Chapter 6

Relationships between GPs and Other Models

In this chapter we discuss a number of concepts and models that are related to Gaussian process prediction. In section 6.1 we cover reproducing kernel Hilbert spaces (RKHSs), which define a Hilbert space of sufficiently-smooth functions corresponding to a given positive semidefinite kernel k.

As we discussed in chapter 1, there are many functions that are consistent with a given dataset $\mathcal{D}$. We have seen how the GP approach puts a prior over functions in order to deal with this issue. A related viewpoint is provided by *regularization* theory (described in section 6.2) where one seeks a trade-off between data-fit and the RKHS norm of function. This is closely related to the MAP estimator in GP prediction, and thus omits uncertainty in predictions and also the marginal likelihood. In section 6.3 we discuss splines, a special case of regularization which is obtained when the RKHS is defined in terms of differential operators of a given order.

There are a number of other families of kernel machines that are related to Gaussian process prediction. In section 6.4 we describe support vector machines, in section 6.5 we discuss least-squares classification (LSC), and in section 6.6 we cover relevance vector machines (RVMs).

6.1 Reproducing Kernel Hilbert Spaces

Here we present a brief introduction to reproducing kernel Hilbert spaces. The theory was developed by Aronszajn [1950]; a more recent treatise is Saitoh [1988]. Information can also be found in Wahba [1990], Schölkopf and Smola [2002] and Wegman [1982]. The collection of papers edited by Weinert [1982] provides an overview of the uses of RKHSs in statistical signal processing.

We start with a formal definition of a RKHS, and then describe two specific bases for a RKHS, firstly through Mercer's theorem and the eigenfunctions of k, and secondly through the reproducing kernel map.

Definition 6.1 *(Reproducing kernel Hilbert space). Let $\mathcal{H}$ be a Hilbert space of real functions f defined on an index set $\mathcal{X}$. Then $\mathcal{H}$ is called a reproducing kernel Hilbert space endowed with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ (and norm $\|f\|_{\mathcal{H}} = \sqrt{\langle f, f \rangle_{\mathcal{H}}}$) if there exists a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following properties:*

1. *for every $\mathbf{x}$, $k(\mathbf{x}, \mathbf{x}')$ as a function of $\mathbf{x}'$ belongs to $\mathcal{H}$, and*

reproducing property

2. *k has the* reproducing property *$\langle f(\cdot), k(\cdot, \mathbf{x}) \rangle_{\mathcal{H}} = f(\mathbf{x})$.* □

See e.g. Schölkopf and Smola [2002] and Wegman [1982]. Note also that as $k(\mathbf{x}, \cdot)$ and $k(\mathbf{x}', \cdot)$ are in $\mathcal{H}$ we have that $\langle k(\mathbf{x}, \cdot), k(\mathbf{x}', \cdot) \rangle_{\mathcal{H}} = k(\mathbf{x}, \mathbf{x}')$.

The RKHS uniquely determines k, and vice versa, as stated in the following theorem:

Theorem 6.1 *(Moore-Aronszajn theorem, Aronszajn [1950]). Let $\mathcal{X}$ be an index set. Then for every positive definite function $k(\cdot, \cdot)$ on $\mathcal{X} \times \mathcal{X}$ there exists a unique RKHS, and vice versa.* □

The Hilbert space L_2 (which has the dot product $\langle f, g \rangle_{L_2} = \int f(\mathbf{x}) g(\mathbf{x}) d\mathbf{x}$) contains many non-smooth functions. In L_2 (which is not a RKHS) the delta function is the representer of evaluation, i.e. $f(\mathbf{x}) = \int f(\mathbf{x}') \delta(\mathbf{x} - \mathbf{x}') d\mathbf{x}'$. Kernels are the analogues of delta functions within the smoother RKHS. Note that the delta function is not itself in L_2; in contrast for a RKHS the kernel k is the representer of evaluation and is itself in the RKHS.

The above description is perhaps rather abstract. For our purposes the key intuition behind the RKHS formalism is that the squared norm $\|f\|^2_{\mathcal{H}}$ can be thought of as a generalization to functions of the n-dimensional quadratic form $\mathbf{f}^\top K^{-1} \mathbf{f}$ we have seen in earlier chapters.

Consider a real positive semidefinite kernel $k(\mathbf{x}, \mathbf{x}')$ with an eigenfunction expansion $k(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{N} \lambda_i \phi_i(\mathbf{x}) \phi_i(\mathbf{x}')$ relative to a measure μ. Recall from Mercer's theorem that the eigenfunctions are orthonormal w.r.t. μ, i.e. we have $\int \phi_i(\mathbf{x}) \phi_j(\mathbf{x})\, d\mu(\mathbf{x}) = \delta_{ij}$. We now consider a Hilbert space comprised of linear combinations of the eigenfunctions, i.e. $f(\mathbf{x}) = \sum_{i=1}^{N} f_i \phi_i(\mathbf{x})$ with $\sum_{i=1}^{N} f_i^2 / \lambda_i < \infty$. We assert that the inner product $\langle f, g \rangle_{\mathcal{H}}$ in the Hilbert space between functions $f(\mathbf{x})$ and $g(\mathbf{x}) = \sum_{i=1}^{N} g_i \phi_i(\mathbf{x})$ is defined as

inner product $\langle f, g \rangle_{\mathcal{H}}$

$$\langle f, g \rangle_{\mathcal{H}} \;=\; \sum_{i=1}^{N} \frac{f_i g_i}{\lambda_i}. \tag{6.1}$$

Thus this Hilbert space is equipped with a norm $\|f\|_{\mathcal{H}}$ where $\|f\|^2_{\mathcal{H}} = \langle f, f \rangle_{\mathcal{H}} = \sum_{i=1}^{N} f_i^2 / \lambda_i$. Note that for $\|f\|_{\mathcal{H}}$ to be finite the sequence of coefficients $\{f_i\}$ must decay quickly; effectively this imposes a smoothness condition on the space.

We now need to show that this Hilbert space is the RKHS corresponding to the kernel k, i.e. that it has the reproducing property. This is easily achieved as

$$\langle f(\cdot), k(\cdot, \mathbf{x})\rangle_{\mathcal{H}} \;=\; \sum_{i=1}^{N} \frac{f_i \lambda_i \phi_i(\mathbf{x})}{\lambda_i} = f(\mathbf{x}). \tag{6.2}$$

Similarly

$$\langle k(\mathbf{x}, \cdot), k(\mathbf{x}', \cdot)\rangle_{\mathcal{H}} \;=\; \sum_{i=1}^{N} \frac{\lambda_i \phi_i(\mathbf{x}) \lambda_i \phi_i(\mathbf{x}')}{\lambda_i} = k(\mathbf{x}, \mathbf{x}'). \tag{6.3}$$

Notice also that $k(\mathbf{x}, \cdot)$ is in the RKHS as it has norm $\sum_{i=1}^{N} (\lambda_i \phi_i(\mathbf{x}))^2/\lambda_i = k(\mathbf{x}, \mathbf{x}) < \infty$. We have now demonstrated that the Hilbert space comprised of linear combinations of the eigenfunctions with the restriction $\sum_{i=1}^{N} f_i^2/\lambda_i < \infty$ fulfils the two conditions given in Definition 6.1. As there is a unique RKHS associated with $k(\cdot, \cdot)$, this Hilbert space must be that RKHS.

The advantage of the abstract formulation of the RKHS is that the eigenbasis will change as we use different measures μ in Mercer's theorem. However, the RKHS norm is in fact solely a property of the kernel and is invariant under this change of measure. This can be seen from the fact that the proof of the RKHS properties above is not dependent on the measure; see also Kailath [1971, sec. II.B]. A finite-dimensional example of this measure invariance is explored in exercise 6.7.1.

Notice the analogy between the RKHS norm $\|f\|^2_{\mathcal{H}} = \langle f, f\rangle_{\mathcal{H}} = \sum_{i=1}^{N} f_i^2/\lambda_i$ and the quadratic form $\mathbf{f}^\top K^{-1}\mathbf{f}$; if we express K and $\mathbf{f}$ in terms of the eigenvectors of K we obtain exactly the same form (but the sum has only n terms if $\mathbf{f}$ has length n).

If we sample the coefficients f_i in the eigenexpansion $f(\mathbf{x}) = \sum_{i=1}^{N} f_i \phi_i(\mathbf{x})$ from $\mathcal{N}(0, \lambda_i)$ then

$$\mathbb{E}[\|f\|^2_{\mathcal{H}}] \;=\; \sum_{i=1}^{N} \frac{\mathbb{E}[f_i^2]}{\lambda_i} \;=\; \sum_{i=1}^{N} 1. \tag{6.4}$$

Thus if N is infinite the sample functions are not in $\mathcal{H}$ (with probability 1) as the expected value of the RKHS norm is infinite; see Wahba [1990, p. 5] and Kailath [1971, sec. II.B] for further details. However, note that although sample functions of this Gaussian process are not in $\mathcal{H}$, the posterior mean after observing some data will lie in the RKHS, due to the smoothing properties of averaging.

