Clustering by optimal subsets to describe environment interdependencies

J. Glänzel, R. Unger

Preprint 2017-3

Preprintreihe der Fakultät für Mathematik
ISSN 1614-8835

Impressum:
Herausgeber:
Der Dekan der
Fakultät für Mathematik
an der Technischen Universität Chemnitz
Sitz:
Reichenhainer Strae 39
09126 Chemnitz
Postanschrift:
09107 Chemnitz
Telefon: (0371) 531-22000
Telefax: (0371) 531-22009
E-Mail: dekanat@mathematik.tu-chemnitz.de
Internet:
http://www.tu-chemnitz.de/mathematik/
ISSN 1614-8835 (Print)

Clustering by optimal subsets to describe environment interdependencies

August 9, 2017

Janine Glänzel
Fraunhofer Institute for Machine Tools and Forming Technology IWU
Reichenhainer Strasse 88
09126 Chemnitz, Germany
janine.glaenzel@iwu.fraunhofer.de

Roman Unger
Chemnitz University of Technology
Department of Mathematics
09107 Chemnitz, Germany
roman.unger@mathematik.tu-chemnitz.de

Abstract

The paper copes with the problem of finding an optimal subset of interpolation points out of a given large set of computed values, arising from a finite element simulation. This simulation computes environment data, which are on their part input data for finite element simulations of machine tools.
For machine tool manufacturers it is still a seriously problem that the machine works imprecisely and products wastrel if environment values like temperature changes. The change of the environment boundary conditions contribute to the phenomenon through sunlight or cold draught owing to open doors of the machine hall or factory. Resulting thermo-elastic effects on the tool center point are one of the major reasons for positioning errors in machine tools.
A genetic search algorithm for clustering relevant heat transfer coefficient values over the geometric surface through computational fluid dynamics (CFD) simulations will be described. These values are the input data for a developed thermo-elastic correction algorithm.
Keywords: Optimal subset problem, Radial Basis Functions, FEM, CFD
MSC: 90C27 Combinatorial optimization, 97N50 Interpolation and approximation

Contents

1 Introduction

The paper copes with the problem of finding an optimal subset of interpolation points out of a given large set of computed values, arising from a finite element simulation to establish a thermo-elastic correction algorithm for increasing the workpiece quality in production processes.
The quality of workpiece is dependend by the thermo-elastic behavior of the machine tool during the production process. Machine tool deformation occurs due to waste heat from motors and frictional heat from guides, joints and the tool, while coolants act to reduce this influx of heat. Additional thermal influences come from the machine tool's environment and foundation. This leads to inhomogeneous, transient temperature fields inside the machine tool which displace the tool center point (TCP) and thus reduce production accuracy and finally the product quality [1]. Next to approximation strategies

such as characteristic diagram based [8] and structure model based correction [4], the most reliable way to predict the TCP displacement is via structure-mechanical finite element (FE) simulation. A CAD model of a given machine tool serves as the basis for this approach. On it a FE mesh is created. After establishing the partial differential equations (PDEs) describing the heat transfer within the machine tool and with its surroundings, FE simulations are run in order to obtain the temperature fields of the machine tool for specified load regimes. Using linear thermo-elastic expansion, the deformation can then be calculated from each temperature field and the displacement of the TCP read from this deformation field, see [6]. The accuracy of this latter approach depends on the correct modelling of the heat flux within the machine tool and the exchange with its surroundings. In order to calculate the correct amount of heat being exchanged with the environment, one may use known parameters from well-established tables. However, if the surrounding air is in motion or otherwise changing, computational fluid dynamics (CFD) simulations are required to accurately determine these transient parameters. This two-step approach makes realistic thermo-elastic simulations particularly complicated and time-consuming. Negative aspect of this approach is the very computing time intensive CFD simulation. Some methods aiming at real-time thermo-elastic simulations based on model order reduction must therefore rely on the inaccurate predetermined parameter sets [5]. This could be helped if all the necessary CFD simulations could be run in advance and supplied to the thermo-elastic models when they are needed.
Nevertheless the whole output of this CFD simulations is to much amount of data for an effective computation of the correction steps. Therefore a reduction of this data is desirable wherefore the ideas of this paper comes up.
This paper starts by introducing an interpolation method with radial basis functions (RBF) and the description of the optimal subset finding problem. After that a solution algorithm for solving this optimal subset problem is developed, based on the idea of genetic algorithms. In the numerical examples the potency of this algorithm is shown in some benchmark examples, where the optimal solution is known. Afterwards it is also demonstrated that the algorithm works for practical relevant datasets too.

