Decision Making under Uncertainty

Assignment 1: Stochastic Knapsack

Group 2

Martijn Ketelaars (SNR: 1263857, ANR: 120975)
Robbie Reyerse (SNR: 2039047, ANR: 109997)
Rosalien Timmerhuis (SNR: 2031797, ANR: 520618)
Mike Weltevrede (SNR: 1257560, ANR: 756479)

October 25, 2019

Contents

Introduction

This assignment is about the Stochastic Knapsack Problem (SKP). In the general SKP you are choosing a subset among N items ($i = 1, ..., N$) to allocate to a knapsack of capacity K. We denote the set of items by $I = \{1, 2, \ldots, N\}$. Each item $i \in I$ has a random size of w_i units. We assume that item sizes are statistically independent. If the collective size of the selected items exceeds the capacity, a penalty of p is assessed per unit excess. We assume that item $i \in I$ has an associated per-unit revenue r_i with $r_i < p$. Different objective functions are possible when modeling the SKP. In our case we want to maximize the profit.

The binary decision variable x_i takes a value of 1 if item $i \in I$ is allocated to the knapsack and a value of 0 otherwise. All items are considered simultaneously and the values of their sizes are unknown before selection decisions are made. In this assignment, the item size w_i can take only two values, namely d_h_i and d_l_i, with probabilities $P(w_i = d\_h_i) = \pi_i$ and $P(w_i = d\_l_i) = 1 - \pi_i$.

Generation of Problem Instances

Part 1.

Our group number is 2, so we take $g = 2$. Consequently, we have penalty $p = 60$ and capacity $K = 408$. We denote the set of problem instances by $J = \{1, 2, \ldots, 10\}$. We will generate 10 random problem instances. The probabilities π_i and revenues r_i for $i \in I$ we need for this are given in Table 1.

Table 1: Probabilities and revenues for each of the 10 items in each problem instance.

Item i	1	2	3	4	5	6	7	8	9	10
Probability π_i	0.549	0.599	0.649	0.699	0.749	0.799	0.849	0.899	0.949	0.999
Revenue r_i	50	49	48	47	46	45	44	43	42	41

Each item $i \in I$ has a random size w_i. There are two possible sizes: d_l_i (low) and d_h_i (high). Moreover, $d\_l_i = \min\{\gamma_i, 10\}$ where $\gamma_i \sim POI\left(\lceil \frac{i}{2} \rceil\right)$ and d_h_i follows a triangular distribution with lower value equal to $92 - i$, mode equal to $102 - i$ and upper value equal to $112 - i$. With probability π_i the size is equal to d_h_i and with probability $(1 - \pi_i)$ it is equal to d_l_i. For each problem instance $j \in J$ we sampled d_l_i and d_h_i for each $i \in I$. The results are stated in Table 8 in Appendix A. The code for this assignment can be found in the Jupyter notebook file.

Heuristic Algorithm

Part 2.

We want that our simple greedy heuristic algorithm starts to add items that have the highest revenue per unit. We employ the expected value of the size, $\mathbb{E}[w_i]$, of each item $i \in I$ in our greedy algorithm due to the fact that item sizes are random and thus unknown beforehand. Let $i \in I$. Then,

$$
\begin{aligned}
\mathbb{E}[w_i] &= \mathbb{P}(w_i = d\_l_i)\mathbb{E}[d\_l_i] + \mathbb{P}(w_i = d\_h_i)\mathbb{E}[d\_h_i] \\
&= (1-\pi_i)\mathbb{E}[d\_l_i] + \pi_i \mathbb{E}[d\_h_i] \\
&= (1-\pi_i)\Big(\sum_{k=0}^{10} k\mathbb{P}(\gamma_i = k) + 10\mathbb{P}(\gamma_i > 10)\Big) + \pi_i \frac{92 - i + 102 - i + 112 - i}{3} \\
&= (1-\pi_i)\Big(\sum_{k=0}^{10} k\mathbb{P}(\gamma_i = k) + 10\big(1 - \sum_{k=0}^{10}\mathbb{P}(\gamma_i = k)\big)\Big) + \pi_i(102 - i) \\
&= (1-\pi_i)\Big(10 - \sum_{k=0}^{9}(k-10)\mathbb{P}(\gamma_i = k)\Big) + \pi_i(102 - i),
\end{aligned}
$$

where $\mathbb{P}[\gamma_i = k] = (\lambda_i^k e^{-\lambda_i})/k!$ with $\lambda_i = \lceil \frac{i}{2} \rceil$. The first equality follows from the law of total expectation, and the third equality follows from the fact that $\mathbb{E}(X) = \frac{a+b+c}{3}$ if a random variable X follows a triangular distribution with lower value a, mode c and upper value b.

Knowing $\mathbb{E}[w_i]$ for all $i \in I$, our greedy algorithm is as follows.

1. Initialize $W = 0$, and $x_i = 0$ for all $i \in I$.
2. Consider item $i \in I$ with largest r_i.
3. If $W = W + \mathbb{E}(w_i) \leq K$ add item $i \in I$ (i.e. set $x_i = 1$), set $W = W + \mathbb{E}(w_i)$, and set $I = I \backslash \{i\}$. Otherwise, only set $I = I \backslash \{i\}$.
4. If either $I = \emptyset$ or $W = K$ terminate the greedy algorithm. Otherwise, go back to step 3.

