Positive Definite Kernels: Opportunities and Challenges

Michael McCourt

Department of Mathematical and Statistical Sciences
University of Colorado, Denver

Applied Mathematics Seminar
Hunter College
April 10, 2014
Acknowledgements

I would like to extend my thanks to

- John Loustau
- Marcello Lucia
- Sandy Prior
- Thelma Jorge
- Norma Moy
Solving problems numerically

Opportunity

As applications continue to increase in complexity, and computer hardware becomes more powerful and inexpensive, *better* numerical methods will be required to conduct the required simulations.
Solving problems numerically

Opportunity

As applications continue to increase in complexity, and computer hardware becomes more powerful and inexpensive, better numerical methods will be required to conduct the required simulations.

My Strategy

Kernel-based methods are a useful tool for computations including:

- Function approximation and interpolation,
- Integration,
- Partial differential equations (BVPs),
- Machine and statistical learning, and
- Optimization and surrogate modeling.
Topics of Discussion

- Introduce positive definite kernels and their properties
- Discuss the scattered data interpolation problem
  - Explain how a kernel basis can be used to solve that problem
  - Explore the impact of different kernel choices in the quality of the approximation
- Explore various kernel-based schemes for solving PDEs
  - Method of fundamental solutions
  - Kernel collocation
  - RBF-based finite differences
What is a kernel?

Kernel functions

- Essentially, any function of two variables: $K(x, z)$ is a kernel.
- I only care about symmetric kernels: $K(x, z) = K(z, x)$.
- **Positive definite kernels** can be interpreted as functional analysis analogs of positive definite matrices.

\[
v^T K v > 0, \quad \text{or} \quad \int_{\mathbb{R}^n} \int_{\mathbb{R}^n} K(x, z)v(x)v(z) \, dx \, dz > 0.
\]

- A reproducing kernel Hilbert space $\mathcal{H}_K$ is a space of functions which are in some way *generated* by the kernel $K$:

\[
\langle K(x, \cdot), f \rangle_{\mathcal{H}_K} = f(x), \quad \text{if} \ f \in \mathcal{H}_K.
\]
What is a kernel?

Kernel functions

- Essentially, any function of two variables: $K(x, z)$ is a kernel.
- I only care about symmetric kernels: $K(x, z) = K(z, x)$.
- **Positive definite kernels** can be interpreted as functional analysis analogs of positive definite matrices.

$$
\mathbf{v}^T K \mathbf{v} > 0, \quad \text{or} \quad \int_{\mathbb{R}^n} \int_{\mathbb{R}^n} K(x, z) v(x) v(z) \, dx \, dz > 0.
$$

- A reproducing kernel Hilbert space $\mathcal{H}_K$ is a space of functions which are in some way *generated* by the kernel $K$:

$$
\langle K(x, \cdot), f \rangle_{\mathcal{H}_K} = f(x), \quad \text{if } f \in \mathcal{H}_K.
$$

**Note:** This term $K(x, z)$ is unrelated to the kernel (null space) of a matrix.
Positive Definite Kernels

Positive Definite Kernel and Radial Basis Function (RBF) are used interchangeably ... but

- Not all kernels are radial: \( K(x, z) = \min(x, z) - xz \)
- Not all RBFs are positive definite: \( \phi(r) = \sqrt{r^2 + c^2} \)

Even so, I may switch back and forth between the terms, thus

\[
\text{kernel} = \text{RBF}
\]

unless otherwise specified.
Sample Kernels

Because all kernels (for now) are radial, we define \( r = \| \mathbf{x} - \mathbf{z} \| \).

