Comparing Correlation Functions and Exploring Efficiency and Identifiability Issues for the Gaussian Process

Ivor Walker
Supervised by Dr Michail Papathomas

Abstract

Contents

1 Defining the Gaussian Process

1.1 Weight-space view [4]

1.1.1 Standard linear model

The standard linear model summises that we have some data-generating function $f(.)$ that linearily combines training data X and parameters of some model W to produce an output y:

$$\begin{aligned} y &= f(X) + \epsilon \\ f(X) &= X^T W \end{aligned} \tag{1}$$

We add a noise term ϵ because y is rarely a perfect observation of $f(X)$ (e.g. measurement error). The standard linear model assumes that ϵ is drawn from a Gaussian distribution $\epsilon \sim N(0, \sigma_n^2 I)$. We add a covariance matrix I to describe how the noise for one observation is related to the noise of another observation.

We can combine our expressions and assumptions for $f(X)$ and ϵ to produce a conditional distribution. Effectively a distribution of errors, this is the distribution from which y is drawn from after knowing perfectly X and W:

$$p(y|X, W) = \mathcal{N}(y|X^T W, \sigma_n^2 I)$$

1.1.2 Determining weights

Our first task is to find W as we typically do not know these in advance. Frequentist approaches focus on arriving at a single estimate of W ($\hat{W}$) via "maximum likelihood estimation" (MLE). $p(y|X, W)$ is at its highest density around the expected value $y|X^T W$ so we can use optimisation methods to find the $\hat{W}$ at the maximum of $p(y|X, W)$. Because we assume $E[p(y|X, W)] = 0$, the values of W at the maximum of $p(y|X, W)$ are also the values of W that pushes the squared error $||y - X^T W||^2$ closest to zero. We can communicate uncertainty surrounding $\hat{W}$ by computing "standard errors", or the ratio between the variance of our errors and the variance of X. A high variance in errors shows that $\hat{W}$ i, but a broader range of X makes it easier to estimate W.

Instead of producing point estimates for $\hat{W}$ and uncertainty, Bayesian statistics treats W as a random variable and specifies an expected value and a variance. Placing W in a probabilistic framework allows us to propagate uncertainty throughout the model and to encode beliefs (e.g. from domain experts) about the weights before observing the data.

We start with a "prior" distribution of W, which the Bayesian linear model assumes:

$$p(W) \sim N(0, \Sigma_p) \tag{2}$$

Then, we observe the data and update our beliefs about the weights using Bayes' theorem to produce a "posterior" distribution $p(W|X, y)$.

$$p(W|X, y) = \frac{p(y|X, W)p(W)}{p(y|X)}$$

$p(y|X, W)$ is the density of the residuals after applying $p(W)$ to X, W under our assumed noise model ϵ, and $p(y|X)$ is the marginal likelihood - how likely the data is given the model.

1.1.2.1 Deriving our posterior

To understand the relationship between $p(W|X, y)$ and W, we can ignore terms that do not vary with W (e.g. our marginal likelihood) by absorbing them into the proportionality constant:

$$p(W|X, y) \propto p(y|X, W)p(W) \tag{3}$$

TODO fix We can get a probability density function (PDF) for our error distribution by representing $Y|X^T W$ in squared error form and substituting it into the Gaussian PDF:

$$p(y|X, W) = exp\left(-\frac{1}{2\sigma_n^2}||y - X^T W||^2\right) \tag{4}$$

Reframing $p(W)$ as a PDF:

$$p(W) = \frac{1}{[\sqrt{\sigma_p}]\sqrt{2\pi}} exp\left(-\frac{1}{2}\frac{([W] - [0])}{[\Sigma_p]}\right)$$

The first term can be absorbed into the proportionality constant. Rewriting the second term as a negative exponential:

$$p(W) \propto exp\left(-\frac{1}{2}W^T \Sigma_p^{-1} W\right) \tag{5}$$

Putting both expressions for 4 and 5 into 3:

$$p(W|X, y) \propto exp\left(-\frac{1}{2\sigma_n^2}||y - X^T W||^2\right) exp\left(-\frac{1}{2}W^T \Sigma_p^{-1} W\right)$$

Expanding $||y - X^T W||^2$ to $y^T y - 2y^T XW + W^T X^T XW$:

$$p(W|X,y) \propto exp\left(-\frac{1}{2\sigma_n^2}(y^T y - 2y^T XW + W^T X^T XW)\right) exp\left(-\frac{1}{2}W^T \Sigma_p^{-1} W\right)$$

Putting both exponentials together by adding their powers:

$$p(W|X,y) \propto exp\left(\frac{1}{\sigma_n^2}(y^T y - 2y^T XW + W^T X^T XW) + \left(-\frac{1}{2}W^T \Sigma_p^{-1} W\right)\right)$$

Rearranging the inside term to be a quadratic, linear and constant term in W:

$$p(W|X,y) \propto exp\left(\frac{1}{2}W^T\left(\frac{1}{\sigma_n^2}X^T X + \Sigma_p^{-1}\right)W - \left(\frac{1}{\sigma_n^2}y^T X\right)W + \frac{1}{2}y^T y\right)$$

We can ignore the constant final term. Introducing these terms to simplify this result:

$$\begin{aligned} A &= \Sigma_p^{-1} + \frac{1}{\sigma_n^2}X^T X \\ b &= \frac{1}{\sigma_n^2}y^T X \\ p(W|X,y) &\propto exp\left(-\frac{1}{2}W^T AW + b^T W\right) \end{aligned} \tag{6}$$

1.1.2.2 Deriving the properties of the posterior by completing the square Now we have a simplified form of the posterior's PDF, we need to get it into a Gaussian form to recover the properties of the posterior distribution.

Bringing all terms inside the exponential to a single term:

$$-\frac{1}{2}W^T AW + b^T W = \frac{1}{2}\left(-W^T AW + 2b^T W\right)$$

Completing the square on our new inner term $W^T AW - 2b^T W$

$$W^T AW - 2b^T W = (W - A^{-1}b)^T A(W - A^{-1}b) - b^T A^{-1}bp(W|X,y) \propto exp\left(-\frac{1}{2}\left((W - A^{-1}b)^T A(W - A^{-1}b) - b^T A^{-1}b\right)\right) \tag{7}$$

Looking at the Gaussian PDF:

$$N(W|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}}exp\left(-\frac{1}{2}(W-\mu)^T\Sigma^{-1}(W-\mu)\right) \tag{8}$$

Our expression lines up with the Gaussian PDF's "kernel" term $exp\left(-\frac{1}{2}(W-\mu)^T\Sigma^{-1}(W-\mu)\right)$, where $\mu = A^{-1}b$ and $\Sigma = A^{-1}$ ($\Sigma^{-1} = A$). Therefore, 7 can be represented as a Gaussian distribution:

$$p(W|X,y) \sim N(A^{-1}b, A^{-1}) \tag{9}$$

Inside our definition of A at **??**, we are missing an expression for Σ_p. Assuming independence of noise under the linear model, our weight variance Σ_p under the Bayesian linear model is "isotropic", meaning it is the same in all directions.

$$\Sigma_p = \tau^2 I$$

Because we assume independence, I is an "identity matrix where each diagonal element is 1 and all off-diagonal elements are 0. τ^2 is a scalar variance term, chosen as a prior.

Substituting the isotropic prior Σ_p into A:

$$A = \Sigma_p^{-1} + \frac{1}{\sigma_n^2}X^T X = \left[\tau^2 I\right]^{-1} + \frac{1}{\sigma_n^2}X^T X = \frac{1}{\tau^2}I + \frac{1}{\sigma_n^2}X^T X$$

Simplifying:

$$A = \frac{1}{\sigma_n^2}\left(X^T X + \frac{\sigma_n^2}{\tau^2}I\right) \tag{10}$$

1.1.2.3 Gaussian posteriors and ridge regression So far we have worked exclusively within the Bayesian paradigm, but we can draw some value of W from our posterior distribution to relate it to a frequentist framework. For Gaussian posteriors, our expected value of W $A^{-1}b$ is also its mode. This is called the maximum a posteriori (MAP) estimate of W, and is due to symmetries in linear model and posterior and is not the case in general. Our MAP estimate does not matter within the Bayesian framework but is equivelant to our frequentist $\hat{W}$.