Thus, in each iteration we try to add the item (from the set of remaining items) for which its revenue is highest. If we are not able to add this item (because its expected weight exceeds the remaining capacity of the knapsack) and there are still other items that have not been considered, it may as well be possible that one of those items can be added instead. We continue until either there are no more items to consider or the knapsack is full.

Monte Carlo Simulation

Part 3.

First, we apply our greedy algorithm. The low and high item-weights are known for the specific problem instance, so the expectation of the size simplifies to:

$$\mathbb{E}(w_i) = (1-\pi_i)d\_l_i + \pi_i d\_h_i \text{ for all } i \in I.$$

We use this expression for the expected weight in our heuristic algorithm to determine a selection strategy. The heuristic algorithm prescribes that we select the first six items, i.e. $x = (1,1,1,1,1,1,0,0,0,0)$. For this selection vector x, we run a Monte Carlo simulation to generate item sizes to calculate the profit for the stochastic knapsack problem. The weight of each item follows a Bernouilli distribution. Therefore, we

sample a weight from each item $i \in I$ using $V_i \sim Unif(0,1)$. If $v_i < \pi_i$, then we assign a high weight (d_h_i) to item $i \in I$. Otherwise, we assign a low weight (d_l_i). The profit for the stochastic knapsack problem for a given x and w is given by

$$\sum_{i=1}^{10} w_i r_i x_i - p \max\{0, \sum_{i=1}^{10} w_i x_i - K\}.$$

Second, to get a feeling about the performance of the heuristic, we calculate the mean of the objective value for a sample of scenarios of all the problem instances. Generating 1000 scenarios results in a mean of 16714.70 with σ of 132.31. Note that these values change with each random sample of 1000 scenarios. To get a feeling about the distribution of the profits, consider Figure 1.

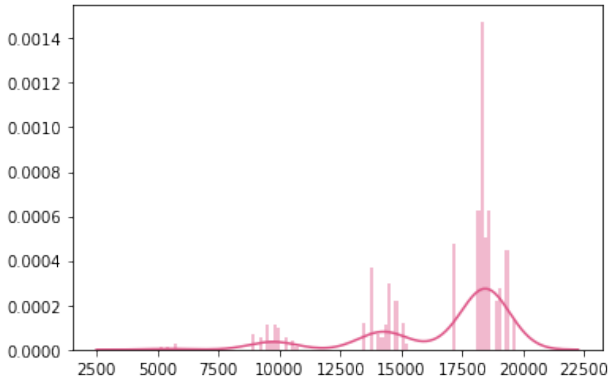

Figure 1: Distribution profits in sample $M = 1000$ when applying the heuristic

Now, we want to calculate the optimal number of runs for running our Monte Carlo simulation, to find the corresponding 95% confidence interval for the sample mean in the end. For this, we first need to get an estimate for the standard deviation of the distribution of the profits using a sample run (we use sample size equal to 1000). Note that this standard deviation is defined as

$$\left(\frac{1}{M(M-1)} \sum_{j=1}^{M} (profits_i - \overline{profits}) \right)^{\frac{1}{2}}$$

where M is the number of runs, since we take the standard deviation of the sample mean. The M in the denominator is caused by the fact that we want a confidence interval of the sample mean. The variance of the sample mean is $\frac{\sigma^2}{M}$, so that is why there is an extra M in comparison with definition of the sample variation in most literature.

Then, we select a value for α equal to 0.05, as the confidence level for our preferred half-width, defined by ε. An α of 0.05 in a two-sided situation results in $z \approx 1.96$. We would like to get the half-width to be 0.1% of our sample mean profit. With the following formula we calculate our number of runs.

$$n = \left\lceil \left(\frac{z \cdot \sigma}{\varepsilon} \right)^2 \right\rceil$$

In this formula, σ the sample standard deviation. Our preferred half-width (around 17, decided by the randomness of our initialization) can be seen to correspond to a number of runs around 240 (again, dependent of random outcomes). In Figure 2 below you can see this correspondence.

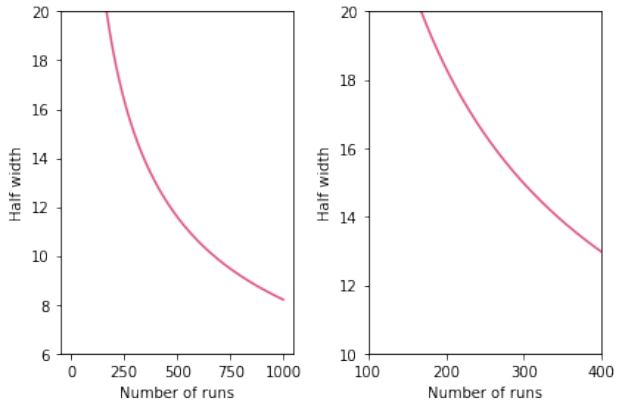

Figure 2: Computed halfwidth per number of runs

This means we know our number of runs. To make sure that we have enough runs, we round the number of runs up to the nearest hundred, thus number of runs is 300 here. Then, the corresponding 95% confidence interval for the sample mean is calculated and stated below:

$$\text{Confidence interval: } [16702.54, 16740.76]$$

Stochastic Programming Models

Part 4.

