol{\alpha}$):

$$\ln p(\mathbf{w} \,|\, \mathbf{t}, \boldsymbol{\alpha}) = \ln \frac{p(\mathbf{t} \,|\, \mathbf{w}) p(\mathbf{w} \,|\, \boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\alpha})} \tag{2.136}$$

$$= \sum_{n}^{N} [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)] - \frac{1}{2} \mathbf{w}^T \mathbf{A} \mathbf{w} + \mathrm{const} \tag{2.137}$$

For a fixed $\boldsymbol{\alpha}$, this expression can be maximized w.r.t. to $\mathbf{w}$ by using the Iterated Reweighted Least Squares (IRLS) algorithm (see, e.g. Nabney, 2002, pp. 132ff).

Having obtained the mode of $\ln p(\mathbf{w} \,|\, \mathbf{t}, \boldsymbol{\alpha})$, we can thus employ the Laplace approximation, and fit a Gaussian with mean $\boldsymbol{\mu}_{\mathrm{LP}}$ and variance $\mathbf{\Sigma}_{\mathrm{LP}}$, i.e. $\ln p(\mathbf{w} \,|\, \mathbf{t}, \boldsymbol{\alpha}) \simeq \mathcal{N}(\boldsymbol{\mu}_{\mathrm{LP}}, \mathbf{\Sigma}_{\mathrm{LP}})$ with:

$$\boldsymbol{\mu}_{\mathrm{LP}} = \mathbf{A}^{-1} \mathbf{\Phi}^T (\mathbf{t} - \mathbf{y}) \tag{2.138}$$

$$\mathbf{\Sigma}_{\mathrm{LP}} = (\mathbf{\Phi}^T \mathbf{B} \mathbf{\Phi} + \mathbf{A})^{-1} \tag{2.139}$$

where we have defined

$$\mathbf{B} \equiv \operatorname{diag}[y_n(1 - y_n)]_{n=1}^{N} \tag{2.140}$$

We are therefore now in a position to approximate the marginalization of $\mathbf{w}$:

$$p(\mathbf{t} \,|\, \boldsymbol{\alpha}) = \int p(\mathbf{t} \,|\, \mathbf{w}) p(\mathbf{w} \,|\, \boldsymbol{\alpha}) d\mathbf{w} \tag{2.141}$$

$$\simeq \sqrt{(2\pi)^M |\mathbf{\Sigma}_{\mathrm{LP}}| p(\mathbf{t} \,|\, \boldsymbol{\mu}_{\mathrm{LP}}) p(\boldsymbol{\mu}_{\mathrm{LP}} \,|\, \boldsymbol{\alpha})} \tag{2.142}$$

And thus the approximate log marginal likelihood can again be written in the form

$$\ln p(\mathbf{t} \,|\, \boldsymbol{\alpha}) = -\frac{1}{2} \left(N \ln(2\pi) + \ln |\mathbf{C}| + \tilde{\mathbf{t}}^T \mathbf{C}^{-1} \tilde{\mathbf{t}} \right) \tag{2.143}$$

$$C \equiv \mathbf{B} + \mathbf{\Phi} \mathbf{A} \mathbf{\Phi}^T \tag{2.144}$$

$$\tilde{\mathbf{t}} = \mathbf{\Phi} \boldsymbol{\mu}_{\mathrm{LP}} + \mathbf{B}^{-1} (\mathbf{t} - \mathbf{y}) \tag{2.145}$$

So that the fRVM scheme (Faul and Tipping, 2002) can be applied again.

2.10.1 IRLS in more detail

It is instructive to have a closer look at IRLS and the approximation that are involved in the classification setup, so we shall present a quick overview of the algorithm. We know that the gradient at the minimum, $\nabla \mathbf{f}(\hat{\mathbf{w}}_{\min})$, must be zero, and we can do a first order Taylor expansion of it around the current value $\hat{\mathbf{w}}_{\mathrm{now}}$. Doing that and re-arranging shows us that...

$$\mathbf{0} = \nabla \mathbf{f}(\hat{\mathbf{w}}_{\min}) \simeq \nabla \mathbf{f}(\hat{\mathbf{w}}_{\mathrm{now}}) + \nabla^2 \mathbf{f}(\hat{\mathbf{w}}_{\mathrm{now}})(\hat{\mathbf{w}}_{\min} - \hat{\mathbf{w}}_{\mathrm{now}}) \tag{2.146}$$

$$\Rightarrow$$

$$\hat{\mathbf{w}}_{\min} \simeq \hat{\mathbf{w}}_{\mathrm{now}} - (\nabla^2 \mathbf{f}(\hat{\mathbf{w}}_{\mathrm{now}}))^{-1} \nabla \mathbf{f}(\hat{\mathbf{w}}_{\mathrm{now}}) \tag{2.147}$$

…we can derive a "guess" for the location of the minimum $\hat{\mathbf{w}}_{\mathrm{min}}$ based on the current position $\hat{\mathbf{w}}_{\mathrm{now}}$ and the gradient and inverse Hessian at that position. This above (quadratic) approximation would be an equality if $\mathbf{f}$ where quadratic, so it is just a good estimate when $\hat{\mathbf{w}}_{\mathrm{now}}$ is sufficiently close to $\hat{\mathbf{w}}_{\mathrm{min}}$. It does however suggest the following update scheme – the well-known Newton-Raphson algorithm:

$$\hat{\mathbf{w}}_{\mathrm{next}} := \hat{\mathbf{w}}_{\mathrm{now}} - \mathbf{H}^{-1}\nabla\mathbf{f}(\hat{\mathbf{w}}_{\mathrm{now}}) \qquad \mathbf{H} \equiv \nabla^2\mathbf{f}(\hat{\mathbf{w}}_{\mathrm{now}}) \tag{2.148}$$

The reason Newton-Raphson is particularly attractive for GLM optimization is firstly, that overshooting the minimum is not much of an issue with GLMs and secondly, that $\hat{\mathbf{w}}_{\mathrm{now}}$ can be initialized with a value quite close to the minimum (which is the main reason it can be several times faster for this particular application than methods like scaled-conjugate gradients or quasi Newton (Nabney, 2002, p. 133f))

2.11 The derivative of the trace of S in the general case

First we note that we can rewrite

$$\begin{aligned}
\mu_{\mathrm{LP}} &= \mathbf{A}^{-1}\mathbf{\Phi}(\mathbf{t}-\mathbf{y}) && (2.149)\\
&= (\mathbf{\Phi}^{\mathsf{T}}\mathrm{B}\mathbf{\Phi} + \mathbf{A} - \mathbf{\Phi}^{\mathsf{T}}\mathbf{B}\mathbf{\Phi})^{-1}\mathbf{\Phi}(\mathbf{t}-\mathbf{y}) && (2.150)\\
&= (\mathbf{\Sigma}_{\mathrm{LP}}^{-1} - \mathbf{\Phi}^{\mathsf{T}}\mathbf{B}\mathbf{\Phi})^{-1}\mathbf{\Phi}(\mathbf{t}-\mathbf{y}) && (2.151)\\
&= (\mathbf{I} - \mathbf{\Sigma}_{\mathrm{LP}}^{-1}\mathbf{\Phi}^{\mathsf{T}}\mathbf{B}\mathbf{\Phi})^{-1}\mathbf{\Sigma}_{\mathrm{LP}}\mathbf{\Phi}^{\mathsf{T}}(\mathbf{t}-\mathbf{y}) && (2.152)\\
&= \mathbf{\Sigma}_{\mathrm{LP}}\mathbf{\Phi}^{\mathsf{T}}\mathbf{B}\mathbf{\Phi}\mu_{\mathrm{LP}} + \mathbf{\Sigma}_{\mathrm{LP}}\mathbf{\Phi}^{\mathsf{T}}(\mathbf{t}-\mathbf{y}) && (2.153)\\
&= \mathbf{\Sigma}_{\mathrm{LP}}\mathbf{\Phi}^{\mathsf{T}}\mathbf{B}(\mathbf{\Phi}\mu_{\mathrm{LP}} + \mathbf{B}^{-1}(\mathbf{t}-\mathbf{y})) && (2.154)\\
&= \mathbf{\Sigma}_{\mathrm{LP}}\mathbf{\Phi}^{\mathsf{T}}\mathbf{B}\tilde{\mathbf{t}} && (2.155)
\end{aligned}$$

Comparing this expression to (2.54) shows that the linearization of the classification problem effected by the Laplace approximation corresponds to solving a (roughly) equivalent regression problem with *heteroscedastic* noise (Tipping and Faul, 2003), i.e.

$$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{B}) \tag{2.156}$$

This is good news, because it gives reason to assume that we can reuse much of our existing machinery from the regression RVM implementation. However, from (2.155) we can also see that the equivalent smoothing matrix for this peseudo-regression problem is given by:

$$\mathbf{S} = \mathbf{\Phi}\mathbf{\Sigma}_{\mathrm{LP}}\mathbf{\Phi}^{\mathsf{T}}\mathbf{B} \tag{2.157}$$

The bad news is that the $\mathbf{B}$ term taking the place occupied by σ^{-2} in the regression-version of $\mathbf{S}$ (which is $\mathbf{\Phi}\mathbf{\Sigma}\mathbf{\Phi}^{\mathsf{T}}\sigma^{-2}$, and given in (2.55)) means that even with orthogonal $\mathbf{\Phi}$ we can no longer compute the DF $= \operatorname{tr}\mathbf{S}$ using a simple expression like (2.58)[43], making it infeasible to completely isolate the contribution of a single α_i for the purpose of maximizing $\hat{\ell}(\alpha_i)$.

Nevertheless, with a bit of work it is still possible to obtain an expression for the derivative of $\operatorname{tr}\mathbf{S}$ that allows a quite similar optimization scheme to the one that has already been used for the orthonormal case (see Appendix A.8 for details). The key result is that we can write:

$$\ell'(\alpha_i + \Delta) = -(\Delta\mathbf{\Sigma}_{[i,i]} + 1)^{-2} \times \mathrm{const} \tag{2.158}$$

[43] $\mathrm{DF} = \operatorname{tr}\mathbf{S} = \sum_{m=1}^{M}(1+\sigma^2\alpha_m)^{-1}$; this identity which only applies to the regression case with orthonormal $\mathbf{\Phi}$ relies on using the cyclic trace identity (see B.1) to re-arrange to $\mathbf{\Phi}^{\mathsf{T}}\mathbf{\Phi} = \mathbf{I}$ inside the trace

allowing us to find the optimal $\alpha_i^\star$ by maximizing Δ relative to some starting value for α_i (e.g. the **None**-solution); the solution again turns out to be the root of a polynomial. In summary, the extra work required for the classification case amounts to finding $\boldsymbol{\mu}_{LP}$ via IRLS, substituting $\mathbf{B}$ for σ^{-2} in all expressions and $\tilde{\mathbf{t}}$ for $\mathbf{t}$, and then proceeding as before, only that this time we have to choose an appropriate starting value for α_i in order to then find $\alpha_i^\star = \alpha_i + \text{argmax}_\Delta\, \ell'(\alpha_i + \Delta)$, again via finding the roots of a (different) polynomial. On the other hand, we no longer need to maximize the posterior wrt. to σ^2 as the "noise" values $\mathbf{B}_{[i,i]]}$ are found during IRLS.

