Radial Basis Functions and Kriging Metamodels for Aerodynamic Optimization

Praveen Chandrashekarappa, Regis Duvigneau

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Radial Basis Functions and Kriging Metamodels for Aerodynamic Optimization

Praveen. C — R. Duvigneau

N° 6151

March 2007

Thème NUM

ISSN 0249-6399 ISRN INRIA/RR--6151--FR+ENG

Radial Basis Functions and Kriging Metamodels for Aerodynamic Optimization

Praveen. C , R. Duvigneau

Thème NUM — Systèmes numériques
Projet OPALE

Rapport de recherche n° 6151 — March 2007 — 40 pages

Abstract: Population-based optimization methods like genetic algorithms and particle swarm optimization are very general and robust but can be costly since they require large number of function evaluations. The costly function evaluations can be replaced by cheaper models which are refered to as *surrogate* or *meta* models*. Here we consider data-fitting models, particularly radial basis functions and kriging. We study the performance of these interpolation models on some analytical functions and aerodynamic data. Both the models have parameters which must be selected carefully to ensure good accuracy. For RBF, we implement a leave-one-out validation technique and for kriging, the parameters are determined by maximizing the probability density of the available data using a particle swarm optimization. The metamodels are then implemented in the shape optimization platform FAMOSA.

Key-words: Metamodels, radial basis functions, kriging, optimization

Project Team Opale: `http://www-sop.inria.fr/opale`

* A metamodel is a model of a model

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles, BP 93, 06902 Sophia Antipolis Cedex (France)
Téléphone : +33 4 92 38 77 77 — Télécopie : +33 4 92 38 77 65

Fonctions à base radiale et Krigeage pour l'optimisation aérodynamique

Résumé : Les méthodes d'optimisation basées sur des populations comme les algorithmes génétiques et l'optimisation par essaim de particules sont très générales et robustes, mais souvent coûteuses car elles nécessitent un nombre important d'évaluations de la fonctionnelle. Ces évaluations coûteuses peuvent être remplacées par des modèles plus économiques connus sous le nom de modèles approchés ou metamodèles[†]. On considère ici des modèles d'approximation, plus précisémemt des fonctions à base radiale (RBF) et des modèle de Krigeage. On étudie la performance de ces modèles pour des fonctions analytiques puis sur des données aérodynamiques. Les deux modèles ont des paramètres qui doivent être choisis prudemment pour assurer une bonne précision. Pour le modèle RBF, on implémente une technique de validation 'leave-one-out' et pour le Krigeage les paramétres sont déterminés en maximisant la densité de probabilité pour les données disponibles en utilisant une optimisation par essaim de particules. Les métamodèles sont ensuite implémentés dans la plateforme FAMOSA.

Mots-clés : Métamodèles, fonctions à base radiale, Krigeage, optimisation

[†] Un métamodèle est un modèle de modèle

1 Introduction

Optimization methods like genetic algorithms, particle swarm optimization, etc. have been found to be ideal for solving large scale problems. Among their many advantages are their ability to handle non-smooth functions (since gradient information is not required) and the possibility of finding global optimal solutions. A distinguishing feature of these methods is that they operate with a *population/swarm*, i.e., they make use of multiple candidate solutions at each step of their iteration. This requires the computation of the *cost/fitness* function for each candidate in every optimization iteration. The ability to locate the global optimum depends on sufficient exploration of the design space which requires using a sufficiently large population size. This is especially true when the cost function is multi-modal and the dimension of the design variable space is high. With the increasing use of high fidelity models, e.g. Navier-Stokes equations for flow analysis, the computation of the cost function for a single design can be costly in terms of time and resource utilization. The combination of such high-fidelity analysis tools with population-based optimization techniques can render them impractical or severely limit the size of the population that can be used.

To overcome this barrier, several researchers have used surrogate models [2, 9, 6, 10, 16] in place of the costly evaluation tool. These surrogate models are inexpensive compared to the exact model. There are several ways in which a surrogate model can be developed.

1.1 Classification of surrogate models

- Data-fitting models: An approximation to the cost function is constructed using available data. This data may be either generated specifically for constructing the model or may be taken from the initial few iterations of the optimization method. Examples of data-fitting models are polynomials (usually quadratic, also known as response surface models), artificial neural networks (like multi-layer perceptron, radial basis function networks) and Gaussian process models (kriging). These models can be either global, which make use of all available data, or local,

which make use of only a small set of data around the point where the function is to be approximated. Global models have been used as a complete replacement of the original cost function with optimization being carried out on the surrogate model. Local models have been typically used as preconditioners to accelerate the exploration of the search space.

- Variable convergence models: The cost function usually depends on the numerical solution of a PDE. Most numerical methods are iterative in nature and contain a stopping criterion which is measured in terms of a solution residual. To get an accurate solution a small value of the residual is usually used. Such an accurate solution maybe unnecessary when all we want is an estimate of a cost function which is usually some integral that converges much faster. In such a situation the stopping criterion can be relaxed thereby considerably reducing the time taken by a single computation.

- Variable resolution models: In these models, a hierarchy of grids is used and the surrogate model is just the costly evaluation tool but run on a coarse grid.

- Variable fidelity models: In these models, a hierarchy of physical models are used, for example Euler equations (surrogate model) and RANS equations (exact model). Even when a high fidelity model like RANS is used, one can use a wall function approximation as a surrogate model and a turbulence model applied upto to the wall as the exact model.

In this report we consider data-fitting models, particularly radial basis functions and gaussian random process models, also known as kriging. Both these methods have been found to be effective in interpolation of high dimensional data with small number of data points as compared to polynomial based methods.

1.2 Interpolation problem

Given a set of N distinct points

Name	$\phi(r)$	Parameters	Smoothness
Gaussian	$\exp(-r^2/a^2)$	$-$	C^∞
Inverse multiquadric	$(r^2+a^2)^{s/2}$	$s<0$	C^∞
Sobolev spline	$r^s K_s(r)$	$s>0$	$C^{\lfloor s \rfloor}$

Table 1: Unconditionally positive definite functions

$$X_N = \{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^d$$

and the values of the function at these locations

$$F_N = [f_1, f_2, \ldots, f_N]^\top$$

we have to infer the value of the function f_{N+1} at a new point x_{N+1}.

2 Radial basis function models

Radial basis functions [3, 19] have been found to be very accurate for interpolation in high dimensions and are ideal for interpolation of scattered data. Due to their spectral convergence property and meshless nature, they are also being used for solving PDEs [11, 12, 22]. Radial basis function interpolation seeks an approximation $\hat{f}$ of the form

$$\hat{f}(x) = \sum_{n=1}^{N} w_n \Phi(x - x_n)$$

where $\Phi(x) = \phi(\|x\|)$ is a radial function. Examples of radial basis functions are given in Table 1. In the RBF terminology, the positions $x_n, n = 1, \ldots, N$ are called the *RBF centers*.

The coefficients $w = [w_1, w_2, \ldots, w_N]^\top$ are determined from the interpolation conditions

$$\hat{f}(x_m) = f_m, \quad m = 1, 2, \ldots, N$$

which can be written in matrix form as[1]

$$Aw = F$$

The matrix A has elements $A_{mn} = \Phi(x_m - x_n)$ and is symmetric since Φ is a radial function. For the functions in Table 1, the matrix A is also *positive-definite* for every set of N distinct points in $\mathbb{R}^d$; this is true for any value of N or d. Such functions are said to be *unconditionally positive definite*. There are radial basis functions which do not have this property; some examples are given in Table 2. In this case, we can make the RBF *conditionally positive definite* by adding a suitable family of polynomials.

$$\hat{f}(x) = \sum_{n=1}^{N} w_n \Phi(x - x_n) + \sum_{l=1}^{M(q)} \alpha_l p_l(x)$$

where $p_l, l = 1, ...M(q)$ forms a basis for $\mathbb{P}_q$, the space of polynomials of degree $\leq q$. The equations to determine the coefficients w and α are

$$\begin{aligned} \sum_{n=1}^{N} w_n \Phi(x_m - x_n) + \sum_{l=1}^{M} \alpha_l p_l(x_m) &= f_m, \quad m = 1, \ldots, N \\ \sum_{n=1}^{N} w_n p_l(x_n) &= 0, \quad l = 1, \ldots, M \end{aligned}$$

The above set of equations is guaranteed to have a unique solution for any disjoint data set. If $N \geq M$ and the unknown function $f \in \mathbb{P}_q$ then the above interpolation will exactly reproduce the function.

Note: The basis functions can be either decreasing (for example the Gaussian) or increasing (for example the thin plate splines) functions, and lead to full

[1]For brevity of notation, the subscript N will be dropped in this section.

Name	$\phi(r)$	Parameters	q
Spline	r^s	$s > 0, s \notin 2\mathbb{N}$	$q \geq \lceil s/2 \rceil$
Thin-plate spline	$r^s \log r$	$s > 0, s \in 2\mathbb{N}$	$q > s/2$
Multi-quadric	$(r^2 + a^2)^{s/2}$	$s > 0, s \notin 2\mathbb{N}$	$q \geq \lceil s/2 \rceil$

Table 2: Conditionally positive definite functions

matrices. There are also basis functions with compact support which lead to sparse coefficient matrices; however these give only algebraic convergence while the smooth basis functions give exponential convergence.

