Scenario generation by selection from historical data

Michal Kaut*

January 29, 2021

In this paper, we present and compare several methods for generating scenarios for stochastic-programming models by direct selection from historical data. The methods range from standard sampling and k-means, through iterative sampling-based selection methods, to a new moment-based optimization approach. We compare the models on a simple portfolio-optimization model and show how to use them in a situation when we are selecting whole *sequences* from the data, instead of single data points.

1 Introduction

In stochastic programming, we typically need to discretize the distribution of uncertain parameters. In a two-stage setting, this means generating a set of *scenarios* that approximates an assumed 'true' distribution. In this paper, we consider the situation where the 'true' distribution is given by a set of historical data and the scenarios must be its subset. In other words, we want to form the scenarios by selecting points from the data set in such a way that the selection, with associated probabilities, is a good approximation of the empirical distribution of the whole data set.

The main motivation for this study are stochastic versions of the TIMES (Loulou and Lettila, 2016; Loulou et al., 2016) and EMPIRE (Skar et al., 2016) energy-system models. There, we may have many years of hourly data for measurements of different types (such as wind-turbine and PV production) and from different locations/regions. The optimization model then requires a number of scenarios, each including values for one day, i.e., a sequence of 24 values for each measurement. While it is theoretically possible to model these sequences using some multi-variate stochastic process, it might not be easy to find a process that can correctly capture all spacial and temporal dependencies between the measurements. Using actual observations, on the other hand, guarantees that values within each scenario are consistent and realistic, since they actually happened (Seljom and Tomasgard, 2015). The same reasoning can be applied also when each

*SINTEF, Trondheim, Norway; michal.kaut@sintef.no. ORCID 0000-0002-7251-5236.

observation is just a single value (not a sequence), in cases with complicated non-linear dependencies between the different parameters.

Another possible situation where we want to generate scenarios by selection is when the model's users provide their own data set. In such a case, the users may have more confidence in the model and its results if they know that it is using their actual data, instead of 'synthetic' values.

Note that the assumption that the data set represents the 'true' distribution excludes situations where we need to generate scenarios for *conditional* distributions, such as operational models where we want scenarios for *now*, given the latest data, or scenarios for distributions within a multi-stage scenario tree, conditional on values in the preceding nodes of the tree. These situations require a different approach and are therefore left for future research.

The requirement of only using existing values limits the selection of scenario-generation methods, as some result in 'synthetic' values. However, the following methods are applicable, either directly or with a slight modification:

sampling from the data set. By itself, this is not a suitable method for small scenario sets, as the sample distribution can be, by chance, very different from the whole data set. However, Seljom and Tomasgard (2015) show how to improve the match by sampling multiple scenario sets and then choosing the one whose distribution is closest to the data. They use the sum of differences in the first four marginal moments, as well as correlations, as their distance measure.

k-**means clustering** (Lloyd, 1982; MacQueen, 1967), where we divide the data set into a specified number of clusters. In the scenario-generation context, one would then normally use the the clusters' means as scenarios (e.g., Munoz and Watson, 2015). While this would result in synthetic values, we can easily avoid this by selecting the data points closest to the clusters' means instead.

scenario-reduction methods (Dupačová et al., 2003; Heitsch and Römisch, 2003), since the data can be interpreted as a large scenario set. Note that Rujeerapaiboon et al. (2018) call the data-selection approach 'discrete scenario reduction problem', which they contrast to 'continuous scenario reduction problem' where the values can be chosen freely.

In this paper, we present several alternative methods that fit into the described framework:

- new MIP formulation of the moment-matching problem, i.e., minimizing the difference in moments and correlations between the scenarios and the data.
- modification of Wasserstein-distance-minimizing heuristic from Pflug and Pichler (2015), adjusted for the case of data selection.
- variant of the sampling-based approach from Seljom and Tomasgard (2015), using Wasserstein distance as a metric.

For the sake of clarity, we describe and test all the methods on a single-period case,

i.e., the case where data points consist of a single value for each parameter.

The rest of the paper is organized as follows: we start by formulating the one-period problem in Section 2, followed by presentation of the scenario-generation methods in Section 3. In Section 4, we test the methods on a portfolio-optimization problem with CVaR as a risk measure. Finally, Section 5 discusses extending the methods to the multi-period case, i.e., the case where data consist of *sequences* of values for each parameter.

2 One-period selection problem

The scenario-selection problem can be summarized as follows: we have data set $\mathcal{D}$ containing N data points for P parameters, i.e., $D_n \in \mathbb{R}^P$ for $n \in \mathcal{N} = \{1, \ldots, N\}$. From the set, we want to select S values such that the empirical distribution of the subset is as close to the empirical distribution of the whole set as possible. This raises two questions: how to measure the distance between distributions, and how to find a subset that minimizes the chosen metric.

For the distance measure, a natural choice seems to be the Kolmogorov-Smirnov statistic, i.e., the supremum of the absolute distance between the two distribution functions. Unfortunately, this distance does not scale well for multi-variate data, so it is not suitable for our purpose. Another choice is the Wasserstein (or Kantorovich–Rubinstein) metric, also known as the 'earth mover's distance' in computer science. This metric has a known connection to scenario generation and stochastic programming, see for example Pflug (2001); Pflug and Pichler (2011, 2014). Finally, we can follow Seljom and Tomasgard (2015) and measure the distance in terms of differences in marginal moments and correlations. This approach has also an existing connection to scenario generation, starting with Høyland and Wallace (2001).

Once we have chosen a measure, we have to find a method for identifying a subset that minimizes it. One option is the aforementioned 'sample and evaluate' approach from Seljom and Tomasgard (2015), i.e., to randomly select a large number of candidate subsets, evaluate the given measure (they use moments and correlations) on all of them and then select the one that minimizes it. While this approach works, it provides only statistical guarantees about the quality of the identified subset. Indeed, if there are only few subsets that are significantly better than the rest, they might not be discovered by this approach. Moreover, random selection implies equiprobable scenarios, which limits the achievable match.

For this reason, we propose to use optimization, in particular mixed integer linear programming (MIP), for the task. This is applicable both to the Wasserstein distance and, perhaps surprisingly, to the moment based approach, since we are selecting from a set of known values. For the same reason, it is possible to have the output probabilities as variables in both the MIP models, which should help to achieve a better match. This is a potential advantage over the 'sample and evaluate' approach.

In addition, we will test the already-mentioned k-means algorithm, where we split the data set into S clusters and then from each cluster select the data point that is closest

to the cluster's mean. We use the relative cluster sizes as probabilities assigned to the selected data points.

