Stable Kernel-Based Interpolation using Gaussian Eigenfunctions

Michael McCourt
Department of Mathematical and Statistical Sciences
University of Colorado, Denver

CCM Seminar
University of Colorado, Denver
August 27, 2013

Acknowledgements

I would like to extend my thanks to
- Lynn Bennethum
- Julien Langou
- Jan Mandel

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 goal

Kernel-based methods are a useful tool for computations including:
- Function approximation and interpolation,
- Integration,
- Boundary value problems,
- Machine and statistical learning, and
- Optimization and surrogate modeling.

I want to expose more applications to kernel-based methods.

RBFs in applications: Why aren’t there more?

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

Possible barriers to widespread RBF usage
- Mathematical complexity
- Stability concerns
- Computational cost
- Presence of one or more free parameters
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 \(s_i\) such that \(s_i(x_j) = y_j, j = 1, \ldots, N\).

Consider here multivariate kernel-based interpolation using a data-dependent linear function space

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

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

To find \(c_j\) solve the interpolation equations

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

which gives a linear system \(Kc = y; K\) is symmetric positive definite.

Kernels vs. Polynomials

The first interpolation scheme that comes to mind for many is polynomial interpolation:

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

The kernel-based scheme takes the form

\[ s_i(x) = c_1 K(x, x_1) + c_2 K(x, x_2) + c_3 K(x, x_3) + \ldots + c_N K(x, x_N). \]

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.

Sample Kernels

We will restrict our kernels (for now) to be radial: \(r = \|x - z\|\).

Gaussians

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

Inverse Multiquadrics

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

\(C_2\) Matérn

\[ K(x, z) = (1 + r) \exp\left(-\epsilon r\right) \]

\(C_4\) Wendland

\[ K(x, z) = (1 - \epsilon r)^6 \left(35(\epsilon r)^2 + 18\epsilon r + 3\right) \]
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.

My first results today will address this ill-conditioning.

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 \( \varepsilon \) can have a great effect on the accuracy and condition of the interpolant.

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.

This is what we want to see ... but how can we make it happen?
Eigenfunction (Mercer) Series Using Hilbert-Schmidt Theory

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

where

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

with \( H_n \) Hermite polynomials,

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

These eigenpairs solve the Hilbert-Schmidt eigenvalue problem,

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

Properties of this Eigenfunction Series

- The \( \{ \varphi_n \}_{n=0}^{\infty} \) are (weighted) \( L_2 \)-orthonormal,

\[ \int_{-\infty}^{\infty} \varphi_m(x) \varphi_n(x) \rho(x) \, dx = \delta_{mn}. \]

- The eigenfunction expansion is the optimal \( M \)-term approximation to the Gaussian in the \( L_2, \rho \) sense.

- The eigenvalues decay quickly as \( \varepsilon \to 0 \),

\[ \lambda_n \sim \varepsilon^{2n} \left( \frac{1}{\alpha^2 + \delta^2 + \varepsilon^2} \right)^n. \]

- Multiple dimensions follow neatly from the tensor product structure of the Gaussian.

\[ e^{-\varepsilon^2 ||x_1-z_1||^2 + \ldots + \varepsilon^2 ||x_d-z_d||^2} = e^{-\varepsilon^2 ||x_1-z_1||^2} \ldots e^{-\varepsilon^2 ||x_d-z_d||^2} \]

Developing the Hilbert-Schmidt SVD (GaussQR)

The eigenfunction structure is

\[ K(x, z) = \sum_{m=0}^{\infty} \lambda_m \varphi_m(x) \varphi_m(z), \]

\[ = \begin{pmatrix} \varphi_1(x) & \cdots & \varphi_N(x) \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_N \end{pmatrix} \begin{pmatrix} \varphi_1(z) \\ \vdots \\ \varphi_N(z) \end{pmatrix}, \]