We consider the Expected Value (EV) model and Conditional Value at Risk (CVaR) model, which have an objective of maximizing expected profit and maximizing CVaR, respectively. We notice that $\sum_{i=1}^{10} w_i$ can take on 2^{10} possible values due to the fact that for each w_i there are two possibilities, that is, low (d_l_i) or high (d_h_i). Let U denote the set of scenarios and let w_{iu} denote the weight of item $i \in I$ in scenario $u \in U$. Moreover, we denote the probability that scenario $u \in U$ occurs with P_u. For example, the probability that scenario $\{d\_h_1, d\_l_2, d\_h_3, d\_h_4, d\_h_5, d\_h_6, d\_h_7, d\_h_8, d\_h_9, d\_h_{10}\}$ occurs is given by $\pi_1(1-\pi_2)\pi_3\pi_4\pi_5\pi_6\pi_7\pi_8\pi_9\pi_{10}$. Finally, let $\varepsilon_u = \max\{0, \sum_{i=1}^{10} w_{iu}x_i - K\}$ denote the excess weight in scenario $u \in U$. Consequently, we have the following MILP for the expected value model:

Expected Value Model

$$\max_{x\in\{0,1\}^n} \sum_{u\in U} P_u \left(\sum_{i=1}^{10} r_i w_{iu} x_i - p\varepsilon_u \right) \tag{1}$$

We simplify this (for our programming) by additionally letting ε_u be a decision variable and by adding the

following two constraints: $\varepsilon_u \geq 0$ and $\varepsilon_u \geq \sum_{i=1}^{10} w_{iu}x_i - K$. This is equivalent to $\varepsilon_u = \max\{0, \sum_{i=1}^{10} w_{iu}x_i - K\}$ in (1). Hence, (1) becomes:

$$\max_{x,\varepsilon_u} \sum_{u\in U} P_u \left(\sum_{i=1}^{10} r_i w_{iu} x_i - p\varepsilon_u \right)$$

subject to

$$\begin{aligned} & x_i \in \{0,1\} && \text{for all } i \in I \\ & \varepsilon_u \geq 0 && \text{for all } u \in U \\ & \varepsilon_u \geq \sum_{i=1}^{10} w_{iu}x_i - K && \text{for all } u \in U. \end{aligned}$$

Conditional Value at Risk Model

Let $\alpha \in (0,1)$ and let $\beta \in [0,1]$. The Conditional Value at Risk model is stated below, we set $\beta = 1$ in order to obtain a model that maximizes CVaR. Note that $\beta = 0$ gives the EV model.

$$\max_{x,\varepsilon_u,\eta,S_u} (1-\beta) \left[\sum_{u\in U} P_u \left(\sum_{i=1}^{10} r_i w_{iu} x_i - p\varepsilon_u \right) \right] + \beta \left[\eta - \frac{1}{1-\alpha} \sum_{u\in U} P_u S_u \right]$$

subject to

$$\begin{aligned} & x_i \in \{0,1\} && \text{for all } i \in I \\ & \varepsilon_u \geq 0 && \text{for all } u \in U \\ & \varepsilon_u \geq \sum_{i=1}^{10} w_{iu}x_i - K && \text{for all } u \in U \\ & S_u \geq 0 && \text{for all } u \in U \\ & S_u \geq \eta - \left(\sum_{i=1}^{10} r_i w_{iu} x_i - p\varepsilon_u \right) && \text{for all } u \in U. \end{aligned}$$

The CVaR model is defined as the expected value of of the profit smaller than the (1-α)-quantile of the profit distribution. In contrast to the expected value model, we add two more decision variables: η and S_u. You can see η as the (non pre-fixed) threshold for the profit to be smaller than η in the (1-α)-quantile. S_u is the shortage in scenario u.

Part 5.

We implement the mathematical models in Part 4. in Python and use Gurobi to solve them. There are two models (EV and CVaR) and 10 instances which means that we obtain 20 solutions in total. The maximum objective values are given in Table 2.

Table 2: Maximum objective value for EV and CVaR models.

	Instance j									
	1	2	3	4	5	6	7	8	9	10
EV	17013.27	16938.96	16985.46	16972.53	16968.32	16973.39	16993.50	16970.52	16996.23	16938.09
CVaR	13880.20	13737.98	13648.55	13813.54	13754.35	13706.80	13900.15	13708.58	13884.30	13769.59

The selection strategy following from the EV model is the same for each instance. In particular, it is equal to $x = (1, 1, 1, 1, 1, 1, 1, 0, 0, 0)$, which means that we select the first seven items. For the CVaR model there are four variations in the item selection. For instances 1, 4 and 9 it is equal to $x = (1, 0, 0, 0, 1, 1, 1, 1, 1, 1)$, so we take the first item and the last six items. For instances 2, 3, 5, 7, and 8 it is equal to $x = (0, 0, 0, 0, 1, 1, 1, 1, 1, 1)$, so we take the last six items. For instance 6 we have $x = (0, 1, 1, 0, 0, 1, 1, 1, 1, 1)$ and for instance 10 we have $x = (0, 1, 0, 1, 0, 1, 1, 1, 1, 1)$. Lastly, the optimal values of η for each instance in the CVaR model are given in Table 3.

Table 3: The optimal value of η in the CVaR model for each instance.

	Instance j									
	1	2	3	4	5	6	7	8	9	10
η	14375	14920	14928	14365	15019	14281	15144	15046	14431	14330

Part 6.

