Comparing different interpolation methods on two-dimensional test functions

Thomas Mühlenstädt, Sonja Kuhnt

TU Dortmund

ENBIS EMSE, St. Etienne

01.07.2009
Introduction

Interpolation methods
- Kriging
- Thin plate spline
- Natural neighbor interpolation
- Kernel interpolation

Comparison

Summary
Simulation model for real world phenomena

- Deterministic
- Computational expensive

Needed: Easy to calculate surrogate

Setting:

- Design $D = \{\vec{x}_1, \ldots, \vec{x}_n\}$, $\vec{x}_i = (x_{i,1}, x_{i,2})$
- One-dimensional output $y_1, \ldots, y_n$
- $y_i = f(\vec{x}_i)$, $f$ unknown

Fang et al. (2006)
Multivariate interpolation

Treated approaches:

- Kriging (Gaussian Random Fields)
- Thin plate spline (TPS)
- Natural neighbor interpolation (NNI)
- Kernel interpolation (KI)
Kriging

\[ Y(\vec{x}_i) = g_\beta(\vec{x}_i) + Z(\vec{x}_i), \quad 1 \leq i \leq n, \]

- \( g_\beta(\vec{x}_i) \) regression part (here: \( g_\beta = \beta \in \mathbb{R} \))
- \( Z(\vec{x}) \sim (0, \sigma^2) \) normally distributed
- \( Z(\vec{x}_1) \) and \( Z(\vec{x}_2) \), \( \vec{x}_1 \neq \vec{x}_2 \) explicitly dependent:
  \[ \text{cor}_{\theta}(Z(\vec{x}_1), Z(\vec{x}_2)) \to 1 \quad \text{for} \quad \vec{x}_1 \to \vec{x}_2 \]

*Santner et al. (2003)*
Kriging

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

- Estimation of parameters \( \beta, \theta, \sigma^2 \): REML
- Optimization of Log-likelihood: by \( R \) command `optim` for 50 different initial values
Thin plate spline (TPS)

- Generalization of cubic splines
- Solves the following optimization problem:

  \[
  \text{Search } f^*, \text{ such that } I(f) \text{ is minimized in a suitable functional space under the constraint of interpolation}
  \]

  \[
  I(f) = \int_{\mathbb{R}^2} \left( \frac{\partial^2 f(\vec{x})}{\partial x_1^2} \right)^2 + 2 \left( \frac{\partial^2 f(\vec{x})}{\partial x_1 \partial x_2} \right)^2 + \left( \frac{\partial^2 f(\vec{x})}{\partial x_2^2} \right)^2 \, d\vec{x}
  \]

  \text{Micula (2002)}
Thin plate spline (TPS)

- Can be solved explicitly:

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

- \[ \phi(r) = r^2 \log(r) \]
- Computation \( \lambda_1, \ldots, \lambda_{n+3} \): system of linear equations
Natural neighbor interpolation (NNI)

- Weighted mean of the \( y \)-values
- Strictly local method
- Uses the Voronoi diagramm for weighting

Sibson (1980)
Natural neighbor interpolation (NNI)
Natural neighbor interpolation (NNI)
Kernel interpolation (KI)

- Weight locally fitted linear functions under the constraint of interpolation
- Split up the convex hull of the design into simplices $S_j$ (Delaunay triangulation)
- Fit a linear function $\hat{y}_j(x)$ to each simplex $S_j$
- Weight the linear functions:

$$\frac{\sum_{j=1}^{N} g_j(\tilde{x})\hat{y}_j(\tilde{x})}{\sum_{j=1}^{N} g_j(\tilde{x})}, \quad g_j(\tilde{x}) = \frac{1}{\prod_{i=0}^{2} \| \tilde{x}_i^j - \tilde{x} \|_2^2}$$

*Mühlstädt and Kuhnt (2009)*
Kernel interpolation (KI)
Kernel interpolation ($KI$)
Kernel interpolation ($KI$)
Kernel interpolation (\(KI\))

- Interpolation
- \(KI\)
Kernel interpolation (KI)
Kernel interpolation (KI)

- **Introduction**
- **Interpolation**
  - Kriging
  - TPS
  - NNI
  - KI
- **Comparison**
- **Summary**
- **References**
Kernel interpolation ($KI$)
Kernel interpolation ($KI$)
Kernel interpolation ($KI$)
Kernel interpolation (KI)
Kernel interpolation ($KI$)
Kernel interpolation (KI)

Kernel interpolation (KI) is a method used for function approximation. It involves the use of a kernel function to approximate a given function. The diagram illustrates the process, with various weight functions and linear functions used in the approximation.
Kernel interpolation ($KI$)
Kernel interpolation (KI)

The diagram illustrates the concept of kernel interpolation (KI) in the context of four-dimensional space. The graph shows a function f(x) plotted against x, with a range from -1 to 1. The function is represented by a series of linear segments, each connected by points. The right side of the diagram displays a bar chart indicating the weight distribution for linear functions.
Kernel interpolation (KI)
Kernel interpolation (KI)

- **Kernel interpolation (KI)**
- Linear function
- Weight
- Comparison
- Summary
- References
Kernel interpolation ($KI$)
Kernel interpolation ($KI$)
Kernel interpolation (\(KI\))

Properties of the Kernel interpolation:

- Continuous
- Differentiable (also at observation points)
- Able to predict outside of observation range
- Exactly reproduces linear functions
Comparison of interpolation methods

Which interpolation method should be used?

