C. E. Rasmussen & C. K. I. Williams, Gaussian Processes for Machine Learning, the MIT Press, 2006, ISBN 026218253X.  www.GaussianProcess.org/gpml

# Chapter 9

# Further Issues and Conclusions

In the previous chapters of the book we have concentrated on giving a solid grounding in the use of GPs for regression and classification problems, including model selection issues, approximation methods for large datasets, and connections to related models. In this chapter we provide some short descriptions of other issues relating to Gaussian process prediction, with pointers to the literature for further reading.

So far we have mainly discussed the case when the output target $y$ is a single label, but in section 9.1 we describe how to deal with the case that there are multiple output targets. Similarly, for the regression problem we have focussed on i.i.d. Gaussian noise; in section 9.2 we relax this condition to allow the noise process to have correlations. The classification problem is characterized by a non-Gaussian likelihood function; however, there are other non-Gaussian likelihoods of interest, as described in section 9.3.

We may not only have observations of function values, by also on derivatives of the target function. In section 9.4 we discuss how to make use of this information in the GPR framework. Also it may happen that there is noise on the observation of the input variable $\mathbf{x}$; in section 9.5 we explain how this can be handled. In section 9.6 we mention how more flexible models can be obtained using mixtures of Gaussian process models.

As well as carrying out prediction for test inputs, one might also wish to try to find the global optimum of a function within some compact set. Approaches based on Gaussian processes for this problem are described in section 9.7. The use of Gaussian processes to evaluate integrals is covered in section 9.8.

By using a scale mixture of Gaussians construction one can obtain a multivariate Student's $t$ distribution. This construction can be extended to give a Student's $t$ process, as explained in section 9.9. One key aspect of the Bayesian framework relates to the incorporation of prior knowledge into the problem

formulation. In some applications we not only have the dataset $\mathcal{D}$ but also additional information. For example, for an optical character recognition problem we know that translating the input pattern by one pixel will not change the label of the pattern. Approaches for incorporating this knowledge are discussed in section 9.10.

In this book we have concentrated on supervised learning problems. However, GPs can be used as components in unsupervised learning models, as described in section 9.11. Finally, we close with some conclusions and an outlook to the future in section 9.12.

## 9.1 Multiple Outputs

Throughout this book we have concentrated on the problem of predicting a single output variable $y$ from an input $\mathbf{x}$. However, it can happen that one may wish to predict multiple output variables (or channels) simultaneously. For example in the robot inverse dynamics problem described in section 2.5 there are really seven torques to be predicted. A simple approach is to model each output variable as independent from the others and treat them separately. However, this may lose information and be suboptimal.

One way in which correlation can occur is through a correlated noise process. Even if the output channels are *a priori* independent, if the noise process is correlated then this will induce correlations in the posterior processes. Such a situation is easily handled in the GP framework by considering the joint, block-diagonal, prior over the function values of each channel.

Another way that correlation of multiple channels can occur is if the prior already has this structure. For example in geostatistical situations there may be correlations between the abundances of different ores, e.g. silver and lead. This situation requires that the covariance function models not only the correlation structure of each channel, but also the cross-correlations between channels. Some work on this topic can be found in the geostatistics literature under the name of cokriging, see e.g. Cressie [1993, sec. 3.2.3]. One way to induce correlations between a number of output channels is to obtain them as linear combinations of a number of latent channels, as described in Teh et al. [2005]; see also Micchelli and Pontil [2005]. A related approach is taken by Boyle and Frean [2005] who introduce correlations between two processes by deriving them as different convolutions of the same underlying white noise process.

cokriging

## 9.2 Noise Models with Dependencies

The noise models used so far have almost exclusively assumed Gaussianity and independence. Non-Gaussian likelihoods are mentioned in section 9.3 below. Inside the family of Gaussian noise models, it is not difficult to model dependencies. This may be particularly useful in models involving time. We simply add terms to the noise covariance function with the desired structure, including

coloured noise

hyperparameters. In fact, we already used this approach for the atmospheric carbon dioxide modelling task in section 5.4.3. Also, Murray-Smith and Girard [2001] have used an autoregressive moving-average (ARMA) noise model (see also eq. (B.51)) in a GP regression task. ARMA