To analyze the effects of changes in α, p, and k we create for loops that change their values with each iteration. We run the for loop 10 times for each variable and plot the results.

For α, we decrease the value by 0.05 during each iteration. So we start with a value of 0.95 and end at the value 0. We will use 0.05 because it is small enough to not make a large change with any one iteration, but over 20 iterations it is large enough that you can easily see any trending changes in CVaR. Doing this shows that as α decreases, the value of CVaR increases, which can be seen in Figure 3. This is what we intuitively expect, as with an α of 0.95 we look at a CVaR in the worst 5% of the cases. When we have an α of 0.50 we look at a CVaR in the worst 50% of the cases. The CVaR we calculate with an α of 0.95 is 13,880.20, while the CVaR we calculate with an α of 0.50 is 15,809.98. Which is an increase of 1,929.78. Also, we see that as α approaches 0, it converges to the EV model, and at $\alpha = 0$, we get the equivalent of the EV model, which can be seen in figure 3, where the profit intersects with the EV. Changing the value of α does change the item selection strategy. As the value of α decreases, the selection strategy moves to the EV model's selection strategy, and so the first 7 items are chosen, such that at $\alpha = 0$ $x = (1, 1, 1, 1, 1, 1, 1, 0, 0, 0)$. At $\alpha = 0.95$ $x = (1, 0, 0, 0, 1, 1, 1, 1, 1, 1)$.

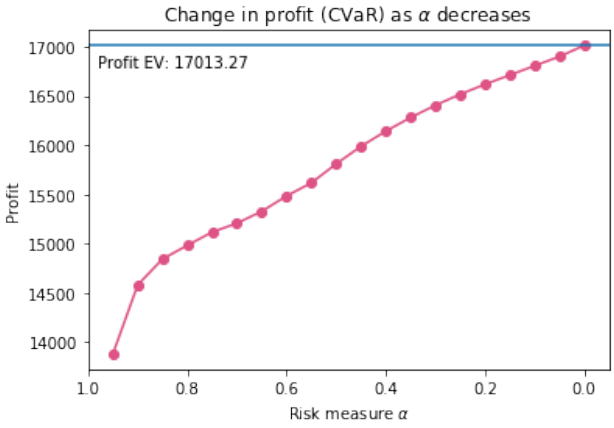

Figure 3: Change in profit (CVaR) as α decreases.

For penalty p, we want to see what happens to CVaR as we lower or increase its value, this will either make it more or less attractive to exceed the knapsack capacity. We change p by 5 for each iteration, as this is small enough that we do not have any large changes in any given iteration, but large enough that over 20 iterations we can see any trends that emerge. We find that CVaR increases as p decreases and that CVaR decreases as p increases, which can be seen in Figure 4. This is what we intuitively expect, it is cheaper to exceed the knapsack as the penalty decreases, while it is more expensive to exceed the knapsack as the penalty increases. To be slightly more specific, it is a declining increase. The revenues of the items vary from 50 to 41. Therefore, p values from 60 to 50, are still higher than the revenue, but excess is less penalized. When the penalty per unit is lower than the revenue per unit, it always pays to add this item to the knapsack. So when penalty is below 51, there always exist an item that is profitable to add despite the knapsack being full. This means that whenever the penalty is below 51, the selection strategy for the knapsack includes all 10 items, even though this exceeds the knapsacks capacity, as this is the most profitable strategy. Note that the lower p the higher the resulting revenue $r_i - p$ and therefore, the profit continues increasing. When p=60, which is our penalty, we have a selection strategy of $x = (1, 0, 0, 0, 1, 1, 1, 1, 1, 1)$. But when we lower the value of p to 55 we get a selection strategy of $x = (0, 0, 1, 0, 1, 1, 1, 1, 1, 1)$. When p is below 51, the selection strategy is $x = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$, as this maximizes profit.

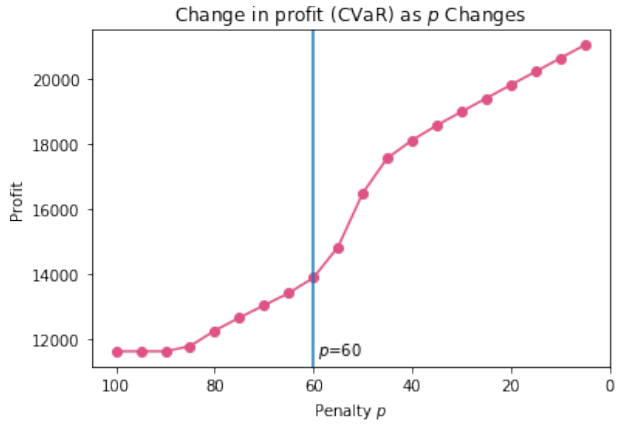

Figure 4: Change in profit (CVaR) as p changes.

