Identifying Critical Demand Scenarios for the Robust Capacitated Network Design Problem using Principal Component Analysis

Andreas Bley and Philipp Hahn

Institute of Mathematics, University of Kassel, Germany

November 30, 2021

Abstract

In this paper, we consider the single-commodity robust network design problem. Given an undirected graph with capacity installation costs on its edges and a set S of scenarios with associated flow balance vectors that represent different scenarios of node supplies and demands, the goal is to find integer edge capacities that minimize the total installation cost and permit a feasible single commodity flow for each scenario. This problem arises, for example, in the design of power networks, which are dimensioned to accommodate many different load scenarios.

We propose a new method to identify a small subset S' of the given scenarios, such that solving the robust network design problem for the smaller scenario set S' leads to almost the same capacities as solving it for the full scenario set S. By considering only the scenarios in S', the size of the model that needs to be solved can be reduced substantially, while the error introduced by neglecting the remaining scenarios is kept very small. Our method only employs simple techniques from statistical data analysis, namely principal component analysis (PCA), and convex hull computations in low dimensions. Thus, its computational effort is very small and it is easily applicable to more complex network design problems.

We evaluate the effectiveness of the method in computational experiments for instances stemming from offshore power grid planning or telecommunication networks. Our results show that the proposed techniques are indeed well suited to identify small scenario subsets that lead to significantly reduced models with high quality solutions.

Keywords: Robust network design, Scenario reduction, Principal component analysis

1 Introduction

We consider the single-commodity robust capacitated network design problem, which arises, for example, in the planning of power grids and other transport networks that need to be operational in various different load scenarios. Given a graph and a finite set of flow balance vectors, which describe the node supplies and demands of the different load scenarios, the task is to find minimum cost edge capacities that are sufficiently large to (non-simultaneously) permit single-commodity flows for all scenarios' flow balances. In this paper, we focus on instances that contain a large number of scenarios and we show how the number of scenarios whose flows need to be considered in the solution process can be effectively reduced using techniques from statistical data analysis.

More formally, the (single-commodity) robust capacitated network design problem RCND is defined as follows: We are given an undirected Graph (V, E) and, for each edge $ij \in E$, a capacity unit u_{ij}, which can be installed in integer multiples at cost c_{ij} per unit on edge ij. We denote the number of nodes by $n = |V|$. Each edge $ij \in E$ can carry flow in both of its directions, which are represented by the directed arcs (i, j) and (j, i). The set of all directed arcs corresponding

to the edges in E is denoted by $A = \{(i,j), (j,i) \mid ij \in E\}$. Furthermore, we are given a finite set of scenarios $S = \{1, \ldots, m\}$, $m \in \mathbb{N}$, and, for each scenario $s \in S$, a flow balance vector $d^s = (d^s_j)_{j \in V}$. The value $d^s_j \in \mathbb{R}$ denotes the supply or demand of node j in load scenario s. The task is to install integer multiples of the capacity units u_{ij} on the edges $ij \in E$ such that for each $s \in S$ a single-commodity flow with node balances d^s not exceeding the installed edge capacities exists and the total installation cost is minimized.

Using non-negative integer variables $x_{ij} \in \mathbb{Z}$ for the number of capacity units installed on edges $ij \in E$ and non-negative continuous variables $f^s_{(i,j)} \in \mathbb{R}$ for flows sent via arcs $(i,j) \in A$ in scenarios $s \in S$, we obtain the following natural mixed-integer linear programming formulation of the problem:

$$\begin{aligned}
\text{minimize} \quad & \sum_{ij \in E} c_{ij} x_{ij} && && \text{(RCND)}\\
\text{s.t.} \quad & f^s_{(i,j)} + f^s_{(j,i)} \leq x_{ij} u_{ij} && ij \in E,\, s \in S && (1)\\
& \sum_{(j,i) \in \delta^-(i)} f^s_{(j,i)} - \sum_{(i,j) \in \delta^+(i)} f^s_{(i,j)} = d^s_i && i \in V,\, s \in S && (2)\\
& f^s_{(i,j)} \geq 0 && (i,j) \in A,\, s \in S\\
& x_{ij} \geq 0 && ij \in E\\
& x_{ij} \in \mathbb{Z} && ij \in E
\end{aligned}$$

The objective is to minimize the sum of all capacity installation costs. The capacity constraints (1) ensure that on each edge the installed capacity is sufficiently large to accommodate the total flow sent along this edge in each scenario. The flow balance constraints (2) ensure that in each scenario the corresponding flow satisfies the given node supplies and demands. Note that the edge capacity variables x_{ij} are independent of the scenarios, while the flows f^s_{ij} depend on the scenario.

In practice, the RCND problem arises in the planning of power grids, for example. In this context, the nodes of the graph represent the hubs and buses of a power grid and edges represent the potential transmission lines, whose capacities need to be dimensioned. Often, the number of potential lines and, thus, the network topology considered in the planning of these networks is very large. Furthermore, the varying power consumption of customers and the volatile production, especially of renewable generation units, lead to many time-dependent load scenarios. In strategic network design and expansion planning, these load scenarios are typically represented by time-series spanning a whole year at a resolution of 1 hour. In operational planning, finer resolutions and shorter time periods are used. For problem instances of practical interest, the ILP models obtained for these problems are very hard to solve, because they involve a huge number of load scenarios and, thus, are extremely large.

In this paper, we present a novel approach to select a small subset S' of the given scenario set S, such that solving the RCND problem for the smaller scenario set S' leads to almost the same installation costs as solving it for the full scenario set S. One easily verifies that only scenarios s' whose flow balance vector $d^{s'}$ is a vertex of the convex hull $C = \text{conv}\{d^s \mid s \in S\} \subset \mathbb{R}^n$ of all scenarios' flow balances need to be considered in the RCND problem. If the chosen edge capacities permit single-commodity flows f^{s_1} to f^{s_k} for the flow balance vectors d^{s_1} to d^{s_k}, respectively, then, for any convex combination $d = \alpha_1 d^{s_1} + \ldots + \alpha_k d^{s_k}$ with $\alpha_1 + \ldots + \alpha_k = 1$ and $\alpha_i \geq 0$ of the flow balances the corresponding convex combination $f = \alpha_1 f^{s_1} + \ldots + \alpha_k f^{s_k}$ of the flows will be a valid flow that does not exceed the chosen edge capacities either. Hence, removing from S all scenarios that do not define a vertex of C will not change the set of valid edge capacities and, thus, the optimal solution of the RCND problem. However, determining the convex hull of large point sets is computationally very costly, unless the dimension is very small (less than 8). Therefore, in practice the vertices of C cannot be calculated directly. Even more, in larger networks typically most of the given flow balance vectors actually do define vertices of C. Due to the rather high dimension n of the space of flow balances, already small variations in the flow balances lead to extremal points. Thus, eliminating only the scenarios not corresponding to vertices of C barely reduces the size of S in problem instances of practical interest.

The method we propose to reduce the scenario set S relies on principal component analysis (PCA), which is a very fast algorithm from statistical data analysis, and convex hull computations in low dimensional spaces. The details of the selection procedure are described in Section 3. Roughly, the method works as follows. In a first step, we compute the directions of largest variation within the set of flow balance vectors of the given scenarios. Using the PCA method, we obtain the principal components, which are the statistically independent, i.e., orthogonal directions of largest variation of the flow balance set, in order of decreasing variance in these directions. Typically, very few of the first principal components suffice to capture most of the variance in the original set of flow balance vectors. If this is the case, the n-dimensional flow balance data can be projected onto the lower dimensional subspace spanned by these directions without losing too much information about its variance. In the second step, we project all given flow balance vectors onto several subspaces, each spanned by one or more of the most important principal components. For each of these projections, we then compute the vertex set of the convex hull and eventually add the scenarios corresponding to the vertices to S'. Note that only scenarios, whose flow balances define vertices in the original n-dimensional space of the flow balances can define vertices in the lower dimensional projections, and that the projections considered in the proposed method are spanned by the directions of largest variation within the flow balances. Thus, the set S' of chosen scenarios only contains scenarios, whose flow balances indeed define vertices in the original space of the flow balances and which are also extremal in the most critical directions. Our experiments show that for many real-world problem instances the scenario set S' identified this way suffices to achieve a good approximation of the RCND solution obtained for the full scenario set S, although only a very small number of the given scenarios is chosen. In a number of cases, the chosen scenarios in fact lead to the same optimal value as the original problem.

A comprehensive overview on single-commodity robust network design can be found in [13]. Among other results, an alternative ILP formulation using cut set inequalities is introduced and used for solving the problem in a branch and cut algorithm.

As mentioned before, the ILP models we are dealing with are extremely large, in particular, due to the large sets of scenarios. Thus, one could benefit from reducing the scenario set to a subset of critical scenarios. In [8], the concept of domination between traffic matrices is introduced. For multi-commodity traffic matrices D_1 and D_2, D_1 is said to dominate D_2 if for any capacitated network that accommodates a feasible flow for D_1, the network can also accommodate a feasible flow for D_2. This concept of dominance is generalizable to single commodity problems (and multiple demand vectors) in a straight forward manner. For the RCND, it would mean that $d_1, \ldots, d_n$ are said to dominate d_{n+1} if any capacity installation that is feasible for the first n demand vectors also can accommodate a feasible flow for demand vector d_{n+1}. In fact, we have already discussed that each demand vector in the interior of the convex hull of all demand vectors is dominated by the vertex set of the convex hull. The dominance criterion in [8] was used in [6] to reduce the uncertainty set of a multi-commodity robust network design problem. In our setting, removing all scenarios that lie in the interior of the convex hull is an exact reduction of the problem, in the sense that any capacity installation is feasible for the reduced model if and only if it is also feasible for the full model. However, as discussed before, it is computationally difficult to determine the convex hull in high dimensions and thus, in practice often inexact methods are used to reduce the scenario set.

In power grid planning, a selection of critical scenarios is often done by clustering techniques, such as k-means clustering, etc.. First, the given scenarios are grouped into clusters according to some similarity measure. Then, a single representative of each cluster is included in the set S'. This representative either is chosen from the given scenario set or it is constructed as a new artificial scenario by averaging the scenarios in its cluster. When operational costs are involved, additional weight factors corresponding to the number of scenarios in the clusters are used to weight the operational costs of the representatives according to the size of the clusters they represent. The methods proposed in the literature typically aim at choosing or constructing a representative that is at the center of the cluster with respect to the chosen similarity measure. Extremal scenarios thus will not be included in S', because they are not centers of their cluster. Solving a robust capacitated

network design problem for a reduced scenario set S' obtained via clustering methods often leads to a network that is inaptly dimensioned for the extremal scenarios contained in the original scenario set S. A simple approach commonly used to identify at least some extremal scenarios is to determine for each node of the underlying graph the scenario attaining the maximum demand or supply at that node. These peak scenarios then are used either alone or together with the cluster centers to solve the network design problem. Note, when operational costs are considered, reductions that use only clustering methods to find representatives typically underestimate the investment costs. Thus, adding the peak demand scenarios to the set of cluster representatives can guide the models towards better solutions. An overview of common clustering techniques is given in [12], their detailed application in energy systems modeling is discussed in [5].

Another common method for finding critical scenarios is outlier detection. However, outlier detection algorithms are inappropriate to identify the scenarios that are critical for the dimensioning of the network. Similar to the clustering approaches, these methods group the given scenarios in one or more clusters according to some similarity measure. Scenarios that are not similar enough to any of the clusters then are considered to be an outlier. These methods effectively identify scenarios that are (very) different from the largest scenario clusters, but they do not aim for extremal scenarios at the border of the convex hull. Hence, they might identify outlier scenarios in the center of the convex hull, if these are isolated scenarios. Even worse, they might not identify a large number of scenarios that are critical for the network capacities, because these scenarios form one or a few clusters containing too many scenarios. An overview on outlier detection methods can be found in [16] and [15].

In the clustering and outlier detection methods described above, the demands and supplies of the given scenarios are usually considered only independently at single nodes or aggregated for some node sets. More general correlations of the demands and supplies, which are very strong in real-world networks, are not taken into account. The PCA-based approach we propose in this article explicitly analyses and exploits the correlations among all scenarios and node demands. This leads to a much better insight in what is critical for the whole network.