## 9.3 Non-Gaussian Likelihoods

Our main focus has been on regression with Gaussian noise, and classification using the logistic or probit response functions. However, Gaussian processes can be used as priors with other likelihood functions. For example, Diggle et al. [1998] were concerned with modelling count data measured geographically using a Poisson likelihood with a spatially varying rate. They achieved this by placing a GP prior over the log Poisson rate.

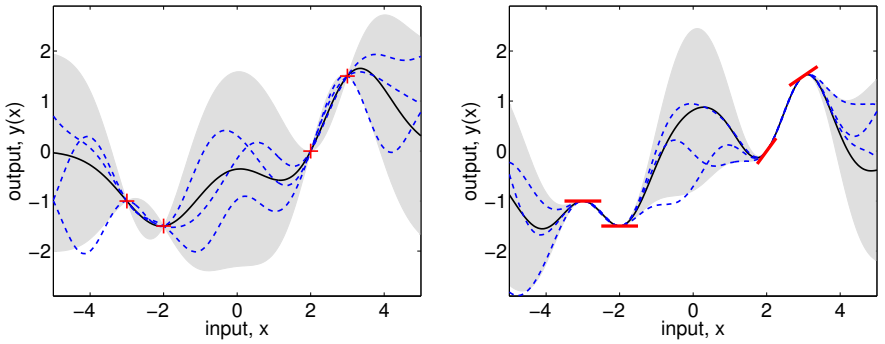
Goldberg et al. [1998] stayed with a Gaussian noise model, but introduced heteroscedasticity, i.e. allowing the noise variance to be a function of $\mathbf{x}$. This was achieved by placing a GP prior on the log variance function. Neal [1997] robustified GP regression by using a Student's $t$-distributed noise model rather than Gaussian noise. Chu and Ghahramani [2005] have described how to use GPs for the ordinal regression problem, where one is given ranked preference information as the target data.

## 9.4 Derivative Observations

Since differentiation is a linear operator, the derivative of a Gaussian process is another Gaussian process. Thus we can use GPs to make predictions about derivatives, and also to make inference based on derivative information. In general, we can make inference based on the joint Gaussian distribution of function values and partial derivatives. A covariance function $k(\cdot,\cdot)$ on function values *implies* the following (mixed) covariance between function values and partial derivatives, and between partial derivatives

$$\operatorname{cov}\big(f_i, \frac{\partial f_j}{\partial x_{dj}}\big) = \frac{\partial k(\mathbf{x}_i,\mathbf{x}_j)}{\partial x_{dj}}, \qquad \operatorname{cov}\big(\frac{\partial f_i}{\partial x_{di}}, \frac{\partial f_j}{\partial x_{ej}}\big) = \frac{\partial^2 k(\mathbf{x}_i,\mathbf{x}_j)}{\partial x_{di}\partial x_{ej}}, \tag{9.1}$$

see e.g. Papoulis [1991, ch. 10] or Adler [1981, sec. 2.2]. With $n$ datapoints in $D$ dimensions, the complete joint distribution of $f$ and its $D$ partial derivatives involves $n(D+1)$ quantities, but in a typical application we may only have access to or interest in a subset of these; we simply remove the rows and columns from the joint matrix which are not needed. Observed function values and derivatives may often have different noise levels, which are incorporated by adding a diagonal contribution with differing hyperparameters. Inference and predictions are done as usual. This approach was used in the context of learning in dynamical systems by Solak et al. [2003]. In Figure 9.1 the posterior process with and without derivative observations are compared. Noise-free derivatives may be a useful way to enforce known constraints in a modelling problem.