Gaussians

\[
K(\mathbf{x}, \mathbf{z}) = \exp \left( -\varepsilon^2 r^2 \right)
\]

Inverse Multiquadrics

\[
K(\mathbf{x}, \mathbf{z}) = \frac{1}{\sqrt{1 + \varepsilon^2 r^2}}
\]

\( C_2 \) Matérn

\[
K(\mathbf{x}, \mathbf{z}) = (1 + r) \exp (-\varepsilon r)
\]

\( C_4 \) Wendland

\[
K(\mathbf{x}, \mathbf{z}) = (1 - \varepsilon r)^6 \left( 35(\varepsilon r)^2 + 18\varepsilon r + 3 \right)
\]
These RBFs may be thought of as a similarity measure.
Fundamental Application (Scattered Data Fitting)

Given data \((x_j, y_j), j = 1, \ldots, N,\) with \(x_j \in \mathbb{R}^d, y_j \in \mathbb{R},\) find a (continuous) function \(s\) such that \(s(x_j) = y_j, j = 1, \ldots, N.\)
Fundamental Application (Scattered Data Fitting)

Given data \((x_j, y_j), j = 1, \ldots, N\), with \(x_j \in \mathbb{R}^d\), \(y_j \in \mathbb{R}\), find a (continuous) function \(s\) such that \(s(x_j) = y_j, j = 1, \ldots, N\).

Consider kernel-based interpolation - the approximation is a linear combination of RBFs (not polynomials)

\[
s(x) = \sum_{j=1}^N c_j K(x, x_j), \quad x \in \Omega \subseteq \mathbb{R}^d
\]

with \(K : \Omega \times \Omega \to \mathbb{R}\) a positive definite kernel.

To find \(c_j\) solve the interpolation equations

\[
s(x_i) = f(x_i) = y_i, \quad i = 1, \ldots, N
\]

which gives a linear system \(Kc = y\); \(K\) is symmetric positive definite.
Fundamental Application (Scattered Data Fitting)

Given data \((x_j, y_j), j = 1, \ldots, N,\) with \(x_j \in \mathbb{R}^d, y_j \in \mathbb{R},\) find a (continuous) function \(s\) such that \(s(x_j) = y_j, j = 1, \ldots, N.\)
Fundamental Application (Scattered Data Fitting)

Given data \((x_j, y_j), \ j = 1, \ldots, N, \) with \(x_j \in \mathbb{R}^d, \ y_j \in \mathbb{R}, \) find a (continuous) function \(s\) such that \(s(x_j) = y_j, \ j = 1, \ldots, N.\)
Fundamental Application (Scattered Data Fitting)

Given data \((x_j, y_j), j = 1, \ldots, N,\) with \(x_j \in \mathbb{R}^d, y_j \in \mathbb{R},\) find a (continuous) function \(s\) such that \(s(x_j) = y_j, j = 1, \ldots, N.\)
Fundamental Application (Scattered Data Fitting)

Given data \((x_j, y_j), j = 1, \ldots, N,\) with \(x_j \in \mathbb{R}^d, y_j \in \mathbb{R},\) find a (continuous) function \(s\) such that \(s(x_j) = y_j, j = 1, \ldots, N.\)
Fundamental Application (Scattered Data Fitting)

Given data $(x_j, y_j)$, $j = 1, \ldots, N$, with $x_j \in \mathbb{R}^d$, $y_j \in \mathbb{R}$, find a (continuous) function $s$ such that $s(x_j) = y_j$, $j = 1, \ldots, N$. 
Fundamental Application (Scattered Data Fitting)

Given data \((x_j, y_j), j = 1, \ldots, N,\) with \(x_j \in \mathbb{R}^d, y_j \in \mathbb{R},\) find a (continuous) function \(s\) such that \(s(x_j) = y_j, j = 1, \ldots, N.\)
Fundamental Application (Scattered Data Fitting)

Given data \((x_j, y_j), j = 1, \ldots, N,\) with \(x_j \in \mathbb{R}^d, y_j \in \mathbb{R},\) find a (continuous) function \(s\) such that \(s(x_j) = y_j, j = 1, \ldots, N.\)
Kernels vs. Polynomials

A polynomial interpolation scheme (in 1D) looks like:

\[ p_f(x) = a_1 + a_2 x + a_3 x^2 + \ldots + a_n x^{n-1}. \]

The kernel-based scheme takes the form

\[ s(x) = c_1 K(x, x_1) + c_2 K(x, x_2) + c_3 K(x, x_3) + \ldots + c_n K(x, x_n). \]

Why is this kernel (RBF) approach preferable?
Kernels vs. Polynomials

A polynomial interpolation scheme (in 1D) looks like:

\[ p_f(x) = a_1 + a_2 x + a_3 x^2 + \ldots + a_{n-1} x^{n-1}. \]

The kernel-based scheme takes the form

\[ s(x) = c_1 K(x, x_1) + c_2 K(x, x_2) + c_3 K(x, x_3) + \ldots + c_n K(x, x_n). \]

Why is this kernel (RBF) approach preferable?

