Parameterization Schemes and Their Quality in Kernel Interpolation

Michael McCourt

January 2, 2014

Abstract

When performing interpolation with a kernel basis, the choice of kernel can play a significant role in the accuracy of the interpolation. Unfortunately, choosing a "good" kernel is a difficult proposition because of the wide variety of kernels available. To simplify this process, often a family of kernels is considered which differ by one or more free parameters while still retaining many of the same interpolation properties. Gaussians are an example of one such kernel, whose shape parameter allows for variable localization. Our goal is to consider several parameterization schemes for Gaussians and to develop a mechanism for comparing these schemes so that we can determine which is effective under what circumstances.

1 Introduction

Kernel interpolation is discussed in [2, 8, 7] and many other books. It is a useful tool for approximating scattered data and appears in both approximation theory (through reproducing kernel Hilbert spaces) and in spatial statistics (through Kriging). We believe that by leveraging knowledge from both of these fields we can improve upon the existing literature. There is another writeup of this content by Fred Hickernell available at

`http://math.ucdenver.edu/~mmccourt/hsp.pdf`

although the notation is somewhat in conflict with what I have written here. Still, it is a fantastic writeup and may help to supplement this discussion.

In both these settings, we are provided data

$$\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}, \qquad \boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}, \qquad \boldsymbol{x}_i \neq \boldsymbol{x}_j \text{ for } i \neq j,$$

where $\boldsymbol{x}_i \in \Omega \subseteq \mathbb{R}^d$ and $y_i \in \mathbb{R}$. Our goal is to somehow make "good" guesses about values of y at $\boldsymbol{x} \in \Omega$ locations where data does not exist.

A kernel $K : \Omega \times \Omega \to \mathbb{R}$ is any symmetric function of two variables. For this work, we restrict K to the set of *positive definite* kernels, which are kernels such that

$$\int_\Omega \int_\Omega K(\boldsymbol{x}, \boldsymbol{z})\, v(\boldsymbol{x})\, v(\boldsymbol{z})\, \mathrm{d}\boldsymbol{x}\, \mathrm{d}\boldsymbol{z} > 0$$

for any $v \in L^2(\Omega)$ not identically zero. This has another (more useful) definition involving the eigenvalues of a specific linear operator, but we'll worry about that if we need it.

Kernels can take lots of different forms and have lots of different properties, but probably the most common form of a kernel is a *radial basis function*, which is a kernel of the form

$$K(\boldsymbol{x}, \boldsymbol{z}) = \phi(\|\boldsymbol{x} - \boldsymbol{z}\|), \qquad \boldsymbol{x}, \boldsymbol{z} \in \Omega.$$

Notice that $\phi : \mathbb{R}^+ \to \mathbb{R}$ is a function of one variable instead of $\mathbb{R}^d$ which is one of the many reasons why radial kernels are nice. The kernel of main interest right now will be the Gaussian kernel

$$K(\boldsymbol{x}, \boldsymbol{z}) = \exp\left(-\varepsilon^2 \|\boldsymbol{x} - \boldsymbol{z}\|^2\right); \tag{1.1}$$

the norm used above is the 2-norm and the value $\varepsilon > 0$ is a *shape parameter* which can be chosen freely. The "correct" choice of ε is an important component in making "good" guesses about unobserved $(\boldsymbol{x}, y)$ values.

1.1 Scattered Data Interpolation

In the interpolation setting, we assume that data we are given $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ has been generated by some function f such that $y_i = f(\boldsymbol{x}_i)$. That function is deterministic, and we assume $f \in L^2(\Omega)$; it is also common to assume that f is in a certain Hilbert space, but we will state when that assumption is made explicitly.

To make guesses of y values at unobserved $\boldsymbol{x}$ locations, we create an interpolant s of f, and then evaluate $s(\boldsymbol{x})$. Because we are performing kernel interpolation, s uses a linear combination of kernel functions

$$s(\boldsymbol{x}) = \sum_{k=1}^{\hat{N}} c_k K(\boldsymbol{x}, \boldsymbol{z}_k),$$

where $\boldsymbol{z}_k$ for $1 \leq k \leq \hat{N}$ are the so-called *kernel centers* that distinguish the different elements of the basis.

Often times, we will prefer to write the interpolant as a vector inner product,

$$s(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^T \boldsymbol{c}, \tag{1.2}$$

using

$$\boldsymbol{k}(\boldsymbol{x}) = \begin{pmatrix} K(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ K(\boldsymbol{x}, \boldsymbol{x}_N) \end{pmatrix}, \qquad \boldsymbol{c} = \begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix}.$$

To be explicit, we should probably write $\boldsymbol{k}_{\mathcal{X}}(\boldsymbol{x})$ to remind ourselves that our basis is data-dependent, but we will omit it for now. To solve the interpolation problem we require $s(\boldsymbol{x}_i) = y_i$ for $1 \leq i \leq N$, which produces the linear system

$$\begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix} = \begin{pmatrix} s(\boldsymbol{x}_1) \\ \vdots \\ s(\boldsymbol{x}_N) \end{pmatrix} = \begin{pmatrix} \boldsymbol{k}(\boldsymbol{x}_1)^T \boldsymbol{c} \\ \vdots \\ \boldsymbol{k}(\boldsymbol{x}_N)^T \boldsymbol{c} \end{pmatrix} = \begin{pmatrix} \boldsymbol{k}(\boldsymbol{x}_1)^T \\ \vdots \\ \boldsymbol{k}(\boldsymbol{x}_N)^T \end{pmatrix} \boldsymbol{c},$$

or, more succinctly,

$$\mathsf{K}\boldsymbol{c} = \boldsymbol{y}, \qquad \mathsf{K} = \begin{pmatrix} \boldsymbol{k}(\boldsymbol{x}_1)^T \\ \vdots \\ \boldsymbol{k}(\boldsymbol{x}_N)^T \end{pmatrix}. \tag{1.3}$$

Because K is a positive definite kernel, we are guaranteed that K is a symmetric positive definite matrix. This implies that kernel-interpolation is well-defined in any dimension, in contrast to polynomial interpolation. Using this definition of $\boldsymbol{c}$ in (1.2) gives

$$s(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^T \mathsf{K}^{-1} \boldsymbol{y}, \tag{1.4}$$

which explicitly shows how values at unobserved locations are a weighted averaging of the given data.

To try to estimate the error associated with your kernel interpolation, some theory exists to help, although it may not be as useful as we would like. We can bound the pointwise error of the interpolant (a big deal) by the native space norm of the function (incomputable):

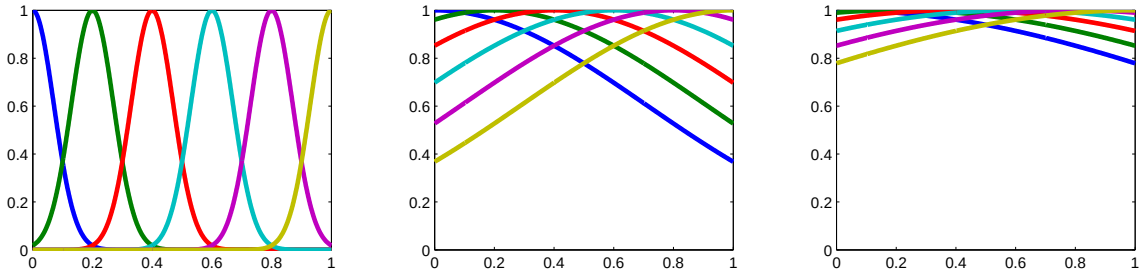
$$|f(\boldsymbol{x}) - s(\boldsymbol{x})| \leq P_{K,\mathcal{X}}(\boldsymbol{x}) \|f\|_{\mathcal{H}_K(\Omega)}, \qquad f \in \mathcal{H}_K(\Omega), \tag{1.5}$$

where $P_{K,\mathcal{X}}(\boldsymbol{x})$ is the so-called *power function*. The norm in $\mathcal{H}_K(\Omega)$ is called the *native space* norm, and that means the Hilbert space norm of f for the Hilbert space induced by the kernel K. Computing that is not possible without the Hilbert-Schmidt decomposition of the kernel, but note that the value of $\|f\|_{\mathcal{H}_K(\Omega)}$ is dependent on the kernel K. The power function is defined as

$$P_{K,\mathcal{X}}(\boldsymbol{x}) = \sqrt{K(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}(\boldsymbol{x})^T \mathsf{K}^{-1} \boldsymbol{k}(\boldsymbol{x})} \tag{1.6}$$

and comes about from some standard reproducing kernel manipulations [2] written up in the appendix.

In the definition of the Gaussian (1.1), there is an ε value which serves as the shape parameter of the basis: small ε produces flat kernels, and large ε produces peaked kernels. These differences can be seen in Figure 1.



(a) $\varepsilon = 10$, condition ≈ 1 (b) $\varepsilon = 1$, condition $\approx 10^6$ (c) $\varepsilon = .5$, condition $\approx 10^9$

Figure 1: Gaussian basis functions centered at 6 points in [0,1]. The choice of shape parameter affects the "width" or "locality" of the kernels. The condition of the K interpolation matrix is also listed.

1.1.1 Hilbert-Schmidt SVD

The following derivation took place in [1] so I'm not going to go too in depth here. I just want to introduce the notation for use later as necessary.

Positive definite kernels have eigenvalues and eigenfunctions defined by the Hilbert-Schmidt integral operator

$$\int_\Omega K(\boldsymbol{x},\boldsymbol{z})\varphi_n(\boldsymbol{z})\rho(\boldsymbol{z})\mathrm{d}\boldsymbol{z} = \lambda_n\varphi_n(\boldsymbol{x}), \tag{1.7}$$

where ρ is a suitable weight function. For Gaussians (our main focus right now) this discussion appears primarily in [3], where the values of ρ, λ_n and φ_n are explicitly stated. We assume that $\Omega = \mathbb{R}^d$. It is relevant to note that as $\varepsilon \to 0$, $\lambda_n \approx \varepsilon^{2n}$, which is the main cause of Gaussian ill-conditioning.

These eigenvalues are all positive, because K is a positive definite kernel, and decreasing, i.e., $\lambda_1 \geq \lambda_2 \geq \ldots > 0$; their (infinite) sum is finite because the Hilbert-Schmidt operator is a trace class operator. The eigenfunctions satisfy the orthonormality property

$$\int_\Omega \varphi_m(\boldsymbol{x})\varphi_n(\boldsymbol{x})\rho(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = \begin{cases} 1 & m = n, \\ 0 & m \neq n, \end{cases}$$

which might also be written as $\langle \varphi_m, \varphi_n \rangle_{L_2(\rho)} = \delta_{m,n}$. Another important orthogonality property of the eigenfunctions involves the Hilbert space inner product. Recall first the reproducing property

$$\langle f, K(\cdot,\boldsymbol{x})\rangle_{\mathcal{H}_K} = f(\boldsymbol{x}), \qquad f \in \mathcal{H}_K$$

which was discussed and used in the Appendix. Because $\varphi_m \in \mathcal{H}_K$ for $n = 1, 2, \ldots$, and using (1.7) to write $\varphi_m = \frac{1}{\lambda_m}\int_\Omega K(\cdot,\boldsymbol{z})\varphi_m(\boldsymbol{z})\rho(\boldsymbol{z})\mathrm{d}\boldsymbol{z}$, we know that

$$\begin{aligned} \langle \varphi_m, \varphi_n \rangle_{\mathcal{H}_K} &= \left\langle \frac{1}{\lambda_m}\int_\Omega K(\cdot,\boldsymbol{z})\varphi_m(\boldsymbol{z})\rho(\boldsymbol{z})\mathrm{d}\boldsymbol{z}, \varphi_n \right\rangle_{\mathcal{H}_K} \\ &= \frac{1}{\lambda_m}\int_\Omega \langle K(\cdot,\boldsymbol{z}), \varphi_n\rangle_{\mathcal{H}_K}\,\varphi_m(\boldsymbol{z})\rho(\boldsymbol{z})\mathrm{d}\boldsymbol{z} \\ &= \frac{1}{\lambda_m}\int_\Omega \varphi_m(\boldsymbol{z})\varphi_m(\boldsymbol{z})\rho(\boldsymbol{z})\mathrm{d}\boldsymbol{z} = \frac{\delta_{mn}}{\lambda_m} \end{aligned} \tag{1.8}$$