Another view of the RKHS can be obtained from the *reproducing kernel map* construction. We consider the space of functions f defined as

$$\Big\{ f(\mathbf{x}) \;=\; \sum_{i=1}^{n} \alpha_i k(\mathbf{x}, \mathbf{x}_i) \;:\; n \in \mathbb{N},\; \mathbf{x}_i \in \mathcal{X},\; \alpha_i \in \mathbb{R} \Big\}. \tag{6.5}$$

Now let $g(\mathbf{x}) = \sum_{j=1}^{n'} \alpha'_j k(\mathbf{x}, \mathbf{x}'_j)$. Then we define the inner product

$$\langle f, g \rangle_{\mathcal{H}} \; = \; \sum_{i=1}^{n} \sum_{j=1}^{n'} \alpha_i \alpha'_j k(\mathbf{x}_i, \mathbf{x}'_j). \tag{6.6}$$

Clearly condition 1 of Definition 6.1 is fulfilled under the reproducing kernel map construction. We can also demonstrate the reproducing property, as

$$\langle k(\cdot, \mathbf{x}), f(\cdot) \rangle_{\mathcal{H}} \; = \; \sum_{i=1}^{n} \alpha_i k(\mathbf{x}, \mathbf{x}_i) \; = \; f(\mathbf{x}). \tag{6.7}$$

6.2 Regularization

The problem of inferring an underlying function $f(\mathbf{x})$ from a finite (and possibly noisy) dataset without any additional assumptions is clearly “ill posed”. For example, in the noise-free case, *any* function that passes through the given data points is acceptable. Under a Bayesian approach our assumptions are characterized by a prior over functions, and given some data, we obtain a posterior over functions. The problem of bringing prior assumptions to bear has also been addressed under the *regularization* viewpoint, where these assumptions are encoded in terms of the smoothness of f.

We consider the functional

$$J[f] = \frac{\lambda}{2} \|f\|^2_{\mathcal{H}} + Q(\mathbf{y}, \mathbf{f}), \tag{6.8}$$

where $\mathbf{y}$ is the vector of targets we are predicting and $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))^\top$ is the corresponding vector of function values, and λ is a scaling parameter that trades off the two terms. The first term is called the *regularizer* and represents smoothness assumptions on f as encoded by a suitable RKHS, and the second term is a data-fit term assessing the quality of the prediction $f(\mathbf{x}_i)$ for the observed datum y_i, e.g. the negative log likelihood.

regularizer

(kernel) ridge regression

Ridge regression (described in section 2.1) can be seen as a particular case of regularization. Indeed, recalling that $\|f\|^2_{\mathcal{H}} = \sum_{i=1}^{N} f_i^2/\lambda_i$ where f_i is the coefficient of eigenfunction $\phi_i(\mathbf{x})$, we see that we are penalizing the weighted squared coefficients. This is taking place in feature space, rather than simply in input space, as per the standard formulation of ridge regression (see eq. (2.4)), so it corresponds to *kernel* ridge regression.

representer theorem

The *representer theorem* shows that each minimizer $f \in \mathcal{H}$ of $J[f]$ has the form $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}, \mathbf{x}_i)$.[1] The representer theorem was first stated by Kimeldorf and Wahba [1971] for the case of squared error.[2] O’Sullivan et al. [1986] showed that the representer theorem could be extended to likelihood

[1] If the RKHS contains a null space of unpenalized functions then the given form is correct modulo a term that lies in this null space. This is explained further in section 6.3.

[2] Schoenberg [1964] proved the representer theorem for the special case of cubic splines and squared error. This was result extended to general RKHSs in Kimeldorf and Wahba [1971].

functions arising from generalized linear models. The representer theorem can be generalized still further, see e.g. Schölkopf and Smola [2002, sec. 4.2]. If the data-fit term is convex (see section A.9) then there will be a unique minimizer $\hat{f}$ of $J[f]$.

For Gaussian process prediction with likelihoods that involve the values of f at the n training points only (so that $Q(\mathbf{y}, \mathbf{f})$ is the negative log likelihood up to some terms not involving $\mathbf{f}$), the analogue of the representer theorem is obvious. This is because the predictive distribution of $f(\mathbf{x}_*) \triangleq f_*$ at test point $\mathbf{x}_*$ given the data $\mathbf{y}$ is $p(f_*|\mathbf{y}) = \int p(f_*|\mathbf{f})p(\mathbf{f}|\mathbf{y})\, d\mathbf{f}$. As derived in eq. (3.22) we have

$$\mathbb{E}[f_*|\mathbf{y}] \;=\; \mathbf{k}(\mathbf{x}_*)^\top K^{-1}\mathbb{E}[\mathbf{f}|\mathbf{y}] \tag{6.9}$$

due to the formulae for the conditional distribution of a multivariate Gaussian. Thus $\mathbb{E}[f_*|\mathbf{y}] = \sum_{i=1}^n \alpha_i k(\mathbf{x}_*, \mathbf{x}_i)$, where $\boldsymbol{\alpha} = K^{-1}\mathbb{E}[\mathbf{f}|\mathbf{y}]$.

The regularization approach has a long tradition in inverse problems, dating back at least as far as Tikhonov [1963]; see also Tikhonov and Arsenin [1977]. For the application of this approach in the machine learning literature see e.g. Poggio and Girosi [1990].

In section 6.2.1 we consider RKHSs defined in terms of differential operators. In section 6.2.2 we demonstrate how to solve the regularization problem in the specific case of squared error, and in section 6.2.3 we compare and contrast the regularization approach with the Gaussian process viewpoint.

6.2.1 Regularization Defined by Differential Operators *

For $\mathbf{x} \in \mathbb{R}^D$ define

$$\|O^m f\|^2 \;=\; \int \sum_{j_1+\ldots+j_D=m} \Big(\frac{\partial^m f(\mathbf{x})}{\partial x_1^{j_1} \ldots x_D^{j_D}}\Big)^2 d\mathbf{x}. \tag{6.10}$$

For example for $m = 2$ and $D = 2$

$$\|O^2 f\|^2 \;=\; \int \Big[\Big(\frac{\partial^2 f}{\partial x_1^2}\Big)^2 + 2\Big(\frac{\partial^2 f}{\partial x_1 \partial x_2}\Big)^2 + \Big(\frac{\partial^2 f}{\partial x_2^2}\Big)^2\Big]\, dx_1\, dx_2. \tag{6.11}$$

Now set $\|Pf\|^2 = \sum_{m=0}^M a_m \|O^m f\|^2$ with non-negative coefficients a_m. Notice that $\|Pf\|^2$ is translation and rotation invariant.

null space

In this section we assume that $a_0 > 0$; if this is not the case and a_k is the first non-zero coefficient, then there is a *null space* of functions that are unpenalized. For example if $k = 2$ then constant and linear functions are in the null space. This case is dealt with in section 6.3.

$\|Pf\|^2$ penalizes f in terms of the variability of its function values and derivatives up to order M. How does this correspond to the RKHS formulation of section 6.1? The key is to recognize that the complex exponentials $\exp(2\pi i \mathbf{s} \cdot \mathbf{x})$ are eigenfunctions of the differential operator if $\mathcal{X} = \mathbb{R}^D$. In this case

$$\|Pf\|^2 = \int \sum_{m=0}^M a_m (4\pi^2 \mathbf{s} \cdot \mathbf{s})^m |\tilde{f}(\mathbf{s})|^2 d\mathbf{s}, \tag{6.12}$$

where $\tilde{f}(\mathbf{s})$ is the Fourier transform of $f(\mathbf{x})$. Comparing eq. (6.12) with eq. (6.1) we see that the kernel has the power spectrum

$$S(\mathbf{s}) = \frac{1}{\sum_{m=0}^{M} a_m (4\pi^2 \mathbf{s}\cdot\mathbf{s})^m}, \tag{6.13}$$

and thus by Fourier inversion we obtain the stationary kernel

$$k(\mathbf{x}) = \int \frac{e^{2\pi i \mathbf{s}\cdot\mathbf{x}}}{\sum_{m=0}^{M} a_m (4\pi^2 \mathbf{s}\cdot\mathbf{s})^m}\, d\mathbf{s}. \tag{6.14}$$

A slightly different approach to obtaining the kernel is to use calculus of variations to minimize $J[f]$ with respect to f. The Euler-Lagrange equation leads to

$$f(\mathbf{x}) \;=\; \sum_{i=1}^{n} \alpha_i G(\mathbf{x}-\mathbf{x}_i), \tag{6.15}$$

with

$$\sum_{m=0}^{M} (-1)^m a_m \nabla^{2m} G \;=\; \delta(\mathbf{x}-\mathbf{x}'), \tag{6.16}$$

Green's function
≡ kernel

where $G(\mathbf{x},\mathbf{x}')$ is known as a Green's function. Notice that the Green's function also depends on the boundary conditions. For the case of $\mathcal{X} = \mathbb{R}^D$ by Fourier transforming eq. (6.16) we recognize that G is in fact the kernel k. The differential operator $\sum_{m=0}^{M}(-1)^m a_m \nabla^{2m}$ and the integral operator $k(\cdot,\cdot)$ are in fact inverses, as shown by eq. (6.16). See Poggio and Girosi [1990] for further details. Arfken [1985] provides an introduction to calculus of variations and Green's functions. RKHSs for regularizers defined by differential operators are *Sobolev spaces*; see e.g. Adams [1975] for further details on Sobolev spaces.

