Advanced surrogate models within the robustness evaluation

Dirk Roos[2], Thomas Most[1], Jörg Unger[1] & Johannes Will[2*]

[1] Institute of Structural Mechanics, Bauhaus University Weimar, Germany

[2] DYNARDO – Dynamic Software and Engineering GmbH, Weimar, Germany

Abstract

In real case applications within the virtual prototyping process, it is not always possible to reduce the complexity of the physical models and to obtain numerical models which can be solved quickly. Usually, every single numerical simulation takes hours or even days. Although the progresses in numerical methods and high performance computing, in such cases, it is not possible to explore various model configurations, hence efficient surrogate models are required.

The paper gives an overview about advanced methods of meta-modeling. In addition, some new aspects are introduced to impove the accuracy and predictability of surrogate models, commonly used in numerical models for automotive applications. Whereby, the main topic is reducing the neccessary number of design evaluations, e.g. finite element analysis within global variance-based sensitivity and robustness studies. In addition, the similar approach can be used to perform optimization and stochastic analysis and to create synthetic meta-models for experimental data.

Keywords: surrogate models, meta-modeling, regression analysis, moving least square approximation, neural networks, multi-layer perceptron, support vector regression, robustness evaluation

*Contact: Dr.-Ing. Johannes Will, DYNARDO – Dynamic Software and Engineering GmbH, Luthergasse 1d, D-99423 Weimar, Germany, E-Mail: johannes.will@dynardo.de

1 Introduction

1.1 Meta-modeling

Meta-modeling is one of the most popular strategy for design exploration within nonlinear optimization (see e.g. Booker et al. (1999); Giunta and Watson (1998); Simpson et al. (2003)) and stochastic analysis (see e.g. Choi et al. (2001); Youn et al. (2004); Yang and Gu (2004); Rais-Rohani and Singh (2004)). Moreover, the engineer has to calculate the general trend of physical phenomena or would like to re-use design experience on different projects. Due to the inherent complexity of many engineering problems it is quite alluring to approximate the problem and to solve other design configurations in a smooth sub-domain by applying a surrogate model (Sacks et al. (1989); Simpson et al. (2001)). Starting from a reduced number of simulations, a surrogate model of the original physical problem can be used to perform various possible design configurations without computing any further analyses. So, the engineer may apply a classical design of experiment or stochastic sampling methods for more than $n = 10...15$ input parameters to calculate the simulations.

For a global variance-based sensitivity analysis it is recommended to scan the design space with latin hypercube sampling and to estimate the sensitivity with the multivariate statistic based on surrogate models. Results of a global sensitivity study are the global sensitivities of the optimization or random variables due to important responses. So, it is possible to identify the sub domains for optimization and reliability analysis.

1.2 Latin hypercube sampling

In order to obtain meaningful correlations between the input and output variables it is essential to precisely capture the input correlations in the simulated values. Monte Carlo-based methods use digital generation of pseudo-random numbers to produce artificial sample values for the input variables. Typically, plain Monte Carlo methods are fairly well able to represent individual statistics of the random variables. At small sample sizes N, however, the prescribed correlation structure may be rather heavily distorted. Unfortunately, solving of many real–world engineering problems is very expensive so that only a small number of samples can be accepted.

Significant improvement can be made by utilizing the latin hypercube sampling method (see Florian (1992)). Latin hypercube sampling is an advanced Monte Carlo simulation and a further development of the stratified sampling methodology. In order to reduce the necessary number of samples, each class of any random variable is considered in the same manner (McKay et al. (1979)). First, the marginal distribution or cumulative distribution function X_i is subdivided into N classes $\mathbb{D}_j$ with the same probability

$$P\left[x_i \in \mathbb{D}_j\right] = \frac{1}{N}, \qquad i = 1, ..., n, \quad j = 1, ..., N$$

So N^n hypercubes are created with the probability N^{-n}. Comparing plain Monte Carlo with latin hypercube sampling it is easily seen that latin hypercube sampling covers the space of random variables in a significantly superior way. In particular, plain Monte Carlo methods introduce unwanted correlation into the samples which becomes very pronounced if the number of samples is small.

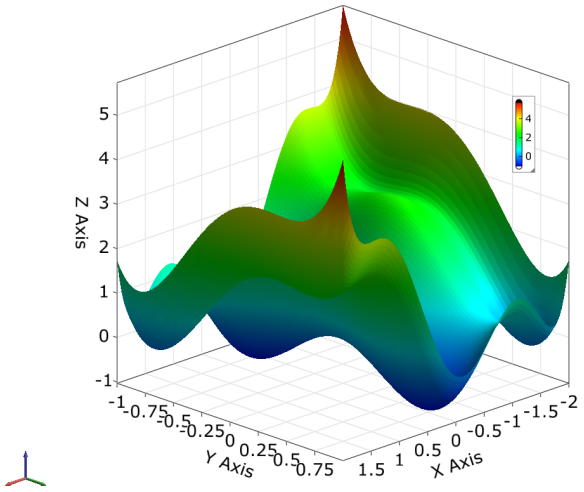


Figure 1: Original model response function $z(x, y)$.

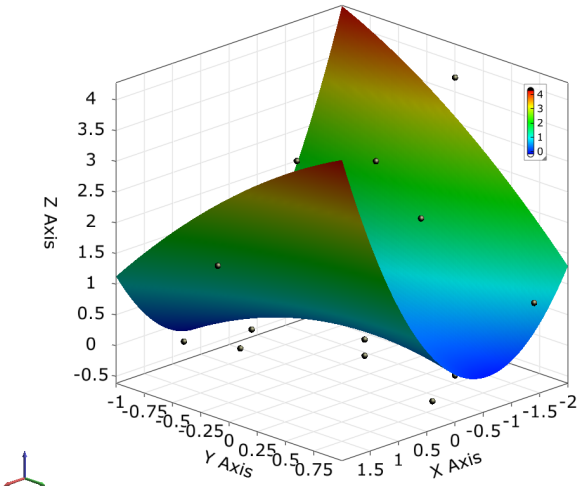


Figure 2: Polynomial least square approximation of a given set of support points. Quadratic approximation function $\hat{z}(x, y)$ using quadratic regression.

1.3 Surrogate models

1.3.1 Polynomial least square approximation

These well-distributed results can be used to create the meta-models. To simulate complex real world systems the response surface methodology is becoming very popular and is widely applied in many scientific areas. In general, response surface methodology is a statistical method for constructing smooth approximations to functions in a multi-dimensional space. In design studies, e.g. design optimization or reliability analysis, a response surface is generated with appropriate approximation functions on a suitable set of discrete support points distributed throughout the design space of interest.

A commonly used approximation method of model responses, objectives, constraints and state functions

$$y(\mathbf{x}) \mapsto \hat{y}(\mathbf{x})$$

is the regression analysis. Usually, the approximation function is a first or second order polynomial (Box and Draper (1987); Myers (1971); Myers and Montgomery (1995)) as shown in Figure 2. As an example in the $(n = 2)$-dimensional case, k-responses $(k = 0, ..., N)$ will be approximated using a least square quadratic polynomial in the following form:

$$y_k = \beta_0 + \beta_1 x_{1k} + \beta_2 x_{2k} + \beta_{11} x_{1k}^2 + \beta_{22} x_{2k}^2 + 2\beta_{12} x_{1k} x_{2k} + \varepsilon_k \tag{1}$$

Herein the term ε_k represents the approximation errors. Equation (1) can be written in matrix notation to

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{2}$$

with

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}; \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_m \end{bmatrix}; \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_{11} \\ \beta_{22} \\ 2\beta_{12} \end{bmatrix}$$

and

$$\mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{21} & x_{11}^2 & x_{21}^2 & x_{11}x_{21} \\ 1 & x_{12} & x_{22} & x_{12}^2 & x_{22}^2 & x_{12}x_{22} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{1N} & x_{2N} & x_{1N}^2 & x_{2N}^2 & x_{1N}x_{2N} \end{bmatrix}$$

The approximate coefficients $\hat{\boldsymbol{\beta}}$ can be calculated using the least square postulate

$$S = \sum_{k=1}^{m} \varepsilon_k^2 = \boldsymbol{\varepsilon}^T \boldsymbol{\varepsilon} \rightarrow \min$$

In due consideration of (2), the least square error S is

$$S(\hat{\boldsymbol{\beta}}) = (\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})^T (\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}) \rightarrow \min \tag{3}$$

An expansion of the right hand side of (3) yields

$$\begin{aligned} S(\hat{\boldsymbol{\beta}}) &= \mathbf{y}^T\mathbf{y} - (\mathbf{X}\hat{\boldsymbol{\beta}})^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\hat{\boldsymbol{\beta}} + (\mathbf{X}\hat{\boldsymbol{\beta}})^T\mathbf{X}\hat{\boldsymbol{\beta}} \\ &= \mathbf{y}^T\mathbf{y} - \hat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\hat{\boldsymbol{\beta}} + \hat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{X}\hat{\boldsymbol{\beta}} \\ &= \mathbf{y}^T\mathbf{y} - 2\hat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{y} + \hat{\boldsymbol{\beta}}^T\mathbf{X}^T\mathbf{X}\hat{\boldsymbol{\beta}} \rightarrow \min \end{aligned}$$

The least square error S may be a minimum in case that the partial gradients

$$\frac{\partial S(\hat{\boldsymbol{\beta}})}{\partial \hat{\boldsymbol{\beta}}} = -2\mathbf{X}^T\mathbf{y} + 2(\mathbf{X}^T\mathbf{X})\hat{\boldsymbol{\beta}} = 0 \tag{4}$$

are zero. Using equation (4), the coefficients $\hat{\boldsymbol{\beta}}$ are given with the linear equation system

$$(\mathbf{X}^T\mathbf{X})\hat{\boldsymbol{\beta}} = \mathbf{X}^T\mathbf{y}$$

In case that $\mathbf{X}^T\mathbf{X}$ is non-singular, the coefficients $\hat{\boldsymbol{\beta}}$ are given by

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

So, the predicted response values are known as

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}}$$

Of course the accuracy of the approximation compared to the real problem has to be checked and verified. For reasonably smooth problems, the accuracy of response surface approximations improves as the number of points increases. However, this effect decreases

with the degree of oversampling. An attractive advantage of the response surface methodology is the smoothing by approximating the subproblem. Especially for noisy problems like crash analysis, for which the catch of global trends is more important and the local noise may not be meaningful, a smoothing of the problem may be advantageous.

However, the recommended area of application is restricted to reasonably smooth problems with a small number of input variables, because linear and quadratic functions are possibly weak approximations near and far from certain support points. And using polynomials higher than second order may only result in higher local accuracy with many sub-optima. Because of that in the last years, different advanced surrogate models have been developed to impove the accuracy and predictability of surrogate models.

1.3.2 Stepwise response surfaces

Within the stepwise response surface approach the response surface is determined by backward stepwise regression (Madsen et al. (1986)), so that the square and cross terms can be absorbed into the model automatically according to their actual contribution, which is calculated by repeated variance analysis. The quadratic backward stepwise regression begins with a model that includes all constant, linear and quadratic terms of the least square polynomial approximation (1). By deleting trivial regressors one at a time this approach develops a stepwise final regression model which only contains regression coefficients which have large effects on the responses. This method is often applied for multiobjective optimization of vehicles. e.g. in Gu et al. (2001); Yu et al. (2002); Liao et al. (2007). Therewith the number n of linear and quadratic terms of the regression model (8) can be dramatically reduced.

1.3.3 Shepard interpolation

A well known method that among all scattered data for an arbitraty number of variables is the Shepard (1968) method. Shepard's method and its generalization (e.g. Du (1996)) is a statistical interpolation averaging the known values of the original function which exactly interpolates the values of the data. The most relevant drawback of this method is that the interpolated values are always constrainted between the maximum and minimum values of the data set.

1.3.4 Kriging models

Kriging was originally developed to model spatial variations in measured geological models (Matheron (1963)). These models are inherently random, and in most cases the variation about a constant mean value is gaussian. Furthermore, the use of Kriging models is also becoming popularity for approximating optimization and stochastic problems (Martin and Simpson (2003)). Kriging is an accurate surrogate model for this type of application due to its flexibility to approximate many different and complex response functions. In addition, it is also suitable for deterministic models since it interpolates the support data points and provide a confidence interval about a prediction result of the approximation model. The flexibility of this method is a result of using a set of parameters to define the model but the process of selecting the best set of parameters for a Kriging model has a few drawbacks. These parameters must be found via a constrained iterative search,

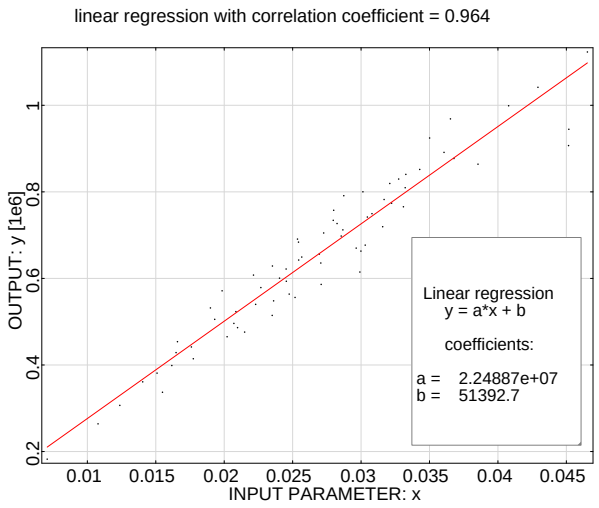


Figure 3: Given set of samples of input parameter $x_i = x$ and response $x_j = y$, linear regression function with correlation coefficient of $\rho_{ij} = 0.964$.

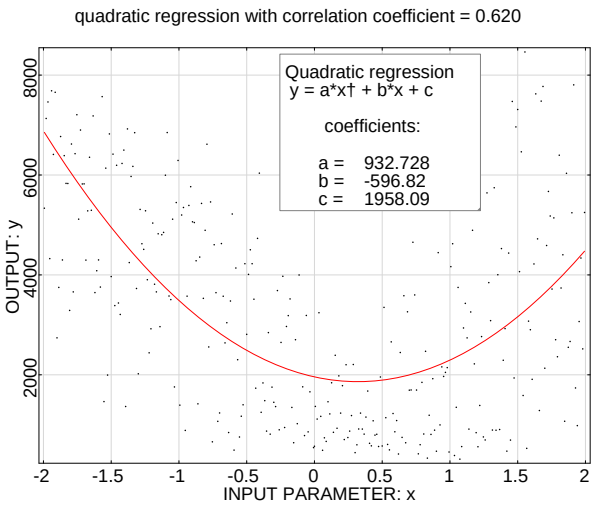


Figure 4: Given set of samples of input parameter $x_i = x$ and response $x_j = y$, quadratic regression function with correlation coefficient of $\rho_{ij} = 0.62$.

a computationally expensive process with respect of the assumption gaussian random process (Jones et al. (1998)).

2 Multivariate Statistic

2.1 Analysis of Correlation

Various statistical analysis procedures are available for the subsequent evaluation of correlation of input parameters and the responses. For example, the coefficients of correlation are calculated from all pairwise combinations of both input variables and response according to:

$$\rho_{ij} = \frac{1}{N-1} \frac{\sum_{k=1}^{N} \left(x_i^{(k)} - \mu_{x_i}\right)\left(x_j^{(k)} - \mu_{x_j}\right)}{\sigma_{x_i}\sigma_{x_j}} \tag{5}$$

The quantity ρ_{ij}, called the linear correlation coefficient, measures the strength and the direction of a linear relationship between two variables, as shown in Figure 3. The linear correlation coefficient is sometimes referred to as the Pearson product moment correlation coefficient. The quadratic coefficients of correlation

$$\rho_{ij} = \frac{1}{N-1} \frac{\sum_{k=1}^{N} \left(\hat{y}^{(k)}(x_i) - \mu_{\hat{y}(x_i)}\right)\left(x_j^{(k)} - \mu_{x_j}\right)}{\sigma_{\hat{y}(x_i)}\sigma_{x_j}}$$

is defined as the linear coefficient of correlation (see Equation (5)) between the least-squares fit of a quadratic regression $\hat{y}(x_i)$ of the variable x_j and x_j themselves on the samples $x_i^{(k)}, x_j^{(k)}$, as shown in Figure 4. A correlation greater than 0.7 is generally described as strong, whereas a correlation less than 0.3 is generally described as weak. These

values can vary based upon the type of data being examined. All pairwise combinations (i, j), values can be assembled into a correlation matrix C_{XX}, as shown in Figure 8.

2.2 Analysis of Prediction

The coefficient of determination

$$R_j^2 = \frac{1}{N-1} \quad \frac{\sum_{k=1}^{N} \left(\hat{y}^{(k)}(x_i) - \mu_{\hat{y}(x_i)}\right) \left(x_j^{(k)} - \mu_{x_j}\right)}{\sigma_{\hat{y}(x_i)} \sigma_{x_j}}$$

with $i = 1, ..., n$ is a value which indicates the shaking (variance or fluctuation) of responses j of the approximation model, depending on the regression model terms. It is a measure thats allow to predict the dependence of a response value from a set of input parameter $i = 1, ..., n$ in a special regression model context. This R^2 value varies between 0 and 1. The coefficient of determination represents the percent of the data that is the closest to the regression model best fit. For example, $R^2 = 0.868$ based on a linear regression, which means that 86.8% of the total variation in the response value y can be explained by the linear relationship between the regression model containing input parameter. However, a high value of R^2 not allways implies a good regression model, because adding of additional approximation model terms increase the coefficient.

Adjusted coefficient of determination This inadequacy leads to the *adjusted* R^2 coefficient

$$R_{adj}^2 = 1 - \frac{N-1}{N-p} \left(1 - R^2\right)$$

who not increase with the number of model terms for a small sample size N. Whereat N is the number of sample points and p the number of regression coefficients. In fact the coefficient decreases with unnecessary model terms. With the comparison of R^2 and R_{adj}^2 it is possible to predict the regression model. A high difference between the coefficients indicates that unnecessary terms are included in the model.