Note: In the present work we consider only interpolating RBFs which exactly reproduce the input data. One can also use fitted RBF models which may not exactly interpolate the data. RBF models can also be considered where the centers do not coincide with the location of the data points.

2.1 Effect of attenuation factor

The attenuation factor in radial basis functions has a critical influence on the accuracy of the interpolation model. We illustrate this with a numerical example. The RBF interpolant is constructed for Test Function 4 (see appendix) using the data from 10 equally spaced points in $[0, 2]$. The error between the interpolant and the exact function is evaluated on a grid of 100 points. Table (3) shows the error and condition number for different attenuation factors. We notice that for both very small and very large values of a the error is high. The condition number of the coefficient matrix A is seen to increase with increasing values of the attenuation factor. The high error at large values of a is due to the numerical instability resulting from ill-conditioning of the matrix A. Note that as long as the condition number is not very high, the interpolant exactly reproduces the training data. The RBF interpolant and the exact function are plotted in Figures 1. Figure 2 shows the variation of average error and condition number with attenuation factor. We notice that the error has a minimum for a particular value of attenuation factor $a_o \approx 0.45$

a	0.01	0.1	0.5	1.0	2.0
Mean error	1.29	0.142	0.0114	0.0978	16.3
Max error	1.24	0.213	0.0181	0.156	19.6
Condition no.	1.0	3.4	1.1×10^9	3.4×10^{14}	2.8×10^{18}

Table 3: Effect of attenuation factor on the error of RBF interpolation for Test 4

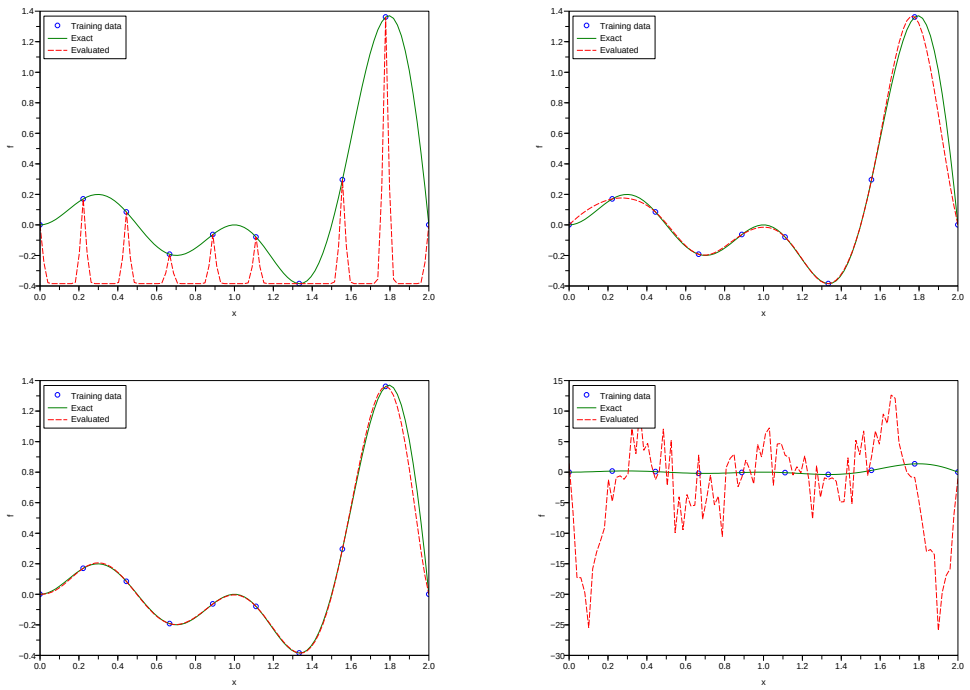


Figure 1: RBF interpolant for Test 4: (a) $a = 0.01$, (b) $a = 0.1$, (c) $a = 0.5$, and (d) $a = 1.0$

with average error of 2.87e-4. A theoretical justification for the existence of an optimal value for the attenuation factor has been recently given in [13].

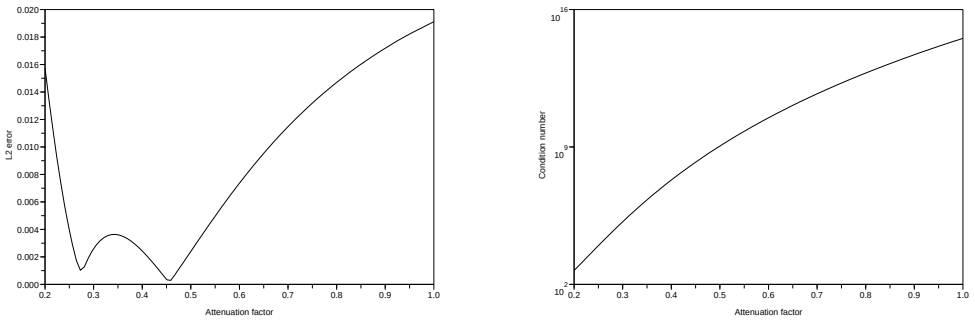


Figure 2: Variation of L_2 error and condition number with attenuation factor

2.2 Optimization of attenuation factor

In [17], several empirical methods for choosing the attenuation factor are discussed; see this paper for further references. Some researchers had expressed the hope that there may be a universally optimal value of the attenuation factor. Based on numerical experiments, Rippa [17] concludes that the best attenuation factor depends on the number and distribution of data points, on the function f and on the precision of the computation[2].

An obvious way to optimize the attenuation factor is to divide the available data into two subsets, a *training set* and a *testing set*; we can use the training set to construct the RBF model and use it to evaluate the function on the testing set. The attenuation factor can be optimized so that the error of interpolation on the testing set is minimized. However, in practical optimization problems, we may not have sufficient number of data points to perform the above sub-division. An alternative approach is the *leave-one-out* technique.

Let $\hat{f}^{(n)}(x;a)$ denote the RBF interpolant constructed using the data points

$$X^{(n)} = \{x_1, x_2, \ldots, x_{n-1}, x_{n+1}, \ldots, x_N\}$$

[2]An interesting result in [4] states that there exists an attenuation factor and location of the RBF centers which leads to an interpolation formula uniformly accurate for all functions in a compact subset of $C(K)$, where K is a compact subset of $\mathbb{R}^d$.

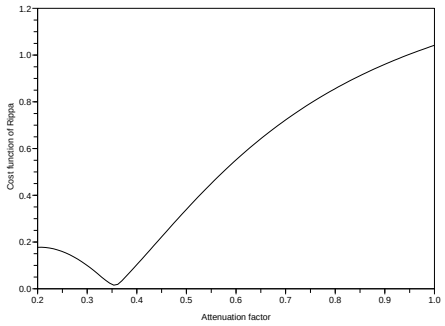


Figure 3: Variation of $C(a)$ with attenuation factor for Test 4

i.e., by ignoring the n'th data point in the full data set. This interpolant can be used to estimate the function value at the ignored point x_n and the corresponding error $E_n = f_n - \hat{f}^{(n)}(x_n; a)$ can also be computed. By ignoring each data point successively and constructing an interpolant we obtain an error vector

$$E(a) = [E_1, E_2, \ldots, E_N]^\top$$

Rippa [17] suggests minimizing some norm of the above error vector with respect to the attenuation factor, i.e., find a_* such that

$$a_* = \text{argmin } \|E(a)\|$$

Rippa gives some numerical examples to show that the function $C(a) = \|E(a)\|$ behaves similar to the actual error. In particular, they achieve their minimum at similar values of attenuation factor. Figure (3) plots $C(a)$ for Test 4 which indicates the existence of an optimum value $a_* \approx 0.353$.

2.3 Efficient implementation

The computation of $C(a)$ requires the solution of N linear equations each of order $(N-1)\times(N-1)$. If the linear system is solved using LU decomposition the total number of operations is of order N^4 which can be very expensive even for moderate size data sets. An efficient algorithm is given in [17] which requires only one LU decomposition at a cost of $O(N^3)$. Below, we essentially reproduce the algorithm as given in [17].