We now give two specific examples of kernels derived from differential operators.

Example 1. Set $a_0 = \alpha^2$, $a_1 = 1$ and $a_m = 0$ for $m \geq 2$ in $D = 1$. Using the Fourier pair $e^{-\alpha|x|} \leftrightarrow 2\alpha/(\alpha^2 + 4\pi^2 s^2)$ we obtain $k(x-x') = \frac{1}{2\alpha} e^{-\alpha|x-x'|}$. Note that this is the covariance function of the Ornstein-Uhlenbeck process, see section 4.2.1.

Example 2. By setting $a_m = \frac{\sigma^{2m}}{m! 2^m}$ and using the power series $e^y = \sum_{k=0}^{\infty} y^k/k!$ we obtain

$$k(\mathbf{x}-\mathbf{x}') \;=\; \int \exp(2\pi i \mathbf{s}\cdot(\mathbf{x}-\mathbf{x}'))\exp(-\frac{\sigma^2}{2}(4\pi^2\mathbf{s}\cdot\mathbf{s}))d\mathbf{s} \tag{6.17}$$

$$\;=\; \frac{1}{(2\pi\sigma^2)^{D/2}}\exp(-\frac{1}{2\sigma^2}(\mathbf{x}-\mathbf{x}')^\top(\mathbf{x}-\mathbf{x}')), \tag{6.18}$$

as shown by Yuille and Grzywacz [1989]. This is the squared exponential covariance function that we have seen earlier.

6.2.2 Obtaining the Regularized Solution

The representer theorem tells us the general form of the solution to eq. (6.8). We now consider a specific functional

$$J[f] \;=\; \frac{1}{2}\|f\|^2_{\mathcal{H}} + \frac{1}{2\sigma_n^2}\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2, \tag{6.19}$$

which uses a squared error data-fit term (corresponding to the negative log likelihood of a Gaussian noise model with variance σ_n^2). Substituting $f(\mathbf{x}) = \sum_{i=1}^{n}\alpha_i k(\mathbf{x}, \mathbf{x}_i)$ and using $\langle k(\cdot, \mathbf{x}_i), k(\cdot, \mathbf{x}_j)\rangle_{\mathcal{H}} = k(\mathbf{x}_i, \mathbf{x}_j)$ we obtain

$$\begin{aligned} J[\boldsymbol{\alpha}] \;&=\; \frac{1}{2}\boldsymbol{\alpha}^\top K\boldsymbol{\alpha} + \frac{1}{2\sigma_n^2}|\mathbf{y} - K\boldsymbol{\alpha}|^2 \\ &=\; \frac{1}{2}\boldsymbol{\alpha}^\top(K + \frac{1}{\sigma_n^2}K^2)\boldsymbol{\alpha} - \frac{1}{\sigma_n^2}\mathbf{y}^\top K\boldsymbol{\alpha} + \frac{1}{2\sigma_n^2}\mathbf{y}^\top\mathbf{y}. \end{aligned} \tag{6.20}$$

Minimizing J by differentiating w.r.t. the vector of coefficients $\boldsymbol{\alpha}$ we obtain $\hat{\boldsymbol{\alpha}} = (K + \sigma_n^2 I)^{-1}\mathbf{y}$, so that the prediction for a test point $\mathbf{x}_*$ is $\hat{f}(\mathbf{x}_*) = \mathbf{k}(\mathbf{x}_*)^\top(K + \sigma_n^2 I)^{-1}\mathbf{y}$. This should look very familiar—it is exactly the form of the predictive mean obtained in eq. (2.23). In the next section we compare and contrast the regularization and GP views of the problem.

regularization network

The solution $f(\mathbf{x}) = \sum_{i=1}^{n}\alpha_i k(\mathbf{x}, \mathbf{x}_i)$ that minimizes eq. (6.19) was called a *regularization network* in Poggio and Girosi [1990].

6.2.3 The Relationship of the Regularization View to Gaussian Process Prediction

The regularization method returns $\hat{f} = \text{argmin}_f\; J[f]$. For a Gaussian process predictor we obtain a posterior distribution over functions. Can we make a connection between these two views? In fact we shall see in this section that $\hat{f}$ can be viewed as the *maximum a posteriori* (MAP) function under the posterior.

Following Szeliski [1987] and Poggio and Girosi [1990] we consider

$$\exp\left(-J[f]\right) \;=\; \exp\big(-\frac{\lambda}{2}\|Pf\|^2\big) \;\times\; \exp\left(-Q(\mathbf{y}, \mathbf{f})\right). \tag{6.21}$$

The first term on the RHS is a Gaussian process prior on f, and the second is proportional to the likelihood. As $\hat{f}$ is the minimizer of $J[f]$, it is the MAP function.

To get some intuition for the Gaussian process prior, imagine $f(\mathbf{x})$ being represented on a grid in $\mathbf{x}$-space, so that f is now an (infinite dimensional) vector $\mathbf{f}$. Thus we obtain $\|Pf\|^2 \simeq \sum_{m=0}^{M} a_m(D_m\mathbf{f})^\top(D_m\mathbf{f}) = \mathbf{f}^\top(\sum_m a_m D_m^\top D_m)\mathbf{f}$ where D_m is an appropriate finite-difference approximation of the differential operator O^m. Observe that this prior term is a quadratic form in $\mathbf{f}$.

To go into more detail concerning the MAP relationship we consider three cases: (i) when $Q(\mathbf{y}, \mathbf{f})$ is quadratic (corresponding to a Gaussian likelihood);

(ii) when $Q(\mathbf{y}, \mathbf{f})$ is not quadratic but convex and (iii) when $Q(\mathbf{y}, \mathbf{f})$ is not convex.

In case (i) we have seen in chapter 2 that the posterior mean function can be obtained exactly, and the posterior is Gaussian. As the mean of a Gaussian is also its mode this is the MAP solution. The correspondence between the GP posterior mean and the solution of the regularization problem $\hat{f}$ was made in Kimeldorf and Wahba [1970].

In case (ii) we have seen in chapter 3 for classification problems using the logistic, probit or softmax response functions that $Q(\mathbf{y}, \mathbf{f})$ is convex. Here the MAP solution can be found by finding $\hat{\mathbf{f}}$ (the MAP solution to the n-dimensional problem defined at the training points) and then extending it to other $\mathbf{x}$-values through the posterior mean conditioned on $\hat{\mathbf{f}}$.

In case (iii) there will be more than one local minimum of $J[f]$ under the regularization approach. One could check these minima to find the deepest one. However, in this case the argument for MAP is rather weak (especially if there are multiple optima of similar depth) and suggests the need for a fully Bayesian treatment.

While the regularization solution gives a part of the Gaussian process solution, there are the following limitations:

1. It does not characterize the uncertainty in the predictions, nor does it handle well multimodality in the posterior.

2. The analysis is focussed at approximating the first level of Bayesian inference, concerning predictions for f. It is not usually extended to the next level, e.g. to the computation of the marginal likelihood. The marginal likelihood is very useful for setting any parameters of the covariance function, and for model comparison (see chapter 5).

In addition, we find the specification of smoothness via the penalties on derivatives to be not very intuitive. The regularization viewpoint can be thought of as directly specifying the *inverse* covariance rather than the covariance. As marginalization is achieved for a Gaussian distribution directly from the covariance (and not the inverse covariance) it seems more natural to us to specify the covariance function. Also, while non-stationary covariance functions can be obtained from the regularization viewpoint, e.g. by replacing the Lebesgue measure in eq. (6.10) with a non-uniform measure $\mu(\mathbf{x})$, calculation of the corresponding covariance function can then be very difficult.

6.3 Spline Models

In section 6.2 we discussed regularizers which had $a_0 > 0$ in eq. (6.12). We now consider the case when $a_0 = 0$; in particular we consider the regularizer to be of the form $\|O^m f\|^2$, as defined in eq. (6.10). In this case polynomials of degree up to $m-1$ are in the *null space* of the regularization operator, in that they are not penalized at all.

