Efficient detection of permissible design spaces in an early design stage

Marco Götz[1], Martin Liebscher[2], Wolfgang Graf[1]

[1]Institute for Structural Analysis, Technische Universität Dresden, 01062 Dresden, Germany
[2]DYNAmore GmbH, Industriestr. 2, 70565 Stuttgart, Germany

1 Introduction

The design of structures and processes is one of the main challenges for engineers. The traditional method 'try and error' (exploration of different variants) is due to an increasing number of design parameters not further possible. State of art is the application of various optimization methods, varying on the type of optimization task e.g. topology optimization, nonlinear parameter optimization or multi-criteria optimization under consideration of different constraints. Common for the most methods is the non-applicability in an early design stage, because of a lack of information. Today's design process is characterized by division of work and decomposition due to the multidisciplinarity of design. Therefore it is required to have knowledge about the structure or process of interest. In an early design stage this information is a priori not available. In this contribution, a method is presented to get additional information by detecting permissible design spaces. These design spaces gives the engineer independent continuous ranges for each design parameter and allow the necessary decomposition for further design stages. The detection of permissible design spaces is feasible by a visual interpretation only two-dimensionally. For n-dimensional tasks the presented approach is a possibility to overcome this problem. The approach detects for existing data sets in higher dimension permissible design spaces with regard to design constraints. A two-dimensional example illustrates the principle of the procedure and an example from automotive industry shows its applicability.

2 General approach

The general aim of this approach is to detect permissible design spaces to describe continuous permissible sets X^+ inside the design space $X \subset \mathbb{R}^n$ under consideration of equality and inequality design constraints ($g_j : \mathbb{R}^n \to \mathbb{R}$ and $h_k : \mathbb{R}^n \to \mathbb{R}$)

$$\begin{aligned} X^+ = \{x \in X \subset \mathbb{R}^n \mid \forall\, j \in \{1,\ldots,a_g\} : g_j(x_{g,j}(x)) \le 0 \\ \wedge \forall\, k \in \{1,\ldots,a_h\} : h_k(x_{h,k}(x)) = 0\}. \end{aligned} \tag{1}$$

Due to the computational effort calculating the equality and inequality constraints (e.g. FEM computations) it is only possible to evaluate them at a few points $\mathcal{H}_x \subset X$. Therefore the characterization of X^+ is only approximately possible. The (approximate) characterization of permissible design spaces can be done by a four step procedure, introduced in [7] and [11] and shown in Fig. 1.

Step 1 Generate point set $\mathcal{H} = \{(x, \xi(x)) \mid x \in \mathcal{H}_x\}$. With the help of this point set the permissible space should be well characterized. $\mathcal{H}_x$ can be generated by any DoE procedure, or reuse of existing data sets for $\mathcal{H}_x$, e.g. from further designs, reliability or sensitivity investigations. This possibility is one of the main benefits of the approach. The objective function $\xi(x)$ maps the input variables to the result variables and could be e.g. a FEA.

Step 2 Divide the point set $\mathcal{H}$ into permissible

$$\begin{aligned}\mathcal{H}^+ = \{(x,\xi(x)) \mid x \in \mathcal{H}, \forall\, j \in \{1,\ldots,a_g\} : g_j(x_{g,j}(x)) \le 0 \\ \wedge \forall\, k \in \{1,\ldots,a_h\} : h_k(x_{h,k}(x)) = 0\}\end{aligned} \tag{2}$$

and non-permissible

$$\begin{aligned}\mathcal{H}^- = \{(x,\xi(x)) \mid x \in \mathcal{H}, \forall\, j \in \{1,\ldots,a_g\} : g_j(x_{g,j}(x)) > 0 \\ \wedge \forall\, k \in \{1,\ldots,a_h\} : h_k(x_{h,k}(x)) \ne 0\}\end{aligned} \tag{3}$$

points.

Step 3 Classify the point set $\mathcal{H}_x^+ = \{x \mid (x,\xi(x)) \in \mathcal{H}^+\}$. This step is necessary, due to the inverse evaluation of the constraints non-connected permissible design spaces can occur. Information about the general behaviour of the design task will be provided and the computation of the permissible spaces is simplified. The classification can be accomplished by means of cluster analysis technologies, e. g. well established cluster algorithms: k-means [5] or fuzzy c-means [3].

Step 4 Detect permissible design spaces. Based on the classified point set $\mathcal{H}_x^+$, the permissible design spaces X^+ are described approximately.

The further parts of this contribution give some remarks about application of cluster analysis and an efficient method for detecting permissible spaces will be introduced.