Coefficients of importance An important prediction value to explain the influence of a single input parameter l on a chosen output parameter j, depending on the regression model is the coefficients of importance

$$\mathrm{COI}_{jl} = R_j^2 - \frac{1}{N-1} \quad \frac{\sum_{k=1}^{N} \left(\hat{y}^{(k)}(x_i) - \mu_{\hat{y}(x_i)}\right) \left(x_j^{(k)} - \mu_{x_j}\right)}{\sigma_{\hat{y}(x_i)} \sigma_{x_j}}$$

which an regression model $\hat{y}\left(x_i | i \in \{1, ..., n\} \wedge i \notin \{l\}\right)$. In addition the adjusted coefficients of importance is given by

$$\mathrm{COI}_{jl}^{adj} = 1 - \frac{N-1}{N-p} \left(1 - \mathrm{COI}_{jl}\right)$$

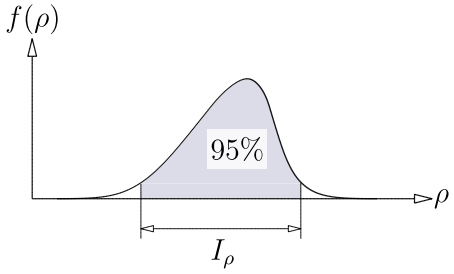


Figure 5: Confidence Interval for coefficient of correlation ρ.

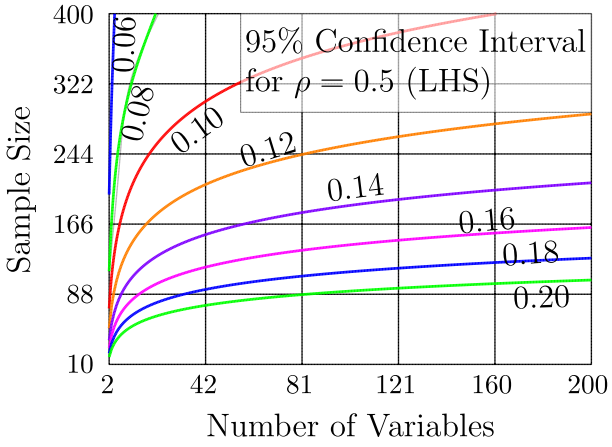


Figure 6: 95% Confidence intervals to estimate a coefficient of correlation of $\rho = 0.5$ using LHS.

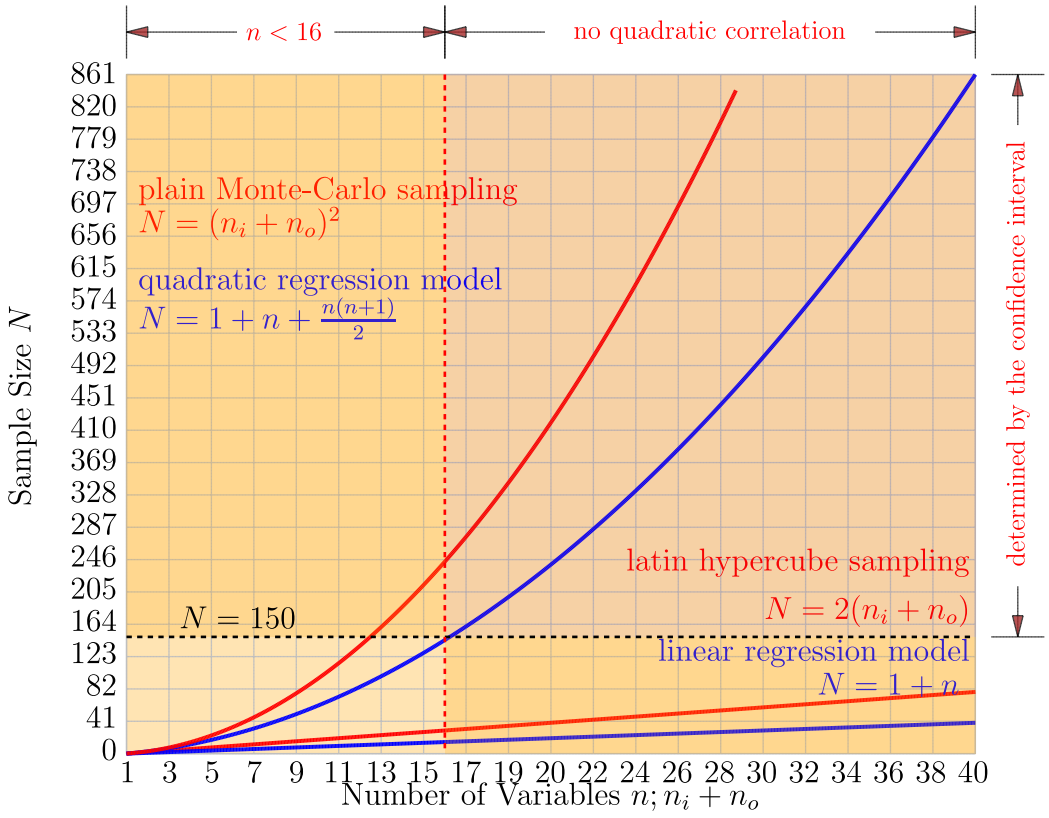


Figure 7: Costs of the robustness evaluation depending on the sampling method, the number of input and oputput parameters n_i, n_o and the regression model to estimate the prediction values. A commonly used maximal sample number is $N = 150$. So, in case of a full polynomial regression model the number of input parameters is restricted to $n < 16$.

2.3 Costs of the robustness evaluation

2.3.1 Minimal sample size

In order to obtain stable statistics for large linear correlation coefficients only, there is a relation between the number of design variables $n = n_i$, the number of responses n_o and the necessary number of samples. The simulation procedure will produce N samples for all parameter x_i denoted by $x_i^k; \quad i = 1 \ldots n; \quad k = 1 \ldots N$. In general it is recommended that the number of samples N be at least equal, but better higher than the number n of design variables.

The recommended minimum number of samples depends on the number of input and oputput parameters and is given by $N = (n_i + n_o)^2$ for plain Monte Carlo sampling and $N = 2(n_i + n_o)$ for latin hypercube sampling, as shown in Figure 7. This rough estimation may be sufficiently for a relative small number of input parameters up to $n < 40$. A more precise estimation is given as a results of a convergence study of the confidence intervals.

2.3.2 Confidence intervals

Of course, the correlation coefficients are random values themselves. Whereby the variance depends on the number of the calculated samples N. According to a specified confidence interval I_p of e.g. 95% the possible lower and upper bounds of the estimated coefficients of correlation ρ_{ij} can be evaluated, as shown in Figure 5.

The confidence intervals for the estimated coefficients of correlation ρ_{ij} in Figure 6 are computed based on the Fisher's z-transformation. The interval for a significance level of α (i.e. a confidence level of $1 - \alpha$) is given by

$$\left[\tanh\left(z_{ij} - \frac{z_c}{\sqrt{N-3}} \right), \ \tanh\left(z_{ij} + \frac{z_c}{\sqrt{N-3}} \right) \right]$$

In this Equation, N is the number of samples used for the estimation of ρ_{ij}. The critical value z_c is computed by using the Bonferroni-corrected value for the significance level $\alpha' = \alpha/k$ with k being the number of confidence tests. The transformed variable z is computed from

$$z_{ij} = \frac{1}{2} \log \frac{1 + \rho_{ij}}{1 - \rho_{ij}} \tag{6}$$

and the critical value z_c is given by

$$z_c = \Phi^{-1}(1 - \alpha'/2) \tag{7}$$

where $\Phi^{-1}(.)$ is the inverse cumulative Gaussian distribution function.

In order to study the effect of latin hypercube sampling on the reduction of statistical uncertainty, a comparison of the estimation errors (standard deviations) of the correlation coefficients is carried out. The most important domain in the range of $2 < n < 200$, coefficient of correlation $\rho = 0.5$ is detailed shown in Figure 6. For example, for $n = 160$ input parameters and a tolerated maximum error of 14 percent according to a 95%-confidence interval the necessary number of samples is $N = 200$ using latin hypercube sampling.

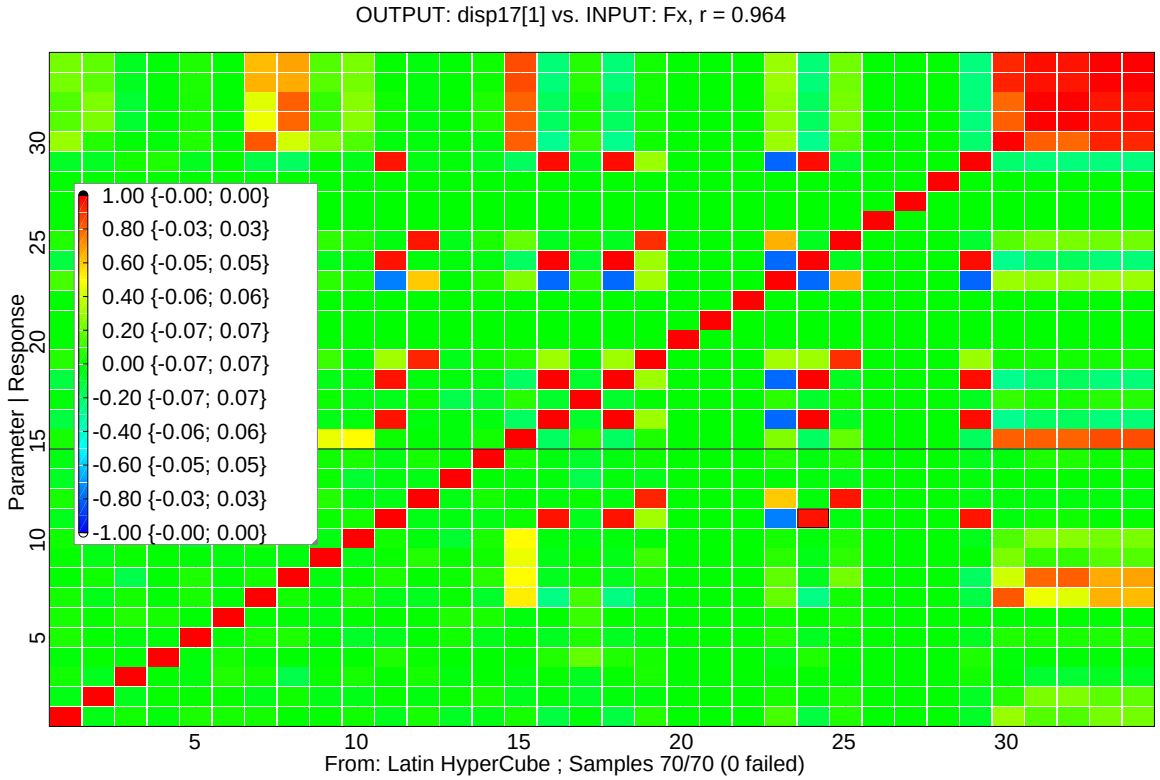


Figure 8: Matrix C_{XX} of the linear correlation coefficients with the possible lower and upper bounds according to a specified confidence interval I_p of 95%.

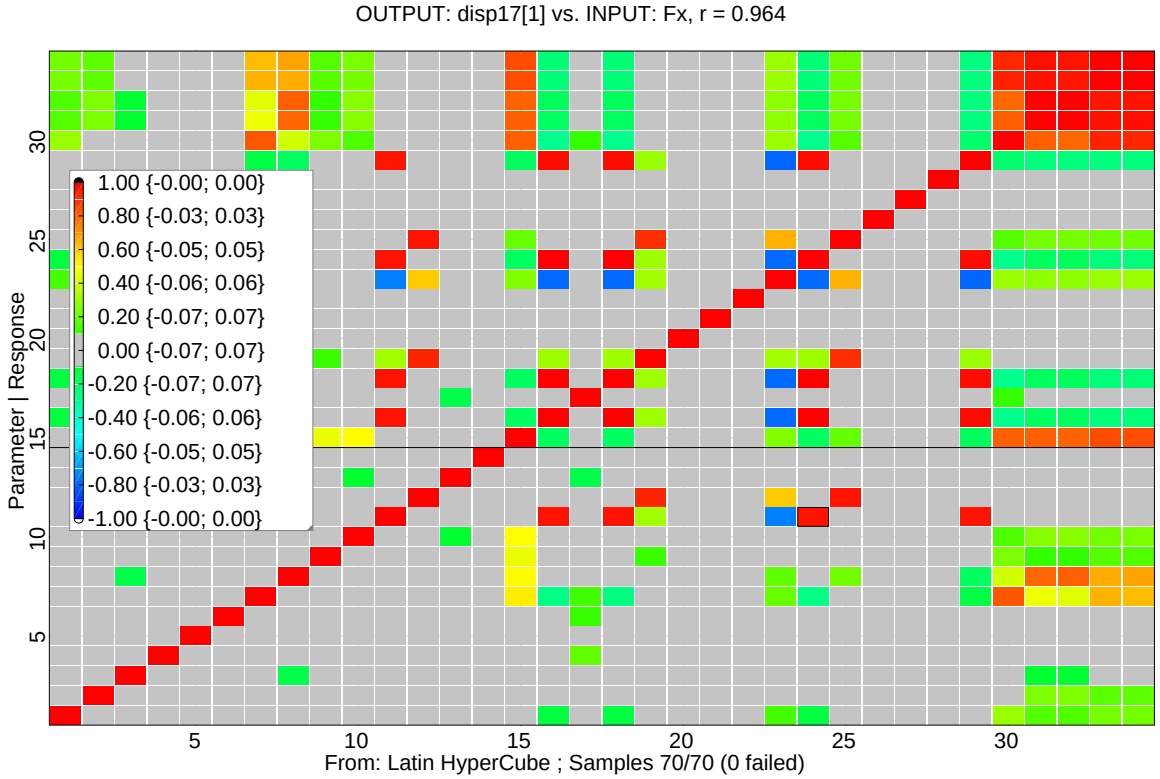


Figure 9: Matrix C_{XX} of the most significance linear correlation coefficients used for reduced polynomial regression models.

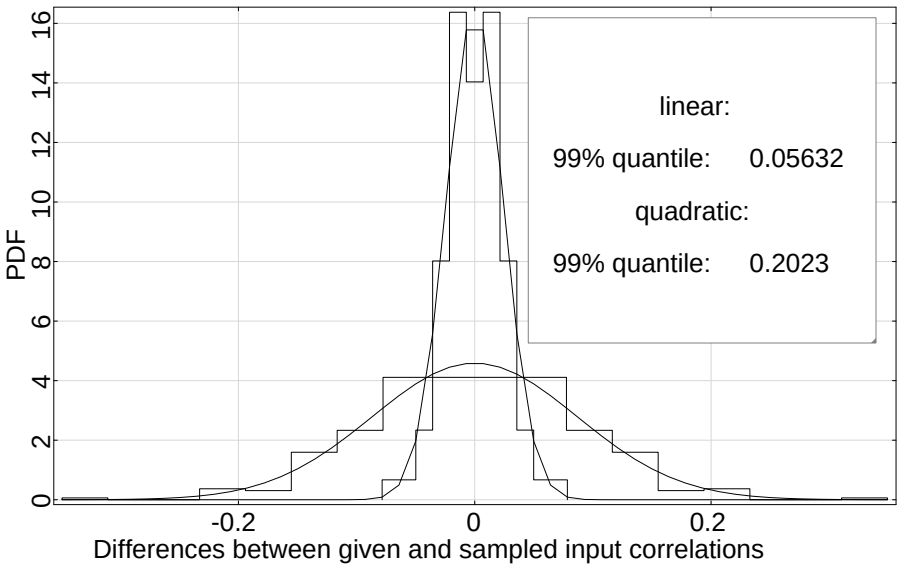


Figure 10: Histogram of the differences between the user-defined and sampled quadratic and linear input correlation matrix C_{XX}. Fitted probability density function with the 99%–fractiles of the coefficients of correlation.

2.3.3 Cost of the regression model

The regression models for the coefficient of importance require a minimum number of samples

$$N = 1 + n + \frac{n(n+1)}{2} \tag{8}$$

depending on the constant, linear and quadratic terms of the regression model. So, in case of a full polynomial regression model and an acceptable maximal sample number of $N = 150$ the number of input parameters is restricted to $n < 16$.

To eliminate this limitation other surrogate models e.g. reduced polynomial regression model, moving least square approximation, artificial neural networks and support vector regression are described in the next sections of this paper.

3 Significance filter and reduced polynomial regression model

The meta-modeling approaches reported in the literature are usually based on the assumption that a simulation model generates a single, i.e. scalar value response. But most complex engineering simulations yield to multiple responses. To reduce the required number of support points a possible approach is to create a separate meta-model for each response individually.

A significance filter to reduce the number n of relevant input parameters is based on the differences between the user-defined and sampled quadratic and linear input correlation matrix C_{XX}. Figure 8 shows the linear correlation matrix as a result of the latin hypercube sampling approach. In this example the user-defined input correlation matrix is simple the

unite matrix. So, the histogram of the differences can be calculated, as shown in Figure 10, and can be fitted by a probability density function with the 95% – 99%-fractiles of the coefficients of correlation. These quantiles are used as a significance filters for the relevant input parameters with a coefficients of correlation larger than these bounds. Result is a reduced polynomial regression model of the responses, as shown in Figure 9.