Substituting our full expressions for A 10 and b 6 into our MAP estimation:

$$W_{\text{MAP}} = A^{-1}b = \left[\frac{1}{\sigma_n^2}(X^TX + \frac{\sigma_n^2}{\tau^2}I\right]^{-1} \cdot \left[\frac{1}{\sigma_n^2}y^TX\right]$$

Inverting LHS term of A:

$$A^{-1} = \frac{\sigma_n^2}{X^TX + \frac{\sigma_n^2}{\tau^2}I} = \sigma_n^2\left(X^TX + \frac{\sigma_n^2}{\tau^2}I\right)^{-1}$$

Substituting this back into W_{MAP} cancels out the σ_n^2 term in A with the $\frac{1}{\sigma_n^2}$ term in B:

$$W_{\text{MAP}} = \sigma_n^2\left(X^TX + \frac{\sigma_n^2}{\tau^2}I\right)^{-1} \cdot \frac{1}{\sigma_n^2}y^TX = \left(X^TX + \frac{\sigma_n^2}{\tau^2}I\right)^{-1} \cdot y^TX$$

This is equivelant to the solution to ridge regression, where $\lambda = \frac{\sigma_n^2}{\tau^2}$.

$$W_{\text{ridge}} = \left(X^TX + \lambda I\right)^{-1}X^Ty$$

Ridge regression introduces some bias to lower the variance in a frequentist linear model by shrinking weights, where the λ term controls the amount of shrinkage applied to the weights. This is traditionally useful where variance is particularly high (e.g. multicollinearity) and can be reduced at the cost of little bias.

Our MAP estimation in Bayesian linear regression with isotropic priors is equivalent to ridge regression, where the amount of bias we introduce depends on our confidence in our priors. The more we trust our prior, the higher our λ and the more we shrink our weights towards zero. A lower τ means we are more confident in $p(W)$ and have better priors, whereas a higher σ_n means lower confidence in $p(y|X, W)$ and worse weights that should be shrunk closer to zero.

1.1.3 Predictive distribution

1.1.3.1 Deriving the predictive distribution Our second task is to make predictions y_* using new input data X_* and our previously learned weights W. Frequentist methods simply multiply $\hat{W}$ by X_*, but this does not propagate uncertainty in W. In this Bayesian framework, we form a "predictive distribution" which we sample from to get our noise-free function evaluations $f(X_*)$ (denoted f_*) and add ϵ to get our noisy predictions y_*.

$$p(f_*|X_*, X, y) = \int p(f_*|X_*, W) \cdot p(W|X, y)dW$$

$p(f_*|X_*, W)$ is what we think the function looks like after producing a prediction using X_* and perfect knowledge of W. $p(W|X, y)$ is our familiar 9 posterior distribution of weights. $p(f_*|X_*, W) \cdot p(W|X, y)$ is the joint distribution of our predictions and our posterior weights, which gets us the conditional distribution $p(f_*, W|X_*, X, y)$ by definition of conditional probability. Because $p(f_*, W|X_*, X, y)$ relies on our perfect knowledge of W, which we lack, we integrate over all possible W to get the final predictive distribution $p(f_*|X_*, X, y)$

$p(f_*|X_*, W)$ is our error distribution, which we assume to be distributed normally and independently with our I identity matrix:

$$p(f_*|X_*, W) = \mathcal{N}(f_*|W^TX_*, \sigma_n^2I)$$

Substituting into 8 and absorbing the LHS term into the proportionality constant:

$$p(f_*|X_*, w) \propto exp\left(-\frac{1}{2}\frac{1}{\sigma_n^2}(f_* - W^TX_*)^2\right)$$

Multiplying $P(f_*|X_*, W)$ and $p(W|X, y)$ to get our conditional $p(f_*, W|X_*, X, y)$, and add the exponents:

$$p(f_*, W|X_*, X, y) \propto exp\left(\frac{1}{2}(-W^TAW + 2b^TW) + \left(-\frac{1}{2}\frac{1}{\sigma_n^2}(f_* - W^TX_*)^2\right)\right)$$

Combining the terms inside the exponent:

$$p(f_*, W|X_*, X, y) \propto exp\left(-\frac{1}{2}\left(W^TAW - 2b^TW + \frac{1}{\sigma_n^2}(f_* - W^TX_*)^2\right)\right)$$

Expanding the squared term:

$$p(f_*, W|X_*, X, y) \propto exp\left(-\frac{1}{2}\left(W^TAW - 2b^TW + \frac{1}{\sigma_n^2}(f_*^2 - 2f_*W^TX_* + W^TX_*X_*^TX_*)\right)\right)$$

Similar to our posterior, we can rearrange this to be a quadratic, linear and constant term in W:

$$p(f_*, W|X_*, X, y) \propto exp\left(-\frac{1}{2}\left(W^T\left(A + \frac{1}{\sigma_n^2}X_*X_*^T\right)W - 2\left(b + \frac{1}{\sigma_n^2}f_*X_*\right)^TW + \frac{1}{\sigma_n^2}f_*^2\right)\right) \tag{11}$$

We can define new terms A_* and b_* to simplify this expression:

$$A_* = A + \frac{1}{\sigma_n^2} X_* X_*^T$$
$$b_* = b + \frac{1}{\sigma_n^2} f_* X_*$$

Substituting into 11:

$$p(f_*, W|X_*, X, y) \propto exp\left(-\frac{1}{2}\left(W^T A_* W - 2b_*^T W + \frac{1}{\sigma_n^2} f_*^2\right)\right)$$

Integrating out W to get our predictive distribution:

$$p(f_*|X_*, X, y) = \int p(f_*, W|X_*, X, y) dW \propto \int exp\left(-\frac{1}{2}\left(W^T A_* W - 2b_*^T W + \frac{1}{\sigma_n^2} f_*^2\right)\right) dW \tag{12}$$

Factoring out $\frac{1}{\sigma_n^2} f_*^2$ as it does not depend on W (since $\int exp(X)dX = exp(X)$):

$$= exp\left(-\frac{1}{2}\frac{1}{\sigma_n^2} f_*^2\right) \times \int exp\left(-\frac{1}{2}\left(W^T A_* W - 2b_*^T W\right)\right) dW$$

Evaluating the RHS multivariate Gaussian integral:

$$\int exp\left(-\frac{1}{2}\left(W^T A_* W - 2b_*^T W\right)\right) dW = \frac{(2\pi)^{D/2}}{\sqrt{|A_*|}} exp\left(\frac{1}{2} b_*^T A_*^{-1} b_*\right)$$

Substituting back into 12:

$$p(f_*|X_*, X, y) \propto exp\left(-\frac{1}{2}\frac{1}{\sigma_n^2} f_*^2\right) + \frac{(2\pi)^{D/2}}{\sqrt{|A_*|}} \cdot exp\left(\frac{1}{2} b_*^T A_*^{-1} b_*\right)$$

Now that no part of our expression is dependent on W, we need an expression for everything that depends on f_*.

Absorbing the second term (since it does not depend on f_*) into the proportionality constant, and combining the remaining exponential terms by adding their powers:

$$p(f_*|X_*, X, y) \propto \exp\left(-\frac{1}{2}\frac{1}{\sigma_n^2} f_*^2 + \frac{1}{2} b_*^T A_*^{-1} b_*\right)$$

Similar to deriving properties from our posterior, we can rearrange this expression and complete the square to derive the properties of our predictive distribution:

$$p(f_*|X_*, W) \sim N(X_*^T A^{-1} b, X_*^T A^{-1} X_*) \tag{13}$$

The variance is quadratic in X_* with A^{-1}, showing that predictive uncertainties grow with size of X_*.

1.1.4 Projections of inputs into feature space

One problem with this model is that it assumes a linear relationship between X and y. We can project our inputs into a higher dimensional feature space and apply a linear model in this space to express non-linear relationships between X and y.

Defining $\phi(X)$ as a function that maps a D-dimensional input vector X into an N dimensional feature space, our standard linear model becomes:

$$f(X) = \phi(X)^T W$$

For example, a scalar x could be projected into the space of powers of x: $\phi(x) = [1, x, x^2, \ldots, x^d]^T$ for a polynomial basis expansion of degree d to represent a d-power relationship between x and y. . Substituting $\phi(X)$ for X in 13:

$$p(f_*|X_*, X, y) = N(\phi(X_*)^T A_\phi^{-1} b_\phi, \phi(X_*)^T A_\phi^{-1} \phi(X_*)) \tag{14}$$

Where A_ϕ and b_ϕ are now:

$$A_\phi = \Sigma_p^{-1} + \frac{1}{\sigma_n^2}\phi(X)^T\phi(X) b_\phi = \frac{1}{\sigma_n^2}\phi(X)^T y$$

1.1.5 Computational issues

1.1.5.1 Avoiding inversion of A_ϕ 14 requires inverting the $N \times N$ matrix A_ϕ, where N is dimension of feature space, to get the expected value and variance.

Typically, matrices are inverted using Gaussian elimination. We need to perform a "forward" pass which requires N pivots on every row and column, N eliminations per pivot, and up to $2N$ columns to update, resulting in an $O(N^3)$ time complexity. Then, we need to perform a backwards pass in the opposite direction which is another $O(N^3)$ operation. Finally, we need to multiply the inverse by the RHS vector b_ϕ, which is an $O(N^2)$ operation but appears trivial next to these two cubic steps.