(a) (b)

Figure 9.1: In panel (a) we show four data points in a one dimensional noise-free regression problem, together with three functions sampled from the posterior and the 95% confidence region in light grey. In panel (b) the same observations have been augmented by noise-free derivative information, indicated by small tangent segments at the data points. The covariance function is the squared exponential with unit process variance and unit length-scale.

## 9.5 Prediction with Uncertain Inputs

It can happen that the input values to a prediction problem can be uncertain. For example, for a discrete time series one can perform multi-step-ahead predictions by iterating one-step-ahead predictions. However, if the one-step-ahead predictions include uncertainty, then it is necessary to propagate this uncertainty forward to get the proper multi-step-ahead predictions. One simple approach is to use sampling methods. Alternatively, it may be possible to use analytical approaches. Girard et al. [2003] showed that it is possible to compute the mean and variance of the output analytically when using the SE covariance function and Gaussian input noise.

More generally, the problem of regression with uncertain inputs has been studied in the statistics literature under the name of errors-in-variables regression. See Dellaportas and Stephens [1995] for a Bayesian treatment of the problem and pointers to the literature.

## 9.6 Mixtures of Gaussian Processes

In chapter 4 we have seen many ideas for making the covariance functions more flexible. Another route is to use a mixture of different Gaussian process models, each one used in some local region of input space. This kind of model is generally known as a mixture of experts model and is due to Jacobs et al. [1991]. In addition to the local expert models, the model has a *manager* that (probabilistically) assigns points to the experts. Rasmussen and Ghahramani [2002] used Gaussian process models as local experts, and based their manager on another type of stochastic process: the Dirichlet process. Inference in this model required MCMC methods.

C. E. Rasmussen & C. K. I. Williams, Gaussian Processes for Machine Learning, the MIT Press, 2006, ISBN 026218253X.  www.GaussianProcess.org/gpml



## 9.7 Global Optimization

Often one is faced with the problem of being able to evaluate a continuous function $g(\mathbf{x})$, and wishing to find the global optimum (maximum or minimum) of this function within some compact set $A \subset \mathbb{R}^D$. There is a very large literature on the problem of global optimization; see Neumaier [2005] for a useful overview.

Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, g(\mathbf{x}_i))|i=1,\ldots,n\}$, one appealing approach is to fit a GP regression model to this data. This will give a mean prediction and predictive variance for every $\mathbf{x} \in A$. Jones [2001] examines a number of criteria that have been suggested for where to make the next function evaluation based on the predictive mean and variance. One issue with this approach is that one may need to *search* to find the optimum of the criterion, which may itself be multimodal optimization problem. However, if evaluations of $g$ are expensive or time-consuming, it can make sense to work hard on this new optimization problem.

For historical references and further work in this area see Jones [2001] and Ritter [2000, sec. VIII.2].

## 9.8 Evaluation of Integrals

Another interesting and unusual application of Gaussian processes is for the evaluation of the integrals of a deterministic function $f$. One evaluates the function at a number of locations, and then one can use a Gaussian process as a posterior over functions. This posterior over functions induces a posterior over the value of the integral (since each possible function from the posterior would give rise to a particular value of the integral). For some covariance functions (e.g. the squared exponential), one can compute the expectation and variance of the value of the integral analytically. It is perhaps unusual to think of the value of the integral as being random (as it does have one particular deterministic value), but it is perfectly in line of Bayesian thinking that you treat all kinds of uncertainty using probabilities. This idea was proposed under the name of Bayes-Hermite quadrature by O'Hagan [1991], and later under the name of Bayesian Monte Carlo in Rasmussen and Ghahramani [2003].

combining GPs with MCMC

Another approach is related to the ideas of global optimization in the section 9.7 above. One can use a GP model of a function to aid an MCMC sampling procedure, which may be advantageous if the function of interest is computationally expensive to evaluate. Rasmussen [2003] combines Hybrid Monte Carlo with a GP model of the log of the integrand, and also uses derivatives of the function (discussed in section 9.4) to get an accurate model of the integrand with very few evaluations.