4 Advanced Moving Least Square Approximation

Moving least square (MLS) functions can approximate locally clustered support point samples with higher local approximation quality. In addition MLS improve the response surface model using additional support points. MLS is formulated as

$$\hat{y}(\mathbf{x}) = \sum_{i=1}^{n_b} h_i(\mathbf{x}) a_i(\mathbf{x}) = \mathbf{h}^T(\mathbf{x})\ \mathbf{a}(\mathbf{x}) \tag{9}$$

with a predefined number of basis terms n_b, a vector of basis functions $\mathbf{h}$ and the associated vector of the coefficients $\mathbf{a}$. Lancaster and Salkauskas (1986) formulates a local MLS approximation as

$$\hat{y}(\mathbf{x}, \mathbf{x}_j) = \sum_{i=1}^{n_b} h_i(\mathbf{x}_j) a_i(\mathbf{x}) = \mathbf{h}^T(\mathbf{x}_j)\ \mathbf{a}(\mathbf{x})$$

with $j = 1, ..., n_s$ support points. The approximate coefficient vector $\mathbf{a}$ can be calculated using the weighted least square postulate

$$\begin{aligned} S(\mathbf{x}) &= \sum_{j=1}^{n_s} w(\mathbf{x} - \mathbf{x}_j) \left(\hat{y}(\mathbf{x}, \mathbf{x}_j) - y(\mathbf{x}_j)\right)^2 \\ &= \sum_{j=1}^{n_s} w(\mathbf{x} - \mathbf{x}_j) \left(\sum_{i=1}^{n_b} h_i(\mathbf{x}_j) a_i(\mathbf{x}) - y(\mathbf{x}_j) \right)^2 \\ &= (\mathbf{H}\mathbf{a} - \mathbf{g})^T \mathbf{W}(\mathbf{x}) (\mathbf{H}\mathbf{a} - \mathbf{g}) \rightarrow \min \end{aligned} \tag{10}$$

with the weighting function $w(\mathbf{x} - \mathbf{x}_j)$ and

$$\begin{aligned} \mathbf{g} &= [y(\mathbf{x}_1) \quad y(\mathbf{x}_2) \quad ... \quad y(\mathbf{x}_{n_s})]^T \\ \mathbf{H} &= [\mathbf{h}(\mathbf{x}_1) \quad \mathbf{h}(\mathbf{x}_2) \quad ... \quad \mathbf{h}(\mathbf{x}_{n_s})]^T \\ \mathbf{h}(\mathbf{x}_j) &= [h_1(\mathbf{x}_j) \quad h_2(\mathbf{x}_j) \quad ... \quad h_{n_b}(\mathbf{x}_j)]^T \\ \mathbf{W}(\mathbf{x}) &= \mathrm{diag}[w(\mathbf{x} - \mathbf{x}_1) \quad w(\mathbf{x} - \mathbf{x}_2) \quad ... \quad w(\mathbf{x} - \mathbf{x}_{n_s})] \end{aligned}$$

The least square error $S(\mathbf{x})$ may be a minimum in case that the partial gradients are zero.

$$\frac{\partial S(\mathbf{x})}{\partial \mathbf{a}} = 0$$

So using the Equation (10) a linear equation system gives an estimation of the coefficient vector $\mathbf{a}$

$$\mathbf{a}(\mathbf{x}) = \mathbf{M}^{-1}(\mathbf{x})\ \mathbf{B}(\mathbf{x})\ \mathbf{g} \tag{11}$$

with

$$\mathbf{M}(\mathbf{x}) = \mathbf{H}^T\ \mathbf{W}(\mathbf{x})\ \mathbf{H}$$
$$\mathbf{B}(\mathbf{x}) = \mathbf{H}^T\ \mathbf{W}(\mathbf{x})$$

Cause the matrix of the basis function $\mathbf{M}(\mathbf{x})$ should be non-singular always a sufficient number of n_s immediate neighbor support points have to be available. The number must be at least as large as number of the basis terms. The Equation (11) inserted in (9) gives the approximation function

$$\hat{y}(\mathbf{x}) = \mathbf{h}^T(\mathbf{x})\ \mathbf{M}^{-1}(\mathbf{x})\ \mathbf{B}(\mathbf{x})\ \mathbf{g}$$

An accurate as possible approximation quality requires a weighting function which is larger than zero $w(\mathbf{x} - \mathbf{x}_j) > 0$ and monotonically decreasing $w(\|\mathbf{x} - \mathbf{x}_j\|)$ inside of a small sub space $\Omega_s \subset \Omega$. So the influence of supports far from the actual coordinates is unimportant. An uniform weighting is given by a symmetry condition $w(\mathbf{x} - \mathbf{x}_j) = w(\mathbf{x}_j - \mathbf{x}) = w(\|\mathbf{x} - \mathbf{x}_j\|)$. Usually, an exponential function is used in this way:

$$w(\|\mathbf{x} - \mathbf{x}_j\|) = \begin{cases} e^{-\left(\frac{\|\mathbf{x} - \mathbf{x}_j\|}{D\alpha}\right)^2} & \|\mathbf{x} - \mathbf{x}_j\| \leq D \\ 0 & \|\mathbf{x} - \mathbf{x}_j\| > D \end{cases} \tag{12}$$

with a constant

$$\alpha = \frac{1}{\sqrt{-\ log\ 0.001}}$$

and a influence radius D to choose. It is obvious that the smaller D the better the response values of the support points fit the given values. But as mentioned above at least n_b support points have to be available in every point to be approximated. Therefore it is possible that a D has to be chosen which leads to a large shape function error at the support points - see Figures 11, 12 and 13. To avoid these problems a new regularized weighting function was introduced by Most and Bucher (2005):

$$w_R(\|\mathbf{x} - \mathbf{x}_j\|) = \begin{cases} \dfrac{\hat{w}_R(\|\mathbf{x} - \mathbf{x}_j\|)}{\sum_{i=1}^{n_s} \hat{w}_R(\|\mathbf{x} - \mathbf{x}_i\|)} & \|\mathbf{x} - \mathbf{x}_j\| \leq D \\ 0 & \|\mathbf{x} - \mathbf{x}_j\| > D \end{cases}, \ 0 < j \leq n_s \tag{13}$$

with

$$\hat{w}_R(d) = \frac{\left(\left(\frac{d}{D}\right)^2 + \varepsilon\right)^{-2} - (1+\varepsilon)^{-2}}{(\varepsilon)^{-2} - (1+\varepsilon)^{-2}}\ ;\ \varepsilon \ll 1 \tag{14}$$

It is recommended by the authors to use the value

$$\varepsilon = 10^{-5}$$

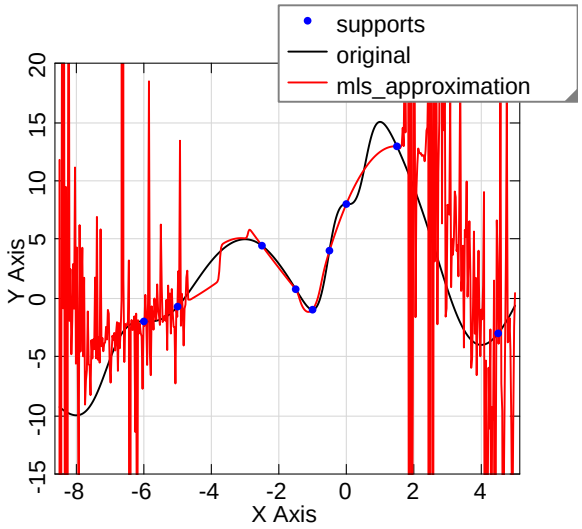


Figure 11: MLS approximation with weighting function (12) and $D = 1$. For some support points ($-2.5 \leq x \leq 0$) the approximation error is very small but for coordinates where the support points have larger distances the shape function is not continuous.

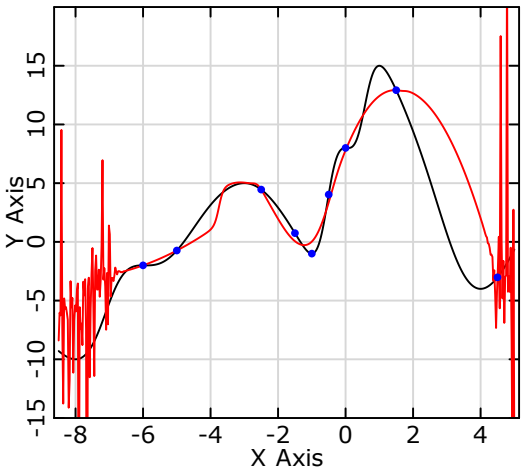


Figure 12: MLS approximation with weighting function (12) and $D = 2$. Now the interval where the shape function is continuous is larger but the error for points with $-2.5 \leq x \leq 0$ increases and for marginally coordinates where are no support points the shape function is still not continuous.

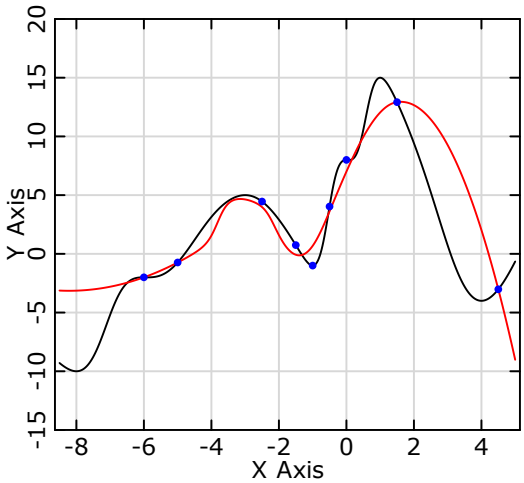


Figure 13: MLS approximation with weighting function (12) and $D = 3$. The shape function is completely continuous in the contemplated interval but the shape function error at the support points is very large.

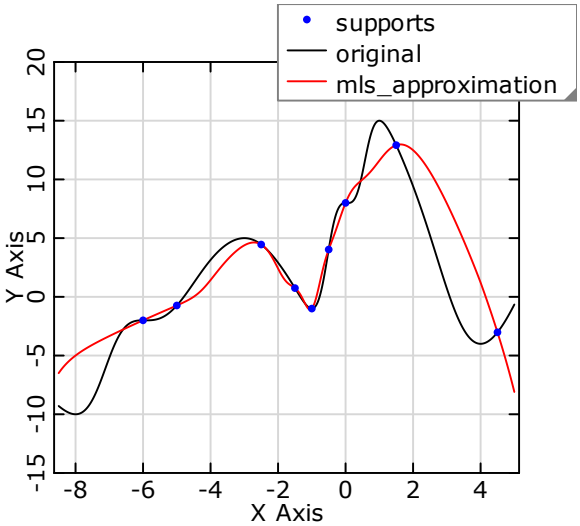


Figure 14: MLS approximation with regularized weighting function (14) and $D = 10$.

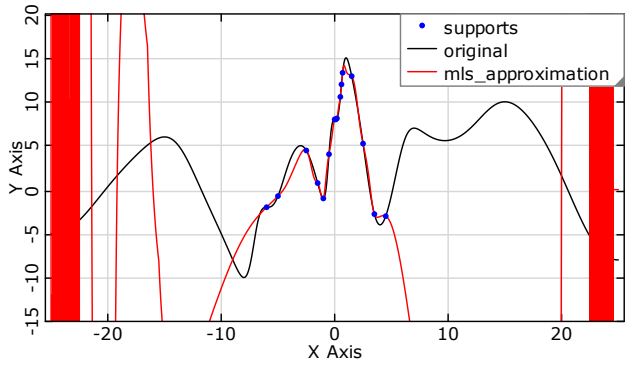


Figure 15: MLS approximation with weighting function (14) and $D = 20$.

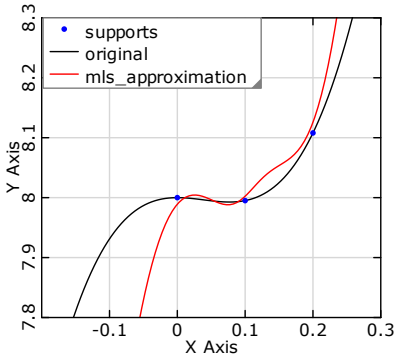


Figure 16: Cut-out of Figure 15.

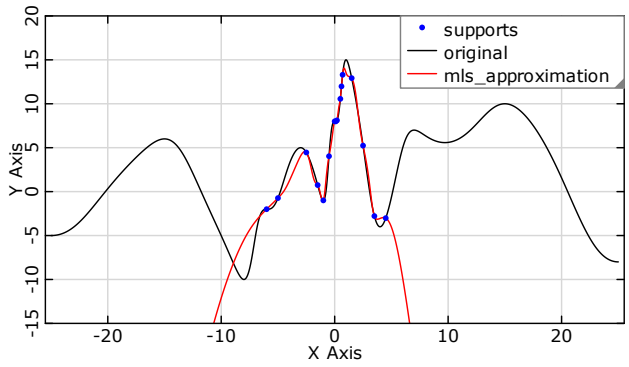


Figure 17: MLS approximation with weighting function (14) and $D = 30$.

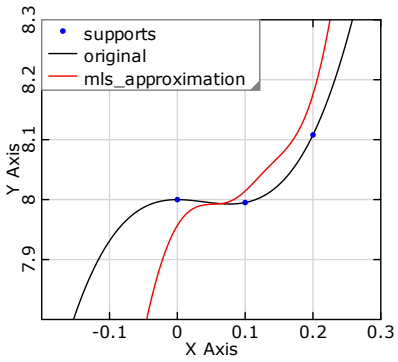


Figure 18: Cut-out of Figure 17.

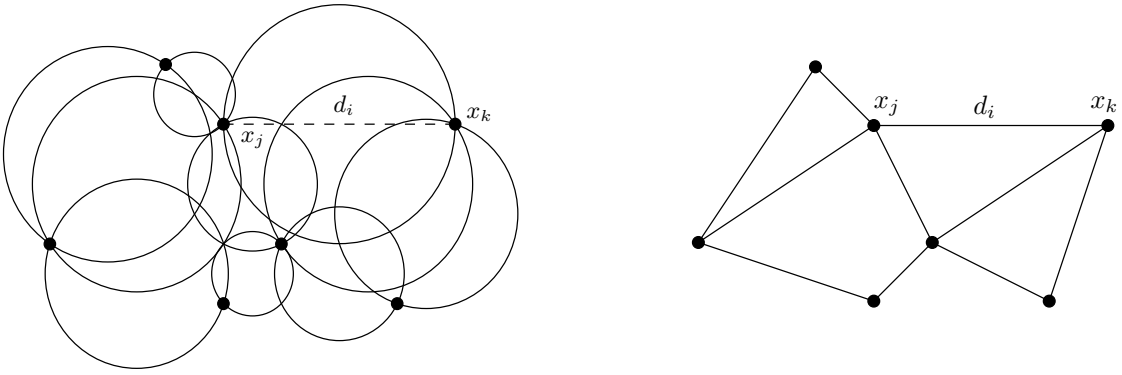


Figure 19: Circles to check the conditions in Equation (15) and the resulting distances.

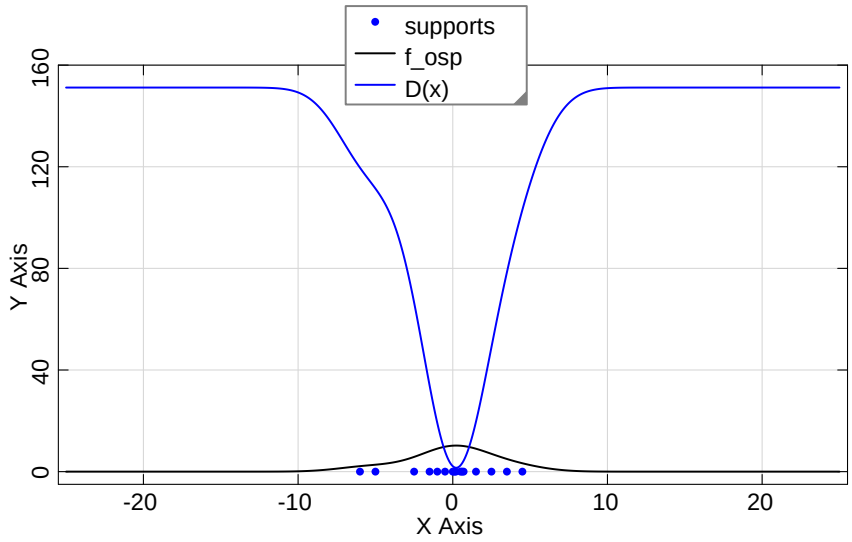


Figure 20: Function representing the occurrence of the supports from the Figures 15 and 17 and the belonging function $d(\mathbf{x})$.

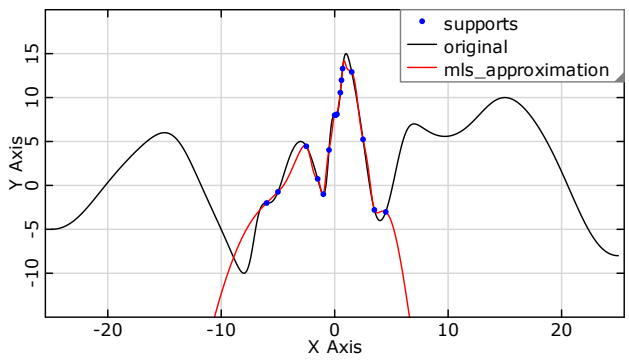


Figure 21: MLS with weighting function (14) and the described function $d(\mathbf{x})$.

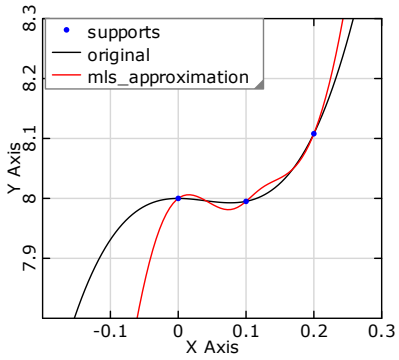


Figure 22: Cut-out of Figure 21.

