A Bayesian Approach to Determining a Good Shape Parameter for Meshfree Approximation

Michael McCourt

January 6, 2009

Abstract

The efficacy of approximation by radial basis functions is often determined by a scale parameter σ. Determining the appropriate scale paramteter is a nontrivial problem not only because of the nonlinearity of the kernel functions, but also because defining the quality of a σ is not straightforward. Many of the techniques for defining a *good* σ come from statistics, specifically frequentist statistics. While cross-validation, maximum likelihood estimation and minimizing the power function are commonly used, Bayesian inference is rarely utilized. This paper discusses a Bayesian analysis of the σ optimization and derives a Markov Chain Monte Carlo technique for determining a good shape parameter. Along the way we will cover some numerical issues which arise and compare the requirements of MCMC to other techniques.

1 The Gaussian Process Model for Radial Basis Interpolation

Much of the material from in the Fasshauer book [4] discusses meshfree approximation from a numerical analysis point of view - which is logical because he is a numerical analyst. In [14], the use of positive definite functions for predicting the output of Gaussian processes is discussed. This analyzes the same problem but puts different assumptions to use: rather than assuming that the underlying function we want to approximate has some level of smoothness (for which RBF are appropriate) we assume that the data we have is a realization of a Gaussian process and that the correlation between design points is an RBF. The following discussion originated in [11] and [10] with adaptations for more general RBF.

1.1 Defining the problem

Suppose we want to fit the model

$$Y(x) = \sum_{j=1}^{N_f} \beta_j f_j(x) + Z(x)$$

to design points (x_i, y_i), $1 \leq i \leq n$ with the goal to be able to make accurate predictions for future (x_0, y_0). It is assumed that we are not able to choose the (x_i, y_i); being able to do so will not affect the formulation of the problem, although it may affect the quality of prediction. The f_j terms are deterministic functions - eg. they may be polynomials if our model expects polynomial reproduction. $Z(x)$ is a Gaussian Process with zero mean and covariance

$$\text{Cov}(Z(u), Z(v)) = \epsilon^2 \Phi(u, v) \tag{1.1}$$

where $\Phi(u, v) = \phi(\sigma\|u - v\|)$ is the radial basis function (or kernel) we have chosen, with σ the shape parameter. ϵ^2 is the process variance.

For simplicity, we will ignore the regression terms by setting $\beta_j \equiv 0$ – this means we presume the data (x_i, y_i) is a realization of Z. These terms can be handled separately in general because Z is assumed to be zero mean and thus $\mathbb{E}[Y] = \sum_{j=1}^{N_f} \beta_j f_j(x)$. A more general statistical formulation and discussion is found in [15].

Once we assume $\beta_j \equiv 0$, for any given shape parameter σ, the best linear unbiased predictor for a new x_0 is

$$\hat{y}_0(\sigma) = \Phi(x_0)^T \Phi^{-1} y \tag{1.2}$$

where Φ is the covariance matrix, y is the vector of design values, $\Phi(x_0)$ is the vector of correlations between x_0 and the design points. This however assumes that we have chosen a suitable σ.

1.2 Numerical demonstration of σ dependence

The choice of σ is very important in making accurate predictions. As an example, consider the input data generated by evaluating $(1 + x^2)^{-1}$ at 6 evenly space points between 0 and 1.

x	y
0	1.0000
0.2	0.9806
0.4	0.9285
0.6	0.8575
0.8	0.7809
1.0	0.7071

When we compare the mean square errors of the unique polynomial interpolant and the Gaussian RBF interpolant for various values of σ the result is Figure 1. It is apparent in that graph that for some values of σ, the RBF interpolant outperforms the simple polynomial interpolation. This is a motivation for us to find the *best* σ possible.