2 Radial Basis Functions

The basic principle in the use of radial basis functions is solving an interpolation problem for m arbitrary sample points x_1, $x_2 \cdots x_m$ in $\mathbb{R}^d$ with given values $w_1, w_2 \cdots w_m$ in $\mathbb{R}$ to find a function $f : \mathbb{R}^d \to \mathbb{R}$, fulfilling the interpolation condition

$$f(x_i) \quad = \quad w_i \qquad \text{for} \quad i = 1 \cdots m. \tag{1}$$

The first step is to find an ansatz for (1). A function η_i: $\mathbb{R}^d \to \mathbb{R}$ is a radial basis function if a function $\phi : \mathbb{R} \to \mathbb{R}$ exists, which satisfies $\eta_i(x) = \phi(||x - x_i||)$ for a fixed point $x_i \in \mathbb{R}^d$. Commonly used types of RBF ansatz functions with the structure $\phi(r)$ with $r = ||x - x_i||$ are so called Gaussian, multiquadic, polyharmonic spline function, for more details see [12].
In our case we used a simple polyharmonic spline, which sounds $\phi(r) = r$. Other variations, especially for large data sets are RBFs with compact support, e.g. the "Wendland functions" [2]. These basic functions are characterized through a sparse interpolation system.
A usual ansatz for an interpolation function is described by

$$f(x) \quad = \quad \sum_{i=1}^{m} \beta_i \phi_i(x). \tag{2}$$

Inserting condition (1) in (2) we get linear equations

$$f(x_i) = \sum_{j=1}^{m} \beta_j \phi_j(x_i) \quad = \quad w_i \qquad \text{for} \quad i = 1 \cdots m,$$

which can be written as a linear system by

$$\Phi \beta^{RBF} \quad = \quad w. \tag{3}$$

The components of (3) are defined by the system matrix $\Phi := [\phi_j(x_i)]$ for $i, j = 1 \cdots m$, the vector $\beta^{RBF} = [\beta_1, \beta_2 \cdots \beta_m]^T$ of the unknown coefficients and the right hand side $w = [w_1, w_2 \cdots w_m]^T$. The system matrix Φ is obviously symmetric. In [2], it is shown that Φ is positive definite for a vast variety of RBFs. The only disadvantage of this RBF ansatz is that a large number of sample points are needed to get a sufficiently exact approximation

of a constant or linear function. One possible way to cope with this is to add a polynomial part $p(x)$.

$$f(x) \quad = \quad \sum_{i=1}^{m} \beta_i^{RBF} \phi_i(x) + p(x)$$

The d-variate polynomial $p \in \pi_n(R^d)$ of degree at most n is defined as

$$p(x) \quad = \quad \sum_{j=1}^{k} \beta_j^{POLY} p_j(x),$$

with $k = dim(\pi_n(R^d))$ and basis polynomials p_j for $j = 1 \ldots k$. Consequently, this ansatz has $(m + k)$ unknown coefficients, while the interpolation system (1) consists of only m equations. Therefore an added condition is imposed by

$$\sum_{j=1}^{m} \beta_j^{RBF} p(x_j) \quad = \quad 0 \quad \text{for all} \quad p \in \pi_n(R^d).$$

The result is a linear matrix system of dimension $(m + k)$

$$\begin{bmatrix} \Phi & P \\ P^T & 0 \end{bmatrix} \begin{bmatrix} \beta^{RBF} \\ \beta^{POLY} \end{bmatrix} = \begin{bmatrix} w \\ 0 \end{bmatrix} \tag{4}$$

A simple and good choice for $\pi_n(R^d)$ are linear polynomials, e.g.

$$\begin{aligned} p_1(x) &= x_1 \\ p_2(x) &= x_2 \\ &\cdots \\ p_d(x) &= x_d \\ p_k(x) &= 1 \end{aligned}$$

where $x \in \mathbb{R}^d$, $k = d + 1$ and x_i is the i-th component of x.
The solution vector $\beta = [\beta^{RBF}, \beta^{POLY}]^T$ of eq. (4) contains the coefficients for the whole interpolation function

$$f(x_i) = \sum_{i=1}^{m} \beta_i \phi_i(x) + \sum_{j=1}^{k} \beta_{(m+j)} p_j(x). \tag{5}$$