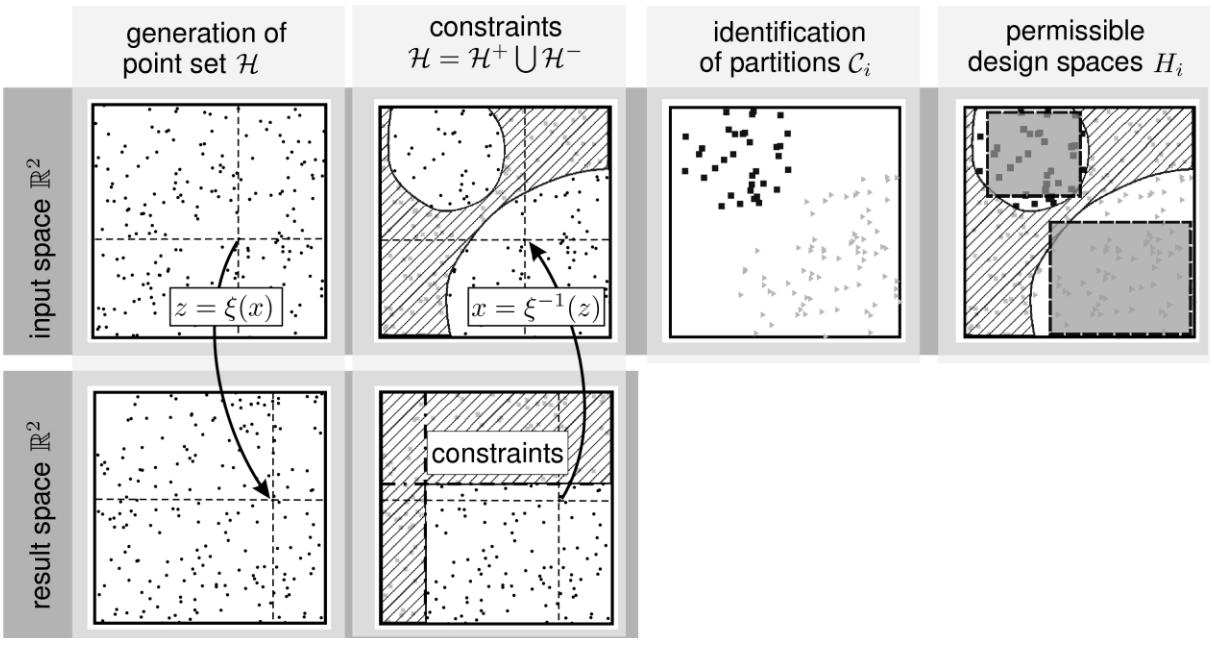
generation of point set $\mathcal{H}$
constraints $\mathcal{H} = \mathcal{H}^+ \bigcup \mathcal{H}^-$
identification of partitions $\mathcal{C}_i$
permissible design spaces H_i
input space $\mathbb{R}^2$
result space $\mathbb{R}^2$
$z = \xi(x)$
$x = \xi^{-1}(z)$
constraints

Fig. 1: General approach for detection of permissible design spaces in point sets $\mathcal{H}$ $(\xi : \mathbb{R}^2 \to \mathbb{R}^2)$

3 Identification of connected subspaces

The identification of connected subspaces (step 3 in the general approach, see section 2) can be done by means of cluster analysis, which detects groups in data. A group is characterized by a high amount of similar properties in a local area. The point set $\mathcal{H}_x^+$ is partitioned into n_c subsets $\mathcal{C}_i, i \in \{2,\ldots,n_c\}$, called clusters. The set of all clusters is called a cluster configuration K. Based on $\mathcal{H}_x^+$ various cluster configurations $K \in \mathcal{O}$ are possible

$$\mathcal{O} := \{K \subseteq \mathcal{P}^{(\mathcal{H}_x^+)}\}. \quad (4)$$

$\mathcal{P}$ indicates a power set.

Each cluster configuration K holds Eqs. (5) - (7).

$$\mathcal{C}_i \cap \mathcal{C}_j = \varnothing, \qquad \mathcal{C}_i \neq \mathcal{C}_j \text{ (in pairs disjoint)} \quad (5)$$

$$\mathcal{C}_i \neq \varnothing \text{ (non-empty)} \quad (6)$$

$$\bigcup_i^{n_c} \mathcal{C}_i = \mathcal{H}_x^+ \text{ (reproduces } \mathcal{H}_x^+) \quad (7)$$

In general, points inside a cluster $\mathcal{C}_i$ should be as homogeneous as possible and points of different clusters $\mathcal{C}_i, \mathcal{C}_j, i \neq j$, should be as heterogeneous as possible. Homogeneity characterizes the similarity of points p, q inside a cluster $\mathcal{C}_i$ and can be evaluated with distance measures $\mathrm{d} = \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$. Similarity increases, if the sum of distances between all points reaches a minimum

$$\sum_{p,q \in \mathcal{C}_i} d(p,q) \to \min. \quad (8)$$

A point set is heterogeneous, if the properties are as dissimilar as possible. The distances between all points should reach a maximum

$$\sum_{p \in \mathcal{C}_i, q \in \mathcal{C}_j, i \neq j} d(p,q) \to \max. \quad (9)$$

The computation of the best possible cluster configuration is an optimization task

$$D(K) \to \min, \text{ with} \quad (10)$$

$$D : \mathcal{O} \to \mathbb{R} : K \mapsto \sum_{\mathcal{C}_i \in K} \sum_{p \in \mathcal{C}_i} d(p, v_i)^2. \quad (11)$$

D describes the distances $d = (p, v_i)$ between points p from a cluster $\mathcal{C}_i$ and the empirical mean value v_i summed for all points inside the cluster $\mathcal{C}_i$ and for all clusters i.