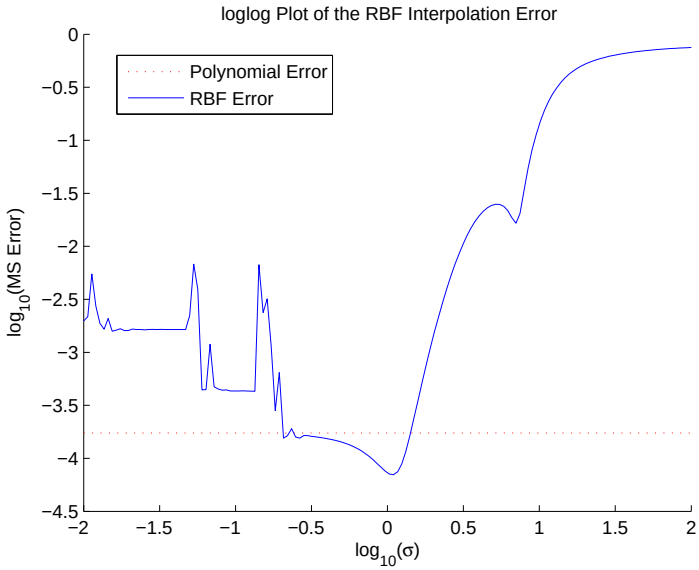

Figure 1: For some values of σ the RBF interpolant is better than a polynomial.

Of course, finding the *best* σ is not a straightforward problem; even defining what *best* means is a subjective decision. This can be done in numerous ways, but the goal of this paper is to analyze this predicament in a Bayesian framework.

1.3 Finding the posterior distribution

Standard techniques for determining an appropriate shape parameter involve optimizing some function - in [4] potential objective functions include the kriging variance, the leave-one-out cross validation error,

and the likelihood function. Here we want to consider the posterior distribution $p(\sigma, \epsilon^2|y)$ to which we can do some standard manipulations,

$$\begin{aligned} p(\sigma, \epsilon^2|y)p(y) &= p(y|\sigma, \epsilon^2)p(\sigma, \epsilon^2) \\ p(\sigma, \epsilon^2|y) &\propto p(y|\sigma, \epsilon^2)p(\sigma, \epsilon^2) \\ p(\sigma, \epsilon^2|y) &\propto L(\sigma, \epsilon^2)p(\sigma, \epsilon^2). \end{aligned}$$

We define $L(\sigma, \epsilon^2)$ as the likelihood of the parameters given the data, and since we are assuming that the design points are not in our control, L is only a function of the parameters. Notice the presence of *proportional to* statements rather than equalities: because we are only interested in the shape of the posterior distribution we needn't carry around scaling or shifting terms.

For this paper we are going to take advantage of our assumption that the data is generated by a Gaussian Process. From this we know the likelihood of the parameters given the data is

$$L(\sigma, \epsilon^2) \propto \frac{\exp\left(-\frac{1}{2\epsilon^2} y^T \Phi^{-1} y\right)}{(\epsilon^2)^{n/2}\sqrt{\det(\Phi)}}.$$

See [10]. We may also make the assumption that ϵ^2 and σ are independent, which means that

$$p(\epsilon^2, \sigma) = p(\epsilon^2)p(\sigma).$$

If we make no prior assumptions about ϵ^2, that would yield the improper distribution

$$p(\epsilon^2) = \begin{cases} 1, & \epsilon^2 \geq 0 \\ 0, & \text{else} \end{cases}$$

which can be put in the posterior distribution

$$p(\sigma, \epsilon^2|y) \propto \frac{\exp\left(-\frac{1}{2\epsilon^2} y^T \Phi^{-1} y\right)}{(\epsilon^2)^{n/2}\sqrt{\det(\Phi)}} p(\sigma), \qquad \epsilon^2 \geq 0.$$

Because of the special structure of this distribution, ϵ^2 can be analytically integrated out to yield

$$\begin{aligned} P(\sigma) &= \int_0^\infty p(\sigma, \epsilon^2|y)p(\sigma)d\epsilon^2, \\ &= (y^T \Phi^{-1} y)^{-n/2} \det(\Phi)^{-1/2} p(\sigma). \end{aligned}$$

which is the distribution for the shape parameter given the assumptions we have made. Note that P can be interpreted as $p(\sigma|y)$, the profile posterior over all ϵ^2.