3 The Optimal Subset Problem

The CFD-simulation in Ansys computes the values of heat transfer coefficients, velocity vectors or other characterising values in a huge number of nodes on the surface of the computed domain.
Because of very large node numbers it is desirable for fast evaluations, to reduce this number of nodes but keep a high accuracy. In the paper this is done by choosing a 'good subset' with a fixed size m of node values which will be used to build an interpolation function, based on radial basis functions.
The mathematical formulation for choosing this optimal nodes for the interpolation can be done in the following way.
Given a set $V = \{1, 2, \cdots N\}$ which corresponds with the nodes $x_1, x_2, \cdots x_N$ of the finite element simulation and her computed values $w_1, w_2, \cdots w_N$ in this nodes. Furthermore define a number $m < N$ and a weighting function

$$f : S \subset V \to \mathbb{R}_0^+ \quad \text{with} \quad |S| = m \tag{6}$$

which maps a m-sized subset S to a real number greater or equal zero. The 'Optimal Subset Problem' is minimizing this weighting function f as

$$\begin{array}{l} \min \\ S \subset V, \\ |S| = m \end{array} \quad f(S) \quad . \tag{7}$$

In our application the weighting function f itself is a function, calculating the interpolation error which occurs when the m node values of S are used to build the radial basis interpolation function f_S like in equation (5), evaluate it in all N nodes of the set V and compare the interpolated values with the given values w_i. Possible error measures are the sum of squares

$$f(S) := \sqrt{\sum_{i=1}^{N} (f_S(x_i) - w_i)^2} \tag{8}$$

or a pointwise computed maximum error

$$f(S) := \max_{i=1 \cdots N} |f_S(x_i) - w_i)| \quad . \tag{9}$$

Clearly the value of $f(S)$ is (up to small roundoff errors) zero if $m = N$ that is $S = V$, but it becomes greater then zero for $m < N$.

The challenge is now to find out the optimal subset S which minimizes (7). For very small sets V and small numbers m this can be done by computing all possible subsets $S \subset V$ and compare the values of (6), but in all practical relevant cases this becomes impossible due to the combinatorical explosion. The number of different subsets $S \subset V$ of size m is given by the binomial coefficient $\binom{N}{m}$ which is defined as

$$\binom{N}{m} := \frac{N!}{m!(N-m)!},$$

some values are shown in the following table 1.

N	m	$\binom{N}{m}$
64	5	7624512
64	6	74974368
64	7	$\approx 6 \cdot 10^8$
64	20	$\approx 1.9 \cdot 10^{16}$
8065	100	$\approx 2.6 \cdot 10^{232}$

Table 1: Some values for N over m

The subset problems, belonging to the first three lines with $\binom{64}{5}$, $\binom{64}{6}$, $\binom{64}{7}$ are computeable explicit by a full search and will be used in the section 5.1 as a benchmark value, because they provide the global minimum of the weighting function (6). The case $\binom{64}{20}$ is not computable by a full search with reasonable effort and it will be used to show the minimization algorithm works.
The last line with $\binom{8065}{100}$ is a practical use case, the immense number of different subsets makes it impossible for all times to perform a full search by the computation of all cases, but the example in section 5.3 will show how the minimization algorithm works also in this case.
In the next section an optimization algorithm for finding the optimal subset will be developed.

4 Genetic Algorithms

The minimization of a function is a widely used technique for solving optimization problems. Depending on the properties of the function which is to minimize (continuity, differentiability, ...) many effective algorithms exist for such classes of functions.
But the challenge of our minimization problem (7) is, that our weighting function is not continous, moreover it is defined over a discrete set. All algorithms, depending on gradients are not applicable for such a problem. In case of very small sets it is possible to perform an exhaustive search, but for all practical purposes the computational effort for this idea is impossible to tackle.
For moderate sizes there exist branch and bound algorithms, based on ideas of Narendra and Fukunaga for feature subset selection in statistics [7] or their implementations, given by Ridout [9] or [11].
They need a subset monotonicity condition, such that for two subsets $S_1 \subset V$ and $S_2 \subset V$ with $S_1 \subset S_2$ the weighting function (6) fulfills – in case of minimization – the condition $f(S_2) \leq f(S_1)$. This holds for our problem (7) too.
Nevertheless in this papers the considered and computable problem sizes are in the magnitude of $\binom{24}{12}$ (Narendra) or $\binom{35}{15}$(Ridout) which is much lesser than our aimed problem sizes.
However there exist one widely used technique for such problems, it is the class of genetic algorithms. The basic idea of this algorithms is to duplicate the selection of the fittest in natural evolution processes.
A solution candidate is seen as an individual, a bulk of individuals is seen as population and the population is developed from one generation to the next generation by annihilating individuals with poor fitness, selecting individual with good fitness for reproduction and create a new generation.
In principle, after fixing a population size p, a genetic algorithm consist of the following steps:

1. **Initial population:** Generating the initial population by randomly creation of p individuals.