Recently, Bukenberger and Webster [1] described an approach using PCA to identify representative scenarios for the transmission network expansion planning problem TNEP. In this approach, first the performance of each scenario is estimated by solving the single scenario optimization problem for each scenario and several candidate networks. Then, PCA is applied on the resulting objective values of these optimization problems in order to identify the scenarios that eventually will be used as representative scenario set for the solution of the multi-scenario TNEP problem. This approach differs substantially from the method we propose in this paper. Bukenberger and Webster's approach tries to identify the scenarios that cause expensive capacity installations and, for this purpose, relies on solving a computationally expensive single scenario optimization problem for each scenario in the first stage. Our approach, on the other hand, applies PCA directly to the raw demand data in order to identify extremal load scenarios, which is computationally much less demanding.

The remainder of this article is organized as follows. In Section 2, we briefly introduce the PCA method. Subsequently, we present several strategies to choose the scenarios for the critical set S' in Section 3. In Section 4, we report the results of a computational study conducted to evaluate the effectiveness of the proposed method for a number of real-world and benchmark instances. Section 5 finally contains a summary and an outlook on potential future research.

2 Principal Component Analysis

Principal component analysis (PCA) is a classical method from statistical data analysis for finding the directions of largest variation in multidimensional random data or measurements. These directions are called the principal components. Often, PCA is used to reduce the dimension of the data by projecting it on a lower dimensional space spanned by only a few principal components, which contain most of the variance of the data. If the large variations in the data correspond to some meaningful signal and small variations to additional noise, this approach can be very

effective in filtering the noise from the signal data. One of the major advantages of the method is its very small computational overhead, even huge data sets can be easily analyzed.

Let us briefly describe the PCA method. We are given a set of m measurements of some n-dimensional data. Alternatively, these measurements can be viewed as m realizations of an n-dimensional random variable. The measurements are encoded as a matrix

$$D = \begin{pmatrix} d_{11} & \dots & d_{1n} \\ \vdots & & \vdots \\ d_{m1} & \dots & d_{mn} \end{pmatrix} \in \mathbb{R}^{m \times n}.$$

For each $i \in \{1, \dots, m\}$, the n-dimensional row-vector $D_{i,\bullet} = (d_{i,1}, \dots, d_{i,n})$ encodes the i-th measurement of the n-dimensional data or, alternatively, realization of the multivariate random variable. The values in column j, $D_{\bullet,j} = (d_{1,j}, \dots, d_{m,j})^T$, $j \in \{1, \dots, n\}$, are the m measurements or realizations of the j-th data or random variable.

In the optional first step of the PCA method, the data is standardized in such a way that each random variable has an expected value of 0 and a variance of 1. The purpose of this step is to eliminate the effects that the different scales and offsets of the variables can have on the numerical variation in the data set. As differences in scales and offsets may have a huge impact on the observed directions of largest variation, a standardized, scale-free view on the data is preferable in many applications. Nonetheless, this step is optional and it strongly depends on the application which type of standardization is beneficial.

In the second step, the covariance matrix $C = \frac{1}{m-1} D \cdot D^T$ of the (standardized) random variables, its eigenvalues λ_j and the corresponding eigenvectors p_j, $j \in \{1, \dots, n\}$, are calculated. The eigenvalues are sorted in order of non-increasing eigenvalues, i.e., such that $\lambda_1 \geq \dots \geq \lambda_n$. The j-th eigenvector in this order is called the j-th principle component (PC). As the covariance matrix C is positive semi-definite, the eigenvectors p_1 to p_n form an orthogonal system, i.e., each principal component is orthogonal to all other principal components, and that the matrix

$$P = \begin{pmatrix} p_1 & \dots & p_n \end{pmatrix},$$

formed by the eigenvectors diagonalizes the covariance matrix. This also implies that the sum of all eigenvalues yields the total variance in the given data. Even more, each single eigenvalue λ_j, $j \in 1, \dots, n$, measures how much variance can be explained by the j-th principal component, i.e., is retained in the orthogonal projection of the m measurements onto the space spanned by the corresponding eigenvector p_j.

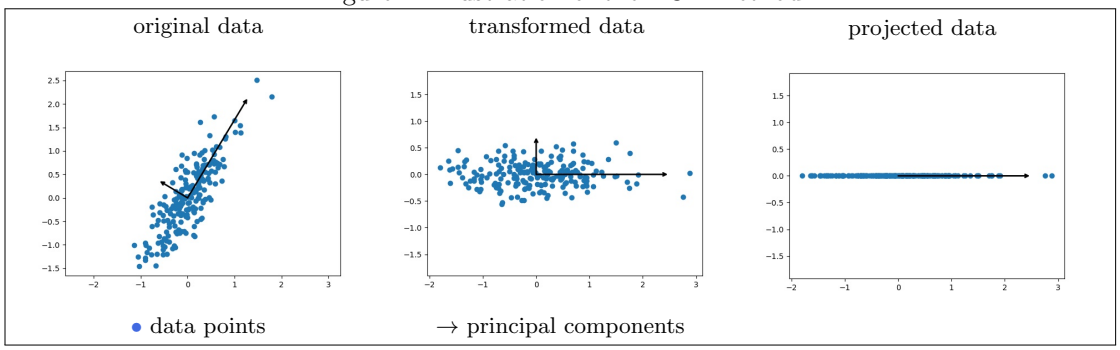
In Step 3, one typically transforms the data from the original coordinate system to the new orthogonal system defined by the principle components. This is easily achieved by multiplying D with the matrix P. We obtain the transformed data

$$\begin{aligned} \bar{D} &= D \cdot P \\ &= \begin{pmatrix} \bar{d}_{11} & \dots & \bar{d}_{1n} \\ \vdots & & \vdots \\ \bar{d}_{m1} & \dots & \bar{d}_{mn} \end{pmatrix} \end{aligned}$$

with $\bar{D}_{\bullet j}^T \cdot \bar{D}_{\bullet i} = 0$ for $i \neq j$. Figure 1 illustrates the PCA method for a 2 dimensional data set. The first figure shows the given 2-dimensional data with respect to its original dimensions. The data that was shifted to have an expected value of zero in both dimensions, but not standardized to variance 1. The second figure shows the same data with respect to the coordinates given by the two principal components, which are also illustrated as the two orthogonal directions of greatest variation in the first figure.

In the last step, the m measurements are projected to a low dimensional space spanned by a subset of the principal components, typically the first k principal components that together cover a prescribed percentage of the total variance. This can be done very easily in the coordinate system defined by PCs by considering only the coordinates of the chosen principal components

Figure 1: Illustration of the PCA method



and setting all other coordinates to zero. The third figure in Figure 1 illustrates the projection of the (transformed) data set on the first principal component, which explains 92% of the total variance within the data.

3 Scenario Selection Strategies

In this section we describe our approach to reduce the scenario sets of robust capacitated network design problems stemming from real-world applications.

Suppose we are given an instance of the RCND problem with flow balance vectors $d^s \in \mathbb{R}^n$, $s \in S$, with S being a scenario set that is very large. Our goal is to choose a small subset $S' \subseteq S$ of scenarios, whose flow balance vectors lead to (almost) the same investment cost for edge capacities as the full scenario set S.

In the first phase of our approach, we identify the directions of largest variance within the given flow balances. For this purpose, we consider the given flow balance vectors d^s as row-vectors and apply the PCA method described in the previous section to the matrix

$$D = \begin{pmatrix} d^1 \\ \vdots \\ d^m \end{pmatrix} = \begin{pmatrix} d^1_1 & \dots & d^1_n \\ \vdots & & \vdots \\ d^m_1 & \dots & d^m_n \end{pmatrix}$$

formed by these rows. Note that each column corresponds to exactly one node of the node set V.

In the optional standardization step of the PCA method, we only shift the flow balances such that the resulting empirical mean values are zero at each node. This is done by subtracting from the flow balance value d^s_v of each scenario s and each node v the mean flow balance $\frac{1}{m}\sum_s d^s_v$ at node v over all scenarios. However, we do not rescale the flow balance values to obtain the same variance of the flow balances at all nodes, because the absolute differences of the flow balances directly relate to the values of the flows and, consequently, of the capacities needed in the network. Hence, it is important to take into account the different orders of magnitude in the flow balances at different nodes when searching for the scenarios that are determining the necessary edge capacities.

For the sake of notational simplicity, we assume that the given flow balances d^s already have an empirical mean of zero at each node and, thus, matrix D is used in the following steps of our algorithm. The PCA method then yields the principal components $p_j \in \mathbb{R}^n$, $j \in \{1, \dots, n\}$, with their corresponding eigenvalues $\lambda_j \geq 0$ in non-increasing order of the eigenvalues. As mentioned in the previous section, the principal components are the directions of the largest variation within the flow balances that are orthogonal, i.e., statistically independent from each other: The first principal component is the direction of the largest variation, the second principal component is the direction of the largest variation that is orthogonal to the first one, etc.. Moreover, the corresponding eigenvalue λ_j measures the variance of the flow balances in the direction given by the principal component p_j.

In the second phase of our approach, we then select the scenarios for the subset S' used in the solution of the RCND problem. On the one hand, we wish to retain as much information about the variance in the flow balances as possible while, on the other hand, choosing only a small number of scenarios. Therefore, we choose scenarios that are extremal with respect to the directions given by the principal components with the largest eigenvalues. As these components represent the directions of the largest variations in the flow balances, and often a few of these directions together suffice to capture a very large percentage of the overall variation, the scenarios that are extremal with respect to only these direction often very well represent the overall variation in the full scenario set.

We have developed two strategies to select the scenarios for S'.

Min-Max Strategy (MM). In our first strategy, we first choose the $k \in \mathbb{N}$ principal components with the largest eigenvalues. For each of these principal components, we then pick the two scenarios that attain the minimum and maximum with respect to the direction given by the principal component, i.e.,

$$S' = \bigcup_{i \in 1,\dots,k} \left\{ \arg\max_{s \in S} \{D_{s\bullet} \cdot p_i\}, \arg\min_{s \in S} \{D_{s\bullet} \cdot p_i\} \right\} .$$

In the projection onto the space spanned by a single chosen principal component p_i, all scenarios lie between the two chosen extreme scenarios. Hence, the flow balances' range of variation in such a direction is fully retained in the set S'.

The computational effort for picking the two extreme scenarios per principal components is very small. Thus, this strategy is well-suited if the number of principal components that are necessary to cover the desired percentage of the overall variation in the given scenario set is rather large and one cannot afford to pick too many scenarios per principal component. We denote this method by MM.

The main drawback of this method is that at most 2 scenarios per principal component, in particular the most important principal components, are chosen due to the fact that each principal component is considered only individually and independent of other principal components. Recall that the principal components are orthogonal to each other. Hence, the variation of the flow balances in the direction(s) of the first principal component(s) is completely ignored when considering the variation with respect to the following principal components. In particular, large variations in directions that are non-trivial combinations of the principal components are underestimated in the projection onto the spaces spanned by single principal components and not covered by picking only the extreme scenarios for single principal components. In many real-world instances, the first principal component alone already accounts for more than 50% of the overall variance. In such cases it does not seem reasonable to completely neglect this direction of variance in the flow balances after choosing only two scenarios with respect to it. Our second strategy overcomes this weakness by considering combinations of principal components and thus, extracting more relevant scenarios.

Convex Hull Strategy ($\mathbf{CH}^b$). Instead of considering the chosen principal components individually, our second strategy forms groups of $b \in \mathbb{N}$ consecutive principal components, which are considered simultaneously. For each such group, it first computes the projection of the given flow balances on the b-dimensional subspace spanned by the b principal components of the group and then determines the vertices of the convex hull of the projected flow balances. The scenarios corresponding to these vertices then are chosen for S'.

More precisely, we are given the parameters $b \in \mathbb{N}$ and $k' \in \mathbb{N}$ for the size and for the number of the principal component groups. Group i, $i \in \{1, \dots, k'\}$, then consist of the principal components $p_{(i-1)\cdot b+1}$ to $p_{i \cdot b}$ (in order of non-increasing eigenvalues). In total, the $k = k' \cdot b$ principal components with the largest eigenvalues are considered. For each group $i \in \{1, \dots, k'\}$, we first compute the projection of all flow balance vectors into the subspace spanned by its principal

components. In the coordinates given by these principal components, the resulting set is

$$P_i = \left\{ \hat{d}_i^s = (\bar{d}^s_{(i-1)\cdot b+1}, \ \dots \ , \bar{d}^s_{i\cdot b}) \mid s \in S \right\} \subset \mathbb{R}^b,$$

where $\bar{d}^s_j = D_{s\bullet} \cdot p_j$. Then, we compute the convex hull $\operatorname{conv}(P_i)$ of this set and identify the set S_i of those scenarios s, whose projected flow balances $\hat{d}_i^s$ are vertices of $\operatorname{conv}(P_i)$. In total, we pick for S' all scenarios that correspond to some vertex in any of the projections, i.e.,

$$S' = \bigcup_{i=1,\dots,k'} S_i.$$

We denote this strategy by CH^b, depending on the dimension b.