The RBF interpolant using the data points $X^{(n)}$ is given by

$$\hat{f}^{(n)}(x) = \sum_{m=1, m\neq n}^{N} w_m^{(n)} \Phi(x - x_m)$$

where the coefficients $w^{(n)}$ are determined by solving the interpolation problem $\hat{f}^{(n)}(x_r) = f(x_r), r = 1, \ldots, n-1, n+1, \ldots, N$. We denote this in matrix notation as

$$A^{(n)} w^{(n)} = F^{(n)}$$

where $A^{(n)}$ is obtained from A by removing the n'th row and n'th column, and $F^{(n)} = (f_1, \ldots, f_{n-1}, f_{n+1}, \ldots, f_N)^\top$. We note that if $y \in \mathbb{R}^N$ is such that $y_n = 0$ then

$$Ay = z \Longrightarrow A^{(n)}(y_1, \ldots, y_{n-1}, y_{n+1}, \ldots, y_N)^\top = (z_1, \ldots, z_{n-1}, z_{n+1}, \ldots, z_N)^\top \tag{1}$$

Now consider the solution $u^{[n]}$ to the system

$$Au^{[n]} = e^{[n]} \tag{2}$$

where $e^{[n]}$ is the n'th column of the $N \times N$ identity matrix. It is easy to verify that $u_n^{[n]} \neq 0$. Indeed, if $u_n^{[n]} = 0$ then by (1) and (2) we conclude that

$$A^{(n)}(u_1^{[n]}, \ldots, u_{n-1}^{[n]}, u_{k+1}^{[n]}, \ldots, u_N^{[n]})^\top = 0$$

which implies, by the non-singularity of $A^{(n)}$ that $u^{[n]} = 0$, which is impossible because $u^{[n]}$ is the solution to (1). Let us now consider the vector $v^{[n]} \in \mathbb{R}^N$ defined by

$$v^{[n]} = w - \frac{w_n}{u_n^{[n]}} u^{[n]}$$

Then we have that

$$Av^{[n]} = Aw - \frac{w_n}{u_n^{[n]}} Au^{[n]} = F - \frac{w_n}{u_n^{[n]}} e^{[n]} = \left(f_1, \ldots, f_{n-1}, f_n - \frac{w_n}{u_n^{[n]}}, f_{n+1}, \ldots, f_N \right)^\top$$

and since $v_n^{[n]} = 0$, we use (1) to conclude that

$$w^{(n)} = \left(v_1^{[n]}, \ldots, v_{n-1}^{[n]}, v_{n+1}^{[n]}, \ldots, v_N^{[n]} \right)^\top$$

This implies that

$$\begin{aligned}
\hat{f}^{(n)}(x_n) &= \sum_{m=1, m \neq n}^{N} w_m^{(n)} \Phi(x_n - x_m) \\
&= \sum_{m=1, m \neq n}^{N} v_m^{[n]} \Phi(x_n - x_m) \\
&= \sum_{m=1}^{N} v_m^{[n]} \Phi(x_n - x_m) \\
&= \left(A v^{[n]} \right)_n \\
&= f_n - \frac{w_n}{u_n^{[n]}}
\end{aligned}$$

which gives the following simple formula for the error of interpolation at the excluded point x_n

$$E_n = f_n - \hat{f}^{(n)}(x_n) = \frac{w_n}{u_n^{[n]}}$$

If we use LU decomposition to solve the linear equation systems, the cost of one LU decomposition of the matrix A is $O(N^3)$, while the cost of solving the N linear equations (2) is $O(N^2)$ so that the total cost is $O(N^3)$.

Rippa has used Brent's method which is a bracketing algorithm for locating the minimum. In our tests we found that it is not possible to predict in advance a suitable bracketing interval since this depends on the data set, the function and dimension of the problem. Hence we have used Particle Swarm Optimization (PSO) to locate the minimum of the cost function $C(a)$. Since this is a one dimensional minimization problem a small number of particles should be sufficient; we have used five particles in the swarm and tests indicate that the minimum point can be located with less than 100 iterations.

2.4 Some practical issues

The radial basis functions depend on the Euclidean distance between two data points. If the components of the independent variables $x \in \mathbb{R}^d$ have widely different scales then the Euclidean norm may not be appropriate. In [9] a weighted norm has been used in place of the Euclidean norm, where the weights depend on the gradient of the function. Here, the independent variables $\{x_n, n = 1, \ldots, N\}$ and function values $\{f_n, n = 1, \ldots, N\}$ are scaled before constructing the RBF model. The independent variables $x \in \mathbb{R}^d$ are scaled so that each component of x lies in the interval $(-1/2, +1/2)$ while function values are scaled to lie in the interval $(0, 1)$. If the function is constant, then in the scaled space all the function values will be zero and the coefficients w are also zero. A constant function is thus recovered exactly for any value of attenuation factor. Note that this avoids the difficulty of reproducing constant functions with RBF which otherwise requires very flat ($a \to \infty$) basis functions.

The coefficient matrix A can become ill-conditioned for *large* values of attenuation factor, and also for very large and dense data sets. What is a large attenuation factor depends on the number of data points, their distribution

and the dimension d. The best attenuation factor usually leads to a highly ill-conditioned coefficient matrix. An uncertainty principle established in [24] states that the attainable error and the condition number of the RBF interpolation matrix cannot both be small at the same time. When the matrix is highly ill-conditioned, it is not possible to compute the interpolant with finite precision arithmetic since the solution of linear algebraic equations becomes unstable. In [8] a method is proposed to compute the RBF interpolant for such ill-conditioned cases. However this is costly for our present purpose and we use a simple limiting approach. While minimizing the cost function $C(a)$ we compute the condition number of the coefficient matrix A; if it is larger than some specified value, then the cost function is not computed but is set to an arbitrarily large positive number. The particles in PSO are then naturally pulled towards regions of well conditioned attenuation factors. In the present computations, the upper limit on the condition number is set to $1/\epsilon$ where ϵ is the machine precision.

2.5 Numerical examples

The RBF metamodel is applied to some analytical functions and aerodynamic data. The programs for constructing the metamodel are written in Scilab. The linear equations are solved using the functions `lufact` and `lusolve`, while the condition number is computed using `cond`. For the test cases using large datasets as in section 2.5.3 and 2.5.5 the programs are written in Fortran 77 and linear equations are solved using the LINPACK routines `dgeco` and `dgesl`; the routine `dgeco` also computes the condition number.

2.5.1 1-D test functions

The RBF model is constructed for test functions 1-4 by optimizing the attenuation factor using Rippa method; in each case 10 equally spaced data points are used, while the reconstructed function and errors are evaluated on 100 equally spaced points. For Test 4, we study the dependence of error as a function of number of data points for different choices of attenuation factor. In Figure 4 the mean error is plotted for attenuation factors 0.1, 0.5, 1.0, $1/N$

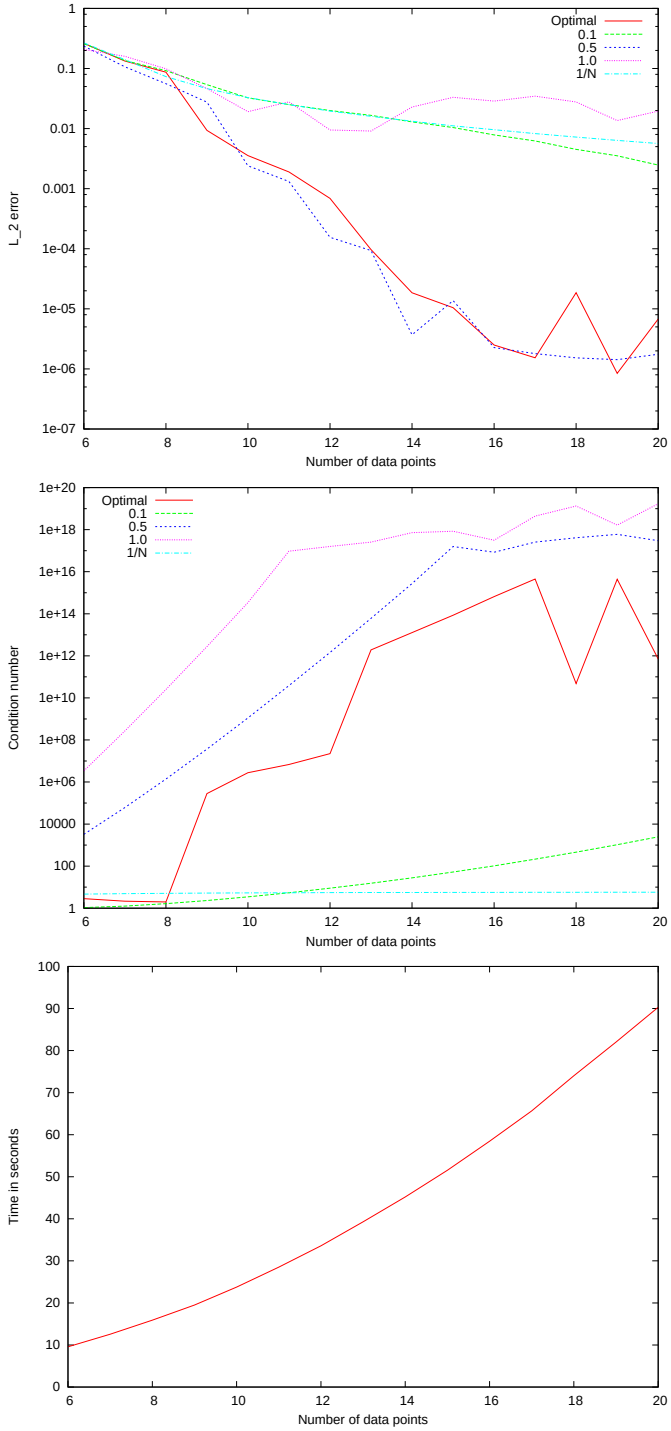


Figure 4: RBF error, condition number and time to construct metamodel (only for Rippa method) versus number of data

