Comparing different interpolation methods on two-dimensional test functions

Thomas Mühlenstädt, Sonja Kuhnt

May 28, 2009

Keywords: Interpolation, computer experiment, Kriging, Kernel interpolation, Thin plate spline, Natural neighbor interpolation.

Introduction

In the context of computer experiments a common way of dealing with expensive simulation models is to evaluate the simulation at a number of well designed input values and to replace the simulation by an easy to calculate surrogate model. Then the surrogate model is used for optimization, sensitivity analysis or other applications at hand. The surrogate model has to interpolate the observations as there is usually no random error for computer experiments. There exist different multivariate interpolation methods which can be applied as surrogate model. The aim here is to provide a comparison of four of these: Kriging, Kernel interpolation, Natural neighbor interpolation and Thin plate splines.

The article is organized as follows: After a short review of the four interpolation methods, in Section 2 details of the chosen designs are explained and the interpolation methods are compared to each other in Section 3. A summary concludes our article.

1 Interpolation methods

Let $\vec{x}_1, \dots, \vec{x}_n$ denote the design D of the input variables with $\vec{x}_i = (x_{i,1}, x_{i,2})'$ and let $y_1, \dots, y_n$ be observations of a one dimensional output variable.

Kriging is the standard approach to model the output of non-random computer experiments [Fang et al. (2006)]. The assumed model is given by

$$Y(\vec{x}_i) = g_\beta(\vec{x}_i) + Z(\vec{x}_i), 1 \le i \le n, \tag{1}$$

with $Z = (Z(\vec{x}_1), \ldots, Z(\vec{x}_n))$ being a normally distributed random vector with zero mean and covariance matrix $\sigma^2\Gamma, \sigma^2 > 0$. In this paper $g_\beta(\vec{x})$ is assumed to be constant over the design space: $g_\beta(\vec{x}) = \beta \in \mathbb{R}$. The correlation matrix depends on the distance between two input points $\vec{x}_1$ and $\vec{x}_2$. In order to interpolate the data, the correlation between two observations must equal 1 if $\vec{x}_1 = \vec{x}_2$. We use the following correlation function:

$$cor(Z(\vec{x}_i), Z(\vec{x}_{i'})) = \exp\left(-\sum_{d=1}^{2} \theta_i \left|x_{i,d} - x_{i',d}\right|^2\right). \tag{2}$$

The parameter vector $\vec{\theta}$ needs to be estimated from the data, which is done by REML estimation. The loglikelihood is optimized in a similar manner to the algorithm described by Fang et al. (2006), p. 149. The optimization is repeated for a number of different initial values. Kriging is applied in many practical situations and is well known for it's high prediction accuracy.

As an alternative to Kriging [Mühlenstädt and Kuhnt (2009)] just recently suggested *Kernel interpolation(KI)*. First the Delaunay triangulation of the design is determined. For every simplex S_j in the triangulation, a linear function $\hat{y}_j(\vec{x})$ is fitted to the vertices of the simplex. Locally these give reasonable fits to the data. In order to obtain one global fit $\hat{y}(\vec{x})$, the local fits are combined by a weighted average:

$$\hat{y}(\vec{x}) := \begin{cases} y_i, & \vec{x} = \vec{x}_i, i = 1, \ldots, n; \\ \frac{\sum_{j=1}^{N} g_j(\vec{x})\hat{y}_j(\vec{x})}{\sum_{j'=1}^{N} g_{j'}(\vec{x})}, & \text{elsewhere}, \end{cases} \tag{3}$$

with N being the number of simplices. Here we use the weight function

$$g_j(\vec{x}) = \frac{1}{(\prod_{i=0}^{2} \|\vec{x} - \vec{x}_i^j\|_2)^2}, \tag{4}$$

where $\vec{x}_i^j$ are the vertices of simplex S_j. The function $\hat{y}(\vec{x})$ is continuous and differentiable (see Mühlenstädt and Kuhnt (2009)). This new approach is combining two methods of interpolation: Piecewise linear interpolation and inverse distance weighting, which are both not very competitive on their own.