Note, that the Min-Max strategy described above is just the special case of the convex hull strategy for dimension $b = 1$. The computational effort of the convex hull strategy strongly depends on the dimension b of the space where the convex hull and its vertices are computed. For dimensions up to 8 and flow balance scenarios stemming from real-world network design problems, this can be done efficiently with current desktop computers. For larger dimensions, these computation may be too time consuming.

Since our goal is to find a rather small number of critical scenarios, we try to further reduce the set of selected scenarios.

Symmetric Convex Hull Strategy (SCH^b). In the RCND problem, edge capacities are undirected. The capacity of any edge must exceed the sum of the flows carried in both directions of the edge. Edge capacities admit a feasible flow f with flow balances d^s if and only if they admit the flow $-f$ with flow balances vector $-d^s$. Consequently, the capacities obtained by solving RCND for the scenario subset S' admit feasible flows not only for any flow balance that is a convex combination of the original flow balances d^s with $s \in S'$. Due to the symmetry, they also admit feasible flows for any flow balance that is a convex combination of all original and all negated flow balances d^s and $-d^s$ with $s \in S'$. We can exploit this symmetry to further reduce the scenario set S' as follows.

In the first phase of our approach, we apply the PCA method to the matrix D formed by the original (shifted) flow balances to determine the principal components p_i and their eigenvalues λ_i as described above. Before entering the second phase, we replace matrix D by the matrix

$$D_2 = \begin{pmatrix} D \\ -D \end{pmatrix}$$

obtained by appending the rows of $-D$ to D. This corresponds to adding the flow balances $-d^s$ for all $s \in S$ explicitly to the set of considered flow balances.

In the second phase, we then use the principal components and eigenvalues determined in the first phase, but matrix D_2 instead of D to pick scenarios for S' with the convex hull strategy. Note the that, by using D_2 instead of D, we obtain the projections $P_{2,i} = P_i \cup (-P_i)$, which contain two points per scenario, one corresponding to the original and one to the negated flow balance. A single scenario thus can correspond to up to two different vertices of $\operatorname{conv}(P_{2,i})$. As above, the strategy then picks for S' any scenario that corresponds to any vertex of $\operatorname{conv}(P_{2,i})$. Of course, scenarios that correspond to two vertices are added to the set S' only once.

One easily verifies that a scenario that corresponds to a vertex of $\operatorname{conv}(P_{2,i})$ obtained from D_2 also corresponds to a vertex of $\operatorname{conv}(P_i)$. Thus, the set of scenarios chosen with the symmetric convex hull strategy is always a subset of the set of scenarios that would have been chosen with the corresponding convex hull strategy without exploiting the symmetry. In practice, the set obtained with the symmetric convex hull strategy often is substantially smaller. This is illustrated in Figure 2 for a 2-dimensional data set, where CH identifies 6 scenarios that are reduced by SCH to just 3 scenarios.

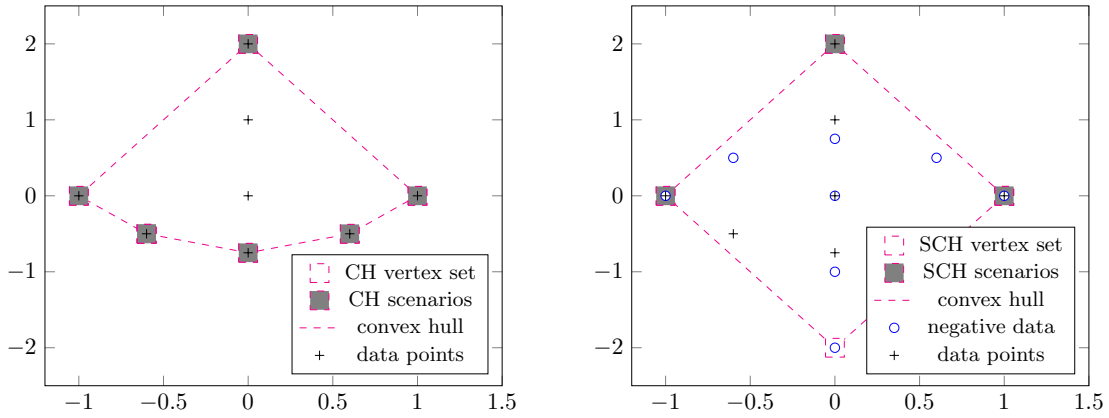


Figure 2: Symmetric convex hull method. The first figure shows the projections of the flow balances of 8 scenarios, whose convex hull is defined by 6 scenarios. These are chosen by the convex hull strategy. The second figure shows the convex hull of the 16 original and negated flow balances of the same 8 scenarios, which has only 4 vertices that correspond to only 3 different scenarios. These are chosen by the symmetric convex hull strategy.

Note that the symmetry argument is only valid in the original n-dimensional space of the flow balances, but the computation of the vertices of the symmetrized convex hull is done in lower dimensional spaces spanned by only a few principal components. Thus, flow balances, whose projections are not vertices of the convex hull in these low dimensional spaces, may still be vertices in the original space. Hence, removing the corresponding scenarios can lead to a loss of accuracy. Consequently, the symmetric convex hull method does not necessarily improve upon the normal convex hull method in terms of the quality of the solution. Both methods are meaningful.

Also note that our scenario selection strategies do not take the underlying network structure into account. The set of chosen scenarios depends only on the number of nodes and the flow balance vectors of the given scenarios. Hence, the selection procedure has to be executed only once if different network topologies on the same node set are studied for the same set of load scenarios.

4 Computational Experiments

In order to assess the effectiveness of the proposed scenario selection strategies, we compare the solution times and the objective values obtained with the natural ILP formulation (RCND) for the scenario subsets chosen by the different strategies to those obtained for the original, full scenario set for various instances.

4.1 Instances

In our experiments, we consider two different types of instances. The first group of instances stem from offshore power grid planning problems and originate from the research project *North Sea Offshore Network* (NSON) [3, 4]. The original power grid planning problems contain 3 different types of nodes: supply nodes (with negative flow balance), transmission nodes (with flow balance zero), and trading nodes (with arbitrary flow balance). Also, the total demand of all supply and trading nodes is not necessarily equal to zero in these problems, as the supply nodes' given flow balances represent the maximum potential power generation at these nodes. In the corresponding RCND instance, we thus introduced an artificial sink node and edges from each supply node to the sink, that provide sufficiently large capacities at zero cost to model the curtailment of the power production. For each scenario, the sink node's flow balance is set to the negative sum of all other nodes balances. In order to avoid that the sink node is also used for power transport among the

Instance	*NSON-386*	*NSON-35*
$\|V\|$	386	35
$\|E\|$	1143	449
$\|S\|$	8419	8760
capacity units	250, 400	250, 400
demands range at suppliers	$-1007 \dots 0$	$-1760 \dots 2394$
demands range at consumers	$-6365 \dots 18126$	$-3385 \dots 6485$
units installed per edge in opt	$0 \dots 139$	$0 \dots 3$

Table 1: Summary of NSON instances

other nodes, we let A_0 denote the set of all arcs emanating from the sink and forbid flow on the arcs of A_0 by introducing the additional constraints

$$f_{ij}^s \leq 0 \quad (i,j) \in A_0, s \in S.$$

Note that his leads to a generalization of the RCND problem, which slightly changes the character of the problem. Due to the upper bound (of zero) for the flows on the arcs of A_0, the problem is no longer symmetric (unless A_0 is symmetric): A capacity installation that is feasible for a flow balance vector d is not necessarily also valid for the negative flow balance $-d$.

We consider the two instances *NSON-386* and *NSON-35*, which are summarized in Table 1. The original power grid of *NSON-386* contains 385 nodes and 3876 edges. In order to obtain an instance that is solvable within reasonable time, we removed all but the 847 edges obtained by computing 3 times an inclusion-wise maximal forest of minimum cost in the network containing only those edges, that have not yet been chosen before. (As the original graph is not 3-edge-connected, the later forests are no spanning trees.) Together with the artificial sink and the edges emanating from the sink, the resulting graph of *NSON-386* then contains 386 nodes and 1143 edges. For *NSON-386*, we are given 8419 demand scenarios with flow balances between -1007 and 0 at the supply nodes and between -6365 and 18127 at the trading nodes. Depending on the edge type, capacity units of 250 or 400 can be installed. In an optimal solution, between 0 and 139 capacity units are actually installed at each edge.

Instance *NSON-35* originates from a smaller offshore power planning problem. Its graph contains 35 nodes including the sink and 430 normal edges plus 19 edges to the sink. For *NSON-35*, we are given 8760 demand scenarios with flow balances between -3385 and 6485. Capacity units of 250 or 400 can be installed at the edges. In an optimal solution, between 0 and 3 units are actually installed on each edge.

Our second group of test instances is based on benchmark problems for robust capacitated network design with multi-commodity network flows from the Survivable Network Design Library SNDlib [9, 10], which originate from telecommunication networks. For our study, we focused on the dynamic traffic instances *Abilene*, *Geant* and *Brain*, which contain a large number of demand scenarios.

In order to construct single-commodity instances, we accumulated the multi-commodity demands of the original multi-commodity instances to a single commodity flow balance vector per scenario as follows. For each scenario s, let $\left(D_{ij}^s\right)_{ij}$ denote the matrix containing the original demand values for each source sink pair. We start with flow balances of zero. For each pair of nodes $i \neq j$, the sum of the two demand values $D_{ij}^s + D_{ji}^s$ then is added to the flow balance of node j and subtracted from the flow balance at i if $D_{ij}^s \geq D_{ji}^s$, and vice versa if $D_{ij}^s < D_{ji}^s$. Note that, using this aggregation, different commodities' demands can cancel out one another at some nodes. The absolute value of the resulting single-commodity flow balance at a node can be much smaller than the total emanating and the total terminating original point-to-point demand at that node. To avoid numerical problems, we have scaled and rounded the resulting flow balances so that their range remains reasonable with respect to the capacity units. In our experiments, we have also tried other aggregation procedures, which, in the end, have lead to quite similar results.

Instance	*Abilene*	*Brain*	$Brain_{agg}$	*Geant*
$\|V\|$	12	161	9	22
$\|E\|$	15	166	14	36
$\|S\|$	48096	8991	8991	10761
capacity units	40	10^7	10^7	5000
demands range	-7385 $\ldots 8196$	$-9.5 \cdot 10^7$ $\ldots 1.05 \cdot 10^8$	$-9.7 \cdot 10^7$ $\ldots 1.12 \cdot 10^8$	-30612 $\ldots 26350$
units installed per edge in opt	$0 \ldots 197$	$0 \ldots 11$	$0 \ldots 7$	$0 \ldots 5$

Table 2: Summary of SNDlib instances

Note that the graph of the *Brain* instance has a very special structure. It contains 9 core nodes, which form a dense subgraph. All other nodes are leafs (i.e., of degree one) that are connected to exactly one of the central nodes. The capacity required for each edge to such a leaf node thus is determined by the scenario where the absolute value of the flow balance of the corresponding leaf node attains its maximum. For each edge, the corresponding critical scenario can be easily determined a priori in linear time. If, however, for one of these edges the critical scenario is not in the scenario subset considered when solving the RCND problem, the resulting solution will very likely not install sufficient capacity on this edge. Therefore, we also consider an aggregated version of the *Brain* instance, where only the central nodes and the edges among them are considered and all leaf node demands are aggregated with their corresponding central node's demand. We call this instance $Brain_{agg}$.

A summary of the SNDlib based RCND instances is provided in Table 2.