For more information about radial basis functions and Gaussian Processes please reference [4], [16] or [14]. We now need to choose a prior distribution for σ.

1.4 Prior knowledge

There is still a term $p(\sigma)$ in $\tilde{P}$ which in Bayesian inference is referred to as the prior distribution [5]. This can be used to incorporate any assumptions we make about σ based on physical characteristics of the underlying problem. For our purposes here, we are going to consider the use of 3 categories of priors and then choose an appropriate distribution. Possible options include

- Uninformative prior - $p(\sigma) = 1$
- Parametric prior - $p(\sigma) = \Gamma(\sigma; \alpha, \beta)$
- Black Box prior - $p(\sigma) = \Psi(\sigma)$

If we have no knowledge of which shape parameter values are more appropriate, then the uninformative prior is the logical choice. As always, we need to be worried about the use of an improper prior distribution possibly yielding an improper posterior distribution (refer to [2]); here the form of the likelihood guarantees a proper posterior. Even so, to assume that we know nothing about σ is to sell ourselves short: in [4] there are numerous examples with varying test functions, design points and kernels and the *optimal* shape parameter is rarely outside of $(0, 40)$.

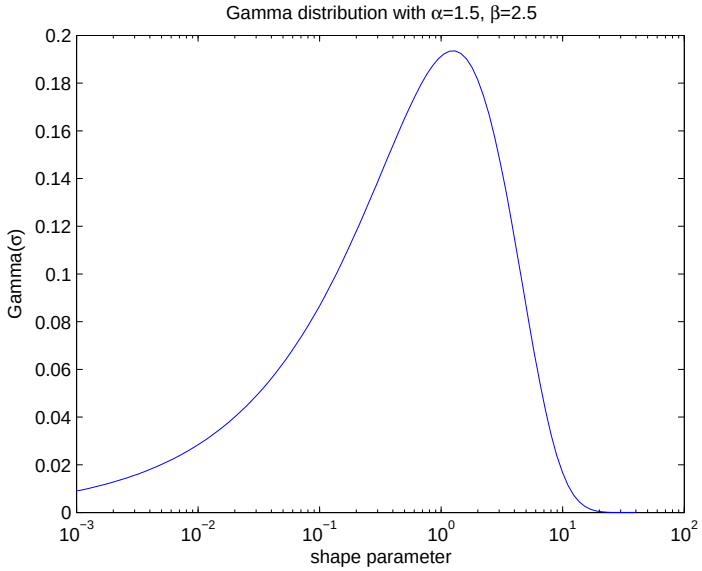

Figure 2: A sample prior distribution with mean 3.75 and mode 1.25

The question then becomes: how do we incorporate the belief that the shape parameters will generally be in the range $(0, 40)$? One choice is to use a Gamma distribution, which allows for values outside that domain but has most of its mass centered around more likely values. See Figure 2 for an example. The distribution can be adjusted if we feel that the design points are on a smaller or larger scale such that different parameters would be appropriate.

The third option listed earlier is the black box prior which loosely speaking involves using a complicated function to compute a $p(\sigma)$ value based on information which cannot be described using a parametric distribution. One example of this might be to evaluate the power function (kriging variance) at various points in the domain and then use a weighted average as the $p(\sigma)$ value. The output of the function need not yield a proper prior, only values whose ratio indicates which choice of σ is preferred.