In nice, compressed, vector notation,

\[ K(x, z) = \phi(x)^T \Lambda \phi(x) \]
Developing the Hilbert-Schmidt SVD (GaussQR)
The scattered data fitting problem asks us to find an interpolant
\[ s(x) = \sum_{k=1}^{N} c_k K(x, x_k), \]
\[ = k(x)^T c. \]
The coefficient vector \( c \) is defined by \( Kc = y \):
\[
\begin{pmatrix}
K(x_1, x_1) & \cdots & K(x_1, x_N) \\
\vdots & \ddots & \vdots \\
K(x_N, x_1) & \cdots & K(x_N, x_N)
\end{pmatrix}
\begin{pmatrix}
c_1 \\
\vdots \\
c_N
\end{pmatrix}
= \begin{pmatrix}
k(x_1)^T \\
\vdots \\
k(x_N)^T
\end{pmatrix}
\begin{pmatrix}
c_1 \\
\vdots \\
c_N
\end{pmatrix}
= \begin{pmatrix}
y_1 \\
\vdots \\
y_N
\end{pmatrix},
\]
which is just \( s(x_j) = y_j \), \( 1 \leq j \leq N \). We can write
\[ s(x) = k(x)^T K^{-1} y. \]

Developing the Hilbert-Schmidt SVD (GaussQR)
Now you gotta trust me ... Break \( \Phi \) and \( \Lambda \) into blocks.
\[ \Phi = (\Phi_1 \Phi_2), \quad \Lambda = \begin{pmatrix} \Lambda_1 \\ \Lambda_2 \end{pmatrix}, \]
so that \( \Phi_1 \) and \( \Lambda_1 \) are \( N \times N \). Let’s rewrite our basis using this structure
\[
k(x)^T = \Phi(x)^T \Lambda \Phi^T = \Phi(x)^T \begin{pmatrix} \Lambda_1 & \Phi_2^T \\ \Phi_2 & \Lambda_2 \end{pmatrix}
= \Phi(x)^T \begin{pmatrix} I_N \\ \Lambda_2 \Phi_2^T \Phi_1^{-T} \Lambda_1^{-1} \end{pmatrix} \Lambda_1 \Phi_1^T.
\]
Define a new function
\[ \psi(x)^T = \Phi(x)^T \begin{pmatrix} I_N \\ \Lambda_2 \Phi_2^T \Phi_1^{-T} \Lambda_1^{-1} \end{pmatrix} \]
to write
\[ k(x)^T = \psi(x)^T \Lambda_1 \Phi_1^T. \]

Developing the Hilbert-Schmidt SVD (GaussQR)
We can rewrite each row of \( K \) as
\[
k(x)^T = (K(x, x_1) \cdots K(x, x_N)) = (\phi(x)^T \Lambda \phi(x_1) \cdots \phi(x)^T \Lambda \phi(x_N)) = \phi(x)^T \Lambda (\phi(x_1) \cdots \phi(x_N)).
\]
Stacking these gives the \( K \) matrix
\[
K = \begin{pmatrix}
k(x_1)^T \\
\vdots \\
k(x_N)^T
\end{pmatrix} = \begin{pmatrix}
\phi(x_1)^T \\
\vdots \\
\phi(x_N)^T
\end{pmatrix} \Lambda (\phi(x_1) \cdots \phi(x_N)),
\]
which I will compress down to
\[ K = \Phi \Lambda \Phi^T, \quad \text{and} \quad k(x)^T = \phi(x)^T \Lambda \Phi. \]

Developing the Hilbert-Schmidt SVD (GaussQR)
Use this to rewrite \( K \), which was just rows of \( k \):
\[
K = \begin{pmatrix}
k(x_1)^T \\
\vdots \\
k(x_N)^T
\end{pmatrix} = \begin{pmatrix}
\psi(x_1)^T \\
\vdots \\
\psi(x_N)^T
\end{pmatrix} \Lambda_1 \Phi_1^T = \psi \Lambda_1 \Phi_1^T.
\]
Now we plug this in to \( s \) ...
\[
s(x) = k(x)^T K^{-1} y = \psi(x)^T \Lambda_1 \Phi_1^T \psi \Lambda_1^{-1} \psi^{-1} y = \psi(x)^T \psi^{-1} y.
\]
and all the ill-conditioning from \( \Lambda_1 \) which would normally have appeared in \( K^{-1} \) has been resolved. Our new stable basis is \( \psi \).
Stable Interpolation Results

\[ f(x) = \sin(x/2) - 2 \cos(x) + 4 \sin(\pi x), \quad x \in [-4, 4] \]