3 Optimization methods for the one-period case

The common part of the optimization models are the binary selection variables x_n and selection probabilities p_n, with the following requirements:

$$\sum_n x_n = S\,. \tag{1}$$

$$p_n \le P^{\max} x_n \qquad n \in \mathcal{N} \tag{2}$$

$$p_n \ge P^{\min} x_n \qquad n \in \mathcal{N} \tag{3}$$

$$\sum_n p_n = 1 \tag{4}$$

Equation (1) ensures that we select exactly S data points and the rest causes the probabilities to be distributed only between the selected data points. Note that the lower bound $P^{\min}$ is required to avoid zero probabilities of selected points, which would in effect mean less than S scenarios. On the other hand, $P^{\max}$ is optional and serves to enforce a more even distribution of probabilities. One possibility is to introduce a new parameter λ, and define

$$P^{\min} = \frac{1}{\sqrt{\lambda} S} \qquad\qquad P^{\max} = \frac{\sqrt{\lambda}}{S} \tag{5}$$

This guarantees that the highest probability is at most λ times larger than the smallest one, independent on S.

If we instead want equiprobable scenarios, we can simply replace (2)–(4) with

$$p_n = \frac{1}{S} x_n \qquad\qquad n \in \mathcal{N}\,. \tag{6}$$

In this case, it is actually possible to substitute p_n out of the model.

It is important to realize that this basic structure includes N binary variables, one for each data point we can select. This implies that there is a limit for how large data sets this approach will be applicable to.

3.1 Minimizing the Wasserstein distance

The scenario-selection problem can be seen as distributing probabilities from the original empirical distribution (assumed to be $1/N$ on every data point) to only the S selected points. To minimize the Wasserstein distance, we want to do this while minimizing the amount of moved probability, multiplied by the move distances. For this, we only need to add the following to (1)–(4) or (1)+(6):

$$\text{minimize} \sum_{ij} \|D_i - D_j\|^r \cdot \pi_{ij} \tag{7}$$

$$\sum_j \pi_{ij} = P_i \qquad i \in \mathcal{N} \tag{8}$$

$$\sum_i \pi_{ij} = p_j \qquad j \in \mathcal{N}\,, \tag{9}$$

where P_i is the probability assigned to data point i (usually $1/N$), π_{ij} is the probability moved from i to j, $\|D_i - D_j\|$ is an appropriate metric, and r is the order of the Wasserstein distance. Note that the distances can be pre-computed, so they are not limited to linear formulas. Also note that (4) may be dropped, as it is implied by (8) and (9). The optimization problem (1)–(4) + (7)–(9) is referred to as the *linear transportation problem* in Heitsch and Römisch (2003) and *optimal quantization* in Löhndorf (2016); Pflug and Pichler (2015), suggesting its relation to different fields of science.

While this model is simple to implement, it does not scale well because of the added variables π_{ij}: the model has N binary and $N^2 + N$ continuous variables, so it becomes intractable beyond ca. thousand data points.

For this reason, one usually resolves to using a heuristic to solve the problem approximately. For this there are two common approaches: one, discussed here, uses a variant of the Lloyd's algorithm (Lloyd, 1982), while the second approach leads to the scenario-reduction techniques discussed in Section 3.5.

In this section, we present a modified version of Algorithm 2 from Pflug and Pichler (2015), which can be summarized as follows:

0. initialize with selection set $\mathcal{Z}^0 = \{z_1^0, \ldots, z_S^0\}$, $D^0 = \infty$, and $k = 1$.
1. find Voronoi partition generated by $\mathcal{Z}^k$, i.e., sets $\mathcal{V}^k(z_s^k)$ for each $s \in \{1, \ldots, S\}$.
2. compute the Wasserstein distance D^k corresponding to $\mathcal{Z}^k$ and partitions $\mathcal{V}^k(z_s^k)$.
3. stop if $D^k \geq D^{k-1}$ (no more improvement)
4. $\mathcal{Z}^{k+1} = \{z_1^{k+1}, \ldots, z_S^{k+1}\}$, where z_s^{k+1} is the 'centre of order r' of partition $\mathcal{V}^k(z_s^k)$.
5. set $k = k + 1$ and goto 1.

Since we operate on a discrete set of points, the Voronoi partitions consist of the original data points $D_i \in \mathcal{N}$, where each D_i is assigned to a partition of its closest z_s^k. Step 1 of the algorithm therefore becomes:

$$D_i \in \mathcal{V}^k(z_s^k) \text{ if } s = \underset{s \in \mathcal{S}^k}{\arg\min} \|D_i - z_s^k\|$$

and the Wasserstein distance in Step 2 becomes

$$D^k = \sum_s \sum_{D_i \in V^k(z_s^k)} P_i \|D_i - z_s^k\|^r \,.$$

Step 4, on the other hand, involves integration also in the discrete case. An important exception is the case where $\|.\|$ is the Euclidean metric and $r = 2$. There, the partition

centres coincide with conditional means, easily computed in the discrete case. In this case, the algorithm becomes the standard Lloyd's algorithm for the k-means problem, which we treat separately in Section 3.4.

Here, we try another approach, where we require also the points z_s^k to be selected from the data set $\mathcal{N}$, i.e., $z_s^k = D_{i_s^k}$. In this case, Step 4 simplifies to

$$z_s^{k+1} = D_{i_s^{k+1}} = \operatorname*{arg\,min}_{D_i \in \mathcal{V}^k(z_s^k)} \sum_{j \in \mathcal{V}^k(z_s^k)} \|D_i - D_j\|^r ,$$

making the whole algorithm easily calculable.[1] Moreover, all computed distances are between the original data points, so they can be pre-computed, speeding up the algorithm. When the algorithm stops, $\mathcal{Z}^k$ becomes the scenario set, with probabilities

$$P_{z_s^k} = \sum_{D_i \in \mathcal{V}^k(z_s^k)} P_{D_i},$$

equal to $|\mathcal{V}^k(z_s^k)|/N$ if the data are equiprobable.

As for the initialization in Step 0, Pflug and Pichler (2015) recommend generating the initial set using Algorithm 1 from the same paper. However, in our case, this algorithm is slow compared to the rest. Therefore, we run Algorithm 2 from multiple randomly generated initial sets and use the selection with the smallest Wasserstein distance. This is significantly faster than using Algorithm 1 and our testing shows no difference in the final distance.

Note that unlike the MIP formulation, the heuristic does not provide any control of the resulting probabilities. Hence, should our optimization problem require equiprobable scenarios, we would simply have to disregard the sizes of the Voronoi sets and assign the same probability to the all resulting scenarios. This could have negative impact on the quality of the approximation, since an outlier data point could end up in a set of its own and then get the same probability as other scenarios.

3.2 Minimizing difference in moments and correlations

Following Høyland and Wallace (2001), we use the first four moments and correlations, with the following definitions:

$$\text{mean} \qquad \mu_p = \mathrm{E}[D_{*,p}] \tag{10}$$

$$\text{variance} \qquad \sigma_p^2 = \mathrm{E}\big[(D_{*,p} - \mu_p)^2\big] \tag{11}$$

$$\text{skewness} \qquad \gamma_p = \mathrm{E}\left[\left(\frac{D_{*,p} - \mu_p}{\sigma}\right)^3\right] = \mathrm{E}\big[(D_{*,p} - \mu_p)^3\big]/\sigma_p^3 \tag{12}$$

$$\text{kurtosis} \qquad \kappa_p = \mathrm{E}\left[\left(\frac{D_{*,p} - \mu_p}{\sigma}\right)^4\right] = \mathrm{E}\big[(D_{*,p} - \mu_p)^4\big]/\sigma_p^4 \tag{13}$$

$$\text{correlation} \qquad \rho_{pq} = \mathrm{E}[(D_{*,p} - \mu_p)(D_{*,q} - \mu_q)]/(\sigma_p\,\sigma_q) \tag{14}$$

[1] Actually, this turns out to be a known algorithm for solving the k-medoids problem (Maranzana, 1963; Park and Jun, 2009).

where $D_{*,p}$ denotes the p-th component of all data points, i.e., all data for parameter $p \in \mathcal{P} = \{1, \ldots, P\}$.