In the case that $\mathcal{X} = \mathbb{R}^D$ we can again use Fourier techniques to obtain the Green's function G corresponding to the Euler-Lagrange equation $(-1)^m \nabla^{2m} G(\mathbf{x}) = \delta(\mathbf{x})$. The result, as shown by Duchon [1977] and Meinguet [1979] is

$$G(\mathbf{x}-\mathbf{x}') \;=\; \begin{cases} c_{m,D}|\mathbf{x}-\mathbf{x}'|^{2m-D}\log|\mathbf{x}-\mathbf{x}'| & \text{if } 2m > D \text{ and } D \text{ even} \\ c_{m,D}|\mathbf{x}-\mathbf{x}'|^{2m-D} & \text{otherwise,} \end{cases} \tag{6.22}$$

where $c_{m,D}$ is a constant (Wahba [1990, p. 31] gives the explicit form). Note that the constraint $2m > D$ has to be imposed to avoid having a Green's function that is singular at the origin. Explicit calculation of the Green's function for other domains $\mathcal{X}$ is sometimes possible; for example see Wahba [1990, sec. 2.2] for splines on the sphere.

Because of the null space, a minimizer of the regularization functional has the form

$$f(\mathbf{x}) \;=\; \sum_{i=1}^{n} \alpha_i G(\mathbf{x}, \mathbf{x}_i) + \sum_{j=1}^{k} \beta_j h_j(\mathbf{x}), \tag{6.23}$$

where $h_1(\mathbf{x}), \ldots, h_k(\mathbf{x})$ are polynomials that span the null space. The exact values of the coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ for a specific problem can be obtained in an analogous manner to the derivation in section 6.2.2; in fact the solution is equivalent to that given in eq. (2.42).

To gain some more insight into the form of the Green's function we consider the equation $(-1)^m \nabla^{2m} G(\mathbf{x}) = \delta(\mathbf{x})$ in Fourier space, leading to $\tilde{G}(\mathbf{s}) = (4\pi \mathbf{s} \cdot \mathbf{s})^{-m}$. $\tilde{G}(\mathbf{s})$ plays a rôle like that of the power spectrum in eq. (6.13), but notice that $\int \tilde{G}(\mathbf{s}) d\mathbf{s}$ is infinite, which would imply that the corresponding process has infinite variance. The problem is of course that the null space is unpenalized; for example any arbitrary constant function can be added to f without changing the regularizer.

Because of the null space we have seen that one cannot obtain a simple connection between the spline solution and a corresponding Gaussian process problem. However, by introducing the notion of an *intrinsic random function* (IRF) one can define a *generalized covariance*; see Cressie [1993, sec. 5.4] and Stein [1999, section 2.9] for details. The basic idea is to consider linear combinations of $f(\mathbf{x})$ of the form $g(\mathbf{x}) = \sum_{i=1}^{k} a_i f(\mathbf{x} + \boldsymbol{\delta}_i)$ for which $g(\mathbf{x})$ is second-order stationary and where $(h_j(\boldsymbol{\delta}_1), \ldots, h_j(\boldsymbol{\delta}_k))\mathbf{a} = 0$ for $j = 1, \ldots, k$. A careful description of the equivalence of spline and IRF prediction is given in Kent and Mardia [1994].

IRF

The power-law form of $\tilde{G}(\mathbf{s}) = (4\pi \mathbf{s} \cdot \mathbf{s})^{-m}$ means that there is no characteristic length-scale for random functions drawn from this (improper) prior. Thus we obtain the *self-similar* property characteristic of fractals; for further details see Szeliski [1987] and Mandelbrot [1982]. Some authors argue that the lack of a characteristic length-scale is appealing. This may sometimes be the case, but if we believe there is an appropriate length-scale (or set of length-scales) for a given problem but this is unknown in advance, we would argue that a hierarchical Bayesian formulation of the problem (as described in chapter 5) would be more appropriate.

spline interpolation

Splines were originally introduced for one-dimensional interpolation and smoothing problems, and then generalized to the multivariate setting. Schoenberg [1964] considered the problem of finding the function that minimizes

$$\int_a^b (f^{(m)}(x))^2\, dx, \tag{6.24}$$

natural polynomial spline

smoothing spline

where $f^{(m)}$ denotes the m'th derivative of f, subject to the interpolation constraints $f(x_i) = f_i$, $x_i \in (a, b)$ for $i = 1, \ldots, n$ and for f in an appropriate Sobolev space. He showed that the solution is the natural polynomial spline, which is a piecewise polynomial of order $2m - 1$ in each interval $[x_i, x_{i+1}]$, $i = 1, \ldots, n-1$, and of order $m - 1$ in the two outermost intervals. The pieces are joined so that the solution has $2m - 2$ continuous derivatives. Schoenberg also proved that the solution to the univariate smoothing problem (see eq. (6.19)) is a natural polynomial spline. A common choice is $m = 2$, leading to the cubic spline. One possible way of writing this solution is

$$f(x) \;=\; \sum_{j=0}^{1} \beta_j x^j + \sum_{i=1}^{n} \alpha_i (x - x_i)_+^3, \quad \text{where } (x)_+ \;=\; \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{6.25}$$

It turns out that the coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ can be computed in time $\mathcal{O}(n)$ using an algorithm due to Reinsch; see Green and Silverman [1994, sec. 2.3.3] for details.

Splines were first used in regression problems. However, by using generalized linear modelling [McCullagh and Nelder, 1983] they can be extended to classification problems and other non-Gaussian likelihoods, as we did for GP classification in section 3.3. Early references in this direction include Silverman [1978] and O'Sullivan et al. [1986].

There is a vast literature in relation to splines in both the statistics and numerical analysis literatures; for entry points see citations in Wahba [1990] and Green and Silverman [1994].

∗ 6.3.1 A 1-d Gaussian Process Spline Construction

In this section we will further clarify the relationship between splines and Gaussian processes by giving a GP construction for the solution of the univariate cubic spline smoothing problem whose cost functional is

$$\sum_{i=1}^{n} \big(f(x_i) - y_i\big)^2 + \lambda \int_0^1 \big(f''(x)\big)^2\, dx, \tag{6.26}$$

where the observed data are $\{(x_i, y_i) | i = 1, \ldots, n,\ 0 < x_1 < \cdots < x_n < 1\}$ and λ is a *smoothing parameter* controlling the trade-off between the first term, the data-fit, and the second term, the regularizer, or complexity penalty. Recall that the solution is a piecewise polynomial as in eq. (6.25).

Following Wahba [1978], we consider the random function

$$g(x) = \sum_{j=0}^{1} \beta_j x^j + f(x) \tag{6.27}$$

where $\boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \sigma_\beta^2 I)$ and $f(x)$ is a Gaussian process with covariance $\sigma_f^2 k_{\mathrm{sp}}(x, x')$, where

$$k_{\mathrm{sp}}(x, x') \triangleq \int_0^1 (x-u)_+ (x'-u)_+ \, du = \frac{|x - x'| v^2}{2} + \frac{v^3}{3}, \tag{6.28}$$

and $v = \min(x, x')$.

To complete the analogue of the regularizer in eq. (6.26), we need to remove any penalty on polynomial terms in the null space by making the prior vague, i.e. by taking the limit $\sigma_\beta^2 \to \infty$. Notice that the covariance has the form of contributions from explicit basis functions, $\mathbf{h}(x) = (1, x)^\top$ and a regular covariance function $k_{\mathrm{sp}}(x, x')$, a problem which we have already studied in section 2.7. Indeed we have computed the limit where the prior becomes vague $\sigma_\beta^2 \to \infty$, the result is given in eq. (2.42).

Plugging into the mean equation from eq. (2.42), we get the predictive mean

$$\bar{f}(x_*) \; = \; \mathbf{k}(x_*)^\top K_y^{-1} (\mathbf{y} - H^\top \bar{\boldsymbol{\beta}}) + \mathbf{h}(x_*)^\top \bar{\boldsymbol{\beta}}, \tag{6.29}$$

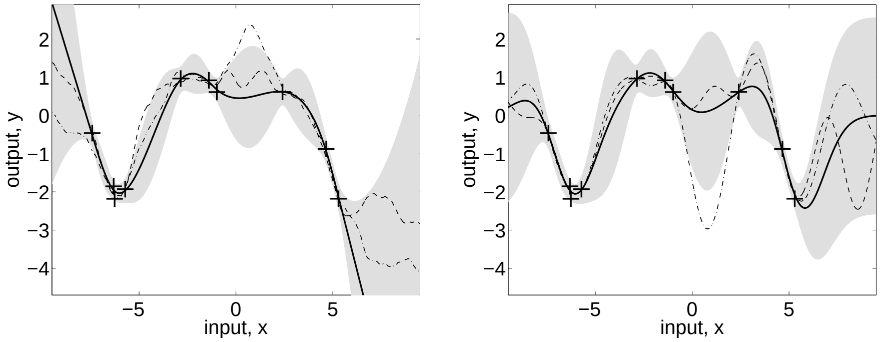
where K_y is the covariance matrix corresponding to $\sigma_f^2 k_{\mathrm{sp}}(x_i, x_j) + \sigma_n^2 \delta_{ij}$ evaluated at the training points, H is the matrix that collects the $\mathbf{h}(x_i)$ vectors at all training points, and $\bar{\boldsymbol{\beta}} = (H K_y^{-1} H^\top)^{-1} H K_y^{-1} \mathbf{y}$ is given below eq. (2.42). It is not difficult to show that this predictive mean function is a piecewise cubic polynomial, since the elements of $\mathbf{k}(x_*)$ are piecewise[3] cubic polynomials. Showing that the mean function is a first order polynomial in the outer intervals $[0, x_1]$ and $[x_n, 1]$ is left as exercise 6.7.3.