- Uniqueness is guaranteed in higher dimensions.
- RBF methods are often referred to as meshfree.
- Optimality of these methods is provable in some circumstances.
- Properties such as smoothness are variable.
- The locality of the basis functions can be parameterized.
RBFs in applications: Why aren’t there more?

For many problems, kernel-based approximations are appropriate; sometimes they would be optimal. Why are they not more prevalent?
RBFs in applications: Why aren’t there more?

For many problems, kernel-based approximations are appropriate; sometimes they would be optimal. Why are they not more prevalent?

Possible barriers to widespread RBF usage

1. Mathematical complexity
2. Stability concerns
3. Computational cost
4. Presence of one or more free parameters
Gaussian Kernel Interpolation

Probably the most common kernel in applications is the Gaussian

\[ K(x, z) = \exp \left( -\varepsilon^2 \| x - z \|^2 \right), \]

The \( \varepsilon \) value determines the locality of the basis functions.
Gaussian Kernel Interpolation

Probably the most common kernel in applications is the Gaussian

\[
K(x, z) = \exp \left( -\varepsilon^2 \|x - z\|^2 \right),
\]

The \( \varepsilon \) value determines the locality of the basis functions.

**Opportunity**

Gaussians are great choices for interpolating very smooth functions.

**Problem**

For some (potentially very accurate) choices of \( \varepsilon \), the interpolation matrix \( K \) may be very ill-conditioned.
Scattered Data Fitting with Gaussians

Gaussians can be spectrally accurate, allowing for high accuracy with smaller $N$ (number of input points). For some applications, the cost of producing a data point is significant, so this is useful.

The choice of shape parameter $\epsilon$ can have a great effect on the accuracy and condition of the interpolant.

- $\epsilon=10$, $\kappa \approx 1$
- $\epsilon=1$, $\kappa \approx 10^6$
- $\epsilon=.5$, $\kappa \approx 10^9$
Naive Gaussian Interpolation Results

As $\varepsilon \to 0$ the $K(\cdot, x_i)$ and $K(\cdot, x_j)$ start to resemble each other, producing a matrix which looks increasingly like a matrix of all ones. 

\[
\begin{pmatrix}
1 & \ldots & 1 \\
\vdots & \ddots & \vdots \\
1 & \ldots & 1
\end{pmatrix}
\]
Better Gaussian Interpolation Results

This is what we want to see:

- No instability for smaller $\varepsilon$
- Recovering the $\varepsilon \to 0$ polynomial limit [Fornberg, 2002]

How can we produce this result?
Eigenfunction (Mercer) Series Using Hilbert-Schmidt Theory

\[ K(x, z) = e^{-\varepsilon^2(x-z)^2} = \sum_{n=0}^{\infty} \lambda_n \varphi_n(x) \varphi_n(z) \]
Eigenfunction (Mercer) Series Using Hilbert-Schmidt Theory

\[ K(x, z) = e^{-\varepsilon^2(x-z)^2} = \sum_{n=0}^{\infty} \lambda_n \varphi_n(x) \varphi_n(z) \]

where

\[ \lambda_n = \sqrt{\frac{\alpha^2}{\alpha^2 + \delta^2 + \varepsilon^2}} \left( \frac{\varepsilon^2}{\alpha^2 + \delta^2 + \varepsilon^2} \right)^n, \quad \varphi_n(x) = \gamma_n e^{-\delta^2 x^2} H_n(\alpha \beta x) \]