For K, we want to see what happens to CVaR as knapsack capacity increases and decreases, results can be found in Figure 5 below. The original knapsack size is indicated by the blue line. We increase K by 10 for each iteration, as this will show us any emerging trends that occur from the increase in K. Intuitively we expect CVaR to strictly increase when the capacity increases and strictly decrease when the capacity decreases. This is because if the capacity of the knapsack increases, then the number of things you can fit inside of it increases, and vice versa. This is exactly what we see in the figure. Even if the increase in knapsack size is not large enough to add an extra item, CVaR still increases because there is less penalty for the excess weight. This is why while the selection strategy may not change with every iteration, the CVaR always does. For example, the selection strategy for $k = 408$, which is our knapsack size, is $x = (1, 0, 0, 0, 1, 1, 1, 1, 1, 1)$. But when we make $k = 458$, the selection strategy becomes $x = (1, 0, 1, 0, 1, 1, 1, 1, 1, 1)$

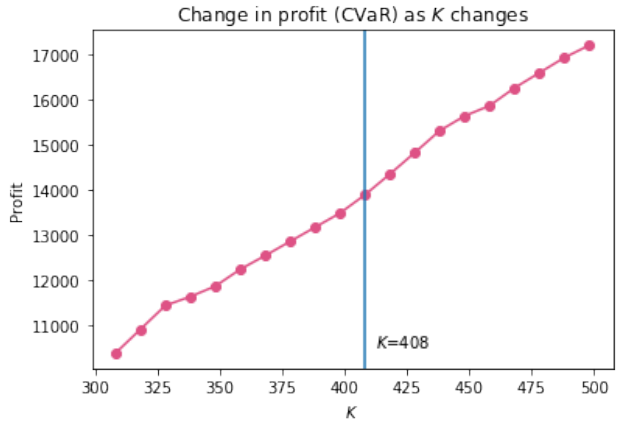

Figure 5: Change in profit (CVaR) as K changes.

Sample Average Approximation

Part 7.

In this part we describe and implement a Sample Average Approximation (SAA) algorithm to solve the EV and CVaR problems. Ultimately, we apply the SAA scheme to solve the stochastic knapsack problem for the first instance ($j = 1$) for both models.

a. The SAA model is an approximation of the true model. The idea is to reduce the scenario set to a manageable size by using Monte Carlo simulation. Thus, we generate a random sample $W^1, W^2, \ldots, W^N$ of size N of the weights, where W_i^u denotes the sampled weight of item $i \in I$ in scenario $u \in U$. This sample is independent identically distributed. Given a sample of scenarios U of size N, the mathematical programming model formulation for SAA is as follows

$$\max_{x,\varepsilon_u,\eta,S_u} (1-\beta)\left[\frac{1}{N}\sum_{u=1}^{N}\left(\sum_{i=1}^{10} r_i W_i^u x_i - p\varepsilon_u\right)\right] + \beta\left[\eta - \frac{1}{1-\alpha}\frac{1}{N}\sum_{u=1}^{N} S_u\right] \qquad \text{(SAA)}$$

subject to

$$\begin{aligned}
& x_i \in \{0,1\} && \text{for all } i \in I \\
& \varepsilon_u \geq 0 && \text{for all } u \in U \\
& \varepsilon_u \geq \sum_{i=1}^{10} W_i^u x_i - K && \text{for all } u \in U \\
& S_u \geq 0 && \text{for all } u \in U \\
& S_u \geq \eta - \left(\sum_{i=1}^{10} r_i W_i^u x_i - p\varepsilon_u\right) && \text{for all } u \in U.
\end{aligned}$$

The generation process of the weights is done in the same way as for Monte Carlo in Part 3. For each scenario we generate $V_i \sim Unif(0,1)$ for all $i \in I$. Consequently, if $V_i \leq \pi_i$, then $W_i^u = d\_h_i$, and $W_i^u = d\_l_i$ otherwise. In other words, if the random uniform number is smaller than π_i the weight of item $i \in I$ is set equal to the high weight of d_h_i. Due to the fact that with 10 items both the EV and CVaR model can be solved exactly efficiently, it is unnecessary to do Sample Average Approximation. Nonetheless, in order to compare the outcome provided by the SAA scheme with the heuristic and exact solution, we take $N < 1024$ (exact number of scenarios). In particular, we take $N = 1000$ to also account for a reduction in the variance, but smaller than the total amount of scenarios.

Moreover, in Shapiro and Philpott (2007)[1] the following four conditions are given that ensures that the SAA method solves the true problem with a reasonable accuracy:

(i) It is possible to generate a sample of realizations of the random vector ξ (here: weights),

(ii) For moderate values of the sample size it is possible to efficiently solve the obtained SAA problem,

[1] Shapiro, Alexander & Philpott, A.. (2007). A tutorial on stochastic programming.

(iii) The true problem has relatively complete recourse,

(iv) Variability of the second-stage function is not "too large".

We argue that in our case all four conditions are met. First, we can generate a sample of realizations because we can sample from a Bernoulli distribution using the uniform distribution. Second, because the chosen sample size N is not too far away from 1024, we know that we can solve the SAA problem efficiently. Third, the second-stage problem is always feasible because it, in fact, is merely the excess weight given an x in the EV model. Likewise for the CVaR model. Finally, there should not be too much variability in the excess weight due to the fact that $p > r_i$ for all $i \in I$. In other words, the unit excess weight penalty is larger than the per unit revenue.