These eigenvalues/functions allow us to write the kernel K as a Mercer's series (or Hilbert-Schmidt series)

$$K(\boldsymbol{x},\boldsymbol{z}) = \sum_{m=1}^{\infty} \lambda_m\varphi_m(\boldsymbol{x})\varphi_m(\boldsymbol{z})$$

which means that the kernel basis vector $\boldsymbol{k}$ can be written as

$$\boldsymbol{k}(\boldsymbol{x})^T = \boldsymbol{\phi}(\boldsymbol{x})^T\Lambda\Phi^T$$

for $\boldsymbol{\phi}(\boldsymbol{x})^T = \begin{pmatrix}\varphi_1(\boldsymbol{x}) & \cdots & \varphi_N(\boldsymbol{x}) & \cdots\end{pmatrix}$, which is an infinite length vector, and

$$\Lambda = \begin{pmatrix} \lambda_1 & & & | & \\ & \ddots & & | & \\ — & — & \underline{\lambda_N} & | & — \\ & & & | & \ddots \end{pmatrix} = \begin{pmatrix} \Lambda_1 & \\ & \Lambda_2 \end{pmatrix}, \qquad \Phi = \begin{pmatrix} \boldsymbol{\phi}(\boldsymbol{x}_1)^T \\ \vdots \\ \boldsymbol{\phi}(\boldsymbol{x}_N)^T \end{pmatrix} = \begin{pmatrix} \Phi_1 & \Phi_2 \end{pmatrix}$$

where $\Lambda_1, \Phi_1 \in \mathbb{R}^{N\times N}$ and Λ_2 and Φ_2 are the (infinite-sized) rest of the matrices Λ and Φ respectively. Eigenfunctions can be chosen so that Φ_1^{-1} exists, and, because K is a positive definite kernel, $\lambda_n > 0$ for $1 \le n < \infty$ thus we know that Λ_1^{-1} exists. This allows us to write

$$\Lambda\Phi^T = \begin{pmatrix} \Lambda_1 & \\ & \Lambda_2 \end{pmatrix} \begin{pmatrix} \Phi_1^T \\ \Phi_2^T \end{pmatrix} = \begin{pmatrix} \mathsf{I}_N \\ \Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1} \end{pmatrix} \Lambda_1\Phi_1^T.$$

Using this in our $\boldsymbol{k}$ definition above gives

$$\boldsymbol{k}(\boldsymbol{x})^T = \underbrace{\boldsymbol{\phi}(\boldsymbol{x})^T \begin{pmatrix} \mathsf{I}_N \\ \Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1} \end{pmatrix}}_{\boldsymbol{\psi}(\boldsymbol{x})^T} \Lambda_1\Phi_1^T = \boldsymbol{\psi}(\boldsymbol{x})^T\Lambda_1\Phi_1^T. \tag{1.9}$$

Substituting this into our interpolation matrix K from (1.3) gives

$$\mathsf{K} = \begin{pmatrix} \boldsymbol{k}(\boldsymbol{x}_1)^T \\ \vdots \\ \boldsymbol{k}(\boldsymbol{x}_N)^T \end{pmatrix} = \begin{pmatrix} \boldsymbol{\psi}(\boldsymbol{x}_1)^T\Lambda_1\Phi_1^T \\ \vdots \\ \boldsymbol{\psi}(\boldsymbol{x}_N)^T\Lambda_1\Phi_1^T \end{pmatrix} = \begin{pmatrix} \boldsymbol{\psi}(\boldsymbol{x}_1)^T \\ \vdots \\ \boldsymbol{\psi}(\boldsymbol{x}_N)^T \end{pmatrix} \Lambda_1\Phi_1^T = \Psi\Lambda_1\Phi_1^T, \tag{1.10}$$

which is the *Hilbert-Schmidt SVD*. This is a useful decomposition because much of the ill-conditioning of K resides in Λ_1 (subject to some appropriate design choices discussed in [3]). Using (1.10) and (1.9) in (1.4) gives

$$s(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^T\mathsf{K}^{-1}\boldsymbol{y} = \boldsymbol{\psi}(\boldsymbol{x})^T\Lambda_1\Phi_1^T(\Psi\Lambda_1\Phi_1^T)^{-1}\boldsymbol{y} = \boldsymbol{\psi}(\boldsymbol{x})^T\Psi^{-1}\boldsymbol{y} \tag{1.11}$$

which shows that the Hilbert-Schmidt SVD allows for a change of basis from the unstable basis $\boldsymbol{k}(\boldsymbol{x})$ to the stable basis $\boldsymbol{\psi}(\boldsymbol{x})$.

1.2 Kriging

This section I am less confident about, especially regarding notation, so please correct as needed. We'll start with the idea of a probability space $(\mathcal{W}, \mathcal{A}, P)$ where $\mathcal{W}$ is the sample space of all possible outcomes, $\mathcal{A}$ is a σ-algebra and P is a probability measure. Normally, I think statisticians and probabilists would use Ω instead of $\mathcal{W}$, but we're already using Ω for something else.

We need to define a parameter space Ω, which for our purposes right now will be $\Omega = \mathbb{R}^d$. In numerical analysis, this is the domain over which we would consider evaluating our interpolant. A function $Y : \Omega, \mathcal{W} \to \mathbb{R}$ (evaluated as $Y(\boldsymbol{x}, \omega)$ for $\boldsymbol{x} \in \Omega$ and $\omega \in \mathcal{W}$) is a random field if, for every $\boldsymbol{x} \in \Omega$, Y is an $\mathcal{A}$-measurable function of ω. Our notation for this is

$$\text{Random Field:} \quad Y = \{Y_{\boldsymbol{x}}\}_{\boldsymbol{x}\in\Omega}.$$

Note that

- For a fixed $\boldsymbol{x}$, $Y_{\boldsymbol{x}} = Y(\boldsymbol{x}, \cdot)$ is a random variable.
- For a fixed ω, $y(\cdot) = Y(\cdot, \omega)$ is a deterministic function of $\boldsymbol{x}$ referred to as a realization of the random field.

The mean of Y is a function μ_Y which is defined at any point $\boldsymbol{x} \in \Omega$ as

$$\mu_Y(\boldsymbol{x}) = \mathbb{E}(Y_{\boldsymbol{x}}) = \int_{\mathcal{W}} Y_{\boldsymbol{x}}(\omega)\mathrm{d}P(\omega) = \int_{\mathbb{R}} y\mathrm{d}F_{Y_{\boldsymbol{x}}}(y)$$

where $F_{Y_{\boldsymbol{x}}}$ is the cumulative distribution function of $Y_{\boldsymbol{x}}$. For our purposes we will assume that $Y_{\boldsymbol{x}}$ is continuous, thus we can write

$$\mu_Y(\boldsymbol{x}) = \int_{\mathbb{R}} y p_{Y_x}(y) \mathrm{d}y$$

for density function p_{Y_x}. Likewise, we will define the covariance kernel of Y as

$$K(\boldsymbol{x}, \boldsymbol{z}) = \mathrm{Cov}(Y_{\boldsymbol{x}}, Y_{\boldsymbol{z}}) = \mathbb{E}((Y_{\boldsymbol{x}} - \mu_Y(\boldsymbol{x}))(Y_{\boldsymbol{z}} - \mu_Y(\boldsymbol{z}))) = \mathbb{E}(Y_{\boldsymbol{x}} Y_{\boldsymbol{z}}) - \mu_Y(\boldsymbol{x})\mu_Y(\boldsymbol{z}), \tag{1.12}$$

after some appropriate manipulations. We are not using the notation K_Y here, unlike μ_Y, to later emphasize the similarities between Kriging and kernel approximation.

Our assumption here is that the random field Y is a *Gaussian random field*, which we denote as

$$Y \sim GF(\mu_Y, K).$$

This implies that, for a finite set of points $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \in \Omega$, the vector of random variables $\boldsymbol{Y}_{\mathcal{X}} = \begin{pmatrix} Y_{\boldsymbol{x}_1} & \cdots & Y_{\boldsymbol{x}_N} \end{pmatrix}^T$ has the distribution

$$\boldsymbol{Y}_{\mathcal{X}} \sim \mathcal{N}(\boldsymbol{\mu}, \mathsf{K}), \tag{1.13}$$

where $\boldsymbol{\mu} = \mathbb{E}(\boldsymbol{Y}_{\mathcal{X}})$ and $(\mathsf{K})_{i,j} = \mathrm{Cov}(Y_{\boldsymbol{x}_i}, Y_{\boldsymbol{x}_j})$. This multivariate normal distribution has the density

$$p_{\boldsymbol{Y}_{\mathcal{X}}}(\boldsymbol{y}) = \frac{1}{\sqrt{2\pi \det(\mathsf{K})}} \exp\left(-\frac{1}{2}(\boldsymbol{y} - \boldsymbol{\mu})^T \mathsf{K}^{-1} (\boldsymbol{y} - \boldsymbol{\mu})\right), \tag{1.14}$$

and the term $(\boldsymbol{y}-\boldsymbol{\mu})^T \mathsf{K}^{-1} (\boldsymbol{y}-\boldsymbol{\mu})$ is sometimes referred to as the Mahalanobis distance. Note that K will be symmetric positive definite for a positive definite covariance kernel K, which must be positive definite or else a negative variance could occur. As a result, K^{-1} must exist, though it may be ill-conditioned.

At this point, we are going to restrict our concern to **zero-mean** Gaussian processes, which demands that $\mu_Y \equiv 0$; in Kriging literature, this is referred to as "simple Kriging". [MJM] What penalties does this restriction incur? This restriction simplifies the situation significantly, most notably by eliminating the mean terms from (1.12),

$$K(\boldsymbol{x}, \boldsymbol{z}) = \mathrm{Cov}(Y_{\boldsymbol{x}}, Y_{\boldsymbol{z}}) = \mathbb{E}(Y_{\boldsymbol{x}} Y_{\boldsymbol{z}}), \tag{1.15}$$

and from (1.14)

$$p_{\boldsymbol{Y}_{\mathcal{X}}}(\boldsymbol{y}) = \frac{1}{\sqrt{2\pi \det(\mathsf{K})}} \exp\left(-\frac{1}{2}\boldsymbol{y}^T \mathsf{K}^{-1} \boldsymbol{y}\right), \tag{1.16}$$

This also produces a duality between the reproducing kernel Hilbert space $\mathcal{H}_K(\Omega)$ defined in Section 1.1 and the Hilbert space $\mathcal{H}_Y$ which is the set of all linear combinations of random variables $Y_{\boldsymbol{x}}$ together with their $L_2(\Omega, \mathcal{A}, P)$-limits. The *Loève representation theorem* says that the inner products in these Hilbert spaces are identical, i.e.,

$$\langle Y_{\boldsymbol{x}}, Y_{\boldsymbol{z}} \rangle_{\mathcal{H}_Y} = \mathbb{E}(Y_{\boldsymbol{x}} Y_{\boldsymbol{z}}) = K(\boldsymbol{x}, \boldsymbol{z}) = \langle K(\cdot, \boldsymbol{x}), K(\cdot, \boldsymbol{z}) \rangle_{\mathcal{H}_K},$$

where the final identity occurs because of the reproducing property of K. We believe that this implies a connection between the eigenfunctions φ_m defined in Section 1.1.1 and elements of the Karhunen-Loève expansion, but I'm not going to write about that here.

The biggest advantage of assuming zero-mean Gaussian fields comes when making predictions for unobserved values of $Y_{\boldsymbol{x}_0}$, $\boldsymbol{x}_0 \notin \mathcal{X}$. After a lengthy derivation (which can be found in a talk I gave at one point) involving the joint distribution of $(\boldsymbol{Y}_{\mathcal{X}}, Y_{\boldsymbol{x}_0})$, it can be shown that the conditional density for the random variable $Y_{\boldsymbol{x}_0}$ given a realization $\boldsymbol{y}$ of the multivariate normal random variable $\boldsymbol{Y}_{\mathcal{X}}$ is

$$p_{Y_{\boldsymbol{x}_0}}(y | \boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y}) \propto \exp\left(-\frac{1}{2}(y - \bar{\mu}) C^{-1} (y - \bar{\mu})\right), \tag{1.17}$$

where $\bar{\mu} = \mu_Y(\boldsymbol{x}_0) + \boldsymbol{k}(\boldsymbol{x})^T \mathsf{K}^{-1} (\boldsymbol{y} - \boldsymbol{\mu})$ and $C = K(\boldsymbol{x}_0, \boldsymbol{x}_0) - \boldsymbol{k}(\boldsymbol{x}_0)^T \mathsf{K}^{-1} \boldsymbol{k}(\boldsymbol{x}_0)$. This implies that $Y_{\boldsymbol{x}_0} | \boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y}$ is a normal random variable (no surprise there, that the conditional of a normal is normal) with mean $\bar{\mu}$ and variance C; it is not a covariance here because we consider only one point $\boldsymbol{x}_0$.

By enforcing our zero-mean assumption, both $\mu_Y(\boldsymbol{x}_0) = 0$ and $\boldsymbol{\mu} = 0$, meaning that

$$Y_{\boldsymbol{x}_0}|\boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{k}(\boldsymbol{x}_0)^T \mathsf{K}^{-1}\boldsymbol{y} \ , \ K(\boldsymbol{x}_0, \boldsymbol{x}_0) - \boldsymbol{k}(\boldsymbol{x}_0)^T \mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x}_0) \right). \tag{1.18}$$

Thus, the best linear unbiased predictor for the zero-mean Gaussian field Y at a point $x_0 \in \Omega$ is

$$\mathbb{E}(Y_{\boldsymbol{x}_0}|\boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y}) = \boldsymbol{k}(\boldsymbol{x}_0)^T \mathsf{K}^{-1}\boldsymbol{y}, \tag{1.19}$$

which through some miracle of mathematics is identical to (1.2) if you consider the function values in the numerical analysis setting $\boldsymbol{y}$ to be equal to the Gaussian random field realization $\boldsymbol{y}$. In another awesome twist, the power function (1.6) evaluated at $\boldsymbol{x}_0$ perfectly matches the so-called Kriging variance:

$$\mathrm{Var}(Y_{\boldsymbol{x}_0}|\boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y}) = K(\boldsymbol{x}_0, \boldsymbol{x}_0) - \boldsymbol{k}(\boldsymbol{x}_0)^T \mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x}_0) = P_{K,\mathcal{X}}(\boldsymbol{x}_0)^2.$$

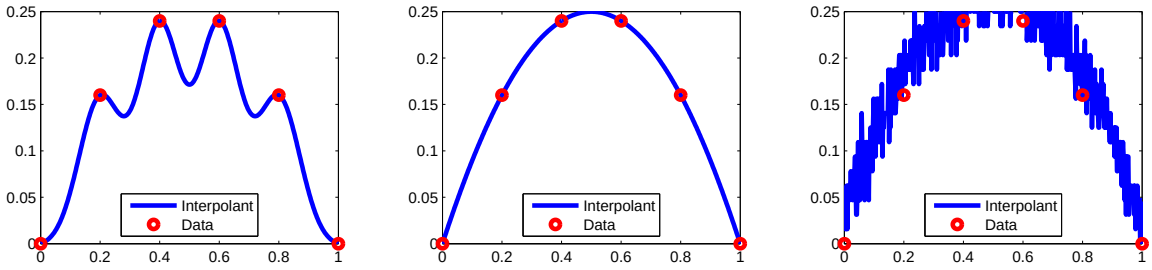
This suggests that minimizing the power function, a wholly analytic approach to reducing the error in the interpolant, has the effect of minimizing the variance of our Kriging prediction, which is a wholly statistical approach to reducing the "error" in the prediction.