Equation (13) in combination with Equation (11) can be simplified:

$$\begin{aligned}\mathbf{a}(\mathbf{x}) &= \mathbf{M}^{-1}(\mathbf{x})\ \mathbf{B}(\mathbf{x})\ \mathbf{g} \\ &= [\mathbf{H}^T\ \mathbf{W}(\mathbf{x})\ \mathbf{H}]^{-1}\mathbf{H}^T\ \mathbf{W}(\mathbf{x}) \\ &= \left[\frac{1}{\sum_{i=1}^{n_s} \hat{w}_R(\|\mathbf{x}-\mathbf{x}_i\|)}\ \mathbf{H}^T\ \hat{\mathbf{W}}(\mathbf{x})\ \mathbf{H}\right]^{-1} \frac{1}{\sum_{i=1}^{n_s} \hat{w}_R(\|\mathbf{x}-\mathbf{x}_i\|)}\ \mathbf{H}^T\ \hat{\mathbf{W}}(\mathbf{x}) \\ &= [\mathbf{H}^T\ \hat{\mathbf{W}}(\mathbf{x})\ \mathbf{H}]^{-1}\mathbf{H}^T\ \hat{\mathbf{W}}(\mathbf{x})\end{aligned}$$

Because of that Equation (14) can be used as weighting function instead of the more complex expression in Equation (13). It can be seen in Figure 14 this new regularized weighting function works better than the exponential function. But if the ratio of the minimal distance among the supports to the extent of areas where are no supports becomes worse the same problems occur again - see Figures 15, 16 and 17, 18. The regularized weighting function (14) works very good for this example but if the minimal distance among the support points gets smaller and the areas where are no support points become larger there are still some problems.

As a matter of fact a large D is needed to approximate for coordinates where no support points are around and a small D is needed for coordinates where are a lot of support points in order to reach a minimal approximation error. To comply with this conditions it is necessary to use a function $d(\mathbf{x})$ for the influence radius instead of a constant D. One possibility to get such a function is to properly scale and flip horizontal a function f_{osp}, which represents the occurrence of the support points:

$$\begin{aligned}f_{\text{osp}}(\mathbf{x}) &= \sum_{i=1}^{n_s} e^{-\left(\frac{\|\mathbf{x}-\mathbf{x}_i\|}{c_1}\right)^2} \\ d(\mathbf{x}) &= c_2 - c_3 \cdot f_{\text{osp}}(\mathbf{x})\end{aligned}$$

Whereby c_1 representing a certain value of the distances among the support points and the scaling factors c_2 and c_3. As mentioned several times the distances among the support points play an important role in order to get a suitable function $d(\mathbf{x})$. They are calculated as follows. Let $\mathbb{S} \subseteq \mathbb{R}^n$ be a finite set of support points and $\mathbb{D}_{\text{ist}}(\mathbb{S}) \subseteq \mathbb{R}$ the set of contemplated distances among the support points. It is valid

$$\begin{aligned}\mathbb{D}_{\text{ist}}(\mathbb{S}) = \{\ d \in \mathbb{R} \setminus \{0\} : \exists\ \mathbf{x}_j, \mathbf{x}_k \in \mathbb{S}\ (\ d = \|\mathbf{x}_j - \mathbf{x}_k\| \wedge \\ \forall\ \mathbf{x}_l \in \mathbb{S}\ \left(\left\|\frac{\mathbf{x}_j + \mathbf{x}_k}{2} - \mathbf{x}_l\right\| \geq \frac{d}{2}\right)\ )\ \}\end{aligned} \tag{15}$$

This means that only distances between two nodes $\mathbf{x}_j$ and $\mathbf{x}_k$ are regarded if no other node is closer to their center than themselves. In the 2-dimensional space the contemplated distances are similar to the Delaunay triangulation - see Figure 19. In higher dimensions the check of the conditions in Equation (15) is also simply practicable.

Needed values for the scaling factors of the function $d(\mathbf{x})$ are $\min \mathbb{D}_{\text{ist}}$ and $\max \mathbb{D}_{\text{ist}}$. To avoid that a little change of the support points cause a great change of these values

a probability density function $f_X(x)$ is fitted to the elements of $\mathbb{D}_{\text{ist}}$. The belonging distribution function is $F_X(x)$ and the values of the inverse ${F_X}^{-1}(x)$ for $x = 0.1$ and $x = 0.9$ are used respectively. With

$$d_{\min} = {F_X}^{-1}(0.1), \quad d_{\max} = {F_X}^{-1}(0.9) \quad \text{and} \quad d_{\text{xmax}} = {F_X}^{-1}(0.99)$$

the scaling factors are adopted as follows

$$c_1 = d_{\text{xmax}}, \quad c_2 = d_{\max} \cdot 10^1 \quad \text{and} \quad c_3 = \frac{(d_{\max} - d_{\min}) \cdot 10^1}{\max\limits_{\mathbf{x}} f_{\text{osp}}(\mathbf{x})}$$

Thereby it is factored in that in order to reach a minimal approximation error Most and Bucher (2005) assumed

$$\left(\frac{d_{\min}}{D}\right)^2 \gg \varepsilon$$

which is equivalent to

$$D^2 \ll d_{\min}^2 \cdot 10^5 \quad \text{resp.} \quad D \ll d_{\min} \cdot 10^{\sqrt{5}}$$

and leads to the predefinition

$$\begin{aligned} d(\mathbf{x}_i) &= d_{\min} \cdot 10^1, \quad \text{for} \quad \mathbf{x}_i \quad \text{with} \quad f_{\text{osp}}(\mathbf{x}_i) = \max_{\mathbf{x}} f_{\text{osp}}(\mathbf{x}) \quad \text{and} \\ d(\mathbf{x}_i) &= d_{\max} \cdot 10^1, \quad \text{for} \quad \mathbf{x}_i \quad \text{with} \quad f_{\text{osp}}(\mathbf{x}_i) = 0 \end{aligned}$$

Another problem within optimization and stochastic analysis could be the fact that the approximation for marginal coordinates follow the global trend of the given support points. This may lead to marginal approximation values which differ from the approximation values of the support points for orders of magnitudes - see Figure 30. Then it could be useful to add some support points (border points) to force the approximation for marginal coordinates to averaged values - see Figure 31. Additional support points $\mathbb{S}_{\text{add}} \subseteq \mathbb{R}^n$ are introduced as follows: For given lower and upper bounds of coordinates $\mathbf{b}_l, \mathbf{b}_u \in \mathbb{R}^n$ it is valid

$$\mathbb{S}_{\text{add}} = \mathbb{S}_{\text{marg}} \cup \mathbb{S}_{\text{inn}}$$

with

$$\begin{aligned} \mathbb{S}_{\text{marg}} = \{\, \mathbf{x} \in \mathbb{R}^n : \forall\ 0 < i \leq n\ \exists\ k \in \mathbb{N},\ 0 \leq k < \frac{b_u^i - b_l^i}{c_4} \\ (\ x^i = b_l^i + k \cdot c_4\ \vee\ x^i = b_u^i\ ) \\ \wedge \exists\ 0 < j \leq n\ (\ x^j = b_l^j \vee x^j = b_u^j\ )\ \} \end{aligned} \tag{16}$$

with c_4 as a value which depends on the maximal distance between the edges of the regarded space characterized by $\mathbf{b}_l$ and $\mathbf{b}_u$ and the support points.

$$\begin{aligned} \mathbb{S}_{\text{inn}} = \{\, \mathbf{x} \in \mathbb{R}^n : \exists\ \mathbf{z} \in \mathbb{S}_{\text{marg}},\ \mathbf{s} \in \mathbb{S}(\mathbf{z}),\ k \in \mathbb{N}^+ \\ (\ \mathbf{x} = \mathbf{z} + (\mathbf{s} - \mathbf{z})(1 - \frac{1}{2^k})\ \wedge \frac{\|\mathbf{s} - \mathbf{z}\|}{2^k} > c_5\ )\ \} \end{aligned} \tag{17}$$

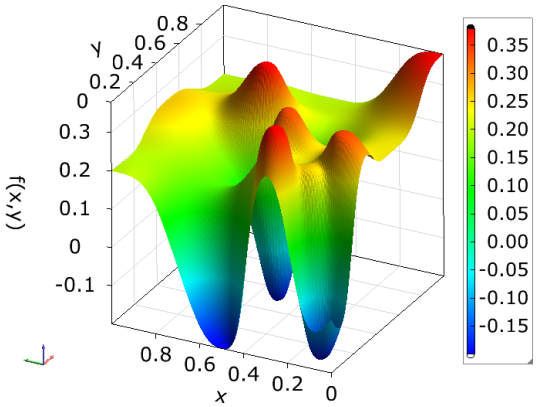


Figure 23: Original function

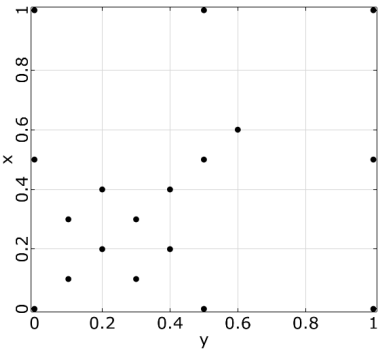


Figure 24: Support points

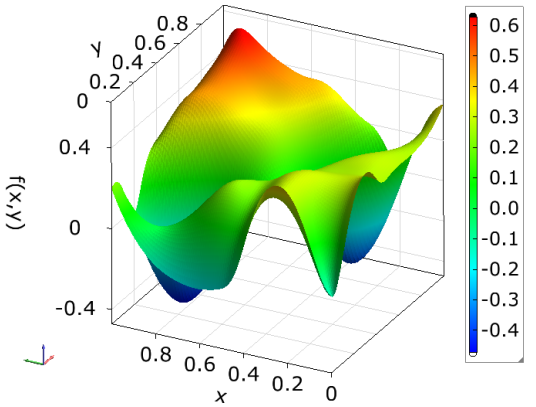


Figure 25: MLS approximation of Figure 23 with supports of Figure 24, weighting function (12) and $D = 0.6$

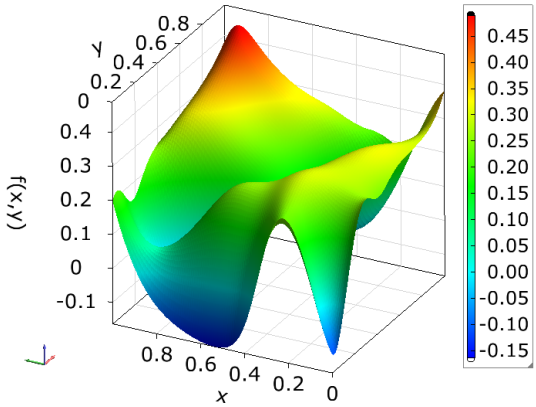


Figure 26: MLS approximation of Figure 23 with supports of Figure 24, weighting function (12) and $D = 0.7$

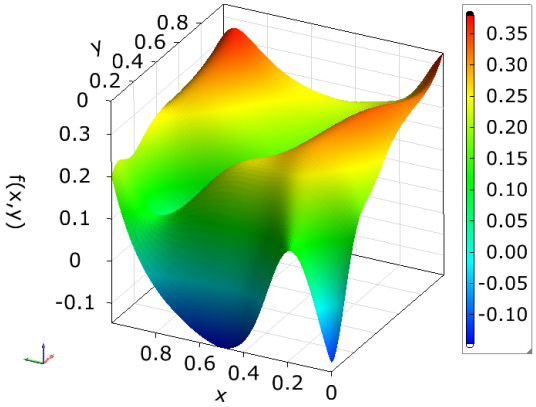


Figure 27: MLS approximation of Figure 23 with supports of Figure 24, weighting function (12) and $D = 0.8$

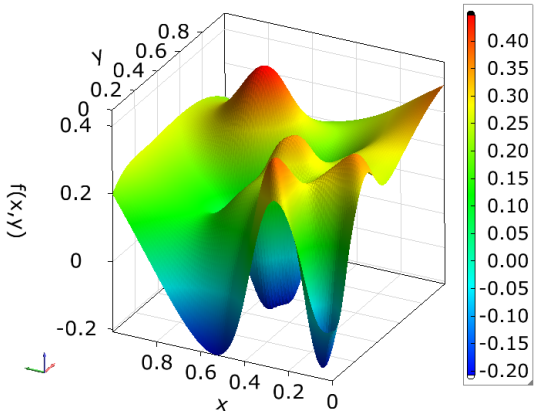


Figure 28: MLS approximation of Figure 23 with supports of Figure 24, weighting function (14) and $D = 3$

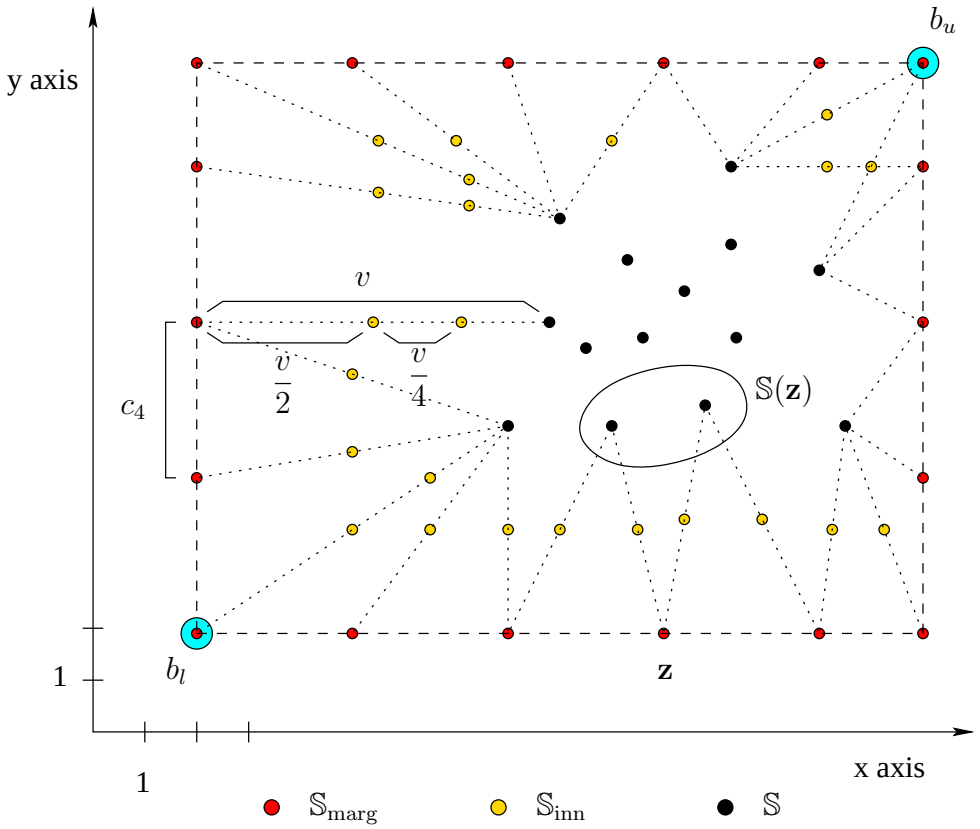


Figure 29: Adding support points with Equations (16) and (17) and $c_5 \approx 3.3$.

with

$$\mathbb{S}(\mathbf{z}) = \{\ \mathbf{s} \in \mathbb{S} : \forall\ \mathbf{x} \in \mathbb{S}\ (\ \|\mathbf{z} + \mathbf{s} - 2\mathbf{x}\| \geq \|\mathbf{z} - \mathbf{s}\|\ )\ \}$$

This simply means that a net of support points with marginal coordinates are added around the given supports ($\mathbb{S}_{\text{marg}}$) and, if it is necessary, originated in this points further support points which lead to the given (inner) supports are also added ($\mathbb{S}_{\text{inn}}$) - see Figure 29 for $n = 2$. The conditions for the elements in $\mathbb{S}(\mathbf{z})$ are equivalent to the conditions for the elements in $\mathbb{D}_{\text{ist}}(\mathbb{S})$ that is $\mathbf{s} \in \mathbb{S}(\mathbf{z}) \rightarrow \|\mathbf{s} - \mathbf{z}\| \in \mathbb{D}_{\text{ist}}(\ \mathbb{S} \cup \{\mathbf{z}\}\ )$.

The values for the additional support points are calculated with the Shepard interpolation. Whereby the weighted interpolation function of the response surface is

$$\hat{y}(\mathbf{x}) = \frac{\sum_{i=1}^{n} y(\mathbf{x}_i) \left(\frac{1}{\|\mathbf{x} - \mathbf{x}_i\| + \epsilon} \right)^m}{\sum_{i=1}^{n} \left(\frac{1}{\|\mathbf{x} - \mathbf{x}_i\| + \epsilon} \right)^m} \quad m = 1, ..., 5$$

The smoothing of the interpolation is numerically controlled by the smoothing exponent m.