Chapter 3

Classification Results and Summary

It is our opinion that the effects of smoothness are more difficult to understand than in the regression case: unlike (low-dimensional) regression and particularly denoising where overfitting is often easily visually diagnosed, the extent of the overfit and the quality of the improvement achieved by regularization is not always as easily recognized for classification. Firstly classification problems involving only two or three dimensions are far less common than equivalent regression problems (and the 3D case is often much harder to visualize). Furthermore, what we might consider as overfitting in classification consists of more than a single phenomenon: apart from overly "wiggly" decision boundaries we also have to consider "black-and-white thinking" or unusually sharp and clear-cut decision boundaries (to elaborate the latter distinction: assume a pair of points that are on opposite sides of the decision boundary, but in close spatial proximity – if our model often, and with great confidence, assigns class A membership probabilities to each member of this pair that are respectively close to 0 and 1, we might consider this a form of overfitting even if the decision boundary itself is not overly wiggly, because in addition to decision boundary smoothness we also expect the posterior likelihood of class membership for adjacent points not too differ too sharply). To allow us to examine and tease these issues apart in more detail we wrote a simple script to generate 2D toy classification data with adjustable "wiggliness" and known class probabilities (see Figure 3.1).

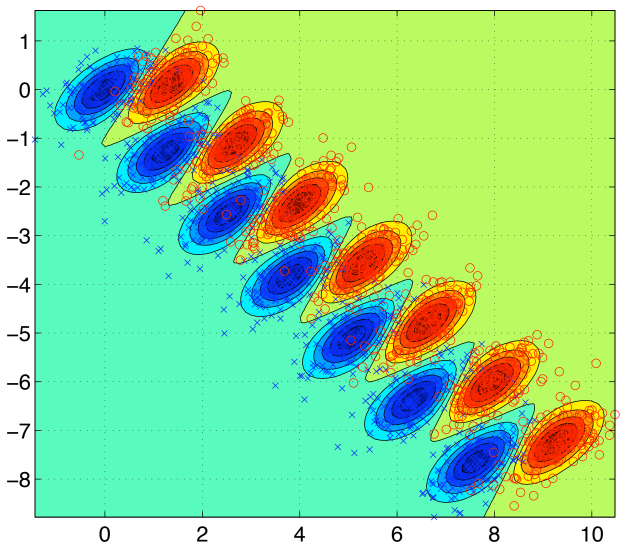


Figure 3.1: Gausssnake fake data example. Gausssnake is a simple matlab script to create "wiggly" toy data from a mixture of Gaussians with known decision boundaries and class probabilities specifically to explore the effects of the smoothness prior for classification data. User adjustable parameters govern the separation of the two classes, and the number and shape of the Gaussians. The black line is the decision boundary, the examplars for classes A and B are respectively displayed in blue and red, and the contour colors indicate the class probabilities.

Unfortunately we found that the implementation of the sRVM classification described in the preceding section, although functioning to some extent, performed rather poorly due to convergence issues. It was impossible to reach convergence with the stricter criterion we described in the RVM for regression implementation and although using the slightly laxer criterion suggested by (Tipping and Faul, 2003) allowed convergence, the results, although not terrible, were not very consistent and generally not superior to the plain RVM. This held true both for our gausssnake data (see Figure 3.2) as well as standard data sets such as tremor, iris or ionosphere – despite the fact that the latter clearly suffered from overfitting with the plain RVM. In view of the fact that local maxima appear to be a fundamental issue with the RVM (section 2.6.6) we decided that rather than spending more even more tuning effort on the RVM[1], a fundamentally less greedy approach was called for and we opted to attempt an evolutionary scheme.

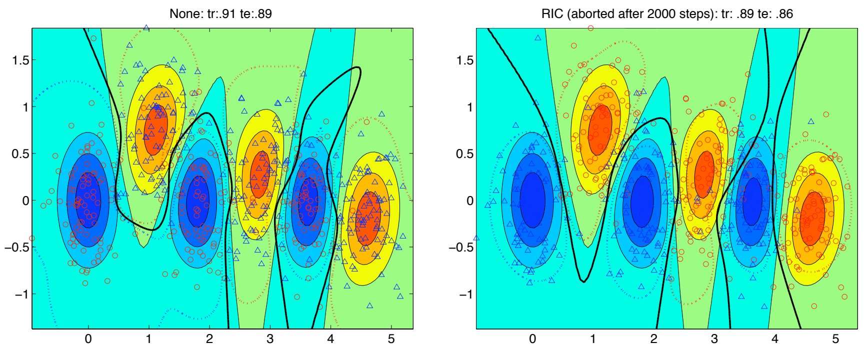


Figure 3.2: Some gausssnake results with None prior (left) and with RIC prior (right).

[1]We did nonetheless still attempt different implementation variants of the sRVM classification scheme: thus we tried first computing the α_i^{NONE} solution and then updating that value for the same i in the next step to find the mode of $\hat{\ell}$ using our formula for $\ell(\alpha_i + \Delta)$. We also attempted alternating between finding α_i^{NONE} and then the optimal Δ for $i \neq j$. However, frustratingly, none of these variants converged properly.

3.0.1 Something completely different: TP/FP-rate Evolutionary Optimization of ℓ_q for penalized logistic kernel regressors

The RVM does provide both a probabilistic output for class-membership and a measure of confidence on a per-data point basis (the latter, as has been mentioned, with the important qualification that large distance from the training set points in feature space will result in unreasonably high confidence).

However, in many settings only a bulk estimate of prediction certainty is needed. Specifically, if the two classes in dichomatization map neatly to two possible actions (i.e. A or B, as opposed to A, B or consult a specialist if unsure) and the costs of wrongly choosing option A over option B and B over A are fixed, the ability to choose the point on the ROC curve[2] that minimizes the sum of the probability of each misclassification weighted by its cost is completely sufficient.

We made a preliminary attempt at a TP/FP ROC curve optimization with evolutionary methods and the results appeared promising, although more work is needed to judge the full potential. We again assume two classes one which we will call "positives" and one which we will call "negatives". Our approach is based on the work of Smith (2006); Everson and Fieldsend (2006) and consists of the following basic idea: it is possible to recast finding the "right" sparse classifier as a problem of multi-objective optimisation, in which we do not use the classification error as an objective directly but instead maximize for True Positive Rate (the number of positive training samples correctly classified as such as a fraction of the total number of positive training samples), whilst minimizing the False Positive Rate (FPR; the fraction of training samples that are incorrectly classified as positive, divided by the number of negative training samples). An ROC curve simply graphs different models in TP/FP space (typically the abscissa is used for TPR and the ordinate FPR) and provides a convenient way for an operator to choose the model that delivers the best trade-off for his criteria.

By generating a manifold of mutually non-dominant[3] models according to the above criteria, we can create models that will also generally yield a low overall classification error. The optimal manifold (i.e. the one that covers the largest possible area) is known as the *Pareto set*. To approximately generate this optimal manifold we pick the current model and apply a random mutation. We accept the mutant (making it our current model) if it is not dominated (in which case we discard everything *it* dominates) otherwise we try again. The process is repeated for a pre-chosen number of steps (say 10^5). If we add (negative) model complexity ("simplicity") as an additional criterion (here we have several choices, including a Lasso style ℓ_1-penalty or some variant of the smoothness prior), we can further reap similar benefits as we did with the sRVM without having to commit to a certain level of smoothness a-priori.

We created a initial implementation of a scheme along those lines learning a set of weights $\mathbf{w}$ for kernel classifier using a type of Lasso penalty. Despite its simplicity, it gave quite promising results on standard data sets such as ionosphere and we aim to further develop these ideas, because there is considerable potential to side-stepping issues more traditional, single objective schemes such as the RVM or LASSO, have with local minima or analytically awkward penalty functions such as ℓ_0.

[2]Receiver Operator Characteristic curve. For our purposes a graph of the True Positive Rate (TPR) against the False Positive Rate (FPR) with each point on the graph a different binary classifier model under evaluation. The TPR is the fraction of positive (class A) members that are correctly classified as such. The FPR is the fraction of negative (class B) members that are misclassified. Taking a disease detection example, the TPR would tell us about the likelihood that the model would detect our illness if we were in fact ill, whereas the FPR would tell us about the likelihood the model would incorrectly deem us ill, despite being healthy. Typically either of these possible errors (non-detection or false alarm) will be considered worse than the other, but that does of course not mean that we would always want to trade-off a reduced likelihood of the more critical error occurring for an increased likelihood of the less critical error occurring – otherwise we would just pick models that always gave the same answer (and TPR=1 or FPR=0). This is were the ROC curve comes handy: given a set of models, their ROC curve allows us to visualize all available trade-offs between increasing the chance of detection and increasing the risk of a false alarm.

[3]Models $\mathcal{M}_A$ and $\mathcal{M}_B$ are mutually non-dominant if neither one fares better on *all* the objectives.

Implementation details of our evolutionary ROC/Simplicity Pareto curve approximator

Mutating a model is simple: at each step we chose one of two perturbations: either a) we make a Laplacian perturbation to each individual weight (i.e. we add a value sampled from a Laplacian with zero mean and a fixed standard deviation) with 10% probability or b) we pick a random weight and "flip it" (turn it to zero if it is nonzero or pick a nonzero value if it is zero; the exact value is again taken from a zero mean Laplacian). After that we score the mutant based on its TPR, negative FPR and negative complexity; we will now deal with measuring the latter.

A suitable complexity penalty for evolutionary optimization Since the ℓ_1 norm is not sparsity enforcing per se (i.e. in the absence of a mode-finding scheme), the question arises which other norm to chose. An obvious and established choice is ℓ_0, which is the most sparsity enforcing of all ℓ_q-norms[4] – it just counts the number of non-zero components. A principal reason it has not found wider application is its analytical awkwardness, but this makes it doubly attractive for an optimization algorithm that is blind to such details.

However, upon reflection the ℓ_0 norm has a certain shortcoming: to it, but not necessarily to us, all non-zero components are created equal; all things being equal, given different $\mathbf{w}$ it seems desirable to always judge the $\mathbf{w}$ with the least weight-mass (for some definition of weight-mass) the sparsest.

L_{0^+}**: a more orderly variant of the** ℓ_0 **penalty** Fortunately this flaw is easily corrected without reducing sparsity-enforcement: since the ℓ_0 norm is always an integer we can simply add a suitable fractional component as a tie-breaker; any monotonously increasing function $\mathbb{R} \to [0,1]$ of a non-degenerative norm such as the Euclidean or Manhattan norm will do: we opted for the latter (ℓ_1) as our tie-breaking norm and use the standard sigmoid as a squashing function[5]. However, although it does not make difference analytically as far as the imposed partial ordering over $\{\mathbf{w}\}_{\mathbf{w}\in\mathbf{W}}$ is concerned, after some experimenting we decided to add a logarithm before taking the sigmoid for numerical reasons, giving[6]:

$$\begin{aligned} L_{0^+}(\mathbf{w}) &\equiv \|\mathbf{w}\|_0 + \sigma(\log(\|\mathbf{w}\|_1)) && (3.1)\\ &= \sum_i I[w_i \neq 0] + \frac{1}{1+e^{-\log \sum_i |w_i|}} && (3.2)\\ &= \sum_i I[w_i \neq 0] + \frac{1}{1+(\sum_i |w_i|)^{-1}} && (3.3) \end{aligned}$$

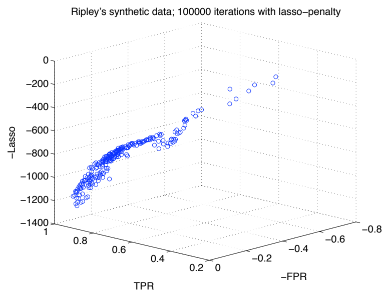
Evolutionary Optimization Results

Some preliminary tests gave rather promising results; pictured below the outcomes of a run for 10^5 optimization steps on Ripley's synthetic data (Ripley, 1996), both with Lasso and our L_{0^+} penalty (left, arbitrarily the model with the lowest overall classification error and right the approximate Pareto front from which it was taken).