with \( H_n \) Hermite polynomials,

\[ \beta = \left( 1 + \left( \frac{2\varepsilon}{\alpha} \right)^2 \right)^{\frac{1}{4}}, \quad \gamma_n = \sqrt{\frac{\beta}{2^n \Gamma(n+1)}}, \quad \delta^2 = \frac{\alpha^2}{2} \left( \beta^2 - 1 \right) \].
Eigenfunction (Mercer) Series Using Hilbert-Schmidt Theory

\[
K(x, z) = e^{-\varepsilon^2(x-z)^2} = \sum_{n=0}^{\infty} \lambda_n \varphi_n(x) \varphi_n(z)
\]

where

\[
\lambda_n = \sqrt{\frac{\alpha^2}{\alpha^2 + \delta^2 + \varepsilon^2}} \left( \frac{\varepsilon^2}{\alpha^2 + \delta^2 + \varepsilon^2} \right)^n, \quad \varphi_n(x) = \gamma_n e^{-\delta^2 x^2} H_n(\alpha \beta x)
\]

with \(H_n\) Hermite polynomials,

\[
\beta = \left(1 + \left(\frac{2\varepsilon}{\alpha}\right)^2\right)^{\frac{1}{4}}, \quad \gamma_n = \sqrt{\frac{\beta}{2^n \Gamma(n+1)}}, \quad \delta^2 = \frac{\alpha^2}{2} \left(\beta^2 - 1\right).
\]

These eigenpairs solve the Hilbert-Schmidt eigenvalue problem,

\[
\int_{-\infty}^{\infty} e^{-\varepsilon^2(x-z)^2} \varphi_n(z) \rho(z) \, dz = \lambda_n \varphi_n(x), \quad \rho(x) = \frac{\alpha}{\sqrt{\pi}} e^{-\alpha^2 x^2}.
\]
Eigenfunctions in the $\varepsilon \to 0$ limit

Studying the limits as $\varepsilon \to 0$ shows

$$\lim_{\varepsilon \to 0} \beta = \lim_{\varepsilon \to 0} \left(1 + \left(\frac{2\varepsilon}{\alpha}\right)^2\right)^{\frac{1}{4}} = 1,$$

$$\lim_{\varepsilon \to 0} \delta^2 = \lim_{\varepsilon \to 0} \frac{\alpha^2}{2} \left(\beta^2 - 1\right) \approx \lim_{\varepsilon \to 0} \varepsilon^2,$$

which in turn allows us to conclude that

$$\lim_{\varepsilon \to 0} \varphi_n(x) = \lim_{\varepsilon \to 0} \gamma_n e^{-\delta^2 x^2} H_n(\alpha \beta x) = \gamma_n H_n(\alpha x).$$

This supports the polynomial limit proved more than 10 years ago.

Infinite Smoothness Only

This limit is true only for analytic kernels - kernels with finite smoothness have (often) a piecewise polynomial spline limit.
Eigenvalues in the $\varepsilon \rightarrow 0$ limit

As $\varepsilon \rightarrow 0$, the eigenvalues $\lambda_n$ decay increasingly fast

$$\lim_{\varepsilon \to 0} \lambda_n = \lim_{\varepsilon \to 0} \sqrt{\frac{\alpha^2}{\alpha^2 + \delta^2 + \varepsilon^2}} \left( \frac{\varepsilon^2}{\alpha^2 + \delta^2 + \varepsilon^2} \right)^n$$

$$= \sqrt{\frac{\alpha^2}{\alpha^2}} \left( \frac{1}{\alpha^2} \right)^n \lim_{\varepsilon \to 0} \varepsilon^{2n}$$

This decay is a major cause of the ill-conditioning in the kernel interpolation problem (we will see this).

