An Introduction to Gaussian Process Regression

P.L.Green
Institute for Risk and Uncertainty
School of Engineering
University of Liverpool
Liverpool, L69 7ZF
United Kingdom

p.l.green@liverpool.ac.uk
https://www.liverpool.ac.uk/engineering/staff/peter-green/

September 3, 2018

1 Introduction

This document is supposed to provide a tutorial-style introduction to Gaussian Process (GP) regression. It comes with some example Gaussian Process code, written in Python, that is free and should be relatively easy to use. The code is by no-means optimised, rather, it is supposed to be easy to read and easy to understand for someone who is new to GP regression.

2 The Problem

Say we believe that a system's input, $\boldsymbol{x}$, is related to its output via a relationship $f(\boldsymbol{x})$. The function, f, is currently unknown. To help us find f, we conduct a series of experiments where, for N different $\boldsymbol{x}$ values, we observe the system's output (denoted y). Our experiment therefore yields the data $(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_N, y_N)$. As with any experiment, our observations are corrupted by some errors. In the following, we treat these errors as being Gaussian and we assume the following 'noise model':

$$y_i = f(\boldsymbol{x}_i) + \epsilon, \qquad \epsilon \sim \mathcal{N}(\epsilon; 0, \sigma^2) \tag{1}$$

that is, we assume that each observation y_i is equal to the function, f, evaluated at input, $\boldsymbol{x}_i$, but corrupted with Gaussian noise of variance σ^2.

3 Bayesian Linear Regression

Once we have our data, $(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_N, y_N)$, we can use it to try and infer the function f. Say we assume that f takes the form

$$f(\boldsymbol{x}_i) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x}_i) \tag{2}$$

where $\boldsymbol{\theta}$ is a vector of parameters which we need to estimate and $\boldsymbol{\phi}$ is a 'basis function'. For example, if we are considering univariate inputs and set

$$\boldsymbol{\theta} = (\theta_0, \theta_1, \theta_2)^T, \qquad \boldsymbol{\phi}(x_i) = (1, x_i, x_i^2)^T \tag{3}$$

then we would be trying to fit a quadratic function to our data. Equation (2) is convenient because it is a linear function of the parameters we need to estimate - this usually makes it easier to find closed-form expression for the 'optimum choice' of $\boldsymbol{\theta}$.

Defining

$$\boldsymbol{f} = (f_1, f_2, ..., f_N)^T, \qquad \text{where} \qquad f_i \equiv f(\boldsymbol{x}_i) \tag{4}$$

we can write

$$f_i = \sum_k \theta_k \phi_k(\boldsymbol{x}_i). \tag{5}$$

Furthermore, defining the matrix

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_1(\boldsymbol{x}_1) & \phi_2(\boldsymbol{x}_1) & ... \\ \phi_1(\boldsymbol{x}_2) & \phi_2(\boldsymbol{x}_2) & \\ \vdots & & \ddots \end{bmatrix} \tag{6}$$

lets us write $\boldsymbol{f} = \boldsymbol{\Phi}\,\boldsymbol{\theta}$.

At this point, a Bayesian approach would typically involve defining a prior probability distribution, $p(\boldsymbol{\theta})$, that is supposed to represent our prior belief in $\boldsymbol{\theta}$. We would then write down Bayes theorem:

$$p(\boldsymbol{\theta} \,|\, \boldsymbol{y}) \propto p(\boldsymbol{y} \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta}) \tag{7}$$

where $\boldsymbol{y} = (y_1, ..., y_N)$ is our observation data and the likelihood, $p(\boldsymbol{y} \,|\, \boldsymbol{\theta})$, is formed based on our noise model (equation (1) in this case). This approach gives us an expression for the posterior distribution, $p(\boldsymbol{\theta} \,|\, \boldsymbol{y})$, which allows us to analyse the probability that our regression parameters can take particular values, conditional on our observation data.

If we adopt such a Bayesian approach and happen choose the prior over our parameters to be

$$p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}; \boldsymbol{0}, \alpha^{-1}\boldsymbol{I}) \tag{8}$$

then, after 'feeding it through' our model ($\boldsymbol{f} = \boldsymbol{\Phi}\,\boldsymbol{\theta}$), it follows that the covariance matrix over $\boldsymbol{f}$ is

$$\mathrm{Cov}[\boldsymbol{f}] = \mathrm{E}[\boldsymbol{f}\,\boldsymbol{f}^T] = \boldsymbol{\Phi}\,\mathrm{E}[\boldsymbol{\theta}\,\boldsymbol{\theta}^T]\boldsymbol{\Phi}^T = \alpha^{-1}\boldsymbol{\Phi}\boldsymbol{\Phi}^T \tag{9}$$

(which we'll write just as $\mathrm{Cov}[\boldsymbol{y}] = \boldsymbol{K}$ from now on). This result means that, regardless of the inputs used in our experiments, saying that our prior over $\boldsymbol{\theta}$ is Gaussian means we automatically assume a Gaussian prior over $\boldsymbol{f}$:

$$p(\boldsymbol{f}) \propto \exp\left(-\frac{1}{2}\,\boldsymbol{f}^T\,\boldsymbol{K}^{-1}\,\boldsymbol{f}\right). \tag{10}$$

where the individual elements of $\boldsymbol{K}$ are given by

$$K_{ij} = \alpha^{-1} \sum_i \phi_i(\boldsymbol{x}_i)\phi_i(\boldsymbol{x}_j) \tag{11}$$

