Lecture 22: Gaussian Processes

The notes below are based on references [**?**, Sec. 6.4] and [**?**, Ch. 2,4,5] and [**?**, Ch. 2].

Gaussian Process: Function-Space View

Gaussian Process Definition. A GP $f(\boldsymbol{x})$ is a *random function* of an argument $x \in \mathcal{X}$ where $\mathcal{X}$ is an input domain. For example, if the GP is a temporal process then $\mathcal{X} = \mathbb{R}$ is the time domain, or if the GP is a spatial process then $\mathcal{X} = \mathbb{R}^2$ or $\mathbb{R}^3$. If the GP models a dynamical system then the input may be the system state $\mathcal{X} = \mathbb{R}^n$. Often the output $f(\boldsymbol{x})$ is a scalar but this is not strictly required. A GP is completed specified by its mean function and covariance function. Let $m(\boldsymbol{x})$ denote the mean over the input space and let $k(\boldsymbol{x}, \boldsymbol{x}')$ denote the covariance function (also called the *kernel*). Formally,

$$m(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})] \tag{1}$$
$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[\{f(\boldsymbol{x}) - m(\boldsymbol{x})\}\{f(\boldsymbol{x}') - m(\boldsymbol{x}')\}] = \text{Cov}(\boldsymbol{x}, \boldsymbol{x}') \ . \tag{2}$$

From the above definition it follows that:

1. The kernel is positive since

$$k(\boldsymbol{x}, \boldsymbol{x}) = \mathbb{E}[\{f(\boldsymbol{x}) - m(\boldsymbol{x})\}^2\}] = \text{Var}(\boldsymbol{x}) \geq 0 \tag{3}$$

2. The kernel is symmetric since

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[\{f(\boldsymbol{x}) - m(\boldsymbol{x})\}\{f(\boldsymbol{x}') - m(\boldsymbol{x}')\}] \tag{4}$$
$$= \mathbb{E}[\{f(\boldsymbol{x}') - m(\boldsymbol{x})\}\{f(\boldsymbol{x}) - m(\boldsymbol{x}')\}] \tag{5}$$
$$= k(\boldsymbol{x}', \boldsymbol{x}) \tag{6}$$

These resemble our familiar definitions of mean and covariance, however they pertain to random *functions* rather than random variables or random vectors. The GP definition above is compactly summarized with the notation:

$$f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')) \ . \tag{7}$$

Covariance Functions (Kernels). At the heart of GPs is the covariance function or kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ that encodes a sense of "smoothness" of the GP over the input space. If the two nearby points, $\boldsymbol{x}$ and $\boldsymbol{x}'$, are highly correlated, then $k(\boldsymbol{x}, \boldsymbol{x}')$ will be large. Conversely, if the two points are far away then $k(\boldsymbol{x}, \boldsymbol{x}')$ will be small. The sense of "close" or "far" is encoded by the shape of the covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$, which is often parameterized by a time or length scale and other constants called *hyperparameters*. The distance between two input points is sometimes called the *lag* $\boldsymbol{h} = \boldsymbol{x} - \boldsymbol{x}'$ (a term common in the geostatistics). When the covariance function depends only on the lag (i.e., $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{h})$ it is called *stationary*. Moreover, if the covariance function depends only on the magnitude of the lag vector $k(\boldsymbol{x}, \boldsymbol{x}') = k(||\boldsymbol{h}||)$it is called *isotropic* (i.e., direction-independent). Stationary, isotropic covariance functions are perhaps the most common and include:

- Squared exponential:

$$k_{SE}(||\boldsymbol{h}||) = \sigma^2 \exp\left(-\frac{||\boldsymbol{h}||^2}{2L}\right) \tag{8}$$

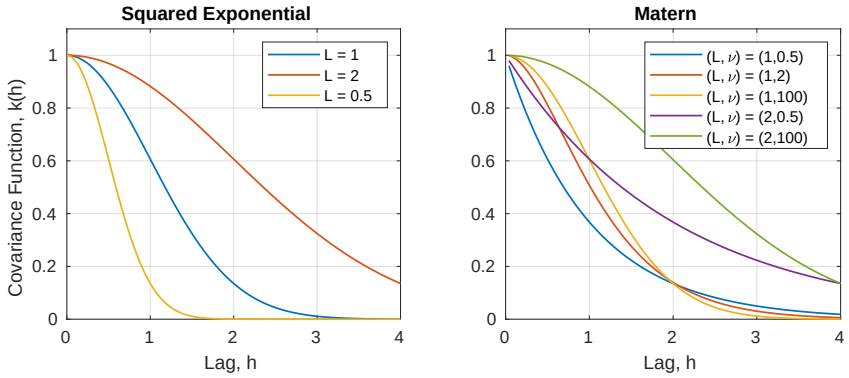
  where the hyperparameters are $\boldsymbol{\theta} = [L, \sigma]^{\mathrm{T}}$ and L is a length-scale parameter, and σ^2 is a variance parameter.

- Matérn class

$$k_{\text{Matern}}(||\boldsymbol{h}||) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\left(\frac{||\boldsymbol{h}||}{L}\right)\right)^{2} K_{\nu}\left(\sqrt{2\nu}\left(\frac{||\boldsymbol{h}||}{L}\right)\right) \tag{9}$$

  where the hyperparameters are $\boldsymbol{\theta} = [L, \sigma, \nu]^{\mathrm{T}}$ and, as before, L is a length-scale parameter, σ^2 is a variance parameter, and ν is a positive constant shape parameter. In this expression $K_{\nu}(\cdot)$ is the modified Bessel function of the second kind and $\Gamma(\cdot)$ is the gamma function.

These covariance functions are plotted below for different values of the hyperparameters (with $\sigma = 1$ in both cases).