Avoiding Instability

We can use the eigenexpansion of the Gaussians to create a stable basis for them.
A Matrix-Vector Description of the Interpolant

Our interpolant

\[ s(x) = \sum_{k=1}^{N} c_k K(x, x_k) \]

can be written using vectors and matrices

\[ k(x) = \begin{pmatrix} K(x, x_1) \\ \vdots \\ K(x, x_N) \end{pmatrix}, \quad K = \begin{pmatrix} k(x_1)^T \\ \vdots \\ k(x_N)^T \end{pmatrix} \]

as

\[ s(x) = k(x)^T c = k(x)^T K^{-1} y, \]

because \( Kc = y \) (the interpolation equations).
The Hilbert-Schmidt SVD

We can create (analytically) a special decomposition of $K$ which will allow for a stable evaluation of $s$.

$$K = \Psi \Lambda_1 \Phi_1^T$$

We call this the Hilbert-Schmidt SVD (HS-SVD) because

- $\Lambda_1$ - Positive decreasing values ($\lambda_n$) in a diagonal matrix
- $\Psi, \Phi_1^T$ - Generated by orthogonal eigenfunctions
Using the Hilbert-Schmidt SVD

Eliminating the Ill-conditioning

$\Lambda$ contains most of the ill-conditioning; by removing it analytically, we can stably evaluate $s$:

\[
s(x) = k(x)^T K^{-1} y
= \psi(x)^T \Lambda_1 \Phi_1^T \Phi_1 \Phi_1^T \Lambda_1^{-1} \psi^{-1} y
= \psi(x)^T \psi^{-1} y.
\]

All the ill-conditioning from $\Lambda_1$ which would normally have appeared in $K^{-1}$ has been resolved. Our new stable basis is $\psi$. 
A New, More Stable, Basis

Old Basis, $\varepsilon=10$

New Basis, $\varepsilon=10$

Old Basis, $\varepsilon=.5$

New Basis, $\varepsilon=.5$
A New, More Stable, Basis

This stable basis $\psi$ allows us to evaluate the interpolant

$$s(x) = \psi(x)^T \psi^{-1} y,$$

rather than

$$s(x) = k(x)^T (\psi \Lambda_1 \phi_1^T)^{-1} y.$$
Gaussian Random Fields - Kriging

In addition, there is an alternate explanation of the interpolation solution

\[ s(x) = k(x)^T K^{-1} y. \]

Suppose, instead, that our data is one realization of a Gaussian random field

Random Field: \( Y = \{ Y_x \}_{x \in \Omega}. \)

This random field has the properties:

- For a fixed \( x, \) \( Y_x = Y(x, \cdot) \) is a random variable.
- For a fixed \( \omega, \) \( y(\cdot) = Y(\cdot, \omega) \) is a deterministic function of \( x \) referred to as a realization of the random field.

The mean of \( Y \) is a function \( \mu_Y \) which is defined at any point \( x \in \Omega \) as

\[
\mu_Y(x) = E(Y_x) = \int_{\mathcal{W}} Y_x(\omega) dP(\omega) = \int_{\mathbb{R}} y dF_{Y_x}(y)
\]
Gaussian Random Fields - Covariance

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.

Much like the Gaussian random variable, Gaussian random fields are defined entirely by their mean and covariance:

$$K(x, z) = \text{Cov}(Y_x, Y_z) = E((Y_x - \mu_Y(x))(Y_z - \mu_Y(z))) = E(Y_x Y_z) - \mu_Y(x) \mu_Y(z)$$  \hspace{1cm} (1)
Random Fields to Random Variables

For a finite set of points $\mathcal{X} = \{x_1, \ldots, x_N\} \in \Omega$, the vector of random variables $\mathbf{Y}_\mathcal{X} = (Y_{x_1} \cdots Y_{x_N})^T$ has the distribution

$$\mathbf{Y}_\mathcal{X} \sim \mathcal{N}(\mathbf{\mu}, \mathbf{K}), \quad (2)$$

where $\mathbf{\mu} = \mathbf{E}(\mathbf{Y}_\mathcal{X})$ and $(\mathbf{K})_{i,j} = \text{Cov}(Y_{x_i}, Y_{x_j})$. This multivariate normal distribution has the density

$$p_{\mathbf{Y}_\mathcal{X}}(\mathbf{y}) = \frac{1}{\sqrt{2\pi} \det(\mathbf{K})} \exp \left( -\frac{1}{2} \mathbf{y}^T \mathbf{K}^{-1} \mathbf{y} \right). \quad (3)$$
Duality Between Analysis and Gaussian Random Fields

Question
Why am I talking about this?