Lastly, we solve the mathematical program (SAA) M times with a given sample of size N, which gives optimal values that we denote by $\hat{v}_N^1, \hat{v}_N^2, \ldots, \hat{v}_N^M$. The SAA solution is then the one that corresponds to the maximum value of $\hat{v}_N^1, \hat{v}_N^2, \ldots, \hat{v}_N^M$. This is the solution that we will evaluate as discussed in b.

b. By solving the SAA model M times given a sample of size N we obtain a candidate solution, say $\hat{x}$, and a corresponding objective value $g(\hat{x})$. Note that in the CVaR model we also have the optimal value of η, which we define to be equal to $\hat{\eta}$. Correspondingly, we have $g(\hat{x}, \hat{\eta})$, which we denote by $g(\hat{x})$ for brevity. We evaluate the quality of this solution by comparing it to the true optimal solution. To this end, we define the optimality gap:

$$\text{gap}(\hat{x}) \coloneqq v^* - g(\hat{x}),$$

where v^* is the true maximum value. Note that $v^* \geq g(\hat{x})$. First, we determine an upper bound for v^* using the fact that $v^* \geq \mathbb{E}(\hat{v}_N)$, i.e. $\mathbb{E}(\hat{v}_N)$ is an upper bound for v^*. In order to estimate $\mathbb{E}(\hat{v}_N)$, we solve the SAA problem M times given a sample of size N and then average the optimal values. It suffices in this assignment to take a small value of M, so we take $M = 10$. Denote by $\hat{v}_N^1, \hat{v}_N^2, \ldots, \hat{v}_N^M$ the optimal values. Consequently, the sample mean is equal to

$$\bar{v}_{N,M} \coloneqq \frac{1}{M} \sum_{j=1}^{M} \hat{v}_N^j.$$

Furthermore, because the samples are independent we know that $\hat{v}_N^1, \hat{v}_N^2, \ldots, \hat{v}_N^M$ are independent and hence we estimate the variance of the sample mean by

$$\hat{\sigma}_{N,M}^2 \coloneqq \frac{1}{M(M-1)} \sum_{j=1}^{M} (\hat{v}_N^j - \bar{v}_{N,M})^2.$$

Then, an approximate $100(1-\gamma)\%$ upper bound for $\mathbb{E}(\hat{v}_N)$ is given by

$$U_{N,M} \coloneqq \bar{v}_{N,M} + t_{\gamma,\nu} \hat{\sigma}_{N,M},$$

where $\nu = M - 1$ and $t_{\gamma,\nu}$ is the γ-critical value of the t-distribution with ν degrees of freedom. This bound is also a valid statistical upper bound for v^* because $v^* \geq \mathbb{E}(\hat{v}_N)$.

Second, we determine a lower bound for $g(\hat{x})$. The true value of $g(\hat{x})$ can be estimated using Monte Carlo simulation again. Hence, we generate an independent identically distributed random sample $W^1, W^2, \ldots, W^{N'}$ of size N' and estimate $g(\hat{x})$ by the corresponding sample average. Let

$$Q(\hat{x}, W^u) = (1-\beta)\left(\sum_{i=1}^{10} r_i W_i^u \hat{x}_i - p\varepsilon_u\right) + \beta\left[\hat{\eta} - \frac{1}{1-\alpha} S_u\right], \tag{2}$$

where $\varepsilon_u = \max\{0, \sum_{i=1}^{10} W_i^u \hat{x}_i - K\}$ and $S_u = \max\{0, \hat{\eta} - (\sum_{i=1}^{10} r_i W_i^u \hat{x}_i - p\varepsilon_u)\}$ for all $u \in U$. Then, the sample average is given by

$$\hat{g}_{N'}(\hat{x}) = \frac{1}{N'} \sum_{u=1}^{N'} Q(\hat{x}, W^u),$$

and the sample variance of $\hat{g}_{N'}(\hat{x})$ is given by

$$\hat{\sigma}^2_{N'}(\hat{x}) = \frac{1}{N'(N'-1)} \sum_{u=1}^{N'} \left[Q(\hat{x}, W^u) - \hat{g}_{N'}(\hat{x})\right]^2.$$

Here we can use a relatively large sample size N' because we only need to solve the second-stage problems which are trivial problems. Therefore, we take $N' = 10000$. Recall that in the EV model we set $\beta = 0$ and in the CVaR model we set $\beta = 1$. Moreover, in the CVaR model we obtain a solution $(\hat{x}, \hat{\eta})$ that we evaluate. Finally, an approximate $100(1-\gamma)\%$ lower bound for $g(\hat{x})$ is given by

$$L_{N'} \coloneqq \hat{g}_{N'}(\hat{x}) - z_\gamma \hat{\sigma}_{N'}(\hat{x}),$$

where z_γ is the γ-critical value of the standard normal distribution. Finally, we set $\gamma = 5\%$ and obtain a 90% confidence bound on the true $\text{gap}(\hat{x})$ given by

$$\widehat{\text{gap}}(\hat{x}) \coloneqq U_{N,M} - L_{N'}.$$

The SAA scheme for the first instance

We apply the SAA approach for the first instance for both the EV and CVaR model. The selection strategies are give in Table 4. The objective values ($g(\hat{x})$) are equal to 17109.18 and 14456.86 for the EV and CVaR models, respectively. Moreover, η is equal to 15370 in the CVaR model.

	EV Model									
$\hat{x}_{EV}$	1	1	1	1	1	1	1	0	0	0
	CVaR Model									
$\hat{x}_{CVaR}$	0	0	0	0	1	1	1	1	1	1

Table 4: The SAA solutions for both the EV and CVaR models.

