# Efficient optimization of structural designs using methods of global sensitivity analysis with reduced meta-models

Uwe Reuter, Gustavo Canon Falla
Department of Civil Engineering, Technische Universität Dresden, Germany

Martin Liebscher, Zeeshan Mehmood
DYNAmore GmbH, Dresden, Germany

**Summary:**

Methods of global sensitivity analysis are used to identify significant parameters in order to perform computationally less expensive optimization of design structures. In most engineering problems, only a few number of design points are available to model structure response for sensitivity analysis. Usually, the initial meta-model is not a good predictor of the actual model response. The optimum solution from the initial meta-model might not lie in the region corresponding to the region where the optimal solution to the actual model response lies. In this paper, optimization of design structures is performed using methods of global sensitivity analysis on reduced meta-models, such as classification based global sensitivity methods. These methods identify significant parameters using only the approximation of the level sets of the model response. The optimization is then carried out on the initial meta-model but only on the domain of significant parameters, under the assumptions: (a) search for an optimum is effective on the domain on which the model response varies the most (b) variation of the model response at the level sets is relatively less prone to approximation errors as compared to the full approximation in very high dimensional models. The results of the optimization using global sensitivity methods with reduced meta-models are compared with already existing methods which use full approximation of the model response for their realization.

**Keywords:**

Optimization, sensitivity analysis, variable screening, meta-models, Support Vector Machines.

## 1 Introduction

In the context of meta-model based optimization of the structural response, before the optimization process, each individual model parameter $X_{i,\, i=1,2,\cdots,n}$ is compared with the other remaining parameters $X_1, \cdots X_{i-1}, X_{i+1}, \cdots, X_n$ in order to evaluate its influence on the model response $Z$. This procedure is known as sensitivity analysis. The normalized sensitivity measure $S_i$ for a model parameter $X_{i,\, i=1,2,\cdots,n}$ is given by:

$$S_i = \frac{\widetilde{S}_i}{\sum_{j=1}^{n} \widetilde{S}_j} \tag{1}$$

where $\widetilde{S}_i$ represents the influence of $X_i$ on $Z$ according to a specific sensitivity measure. A number of significant parameters are then chosen to direct the search for the optimal solution [8]. Generally known global sensitivity methods for non-linear models such as Sobol Indices [11] and neural network based methods [7, 9] require a substantially good approximation of the model response. Recently, different classification based methods [10] have been proposed which differs from the above mentioned methods due to the granularity of the model approximation. This approach uses the approximation of only the level sets to extract sensitivity information. The approximation for the level sets requires relatively lesser sample points as compared to the number of sample of points for a full approximation. Support Vector Machines (SVMs) (see e. g. [16]) are used to identify the level sets by partitioning the model response into a set of disjoint classes. The sensitivity measure for each parameter is then calculated by means of class changes on that parameter domain through Monte Carlo simulation [10].

In this paper, a comparison is made between the results of the optimization using classification based sensitivity measures and the optimization using other meta-model based sensitivity measures. For this purpose, optimization software LS-OPT is used for full model approximation and optimization using genetic algorithms. However, classification based sensitivity measures are calculated with a reduced meta-model using PySen, which is a sensitivity analysis software tool written in Python programming language. A wrapper script is used to automate the optimization study for increasing number of sample points in order to evaluate the influence of the meta-model approximation on the sensitivity measures and in turn on the optimization results as well.

In the next section, different global sensitivity analysis methods are briefly explained, which require full approximation of the model as well as the new classification based sensitivity methods with reduced meta-models. Section 3 highlights the optimization using sensitivity measures and the software test system used for the comparison. Optimization results of a high dimensional crash test example are presented in Section 4. Section 5 concludes the paper.

## 2 Global sensitivity analysis methods

### 2.1 Variance based methods

These methods implicitly assume that the moments (e.g. variance) are sufficient to describe the variation in the model response. One such method is the variance based decomposition [11, 12]. A response $Z = f(\underline{X}), \underline{X} = (X_1, \cdots, X_n)$, can be represented as

$$f(\underline{X}) = f_o + \sum_{i}^{n} f_i(X_i) + \sum_{1 \le i < j \le n} f_{ij}(X_i, X_j) + \cdots + f_{1,2,\cdots,n}(X_1, \cdots, X_n) \tag{2}$$

Such a decomposition of $f(\underline{X})$ is termed as variance based decomposition. The function $f(\underline{X})$ is characterized by its variance $V$, which can be decomposed into partial variances associated with $X_1, X_2 \cdots, X_n$ according to Eq. (2) as

$$V = \sum_{i=1}^{n} V_i + \sum_{1 \le i < k \le n} V_{i,k} + \cdots + V_{1,2,\cdots,n} \tag{3}$$