Things to note:
- Structure exists which can't be resolved with RBF-Direct
- RBF-QR can get the same accuracy as RBF-Direct for fewer points

Applications and extensions of GaussQR

Now that we have introduced the concept of kernel methods, and established the foundation of GaussQR, where have we gone since?

- Multiphysics coupling
- Boundary value problems
- Collocation
- Method of Particular Solutions
- Designer kernels
- Kernel parametrization through statistics
- Improved computational efficiency

Stable Gaussians for Boundary Value Problems

In the meshfree coupling section, we used the stable Gaussian basis to accurately approximate derivatives of interpolants along the interface manifold.

Using this derivative approximation scheme, we can also solve boundary value problems of the form

\[ \mathcal{L}u = f, \quad \text{on the interior } \Omega, \]

\[ Bu = g, \quad \text{on the boundary } \partial \Omega, \]

by assuming that the solution takes the form

\[ u(x) = \sum_{k=1}^{N} a_k K(x, x_k). \]
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 Fasshauer’s direct collocation method.

![Graph](image)

Likelihood functions for Gaussian Processes

The scattered data approximation problem has an analog in statistics called Kriging.

If, instead of trying to recover a function, we treat our scattered data as one realization of a Gaussian Process, we can prescribe a positive definite kernel $K$ as the presumed covariance between realizations of the Gaussian Process. The likelihood function of a zero-mean Gaussian Process (the probability of $\varepsilon$ given the data $(x_k, y_k), 1 \leq k \leq n$) is

$$P(\varepsilon | y) = (2\pi \det(K))^{-1/2} \exp\left(\frac{-1}{2} y^T K^{-1} y\right)$$

The matrix $K$ is the scattered data interpolation matrix.

Maximum Likelihood Estimation

We can use Maximum Likelihood Estimation to optimize our shape parameter. Maximizing

$$P(\varepsilon | y) = (2\pi \det(K))^{-1/2} \exp\left(\frac{-1}{2} y^T K^{-1} y\right)$$

requires evaluating $\det(K)$, which, given its ill-conditioning, is a dicey proposition.

Using the Hilbert-Schmidt SVD this determinant can be computed in pieces

$$\det(K) = \det(\Psi \Lambda_1 \Phi_1) = \det(\Psi) \det(\Lambda_1) \det(\Phi_1)$$

The values $\det(\Psi)$ and $\det(\Phi_1)$ can be computed stably (we presume) using standard techniques.

Evaluating $\det(\Lambda_1)$ can be done analytically because it is a diagonal matrix populated by our Hilbert-Schmidt eigenvalues.

![Graph](image)
Summary

- Kernel-based methods are powerful tools for many common problems in computational mathematics.
  - High (maybe spectral) accuracy
  - Meshfree
- They are not widely used in applications because of their computational cost and potential instability.
- We have begun to address stability concerns by changing basis.
  - This stable approach RBF-QR allows for interpolation, PDE, and statistical problems to be solved accurately.
- RBF-QR also allows us to create designer kernels without a closed form.
  - These can be designed to satisfy properties of interest.
    - Boundary conditions
    - Smoothness

Current Work/Applications

- EEG/MEG simulations using kernel methods to solve the forward component of the inverse problem.
- Surrogate modeling for parameter optimizing in stochastic dynamics.
- Incorporating GaussQR into ENO/WENO methods.

Future work and long term plans

- Study generalized Sobolev spaces for involving $\epsilon$ in Native space norm computations.
- Improve the quality of statistical methods for parameterizing kernels.
- Improve the computational feasibility of kernel methods.
  - Hierarchically semiseparable methods
  - Treecodes exploiting the Mercer’s series
- Find more applications where kernels are useful.