In order to accomplish the classification, which divides the point set into clusters three inputs are necessary, see e.g. [3], [5]: the number of clusters n_c, the point of initialization and a distance measure. Due to the missing of a priori knowledge about the data this information cannot be provided. This means that a variation of parameters is necessary. The quality of resulting cluster configurations can be evaluated a posterior with measurements [12].

4 Detection of permissible design spaces

The results of Step 3 – evaluated cluster analysis – are point sets $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_i$ with an arbitrary structure in space. Based on these points permissible design spaces will be detected. For further usage of the design spaces it is necessary, to describe them through continuous spaces. In general it is possible to describe a point set $\mathcal{C}_i \in K$ with a convex hull. A convex hull holds the condition

$$(1-t) \cdot x_1 + t \cdot x_2 \in \mathcal{C}_i, \qquad x_1, x_2 \in \mathcal{C}_i, t \in [0,1]. \quad (12)$$

The computation of this hull is ambiguous and yields to interacting parameters. Another possibility is to detect hypercuboids to characterize the point set $\mathcal{C}_i \in K$. This approach has the benefit of non-interacting parameters, which results on the one hand in smaller permissible design spaces. But on the other hand it allows the separation of design parameters, which is necessary for an efficient further workflow. A hypercuboid is defined as

$$H = [a_1, b_1] \times \ldots \times [a_n, b_n] \subset X, \qquad a_j < b_j, \; j \in \{1, \ldots, n\}. \quad (13)$$

The main property of a permissible hypercuboid is

$$H \cap \mathcal{H}_x^- = \varnothing . \tag{14}$$

This part of the contribution shows different kinds of permissible spaces, gives an overview about possible approaches to detect hypercuboids and introduces a new approach to compute hypercuboids efficiently.

4.1 Kinds of permissible spaces

Due to the lack of information there are in general different possible shapes of permissible and non-permissible spaces. Fig. 2 shows structures for $X \subset \mathbb{R}^2$, whereby $\mathcal{H}_x^- = \{x \mid (x, \xi(x)) \in \mathcal{H}^-\}$. Common for engineering application are the properties of small corridors, holes and in higher dimensions a combination of them. Not preprocessed data can also have the property of non-sensitivity for some design variables. The occurrence of non-permissible spaces as patches is unusual for engineering data. Such a patch can easily be described by means of a bounding box of all non-permissible points and is further not considered.

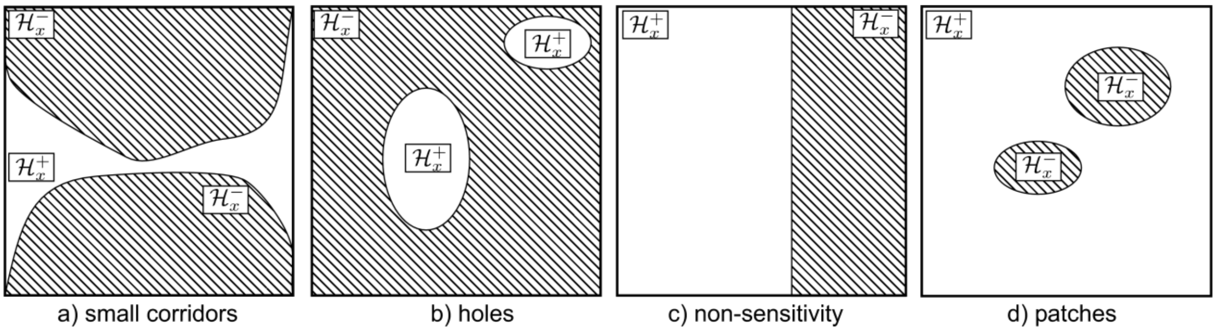

Fig. 2: Different shapes of permissible and non-permissible sets

4.2 Approaches for detection of permissible hypercuboids

The main idea of the presented approach is to detect independent, non-connected permissible spaces and describe them with hypercuboids. In general, the detection of permissible hypercuboids is split into two major phases. Phase I is the detection of a permissible hypercuboid inside the points of the cluster. Phase II tries to enlarge the permissible hypercuboid above the bounds of the relating cluster inside the design space.