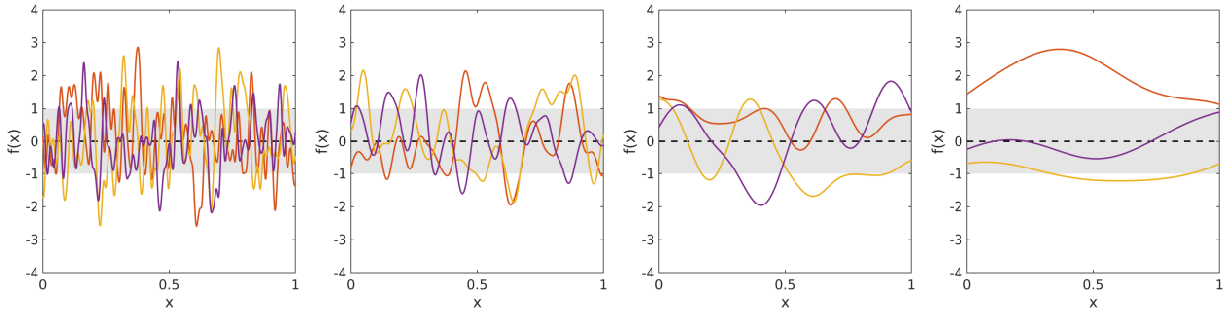
The main requirement for the covariance function is for it to be positive definite. Olea [**?**, Defn. 2.3] defines a positive definite function as follows. Let m be a positive integer, let $\{\alpha_1, \alpha_2, \ldots, \alpha_m\}$ be a set of real or complex numbers, and let $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \boldsymbol{x}_m\}$ be a set of points in an n-dimensional Euclidean space. The function $\phi(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is a positive definite function if

$$\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j \phi(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0\,. \tag{10}$$

Since the product of two positive definite functions is also positive definite the kernels can be multiplied to form new (valid) kernels[1]. This is often used when modeling spatiotemporal processes. For example, let the input space consist of vectors $\boldsymbol{x} = [\boldsymbol{s}, t]^{\mathrm{T}}$ where $\boldsymbol{s} \in D \subset \mathbb{R}^2$ is a spatial coordinate and $t \in \mathbb{R}^2$ is the time. Spatial correlations can be modeled by a kernel of the form $k_s(\boldsymbol{s}, \boldsymbol{s}')$ whereas temporal correlations can be modeled by a kernel of the form $k_t(t, t')$. Then a spatiotemporal kernel can be formed as

$$k(\boldsymbol{x}, \boldsymbol{x}') = k_s(\boldsymbol{s}, \boldsymbol{s}') k_t(t, t')\,. \tag{11}$$

[1]For an extensive overview see the Kernel Cookbook [Link]



(a) $L = 0.0001$ (b) $L = 0.001$ (c) $L = 0.01$ (d) $L = 0.1$

Figure 1: Gaussian process realizations on the interval of $x \in [0, 1]$ using a squared exponential kernel with $\sigma^2 = 1$ with varying lengthscales L.

Realizations. Since the GP is a random function there are infinitely many *realizations* (i.e., example functions) that can be sampled from it. On average, the ensemble of these realizations satisfies the mean $m(\boldsymbol{x})$ and covariance $k(\boldsymbol{x}, \boldsymbol{x}')$ of the GP. We can numerically simulate a GP over this input space as follows. First choose a set of Q points over which to define the realizaton $\bar{\boldsymbol{X}} = \{\bar{\boldsymbol{x}}_1, \ldots, \bar{\boldsymbol{x}}_Q\}$ and evaluate the mean at each point:

$$\boldsymbol{\mu}(\bar{\boldsymbol{X}}) = \begin{bmatrix} m(\bar{\boldsymbol{x}}_1) \\ \vdots \\ m(\bar{\boldsymbol{x}}_Q) \end{bmatrix}. \tag{12}$$

Then, compute the covariance matrix relating all of the grid points to each other as

$$\boldsymbol{K}(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}}) = \begin{bmatrix} k(\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_1) & k(\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_2) & \cdots & k(\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{x}}_Q) \\ k(\bar{\boldsymbol{x}}_2, \bar{\boldsymbol{x}}_1) & \vdots & \ddots & k(\bar{\boldsymbol{x}}_2, \bar{\boldsymbol{x}}_Q) \\ \vdots & \vdots & \ddots & \vdots \\ k(\bar{\boldsymbol{x}}_Q, \bar{\boldsymbol{x}}_1) & k(\bar{\boldsymbol{x}}_Q, \bar{\boldsymbol{x}}_2) & \cdots & k(\bar{\boldsymbol{x}}_Q, \bar{\boldsymbol{x}}_Q) \end{bmatrix} \tag{13}$$

A GP realization is then found by sampling from the distribution

$$\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{\mu}(\bar{\boldsymbol{X}}), \boldsymbol{K}(\bar{\boldsymbol{X}}, \bar{\boldsymbol{X}})) \tag{14}$$

For example, in MATLAB the `mvrnd` function can be used.

GP Regression. Rather than simulation GP realizations, a more common usage of GPs is to perform regression—given some set of partial observations $\boldsymbol{f}$ at locations $\boldsymbol{X}$ we wish to use the assumed mean and covariance to predict the values of the process $\boldsymbol{f}_*$ at the grid points $\boldsymbol{X}_*$ (or a single point, if desired). Suppose that the number of observations is $|\boldsymbol{X}| = N$ and the number of grid points is $|\boldsymbol{X}^*| = M$. To perform the regression we begin by forming a new stacked vector of the observations $[\boldsymbol{f}^\mathrm{T}, \boldsymbol{f}_*^\mathrm{T}]^\mathrm{T}$. For simplicity, assume the mean of the GP is zero. Then this new random vector is distributed according to the joint distribution:

$$\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) & \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_*) \\ \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}) & \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right) \tag{15}$$

where

- $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) \in \mathbb{R}^{N \times N}$ is the covariance matrix relating observation points to each other as in (**??**).
- $\boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}) \in \mathbb{R}^{M \times N}$ is the covariance matrix relating the test/prediction grid points to the observation points, similar to (**??**). Note that $\boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}) = \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_*)^{\mathrm{T}}$.
- $\boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*) \in \mathbb{R}^{M \times M}$ is the covariance matrix relating the test/prediction grid points to each other.