4.2 Implementation

To assess the effectiveness of the proposed scenario selection, we implemented the strategies and the natural mixed-integer linear programming formulation (RCND) in Python, version 2.7. For the computation of the principle components, we used the StandardScaler and the PCA algorithm available in the sci-kit libraries sklearn.decomposition and sklearn.preprocessing, respectively [11]. For determining the vertices of the convex hull, we used the corresponding algorithms from the scipy.spatial library [14]. Mixed-integer linear programs were solved using CPLEX 12.9 [2], a commercial integer programming solver, via its Python API. Note that for many of the considered test instances the natural formulation (RCND) cannot be solved to optimality (within the given time limits) when using CPLEX or GUROBI with default settings. However, formulation (RCND) naturally decomposes in one flow-subproblem per scenario and the master problem of finding the integer capacity installation. With its automatic Benders decomposition enabled (parameters.benders.strategy=3), CPLEX is able to detect this decomposition and solve all instances using a Benders-type algorithm. This allows us to find solutions even for the full RCND instances involving the complete scenario set and, thus, to benchmark the solution quality and speedup obtained for the scenario subsets chosen by the proposed strategies. For this purpose, we leave all other parameters at the default settings.

The computational experiments were conducted on a computing server featuring two 14-core Intel Xeon (R) E5-2690 v4, 2.6 GHz processors, 256 GB of RAM, and running Linux x86-64. Unless stated otherwise, all PCA and convex hull computations could be done in (far) less than 1 second. Hence, we report only the MILP solution times and mention the times needed for PCA and convex hull computations only if they make a substantial difference.

4.3 Results for the NSON Instances

We first present the results obtained for the NSON instances stemming from power grid planning, as this application motivated our work.

Nr. PC	**explained var.**	**cumulated var.**
1	$68.8 \cdot 10^{-2}$	$68.8 \cdot 10^{-2}$
2	$16.5 \cdot 10^{-2}$	$85.3 \cdot 10^{-2}$
3	$8.7 \cdot 10^{-2}$	$94.1 \cdot 10^{-2}$
4	$4.8 \cdot 10^{-2}$	$99.0 \cdot 10^{-2}$
5	$0.4 \cdot 10^{-2}$	$99.4 \cdot 10^{-2}$
...	...	...
20	$1.2 \cdot 10^{-12}$	1
21	$1.0 \cdot 10^{-12}$	1
22	$9.7 \cdot 10^{-13}$	1
23	$4.4 \cdot 10^{-33}$	1
...	...	...
35	$4.0 \cdot 10^{-33}$	1

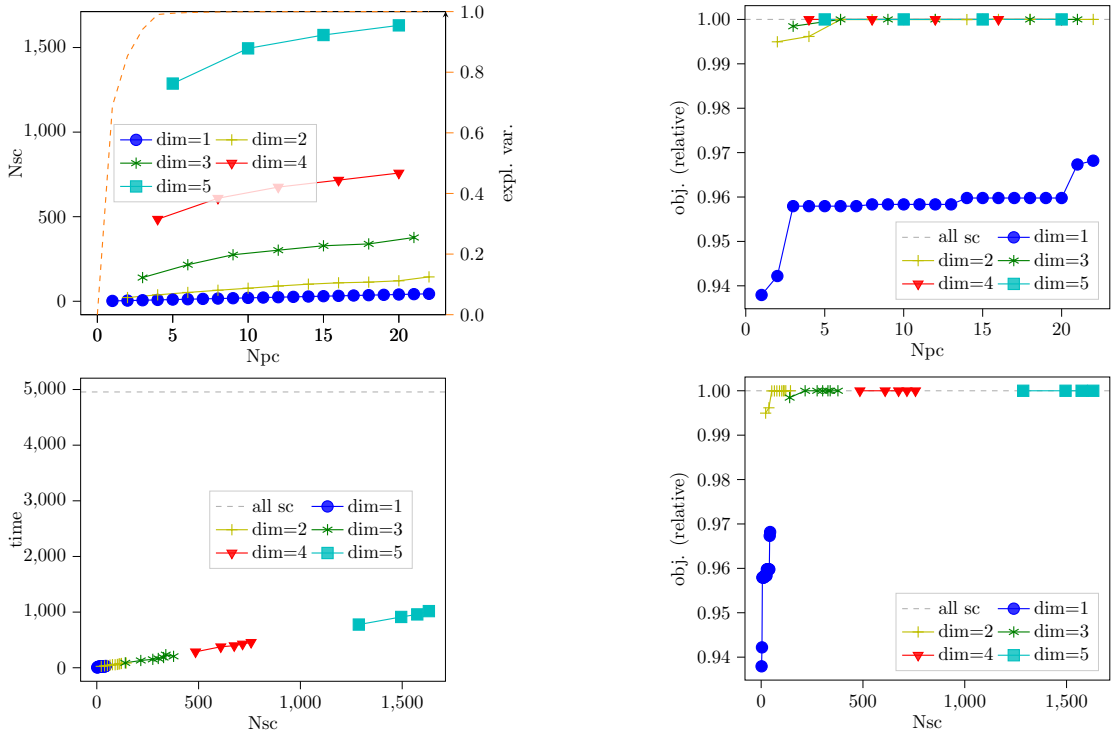
Table 3: PCA results for instance *NSON-35*

Before addressing the effectiveness of the model reductions obtained with the proposed scenario selection strategies, let us briefly discuss the results of the first phase of our selection strategies, where we identify the principal components in the set of flow balances of the given scenarios using the PCA method as described in Sections 2 and 3. We discuss these results only for the smaller instance *NSON-35*, as the results for the other instances are very similar.

Instance *NSON-35* contains a total of 8760 scenarios, whose flow balance vectors describe the supplies and demands at the 35 nodes of the network. As the demand data has dimension 35, we can have up to 35 principal components. However, 11 of the 35 nodes are transmission nodes, which have a flow balance value of zero in each scenario, and one node is the artificial sink, whose flow balance is equal to the negative sum of all other nodes' balances. Thus, the dimension of the space actually spanned by the demand data is at most 23, i.e., the 8760 given flow balances vary in at most 23 independent directions. Table 3 shows how much of this variation is explained by the first principal components identified by the PCA method. For the i-the principal component (in order of decreasing eigenvalue), column *explained variance* shows which percentage of the overall variance is retained in the projection of the flow balances to the space spanned by the i-th principal component p_i alone, while column *cumulated variance* shows the percentage of the variance retained in the projection to the space spanned by the principal components 1 through i. We see that already very few independent directions suffice to describe the actual variation in the given flow balances: Roughly 69 percent of the overall variance is explained by the first principal component alone, the first 4 principal components already capture more than 99 percent of the overall variation, and the variation associated with later principal components is negligible. This situation is typical for demand data stemming from real-world networks. Furthermore, we see from Table 3 that the given demand data actually has dimension only 22, not 23.

Figure 3 then illustrates the results obtained with the scenario selection strategy CH^b for this problem instance. First, we look at the number of scenarios that are identified by this strategy. The top left subfigure shows the number of scenarios (Nsc) chosen for S' by this strategy for different subspace dimensions b and varying numbers of principal components (Npc) that have been considered. Recall that for dimension $b = 1$ strategy CH^1 is just the min-max strategy MM. The number of scenarios chosen by CH^1 is 2 times the number of principal components considered, ranging from 2 scenarios for 1 principal component to 44 for 22 principal components. However, a single scenario could potentially be extreme in more than one principal component and the resulting selected scenarios could be less than 2 times the number of principal components. For higher subspace dimensions b, more scenarios per principal component define vertices in the respective subspaces and thus are chosen. Consequently, we see that larger subspace dimensions lead to larger numbers of chosen scenarios for the same number of principal components considered. While CH^1 chooses only $0.02 - 0.6\%$ of the given scenarios, CH^2 chooses $0.25 - 1.7\%$, CH^3 chooses $1.6 - 4.4\%$, CH^4 chooses $5.5 - 8.7\%$, and CH^5 already chooses $14.6\% - 18.7\%$ of the given

Figure 3: Results for *NSON-35* with strategy CH^b.



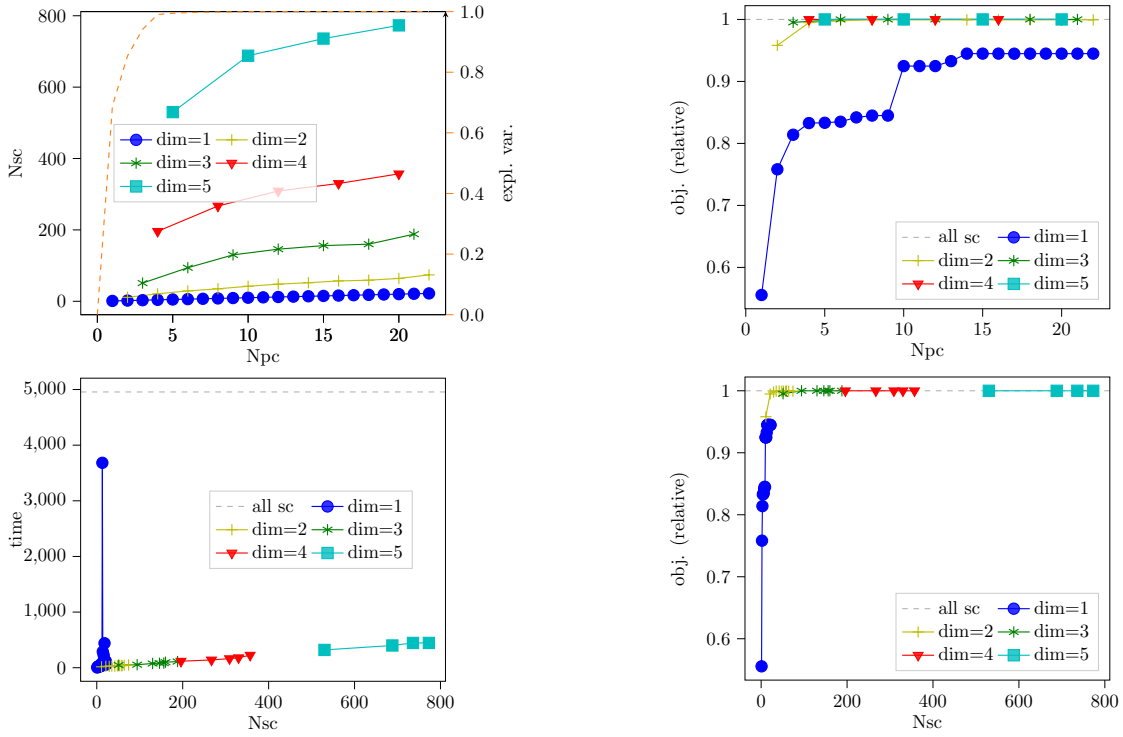
scenarios. The additional dashed line starting at the left bottom shows which percentage of the overall variance in the given flow balance data is cumulatively explained by the first i principal components.

Next, we investigate the objective value that is obtained with the chosen scenario subsets. The top right subfigure of Figure 3 shows the objective values obtained for the scenario subsets chosen by the CH^b strategy for different subspace dimensions b and varying numbers of principal components (Npc) considered. The dashed horizontal line indicates the objective value of the full problem involving all scenarios. The values obtained for the scenario subsets chosen for $b = 1$ (i.e., with min-max strategy) are strictly less than the full problem's value, even if many principal components are considered. Nevertheless, already this simple strategy yields very good results for this problem instance, achieving up to 96.8% of the full problem's optimal objective value when considering only 44 of the 8760 given scenarios. For $b = 2$ and $b = 3$, the scenario subsets chosen with strategy CH^b for 6 or more principal components yield the full problem's optimum, for $b \geq 4$ the optimum is already achieved with the first b principal components.

As mentioned above, the number of scenarios chosen per principal component by strategy CH^b increases as b increases. Furthermore, both the quality of the solution obtained for a scenario subset as well as the time needed to solve the corresponding model obviously depend on the size of the scenario subset. Therefore, we assess the efficiency of the scenario selection strategies primarily with respect to the number of scenarios that have been chosen, instead of the number of principal components that have been considered. The bottom right subfigure of Figure 3 shows the objective values obtained for different values of b depending on the number of scenarios (Nsc) chosen by strategy CH^b. We see that CH^2 performs better than the simpler min-max method CH^1 when the same number of scenarios are chosen. However, CH^2 also yields a better result than CH^3 for one case where the same number of scenarios was chosen, so a higher subspace dimension b not necessarily leads to a better selection of scenarios.

The bottom left subfigure of Figure 3 finally shows the times (in seconds) needed to solve the reduced models depending on the number of scenarios (Nsc) chosen by strategy CH^b for different

Figure 4: Results for *NSON-35* with strategy SCH^b.



values of b depending. The speedup with respect to the full original model contain all scenarios is significant for all parameter settings. While solving the full problem takes about 5000 seconds, the solution times for the reduced models range from 4-32 seconds for CH^1 and 25-85 seconds for CH^2 to 776-1018s for CH^5. For this instance, the solution times increase almost linearly with the number of scenarios in the model.