Now, let us evaluate the quality of these two candidate solutions. First, we determine a 95% upper bound for v^*. To this end, we need the sample mean and the sample standard deviation of the sample mean. For the EV model we get that $\bar{v}_{N,M} = 17015.98$ and $\hat{\sigma}_{N,M} = 17.64$. For the CVaR model we get that $\bar{v}_{N,M} = 14105.19$ and $\hat{\sigma}_{N,M} = 62.96$. Consequently, an approximate 95% upper bound for v^* is equal to 17048.32 for the EV model and 14220.60 for the CVaR model.

Second, we determine a 95% lower bound for $g(\hat{x})$. For the EV model we get that $\hat{g}_{N'}(\hat{x}_{EV}) = 16998.79$ and $\hat{\sigma}_{N'}(\hat{x}_{EV}) = 19.26$. For the CVaR model we get that $\hat{g}_{N'}(\hat{x}_{CVaR}) = 13651.72$ and $\hat{\sigma}_{N'}(\hat{x}_{CVaR}) = 111.20$. Hence, a 95% lower bound for $g(\hat{x})$ is equal to 16967.11 for the EV model and 13468.82 for the CVaR model.

Finally, a 90% confidence bound on the true gap can be determined. For the EV model it is equal to 81.21 and for the CVaR model it is equal to 751.78.

Analysis

Part 8.

In this part we will compare the solutions of the different methods for the first instance. The (maximum) results of the different methods are given in Table 5. The value for the heuristic is the average profit over 1000 runs for the first instance, where we pick the first six items. This makes for a fair comparison with the objective value following from the EV and SAA scheme.

Table 5: Objective values (EV and CVaR) for each solution approach

	Exact	SAA	Heuristic
EV	17013.27	17109.18	16714.70
CVaR	13880.20	14456.86	-

First, we see that the objective values following from all three solution methods are quite similar. In particular, the objective values following from the exact and SAA method are alike. The objective value following from the heuristic approach is lower. There is a specific reason for this. In both the exact and SAA solution we select the first seven items, i.e. $x = (1, 1, 1, 1, 1, 1, 1, 0, 0, 0)$, whereas in the heuristic solution we pick the first six items, i.e. $x = (1, 1, 1, 1, 1, 1, 0, 0, 0, 0)$. The reason that we only select six items in the greedy algorithm is that it only keeps adding items as long as they fit, where we look at the expected weight of an item specifically. It does not check whether adding another time and exceeding the knapsack capacity produces a better result. As a result of only selecting the first six items, the objective value following from the heuristic approach is lower. Moreover, the difference in the objective value of the exact and SAA approach differ due to the fact that in the SAA approach we optimize over a random sample of 1000 scenarios. In other words, the SAA is an approximation of the true model, and in this case the approximation is relatively good given that we optimize over a random sample of 1000 scenarios. The heuristic approach also gives an approximation, but it is not as good as the approximation from the SAA scheme.

Second, for the CVaR model we see a bigger difference between the optimal solution following from the exact and SAA approaches. This is due to the fact that in both cases we select different items. The exact solution gives $x = (1, 0, 0, 0, 1, 1, 1, 1, 1, 1)$ and SAA gives $x = (0, 0, 0, 0, 1, 1, 1, 1, 1, 1)$. This indicates that the SAA approximation for the CVaR model is not as good as for the EV model. As we saw in Part 7., there is

a larger 90% confidence bound on true gap for the SAA candidate solution in the CVaR model compared to the EV model. In fact, the confidence bound on the true gap in the CVaR model is 751.78. Furthermore, the given corresponding standard deviations are larger. In order to improve the approximation, we can optimize over a large sample, e.g. a random sample of 2000 scenarios. However, this is superfluous as we already know that we can solve it exactly for the case with 1024 scenarios.

Part 9.

In this part, we discuss advantages and disadvantages of the used approaches for varying problem sizes and random distributions.

First, we will cover the advantages and disadvantages of certain solution approaches for larger instances, such as ($N > 20$). When there are 20 items, for example, then what we get is $2^{20} = 1,048,576$ different scenarios. While it is possible to get an exact solution, it has a very large computational cost and is thus not practical. Increasing the instances to 25 increases the number of scenarios to 33,554,432. Thus, we see that there is an exponential increase in computation time as we increase the number of instances. In case of larger instances (e.g. $N > 20$), the heuristic algorithm and the SAA method make more sense. The heuristic algorithm as it is faster to compute than the SAA solution, though it is not as precise. In fact, it is most appropriate to use the SAA method when the number of scenarios becomes too large because the optimization model can be solved efficiently when we take a sample of scenarios.

Next, we will cover the advantages and disadvantages of these solution approaches if the item weights follow different distributions, such as Normal, Poisson, etc.
The exact solution approach is particularly useful when using discrete distributions for the weights, provided that the number of possible scenarios is not too large. The reason for this is that the expected profit can be written as a weighted sum of the profit in each scenario. In case the discrete distribution is unbounded, i.e. it can take on an infinite number of values, which is the case for the Poisson, we can truncate it and obtain a finite sum. When dealing with discrete distributions, the heuristic algorithm and SAA approach are not useful unless the number of scenarios becomes too large. Both are an approximation, so if we can derive an explicit expression for the expected profit, the exact solution approach is always the one to use.