## 9.9 Student's $t$ Process

scale mixture

A Student's $t$ process can be obtained by applying the *scale mixture* of Gaussians construction of a Student's $t$ distribution to a Gaussian process [O'Hagan, 1991, O'Hagan et al., 1999]. We divide the covariances by the scalar $\tau$ and put a gamma distribution on $\tau$ with shape $\alpha$ and mean $\beta$ so that

$$\tilde{k}(\mathbf{x},\mathbf{x}') \;=\; \tau^{-1}k(\mathbf{x},\mathbf{x}'), \qquad p(\tau|\alpha,\beta) \;=\; \frac{\alpha^\alpha}{\beta^\alpha\Gamma(\alpha)}\tau^{\alpha-1}\exp\Big(-\frac{\tau\alpha}{\beta}\Big), \quad (9.2)$$

where $k$ is any valid covariance function. Now the joint prior distribution of any finite number $n$ of function values $\mathbf{y}$ becomes

$$\begin{aligned} p(\mathbf{y}|\alpha,\beta,\boldsymbol{\theta}) \;&=\; \int \mathcal{N}(\mathbf{y}|\mathbf{0},\tau^{-1}K_y)p(\tau|\alpha,\beta)d\tau \\ &=\; \frac{\Gamma(\alpha+n/2)(2\pi\alpha)^{-n/2}}{\Gamma(\alpha)|\beta^{-1}K_y|^{-1/2}}\Big(1+\frac{\beta\mathbf{y}^\top K_y^{-1}\mathbf{y}}{2\alpha}\Big)^{-(\alpha+n/2)}, \end{aligned} \quad (9.3)$$

which is recognized as the zero mean, multivariate Student's $t$ distribution with $2\alpha$ degrees of freedom: $p(\mathbf{y}|\alpha,\beta,\boldsymbol{\theta}) = \mathcal{T}(0,\beta^{-1}K_y,2\alpha)$. We could state a definition analogous to definition 2.1 on page 13 for the Gaussian process, and write

$$f \;\sim\; \mathcal{TP}(0,\beta^{-1}K,2\alpha), \quad (9.4)$$

cf. eq. (2.14). The marginal likelihood can be directly evaluated using eq. (9.3), and training can be achieved using the methods discussed in chapter 5 regarding $\alpha$ and $\beta$ as hyperparameters. The predictive distribution for test cases are also $t$ distributions, the derivation of which is left as an exercise below.

Notice that the above construction is clear for noise-free processes, but that the interpretation becomes more complicated if the covariance function $k(\mathbf{x},\mathbf{x}')$

noise entanglement

contains a noise contribution. The noise and signal get entangled by the common factor $\tau$, and the observations can no longer be written as the sum of independent signal and noise contributions. Allowing for independent noise contributions removes analytic tractability, which may reduce the usefulness of the $t$ process.

Exercise Using the scale mixture representation from eq. (9.3) derive the posterior predictive distribution for a Student's $t$ process.

Exercise Consider the generating process implied by eq. (9.2), and write a program to draw functions at random. Characterize the difference between the Student's $t$ process and the corresponding Gaussian process (obtained in the limit $\alpha \to \infty$), and explain why the $t$ process is perhaps not as exciting as one might have hoped.

## 9.10 Invariances

It can happen that the input is apparently in vector form but in fact it has additional structure. A good example is a pixelated image, where the 2-d array

of pixels can be arranged into a vector (e.g. in raster-scan order). Imagine that the image is of a handwritten digit; then we know that if the image is translated by one pixel it will remain the same digit. Thus we have knowledge of certain *invariances* of the input pattern. In this section we describe a number of ways in which such invariances can be exploited. Our discussion is based on Schölkopf and Smola [2002, ch. 11].