So far k_{sp} has been produced rather mysteriously "from the hat"; we now provide some explanation. Shepp [1966] defined the l-fold integrated Wiener process as

$$W_l(x) = \int_0^1 \frac{(x-u)_+^l}{l!} Z(u), \qquad l = 0,\ 1,\ \ldots \tag{6.30}$$

where $Z(u)$ denotes the Gaussian white noise process with covariance $\delta(u - u')$. Note that W_0 is the standard Wiener process. It is easy to show that $k_{\mathrm{sp}}(x, x')$ is the covariance of the once-integrated Wiener process by writing $W_1(x)$ and $W_1(x')$ using eq. (6.30) and taking the expectation using the covariance of the white noise process. Note that W_l is the solution to the stochastic differential equation (SDE) $X^{(l+1)} = Z$; see Appendix B for further details on SDEs. Thus

[3] The pieces are joined at the datapoints, the points where the $\min(x, x')$ from the covariance function is non-differentiable.



(a), spline covariance (b), squared exponential cov.

Figure 6.1: Panel (a) shows the application of the spline covariance to a simple dataset. The full line shows the predictive mean, which is a piecewise cubic polynomial, and the grey area indicates the 95% confidence area. The two thin dashed and dash-dotted lines are samples from the posterior. Note that the posterior samples are not as smooth as the mean. For comparison a GP using the squared exponential covariance function is shown in panel (b). The hyperparameters in both cases were optimized using the marginal likelihood.

for the cubic spline we set $l = 1$ to obtain the SDE $X'' = Z$, corresponding to the regularizer $\int (f''(x))^2 dx$.

We can also give an explicit basis-function construction for the covariance function k_{sp}. Consider the family of random functions given by

$$f_N(x) \;=\; \frac{1}{\sqrt{N}} \sum_{i=0}^{N-1} \gamma_i (x - \frac{i}{N})_+, \tag{6.31}$$

where $\boldsymbol{\gamma}$ is a vector of parameters with $\boldsymbol{\gamma} \;\sim\; \mathcal{N}(\mathbf{0},\, I)$. Note that the sum has the form of evenly spaced "ramps" whose magnitudes are given by the entries in the $\boldsymbol{\gamma}$ vector. Thus

$$\mathbb{E}[f_N(x) f_N(x')] \;=\; \frac{1}{N} \sum_{i=0}^{N-1} (x - \frac{i}{N})_+ (x' - \frac{i}{N})_+. \tag{6.32}$$

Taking the limit $N \to \infty$, we obtain eq. (6.28), a derivation which is also found in [Vapnik, 1998, sec. 11.6].

Notice that the covariance function k_{sp} given in eq. (6.28) corresponds to a Gaussian process which is MS continuous but only once MS differentiable. Thus samples from the prior will be quite "rough", although (as noted in section 6.1) the posterior mean, eq. (6.25), is smoother.

The constructions above can be generalized to the regularizer $\int (f^{(m)}(x))^2\, dx$ by replacing $(x - u)_+$ with $(x - u)_+^{m-1}/(m-1)!$ in eq. (6.28) and similarly in eq. (6.32), and setting $\mathbf{h}(x) = (1, x, \ldots, x^{m-1})^\top$.

Thus, we can use a Gaussian process formulation as an alternative to the usual spline fitting procedure. Note that the trade-off parameter λ from eq. (6.26)

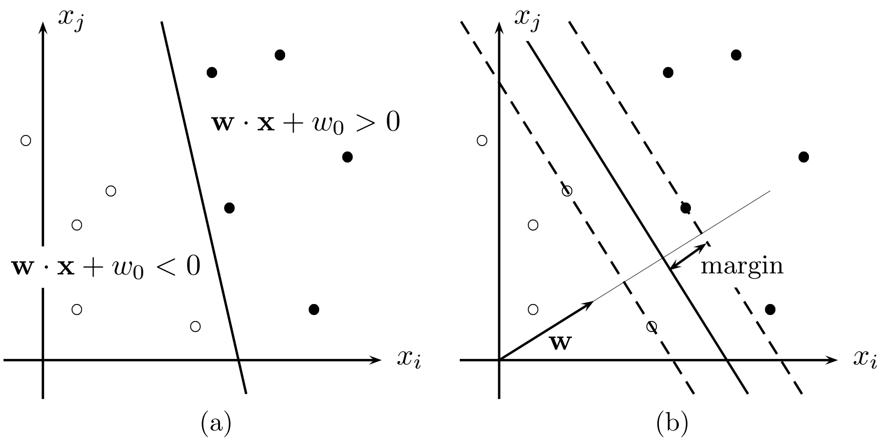

x_j
$\mathbf{w} \cdot \mathbf{x} + w_0 > 0$
$\mathbf{w} \cdot \mathbf{x} + w_0 < 0$
x_i
(a)
x_j
margin
$\mathbf{w}$
x_i
(b)

Figure 6.2: Panel (a) shows a linearly separable binary classification problem, and a separating hyperplane. Panel (b) shows the maximum margin hyperplane.

is now given as the ratio σ_n^2/σ_f^2. The hyperparameters σ_f^2 and σ_n^2 can be set using the techniques from section 5.4.1 by optimizing the marginal likelihood given in eq. (2.45). Kohn and Ansley [1987] give details of an $\mathcal{O}(n)$ algorithm (based on Kalman filtering) for the computation of the spline and the marginal likelihood. In addition to the predictive mean the GP treatment also yields an explicit estimate of the noise level and predictive error bars. Figure 6.1 shows a simple example. Notice that whereas the mean function is a piecewise cubic polynomial, samples from the posterior are not smooth. In contrast, for the squared exponential covariance functions shown in panel (b), both the mean and functions drawn from the posterior are infinitely differentiable.

6.4 Support Vector Machines *

Since the mid 1990's there has been an explosion of interest in kernel machines, and in particular the support vector machine (SVM). The aim of this section is to provide a brief introduction to SVMs and in particular to compare them to Gaussian process predictors. We consider SVMs for classification and regression problems in sections 6.4.1 and 6.4.2 respectively. More comprehensive treatments can be found in Vapnik [1995], Cristianini and Shawe-Taylor [2000] and Schölkopf and Smola [2002].

6.4.1 Support Vector Classification

For support vector classifiers, the key notion that we need to introduce is that of the *maximum margin* hyperplane for a linear classifier. Then by using the "kernel trick" this can be lifted into feature space. We consider first the separable case and then the non-separable case. We conclude this section with a comparison between GP classifiers and SVMs.

The Separable Case

Figure 6.2(a) illustrates the case where the data is linearly separable. For a linear classifier with weight vector $\mathbf{w}$ and offset w_0, let the decision boundary be defined by $\mathbf{w} \cdot \mathbf{x} + w_0 = 0$, and let $\tilde{\mathbf{w}} = (\mathbf{w}, w_0)$. Clearly, there is a whole version space of weight vectors that give rise to the same classification of the training points. The SVM algorithm chooses a particular weight vector, that gives rise to the "maximum margin" of separation.

functional margin

Let the training set be pairs of the form $(\mathbf{x}_i, y_i)$ for $i = 1, \ldots, n$, where $y_i = \pm 1$. For a given weight vector we can compute the quantity $\tilde{\gamma}_i = y_i(\mathbf{w} \cdot \mathbf{x} + w_0)$, which is known as the *functional margin.* Notice that $\tilde{\gamma}_i > 0$ if a training point is correctly classified.

If the equation $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + w_0$ defines a discriminant function (so that the output is $\text{sgn}(f(\mathbf{x}))$), then the hyperplane $c\mathbf{w} \cdot \mathbf{x} + cw_0$ defines the same discriminant function for any $c > 0$. Thus we have the freedom to choose the scaling of $\tilde{\mathbf{w}}$ so that $\min_i \ \tilde{\gamma}_i = 1$, and in this case $\tilde{\mathbf{w}}$ is known as the *canonical* form of the hyperplane.

geometrical margin

The *geometrical margin* is defined as $\gamma_i = \tilde{\gamma}_i / |\mathbf{w}|$. For a training point $\mathbf{x}_i$ that is correctly classified this is simply the distance from $\mathbf{x}_i$ to the hyperplane. To see this, let $c = 1/|\mathbf{w}|$ so that $\hat{\mathbf{w}} = \mathbf{w}/|\mathbf{w}|$ is a unit vector in the direction of $\mathbf{w}$, and $\hat{w}_0$ is the corresponding offset. Then $\hat{\mathbf{w}} \cdot \mathbf{x}$ computes the length of the projection of $\mathbf{x}$ onto the direction orthogonal to the hyperplane and $\hat{\mathbf{w}} \cdot \mathbf{x} + \hat{w}_0$ computes the distance to the hyperplane. For training points that are misclassified the geometrical margin is the negative distance to the hyperplane.