We can mitigate this for a particular class of high-dimensional $N > n$ problems by restating the predictive distribution in terms of the number of training data points n which would require inverting an $n \times n$ matrix instead. For polynomial basis expansions, N is degree D multiplied by number of features, so N can be very large or even infinite (e.g. SE).

Substituting b_ϕ into our predictive distribution mean:

$$\mathbb{E}_{p(f_*|X_*,X,y)}[f_*] = \phi(X_*)^T \cdot A_\phi^{-1} \cdot \left[\frac{1}{\sigma_n^2}\phi(X)^T y\right]$$

Rearranging to isolate $A_\phi^{-1}\phi(X)$

$$= \frac{1}{\sigma_n^2}\left[A_\phi^{-1}\phi(X)\right]^T y$$

We can use the Sherman-Morrison identity to get an expression for A_ϕ^{-1} directly, where $K = \phi(X)^T\Sigma_p\phi(X)$

$$A_\phi^{-1} = \Sigma_p - \Sigma_p\phi(X)(K + \sigma_n^2 I)^{-1}\phi(X)^T\Sigma_p$$

For the mean, we can use the Sherman-Morrison identity again to get an expression for $A_\phi^{-1}\phi(X)$

$$A_\phi^{-1}\phi(X) = \sigma_n^2\Sigma_p\phi(X)(K + \sigma_n^2 I)^{-1} \tag{15}$$

Substitute in 15 into our 14:

$$\mathbb{E}_{p(f_*|X_*,X,y)}[f_*] = \phi(X_*)\frac{1}{\sigma_n^2}\left[\sigma_n^2\Sigma_p\phi(X)(K + \sigma_n^2 I)^{-1}\right]^T y$$

$\frac{1}{\sigma_n^2}$ and σ_n^2 cancel out, leaving us with this final expression for the mean:

$$\mathbb{E}_{p(f_*|X_*,X,y)}[f_*] = \phi(X_*)^T \cdot \Sigma_p\phi(X)(K + \sigma_n^2 I)^{-1}y \tag{16}$$

For the variance, we cannot use the Sherman-Morrison identity to arrive at an expression for $A_\phi^{-1}\phi(X_*)$ because $\phi(X_*)$ is an arbitrary N-vector, not one of the columns of $\phi(X)$. Instead, we use the A_ϕ^{-1} expression we derived earlier to get an expression for $A_\phi^{-1}\phi(X_*)$:

$$A_\phi^{-1}\phi(X_*) = \Sigma_p \cdot \phi(X_*) - \Sigma_p\phi(X)(K + \sigma_n^2 I)^{-1}\phi(X)^T\Sigma_p \cdot \phi(X_*)$$

Substituting this into 14:

$$\mathrm{Var}_{p(f_*|X_*,X,y)}[f_*] = \phi(X_*)^T\Sigma_p\phi(X_*) - \phi(X_*)^T\Sigma_p\phi(X)(K + \sigma_n^2 I)^{-1}\phi(X)^T\Sigma_p\phi(X_*) \tag{17}$$

With our alternative mean 16 and variance 17, we can form an alternative expression for our predictive distribution:

$$\begin{aligned} p(f_*|X_*, X, y) = \mathcal{N}(\\ \phi(X_*)^T\Sigma_p\phi(X)(K + \sigma_n^2 I)^{-1}y, \\ \phi(X_*)^T\Sigma_p\phi(X_*) - \phi(X_*)^T\Sigma_p\phi(X)(K + \sigma_n^2 I)^{-1}\phi(X)^T\Sigma_p\phi(X_*) \\ ) \end{aligned} \tag{18}$$

With this alternative formulation, we need to invert the $n \times n$ matrix $K + \sigma_n^2 I$ only. Geometrically, n datapoints can span at most n dimensions in the feature space - if $N > n$, the data forms a subspace of the feature space.

1.1.5.2 Kernels and the kernel trick In 18, $\phi(.)$ is always an inner product of a positive definite correlation matrix Σ_p, but with different arrangements of $\phi(X)$ and $\phi(X_*)$. We can define $k(X, X') = \phi(X)^T\Sigma_p\phi(X')$ as a covariance function or kernel, where X and X' are either X or X_*. For example, in 18 the definition of $K = \phi(X)^T\Sigma_p\phi(X)$ becomes $K = k(X, X)$.

Introducing $\psi(X)$ to better represent $k(X, X')$ as an inner product:

$$\psi(X) = \phi(X)\Sigma_p^{1/2}$$
$$k(X, X') = \psi(X)^T\psi(X')$$

These inner product representations require us to compute $\phi(X)$ and $\phi(X')$ in the feature space. A higher-dimensional feature space requires more compute to evaluate $\phi(X)$ and more memory to store $\phi(X)$ and $\phi(X')$.

Instead, the representer theorem guarantees that we can find an equivelant kernel that does not require us to explicitly compute $\phi(X)$ or $\phi(X')$ in the feature space. With this "kernel trick" we avoid the associated memory and computational costs of explicitly computing $\phi(X)$ and $\phi(X')$. Since computing the kernel directly is more convenient than the feature vectors themselves, these kernels become the object of primary interest.

For example, if we had some polynomial transformation $\phi(X) = [1, x^1, ..., x^D]^T$ and Σ_p as an identity matrix, we could define $k(X, X')$ as inner products:

$$\psi(X) = [1, x^1, ..., x^D]^T k(X, X') = \psi(X)^T \psi(X')$$

This approach requires arranging ϕ and $\phi(X')$ into a D sized vector, then taking the dot product. This is trivial for small D, but as D becomes infinite (e.g. RBF kernel), arranging a D sized vector requires too much memory and the dot product becomes computationally expensive.

Instead, we can define $k(X, X')$ as an equivelant function of X and X' directly:

$$k(X, X') = (1 + X \cdot X')^D$$

This is the polynomial kernel, which is equivalent to our original polynomial basis expansion $\phi(X)$ without explicitly computing $\phi(X)$.

1.2 Function-space view [4]

1.2.1 Gaussian processes (GP)

1.2.1.1 Bayesian linear model We can define our Bayesian linear model of a real process $f(X)$ entirely in terms of mean function $m(X)$ and covariance function $k(X, X')$:

$$\begin{aligned} m(X) &= \phi(X)^T \mathbb{E}[W] = \phi(X)^T [0] = 0 \\ k(X, X') &= \phi(X)^T \mathbb{E}[WW^T] \phi(X') = \phi(X)^T \Sigma_P \phi(X') \end{aligned}$$

Our covariance function here is in inner product form. The kernel trick here uses the squared exponential (SE) covariance function, also known as the radial basis function (RBF) or Gaussian kernel:

$$k(f(X), f(X')) = \exp\left(-\frac{1}{2}\frac{|X - X'|^2}{l^2}\right)$$

It can be shown that SE corresponds to a Bayesian linear regression model with infinite basis functions.

1.2.1.2 Function evaluations to a random function We can choose a subset X_{*1} from our test data X_* and apply it to our model get some function evaluations $f(X_{*1})$. $f(X_{*1})$ can be described as a multivariate Gaussian distribution, e.g. in the Bayesian linear model $f(X_{*1}) \sim N(0, k(X_{*1}, X_{*1}))$. Each output $f(X_{\theta*1})$ in our $f(X_{*1})$ vector is a random variable with mean 0 and covariance with each other $K_{\theta\theta'} = k(X_{\theta*}, X_{\theta'*})$. There exists some random function $g(X_{*1})$ for our subsets such that $f(X_{*1}) = g(X_{*1})$. We only know the value of $g(X_{*1})$ at the points X_{*1}, so $g(X_{*1}) = X_{*1} : f(X_{*1})$. Because $g(X)$ entirely consists of random points, we can think of $g(X_{*1})$ as a random function and our distribution $f(X)$ can be seen as a distribution of these random $g(X)$ functions. We can recover our individual $g(X_{*1}$ thanks to consistency - if we marginalised out our subset from the entire distribution $f(X_*)$, we would recover the subset distribution $N(0, K_*(X_{*1}, X_{*1}))$ that describes our random function $g(X_{*1})$.

1.2.1.3 Definition of a GP A GP is a collection of random variables, any finite number of which have a joint Gaussian distribution. Ultimately, GPs describe a distribution of random functions where each drawn function is a $g(X)$ sample from the GP.

$$\begin{aligned} f(X) &\sim \mathcal{GP}(m(X), k(X, X')), \\ m(X) &= \mathbb{E}[f(X)], \\ k(X, X') = \text{Cov}(f(X), f(X')) &= \mathbb{E}[(f(X) - m(X))(f(X') - m(X'))] \end{aligned}$$

1.2.1.4 Consistency requirement This definition implies a consistency requirement - any group of functions drawn from our GP can be described by the same distribution as our GP. For example, if our GP implies that $(f(X_1), f(X_2)) \sim \mathcal{N}(\mu, \Sigma)$, then $(f(X_1) \sim \mathcal{N}(\mu_1, \Sigma) and f(X_2) \sim \mathcal{N}(\mu_2, \Sigma))$ where $\mu_\theta = m(X_\theta)$ and $\Sigma_{\theta\theta} = k(X_\theta, X_\theta)$. This requirement is also called the marginalisation property, because to get the smaller distribution of $f(X_1)$ we marginalise out the larger distribution of $f(X_1), f(X_2)$ by integrating the larger distribution wrt $f(X_2)$. Consistency is automatically gained if our covariance function specifies entries in a covariance matrix.