4 Gaussian Process Regression

In the previous section we defined a function f that was linear in our regression parameters, $\boldsymbol{\theta}$, before defining a Gaussian prior over $\boldsymbol{\theta}$. This led to a Gaussian prior over $\boldsymbol{f}$, whose covariance matrix is given by equation (11). Key to the construction of GPs is the idea that, as we know that the aforementioned Bayesian approach will always lead to a Gaussian distribution over $\boldsymbol{f}$, we may as well define $p(\boldsymbol{f})$ directly and skip the part where we define a basis function and a prior over $\boldsymbol{\theta}$. Specifically, we simply write

$$p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}; \mathbf{0}, \boldsymbol{K}) \tag{12}$$

where the matrix $\boldsymbol{K}$ can be anything we like (so long as it is a valid covariance matrix). For this reason, GPs are sometimes described as a method that 'assigns a prior over the function values, $\boldsymbol{f}$'. The matrix $\boldsymbol{K}$ (the 'gram matrix') is usually defined using a *kernel function*, k, such that $\boldsymbol{K}_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. A common choice of kernel function, for example, is

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{1}{2l^2}(\boldsymbol{x}_i - \boldsymbol{x}_j)^T(\boldsymbol{x}_i - \boldsymbol{x}_j)\right) \tag{13}$$

where we have introduced l, a 'length scale' parameter. So how do the kernel function (equation (13)) and length scale parameter alter our prior belief about the function f? Consider the case where we have a vector of 100 1D inputs, that are evenly spaced between 0 and 10. We choose the kernel described by equation (13) and plot the matrix $\boldsymbol{K}$ for $l = 10, 1$ and 0.1. The resulting gram matrices are shown in Figure 1.

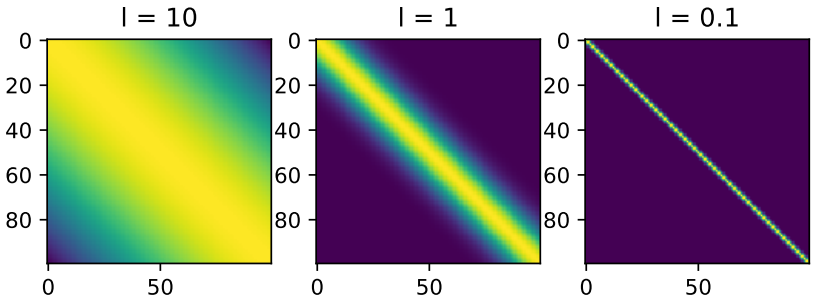

Figure 1: Plotting the gram matrix, $\boldsymbol{K}$, for different values of length scale, l, using the kernel function described by equation (13).

Figure 1 shows us that, by using larger values of l, we are introducing greater correlation between our inputs. So what does this signify? Generating 100 samples of $\boldsymbol{f}$, from $\mathcal{N}(\boldsymbol{f}; \mathbf{0}, \boldsymbol{K})$ for different values of l, we realise the results shown in Figure 2.

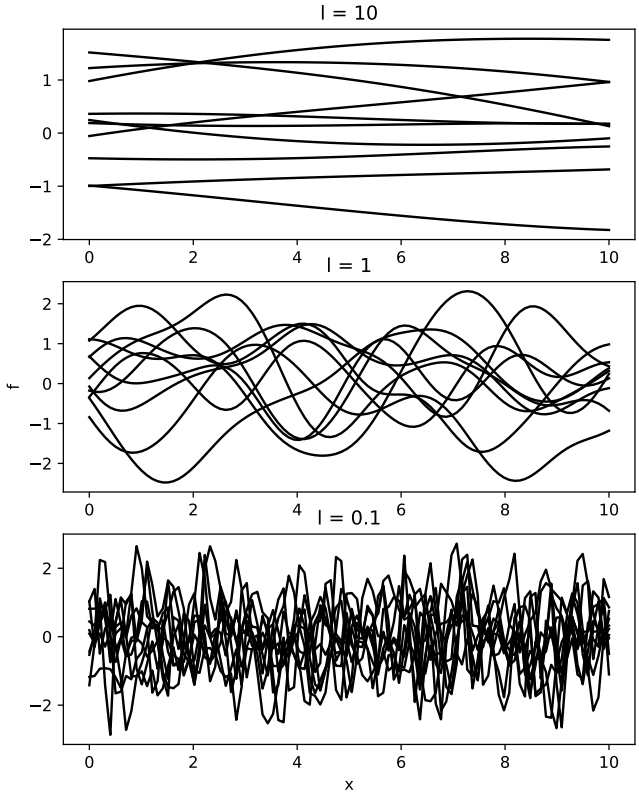

Figure 2: 100 samples of $\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{f}, \boldsymbol{0}, \boldsymbol{K})$, for different values of length scale, l.

Figure 2 shows us that, by increasing the correlation between our inputs, we are inducing *smoothness* in our prior function f. As we shall see shortly, 'smoothness' is quite an intuitive and useful quantity when it comes to performing regression.