Equation (**??**) represents the joint distribution between the random vectors $\boldsymbol{f}_*$ (the prediction/grid values) and $\boldsymbol{f}$ (the observed values). To perform the regression we first review the following fact.

Partitioned Gaussian Vectors. Consider a Gaussian random vector partitioned as $\boldsymbol{z} = [\boldsymbol{x}^T, \boldsymbol{y}^T]^T$ with mean

$$E[\boldsymbol{z}] = \begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix} \tag{16}$$

and partitioned covariance matrix

$$\boldsymbol{P}_f = \begin{bmatrix} \boldsymbol{P}_{xx} & \boldsymbol{P}_{xy} \\ \boldsymbol{P}_{yx} & \boldsymbol{P}_y \end{bmatrix} \tag{17}$$

The symmetry of $\boldsymbol{P}_z$ implies that $\boldsymbol{P}_{xy} = \boldsymbol{P}_{yx}^T$. It can be shown that the conditional probability of $\boldsymbol{y}$ given $\boldsymbol{x}$ is also a Gaussian vector with mean and covariance:

$$\boldsymbol{\mu}_{y|x} = \boldsymbol{\mu}_y + \boldsymbol{P}_{yx}\boldsymbol{P}_{xx}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_x) \tag{18}$$

$$\boldsymbol{P}_{y|x} = \boldsymbol{P}_{yy} - \boldsymbol{P}_{yx}\boldsymbol{P}_{xx}^{-1}\boldsymbol{P}_{xy}\ . \tag{19}$$

For a derivation of this fact refer to the Appendix notes on Gaussian random vectors.

Applying the above conditioning principle to (**??**) we find that the mean value $\boldsymbol{f}_* \in \mathbb{R}^{M \times 1}$ at the grid points $\boldsymbol{X}_*$ is conditioned on the observed values $\boldsymbol{f} \in \mathbb{R}^{N \times 1}$ is a Gaussian random vector

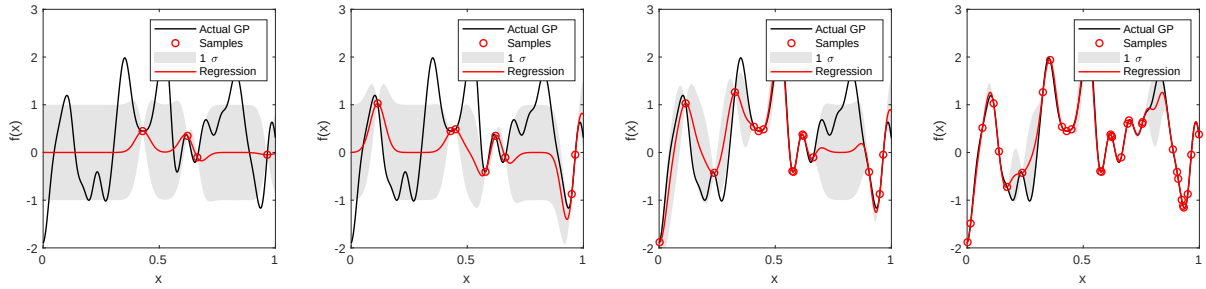
$$\boldsymbol{f}_* \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{f}_*|\boldsymbol{f}}, \boldsymbol{P}_{\boldsymbol{f}_*|\boldsymbol{f}}) \tag{20}$$

where

$$\boldsymbol{\mu}_{\boldsymbol{f}_*|\boldsymbol{f}} = \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X})\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})^{-1}\boldsymbol{f} \tag{21}$$

$$\boldsymbol{P}_{\boldsymbol{f}_*|\boldsymbol{f}} = \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*) - \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X})\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})^{-1}\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_*) \tag{22}$$

Notice that the predicted values $\boldsymbol{\mu}_{\boldsymbol{f}_*|\boldsymbol{f}}$ in (**??**) are a linear combination of the existing data $\boldsymbol{f}$ and the matrix $\boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X})\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})^{-1}$ can be viewed as a matrix of weights that determine the contribution of each data point to each prediction. A useful feature of GP regression is that it is accompanied immediately by a covariance matrix $\boldsymbol{P}_{\boldsymbol{f}_*|\boldsymbol{f}}$ given by (**??**). The diagonal elements of this matrix describe the uncertainty at each estimation point. As an example in Fig. **??**, the resulting mean and associated uncertainty are plotted with increasing number of measurements. Both (**??**)–(**??**) involve the term $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})^{-1}$ which requires inverting a $N \times N$ matrix where N is the number of data points. For large data sets this can be prohibitive and numerous techniques have been proposed to approximate the regression.



(a) $N = 4$ (b) $N = 8$ (c) $N = 16$ (d) $N = 32$

Figure 2: Gaussian process regression with an increasing number of measurements.

GP Regression with Noisy Measurements. Suppose now that the measurements of the GP are corrupted by Gaussian noise

$$y = f(\boldsymbol{x}) + \epsilon \tag{23}$$

where $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$. In this setting, $\boldsymbol{f}$ is considered a "latent" or "hidden" variable. The vector of noisy observations $\boldsymbol{y}$ is dependent on the latent variable $\boldsymbol{f}$ by

$$p(\boldsymbol{y}|\boldsymbol{f}) = \mathcal{N}(\boldsymbol{y}|\boldsymbol{f}, \sigma_n^2 \mathbf{1}) \tag{24}$$

We assumed that $\boldsymbol{f}$ is zero-mean and has covariance given by $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})$, thus

$$p(\boldsymbol{f}) = \mathcal{N}(\mathbf{0}, \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})) \tag{25}$$

The margnial distribution conditioned on the input values is

$$p(\boldsymbol{y}) = \int p(\boldsymbol{y}|\boldsymbol{f}) p(\boldsymbol{f}) d\boldsymbol{f} \tag{26}$$

where the terms $p(\boldsymbol{y}|\boldsymbol{f})$ and $p(\boldsymbol{f})$ are given above. Using properties of Gaussian random vectors it follows that the joint distribution is

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \mathbf{1} & \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_*) \\ \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}) & \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*) \end{bmatrix}\right) \tag{27}$$