[4]in fact it so sparsity enforcing it no longer is a norm in the strict sense, since scalar multiplication no longer *positive homogeneous*, viz: $\alpha\ell_0(\mathbf{w}) \neq \ell_0(\alpha\mathbf{w})$.

[5]One criticism that can be leveled against this "improvement" is that it introduces a sensitiveness to the scaling of the basis functions. The simplest answer is that this can be addressed by normalization and that furthermore the effect is small since after all we are still dealing with what is predominantly an ℓ_0 norm. A more subtle and principled answer would be to restrict the tie-breaking to corresponding weight indices w_i, but this seems overly complicated so we will stick with the former approach.

[6]This choice reduces problems due to finite precision of floating point numbers because it brings the magnitude of the tie-breaking part closer to that of the ℓ_0 norm, minimizing precision loss on addition.



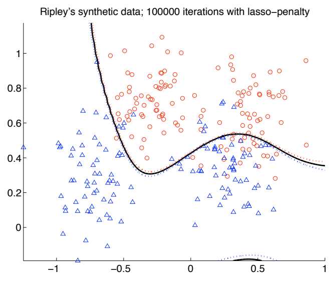
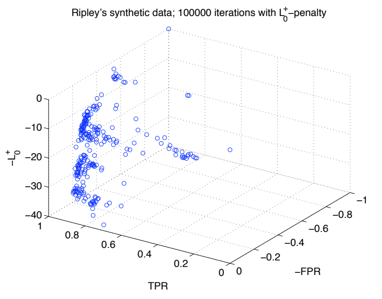


Figure 3.3: Evolutionary multi-objective optimization on Ripley's synthetic data, using the Lasso penalty.



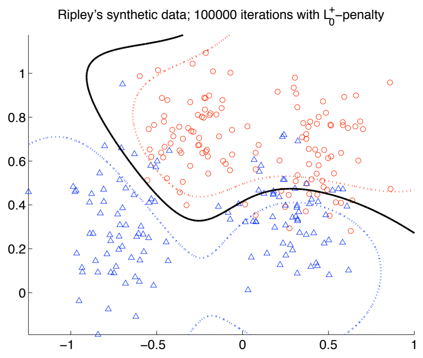


Figure 3.4: Evolutionary multi-objective optimization on Ripley's synthetic data, using the L_{0^+} penalty.

Chapter 4

Overall Conclusion

The central motive of this thesis was adequate smoothness enforcement in Relevance Vector Machines as part of the probabilistic model (as opposed to kernel/dictionary choice) and we examined its effects on both regression and classification.

For regression the situation was rather clear-cut: We provided strong empirical evidence to show that sparsity-enforcement in a plain RVM context is heavily dependent[1] on dictionary choice and that dictionary choice is, by itself, often an insufficient mechanism to obtain good results – both because it renders certain types of dictionaries with very attractive properties (wavelets in particular) practically useless for RVM regression and also because it proves to be too crude a tool to handle differential smoothing for more complex, especially multi-scale, data sets (the Doppler data set being a prime example).

By contrast, our straightforward smoothness prior extension to the RVM dealt very well with all these different scenarios in our experiments, without noticeably leading to oversmoothing in setups such as Sinc/linear spline regression where the traditional RVM shines. Furthermore the link between wavelet shrinkage and RVM regression we established opens up exciting new avenues for RVM regression and denoising, inter alia because, where applicable, wavelets (and other types of orthogonal dictionaries that can be implemented as efficient transforms, such as e.g. the discrete cosine transform) allow tackling problems of far greater scale. This is because the properties of these dictionaries allow for substantial improvements in computational complexity and numerical robustness (the largest denoising problems we successfully tackled with wavelet dictionaries were more than two orders of magnitude larger than the largest problems we could solve with more conventional kernels).

In addition to these empirical results, we gave a number of theoretical derivations to establish that our proposed prior structure meets several "sanity checks". Thus we demonstrated that the smoothness prior, unlike some other proposed prior types, yields scale invariant solutions for constant SNR (Appendix A.3), that it preserves the uniqueness of the maximum in $\hat{\ell}(\alpha_i)$ and, moreover, that this new maximum always ensures greater sparsity than the plain RVM solution (Appendix A.4). We also gave a proof that an alternative sparseness enforcing prior that has often been mentioned in the context of RVM regression and classification (e.g. in Tipping, 2001) is in fact incompatible with the fRVM scheme (Appendix A.6).

Whilst our attempts to advantageously employ a smoothness prior for regression have thus been rather successful, the picture for MCMC-based sRVM regression schemes and sRVM classification looks more ambiguous. One of our chief motivations for attempting an MCMC-based implementation still stands: although our empirical analysis of multiple local minima in sRVM regression/denoising was simple, it clearly demonstrated that the greedy fRVM scheme has the

[1]Although this fact may in retrospect seem unsurprising, it certainly has not seen much discussion in the literature, in spite of its significant practical implications.

undesirable tendency to get stuck in local minima. We saw that these problems can be to some extent masked by some popular kernel choices like Gaussian- or linear-spline kernels for which different local minima in $\boldsymbol{\alpha}/\mathbf{w}$ space can give very similar results in $\mathbf{y}$ space and our positive experience with wavelets furthermore suggests that local minima are not much an issue with orthogonal dictionaries. Although this latter fact in a way adds to the attractiveness of the smoothness prior extension (since wavelets and similar orthonormal dictionaries only become a viable choice for RVM regression thanks to the additional complexity and overfitting control the smoothness prior provides), the issue of local minima remains troubling nonetheless, both in the context of RVM in general and sRVM regression in particular – for another possibility afforded by the smoothness prior extension is the use of overcomplete dictionaries and here local minima both do occur and give rise to visibly different posterior estimates $\hat{\mathbf{y}}$ (as our toy BlocksSinc example has shown). However on the basis of our experiences we no longer feel that MCMC is a good way to address the lack of search space exploration and certain defects in probabilistic rigour the RVM suffers from – a reliably working version proved surprisingly elusive and the severe performance degradation we observed (compared to the fRVM scheme) in combination with the RVM's degenerate variance structure (which somewhat circumscribes the benefits MCMC can bring in terms of added probabilistic principledness) make it appear unlikely that an MCMC based RVM implementation could ever offer a compelling trade-off between performance and rigour. This leaves open the question of suitable alternatives for a less greedy and more principled fRVM variant or other form of sparse Bayesian regressor and classifier. An area of direct relevance to RVMs that has seen a great deal of attention of late are Gaussian Process models (GPMs) (see Rasmussen and Williams, 2006, for a recent overview that includes discussion of the RVM). However, although GPMs hold a lot of promise and are theoretically well-founded, efficient implementation for medium-to-large data sets ($N >> 1000$) is non-trivial and an area of active research – this is particularly true for the (significantly more complicated) classification case where GPMs are currently of doubtful practical value (Rasmussen and Williams, 2006).

As for classification, we have seen that although we have presented a theoretically neat derivation to effect efficient component-wise updates even for the most general case, with non-orthogonal $\mathbf{\Phi}$ and non-diagonal $\mathbf{B}$, a successful practical exploitation of the scheme ran afoul of convergence issues. In view of the limitations of the fRVM scheme with respect to local minima, we decided to bite the bullet and attempt a completely different, evolutionary optimization-based strategy.

We also think that evolutionary optimization could hold some promise for simpler and efficient, albeit possibly less Bayesian, approaches to non-greedy kernel learning – but since the particular scheme we considered (generating Pareto-optimal Receiver Operator Characteristic (ROC) curves via evolutionary optimization) is only directly applicable to classification problems and since we have not yet explored an adaptation for regression any musings concerning its potential for regression problems remain purely speculative and whereas the initial and very preliminary results did look very promising further effort in this direction is required.

In summary, although the RVM manages to unite a surprising number of seemingly conflicting requirements and uses, there is a price to pay – there are issues with local maxima and pathological posterior confidence (Rasmussen and Quiñonero-Candela, 2005). But despite such shortcomings, in our opinion the RVM provides a very powerful and versatile if somewhat eclectic mix of efficiency, sparsity and probabilistic interpretability, with little up-front modeling commitment on part of the user and for all the major tasks in machine learning: classification, regression, and in particular with our smoothness prior extension, also denoising – including Bayesian wavelet shrinkage.

Bibliography

Ahmed, N., Natarajan, T., and Rao, K. R. (1974). Discrete Cosine Transform. *IEEE Transactions on Computers, 90-93, Jan 1974.*, C-23:90–93.

Akaike, H. (1974). A new look at statistical model identification. *IEEE Transactions on Automatic Control*, 25:461–464.

Arfken, G. (1985). *Mathematical methods for physicists*. Academic Press.

Bernardo, J. and Smith, A. (1994). *Bayesian Theory*. Wiley.

Bishop, C. (2006). *Pattern recognition and machine learning*. Springer, Singapore.

Bishop, C. and Tipping, M. (2000). Variational relevance vector machines. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, pages 46–53.

Bobin, J., Moudden, Y., Starck, J.-L., and Elad, M. (2005). Multichannel morphological component analysis. In *Proceedings of Spars05*, pages 103–106, Rennes, France.

Bottou, L., Chapelle, O., DeCoste, D., and Weston, J., editors (2007). *Large-scale kernel machines*. MIT Press, Boston.

Bottou, L. and Lin, C. (2007). Support Vector Machine solvers. In Bottou et al. (2007), pages 1–27.

Boyd, S. P. and Vandenberghe, L. (2004). *Convex Optimization*. CUP, Cambridge.

Chen, S. S., Donoho, D. L., and Saunders, M. A. (1999). Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61.

Chipman, H., Kolaczyk, E., and McCulloch, R. (1997). Adaptive Bayesian wavelet shrinkage. *Journal of the American Statistical Association*, 92:1413–1421.

Clarkson, E. and Barrett, H. (2001). High-pass filters give histograms with positive kurtosis. *Optics Letters*, 26(16):1253–1255.

Clyde, M., Parmigiani, G., and Vidakovic, B. (1998). Multiple shrinkage and subset selection in wavelets. *Biometrika*, 85:391–401.

Daubechies, I. (1992). *Ten lectures on wavelets*. SIAM.

Denison, D., Holmes, C., Mallick, B., and Smith, A. (2002). *Bayesian Methods for Nonlinear Classification and Regression*. Wiley.

Do, M. and Vetterli, M. (2003). The finite ridgelet transform for image representation. *IEEE Transactions on Image Processing*, 12(1):16–28.

Donoho, D. and Johnstone, I. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455.