All of the following results figures are composed in the same way.

Figure 4 illustrates the results obtained with the scenario selection strategy SCH^b for the same problem instance *NSON-35*. Note that this problem is not fully symmetric, as the edges between the artificial sink and the supply nodes can carry flow in only one direction. Hence, the symmetric convex hull strategy might wrongfully discard scenarios, which are actually necessary to determine the capacities. In fact, SCH^b typically chooses roughly only half as many scenarios as CH^b did for the same subspace dimension b and the same number of principal components. Consequently, also the solution time for the reduced model reduces to approximately 50%, with only one exception for $b = 1$, which seems to be caused by numerical difficulties. Apparently, in this case the solver has trouble finding the (near) optimal solutions and generates a huge branch and bound tree using default settings. With a search strategy focusing on finding feasible solutions (parameter emphasis.mip 4) or a different branching strategy (parameter variableselect 2), however, it is able to solve the problem to optimality in roughly 120 seconds. Note that even though SCH^b drops almost half of the scenarios, the quality of the reduced model hardly deteriorates. For $b = 1$, where already the scenario set chosen by CH^1 was too small to achieve the full problem's optimum, the additional reduction of the scenario set worsens the solution quality. Moreover, for $b = 2$ we see a small drop in accuracy but only to 99.94% of the optimal objective. For all $b \geq 3$, the full problem's optimal objective is still achieved when 4 or more principal components are considered.

The corresponding results for the larger NSON instance *NSON-386* are shown in Figures 5 and 6. As the underlying graph contains 386 nodes, the given flow balances have dimension 386. However, the first 20 principal components identified by the PCA method already cover

Figure 5: Results for *NSON-386* with strategy CH^b

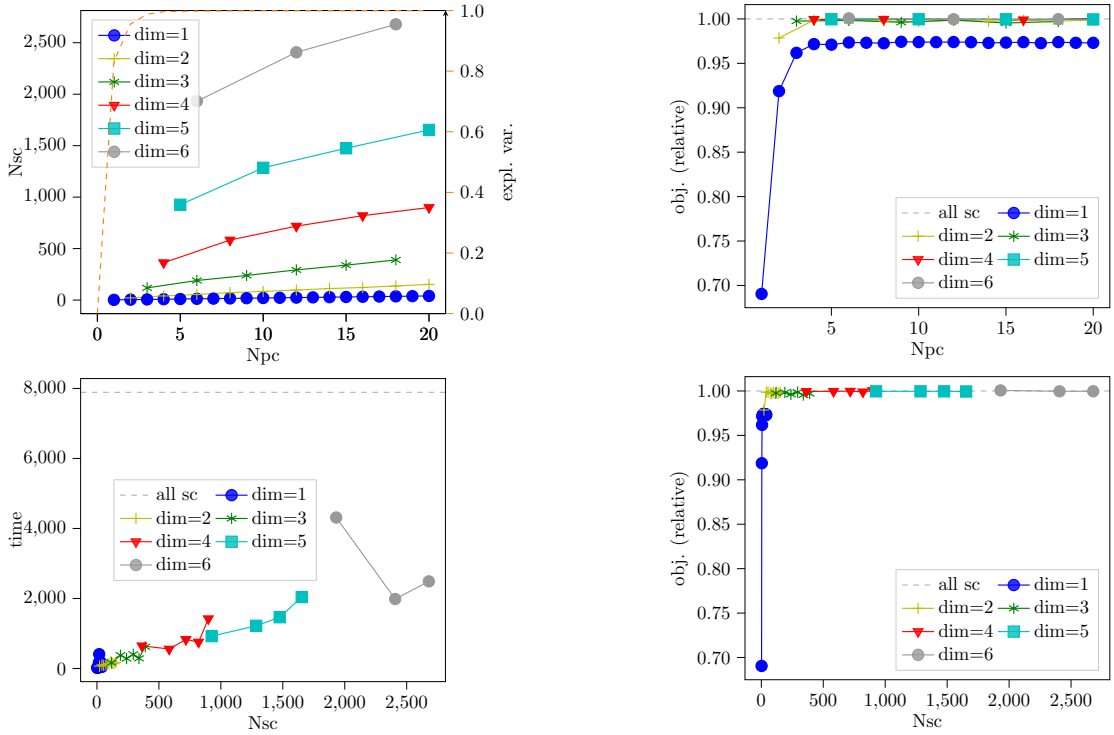
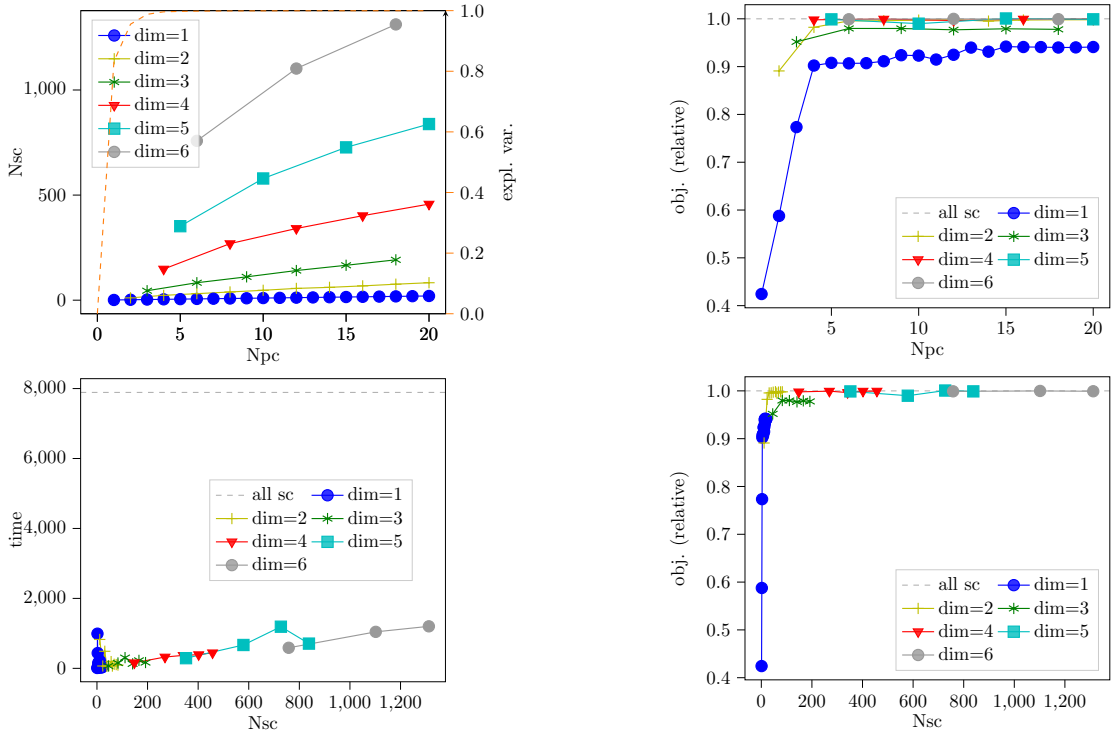


Figure 6: Results for *NSON-386* with strategy SCH^b



more than 99.97% of the variance in the given flow balances. Since the *NSON-386* instance is anyway computationally challenging, only these 20 most important principal components have been considered in our scenario selection strategies. The results show that this suffices to identify a scenario subset that leads to a very good approximation of the full problem. Note that, for this particular large instance, we solve the integer linear models only to an optimality gap of 1% in order to keep the computational times small. As a consequence, the reported objective values depend on the solutions and the best bounds found during the solution processes and are accurate only within this bound. For the strategy CH^b the number of identified scenarios behaves similar as for the smaller instance *NSON-35*. Also, the model accuracy achieved with the chosen scenario subsets is comparable to that of the 35 nodes instance. For the non-symmetric strategy CH^b, we observe a gap of 2.6% for $b = 1$, which cannot be closed by considering more principal components. Higher values of b lead to better results. For $b = 2$ with 4 or more principal components, the scenario set chosen by CH^b already achieves 99.86% of the full model's optimal objective value. With $b = 5$ applied to 10 or more principal components, we even obtained this optimal value. However, note that CH^5 applied to 10 principal components choses 1285 scenarios out of 8419 scenarios in the full scenario set, which is around 15%. The solution time of the reduced mixed-integer linear model, again, increases more or less linearly with the number of chosen scenarios. However, there are some cases where models with more scenarios are solved faster than models with less scenarios, which is somewhat unexpected. This effect, however, was only observed for the mixed-integer models, not for the corresponding linear relaxations, whose solution times indeed increased approximately linear with the number of scenarios. Therefore, we assume that this effect again is caused by some (unfortunate) branching decisions in the branch and bound procedure.

For the symmetric strategy SCH^b, we again see reductions in the number of selected scenarios and in the solution times of around 50%. Concerning the objective values achieved by the reduced models, we can make an interesting observation. While for most subspace dimensions b we do not loose much accuracy, for $b = 3$ the results get substantially worse, even when considering 18 principal components. Apparently, some of the critical scenarios can only be identified when the 3rd and the 4th principal components are considered together in one group, as the corresponding flow balances are vertices only when projected in a subspace spanned by both of these directions. For $b = 3$, however, components 3 and 4 will not be in the same group. Without the additional symmetry reduction in the SCH^b strategy, enough other scenarios that are chosen seem to compensate for the missing ones, with the symmetry reduction this does not seem to be the case anymore.

4.4 Results for the SND Instances

Next, we present the results obtained for the SND instances stemming from network design problems in telecommunications.

Figures 7 and 8 show the results obtained for the *Geant* instance for the scenario selection strategies CH^b and SCH^b, respectively. Here, we have considered all 21 principal components identified by the PCA method. The number of identified scenarios shows a similar growth with respect to the number of principal components considered and the subspace dimension b as for the *NSON*-instances. For $b = 1$, the min-max strategy CH^1 selects only 2-38 scenarios. Nevertheless, these scenario sets already yield $87-96\%$ of the optimal objective value of the full model containing all 10761 scenarios. Increasing the dimension to $b = 2$ improves the objective value obtained with the chosen scenario subsets to 99.5% of the full model's value. For $b > 2$ this value even is achieved when enough principal components are considered. Note that CH^5 achieves the full model's optimum already when applied to only the first group containing the 5 most import principal components. Thus, the scenarios corresponding to the vertices of the flow balances' projections into this 5 dimensional subspace already suffice to obtain the optimal solution of this RCND instance. Furthermore, for all dimensions b, we see that considering more principal components after some point does not help to improve the objective further. With respect to the solution time, we see a tremendous speedup with all strategies. While the original instance needs 77 seconds to be solved, even the rather large scenario subset (of 668 scenarios) chosen for $b = 5$

Figure 7: Results for *Geant* with strategy CH^b.