Again, applying the conditioning principle to (**??**) we find that the mean value $\boldsymbol{f}$ at the grid points $\boldsymbol{X}$ is conditioned on the observed values $\boldsymbol{f}$ is a Gaussian random vector

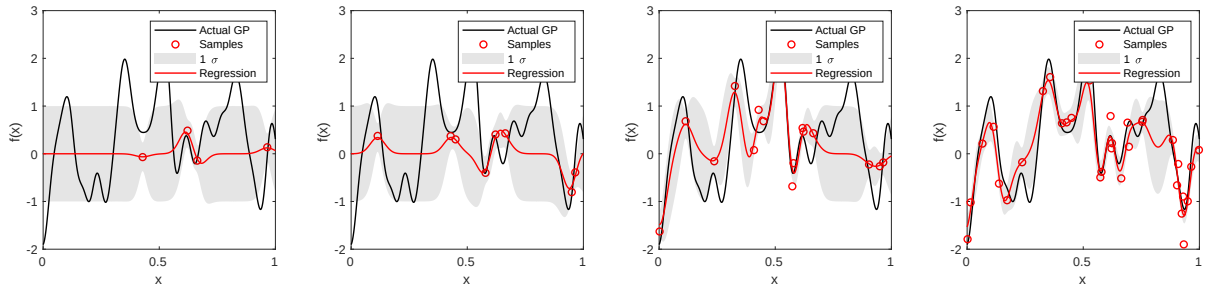
$$\boldsymbol{f}_* \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{f}_*|\boldsymbol{f}}, \boldsymbol{P}_{\boldsymbol{f}_*|\boldsymbol{f}}) \tag{28}$$

where

$$\boldsymbol{\mu}_{\boldsymbol{f}_*|\boldsymbol{f}} = \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \mathbf{1})^{-1} \boldsymbol{y} \tag{29}$$

$$\boldsymbol{P}_{\boldsymbol{f}_*|\boldsymbol{f}} = \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*) - \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X})(\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2 \mathbf{1})^{-1} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_*) \tag{30}$$

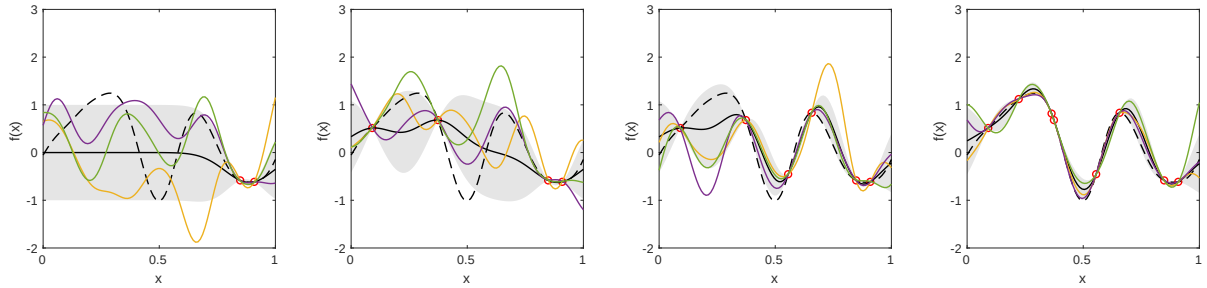
The figures below illustrate the GP regression with noisy data.



(a) $N = 4$ (b) $N = 8$ (c) $N = 16$ (d) $N = 32$

Figure 3: Gaussian process regression with an increasing number of measurements curropted by additive Gaussian noise.

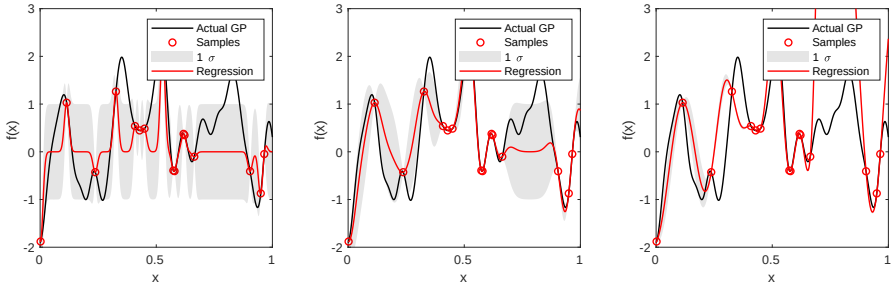
GP Realizations (Conditioned on Data). As before, we can generate sample realizations of the GP that are now conditioned on the data. This amounts to generate a random vector from the normal distribution given by the posterior (**??**) or (**??**) for the noise-fee or noisy cases, respectively. Example GP realizations conditioned on data are shown below.



(a) $N = 2$ (b) $N = 4$ (c) $N = 6$ (d) $N = 8$

Figure 4: Gaussian process realizations conditioned on data for an increasing number of measurements. The solid black line is the regression, the dashed black line is the actual GP, and the colored lines are various GP realizations

Hyperparameter selection. Thusfar we have not commented on how the hyperparameters of the covariance function are chosen. However, as the figures below illustrate, the choice of hyperparameters can greatly affect both the mean and variance of the posterior GP regression. In the following we will overview an approach for optimizing the hyperparameters.