The geometrical margin for a dataset $\mathcal{D}$ is defined as $\gamma_{\mathcal{D}} = \min_i \gamma_i$. Thus for a canonical separating hyperplane the margin is $1/|\mathbf{w}|$. We wish to find the maximum margin hyperplane, i.e. the one that maximizes $\gamma_{\mathcal{D}}$.

optimization problem

By considering canonical hyperplanes, we are thus led to the following optimization problem to determine the maximum margin hyperplane:

$$\begin{aligned} &\text{minimize } \frac{1}{2}|\mathbf{w}|^2 \qquad \text{over } \mathbf{w},\ w_0 \\ &\text{subject to } y_i(\mathbf{w} \cdot \mathbf{x}_i + w_0) \geq 1 \qquad \text{for all } i = 1, \ldots, n. \end{aligned} \tag{6.33}$$

It is clear by considering the geometry that for the maximum margin solution there will be at least one data point in each class for which $y_i(\mathbf{w} \cdot \mathbf{x}_i + w_0) = 1$, see Figure 6.2(b). Let the hyperplanes that pass through these points be denoted H_+ and H_- respectively.

This constrained optimization problem can be set up using Lagrange multipliers, and solved using numerical methods for quadratic programming[4] (QP) problems. The form of the solution is

$$\mathbf{w} \;=\; \sum_i \lambda_i y_i \mathbf{x}_i, \tag{6.34}$$

[4] A quadratic programming problem is an optimization problem where the objective function is quadratic and the constraints are linear in the unknowns.

where the λ_i's are non-negative Lagrange multipliers. Notice that the solution is a linear combination of the $\mathbf{x}_i$'s.

The key feature of equation 6.34 is that λ_i is zero for every $\mathbf{x}_i$ *except* those which lie on the hyperplanes H_+ or H_-; these points are called the *support vectors*. The fact that not all of the training points contribute to the final solution is referred to as the *sparsity* of the solution. The support vectors lie closest to the decision boundary. Note that if all of the other training points were removed (or moved around, but not crossing H_+ or H_-) the same maximum margin hyperplane would be found. The quadratic programming problem for finding the λ_i's is convex, i.e. there are no local minima. Notice the similarity of this to the convexity of the optimization problem for Gaussian process classifiers, as described in section 3.4.

support vectors

To make predictions for a new input $\mathbf{x}_*$ we compute

$$\mathrm{sgn}(\mathbf{w}\cdot\mathbf{x}_* + w_0) \;=\; \mathrm{sgn}\Big(\sum_{i=1}^{n}\lambda_i y_i(\mathbf{x}_i\cdot\mathbf{x}_*) + w_0\Big). \tag{6.35}$$

In the QP problem and in eq. (6.35) the training points $\{\mathbf{x}_i\}$ and the test point $\mathbf{x}_*$ enter the computations only in terms of inner products. Thus by using the kernel trick we can replace occurrences of the inner product by the kernel to obtain the equivalent result in feature space.

kernel trick

The Non-Separable Case

For linear classifiers in the original $\mathbf{x}$ space there will be some datasets that are not linearly separable. One way to generalize the SVM problem in this case is to allow violations of the constraint $y_i(\mathbf{w}\cdot\mathbf{x}_i + w_0) \geq 1$ but to impose a penalty when this occurs. This leads to the *soft margin* support vector machine problem, the minimization of

soft margin

$$\frac{1}{2}|\mathbf{w}|^2 + C\sum_{i=1}^{n}(1 - y_i f_i)_+ \tag{6.36}$$

with respect to $\mathbf{w}$ and w_0, where $f_i = f(\mathbf{x}_i) = \mathbf{w}\cdot\mathbf{x}_i + w_0$ and $(z)_+ = z$ if $z > 0$ and 0 otherwise. Here $C > 0$ is a parameter that specifies the relative importance of the two terms. This convex optimization problem can again be solved using QP methods and yields a solution of the form given in eq. (6.34). In this case the support vectors (those with $\lambda_i \neq 0$) are not only those data points which lie on the separating hyperplanes, but also those that incur penalties. This can occur in two ways (i) the data point falls in between H_+ and H_- but on the correct side of the decision surface, or (ii) the data point falls on the wrong side of the decision surface.

In a feature space of dimension N, if $N > n$ then there will always be separating hyperplane. However, this hyperplane may not give rise to good generalization performance, especially if some of the labels are incorrect, and thus the soft margin SVM formulation is often used in practice.

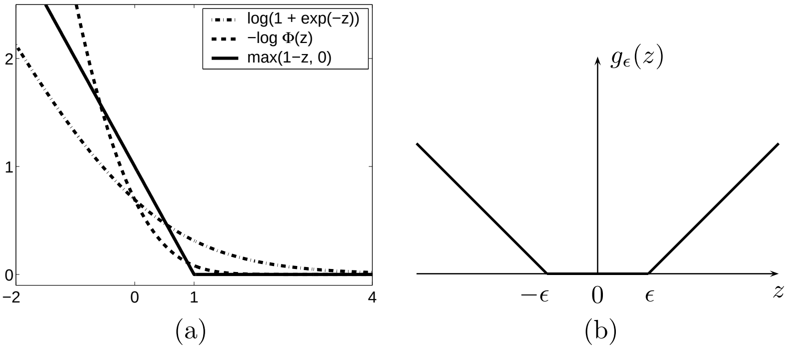


Figure 6.3: (a) A comparison of the hinge error, g_λ and g_Φ. (b) The ϵ-insensitive error function used in SVR.

For both the hard and soft margin SVM QP problems a wide variety of algorithms have been developed for their solution; see Schölkopf and Smola [2002, ch. 10] for details. Basic interior point methods involve inversions of $n \times n$ matrices and thus scale as $\mathcal{O}(n^3)$, as with Gaussian process prediction. However, there are other algorithms, such as the sequential minimal optimization (SMO) algorithm due to Platt [1999], which often have better scaling in practice.

Above we have described SVMs for the two-class (binary) classification problem. There are many ways of generalizing SVMs to the multi-class problem, see Schölkopf and Smola [2002, sec. 7.6] for further details.

Comparing Support Vector and Gaussian Process Classifiers

For the soft margin classifier we obtain a solution of the form $\mathbf{w} = \sum_i \alpha_i \mathbf{x}_i$ (with $\alpha_i = \lambda_i y_i$) and thus $|\mathbf{w}|^2 = \sum_{i,j} \alpha_i \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j)$. Kernelizing this we obtain $|\mathbf{w}|^2 = \boldsymbol{\alpha}^\top K \boldsymbol{\alpha} = \mathbf{f}^\top K^{-1} \mathbf{f}$, as[5] $K\boldsymbol{\alpha} = \mathbf{f}$. Thus the soft margin objective function can be written as

$$\frac{1}{2}\mathbf{f}K^{-1}\mathbf{f} + C\sum_{i=1}^{n}(1 - y_i f_i)_+. \qquad (6.37)$$

For the binary GP classifier, to obtain the MAP value $\hat{\mathbf{f}}$ of $p(\mathbf{f}|\mathbf{y})$ we minimize the quantity

$$\frac{1}{2}\mathbf{f}K^{-1}\mathbf{f} - \sum_{i=1}^{n}\log p(y_i|f_i), \qquad (6.38)$$

cf. eq. (3.12). (The final two terms in eq. (3.12) are constant if the kernel is fixed.)

For log-concave likelihoods (such as those derived from the logistic or probit response functions) there is a strong similarity between the two optimization problems in that they are both convex. Let $g_\lambda(z) \triangleq \log(1 + e^{-z})$, $g_\Phi =$

[5] Here the offset w_0 has been absorbed into the kernel so it is not an explicit extra parameter.

$-\log\Phi(z)$, and $g_{\text{hinge}}(z) \triangleq (1-z)_+$ where $z = y_i f_i$. We refer to g_{hinge} as the *hinge* error function, due to its shape. As shown in Figure 6.3(a) all three data fit terms are monotonically decreasing functions of z. All three functions tend to infinity as $z \to -\infty$ and decay to zero as $z \to \infty$. The key difference is that the hinge function takes on the value 0 for $z \geq 1$, while the other two just decay slowly. It is this flat part of the hinge function that gives rise to the sparsity of the SVM solution.

hinge error function

Thus there is a close correspondence between the MAP solution of the GP classifier and the SVM solution. Can this correspondence be made closer by considering the hinge function as a negative log likelihood? The answer to this is no [Seeger, 2000, Sollich, 2002]. If $Cg_{\text{hinge}}(z)$ defined a negative log likelihood, then $\exp(-Cg_{\text{hinge}}(f)) + \exp(-Cg_{\text{hinge}}(-f))$ should be a constant independent of f, but this is not the case. To see this, consider the quantity

$$\nu(f;C) = \kappa(C)[\exp(-C(1-f)_+) + \exp(-C(1+f)_+)]. \tag{6.39}$$

$\kappa(C)$ cannot be chosen so as to make $\nu(f;C) = 1$ independent of the value of f for $C > 0$. By comparison, for the logistic and probit likelihoods the analogous expression is equal to 1. Sollich [2002] suggests choosing $\kappa(C) = 1/(1+\exp(-2C))$ which ensures that $\nu(f,C) \leq 1$ (with equality only when $f = \pm 1$). He also gives an ingenious interpretation (involving a "don't know" class to soak up the unassigned probability mass) that does yield the SVM solution as the MAP solution to a certain Bayesian problem, although we find this construction rather contrived. Exercise 6.7.2 invites you to plot $\nu(f;C)$ as a function of f for various values of C.