Donoho, D., Johnstone, I., Kerkyacharian, G., and Picard, D. (1995). Wavelet shrinkage: asymptotia? (with discussion). *J. Royal Statistical Soc.*, B 57:45–97.

Donoho, D. L. and Elad, M. (2002). Optimally sparse representation in general (nonorthogonal) dictionaries via 1 minimization. *PNAS*, 100(5):2197–2202.

D'Souza, A., Vijayakumar, S., and Schaal, S. (2004). The Bayesian backfitting relevance vector machine. In *Proceedings of the 21st International Conference on Machine Learning (ICML 2004)*, Banff, Alberta, Canada.

Efron, B., Hasti, T., Johnstone, I., and Tibshirani, R. (2002). Least angle regression. Technical report, Stanford University.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004). Least angle regression. *Annals of Statistics*, 32(2):407–499.

Ehlers, R. and Brooks, S. (submitted). Efficient construction of reversible jump MCMC proposals for autoregressive time series models. http://www.statslab.cam.ac.uk/ steve/mypapers/ehlb02.ps.

Everson, R. and Fieldsend, J. (2006). Multi-objective optimisation of safety related systems: An application to short term conflict alert. *IEEE Transactions on Evolutionary Computation,*, 10(2):187–198.

Faul, A. and Tipping, M. (2002). Analysis of sparse Bayesian learning. In *Advances in Neural Information Processing Systems*, volume 14.

Figueiredo, M. (2003). Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:1150–1159.

Fokoué, E., Goel, P., and Sun, D. (2004). A Prior for Consistent Estimation for The Relevance Vector Machine. Technical report, Statistical and Applied Mathematical Sciences Institute, Research Triangle Park, NC, USA.

Foster, D. P. and Goerg, E. I. (1994). The Risk Inflation Criterion for multiple regression. *Annals of Statistics*, 22(4):1947–1975.

Frigo, M. and Johnson, S. G. (2005). The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231.

Furnival, G. M. and Robert W. Wilson, J. (1974). Regressions by leaps and bounds. *Technometrics*, 16(4):499–511.

Girolami, M. and Rogers, S. (2005). Hierachic Bayesian models for kernel learning. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 241–248, Bonn.

Golub, G. H. and van Loan, C. E. (1996). *Matrix computations*. John Hopkins Press, 3rd edition.

Green, P. J. (1995). Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82:711–713.

Hastie, T. and Tibshirani, R. (1990). *Generalized additive models*. Chapman and Hall, London.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The elements of statistical learning: data mining, inference, and prediction*. Springer, New York.

Hoerl, A. and Kennard, R. (1970). Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67.

Holmes, C. and Denison, G. (1999). Bayesian wavelet analysis with a model complexity prior. In *Bayesian statistics 6: Proceedings of the sixth Valencia international meeting*, pages 769–776, Oxford.

Jansen, M. (2001). *Noise reduction by wavelet thresholding*. Springer, New York.

Jaynes, E. (2003). *Probability Theory: the logic of science*. CUP, Cambridge.

Kovačević, J. and Vetterli, M. (1991). Nonseperable multidimensional perfect filter banks and wavelet bases for $\mathbb{R}^N$. *IEEE Transactions on Information Theory*, 38(2):533–555.

Krishnapuram, B., Hartemink, A., Carin, L., and Figueiredo, M. (2004). A Bayesian approach to joint feature selection and classifier design. *IEEE Transactions on Pattern Analysis and Machine Intelligence - PAMI*, 26(9):1105–1111.

Kroptov, D. and Vetrov, D. (2007). On one method of non-diagonal regularization in sparse Bayesian learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML 2007)*, Cornvallis, OR.

Lam, E. and Goodman, J. (2000). A mathematical analysis of the DCT coefficient distributions for images. *IEEE Trans. Image Processing*, 9:1661–1666.

MacKay, D. (1992). The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736.

MacKay, D. (1995). Probable networks and plausible predictions. *Network: Computation in Neural Systems*, 6:469–505.

MacKay, D. C. (2003). *Information Theory, Inference, and Learning Algorithms*. CUP, Cambridge.

MacKay, D. J. C. (1992). A practical Bayesian framework for backprop networks. In Moody, J. E., Hanson, S. J., and Lippmann, R. P., editors, *Advances in Neural Information Processing Systems 4*, pages 839–846.

Mallat, S. (1999). *A wavelet tour of signal processing*. Academic Press, 2nd edition.

Mallat, S. and Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415.

Moler, C. (2004). *Matlab® The Language of Technical Computing, version 7.01*. The MathWorks Inc., Natick, MA, USA.

Nabney, I. T. (2002). *NETLAB: Algorithms for pattern recognition*. Springer, New York.

Nason, G. P. (1996). Wavelet shrinkage by cross-validation. *Journal of the Royal Statistical Society B*, 58:463–479.

Neal, R. (1993). Probabilistic inference using Markov Chain Monte Carlo Methods. Technical report, Dept. of Computer Science, University of Toronto.

Neumaier, A. (1998). Solving ill-conditioned and singular linear systems: A tutorial on regularization. *SIAM Review*, 40:636–666.

Ng, A. Y. (2004). Feature selection, l1 vs. l2 regularization, and rotational invariance. In *Proceedings of the 21. International Conference on Machine Learning*.

Ogden, R. T. (1997). *Essential wavelets for statistical applications and data analysis*. Birkhäuser, Boston.

Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. (1992). *Numerical Recipes in C*. Cambridge University Press, Cambridge, 2nd edition.

Quiñonero-Candela, J. (2004). *Learning with Uncertainty – Gaussian Processes and Relevance Vector Machines*. PhD thesis, Technical University of Denmark, Lyngby, Denmark.

Quinonero-Candela, J., Rasmussen, C. E., and Williams, C. K. (2007). Approximation methods for Gaussian Process regression. In Bottou et al. (2007), pages 203–224.

Quiñonero-Candela, J., Snelson, E., and Williams, O. (2007). Sensible priors for sparse bayesian learning. Technical report, Microsoft resrch.

Rasmussen, C. E. and Quiñonero-Candela, J. (2005). Healing the relevance vector machine by augmentation. In Raedt, L. D. and Wrobel, S., editors, *Proceedings of the 22nd International Conference on Machine Learning*, pages 689–696.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press, Boston, Mass.

Rezek, I. (2008). $N = 4097$ single channel EEG dataset with human expert marked spindles. private communication.

Ripley, B. (1996). *Pattern recognition and neural networks*. CUP, Cambridge.

Schmolck, A. and Everson, R. (2005). Smoothness priors for sparse Bayesian regression. In *Workshop on Signal Processing with Adaptative Sparse Structured Representations*, Rennes, France.

Schmolck, A. and Everson, R. M. (2007). Smooth relevance vector machine: a smoothness prior extension of the RVM. *Machine Learning*, 68(2):107–135.

Schölkopf, B. and Smola, A. (2002). *Learning with kernels*. MIT Press, Cambridge, Mass.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6:461–464.

Smith, K. I. (2006). *A Study of Simulated Annealing Techniques for Multi-Ob jective Optimisation*. PhD thesis, University of Exeter.

Smola, A. and Schmölkopf, B. (2000). Sparse greedy matrix approximation for machine learning. In Langley, P., editor, *Proceedings of the 17th International Conference on Machine Learning*, pages 911–918, San Fransisco. Morgan Kaufman.

Starck, J., Candes, E., and Donoho, D. (2002). The curvelet transform for image denoising. *IEEE Transactions on Image Processing*, 11(6):670–684.

Stein, C. M. (1981). "estimation of the mean of a multivariate normal distribution". *The Annals of Statistics*, 9(6):1135–1151.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (B)*, 58:267–288.

Tikhonov, A. (1943). On the stability of inverse problems. *Dokl. Akad. Nauk SSSR*, 39(5):195–198.

Tipping, M. (2000). The relevance vector machine. In Solla, A., Leen, T., and Müller, K.-R., editors, *Advances in Neural Information Processing Systems*, volume 12, pages 652–658, Cambridge, Mass. MIT Press.

Tipping, M. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244.

Tipping, M. and Faul, A. (2003). Fast marginal likelihood maximisation for sparse Bayesian models. In *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*.

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley-Interscience, Hoboken, NJ.

Vidakovic, B. (1998). Nonlinear wavelet shrinkage with Bayes rules and Bayes factors. *Journal of the American Statistical Association*, 93:173–179.

Vidakovic, B. (1999). *Statistical Modelling by wavelets*. Wiley, New York.

Wahaba, G. (1984). Cross-validated spline methods for estimation of multivariate functions from data on functionals. In *Statistics: An Apraisal*, pages 205–25. Iowa State Press, Ames, Iowa.

Williams, P. M. (1994). Bayesian regularisation and pruning using a Laplace prior. Technical Report 312, School of Cognitive and Computing Sciences, University of Sussex.

Wipf, D. and Rao, B. (2004). Perspectives on sparse Bayesian learning. In Thrun, S., Saul, L., and Schölkopf, B., editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA. MIT Press.

Wipf, D. and Rao, B. (2005). Finding sparse representations in multiple response models via Bayesian learning. In *Proceedings of Spars05*, pages 155–158, Rennes.

Appendix A

Appendix

A.1 Kernel functions

For completeness, here we list the kernels generating the basis functions used in this paper.

$$\begin{aligned} K_{\text{lspline}}(x_m, x_n) &= 1 + r^{-2}x_m x_n + r^{-3}x_m x_n \min(x_m, x_n) \\ &\quad - \frac{x_m + x_n}{2} r^{-3} \min(x_m, x_n)^2 + r^{-3}\frac{\min(x_m, x_n)^3}{3} \qquad \text{(A.1)} \\ K_{\text{gauss}}(x_m, x_n) &= \exp(-(x_m - x_n)^2/r^2) \qquad \text{(A.2)} \\ K_{\text{tpspline}}(x_m, x_n) &= |x_m - x_n|^2 r^{-2} \log(|x_m - x_n + \delta_{mn}|/r) \qquad \text{(A.3)} \end{aligned}$$

Here r sets the width of the kernel. The quoted values of r are relative to data defined so that $-10 \leqslant x \leqslant 10$. We use the shorthand "gauss 3.0" etc. in the text, to denote the gaussian kernel defined below with a width parameter $r = 3.0$.

We use the variant of the discrete cosine transform (Ahmed et al., 1974) used by Matlab®'s `dct` function (Moler, 2004):

$$\phi_m(\mathbf{x}) = r_m \sum_{d=1}^{D} x_d \cos\frac{\pi(2d-1)(m-1)}{2D}, \qquad m = 1, ..., M, \quad M = D \qquad \text{(A.4)}$$

$$r_m = \begin{cases} \sqrt{\frac{1}{M}} & m = 1 \\ \sqrt{\frac{2}{M}} & 2 \leqslant m \leqslant M \end{cases} \qquad \text{(A.5)}$$

The symmlet family of wavelets is due to Daubechies, all our examples use the symmlet8 wavelet (the 8 here is not a width parameter) from that family as defined in (Daubechies, 1992). For a general discussion of wavelet bases such as Haar and symmlets see e.g. (Mallat, 1999).