Because there are many possible choices for black box prior (and they likely require extensive computation) we will focus our work on using

$$p(\sigma) = \Gamma(\sigma; \alpha, \beta) = \frac{1}{\beta^\alpha \Gamma(\alpha)} \sigma^{\alpha-1} e^{-\frac{\sigma}{\beta}}$$

where $\alpha = 1.5$ and $\beta = 2.5$ as depicted in Figure 2.

2 Determining an Appropriate Shape Parameter

We are now able to evaluate (up to a constant) the posterior probability of σ with the function

$$P(\sigma) = (y^T \Phi^{-1} y)^{-n/2} \det(\Phi)^{-1/2} \Gamma(\sigma; \alpha, \beta),$$

where, of course, Φ is the interpolation matrix and is dependent on σ. The process of handling P is threatened by over/underflow, so instead we will evaluate its logarithm which is

$$\tilde{P}(\sigma) = -\log(y^T \Phi^{-1} y) - \frac{1}{n} \log\det(\Phi) + \frac{2}{n} \log p(\sigma). \tag{2.1}$$

2.1 Using MCMC to simulate $p(\sigma|y)$

Our solution is to use Metropolis-Hastings (See [6],[8]) to generate a sequence of random variables $\{\sigma_i\}$ which follow the distribution $p(\sigma|y)$. The proposal distribution will be a normal distribution with mean at the current σ_i. The choice of variance γ^2 is vital because if our variance is too great then the random walk will take too few steps and if it is too small then the steps will be too short to fully explore the domain. This dilemma means that γ^2 will need to be problem dependent.

Because random walk Metropolis-Hastings has a symmetric proposition, the acceptance ratio for a step from σ_i to σ^* reduces to

$$\begin{aligned}\alpha &= \frac{P(\sigma^*)N(\sigma_i;\sigma^*,\gamma^2)}{P(\sigma_i)N(\sigma^*;\sigma_i,\gamma^2)} \\ &= \frac{P(\sigma^*)}{P(\sigma_i)}\end{aligned}$$

This means that at each step of the Markov Chain we will need to evaluate $\tilde{P}$ once and draw one Normal random variable. Drawing a Normal random variable is straightforward, however evaluating $\tilde{P}$ may be problematic depending on the condition of Φ.

2.2 Using Riley's Algorithm to invert Φ

More than 50 years ago, a regularization technique for solving ill-conditioned symmetric positive definite linear systems was developed in [13]. It resurfaced in [12] and Fasshauer recently gave a talk at a colloquium in Syracuse on its efficiency when compared to the SVD. While it does not have an official name, we will call this Riley's Algorithm.

Assuming the goal is to solve the system $Ax = b$ where A is SPD and ill-conditioned, Riley's algorithm solves a regularized system

$$C = A + \mu I \qquad \mu > 0 \tag{2.2}$$

which can be factored safely with the Cholesky decomposition. If all we concerned ourselves with was solving $Cy = b$ this would be ridge regression or Tikhonov regularization. But we can take another step by noting the identity

$$A^{-1} = \frac{1}{\mu}\sum_{k=1}^{\infty}(\mu C^{-1})^k,$$

which gives us a simple iteration method for approximating the solution to $Ax = b$

$$x_{i+1} = y + \mu C^{-1}x_i \tag{2.3}$$

where $y = C^{-1}b$. This technique allows us to find x by only performing a stable Cholesky decomposition on C.

Choosing the regularization parameter μ to maximize stability but minimize the summation length is not a straightforward procedure, but we won't concern ourselves with it here. What we do need to be concerned with is that evaluating $\tilde{P}$ requires both $\Phi^{-1}y$ and $\det(\Phi)$. Unfortunately, using Riley's algorithm for the linear solve does not provide us with a determinant revealing factorization as we would have with a direct solver. In the next section we will analyze a Monte Carlo technique for approximating the log-determinant of Φ.