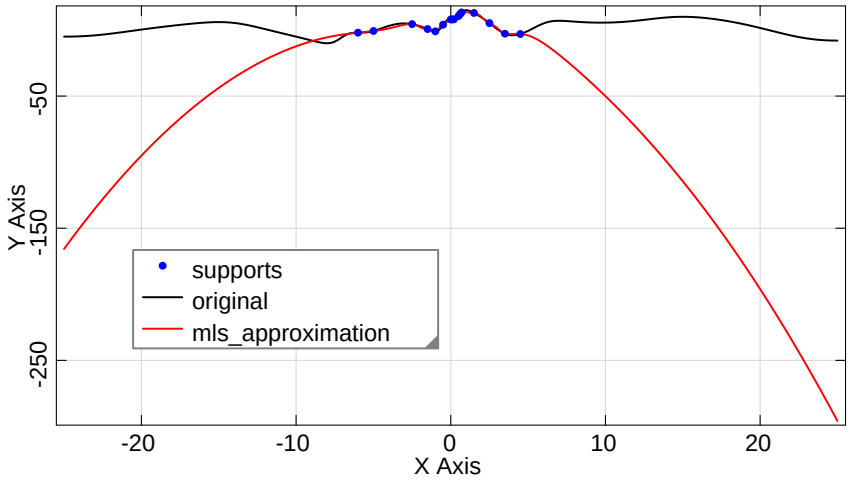


Figure 30: Figure 21 with visualizing the whole values range

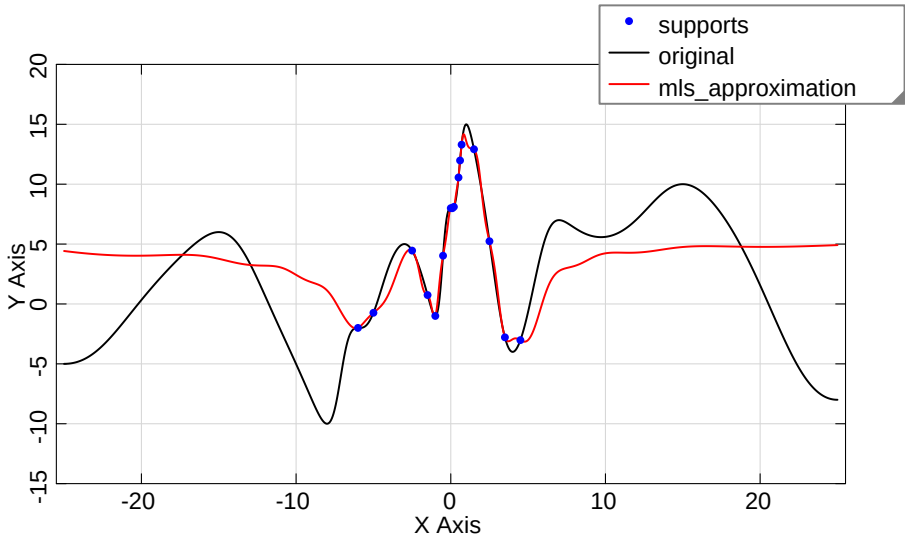


Figure 31: Example of Figure 21 with additional support points (border points)

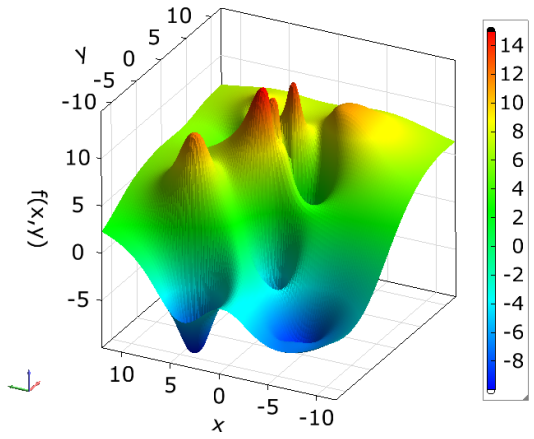


Figure 32: Original function

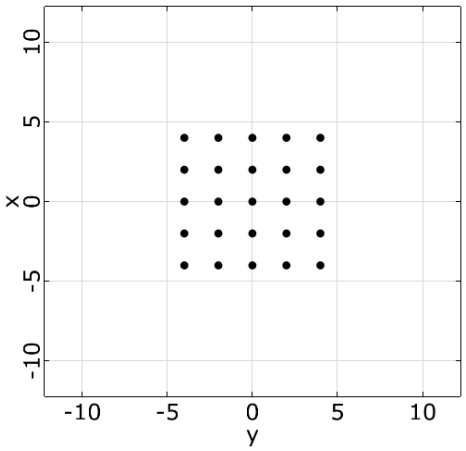


Figure 33: Support points

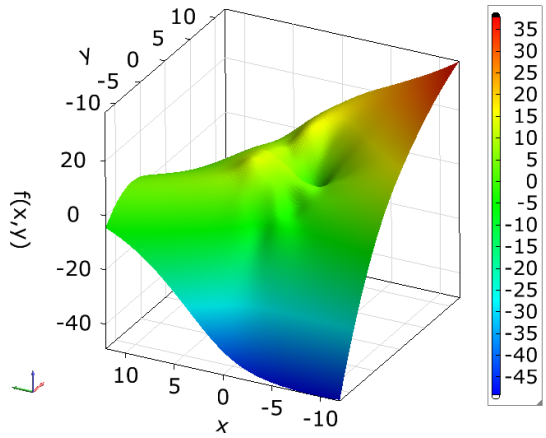


Figure 34: MLS approximation of Figure 32 with supports of Figure 33

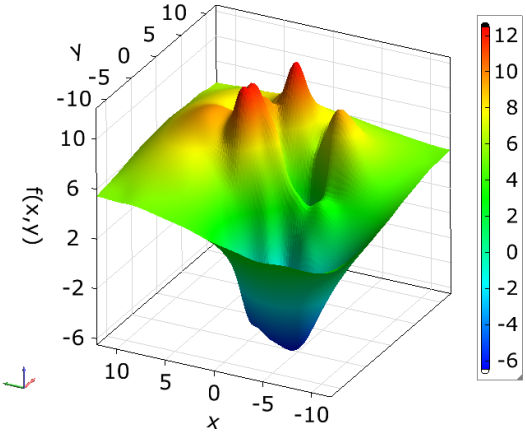


Figure 35: MLS approximation of Figure 32 with supports of Figure 33 and additional supports (border points) like it is shown in Figure 29

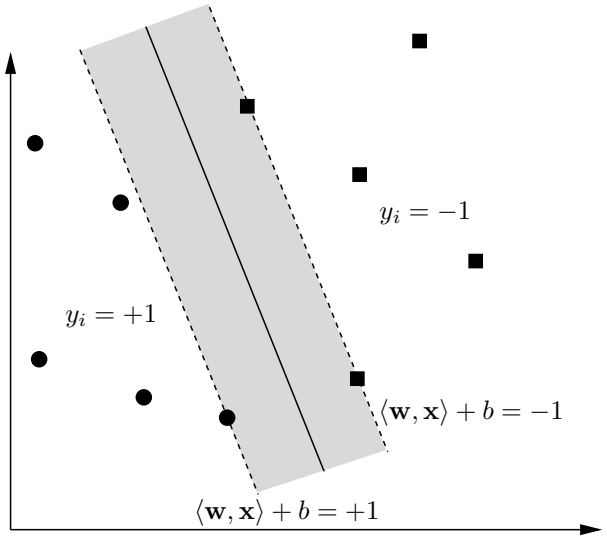


Figure 36: SVM: linear separation by a hyperplane

5 Support Vector Machines

5.1 Classification

A very efficient tool for classification purposes are Support Vector Machines (SVM), which is a method from the statistical learning theory. This method was firstly proposed by Vapnik and Chervonenkis (1974) and became popular in the last decade. Fundamental publications from this period are Cortes and Vapnik (1995), Vapnik (1995) and Schoelkopf and Smola (2001). The algorithmic principle is to create a hyperplane, which separates the data into two classes by using the maximum margin principle.

The linear separator is a hyperplane which can be written as

$$f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + \alpha_0 \tag{18}$$

where $\mathbf{w}$ is the parameter vector that defines the normal to the hyperplane and α_0 is the threshold. In Figure 36 a linear separation is shown for a set of points. The two classes are associated with -1 and $+1$. The SVM principle is to maximize the distance between the hyperplane and the two classes. This can be seen in Figure 36. This principle can be written as an optimization problem

$$\max_{w,b} \min_i \{ \|\mathbf{x} - \mathbf{x}_i\| : \langle \mathbf{w}, \mathbf{x} \rangle + \alpha_0 = 0 \} \tag{19}$$

where

$$\min_i \{ \|\mathbf{x} - \mathbf{x}_i\| : \langle \mathbf{w}, \mathbf{x} \rangle + \alpha_0 = 0 \} \tag{20}$$

is the minimum distance from the training points to the hyperplane. By assuming that the minimal distance is equal to one

$$\min_{i=1..n} |\langle \mathbf{w}, \mathbf{x}_i \rangle + \alpha_0| = 1 \tag{21}$$

we obtain for the margin width

$$\Delta = \frac{2|\langle \mathbf{w}, \mathbf{x}_i \rangle + \alpha_0|}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|} \tag{22}$$

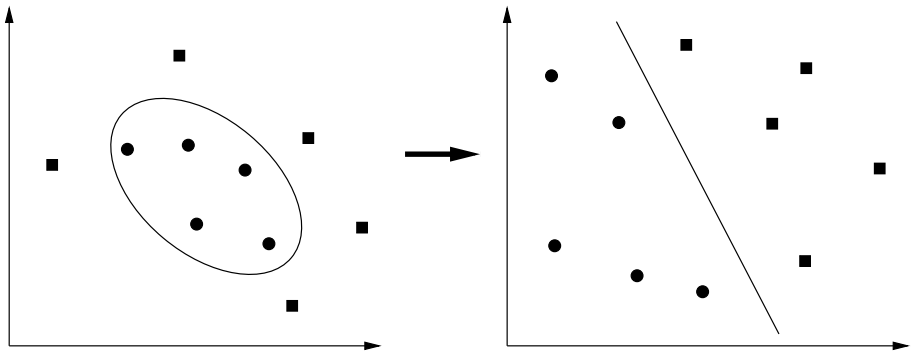


Figure 37: SVM: nonlinear projection into feature space

By introducing Lagrange multipliers $\alpha_i \geq 0$ we obtain the following unconstraint form of the problem

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \quad \text{with} \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{23}$$

whereby only the training points for which the Lagrange multipliers are strictly positive $\alpha_i > 0$, the so-called support vectors, are needed for the function evaluation

$$\mathbf{w} = \sum_{j=1}^{s} \alpha_j y_j \mathbf{x}_j \tag{24}$$

with $s < n$.

For nonlinear separable classes the training data are mapped nonlinearly into a higher-dimensional feature space and a linear separation is constructed there. This is illustrated in Figure 37. The transformation $\psi(\mathbf{x})$ which is realized as an inner product

$$f(\mathbf{x}) = \sum_{i=1}^{s} \alpha_i y_i \langle \psi(\mathbf{x}_i), \psi(\mathbf{x}) \rangle + \alpha_0 \tag{25}$$

can be substituted by a kernel function

$$K(\mathbf{x}, \mathbf{y}) = \langle \psi(\mathbf{x}), \psi(\mathbf{y}) \rangle \tag{26}$$

which leads finally to the expression

$$f(\mathbf{x}) = \sum_{i=1}^{s} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + \alpha_0 \tag{27}$$

where explicit knowledge of the nonlinear mapping is not needed. Often used kernel types are the Gaussian kernel

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|}{2D^2}\right) \tag{28}$$

and the polynomial kernel

$$K(\mathbf{x}, \mathbf{y}) = (\langle \mathbf{x}, \mathbf{y} \rangle + \theta)^p \tag{29}$$

During the training of the support vector machines the Lagrange multiplier of the training points have to be determined by minimizing the primal objective function obtained from Eq.(19) which reads

$$L_p(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{s}\sum_{j=1}^{s} y_i y_j \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) - \sum_{i=1} \alpha_i \tag{30}$$

Many algorithms can be found in literature, see Schoelkopf and Smola (2001). We use one of the fastest methods, the sequential minimal optimization algorithm proposed by Platt (1998) for this training. In this algorithm the Lagrange multipliers will be updated pair-wisely be solving the linear constraint conditions.

5.2 Regression

Regression based on Support Vector Machines was introduced by Drucker et al. (1997). In Smola and Schoelkopf (1998) and Smola and Schoelkopf (2004) a detailed introduction is published. In the SVR approach an error tolerance function

$$L_\epsilon(y) = \begin{cases} 0 & |f(\mathbf{x}) - y| < \epsilon \\ |f(\mathbf{x}) - y| - \epsilon & |f(\mathbf{x}) - y| \geq \epsilon \end{cases} \tag{31}$$

which is called ϵ-insensitive loss function is defined. The optimization task is defined as

$$\text{minimize} \frac{1}{n}\sum_{i=1}^{n} |f(\mathbf{x}_i, \mathbf{w}) - y_i|_\epsilon + \|\mathbf{w}\|^2. \tag{32}$$

The output of the Support Vector Regression reads

$$f(\mathbf{x}) = \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) K(\mathbf{x}_i, \mathbf{x}) + \alpha_0 \tag{33}$$

where α_i^* and α_i are the Lagrange multipliers. The primal form of the objective function reads

$$\begin{aligned} L_p(\boldsymbol{\alpha}^*, \boldsymbol{\alpha}) =& \epsilon \sum_{i=1}^{n} (\alpha_i^* + \alpha_i) - \sum_{i=1}^{n} y_i (\alpha_i^* - \alpha_i) \\ &+ \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) K(\mathbf{x}_i, \mathbf{x}_j) \end{aligned} \tag{34}$$

subjected to the constraints

$$\sum_{i=1}^{n} (\alpha_i^* - \alpha_i) = 0, \quad 0 \leq \alpha_i^*, \alpha_i. \tag{35}$$

Again we use the sequential minimal optimization algorithm for the determination of the Lagrange multipliers.

For deterministic data point values the error tolerance ϵ is chosen very small to represent these values with high accuracy. The kernel radius is taken as the maximum possible value by fulfilling the specified error tolerance. If noisy data should be approximated a

very small ϵ would lead to a small maximum kernel radius and consequently to a strong over-fitting effect in the approximation function. This is shown in the Figures 38 and 39. For an efficient application for noisy data the optimal error tolerance has to be determined. This can be done by using additional information about the noise level or, if these informations are not available, be an additional test data set, where ϵ is modified in a given interval and the optimal value concerning the test data error is taken. In Figure 40 the approximation function for such an optimal ϵ is shown.

6 Artifical neural networks

6.1 Introduction

Artificial neural networks are inspired by biological counterparts. The brain consist of a large number (10^{11}) of highly connected elements (approximately 10^4 connections per element) - neurons. In our simplified approach, a neuron can be considered as consisting of three parts - the dendrites which serve as sensors for electrical signals and carry it into the cell body, the cell body sums up all the inputs and processes the information and finally the axon which transmits the output via the synapsis to other neurons. Within this complex system, a large number of complex information can be stored. Learning during lifetime consists of a modification of the connections and a varying weight of each connection before being processed in the cell body. Artificial neural networks are used in many areas of engineering. The main applications can be divided into pattern recognition and regression.

6.2 Multilayer Perceptron

6.2.1 Layout

The multilayer perceptron is one of the widely used artificial neural networks, especially suitable for regression analysis. It consist of an input layer a certain number of hidden layers and an output layer. Each neuron in the input layer represents a single input parameter. The neurons of the input layer are connected to neurons of the first hidden layer. The number of hidden layers and the number of neurons is variable and should be chosen with respect to the complexity of the system response and the number of available training patterns. Only forward connections are allowed in a multilayer perceptron. The neurons in the final output layer represent the output parameter of the regression model. A schematic drawing of the layout is given in Fig.41 The output a_i^l of neuron i in layer l is calculated as

$$a_i^l = h\left(\sum_{j=1}^{N_i^l} w_{ji}^{l-1} v_j^{l-1} + b_i^l\right), \tag{36}$$

where h is the activation function, N_i^l is the number of connections to the previous layer, w_{ji}^{l-1} corresponds to the weights of each connection and b^l is the bias, which represents the constant part in the activation function. In Fig.42 commonly used activation functions are illustrated.

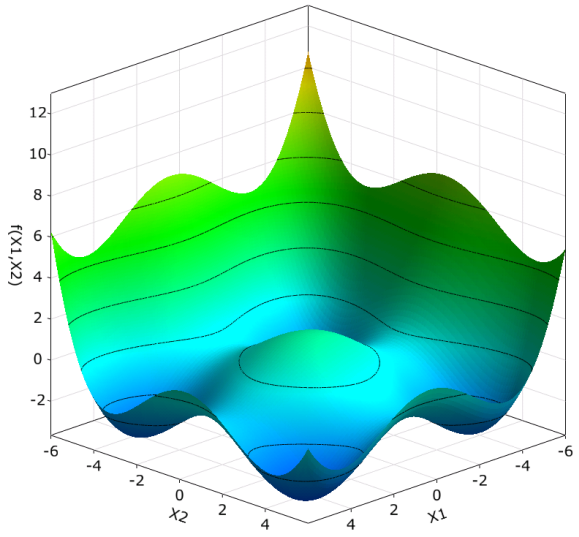


Figure 38: SVR approximation of deterministic data ($\epsilon = 0.001$; $D = 6.00$)

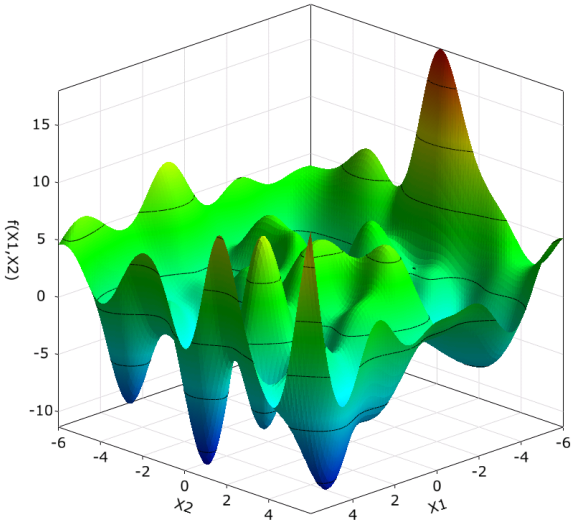


Figure 39: SVR approximation of noisy data with over-fitting ($\epsilon = 0.001$; $D = 1.81$)

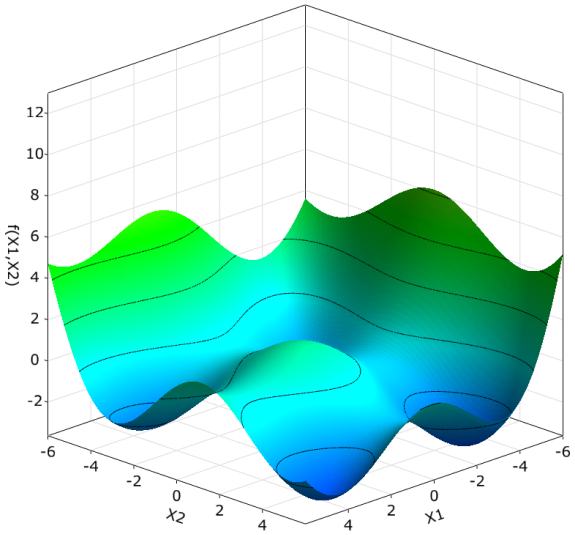


Figure 40: SVR approximation of noisy data with automatic error tolerance ($\epsilon = 0.16$; $D = 6.50$)