1.3 Parameterizing Kernels

So far, we have introduced two ways that we can approach this scattered data interpolation problem, which under certain circumstances yield the same result. We have assumed that the kernel which defines the interpolating basis $\boldsymbol{k}(\boldsymbol{x})$, and defines the covariance of the Gaussian random field Y, is a Gaussian kernel (1.1).

Within that Gaussian kernel $K(\boldsymbol{x}, \boldsymbol{z}) = \exp(-\varepsilon^2\|\boldsymbol{x} - \boldsymbol{z}\|^2)$ is a free parameter $\varepsilon > 0$ which we have until now largely ignored. It can be proved that any value of ε will still produce a positive definite interpolation matrix K, thus the interpolant (1.2) is guaranteed to exist and the covariance of $Y_{\mathcal{X}}$ will be a positive definite matrix.

This theoretical result does not suggest, however, that all ε values will produce equally effective interpolants and predictions, nor does it suggest that the matrices K will be well conditioned. Indeed, the predictive capacity of $s(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^T \mathsf{K}^{-1}\boldsymbol{y}$ is greatly dependent on the shape parameter ε, as is indicated in Figure 2.

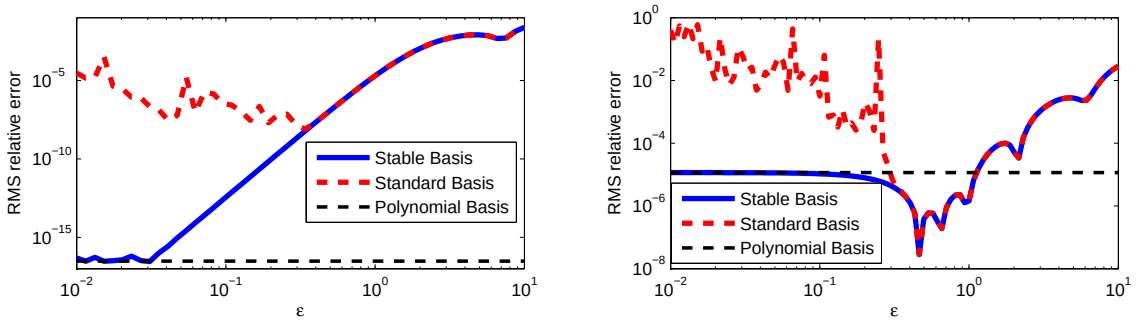


(a) $\varepsilon = 10$, condition ≈ 1 (b) $\varepsilon = 1$, condition $\approx 10^6$ (c) $\varepsilon = .0003$, condition $\approx 10^{19}$

Figure 2: Different shape parameters will produce interpolants that look different. When ε is too large, the basis functions are too localized and predictions suffer. When ε is too small, K is very ill-conditioned, yielding severe cancelation errors, and possibly preventing $s(\boldsymbol{x}_i) = y_i$ for some of the given data. The function which created this data is $f(x) = x(1-x)$.

The severe ill-conditioning which cripples the result in Figure 2c can be overcome with (1.11). But even with a stable result, as $\varepsilon \to 0$ the Gaussian interpolant approaches the polynomial interpolant (or some polynomial approximation in higher dimensions) meaning that kernels have the potential to exceed polynomial accuracy for $\varepsilon > 0$. This is, at least in part, a result of the Runge phenomenon: polynomials are global basis functions, so because Gaussians have the ability to localize they have the ability to minimize the Runge phenomenon. This concept is demonstrated graphically in Figure 3.

Figure 3b demonstrates the opportunity for both success and failure when choosing ε. There is a minimum error attainable for $\varepsilon \approx .4642$, but if ε is chosen slightly smaller or larger the error could be orders of magnitude worse. Of course, this graph can only be created with the true function that generated the data, and that will be unavailable in many applications. This motivates the search for parameterization schemes that may allow us to find good ε values using just the available data.



(a) $f(x) = x^2$ (b) $f(x) = \cos(\pi x)$

Figure 3: In this example, 10 evenly spaced points in $(-1, 1)$ are sampled and Gaussian interpolation is conducted with a range of ε values. These results are compared to the polynomial interpolant of the same data, and the $\varepsilon \to 0$ limit is confirmed. This limit cannot be attained with the standard basis, but the stable basis succeeds. The error is averaged over 100 evenly spaced points.

2 Existing Parameterization Methods

As we work towards trying to identify a suitable shape parameter, we recall existing techniques in the literature which have had variable levels of success. Each of these existing parameterization techniques has some objective function which needs to be minimized, and that is what we will introduce now. [MJM] Maybe talk briefly about how a lot of people just kinda guess, or look at small examples to gain intuition, or something? Also, expert knowledge?

2.1 The power function, a.k.a., the Kriging variance

As we saw in (1.5), the error in a scattered data interpolation is bounded by the power function times the Hilbert space norm of f. Our choice of parameter ε will have an effect on $\|f\|_{\mathcal{H}_K}$, but it is not one that we can immediately understand. [MJM] Maybe I'll type more up about this.

We are, however, able to measure how ε affects the power function, and this motivates one strategy that people have used to optimize their kernels: minimize the power function to minimize the pointwise interpolant error. In this setting, the objective function is

$$\mathrm{C}_{\mathrm{POWER}}(\varepsilon; k) = \|P_{K,\mathcal{X}}\|_k, \tag{2.1}$$

where the k-norm can be chosen in one of several ways (2-norm, ∞-norm, $\mathcal{H}_K$ norm). If, at the time of interpolation, you know at which points $\hat{\mathcal{X}}$ you want to evaluate s, you can choose a discrete norm of $P_{K,\mathcal{X}}(\hat{x}_i)$, $1 \le i \le \hat{N}$, to minimize the error at those points. If you do not know those points, you can just approximate the $L^2(\Omega)$ norm.

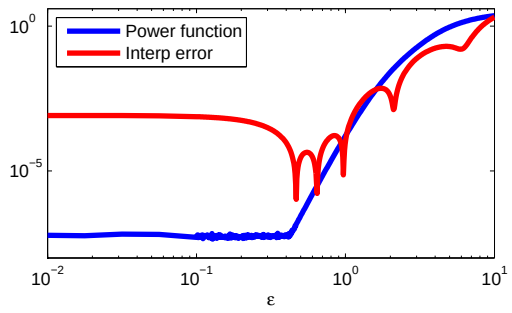
Historically, the power function has been unstable during computation because of the presence of the K^{-1} term, which is notoriously ill-conditioned. Using the Hilbert-Schmidt SVD Section 1.1.1 relieves this difficulty because $\boldsymbol{k}(\boldsymbol{x})^T\mathsf{K}^{-1} = \boldsymbol{\psi}(\boldsymbol{x})^T\Psi^{-1}$:

$$P_{K,\mathcal{X}}(\boldsymbol{x}) = \sqrt{K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{\psi}(\boldsymbol{x})^T\Psi^{-1}\boldsymbol{k}(\boldsymbol{x})}.$$

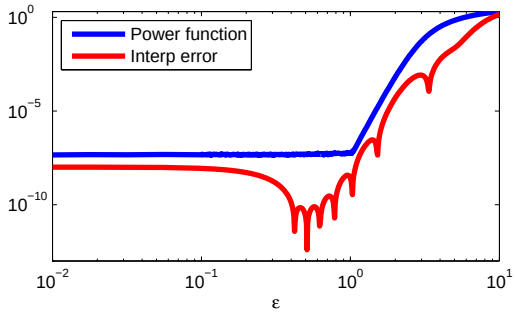
See Figure 4 for some examples measuring the 2-norm of the power function compared to the error of the interpolation.

In these graphs, $\mathrm{C}_{\mathrm{POWER}}(\varepsilon; 2)$ seems to decrease until reaching roughly 10^{-8}; this unfortunately seems to be the result of numerical cancelation which occurs as $\boldsymbol{\psi}(\boldsymbol{x})^T\Psi^{-1}\boldsymbol{k}(\boldsymbol{x}) \to 1$ for $\varepsilon \to 0$. Fixing this may be possible if we perform additional matrix algebra on $P_{K,\mathcal{X}}$ using the Hilbert-Schmidt SVD:

$$\begin{aligned} P_{K,\mathcal{X}}(\boldsymbol{x})^2 &= K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}(\boldsymbol{x})^T\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x}) \\ &= K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{\psi}(\boldsymbol{x})^T\Lambda_1\Phi_1^T\Phi_1^{-T}\Lambda_1^{-1}\Psi^{-1}\Phi_1\Lambda_1\boldsymbol{\psi}(\boldsymbol{x}) \\ &= K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{\psi}(\boldsymbol{x})^T\Psi^{-1}\Phi_1\Lambda_1\boldsymbol{\psi}(\boldsymbol{x}) \end{aligned}$$



(a) $f(x) = \cos(\pi x)$, $N = 10$ uniform points



(b) $f(x) = J_0(4|x|)$, $N = 15$ chebyshev points

Figure 4: In this example, N points in $(-1, 1)$ are sampled and Gaussian interpolation is conducted with a range of ε values. These results are compared to $\mathrm{C}_{\mathrm{POWER}}(\varepsilon; 2)$, which seems to flatten out for small values of ε. This suggests that minimizing $P_{K,\mathcal{X}}$ alone may not be sufficient to obtain an optimal epsilon, though it may help in finding an appropriate region. The error is averaged over 100 evenly spaced points.

To simplify this, we will need to analyze $\Psi^{-1}\Phi_1\Lambda_1$ recalling (1.9):

$$\begin{aligned}
\Psi^{-1}\Phi_1\Lambda_1 &= \left(\begin{pmatrix} \Phi_1 & \Phi_2 \end{pmatrix} \begin{pmatrix} \mathsf{I}_N \\ \Lambda_2 \Phi_2^T \Phi_1^{-T} \Lambda_1^{-1} \end{pmatrix} \right)^{-1} \Phi_1\Lambda_1 \\
&= \left(\Phi_1 + \Phi_2\Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1} \right)^{-1} \Phi_1\Lambda_1 \\
&= \left(\Lambda_1^{-1} + \Lambda_1^{-1}\Phi_1^{-1}\Phi_2\Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1} \right)^{-1} \\
&= \Lambda_1^{1/2} \left(\mathsf{I}_N + \Lambda_1^{-1/2}\Phi_1^{-1}\Phi_2\Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1/2} \right)^{-1} \Lambda_1^{1/2} \\
&= \Lambda_1^{1/2} \left(\mathsf{I}_N + \mathsf{L}\mathsf{L}^T \right)^{-1} \Lambda_1^{1/2},
\end{aligned}$$

where we have defined $\mathsf{L} = \Lambda_2^{1/2}\Phi_2\Phi_1^{-1}\Lambda_1^{-1/2}$. We want to use the von Neumann series to write

$$\left(\mathsf{I}_N + \mathsf{L}\mathsf{L}^T \right)^{-1} = \sum_{k=0}^{\infty} (-1)^k (\mathsf{L}\mathsf{L}^T)^k = \mathsf{I}_N - \mathsf{L}\mathsf{L}^T + (\mathsf{L}\mathsf{L}^T)^2 - \dots , \tag{2.2}$$

which is only allowed if $\|\mathsf{L}\mathsf{L}^T\|_2 \leq 1$. Studying the structure of L shows

$$\begin{aligned}
\|\mathsf{L}\mathsf{L}^T\|_2 \leq \|\mathsf{L}\|_2^2 &\leq \|\Lambda_2^{1/2}\|_2^2 \, \|\Lambda_1^{-1/2}\|_2^2 \, \|\Phi_2\Phi_1^{-1}\|_2^2 \\
&= (\varepsilon^{N+1})^2 \quad (\varepsilon^{-N})^2 \quad \|\Phi_2\Phi_1^{-1}\|_2^2 = \varepsilon^2 \|\Phi_2\Phi_1^{-1}\|_2^2
\end{aligned}$$

using the submultiplicativity of induced matrix norms. This implies that there is an ε small enough such that the series (2.2) can be used. Reinspecting the power function for sufficiently small ε with this expansion shows

$$\begin{aligned}
P_{K,\mathcal{X}}(\boldsymbol{x})^2 &= K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{\psi}(\boldsymbol{x})^T \Lambda_1^{1/2} \left(\mathsf{I}_N - \mathsf{L}\mathsf{L}^T + (\mathsf{L}\mathsf{L}^T)^2 - \dots \right) \Lambda_1^{1/2} \boldsymbol{\psi}(\boldsymbol{x}) \\
&= K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{\psi}(\boldsymbol{x})^T \Lambda_1 \boldsymbol{\psi}(\boldsymbol{x}) + \boldsymbol{\psi}(\boldsymbol{x})^T \Lambda_1^{1/2} \mathsf{L}\mathsf{L}^T \Lambda_1^{1/2} \boldsymbol{\psi}(\boldsymbol{x}) - \dots \\
&\approx K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{\psi}(\boldsymbol{x})^T \Lambda_1 \boldsymbol{\psi}(\boldsymbol{x}),
\end{aligned}$$

where the final approximation becomes more valid as $\varepsilon \to 0$. Because of the swift decay of the eigenvalues it is likely that the terms in $\boldsymbol{\psi}(\boldsymbol{x})^T \Lambda_1 \boldsymbol{\psi}(\boldsymbol{x})$ will need to be separated out to avoid cancelation. Thus we may prefer to write

$$P_{K,\mathcal{X}}(\boldsymbol{x})^2 \approx K(\boldsymbol{x},\boldsymbol{x}) - \sum_{n=1}^{N} \lambda_n \psi_n(\boldsymbol{x})^2 = K(\boldsymbol{x},\boldsymbol{x}) - \lambda_1\psi_1(\boldsymbol{x})^2 - \lambda_2\psi_2(\boldsymbol{x})^2 - \dots , \tag{2.3}$$

so that the magnitude of the first few terms, which are likely on the same order as $K(\boldsymbol{x},\boldsymbol{x})$, do not overwhelm the later terms.