Each partial variance $V_{i_1,\cdots,i_s}$ can be related to each of the sensitivity measures $S_{i_1,\cdots,i_s}$ as

$$S_{i_1,\cdots,i_s} = \frac{V_{i_1,\cdots,i_s}}{V},\ 1 \leq i_1 < \cdots i_s \leq n,\ s = 1,2,\cdots,n \tag{4}$$

In order to evaluate the total effect of a single variable $X_i$, all partial sensitivity measures $S_i$ involving $X_i$ are summed up to define the total sensitivity measure $S_{T_i}$. The total sensitivity measures consider the interactions among all model parameters. In order to quantify which amount of variance $V$ is caused due to a single variable $X_i$, the corresponding total sensitivity measures $S_{T_i}$ can be normalized as

$$norm\, S_{T_i} = \frac{S_{T_i}}{\sum_{k=1}^{n} S_{T_k}} \tag{5}$$

The total sensitivity measure $S_{T_i}$ as shown in Eq. (5) can be numerically computed using the SOBOL approach [12] which uses the Monte Carlo simulation. $S_{T_i}$ associated with each input variable $X_i$ can be computed as $S_{T_i} = 1 - S_{\sim i}$, where $S_{\sim i} = \frac{V_{\sim i}}{V}$ and

$$V_{\sim i} \approx \frac{1}{N}\sum_{k=1}^{N} f\left(X_{\sim ik}^{(1)}, X_{ik}^{(1)}\right) f(X_{\sim ik}^{(1)}, X_{ik}^{(2)}) - f_0^2 \tag{6}$$

The superscripts (1) and (2) indicate that two different samples are generated and mixed. $X_{\sim ik}^{(1)}$ denotes the $k$th sample point with $X_{\sim ik}^{(1)} = (X_{1k}^{(1)}, \cdots, X_{(i-1)}^{(1)}, X_{(i+1)k}^{(1)}, \cdots, X_{nk}^{(1)})$ and $f_0 = \frac{1}{N}\sum_{m=1}^{N} f(X_k)$ is the mean and $N$ is the number of simulation.

## 2.2 Weight based sensitivity methods

If Artificial Neural Networks are used to model the response, the values stored in the static matrix of weights can be used to determine the relative influence of each input variable on the network response [3, 4, 15]. The methods which uses neural network connection weights for calculating sensitivity measures are broadly termed as weight based sensitivity measures. Different equations have been proposed which calculate the products of the weights of the network and then obtaining the sum of the calculated products according to a certain criteria. Garson [3] proposed an equation

$$S_{ik} = \frac{\sum_{j=1}^{L}\left(\frac{w_{ij}}{\sum_{r=1}^{N} w_{rj}} w_{jk}\right)}{\sum_{i=1}^{N}\left(\sum_{j=1}^{L}\left(\frac{w_{ij}}{\sum_{r=1}^{N} w_{rj}} w_{jk}\right)\right)} \tag{7}$$

in which hidden-output connection weights are partitioned into components associated with each input neuron using absolute values of connection weights, where $w_{ij}$ is the weight associated with the input neuron $i$ and the hidden neuron $j$ and $w_{jk}$ is the weight associated with the hidden neuron $j$ and output neuron $k$, $\frac{w_{ij}}{\sum_{r=1}^{N} w_{rj}}$ is normalized value of the connection weight, $N$ is the total number of input neurons and $L$ is the total number of hidden neurons. The subscript $k$ can be dropped if only one output neuron is used for a single model response.

Tchaban et al. [15] proposed an equation

$$S_{ik} = \frac{x_i}{o_k}\sum_{j=1}^{L} w_{ij} w_{jk} \tag{8}$$

where $S_i$ is the overall impact of the input $i$ on the output neuron $k$. Equation (8) states that the impact of the input $i$ on the output neuron $k$ through the hidden neuron $j$ is equal to the product of the impact of the input $i$ on the output of the hidden neuron $j$ with the impact of the hidden neuron $j$ on the network output neuron $k$. $w_{ij}$ is the weight associated with the input neuron $i$ and the hidden neuron $j$ and $w_{jk}$ is the weight associated with the hidden neuron $j$ and output neuron $k$. $L$ is the total number of hidden neurons. A neuron input has a positive impact on the neuron output if the input value multiplied by the weight is positive and vice versa.