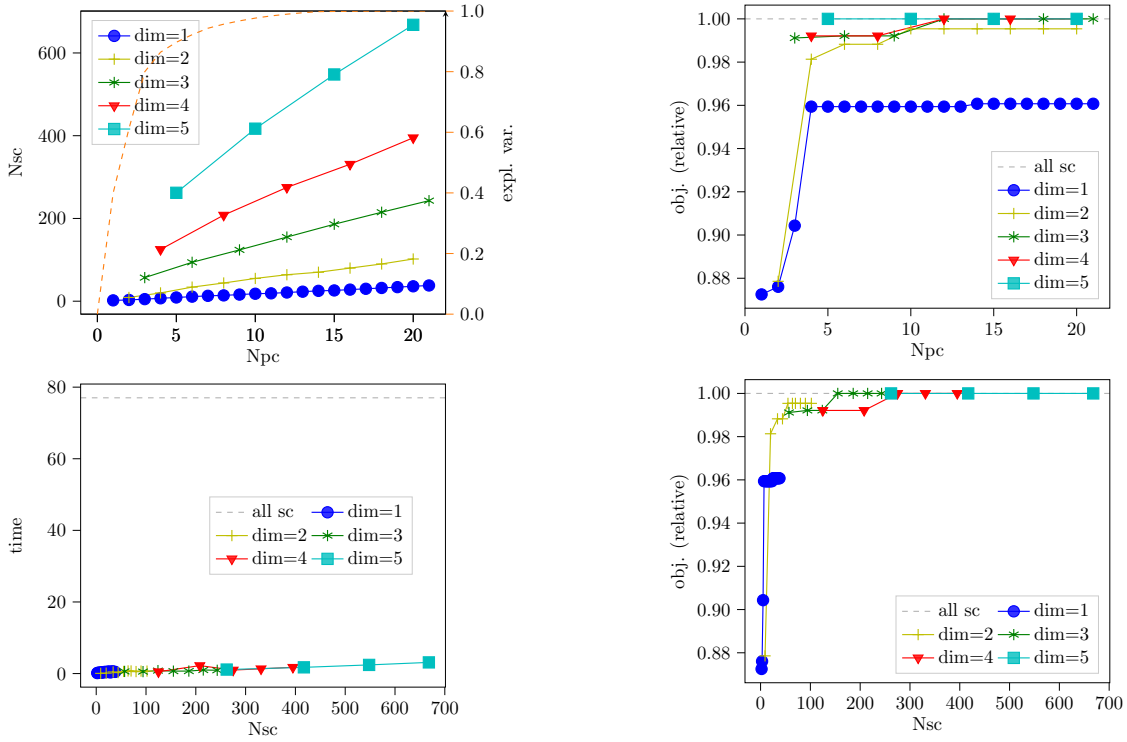
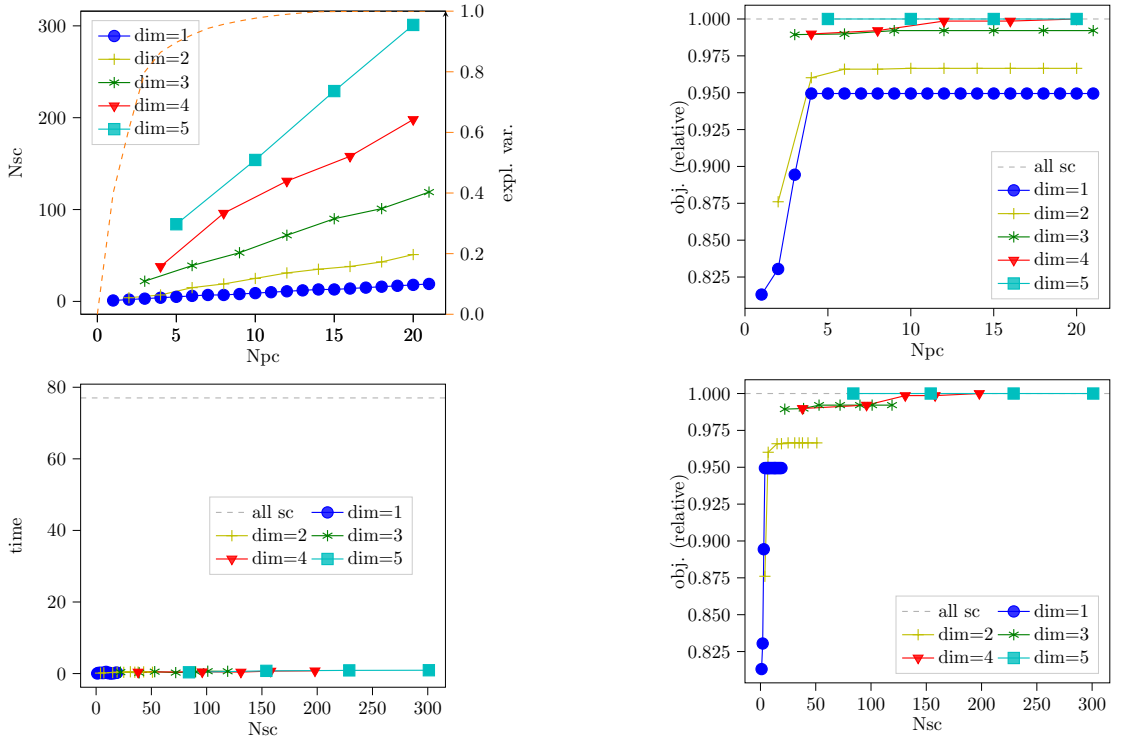


Figure 8: Results for *Geant* with strategy SCH^b.



and 20 principal components led to a reduction to only 3.1 seconds.

The SCH results are as expected. We see only small drops in accuracy, e.g. from $0.87 - 0.96$ for CH^1 to $0.81 - 0.95$ for SCH^1 and from $0.992 - 1$ for CH^4 to $0.990 - 1$ for SCH^4. For $b = 5$, both methods SCH^5 and CH^5 deliver the full model's optimal value already when applied to the first 5 principal components. However, SCH^5 chooses only one third of the number of the scenarios (namely 84 scenarios) chosen by CH^5. Among all methods that yield the full model's value, SCH^5 generated the smallest scenario subset. CH^3 applied to 12 principal components picked 155 scenarios, CH^4 with 12 principal components picked 275 scenarios, and SCH^4 applied to 20 principal components selected 198 scenarios. We see from these results that the symmetric method SCH^b typically allows us to consider more principal components, as it chooses substantially fewer scenarios per group of principal components without sacrificing to much of the accuracy obtained with the resulting reduced model.

Figures 9 and 10 show the corresponding results obtained for the *Abilene* instance. Note that this instance contains more than 48.000 scenarios, but only 12 nodes and 15 edges. Solving the MILP thus essentially means solving the LP relaxation, determining an optimal integer solution is no substantial additional effort in this case. We see that CH^3 applied to the 6 most important principal components identifies 105 scenarios, such that the objective value obtained with the reduced RCND model for only these scenarios is already 99.98% of the value obtained with the full model involving all 48.000 scenarios. This a great reduction in model size and solution time. While solving the full model takes more than 560 seconds, less than 2 seconds are required for the selection of the critical scenarios (including the time for the PCA method and calculating the vertices of the convex hulls) and the solution of the reduced ILP model. Unfortunately, increasing the number of principal components considered or increasing the dimension b (up to the value of 5) used in the convex hull computation does not reveal all the missing critical scenarios. We can only improve to 99.99% of the full models objective value. For achieving the full models objective value, dimension 7 is needed identifying 1113 scenarios.

Using the symmetric method SCH^b can again help to further reduce the number of scenarios and, thus, the time needed to solve the resulting reduced models without deteriorating the objective value achieved too much. To mention only one example, for subspace dimension $b = 3$ and 6 principal components to be considered, the number of chosen scenarios drops from 105 to only 38, while the objective value only drops from 99.98% to 99.84% of the full model's value. On the other hand, to achieve at least the quality of 99.98% as with the scenario sets chosen with the CH^3 strategy, we had to consider the 7 most important principal components simultaneously (i.e., use SCH^7 for the first seven principal components), which resulted in about 400 scenarios to be chosen and an accuracy of 99.99%.

The results we obtained for the *Brain* instance are illustrated in Table 4 and Figures 11 and 12. Remember that the underlying network of the *Brain* instance has a very special structure with 9 nodes forming a densely connected core network and 152 leaf nodes, that are connected in a star like fashion to the core nodes. In order to find a scenario subset that leads to sufficient edge capacities for all given scenarios, we not only need to find those scenarios, that drive the edge capacities in the (rather small) core network. For each of the leaf node, we also need to find the scenario that maximizes the demand or supply at this node. It thus seems plausible that we need a larger fraction of the given scenarios to obtain a good approximation of the edge capacities than for the instances considered so far. (Unless we explicitly compute the capacities needed for the edges between the core and leaf nodes beforehand, which would allow us to then ignore the scenarios that imply these capacities.)

For the *Brain* instance, again, the effort of solving the corresponding RCND model mostly consists in solving the LP relaxation, the additional effort in creating an integer solution from the LP solution is negligible.

Despite the fact that we have 161 nodes in the network, we see from the PCA results in Table 4 that the flow balance vectors only vary in 131 independent directions, which is the number of

Figure 9: Results for *Abilene* with strategy CH^b.

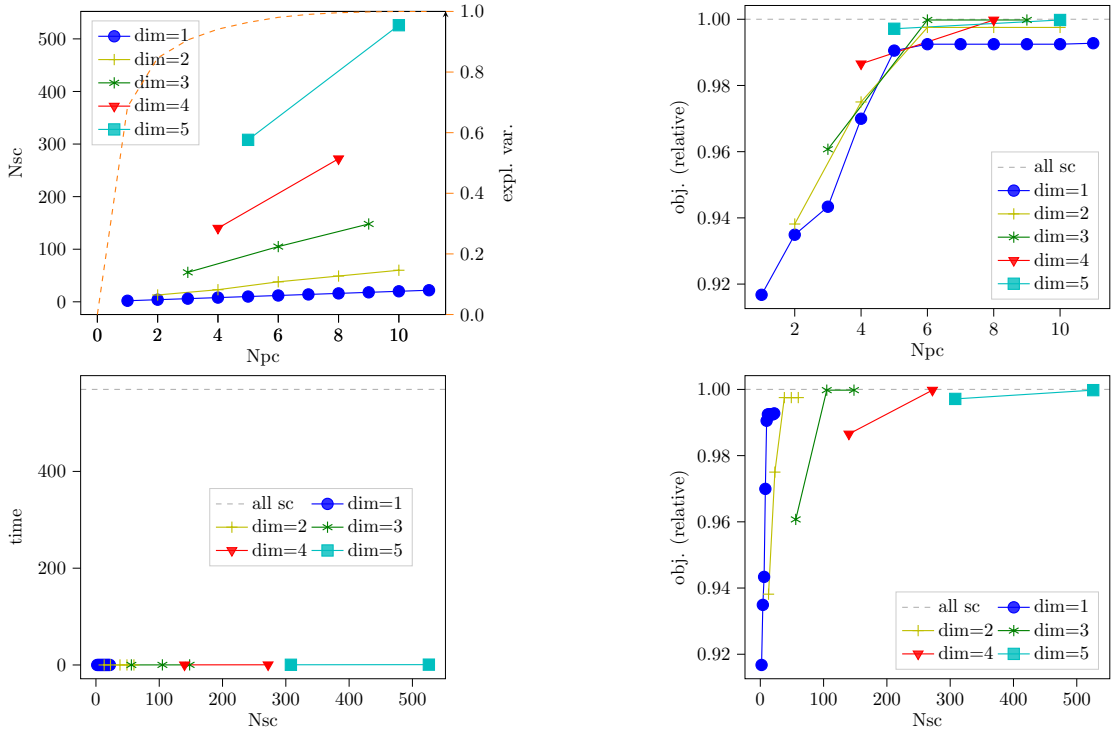


Figure 10: Results for *Abilene* with strategy SCH^b.

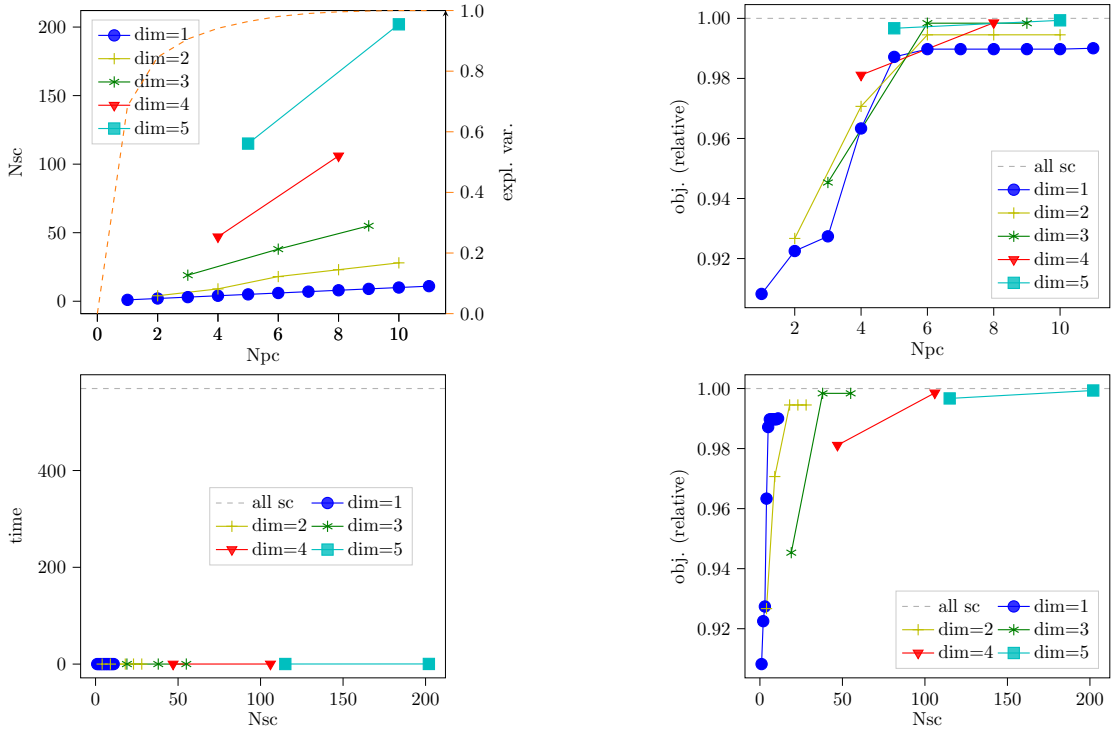


Figure 11: Results for *Brain* with strategy CH^b.