Thin plate splines (TPS) [Micula (2002)] are a multivariate extension of natural cubic splines. A Thin plate spline is a function f^* minimizing

$$I(f) = \int_{\mathbb{R}^2} \left(\frac{\partial^2 f(\vec{x})}{\partial x_1^2}\right)^2 + 2\left(\frac{\partial^2 f(\vec{x})}{\partial x_1 \partial x_2}\right)^2 + \left(\frac{\partial^2 f(\vec{x})}{\partial x_2^2}\right)^2 d\vec{x} \tag{5}$$

in a suitable functional space under the constraint of interpolation. The solution is a special case of interpolation using radial basic functions:

$$f^*(\vec{x}) = \sum_{i=1}^{n} \lambda_i \phi(\|\vec{x} - \vec{x}_i\|_2) + \lambda_{n+1} + \lambda_{n+2}x_1 + \lambda_{n+3}x_2, \tag{6}$$

with $\phi(r) = r^2 log(r)$. The parameters $\lambda_1, \ldots, \lambda_{n+3}$ can be calculated by a system of linear equations.

Natural neighbor interpolation (NNI) was proposed by [Sibson (1980)] and is predicting by a weighted average of the observations. The i-th weight for the prediction at $\vec{x}$ is determined by the (scaled) Lebesgue measure of the intersection of the Voronoi cell of $\vec{x}_i$ w.r.t. $\vec{x}_1, \ldots, \vec{x}_n$(i.e. the data without $\vec{x}$) and the Voronoi cell of $\vec{x}$ w.r.t. $\vec{x}$ and $\vec{x}_1, \ldots, \vec{x}_n$.The points $\vec{x}_i$ which have a non-zero weight are called the natural neighbors of $\vec{x}$. We include natural neighbor interpolation in our study as it is a strictly local method.

2 Designs

We employ three different kinds of designs to evaluate whether or not the performance of the interpolation methods depend on the data structure. As an example of a well designed computer experiment we consider maximin latin hypercube designs according to Morris and Mitchell (1995). The corresponding optimality criterion for this kind of design is

$$\Phi_{25}(D) = \left(\sum_{i=1}^{n} \sum_{j=i+1}^{n} \frac{1}{\|\vec{x}_i - \vec{x}_j\|_2^{25}} \right)^{1/25} . \tag{7}$$

Simulated annealing is applied for searching designs which minimize this criterion. Designs with sample sizes $n = 10, 20, 30$ are constructed.
Although regular grids are not considered to be first choice for computer experiments due to their poor projection properties, it might well be that the data at hand come from a full factorial design as full factorial designs are often D-optimal designs w.r.t. higher order models. Here full factorials with sample size $n = 9, 16, 25, 36$ are considered.
For the last class of designs a random latin hypercube design is constructed and some additional two-dimensional beta-distributed observations are added to the design. The beta distributions are constructed such that there are clusters in the input design. This may reflect two different situations: Firstly the situation of a random sample of design points. Secondly this can be the result of an optimization process where first a space filling design has been chosen and afterwards promising areas of the design space are analyzed by additional design points. All designs are plotted in Figure 1.