Prior knowledge about the problem tells us that certain transformations of the input would leave the class label invariant—these include simple geometric transformations such as translations, rotations,[1] rescalings, and rather less obvious ones such as line thickness transformations.[2] Given enough data it should be possible to learn the correct input-output mapping, but it would make sense to try to make use of these known invariances to reduce the amount of training data needed. There are at least three ways in which this prior knowledge has been used, as described below.

synthetic training examples

The first approach is to generate synthetic training examples by applying valid transformations to the examples we already have. This is simple but it does have the disadvantage of creating a larger training set. As kernel-machine training algorithms typically scale super-linearly with $n$ this can be problematic.

tangent prop

A second approach is to make the predictor invariant to small transformations of each training case; this method was first developed by Simard et al. [1992] for neural networks under the name of "tangent prop". For a single training image we consider the manifold of images that are generated as various transformations are applied to it. This manifold will have a complex structure, but locally we can approximate it by a tangent space. The idea in "tangent prop" is that the output should be invariant to perturbations of the training example in this tangent space. For neural networks it is quite straightforward to modify the training objective function to penalize deviations from this invariance, see Simard et al. [1992] for details. Section 11.4 in Schölkopf and Smola [2002] describes some ways in which these ideas can be extended to kernel machines.

invariant representation

The third approach to dealing with invariances is to develop a representation of the input which is invariant to some or all of the transformations. For example, binary images of handwritten digits are sometimes "skeletonized" to remove the effect of line thickness. If an invariant representation can be achieved for all transformations it is the most desirable, but it can be difficult or perhaps impossible to achieve. For example, if a given training pattern can belong to more than one class (e.g. an ambiguous handwritten digit) then it is clearly not possible to find a new representation which is invariant to transformations yet leaves the classes distinguishable.

[1] The digit recognition problem is only invariant to small rotations; we must avoid turning a 6 into a 9.

[2] i.e. changing the thickness of the pen we write with within reasonable bounds does not change the digit we write.

## 9.11 Latent Variable Models

Our main focus in this book has been on supervised learning. However, GPs have also been used as components for models carrying out non-linear dimensionality reduction, a form of unsupervised learning. The key idea is that data which is apparently high-dimensional (e.g. a pixelated image) may really lie on a low-dimensional non-linear manifold which we wish to model.

Let $\mathbf{z} \in \mathbb{R}^L$ be a latent (or hidden) variable, and let $\mathbf{x} \in \mathbb{R}^D$ be a visible variable. We suppose that our visible data is generated by picking a point in $\mathbf{z}$-space and mapping this point into the data space through a (possibly non-linear) mapping, and then optionally adding noise. Thus $p(\mathbf{x}) = \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$. If the mapping from $\mathbf{z}$ to $\mathbf{x}$ is linear and $\mathbf{z}$ has a Gaussian distribution then this is the factor analysis model, and the mean and covariance of the Gaussian in $\mathbf{x}$-space can easily be determined. However, if the mapping is non-linear then the integral cannot be computed exactly. In the generative topographic mapping (GTM) model [Bishop et al., 1998b] the integral was approximated using a grid of points in $\mathbf{z}$-space. In the original GTM paper the non-linear mapping was taken to be a linear combination of non-linear basis functions, but in Bishop et al. [1998a] this was replaced by a Gaussian process mapping between the latent and visible spaces.

GTM

More recently Lawrence [2004] has introduced a rather different model known as the Gaussian process latent variable model (GPLVM). Instead of having a prior (and thus a posterior) distribution over the latent space, we consider that each data point $\mathbf{x}_i$ is derived from a corresponding latent point $\mathbf{z}_i$ through a non-linear mapping (with added noise). If a Gaussian process is used for this non-linear mapping, then one can easily write down the joint distribution $p(X|Z)$ of the visible variables conditional on the latent variables. Optimization routines can then be used to find the locations of the latent points that optimize $p(X|Z)$. This has some similarities to the work on regularized principal manifolds [Schölkopf and Smola, 2002, ch. 17] except that in the GPLVM one integrates out the latent-to-visible mapping rather than optimizing it.

GPLVM

## 9.12 Conclusions and Future Directions

In this section we briefly wrap up some of the threads we have developed throughout the book, and discuss possible future directions of work on Gaussian processes.