2. **Assessment:** Perform a rating of all individuals by evaluating the fitness function for each one.

3. **Stopping criterion:** Use the best individual as solution if the best fitness value reaches a given threshold, or if it not changes over a fixed number of generations.

4. **Reproduction:** Annihilate individuals with poor fitness, use individuals with good fitness for reproduction to create a new generation.

5. **Mutation:** Mutate some new generated individuals with a (small) probability. This mutation step offers the inclusion of new solution candidates to the population.

6. **Loop to next generation:** Go to step 2.

Especially for the reproduction step there exists many different variants for the selection of parent individuals and their reproduction. For a more detailed introduction to genetic algorithms see for example [3] or [10]. In our adoption of the algorithm, the important steps from above are done in the following way.

1. **Initial population:** In our use case the individuals are subsets $S \subset V$ of node numbers, i.e. integers out of $\{1, 2, \cdots N\}$, such the initial population is generated as a family of p random subsets $S \subset V$.

2. **Assessment:** The assessment of each individual is done by setting up the RBF interpolation system with these basis nodes corresponding to the individuals set S, evaluate this system in all N nodes, compare the interpolated values with the given values like given in equation (8) and use the result as fitness value. Please note, that these settings perform a minimization, such that best fitness value is zero.

4. **Reproduction:** During the reproduction step a small number of individuals with the best fitness values, e.g. the elite are transfered unchanged to the new population. The remaining individuals of the new population are created by mixing individuals out of the upper half of the population. For two parent sets S_1 and S_2 this is done by taking m elements out of $S_1 \bigcup S_2$ to create a new individual.

5. **Mutation:** The mutation step is performed by changing some elements of the newly created set S to a random value out of $\{1, 2 \cdots N\}$.

5 Numerical Examples

In the numerical examples we consider three different use cases. In the first example a benchmark to the optimization algorithm is done, because of the moderate number of different subsets it is possible to do a full search and find the optimal solution exactly to compare it with the solution found by the optimization algorithm.
The next two examples consider practical use cases, where the optimization algorithm works to find an optimal subset. Comparision with a full computed example is impossible.
In all numerical examples the assessment of the solutions during minimization is done by the error measure, given in equation (8).

5.1 Benchmarks examples with $N = 64, m = 5, 6, 7$

This example is constructed for benchmarking the optimization algorithm in comparison with a known global minimum. This global minimum ist computed by the assessment of all $\binom{64}{m}$ cases for $m = 5, 6, 7$. For some details of the computation of all this subsets see sections A.1 and A.2 in the appendix.
The exact numbers of how many different subsets exists in this cases are given in table 1. The original ansys data are shown in figure 1.

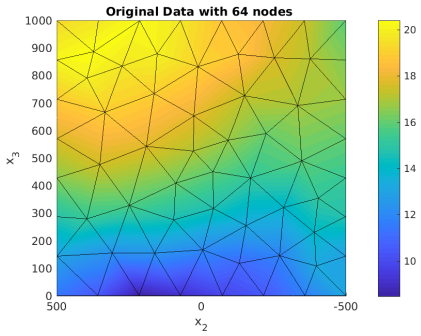


Figure 1: Original mesh data with 64 nodes

In the following tables 2, 3, 4 the best six sets with their interpolation error are shown. Moreover, for an illustration that there really exists 'good and bad sets' in table 2 the set with the largest error up to sets producing not a

number results is given in the last line too. The corresponding nodes to this set are located all on a straight line on the right boundary of the domain. Therefore the interpolation with this set is one of the worstest variants.

error	set S with $\|S\| = 5$
4.180556e+00	13 16 45 48 61
4.268491e+00	18 29 45 48 61
4.291127e+00	2 18 45 48 61
4.301570e+00	3 15 45 48 61
4.344761e+00	1 18 45 48 61
4.346238e+00	13 20 45 48 61
1.771360e+04	42 56 59 60 61

Table 2: The best six sets and the worst one (without NaN) for $m = 5$

When running the genetic algorithm for solving the optimization problem it runs after about 800 generations into the global minimum. For the evolution of the best error over the generations see figure 2.