2.1.1 The Golomb-Weinberger Error bound

At the end of the Appendix, we suggest that the standard error bound (1.5) could be made computable if our predictive accuracy on Ω is good enough:

$$\|f - s\|_{\mathcal{H}_K} \leq \delta_\varepsilon \|s\|_{\mathcal{H}_K}.$$

This term δ_ε is presumably a small number if we have done a good job of approximating f; the ε term is left there just to show that it is not independent of ε as is suggested in Figure 2 for large ε.

Using this assumption, and (3.3), we can write (1.5) as

$$|f(\boldsymbol{x}) - s(\boldsymbol{x})| \leq \delta_\varepsilon \|s\|_{\mathcal{H}_K} P_{K,\mathcal{X}}(\boldsymbol{x}), \tag{2.4}$$

which is referred to as the Golomb-Weinberger bound. We determined $\|s\|_{\mathcal{H}_K} = \sqrt{\boldsymbol{y}^T \mathsf{K}^{-1} \boldsymbol{y}}$ in (3.4), so by assuming that we have done a decent job approximating f, i.e., $\|f - s\|_{\mathcal{H}_K}$ is small, our new criterion for optimizing ε is

$$\mathrm{C}_{\mathrm{GW}}(\varepsilon; k) = \sqrt{\boldsymbol{y}^T \mathsf{K}^{-1} \boldsymbol{y}}\, \|P_{K,\mathcal{X}}\|_k. \tag{2.5}$$

We have already discussed in Section 2.1 how computing the power function can be numerically unstable and requires the Hilbert-Schmidt SVD; we also discussed how $P_{K,\mathcal{X}}$ is subject to cancelation, but we'll forget that for a moment. The presence of K^{-1} in the $\boldsymbol{y}^T \mathsf{K}^{-1} \boldsymbol{y}$ term seems dangerous because of ill-conditioning in K. Indeed, direct computation of this term is not recommended, and even with the Hilbert-Schmidt SVD the situation is still troubling. Simple substitution leads to

$$\|s\|^2_{\mathcal{H}_K} = \boldsymbol{y}^T \mathsf{K}^{-1} \boldsymbol{y} = \boldsymbol{y}^T (\Psi \Lambda_1 \Phi_1^T)^{-1} \boldsymbol{y} = \boldsymbol{y}^T \Phi_1^{-T} \Lambda_1^{-1} \Psi^{-1} \boldsymbol{y} = \boldsymbol{y}_\Phi^T \Lambda_1^{-1} \boldsymbol{y}_\Psi, \tag{2.6}$$

where $\Phi_1 \boldsymbol{y}_\Phi = \boldsymbol{y}$ and $\Psi \boldsymbol{y}_\Psi = \boldsymbol{y}$. Those terms are (presumably) safe to compute, or else the stable basis cannot solve the interpolation problem.

Even so, this term $\boldsymbol{y}_\Phi^T \Lambda_1^{-1} \boldsymbol{y}_\Psi$ may not be safe to compute as it is a weighted inner product between $\boldsymbol{y}_\Phi^T$ and $\boldsymbol{y}_\Psi$. This computation is therefore subject to numerical cancelation if the vectors are on different scales; in experiments we have even seen this value computed to be a negative number, despite the fact that $\|s\|_{\mathcal{H}_K} \geq 0$.

Although this direct computation has proven unreliable in practice, further manipulations will allow us to bound $\|s\|_{\mathcal{H}_K}$, and the bound will become tighter as $\varepsilon \to 0$. The first step is to define $\Psi \boldsymbol{b} = \boldsymbol{y}$ - our goal will be to compute

$$\boldsymbol{y}^T \mathsf{K}^{-1} \boldsymbol{y} = \boldsymbol{b}^T \mathsf{B} \boldsymbol{b}$$

for some matrix B. This vector $\boldsymbol{b} = \Psi^{-1} \boldsymbol{y}$ is safe to compute if $\boldsymbol{\psi}$ is indeed a stable basis, and this symmetric inner product will not be subject to the same numerical issues as (2.6). To find B we may perform some manipulations involving the HS-SVD $\mathsf{K} = \Psi \Lambda_1 \Phi_1^T$,

$$\begin{aligned} \boldsymbol{y}^T \mathsf{K}^{-1} \boldsymbol{y} &= \boldsymbol{y}^T \Phi_1^{-T} \Lambda_1^{-1} \Psi^{-1} \boldsymbol{y} \\ &= (\Psi \boldsymbol{b})^T \Phi_1^{-T} \Lambda_1^{-1} \Psi^{-1} \Psi \boldsymbol{b} \\ &= \boldsymbol{b}^T \Psi^T \Phi_1^{-T} \Lambda_1^{-1} \boldsymbol{b}. \end{aligned} \tag{2.7}$$

Thus $\mathsf{B} = \Psi^T \Phi_1^{-T} \Lambda_1^{-1}$, although at first glance this does not seem very helpful.

To simplify $\Psi^T \Phi_1^{-T} = (\Phi_1^{-1} \Psi)^T$, we will recall the structure of $\boldsymbol{\psi}$ from (1.9):

$$\boldsymbol{\psi}(\boldsymbol{x})^T = \boldsymbol{\phi}(\boldsymbol{x})^T \begin{pmatrix} \mathsf{I}_N \\ \Lambda_2 \Phi_2^T \Phi_1^{-T} \Lambda_1^{-1} \end{pmatrix} \quad \Rightarrow \quad \Psi = \begin{pmatrix} \Phi_1 & \Phi_2 \end{pmatrix} \begin{pmatrix} \mathsf{I}_N \\ \Lambda_2 \Phi_2^T \Phi_1^{-T} \Lambda_1^{-1} \end{pmatrix},$$

where we have used the fact that $\Phi = \begin{pmatrix} \Phi_1 & \Phi_2 \end{pmatrix}$. Substituting this into $\Phi_1^{-1} \Psi$ gives

$$\begin{aligned} \Phi_1^{-1} \Psi &= \Phi_1^{-1} \begin{pmatrix} \Phi_1 & \Phi_2 \end{pmatrix} \begin{pmatrix} \mathsf{I}_N \\ \Lambda_2 \Phi_2^T \Phi_1^{-T} \Lambda_1^{-1} \end{pmatrix} \\ &= \begin{pmatrix} \mathsf{I}_N & \Phi_1^{-1} \Phi_2 \end{pmatrix} \begin{pmatrix} \mathsf{I}_N \\ \Lambda_2 \Phi_2^T \Phi_1^{-T} \Lambda_1^{-1} \end{pmatrix} \\ &= \mathsf{I}_N + \Phi_1^{-1} \Phi_2 \Lambda_2 \Phi_2^T \Phi_1^{-T} \Lambda_1^{-1}. \end{aligned}$$

Using this result in (2.7) gives

$$\begin{aligned}
\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y} &= \boldsymbol{b}^T(\Phi_1^{-1}\Psi)^T\Lambda_1^{-1}\boldsymbol{b} \\
&= \boldsymbol{b}^T\left(\mathsf{I}_N + \Phi_1^{-1}\Phi_2\Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\right)^T\Lambda_1^{-1}\boldsymbol{b} \\
&= \boldsymbol{b}^T\Lambda_1^{-1}\boldsymbol{b} + \boldsymbol{b}^T\Lambda_1^{-1}\Phi_1^{-1}\Phi_2\Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\boldsymbol{b} \qquad (2.8)\\
&= \boldsymbol{b}^T\Lambda_1^{-1}\boldsymbol{b} + (\Lambda_2^{1/2}\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\boldsymbol{b})^T(\Lambda_2^{1/2}\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\boldsymbol{b}) \\
&\geq \boldsymbol{b}^T\Lambda_1^{-1}\boldsymbol{b}, \qquad (2.9)
\end{aligned}$$

where the inequality is true because $\|\Lambda_2^{1/2}\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\boldsymbol{b}\|_2^2 \geq 0$. Here, $\Lambda_2^{1/2}$ is defined as the diagonal matrix with the positive square roots of Λ_2 on its diagonal.

This result implies that if we use the Hilbert-Schmidt SVD we can bound the native space norm of the interpolant from below by $\sqrt{\boldsymbol{b}^T\Lambda_1^{-1}\boldsymbol{b}}$. This term is safely computable because it is a symmetric inner product, meaning that cancelation will not occur:

$$\boldsymbol{b}^T\Lambda_1^{-1}\boldsymbol{b} = \sum_{n=1}^{N}\frac{b_n^2}{\lambda_n}.$$

We can reach this result another way, by instead studying $\|s\|_{\mathcal{H}_K}^2$ with the Hilbert-Schmidt SVD (recalling (1.11) and (1.9))

$$\begin{aligned}
\|s\|_{\mathcal{H}_K}^2 &= \langle s^T, s\rangle_{\mathcal{H}_K} \\
&= \langle \boldsymbol{y}^T\Psi^T\boldsymbol{\psi}(\cdot), \boldsymbol{\psi}(\cdot)^T\Psi\boldsymbol{y}\rangle_{\mathcal{H}_K} \\
&= \boldsymbol{y}^T\Psi^T\left\langle \begin{pmatrix}\mathsf{I}_N\\ \Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\end{pmatrix}^T\boldsymbol{\phi}(\cdot), \boldsymbol{\phi}(\cdot)^T\begin{pmatrix}\mathsf{I}_N\\ \Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\end{pmatrix}\right\rangle_{\mathcal{H}_K}\Psi\boldsymbol{y} \\
&= \boldsymbol{b}^T\begin{pmatrix}\mathsf{I}_N & \Lambda_1^{-1}\Phi_1^{-1}\Phi_2\Lambda_2\end{pmatrix}\left\langle\boldsymbol{\phi}(\cdot), \boldsymbol{\phi}(\cdot)^T\right\rangle_{\mathcal{H}_K}\begin{pmatrix}\mathsf{I}_N\\ \Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\end{pmatrix}\boldsymbol{b} \\
&= \boldsymbol{b}^T\begin{pmatrix}\mathsf{I}_N & \Lambda_1^{-1}\Phi_1^{-1}\Phi_2\Lambda_2\end{pmatrix}\begin{pmatrix}\Lambda_1^{-1} & \\ & \Lambda_2^{-1}\end{pmatrix}\begin{pmatrix}\mathsf{I}_N\\ \Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\end{pmatrix}\boldsymbol{b} \\
&= \boldsymbol{b}^T\left(\Lambda_1^{-1} + \Lambda_1^{-1}\Phi_1^{-1}\Phi_2\Lambda_2\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\right)\boldsymbol{b}.
\end{aligned}$$

The resolution of the inner product is a result of the Hilbert space orthogonality of the eigenfunctions described in (1.8), i.e., $\left\langle\boldsymbol{\phi}(\cdot), \boldsymbol{\phi}(\cdot)^T\right\rangle_{\mathcal{H}_K} = \Lambda^{-1}$.

Remark 1 *Much of what is presented in this paper applied to general kernels, but the lower bound on $\|s^2\|_{\mathcal{H}_K}$ derived below applies only to Gaussian kernels. There are likely similar bounds on other kernels as well. A thorough discussion of the Gaussian eigenvalues is available in [3].*

The bound (2.9) becomes tighter as $\varepsilon \to 0$, because the remainder term in (2.8) is orders of magnitude smaller than the main term. [GEF] There are pictures to illustrate this in my Sicily talk. To prove this, we need to study $\|\Lambda_2^{1/2}\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\boldsymbol{b}\|_2^2$; we assume that $\Phi_2^T\Phi_1^{-T}$ is a well-behaved matrix as $\varepsilon \to 0$. This is a reasonable assumption if the eigenfunctions are properly scaled, a topic which is discussed in [3]. Furthermore, we can note that $\boldsymbol{b}$ is well-behaved as $\varepsilon \to 0$ because otherwise $\boldsymbol{\psi}$ would not be a stable basis.