1.2.2 Predictive distributions with noise-free observations

1.2.2.1 Prior distribution over functions $f(X)$ and $f(X_*)$ are jointly distributed according to the prior:

$$\begin{pmatrix} f(X) \\ f(X_*) \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{pmatrix}\right)$$

1.2.2.2 Posterior distribution of functions To get the posterior distribution of functions given the training data and our prior, we can condition the joint prior distribution on the training data. Intuitively, this is like generating random functions $g(X)$ and rejecting those that do not pass through the training data. Probabilistically, we condition our joint Gaussian prior distribution on the observations $p(f(X_*)|X_*, X, f(X))$.

Substituting $p(W)$ and our conditioning X into the Gaussian multivariate conditioning identity:

$$\begin{aligned} p(f(X_*)|X_*, X, f(X)) \sim N(\\ [0] + [K(X_*,X)][K(X,X)]^{-1}([f(X)] - [0]), \\ [K(X*,X*)] - [K(X_*,X)][K(X,X)]^{-1}[K(X,X_*)] \\ ) \end{aligned}$$

Although we condition on X_*, X, and $f(X)$, we only substitute $f(X)$ because X_* and X are known constants, but $f(X)$ is random because it is a sample from the prior. We also swap $f(X_*)$ and $f(X)$ in our prior to match the conditioning identity, such that our input vector into the conditioning identity is $(f(X_*), f(X))^T$.

Simplifying the last term in the mean:

$$\begin{aligned} p(f(X_*)|X_*, X, f(X)) \sim N(\\ K(X,X_*)K(X,X)^{-1}f(X), \\ K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*) \\ ) \end{aligned} \tag{19}$$

1.2.3 Predictive distributions with noisy observations

1.2.3.1 Noisy observations prior It is typical to not have the noise-free function evaluations $f(X)$ as our training data, but instead our noisy observations y. We can simply add ϵ:

$$\text{Cov}(y_p, y_q) = K(X_p, X_q) + \sigma_n^2 \delta_{pq}$$

δ_{pq} represents our independence condition in 1D. This is the Kronecker delta, which returns 1 if indices (p, q) are equal and 0 otherwise. σ_n^2 is the noise variance, which is a constant for all observations.

In matrix form:

$$\text{Cov}(Y) = k(X,X) + \sigma_n^2 I$$

This gives us this prior:

$$\begin{pmatrix} Y \\ f(X_*) \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} K(X,X) + \sigma_n^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{pmatrix}\right)$$

1.2.3.2 Noisy observations posterior As before, we can form a predictive distribution using the Gaussian multivariate conditioning identity:

$$\begin{aligned} p(f(X_*)|X_*, X, Y) \sim N(\\ K(X_*,X)[K(X,X) + \sigma_n^2 I]^{-1}Y, \\ K(X_*,X_*) - K(X_*,X)[K(X,X) + \sigma_n^2 I]^{-1}K(X,X_*) \\ ) \end{aligned}$$

Substituting $k(X,X') = \phi(X)^T \Sigma_p \phi(X')$ into here gives us the exact same result as 18.

Our variance is independent of the targets y and only depends on our inputs X and X_*. Our variance is two terms: our prior covariance $K(X_*,X_*)$, a term representing the information the observations give us about the function. As before, we can compute the predictive distribution of y_* by adding the noise term $\sigma_n^2 I$ to the variance.

1.2.4 Marginal likelihood

We need some measure of how well our GP fits the data, which we can get by computing the marginal likelihood $p(Y|X)$:

$$p(Y|X) = \int p(Y|f,X)p(f|X)df$$

$p(y|f,X)$ is our familar predictive distribution $p(y|f,X) \sim N(f, \sigma_n^2 I)$, and represents how well f maps X to y. $p(f|X)$ is our prior distribution over weights $\sim N(0,K)$ which we use here to represent the complexity of f. Our weight's mean

will always be the same under our prior, but our $K(X, X')$ tells us how "wiggly" our function is. The closer our $p(f|X)$ distribution is to the true complexity of the function, the higher our marginal likelihood. For example, for SE if our data is close together, then $|X - X'|$ becomes small and our covariance $k(X, X')$ on our function distribution prior is large. Therefore, we get a high variety of functions and a higher probability of sampling a more complex function.

We can express $p(Y|X)$ as a Gaussian integral over the joint distribution of f and Y ($p(Y, f|X)$), and marginalise out f to get this PDF:

$$logp(Y|X) = -\frac{1}{2}Y^T(K + \sigma_n^2 I)^{-1}Y - \frac{1}{2}log|K + \sigma_n^2 I| - \frac{n}{2}log(2\pi) \tag{20}$$

Alternatively, from 1 we know that y is Gaussian. Since both y and f are Gaussian, we can simply add their means and variances:

$$p(Y|X) = N(0, K + \sigma_n^2 I)$$

We can plug these mean and variances into Gaussian PDF 8 to get 20.

1.2.5 Algorithm for predictive distribution

1. Take in inputs X, outputs y, covariance function k, noise level σ_n^2, and test input X_*
2. $L = \text{cholesky}(K(X, X) + \sigma_n^2 I)$
   - Invert our $[K(X, X) + \sigma_n^2 I]$ matrix needed for mean and variance using Cholesky decomposition
3. $alpha = L^T \backslash (L \backslash y)$
   - Prepare the mean of our predictive distribution in linear combination form by computing the α vector
4. $mu = K(X_*, X)^T \cdot alpha$
   - Compute the mean
5. $v = L \backslash K(X_*, X)^T$
   - Prepare to compute variance by computing v, the form in which L is used in the variance
6. $var = K(X_*, X_*) - v^T v$
   - Compute the variance
7. $\log p(Y|X) = -\frac{1}{2}y^T \cdot alpha - \frac{1}{2}log|K(X, X) + \sigma_n^2 I| - \frac{n}{2}log(2\pi)$
   - Compute the log marginal likelihood
8. Return the mean mu, variance var, and log marginal likelihood

TODO Choelsky decomposition if needed, missing background

1.3 Varying the hyperparameters [4]

Our covariance functions has some hyperparameters, e.g. the full form of SE in one dimension contains some free parameters σ_f^2, σ_n^2, and l:

$$k_y(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2}\frac{|x - x'|^2}{l^2}\right) + \sigma_n^2 \delta_{x,x'}$$

Note that our covariance function is for k_y as it is for the noisy targets y, not the function f. σ_f^2 is the signal variance, which controls the overall scale of the function. σ_n^2 is the noise variance, which controls the amount of noise in the observations. $\delta_{X,X'}$ is the Kronecker delta which represents our independence of noise assumption.

l is a "length scale" hyperparameter that controls how sensitive our functions are - if we specify a lower l, we can "artificially" get a high $k(X, X')$. One way to determine l is by the expected number of "upcrossings" that our kernel is expected to make for a given level u. A function performs an upcrossing for u when $u = f(x)$ and $dy/dx > 0$. For example, with $u = 2$ and $y = x^2$, there exists one upcrossing at $(2, \sqrt{2})$ where $dy/dx = 4$, and a downcrossing at $(2, -\sqrt{2})$ where $dy/dx = -4$. For our zero-mean Gaussian processes, the expected number of upcrossings of our (stationary) kernel for a level $0 < u < 1$ is:

$$\mathbb{E}[N_u] = \frac{1}{2\pi}\sqrt{\frac{-k''(0)}{k(0)}}\exp\left(-\frac{u^2}{2k(0)}\right)$$

We can empirically count the number of upcrossings between 0 and 1 and set this equal to the expected number of upcrossings to get a value for l:

$$\frac{1}{2\pi}\sqrt{\frac{-k''(0)}{k(0)}}\exp\left(-\frac{u^2}{2k(0)}\right) = \hat{N}_u \tag{21}$$

A large amount of upcrossings implies our data-generating function wiggles rapidly, so our l becomes smaller to produce a covariance function that in turn produces more flexible functions.