So, to recap, we have our prior

$$p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}; \boldsymbol{0}, \boldsymbol{K}) \tag{14}$$

and we have our noise model, equation (1). As we have assumed that the noise affecting our observations is independent for each data point, it follows that

$$p(\boldsymbol{y} \,|\, \boldsymbol{f}) = \mathcal{N}\left(\boldsymbol{y}; \boldsymbol{f}, \boldsymbol{I}\sigma^2\right) \tag{15}$$

We can then marginalise over $\boldsymbol{f}$, to find the probability of witnessing $\boldsymbol{y}$:

$$p(\boldsymbol{y}) = \int p(\boldsymbol{y} \,|\, \boldsymbol{f}) p(\boldsymbol{f}) d\,\boldsymbol{f} \tag{16}$$

Fortunately, we can find a closed-from expression for $p(\boldsymbol{y})$ using some generic expressions for Gaussian distributions (see Appendix A). After some manipulation, we find that

$$p(\boldsymbol{y}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{0}, \boldsymbol{C}) \tag{17}$$

where

$$\boldsymbol{C} = \boldsymbol{I}\sigma^2 + \boldsymbol{K} \implies C_{ij} = \sigma^2 \delta_{ij} + k(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{18}$$

Now, what we really want to do is analyse the probability that, in response to a new input, $\boldsymbol{x}_*$, we would observe the response y_*. Using the relationship we derived in equation (18), we can write

$$p(\boldsymbol{y}, y_*) = \mathcal{N}\left(\begin{pmatrix} \boldsymbol{y} \\ y_* \end{pmatrix}; \begin{pmatrix} \boldsymbol{0} \\ 0 \end{pmatrix}, \begin{bmatrix} \boldsymbol{C} & \boldsymbol{k} \\ \boldsymbol{k}^T & c \end{bmatrix}\right) \tag{19}$$

where the vector $\boldsymbol{k}$ is defined as

$$\boldsymbol{k}_n = k(\boldsymbol{x}_n, \boldsymbol{x}_*), \qquad n = 1, ..., N \tag{20}$$

and

$$c = \sigma^2 + k(\boldsymbol{x}_*, \boldsymbol{x}_*). \tag{21}$$

By once again exploiting our generic expressions for Gaussian distributions, we are then able to find that

$$p(y_* | \, \boldsymbol{y}) = \mathcal{N}(y_*; m_*, \sigma_*^2) \tag{22}$$

where

$$m_* = \boldsymbol{k}^T \boldsymbol{C}^{-1} \boldsymbol{t}, \qquad \sigma_*^2 = c - \boldsymbol{k}^T \boldsymbol{C}^{-1} \boldsymbol{k}. \tag{23}$$

5 Example: 1D Interpolation

Here we have generated some 'observations' according to:

$$y_n = \sin x_n + \epsilon, \qquad \epsilon \sim \mathcal{N}(\epsilon; 0, \sigma^2) \tag{24}$$

where $\sigma = 0.2$. We are going to use a Gaussian process to 'fill the gaps' between the data that we have gathered. Using the kernel function in equation (13), Figure 3 shows what happens when we set $l = 0.5, 1$ and 5 respectively.

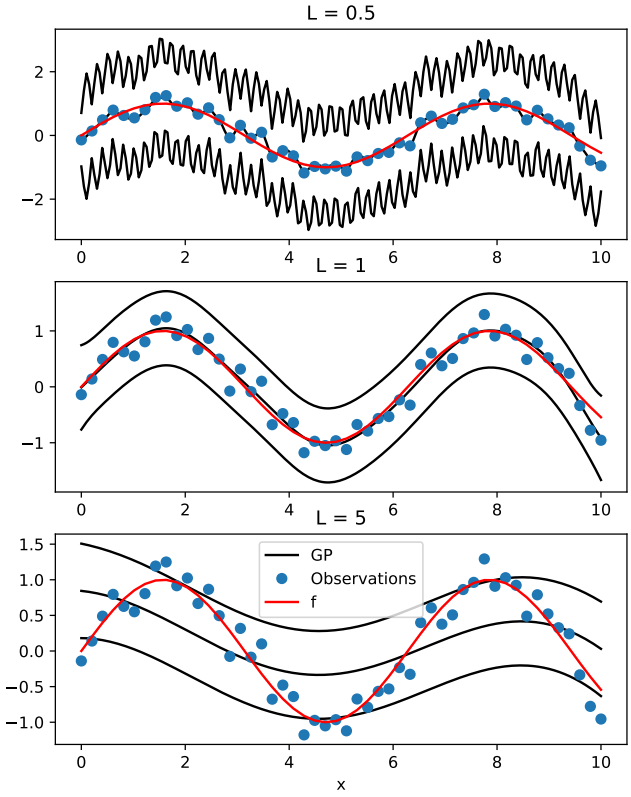

Figure 3: Performing Gaussian Process regression for different values of the length scale, l. Gaussian Process confidence bounds are 3 standard deviations from the mean.

From Figure 3 it should be obvious that, when $l = 0.5$, our interpolating function isn't smooth enough. In fact, it passes through every observation almost exactly - by mistakenly fitting to the noise that is in the data, it has formed a poor representation of the true function, f. This phenomenon is sometimes referred to as 'overfitting'. When $l = 1$, the fit appears sensible while, when $l = 5$, our fit seems overly smooth (in other words, it is not able to react quickly enough to changes in the input, x).

6 Gaussian Process Learning

A useful aspect of GPs is that, unlike our linear regression approach where we had to begin by assuming a functional form for f (equation (2)), GPs are not so tightly constrained. They are sometimes referred to as a form of *non-parametric regression* as they are not limited to only consider one parametric family of regression functions. The GP does, however, have some parameters which require tuning (l and σ is our case). These, perhaps a little confusingly, are often referred to as *hyperparameters*.