A.2 Data sets used

With the exception of the spindle EEG data (Rezek, 2008), we draw mostly from the standard regression data sets that have been used throughout the RVM (Sinc, see e.g. Tipping (2000, 2001); Quiñonero-Candela (2004)) and the Wavelet literature (Blocks, Bumps and Doppler, see in particular Donoho and Johnstone (1994)). The latter three functions are available in executable form as part of Wavelab (`http://www-stat.stanford.edu/~wavelab/`), but for convenience we present all of them here in Matlab® code:

```
function yy = makeSignal(signal, N)
%   Copyright 2006 Alexander Schmolck
ts = [0.1, 0.13, 0.15, 0.23, 0.25, 0.40, 0.44, 0.65, 0.76, 0.78, 0.81];
xx = linspace(0,1,N);
outer = @(f,x,y) f(repmat(x(:)',length(y),1), repmat(y(:), 1, length(x)));
switch lower(signal)
 case 'sinc',
  yy = sinc(linspace(-10,10,N)./pi)';
 case 'heavisine'
  yy = (4.*sin(4.*pi.*xx) - sign(xx-0.3) - sign(0.72-xx))';
 case 'doppler'
  epsilon=0.05;
  yy = ((xx.*(1-xx)).^0.5.*sin(2.*pi.*(1+epsilon)./(xx+epsilon)))';
 case 'blocks'
  K = @(xx) 0.5.*(1+sign(xx));
  hh = [4, -5, 3, -4, 5, -4.2, 2.1, 4.3,-3.1, 2.1,-4.2];
  yy = sum(repmat(hh, N, 1).*K(outer(@minus,xx,ts)),2);
 case 'bumps' %XXX this maybe diverges a bit too much numerically
  ww = [.005,.005,.006,.01,.01,.03,.01,.01,.005,.008,.005];
  hh = [4, 5, 3, 4, 5, 4.2, 2.1, 4.3, 3.1, 5.1, 4.2];
  K = @(xx) 1.0./(1.0 + abs(xx)).^4;
  yy = sum(repmat(abs(hh),N,1).*K(outer(@minus,xx,ts)./repmat(ww, N,1)),2);
 otherwise
  error('Unknown function %s', signal)
end
```

A.3 Scale invariance for constant SNR

The sRVM scheme is invariant to scaling of the signal amplitude provided that the noise variance is also scaled so that the SNR remains constant. Suppose that the targets are scaled so that $\mathbf{t} \to k\mathbf{t}$ and the noise is scaled so that $\sigma^2 \to k^2\sigma^2$, then if the coefficient precisions α_i are scaled as $\alpha_i \to k^{-2}\alpha_i$ then the log posterior $\mathcal{L}(\boldsymbol{\alpha}) + \log p(\alpha \,|\, \sigma^2)$ changes only by an additive constant. To see this, note that the matrix $\mathbf{C}$ (2.69) becomes

$$\tilde{\mathbf{C}} = k^2\sigma^2\mathbf{I} + \sum_m \alpha_m^{-1} k^2 \boldsymbol{\Phi}_m \boldsymbol{\Phi} \tag{A.6}$$

$$= k^2\mathbf{C} \tag{A.7}$$

Consequently, from (2.64), $\mathcal{L}(\boldsymbol{\alpha})$ becomes:

$$\tilde{\mathcal{L}}(\boldsymbol{\alpha}) = -\frac{1}{2}\left[N\log 2\pi + 2M\log k + \log|\mathbf{C}| + (k\mathbf{t}^T)(k^{-2}\mathbf{C}^{-1})(k\mathbf{t})\right] \tag{A.8}$$

$$= \mathcal{L}(\boldsymbol{\alpha}) + 2M\log k \tag{A.9}$$

When the α_i are rescaled along with σ^2, it is clear that DF in (2.58) remains unchanged. Thus the prior is invariant to simultaneous rescaling of $\boldsymbol{\alpha}$ and σ^2 and so the log posterior is

$$\log p(\boldsymbol{\alpha}/k^2, k^2\sigma^2 \,|\, k\mathbf{t}) = \log p(\boldsymbol{\alpha}, \sigma^2 \,|\, \mathbf{t}) + 2M\log k \tag{A.10}$$

Maximum a posteriori solutions for $\boldsymbol{\alpha}$ in the scaled case are thus just the MAP solutions in the unscaled case divided by k^2.

A.4 Uniqueness of local maximum

Here we show that $\hat{\ell}(\alpha_i)$ (2.80) can have at most a single maximum $\hat{\alpha}_i < \infty$ and $\alpha_i^\star < \hat{\alpha}_i$. We drop the explicit indication of which basis function is being dealt with and it is convenient to work in terms of the noise precision $\beta = \sigma^{-2}$.

The derivative of $\hat{\ell}(\alpha)$ is given by (2.81) in which the cubic polynomial $P(\alpha) = B_3\alpha^3 + B_2\alpha^2 + B_1\alpha + B_0$ has coefficients

$$B_0 = s^2\beta^2 \tag{A.11}$$

$$B_1 = s\beta^2 + 2\beta s^2 - \beta^2 q^2 + 2s^2 c\beta \tag{A.12}$$

$$B_2 = 2s\beta + s^2 - 2\beta q^2 + 4s\beta c \tag{A.13}$$

$$B_3 = s - q^2 + 2c\beta \tag{A.14}$$

We note that the numerator of (2.81) is positive for all $\alpha > 0$ so the turning points of $\hat{\ell}(\alpha)$ can be found by examining $P(\alpha)$. A crucial quantity turns out to be the sign of B_3 and we treat positive and negative cases separately. First note that the derivative of
$lhat(\alpha)$, $\hat{\ell}'(\alpha)$ is given by

$$2\hat{\ell}'(\alpha) = \frac{1}{2\alpha(\alpha+s)^2(\alpha+\beta)^2}\left\{[s - q^2 + 2c\beta]\alpha^3 + \text{H.O.T.}\right\} \tag{A.15}$$

so that the gradient at infinite α is always zero. Also $\lim_{\alpha\to\infty}\hat{\ell}(\alpha) = 0$ and so $\hat{\ell}(\alpha)$ is asymptotic to zero from below if $B_3 > 0$ or from above if $B_3 < 0$. As $\alpha \to 0$ then $\hat{\ell}(\alpha) \to -\infty$.

A.4.1 Asymptote from below:

Since $B_3 > 0$ the graph of $P(\alpha) \to \pm\infty$ as $\alpha \to \pm\infty$, and $P(0) = B_0 > 0$. Consequently, $P(\alpha)$ has a least one root for $\alpha < 0$. It must therefore either have zero or two positive roots.

If P has no positive roots the maximum of $\hat{\ell}(\alpha)$ is achieved at infinity (e.g. top-left Figure 2.11).

If there are two positive roots, one corresponds to a maximum and the other to a minimum. Ignoring the degenerate case of an inflexion point, the root for smaller α must be the maximum and the root for larger α

is a minimum. The maximum may be greater or less than the asymptotic value, as illustrated by the bottom row of Figure 2.11.

A.4.2 Asymptote from above:

In this case since $B_3 < 0$ there is at least a single positive root of $P(\alpha)$. Since $\hat{\ell}(\alpha)$ is asymptotic to zero from above at infinity this root must correspond to a maximum. However, we must ensure that there cannot be 3 positive roots.

The derivative of $\hat{\ell}(\alpha)$ can be written as the sum of the derivatives of $\ell(\alpha)$ and $\log p(\alpha_i \mid \sigma^2)$ as follows:

$$\hat{\ell}' = \frac{[s^2 + (s - q^2)\alpha](\beta + \alpha)^2 + 2c\beta\alpha(s + \alpha)^2}{2\alpha(\alpha + s)^2(\alpha + \beta)^2} \tag{A.16}$$

$$\equiv \frac{P_0(\alpha) + 2c\beta\alpha(s + \alpha)^2}{2\alpha(\alpha + s)^2(\alpha + \beta)^2} \tag{A.17}$$

where $P_0(\alpha)$ is the cubic $P(\alpha)$ with $c = 0$. It has a root at $\alpha^\star$, which corresponds to the maximum in the likelihood, and there is an additional repeated root at $\alpha = -\beta$.

The term $2c\beta\alpha(s + \alpha)^2$ arising from the prior is a cubic with a root at $\alpha = 0$ and a repeated root at $\alpha = -s < 0$. It is clearly positive for all $\alpha > 0$. There is therefore a root of $P(\alpha)$ at some $\hat{\alpha} > \alpha^\star$ because $P(\alpha) \geqslant P_0(\alpha)$ for all $\alpha > 0$. Since $P_0(\alpha)$ is monotonically decreasing for $\alpha > \alpha^\star$, while $c\beta\alpha(s + \alpha)^2$ is monotonically increasing they can only intersect once (at $\hat{\alpha}$) so there can be no roots for $\alpha > \hat{\alpha}$ and we conclude that there can only be a single maximum of $\hat{\ell}(\alpha)$ for $\alpha > 0$ in this case, which is illustrated in the bottom right panel of figure 2.11.

A.5 Saddle-points of $\hat{\mathcal{L}}$

In order to determine the nature of the maxima of $\hat{\mathcal{L}}$ the Hessian of $\mathcal{L}$ is required. As shown in (Faul and Tipping, 2002), the off-diagonal terms of the Hessian are:

$$\frac{\partial^2 \mathcal{L}(\hat{\boldsymbol{\alpha}})}{\partial\alpha_i \partial\alpha_j} = \frac{\boldsymbol{\phi}_i^T \mathbf{C}^{-1} \boldsymbol{\phi}_j}{2\alpha_i^2 \alpha_j^2} \left[\boldsymbol{\phi}_i^T \mathbf{C}^{-1} \boldsymbol{\phi}_j - 2(\boldsymbol{\phi}_i^T \mathbf{C}^{-1}\mathbf{t})(\boldsymbol{\phi}_j^T \mathbf{C}^{-1}\mathbf{t})\right] \quad i \neq j \tag{A.18}$$

When the basis functions are orthogonal the matrices $\mathbf{C}$ and therefore $\mathbf{C}^{-1}$ are diagonal (2.69). Consequently $\boldsymbol{\phi}_i^T \mathbf{C}^{-1} \boldsymbol{\phi}_j$ and hence the off-diagonal terms of the Hessian are zero. At a solution located through the maximization procedure the diagonal elements of the Hessian corresponding to finite α_i maxima are necessarily negative. The Hessian is therefore positive semi-definite, with the zeros corresponding to the infinite α_m, switched-off components. Unfortunately, the demonstration (Faul and Tipping, 2002) that the Hessian of the log marginal likelihood is negative semi-definite with general basis functions appears to be flawed and we are unable to provide any assurance that joint optimization of two or more α_i might not yield a better result than successive maximization with respect to each.

A.6 Approaching the Jeffreys' prior

With $p(\alpha_i) = Z\alpha_i^\zeta$ the contribution of the ith component to the posterior becomes:

$$\hat{\ell}(\alpha_i) = \frac{1}{2}\left[\log\alpha_i - \log(\alpha_i + s) + \frac{q^2}{\alpha_i + s}\right] + \log Z + \zeta\log\alpha_i \tag{A.19}$$

$$= \frac{1}{2}\left[(1 + 2\zeta)\log\alpha_i - \log(\alpha_i + s) + \frac{q^2}{\alpha_i + s}\right] + \log Z \tag{A.20}$$

Setting the derivative to 0 to find the MAP solution $\hat{\alpha}_i$ we get:

$$\hat{\ell}'(\alpha_i) = \frac{1}{2}\left[\frac{1 + 2\zeta}{\alpha_i} - \frac{1}{\alpha_i + s} - \frac{q^2}{(\alpha_i + s)^2}\right] = 0 \tag{A.21}$$

From this it becomes apparent that $\hat{\ell}$ has no turning points for finite, positive α_i if $\zeta < -\frac{1}{2}$ because α_i, q^2 and s are always positive. Furthermore, in this case since $\hat{\ell}'(\alpha_i) < 0$ the maximum of $\hat{\ell}(\alpha_i)$ is achieved at $\alpha_i = 0$.