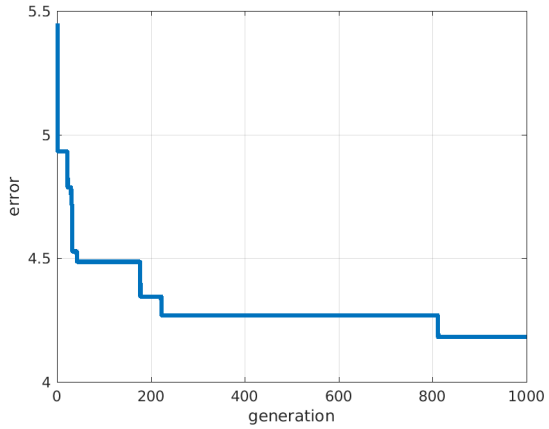
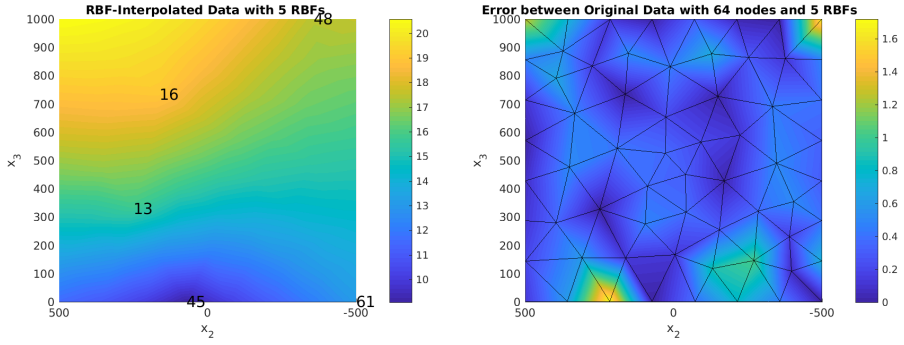


Figure 2: Evolution of the error over the generations for $m = 5$

The resulting interpolation with this optimal set $S = \{13, 16, 45, 48, 61\}$ is shown in figures 3a and 3b. The location of the interpolation centers of this

five radial basis function is marked in figure 3a with the number placed on the center position.



(a) Interpolation (b) Interpolation error

Figure 3: Example for $N = 64$ and $m = 5$ (RBFs).

In the following tables and pictures the results of the same computations for $m = 6$ and $m = 7$ are shown. At first remark that the optimal solution becomes better for increasing value of m. It comes out, that also in this cases the genetic optimization algorithm runs into the global minimum.

error	set S with $\|S\| = 6$
3.908642e+00	13 16 42 45 59 61
3.945560e+00	1 15 42 45 59 61
3.953646e+00	18 20 42 45 59 61
3.956622e+00	3 15 42 45 59 61
3.965478e+00	1 31 32 42 49 59
3.999228e+00	13 24 42 45 59 61

Table 3: The best sets for $m = 6$

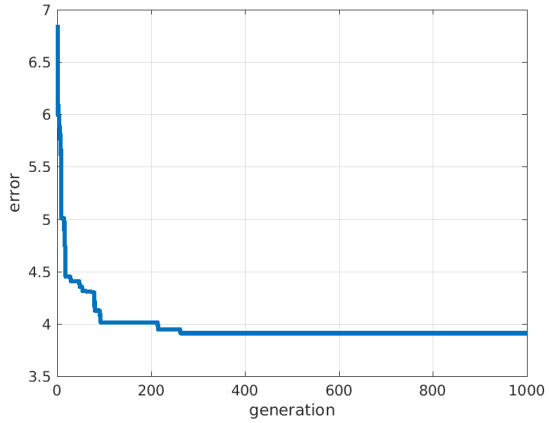
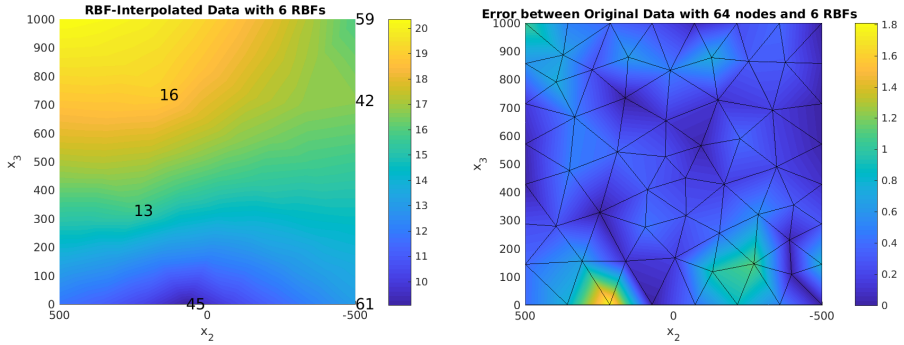