In chapter 2 we saw how Gaussian process regression is a natural extension of Bayesian linear regression to a more flexible class of models. For Gaussian noise the model can be treated analytically, and is simple enough that the GP model could be often considered as a replacement for the traditional linear analogue. We have also seen that historically there have been numerous ideas along the lines of Gaussian process models, although they have only gained a sporadic following.

C. E. Rasmussen & C. K. I. Williams, Gaussian Processes for Machine Learning, the MIT Press, 2006, ISBN 026218253X.  www.GaussianProcess.org/gpml



One may indeed speculate, why are GPs not currently used more widely in applications? We see three major reasons: (1) Firstly, that the application of Gaussian processes requires the handling (inversion) of large matrices. While these kinds of computations were tedious 20 years ago, and impossible further in the past, even naïve implementations suffice for moderate sized problems on an anno 2005 PC. (2) Another possibility is that most of the historical work on GPs was done using fixed covariance functions, with very little guide as to how to choose such functions. The choice was to some degree arbitrary, and the idea that one should be able to *infer* the structure or parameters of the covariance function as we discuss in chapter 5 is not so well known. This is probably a very important step in turning GPs into an interesting method for practitioners. (3) The viewpoint of placing Gaussian process priors over functions is a Bayesian one. Although the adoption of Bayesian methods in the machine learning community is quite widespread, these ideas have not always been appreciated more widely in the statistics community.

Although modern computers allow simple implementations for up to a few thousand training cases, the computational constraints are still a significant limitation for applications where the datasets are significantly larger than this. In chapter 8 we have given an overview of some of the recent work on approximations for large datasets. Although there are many methods and a lot of work is currently being undertaken, both the theoretical and practical aspects of these approximations need to be understood better in order to be a useful tool to the practitioner.

The computations required for the Gaussian process classification models developed in chapter 3 are a lot more involved than for regression. Although the theoretical foundations of Gaussian process classification are well developed, it is not yet clear under which circumstances one would expect the extra work and approximations associated with treating a full probabilistic latent variable model to pay off. The answer may depend heavily on the ability to learn meaningful covariance functions.

The incorporation of prior knowledge through the choice and parameterization of the covariance function is another prime target for future work on GPs. In chapter 4 we have presented many families of covariance functions with widely differing properties, and in chapter 5 we presented principled methods for choosing between and adapting covariance functions. Particularly in the machine learning community, there has been a tendency to view Gaussian processes as a "black box"—what exactly goes on in the box is less important, as long as it gives good predictions. To our mind, we could perhaps learn something from the statisticians here, and ask how and why the models work. In fact the hierarchical formulation of the covariance functions with hyperparameters, the testing of different hypotheses and the adaptation of hyperparameters gives an excellent opportunity to understand more about the data.

We have attempted to illustrate this line of thinking with the carbon dioxide prediction example developed at some length in section 5.4.3. Although this problem is comparatively simple and very easy to get an intuitive understanding of, the principles of trying out different components in the covariance structure

and adapting their parameters could be used universally. Indeed, the use of the isotropic squared exponential covariance function in the digit classification examples in chapter 3 is not really a choice which one would expect to provide very much insight to the classification problem. Although some of the results presented are as good as other current methods in the literature, one could indeed argue that the use of the squared exponential covariance function for this task makes little sense, and the low error rate is possibly due to the inherently low difficulty of the task. There is a need to develop more sensible covariance functions which allow for the incorporation of prior knowledge and help us to gain real insight into the data.

Going beyond a simple vectorial representation of the input data to take into account structure in the input domain is also a theme which we see as very important. Examples of this include the invariances described in section 9.10 arising from the structure of images, and the kernels described in section 4.4 which encode structured objects such as strings and trees.

As this brief discussion shows, we see the current level of development of Gaussian process models more as a rich, principled framework for supervised learning than a fully-developed set of tools for applications. We find the Gaussian process framework very appealing and are confident that the near future will show many important developments, both in theory, methodology and practice. We look forward very much to following these developments.