1.4 Smoothing and equivelant kernels [4]

TODO, missing background

1.4.1 Linear predictors and smoothers

Understanding how our predictive distribution's mean varies with its inputs is difficult because of the first term $[(K(X,X)+\sigma_n^2 I)]^-1$: it is dependent on exact values of X, and it requires the inversion of $K(X,X)+\sigma_n^2 I$. Instead, we can reformulate $E_{p(f(X_*)|X_*,X,Y)}[f(X_*)]$ as an "equivelant kernel" to remove the dependence on X. which uses a kernel smoother (also known as the Nadaraya-Watson estimator)

Firstly, we rewrite our mean such that our mean function is a "linear smoother", or a function of the covariance function and our training data labels Y:

$$\begin{aligned} H(X_*) &= [K(X,X)+\sigma_n^2 I]^{-1} K(X_*,X) \\ \mathbb{E}_{y_*|X_*,X,y)}[f(X_*)] &= H(X_*)^T Y \end{aligned} \tag{22}$$

This vector of functions $H(X_*)$ is called a weight function and contains our problematic term.

Instead of inputting the values of x_i directly into the kernel function, we can use x_i centred around X_*:

$$k_i = k(|x_i - x_{*i}|/l)$$

Where l is some length scale hyperparameter.

TODO explain what we gain from this

Thus, our predictive distribution mean becomes:

$$\mathbb{E}_{p(f(X_*)|X_*,X,Y)}[f(X_*)] = \sum_{i=1}^{n} w_i y_i$$

where $w_i = k_i / \sum_{j=1}^{n} k_j$ is a TODO.

2 Exploring Covariance Functions [4]

2.1 Characteristics of covariance functions [4]

2.1.1 Stationarity and isotropicism

A stationary covariance function is a function of $X - X'$ only, and is invariant to the exact locations of X and X'. An isotropic covariance function is a function only of $|X - X'|$, and is is invariant to the direction of $X - X'$. For example, SE [**eq:se**] is both stationary and isotropic because it is a function of $|X - X'|$ only.

2.1.2 Symmetry and positive semidefiniteness

Given a vector of input points $X_i|i = 1, ..., n$, the Gram matrix K is the $n \times n$ matrix whose (i, j)-th entry is the inner product between X_i and X_j. Since our covariance matrix can be represented as inner products of our vectors of inputs, we can represent it as a Gram matrix. The Gram matrix has two key properties, symmetry and positive semidefiniteness:

$$K_{ij} = K_{ji}$$
$$X^T K X \geq 0$$

TODO prove positive semidefiniteness, background needed

2.1.3 Mean square continuity and differentiability

To understand how smooth the functions drawn from a Gaussian process are, we need to understand how differentiable and continuous they are. A more differentiable function implies that the function contains higher order polynomials which makes it smoother, and a continuous function avoids any reductions in smoothness produced by discontinuities.

Because the functions drawn from the Gaussian distribution are random functions between datapoints, there are infinitely many possible functions and determining if they are all continuous or differentiable is impossible. Instead, we can examine the covariance functiondifferentiable and continuous, and square these results to enable a direct comment about the smoothness of the functions drawn from the Gaussian process.

2.1.3.1 Continuity A Gaussian process $f(X)$ is continuous in mean square at X_* if, as $k \to \infty$:

$$\mathbb{E}[|f(X_k) - f(X_*)|^2] \to \infty$$

A Gaussian process is continuous at X_* if and only if its covariance function is continuous at X_*. For stationary covariance functions this involves checking $k(0, 0)$ only.

2.1.3.2 Differentiability TODO

2.2 Stationary covariance functions [4]

2.2.1 Spectral density for stationary processes

TODO, background needed

2.2.2 Squared exponential (SE)

Here is the already introduced SE:

$$k(X, X*) = \exp\left(-\frac{|X - X'|^2}{2l^2}\right)$$

We can find the value for l analytically using 21:

$$l = \frac{1}{2\pi\hat{N}_u} \exp\left(-\frac{u^2}{2\sigma^2}\right)$$

Setting $u = 0$ makes our term inside the exponential equal to zero:

$$l = \frac{1}{2\pi\hat{N}_0}$$

This covariance function is infinitely differentiable thanks to the exp term, so a GP using SE is infinitely mean-squared differentiable, which produces very smooth functions.

TODO proof of infinite basis functions, background needed

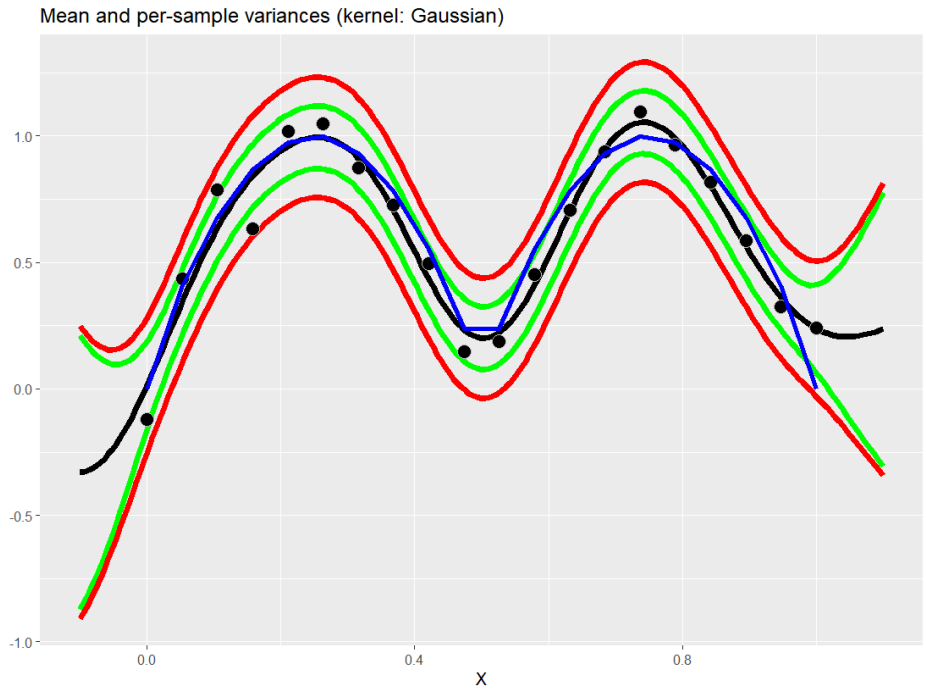


Figure 1: Plot of a Gaussian process using SE applied to a toy dataset. The toy dataset ($n = 15$) is a data-generating function in blue with some Gaussian noise applied to produce the datapoints in black. The black line represents the expected function from the Gaussian process. The green line represents the 90% confidence interval around the predictive distribution without the σ_n^2 term, representing the uncertainty surrounding predictions of the noise-free mean function $f(X)$. The red line represents the 90% confidence interval with σ_n^2, representing the uncertainty surrounding predictions of the noisy observations y.

2.2.3 Matern-class

The Matern class of covariance functions is given by:

$$k(X, X') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}|X - X'|}{l} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}|X - X'|}{l} \right)$$

l is our familiar length scale hyperparameter, but ν controls how differentiable the function is.

TODO Bessel function K_ν, background needed Therefore, a a Gaussian process using a Matern class kernel is k-times MS differentiable if and only if $\nu > k$.

We can simplify this by using half-integers, i.e. $\nu = p + 1/2$ where p is a non-negative integer. In this case, the covariance function becomes a product of a polynomial and an exponential:

$$k_{\nu=p+1/2}(X, X') = \exp\left(-\frac{\sqrt{2\nu}|X - X'|}{l} \right) \frac{\Gamma(p+1)}{\Gamma(2p+1)} \sum_{i=0}^{p} \frac{(p+i)!}{i!(p-i)!} \left(\frac{\sqrt{8\nu r}}{l} \right)^{p-i}$$

$\nu = 1/2$ is equivelant to the exponential covariance function.