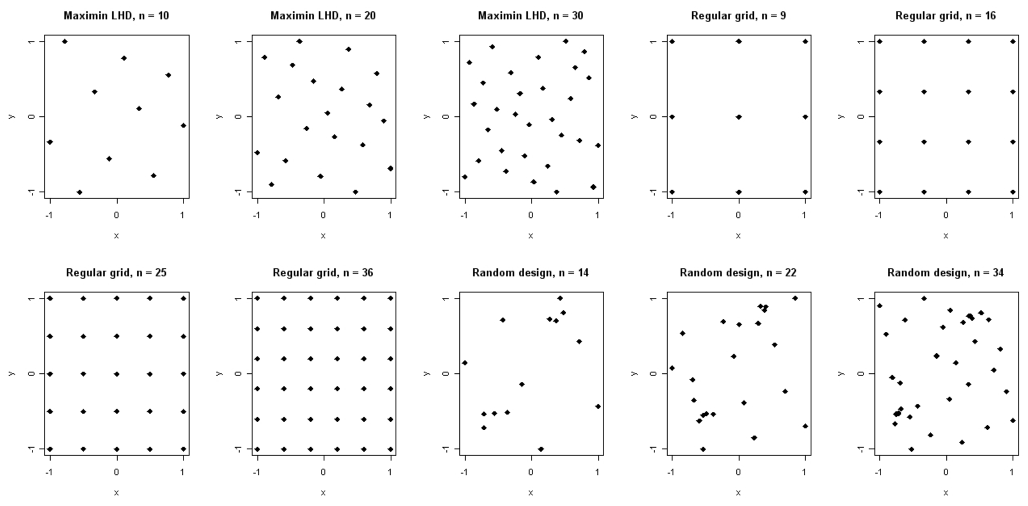

Figure 1: Designs used for comparison: Maximin latin hypercube, full factorial, random design with clusters.

3 Comparison

In order to judge the fit of an interpolator, the root mean square error is used:

$$RMSE(y, \hat{y}, \vec{r}_1, \ldots, \vec{r}_n) := \sqrt{\sum_{i=1}^{n} (y(\vec{r}_i) - \hat{y}(\vec{r}_i))^2}. \tag{8}$$

We always used 10000 points of a Sobol' uniform sequence for calculating the RMSE. In order to have comparable RMSE values, all functions $f(\vec{x})$ have been scaled such that $\min f = 0$ and $\max f = 1$. The most simple method to implement is the *TPS*, as it mainly consists of solving a system of linear equations of order $n + 3$. The implementation of Kriging and *KI* are of comparable complexity if an optimization algorithm (for Kriging) and an algorithm for the computation of the Delaunay triangulation (for *KI*) are available. Although *NNI* is conceptually simple it is by far the most complicated method to implement, as implementing the Voronoi diagram efficiently is demanding. The computation times for 10000 points of the Sobol' sequence for each interpolation method based on the three Maximin designs are shown in Table 1. For the other designs the computation times are comparable. *NNI* is by far the most expensive surrogate in terms of computation time. For Kriging, the computation times differ depending on the example considered as the optimization of the restricted log-likelihood strongly depends on the sample data set. The optimization has been repeated

50 times with different initial values for each data set. Once the correlation parameters are estimated, the Kriging prediction is very fast.

Design	*TPS*	*KI*	Kriging	*NNI*
D_{10}^{Mm}	10	15	13 - 22	839
D_{20}^{Mm}	13	27	69 - 162	1536
D_{30}^{Mm}	15	39	178 - 448	2239

Table 1: Computation times for the prediction of 10000 points based on the Maximin latin hypercube designs in seconds.

The first considered function comes from one of our mechanical engineering projects. The output represents the *die force* in sheet metal forming which depends on friction and blankholder force:

$$f(x,y) = 0.9996(1090.91 + 4xy)\exp\left(x\frac{\pi}{2}\right), \quad D = [0.05, 0.2] \times [5, 30]. \quad (9)$$

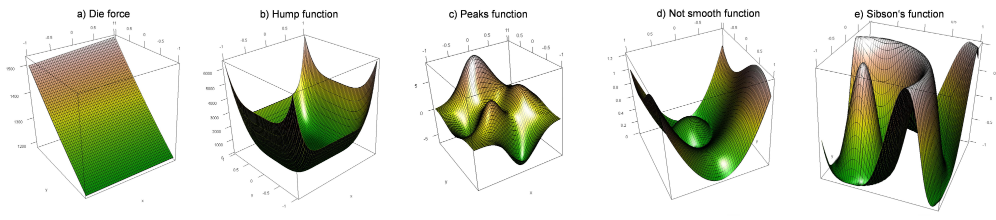

Figure 2: Contour plots.

This function is mainly a linear function in one variable, as can be seen by its contour plot in Figure 2. It is included, as a minimum requirement to an interpolation method is, that it is very precisely reproducing simple functions. The RMSE results are shown in Figure 3. Here *TPS* and *KI* delivered comparable results for all designs. Kriging is delivering by far the best fit (numerically a perfect fit) except for the 3^2 design, for which the optimization of the likelihood delivered relatively large correlation parameters. For all three kinds of designs, the RMSE for *NNI* was magnitudes away from the results of the other methods, showing that the *NNI* fit has been very poor. A function often used as example in optimization is the *Hump* function [Wang and Chen (1996)]:

$$f(x,y) = 1.0316 + 4x^2 - 2.1x^4 + \frac{1}{3}x^6 + xy - 4y^2 + 4y^4, \; D = [-5,5]^2 \quad (10)$$