Writing our hyperparameters together as $\boldsymbol{\theta}_{\text{hyp}}$ such that $\boldsymbol{\theta}_{\text{hpy}} = \{l, \sigma\}$ then, from equation (17) (derived in the previous section), we know that the probability of witnessing our observations conditional on $\boldsymbol{\theta}_{\text{hyp}}$ is given by

$$p(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{\text{hyp}}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{0}, \boldsymbol{C}) = \left((2\pi)^N |\boldsymbol{C}|\right)^{-1/2} \exp\left(-\frac{1}{2}\, \boldsymbol{y}^T\, \boldsymbol{C}^{-1}\, \boldsymbol{y}\right) \tag{25}$$

such that:

$$\log p(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{\text{hyp}}) = -\frac{1}{2}\left(N \log 2\pi + \log|\boldsymbol{C}| + \boldsymbol{y}^T \boldsymbol{C}^{-1} \boldsymbol{y}\right). \tag{26}$$

Finding the maximum of equation (26), we can therefore find the maximum likelihood estimates of $\boldsymbol{\theta}_{\text{hyp}}$. This is typically what happens during the 'learning process' for a GP. Conveniently, to help our optimisation routine, we are usually able to evaluate the gradient of the log-likelihood with respect to our hyperparameters using closed-form expressions. For example, noting that:

$$\frac{\partial \log|\boldsymbol{C}|}{\partial \theta_{\text{hyp},i}} = \text{Tr}\left(\boldsymbol{C}^{-1}\frac{\partial \boldsymbol{C}}{\partial \theta_{\text{hyp},i}}\right), \qquad \frac{\partial \boldsymbol{C}^{-1}}{\partial \theta_{\text{hyp},i}} = -\boldsymbol{C}^{-1}\frac{\partial \boldsymbol{C}}{\partial \theta_{\text{hyp},i}}\boldsymbol{C}^{-1} \tag{27}$$

then it can be shown that

$$\frac{\partial}{\partial \theta_{\text{hyp},i}} \log p(\boldsymbol{y} \,|\, \boldsymbol{\theta}_{\text{hyp}}) = -\frac{1}{2}\left(\text{Tr}\left(\boldsymbol{C}^{-1}\frac{\partial \boldsymbol{C}}{\partial \theta_{\text{hyp},i}}\right) - \boldsymbol{y}^T \boldsymbol{C}^{-1}\frac{\partial \boldsymbol{C}}{\partial \theta_{\text{hyp},i}}\boldsymbol{C}^{-1}\boldsymbol{y}\right). \tag{28}$$

7 Example Python Code

Code accompanying this tutorial is written in Python (version 2), is heavily commented and is designed to be readable. The module 'LIRU_GP' contains all of the functions needed to train a GP for regression. The scripts '1D_regression_example' and '2D_regression_example', as the names imply, use LIRU_GP to perform regression on a 1D and 2D problem respectively. '1D_regression_example' will plot results similar to those shown in Figure 4 while '2D_regression_example' will produce results similar to those shown in Figure 5.

Note that, for more optimised and generic code, a variety of GP packages are offered though Python (see `https://sheffieldml.github.io/GPy/` for example). As stated previously, our code is intended to be readable, rather than generic.

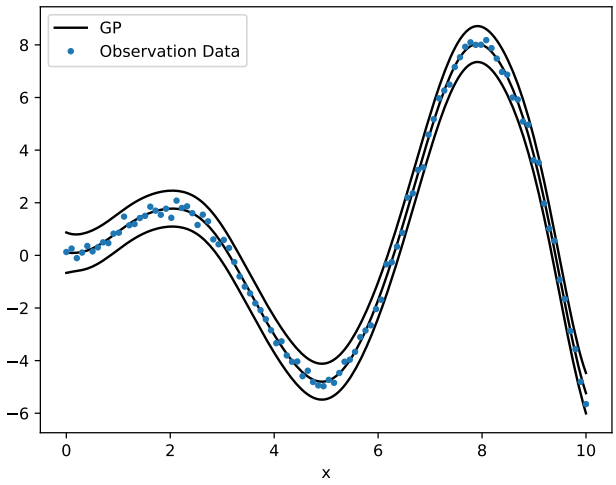

Figure 4: Example output of '1D_regression_example'.

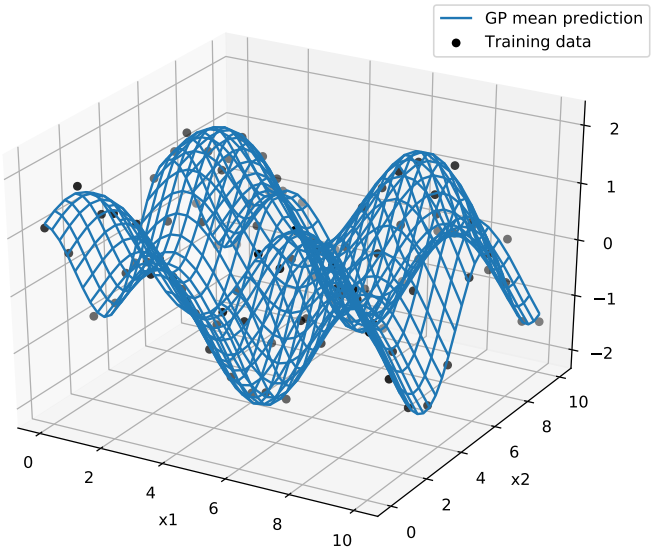

Figure 5: Example output of '2D_regression_example'.

A Properties of Gaussians

Before we start we should note some useful properties. All this stuff if proved in Appendix B (which is just a slightly longer version of what you can find in Bishop [1]).