2.3 Using Orthogonal Draws to approximate the log-determinant

For a matrix whose eigenvalues are within $(-1, 1)$, [1] describes an algorithm which will approximate its log-determinant. Unfortunately, no matrices which are generated by radial basis interpolation satisfy that requirement. It is often the case though that many of the eigenvalues of Φ are satisfactory with only a few delinquent. Matrices of this form can be analyzed using the technique of orthogonal draws described in [9] where the problematic eigenspace is found and then the random vectors are projected away from it.

See Appendix A for a Matlab implementation of orthogonal draws.

3 A Numerical Example

Now that we have set the table, let's look at a specific 1D example of shape parameter selection by MCMC.

Suppose our design points/values are generated by the test function

$$f(x) = 1 + x$$

evaluated at 30 evenly spaced points between $[0, 1]$. If we choose to use truncated power functions as our RBF

$$\phi(r) = (1 - \epsilon r)_+,$$

the spectrum of the resulting matrix is likely to have only a few eigenvalues greater than 1. Note that Φ is positive definite for x in one dimension [4], and sample spectra for $\sigma = 1$ and $\sigma = .01$ are shown in Figure 3. The condition numbers of those matrices are roughly 10^{-3} and 10^{-5} respectively.

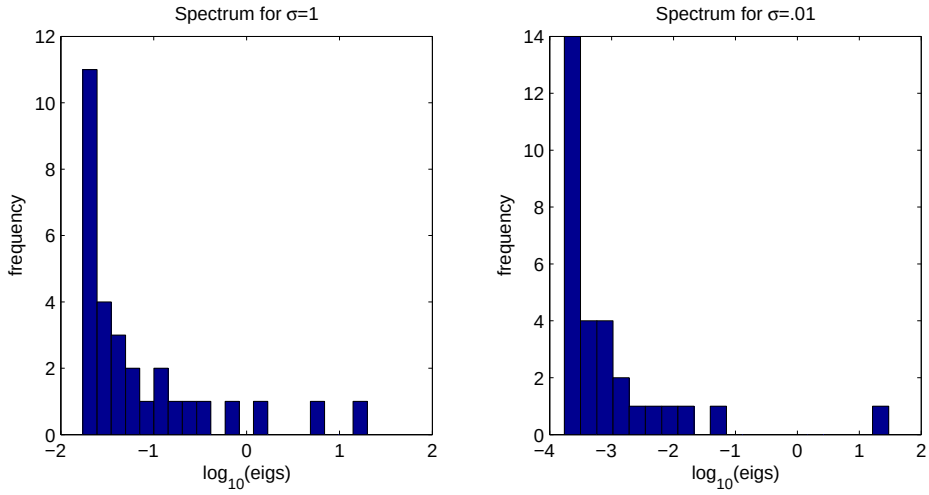

Figure 3: Only a few eigenvalues of Φ need to be handled with orthogonal draws.

Using the techniques described earlier, we can evaluate $\tilde{P}$ for various σ (even though Φ may not be terribly ill-conditioned) and conduct an MCMC simulation to find $p(\sigma|y)$. 1000 MCMC steps were conducted with 100 burn-in and the γ^2 variance in the random walk is chosen arbitrarily to be .1. The resulting distribution can be found in Figure 4.