(a) $0.1L$ (too short) (b) L (correct) (c) $L = 4$ (too long)

Figure 5: Hyperparameter selection

Hyperparameter Optimization. If the hyperparameters are unknown an optimization problem may be formulated to determine them, often with the aim of minimizing the the marginal likelihood $p(\boldsymbol{y}|\boldsymbol{X},\boldsymbol{\theta})$, which is the probability of the observed (noisy) data $\boldsymbol{y}$ given the model. If the covariance function is fixed, then the statistical model relating the observations is only a function of the observation locations $\boldsymbol{X}$ and the model hyperparameters $\boldsymbol{\theta}$. The marginal likelihood can be computed as follows

$$p(\boldsymbol{y}|\boldsymbol{X};\boldsymbol{\theta}) = \int p(\boldsymbol{y}|\boldsymbol{f},\boldsymbol{X})p(\boldsymbol{f}|\boldsymbol{X};\boldsymbol{\theta})d\boldsymbol{f} \tag{31}$$

which is normally distributed with covariance

$$\boldsymbol{K}_y = \sigma_n^2 \mathbf{1} + \boldsymbol{K} \tag{32}$$

(see [**?**, Sec. 6.4.2-6.4.3] and [**?**, Sec. 5.4.1]). It is often more useful to work with the log likelihood

$$\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_y^{-1}\boldsymbol{y} - \frac{1}{2}\log|\boldsymbol{K}_y| - \frac{n}{2}\log 2\pi \tag{33}$$

For optimization, we need the partial derivatives (with respect to θ_j for all $j = 1, \ldots, |\boldsymbol{\theta}|$) :

$$\frac{\partial}{\partial\theta_j}\log p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) = -\frac{\partial}{\partial\theta_j}\left(\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\boldsymbol{K}_y^{-1}\boldsymbol{y}\right) - \frac{\partial}{\partial\theta_j}\left(\frac{1}{2}\log|\boldsymbol{K}_y|\right) - \underbrace{\frac{\partial}{\partial\theta_j}\left(\frac{n}{2}\log 2\pi\right)}_{=0} \tag{34}$$

$$= -\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\left(\frac{\partial}{\partial\theta_j}\boldsymbol{K}_y^{-1}\right)\boldsymbol{y} - \frac{1}{2}\frac{\partial}{\partial\theta_j}\log|\boldsymbol{K}_y| \tag{35}$$

$$= -\frac{1}{2}\boldsymbol{y}^{\mathrm{T}}\left(-\boldsymbol{K}_y^{-1}\frac{\partial\boldsymbol{K}_y}{\partial\theta_j}\boldsymbol{K}_y^{-1}\right)\boldsymbol{y} - \frac{1}{2}\mathrm{tr}\left(\boldsymbol{K}_y^{-1}\frac{\partial\boldsymbol{K}_y}{\partial\theta_j}\right) \tag{36}$$

where the following identities described below were used.

Aside: The follow identity holds when differentiating a matrix $\boldsymbol{U}(x)$ with respect to a scalar x:

$$\frac{\partial\boldsymbol{U}^{-1}}{\partial x} = -\boldsymbol{U}^{-1}\frac{\partial\boldsymbol{U}}{\partial x}\boldsymbol{U}^{-1} \tag{37}$$

or a scalar-by-scalar with matrices involved

$$\frac{\partial\log|\boldsymbol{U}|}{\partial x} = \mathrm{tr}\left(\boldsymbol{U}^{-1}\frac{\partial\boldsymbol{U}}{\partial x}\right) \tag{38}$$

Link: For a more extensive list of identities see here.

The choice of kernel will determine the partial derivative $\frac{\partial\boldsymbol{K}_y}{\partial\theta_j}$. We can now formulate an optimization problem to find the optimal hyperparameters $\boldsymbol{\theta}^*$ that maximize (**??**) (equivalently minimize negative of (**??**)) using any standard gradient-descent solver. The expression (**??**) gives the gradient of the cost function and is used to iteratively move the candidate hyperparameter $\boldsymbol{\theta}$ to a lower-cost location. However, since the marginal likelihood is nonconvex there may be multiple local minimia.

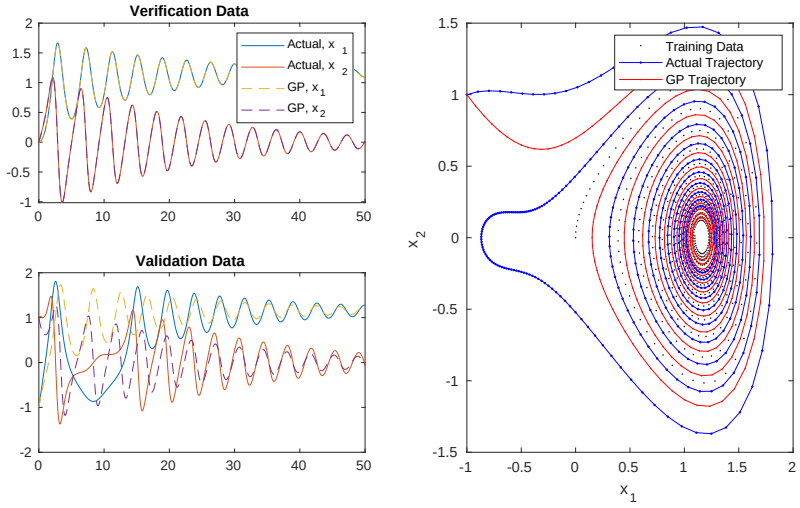
Cross Validation. As with any machine learning algorithm, the trained model should be *cross-validated*. The basic idea is to split the data into a training set used to learn the model and a validation set to test the learned model (i.e., estimate its performance on previously unobserved data). However, drawbacks of this approach are that only a fraction of the full data can be used for training and, if the validation data set is small, then the estimated performance can have a large variance. To address this issue k-fold cross validation is often used. The data is split into k disjoint subsets and then trained separately, k times, each time leaving out a different subset of the data for validation. The hyperparameters learned from each of the model are then combined, for example, by averaging.

Model Selection. Our previous discussion concerned hyperparameter optimization under a known model. However, we may wish to evaluate several different models (e.g., GP models with different kernels) from a library of hypothesized candidate models. This problem can be viewed as a nested optimization problem wherein at the outer level we optimize the choice of model and at the inner level we optimize the chosen model's hyperparameters.