Test	a_*	Cond. no.	Mean err	Max. err.	Med. err.
1	1.1553	4.48E+15	2.2156E-08	1.6863E-07	2.8561E-09
2	1.1551	4.49E+15	1.8961E-07	1.3229E-06	3.3238E-08
3	0.2960	1.24E+05	1.5056E-03	1.0669E-02	2.2002E-04
4	0.3563	2.75E+06	3.5499E-03	3.3894E-02	1.2096E-03

Table 4: RBF results for 1-D test functions

and the Rippa method. The value of $1/N$ is equal to the spacing between the data points. We see from the figure that all the attenuation factors perform poorly compared to the Rippa value, except for the value of $a = 0.5$; even the choice of the data spacing $a = 1/N$, is seen to give very high errors. Figure 4 also shows the variation of the condition number and the time taken in the case of Rippa optimization. For 20 data points, the time to construct the RBF model by optimizing the attenuation factor is about 90 seconds. This time is nearly independent of the dimension of the independent space d and depends mostly on the size of the data set N. The optimized attenuation factor, condition number of the linear system and errors for the 1-D test cases are listed in Table 4. For the first two functions which are simple, we see very small approximation errors while for the last two, the errors are higher. The reconstructed function, data points and exact function are plotted in Figure 5.

2.5.2 Rastrigin function in 2-D

The Rastrigin function in 2-D is reconstructed on $[-1, +1] \times [-1, +1]$ using 5×5 to 10×10 data points which are distributed on a uniform grid. The reconstructed function is plotted on a uniform grid of 100×100 points and the errors are also evaluated on this grid. Figure (6) shows the reconstructed and exact function contours along with the location of the data points. For the case of 5×5 data points the accuracy is very poor even qualitatively while for 10×10 points the reconstructed function is qualitatively better. In Table 5, the error for increasing number of data points is given and we notice a monotonic decrease in both the mean and maximum errors.

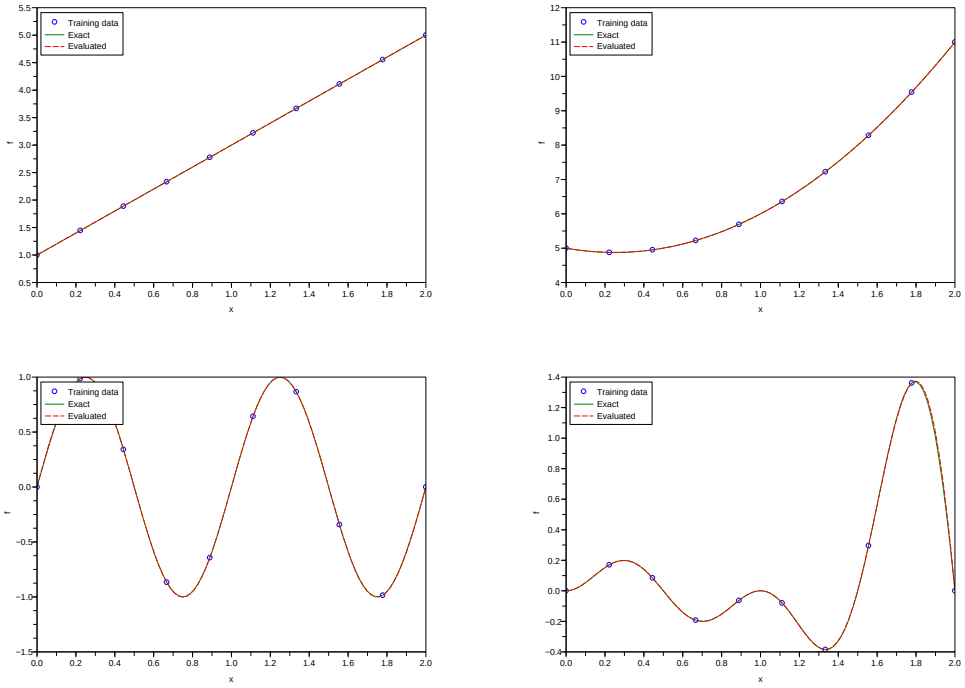
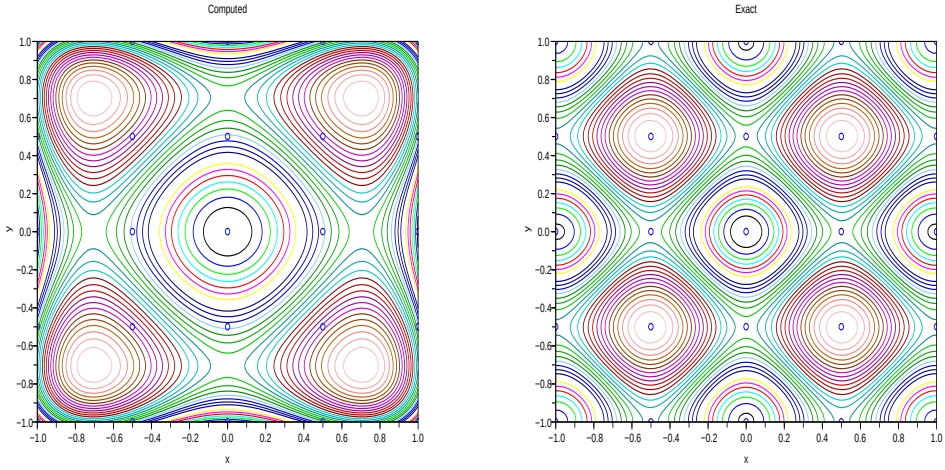


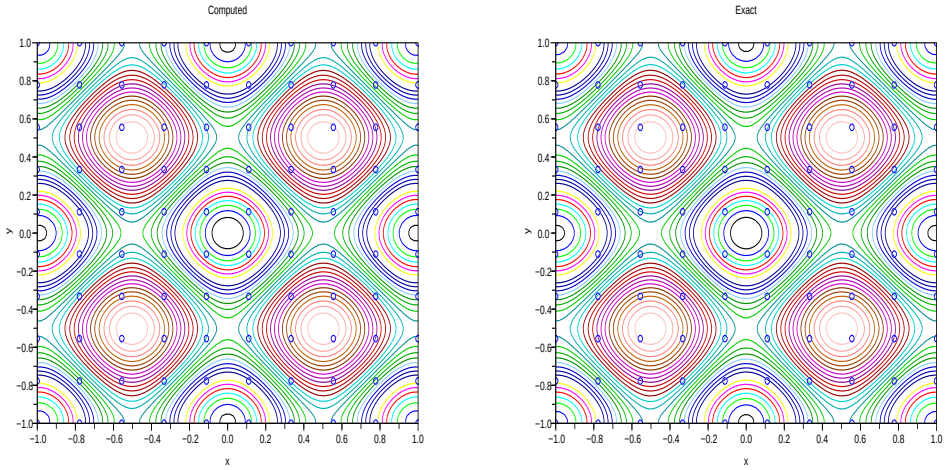
Figure 5: 1-D test functions reconstructed with RBF using 10 data points

N	a	Cond. no.	Mean err	Max err
5×5	2.4212315	4.41e+15	1.073537e+01	3.468763e+01
6×6	1.3368658	4.36e+15	6.192396e+00	1.956525e+01
7×7	0.8935090	4.37e+15	2.208371e+00	1.022841e+01
8×8	0.6578556	4.40e+15	9.937358e-01	4.675421e+00
9×9	0.4236043	7.04e+12	3.324116e-02	2.017713e-01
10×10	0.4166673	1.87e+15	2.318219e-02	1.446371e-01

Table 5: RBF results for 2-D Rastrigin function



5×5 data points



10×10 data points

Figure 6: 2-D Rastrigin function reconstructed in 100×100 grid using RBF with optimized attenuation factor

2.5.3 20-D aerodynamic data

In this test, we take aerodynamic data (drag coefficient C_d and lift coefficient C_l) for a 3-D wing whose shape is parameterized in terms of the displacements of 20 control points of an FFD box. The wing shape is optimized using simplex method so that it minimizes the cost function [1]

$$J = \frac{C_d}{C_{d_o}} + 10^4 \cdot \max\left[1 - \frac{C_l}{C_{l_o}}, 0\right]$$

A set of 500 data points in $\mathbb{R}^{20}$ is generated by running the CFD solver around the optimal point Y^* in the following way: $Y = Y^* + Y_r$, where $Y_r \in \mathbb{R}^{20}$ is a random vector each of whose elements is taken from $\{-1, 0, +1\}$ in a uniform random manner. RBF models are constructed for C_d and C_l using the Rippa method which gives the attenuation factors as 58.25 and 309.45 respectively[3]. In order to test the accuracy of the constructed model, a further set of 100 data points is generated by running the CFD solver; these data points are taken as $Y = Y^* + Y_r$ where the elements of Y_r are uniform random numbers from $(-0.5, +0.5)$. For the drag coefficient, the average and maximum relative errors are 2.78×10^{-5} and 1.10×10^{-4} respectively, while for the lift coefficient they are 1.66×10^{-5} and 6.24×10^{-5} respectively. These errors are very small indicating the approximation power of RBF.