It turns out, there is a duality between the reproducing kernel Hilbert space $\mathcal{H}_K$ and the Hilbert space $\mathcal{H}_Y$ which is the set of all linear combinations of random variables $Y_x$ together with their $L_2$-limits.

The Loève representation theorem says that the inner products in these Hilbert spaces are identical, i.e.,

$$\langle Y_x, Y_z \rangle_{\mathcal{H}_Y} = E(Y_x Y_z) = K(x, z) = \langle K(\cdot, x), K(\cdot, z) \rangle_{\mathcal{H}_K},$$

where the final identity occurs because of the reproducing property of $K$. 
Predicting the Random Field at Unobserved Values

Similarly to the goal of interpolation
  - Given data from a function approximate that function,
the goal for studying Gaussian random fields is
  - Given data from a random field, predict other values from that field.

One way to make predictions is to study the random variable

\[ Y_{x_0} | Y = y, \]

Making predictions
What is the distribution of possible values \( Y_{x_0} \) can take, given the observed data?
Predicting the Random Field at Unobserved Values

An important result is the following:

\[ Y_{x_0} | Y = y \sim \mathcal{N}(k(x_0)^TK^{-1}y, K(x_0, x_0) - k(x_0)^TK^{-1}k(x_0)). \]

This gives us the exact distribution for our predictions - most importantly:

\[ E(Y_{x_0} | Y \chi = y) = k(x_0)^TK^{-1}y. \]

This is the same result as what we saw in the interpolation case, which suggests the following:

**Duality**

The interpolation and spatial statistics analysis of scattered data can (and often do) produce the same result.
Solving PDEs with Kernels

We have used a linear combination of kernels to solve a scattered data approximation problem.

A similar strategy can be used to solve boundary value problems of the form

\[ \mathcal{L}s = f, \quad \text{on interior,} \]
\[ \mathcal{B}s = g, \quad \text{on boundary,} \]

for \( \mathcal{L} \) a linear differential operator.

Note

Nonlinear and time-stepping problems can also be solved, though we will not talk about them here.
Solving PDEs numerically: Approximate solution

Our focus in this talk will be on constructing a function which approximates the true solution.

Goal
Create a function

\[ s(x) = \sum_{k=1}^{N} c_k K(x, z_k) \]

which “approximately” solves the PDE in some sense we deem appropriate.

Appropriate approximation

- How do we enforce the PDE and BC on our \( s \)?
- What are these \( z_k \) centers?
Solving PDEs numerically: Approximate solution
Our focus in this talk will be on constructing a function which approximates the true solution.

Goal
Create a function

\[ s(x) = \sum_{k=1}^{N} c_k K(x, z_k) \]

which “approximately” solves the PDE in some sense we deem appropriate.

Appropriate approximation

- How do we enforce the PDE and BC on our \( s \)?
  - Choose points \( x_k \) on the interior/boundary at which the PDE is collocated (solved).
- What are these \( z_k \) centers?
Kernel Collocation - Point Selection

Historical note

Almost all papers use \( x_k = z_k \), that is, the centers and collocation points are the same - this matches the interpolation setting.

Recently, researchers have challenged this strategy, but it is still the standard and the only setting we consider here.

Matching the centers and collocation points \( (N = N_L + N_B) \) yields:

\[
\begin{bmatrix}
\mathcal{L} K(x_1, x_1) & \cdots & \mathcal{L} K(x_1, x_N) \\
\cdots & & \cdots \\
\mathcal{L} K(x_{N_L}, x_1) & \cdots & \mathcal{L} K(x_{N_L}, x_N) \\
\mathcal{B} K(x_{N_L+1}, x_1) & \cdots & \mathcal{B} K(x_{N_L+1}, x_N) \\
\cdots & \cdots & \cdots \\
\mathcal{B} K(x_{N_L+N_B}, x_1) & \cdots & \mathcal{B} K(x_{N_L+N_B}, x_N)
\end{bmatrix}
\begin{bmatrix}
c_1 \\
\vdots \\
c_N
\end{bmatrix}
= 
\begin{bmatrix}
f(x_1) \\
\vdots \\
f(x_{N_L}) \\
g(x_{N_L+1}) \\
\vdots \\
g(x_{N_L+N_B})
\end{bmatrix}
\]
Kernel Collocation - Stable Basis