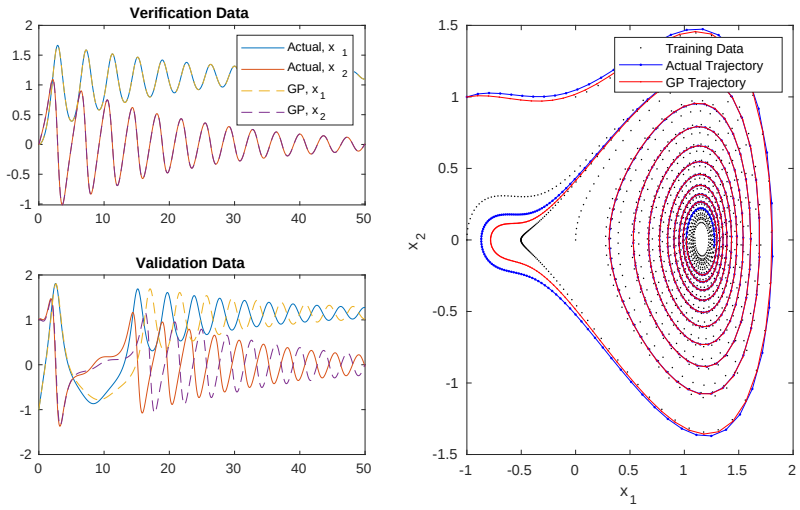
Example (Double Well, Strogatz Ch. 12.5). A well known example of a bi-stable system is the model:

$$\ddot{x} + \delta\dot{x} - x + x^3 = F\cos\omega t \tag{39}$$

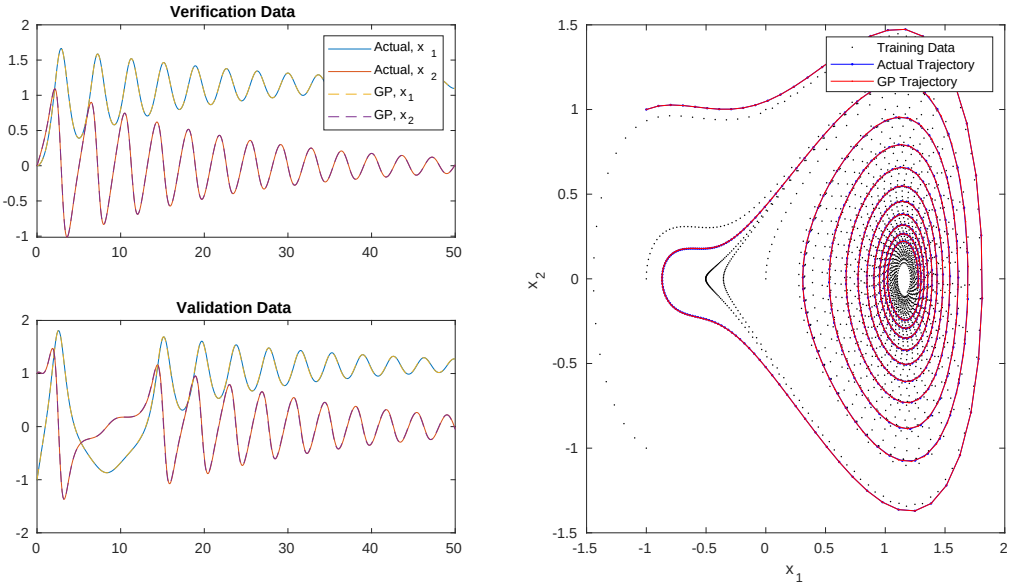
In the following we make the assumption that $\omega = 0$ (i.e., a constant force) and assume the parameter $\delta = 0.1$ and $F = 0.4$. The system is converted into a two-state model with $x_1 = x$ and $x_2 = \dot{x}$ and simulated for 50 seconds using `ode45` in MATLAB. To generate the training data we begin by choosing the initial condition $(x_1, x_2) = (0, 0)$. For validation we use $(x_1, x_2) = (-1, 1)$. The GP is trained with a squared exponential kernel with hyperparameters of unit length and unit variance. The resulting GP-predicted trajectory is shown below. Since the training data does not cover the space of the validation trajectory we see a mismatch in the initial response.



To alleviate this issue we can add additional training data for the initial conditions $(x_1, x_2) = (0, 1)$ and $(x_1, x_2) = (-1, 0)$. The results are shown below. There is an improvement however the predicted response appears out of phase with the actual response.



Adding two more initials to the training data set, $(x_1, x_2) = (1, 1)$ and $(x_1, x_2) = (-1, -1)$, appears to alleviate this issue for this validation data set. For the GP predictions to be accurate over a wide range of initial conditions additional training data is needed.



Gaussian Processes: Weight-Space View

In the functions-space view described above GP-based predictions were obtained by condition the joint probability distribution of the training data and prediction points on the training data to obtain a posterior distribution. Another approach of obtaining the regression is the so-called weight-space view in which the GP regression is viewed as an optimization problem to identify optimal weights. To simplify the discussion suppose the mean is a known constant m and again.

The simple kriging estimator $f_*(\boldsymbol{x}_*)$ at site $\boldsymbol{x}_*$ is a linear combination of k variables at sites $\boldsymbol{x}_i$ for $i = 1, \ldots, k$:

$$f_*(\boldsymbol{x}_*) = m + \sum_{i=1}^{k} \lambda_i (f(\boldsymbol{x}_i) - m) \tag{40}$$

where λ_i is a set of weights. The estimation variance at site $\boldsymbol{x}_*$ is

$$\begin{aligned}
\sigma^2(\boldsymbol{x}_*) &= \text{Var}(f_*(\boldsymbol{x}_*) - f(\boldsymbol{x}_*)) && (41)\\
&= \text{Var}(m + \sum_{i=1}^{k} \lambda_i (f(\boldsymbol{x}_i) - m) - f(\boldsymbol{x}_*)) && (42)\\
&= \text{Var}\left(\sum_{i=1}^{k} \lambda_i (f(\boldsymbol{x}_i) - m) - [f(\boldsymbol{x}_*) - m] \right) && (43)\\
&= \text{Var}\left(\sum_{i=1}^{k} \lambda_i \epsilon(\boldsymbol{x}_i) - \epsilon(\boldsymbol{x}_*) \right) . && (44)
\end{aligned}$$

where $\epsilon(\boldsymbol{x}) = f(\boldsymbol{x}) - E[f(\boldsymbol{x})] = f(\boldsymbol{x}) - m$ is the residual. Note that

$$E[\epsilon(\boldsymbol{x})] = E[f(\boldsymbol{x}) - m] = 0 \tag{45}$$

Changing the index from $i = 1$ to $i = 0$ and defining $\lambda_0 = -1$:

$$\begin{aligned}
\sigma^2(\boldsymbol{x}_*) &= \text{Var}\left(\sum_{i=1}^{k} \lambda_i \epsilon(\boldsymbol{x}_i) + \lambda_0 \epsilon(\boldsymbol{x}_*) \right) && (46)\\
&= \text{Var}\left(\sum_{i=0}^{k} \lambda_i \epsilon(\boldsymbol{x}_i) \right) && (47)
\end{aligned}$$