Equation (8) has been modified in [7] for more than one hidden layer in which the impact or the sensitivity measure $S_i$ is formulated for the number of layers $l \in \{1, \cdots, s\}$ and with $j_l \in \{1, \cdots, N_l\}$ neurons per layer $l$ respectively as

$$S_i = \sum_{j_{s-1}=1}^{N_{s-1}} \cdots \sum_{j_2=1}^{N_2} \left| w^1_{i,j_2} . w^2_{j_2,j_3} . \ldots . w^{s-1}_{j_{s-1},1} \right| \tag{9}$$

## 2.3 Derivative based sensitivity methods

Partial derivatives of a model response can be used to determine the influence of the input parameters since they represent the instant slope of the model response between each pair of input $X_i$ and response $Z$. Thus, the local sensitivity of a function $Z = f(\underline{X})$ with $\underline{X} = (X_1, X_2, \cdots, X_n)^T$ at a certain point can be represented by the partial derivatives. In order to calculate the global sensitivity measures, the partial derivatives can be integrated over the complete input space $H^n$ as

$$\tilde{S}_i = \int_{H^n} g_i(\cdot) \ . \tag{10}$$

In [1], $\left| \frac{\partial f(\underline{X})}{\partial X_i} \right|$ is taken as the value of the partial derivative $g_i(\cdot)$ at $X_i$, whereas $g_i(\cdot) = \left( \frac{\partial f(\underline{X})}{\partial X_i} \right)^2$ is used in [13] as a value of the partial derivative for calculating derivative based global sensitivity measure. For a discrete number of points Eq. (10) can be approximated as

$$\tilde{S}_i \approx \frac{1}{N} \sum_{j=1}^{N} g_i\left(\underline{X}_j\right) \tag{11}$$

by using Monte Carlo sampling methods with $N$ number of points $\underline{X}_{j,\ j=1,2,\cdots,N}$.

If an Artificial Neural Network with a single hidden layer is used as a meta-model, the equation

$$g_i(\cdot) = \left| f'(net_k) \sum_{j=1}^{L} w_{ij} f'(net_j)\, w_{jk} \right| \tag{12}$$

(see [6]) can be used for calculating sensitivity measures, where $f'(net_j)$ and $f'(net_k)$ are the derivatives of the activation function of the hidden neuron j and the output neuron k respectively, $net_k$ is the output of the $k^{th}$ neuron of the output layer, and $net_j$ is the output of the $j^{th}$ neuron of the hidden layer. $w_{ij}$ is the weight between the $i^{th}$ neuron of the input layer and the $j^{th}$ neuron of the hidden layer and $w_{jk}$ is the weight between the $j^{th}$ neuron of the hidden layer and the $k^{th}$ neuron of the output layer. The examples presented in this paper use Eq. (12) as derivative based sensitivity measure.

## 2.4 Classification based sensitivity methods

Classification based sensitivity measures are calculated using only the level sets (or class partitioning) of the model response [10]. This level set based approximation of the model response is termed as a reduced meta-model. Once the level sets are identified with the help of Support Vector Machines, the sensitivity of the model response $Z = f(\underline{X})$ for a parameter $X_i$ is calculated by generating points by varying $X_i$ through Monte Carlo simulation according to its probability distribution function at repeatedly fixed values of the rest of the parameters $X_1, \cdots X_{i-1}, X_{i+1}, \cdots, X_n$. For each generated point the membership to a class is determined and an average change in the class membership for all points on the domain of each parameter $X_i$ is considered as the sensitivity measure $\tilde{S}_i$ for $X_i$ [10]. The change in class membership can be determined using different methods such as Vertical Class Jump Method (VCJM), Horizontal Class Jump Method (HCJM), and Boundary Method (BM).

For Vertical Class Jump Method, $N_{\text{sim}}$ points are generated for each input parameter $X_i$ according to its underlying probability distribution function for repeatedly fixed values of the rest of the parameters. For each generated point, the corresponding class index is determined. If the class index for a point in consideration is different than the class index of the previous point, a counter is incremented indicating a change of class. In this way, the value of the counter assigned to each parameter $X_i$ provides the influence of $X_i$ on model response $Z$ (see [10]).

Similarly for Horizontal Class Jump Method, the counter is not just incremented, instead the value of the counter is increased according to the number of subclasses (see [10]) present between the class of the point in consideration and the class of the previous point. In this method, the absolute difference of subclass indexes are considered and therefore it is termed as Horizontal Class Jump Method Delta (HCJMD). In this paper, HCJMD is used to calculated reduced meta-model based sensitivity measures.

Compared to the Horizontal Class Jump Method, the Boundary Method uses an approximate approach by taking into account the number of subclass boundaries which exist at each line defined by the repeatedly fixed values $X_1, \cdots, X_{i-1}, X_{i+1}, \cdots X_n$. Thus, the average number of the subclasses intersecting all lines represents a measure for the sensitivity for $X_i$ (see [10]).