Figure 4: Evolution of the error over the generations for $m = 6$



(a) Interpolation (b) Interpolation error

Figure 5: Example for $N = 64$ and $m = 6$ (RBFs).

error	set S with $\lvert S\rvert = 7$
3.447288e+00	8 13 15 49 51 59 61
3.479505e+00	13 15 42 49 51 59 61
3.500212e+00	1 8 32 49 51 59 61
3.503142e+00	3 15 42 45 51 59 61
3.545113e+00	8 13 15 22 49 59 61
3.589773e+00	3 15 22 42 45 59 61

Table 4: The best sets for $m = 7$

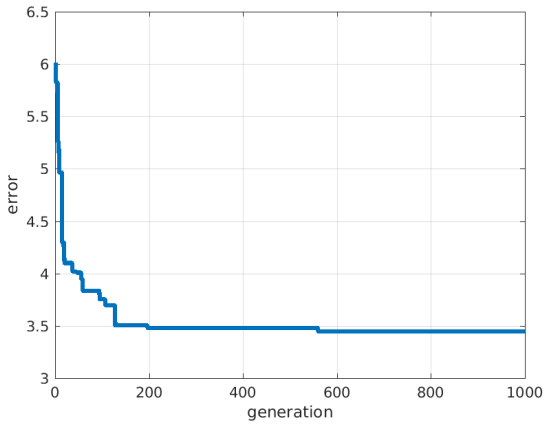
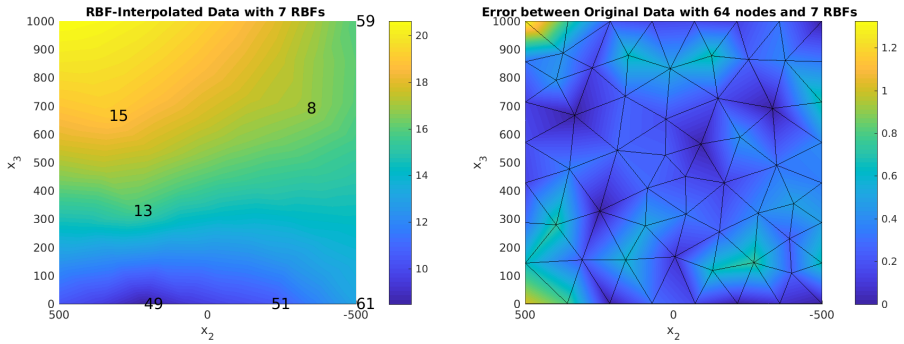


Figure 6: Evolution of the error over the generations for $m = 7$



(a) Interpolation (b) Interpolation error

Figure 7: Example for $N = 64$ and $m = 7$ (RBFs).

5.2 Use case with $N = 64, m = 20$

In this example the algorithm chooses 20 nodes out of 64 to create the interpolation system. It can reduce the interpolation error to 1.7308 by using the subset $S = \{5, 8, 11, 13, 19, 24, 25, 34, 35, 36, 38, 42, 45, 49, 54, 57, 58, 59, 61, 63\}$, see figure 9a for the spatial distribution of this nodes.

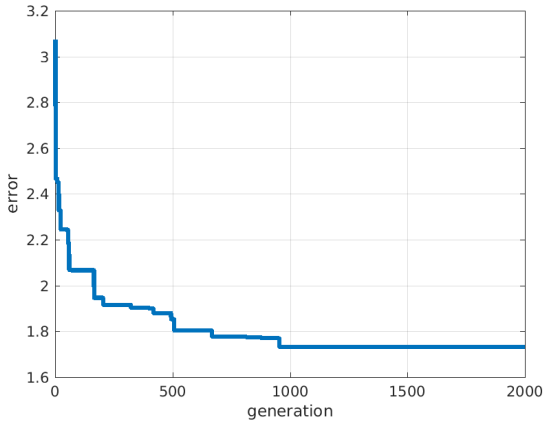
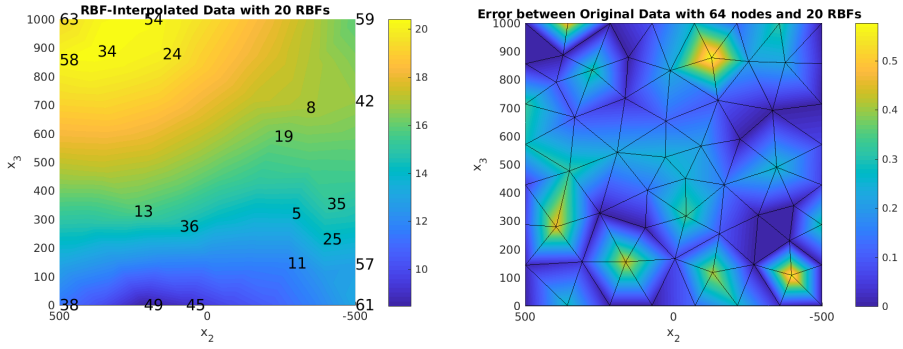