2.5.4 8-D aerodynamic data

This is similar to the previous example but with 8 design variables. A set of 979 points[4] is generated in $\mathbb{R}^8$ using latin hypercube sampling in the range $[Y^* - 50, Y^* + 50]^8$. The optimized attenuation factors for C_d and C_l are found to be 1.2692 and 1.6800 respectively and the condition number of the coefficient matrix in RBF was 0.4652E+08 and 0.1135E+10 respectively. With the optimized attenuation factor, the error of interpolation on a different set of 98 data points was found to be 5.178E-3 and 1.682E-3 respectively. In order

[3] The data was not scaled in this case

[4] Actually 1000 points were generated using latin hypercube sampling but a few of them lead to problems in grid deformation and are not taken for generating the metamodel.

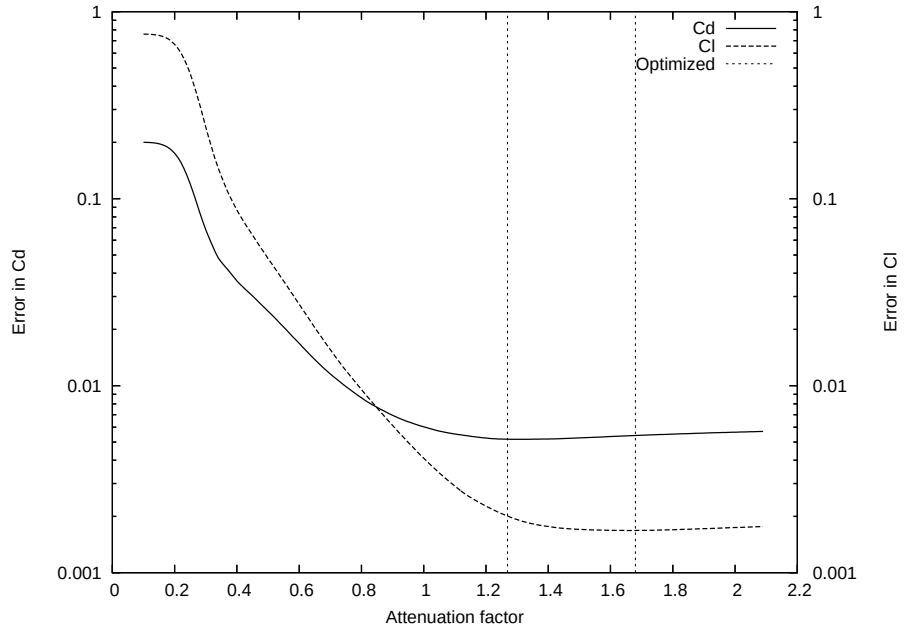


Figure 7: Average error of interpolation as a function of attenuation factor for 8-D aerodynamic data

to see the variation of accuracy of the model with the attenuation factor, we compute the error of interpolation on the test data for different attenuation factors and plot them in figure 7. The two vertical lines represent the optimized attenuation factor. We can see that the minimum error of interpolation occurs at an attenuation factor quite close to the values determined from the Rippa method.

The RBF metamodel was also used to solve a constrained minimization problem[5]:

$$\min C_d \quad \text{such that} \quad C_l = C_{l_0} \quad \text{in} \quad [Y^* - 40, Y^* + 40]^8$$

Starting from a solution obtained with a simplex method ($C_d = 0.01399$, $C_l = 0.31870$), the metamodel yielded the values $C_d = 0.01365$ and $C_l = 0.31870$, while an exact CFD computation gives $C_d = 0.01376$ and $C_l = 0.31847$. This

[5]This problem was solved using FFSQP.

however does not yield a converged solution to the original problem since we performed only a local minimization. A sequence of such local metamodel-based minimizations can be performed to obtained a converged solution. We have implemented such an approach using a trust region strategy which will be presented in a separate report.

2.5.5 20-D quadratic polynomial: gradient and hessian computation

A metamodel can also be used to estimate the gradient and hessian of the unknown function by differentiating the metamodel. We test the accuracy of RBF in estimating the gradient and hessian of a quadratic polynomial in 20-dimensions. A data set of 500 is generated in the same way as in section 2.5.3 but using the test function

$$f(x) = \frac{1}{10}\sum_{i=1}^{20} i x_i^2$$

The RBF model is constructed using two attenuation factors, an empirical formula [15] and the Rippa method. The empirical attenuation factor is given by

$$a = d_{\max}(Nd)^{-1/d}$$

where $d_{\max}$ is the maximum distance between the data points. For the data set used in this test, the empirical formula gives $a = 5.6$ while the Rippa method gives $a = 330.0$. The gradient and hessian are estimated at the origin and are given in Table 6. The exact gradient is zero and the hessian is diagonal with diagonal entries $\lambda_i = i/5$ which are also the eigenvalues of the hessian. We note that with Rippa method, the gradient and hessian are computed with higher accuracy as compared to the empirical formula. This example again demonstrates the importance of the choosing a good value of the attenuation factor in RBF interpolation. A metamodel which gives accurate estimates of gradient can be used to accelerate a genetic or particle swarm method by

Gradient		Hessian	
Empirical	Rippa	Empirical	Rippa
9.680E-02	3.139E-05	1.067872	0.200279
-1.073E-01	-1.951E-05	1.242852	0.400214
2.246E-01	5.434E-05	1.407835	0.600247
2.158E-01	4.978E-05	1.789115	0.800265
-2.464E-01	-5.708E-05	1.915191	1.000399
-4.709E-03	6.331E-06	2.036542	1.200228
-4.795E-02	-9.139E-06	2.325163	1.400180
1.191E-01	3.303E-05	2.736411	1.600317
1.752E-01	4.678E-05	2.786086	1.800187
1.192E-01	3.079E-05	2.916450	2.000256
-2.297E-02	-5.495E-06	3.076823	2.200179
-7.560E-02	-1.192E-05	3.245960	2.400180
8.363E-02	1.025E-05	3.506583	2.600174
-1.552E-01	-4.583E-05	3.702192	2.800150
-1.522E-01	-3.710E-05	3.853749	3.000159
1.327E-01	3.629E-05	4.266155	3.200221
1.344E-01	2.040E-05	4.649631	3.400349
1.472E-01	4.406E-05	4.714389	3.600319
-1.423E-02	8.539E-06	4.996111	3.800309
1.136E-01	2.780E-05	5.337426	4.000272

Table 6: Gradient and hessian for 20-D quadratic polynomial computed from RBF. The exact gradient is zero and the hessian is diagonal with diagonal terms equal to its eigenvalues, $\lambda_i = i/5, i = 1, \ldots, 20$

combining it with a deterministic (gradient-based) method, maybe in a trust-region framework.

3 Gaussian process models

Gaussian process models [14, 18, 23] (also called kriging) treat the response of some experiment as if it were a realization of a stochastic process. In physical experiments, there are independent random errors which make the assumption of a stochastic process plausible. But in a computer experiment, where the response is the output of a computer code which is deterministic, its plausibility is less clear. The output of a computer code also has some high-frequency, low-amplitude errors due to convergence and stopping criteria, limiters, etc., but the general trends are different. In this regard, Torczon et al. state [20]: *we regard the supposition of an underlying stochastic process as nothing more than a convenient fiction. The value of this fiction lies in its power to suggest plausible ways of constructing useful approximations and we will try to avoid invoking it excessively. When we do invoke it, it should be appreciated that optimality criteria such as BLUP and MLE are defined with respect to the fictional stochastic process and should not be invested with more importance than the practical utility of the approximations in which they result.*

In the following, we adopt the Bayesian viewpoint of Gaussian processes [14, 23]. The vector of known function values F_N is assumed to be one realization of a multivariate Gaussian process with joint probability density

$$p(F_N|X_N) = \frac{\exp\left(-\frac{1}{2}F_N^\top C_N^{-1} F_N\right)}{\sqrt{(2\pi)^N \det(C_N)}} \tag{3}$$

where C_N is the $N \times N$ covariance matrix. The element C_{mn} of the covariance matrix gives the correlation between the function values f_m and f_n. This is expressed in terms of a covariance function c, i.e., $C_{mn} = c(x_m, x_n)$. When adding a new point x_{N+1}, the resulting vector of function values F_{N+1} is assumed to be a realization of the $(N+1)$-variable Gaussian process with joint probability density

$$p(F_{N+1}|X_{N+1}) = \frac{\exp\left(-\frac{1}{2}F_{N+1}^\top C_{N+1}^{-1} F_{N+1}\right)}{\sqrt{(2\pi)^{N+1} \det(C_{N+1})}} \tag{4}$$