## 3 Optimization using sensitivity measure with reduced meta-models

In most engineering problems, only a few number of design points are available to model and optimize a structural response. Therefore, the initial meta-model is usually not a good predictor of the actual structural response. The optimum solution from the initial meta-model might not lie in the region corresponding to the region where the optimal solution to the actual structural response lies. Structural responses are mostly dominated by a small number of significant parameters. One way to identify these significant parameters is according to their corresponding sensitivity measures. The search space of the optimization algorithm is then confined only within the domain of the significant variables on the initial meta-model under the assumption that search for an optimum is effective on the domain on which the model response varies the most.

Consider a simple three dimensional function $f(X_1, X_2) = X_1^3 - X_1^2 - 8X_1 + X_2^2$, where $X_1 \sim U(-3,4)$ and $X_2 \sim U(-5,5)$. Figure 1 shows the graph of $f(X_1, X_2)$ and the plausible search space for the optimal solution on the domain of significant variable $X_1$ with $X_2$ set to arbitrary fixed value for the optimization algorithm. Thus, the domain of significant variables provides a plausible search space for non-gradient based optimization algorithms (e.g. Genetic Algorithms). One way to identify this search space is through sensitivity measures using full meta-models and another using classification based sensitivity measures with reduced meta-models. The classification based sensitivity measures identify significant variables without using the full approximation of the model response [10], which can be prone to approximation errors. Figure 2 shows the optimization results for the example function $f(X_1, X_2)$ using a set of 5, 15, and 30 sample points. The example function is approximated using Artificial Neural Networks with 8 neurons in a single hidden layers using LS-OPT [14] as well as using a Pseudo Multiclass Support Vector Machine for the approximation of 5 level sets as a reduced meta-model for each set of points. The optimization results for a set of 30 points shows that a search for optimum is more plausible on the domain of significant variable using $S_i^{class}$ as compared to $S_i^{sobol}$ and the optimization results are significantly more closer to the analytical optimum value of $-12.0$. The process of optimization using reduced meta-model is depicted in Fig. 3.

### *Software test system*

In order to evaluate and compare the results of the optimization using reduced meta-model with full approximation of the model response, a software test system is developed using LS-OPT [14]. LS-OPT is an optimization tool which helps in design optimization, design of experiments, reliability studies and sensitivity analysis. It uses meta-models for optimization and sensitivity analysis. Since the accuracy of the meta-models is dependent on many factors such as the size of the sub-region and the number and distribution of the design points, LS-OPT provides several point selection procedures such as factorial, composite, D-optimal, Latin Hypercube and space filling for this purpose. LS-OPT acts as a powerful design point and response generator with its parallel processing and diverse remote job scheduling and

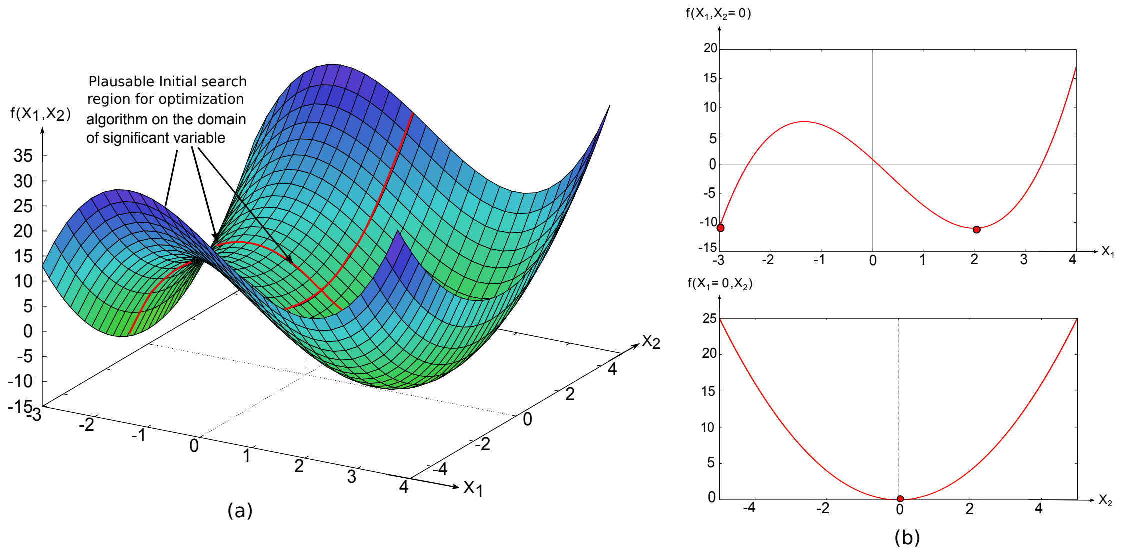


Figure 1: Significant variables and optimization: (a) Search space on the domain of significant variable (b) Cross sectional plots