As a Jeffreys' prior corresponds to $\zeta = -1$, one would always obtain a model in which all the components are active.

Since it is well-known that the MAP solution is not invariant over parametrization, enquiring minds might wonder if this result is not in fact specific to differentiation in real space. Indeed it it turns out to be, but the setting the derivative in logspace, $\frac{d}{d \log \alpha_i} \hat{\ell} = 0$, gives the following solution for $\zeta = -1$:

$$\alpha_i = \frac{\sqrt{s^2 + 6q^2 s + q^4} - 3s - q^2}{4} \tag{A.22}$$

The Schwarz inequality suggests that this α_i will be negative (and hence invalid) in most cases and indeed in several trials we found α_i to be negative without exception.

With a Gamma prior for α_m (2.129) the derivative of the contribution to the log posterior from the ith basis function is:

$$\hat{\ell}'(\alpha_i) = \frac{1}{2}\left[\frac{1}{\alpha_i} - \frac{1}{\alpha_i + s} + \frac{q^2}{\alpha_i + s}\right] + \frac{a-1}{\alpha_i} - b \tag{A.23}$$

$$= \frac{1}{2}\left[\frac{2a-1}{\alpha_i} - \frac{1}{\alpha_i + s} - \frac{q^2}{(\alpha_i + s)^2} - 2b\right] \tag{A.24}$$

In this case is it clear that $\hat{\ell}$ has no turning points for finite, positive α_i if $a < 1/2$, although there will be no positive α_i for larger a when b is larger. Consequently the Gamma prior forces all components to be active when $a < 1/2$, which of course includes the Jeffreys' prior in the limit $a, b \to 0$.

A.7 Efficiently calculating the full likelihood and posterior

As $\mathbf{C}$ is a $N \times N$ matrix, its inversion and computation of the determinant are very costly procedures – $O(N^3)$. Fortunately with the help of the Woodbury matrix inversion identity and matrix determinant lemma (see (B.3), (B.4)) and using (2.53) and (2.79) the marginal log likelihood may instead be advantageously expressed as (for regression):

$$\mathcal{L} = -\frac{1}{2}\left[N \log(2\pi) + N \log \sigma^2 - \sum_{s} \log \alpha_s + \log|\mathbf{\Sigma}| + \left(\sigma^{-2}\mathbf{t}^T\mathbf{t} - \mu^T\mathbf{\Sigma}^{-1}\mu\right)\right] \tag{A.25}$$

or by

$$\mathcal{L} = -\frac{1}{2}\left[2\sum_{n}^{N}(t_n \log \hat{y}_n + (1-t_n)\log(1-\hat{y}_n)) + \hat{\mathbf{w}}_{LP}^T \mathbf{A} \hat{\mathbf{w}}_{LP} - \log|\mathbf{A}| - \log|\mathbf{\Sigma}|\right] \tag{A.26}$$

(for classification) ()

The efficient expression for $\hat{\mathcal{L}}$, the log posterior is then given by (for regression and approximately orthogonal $\mathbf{\Phi}$)

$$\hat{\mathcal{L}} = \mathcal{L} + \sum_{s} \frac{-c}{1 + \sigma^2 \alpha_s} + S \log Z + \log \mathcal{IG}(\sigma^2 \mid g, h) \tag{A.27}$$

or by (for classification)

$$\hat{\mathcal{L}} = \mathcal{L} + (-c \operatorname{tr} \mathbf{S}) \tag{A.28}$$

and may be computed in $O(S^3)$ time, where S is the number of *included* components (as the matrix $\mathbf{\Sigma}$ is $S \times S$). Although the stated algorithm does not depend on it this expression is useful for obtaining the posterior likelihood of the solution the sRVM algorithm finds and may also be used for convergence testing.

$$\begin{aligned}
\mathbf{\Sigma} =& (\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi} + \mathbf{A})^{-1} \qquad \mu = \mathbf{\Sigma}\mathbf{\Phi}^T\mathbf{B}\tilde{\mathbf{t}} \qquad \text{(for comparison)} && \text{(A.29)}\\
\tilde{\mathbf{t}} =& \mathbf{\Phi}\mu + \mathbf{B}^{-1}(\mathbf{t} - \mathbf{y}) && \text{(A.30)}\\
\mathbf{C} =& \mathbf{B} + \mathbf{\Phi}\mathbf{A}\mathbf{\Phi}^T && \text{(A.31)}\\
\mathbf{C}^{-1} =& \mathbf{B}^{-1} - \mathbf{B}^{-1}\mathbf{\Phi}(\mathbf{A}^{-1} + \mathbf{\Phi}^T\mathbf{B}^{-1}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{B}^{-1} && \text{(A.32)}\\
|\mathbf{C}| =& |\mathbf{A}|\,|\mathbf{B}|\,|\mathbf{A}^{-1} + \mathbf{\Phi}^T\mathbf{B}^{-1}\mathbf{\Phi}| && \text{(A.33)}\\
\mathcal{L} =& -\frac{1}{2}\left(N\log(2\pi) + \log|\mathbf{C}| + \tilde{\mathbf{t}}\mathbf{C}^{-1}\tilde{\mathbf{t}}\right) && \text{(A.34)}\\
=& -\frac{1}{2}(N\log(2\pi) + \sum_S \log \mathbf{B}_{[s,s]} + \sum_S \log\alpha_s + \log|\mathbf{A}^{-1} + \mathbf{\Phi}^T\mathbf{B}^{-1}\mathbf{\Phi}| && \text{(A.35)}\\
& + \tilde{\mathbf{t}}^T\left(\mathbf{B}^{-1} - \mathbf{B}^{-1}\mathbf{\Phi}(\mathbf{A}^{-1} + \mathbf{\Phi}^T\mathbf{B}^{-1}\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{B}^{-1}\right)\tilde{\mathbf{t}}) && \text{(A.36)}
\end{aligned}$$

A.8 Computing $\frac{d}{d\alpha_i}(\operatorname{tr}\mathbf{S})$ for $\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi} \not\simeq \sigma\mathbf{I}$

In this section we explore how to obtain $\frac{d}{d\alpha_i}(\operatorname{tr}\mathbf{S})$,when $\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi} \not\simeq \sigma\mathbf{I}$ so that the approximation (2.58) is not longer appropriate, so that we no longer can use (2.81) to maximize $\hat{\ell}$ w.r.t. α_i. Crucially this is always the case for classification problems.

We are interested in the contribution of α_i to the trace of $\mathbf{S} = \mathbf{\Phi}\mathbf{\Sigma}\mathbf{\Phi}^T\mathbf{B}$, in the general case. For notational convenience we introduce:

$$\mathbf{\Lambda}^{-1} \quad \equiv \quad \mathbf{\Sigma} = (\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi} + \mathbf{A})^{-1} \tag{A.37}$$

The change $\alpha_i \leftarrow \alpha_i + \Delta$ and thus $\mathbf{\Lambda}_{[i,i]} \leftarrow \mathbf{\Lambda}_{[i,i]} + \Delta$ may be written as $\mathbf{\Lambda}_\Delta = \mathbf{\Lambda} + \Delta\mathbf{e}_i\mathbf{e}_i^T$, where $e_{ij} = \delta_{ij}$. Now by the Sherman-Morrison identity we have

$$\begin{aligned}
\mathbf{\Lambda}_\Delta^{-1} = (\mathbf{\Lambda} + \Delta\mathbf{e}_i\mathbf{e}_i^T)^{-1} = \mathbf{\Sigma}_\Delta \quad &= \quad \mathbf{\Sigma} - \frac{\mathbf{\Sigma}\Delta\mathbf{e}_i(\mathbf{\Sigma}^T\mathbf{e}_i)^T}{1 + \Delta\mathbf{e}_i^T\mathbf{\Sigma}^T\mathbf{e}_i} && \text{(A.38)}\\
&= \quad \mathbf{\Sigma} - \frac{\Delta\mathbf{\Sigma}_{[:,i]}(\mathbf{\Sigma}_{[i,:]})^T}{1 + \Delta\mathbf{\Sigma}_{[i,i]}} && \text{(A.39)}\\
&= \quad \mathbf{\Sigma} - \frac{\boldsymbol{\sigma}_i\boldsymbol{\sigma}_i^T}{\Delta^{-1} + \sigma_{ii}} && \text{(A.40)}\\
&\equiv \quad \mathbf{\Sigma} - (\Delta^{-1} + \sigma_{ii})^{-1}\mathbf{Q} && \text{(A.41)}
\end{aligned}$$

where we have introduced $\mathbf{Q} \equiv \mathbf{\Sigma}_{[:,i]}(\mathbf{\Sigma}_{[i,:]})^T = \boldsymbol{\sigma}_i\boldsymbol{\sigma}_i^T$

Thus the change in the trace of $\mathbf{S}$ resulting from $\alpha_i \leftarrow \alpha_i + \Delta$ can be stated as:

$$\begin{aligned}
\operatorname{tr}\mathbf{S}_\Delta - \operatorname{tr}\mathbf{S} \quad &= \quad \operatorname{tr}(\mathbf{S}_\Delta - \mathbf{S}) && \text{(A.42)}\\
&= \quad \operatorname{tr}(\mathbf{\Phi}\mathbf{\Sigma}_\Delta\mathbf{\Phi}^T\mathbf{B} - \mathbf{\Phi}\mathbf{\Sigma}\mathbf{\Phi}^T\mathbf{B}) && \text{(A.43)}\\
&= \quad \operatorname{tr}(\mathbf{\Sigma}_\Delta\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi} - \mathbf{\Sigma}\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}) && \text{(A.44)}\\
&= \quad \operatorname{tr}\left\{(\mathbf{\Sigma}_\Delta - \mathbf{\Sigma})\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}^T\right\} && \text{(A.45)}\\
&= \quad \operatorname{tr}\left\{-(\Delta^{-1} + \sigma_{ii})^{-1}\mathbf{Q}\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}\right\} && \text{(A.46)}\\
&= \quad -\operatorname{tr}\left\{(\Delta^{-1} + \sigma_{ii})^{-1}\mathbf{Q}\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}\right\} && \text{(A.47)}
\end{aligned}$$

Taking the limit we see that:

$$\begin{aligned}
\frac{d}{d\alpha_i}\operatorname{tr}\mathbf{S} \quad &= \quad \lim_{\Delta\to 0}\frac{\operatorname{tr}\mathbf{S}_\Delta - \operatorname{tr}\mathbf{S}}{\Delta} && \text{(A.48)}\\
&= \quad -\operatorname{tr}\left\{(1 + 0\sigma_{ii})^{-1}\mathbf{Q}\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}\right\} && \text{(A.49)}\\
&= \quad -\operatorname{tr}\left\{\mathbf{Q}\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}\right\} && \text{(A.50)}\\
&= \quad -\boldsymbol{\sigma}^T\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi}\boldsymbol{\sigma} && \text{(A.51)}
\end{aligned}$$

Since $\hat{\ell}(\alpha_i) = \ell(\alpha_i) + \left[\log p(\alpha_i \mid \sigma^2) = -c\operatorname{tr}\mathbf{S}\right]$, we see that the derivative of the posterior log likelihood function for the i-th component, $\hat{\ell}'$, is given by:

$$\hat{\ell}'(\alpha_i) = \ell'(\alpha_i) + c\,\mathrm{tr}(\mathbf{Q}\mathbf{\Phi}^\mathrm{T}\mathbf{B}\mathbf{\Phi}) \tag{A.52}$$

For an orthogonal design matrix, $\mathbf{\Phi}^\mathrm{T}\mathbf{\Phi} = \mathbf{I}$ and the regression case $\mathbf{B} = \sigma^{-2}\mathbf{I}$, this simplifies to the familiar[1] $\hat{\ell}'(\alpha_i) = \ell'(\alpha_i) + c\sigma^2(1+\alpha\sigma^2)^{-2}$, which only depends on a single α_i. Unfortunately, in the general case with diagonal $\mathbf{B}$ and non-orthogonal $\mathbf{\Phi}$ we see that the second term on the RHS of this equation no longer depends exclusively on a single α_i, since the $\mathbf{Q}$ term contains contributions from all (non-∞)α_m. Moreover even ignoring these cross-terms, finding the roots of this equation appears non-trivial.