Formulas (13)–(14) are nonlinear due to the scaling by σ_p, so we use unscaled versions instead. Note that this means that if the sample variance of p differs from σ_p^2, then the scaled versions will not be calculated correctly.

Since we have a discrete distribution, expected values simplify to sums over $n \in \mathcal{N}$, with probabilities p_n from either (1)–(4) or (1)+(6). In particular, the scenario-based expected value of the m-th power of parameter $p \in \mathcal{P}$ is

$$r_{p,m} = \mathrm{E}\big[D_{*,p}^m\big] = \sum_n p_n \, D_{np}^m \tag{15}$$

and, using the binomial expansion, the *m-th central moment* as

$$c_{p,m} = \mathrm{E}\big[(D_{*,p} - \mu_p)^m\big] = \sum_{k=0}^{m} \binom{m}{k} (-1)^{m-k} \, \mathrm{E}\big[D_{*,p}^k\big] \mu_p^{m-k} \, . \tag{16}$$

The first equation is linear in p_n, since D_n is data. To make the second equation linear in $\mathrm{E}\big[D_{*,p}^k\big]$, and hence also in p_n, we need μ to be a constant. In other words, we have to replace the actual sample mean by its target value. Hence, the computed moments will be exact only if the sample mean is exactly equal to μ_p. Alternatively, we can avoid the approximation by matching directly the expected powers $\mathrm{E}\big[D_{*,p}^m\big]$.[2]

For correlations, we use the fact that

$$\mathrm{E}[(D_{*,p} - \mu_p)(D_{*,q} - \mu_q)] = \mathrm{E}[D_{*,p} \, D_{*,q}] - \mu_p \, \mu_q$$

and

$$\mathrm{E}[D_{*,p} \, D_{*,q}] = \sum_n p_n \, D_{np} \, D_{nq} \, ,$$

where the second equation is linear in p_n since D is data.

Just like for the moments, we have the choice of matching the correlations, with μ_i and σ_i replaced by their target values, or matching only $\mathrm{E}[D_i \, D_j]$.

Putting it all together, the complete model for the moment-based approach, in addition to (1)–(4) or (1)+(6), is:

$$\text{minimize} \ \sum_m W_m \sum_p (d_{pm}^+ + d_{pm}^-) + \sum_{p<q} W_{pq} (\delta_{pq}^+ + \delta_{pq}^-) \tag{17}$$

[2] Another way of avoiding the approximation is adding $r_{p1} = \mu_p$ as a constraint to the model. This, however, could make the model infeasible, especially in cases with small S.

subject to:

$$r_{pm} = \sum_n p_n \, D_{pn}^m \qquad p \in \mathcal{P},\, m \in \mathcal{M} \qquad (18)$$

$$c_{pm} = \sum_{k=0}^{m} \binom{m}{n} (-1)^{m-k} \, r_{pm} \, \mu_p^{m-k} \qquad p \in \mathcal{P},\, m \in \mathcal{M} \qquad (19)$$

$$d_{pm}^+ \geq c_{pm}/\sigma_p^m - M_{pm} \qquad p \in \mathcal{P},\, m \in \mathcal{M} \qquad (20)$$

$$d_{pm}^- \geq M_{pm} - c_{pm} \qquad p \in \mathcal{P},\, m \in \mathcal{M} \qquad (21)$$

$$r_{pq} = \sum_n p_n \, D_{np} \, D_{nq} \qquad p, q \in \mathcal{P},\, p<q \qquad (22)$$

$$s_{pq} = (r_{pq} - \mu_p \, \mu_q)/(\sigma_p \, \sigma_q) \qquad p, q \in \mathcal{P},\, p<q \qquad (23)$$

$$\delta_{pq}^+ \geq s_{pq} - \Sigma_{pq} \qquad p, q \in \mathcal{P},\, p<q \qquad (24)$$

$$\delta_{pq}^- \geq \Sigma_{pq} - s_{pq} \qquad p, q \in \mathcal{P},\, p<q \qquad (25)$$

where $\mathcal{M} = \{1, 2, 3, 4\}$ is the set of considered moments, M_{pm} are the target values of the central moments of parameter $p \in \mathcal{P}$, $\mu_p = M_{p1}$ and $\sigma_p = \sqrt{M_{p2}}$. In addition, we define $r_{p0} = 1$ in (19).

Moreover, W_m and W_{pq} are weights for distances in moments and correlations, respectively. Since we usually expect the sensitivity of an optimization problem to decrease with the order of the moment of the input data (see, e.g., Chopra and Ziemba, 1993), it is natural to use a decreasing sequence of weights. Another argument for higher weights on mean and variance is that, as we have already seen, mismatch in those moments implies error in evaluation of the higher moments, as well as correlations. In our test case, we have used $W = \{10, 5, 2, 1\}$ and $W_{pq} = 3$ for all p, q.

Note that any of r_{pm}, c_{pm}, r_{pq}, and s_{pq} can be easily substituted out of the model, reducing the number of variables but creating a denser LP matrix. Our testing with FICO™ Xpress indicates that this has some impact on solution times, but the exact choice of what to substitute is likely to differ between solvers and solution algorithms, so we do not present the results here.

3.3 k-means clustering

The k-means clustering algorithm (Lloyd, 1982; MacQueen, 1967) is used to divide a data set into a given number of clusters. As we have seen in Section 3.1, it is also a special case of the heuristic for minimizing the mass-transportation problem (1)–(4) + (7)–(9), with Euclidean distance and $r = 2$. Once the algorithm finishes, we need to select a representative scenario from each cluster. For this, we simply use the scenario closest to the cluster's mean. We then use the relative cluster sizes as the probabilities assigned to the scenarios.

The advantage of k-means is that it is a standard method with many available implementations and that the method is very fast. On the other hand, there is no guarantee that the resulting scenarios constitute a good approximation of the distribution.

Note that this method, just like the Wasserstein heuristic, does not provide control over the resulting cluster sizes and hence the scenario probabilities. However, there are alternative versions of the k-means method that provide some control over cluster sizes, such as the 'constrained k-means clustering' method from Bennett et al. (2000). The downside of this variant is longer run-time, compared to the standard k-means.

3.4 Sampling-based approaches

The 'sampling and evaluate' approaches provide another way to generate scenarios. One obvious advantage is their simplicity: we randomly sample scenarios from the data, compute their distance from the target distribution, and keep track of the best scenario set. They are also flexible, as they work with any distance measure that we can implement and that can be evaluated quickly enough.