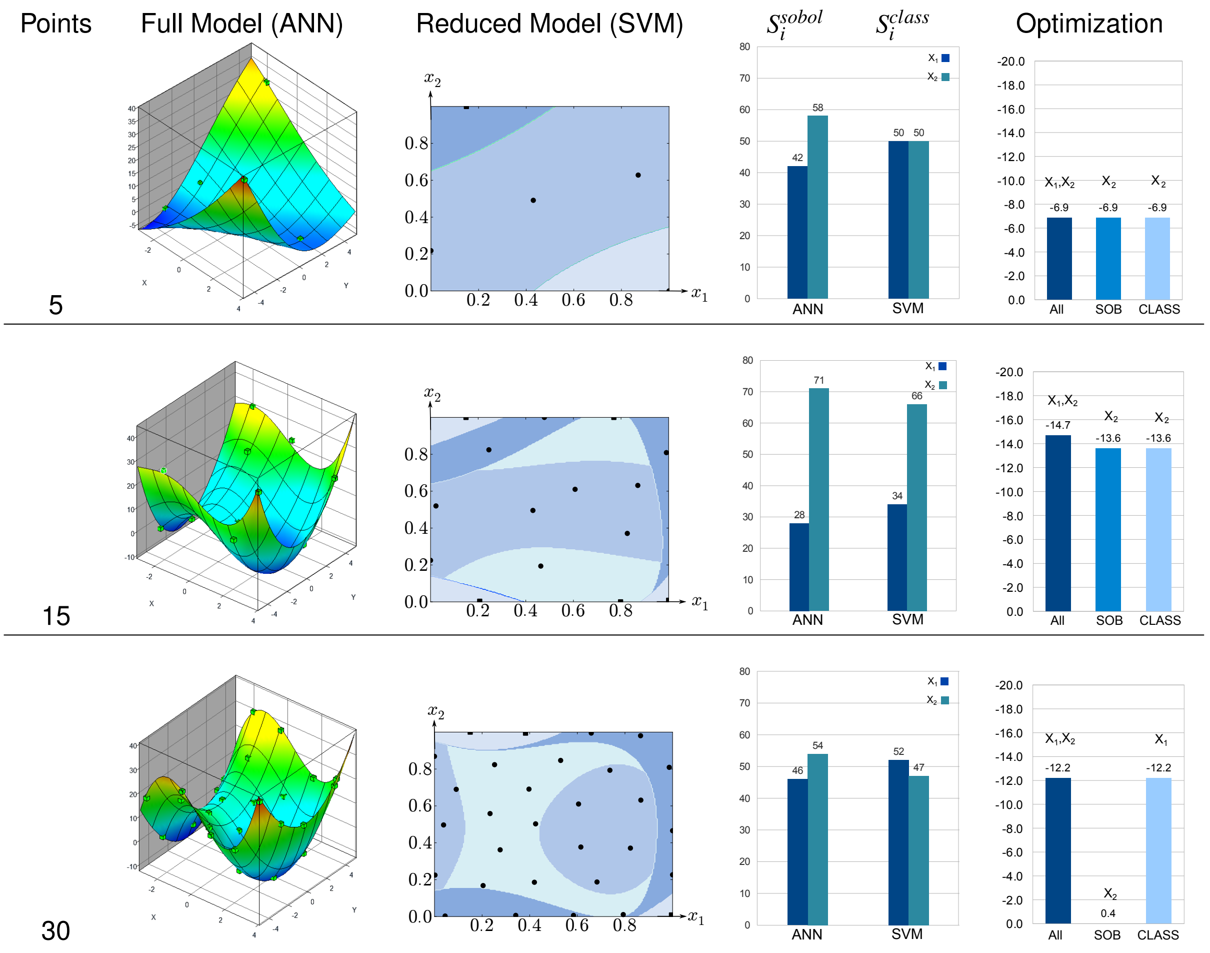


Figure 2: Optimization of example function

queuing facilities. In order to calculate classification based sensitivity measures a meta-model based sensitivity analysis tool PySen is attached to the system. It is written in Python programming language and contains a multi-layered neural network implementation for calculating different global sensitivity measures as well as a Pseudo Multiclass SVM for calculating classification based sensitivity measures. The wrapper is an umbrella layer that automates the execution of different test cases for optimization, see

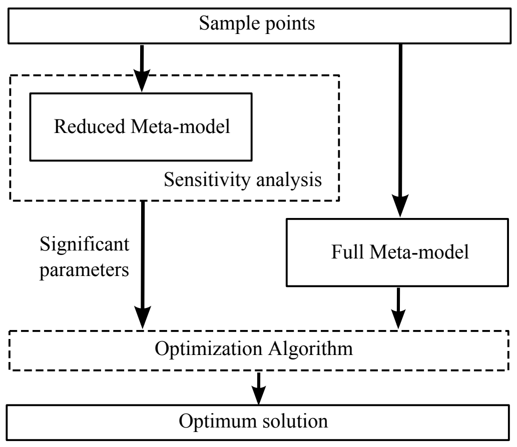


Figure 3: Optimization with reduced meta-models

Fig. 4. It contains a template generator which accepts input files with simple statements and invokes LS-OPT with automatically generated command files. It provides a simplified interface to LS-OPT focusing on executions related to sensitivity analysis and optimization. The wrapper also automates the execution of the test cases for different number of design points in order to analyze their effects on the model sensitivity and optimization.