Say we have a partitioned Gaussian $p(\boldsymbol{x}) \propto \exp(-\frac{1}{2}\phi)$ where

$$\phi = \left((\boldsymbol{x}_a - \boldsymbol{\mu}_a)^T, (\boldsymbol{x}_b - \boldsymbol{\mu}_b)^T\right) \begin{bmatrix} \boldsymbol{A}_{aa} & \boldsymbol{A}_{ab} \\ \boldsymbol{A}_{ba} & \boldsymbol{A}_{bb} \end{bmatrix} \begin{pmatrix} \boldsymbol{x}_a - \boldsymbol{\mu}_a \\ \boldsymbol{x}_b - \boldsymbol{\mu}_b \end{pmatrix}. \tag{29}$$

$$= \left((\boldsymbol{x}_a - \boldsymbol{\mu}_a)^T, (\boldsymbol{x}_b - \boldsymbol{\mu}_b)^T\right) \begin{bmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{bmatrix}^{-1} \begin{pmatrix} \boldsymbol{x}_a - \boldsymbol{\mu}_a \\ \boldsymbol{x}_b - \boldsymbol{\mu}_b \end{pmatrix}. \tag{30}$$

Written in terms of precision matrices we have the properties that:

Distribution	Precision	Mean
$p(\boldsymbol{x}_a\|\boldsymbol{x}_b)$	$\boldsymbol{A}_{aa}$	$\boldsymbol{A}_{aa}^{-1}\left(\boldsymbol{A}_{aa}\boldsymbol{\mu}_a - \boldsymbol{A}_{ab}(\boldsymbol{x}_b - \boldsymbol{\mu}_b)\right)$
$p(\boldsymbol{x}_b\|\boldsymbol{x}_a)$	$\boldsymbol{A}_{bb}$	$\boldsymbol{A}_{bb}^{-1}\left(\boldsymbol{A}_{bb}\boldsymbol{\mu}_b - \boldsymbol{A}_{ba}(\boldsymbol{x}_a - \boldsymbol{\mu}_a)\right)$
$p(\boldsymbol{x}_a)$	$\boldsymbol{A}_{aa} - \boldsymbol{A}_{ab}\boldsymbol{A}_{bb}^{-1}\boldsymbol{A}_{ba}$	$\boldsymbol{\mu}_a$
$p(\boldsymbol{x}_b)$	$\boldsymbol{A}_{bb} - \boldsymbol{A}_{ba}\boldsymbol{A}_{aa}^{-1}\boldsymbol{A}_{ab}$	$\boldsymbol{\mu}_b$

Written in terms of covariance matrices these can also be written as:

Distribution	Covariance	Mean
$p(\boldsymbol{x}_a\|\boldsymbol{x}_b)$	$\boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}$	$\boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\boldsymbol{x}_b - \boldsymbol{\mu}_b)$
$p(\boldsymbol{x}_b\|\boldsymbol{x}_a)$	$\boldsymbol{\Sigma}_{bb} - \boldsymbol{\Sigma}_{ba}\boldsymbol{\Sigma}_{aa}^{-1}\boldsymbol{\Sigma}_{ab}$	$\boldsymbol{\mu}_b + \boldsymbol{\Sigma}_{ba}\boldsymbol{\Sigma}_{aa}^{-1}(\boldsymbol{x}_a - \boldsymbol{\mu}_a)$
$p(\boldsymbol{x}_a)$	$\boldsymbol{\Sigma}_{aa}$	$\boldsymbol{\mu}_a$
$p(\boldsymbol{x}_b)$	$\boldsymbol{\Sigma}_{bb}$	$\boldsymbol{\mu}_b$

Finally, for the case where

$$p(\boldsymbol{x}_a) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{A}^{-1}), \qquad p(\boldsymbol{x}_b|\boldsymbol{x}_a) = \mathcal{N}(\boldsymbol{H}\boldsymbol{x}_a + \boldsymbol{b}, \boldsymbol{L}^{-1}) \tag{31}$$

then we have

Distribution	Covariance	Mean
$p(\boldsymbol{x}_a\|\boldsymbol{x}_b)$	$(\boldsymbol{A} + \boldsymbol{H}^T\boldsymbol{L}\boldsymbol{H})^{-1}$	$(\boldsymbol{A} + \boldsymbol{H}^T\boldsymbol{L}\boldsymbol{H})^{-1}\left(\boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{H}^T\boldsymbol{L}(\boldsymbol{x}_b - \boldsymbol{b})\right)$
$p(\boldsymbol{x}_b)$	$\boldsymbol{L}^{-1} + \boldsymbol{H}\boldsymbol{A}^{-1}\boldsymbol{H}^T$	$\boldsymbol{H}\boldsymbol{\mu} + \boldsymbol{b}$

These expressions are generic - they will simplify down a lot for the examples shown here.

B Deriving the properties of Gaussian distributions

B.1 Partitioned Gaussian

We have a Gaussian distribution $p(\boldsymbol{x}) = \exp(-\frac{1}{2}\phi(\boldsymbol{x}))$ where

$$\phi = (\boldsymbol{x} - \boldsymbol{\mu})^T\boldsymbol{A}(\boldsymbol{x} - \boldsymbol{\mu}) \tag{32}$$

where $\boldsymbol{A}$ is the precision matrix. We are going to partition this up:

$$\boldsymbol{x} = (\boldsymbol{x}_a, \boldsymbol{x}_b)^T, \qquad \boldsymbol{\mu} = (\boldsymbol{\mu}_a, \boldsymbol{\mu}_b)^T, \qquad \boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{aa} & \boldsymbol{A}_{ab} \\ \boldsymbol{A}_{ba} & \boldsymbol{A}_{bb} \end{bmatrix} \tag{33}$$

such that $p(\boldsymbol{x}) = p(\boldsymbol{x}_a, \boldsymbol{x}_b) \propto \exp(-\frac{1}{2}\phi)$ where