We need to consider the magnitude of terms in $\Lambda_2^{1/2}\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}$ as $\varepsilon \to 0$, under the assumption that $\Phi_2^T\Phi_1^{-T}$ is at its limit. Rows of $\Phi_2^T\Phi_1^{-T}$ are scaled by $\Lambda_2^{1/2}$ and columns are scaled by Λ_1^{-1}. Note that as $\varepsilon \to 0$,

$$\left(\Lambda_1^{-1}\right)_{ii} \approx \varepsilon^{-2i}, \qquad \left(\Lambda_2^{1/2}\right)_{ii} \approx \varepsilon^{i+N}. \qquad (2.10)$$

Recalling a property of matrix norms that I found on Wikipedia (I'm sure it's also in [5]), we know that $\|\mathsf{A}\boldsymbol{x}\|_a \leq \|\mathsf{A}\|_b\|\boldsymbol{x}\|_a$ for a, b such that $\|\cdot\|_b$ is an induced matrix norm and $\|\cdot\|_a$ is a valid vector norm. Using that, (2.10), and

the submultiplicativity of induced matrix norms, we know

$$\begin{aligned}\lim_{\varepsilon\to 0}\left\|\Lambda_2^{1/2}\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\boldsymbol{b}\right\|_2 &\le \lim_{\varepsilon\to 0}\left\|\Lambda_2^{1/2}\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\right\|_\infty \|\boldsymbol{b}\|_2\\ &\le \|\boldsymbol{b}\|_2\left\|\Phi_2^T\Phi_1^{-T}\right\|_\infty \lim_{\varepsilon\to 0}\left\|\Lambda_2^{1/2}\right\|_\infty \left\|\Lambda_1^{-1}\right\|_\infty\\ &\le \|\boldsymbol{b}\|_2\left\|\Phi_2^T\Phi_1^{-T}\right\|_\infty \varepsilon^{1+N}\varepsilon^{-2N}\end{aligned}$$

This lets us say that

$$\lim_{\varepsilon\to 0}\left\|\Lambda_2^{1/2}\Phi_2^T\Phi_1^{-T}\Lambda_1^{-1}\boldsymbol{b}\right\|_2^2 \le \varepsilon^{2-2N}\left\|\Phi_2^T\Phi_1^{-T}\right\|_\infty^2\|\boldsymbol{b}\|_2^2.$$

When we compare that to

$$\boldsymbol{b}^T\Lambda_1^{-1}\boldsymbol{b} = \left\|\Lambda_1^{-1/2}\boldsymbol{b}\right\|_2^2 \le \left\|\Lambda_1^{-1/2}\right\|_\infty^2\|\boldsymbol{b}\|_2^2 = \left(\varepsilon^{-N}\right)^2\|\boldsymbol{b}\|_2^2 = \varepsilon^{-2N}\|\boldsymbol{b}\|_2^2,$$

we see that, as $\varepsilon\to 0$, the remainder term of (2.8) grows two orders of magnitude more slowly than the bound in (2.9).

[GEF] So if you combine the two ideas, does $\mathrm{C_{GW}}(\varepsilon;k)$ give you a good criterion? I've used the $\mathrm{C_{GW}}(\varepsilon;\infty)$ criterion—without the HS-SVD—before. See Fig. 20 in the past, present, future paper. For me, it seemed to be right there with MLE. [MJM] I think we're still subject to the same cancelation for the power function. The fix I just typed up may correct that. I'll need to test it though.

2.2 Cross-Validation

This writeup is stolen liberally from the Hickernell writeup that I mentioned earlier. The idea of cross-validation is essentially:

- Given the scattered data $\mathcal{X}$ and $\boldsymbol{y}$, partition the design into disjoint, nonempty groups $\mathcal{X}_I$ and $\mathcal{X}_O$ so that $\mathcal{X}_I\cup\mathcal{X}_O=\mathcal{X}$. The set $\mathcal{X}_I$ will be used to create an auxiliary approximation s_I and the set $\mathcal{X}_O$ will be used to judge the accuracy of s_I.
  - The sets $\mathcal{X}_I$ and $\mathcal{X}_O$ block up the interpolation matrix K, its inverse A, and the vectors $\boldsymbol{y}$ and $\boldsymbol{c}$ as

$$\mathsf{K}=\begin{pmatrix}\mathsf{K}_{II} & \mathsf{K}_{IO}\\ \mathsf{K}_{OI} & \mathsf{K}_{OO}\end{pmatrix},\qquad \mathsf{A}=\begin{pmatrix}\mathsf{A}_{II} & \mathsf{A}_{IO}\\ \mathsf{A}_{OI} & \mathsf{A}_{OO}\end{pmatrix},\qquad \boldsymbol{y}=\begin{pmatrix}\boldsymbol{y}_I\\ \boldsymbol{y}_O\end{pmatrix},\qquad \boldsymbol{c}=\begin{pmatrix}\boldsymbol{c}_I\\ \boldsymbol{c}_O\end{pmatrix}.$$

    Recall that $\mathsf{K}\boldsymbol{c}=\boldsymbol{y}$, and thus $\boldsymbol{c}=\mathsf{A}\boldsymbol{y}$; because K and A are symmetric, we know that $\mathsf{K}_{IO}=\mathsf{K}_{OI}^T$ and $\mathsf{A}_{IO}=\mathsf{A}_{OI}^T$. Also recall that the parameter ε appears in K.

- The available data at $\mathcal{X}_0$ is $\boldsymbol{y}_0$, and the prediction at the points $\mathcal{X}_0$ using the data $(\mathcal{X}_I,\boldsymbol{y}_I)$ is $\mathsf{K}_{OI}\mathsf{K}_{II}^{-1}\boldsymbol{y}_I$ which can be determined by applying the structure of (1.4) at x locations in $\mathcal{X}_O$. Therefore, studying

$$\left|\,y_O-\mathsf{K}_{OI}\mathsf{K}_{II}^{-1}\boldsymbol{y}_I\right|$$

  tells us something about how good our approximation is. The matrix $\mathsf{K}_{OI}\mathsf{K}_{II}^{-1}$ can be thought of as an operator which takes in values on $\mathcal{X}_I$ and interpolates them to $\mathcal{X}_O$ [6].

- By considering a set of p partitions $\mathcal{O}=\{\mathcal{X}_O^{(1)},\dots,\mathcal{X}_O^{(p)}\}$ such that

$$\mathcal{X}_O^{(i)}\cap\mathcal{X}_O^{(j)}=\emptyset,\ i\ne j,\quad\text{and}\quad\bigcup_{i=1}^{p}\mathcal{X}_O^{(i)}=\mathcal{X}$$

  with $\mathcal{X}_I^{(i)}=\mathcal{X}\setminus\mathcal{X}_O^{(i)}$, we could consider instances with certain pieces of data omitted for certain i values. Then we could consider the residual left-over by our interpolants evaluated at those points:

$$\mathrm{C_{CV}}(\varepsilon;\mathcal{O},k)=\sum_{\mathcal{X}_O\in\mathcal{O}}\left\|\boldsymbol{y}_O-\mathsf{K}_{OI}\mathsf{K}_{II}^{-1}\boldsymbol{y}_I\right\|_k, \tag{2.11}$$

  where k is probably 1, 2 or ∞.

The criterion to be minimized for a good ε value is (2.11). Of course, the presence of K_{II}^{-1} suggests ill-conditioning may be a problem. If we choose to implement the Hilbert-Schmidt SVD here, we should block up the interpolation matrix

$$\mathsf{K} = \begin{pmatrix} \mathsf{K}_{II} & \mathsf{K}_{IO} \\ \mathsf{K}_{OI} & \mathsf{K}_{OO} \end{pmatrix} = \begin{pmatrix} \Psi_{II}\Lambda_I\Phi_I^T & \Psi_{IO}\Lambda_O\Phi_O^T \\ \Psi_{OI}\Lambda_I\Phi_I^T & \Psi_{OO}\Lambda_O\Phi_O^T \end{pmatrix} = \begin{pmatrix} \Psi_{II} & \Psi_{IO} \\ \Psi_{OI} & \Psi_{OO} \end{pmatrix} \begin{pmatrix} \Lambda_I\Phi_I^T & \\ & \Lambda_O\Phi_O^T \end{pmatrix}. \tag{2.12}$$

Using this block structure, we see that $\mathsf{K}_{II} = \Psi_{II}\Lambda_I\Phi_I^T$ and $\mathsf{K}_{OI} = \Psi_{OI}\Lambda_I\Phi_I^T$, so therefore we can write the criterion as

$$\mathrm{C_{CV}}(\varepsilon; \mathcal{O}, k) = \sum_{\mathcal{X}_O \in \mathcal{O}} \left\| \boldsymbol{y}_O - \Psi_{OI}\Psi_{II}^{-1}\boldsymbol{y}_I \right\|_k, \tag{2.13}$$

which is presumably safer to compute.

The expression in (2.11) can be rewritten in a slightly different way by exploiting the block structure of the relevant vectors and matrices. Note that $\boldsymbol{c} = \mathsf{A}\boldsymbol{y}$ implies

$$\boldsymbol{c}_O = \mathsf{A}_{OI}\boldsymbol{y}_I + \mathsf{A}_{OO}\boldsymbol{y}_O.$$

We can also invoke the fact that $\mathsf{AK} = \mathsf{I}_N$ (where I_N is the $N \times N$ identity) to note that

$$\mathsf{A}_{OI}\mathsf{K}_{II} + \mathsf{A}_{OO}\mathsf{K}_{IO}^T = 0 \quad \Rightarrow \quad \mathsf{A}_{OI} = -\mathsf{A}_{OO}\mathsf{K}_{IO}^T\mathsf{K}_{II}^{-1}.$$

Plugging this in above gives

$$\begin{aligned} \boldsymbol{c}_O &= -\mathsf{A}_{OO}\mathsf{K}_{OI}\mathsf{K}_{II}^{-1}\boldsymbol{y}_I + \mathsf{A}_{OO}\boldsymbol{y}_O \\ \mathsf{A}_{OO}^{-1}\boldsymbol{c}_O &= -\mathsf{K}_{OI}\mathsf{K}_{II}^{-1}\boldsymbol{y}_I + \boldsymbol{y}_O \end{aligned}$$

which allows us to write our optimization criterion as

$$\mathrm{C_{CV}}(\varepsilon; \mathcal{O}, k) = \sum_{\mathcal{X}_O \in \mathcal{O}} \left\| \mathsf{A}_{OO}^{-1}\boldsymbol{c}_O \right\|_k. \tag{2.14}$$

To involve the Hilbert-Schmidt SVD, we need to study (2.12) and define

$$\begin{pmatrix} \Psi_{II} & \Psi_{IO} \\ \Psi_{OI} & \Psi_{OO} \end{pmatrix} \begin{pmatrix} \mathsf{B}_{II} & \mathsf{B}_{IO} \\ \mathsf{B}_{OI} & \mathsf{B}_{OO} \end{pmatrix} = \begin{pmatrix} \mathsf{I}_{N_I} & \\ & \mathsf{I}_{N_O} \end{pmatrix},$$

where $\mathcal{X}_I$ and $\mathcal{X}_O$ have N_I and N_O points in them respectively. Manipulations similar to those above show that

$$\mathsf{B}_{OO}^{-1} = \Psi_{OO} - \Psi_{OI}\Psi_{II}^{-1}\Psi_{IO},$$

which we will use shortly. We also need to define

$$\begin{pmatrix} \Psi_{II} & \Psi_{IO} \\ \Psi_{OI} & \Psi_{OO} \end{pmatrix} \begin{pmatrix} \boldsymbol{b}_I \\ \boldsymbol{b}_O \end{pmatrix} = \begin{pmatrix} \boldsymbol{y}_I \\ \boldsymbol{y}_O \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} \boldsymbol{b}_I \\ \boldsymbol{b}_O \end{pmatrix} = \begin{pmatrix} \mathsf{B}_{II} & \mathsf{B}_{IO} \\ \mathsf{B}_{OI} & \mathsf{B}_{OO} \end{pmatrix} \begin{pmatrix} \boldsymbol{y}_I \\ \boldsymbol{y}_O \end{pmatrix}, \qquad \begin{pmatrix} \boldsymbol{b}_I \\ \boldsymbol{b}_O \end{pmatrix} = \begin{pmatrix} \Lambda_I\Phi_I^T & \\ & \Lambda_O\Phi_O^T \end{pmatrix} \begin{pmatrix} \boldsymbol{c}_I \\ \boldsymbol{c}_O \end{pmatrix}$$

This implies that, in terms of the HS-SVD system, $\boldsymbol{c}_O = \Phi_O^{-T}\Lambda_O^{-1}\boldsymbol{b}_O$, and sufficient study of (2.12) knowing that $\mathsf{AK} = \mathsf{I}_N$ gives us $\mathsf{A}_{OO} = \Phi_O^{-T}\Lambda_O^{-1}B_{OO}$. Combining these in (2.14) gives us

$$\mathrm{C_{CV}}(\varepsilon; \mathcal{O}, k) = \sum_{\mathcal{X}_O \in \mathcal{O}} \left\| \mathsf{B}_{OO}^{-1}\boldsymbol{b}_O \right\|_k = \sum_{\mathcal{X}_O \in \mathcal{O}} \left\| (\Psi_{OO} - \Psi_{OI}\Psi_{II}^{-1}\Psi_{IO})^{-1}\boldsymbol{b}_O \right\|_k, \tag{2.15}$$

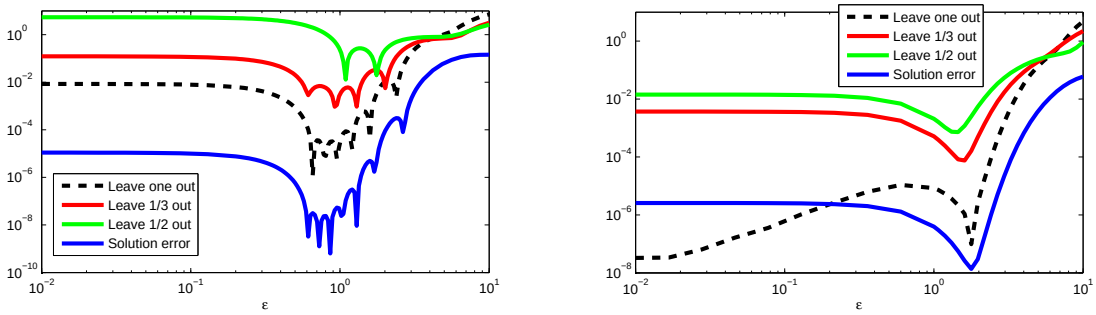
which is almost certainly more stable to compute.