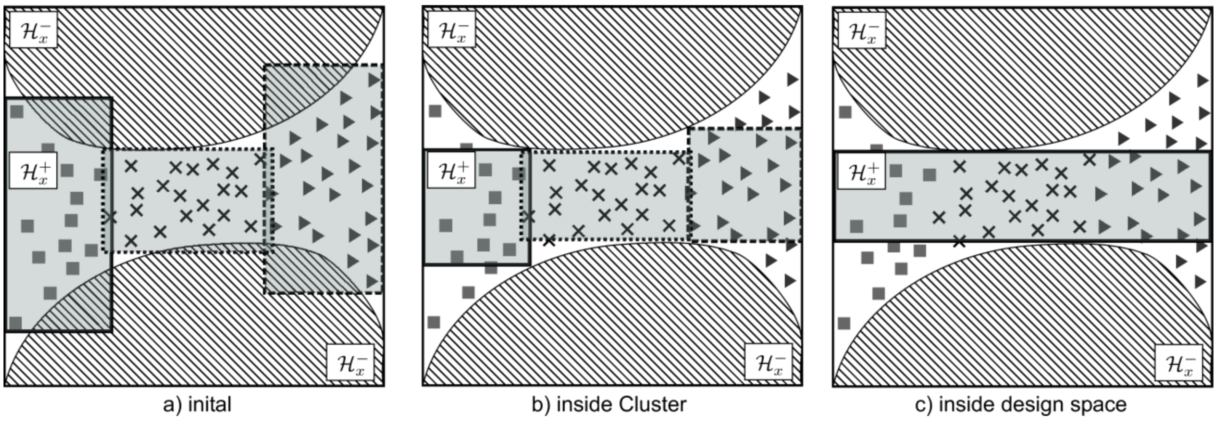

Fig. 2: Calculation of permissible hypercuboids

The initial hypercuboid can be found as the bounding box B around the existing points $x_1, \ldots, x_{p_i} \in \mathcal{C}_i$ of the cluster $\mathcal{C}_i$

$$\vartheta_i^{\min} = (\min(x_{1,1}, \ldots, x_{1,p_i}), \ldots, \min(x_{n,1}, \ldots, x_{n,p_i})) \tag{15}$$

$$\vartheta_i^{\max} = (\max(x_{1,1},\ldots,x_{1,p_i}),\ldots,\max(x_{n,1},\ldots,x_{n,p_i})) . \tag{16}$$

According to Eqs. (15) and (16) the bounding box can be written as

$$B_i = [\vartheta_{i,1}^{\min},\vartheta_{i,1}^{\max}]\times\ldots\times[\vartheta_{i,n}^{\min},\vartheta_{i,n}^{\max}] . \tag{17}$$

In general this hypercuboid does not always hold the permissibility condition in Eq. (14). Therefore in phase I the bounding box is shrank in order to obtain a permissible hypercuboid H_i. Additionally the computation can have the aim to find a permissible hypercuboid with maximal volume

$$V(H_i) \rightarrow \max \mid H_i \subseteq B_i \wedge H_i \cap \mathcal{H}_x^- = \varnothing \tag{18}$$

or with the largest possible smallest interval

$$\min(b_{i,j} - a_{i,j}) \; \forall \; j \in \{1,\ldots,n\} \rightarrow \max \mid H_i \subseteq B_i \wedge H_i \cap \mathcal{H}_x^- = \varnothing . \tag{19}$$

The second aim is applicable especially for small corridors.

Two major types characterize the different methods to shrink the initial bounding box. On the one hand there are algorithms shrinking the bounding box step by step, delineated in Fig. 3. These algorithms have a huge dependence on the point set $\mathcal{H}_x$ and the regulation selecting the dimensions and bounds (the first step). Algorithms working like this are efficient for small datasets for a small number of dimensions only. For a higher number of dimensions and bigger datasets it is disadvantageous that the amount of loop-runs and therewith the evaluation of permissibility is very high. On the other hand there are algorithms which compute the permissible hypercuboid by a systematic sweep for possible hypercuboids. The sweeping can be done with three approaches. The first approach a) has all dimensions of all points k as degree of freedom and is applicable for a small point set for a small number of dimensions. The second approach b) – interval approach – splits each dimension in p intervals, all interval combinations are possible. This approach is versatile and will be explained in detail in the next section. The last approach c) only has all permissible points k as degree of freedom. This has the benefit of independency from the number of dimension. The accuracy decreases for the mentioned possibilities, but the number of possible combinations, see Eq. (20) decreases extremely. For the parameter $n = 4, k = 50, p = 7$ the following numbers of combinations are obtained

$$a) \binom{k^n+1}{2} \approx 1.95\cdot10^{13} \qquad b) \binom{p^n+1}{2} \approx 2.88\cdot10^{6} \qquad c) \binom{k+1}{2} = 1275 . \tag{20}$$

```
Algorithm for shrink dimensionally
until $H_i \cap \mathcal{H}_x^- = \varnothing$ do:
    1. select dimension and
       bound due to regulations
    2. shrink selected bound of
       $H_i$ in selected dimension
       to the nearest
       non-permissible point
```