Using the rule of conditional probabilities

$$p(A|B) = \frac{p(A,B)}{p(B)}$$

we can write the probability density for the unknown function value f_{N+1}, given the data (X_N, F_N) as

$$p(f_{N+1}|X_{N+1}, F_N) = \frac{p(F_{N+1}|X_{N+1})}{p(F_N|X_N)} \tag{5}$$

In order to simplify this expression we notice that the $(N+1)$-variable covariance matrix C_{N+1} can be written as

$$C_{N+1} = \begin{bmatrix} C_N & k \\ k^\top & \kappa \end{bmatrix}$$

where

$$k = [c(x_1, x_{N+1}), c(x_2, x_{N+1}), \ldots, c(x_N, x_{N+1})]^\top$$

and

$$\kappa = c(x_{N+1}, x_{N+1})$$

This gives

$$C_{N+1}^{-1} = \begin{bmatrix} M & m \\ m^\top & \mu \end{bmatrix}$$

where

$$M = C_N^{-1} + \frac{1}{\mu} m m^\top, \quad m = -\mu C_N^{-1} k, \quad \mu = (\kappa - k^\top C_N^{-1} k)^{-1}$$

Then, the probability density for the function value at the new point is

$$p(f_{N+1}|X_{N+1}, F_N) \propto \exp\left[-\frac{(f_{N+1} - \hat{f}_{N+1})^2}{2\sigma^2_{f_{N+1}}}\right]$$

where

$$\hat{f}_{N+1} = k^\top C_N^{-1} F_N, \qquad \sigma^2_{f_{N+1}} = \kappa - k^\top C_N^{-1} k \tag{6}$$

Thus the probability density for the function value at the new point x_{N+1} is also Gaussian with mean $\hat{f}_{N+1}$ and standard deviation $\sigma_{f_{N+1}}$. The availability of the variance of the estimated function value is an important advantage of this method.

Note: The kriging model reproduces the input data exactly as long as the covariance matrix has full rank. To see this, let us predict the function value at $x_{N+1} = x_1$. Note that in this case, the vector k is identical to the first column of C_N. Thus, since C_N is symmetric, we have

$$k^\top C_N^{-1} = [1, 0, \ldots, 0]$$

which gives $\hat{f}_{N+1} = k^\top C_N^{-1} F_N = f_1$.

3.1 Covariance function

The covariance function must reflect the characteristics of the output of the computer code. For a smooth response, a covariance function with sufficient number of derivatives is preferable, whereas an irregular response might require a covariance function with no derivatives. In the absence of any knowledge regarding the unknown function, the most commonly used correlation function is an exponential; in this work we take the correlation function of the form

$$c(x, y) = \theta_1 \exp\left[-\frac{1}{2}\sum_{i=1}^{d} \frac{(x_i - y_i)^2}{r_i^2}\right] + \theta_2$$

where $\Theta = (\theta_1, \theta_2, r_1, r_2, \ldots, r_d)$ are some parameters to be determined. The first term is a distance-dependent correlation between the function values at two data points; if their distance is small compared to the length scales r_i, the exponential term is close to one while it decays exponentially as their distance increases. The parameter θ_1 scales this correlation. In the second term, θ_2 gives an offset of the function values from zero. It is also common to use a single length scale r which reduces the number of parameters to three irrespective of the dimension d of the independent variables. This choice may be sufficient for constructing local kriging models.

3.2 Optimization of hyper-parameters

It now remains to find the parameters Θ in the correlation function. These parameters are determined by maximizing the joint probability density $p(F_N|X_N)$. This is equivalent to minimizing the log-likelihood function given by

$$\mathfrak{L} = F_N^\top C_N^{-1} F_N + \log\det(C_N)$$

This function is known to be multi-modal; hence robust techniques like Genetic Algorithms [5] or Particle Swarm Optimization [21] might be preferable. Many researchers have combined such techniques with a final gradient search to locate the minimum more accurately [2]. In this work we have used PSO technique but without any gradient search.

There are many practical issues that must be taken care of so that all the computations are stable; we follow [2] in this respect. The hyper-parameters must be non-negative; hence it is better to work with the logarithm of the parameters so that they always remain positive. The optimization using PSO is thus applied to the logarithm of the hyper-parameters. The correlation matrix can be ill-conditioned in which case the computation of its inverse will not be accurate. If the condition number is above a specified tolerance, then the log-likelihood is taken to be a large positive value. If the condition number is within the specified tolerance, an LU decomposition of C_N is first performed; then $C_N^{-1}F_N$ and $C_N^{-1}k$ are computed using the LU decomposition. The logarithm of the determinant of C_N is also computed using the LU decomposition

$$\log\det(C_N) = \sum_{n=1}^{N} \log L_{nn} + \sum_{n=1}^{N} \log U_{nn}$$

This avoids the under-flow problem associated with multiplying a large number of small numbers which would be the case if the matrix C_N is badly scaled. As in the case of RBF, an upper limit of $1/\epsilon$ is imposed on the condition number of C_N, where ϵ is the machine precision.

Again following [2] we scale the coordinates and function values as described in section 2.4. For each particle, the parameters in the covariance function are initialized randomly in the intervals as given below:

$$\begin{aligned} \theta_1 &\in [10^{-3}, 1] \\ \theta_2 &\in [10^{-3}, 1] \\ r_i &\in [10^{-2}, 10], \quad i = 1, \ldots, d \end{aligned}$$

In the PSO method, if a particle goes outside this range then its previous position is restored and it's velocity is set to zero. If we do not restrict the range in this manner, then we found that the PSO can converge to a solution which does not give an accurate interpolation. However it is not clear how to choose the range of parameter values. In some of the test cases presented below we notice that the optimal value of the parameters is close to the boundary of the specified range.

3.3 Numerical examples

3.3.1 1-D test functions

We apply the kriging approximation to reconstruct the 1-D test functions given in appendix A. For Test 4, the reconstructed function for 10 and 20 equally spaced data points is shown in Figure 8. In the same figure we also plot a lower estimate of the function $\hat{f} - \sigma$ and an upper estimate $\hat{f} + \sigma$, where σ is the estimated variance which is also given by the kriging model. For 10

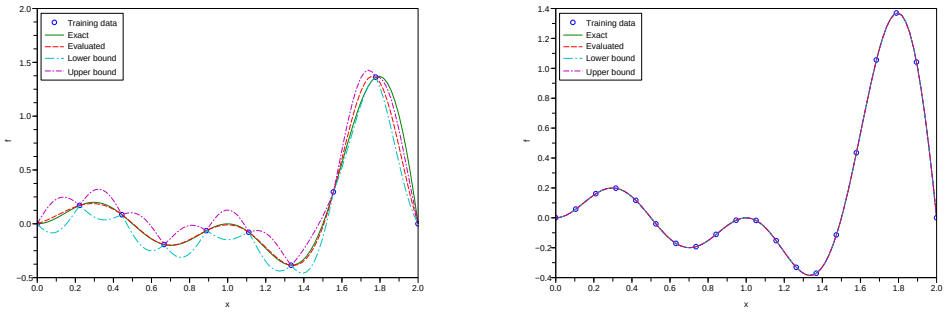


Figure 8: Kriging with 10 and 20 data points

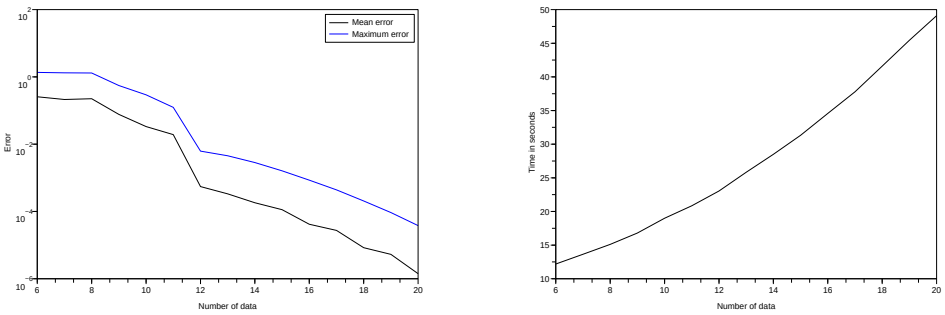


Figure 9: Kriging error and time versus number of data points

data points the function is not well resolved, and we notice that the variance is appreciably large; for 20 data points the function is quite well resolved by the data and the variance is small.

In Figure 9, we plot the variation of the approximation error and time taken to construct the approximation as a function of number of data points. The error is seen to decrease monotonically with increasing number of data points.