$$\phi = \left((\boldsymbol{x}_a - \boldsymbol{\mu}_a)^T, (\boldsymbol{x}_b - \boldsymbol{\mu}_b)^T\right) \begin{bmatrix} \boldsymbol{A}_{aa} & \boldsymbol{A}_{ab} \\ \boldsymbol{A}_{ba} & \boldsymbol{A}_{bb} \end{bmatrix} \begin{pmatrix} \boldsymbol{x}_a - \boldsymbol{\mu}_a \\ \boldsymbol{x}_b - \boldsymbol{\mu}_b \end{pmatrix} \tag{34}$$

$$= (\boldsymbol{x}_a - \boldsymbol{\mu}_a)^T \boldsymbol{A}_{aa} (\boldsymbol{x}_a - \boldsymbol{\mu}_a) + (\boldsymbol{x}_a - \boldsymbol{\mu}_a)^T \boldsymbol{A}_{ab} (\boldsymbol{x}_b - \boldsymbol{\mu}_b)$$

$$+ (\boldsymbol{x}_b - \boldsymbol{\mu}_b)^T \boldsymbol{A}_{ba} (\boldsymbol{x}_a - \boldsymbol{\mu}_a) + (\boldsymbol{x}_b - \boldsymbol{\mu}_b)^T \boldsymbol{A}_{bb} (\boldsymbol{x}_b - \boldsymbol{\mu}_b). \tag{35}$$

B.2 Conditional Gaussian

We are going to take $p(\boldsymbol{x}_a, \boldsymbol{x}_b)$ and find $p(\boldsymbol{x}_a|\boldsymbol{x}_b)$. This essentially involves taking equation (35) and treating all the $\boldsymbol{x}_b$ terms as being constant (which we'll lose when the distribution is normalised). Consequently, defining C as 'some constant' and noting that

$$(\boldsymbol{x}_a - \boldsymbol{\mu}_a)^T \boldsymbol{A}_{ab} (\boldsymbol{x}_b - \boldsymbol{\mu}_b) = (\boldsymbol{x}_b - \boldsymbol{\mu}_b)^T \boldsymbol{A}_{ba} (\boldsymbol{x}_a - \boldsymbol{\mu}_a) \tag{36}$$

(as $\boldsymbol{A}_{ab}^T = \boldsymbol{A}_{ba}$) then we find that

$$\phi = (\boldsymbol{x}_a - \boldsymbol{\mu}_a)^T \boldsymbol{A}_{aa} (\boldsymbol{x}_a - \boldsymbol{\mu}_a) + 2(\boldsymbol{x}_a - \boldsymbol{\mu}_a)^T \boldsymbol{A}_{ab} (\boldsymbol{x}_b - \boldsymbol{\mu}_b) + C \tag{37}$$

$$= \boldsymbol{x}_a^T \boldsymbol{A}_{aa} \boldsymbol{x}_a - 2\boldsymbol{x}_a^T \left(\boldsymbol{A}_{aa}\boldsymbol{\mu}_a - \boldsymbol{A}_{ab}(\boldsymbol{x}_b - \boldsymbol{\mu}_b)\right) + C \tag{38}$$

and so, by completing the square, we find that $p(\boldsymbol{x}_a|\boldsymbol{x}_b)$ is Gaussian with precision matrix

$$\boldsymbol{A}_{a|b} = \boldsymbol{A}_{aa} \tag{39}$$

and mean

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{A}_{aa}^{-1} \left(\boldsymbol{A}_{aa}\boldsymbol{\mu}_a - \boldsymbol{A}_{ab}(\boldsymbol{x}_b - \boldsymbol{\mu}_b)\right) \tag{40}$$

$$= \boldsymbol{\mu}_a - \boldsymbol{A}_{aa}^{-1} \boldsymbol{A}_{ab} (\boldsymbol{x}_b - \boldsymbol{\mu}_b). \tag{41}$$

By inverting the precision matrix (which we won't go into here) we can also show that

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab} \boldsymbol{\Sigma}_{bb}^{-1} (\boldsymbol{x}_b - \boldsymbol{\mu}_b) \tag{42}$$

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab} \boldsymbol{\Sigma}_{bb}^{-1} \boldsymbol{\Sigma}_{ba}. \tag{43}$$

Using the same basic method you can get similar expressions for $p(\boldsymbol{x}_b|\boldsymbol{x}_a)$.

B.3 Marginalised Gaussian

Now we will find $p(\boldsymbol{x}_a) = \int p(\boldsymbol{x}_a, \boldsymbol{x}_b) d\boldsymbol{x}_b$. To do this we will collect together all of the $\boldsymbol{x}_b$ terms in equation (35) and complete the square. By doing this, when we integrate over $\boldsymbol{x}_b$, we will simply get the normalising constant of a Gaussian which can be absorbed into the constant C. We then just collect together all the remaining $\boldsymbol{x}_a$ terms and complete the square again.