Fig. 3: Shrinkage algorithm

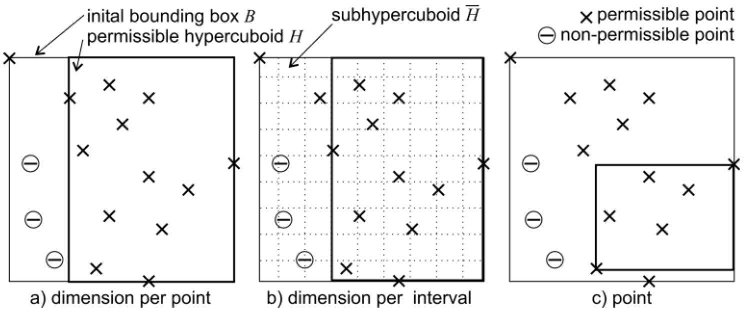

Fig. 4: Different approaches for systematic sweep

The phase II is necessary to enable detecting of permissible design spaces beyond the bounding box B of the relating cluster $\mathcal{C}_i$. Especially for small corridors, as shown in Fig. 2 is this option essential. E.g. the aim of maximizing the volume can be written as

$$V(H_i) \rightarrow \max \mid H_i \subseteq X \wedge H_i \cap \mathcal{H}_x^- = \varnothing . \tag{21}$$

This can result in overlapping hypercuboids, as shown in Fig. 2.

4.3 Interval approach

4.3.1 General formulation

Basis of this approach is also a bounding box $B = B_i$ of the relating cluster $\mathcal{C}_i$, Eq. (17). According to Eq. (13) the bounding box is the Cartesian product of the intervals I_i of all dimensions

$$B = I_1 \times I_i \times \ldots \times I_n \subset X, \quad I_i = [a_i, b_i], \quad i \in \{1, \ldots, n\}. \tag{22}$$

The splitting of the intervals I_i into p subintervals is defined as

$$I_i = I_{i,1} \cup I_{i,l} \cup \ldots \cup I_{i,p}, \quad I_{i,l} = [a_{i,l}, b_{i,l}], \quad l \in \{1, \ldots, p\}. \tag{23}$$

This can be interpreted as a discretization of B. The bounding box is separated into p^d sub-hypercuboids

$$\overline{H_k} = I_{1,l_1} \times I_{j,l_j} \times \ldots \times I_{n,l_n}, \qquad l_j \in \{1, \ldots, p\}, j \in \{1, \ldots, n\} \tag{24}$$

The bounding box can be written as

$$B = \bigcup_{k=1}^{p^d} \overline{H}_k\,. \tag{25}$$

The idea of the interval approach is to compute all possible combinations Eq. (20) of the paling space of two subhypercuboids. This enables a general analysis of the point set and results in good permissible hypercuboids. The combined hypercuboid H out of $\overline{H}_i$ and $\overline{H}_j$ can be written as

$$\begin{aligned} H = [\min\{a_{i,1}, a_{j,1}\}, \max\{b_{i,1}, b_{j,1}\}] \times [\min\{a_{i,k}, a_{j,k}\}, \max\{b_{i,k}, b_{j,k}\}] \times \\ \ldots \times [\min\{a_{i,n}, a_{j,n}\}, \max\{b_{i,n}, b_{j,n}\}], \qquad k \in \{1, n\} \end{aligned}. \tag{26}$$

The most time consuming operation is the permissibility check of the hypercuboid H. Therefore all subhypercuboids containing non-permissible points have to be computed.

$$\mathcal{K}^- = \left\{ \overline{H_i} \mid \overline{H_i} \cap \mathcal{H}^- \neq \varnothing \right\}, \quad i \in \{1, p^d\}. \tag{27}$$

The hypercuboid H is permissible, if

$$H \cap \bigcup_{\overline{H} \in \mathcal{K}^-} \overline{H} = \varnothing. \tag{28}$$

Among all combinations, an optimal hypercuboid can be identified, that fulfils the aim of the computations (Eq. (18) or Eq. (19)) in the best manner. To reduce the computational effort for a high number of dimensions an intelligent preselection of subhypercuboids of interest is necessary. One method is the analysis of the subhypercuboids containing permissible points only

$$\mathcal{K}^+ = \left\{ \overline{H_i} \mid \overline{H_i} \cap \mathcal{H}^+ \neq \varnothing \right\}, \quad i \in \{1, \ldots, p^d\}. \tag{29}$$

But there is a dependency on the point set $\mathcal{H}$. Another way of preselection is explained in section 4.3.3.

4.3.2 Numerical implementation

Due to the high amount of subhypercuboids for a high number of dimensions it is not possible to store all in memory. The definition of a subhypercuboid as the Cartesian product of the intervals Eq. (24) enables to compute the identity (ID) of $\overline{H}_i$ out of the intervals or vice versa, see Fig. (5). This allows a numerically efficient implementation with the use of positional notation system [6] to the base of the number of intervals p. Hence it is not necessary to store all subhypercuboids. The intervals (IV) are vectors of the general interval numbers. The calculation of the ID of $\overline{H}_i$ out of the intervals is

$$ID = \sum_{i=1}^{n} (IV_i \cdot p^{n-i})\,. \tag{30}$$

The calculation of the intervals out of the ID is

$$IV_i = \begin{cases} \bmod\left(\frac{ID}{p}\right), i = n \\ \bmod\left(\frac{\left\langle \frac{IV_{i+1}}{p} \right\rangle}{p}\right), \quad i \neq n, \quad \langle x \rangle = \max\{k \in \mathbb{Z} \mid k \leq x\} \end{cases}. \tag{31}$$