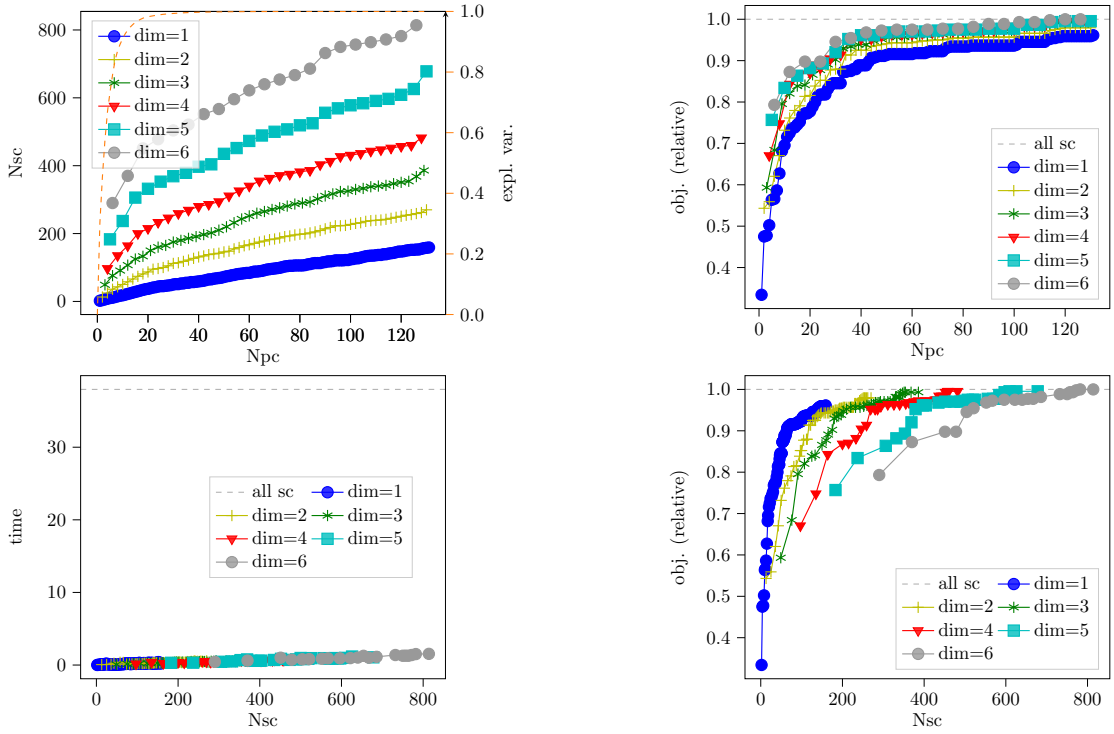
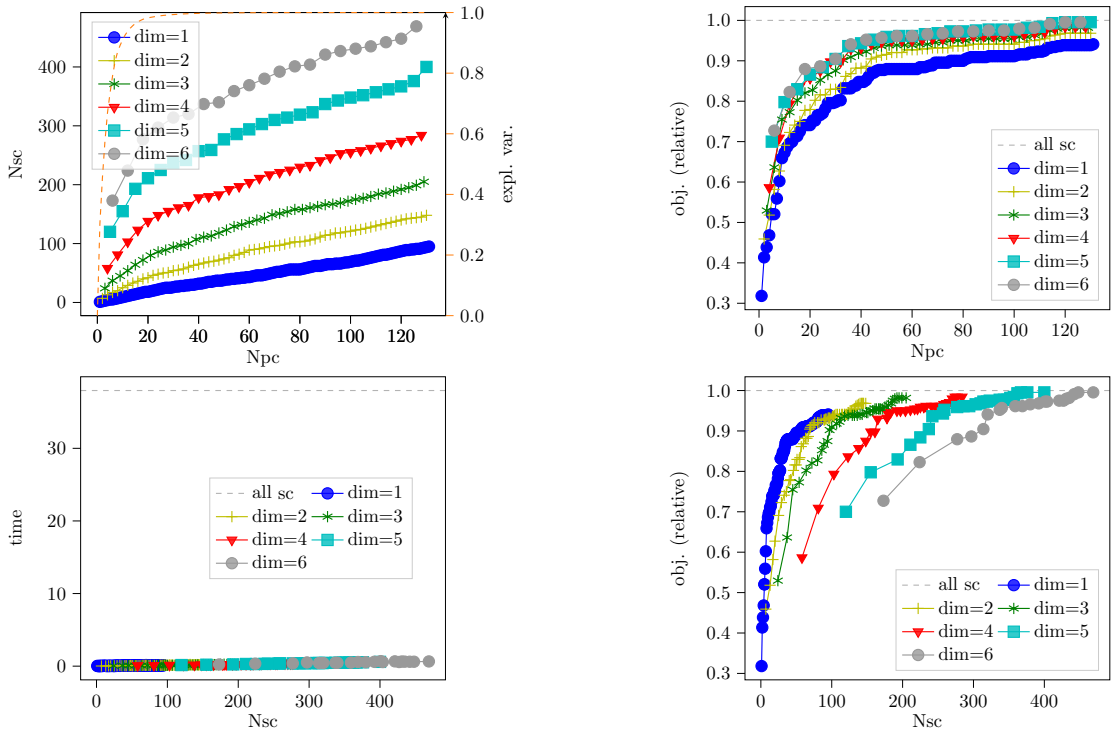


Figure 12: Results for *Brain* with strategy SCH^b.



Nr. PC	**explained var.**	**cumulated var.**
1	$30.22 \cdot 10^{-2}$	$30.2 \cdot 10^{-2}$
2	$15.23 \cdot 10^{-2}$	$45.4 \cdot 10^{-2}$
3	$12.07 \cdot 10^{-2}$	$57.5 \cdot 10^{-2}$
4	$9.36 \cdot 10^{-2}$	$66.9 \cdot 10^{-2}$
...	...	...
10	$1.84 \cdot 10^{-2}$	$91.0 \cdot 10^{-2}$
...	...	...
26	$0.10 \cdot 10^{-2}$	$99.0 \cdot 10^{-2}$
...	...	...
54	$8.30 \cdot 10^{-5}$	$99.9 \cdot 10^{-2}$
...	...	...
107	$1.60 \cdot 10^{-7}$	1
...	...	...
131	$1.42 \cdot 10^{-13}$	1
132	$1.75 \cdot 10^{-33}$	1
...	...	...

Table 4: PCA results for instance *Brain*

principal components identified. As the scenario subsets that are selected when only few principal components are considered are still rather small, many of the scenarios that are critical for edges between core and leaf nodes are still missing. Thus, the objective values obtained for few principal components are worse than for the other problem instances, starting with only $33 - 50\%$ of the value obtained for all scenarios for $b = 1$ and only a few principal components. Nonetheless, even for subspace dimension $b = 1$ we can achieve 96% of the optimal value when considering all 131 principal components. Applying the method with $b = 6$ and more than 120 principal components, we actually obtain the full models optimal value, considering only 782 out of the 8993 given scenarios (around 8.7%). For all subspace dimensions b, we need to consider a large number of principal components in order to achieve very good objective values. This resembles the result of the PCA (Table 4), where we see that, unlike for the instances considered up to this point, many different principal components are required to capture the overall variance in the given flow balances. Combined with the special structure of the network this explains why we need many scenarios and principal components to achieve the results that are close to the optimal objective value of the full model. For the SCH^b method, we see a similar behavior as for the other instances. However, there is an interesting observation. For 126 principal components, SCH^6 identifies 469 scenarios and achieves 99.55% of the full model objective value. On the one hand, this is roughly half a percent less than the value obtained with the scenario set identified by CH^6 for the same number of principal components. On the other hand, when we compare it to method CH^6 for 24 principal components, which picks a similar number of scenarios (478) and achieves an objective value of only 89.77% of the optimal objective, we see that it significantly improves upon the non-symmetric method by almost 10%. Thus, the symmetric variant SCH^b can be really helpful to keep the number of scenarios small when many principal have to be considered due to the distribution of the explained variance.

Finally, Figures 13 and 14 show the results we obtained for the problem instance $Brain_{agg}$, which was derived from *Brain* by aggregating all leaf nodes with their respective core nodes. As the sum of the flow balances over all 9 nodes of the network is zero in each scenario, the dimension of the space spanned by the (aggregated) flow balance vectors is at most 8. Therefore, the vertices identified by the convex hull method for subspace dimension $b = 8$ correspond exactly to the vertices in the original space of the flow balances. Strategy CH^8 actually identifies all scenarios that are necessary to describe the convex hull of the given flow balances and, thus, the corresponding reduced model is equivalent to the original model involving all given scenarios. Note

Figure 13: Results for $Brain_{agg}$ with strategy CH^b.

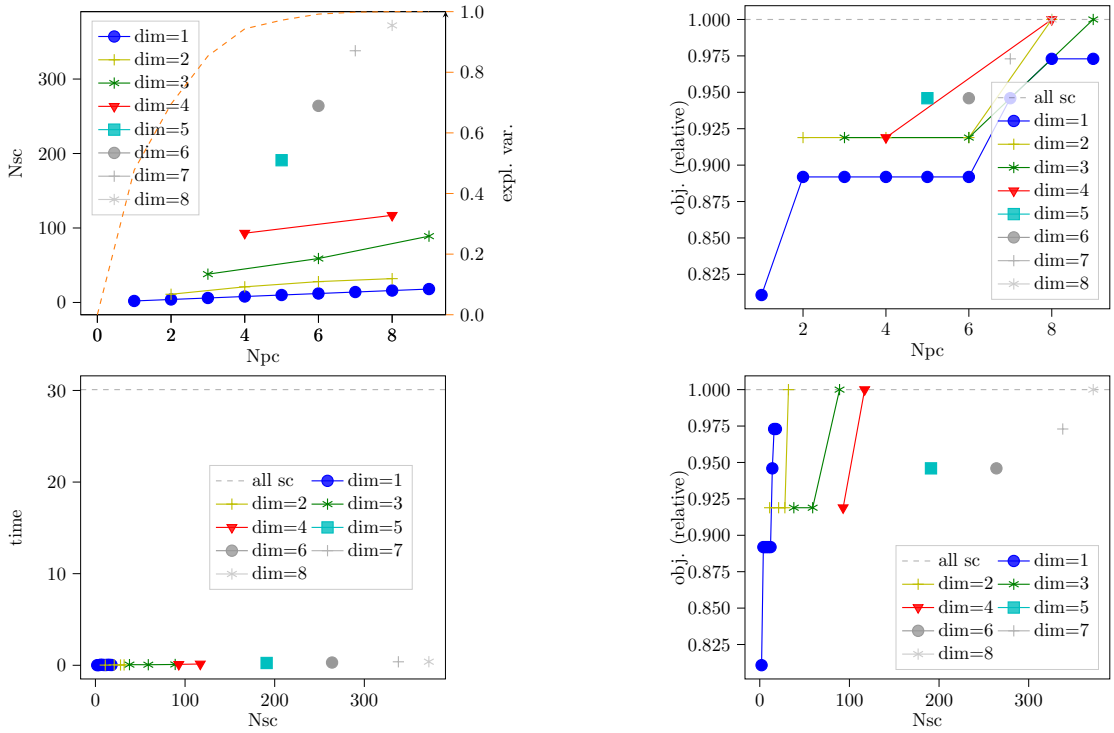
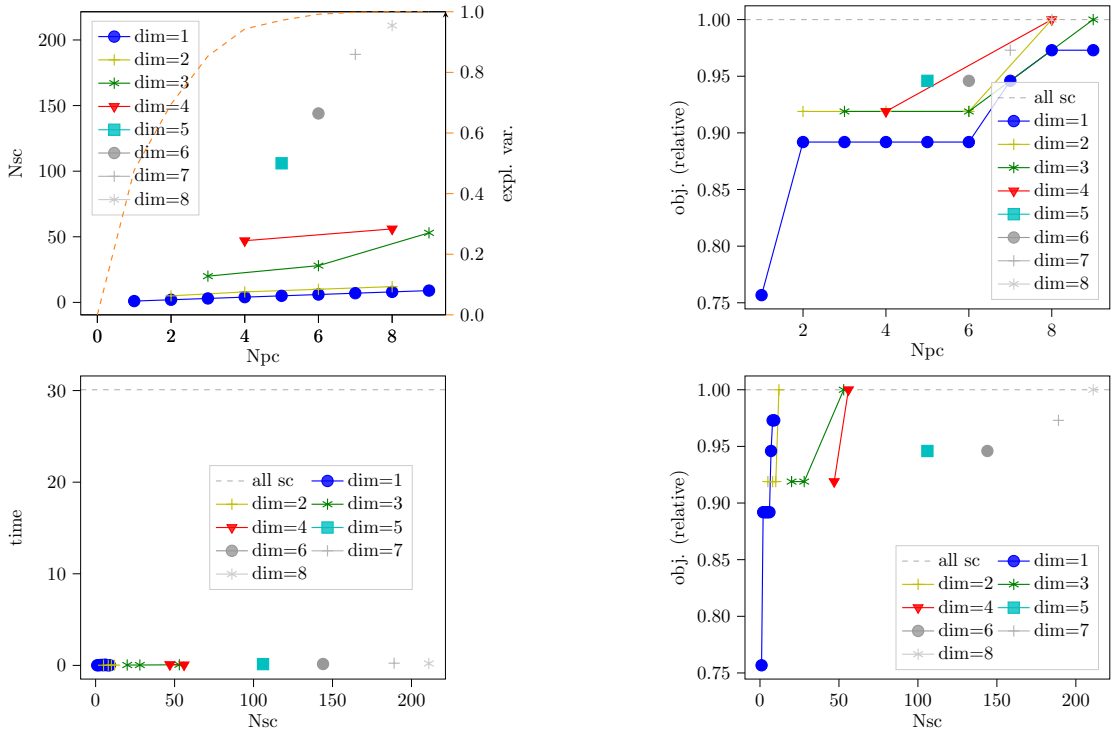