One attraction of the GP classifier is that it produces an output with a clear probabilistic interpretation, a prediction for $p(y=+1|\mathbf{x})$. One can try to interpret the function value $f(\mathbf{x})$ output by the SVM probabilistically, and Platt [2000] suggested that probabilistic predictions can be generated from the SVM by computing $\sigma(af(\mathbf{x})+b)$ for some constants a, b that are fitted using some "unbiased version" of the training set (e.g. using cross-validation). One disadvantage of this rather *ad hoc* procedure is that unlike the GP classifiers it does not take into account the predictive variance of $f(\mathbf{x})$ (cf. eq. (3.25)). Seeger [2003, sec. 4.7.2] shows that better error-reject curves can be obtained on an experiment using the MNIST digit classification problem when the effect of this uncertainty is taken into account.

6.4.2 Support Vector Regression

The SVM was originally introduced for the classification problem, then extended to deal with the regression case. The key concept is that of the ϵ-insensitive error function. This is defined as

$$g_\epsilon(z) = \begin{cases} |z| - \epsilon & \text{if } |z| \geq \epsilon, \\ 0 & \text{otherwise.} \end{cases} \tag{6.40}$$

This function is plotted in Figure 6.3(b). As in eq. (6.21) we can interpret $\exp(-g_\epsilon(z))$ as a likelihood model for the regression residuals (c.f. the squared error function corresponding to a Gaussian model). However, we note that this is quite an unusual choice of model for the distribution of residuals and is basically motivated by the desire to obtain a sparse solution (see below) as in support vector classifier. If $\epsilon = 0$ then the error model is a Laplacian distribution, which corresponds to least absolute values regression (Edgeworth [1887], cited in Rousseeuw [1984]); this is a heavier-tailed distribution than the Gaussian and provides some protection against outliers. Girosi [1991] showed that the Laplacian distribution can be viewed as a continuous mixture of zero-mean Gaussians with a certain distribution over their variances. Pontil et al. [1998] extended this result by allowing the means to uniformly shift in $[-\epsilon, \epsilon]$ in order to obtain a probabilistic model corresponding to the ϵ-insensitive error function. See also section 9.3 for work on robustification of the GP regression problem.

For the linear regression case with an ϵ-insensitive error function and a Gaussian prior on $\mathbf{w}$, the MAP value of $\mathbf{w}$ is obtained by minimizing

$$\frac{1}{2}|\mathbf{w}|^2 + C\sum_{i=1}^{n} g_\epsilon(y_i - f_i) \tag{6.41}$$

w.r.t. $\mathbf{w}$. The solution[6] is $f(\mathbf{x}_*) = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i \cdot \mathbf{x}_*$ where the coefficients $\boldsymbol{\alpha}$ are obtained from a QP problem. The problem can also be kernelized to give the solution $f(\mathbf{x}_*) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x}_*)$.

As for support vector classification, many of the coefficients α_i are zero. The data points which lie inside the ϵ-"tube" have $\alpha_i = 0$, while those on the edge or outside have non-zero α_i.

* 6.5 Least-Squares Classification

In chapter 3 we have argued that the use of logistic or probit likelihoods provides the natural route to develop a GP classifier, and that it is attractive in that the outputs can be interpreted probabilistically. However, there is an even simpler approach which treats classification as a regression problem.

Our starting point is binary classification using the linear predictor $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$. This is trained using linear regression with a target y_+ for patterns that have label $+1$, and target y_- for patterns that have label -1. (Targets y_+, y_- give slightly more flexibility than just using targets of ± 1.) As shown in Duda and Hart [1973, section 5.8], choosing y_+, y_- appropriately allows us to obtain the same solution as Fisher's linear discriminant using the decision criterion $f(\mathbf{x}) \gtrless 0$. Also, they show that using targets $y_+ = +1$, $y_- = -1$ with the least-squares error function gives a minimum squared-error approximation to the Bayes discriminant function $p(\mathcal{C}_+|\mathbf{x}) - p(\mathcal{C}_-|\mathbf{x})$ as $n \to \infty$. Following Rifkin and Klautau [2004] we call such methods least-squares classification (LSC). Note that under a probabilistic interpretation the squared-error criterion is rather an

[6] Here we have assumed that the constant 1 is included in the input vector $\mathbf{x}$.

odd choice as it implies a Gaussian noise model, yet only two values of the target (y_+ and y_-) are observed.

It is natural to extend the least-squares classifier using the kernel trick. This has been suggested by a number of authors including Poggio and Girosi [1990] and Suykens and Vanderwalle [1999]. Experimental results reported in Rifkin and Klautau [2004] indicate that performance comparable to SVMs can be obtained using kernel LSC (or as they call it the regularized least-squares classifier, RLSC).

Consider a single random variable y which takes on the value $+1$ with probability p and value -1 with probability $1-p$. Then the value of f which minimizes the squared error function $E = p(f-1)^2 + (1-p)(f+1)^2$ is $\hat{f} = 2p - 1$, which is a linear rescaling of p to the interval $[-1, 1]$. (Equivalently if the targets are 1 and 0, we obtain $\hat{f} = p$.) Hence we observe that LSC will estimate p correctly in the large data limit. If we now consider not just a single random variable, but wish to estimate $p(\mathcal{C}_+|\mathbf{x})$ (or a linear rescaling of it), then as long as the approximating function $f(\mathbf{x})$ is sufficiently flexible, we would expect that in the limit $n \to \infty$ it would converge to $p(\mathcal{C}_+|\mathbf{x})$. (For more technical detail on this issue, see section 7.2.1 on consistency.) Hence LSC is quite a sensible procedure for classification, although note that there is no guarantee that $f(\mathbf{x})$ will be constrained to lie in the interval $[y_-, y_+]$. If we wish to guarantee a probabilistic interpretation, we could "squash" the predictions through a sigmoid, as suggested for SVMs by Platt [2000] and described on page 145.

When generalizing from the binary to multi-class situation there is some freedom as to how to set the problem up. Schölkopf and Smola [2002, sec. 7.6] identify four methods, namely one-versus-rest (where C binary classifiers are trained to classify each class against all the rest), all pairs (where $C(C-1)/2$ binary classifiers are trained), error-correcting output coding (where each class is assigned a binary codeword, and binary classifiers are trained on each bit separately), and multi-class objective functions (where the aim is to train C classifiers simultaneously rather than creating a number of binary classification problems). One also needs to specify how the outputs of the various classifiers that are trained are combined so as to produce an overall answer. For the one-versus-rest[7] method one simple criterion is to choose the classifier which produces the most positive output. Rifkin and Klautau [2004] performed extensive experiments and came to the conclusion that the one-versus-rest scheme using either SVMs or RLSC is as accurate as any other method overall, and has the merit of being conceptually simple and straightforward to implement.

6.5.1 Probabilistic Least-Squares Classification

The LSC algorithm discussed above is attractive from a computational point of view, but to guarantee a valid probabilistic interpretation one may need to use a separate post-processing stage to "squash" the predictions through a sigmoid. However, it is not so easy to enforce a probabilistic interpretation

[7] This method is also sometimes called one-versus-all.

during the training stage. One possible solution is to combine the ideas of training using leave-one-out cross-validation, covered in section 5.4.2, with the use of a (parameterized) sigmoid function, as in Platt [2000]. We will call this method the probabilistic least-squares classifier (PLSC).