Figure 10 shows the reconstructed functions for Test 1-4 using 10 equally spaced data points. The figures also plot the lower and upper estimate of the

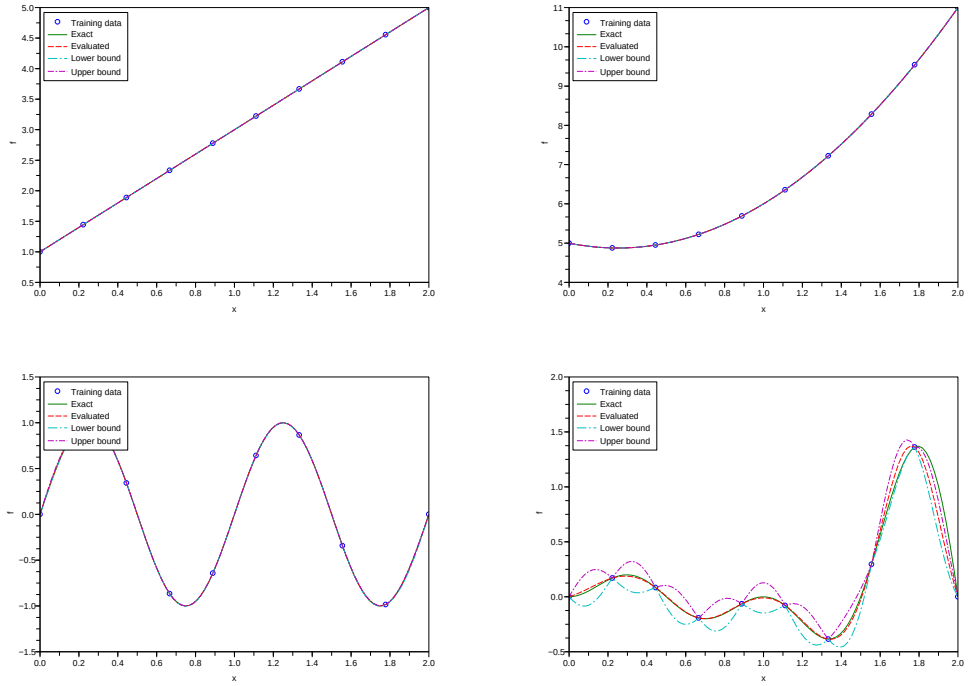


Figure 10: 1-D test functions reconstructed with kriging using 10 data points

functions based on a σ-confidence interval. For Test 1 to 3 the variance is quite small while it is not so for Test 4 which as discussed above is not well resolved by 10 data points. Table 7 lists the parameters of the covariance function and errors obtained for the four test functions. As in the case of RBF, we see excellent approximation of the first two functions but the accuracy degrades considerably for the last two functions.

3.3.2 2-D Rastrigin function

The Rastrigin function in 2-D is reconstructed on $[-1,+1] \times [-1,+1]$ using 5×5 to 10×10 data points which are distributed on a uniform grid. Figure 11 shows the reconstructed and exact function contours along with the location

Test	r	θ_1	θ_2	Mean err	Max. err	Med err
1	0.8136E+0	0.5414E+0	0.8165E-1	0.8275E-7	0.4583E-6	0.1503E-7
2	0.7863E+0	0.9987E+0	0.9931E+0	0.2039E-6	0.1668E-5	0.3344E-7
3	0.2059E+0	0.1000E+1	0.1000E-2	0.1513E-2	0.1091E-1	0.2497E-3
4	0.8013E-1	0.8807E-1	0.5863E-1	0.3327E-1	0.2929E+0	0.9402E-2

Table 7: Kriging results for 1-D test functions

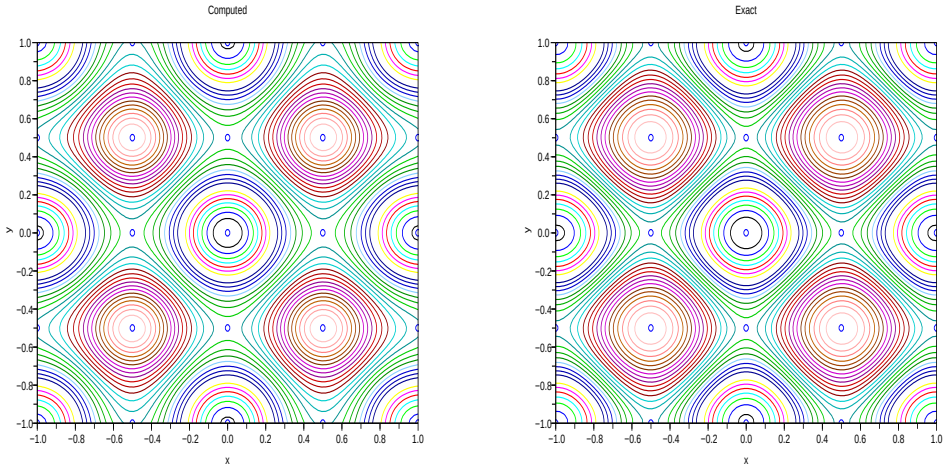
N	r	θ_1	θ_2	Max. var	Mean err	Max err
5×5	0.100	0.118	0.164	12.692	8.738e-1	2.081e+0
6×6	0.336	0.999	0.818	2.726	4.654e+0	1.427e+1
7×7	0.188	0.226	0.178	1.591	7.594e-1	2.882e+0
8×8	0.225	0.416	0.232	2.566	3.069e-1	1.137e+0
9×9	0.264	0.999	0.001	0.154	2.107e-2	1.382e-1
10×10	0.271	0.999	0.997	2.046	2.550e-2	1.623e-1

Table 8: Kriging results for 2-D Rastrigin function

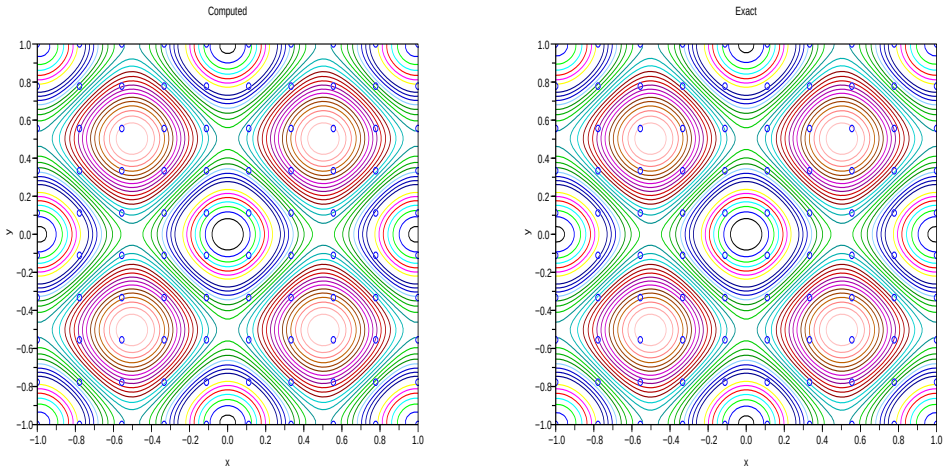
of the data points. In Table 8, the error for increasing number of data points is given and we notice a monotonic decrease in both the mean and maximum errors except for the first and last results. For the case of 5×5 data points the reconstructed function is remarkably accurate atleast qualitatively. However this seems to be only fortuitous; the accuracy is seen to decrease for 6×6 data points as seen in Table 8. For this test, we have used two spatial parameters r_1, r_2 and the minimization of the log-likelihood function gives nearly equal (upto three decimal places) values for them which is to be expected from the symmetry of the function.

3.4 8-D aerodynamic data

This test is identical to section 2.5.4. A single spatial parameter was used in the correlation function. The parameters of the correlation function are given



5×5 data points



10×10 data points

Figure 11: 2-D Rastrigin function using kriging; 5×5 and 10×10 data points.

Quantity	C_d	C_l
r	0.887	1.211
θ_1	4.641×10^{-2}	3.302×10^{-2}
θ_2	0.352	0.277

Table 9: Parameters of correlation function for the 8-D aerodynamic data

Variable	Initial	RBF	Kriging
1	0.162159E+03	0.202000E+03	0.202000E+03
2	-0.156146E+03	-0.176179E+03	-0.179367E+03
3	-0.225295E+03	-0.232854E+03	-0.232523E+03
4	0.126721E+03	0.139730E+03	0.142176E+03
5	0.129458E+03	0.135667E+03	0.136099E+03
6	-0.171660E+03	-0.210900E+03	-0.210900E+03
7	-0.185184E+03	-0.191681E+03	-0.192461E+03
8	0.268190E+03	0.299299E+03	0.300057E+03

Table 10: Metamodel-based shape optimization for 8-D aerodynamic problem

in table 9. The error of interpolation on the test data was found to be 5.331E-3 and 1.697E-3 for C_d and C_l respectively.

The kriging model was also minimized as in section 2.5.4 and yielded nearly identical answers. Table 10 shows the 8 design variables obtained after minimizing the RBF and kriging models.