Often times, cross-validation is conducted in one of two ways:

- **Leave-one-out cross-validation** - All the data except a single point is used to compute the interpolant, and the residual is judged at that point. In this setting, $\mathcal{O} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ and the errors at each of those points are added up to find $CV(\varepsilon; \mathcal{O}, k)$. (2.14) may be preferable to compute in this case because A_{OO} is just a number.

- **Leave-half-out cross-validation** - Half of the data is omitted to create an interpolant and the residual is judged on the other half; then the process is flipped and both results are combined to compute $CV(\varepsilon; \mathcal{O}, k)$. In this setting, $\mathcal{O} = \{\mathcal{X}_O^{(1)}, \mathcal{X}_O^{(2)}\}$ and $\left|\mathcal{X}_O^{(1)}\right| = \left|\mathcal{X}_O^{(2)}\right|$, or as close as possible. (2.11) is almost surely the preferred computation in this case because A_{OO} is roughly size $N/2 \times N/2$ and computing its inverse is costly.

An example of cross-validation results is presented in Figure 5, although the results are somewhat inconclusive: the minimum in Figure 5a seems near the "true" ε, but in Figure 5b the $\mathrm{C_{CV}}$ function for LOOCV seems to drop for $\varepsilon \to 0$ which does not reflect the error curve.



(a) $f(x) = \cos(2\pi x)$, $N = 18$ uniform points

(b) $f(x) = (1 + x^2)^{-1}$, $N = 24$ uniform points

Figure 5: In this example, N points in $(-1, 1)$ are sampled and Gaussian interpolation is conducted with a range of ε values. These results are compared to $\mathrm{C_{CV}}(\varepsilon; \mathcal{O}, 2)$, for three $\mathcal{O}$ choices: one with 2 equally sized sets, 3 equally sized set, and N equally sized sets. The error is averaged over 100 evenly spaced points.

One concern, which is discussed in Section 2.3.1, is the process variance. This is the idea that the kernel K in the Gaussian field could actually be written as $\hat{K} = \sigma K$ for some positive valued σ called the *process variance*. This plays no role here because such a term gets canceled out in the computation of $\mathsf{K}_{IO}^T \mathsf{K}_{II}^{-1} \boldsymbol{y}_I$, in the same way that it is canceled out in $s(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^T \mathsf{K}^{-1} \boldsymbol{y}$. It does potentially play a role in other settings though, most notably the Kriging variance, and is discussed later.

Although often described in a statistical setting, this cross-validation concept is equally well supported from the numerical analysis side. No rigorous structure that I know of exists to explain why this is a useful technique analytically, as opposed to just an idea that makes sense.

2.3 Maximum Likelihood Estimation

This approach leans primarily on the Gaussian processes framework, though we will show later how it can be derived using the Hilbert space framework. Begin by recalling (1.13), that $\boldsymbol{Y}_{\mathcal{X}} \sim \mathcal{N}(\boldsymbol{\mu}, \mathsf{K})$, and, as before, restricting the Gaussian Process Y to have zero mean, i.e., $\mu_Y \equiv 0$. We are going to treat ε as *drawn from a random variable* $\mathcal{E}$ now, with some unknown distribution; in turn we also will need to define the joint random variable $Z = (\mathcal{E}, \boldsymbol{Y}_{\mathcal{X}})$ with density p_Z. We want to study the conditional density function $p_{\mathcal{E}|\boldsymbol{Y}_{\mathcal{X}}}(\varepsilon|\boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y})$, that is, how likely it is that the Gaussian process had covariance K, parameterized by ε, when realizing the data $\boldsymbol{y}$.

This function is often called the *likelihood* function, and it allows us to compare the relative likelihood of values of ε given the existing data. Maximizing this function yields (in some sense) the ε which most likely parameterized the covariance kernel which generated the data $\boldsymbol{y}$. The function $p_{\mathcal{E}|\boldsymbol{Y}_{\mathcal{X}}}(\varepsilon|\boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y})$ is a function of ε for any fixed value of $\boldsymbol{y}$.

A technical note: the notation from Section 1.2 and this section will not perfectly align. In that section we considered ε fixed and the joint distribution $(\boldsymbol{Y}_{\mathcal{X}}, Y_{\boldsymbol{x}_0})$, but here we are studying the joint distribution $(\mathcal{E}, \boldsymbol{Y}_{\mathcal{X}})$. Where we previously had written $p_{\boldsymbol{Y}_{\mathcal{X}}}(\boldsymbol{y})$ in (1.16), the appropriate notation in this section is

$$p_{\boldsymbol{Y}_{\mathcal{X}}|\mathcal{E}}(\boldsymbol{y}|\mathcal{E} = \varepsilon) = \frac{1}{\sqrt{2\pi \det(\mathsf{K})}} \exp\left(-\frac{1}{2} \boldsymbol{y}^T \mathsf{K}^{-1} \boldsymbol{y}\right), \tag{2.16}$$

where although ε does not explicitly appear on the right hand side, it appears within K.

We want to study $p_{\mathcal{E}|\boldsymbol{Y}_{\mathcal{X}}}(\varepsilon|\boldsymbol{Y}_{\mathcal{X}}=\boldsymbol{y})$, but we do not know that density. What we do know is the definition of conditional probability:

$$P(A|B)=\frac{P(A\cap B)}{P(B)} \quad \Leftrightarrow \quad P(A\cap B)=P(A|B)P(B),$$

assuming $P(B)>0$. Using an analogous version of this for density functions instead of the probability function allows us to write

$$\begin{aligned} p_{\mathcal{E}|\boldsymbol{Y}_{\mathcal{X}}}(\varepsilon|\boldsymbol{Y}_{\mathcal{X}}=\boldsymbol{y}) &= \frac{p_Z(\varepsilon,\boldsymbol{y})}{p_{\boldsymbol{Y}_{\mathcal{X}}}(\boldsymbol{Y}_{\mathcal{X}}=\boldsymbol{y})} \\ &= \frac{p_{\boldsymbol{Y}_{\mathcal{X}}|\mathcal{E}}(\boldsymbol{y}|\mathcal{E}=\varepsilon)p_{\mathcal{E}}(\varepsilon)}{p_{\boldsymbol{Y}_{\mathcal{X}}}(\boldsymbol{y})}. \end{aligned} \tag{2.17}$$

We can clean up our likelihood function expression somewhat by ignoring things we have no knowledge of. First off, we do not know what the marginal distribution of ε is - if we did we would just study that to determine an optimal ε parameterization. We also do not know what the marginal distribution of $\boldsymbol{Y}_{\mathcal{X}}$ is: we do not know how $\boldsymbol{Y}_{\mathcal{X}}$ varies independently of ε. To determine this, we would need to compute

$$p_{\boldsymbol{Y}_{\mathcal{X}}}(\boldsymbol{y})=\int_0^\infty p_Z(\boldsymbol{y},\varepsilon)\mathrm{d}\varepsilon$$

which is not going to happen because the joint density p_Z is unknown. More importantly, $p_{\boldsymbol{Y}_{\mathcal{X}}}(\boldsymbol{y})$ is independent of ε, and therefore changing ε will have no effect on that value.

By abandoning $p_{\mathcal{E}}(\varepsilon)$ and $p_{\boldsymbol{Y}_{\mathcal{X}}}(\boldsymbol{y})$, we can suggest that

$$p_{\mathcal{E}|\boldsymbol{Y}_{\mathcal{X}}}(\varepsilon|\boldsymbol{Y}_{\mathcal{X}}=\boldsymbol{y}) \propto p_{\boldsymbol{Y}_{\mathcal{X}}|\mathcal{E}}(\boldsymbol{y}|\mathcal{E}=\varepsilon), \tag{2.18}$$

and we know this function (from (2.16)). Thus the concept of maximizing the likelihood requires maximizing $p_{\boldsymbol{Y}_{\mathcal{X}}|\mathcal{E}}(\boldsymbol{y}|\mathcal{E}=\varepsilon)$. By itself, this function is subject to overflow and underflow, so it is common to instead work with its logarithm:

$$\log\left(p_{\boldsymbol{Y}_{\mathcal{X}}|\mathcal{E}}(\boldsymbol{y}|\mathcal{E}=\varepsilon)\right)=-\frac{1}{2}\log\det\mathsf{K}-\frac{1}{2}\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y}-\frac{1}{2}\log 2\pi.$$

Because we have been in the practice of minimizing functions to find optimal ε parameterizations, we will multiply by -2 and ignore the constant $\log 2\pi$ to create our maximum likelihood criterion

$$\begin{aligned} \mathrm{C}_{\mathrm{MLE}}(\varepsilon) &= -2\log\left(p_{\boldsymbol{Y}_{\mathcal{X}}|\mathcal{E}}(\boldsymbol{y}|\mathcal{E}=\varepsilon)\right)-\log 2\pi \\ &= \log\det\mathsf{K}+\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y} \end{aligned} \tag{2.19}$$

2.3.1 Incorporating the process variance

Mentioned briefly in Section 2.2 was the concept of a process variance $\sigma>0$ which defines the maximum covariance of K; in this setting, our standard kernel K would actually be multiplied by σ to create the kernel $\tilde{K}=\sigma K$ which defines the Gaussian process $Y\sim GF(\mu_Y,\tilde{K})$.

Thus far we have considered only $\sigma=1$, and in fact this is acceptable for prediction purposes: recall that $s(\boldsymbol{x})=\boldsymbol{k}(\boldsymbol{x})^T\mathsf{K}^{-1}\boldsymbol{y}$. Replacing K with $\tilde{K}$ would give

$$s(\boldsymbol{x})=\tilde{\boldsymbol{k}}(\boldsymbol{x})^T\tilde{\mathsf{K}}^{-1}\boldsymbol{y}=\sigma\boldsymbol{k}(\boldsymbol{x})^T(\sigma\mathsf{K})^{-1}\boldsymbol{y}=\boldsymbol{k}(\boldsymbol{x})^T\mathsf{K}^{-1}\boldsymbol{y}$$

thus the interpolant would be the same for any σ. The same computation can be applied to C_{CV} to show that it is invariant for different σ values. The power function $P_{\tilde{K},\mathcal{X}}\to 0$ as $\sigma\to 0$,

$$P_{\tilde{K},\mathcal{X}}(\boldsymbol{x})=\sqrt{\tilde{K}(\boldsymbol{x},\boldsymbol{x})-\tilde{\boldsymbol{k}}(\boldsymbol{x})^T\tilde{\mathsf{K}}^{-1}\tilde{\boldsymbol{k}}(\boldsymbol{x})}=\sqrt{\sigma K(\boldsymbol{x},\boldsymbol{x})-\sigma\boldsymbol{k}(\boldsymbol{x})^T(\sigma\mathsf{K})^{-1}\sigma\boldsymbol{k}(\boldsymbol{x})}=\sqrt{\sigma}P_{K,\mathcal{X}}(\boldsymbol{x}),$$

so when studying the power function it is necessary to fix σ, probably to 1, and vary ε to minimize $\mathrm{C}_{\mathrm{POWER}}$. If the term $\|f\|_{\mathcal{H}_K}$ or $\|s\|_{\mathcal{H}_K}$ is included to compute the Native space norm error or Golomb-Weinberger bound, respectively, (see the Appendix) then the situation becomes more complicated because σ is involved in the Native space norm.