TODO formula

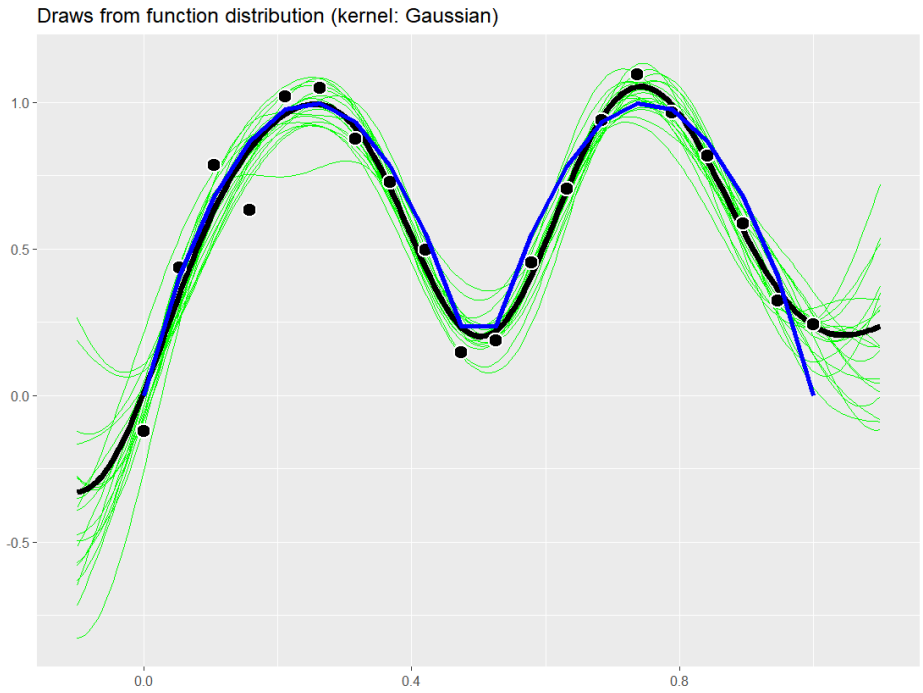
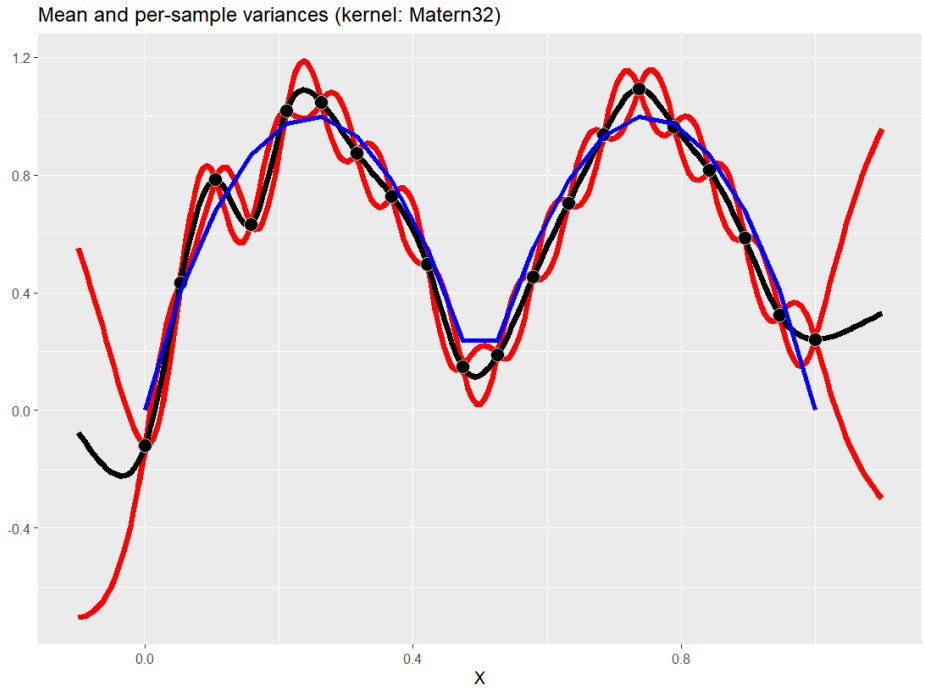
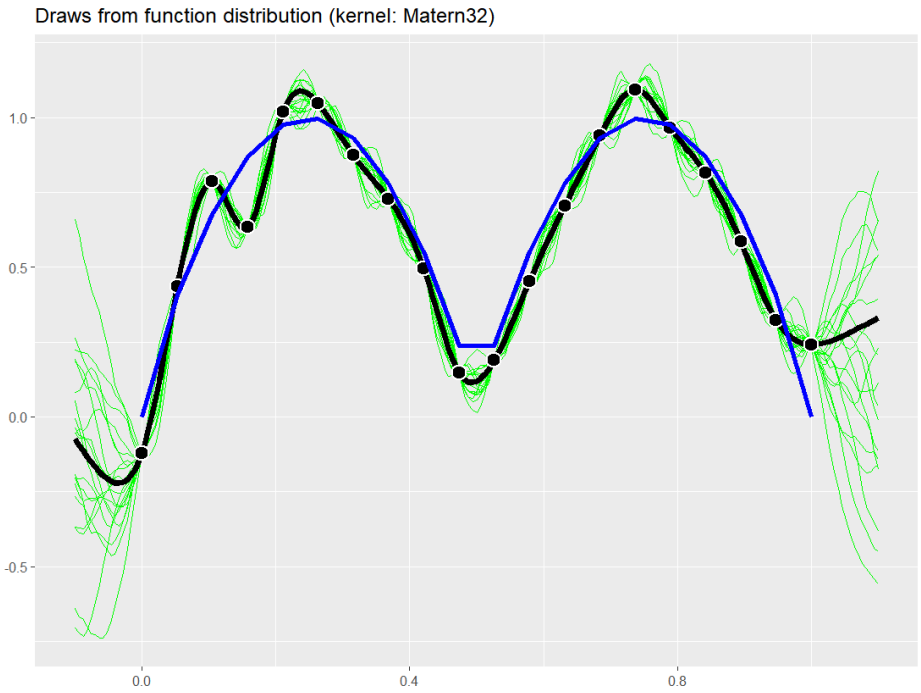
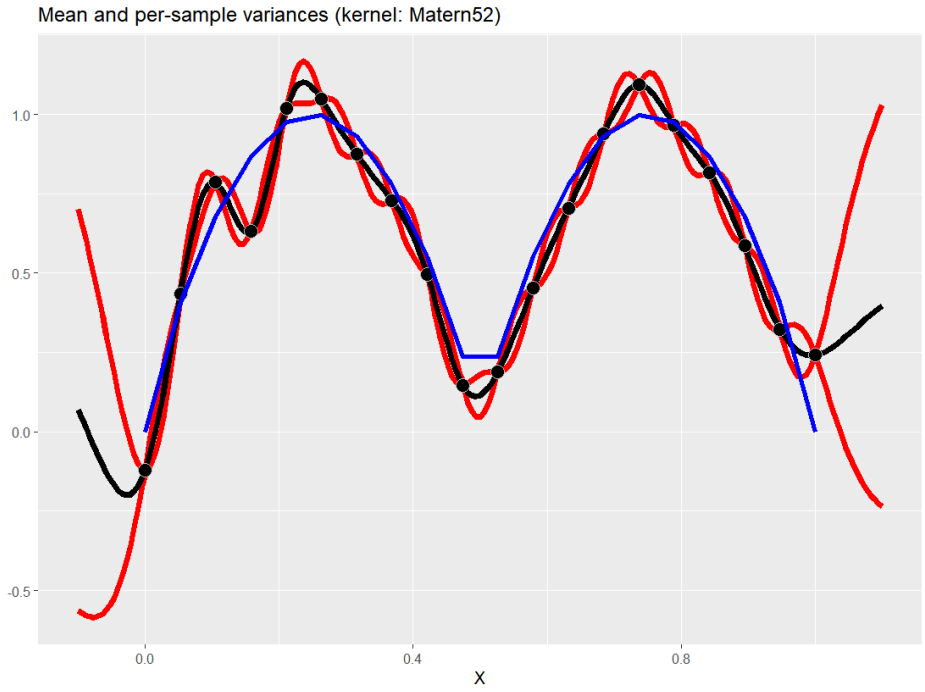
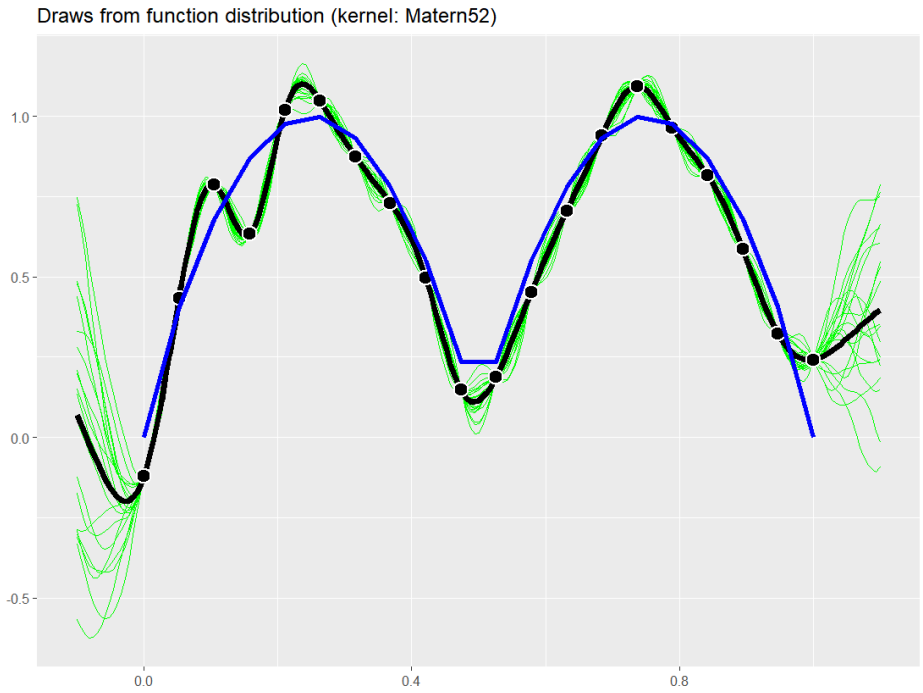


Figure 2: Plots of functions from a Gaussian process using SE applied to the same toy dataset. The blue line and black datapoints and lines are as before, but the green lines here are a sample of functions drawn from the Gaussian process.