For instance, this approach is well suited for use with moments and correlations, since these are both easy to implement and fast to evaluate (Seljom and Tomasgard, 2015).

It is also possible to use it with the Wasserstein distance. With scenario selection given by the sample, we can remove x_n and all $j \in \mathcal{N} \setminus \mathcal{S}$ from (1)–(4) + (7)–(9), which then simplifies to

$$\text{minimize} \sum_{i \in \mathcal{N}, j \in S} \|D_i - D_j\| \cdot \pi_{ij} \tag{7'}$$

$$\sum_{j \in \mathcal{S}} \pi_{ij} = P_i \qquad i \in \mathcal{N} \tag{8'}$$

$$\sum_{i \in \mathcal{N}} \pi_{ij} = p_j \qquad j \in \mathcal{S}\,, \tag{9'}$$

$$p_j \geq P^{\min} \qquad j \in \mathcal{S} \tag{3'}$$

It turns out that this model can be solved analytically (Theorem 2 of Dupačová et al., 2003), by setting

$$\pi_{ij} = \begin{cases} 1 & \text{if } j = \arg\min_{j \in \mathcal{S}} \|D_i - D_j\| \\ 0 & \text{otherwise}\,, \end{cases}$$

as long as we do not require (3') – which is no longer needed to ensure $p_j > 0$, as the construction guarantees $p_j \geq \min_i P_i$. Since π_{ij}'s are either zero or one, this solution assigns every input $i \in \mathcal{N}$ to one of the selected points $j \in \mathcal{S}$, in effect partitioning the set into clusters. This provides a justification for Step 1 of the heuristic presented in Section 3.1.

Note that no such analytical solution exists if we require fixed probabilities. In that case, we would have no choice but to solve the LP (7')–(9'), with $p_j = 1/S$, in each iteration of the sampling algorithm.

3.5 Scenario-reduction techniques

Scenario-reduction techniques from Dupačová et al. (2003); Heitsch and Römisch (2003) provide an alternative way to approximately solve the linear transportation problem

(1)–(4) + (7)–(9). They take advantage of the fact that the problem is easily solvable if we want to either select or remove a single point from the data, and come with two types of methods:

- 'reduction' methods, where we start from the whole data set and remove one point at a time until we reach the require size.
- 'selection' methods, where we start with empty set and then add one point from the data at a time

In both papers, testing shows that the selection methods generate better scenarios (smaller Wasserstein distance from the data set), but can be slow for big S. Heitsch and Römisch (2003) therefore recommend to use the *fast forward selection* method as long as $S \lessapprox N/4$. Since our tests will be well within this limit, we use this method (Algorithm 2.4 of Heitsch and Römisch, 2003) in our tests.

4 Test: portfolio-optimization problem with CVaR

In this section, we test all the approaches presented in the previous section on a simple portfolio-optimization problem, with CVaR as a risk measure (Rockafellar and Uryasev, 2000; Uryasev, 2000). We put CVaR into the objective, with risk weight W_R, so the model is:

$$\text{maximize } (1 - W_R) \sum_{s \in \mathcal{S}} P_s \, y_s + W_R \, c \tag{26}$$

$$\sum_i x_i \leq B \qquad i \in \mathcal{I} \tag{27}$$

$$x_i \leq W_I \, B \qquad i \in \mathcal{I} \tag{28}$$

$$y_s = \sum_i x_i \, (1 + R_{is}) - B \qquad s \in \mathcal{S} \tag{29}$$

$$z_s \geq \zeta - y_s \qquad s \in \mathcal{S} \tag{30}$$

$$c = \zeta - \frac{1}{\alpha} \sum_{s \in \mathcal{S}} P_s \, z_s \tag{31}$$

There, x_i is the amount invested to instrument $i \in \mathcal{I}$, limited by a budged B plus maximum fraction invested into one instrument, W_I. R_{is} are the instrument returns in scenario $s \in \mathcal{S}$, so y_s is the expected profit. ζ represents a threshold, equal to value-at-risk (VaR) at the optimum. Therefore, z_s is the profit below VaR and c the conditional value-at-risk (CVaR) with confidence level α. In our test, we use $\alpha = 0.05$, $B = 1000$, and $W_I = 0.25$.

For our data, we use 5 years of daily close prices for the 25 largest stocks on the Oslo stock market, with data starting in September 2015. From this, we compute weekly price returns, resulting in 1248 data points. We consider three different problem dimension, by using the first 10, 20, or 25 stocks, and test four different scenario-set sizes, $S \in \{10, 20, 50, 100\}$.

Both Wasserstein-based approaches use the Euclidean norm in calculation of the distances, and $r = 2$. For the moment-based optimization model, we set the parameter λ from (5) to 10. Moreover, since Xpress struggles to close the optimality gap of the model (1)–(4) + (17)–(25), we stop it after a specified time limit (see below).

The scenario-generation framework has been implemented in Python. The MIP model were written in Pyomo (Hart et al., 2011, 2017) and solved using FICO™ Xpress solver. All source files are freely available from `https://github.com/SINTEF/scengen-data-select`.[3]

We test the following scenario-generation methods and variants:

WD heuristic is the heuristic described in Section 3.1. The method is initialized with 10 different random selections.

opt. 300 s is the moment-based optimization with a time limit of 300 s.

opt. 1800 s is the moment-based optimization with a time limit of 1800 s.

sampling 1x is using just a single sample, to serve as a benchmark.

sampl. MM 500x is selecting from 500 samples, using the moment-based dist.

sampl. MM 10000x is the same with 10000 samples.

sampl. WD 500x is selecting from 500 samples, using the Wasserstein metric.

sampl. WD 10000x is the same with 10000 samples.

k-**means,** using Python implementation from scikit-learn (Pedregosa et al., 2011). This involves starting the method from 10 different random selections.

scen. reduction, using our Python implementation of the fast forward selection method.

For each combination of scenario-generation method, dimension, and scenario-set size, we generated 25 trees – with the exception of optimization and scenario reduction, where we generate only one since they would all be equal. This gives $(7 \times 25 + 3) \times 3 \times 4 = 2136$ scenario sets.

4.1 Generation times

The average generation times are reported in Fig. 1. We do not show the moment-based optimizations, because they run until their time limit, and the sampling-based methods with 10 000 iterations, since this simply takes 20× longer than 500 iterations. All reported times are from a laptop with 2.8 GHz Intel® i7 processor and 16 GB of RAM.

We see that k-means is the fastest method, followed by the Wasserstein heuristic. However, it should be pointed out that the k-means method in scikit-learn is highly optimized, while our implementation of the Wasserstein heuristic is not. Of the sampling methods, the moment-based sampling method is significantly faster than the one using Wasserstein distance. Finally, scenario reduction is the slowest of the shown methods.