Using appropriate shorthand, we can rewrite this system as

\[
\begin{pmatrix}
K_L \\
K_B
\end{pmatrix}
\begin{pmatrix}
c \\
\end{pmatrix}
= 
\begin{pmatrix}
f \\
g
\end{pmatrix}
\]

The HS-SVD allows us to write

\[
K_L = \Psi_L \Lambda_1 \Phi_1^T
\]
\[
K_B = \Psi_B \Lambda_1 \Phi_1^T
\]

because \( \Lambda_1 \Phi_1^T \) is defined by the centers, not the collocation points. This produces

\[
\begin{pmatrix}
\Psi_L \\
\Psi_B
\end{pmatrix}
\begin{pmatrix}
\Lambda_1 \Phi_1^T
\end{pmatrix}
\begin{pmatrix}
c \\
\end{pmatrix}
= 
\begin{pmatrix}
f \\
g
\end{pmatrix}
\]

and the solution \( s \) is evaluated as

\[
s(\mathbf{x}) = \psi(\mathbf{x})^T \left( \begin{pmatrix}
\Psi_L \\
\Psi_B
\end{pmatrix} \right)^{-1} \begin{pmatrix}
f \\
g
\end{pmatrix} = \psi(\mathbf{x})^T \psi_{L,B}^{-1} \mathbf{y}_{f,g}.
\]
Stable Gaussians for Boundary Value Problems

Implementing the stable basis for this system will avoid the ill-conditioning which would otherwise plague the collocation solution.

This example is a 2D Helmholtz problem, comparing Gaussian collocation to Trefethen’s polynomial method and the standard Gaussian kernel collocation method.
Evaluating the Collocation Solution

Suppose that we wanted to evaluate our PDE solution at our collocation points $x_1, \ldots, x_N$, similarly to a finite difference method:

$$s = \begin{pmatrix} s(x_1) \\ \vdots \\ s(x_N) \end{pmatrix} = \begin{pmatrix} \psi(x_1)^T \psi^{-1}_{L,B} y_{f,g} \\ \vdots \\ \psi(x_N)^T \psi^{-1}_{L,B} y_{f,g} \end{pmatrix} = \begin{pmatrix} \psi(x_1)^T \\ \vdots \\ \psi(x_N)^T \end{pmatrix} \psi^{-1}_{L,B} y_{f,g} = \psi \psi^{-1}_{L,B} y_{f,g}$$

This relationship $s = \psi \psi^{-1}_{L,B} y_{f,g}$ suggests that $\psi \psi^{-1}_{L,B}$ plays a similar role as the standard finite difference matrix (plus boundary conditions).
RBF Differentiation Matrices

Observation
Applying $\psi \psi^{-1}_{L,B} \mathbf{y}_{f,g}$ evaluates a function (the approximate solution $s$) when given data (the PDE’s RHS $\mathbf{y}_{f,g}$).

Idea
Could we reverse the process ... that is, given function values, could we evaluate derivatives of that function?

Given data $\mathbf{y}$ at points $\mathbf{x}_1, \ldots, \mathbf{x}_N$, we can form an associated interpolant $s = \psi(\cdot)^T \psi^{-1} \mathbf{y}$.