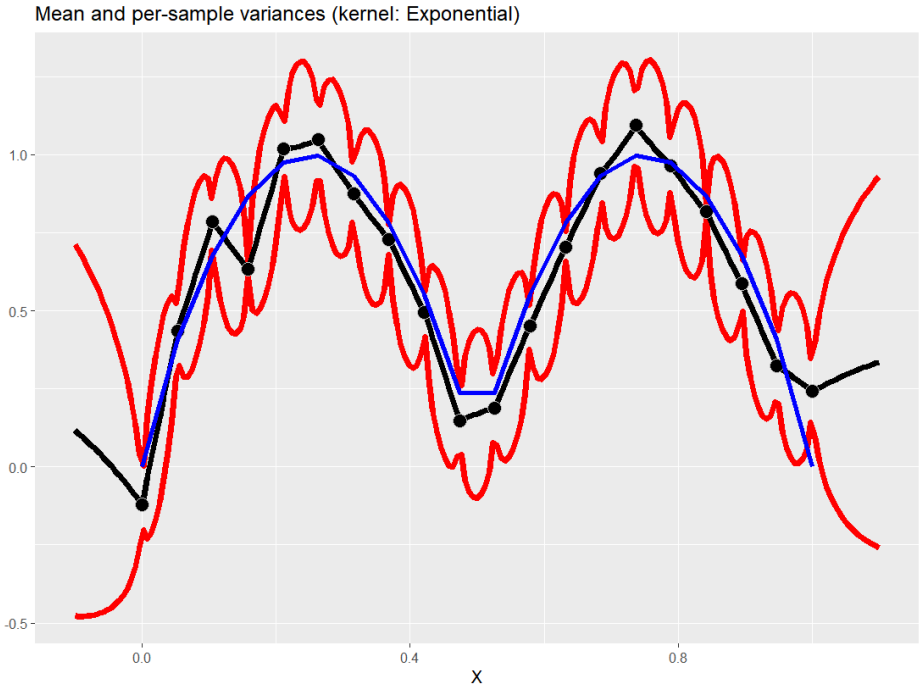

Draws from function distribution (kernel: Matern32)
1.0
0.5
0.0
-0.5
0.0
0.4
0.8

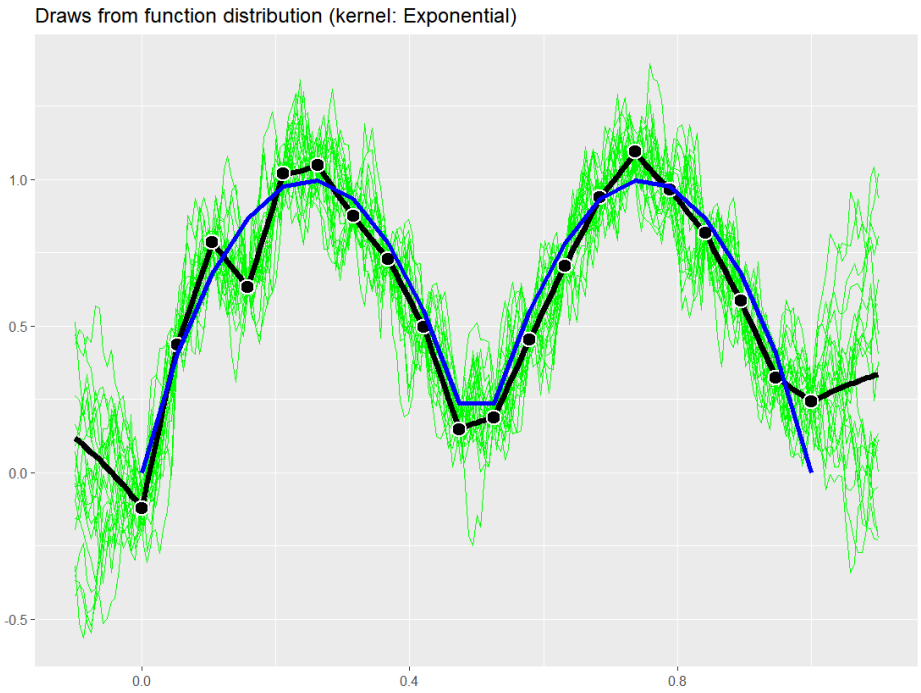

Mean and per-sample variances (kernel: Matern52)
1.0
0.5
0.0
-0.5
0.0
0.4
0.8
X

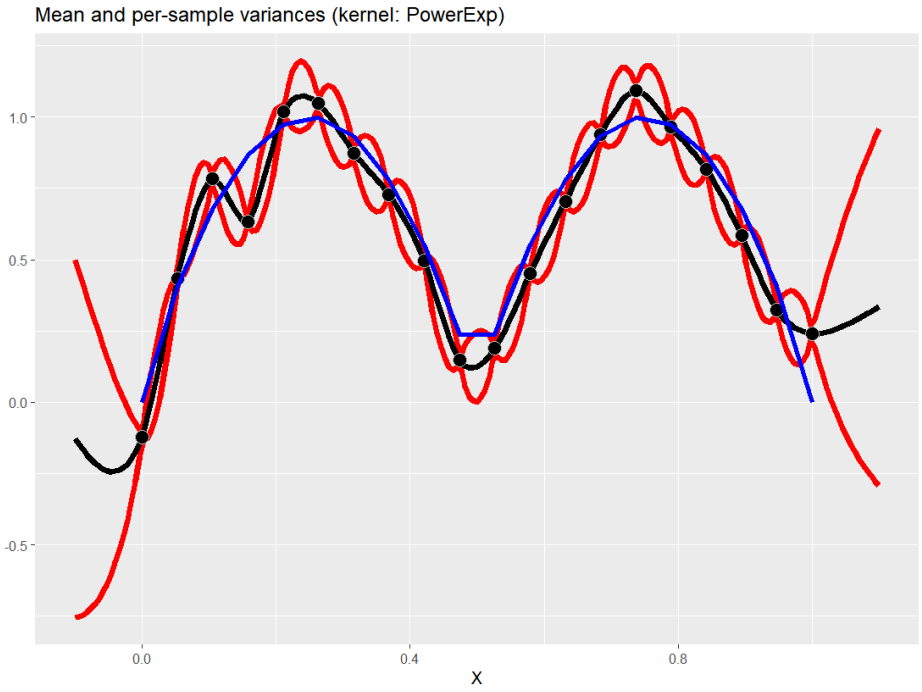


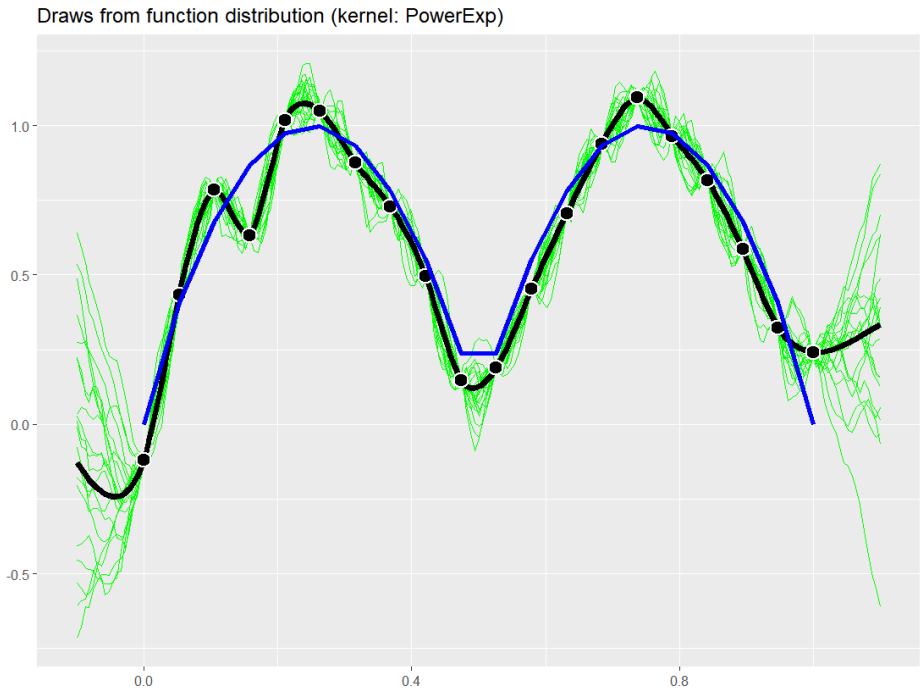
2.2.4 Exponential and γ-exponential

TODO formulas




Draws from function distribution (kernel: Exponential)
1.0
0.5
0.0
-0.5
0.0
0.4
0.8