[3] Comming soon

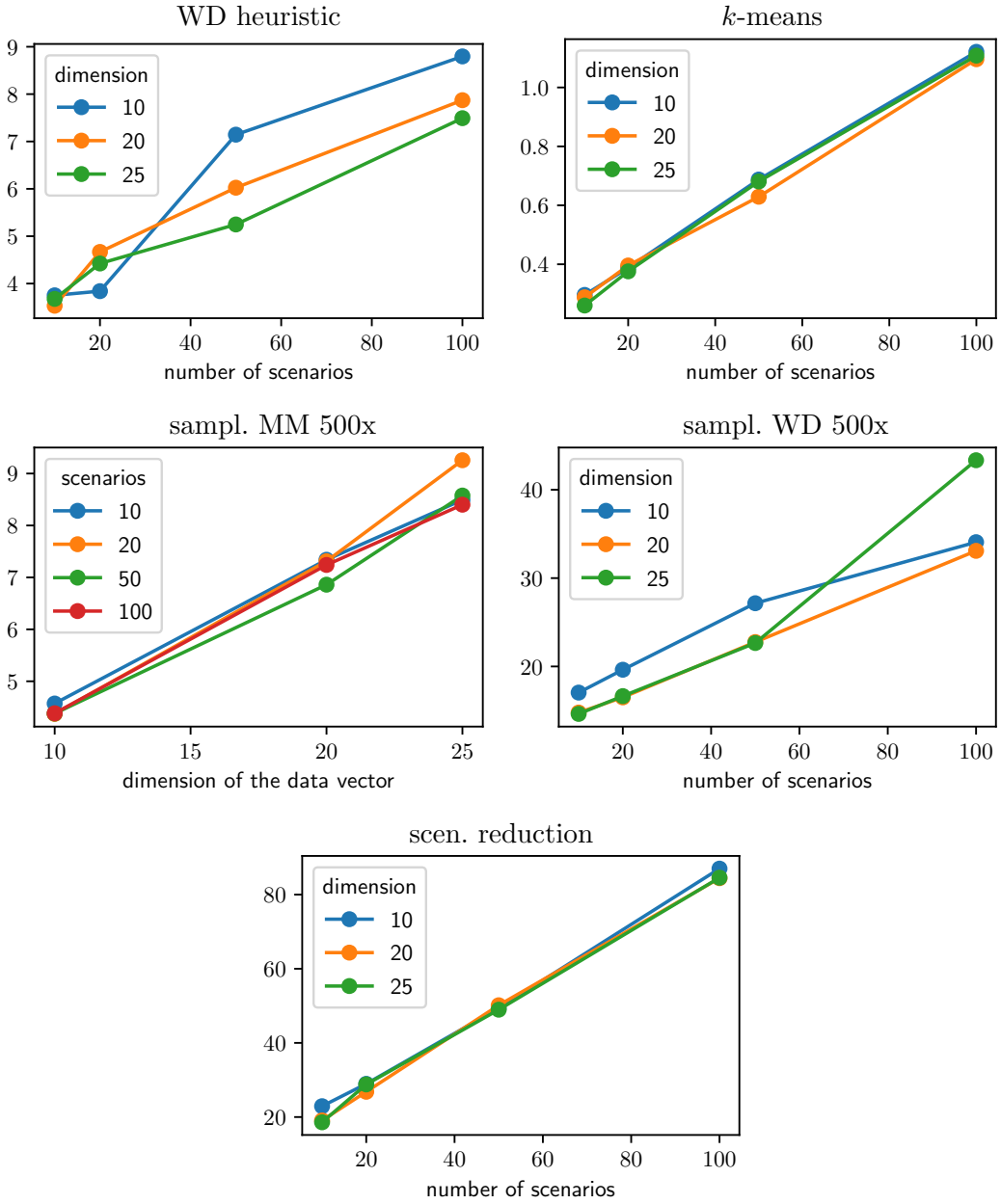

Figure 1: Average scenario-generation times

We can see that runtime of k-means and all Wasserstein-based methods, including scenario reduction, does not depend on dimension. This is expected, since the dimension enters only into the calculation of distances and otherwise does not influence the algorithm. Otherwise, the generation time increases approximately linearly with the number of scenarios, i.e., the number of clusters in k-means and the Wasserstein-distance heuristic.

The situation is reversed for the moment-based sampling, with generation times independent on the number of scenarios and increasing with dimension. The latter is expected, as higher dimension means more moments and correlations to evaluate and compare. Since the time to calculate moments and correlations must increase with the number of scenarios, the result suggests that it is not a significant factor in the overall run-time.

4.2 Out-of-sample evaluations

Next, we perform the out-of-sample evaluations of the trees, in the sense defined in Kaut and Wallace (2007). We do this for six different values of the risk weight, $W_R \in \{0, 0.1, \ldots, 0.5\}$. For each W_R, we solve the optimization problem (26) – (31) on all the generated trees. Next, we take the $6 \times 2136 = 12\,816$ scenario-based solutions and evaluate them on the full data set, i.e., solve the problem using the data set as scenarios, with investment variables x fixed to the values from those solutions. This gives us out-of-sample evaluations of the solutions, and hence a measure of the quality of the scenarios sets they originated from.

In addition, we solve the problem on the data set without any restrictions to get the 'true' optimal solution, for each value of W_R.

4.2.1 Difference in objective value

Since we have, for each scenario set and W_R, both the out-of-sample evaluation and the actual 'true' optimal objective value, we can subtract the two to get a measure of sub-optimality caused by the scenario set; following Pflug and Pichler (2011), we will refer to it as the *discretization error*, but it is also known as the *approximation error* (Pflug, 2001), or *policy error* (Löhndorf, 2016). Then we can compare distributions of the discretization errors across all the test cases, for scenario sets generated by different methods.

The results are presented in Figs. 2 and 3, where the former shows the whole distributions, while the latter zooms in for easier comparisons of medians and quartiles. We omit the sampling-based methods with 10 000 iterations, since there was no visible improvement compared to the variants with 500 iterations.

Unsurprisingly, using a random sample is the worst method. Comparing the two methods selecting from 500 samples, we can see that moments and correlations are a better selection criteria than the Wasserstein distance, for our optimization problem.