Figure 8: Evolution of the error over the generations for $m = 20$

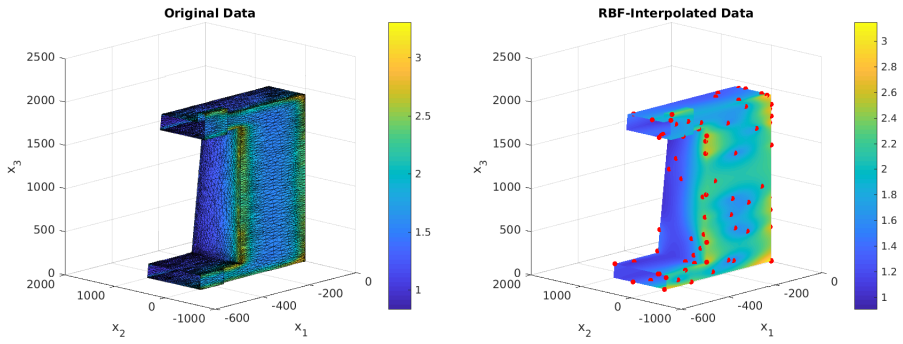


(a) Interpolation (b) Interpolation error

Figure 9: Example for $N = 64$ and $m = 20$ (RBFs).

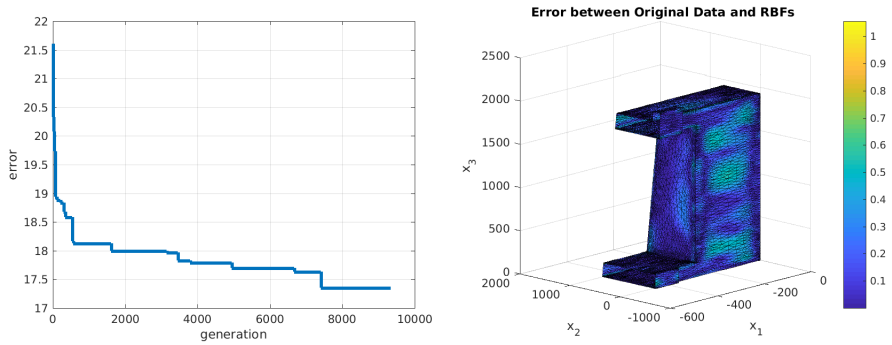
5.3 Practical example with $N = 8065, m = 100$

In this numerical example the computation of a heat transfer coefficient on the Auerbach ACW machine with 8065 nodes is used as input data. This input data are shown in figure 10a. The corresponding interpolation with $m = 100$ RBFs is shown in figure 10b, the interpolation centers of this 100 RBFs are marked with red balls.



(a) Original mesh with 8065 nodes (b) Interpolation with 100 RBFs

Figure 10: Example for $N = 8065$ and $m = 100$ (RBFs).



(a) Evolution of the error (b) Interpolation error

Figure 11: Example for $N = 8065$ and $m = 100$ (RBFs).

6 Conclusion and Outlook

This paper presents an efficient clustering approach for characteristic values out of the fluid simulation to describe the environment interdependencies of machine tools. The fluid simulation calculates a field of heat transfer coefficient around the surface of the machine component as characteristic values. Using the RBF theory to interpolate this field for the following steps to find an optimal subset of nodes.
The implementation is ensued through a genetic algorithm. Some numerical use cases presents the functionality of the developed method. The last part shows the result for an optimal subset of characteristic values on a realistic component, the column of a machine tool.
Finally, the planned work is to use the developed method for the numerical calculation of a whole working day in a factory hall. Therefore many time-intensive fluid simulation are necessary and produce a lot of data. So the next steps are to assign this method for handling the big data of characteristic values for each time step.