- 5 analytical 2-dim. examples
- Aim: high prediction power
- Prediction power: root mean square error (RMSE)
- Different experimental designs
Designs for computer experiments

- Difference to 'standard' DoE: no random error
- Example for a 'good' space filling designs: maximin latin hypercube design
- Also often encountered: factorial designs
- For sequential procedures: designs with clusters

Fang et al. (2006)
Used Designs

Maximin

Full factorial

Cluster

n = 10

n = 20

n = 30

n = 9

n = 16

n = 25

n = 36

n = 14

n = 22

n = 34
Root Mean square error (RMSE)

- Criterium for comparing different predictions:

\[ RMSE(y, \hat{y}, \vec{r}_1, \ldots, \vec{r}_m) := \sqrt{\frac{1}{m} \sum_{i=1}^{m} (y(\vec{r}_i) - \hat{y}(\vec{r}_i))^2} \]

- \( \vec{r}_1, \ldots, \vec{r}_m \) points in the design space
- \( \vec{r}_1, \ldots, \vec{r}_m \) 10000 points of a Sobol’ Uniform Sequence
Implementation aspects

- Easy: Thin plate spline
- Acceptable:
  - Kriging
  - Kernel interpolation (if Delaunay triangulation is available)
- Difficult: Natural neighbor interpolation (Calculation of the Voronoi diagram constrained to design space)
Computation times

<table>
<thead>
<tr>
<th>Design</th>
<th>$TPS$</th>
<th>$KI$</th>
<th>Kriging</th>
<th>$NNI$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_{10}^{Mm}$</td>
<td>10</td>
<td>15</td>
<td>13 - 22</td>
<td>839</td>
</tr>
<tr>
<td>$D_{20}^{Mm}$</td>
<td>13</td>
<td>27</td>
<td>69 - 162</td>
<td>1536</td>
</tr>
<tr>
<td>$D_{30}^{Mm}$</td>
<td>15</td>
<td>39</td>
<td>178 - 448</td>
<td>2239</td>
</tr>
</tbody>
</table>

- Computation times in seconds
- For 10000 prediction points
- Based on the maximin latin hypercube designs
Die force

- Model for the effective die force in sheet metal forming processes depending on friction and blankholder force
- Nearly linear in one variable

\[ f(\bar{x}) = 0.9996(1090.91 + 4x_1 x_2) \exp \left( x_1 \frac{\pi}{2} \right), \]

\[ D = [0.05, 0.2] \times [5, 30] \]
Die force

Die force, RMSE, Maximin LHD

Die force, RMSE, grid

Die force, RMSE, random
Hump function

- Standard example from optimization literature
- Extreme values on boundary

\[ f(x, y) = 1.0316 + 4x^2 - 2.1x^4 + \frac{1}{3}x^6 + xy - 4y^2 + 4y^4, \]
\[ D = [-5, 5]^2 \]
Hump function
Matlabs peaks function

- Good example for a hilly contour

\[
f(\mathbf{x}) = 3(1 - x_1)^2 \exp\left(-x_1^2 - (x_2 + 1)^2\right) - \\
10 \left(\frac{x_1}{5} - x_1^3 - x_2^5\right) \exp\left(-x_1^2 - x_2^2\right) - \\
\frac{1}{3} \exp\left(-(x_1 + 1)^2 - x_2^2\right), \quad D = [-2, 2]^2
\]
Matlabs peaks function
Assumption of smoothness often unrealistic
Continuous but not differentiable

\[ f(x, y) = |x^2 + \sin(0.5\pi y) - y|, \quad D = [0, 1]^2. \]
Not smooth, RMSE, Maximin LHD

Not smooth, RMSE, grid

Not smooth, RMSE, random
Sibson’s function

- Proposed by Sibson (1980) for illustrating NNI
- High complexity for size of sample

\[
f(\bar{x}) = \cos \left( 4\pi \sqrt{(x_1 - 0.25)^2 + (x_2 - 0.25)^2} \right),
\]

\[D = [0, 1]^2\]
Sibson’s function
Summary

- No overall winner
- Decision depends on design
- Kriging often very efficient, especially for higher sample sizes, designs with clusters
- $KI$ and $TPS$ good for small sample sizes
- $NNI$ not recommendable