The situation is also tricky for maximum likelihood estimation, which is why it is under consideration right now. When we introduce a process variance, we are suggesting that σ is a draw from a random variable Σ with unknown distribution. We would need to study the joint distribution $(\Sigma, \mathcal{E}, \boldsymbol{Y}_{\mathcal{X}})$, and our kernel parameterization would require optimizing for both σ and ε by maximizing $p_{\Sigma,\mathcal{E}|\boldsymbol{Y}_{\mathcal{X}}}(\sigma, \varepsilon|\boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y})$, which we will maximize by maximizing $p_{\boldsymbol{Y}_{\mathcal{X}}|\Sigma,\mathcal{E}}(\boldsymbol{y}|\Sigma = \sigma, \mathcal{E} = \varepsilon)$ similarly to (2.18).

We could treat this as a two dimensional optimization problem, but instead we will invoke the technique of *profile likelihood*, where σ will be defined as a function of ε, i.e., $\sigma \equiv \sigma(\varepsilon)$. Our goal now is to choose an optimal process variance σ_{opt} which we will do by maximizing $p_{\Sigma|\mathcal{E},\boldsymbol{Y}_{\mathcal{X}}}(\sigma|\mathcal{E} = \varepsilon, \boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y})$. The term profile likelihood is a bit of a misnomer, because $p_{\Sigma,\mathcal{E}|\boldsymbol{Y}_{\mathcal{X}}}(\sigma(\varepsilon), \varepsilon|\boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y})$ is not derived from a cumulative distribution function and thus it is not a true density and loses some desirable properties. Even so, this is a common technique.

Using the same proportionality logic as in (2.18) we can write that

$$\begin{aligned} p_{\Sigma|\mathcal{E},\boldsymbol{Y}_{\mathcal{X}}}(\sigma|\mathcal{E} = \varepsilon, \boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y}) &\propto p_{\mathcal{E},\boldsymbol{Y}_{\mathcal{X}}|\Sigma}(\varepsilon, \boldsymbol{y}|\Sigma = \sigma) \\ &= p_{\boldsymbol{Y}_{\mathcal{X}}|\Sigma,\mathcal{E}}(\boldsymbol{y}|\Sigma = \sigma, \mathcal{E} = \varepsilon) p_{\mathcal{E}|\Sigma}(\varepsilon|\Sigma = \sigma) \\ &\propto p_{\boldsymbol{Y}_{\mathcal{X}}|\Sigma,\mathcal{E}}(\boldsymbol{y}|\Sigma = \sigma, \mathcal{E} = \varepsilon). \end{aligned}$$

Therefore, our optimal σ can be found by maximizing $p_{\boldsymbol{Y}_{\mathcal{X}}|\Sigma,\mathcal{E}}(\boldsymbol{y}|\Sigma = \sigma, \mathcal{E} = \varepsilon)$; this function is the same as (2.16), except with K replaced by $\tilde{\mathsf{K}}$. As before, instead of maximizing, we will try to minimize the negative log of this function:

$$\begin{aligned} -2\log\left(p_{\boldsymbol{Y}_{\mathcal{X}}|\Sigma,\mathcal{E}}(\boldsymbol{y}|\Sigma = \sigma, \mathcal{E} = \varepsilon)\right) + \log 2\pi &= \log\det\tilde{\mathsf{K}} + \boldsymbol{y}^T\tilde{\mathsf{K}}^{-1}\boldsymbol{y} \\ &= \log\det\sigma\mathsf{K} + \boldsymbol{y}^T(\sigma\mathsf{K})^{-1}\boldsymbol{y} \\ &= N\log\sigma + \log\det\mathsf{K} + \frac{1}{\sigma}\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y} \end{aligned}$$

Differentiating this with respect to σ, setting it equal to 0, and solving for σ gives the optimal profile variance

$$\sigma_{\text{opt}} = \frac{1}{N}\boldsymbol{y}^T\mathsf{K}\boldsymbol{y}. \tag{2.20}$$

Using the profile likelihood strategy, we maximize $p_{\Sigma,\mathcal{E}|\boldsymbol{Y}_{\mathcal{X}}}(\sigma, \varepsilon|\boldsymbol{Y}_{\mathcal{X}} = \boldsymbol{y})$ by minimizing

$$\begin{aligned} -2\log\left(p_{\boldsymbol{Y}_{\mathcal{X}}|\Sigma,\mathcal{E}}(\boldsymbol{y}|\Sigma = \sigma_{\text{opt}}, \mathcal{E} = \varepsilon)\right) + \log 2\pi &= N\log\left(\frac{1}{N}\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y}\right) + \log\det\mathsf{K} + \left(\frac{1}{N}\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y}\right)^{-1}\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y} \\ &= N\log\left(\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y}\right) + \log\det\mathsf{K} - N\log N + N \end{aligned}$$

thus defining our profile likelihood ε parameterization criterion as

$$\mathrm{C}_{\text{MPLE}}(\varepsilon) = N\log\left(\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y}\right) + \log\det\mathsf{K} \tag{2.21}$$

after omitting the constant $-N\log N + N$.

2.3.2 A deterministic derivation of MLE

The title of this section is a bit misleading, because of course likelihoods cannot be discussed outside of a probabilistic setting. The criterion C_{MPLE} is equivalent to a criterion which can be derived deterministically, which we will show now. This too was swiped from the Hickernell writeup.

As before, assume that the function which produced the data $\boldsymbol{y}$ is $f \in \mathcal{H}_K$. Let us expand the notation for our interpolant s to include the data $\boldsymbol{z}$ which generated the interpolant: $s(\cdot; \boldsymbol{z}) = \boldsymbol{k}(\cdot)^T\mathsf{K}^{-1}\boldsymbol{z}$. Also, recall the Hilbert-space norm of an interpolant (derived in (3.4) in the Appendix) is

$$\|s(\cdot; \boldsymbol{z})\|_{\mathcal{H}_K} = \sqrt{\boldsymbol{z}^T\mathsf{K}^{-1}\boldsymbol{z}}. \tag{2.22}$$

Let $V(\varepsilon)$ denote the volume of the ellipsoid in $\mathbb{R}^N$ which contains all $\boldsymbol{z}$ such that $\|s(\cdot;\boldsymbol{z})\|_{\mathcal{H}_K} \leq \|s(\cdot;\boldsymbol{y})\|_{\mathcal{H}_K}$:

$$\begin{aligned} V(\varepsilon) = &\text{ volume of ellipsoid } \left\{z \in \mathbb{R}^N : \|s(\cdot;\boldsymbol{z})\|_{\mathcal{H}_K}^2 \leq \|s(\cdot;\boldsymbol{y})\|_{\mathcal{H}_K}^2\right\} \\ = &\text{ volume of ellipsoid } \left\{z \in \mathbb{R}^N : \boldsymbol{z}^T\mathsf{K}^{-1}\boldsymbol{z} \leq \boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y}\right\} \\ = &\,\vartheta_N \frac{(\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y})^N}{\det \mathsf{K}^{-1}} \\ = &\,\vartheta_N (\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y})^N \det \mathsf{K} = \vartheta_N \exp(\mathrm{C}_{\mathrm{MPLE}}(\varepsilon)), \end{aligned}$$

where ϑ_N is the volume of the $\mathbb{R}^N$ unit sphere. Thus, choosing ε to minimize the volume of the ellipsoid containing function data which would produce "smaller interpolants" (in the $\mathcal{H}_K$ norm) than the observed data $\boldsymbol{y}$ produces is equivalent to maximizing the profile likelihood that ε parameterized the Gaussian field from which $\boldsymbol{y}$ was realized.

This concept of ellipsoid volume is basically employing Occam's Razor: the interpolant that best fits the data should be the simplest, which in this case is measured using the $\mathcal{H}_K$ norm. The ellipsoid described in $V(\varepsilon)$ contains data that would produce a simpler interpolant. By choosing ε to minimize $V(\varepsilon)$ we are minimizing the region from which simpler interpolants could be produced, thus making it less likely that our interpolant is not the simplest.

2.3.3 Impact of the Hilbert-Schmidt SVD

Two pieces appear in (2.21) which may be subject to ill-conditioning, $\log\left(\boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{y}\right)$ and $\log\det\mathsf{K}$. We have already presented (2.8) which will allow the former term to be computed stably, and (2.9) which will bound that term as $\varepsilon \to 0$ and help us avoid potential overflow/underflow prior to applying the logarithm. The term $\log\det\mathsf{K}$ is actually more safe to compute when the analytic form of the eigenvalues is known.

Using the Hilbert-Schmidt SVD,

$$\log\det\mathsf{K} = \log\det\Psi\Lambda_1\Phi_1^T = \log\left(\det\Psi\det\Lambda_1\det\Phi_1^T\right) = \log\det\Psi + \log\det\Lambda_1 + \log\det\Phi_1^T.$$

Presumably, we have already computed a factorization of Φ_1^T when Ψ was computed (recall (1.9)) so computing $\log\det\Phi_1^T$ should come at no cost and with no stability issues. Furthermore, assuming we at some point computed $\Psi^{-1}\boldsymbol{y}$ to evaluate our interpolant, we should have a factorization of Ψ meaning that $\log\det\Psi$ can be computed at no cost and with no stability issues. The final term, $\log\det\Lambda_1$ is straightforward because it is diagonal:

$$\log\det\Lambda_1 = \log\prod_{n=1}^{N}\lambda_n = \sum_{n=1}^{N}\log\lambda_n,$$

and by distributing the logarithm we are able to avoid the underflow issues that would otherwise arise.

[MJM] Add pictures

3 Goals for a Parameterization Judgment Tool

For applications to buy into kernel methods, the presence of a free parameter such as ε (or perhaps a β smoothness parameter) must be seen as a benefit and not a liability. To do this, we need to have successful parameterization methods so that ε can be chosen to produce the most accurate results. The techniques described above show that existing parameterization methods may work or may not work depending on the situation. New schemes can be developed, but we believe that a tool is needed to judge the viability of existing and future schemes within a rigorous context.

Our goal right now is to develop a metric by which the quality of a parameterization scheme can be measured with respect to other schemes, or a single scheme's success can be compared across scattered data problems. This could be thought of analogously to how we measure the convergence rate of numerical algorithms such as Newton's method: when solving $f(\alpha) = 0$, we know

$$|\epsilon_{n+1}| = \frac{|f''(\xi_n)|}{2\,|f'(x_n)|}\,{\epsilon_n}^2, \qquad \epsilon_n = \alpha - x_n, \qquad \xi_n \text{ between } x_n \text{ and } \alpha.$$

This suggests that Newton's method converges quadratically, which is a property that can be compared to other root-finding schemes. On the other hand, this quadratic convergence is only valid when $f'(x) \neq 0$ near $x = \alpha$ and $f''(x) < \infty$ near $x = \alpha$, which is a comparison of this method across the set of problems to which it could be applied.

What I think is the likeliest structure for a parameterization judgment tool will be: how near is the guess to the "true" value as the amount of available data increases. Now, what exactly the "true" value means is up for interpretation ...

- You could argue that the true ε for any given set of data would be the value that minimizes (some norm of) the error. I think that is an unstable definition because it may be subject to small changes in the data, but it is probably the definition most sought by applications people.
- The "true" ε could be the ε value that defines the Hilbert-space $\mathcal{H}_K$ in which f lies.
- In the same way as the previous bullet, the "true" ε could be the ε value defining the covariance kernel of the Gaussian process that generated the available data.

The last two points are similar approaches to defining the "true" ε, only the 2nd bullet considers it from the numerical analysis side, and the 3rd considers it from the statistics side.

Defining the *fill-distance* to be

$$h_{\mathcal{X}} = \sup_{\boldsymbol{x}\in\Omega} \min_{\boldsymbol{x}_j\in\mathcal{X}} \|\boldsymbol{x} - \boldsymbol{x}_j\|_2$$

gives us a mechanism for defining the density of the design of points used in the approximation problem. Basically, the fill distance is the radius of the largest ball that can be squeezed in between points in the design $\mathcal{X}$. This is a useful tool for studying convergence behavior of interpolation schemes; for instance, we know that a kernel K with 2β smooth derivatives has an interpolant s to data generated by $f \in \mathcal{H}_K$ with accuracy

$$|f(\boldsymbol{x}) - s(\boldsymbol{x})| \leq C h_{\mathcal{X}}^{\beta} \sqrt{C_K(\boldsymbol{x})} \|f\|_{\mathcal{H}_K}, \tag{3.1}$$

where C_K is independent of f and $h_{\mathcal{X}}$ is sufficiently small. An example of an optimal ε graph as N increases (and thus $h_{\mathcal{X}}$ decreases) is provided in Figure 6.

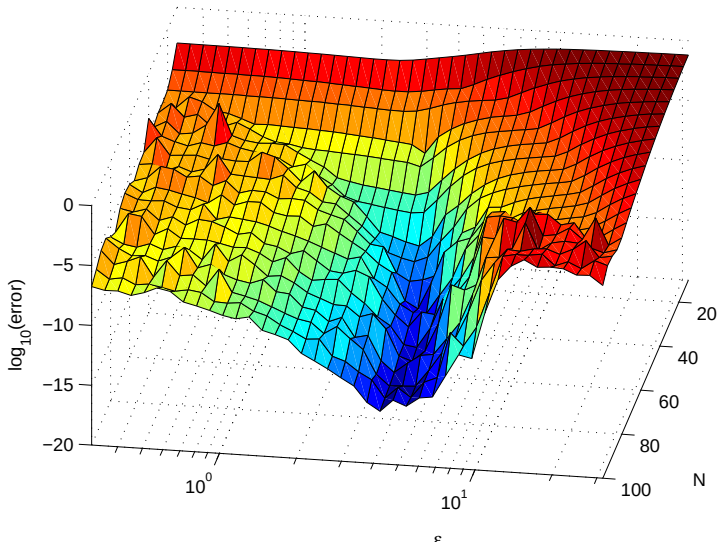


Figure 6: This example involves $f(x) = (1 + 4x^2)^{-1}$ and shows that as $N \to \infty$ there is a rough "convergence" of ε to a consistent value. This example doesn't perfectly reflect the idea of an ε value of an underlying RKHS because f does not belong to a Gaussian RKHS. Even so, the idea still applies here, until machine precision takes over.