7 Acknowledgements

This research is being supported by the German Research Foundation in the context of the Collaborative Research Centers Transregio 96, subproject B09.

A Appendix

In the appendix some remarks to the computation of the considered subsets $S \subset V$ with fixed size m are given. Especially for the full computation of the small benchmark examples with sizes $\binom{64}{5}$, $\binom{64}{6}$, $\binom{64}{7}$ in section 5.1 this was used.

A.1 Relationship of a subset $S \subset V$ and a N-bit integer

For the handling of subsets the usage of an one-to-one mapping from a set $V = \{1, 2, \cdots N\}$ to the bits of a N-bit integer value K, given in the following sense:

Set V:	N	N-1	N-2	$\cdots$	2	1
Bits of integer K:	1	0	1	$\cdots$	0	1

is very useful. Every value of K defines a subset $S \subset V$ by using the 1-bits of K as a marker, that the corresponding element of V belongs to the subset S.
There exists 2^N differents subsets of V, the N-bit integer K can achieve 2^N values.
Moreover, all subsets S with $|S| = m$ corresponds with all integers K where exactly m bits are set to 1.

A.2 Sequential creating all subsets S with $|S| = m$

Using the basic concept out of the last subsection, it would be thinkable in a naive implementation to count the value of K from 1 to 2^N, determine the number of 1-bits in K and use the correponding subset if there are exactly m bits set to 1. But already for small numbers of N this counting consumes so much computation time, that it becomes impossible.
A much better approach is jumping from a given number K which has m bits set to 1 to the next larger number with m bits set to 1.
Looking at the bit representation this can be done by the following algorithm.

1. Start with the lowest significant bit position and find the position where the first change from a 1-bit to a 0-bit occurs.

2. Exchange this 2 bits, i.e. the 1-bit becomes the value 0, and the 0-bit becomes 1.

3. Shift the block of remaining 1-bits right from the position found in step 1 as far as possible to the right.

To clearify this idea consider the following example. Setting $N = 10$, $m = 4$ and starting with $K = 0010011100$ which is the decimal value of $256 + 16 + 8 + 4 = 284$.

K=	0	0	1	0	0	1	1	1	0	0
position of 1 jumps to 0 :	0	0	1	0	0	1	1	1	0	0
swap the bits :	0	0	1	0	1	0	1	1	0	0
remaining 1-block :	0	0	1	0	1	0	1	1	0	0
shift to the right:	0	0	1	0	1	0	0	0	1	1

So the new value of K is 0010100011 respectively $256 + 32 + 2 + 1 = 291$ which the next integer greater then the given value and has m 1-bits too.

References

[1] J. Bryan. International status of thermal error research. *CIRP Annals Manufacturing Technology*, 1990.

[2] M. D. Buhmann. *Radial Basis Functions*. Cambridge University Press, 2004.

[3] D. A. Coley. *An Introduction to Genetic Algorithms for Scientists and Engineers*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1998.

[4] K. Grossmann et al. Challenges in the development of a generalized approach for the structure model based correction. *Applied Mechanics and Materials*, 794:387–394, 2015.

[5] S. Ihlenfeldt et al. Modelling of thermal interaction between environment and machine tool. *Thermo-energetic design of machine tools*, pages 111–124, 2014.

[6] J. Mayr et al. Compensation of thermal effects on machine tools using a fdem simulation approach. *9th International Conference and Exhibition on LASER Metrology, Machine Tool, CMM and Robotic Performance*, 2009.

[7] P. M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Trans. Comput.*, 26(9):917–922, Sept. 1977.

[8] C. Naumann et al. Characteristic diagram based correction algorithms for the thermo-elastic deformation of machine tools. *Proceedings 48th CIRP Conference on Manufacturing Systems*, 2015.

[9] M. S. Ridout. [Algorithm AS 233] An improved branch and bound algorithm for feature subset selection. *Applied Statistics*, 37:139–147, 1988.

[10] S. N. Sivanandam and S. N. Deepa. *Introduction to Genetic Algorithms.* Springer Publishing Company, Incorporated, 1st edition, 2007.

[11] P. Somol, P. Pudil, and J. Kittler. Fast branch & bound algorithms for optimal feature selection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(7):900–912, 2004.

[12] R. Unger. *Obstacle Description with Radial Basis Functions for Contact Problems in Elasticity.* Preprint CSC 09-01 TU Chemnitz, 2009.