We therefore propose the following approach: as we can always come up with a hopefully reasonably close candidate α_i (using e.g. the trivial None-prior solution), we express the desired solution $\alpha_i^\star$ as $\alpha_i + \Delta$ and thus rewrite (A.52) as follows:

$$\hat{\ell}'(\alpha_i + \Delta) = \ell'(\alpha_i + \Delta) + c\,\mathrm{tr}(\mathbf{Q}_\Delta\mathbf{\Phi}^\mathrm{T}\mathbf{B}\mathbf{\Phi}) \tag{A.53}$$

$$\tag{A.54}$$

From

$$\begin{aligned}
\mathbf{Q}_\Delta &= \boldsymbol{\sigma}_{\Delta i}\boldsymbol{\sigma}_{\Delta i}^\mathrm{T} = \mathbf{\Sigma}_{\Delta[:,i]}(\mathbf{\Sigma}_{\Delta[:,i]})^\mathrm{T} && \text{(A.55)}\\
&= (\mathbf{\Sigma}_{[:,i]} - (\Delta^{-1} + \sigma_{ii})^{-1}\boldsymbol{\sigma}_i\sigma_{ii})(\mathbf{\Sigma}_{[:,i]} - (\Delta^{-1} + \sigma_{ii})^{-1}\boldsymbol{\sigma}_i\sigma_{ii})^\mathrm{T} && \text{(A.56)}\\
&= (\boldsymbol{\sigma}_i(1 - (\Delta^{-1} + \sigma_{ii})^{-1}\sigma_{ii}))(\boldsymbol{\sigma}_i(1 - (\Delta^{-1} + \sigma_{ii})^{-1}\sigma_{ii}))^\mathrm{T} && \text{(A.57)}\\
&= (1 - (\Delta^{-1} + \sigma_{ii})^{-1}\sigma_{ii})^{-2}\boldsymbol{\sigma}_i\boldsymbol{\sigma}_i^\mathrm{T} && \text{(A.58)}\\
&= (\Delta\sigma_{ii} + 1)^{-2}\mathbf{Q} && \text{(A.59)}
\end{aligned}$$

we have

$$\begin{aligned}
\frac{d}{d(\alpha_i + \Delta)}\,\mathrm{tr}\,\mathbf{S}_\Delta &= -\,\mathrm{tr}(\mathbf{Q}_\Delta\mathbf{\Phi}^\mathrm{T}\mathbf{B}\mathbf{\Phi}) && \text{(A.60)}\\
&= (\Delta\sigma_{ii} + 1)^{-2}(-\,\mathrm{tr}(\mathbf{Q}\mathbf{\Phi}^\mathrm{T}\mathbf{B}\mathbf{\Phi})) && \text{(A.61)}\\
&= (\Delta\sigma_{ii} + 1)^{-2}\frac{d}{d\alpha_i}\,\mathrm{tr}\,\mathbf{S} && \text{(A.62)}
\end{aligned}$$

So, that as function of Δ, $\hat{\ell}' = 0$ implies:

$$\ell'(\alpha_i + \Delta) = -(\Delta\sigma_{ii} + 1)^{-2} \times \mathrm{const} \tag{A.63}$$

which can be easily solved numerically (e.g. by finding the roots of a cubic polynomial just as in the simple orthogonal/regression case).

By making the substitution $\alpha_i = 0, \Delta = \alpha_i^{\mathrm{new}}$, we obtain the interesting expression

$$\ell'(\alpha_i^{\mathrm{new}}) = -(\alpha_i^{\mathrm{new}}\sigma_{ii} + 1)^{-2} \times \mathrm{const} \tag{A.64}$$

$$\mathrm{const} = -c\frac{d}{d\alpha_i}\,\mathrm{tr}\,\mathbf{S}_{\alpha_i=0} = c\,\mathrm{tr}(\mathbf{Q}_{\alpha_i=0}\mathbf{\Phi}^\mathrm{T}\mathbf{B}\mathbf{\Phi}) \tag{A.65}$$

Reassuringly, for $\mathbf{B} = \sigma^{-2}\mathbf{I}$, $\mathbf{\Phi}^\mathrm{T}\mathbf{\Phi} = \mathbf{I}$ one obtains $\sigma_{ii} = \sigma^2$, $\boldsymbol{\sigma}_i = \sigma^2\mathbf{e}_i$ and $-\frac{d}{d\alpha_i}\,\mathrm{tr}\,\mathbf{S} = \mathrm{tr}\,\mathbf{Q}\sigma^{-2} = \mathrm{tr}(\boldsymbol{\sigma}_i\boldsymbol{\sigma}_i^\mathrm{T}\sigma^{-2}) = \sigma^2$ and thus again the familiar

$$\ell'(\alpha_i^{\mathrm{new}}) = -(\alpha_i^{\mathrm{new}}\sigma^2 + 1)^{-2} \times c\sigma^2 \tag{A.66}$$

The roots of the polynomial in Δ of the numerator of (A.63) are thus (omitting indices and writing $k :=$

[1]To check note $\mathbf{\Sigma} = \sigma^2(\mathbf{I} + \sigma^2\mathbf{A})^{-1}$, $\boldsymbol{\sigma}_i = \sigma_{ii}\mathbf{e}_i = \sigma^2(1 + \sigma^2\alpha_i)^{-1}$

$\text{const} = c\,\mathrm{tr}(\mathbf{Q}_{\alpha_i=0}\mathbf{\Phi}^\mathsf{T}\mathbf{B}\mathbf{\Phi})$, $\xi := \sigma_{ii}$ for compactness):

$$B_0 = 2\alpha k s^2 + s^2 + 4\alpha^2 k s + \alpha s - \alpha q^2 + 2\alpha^3 k \tag{A.67}$$

$$B_1 = 2s^2\xi + 2\alpha s\xi - 2\alpha q^2\xi + 2ks^2 + 8\alpha k s + s - q^2 + 6\alpha^2 k \tag{A.68}$$

$$B_2 = s^2\xi^2 + \alpha s\xi^2 - \alpha q^2\xi^2 + 2s\xi - 2q^2\xi + 4ks + 6\alpha k, \tag{A.69}$$

$$B_3 = s\xi^2 - q^2\xi^2 + 2k \tag{A.70}$$

However there's an issue which cannot arise of the optimization scheme we use in the orthonormal case: as the denominator is given by

$$2\left(\Delta + \alpha\right)\left(s + \Delta + \alpha\right)^2\left(\Delta\xi + 1\right)^2 \tag{A.71}$$

and Δ can be negative, a putative solution could be very unstable or in fact turn out to be a pole.

However, if we pick α_i to be either 0 or the None-solution, Δ will always be positive and these problems will not arise.

Appendix B

Some useful Identities and Results

B.1 Matrix trace, inverse and determinant identities

The following well-known identities can be found in many references – e.g. (Press et al., 1992) or (Golub and van Loan, 1996). In the following **A** and **B** are invertible matrices, **X**, **Y** and **Z** are arbitrary rectangular matrices (of course we assume that they have suitable dimensions for the products they occur in to be well-defined) and **u** and **v** are arbitrary column vectors (with the same caveat). All matrices and vectors are assumed to have real-valued entries simply because complex vectors and matrices play no role in this thesis.

Cyclic trace identity

$$\operatorname{tr}(\mathbf{XY}\ldots\mathbf{Z}) = \operatorname{tr}(\mathbf{ZXY}\ldots) \tag{B.1}$$

Sherman Morrison formula:

$$(\mathbf{B} + \mathbf{u}\mathbf{v}^{\mathsf{T}})^{-1} = \mathbf{B}^{-1} - \frac{\mathbf{B}^{-1}\mathbf{u}\mathbf{v}^{\mathsf{T}}\mathbf{B}^{-1}}{1 + \mathbf{v}^{\mathsf{T}}\mathbf{B}^{-1}\mathbf{u}} \tag{B.2}$$

Woodbury identity:

$$(\mathbf{B} + \mathbf{XAX}^{\mathsf{T}})^{-1} = \mathbf{B}^{-1} - \mathbf{B}^{-1}\mathbf{X}(\mathbf{A}^{-1} + \mathbf{X}^{\mathsf{T}}\mathbf{B}^{-1}\mathbf{X})^{-1}\mathbf{X}^{\mathsf{T}}\mathbf{B}^{-1} \tag{B.3}$$

Matrix determinant lemma:

$$|\mathbf{B} + \mathbf{XAX}^{\mathsf{T}}| = |\mathbf{A}|\,|\mathbf{B}|\,|\mathbf{A}^{-1} + \mathbf{X}^{\mathsf{T}}\mathbf{A}^{-1}\mathbf{X}| \tag{B.4}$$

Note: We give the less general versions because they suffice for this thesis. But one could in fact write **Y** *in place of* $\mathbf{X}^{\mathsf{T}}$ *in the two formulas above and they would remain correct. This shows that* (B.3) *it is the general case of (B.2) above.*

Appendix C

Some tedious calculations

As elsewhere in the thesis $\mathbf{\Phi}^{N\times M}$ is the design matrix ($\mathbf{K}^{M\times M} \equiv \mathbf{\Phi}^T\mathbf{\Phi}$, by definition), $\mathbf{w}^{M\times 1}$ the vector of weights and $\mathbf{y}^{N\times 1}$ the true signal.