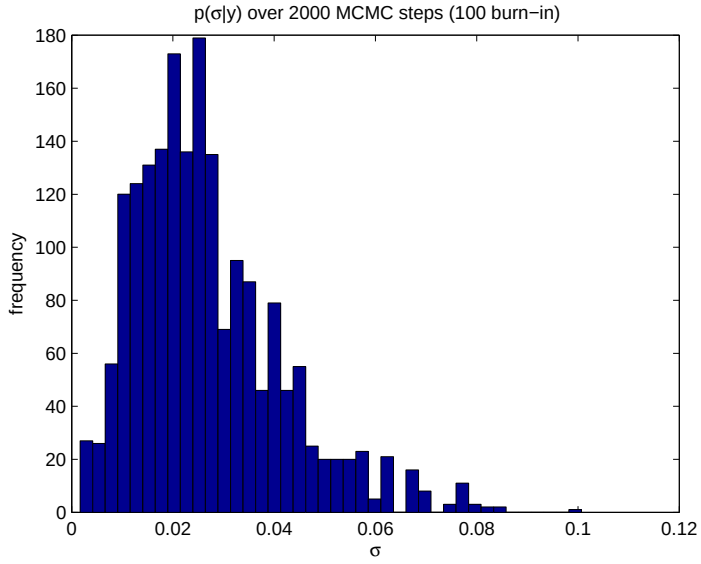

Figure 4: 900 random variable drawn from $p(\sigma|y)$ by the MCMC random walk.

Using this distribution, we can make predictions of new y_0 given an $x_0 \notin (x_i)$. Using the standard Bayesian practice for making predictions and (1.2), our prediction of y_0 is

$$\hat{y}_0 = \frac{1}{N} \sum_{i=1}^{N} \hat{y}_0(\sigma_i),$$

where $\hat{y}_0(\sigma_i)$ is the best linear unbiased predictor for the i^{th} random variable drawn from $p(\sigma|y)$. The mean square error of our sample problem for $\hat{y}_0$ evaluated at 100 evenly spaced points is 6.6947×10^{-15}.

There is one obvious difficulty with this technique of predicting new (x_0, y_0) values: each σ requires inverting the associated Φ. For the sample problem that means inverting 900 more (potentially ill-conditioned) matrices to accompany the 1000 inversions which took place in assembling $p(\sigma|y)$.

Rather than actually finding the best linear unbiased predictor, it is simpler to just approximate the mean of $p(\sigma|y)$ and use that value to make predictions. This is also closer to more standard techniques such as LOOCV and maximum likelihood estimation which produce a unique σ. When we compute the sample mean from our random walk, $\bar{\sigma} = .0267$ and the MS error of $\hat{y}_0(\bar{\sigma})$ for 100 points is 1.2789×10^{-14}. While this is not quite as good as the true estimator, the large reduction in work seems more valuable.

4 Final Thoughts

This project is more of a proof of concept than an actual usable result. While the technique described here for radial basis interpolation is viable, there is as of yet no reason to prefer this method of shape parameter estimation to any other. More work will need to be done to determine when using stochastic simulation is preferable to deterministic techniques.

In looking at the radial basis interpolation problem from a stochastic standpoint, there is an added layer of flexibility now. Rather than simply trying to estimate a good shape parameter, we could also try to find optimal values for the design points or radial basis centers. Even the choice of RBF could be made to be a distribution allowing the model to do even more than in the deterministic case.

The difficulty with this is twofold. First, adding more parameters to the model may result in overfitting; and even if it does not, good priors need to be chosen or the resulting posterior may be troubled. Second, already in this sample problem we discussed, conducting the prediction as it should naturally happen would double the amount of work required to get an answer. Additional complexity in the model would only increase the work further and at some point it would no longer be advantageous to look for more changes.

Unfortunately, each of these topics needs to be considered individually for its merit. It seems logical that stochastic simulation will eventually become an important part of working with radial basis functions, but first we need to work around the numerical difficulties. Already, [10] discusses a technique for making i.i.d. draws from an approximation to $p(\sigma|y)$ which should make the variance in predictions decrease more rapidly. In the past, we have worked on MCMC methods for inferring about good RBF centers and using MCMC to solve ill-conditioned RBF systems - the results were mixed, but promising. Hopefully the statisticians and numerical analysts can continue to work together to solve some of these problems.

References

[1] RP Barry, RK Pace, *Monte Carlo estimates of the log-determinant of sparse matrices* Linear Algebra and Applications, 289:41-54, 1999.

[2] JO Berger, V De Oliveira, B Sans, *Objective Bayesian Analysis of Spatially Correlated Data*, Journal of the American Statistical Association, 96(456):1361-1374, 2001.