Figure 14: Results for $Brain_{agg}$ with strategy SCH^b.



that the number of scenarios selected with the different parameter settings remains relatively small. Even for subspace dimension $b = 8$ and considering all 8 principal components, CH^b identifies only 372 critical scenarios, which is small, e.g., compared to *Geant*'s 1387 scenarios for dimension 8. Using the symmetric method SCH^b, we can further reduce the scenario set, for $b = 8$ from 372 to only 211 chosen scenarios. Furthermore, we obtain the full problem's optimal value already with only 12 scenarios identified by CH^2 for 8 principal components. As shown in Figure 14, the only case where we lose accuracy with respect to the non-symmetric selection strategy is for subspace dimension $b = 1$ and a single principal component, where SCH^1 picks only one of the scenarios picked by CH^1.

Summarizing our results, we see that our methods deliver good scenario sets for all considered test instances, which stem from offshore power grid planning and telecommunication networks. The min-max method CH^1 delivers good results when a relatively large number of principal components was considered. Since at most 2 scenarios are selected per principal component, the selected scenario sets remain very small. On the other hand, one has to be very careful using the corresponding symmetric version SCH^1 for only a small number of principal components. SCH^1 drops around half of the scenarios selected by CH^1, which can have a big impact when there are not many scenarios at hand. For a larger number of principal components, the loss in accuracy is not that big and SCH^1 may be appropriate to choose a very small scenario set.

Using higher subspace dimensions b in the convex hull computations often leads to a significantly better selection of scenarios. In several cases, the scenario set generated by CH^b lead to the same objective value as the full scenario set, even for very small values of b and few principal components. Yet, the number of scenarios chosen by CH^b per per principal component can become very large as b increases. It is thus important to balance the subspace dimension b and the number of considered principal components in order to keep the chosen scenario subset small. Unfortunately, often it is not clear a priory if choosing a larger subspace dimension b and considering less principal components (e.g. *NSON-35*) leads to better results than choosing a smaller b and instead consider more principal components (e.g. *Abilene*). However, our results (e.g. for the *Brain* instance) indicate that it is important to consider more principal components if many principal components are necessary to explain the overall variance in the flow balance data.

The symmetric convex hull method SCH seems well suited to consider more relevant principal components without selecting too many potentially unnecessary scenarios per (group of) principal component(s). The scenario sets chosen by the symmetric version SCH^b typically contain only half as many scenarios as those chosen by CH^b for the same parameters, while the accuracy of corresponding reduced model hardly deteriorates. However, it can happen that relevant scenarios are dropped, as the convex hull computations are performed in low dimensional projections only.

For all methods and instances we observed a tremendous reduction of the times needed to solve the RCND model for the reduced scenario set in comparison to the full scenario sets. The times needed for the PCA and computing the convex hulls was typically less than 1s.

When the selected scenario set is required to be as small as possible, we recommend to use CH^1 since at most 2 scenarios per principal component are chosen. It may be even possible to use SCH^1 to further reduce the set, but one has to be careful when considering only few principal components. When many relevant principal components are required to explain the overall variance in the given flow balance data, then it is also recommended to consider more principal components and keep the dimension of the convex hull rather small instead. When the maximum accuracy is the goal, one should increase the subspace dimension b of the convex hull computation to 8-10 (depending on the numbers of scenarios and nodes). However, note that for dimensions 7 or larger the convex hull computation starts taking a non-negligable amount of time. Dimensions 2 or 3 seem to be a good trade off between the number of scenarios chosen per principal component considered and the gain in the objective value.

Also, note that due to small computational effort of our methods it is also possible to get several candidate sets of scenarios and decide which one to use depending on the sizes of the candidate sets.

5 Conclusion and Outlook

We have proposed a new technique combining principal component analysis and convex hull computations for identifying those scenarios in robust capacitated network design problems, that essentially determine the capacity installation. Aiming for the solution of real-world instances with very large scenarios sets, the goal is to find a very small subset of the given scenarios, such that solving RCND for the chosen subset of scenarios leads to almost the same capacities as solving it for the full given scenario set.

In our experiments, we have seen that already the min-max strategy, which chooses the scenarios that correspond to the minimum and the maximum of the flow balance vectors projected on each principal component independently, finds scenario sets that yield good solutions. Choosing the scenarios that correspond to the vertices of the convex hull of the flow balance vectors projected into b-dimensional spaces spanned by b principal components simultaneously, the convex hull method CH^b for $b \geq 2$ delivers not only larger, but typically also substantially better scenario sets. In many of the test instances, the scenario subsets generated using this approach yield the same capacity installation costs as the full scenario set, despite containing only a very small fraction of the given scenarios. Using a symmetric version SCH^b of this method, the number of chosen scenarios could be reduced further by a factor of roughly 2, often with only very little loss in the resulting model's accuracy.

In practice, robust optimization problems such as RCND are often solved using scenario-based decomposition approaches. These methods typically start with the solution of an initial master model, which involves none or only a few of the given scenarios, and then iteratively refine this model by considering more scenarios and implicitly or explicitly adding sub-models or valid constraints associated with these scenarios. The computational performance of these approaches often strongly depends on scenario set that is included in the initial model and on the order, in which the remaining scenarios are considered. The proposed scenario selection strategies CH^b and SCH^b can be easily integrated into these approaches and extended in such a way, that they identify both a scenario set to be used initially as well as an order in which to consider the remaining scenarios. Our results indicate that the methods are well suited for this purpose, as they construct (a sequence of) scenario subsets based on the data's variance that is retained in the chosen subsets, trying to simultaneously minimize the size of the scenarios set and to maximize the retained variance. Evaluating the impact of the proposed selection strategies on the computational efficiency of such scenario-based decomposition approaches is one line of potential future research.

As mentioned in Section 3, the proposed scenario selection strategies only consider the variation in the different scenarios' flow balances. They do not consider the topology, the capacity and cost data, or any other structural properties of the problem. Hence, they can be easily generalized to other problem types and settings. For future power grid design, for example, it is of great interest to integrate power storage technologies in the network planning. In order to appropriately model storage, it is necessary to consider the time dependencies among consecutive scenarios, i.e., to use time series instead of scenarios. In this context, it is of great interest to adapt the proposed techniques to identify within a given time series a subset of shorter time intervals that are critical (not typical or representative) for the dimensioning of the power grid's elements. We are currently working on these questions within the research project *NSON-II* [7], funded by the German Federal Ministry for Economic Affairs and Energy (BMWi).

Acknowledgments

This work was supported by the German *North Sea Offshore Network* (NSON-DE) project of the funding initiative *Zukunftsfähige Stromnetze* of the German Federal Ministry for Economic Affairs and Energy (BMWi). This support is greatly acknowledged.

References

[1] J. P. Bukenberger and M. D. Webster. "Approximate Latent Factor Algorithm for Scenario Selection and Weighting in Transmission Expansion Planning." In: *IEEE Transactions on Power Systems* 35.2 (Mar. 2020), pp. 1099–1108. ISSN: 1558-0679. DOI: 10.1109/TPWRS.2019.2942925.

[2] IBM ILOG CPLEX: IBM ILOG CPLEX 12.9. *User's Manual for CPLEX*. 2020.

[3] P. Härtel et al. "North Seas Offshore Network (NSON): Challenges and its way forward." In: *Journal of Physics: Conference Series* 1104 (Oct. 2018), p. 012004. DOI: 10.1088/1742-6596/1104/1/012004.

[4] P. Härtel et al. *North Seas Offshore Network (NSON): Machbarkeit und Implikationen verschiedener Offshore-Netzkonzepte in der Nordseeregion*. Tech. rep. available online at http://publica.fraunhofer.de/documents/N-515115.html. Fraunhofer IEE, University of Kassel, University of Hannover, June 2018.

[5] M. Hoffmann et al. "A Review on Time Series Aggregation Methods for Energy System Models." In: *Energies* 13 (Feb. 2020), p. 641. DOI: 10.3390/en13030641.

[6] A. Koster, M. Kutschka, and C. Raack. "Robust Network Design: Formulations, Valid Inequalities, and Computations." In: *Networks* 61 (Mar. 2013). DOI: 10.1002/net.21497.

[7] D. Mende et al. "NSON II: Next Steps in Economical Connection and International Integration of Offshore Wind Energy in the North Seas." In: *Proceedings of the 19th Wind Integration Workshop, Ljubljana, Slovenia*. 2020.

[8] G. Oriolo. "Domination between Traffic Matrices." In: *Mathematics of Operations Research* 33.1 (2008), pp. 91–96. ISSN: 0364765X, 15265471. URL: http://www.jstor.org/stable/25151842.

[9] S. Orlowski et al. "SNDlib 1.0–Survivable Network Design Library." English. In: *Proceedings of the 3rd International Network Optimization Conference (INOC 2007), Spa, Belgium*. http://sndlib.zib.de, extended version accepted in Networks, 2009. Apr. 2007. URL: http://www.zib.de/orlowski/Paper/OrlowskiPioroTomaszewskiWessaely2007-SNDlib-INOC.pdf.gz.

[10] S. Orlowski et al. "SNDlib 1.0–Survivable Network Design Library." English. In: *Networks* 55.3 (2010), pp. 276–286. DOI: 10.1002/net.20371. URL: http://www3.interscience.wiley.com/journal/122653325/abstract.

[11] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[12] A. Saxena et al. "A review of clustering techniques and developments." In: *Neurocomputing* 267.May 2018 (2017), pp. 664–681. ISSN: 18728286. DOI: 10.1016/j.neucom.2017.06.053. URL: http://dx.doi.org/10.1016/j.neucom.2017.06.053.

[13] D. R. Schmidt. "Robust Design of Single-Commodity Networks." PhD thesis. Universität zu Köln, 2014. URL: https://kups.ub.uni-koeln.de/5872/.

[14] P. Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python." In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[15] H. Wang, M. J. Bah, and M. Hammad. "Progress in Outlier Detection Techniques: A Survey." In: *IEEE Access* 7 (2019), pp. 107964–108000. DOI: 10.1109/ACCESS.2019.2932769.

[16] A. Zimek, E. Schubert, and H.-P. Kriegel. "A survey on unsupervised outlier detection in high-dimensional numerical data." In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 5.5 (2012), pp. 363–387. DOI: https://doi.org/10.1002/sam.11161. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sam.11161. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sam.11161.