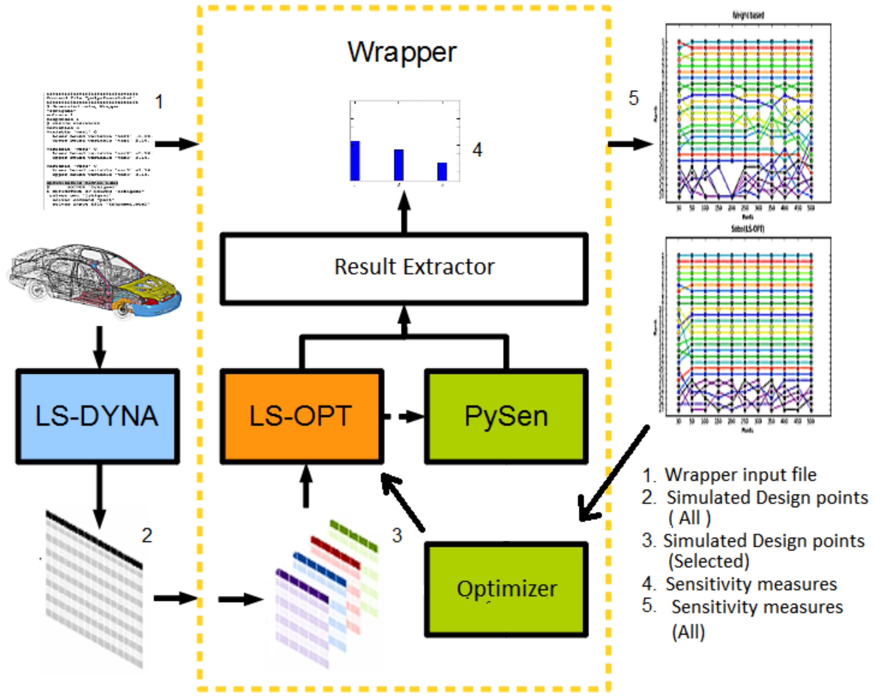
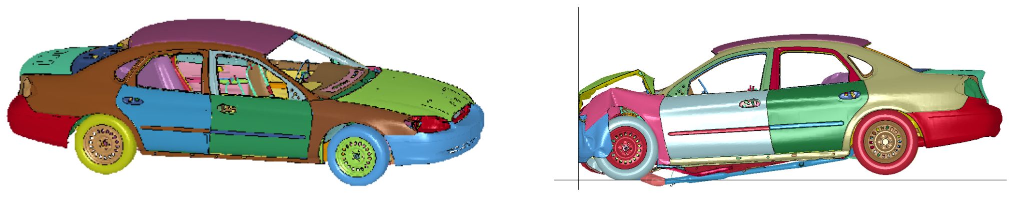

Wrapper
1
2
3
4
5
Result Extractor
LS-DYNA
LS-OPT
PySen
Optimizer
1. Wrapper input file
2. Simulated Design points ( All )
3. Simulated Design points (Selected)
4. Sensitivity measures
5. Sensitivity measures (All)

Figure 4: Test system

## 4 Example: US-NAP

As a structural optimization example, the NCAC Ford Taurus model (courtesy of [2]) is taken for a US-NCAP frontal impact test case and the acceleration of the lower block of the engine is taken as the structural response, see Fig. 5. The response depends upon 27 design variables from which 21 are sheet thickness variables and 6 are discrete material variables.



(a) NCAA Ford Taurus (b) US-NAP 56.6 km/h

Figure 5: Crash test example (courtesy of [2])

This paper uses the sensitivity measures given in [10] (see Fig. 6) and uses them for the optimization of the model response. For the evaluation of the different sensitivity measures, increasing sets of 30 to 3900 simulated points for the chosen response were taken. In this way, the effects of the number of sample points on the sensitivity measures and in turn on the optimization can be evaluated. For classification based methods, 9 level curves (10 classes) of the chosen response were taken for training the SVMs.