To that end, all the terms involving $\boldsymbol{x}_b$ in equation (35) are:

$$(\boldsymbol{x}_b - \boldsymbol{\mu}_b)^T \boldsymbol{A}_{bb} (\boldsymbol{x}_b - \boldsymbol{\mu}_b) + (\boldsymbol{x}_b - \boldsymbol{\mu}_b)^T \boldsymbol{A}_{ba} (\boldsymbol{x}_a - \boldsymbol{\mu}_a) \tag{44}$$

$$= \boldsymbol{x}_b^T \boldsymbol{A}_{bb} \boldsymbol{x}_b - 2\boldsymbol{x}_b^T \left(\boldsymbol{A}_{bb} \boldsymbol{\mu}_b - \boldsymbol{A}_{ba} (\boldsymbol{x}_a - \boldsymbol{\mu}_a) \right) + C \tag{45}$$

$$= (\boldsymbol{x}_b - \boldsymbol{A}_{bb}^{-1} \boldsymbol{m})^T \boldsymbol{A}_{bb} (\boldsymbol{x}_b - \boldsymbol{A}_{bb}^{-1} \boldsymbol{m}) - \boldsymbol{m}^T \boldsymbol{A}_{bb}^{-1} \boldsymbol{m} + C \tag{46}$$

where

$$\boldsymbol{m} = \boldsymbol{A}_{bb} \boldsymbol{\mu}_b - \boldsymbol{A}_{ba} (\boldsymbol{x}_a - \boldsymbol{\mu}_a). \tag{47}$$

Notice we've not included $\boldsymbol{m}^T \boldsymbol{A}_{bb}^{-1} \boldsymbol{m}$ in C as it contains some $\boldsymbol{x}_a$ terms. Once we have integrated over $\boldsymbol{x}_b$ we will be left with the $\boldsymbol{x}_a$ terms left over from equation (35) and $-\boldsymbol{m}^T \boldsymbol{A}_{bb} \boldsymbol{m}$.

With

$$\boldsymbol{m}^T \boldsymbol{A}_{bb}^{-1} \boldsymbol{m} = \left(\boldsymbol{A}_{bb} \boldsymbol{\mu}_b - \boldsymbol{A}_{ba} (\boldsymbol{x}_a - \boldsymbol{\mu}_a) \right)^T \boldsymbol{A}_{bb}^{-1} \left(\boldsymbol{A}_{bb} \boldsymbol{\mu}_b - \boldsymbol{A}_{ba} (\boldsymbol{x}_a - \boldsymbol{\mu}_a) \right) \tag{48}$$

$$\boldsymbol{x}_a^T \boldsymbol{A}_{ab} \boldsymbol{A}_{bb}^{-1} \boldsymbol{A}_{ba} \boldsymbol{x}_a - 2\boldsymbol{x}_a^T (\boldsymbol{A}_{ab} \boldsymbol{\mu}_b + \boldsymbol{A}_{ab} \boldsymbol{A}_{bb}^{-1} \boldsymbol{A}_{ba} \boldsymbol{\mu}_a) + C \tag{49}$$

and, as we have

$$\boldsymbol{x}_a^T \boldsymbol{A}_{aa} \boldsymbol{x}_a - 2\boldsymbol{x}_a^T \boldsymbol{A}_{aa} \boldsymbol{\mu}_a - 2\boldsymbol{x}_a^T \boldsymbol{A}_{ab} \boldsymbol{\mu}_b + C \tag{50}$$

from the left over $\boldsymbol{x}_a$ terms, it follows that after integrating out $\boldsymbol{x}_b$ we are left with

$$\boldsymbol{x}_a^T (\boldsymbol{A}_{aa} - \boldsymbol{A}_{ab} \boldsymbol{A}_{bb}^{-1} \boldsymbol{A}_{ba}) \boldsymbol{x}_a - 2\boldsymbol{x}_a (\boldsymbol{A}_{aa} - \boldsymbol{A}_{ab} \boldsymbol{A}_{bb}^{-1} \boldsymbol{A}_{ba}) \boldsymbol{\mu}_a + C. \tag{51}$$

Consequently, $p(\boldsymbol{x}_a) = \int p(\boldsymbol{x}_a, \boldsymbol{x}_b) d\boldsymbol{x}_b$ has precision matrix

$$\boldsymbol{A}_a = \boldsymbol{A}_{aa} - \boldsymbol{A}_{ab} \boldsymbol{A}_{bb}^{-1} \boldsymbol{A}_{ba} \tag{52}$$

and mean

$$\boldsymbol{A}_a^{-1} (\boldsymbol{A}_{aa} - \boldsymbol{A}_{ab} \boldsymbol{A}_{bb}^{-1} \boldsymbol{A}_{ba}) \boldsymbol{\mu}_a = \boldsymbol{\mu}_a \tag{53}$$

which is nice. Furthermore, taking the inverse of the precision matrix (which we won't do here) we find that

$$\boldsymbol{\Sigma}_a = \boldsymbol{\Sigma}_{aa} \tag{54}$$

which is also nice.

B.4 Bayes' theorem for a linear Gaussian model

Say we have

$$p(\boldsymbol{x}_a) = \mathcal{N}(\mu, \boldsymbol{A}^{-1}), \qquad p(\boldsymbol{x}_b | \boldsymbol{x}_a) = \mathcal{N}(\boldsymbol{H} \boldsymbol{x}_a + \boldsymbol{b}, \boldsymbol{L}^{-1}) \tag{55}$$

and we wish to evaluate Bayes' theorem:

$$p(\boldsymbol{x}_a | \boldsymbol{x}_b) \propto p(\boldsymbol{x}_b | \boldsymbol{x}_a) p(\boldsymbol{x}_a). \tag{56}$$

The easiest way to do is actually to write an expression for $p(\boldsymbol{x}_a, \boldsymbol{x}_b)$ before finding $p(\boldsymbol{x}_a | \boldsymbol{x}_b)$ using the stuff we derived earlier. To that end, writing $\boldsymbol{z}^T = (\boldsymbol{x}_1^T, \boldsymbol{x}_2^T)$ then we have $p(\boldsymbol{z}) \propto \exp(-\frac{1}{2}\phi)$ where

$$\phi = (\boldsymbol{x}_a - \boldsymbol{\mu})^T \boldsymbol{A} (\boldsymbol{x}_a - \boldsymbol{\mu}) + (\boldsymbol{x}_b - \boldsymbol{H} \boldsymbol{x}_a - \boldsymbol{b})^T \boldsymbol{L} (\boldsymbol{x}_b - \boldsymbol{H} \boldsymbol{x}_a - \boldsymbol{b}). \tag{57}$$