Using the definition of variance the expression (**??**) can be expanded and factored as

$$\sigma^2(\boldsymbol{x}_*) = E\left[\left(\sum_{i=0}^{k}\lambda_i\epsilon(\boldsymbol{x}_i)\right)^2\right] - \left(E\left[\sum_{i=0}^{k}\lambda_i\epsilon(\boldsymbol{x}_i)\right]\right)^2 \tag{48}$$

$$= E\left[\left(\sum_{i=0}^{k}\lambda_i\epsilon(\boldsymbol{x}_i)\right)\left(\sum_{j=0}^{k}\lambda_j\epsilon(\boldsymbol{x}_j)\right)\right] - \left(\left[\sum_{i=0}^{k}\lambda_i E[\epsilon(\boldsymbol{x}_i)]\right]\right)^2 \tag{49}$$

$$= E\left[\left(\sum_{i=0}^{k}\lambda_i\epsilon(\boldsymbol{x}_i)\right)\left(\sum_{j=0}^{k}\lambda_j\epsilon(\boldsymbol{x}_j)\right)\right] - \left(\left[\sum_{i=0}^{k}\lambda_i E[\epsilon(\boldsymbol{x}_i)]\right]\right)^2 \tag{50}$$

$$= E\left[\sum_{i=0}^{k}\sum_{j=0}^{k}\lambda_i\lambda_j\epsilon(\boldsymbol{x}_i)\epsilon(\boldsymbol{x}_j)\right] - \left(\left[\sum_{i=0}^{k}\lambda_i \underbrace{E[\epsilon(\boldsymbol{x}_i)]}_{=0}\right]\right)^2 \tag{51}$$

$$= \sum_{i=0}^{k}\sum_{j=0}^{k}\lambda_i\lambda_j E[\epsilon(\boldsymbol{x}_i)\epsilon(\boldsymbol{x}_j)] \tag{52}$$

$$= \sum_{i=0}^{k}\sum_{j=0}^{k}\lambda_i\lambda_j E[(f(\boldsymbol{x}_i)-m)(f(\boldsymbol{x}_j)-m)] \tag{53}$$

$$= \sum_{i=0}^{k}\sum_{j=0}^{k}\lambda_i\lambda_j\left\{E[f(\boldsymbol{x}_i)f(\boldsymbol{x}_j)] - m\underbrace{E[f(\boldsymbol{x}_i)]}_{=m} - m\underbrace{E[f(\boldsymbol{x}_j)]}_{=m} + m^2\right\} \tag{54}$$

$$= \sum_{i=0}^{k}\sum_{j=0}^{k}\lambda_i\lambda_j\left\{\underbrace{E[f(\boldsymbol{x}_i)f(\boldsymbol{x}_j)] - m^2}_{\mathrm{Cov}(f(\boldsymbol{x}_i),f(\boldsymbol{x}_j))}\right\} \tag{55}$$

$$= \sum_{i=0}^{k}\sum_{j=0}^{k}\lambda_i\lambda_j k(\boldsymbol{x}_i,\boldsymbol{x}_j) \tag{56}$$

Change the limits of the summation back to $i=1$ and $j=1$, recall that k is symmetric:

$$\sigma^2(\boldsymbol{x}_*) = \sum_{i=0}^{k}\sum_{j=0}^{k}\lambda_i\lambda_j k(\boldsymbol{x}_i,\boldsymbol{x}_j) \tag{57}$$

$$\begin{aligned}=&\lambda_0\lambda_0 k(\boldsymbol{x}_*,\boldsymbol{x}_*) + \lambda_0\lambda_1 k(\boldsymbol{x}_*,\boldsymbol{x}_1) + \cdots + \lambda_0\lambda_k k(\boldsymbol{x}_*,\boldsymbol{x}_k) +\\ &\lambda_1\lambda_0 k(\boldsymbol{x}_1,\boldsymbol{x}_*) + \lambda_1\lambda_1 k(\boldsymbol{x}_1,\boldsymbol{x}_1) + \cdots + \lambda_1\lambda_k k(\boldsymbol{x}_1,\boldsymbol{x}_k) +\\ &\vdots\\ &\lambda_k\lambda_0 k(\boldsymbol{x}_k,\boldsymbol{x}_*) + \lambda_k\lambda_1 k(\boldsymbol{x}_k,\boldsymbol{x}_1) + \cdots + \lambda_k\lambda_k k(\boldsymbol{x}_k,\boldsymbol{x}_k)\end{aligned} \tag{58}$$

$$= \sum_{i=1}^{k}\sum_{j=1}^{k}\lambda_i\lambda_j k(\boldsymbol{x}_i,\boldsymbol{x}_j) + 2\underbrace{\lambda_0}_{=-1}\sum_{i=1}^{k}\lambda_i k(\boldsymbol{x}_i,\boldsymbol{x}_*) + \lambda_0^2 k(\boldsymbol{x}_*,\boldsymbol{x}_*) \tag{59}$$

$$= \sum_{i=1}^{k}\sum_{j=1}^{k}\lambda_i\lambda_j k(\boldsymbol{x}_i,\boldsymbol{x}_j) - 2\sum_{i=1}^{k}\lambda_i k(\boldsymbol{x}_i,\boldsymbol{x}_*) + k(\boldsymbol{x}_*,\boldsymbol{x}_*) \tag{60}$$