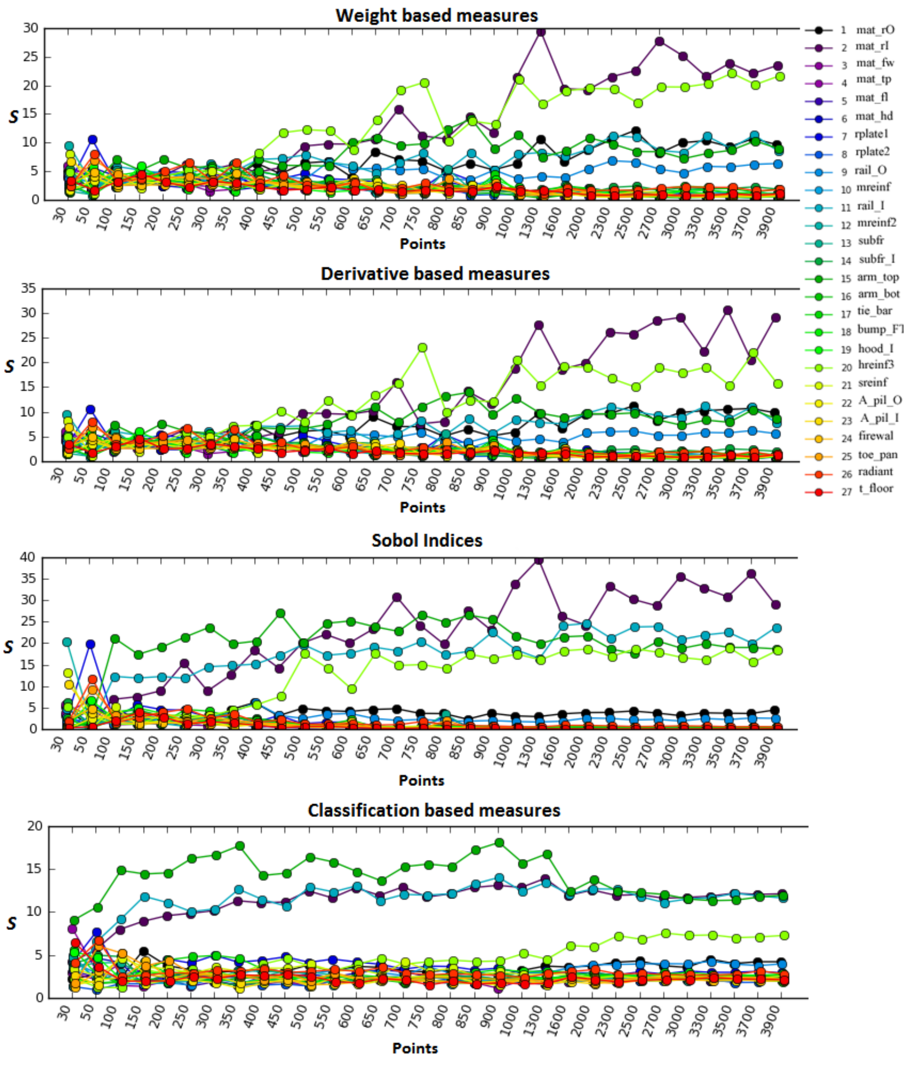
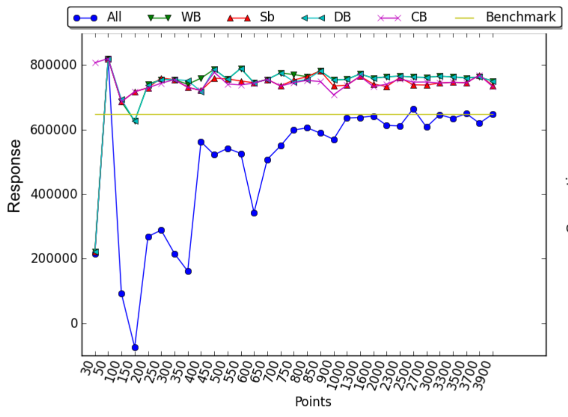
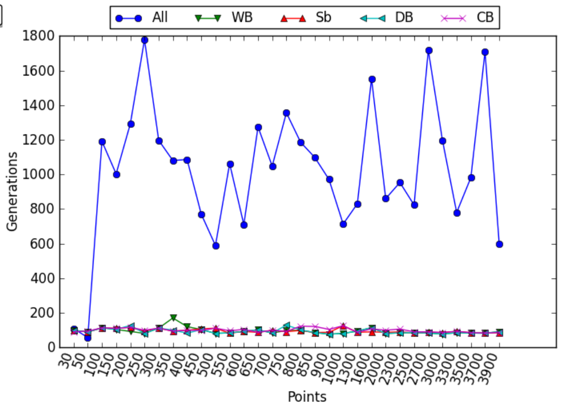


Figure 6: Sensitivity results for US-NCAP [10]