Collecting together all the second order terms we get

$$\boldsymbol{x}_a^T \boldsymbol{A} \boldsymbol{x}_a + \boldsymbol{x}_b^T \boldsymbol{L} \boldsymbol{x}_b - 2\boldsymbol{x}_a^T \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{x}_b + \boldsymbol{x}_a^T \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{H} \boldsymbol{x}_a \tag{58}$$

$$= \begin{pmatrix} \boldsymbol{x}_a^T & \boldsymbol{x}_b^T \end{pmatrix} \begin{bmatrix} \boldsymbol{A} + \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{H} & -\boldsymbol{H}^T \boldsymbol{L} \\ -\boldsymbol{L} \boldsymbol{H}^T & \boldsymbol{L} \end{bmatrix} \begin{pmatrix} \boldsymbol{x}_a \\ \boldsymbol{x}_b \end{pmatrix} \tag{59}$$

and so $\boldsymbol{z}$ has precision matrix

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{A} + \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{H} & -\boldsymbol{H}^T \boldsymbol{L} \\ -\boldsymbol{L} \boldsymbol{H}^T & \boldsymbol{L} \end{bmatrix} \tag{60}$$

and, it can be shown, covariance matrix

$$\boldsymbol{R}^{-1} = \begin{bmatrix} \boldsymbol{A}^{-1} & \boldsymbol{A}^{-1} \boldsymbol{H}^T \\ \boldsymbol{H} \boldsymbol{A}^{-1} & \boldsymbol{L}^{-1} + \boldsymbol{H} \boldsymbol{A}^{-1} \boldsymbol{H}^T \end{bmatrix}. \tag{61}$$

Collecting the first order terms:

$$-2\boldsymbol{x}_a^T \boldsymbol{A} \boldsymbol{\mu} - 2\boldsymbol{x}_b^T \boldsymbol{L} \boldsymbol{b} + 2\boldsymbol{x}_a^T \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{b} \tag{62}$$

$$= -2 \begin{pmatrix} \boldsymbol{x}_a^T & \boldsymbol{x}_b^T \end{pmatrix} \begin{pmatrix} \boldsymbol{A} \boldsymbol{\mu} - \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{b} \\ \boldsymbol{L} \boldsymbol{b} \end{pmatrix} \tag{63}$$

and so

$$\mathrm{E}[\boldsymbol{z}] = \boldsymbol{R}^{-1} \begin{pmatrix} \boldsymbol{A} \boldsymbol{\mu} - \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{b} \\ \boldsymbol{L} \boldsymbol{b} \end{pmatrix} \tag{64}$$

$$= \begin{bmatrix} \boldsymbol{A}^{-1} & \boldsymbol{A}^{-1} \boldsymbol{H}^T \\ \boldsymbol{H} \boldsymbol{A}^{-1} & \boldsymbol{L}^{-1} + \boldsymbol{H} \boldsymbol{A}^{-1} \boldsymbol{H}^T \end{bmatrix} \begin{pmatrix} \boldsymbol{A} \boldsymbol{\mu} - \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{b} \\ \boldsymbol{L} \boldsymbol{b} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{H} \boldsymbol{\mu} + \boldsymbol{b} \end{pmatrix}. \tag{65}$$

From our expressions for marginalised Gaussians it is clear that

$$p(\boldsymbol{x}_b) = \mathcal{N}(\boldsymbol{H} \boldsymbol{\mu} + \boldsymbol{b}, \boldsymbol{L}^{-1} + \boldsymbol{H} \boldsymbol{A}^{-1} \boldsymbol{H}^T). \tag{66}$$

Now, making the substitutions

$$\boldsymbol{A}_{aa} \equiv \boldsymbol{A} + \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{H}, \quad \boldsymbol{\mu}_a \equiv \boldsymbol{\mu}, \quad \boldsymbol{\mu}_b \equiv \boldsymbol{H} \boldsymbol{\mu} + \boldsymbol{b}, \quad \boldsymbol{A}_{ab} \equiv -\boldsymbol{H}^T \boldsymbol{L} \tag{67}$$

we can use our expressions for the conditional Gaussian to find $p(\boldsymbol{x}_a | \boldsymbol{x}_b)$. Firstly, note that the posterior precision matrix is

$$\boldsymbol{A} + \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{H}. \tag{68}$$

Secondly, as

$$(\boldsymbol{A} + \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{H}) \boldsymbol{\mu} + \boldsymbol{H}^T \boldsymbol{L} (\boldsymbol{x}_b - \boldsymbol{H} \boldsymbol{\mu} - \boldsymbol{b}) = \boldsymbol{A} \boldsymbol{\mu} + \boldsymbol{H}^T \boldsymbol{L} (\boldsymbol{x}_b - \boldsymbol{b}) \tag{69}$$

then mean is

$$(\boldsymbol{A} + \boldsymbol{H}^T \boldsymbol{L} \boldsymbol{H})^{-1} \left(\boldsymbol{A} \boldsymbol{\mu} + \boldsymbol{H}^T \boldsymbol{L} (\boldsymbol{x}_b - \boldsymbol{b}) \right). \tag{70}$$

References

[1] Christopher M Bishop. *Pattern Recognition and Machine Learning*, volume 53. 2013.