$$\frac{\partial}{\partial w_i}\frac{1}{2}\|\mathbf{\Phi w}-\mathbf{y}\|^2 = \frac{1}{2}\frac{\partial}{\partial w_i}[\mathbf{w}^T\mathbf{\Phi}^T\mathbf{\Phi w}] - \frac{\partial}{\partial w_i}2\mathbf{y}^T\mathbf{\Phi w} \tag{C.1}$$

$$\frac{\partial}{\partial w_i}-2\mathbf{y}^T\mathbf{\Phi w} = -\phi_i^T\mathbf{y} \tag{C.2}$$

$$\frac{\partial}{\partial w_i}[\mathbf{w}^T\mathbf{\Phi}^T\mathbf{\Phi w}] \equiv \frac{\partial}{\partial w_i}[\mathbf{w}^T\mathbf{K w}] \tag{C.3}$$

$$= \frac{\partial}{\partial w_i}\left[\sum_j w_j \sum_l w_l k_{lj}\right] \tag{C.4}$$

$$= \sum_j \frac{\partial}{\partial w_i}[w_j]\sum_l w_l k_{lj} + \sum_j w_j \frac{\partial}{\partial w_i}\left[\sum_l w_l k_{lj}\right] \tag{C.5}$$

$$= \sum_l w_l k_{li} + \sum_j w_j k_{ij} \tag{C.6}$$

$$= 2\sum_l w_l k_{il} \qquad (k_{ij} = k_{ji}) \tag{C.7}$$

$$= 2\mathbf{K}_{[i,:]}\mathbf{w} \tag{C.8}$$

$$= 2[\mathbf{\Phi}^T\mathbf{\Phi}]_{[i,:]}\mathbf{w} \tag{C.9}$$

$$= 2\boldsymbol{\phi}_i^T\mathbf{\Phi w} \tag{C.10}$$

$$\Longrightarrow \tag{C.11}$$

$$\frac{\partial}{\partial w_i}\frac{1}{2}\|\mathbf{\Phi w}-\mathbf{y}\|^2 = \boldsymbol{\phi}_i^T(\mathbf{\Phi w}-\mathbf{y}) \tag{C.12}$$

$$\Longrightarrow \tag{C.13}$$

$$\frac{\partial}{\partial \mathbf{w}}\frac{1}{2}\|\mathbf{\Phi w}-\mathbf{y}\|^2 = \mathbf{\Phi}^T(\mathbf{\Phi w}-\mathbf{y}) \tag{C.14}$$

$$\Longrightarrow \tag{C.15}$$

$$\frac{\partial}{\partial \mathbf{w}}\frac{1}{2}\|\mathbf{\Phi w}-\mathbf{y}\|^2 = 0 \Rightarrow \mathbf{\Phi}^T(\mathbf{\Phi w}-\mathbf{y}) = \mathbf{0} \tag{C.16}$$

$$\Rightarrow \mathbf{\Phi}^T\mathbf{\Phi w} = \mathbf{\Phi}^T\mathbf{y} \tag{C.17}$$

$$\Rightarrow \mathbf{w} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{y} \tag{C.18}$$

Appendix D

Notation and Glossary

The reader in a hurry may want to skip straight ahead to the adjacent table comprising the most important variables and their place of definition; for everyone else here a few explanatory notes about the notation.

Apart from some minor adjustments, we have tried to follow the notation Tipping and collaborators used (Tipping, 2001; Faul and Tipping, 2002; Tipping and Faul, 2003) rather closely, but readers with a frequentist statistics background will be more used to $\mathbf{X}$ as design matrix (instead of $\mathbf{\Phi}$), $\boldsymbol{\beta}$ as regression weights (instead of $\mathbf{w}$) and $\mathbf{y}$ as targets (instead of $\mathbf{t}$).

Since the various syphilitic abuses of notation we inherit from the literature (in addition to those of our own making) may trip the unwary reader they shall receive attention before everything else:

As is customary, an expression such as $p(\mathbf{w}) = \int p(\mathbf{w} \mid \boldsymbol{\alpha}) p(\boldsymbol{\alpha}) d\boldsymbol{\alpha}$ commits a multitude of sins: despite the appearances the integral is not really indefinite but is understood to be taken over all possible values of the variable of integration (in this case $\boldsymbol{\alpha}$); p looks like but does not denote a single function, but two quite unrelated ones (which would be more cleanly written as say $p_a(\mathbf{w})$ and $p_b(\boldsymbol{\alpha})$)[1] and finally conditioning variables (such as the data points $\mathbf{X}$) may be silently omitted to save space. As is also customary, ranges will be inclusive at both ends, and indices will generally start at 1 rather then 0 and there are exceptions such as w_0 denoting the intercept.

Also, due to the finiteness of the Greek and Latin alphabet, and again established conventions, some identifiers will do double duty – thus, in regression σ is noise std whereas in classification σ refers to the sigmoid function, and, more benignly the vector $\boldsymbol{\phi}_i$ may be regarded as a shorthand for $\boldsymbol{\phi}(\mathbf{x}_i)$, the basis-function-transformed ith input vector.

After these words of warning a few general patterns:

Matrices are bold-uppercase ($\mathbf{\Phi}$), vectors are bold lowercase and columns from matrices are the subscripted bold lowercase equivalent ($\boldsymbol{\phi}_m$); whether the subscript refers to a column or row, alas, depends on the variable, in the case of $\mathbf{X}$ ($\mathbf{\Phi}$) where a row corresponds to a single (transformed) data point, $\mathbf{x}_i$ and $\boldsymbol{\phi}_i$ denote the ith row of the respective matrices, in many other contexts the difference does not matter as the matrix will be symmetric. In rare cases, such as when we wish to avoid notational clashes, we also use $_{[\cdot,\cdot]}$ as an explicit matrix indexing operator to write $\mathbf{\Phi}_{[:,j]}$ for the jth column of $\mathbf{\Phi}$ and $\mathbf{\Phi}_{[i,:]}$ for the ith row.

We loosely use hats to refer to ultimate, model-determined, quantities of interest such as (posterior mean) estimates (e.g. $\hat{\mathbf{y}}$) and tildes to refer to transformed quantities. To show that the influence of the ith component has been removed we use $_{\setminus i}$. We use $\equiv$ to mean "equals by definition", with some latitude as to where on the right or left hand side the defined quantity is actually introduced. The positive part of an expression is denoted by $_{+}$, i.e. $(x)_+ \equiv x$ if $x > 0$ else 0. Another established convention we adopt is writing $\delta_{ij} \equiv 1$ if $i = j$ else 0. A possibly less established convention is that we use $[x_m]_m^M$ analogously to $\{x_m\}_m^M$ to define the vector $\mathbf{x} = [x_1 \ldots x_m]^T$.

[1]This must especially be borne in mind in cases where a distribution is re-parametrized to ensure that the required change of variables is not omitted $\sigma^2 = \beta^{-1} \not\Rightarrow p(\sigma^2) = p(\beta^{-1})$, rather $p(\beta) = p(\sigma^2)|\frac{d\beta}{d\sigma^2}|$.

$\mathbf{y}^{N\times 1}$	the true signal (2.47)
$\mathbf{t}^{N\times 1}$	the observations or targets (2.47)
$\tilde{\mathbf{t}}^{N\times 1}$	the *pseudo-targets* for the linearized classification problem (2.145)
$\boldsymbol{\epsilon}$	the error (the difference between observations and true signal) (2.47); *classification* (2.156)
σ	(*in regression*) the standard deviation of the error (2.47), (2.50)
	(*in classification*) the (completely unrelated) sigmoid activation function (2.130)
β	the noise precision ($\beta = \sigma^{-2}$), (sec. A.4)
$\hat{\mathbf{y}}^{N\times 1}$	the posterior mean prediction for the true signal (2.55)
$\mathbf{\Phi}^{N\times M}$	the design matrix, viz. dictionary of basis functions can also refer to a kernel matrix where the difference does not matter (else see **K**) (2.48)
$\mathbf{\Phi}_{\mathcal{S}}^{N\times S}$	the selected components of the design matrix
$\boldsymbol{\phi}^{M\times 1}$	a *column* of the design matrix
$\boldsymbol{\phi}(\mathbf{x})$	the function that maps an input $\mathbf{x}_j$ into feature space; a single column of the design matrix $\mathbf{\Phi}$ has the form $\boldsymbol{\phi} = [\boldsymbol{\phi}(\mathbf{x}_1)\cdots\boldsymbol{\phi}(\mathbf{x}_N)]^N$
$\mathbf{K}^{N\times N}$	a kernel matrix
D	the dimensionality of a single input $\mathbf{x}_i$ and thus of the input space
N	the number of target observations
M	the number of basis functions viz. components and thus the dimensionality of feature space
S	the number of included basis functions viz. components (or non-zero w_m or finite α_m)
$\mathcal{S}$	the set of currently included components
$\mathbf{w}^{M\times 1}$	the weights ($\mathbf{\Phi}\mathbf{w} = \mathbf{y}$)
$\mathbf{\Sigma}^{M\times M}$	the posterior covariance matrix of the weights (2.53)
$\mathbf{\Sigma}_{LP}^{M\times M}$	the Laplace approximation for the covariance matrix for regression (2.139)
$\mathbf{\Sigma}_{S}^{S\times S}$	the posterior covariance matrix of the included weights
$\boldsymbol{\mu}^{M\times 1}$	the posterior mean of the weights (2.54)
$\boldsymbol{\mu}_{LP}^{M\times 1}$	the MAP solution for the $p(\mathbf{w}\,\vert\,\mathbf{t}, \boldsymbol{\alpha})$,(2.155) for a fixed $\boldsymbol{\alpha}$ used for the Laplace approximation to marginalize over $\mathbf{w}$ in classification
$\boldsymbol{\alpha}^{M\times 1}$	the precisions of the weights (2.49)
$\mathcal{L}(\boldsymbol{\alpha})$	the log-likelihood of the observations, $\log p(\mathbf{t}\,\vert\,\boldsymbol{\alpha}, \sigma^2)$, (2.64)
$\ell(\alpha_i)$	the contribution of the ith component to the likelihood $\mathcal{L}(\boldsymbol{\alpha})$ (2.63), (2.73)
$\mathcal{L}(\boldsymbol{\alpha}_{\setminus i})$	the log-likelihood of the observations without the contribution of the ith component(2.64),(A.25)
$\hat{\mathcal{L}}(\boldsymbol{\alpha})$	the log-posterior of the weight precisions $\boldsymbol{\alpha}$ (A.27)
$\hat{\ell}(\alpha_i)$	the contribution of the ith component to the posterior $\hat{\mathcal{L}}(\boldsymbol{\alpha})$ (2.80)
$\alpha_i^\star$	the value of α_i that maximizes $\hat{\ell}$ (2.79)
c	the hyperparameter that controls the degree of smoothing in the smoothness prior $p(\boldsymbol{\alpha}\vert\sigma^2)$ (2.59)
g, h	hyperparameters for the inverse Gamma distribution over σ^2 (2.50)
DF	the degrees of freedom of the smoothing matrix **S** hence an indicator of the complexity of the model (2.57), (2.58)
$\mathbf{S}^{N\times N}$	the smoothing matrix (2.55); *for classification:* (2.157)
$\mathbf{C}^{N\times N}$	covariance matrix of the data likelihood (2.69)
$\mathbf{C}_{\setminus i}^{N\times N}$	**C** without the influence of the ith component (2.69)
s_i, q_i	convenience variables that can be thought of as indices of sparsity and quality of component i (2.74), (2.75)
$\mathbf{x}_i^{D\times 1}$	a single input that corresponds to a single target t_i , the transpose of the ith *row* of **X**
$\mathbf{X}^{N\times(D+1)}$	a matrix of augmented inputs, each row is a bias term, 1, followed by $\mathbf{x}_i$, a D-dimensional input (2.4)