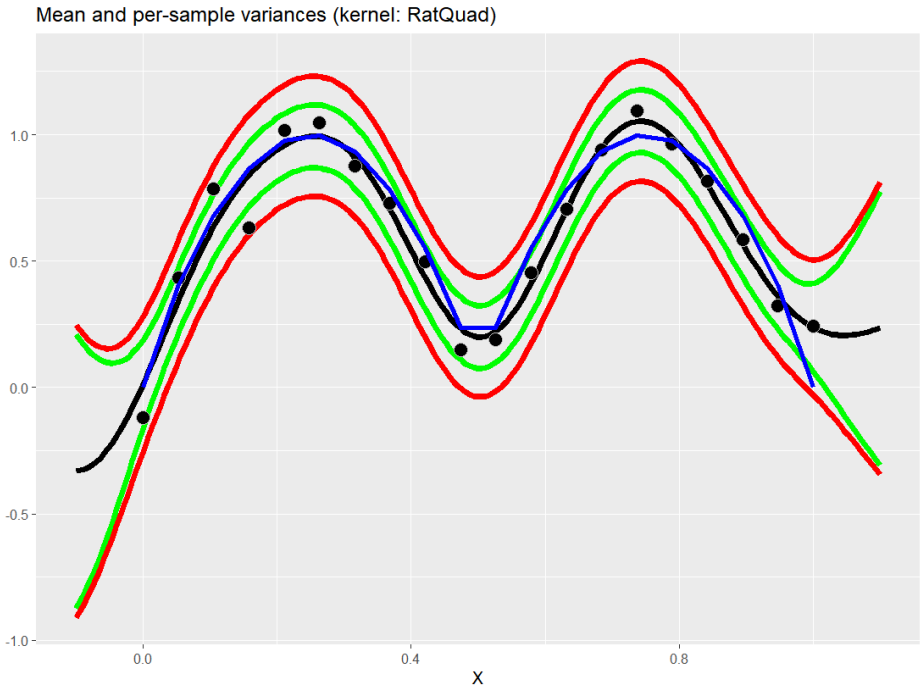

Mean and per-sample variances (kernel: PowerExp)
1.0
0.5
0.0
-0.5
0.0
0.4
0.8
X

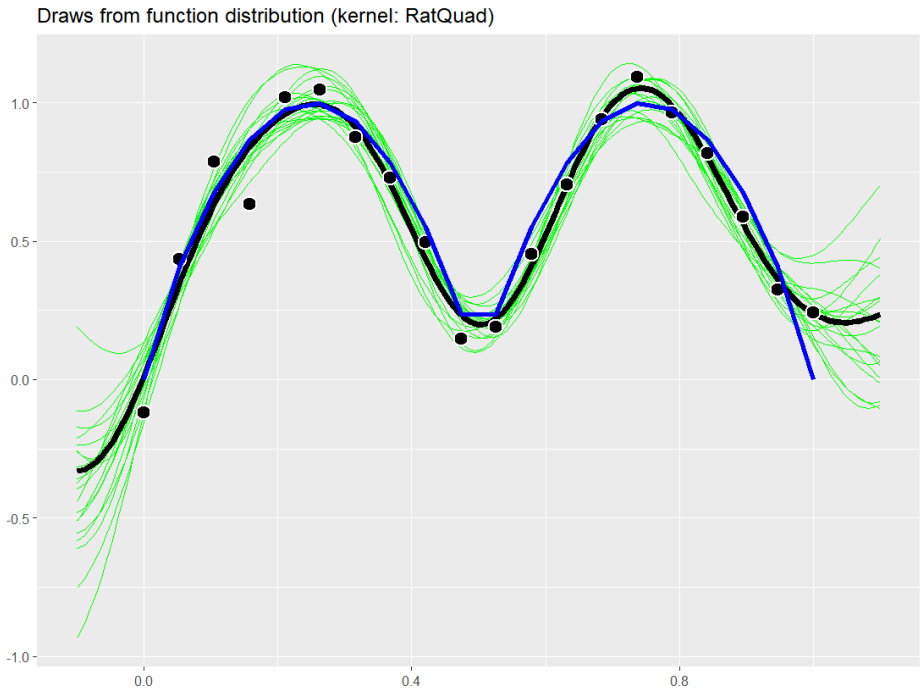


2.2.5 Rational quadratic

The rational quadratic can be seen as an infinite sum of SE with different length-scales.

TODO formulas





2.3 Deriving kernels [5]

TODO

2.4 Learning best kernel from data [1]

TODO

3 Computational Issues

3.1 Matrix inversion [3]

Inverting the $[K(X,X)+\sigma_n^2 I]$ matrix in our predictive distribution scales poorly with the number of training data points n, as inverting the $n \times n$ matrix X that represents our training data is $O(n^3)$. Strategies to approximate the result of this inversion fall into two categories: those that produce a single approximation for the entire dataset, or those that produce several approximations that are "experts" in a particular region of the dataset and combine these local approximations to form a global approximation.

3.2 Global approximations

3.2.1 Subset-of-data

The simplest strategy is to use a subset M of X to reduce the cost of inversion to $O(m^3)$, where m is the number of training points in M. Although this approach does not address the issues of matrix inversion directly,a theoretical graphon analysis proves that choosing M randomly gives an accuracy of $O(log^{-1/4}m)$ for the predictive mean and variance, which produces more accurate predictions with faster runtimes than sparse approximations as n increases. [2] Subset-of-data also requires no analytic assumptions about the kernel.

We can reduce m needed to achieve the same level of accuracy with a "greedy" approach by determining the gain in likelihood from including each data point x_i in X, adding the maximum gain in likelihood point to M and repeating until the size of M reaches m. However, computational savings from reducing m are smaller than the cost of searching X for these centroids $O(n^2m)$. Instead, we can use a "matching pursuit" approach - maintain a cache of the already precomputed kernel values, and use these to compute the gain in likelihood for each point in X in $O(nm^2)$ time. [**matching-pursuit**]

3.2.2 Sparse kernels

A sparse kernel is a particularly designed kernel that imposes $k(X,X')=0$ if $|X-X'|$ is larger than some threshold d to create a sparse covariance matrix. This reduces the number of calculations that need to be performed and computational complexity to $O(an^3)$, where a is the proportion of non-zero entries remaining, but the kernel needs to be carefully designed to work with zeroes and ensure all entries are positive definite to satisfy completeness. TODO sparse RBF

3.2.3 Sparse approximations

TODO, missing background

3.2.3.1 Prior approximation

3.2.3.2 Posterior approximation

3.2.3.3 Structured sparse approximation

3.3 Local approximations

TODO

3.3.1 Naive-local-experts

3.3.2 Mixture-of-experts

3.3.3 Product-of-experts

3.4 Improvements

TODO

3.4.1 Scalability

3.4.2 Capability

4 Applying a Gaussian Process to Astrostatistics

4.1 Introduction

4.2 Methodology

4.2.1 Selecting the covariance function

4.2.2 Resolving computational issues

4.3 Results

4.4 Discussion

4.5 Conclusion

5 Conclusion

References

[1] Jean-Luc Akian et al. "Learning "best" kernels from data in Gaussian process regression. With application to aerodynamics". In: *Journal of Computational Physics* 470 (Aug. 2022), p. 111595. DOI: 10.1016/j.jcp.2022.111595.

[2] Kohei Hayashi, Masaaki Imaizumi, and Yuichi Yoshida. "On Random Subsampling of Gaussian Process Regression: A Graphon-Based Analysis". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, 26–28 Aug 2020, pp. 2055–2065. URL: https://proceedings.mlr.press/v108/hayashi20a.html.

[3] Haitao Liu et al. "When Gaussian Process Meets Big Data: A Review of Scalable GPs". In: *IEEE Transactions on Neural Networks and Learning Systems* 31.11 (2020), pp. 4405–4423. DOI: 10.1109/TNNLS.2019.2957109.

[4] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, Nov. 2005. ISBN: 9780262256834. DOI: 10.7551/mitpress/3206.001.0001. eprint: https://direct.mit.edu/book-pdf/2514321/book_9780262256834.pdf. URL: https://doi.org/10.7551/mitpress/3206.001.0001.

[5] Andrew Wilson and Ryan Adams. "Gaussian Process Kernels for Pattern Discovery and Extrapolation". In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, June 2013, pp. 1067–1075. URL: https://proceedings.mlr.press/v28/wilson13.html.