Define the matrix

$$\boldsymbol{K} = \boldsymbol{K}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & k(\boldsymbol{x}_2, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_k, \boldsymbol{x}_1) \\ k(\boldsymbol{x}_1, \boldsymbol{x}_2) & k(\boldsymbol{x}_2, \boldsymbol{x}_2) & \cdots & k(\boldsymbol{x}_k, \boldsymbol{x}_2) \\ \cdots & \cdots & \cdots & \cdots \\ k(\boldsymbol{x}_1, \boldsymbol{x}_k) & k(\boldsymbol{x}_2, \boldsymbol{x}_k) & \cdots & k(\boldsymbol{x}_k, \boldsymbol{x}_k) \end{bmatrix}, \tag{61}$$

the vector of weights $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \cdots, \lambda_k]^T$, and the vector of covariance of sample sites with the estimation site

$$\boldsymbol{\nu} = [k(\boldsymbol{x}_1, \boldsymbol{x}_*) \quad k(\boldsymbol{x}_2, \boldsymbol{x}_*) \quad \cdots \quad k(\boldsymbol{x}_k, \boldsymbol{x}_*)]^T, \tag{62}$$

then

$$\sigma^2(\boldsymbol{x}_*; \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \boldsymbol{\lambda}) = \boldsymbol{\lambda}^T \boldsymbol{K} \boldsymbol{\lambda} - 2\boldsymbol{\lambda}^T \boldsymbol{\nu} + k(\boldsymbol{x}_*, \boldsymbol{x}_*) . \tag{63}$$

To find the optimal weights $\boldsymbol{\lambda}^*$ that minimize $\sigma^2(\boldsymbol{x}_*)$ we can formulate the following optimization problem:

$$\text{minimize} \quad \sigma^2(\boldsymbol{\lambda}; \boldsymbol{x}_*, \boldsymbol{x}_1, \ldots \boldsymbol{x}_k) \tag{64}$$

where we have rewritten $\sigma^2(\boldsymbol{\lambda}; \boldsymbol{x}_*, \boldsymbol{x}_1, \ldots \boldsymbol{x}_k)$ to emphasize that $\boldsymbol{\lambda}$ is the optimization variable and $\boldsymbol{x}_*, \boldsymbol{x}_1, \ldots \boldsymbol{x}_k$ are fixed parameters defining a problem instance.

The optimal weights $\boldsymbol{\lambda}^*$ are found by minimizing the variance (**??**). Since (**??**) is a linear equation it can only has one critical point that occurs where the first derivative is zero

$$\frac{d\sigma^2}{d\boldsymbol{\lambda}} = 2\boldsymbol{K}\boldsymbol{\lambda} - 2\boldsymbol{\nu} . \tag{65}$$

Setting the above derivative to zero and solving for the optimal weights

$$\boldsymbol{\lambda}^* = \boldsymbol{K}^{-1}\boldsymbol{\nu} \tag{66}$$

Moreover, this point is a minimum if the Hessian

$$\frac{d^2\sigma^2}{d\boldsymbol{\lambda}^2} = 2\boldsymbol{K} \tag{67}$$

is positive definite. The matrix $\boldsymbol{K}$ is positive definite if for all vectors $\boldsymbol{x} \in \mathbb{R}^k$ the scalar $\boldsymbol{x}^T\boldsymbol{K}\boldsymbol{x} = 0$. The simple kriging estimator (**??**) can now be rewritten with the optimal weights as

$$\begin{aligned} f_*(\boldsymbol{x}_*) &= m + \sum_{i=1}^{k} \lambda_i^* (f(\boldsymbol{x}_i) - m) \\ &= m + (\boldsymbol{\lambda}^*)^T \bar{\boldsymbol{f}} \end{aligned} \tag{68}$$

where $\bar{\boldsymbol{f}} = [f(\boldsymbol{x}_1) - m \quad f(\boldsymbol{x}_2) - m \quad \cdots \quad f(\boldsymbol{x}_k) - m]^T$. Note that $\boldsymbol{\nu}$ is analagous to the matrix $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_*)$ in (**??**)–(**??**) and with with the substitution of $m = 0$ the expression for the predicted values is equivalent. Also, using (**??**).

and the corresponding variance is

$$\begin{aligned}
\sigma^2(\boldsymbol{x}_*) &= (\boldsymbol{\lambda}^*)^T \boldsymbol{K} \boldsymbol{\lambda}^* - 2(\boldsymbol{\lambda}^*)^T \boldsymbol{\nu} + k(\boldsymbol{x}_*, \boldsymbol{x}_*) && (69)\\
&= (\boldsymbol{K}^{-1}\boldsymbol{\nu})^T \boldsymbol{K} \boldsymbol{K}^{-1} \boldsymbol{\nu} - 2(\boldsymbol{K}^{-1}\boldsymbol{\nu})^T \boldsymbol{\nu} + k(\boldsymbol{x}_*, \boldsymbol{x}_*) && (70)\\
&= (\boldsymbol{K}^{-1}\boldsymbol{\nu})^T \boldsymbol{\nu} - 2(\boldsymbol{K}^{-1}\boldsymbol{\nu})^T \boldsymbol{\nu} + k(\boldsymbol{x}_*, \boldsymbol{x}_*) && (71)\\
&= \boldsymbol{\nu}^T (\boldsymbol{K}^{-1})^T \boldsymbol{\nu} - 2\boldsymbol{\nu}^T (\boldsymbol{K}^{-1})^T \boldsymbol{\nu} + k(\boldsymbol{x}_*, \boldsymbol{x}_*) && (72)\\
&= k(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{\nu}^T \boldsymbol{K}^{-1} \boldsymbol{\nu} && (73)
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{\mu}_{\boldsymbol{f}_*|\boldsymbol{f}} &= \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}) \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})^{-1} \boldsymbol{f} && (74)\\
\boldsymbol{P}_{\boldsymbol{f}_*|\boldsymbol{f}} &= \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}_*) - \boldsymbol{K}(\boldsymbol{X}_*, \boldsymbol{X}) \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X})^{-1} \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_*) && (75)
\end{aligned}$$

References

[1] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[3] R. A. Olea, *Geostatistics for Engineering and Earth Scientists*. Springer Science+Buisness Media, 1999.