The Wasserstein-distance heuristic is slightly better than any of the sampling-based methods, but worse than both k-means and the moment-based optimization. Further-

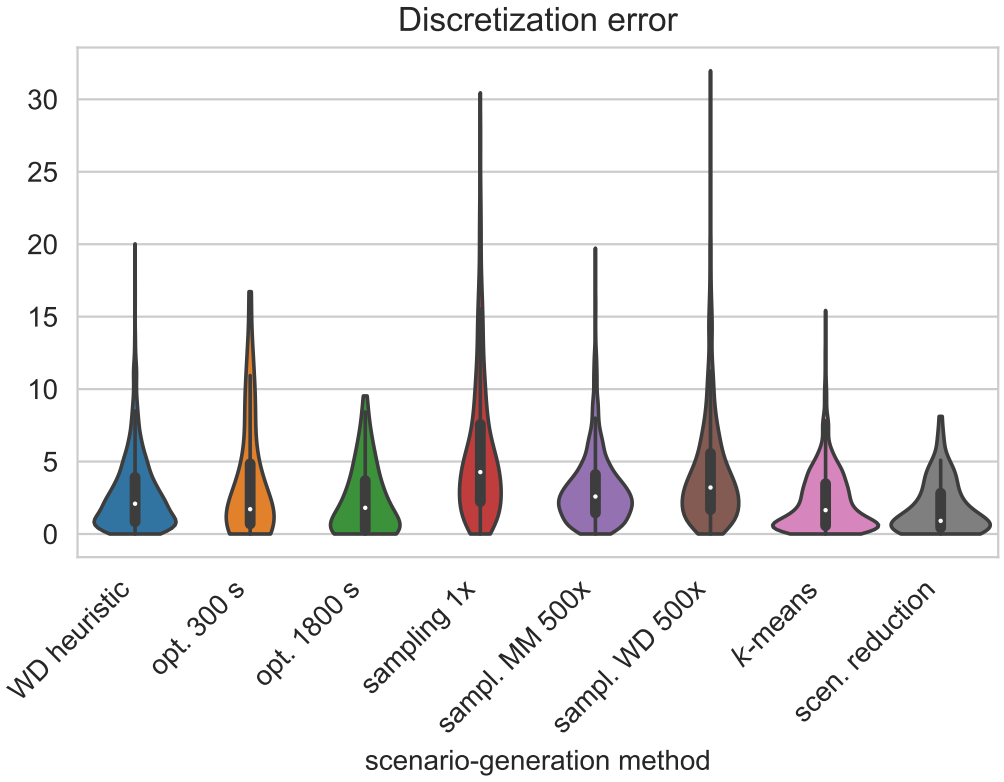

Figure 2: Distribution of discretization errors across all the generated scenario sets. The inside boxes in each plot represent the inner quartiles, with the median denoted by the white dot. Moreover, the middle line shows the estimated range, limited to 1.5 times the inter-quartile range beyond the quartiles.

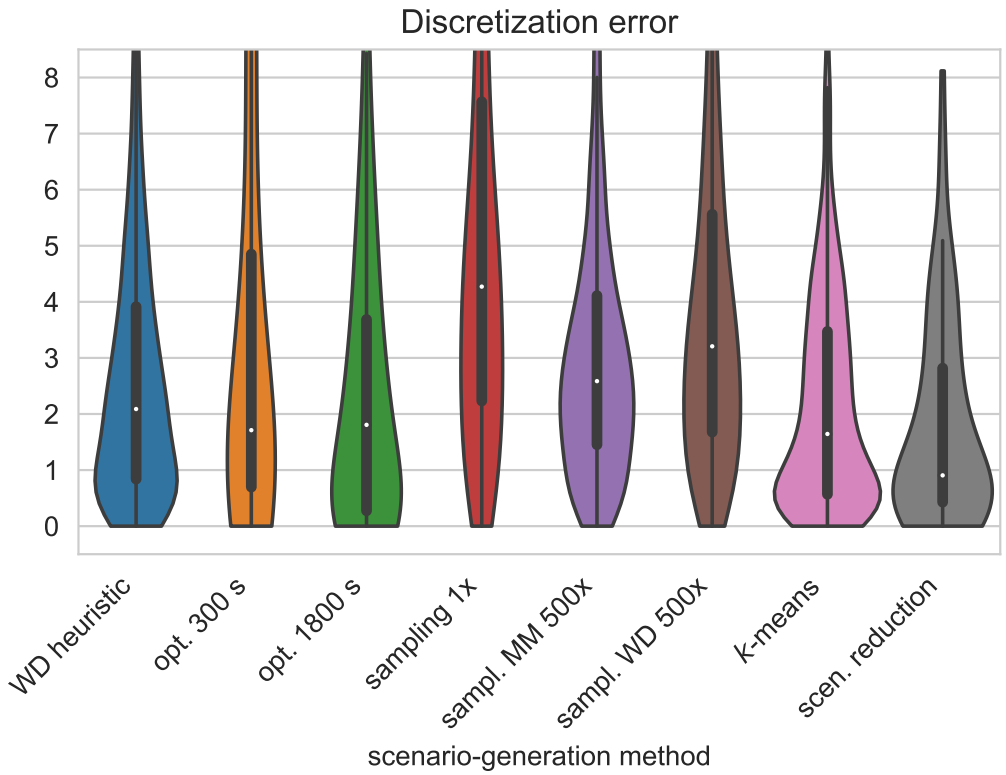

Figure 3: Zoomed-in version of Fig. 2.

more, for the optimization method, we see that giving the solver more time helps to shift the distribution downwards, even though the median remains unchanged.

Finally, scenario reduction is clearly the best method overall, with median discretization error almost half that of the next-best methods.

We have also tested splitting the test cases by dimension, number of scenarios, and values of the risk weight. From this, we could conclude that the above observations hold for most subsets. The biggest difference was when we considered only sets with 10 scenarios, where the moment-based optimization method with 300-second limit performed almost as bad as random sampling. This is probably due to fact that with 10 scenarios, it is difficult to get a good match in means and variances, causing errors in evaluations of the other moments, as mentioned in Section 3.2.

On the other hand, the moment-based method with 1800-second limit was better than scenario reduction for $W_R \in \{0, 0.1\}$. For the risk-neutral case ($W_R = 0$), its median discretization error is virtually zero, suggesting that it found the actual optimal value in (at least) 50% of cases.

Apart from that, we can observe that:

- dimension matters: the discretization error is several times bigger for 25 assets, compared to 10;
- more scenario helps: as expected, the discretization error decreases with the number of scenarios;
- risk-aversion matters: the discretization error increases with W_R;[4]

Moreover, the nature of the optimization problem allows us to plot the solutions in a mean-risk plane and compare them to the 'true' efficient frontier. This way, we can get some additional insights in the way the different methods work.

For example, comparing Figs. 4 and 5 shows that compared to k-means, the Wasserstein-based sampling results in portfolios with higher risk, i.e., with worse CVaR (solutions 'to the left' of the efficient frontier). This suggests that its scenarios do not represent the distributions' tails as well as those based on k-means.

5 Extending the methods to multi-period sequences

So far, we have limited ourselves to the case of single-period data. However, as we have pointed out in the start of the paper, our motivation for the data-selecting approach came from the case where we select whole sequences, such as hourly values for a day or a week. In this section, we will show how to extend the methods from Section 3 to handle this.

The extension means that our data set $\mathcal{D}$ now contains N data series for P parameters, with each series containing T values, i.e., $D_{np} \in \mathbb{R}^T$ for each $n \in \mathcal{N}$ and $p \in \mathcal{P}$.

[4] This is also expected, since scenarios for the risk-neutral case only need to match the means in order to produce correct results. CVaR, on the other hand, requires a good representation of the tails as well, so increasing its weight in the objective makes the scenario-generation process more demanding.