Applying a derivative operator $\mathcal{D}$ to this produces

$$\mathcal{D} s = \mathcal{D} \psi(\cdot)^T \psi^{-1} \mathbf{y}.$$
RBF Differentiation Matrices

Evaluating the derivative of our interpolant at the interpolation points:

\[
\mathbf{s}_D = \begin{pmatrix}
\mathcal{D}s(x_1) & \\
\vdots & \\
\mathcal{D}s(x_N)
\end{pmatrix} = \begin{pmatrix}
\mathcal{D}\psi(x_1)^T \psi^{-1} \mathbf{y} & \\
\vdots & \\
\mathcal{D}\psi(x_N)^T \psi^{-1} \mathbf{y}
\end{pmatrix}
\]

\[
= \begin{pmatrix}
\mathcal{D}\psi(x_1)^T & \\
\vdots & \\
\mathcal{D}\psi(x_N)^T
\end{pmatrix} \psi^{-1} \mathbf{y} = \psi_D \psi^{-1} \mathbf{y}
\]

Our differentiation matrix \(\psi_D \psi^{-1}\) takes in data from a function \(f\) (evaluated at \(x_1, \ldots, x_N\) and stored in \(\mathbf{y}\)) and returns an approximation of \(\mathcal{D}f\) at the same locations.
We can expand our notation to evaluate $D$s at any location, not just the input data locations.

- $\psi^I_L$ - Evaluates $L\psi$ at the interior collocation points
- $\psi^B_B$ - Evaluates $B\psi$ at the boundary collocation points

Using this, the PDE, in pseudospectral form could be written as

$$\left(\begin{array}{cc}
\psi^I_L & \psi^{-1}
\end{array}\right) \mathbf{s} = \left(\begin{array}{c}
f
\end{array}\right) , \quad \text{because} \quad L\mathbf{s} = f$$

$$\left(\begin{array}{cc}
\psi^B_B & \psi^{-1}
\end{array}\right) \mathbf{s} = \left(\begin{array}{c}
g
\end{array}\right) , \quad \text{because} \quad B\mathbf{s} = g$$

This used by Trefethen to describe the differentiation matrix structure of polynomial collocation methods.
RBF Finite Differences

This design of RBF Differentiation matrices motivates one of the most promising new developments in meshfree PDE solvers: RBF-FD.

Strategy

Define a localized differential operator (analogous to finite differences) which can be used approximate derivatives.

Benefits

- No prescribed point distribution would be required (e.g., uniform points for finite differences, Chebyshev nodes for polynomials)
- The locality of the operator is a free parameter, and could even vary within a single PDE.
- Unlike collocation based operators, RBF-FD would be sparse.
RBF Finite Differences - Point Selection

Each point uses nearby points to approximate a differential operator.
RBF Finite Differences - Structure

For each $x_k$, nearby points, $\mathcal{X}_k$, are chosen to approximate the derivatives, and a one row differentiation matrix is computed for each point:

$$\Psi_{\mathcal{L}}^{x_k} (\Psi_{\mathcal{X}})^{-1}.$$

This matrix has the effect of taking in nearby function values and producing a linear combination of them to approximate the $\mathcal{L}$ operator at $x_k$. 
This seems like a really crummy structure which will be much worse than finite differences ...
But a simple reordering of our indices produces the familiar banded structure.
Multiple meanings

- As the number of points used in computing the finite difference stencil increases, this method should converge to traditional collocation.
- For a fixed number of neighbors, but an increasingly large number of collocation points, convergence should occur until saturation.
  - Eventually the quality will be bounded by the limited domain over which points are used to approximate the derivatives.

A problem of scale

For increasingly dense points, different $\varepsilon$ shape parameters (which control the kernel locality) may be needed to maintain convergence order.
RBF Finite Differences - Convergence

N=2070 total collocation points

RMS relative error vs number of neighbors included
RBF Finite Differences - Overview

1. A neighborhood is defined around each collocation point.
2. A linear combination of the points in that neighborhood are used to approximate the derivative at that point.
3. This produces a sparse operator which can be used to solve boundary value problems.
4. Choosing a neighborhood optimally, and then choosing the kernel to use for the approximation in the neighborhood ($\varepsilon$) is still an open problem.
Positive definite kernels (RBFs) can be used as a basis for scattered data interpolation and PDE collocation.
  - Some kernels see severe ill-conditioning.

A strong connection exists between reproducing kernels and Gaussian random fields.

A change of basis using the Hilbert-Schmidt SVD can circumvent the traditional ill-conditioning.

PDE collocation schemes can be constructed using the same strategy as interpolation.
  - RBF based finite differences are a new and exciting strategy.