Classification based method was able to identify parameter 15 as the significant parameter already for the first 30 simulation points. Both classification based method and SOBOL method identify parameters 15, 11, 2, and 20 as the most significant parameters. However, the classification based method returns a lower sensitivity value for parameter 20 as compared to Sobol method. Also, the classification based measures seem to be relatively stable for the increasing number of simulation points as compared to weight based measures. The optimization is carried out using the Genetic Algorithm within LS-OPT with only the significant variables for each set of points. Genetic Algorithm based optimization technique is used due to its population based approach, providing a natural advantage over classical optimization techniques [5]. The significant parameters bias the search to the favorable parts of the solution space possibly around multiple solutions hypothesizing that the response varies significantly on the domain of the significant variables, so as to prevent convergence to a single solution in relation to the poorly fitted initial meta-model. The graph in Fig. 7a shows the optimization results. The reference optimum value is set to the optimum found over the complete domain (All) of model parameters with 3900 sample points. The optimum on the domain of only 2 most significant variables converges quickly to the optimum of 3900 points with classification based measures already close to the optimum with just 30 samples points. It can be seen from the graph that the optimization using all variables returns a very poor optimum value of the model response as compared to the reference solution for initial sets of sample points. One reason for this might be that due to a very high dimensional problem there is a potential that the optimization algorithm might converge to many false local optimums if all variables are used. Optimization over the domain of significant variables also decreases the computational effort in terms of the number of generations maintained by the Genetic Algorithm as shown in Fig. 7b.



(a) Optimization with significant variables



(b) Computational efforts in terms of generations maintained

Figure 7: Optimization results for US-NAP

## 5 Conclusions

Optimization using significant variables reduces the complexity of the optimization process for very high dimensional models and provides relatively good optimum results as compared to the optimization using the complete domain of input variables. The optimization results of a high dimensional crash analysis example using only the significant variables shows the efficiency of reduced meta-model based sensitivity measures for a plausible search space for finding an optimal solution.

## Acknowledgment

The authors gratefully acknowledge the financial support of the European Union (EFRE) and the Free State of Saxony as well as the support of the Center for Information Services and High Performance Computing (ZIH) of the Technische Universität Dresden. The Authors would also like to thank Ingolf Lepenies from DYNAmore GmbH Germany for his valuable discussions.

## References

[1] F. Campolongo, J. Cariboni, and A. Saltelli. An effective screening design for sensitivity analysis of large models. *Environmental Modelling and Software*, 22(10):1509 – 1518, 2007.

[2] FHWA/NHTSA National Crash Analysis Center. Finite Element Model of Ford Taurus – Model Year 2001, Version 3. Technical report.

[3] G. D. Garson. Interpreting neural-network connection weights. *AI Expert*, pages 47–51, 1991.

[4] M. Gevreya, I. Dimopoulosb, and S. Leka. Two-way interaction of input variables in the sensitivity analysis of neural network models. *Ecological Modelling*, 195:43–50, 2006.

[5] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989. ISBN 0201157675.

[6] J. J. Montano and A. Palmer. Numeric sensitivity analysis applied to feedforward neural networks. *Neural Computation & Applications*, 12, 2003.

[7] S. Pannier and W. Graf. Sectional sensitivity measures with artificial neural networks. In *9th LS-DYNA User Forum*, Bamberg, 2010.

[8] U. Reuter, M. Liebscher, and H. Müllerschön. Global sensitivity analysis in structural optimization. In *7th European LS-DYNA Conference*, Salzburg, 2009.

[9] U. Reuter, Z. Mehmood, C. Gebhardt, M Liebscher, H. Müllerschön, and I. Lepenies. Using LS-OPT for meta-model based global sensitivity analysis. In *8th European LS-DYNA Conference*, Strassburg, France, 2011.

[10] U. Reuter, Z. Mehmood, and C. Gebhardt. Efficient classification based methods for global sensitivity analysis. *Computers & Structures*, 110-111:79–92, 2012.

[11] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola. *Global Sensitivity Analysis: The Primer*. John Wiley and Sons Ltd, 2008.

[12] I. M. Sobol. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Computers in Simulation*, 55:271–280, 2001.

[13] I. M. Sobol and S. Kucherenko. Derivative based global sensitivity measures. *Procedia - Social and Behavioral Sciences*, 2(6):7745 – 7746, 2010.

[14] N. Stander, W. Roux, T. Goel, T. Eggleston, and K. Craig. *LS-OPT® User's Manual*. Livermore Software Technology Corporation, 2010.

[15] T. Tchaban, M. J. Taylor, and J. P. Griffin. Establishing Impacts of the Inputs in a Feedforward Neural Network. *Neural Computing & Applications*, 178:309–317, 1998.

[16] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.