It is my hope that a similarly structured bound can exist in a probabilistic sense for the accuracy of a parameterization scheme. Essentially, the situation would play out as

1. Data is generated by a function $f \in \mathcal{H}_K$ or a Gaussian process Y with covariance kernel K.
   (a) The kernel K that appears in both settings is the same. It has some parameter ε that defines it, but ε is unknown to us.

(b) The data is $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_N, y_N)\}$. This defines the design $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ and the vector $\boldsymbol{y}$.

i. Knowing $\mathcal{X}$ defines the fill-distance $h_{\mathcal{X}}$.

2. A parameterization scheme is chosen to guess a $\hat{\varepsilon}$ value to be used while constructing the approximation. For example, this could be computed as

$$\hat{\varepsilon} = \underset{\varepsilon}{\operatorname{argmin}}\, \mathrm{C}_{\mathrm{POWER}}(\varepsilon; 2).$$

(a) This scheme may involve $\boldsymbol{y}$ (e.g., cross-validation) or only $\mathcal{X}$ (e.g., power function).

(b) The accuracy of this scheme is hopefully something of the form

$$|\varepsilon - \hat{\varepsilon}| \leq h_{\mathcal{X}}^{\gamma} \left[C_1(\mathcal{X}) C_2(\boldsymbol{y}) \|f\|_{\mathcal{H}_K} \right],$$

for some $\gamma > 0$ and C_1, C_2. Really, I have no idea if this is what it will look like, it's just a thought for an ideal situation in the deterministic setting.

(c) What is much more likely is that the bound will be of the form

$$P(|\varepsilon - \hat{\varepsilon}| < \alpha) \geq 1 - \nu(\mathcal{X}, \boldsymbol{y}, \alpha), \qquad \lim_{h_{\mathcal{X}} \to 0} \nu(\mathcal{X}, \boldsymbol{y}, \alpha) = 0,$$

because if the data is generated by a Gaussian process, then there's always a chance that it will be a really crummy, uninformative realization.

This last point, Dr. Cobb, is where I think you will be of great help. If we can prove something of this form, then we would be able to judge (empirically at the least) the quality of parameterization schemes which would help inform applications scientists.

With a tool like this, we would be able to make comments on

- **Consistency** - Will the scheme recover the "true" ε for an infinitely dense design? This would only be true if $\lim_{h_{\mathcal{X}} \to 0} \nu(\mathcal{X}, \boldsymbol{y}) = 0$.
- **Convergence rate** - How quickly is the parameterization scheme approaching the "true" ε, and thus how few points are needed before I feel comfortable that I am doing a decent job? This would depend on the rate at which $\lim_{h_{\mathcal{X}} \to 0} \nu(\mathcal{X}, \boldsymbol{y}) = 0$, especially in comparison to other schemes.
- **Stability** - Do small changes in $\mathcal{X}$ and $\boldsymbol{y}$ affect the consistency or convergence of the scheme? I'm not as concerned about this, but it is something that should eventually be studied.
- **Bounding** ε - Most applications don't demand an optimal ε because of noise in the data. Can this convergence study be used to create a region in which the "true" ε lies?
- **Computational cost** - If two schemes have similar convergence properties, is there any reason to not use the cheaper one?
- **Log scale** - The plots used above show ε on a log-scale, and often this is how we consider different ε values. Is it useful to instead study accuracy of the form

$$|\log \varepsilon - \log \hat{\varepsilon}|, \quad \text{or} \quad \log |\varepsilon - \hat{\varepsilon}|?$$

So, do you know of anything that would help us think about this problem?

References

[1] R. Cavoretto, G. Fasshauer, and M. McCourt. Compact Matérn kernels and piecewise polynomial splines viewed from a Hilbert-Schmidt perspective, 2013. submitted.

[2] G. E. Fasshauer. *Meshfree Approximation Methods with* MATLAB. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2007.

[3] G. E. Fasshauer and M. McCourt. Stable evaluation of Gaussian RBF interpolants. *SIAM J. Sci. Comput.*, 34(2):A737—A762, 2012.

[4] M. Golomb and H. F. Weinberger. Optimal approximation and error bounds. In R. E. Langer, editor, *On Numerical Approximation*, pages 117–190. University of Wisconsin Press, 1959.

[5] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2013.

[6] M. McCourt. Using Gaussian eigenfunctions to solve boundary value problems. *Adv. Appl. Math. Mech.*, 5(4):569–594, 2013.

[7] M. L. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer Series in Statistics. Springer, 1999.

[8] H. Wendland. *Scattered Data Approximation*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2005.

Appendix: Deriving the Native Space Error Bound

For this derivation, we need to recall the reproducing property

$$\langle f, K(\cdot,\boldsymbol{x})\rangle_{\mathcal{H}_K} = f(\boldsymbol{x}), \qquad f \in \mathcal{H}_K,$$

where $\mathcal{H}_K$ is the reproducing kernel Hilbert space, or *native space*, generated by K. It is important to also know that $K(\cdot,x) \in \mathcal{H}_K$ for $x \in \Omega$, which implies that

$$\langle K(\cdot,\boldsymbol{x}), K(\cdot,\boldsymbol{z})\rangle_{\mathcal{H}_K} = K(\boldsymbol{x},\boldsymbol{z}), \qquad \boldsymbol{x},\boldsymbol{z} \in \Omega.$$

Note that because $s(\boldsymbol{x})$ is a number,

$$s(\boldsymbol{x}) = s(\boldsymbol{x})^T = \boldsymbol{y}^T \mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x}).$$

The vector $\boldsymbol{y}$ is full of function values ($y_i = f(\boldsymbol{x}_i)$) so we can write

$$\begin{aligned}\boldsymbol{y}^T = \begin{pmatrix} f(\boldsymbol{x}_1) & \cdots & f(\boldsymbol{x}_N)\end{pmatrix} &= \begin{pmatrix}\langle f, K(\cdot,\boldsymbol{x}_1)\rangle_{\mathcal{H}_K} & \cdots & \langle f, K(\cdot,\boldsymbol{x}_N)\rangle_{\mathcal{H}_K}\end{pmatrix} \\ &= \left\langle f, \begin{pmatrix} K(\cdot,\boldsymbol{x}_1) & \cdots & K(\cdot,\boldsymbol{x}_N)\end{pmatrix}\right\rangle_{\mathcal{H}_K} \\ &= \langle f, \boldsymbol{k}(\cdot)^T\rangle_{\mathcal{H}_K}.\end{aligned}$$

That in turn allows us to write

$$s(\boldsymbol{x}) = \langle f, \boldsymbol{k}(\cdot)^T\rangle_{\mathcal{H}_K}\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x}) = \left\langle f, \boldsymbol{k}(\cdot)^T\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x})\right\rangle_{\mathcal{H}_K}.$$

because $\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x})$ is not a function and is unaffected by the inner product. Using this in the error expression gives

$$\begin{aligned}|f(\boldsymbol{x}) - s(\boldsymbol{x})| &= |f(\boldsymbol{x}) - \boldsymbol{y}^T\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x})| \\ &= \left|\langle f, K(\cdot,\boldsymbol{x})\rangle_{\mathcal{H}_K} - \left\langle f, \boldsymbol{k}(\cdot)^T\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x})\right\rangle_{\mathcal{H}_K}\right| \\ &= \left|\left\langle f, K(\cdot,\boldsymbol{x}) - \boldsymbol{k}(\cdot)^T\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x})\right\rangle_{\mathcal{H}_K}\right| \\ &\leq \|f\|_{\mathcal{H}_K}\left\|K(\cdot,\boldsymbol{x}) - \boldsymbol{k}^T(\cdot)\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x})\right\|_{\mathcal{H}_K} = \|f\|_{\mathcal{H}_K}P_{K,\mathcal{X}}(\boldsymbol{x}),\end{aligned}$$

using the Cauchy-Schwarz inequality, and the power function

$$P_{K,\mathcal{X}}(\boldsymbol{x}) = \sqrt{K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}^T(\boldsymbol{x})\mathsf{K}^{-1}\boldsymbol{k}(\boldsymbol{x})}. \tag{1.6}$$

This expression for the power function is obtained by again using the reproducing property on the norm present in the inequality.

The standard error bound derived above

$$|f(\boldsymbol{x}) - s(\boldsymbol{x})| \leq \|f\|_{\mathcal{H}_K} P_{K,\mathcal{X}}(\boldsymbol{x}) \tag{1.5}$$

can be improved (see [4]) to

$$|f(\boldsymbol{x}) - s(\boldsymbol{x})| \leq \|f - s\|_{\mathcal{H}_K} P_{K,\mathcal{X}}(\boldsymbol{x}). \tag{3.2}$$

To see this, first we must prove that $f - s$ is orthogonal to s in the Hilbert-space inner product:

$$\begin{aligned}
\langle f - s, s\rangle_{\mathcal{H}_K} &= \big\langle f - s, \boldsymbol{k}(\cdot)^T \mathsf{K}^{-1}\boldsymbol{y}\big\rangle_{\mathcal{H}_K} \\
&= \big\langle f - s, \boldsymbol{k}(\cdot)^T\big\rangle_{\mathcal{H}_K} \mathsf{K}^{-1}\boldsymbol{y} \\
&= \big(\langle f - s, K(\cdot, \boldsymbol{x}_1)\rangle_{\mathcal{H}_K} \quad \cdots \quad \langle f - s, K(\cdot, \boldsymbol{x}_N)\rangle_{\mathcal{H}_K}\big)\, \mathsf{K}^{-1}\boldsymbol{y} \\
&= \big(f(\boldsymbol{x}_1) - s(\boldsymbol{x}_1)) \quad \cdots \quad f(\boldsymbol{x}_N) - s(\boldsymbol{x}_N))\big)\, \mathsf{K}^{-1}\boldsymbol{y} = \begin{pmatrix}0 & \cdots & 0\end{pmatrix} \mathsf{K}^{-1}\boldsymbol{y} = 0
\end{aligned}$$

because we know that $f(\boldsymbol{x}_i) = s(\boldsymbol{x}_i)$, $1 \leq i \leq N$, if s interpolates f at those points. Using this, we can prove that $\|f - s\|_{\mathcal{H}_K} < \|f\|_{\mathcal{H}_K}$:

$$\|f\|^2_{\mathcal{H}_K} = \|f - s + s\|^2_{\mathcal{H}_K} = \|f - s\|^2_{\mathcal{H}_K} + 2\langle f - s, s\rangle_{\mathcal{H}_K} + \|s\|^2_{\mathcal{H}_K} = \|f - s\|^2_{\mathcal{H}_K} + \|s\|^2_{\mathcal{H}_K} > \|f - s\|^2_{\mathcal{H}_K}, \tag{3.3}$$

which implies that (3.2) is a better bound. Even so, this tighter error bound does not seem to play a significant role in the kernel literature.

Since $\|f\|_{\mathcal{H}(K,\Omega)}$ usually is not computable (remember, we do not even know f, but want to reconstruct it from the data) the standard error bound is not very useful for practical situations. On the other hand, if we assume that our approximation s is good, i.e.,

$$\|f - s\|_{\mathcal{H}_K} \leq \delta_\varepsilon \|s\|_{\mathcal{H}_K}$$

for some not too large constant δ_ε, then the Golomb-Weinberger improved error bound yields a mostly computable error bound

$$|f(\boldsymbol{x}) - s(\boldsymbol{x})| \leq \delta_\varepsilon \|s\|_{\mathcal{H}_K} P_{K,\mathcal{X}}(\boldsymbol{x}).$$

This is indeed computable since $\|s\|_{\mathcal{H}_K} = \sqrt{\boldsymbol{y}^T \mathsf{K}^{-1}\boldsymbol{y}}$:

$$\begin{aligned}
\|s\|^2_{\mathcal{H}_K} = \langle s, s\rangle_{\mathcal{H}_K} &= \langle \boldsymbol{y}^T \mathsf{K}^{-1}\boldsymbol{k}(\cdot), \boldsymbol{k}(\cdot)^T \mathsf{K}^{-1}\boldsymbol{y}\rangle_{\mathcal{H}_K} \\
&= \boldsymbol{y}^T \mathsf{K}^{-1}\langle \boldsymbol{k}(\cdot), \boldsymbol{k}(\cdot)^T\rangle_{\mathcal{H}_K} \mathsf{K}^{-1}\boldsymbol{y} = \boldsymbol{y}^T \mathsf{K}^{-1}\mathsf{K}\mathsf{K}^{-1}\boldsymbol{y} = \boldsymbol{y}^T \mathsf{K}^{-1}\boldsymbol{y}.
\end{aligned} \tag{3.4}$$

Also, recall that this term $\boldsymbol{y}^T \mathsf{K}^{-1}\boldsymbol{y}$ appears in (1.17) for the zero-mean Gaussian field.