Here (and for all remaining functions) there is not one method which outperforms the other methods by magnitudes. For small sample sizes, *TPS* and *KI* deliver the best results, while Kriging performs better for larger sample sizes. *NNI* is the worst method w.r.t. the RMSE for most of the designs for the Hump function.
The Matlab's `peaks` function is an example of a very 'hilly' contour with several local optima:

$$f(x,y) = 3(1-x)^2 \exp(-x^2-(y+1)^2) - 10(\frac{x}{5} - x^3 - y^5)\exp(-x^2-y^2) - \frac{1}{3}\exp(-(x+1)^2-y^2),\ D = [-2,2]^2 \quad (11)$$

Here Kriging performs best for the designs with higher sample sizes and for the random design. For the smaller sample sizes of the full factorial and the Maximin LHD, the performance of the methods vary: For the smaller sample size, Kriging performed comparable (full factorial) or worse (Maximin LHD) than *KI* and *TPS*. For higher sample sizes, Kriging is better than KI and TPS, whereas NNI again delivers the poorest fit.

Most of the examples in literature are smooth functions. However, this does not necessarily need to be the case. Therefor we include a function in this simulation study, which is continuous but not differentiable:

$$f(x,y) = |x^2 + \sin(0.5\pi y) - y|, \qquad D = [0,1]^2. \quad (12)$$

None of the methods fail completely. As for the other examples *KI* and *TPS* perform comparable most of the times and *NNI* performs poor compared to the best method. Kriging performs good except for the Maximin LHD with $n = 20$ and for the 3^2 design. For the random designs with clusters, the *TPS* achieves the best RMSE value for all 3 sample sizes.
Sibson (1980) uses the following function in order to demonstrate *NNI*:

$$f(x,y) = cos(4\pi\sqrt{(x-0.25)^2+(y-0.25)^2}), \quad D = [0,1]^2 \quad (13)$$

This function has a complexity, which is too high for the smaller sample sizes. This function is included, as it is realistic that the computer experiment might have a higher complexity than expected. In such a situation the fit is more like an approximation than an interpolation, which is reflected by the fact, that the RMSE is very high compared to the other examples. Here Kriging gives an inconsistent result: For the Maximin LHD, it has the poorest RMSE value despite of the largest sample size, whereas for the random design, it performs best for all sample sizes. Sibson's function is the only one, where *NNI* performed best w.r.t. the RMSE for several designs.

4 Summary

In this paper we compared well-known and new interpolation methods based on the simulations. Several conclusions can be drawn: There is not one overall winner for every situation. Kriging often performs very well, although it may fail for small sample sizes. Thin plate splines and Kernel interpolation most of the times perform comparable and, especially for small sample sizes, even better than Kriging. For most of the examples, *NNI* has delivered the poorest fit w.r.t. the RMSE and is at the same time computationally intensive. Hence we recommend not to use *NNI*. Maximin designs give the best fit among all considered designs, which confirms considering Maximin LHD as a good way to design computer experiments. For the random design with clusters, Kriging seems to be the method which performs best under this kind of data. All in all, for small sample sizes, it seems to be reasonable to use *KI* or *TPS*, whereas for higher sample size, Kriging is the best choice.

Acknowledgement

The financial support of the DFG (research training group Statistical Modelling, SFB 708) is gratefully acknowledged.

References

Fang, K.-T., Li, R., and Sudjianto, A. (2006). *Design and Modeling for Computer Experiments*. Computer Science and Data Analysis Series. Chapman & Hall/CRC, Boca Raton.

Mühlenstädt, T. and Kuhnt, S. (2009). Kernel interpolation. Technical report, Faculty of Statistics, Technische Universität Dortmund, Dortmund, Germany.

Micula, G. (2002). A variational approach to spline functions theory. *General Mathematics*, 10(1):21–50.

Morris, M. and Mitchell, T. (1995). Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43:381–402.

Sibson, R. (1980). A brief description of natural neighbor interpolation. In Barnett, V., editor, *Interpreting multivariate data*, pages 21–36. Wiley.