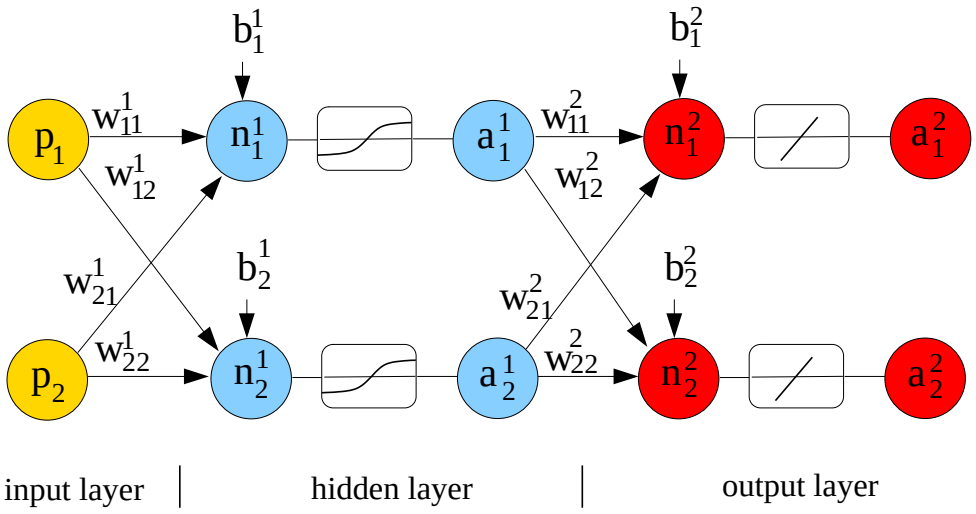


Figure 41: general layout of a multilayer perceptron

6.3 Training

The training of the neural network is an optimization procedure, in which the weights and biases of the neural network are determined. For this purpose, a certain a number of training samples M with corresponding pairs of inputs $\boldsymbol{p}_i$ and outputs $\boldsymbol{o}_i$ is required. The mean square error F, which is the average value of the difference between the approximated response and the outputs, is used as objective in the optimization procedure.

$$F(\boldsymbol{w},\boldsymbol{b}) = \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{e}_i \tag{37}$$

$$\boldsymbol{e}_i = |\boldsymbol{a}(\boldsymbol{p_i}) - \boldsymbol{o}_i|^2 \tag{38}$$

In general, different training algorithms can be applied to solve this optimization procedure. The most important are the standard back-propagation algorithm Rumelhart et al. (1986), RPROP Riedmiller and Braun (1993), the conjugate gradient method Johansson et al. (1992) and the scaled conjugate gradient algorithm Moller (1993). In this paper, the Levenberg Marquardt algorithm Hagan and Menhaj (1994) has been used, since for small systems with up to 1000 free parameters it was found to be faster compared to other methods. For all these methods, the gradient $\boldsymbol{g}$ of the objective function F with respect to the free parameters $\boldsymbol{x}$ (weights $\boldsymbol{w}$ and biases $\boldsymbol{b}$) is to be calculated. This can be performed with a variation of the back-propagation algorithm Hagan et al. (1996):

$$\boldsymbol{G} = \frac{\partial F}{\partial x_j} = \frac{2}{M}\sum_{q=1}^{M}\sum_{i=1}^{N} e_i^q(\boldsymbol{x})\frac{\partial e_i^q(\boldsymbol{x})}{\partial x_j} = \frac{2}{M}\sum_{q=1}^{M}\left[\boldsymbol{J}^q(\boldsymbol{x})\right]^T \boldsymbol{e}^q(\boldsymbol{x}) \tag{39}$$

$$\boldsymbol{J}^q(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial e_1^q(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial e_1^q(\boldsymbol{x})}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial e_N^q(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial e_N^q(\boldsymbol{x})}{\partial x_n} \end{bmatrix}, \tag{40}$$

where $\boldsymbol{J}$ describes the sensitivity of the outputs with respect to the free parameters and N is the dimension of the output vector. The Hessian can be expressed in a similar way

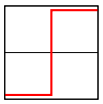
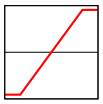
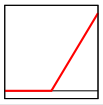
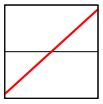
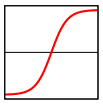
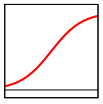
hard limit		$y = -1 \quad x < 0$ $y = +1 \quad x \geq 0$
saturating linear		$y = -1 \qquad x < -1$ $y = x \qquad -1 \leq x < -1$ $y = +1 \qquad x \geq 0$
positive linear		$y = 0 \quad x < 0$ $y = x \quad x \geq 0$
linear		$y = x$
hyperbolic tangent sigmoid		$y = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
log-sigmoid		$y = \frac{1}{1 + e^{-x}}$

Figure 42: activation functions commonly used in multilayer perceptrons

as

$$\boldsymbol{H} = \frac{\partial^2 F}{\partial x_j \partial x_k} = 2 \sum_{q=1}^{M} \sum_{i=1}^{N} \left[\frac{\partial e_i^q(\boldsymbol{x})}{\partial x_k} \frac{\partial e_i^q(\boldsymbol{x})}{\partial x_j} + e_i^q(\boldsymbol{x}) \frac{\partial^2 e_i^q(\boldsymbol{x})}{\partial x_j \partial x_k} \right] \tag{41}$$

$$= \frac{2}{M} \sum_{q=1}^{M} \left\{ \left[\boldsymbol{J}^q(\boldsymbol{x}) \right]^T \boldsymbol{J}^q(\boldsymbol{x}) + 2\boldsymbol{S}^q(\boldsymbol{x}) \right\} \tag{42}$$

$$\tilde{\boldsymbol{H}} = \frac{2}{M} \sum_{q=1}^{M} \left\{ \left[\boldsymbol{J}^q(\boldsymbol{x}) \right]^T \boldsymbol{J}^q(\boldsymbol{x}) \right\}. \tag{43}$$

If we assume $\boldsymbol{S}^q$ to be small, the exact Hessian $\boldsymbol{H}$ can be replaced by $\tilde{\boldsymbol{H}}$. As a result, the update of the parameters in the Newton-iteration can be written as

$$\Delta \boldsymbol{x}^{(k)} = -\tilde{\boldsymbol{H}}^{-1} \boldsymbol{G}. \tag{44}$$

This approach requires the approximated Hessian $\tilde{\boldsymbol{H}}$ to be invertible. However, this cannot be assured, especially if a high number of neurons in the hidden layer is used and the size of the training set is small. In order to overcome this problem, an additional scalar parameter μ is added to all the diagonal elements of $\tilde{\boldsymbol{H}}$. In the limit case of $\mu = 0$ the algorithm converges to the Newton method (with the approximated Hessian), whereas for a large parameter μ the update can be approximated by a steepest descent algorithm with learning rate $\frac{1}{2\mu}$:

$$\Delta \boldsymbol{x}^{(k)} \approx -\frac{1}{2\mu} \boldsymbol{G}^{(k)} \quad \text{, for large } \mu. \tag{45}$$

The parameter μ is initially set to a small value (e.g. 0.01), the update $\Delta \boldsymbol{x}^{(k)}$ and the mean square error $F(\boldsymbol{x}^{(k)} + \Delta \boldsymbol{x}^{(k)})$ are calculated. If a reduction of the mean square error

is obtained, the next iteration step $k+1$ is performed with $\mu^{(k+1)} = \mu^{(k)}/2$, otherwise the iteration step is repeated with a $\mu^{(k)}$ increased by a factor of 2.
The initial weights have been calculated from a uniform distribution in the interval $[-s, s]$ according to Haykin (1999), where s is given by

$$s = \sqrt{\frac{3}{L_i}} \tag{46}$$

and L_i is the number of input connections of the neuron. The initial biases are set to zero. Due to the random initialization the training process was repeated 10 times to decrease the influence of the starting values of the weights.

In order to reduce the influence of over-fitting, an early stopping criterion Amari et al. (1996) is applied. The data set is divided into a training and validation set. The update of the weights and biases is stopped, if the required accuracy of the mean square error for the training samples is reached, the mean square error for the validation set starts to increase or the norm of the gradient of the objective function is smaller than a prescribed value.
In general, two strategies for learning can be applied - sequential learning and batch mode learning. In the first approach, the training samples are presented to the learning algorithm separately in a stochastic order (e.g. randomly) and the free parameters are updated for each of these training samples to reduce the difference between the network output and the training sample. In the second approach, the average error for the whole training set is calculated and the update of the free parameters is performed for the full set at once. In this investigation, the batch mode was applied, since the convergence speed of the method increased dramatically compared to the sequential approach.

In the same way, the number of neurons in the single hidden layer is calculated. Starting with only three neurons, the number of neurons is iteratively increased until no significant decrease of the objective function is obtained in the validation data set.

7 Numerical examples

In following examples we investigate the approximation quality of the different surrogate models by means of a complex nonlinear analytical function in 2D, a nonlinear analytical function in up to 50 dimensions and the applicability in the framework of the significance and importance filters by means of an analytical and industrial example.

7.1 1D test function

The first example is given to analyse the convergence of the approximation quality of the different suroogate models depending on the number N of support points on a non-convex function of the Shepard typ:

$$\hat{y}(\mathbf{x}) = \frac{\sum_{i=1}^{n} y(\mathbf{x}_i) \left(\frac{1}{\|\mathbf{x} - \mathbf{x}_i\| + \epsilon} \right)^2}{\sum_{i=1}^{n} \left(\frac{1}{\|\mathbf{x} - \mathbf{x}_i\| + \epsilon} \right)^2}$$

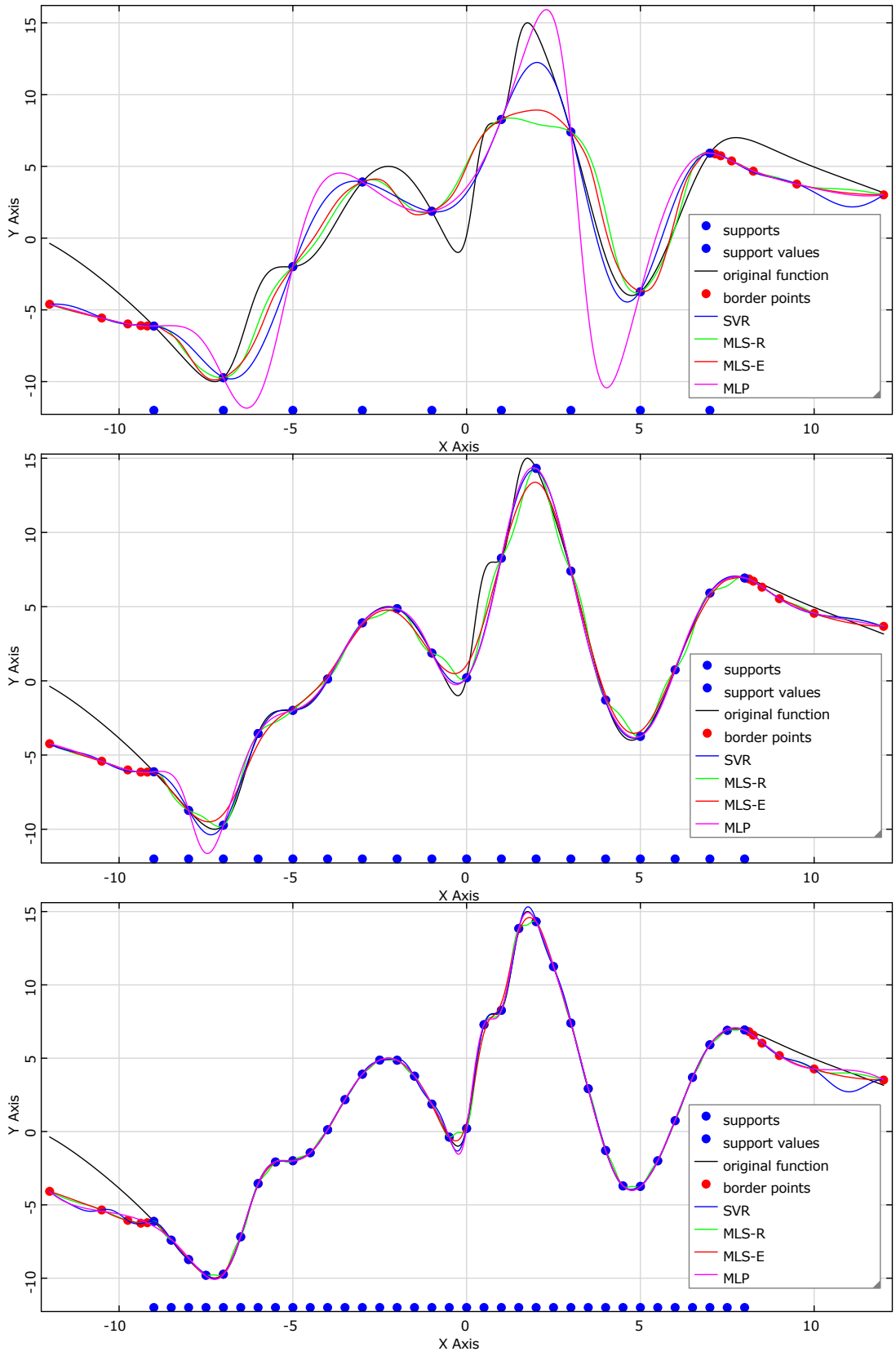


Figure 43: Deterministic 1D test function with $N = 9, 18, 35$ support points.

with $x_i = 0.75 + z_i$, $z_i = \{-15; -8; -6; -3; -1; 0; 1; 4; 7; 15\}$
and $y_i = \{1; -10; -2; 5; -1; 8; 15; -4; 7; 1\}$, as shown in Figure 43. As meta-models Moving Least Squares with regularized (MLS-R) and exponential (MLS-E) weighting function, Support Vector Regression (SVR) and Multilayer Perceptrons (MLP) are investigated. The optimal influence radius of the exponential weighting function is determined based on an adaptive D-approach. The results indicate, that for the deterministic case the SVR approximation converges much faster than these of MLP and MLS. For a relative large number of support points MLP, SVR and MLS the approximation is almost identical.

7.2 2D test function

In this example we investigate the well-known two-dimensional test function Teughels (2003)

$$f(\mathbf{x}) = 0.01 \sum_{i=1}^{2} (x_i + 0.5)^4 - 30x_i^2 + 20x_i \quad \text{with} \quad -6 \leq x_i \leq 6 \tag{47}$$

for the deterministic case and for the case of noisy data values. For the latter purpose we modify the original function by adding Gaussian noise with zero mean and standard error equal to one to each support point value. In the Figures 44 and 45 the two functions are shown. The test data set, which is of the same size as the support data set, is modified in the same manner. Both data sets are obtained by Latin Hypercube Sampling with uniform distribution. The approximation quality is evaluated by the normalized mean error of 10000 regular distributed evaluation points.

In the Figures 46 and 47 the obtained relative mean errors are shown depending on the number of support points. The presented values are the average of 100 random sets for each configuration. As meta-models the polynomial regression, Moving Least Squares with regularized (MLS-Reg) and exponential (MLS-Exp) weighting function, Support Vector Regression (SVR) and Multilayer Perceptrons (MLP) are investigated. The optimal influence radius of the exponential weighting function is determined based on the test data set. The results indicate, that for the deterministic case the MLP approximation converges much faster than these of SVR and MLS. For the noisy function the convergence of MLP, SVR and MLS is almost identical. As expected the polynomial regression can not converge to the exact result due to the high complexity of the function.

7.3 nD test function

In this example we investigate the nonlinear function

$$f(\mathbf{x}) = \sum_{i=1}^{n} exp(-x_i^2) + (5 - x_i)^{-1} \quad \text{with} \quad 0 \leq x_i \leq 4 \tag{48}$$

for different numbers of input variables. In Figure 48 this function is shown for 2 variables. We investigate the different surrogate models for the cases of 5, 10, 20 and 50 input variables nV, whereby for every case $2nV$, $5nV$, $10nV$ and $20nV$ Latin Hypercube samples are used as support points to build the approximation functions. Again a test data set of same size is used to determine the parameters of the MLS, SVR and MLP models. The relative mean error is calculated using 10000 uniformly distributed Monte Carlo samples.