4 Comparison of RBF and Kriging

Table 11 compares the approximation error for 1-D test functions obtained from RBF and Kriging. For Tests 1-3, the errors of the two methods are comparable, though RBF performs slightly better. For Test 4, RBF performs an order of magnitude better than kriging. Comparing the time taken to construct the metamodels from Figures 4 and 9, we see that kriging is about

Test	Mean error		Max. error	
	RBF	Kriging	RBF	Kriging
1	2.215E-8	8.275E-8	1.686E-7	4.583E-7
2	1.896E-7	2.039E-7	1.323E-6	1.668E-6
3	1.505E-3	1.513E-3	1.067E-2	1.091E-2
4	3.549E-3	3.327E-2	3.389E-2	2.929E-1

Table 11: Comparison of RBF and Kriging for 1-D test functions

N	Mean error		Max. error	
	RBF	Kriging	RBF	Kriging
5×5	1.073E+1	8.738E-1	3.368E+1	2.081E+0
6×6	6.192E+0	4.654E+0	1.956E+1	1.427E+1
7×7	2.208E+0	7.594E-1	1.022E+1	2.882E+0
8×8	9.937E-1	3.069E-1	4.675E+0	1.137E+0
9×9	3.324E-2	2.107E-2	2.017E-1	1.382E-1
10×10	2.318E-2	2.550E-2	1.446E-1	1.623E-1

Table 12: Approximation errors for 2-D Rastrigin function

twice as fast as RBF. However, the absolute times are small enough to make this distinction irrelevant, especially when the programs are written in Fortran.

The approximation errors for the 2-D Rastrigin function as shown in Table 12. In this example, Kriging is seen to perform better than RBF. In Figure 12, the reconstructed function with 7×7 data points is shown for both RBF and Kriging. We notice that RBF has poor accuracy near boundaries [7], which is probably the reason for its poor approximation property, while Kriging seems to perform much better everywhere.

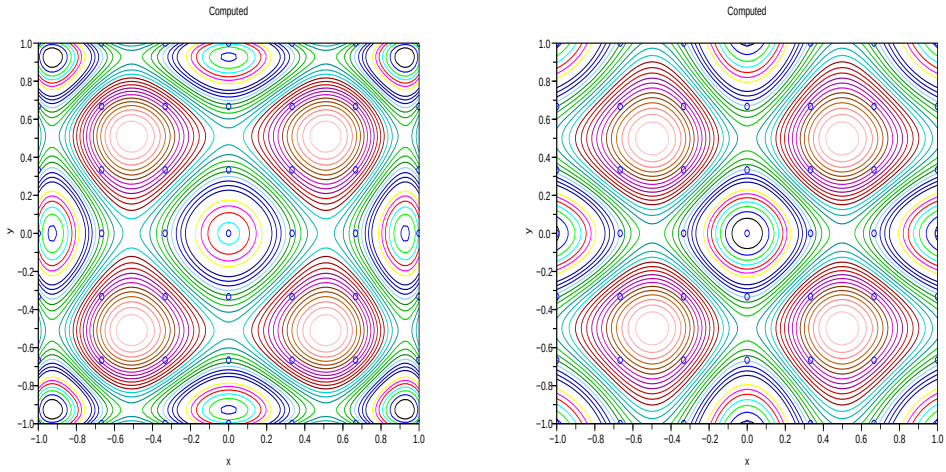


Figure 12: RBF (left) and Kriging (right) approximations using 7×7 data points

5 Summary

We have tested radial basis function and kriging for interpolation of functions given a discrete set of data. The attenuation factor in radial basis function method is seen to have considerable impact on the accuracy. A validation method to optimize the attenuation factor is used and found to give better accuracy compared to choosing it in an empirical manner. The parameters in the correlation function for kriging is determined by maximizing the probability density of the given data. For an 8-dimensional aerodynamic problem, both RBF and kriging methods yielded similar results. However kriging also provides an estimate of the covariance which can be useful to determine the accuracy of the model. Both the methods have been implemented in the shape optimization platform FAMOSA and promising results have been obtained. Metamodels have also been used in a trust-region framework and will be reported in a forthcoming publication.

Acknowledgements

This work has been supported by ANR in the framework of RNTL 2005 program.

A Test functions

1. Linear function in 1-D

$$f(x) = 2x + 1, \quad x \in [0, 2]$$

2. Quadratic function in 1-D

$$f(x) = x^2 + x + 1, \quad x \in [0, 2]$$

3. Sine function in 1-D

$$f(x) = \sin(2\pi x), \quad x \in [0, 2]$$

4. Quadratic-sine function in 1-D

$$f(x) = x(1 - x)\sin(2\pi x), \quad x \in [0, 2]$$

5. d-dimensional Rastrigin function

$$f(x) = 10d + \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i)\right]$$

References

[1] B. Abou El Majd, R. Duvigneau and J.-A. Désidéri, "Aerodynamic Shape Optimization using a Full and Adaptive Multilevel Algorithm", ERCOFTAC Conference Design Optimization : Methods and Applications, Canary Island, Spain, April 2006.

[2] Dirk Büche, Nicol N. Schraudolph and Petros Koumoutsakos, "Accelerating evolutionary algorithms with Gaussian process fitness function models", IEEE Tran. on Systems, Man, and Cybernetics - Part C: Applications and Reviews, Vol. 35, No. 2, May 2005.

[3] M. D. Buhmann, "Radial basis functions", *Acta Numerica*, Vol. 9, pp. 1-38, 2000.

[4] Tianping Chen and Robert Chen, "Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks", *IEEE Transactions on Neural Networks*, Vol. 6, Issue 4, 1995.

[5] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*, John Wiley and Sons, 2001.

[6] Michael Emmerich, Kyriakos Giannakoglou and Boris Naujoks, "Single- and multi-objective evolutionary optimization assisted by Gaussian random field metamodels", *IEEE trans. evol. comput.*, Vol. 10, No. 4, pp. 421-439, 2006.

[7] B. Fornberg, T. A. Driscoll, G. Wright and R. Charles, "Observations on the behavior of radial basis function approximations near boundaries", *Comp. Math. with App.* Vol. 43, Issues 3-5, Pages 473-490, 2002.

[8] B. Fornberg and G. Wright, "Stable computation of multi-quadric interpolants for all values of the shape parameter", *Computers and Mathematics with Applications*, Vol. 48, pp. 853-867, 2004.

[9] K. C. Giannakoglou, "Design of optimal aerodynamic shapes using stochastic optimization methods and computational intelligence", *Prog. Aero. Sci.*, Vol. 38, pp. 43-76, 2002.

[10] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation", in *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, Vol. 9, No. 1, Springer, 2005.

[11] E. Kansa, "Motivation for using radial basis functions to solve PDEs", available on the internet at `http://rbf-pde.uah.edu/kansaweb.pdf`

[12] E. Larsson and B. Fornberg, "A numerical study of some radial basis function based solution methods for elliptic PDEs", *Computers and Mathematics with Applications* Vol. 46, No. 5-6, pp. 891-902. Sept. 2003.

[13] E. Larsson and B. Fornberg, "Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions", *Computers and Mathematics with Applications*, Vol. 49, pp. 103-130, 2005.

[14] David J. C. Mackay, "Gaussian Processes: A replacement for supervised neural networks ?",

[15] H. Nakayama, M. Arakawa and R. Sasaki, "A computational intelligence approach to optimization with unknown objective functions", *Artif. Neural Netw.*, pp. 73-80, 2001.

[16] Y. S. Ong, P. B. Nair, A. J. Keane and K. W. Wong, "Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems", *Knowledge Incorporation in Evolutionary Computation*, editor: Y. Jin, Studies in Fuzziness and Soft Computing Series, pp. 307-331, Springer Verlag, 2004.

[17] Schmuel Rippa, "An algorithm for selecting a good value for the parameter c in radial basis function interpolation", *Adv. Comp. Math.*, Vol. 11, 1999.

[18] J. Sacks, W. J. Welch, T. J. Mitchell and H. P. Wynn, "Design and analysis of computer experiments", *Statistical Science*, Vol. 4, No. 4, pp. 409-435, 1989.

[19] R. Schaback, "Reconstruction of Multivariate Functions From Scattered Data", available on the internet at `http://www.num.math.uni-goettingen.de/schaback/teaching/texte/rbfbook.ps`

[20] V. Torczon and M. W. Trosset, "Using approximations to accelerate engineering design optimization", ISSMO/NASA First Internet Conference on Approximations and Fast Reanalysis in Engineering Optimization, June 14-27, 1998.

[21] G. Venter and J. S. Sobieski, "Particle swarm optimization", *AIAA J.*, Vol. 41, No. 8, 2003.

[22] H. Wendland, "Meshless Galerkin Methods Using Radial Basis Functions", *Mathematics of Computation*, Vol. 68, No. 228, pp. 1521-1531, 1999.

[23] C. K. I. Williams, "Prediction with Gaussian processes: From linear regression to linear prediction and beyond", Technical Report NCRG/97/012, Neural Computing and Research Group, Dept. of Comp. Sci. and App. Math., Aston University, Birmingham, 1997.

[24] Z. M. Wu and R. Schaback, "Local error estimates for radial basis function interpolation of scattered data", *IMA J. Num. Anal.*, Vol. 13, No. 1, pp. 13-27, 1993.

Contents

Unité de recherche INRIA Sophia Antipolis
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399