[3] LP Bos, U Maier, *On the Asymptotics of Fekete-Type Points for Univariate Radial Basis Interpolation*, Journal of Approximation Theory, 119:252-270, 2002.

[4] G Fasshauer, *Meshfree Approximation Methods in Matlab*, World Scientific Press.

[5] A Gelman, JB Carlin, HS Stern and DB Rubin, *Bayesian Data Analysis*, Chapman & Hall CRC.

[6] WK Hastings, *Monte Carlo Sampling Methods Using Markov Chains and Their Applications*, Biometrika, 57(1):97-109, 1970.

[7] B Kolman, DR Hill, *Elementary Linear Algebra*, Pearson Prentice Hall.

[8] J Liu, *Monte Carlo Strategies in Scientific Computing*, Springer.

[9] M McCourt, *A Stochastic Simulation for Approximating the log-Determinant of a Symmetric Positive Definite Matrix*, http://www.thefutureofmath.com/mystuff/logdet.pdf.

[10] B Nagy, JL Loeppky, WJ Welch, *Fast Bayesian Inference for Gaussian Process Models*, University of British Columbia, Stat. Dept. Tech. Report #230, 2007.

[11] B Nagy, JL Loeppky, WJ Welch, *Correlation parameterization in random function models to improve nomral approximation of the likelihood or posterior*, University of British Columbia, Stat. Dept. Tech. Report #229, 2007.

[12] A. Neumaier, *Solving ill-conditioned and singular linear systems: a tutorial on regularization*, SIAM Review, 40(3):636-666, 1998.

[13] JD Riley, *Solving systems of linear equations with a positive definite, symmetric, but possibly ill-conditioned matrix*, Mathematical Tables and Other Aids to Computation, 9(51):96-101, 1955.

[14] ML Stein, *Interpolation of Spatial Data: Some Theory for Kriging*, Springer.

[15] J Sacks, WJ Welch, TJ Mitchell, HP Wynn, *Design and analysis of computer experiments*, Statistical Science, 4:409-423, 1989.

[16] H Wendland, *Scattered Data Approximation*, Cambridge University Press.

Appendix A

This Matlab function combines all the adaptations of the original function so that we can approximate the determinant of A to a certain tolerance. Note that we use `eigs` to find the disruptive subspace because A has the potential to be sparse for compactly supported RBF. If A is a dense matrix, it will probably be faster to use subspace iteration, but because `eigs` allows us to use a function to represent $A * x$ it will work for both situations. Note also that we do the orthogonal projection separately rather than forming the PDP matrix.

```
function mclogdet = DetAppxVecRBF(A,p,tol)
n = length(A);
v = zeros(1,p);
x = randn(n,p);
xTx = sum(x.*x);
c = x;
[V,E] = FindOuterSpace(A);
lam = E(end);
D = (speye(n)-A)/lam;
oldv = ones(1,p);
for j = 1:m
    c = ProjectPAP(D,V,c);
    v = v-n/j*sum(x.*c)./xTx;
    if abs(mean(v)-mean(oldv))/mean(oldv)<tol
        break
    end
    oldv = v;
end
mclogdet = mean(v)+n*log(lam)+sum(log(E));
end

function [V,outereigs] = FindOuterSpace(A)
n = length(A);
k = floor(sqrt(n));
lam = 2;
V = [];
outereigs = [];
opts.issym = 1;
opts.disp = 0;
while lam>1
    [U,T] = eigs(@(x)ProjectPAP(A,V,x),n,k,'la',opts);
    lam = T(k,k);
    V = [V,U];
    outereigs = [outereigs;diag(T)];
end
end

function y = ProjectPAP(A,V,x)
% This multiplies (I-VV')A(I-VV')x
if V==[]
    y = A*x;
else
    y = A*(x-V*(V'*x));
    y = y-V*(V'*y);
end
```