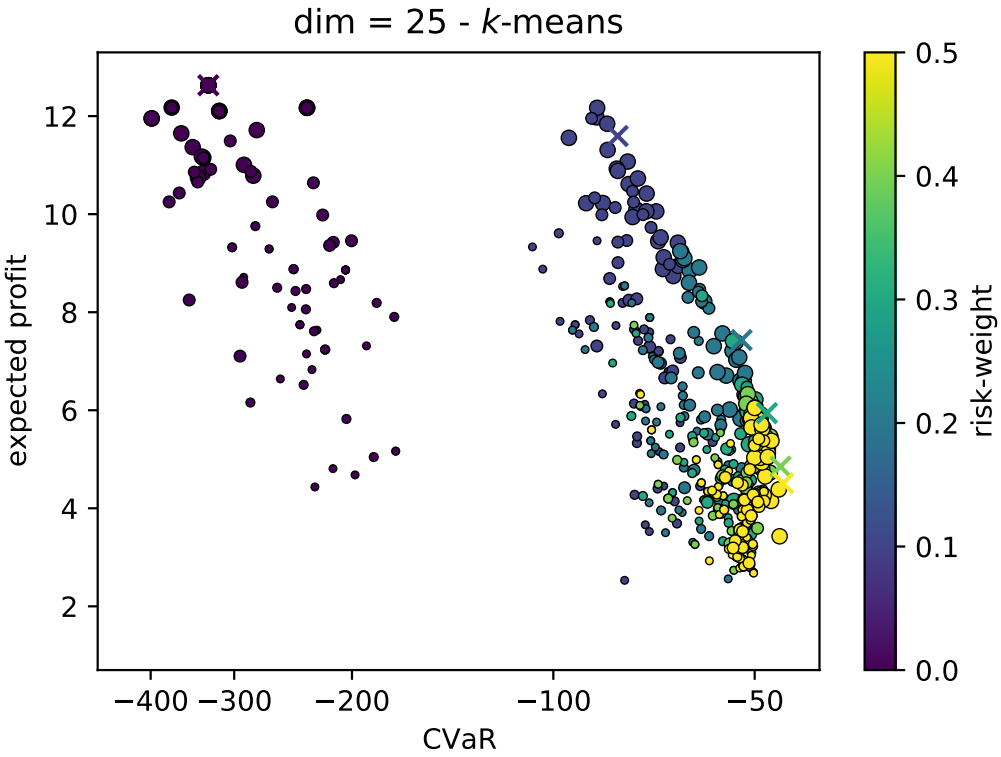

Figure 4: Out-of-sample expected profit and CVaR for solutions based on scenarios generated using the moment-based 'sampl. MM 500x' method, with 25 assets. The point size correspond to the size of the scenario sets, while colour denotes the risk-weight parameter W_R. The crosses show the optimal solution for all tested values of W_R, so they lie on the mean-CVaR efficient frontier. Note the log-scale on the x-axes (CVaR).

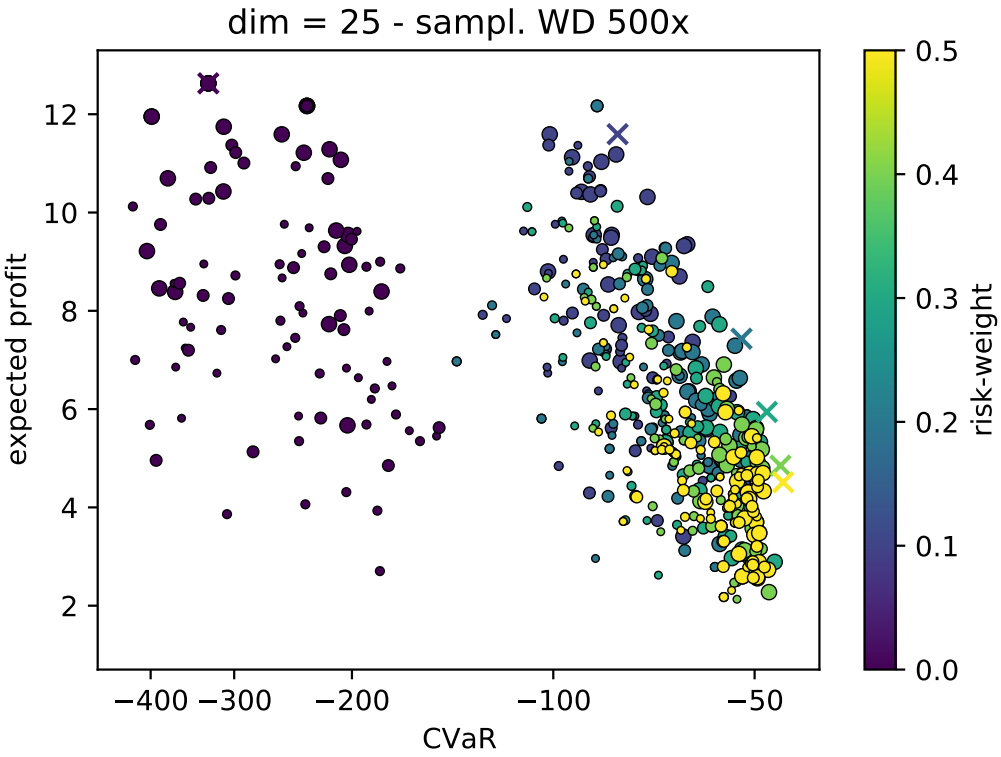

Figure 5: Version of Fig. 4 for the Wasserstein-based 'sampl. WD 500x' method.

However, this data can be interpreted as having N data points for $P \times T$ parameters, converting any temporal dependencies into spacial ones. This way, we can convert the problem into the previous one, at the cost of increasing the dimension T times. For the above-mentioned cases of hourly values for one day or week, this means 24- or 168-fold increase in dimension, which could have significant implications for the scenario-generation methods.

Fortunately, both k-means and Wasserstein-distance-based methods are basically independent of dimension, as seen in Fig. 1. This is because they can operate on a pre-computed set of distances between the data points, so the dimension enters only into pre-processing.

For the methods based on moments and correlations, on the other hand, the amount of evaluated distances grows quadratically with the dimension because of correlations. This will make the sampling with moment matching significantly slower, and the optimization model most likely unsolvable for most practical problems.

There is also another issue to consider: with such a significant increase in dimension, we cannot really expect to get a very good match, unless we increase the number of scenarios correspondingly. In many cases, there might be some natural way to aggregate the sequences in order to decrease the dimension and therefore improve the achievable match. For example, if we are selecting days from a data set with hourly inflows, matching the distribution of total daily inflows could be enough—while the fact that we use historical data would ensure that the intra-day profiles are realistic. Similarly, if we generate scenarios for wind- or solar-power capacity factor, then it could suffice to match the distribution of the daily average values (so we have a realistic distribution of good and bad days). We could also choose a middle way and do a partial aggregation, by using only a couple of values per day (day/night, peak/off-peak, etc).

In other words, while we can use the same methods as for the single-period case, they require some extra considerations. We will present a study describing generation of multi-period scenarios for the TIMES energy-system model in an forthcoming paper.