Because of the computation of all combinations, the check for permissibility has to be done very often. The discretization of the initial bounding box into subhypercuboids allows to check the permissibility of the hypercuboid H Eq. (28) resulting from the current combination Eq. (26) as

$$H \cap \overline{H}_j = \varnothing \mid \forall \overline{H}_j \in \mathcal{K}^-\,. \tag{32}$$

Due to the axis parallelism of both hypercuboids the proof of a zero intersection can numerical be treated as (under consideration of Eq. (13))

$$\begin{aligned} &a_l = \max\{a_{i,l}, a_{j,l}\}, b_l = \min\{b_{i,l}, b_{j,l}\} \\ &\exists\, l \in \{1,\ldots,n\} : a_l > b_l. \end{aligned} \tag{33}$$

If there exists at least one dimension l for all non-permissible subhypercuboids H_i, holding this condition the hypercuboid H_i is permissible.

4.3.3 Spherical enhancements

As remarked in section 4.3.1 it can be efficient to preselect the subhypercuboids and reduce the number of possible combinations. One possibility of preselection is to take only the subhypercuboids inside a n-dimensional ball S

$$S = \{x \in \mathbb{R}^n \mid \|x\| \leq r\} \tag{34}$$

into account. Because of the huge difference between the hypervolumes of a normed hypercuboid and the inside hypersphere (see Fig. 6) the number of combinations of subhypercuboids decreases strongly.

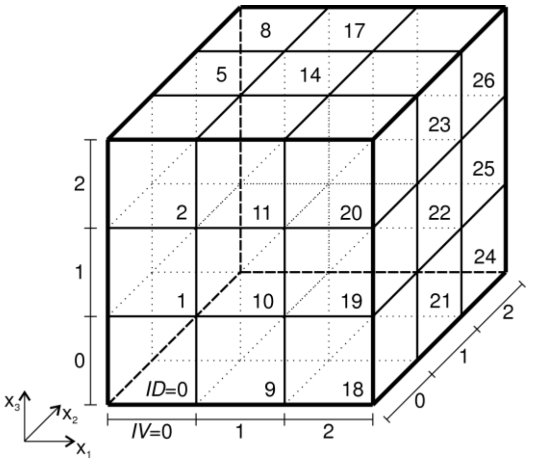

Fig. 5: Discretized hypercububoid ($\mathbb{R}^3$, p=3)

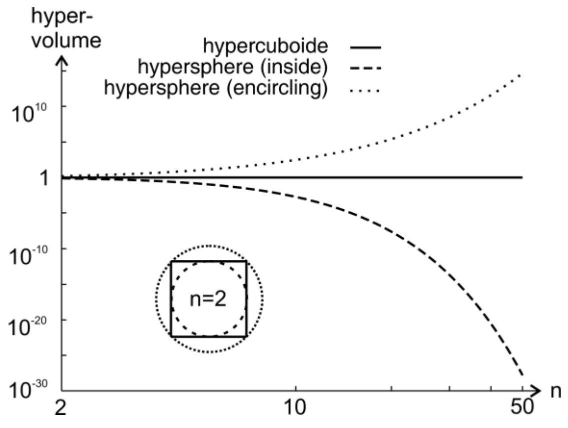

Fig. 6: Comparison of hypervolumes

The set of subhypercuboids can be written as

$$\mathcal{K}^{\circ} = \left\{ \overline{H_i} \mid \overline{H_i} \cap S \neq \varnothing \right\}, i \in \{1, p^d\} . \tag{35}$$

This reduction of numerical effort is only applicable for higher dimensions. For small number of dimensions the not considered hypervolume is too large.

5 Examples

5.1 Analytical function

In [8] the *Michalewicz* function Eq. (36) is used for performance evaluation of genetic algorithms. For visualization of the algorithms discussed in this contribution that function will be used two-dimensionally (n=2, m=10), see Fig. 7. In general it could be solved manually.

$$z = \xi(x) = -\sum_{i=1}^{n} \sin(x_i) \cdot \left(\sin\left(\frac{i \cdot x_i^2}{\pi} \right) \right)^{2 \cdot m} \qquad x_i \in [0, \pi] \tag{36}$$

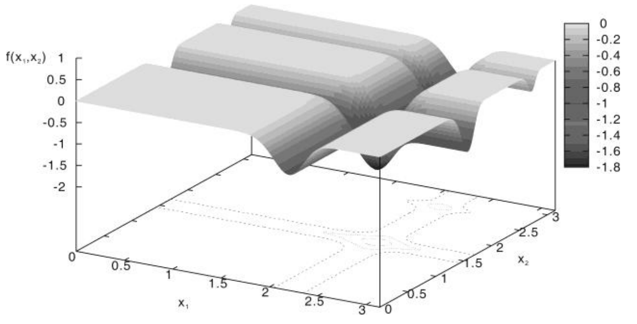

Fig. 7: Michalewicz function (two-dimensionally)