Wang, P. and Chen, D.-S. (1996). Continuous optimization by a variant of simulated annealing. *Computational Optimization and Applications*, 6:59–71.

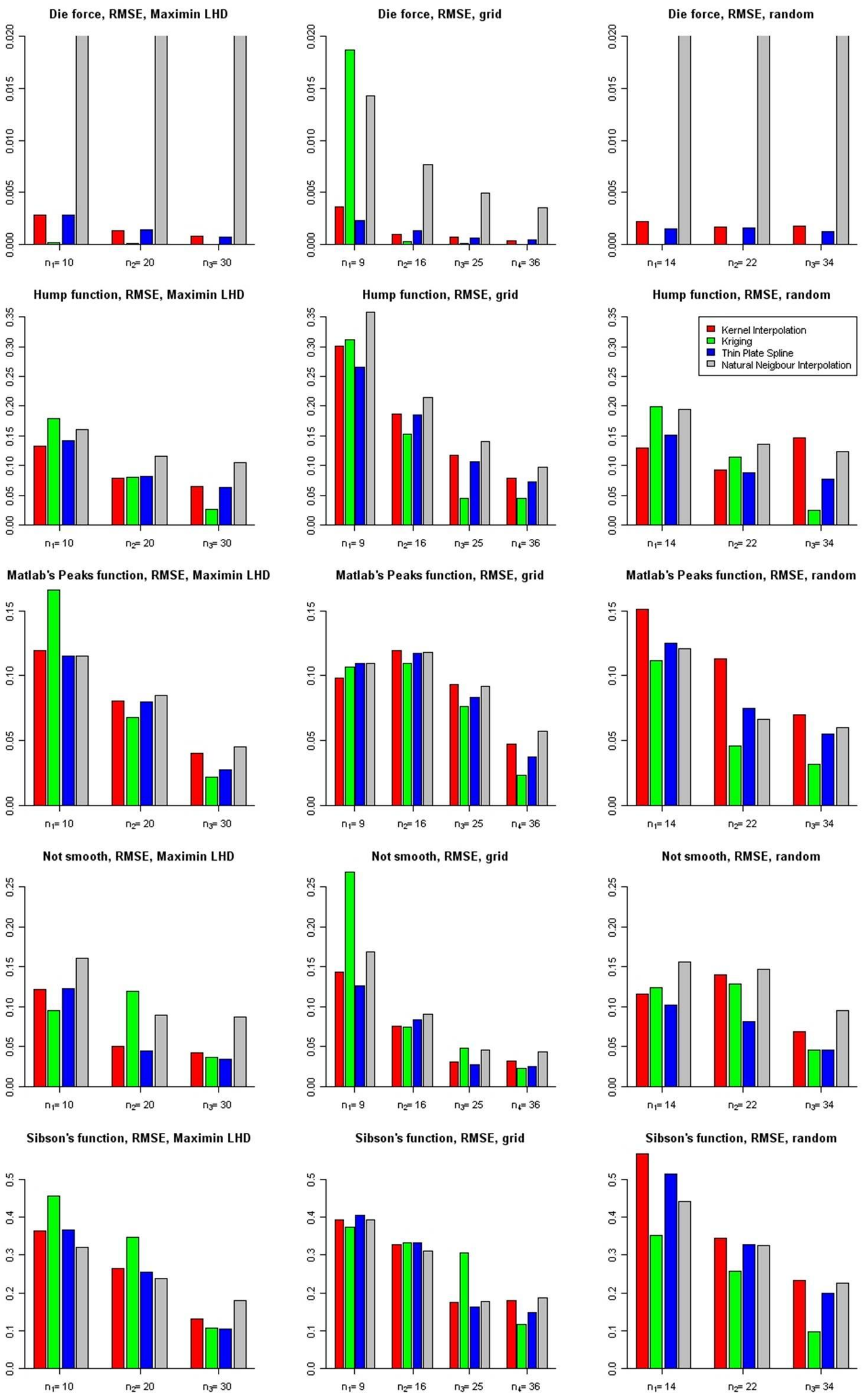

Figure 3: RMSE comparisons for all example functions.