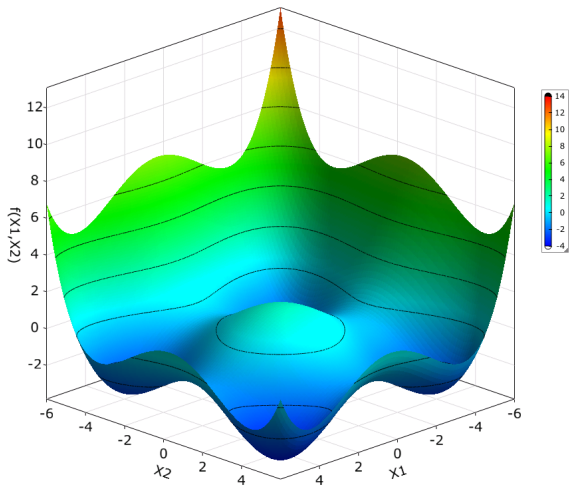


Figure 44: Deterministic 2D test function

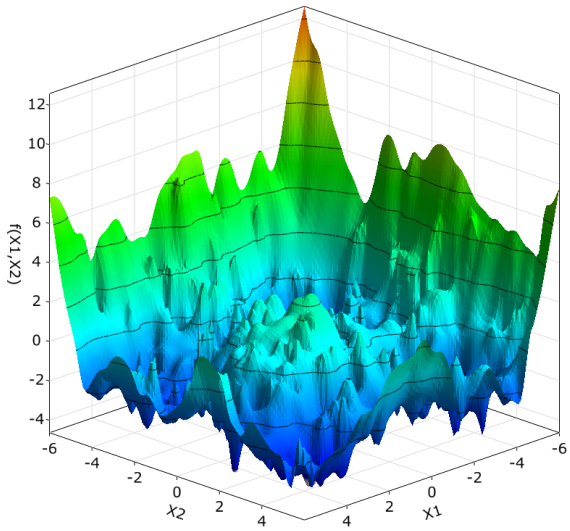


Figure 45: 2D test function with additional Gaussian noise

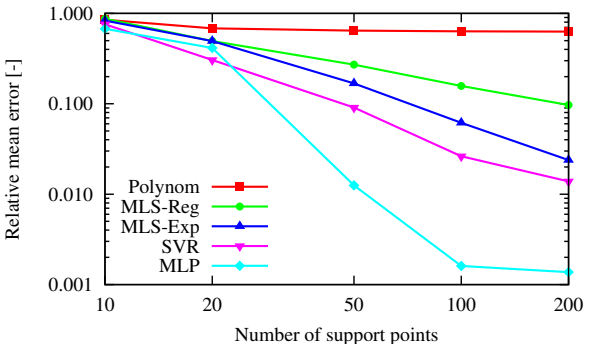


Figure 46: Relative mean errors for the deterministic 2D function

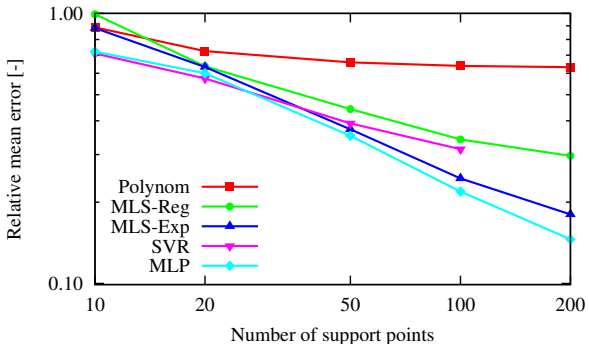


Figure 47: Relative mean errors for the 2D function with additional noise

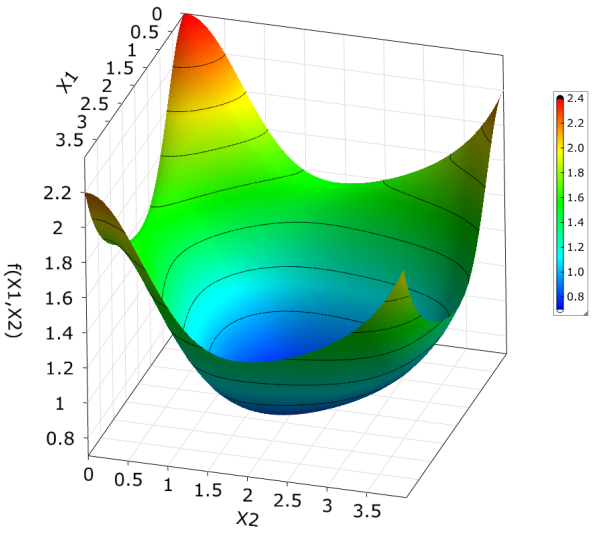


Figure 48: Investigated nonlinear test function for the 2D case

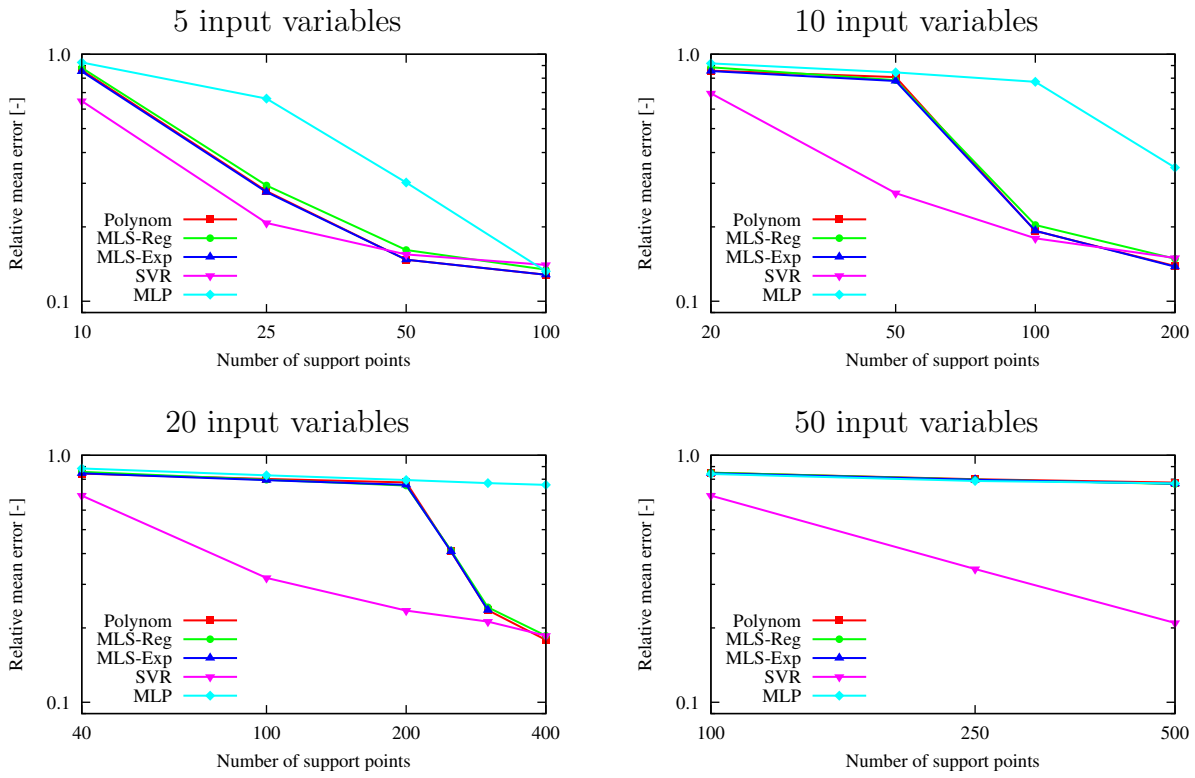


Figure 49: Relative mean errors for the analytical test function with 5, 10, 20 and 50 input variables

In Figure 49 the calculated relative mean errors are shown for the different numbers of input variables depending on the number of support points. Again the results from 100 random sets for each support point configuration are averaged. The figure indicates, that the polynomial regression and the Moving Least Squares approach with regularized and exponential weighting function agree almost exactly with each other. For the cases where enough available support points enable the application of a quadratic base polynomial these approaches show similar approximation errors as Support Vector Regression. If only a linear base polynomial can be applied because of the to small number of support points the polynomial-based approaches show significant higher approximation errors than SVR. This can be observed for the cases with more than 10 input variables. Similar to the polynomial-based models the approximation errors of the Multi-Layer Perceptron increase rapidly for an increasing dimension.

An investigation of this test function with additional Gaussian noise has shown similar results as the deterministic function. For this reason we do present these results at this point.

7.4 Dimension reduction by significance and importance filter

Based on the investigations in the previous example we analyze now the effect of the proposed significance and importance filters. For this purpose we investigate three analytical functions

$$\begin{aligned} f_1(\mathbf{x}) &= \sum_{i=1}^{n} w_i \left[exp(-x_i^2) - (5 - x_i)^{-1} \right] \\ f_2(\mathbf{x}) &= \sum_{i=1}^{n} w_i \left[exp(-x_i^2) \right] \\ f_3(\mathbf{x}) &= \sum_{i=1}^{n} w_i \left[exp(-x_i^2) + (5 - x_i)^{-1} \right] \end{aligned} \tag{49}$$

with

$$0 \leq x_i \leq 4; \quad w_i = 1;\ 1;\ 1;\ \frac{1}{2};\ \frac{1}{3};\ \frac{1}{4};\ \ldots;\ \frac{1}{i-2} \tag{50}$$

where function 1 is weakly nonlinear, function 2 contains nonlinearities of pure exponential type and function 3 is strongly nonlinear. The test functions are shown in Figure 50. The weighting factors w_i are introduced to obtained a set with some important input variables and some more unimportant variables, which can be neglected in building up the approximation models.

In the first step, we analyze the output of the significance and importance filter. In Figure 51 the calculated Coefficient of Importance for the first 10 input variables are shown for the deterministic functions by using 20 input variables with 40 and 100 sampling points. Additionally the probability of the selection of the parameter is shown by assuming a minimal CoI of 0.02. In the figure the mean values of the CoI with the corresponding 90% interval obtained from 100 independent random sampling sets are displayed. The quantile value for the significance filter was chosen with 95%. The figure shows clearly that for function 1 the variation of the CoIs is quite small even for the case with 40 sampling points and the important variables can be identified with high probability. For the nonlinear cases, especially for function 3, 40 sampling points are not sufficient, since

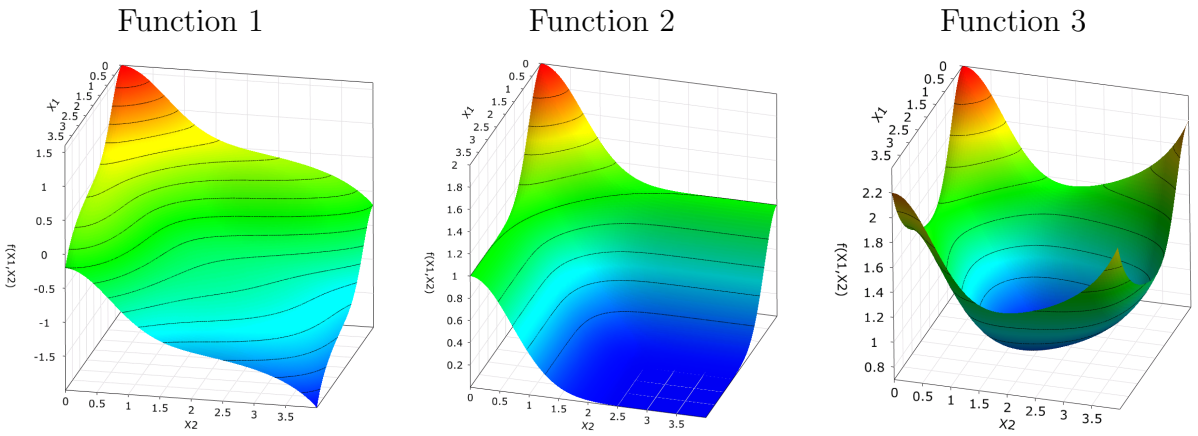


Figure 50: Analytical test functions for the investigations of the variable filters

the variation of the CoIs are very high and the important variables are not selected in all cases. Analogous the probability that unimportant variables will be selected is much higher than for the weakly nonlinear function. If 100 sampling points are taken for the nonlinear cases the variation of the calculated CoIs is much smaller and the important variables can be identified very reliable. A good measure for a sufficient sampling size is the adjusted Coefficient of Determination (CoD) of the full system, which is in our case the system with the remaining variables of the significance filter. If this value is around 0.8 or higher the prediction of the Coefficient of Importance will be sufficiently accurate.

In the next step we investigate the influence of noise on the accuracy of the significance and importance filters. For this purpose we modify the test functions by adding Gaussian noise with zero mean and increasing standard deviation σ_ξ to the sampling point values. In Figure 52 the calculated CoIs for function 3 are shown. From the figure we can observe, that with increasing noise level the CoD of the full system decreases and the variation of the CoIs increases. However the important variables can be identified with sufficient probability even for the third case, where the variation of the noise is about 50% of the variation of the deterministic function.

The final investigation of the filter accuracy is the influence of the problem dimension. For this case we increase the number of input variables up to 100. In Figure 53 the obtained CoIs are shown for the deterministic function 3. The variation of the CoIs and thus the accuracy of the filter is almost similar for 50 input variables as for 20 variables. For the case with 100 variables the accuracy is not sufficient, which indicates that more than 100 sampling point would be required for a strong nonlinear function.

Additionally to the accuracy of the proposed filters we investigate the effect of the filters on the approximation quality of the presented surrogate models. For this purpose we analyze the relative mean error from 10000 Monte Carlo samples similarly to example 7.3. In Figure 54 the calculated errors are shown for function 2 and 3 with 20 variables and 100 support points depending on the value of the importance filter. The averaged number of remaining variables are shown additionally in each diagram. The figure clearly shows, that the application of the filters lead to a significant reduction of the approximation errors for all investigated surrogate models, but the errors increase above a certain CoI value where not all important variables are used for the approximation. The differences between the results of the different meta-models are small. Again the SVR gives shows its best result for a larger number of variables and the MLP for a smaller dimension.

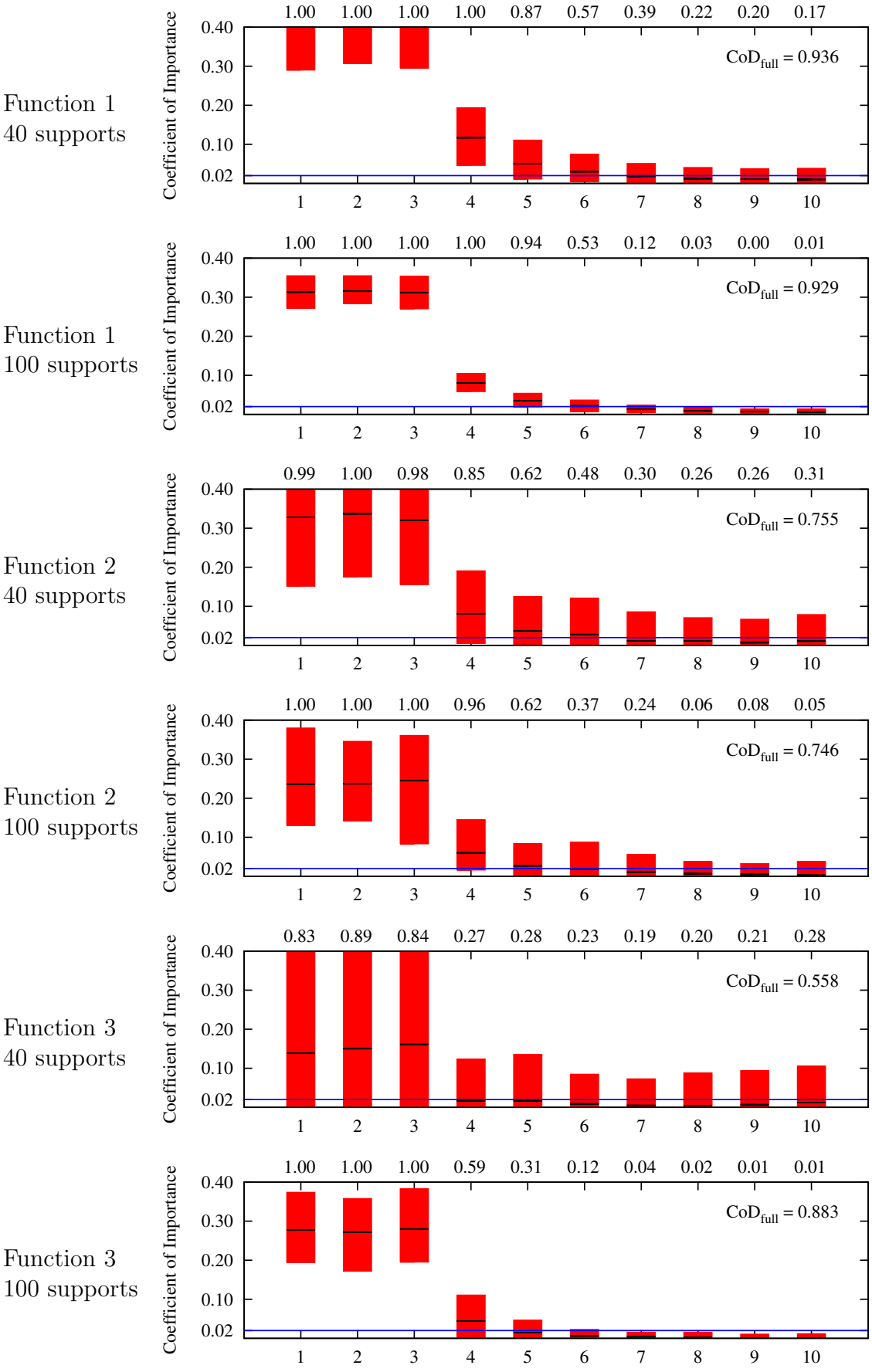


Figure 51: CoI for the first 10 of 20 input variables (deterministic functions)