6 Conclusions

In this paper, we have presented five different methods for selecting representative data points or sequences from historical data, in order to obtain a good approximation of the empirical distribution represented by the data. Such methods are needed in cases where the model users do not want to use synthetic values and therefore allow only historical data in scenarios for a given optimization model.

The methods range from off-the-shelf k-means algorithm, through easy-to-implement scenario reduction and iterative sampling methods with two different distance measures, to new optimization models and Wasserstein-based heuristic.

We have then compared the methods using a simple portfolio-optimization model with CVaR as a risk measure. In our case, the 'fast forward selection' scenario-reduction method worked best, followed by k-means that was fastest. However, these results are likely to be problem-dependent. For example, the sampling-based methods generally

produce equiprobable scenarios, so they cannot provide as good a match as methods with adjustable probabilities. On the other hand, k-means and the Wasserstein-based methods, including scenario reduction, cannot control the output probabilities – if we want equiprobable scenarios, we have to disregard the computed probabilities, which is likely to have an impact on the quality of the scenarios. Consequently, should we generate scenarios for an optimization model that requires equiprobable scenarios, such as the TIMES energy-system model, the sampling-based methods would loose their disadvantage while other methods would not work optimally, so the conclusion could change.

The main contribution of the paper is therefore not the results of the particular test case, but rather the presentation of the methods, including formulation of the moment-based optimization model as well as the Wassertein-metric-based heuristic. In addition, the presented 'sample and evaluate' methods can be useful for comparing different distance measures and identifying the most relevant measure for a given optimization model.

There are two natural avenues for future research: the first is to test the presented methods on a large-scale model, such as TIMES, and the second is to develop methods that can handle conditional distributions. The latter would allow usage for operational models (conditional on the latest data and possibly a forecast), as well as multi-stage models with dependences between stages.

Acknowledgements

This work has been done as a part of research project "Assessment of the Value of Flexibility Services from the Norwegian Energy System" (ASSETS), a Norwegian Research Council project no. 268097.[5]

References

Bennett K, Bradley P, Demiriz A (2000) Constrained k-means clustering. resreport MSR-TR-2000-65, Microsoft, URL `https://www.microsoft.com/en-us/research/publication/constrained-k-means-clustering/`

Chopra V, Ziemba W (1993) The effects of errors in means, variances, and covariances on optimal portfolio choice. The Journal of Portfolio Management 19(2):6–11

Dupačová J, Gröwe-Kuska N, Römisch W (2003) Scenario reduction in stochastic programming: An approach using probability metrics. Mathematical Programming 95(3):493–511, DOI 10.1007/s10107-002-0331-0

Hart WE, Watson JP, Woodruff DL (2011) Pyomo: modeling and solving mathematical programs in Python. Mathematical Programming Computation 3(3):219–260, DOI 10.1007/s12532-011-0026-8

[5]See `https://prosjektbanken.forskningsradet.no/#/project/NFR/268097`.

Hart WE, Laird CD, Watson JP, Woodruff DL, Hackebeil GA, Nicholson BL, Siirola JD (2017) Pyomo – Optimization Modeling in Python, Springer Optimization and Its Applications, vol 67, 2nd edn. Springer International Publishing, DOI 10.1007/978-3-319-58821-6

Heitsch H, Römisch W (2003) Scenario reduction algorithms in stochastic programming. Computational Optimization and Applications 24(2–3):187–206, DOI 10.1023/A:1021805924152

Høyland K, Wallace SW (2001) Generating scenario trees for multistage decision problems. Management Science 47(2):295–307

Kaut M, Wallace SW (2007) Evaluation of scenario generation methods for stochastic programming. Pacific Journal of Optimization 3:257–271

Lloyd SP (1982) Least squares quantization in PCM. IEEE Transactions on Information Theory 28(2):129–137, DOI 10.1109/tit.1982.1056489, prev. published as a Technical Report RR-5497, Bell Labs, 1957.

Löhndorf N (2016) An empirical analysis of scenario generation methods for stochastic optimization. European Journal of Operational Research 255(1):121–132, DOI 10.1016/j.ejor.2016.05.021

Loulou R, Lettila A (2016) Stochastic programming and tradeoff analysis in times. Tech. rep., IEA-ETSAP, URL `https://iea-etsap.org/index.php/documentation`

Loulou R, Goldstein G, Kanudia A, Lettila A, Remme U (2016) Documentation for the TIMES model – part I. Tech. rep., IEA-ETSAP, URL `https://iea-etsap.org/index.php/documentation`

MacQueen J (1967) Some methods for classification and analysis of multivariate observations. In: Cam LML, Neyman J (eds) Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, University of California Press, vol 1, pp 281–297

Maranzana FE (1963) On the location of supply points to minimize transportation costs. IBM Systems Journal 2(2):129–135, DOI 10.1147/sj.22.0129

Munoz FD, Watson JP (2015) A scalable solution framework for stochastic transmission and generation planning problems. Computational Management Science 12(4):491–518, DOI 10.1007/s10287-015-0229-y

Park HS, Jun CH (2009) A simple and fast algorithm for K-medoids clustering. Expert Systems with Applications 36(2):3336–3341, DOI 10.1016/j.eswa.2008.01.039

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in Python. Journal

of Machine Learning Research 12:2825–2830, URL https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html

Pflug GC (2001) Scenario tree generation for multiperiod financial optimization by optimal discretization. Mathematical Programming, 89(2):251–271, DOI 10.1007/PL00011398

Pflug GC, Pichler A (2011) Approximations for probability distributions and stochastic optimization problems. In: Bertocchi M, Consigli G, Dempster MAH (eds) Stochastic Optimization Methods in Finance and Energy, International Series in Operations Research & Management Science, vol 163, Springer, chap 15, pp 343–387, DOI 10.1007/978-1-4419-9586-5_15

Pflug GC, Pichler A (2014) Multistage Stochastic Optimization. Springer Series in Operations Research and Financial Engineering, Springer, DOI 10.1007/978-3-319-08843-3

Pflug GC, Pichler A (2015) Dynamic generation of scenario trees. Computational Optimization and Applications 62(3):641–668, DOI 10.1007/s10589-015-9758-0

Rockafellar R, Uryasev S (2000) Optimization of conditional value-at-risk. Journal of Risk 2:21–42, DOI 10.21314/JOR.2000.038

Rujeerapaiboon N, Schindler K, Kuhn D, Wiesemann W (2018) Scenario reduction revisited: fundamental limits and guarantees. Mathematical Programming DOI 10.1007/s10107-018-1269-1

Seljom P, Tomasgard A (2015) Short-term uncertainty in long-term energy system models — A case study of wind power in Denmark. Energy Economics 49:157–167, DOI 10.1016/j.eneco.2015.02.004

Skar C, Doorman G, Pérez-Valdés G, Tomasgard A (2016) A multi-horizon stochastic programming model for the european power system. techreport 2/2016, CenSES, URL https://www.ntnu.no/censes/working-papers

Uryasev S (2000) Conditional value-at-risk: Optimization algorithms and applications. Financial Engineering News 14:1–5, DOI 10.1109/CIFER.2000.844598