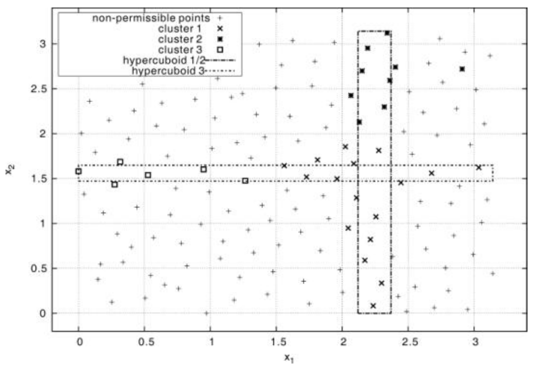

Fig. 8: Permissible design spaces for $z < -0.4$

The aim is to detect permissible design spaces with respect to the design constraint $z < -0.4$. Therefore 150 points $\mathcal{H}$ are generated with a DoE. These points are subdivided into 32 permissible $\mathcal{H}^+ = \{(x, \xi(x)) \mid x \in \mathcal{H}, z < -0.4\}$ and 118 non-permissible $\mathcal{H}^- = \{(x, \xi(x)) \mid x \in \mathcal{H}, z \geq -0.4\}$ points. The clustering and evaluation of several cluster configurations of the permissible points leads to the best configuration: three independent clusters, see Fig. 8. Permissible design spaces, represented by hypercuboids are calculated for each cluster, with the aim to maximize the volume. These hypercuboids are limited to the bounds of the particular cluster (phase I). The possibility to connect and enlarge the hypercuboids (phase II) leads to better permissible design spaces, whereby the hypercuboid for cluster 1 is equal to that of cluster 2. The hypercuboid related to cluster 3 was enlarged. Summarizing, the detection of permissible design spaces results into two hypercuboid, shown in Fig. 8.

5.2 Automotive design

To demonstrate the capabilities of the presented procedure an eight-dimensional numerical function is considered. This function is an evaluation of US-NCAP rating, according to the *overall Vehicle Safety Score* (VSS) [1]. The results of the calculation are *static stability factors* (SSF) and the stars-evaluation for frontal, side and rollover tests. The result of interest for this study is the *side combined star*

$$z = f(x_1, x_2, \ldots, x_8) . \tag{37}$$

The input parameters relevant for the side combined star result are eight injury measures of passengers, e.g., abdomen force or rib deflection, listed in Fig. 9.

input parameter	design space
sidepole front seat HIC36 [-]	0 … 2000
sidepole front seat Pelvis Force [kN]	0 … 50
sideMDB front seat HIC36 [-]	0 … 1000
sideMDB front seat Max Rib Deflection [mm]	0 … 500
sideMDB front seat Abdomen Force [kN]	0 … 20
sideMDB front seat Pubic Force [kN]	0 … 20
sideMDB rear seat HIC36 [-]	0 … 1000
sideMDB rear seat Pelvis Force [kN]	0 … 20

Fig. 9: Input parameters

DoE	$z > 5^*$
1.	0.00 %
2.	0.92 %
3.	21.60 %
4.	99.48 %
(4. only for validation)	

Fig. 10: Convergence of DoEs

The aim of the investigation is to find permissible design spaces holding $z = 5^*$. For a better numerical stability, the not existing decimals of the star result are allowed. A first DoE with 10,000 sample points was computed inside the design space given in Fig. 9. This point set contains no result which fulfils the constraint, hence this demonstrates the high influence of dimensionality for this example. To overcome this, the DoE was repeated, with decreasing design spaces. The smaller design spaces can be calculated with an iterative application of the introduced approach. The success of the iteration can be seen for this example Fig. 10. The examples result is one permissible design space, see Figs. 11, 12. The used visualization technology – parallel coordinate plot [4], allows a simultaneous visualization of the permissible design space for all design parameter. This overview enables the evaluation of the result and a more detailed understanding of the model behaviour. The volume of that design space is $V_{DoE3} = 2.3 \cdot 10^{9}$ which is about $V_{DoE3} = 5.75 \cdot 10^{-9} \cdot V_{DesignSpace}$. The huge difference in volume between the initial design space and the permissible design space explains the requirement of the iterative application of the method. The design engineer has now the possibility to select the best value (dependent on other criteria's besides permissibility) for each design variable independent from all others, inside the detected ranges. This enables easy work sharing procedures for the more detailed design stages.