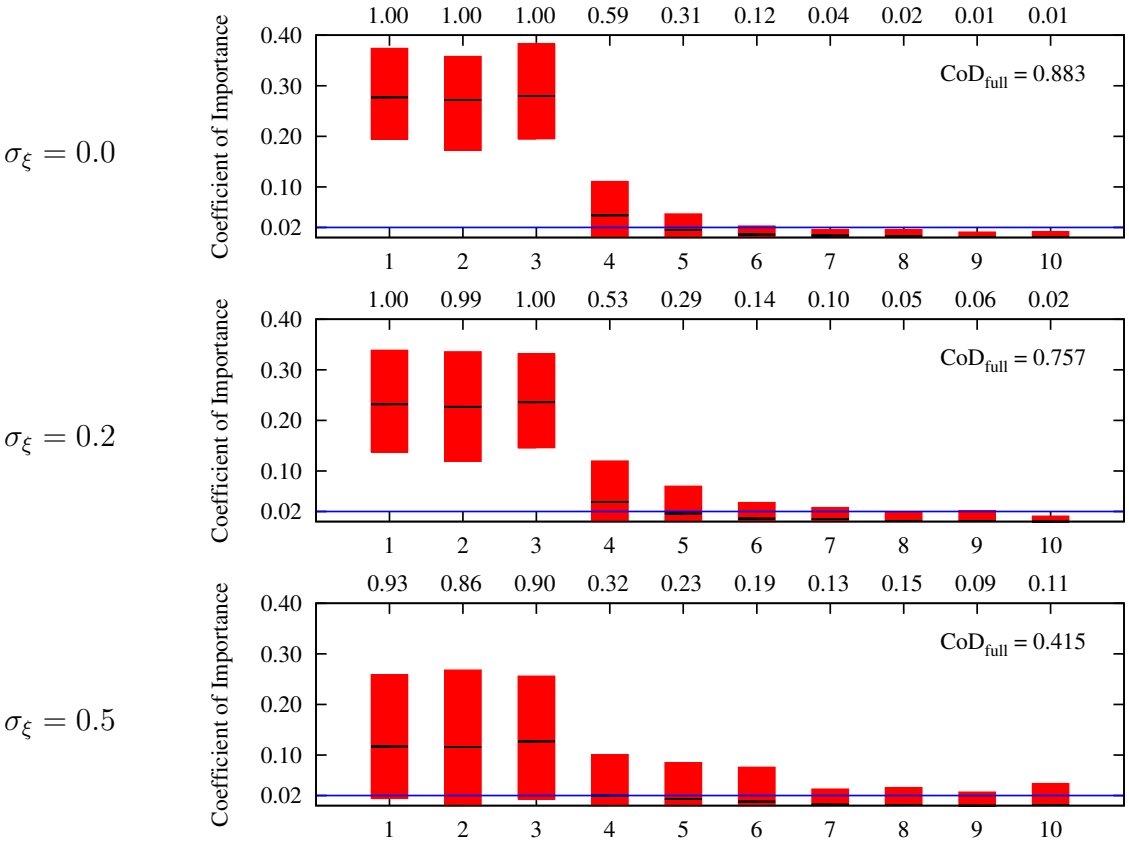


Figure 52: CoI for increasing noise ratio (function 3, 100 supports, 20 input variables)

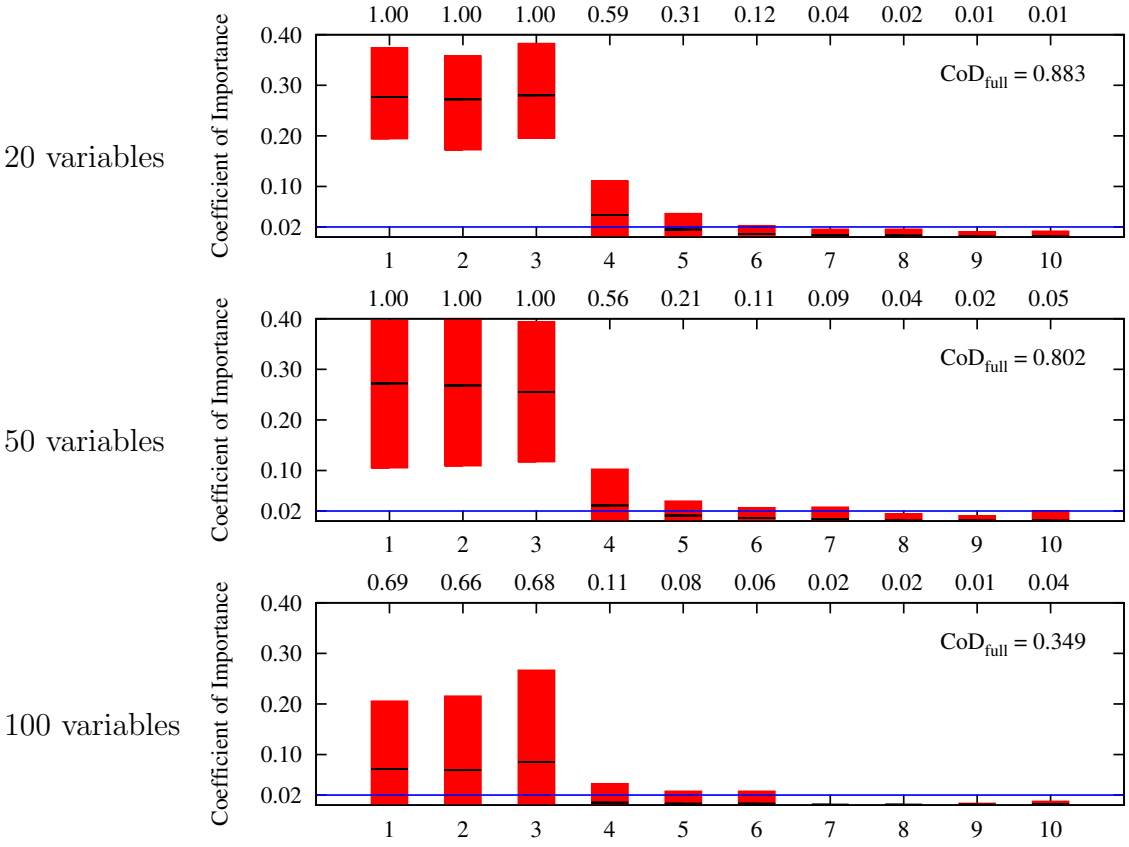


Figure 53: CoI for increasing dimension (deterministic function 3, 100 supports)

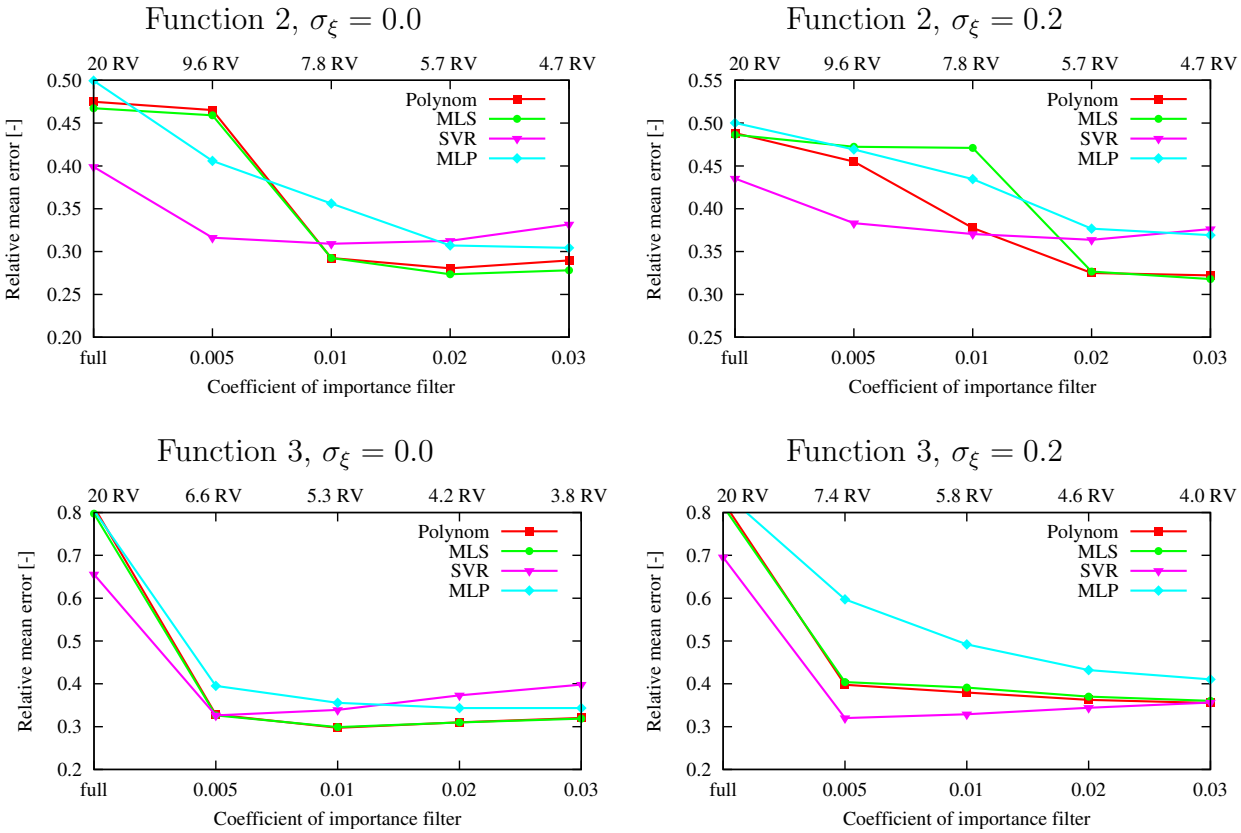


Figure 54: Approximation errors for the surrogate models depending on the applied filter parameters (function 3, 20 input variables, 100 support points)

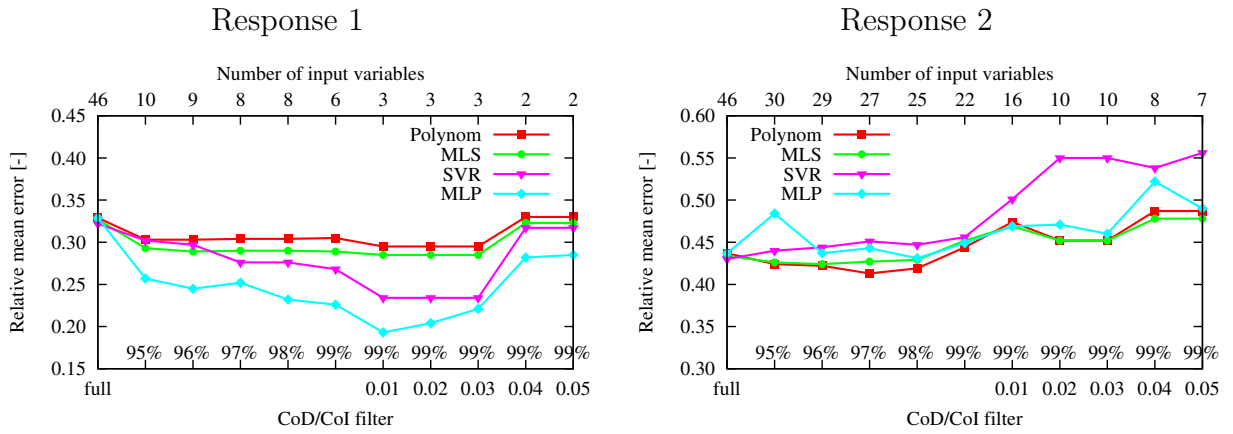


Figure 55: Industrial application: approximation errors for the surrogate models depending on the applied filter parameters

7.5 Industrial application

In this final example we investigate the applicability of the presented filters and approximation models for a real problem, where we have 46 input variables and several output quantities which are anonymous. Again we apply first the significance and importance filter and use the remaining set of input variables to build up the approximation model. We use 100 support points for the surrogate model and 100 test samples to determine the model parameters and to evaluate the approximation error. In Figure 55 the calculated relative approximation errors for two representative responses are shown depending on the filter parameters. The figure indicate a significant reduction of the approximation errors for all surrogate models for response 1, whereby the MLP approach gives the best results. The approximation errors for response 2 can be reduced slightly only for the polynomial based approaches for a significance filter of 97% which corresponds to 27 remaining input variables. Further reduction of the variable number leads to increasing approximation errors. The SVR and MLP approximation errors are for the reduced systems always larger as for the full system. This shows, that not for all cases a significant better result can be achieved. This may be caused by several reasons, e.g. that response 2 contains a large noise fractions or that to many input variables are important for the approximation. The usage of an increased number of sampling points could possibly solve this problem.

8 Conclusions

In this paper we have investigated several advanced surrogate models concerning their applicability in the framework of a robustness evaluation. Based on the results of the investigated numerical examples we can summarize, that we can not define one best model type which is promising for all application. We found that Artificial Neural Networks and Moving Least Squares work very nicely for low dimensional problems with less than 10 input variables. For higher dimensional problems the Support vector Regression method gives more accurate results especially if the number of available support points is relatively small. All types of models work similar for noisy data as for the deterministic case, if we determine the model parameters on the basis of an additional test data set. If this test data set is not available, the support point set can be subdivided in training and test data and after the model parameters have been determined the full data set can be used for the approximation. Our recommendation for the choice of the most proper surrogate model for a specific problem is to compare the approximation quality of the available models based on a test data set and select the best approach.

In many real applications not all input variables contribute significantly to the response functions. For this cases a selection of important variables and an approximation on the reduced variable set can increase the approximation accuracy dramatically. For this purpose we developed an efficient significance and important filter to identify the important input variables. We could show that this filter combination works very reliable even for noisy response data and its application can improve the applied approximation model significantly. For real applications this stands for a dramatic reduction of the required computational costs within the robustness analysis.

References

S. Amari, N. Murata, K.-R. Müller, M. Finke, and H. Yang. Statistical theory of overtraining — is cross-validation asymptotically effective? In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 176–182. The MIT Press, 1996.

A. J. Booker, Jr. Dennis, J. E., P. D. Frank, D. B. Serafini, V. Torczon, and M. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural Optimization*, 17(1):1 – 13, 1999.

G. E. P. Box and N. R. Draper. *Empirical Model Building and Response Surfaces*. John Wiley and Sons, New York,USA, 1987.

K. K. Choi, Byeng D. Youn, and Ren-Jye Yang. Moving least square method for reliability-based design optimization. In *WCSMO-4*, Dalian, China, 2001. Center for Computer-Aided Design and Department of Mechanical Engineering. URL http://www.icaen.uiowa.edu/~byoun/WCSMO-4.pdf.

D. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. *Advances in Neural Information Processing Systems*, 9:155–161, 1997.

C. Du. An interpolation method for grid-based terrain modelling. *THE COMPUTER JOURNAL*, 39(10):837 – 843, 1996.

A. Florian. An efficient sampling scheme: Updated Latin Hypercube Sampling. *Probabilistic Engineering Mechanics*, 7:123–130, 1992.

A. Giunta and L. T. Watson. A comparison of approximation modeling technique: Polynomial versus interpolating models. In *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization*, pages 392 – 404. AIAA, St. Louis, MO, 1998.

L. Gu, R.J. Yang, C.H. Tho, M. Makowskit, O. Faruquet, and Y.Li. Optimisation and robustness for crashworthiness of side impact. *International Journal of Vehicle Design*, 26(4):348 – 360, 2001.

M. T. Hagan, H. B. Demuth, and M. Beale. *Neural Network Design*. PWS Publishing Company, 1996.

Martin H. Hagan and Mohammad B. Menhaj. Training feedforward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6):989–993, 1994.

Simon Haykin. *Neural Networks, a comprehensive foundation*, chapter 4.6, pages 181–182. Prentice Hall, second edition, 1999.

E. M. Johansson, F. U. Dowla, and D. M. Goodman. Backpropagation learning for multilayer feed-forward neural networks using the conjugate gradient method. *International Journal of Neural Systems*, 2(4):291–301, 1992.

D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13:455 – 492, 1998.

P. Lancaster and K. Salkauskas. *Curve and surface fitting; an introduction*. Academic Press, London, 1986.

X. Liao, Q. Li, X. Yang, and W. Zhang. Multiobjective optimization for crash safety design of vehicles using stepwise regression model, 2007. URL http://www.springerlink.com/content/577x13588270v270.

H. O. Madsen, Steen Krenk, and Niels C. Lind. *Methods of Structural Safety*. Prentice-Hall, Englewood Cliffs, 1986.

J.D. Martin and T.W. Simpson. A study on the use of kriging models to approximate deterministic computer models. In *Proceedings of DETC ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference Chicago, September 2-6*. Illinois USA, 2003.

G. Matheron. Principles of geostatistics. *Economic Geology*, 58:1246 – 1266, 1963.

M. D. McKay, W. J. Conover, and R. J. Beckmann. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

Martin F. Moller. Scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6(4):525–533, 1993.

T. Most and C. G. Bucher. A moving least squares weighting function for the element-free galerkin method which almost fulfills essential boundary conditions. *Structural Engineering and Mechanics*, 21(3):315 – 332, 2005.

R. H. Myers. *Response Surface Methodology*. Allyn and Bacon Inc., BostonUSA, 1971.

R. H. Myers and D. C. Montgomery. *Response Surface Methodology - Process and Product Optimization Using Designed Experiments*. John Wiley & Sons, Inc., New York, 1995.

J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1998.

M. Rais-Rohani and M. N. Singh. Comparison of global and local response surface techniques in reliabilty-basedoptimization of composite structures. *Structural and Multidisciplinary Optimization*, 26:333 – 345, 2004.

Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *Proc. of the IEEE Intl. Conf. on Neural Networks*, pages 586–591, San Francisco, CA, 1993.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409 – 435, 1989.

B. Schoelkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2001.

D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proc. ACM Natl Conf.*, pages 517 – 523. 1968.

T. W. Simpson, J. Peplinski, P. N. Koch, and J. K. Allen. Metamodels for computer-based engineering design: Survey and recommendations. *Engineering with Computers*, 17(2):129 – 150, 2001.

T. W. Simpson, A. J. Booker, S. Ghosh, A. Giunta, Koch, P. N., and R. J. Yang. Approximation methods in multidisciplinary analysis and optimization: A panel discussion. *Structural and Multidisciplinary Optimization*, 2003.

A. J. Smola and B. Schoelkopf. A tutorial on support vector regression. NeuroCOLT2 Technical Report NC2-TR-1998-030, 1998.

A. J. Smola and B. Schoelkopf. A tutorial on support vector regression. *Statistics and Computing*, 14:199–222, 2004.

A. Teughels. *Inverse modelling of civil engineering structures based on operational data*. PhD thesis, Katholieke Universiteit te Leuven, 2003.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974.

R. J. Yang and L. Gu. Experience with approximate reliabiltity-based optimization. *Structural and Multidisciplinary Optimization*, 26:152 – 159, 2004.

B. D. Youn, K. K. Choi, R. J. Yang, and L. Gu. Reliability-based design optimization for crashworthiness of vehicle sideimpacts. *Structural and Multidisciplinary Optimization*, 26:272 – 283, 2004.

L. Yu, Das P.K., and Y. Zheng. Stepwise response surface method and its application in reliability analysis of ship hull structure. *Journal of Offshore Mechanics and Arctic Engineering*, 124(4):226 – 230, 2002.