In section 5.4.2 we saw how to compute the Gaussian leave-one-out (LOO) predictive probabilities, and that training of hyperparameters can be based on the sum of the log LOO probabilities. Using this idea, we express the LOO probability by squashing a linear function of the Gaussian predictive probability through a cumulative Gaussian

$$\begin{aligned} p(y_i|X, \mathbf{y}_{-i}, \boldsymbol{\theta}) \;&=\; \int \Phi\big(y_i(\alpha f_i + \beta)\big)\mathcal{N}(f_i|\mu_i, \sigma_i^2)\, df_i \\ &=\; \Phi\Big(\frac{y_i(\alpha\mu_i + \beta)}{\sqrt{1+\alpha^2\sigma_i^2}}\Big), \end{aligned} \tag{6.42}$$

where the integral is given in eq. (3.82) and the leave-one-out predictive mean μ_i and variance σ_i^2 are given in eq. (5.12). The objective function is the sum of the log LOO probabilities, eq. (5.11) which can be used to set the hyperparameters as well as the two additional parameters of the linear transformation, α and β in eq. (6.42). Introducing the likelihood in eq. (6.42) into the objective eq. (5.11) and taking derivatives we obtain

$$\begin{aligned} \frac{\partial L_{\text{LOO}}}{\partial \theta_j} \;&=\; \sum_{i=1}^{n} \frac{\partial \log p(y_i|X, \mathbf{y}, \boldsymbol{\theta})}{\partial \mu_i}\frac{\partial \mu_i}{\partial \theta_j} + \frac{\partial \log p(y_i|X, \mathbf{y}, \boldsymbol{\theta})}{\partial \sigma_i^2}\frac{\partial \sigma_i^2}{\partial \theta_j} \\ &=\; \sum_{i=1}^{n} \frac{\mathcal{N}(r_i)}{\Phi(y_i r_i)} \frac{y_i\alpha}{\sqrt{1+\alpha^2\sigma_i^2}}\Big(\frac{\partial \mu_i}{\partial \theta_j} - \frac{\alpha(\alpha\mu_i+\beta)}{2(1+\alpha^2\sigma_i^2)}\frac{\partial \sigma_i^2}{\partial \theta_j}\Big), \end{aligned} \tag{6.43}$$

where $r_i = (\alpha\mu_i + \beta)/\sqrt{1+\alpha^2\sigma_i^2}$ and the partial derivatives of the Gaussian LOO parameters $\partial\mu_i/\partial\theta_j$ and $\partial\sigma_i^2/\partial\theta_j$ are given in eq. (5.13). Finally, for the linear transformation parameters we have

$$\begin{aligned} \frac{\partial L_{\text{LOO}}}{\partial \alpha} \;&=\; \sum_{i=1}^{n} \frac{\mathcal{N}(r_i)}{\Phi(y_i r_i)} \frac{y_i}{\sqrt{1+\alpha^2\sigma_i^2}} \frac{\mu_i - \beta\alpha\sigma_i^2}{1+\alpha^2\sigma_i^2}, \\ \frac{\partial L_{\text{LOO}}}{\partial \beta} \;&=\; \sum_{i=1}^{n} \frac{\mathcal{N}(r_i)}{\Phi(y_i r_i)} \frac{y_i}{\sqrt{1+\alpha^2\sigma_i^2}}. \end{aligned} \tag{6.44}$$

These partial derivatives can be used to train the parameters of the GP. There are several options on how to do predictions, but the most natural would seem to be to compute predictive mean and variance and squash it through the sigmoid, parallelling eq. (6.42). Applying this model to the USPS 3s vs. 5s binary classification task discussed in section 3.7.3, we get a test set error rate of $12/773 = 0.0155\%$, which compares favourably with the results reported for other methods in Figure 3.10. However, the test set information is only 0.77 bits,[8] which is very poor.

[8] The test information is dominated by a single test case, which is predicted confidently to belong to the wrong class. Visual inspection of the digit reveals that indeed it looks as though the testset label is wrong for this case. This observation highlights the danger of not explicitly to allowing for data mislabelling in the model for this kind of data.

6.6 Relevance Vector Machines *

Although usually not presented as such, the relevance vector machine (RVM) introduced by Tipping [2001] is actually a special case of a Gaussian process. The covariance function has the form

$$k(\mathbf{x}, \mathbf{x}') \;=\; \sum_{j=1}^{N} \frac{1}{\alpha_j} \phi_j(\mathbf{x}) \phi_j(\mathbf{x}'), \tag{6.45}$$

where α_j are hyperparameters and the N *basis functions* $\phi_j(\mathbf{x})$ are usually, but not necessarily taken to be Gaussian-shaped basis functions centered on each of the n training data points

$$\phi_j(\mathbf{x}) \;=\; \exp\Big(-\frac{|\mathbf{x}-\mathbf{x}_j|^2}{2\ell^2}\Big), \tag{6.46}$$

where ℓ is a length-scale hyperparameter controlling the width of the basis function. Notice that this is simply the construction for the covariance function corresponding to an N-dimensional set of basis functions given in section 2.1.2, with $\Sigma_p = \mathrm{diag}(\alpha_1^{-1}, \ldots, \alpha_N^{-1})$.

The covariance function in eq. (6.45) has two interesting properties: firstly, it is clear that the feature space corresponding to the covariance function is finite dimensional, i.e. the covariance function is *degenerate*, and secondly the covariance function has the odd property that it *depends on the training data*. This dependency means that the prior over functions depends on the data, a property which is at odds with a strict Bayesian interpretation. Although the usual treatment of the model is still possible, this dependency of the prior on the data may lead to some surprising effects, as discussed below.

Training the RVM is analogous to other GP models: optimize the marginal likelihood w.r.t. the hyperparameters. This optimization often leads to a significant number of the α_j hyperparameters tending towards infinity, effectively removing, or *pruning*, the corresponding basis function from the covariance function in eq. (6.45). The basic idea is that basis functions that are not significantly contributing to explaining the data should be removed, resulting in a *sparse* model. The basis functions that survive are called *relevance vectors*. Empirically it is often observed that the number of relevance vectors is smaller than the number of support vectors on the same problem [Tipping, 2001].

relevance vectors

The original RVM algorithm [Tipping, 2001] was not able to exploit the sparsity very effectively during model fitting as it was initialized with all of the α_is set to finite values, meaning that all of the basis functions contributed to the model. However, careful analysis of the RVM marginal likelihood by Faul and Tipping [2002] showed that one can carry out optimization w.r.t. a single α_i analytically. This has led to the accelerated training algorithm described in Tipping and Faul [2003] which starts with an empty model (i.e. all α_is set to infinity) and adds basis functions sequentially. As the number of relevance vectors is (usually much) less than the number of training cases it will often be much faster to train and make predictions using a RVM than a non-sparse

GP. Also note that the basis functions can include additional hyperparameters, e.g. one could use an automatic relevance determination (ARD) form of basis function by using different length-scales on different dimensions in eq. (6.46). These additional hyperparameters could also be set by optimizing the marginal likelihood.

The use of a degenerate covariance function which depends on the data has some undesirable effects. Imagine a test point, $\mathbf{x}_*$, which lies far away from the relevance vectors. At $\mathbf{x}_*$ all basis functions will have values close to zero, and since no basis function can give any appreciable signal, the predictive distribution will be a Gaussian with a mean close to zero and *variance close to zero* (or to the inferred noise level). This behaviour is undesirable, and could lead to dangerously false conclusions. If the $\mathbf{x}_*$ is far from the relevance vectors, then the model shouldn't be able to draw strong conclusions about the output (we are extrapolating), but the predictive uncertainty becomes very small—this is the opposite behaviour of what we would expect from a reasonable model. Here, we have argued that for localized basis functions, the RVM has undesirable properties, but as argued in Rasmussen and Quiñonero-Candela [2005] it is actually the degeneracy of the covariance function which is the core of the problem. Although the work of Rasmussen and Quiñonero-Candela [2005] goes some way towards fixing the problem, there is an inherent conflict: degeneracy of the covariance function is good for computational reasons, but bad for modelling reasons.

6.7 Exercises

1. We motivate the fact that the RKHS norm does not depend on the density $p(\mathbf{x})$ using a finite-dimensional analogue. Consider the n-dimensional vector $\mathbf{f}$, and let the $n \times n$ matrix Φ be comprised of non-colinear columns $\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_n$. Then $\mathbf{f}$ can be expressed as a linear combination of these basis vectors $\mathbf{f} = \sum_{i=1}^n c_i \boldsymbol{\phi}_i = \Phi \mathbf{c}$ for some coefficients $\{c_i\}$. Let the $\boldsymbol{\phi}$s be eigenvectors of the covariance matrix K w.r.t. a diagonal matrix P with non-negative entries, so that $KP\Phi = \Phi\Lambda$, where Λ is a diagonal matrix containing the eigenvalues. Note that $\Phi^\top P \Phi = I_n$. Show that $\sum_{i=1}^n c_i^2/\lambda_i = \mathbf{c}^\top \Lambda^{-1} \mathbf{c} = \mathbf{f}^\top K^{-1} \mathbf{f}$, and thus observe that $\mathbf{f}^\top K^{-1} \mathbf{f}$ can be expressed as $\mathbf{c}^\top \Lambda^{-1} \mathbf{c}$ for any valid P and corresponding Φ. Hint: you may find it useful to set $\tilde{\Phi} = P^{1/2}\Phi$, $\tilde{K} = P^{1/2} K P^{1/2}$ etc.

2. Plot eq. (6.39) as a function of f for different values of C. Show that there is no value of C and $\kappa(C)$ which makes $\nu(f; C)$ equal to 1 for all values of f. Try setting $\kappa(C) = 1/(1 + \exp(-2C))$ as suggested in Sollich [2002] and observe what effect this has.

3. Show that the predictive mean for the spline covariance GP in eq. (6.29) is a linear function of x_* when x_* is located either to the left or to the right of all training points. Hint: consider the eigenvectors corresponding to the two largest eigenvalues of the training set covariance matrix from eq. (2.40) in the vague limit.