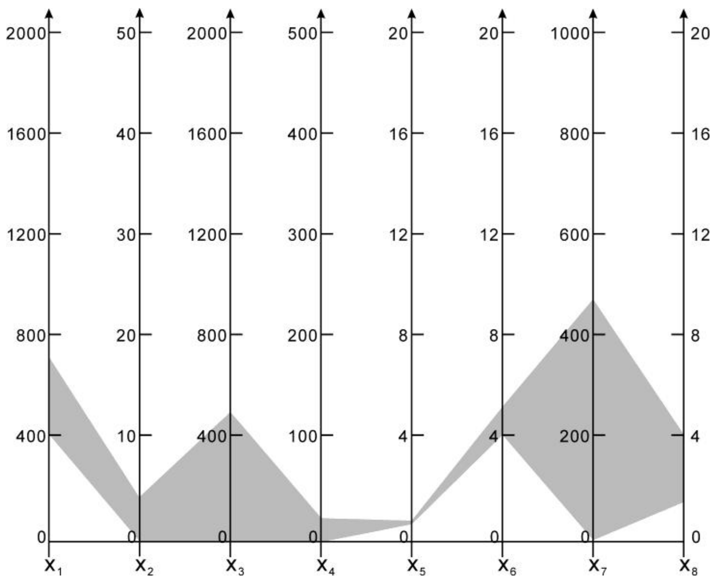

Fig.11: Parallel coordinate plot of the permissible hypercuboid (scaled to the initial design space)

parameter	range
x_1	399.0 … 715.5
x_2	0.0 … 4.13
x_3	0.0 … 486.4
x_4	0.2 … 16.7
x_5	0.5 … 0.7
x_6	3.9 … 4.9
x_7	1.7 … 466.9
x_8	1.6 … 4.0

Fig. 12: Permissible design space

6 Summary

The introduced approach for detecting permissible design spaces can be summarized by four main steps:

1. Generate point set $\mathcal{H}$
2. Divide the point set $\mathcal{H}$ into permissible $\mathcal{H}^{+}$ and non-permissible points $\mathcal{H}^{-}$
3. Classify the point set $\mathcal{H}_x^{+}$
4. Detect of permissible design spaces H_i

This procedure can handle any kind of data, which makes it versatile. The classification is realized by cluster algorithms. Due to the preconditions for cluster algorithms an evaluation of several cluster configurations selects the best ones and reduces the numerical effort for computing the permissible design space. In this contribution different possibilities for detecting permissible hypercuboids are presented. The interval approach is introduced. The capability of the approach is demonstrated by two examples. The example in 5.2 and the spherical enhancement in section 4.3.3 show the influence of dimensionality. This effect exists due to the Curse of Dimensionality [2] and demonstrates the importance of reasonable selection of parameters and the research for sensitivity. Sensitivity analysis detects main influence parameter and can reduce the influence of dimensionality.

Common for all engineering application is uncertainty for parameters and models. The approach is extensible for uncertain data [10]. The uncertain data can be modelled with fuzziness, randomness or fuzzy randomness [9]. This extension would allow the consideration of further aims computing the permissible design space, as reliability or robustness. Furthermore the introduced procedure enables the consideration of static data only. For regarding time-dependent or other functional behaviour other algorithms for classification and detection of permissible hypercuboids are necessary, but the general sequences are the same.

7 Literature

[1] *[Docket No. NHTSA-2006-26555]: Consumer Information - New Car Assessment Program* (2008), National Highway Traffic Safety Administration – Department of Transportation.

[2] Bellman, R. E.: “Dynamic Programming”, Princeton University Press, 1957

[3] Bezdek, J. C., Ehrlich, R., and Full, W.: “Fcm: The fuzzy c-means clustering algorithm”, Computers and Geosciences 10, 1984, 191–203

[4] Inselberg, A.: “The plane with parallel coordinates”, The Visual Computer 1, 1985, 69–91

[5] Jain, A. K.: “Data clustering: 50 years beyond k-means”, Pattern Recognition Letters 31, 2010, 651 – 666

[6] Knuth, D.: “The Art of Computer Programming, Volume 2: Seminumerical Algorithms”, 3 ed., Addison-Wesley, Massachusetts, 1997

[7] Liebscher, M.: “Dimensionierung und Bewertung von Tragwerken bei Unschärfe – Lösung des inversen Problems mit Methoden der explorativen Datenanalyse”, Veröff. Institut für Statik und Dynamik der Tragwerke, Heft 13, TU Dresden, 2007

[8] Michalewicz, Z.: “Genetic algorithms + data structures = evolution programs”, 3 ed., Springer-Verlag, London, 1996

[9] Möller, B., and Beer, M.: “Fuzzy Randomness - Uncertainty in Civil Engineering and Computational Mechanics”, Springer-Verlag, Berlin, 2004

[10] Pannier, S.: “Effizienter numerischer Entwurf von Strukturen und Prozessen bei Unschärfe” Veröff. Institut für Statik und Dynamik der Tragwerke, Heft 23, TU Dresden, 2011

[11] Pannier, S., Grossenbacher, K., Liebscher, M., Ganser, M., Graf, W., Müllerschön, H., Lipp, A., and Kaliske, M.: “Increasing reliability of metal forming processes in early design stages”, 9. LS-DYNA Anwenderforum, Bamberg, 2010

[12] Piotrow, A., Liebscher, M., Pannier, S., and Graf, W.: “Grouping detection of uncertain structural process by means of cluster analysis”, 7th European LS-DYNA Conference, Salzburg, 2009

Acknowledgement

The authors gratefully acknowledge the financial support of Federal Ministry of Economics and Technology on the basis of a decision by the German Bundestag according to the Central Innovation Programme SME (ZIM).