In contrast to discrete distributions, deriving an explicit expression for the expected profit becomes much more complicated when weights follow a continuous distribution. Moreover, as a result it becomes more complicated to derive an exact solution. Nonetheless, Merzifonluoglu and Geunes (2012) give an exact algorithm for the stochastic knapsack problem when items follow normally distributed item sizes. Irrespective of this, the heuristic algorithm and the SAA approach are more useful when dealing with continuous distributions, provided that we can generate a random sample of scenarios.

Bonus: Antithetic variates

In the SAA scheme we solve the mathematical program (SAA) M times given a sample of size N. We can use antithetic variates to reduce the variance between replications. When we generate random weights $W^1, W^2, \ldots, W^N$ in one replication, we use $(0, 1)$ uniform numbers $V^1, V^2, \ldots, V^N$. In the next replication we can recycle these numbers and use $1-V^1, 1-V^2, \ldots, 1-V^N$ to generate random weights $W^1, W^2, \ldots, W^N$. More specifically, we do the following.

- Generate a random sample $W^1, \ldots, W^N$ using $(0,1)$ uniform random numbers $V^1, \ldots, V^N$ and solve the SAA model to obtain $\hat{v}_N^{j,1}$
- Generate another random sample $W^1, \ldots, W^N$ using $(0,1)$ uniform random numbers $1-V^1, \ldots, 1-V^N$ and solve the SAA model to obtain $\hat{v}_N^{j,2}$
- Let $\hat{v}_N^j = (\hat{v}_N^{j,1} + \hat{v}_N^{j,2})/2$
- Repeat

Hence, we get M pairs $(\hat{v}_N^{1,1}, \hat{v}_N^{1,2}), (\hat{v}_N^{2,1}, \hat{v}_N^{2,2}), \ldots, (\hat{v}_N^{M,1}, \hat{v}_N^{M,2})$. Likewise Part 7, the SAA solution $\hat{x}$ is the one that corresponds to the maximum value of all $M \times 2$ obtained solutions. As a result of using antithetic variates, the sample variance of $\bar{v}_{N,M}$, given by

$$\hat{\sigma}^2_{N,M} \coloneqq \frac{1}{M(M-1)} \sum_{j=1}^{M} (\hat{v}_N^j - \bar{v}_{N,M})^2,$$

should be smaller.

Furthermore, in a similar fashion we can use antithetic variates to reduce the sample variance of $\hat{g}_{N'}(\hat{x})$:

- Generate a random sample $W^1, \ldots, W^{N'}$ using $(0,1)$ uniform random numbers $V^1, \ldots, V^{N'}$ to obtain $Q_1(\hat{x}, W^1), \ldots, Q_1(\hat{x}, W^{N'})$ (see equation (2))
- Generate another random sample $W^1, \ldots, W^{N'}$ using $(0,1)$ uniform random numbers $1 - V^1, \ldots, 1 - V^{N'}$ to obtain $Q_2(\hat{x}, W^1), \ldots, Q_2(\hat{x}, W^{N'})$ (see equation (2))
- Let $Q(\hat{x}, W^u) = (Q_1(\hat{x}, W^u) + Q_2(\hat{x}, W^u))/2$ for all $u = 1, \ldots, N'$.

Consequently, the sample variance of $\hat{g}_{N'}(\hat{x})$, given by,

$$\hat{\sigma}^2_{N'}(\hat{x}) = \frac{1}{N'(N'-1)} \sum_{u=1}^{N'} \left[Q(\hat{x}, W^u) - \hat{g}_{N'}(\hat{x}) \right]^2.$$

should be smaller.

To analyze the effect of the use of antithetic variates with respect to the result in Part 7, we set $M = 5$ and $N' = 5000$ due to the fact that we have $M \times 2$ and $N' \times 2$ pairs. We start with the EV model. In Table 6 one can see that using antithetic variates in this particular EV model does give a reduction in standard deviation in one case. For the other case it is slightly worse. Note that the standard deviations are already small to begin with. Therefore, we cannot expect to get a large reduction as there is always randomness involved. Nonetheless, the 90% confidence bound on the true gap has been reduced to 27.73 when using antithetic variates. Without using antithetic variates it was equal to 81.21.

Table 6: Change in standard deviation as a result of applying antithetic variates (AV) in the EV model.

	Without AV	With AV
$\hat{\sigma}^2_{N,M}$	17.64	9.26
$\hat{\sigma}^2_{N'}(\hat{x})$	19.26	19.74

Second, the CVaR model. In contrast to the EV model, Table 7 shows that in both cases the use of antithetic variates reduces the standard deviation. In particular, in the first case the standard deviation is more than halved. Lastly, the 90% confidence bound on the true gap has been reduced to 454.99 in the antithetic variates case in comparison to 751.78 in the case without using antithetic variates.

Table 7: Change in standard deviation as a result of applying antithetic variates (AV) in the CVaR model.

	Without AV	With AV
$\hat{\sigma}^2_{N,M}$	62.96	28.52
$\hat{\sigma}^2_{N'}(\hat{x})$	111.20	100.81

Appendix

Table

Table 8: Item weights per problem instance (generated with a random seed equal to the instance number j)

		Item i									
Instance j	Weight	1	2	3	4	5	6	7	8	9	10
1	d_h_i	99.53	100.73	98.37	106.48	91.52	92.46	90.68	95.67	90.12	91.66
	d_l_i	2	1	4	4	1	7	5	3	8	4
2	d_h_i	108.10	100.34	101.15	95.94	99.08	100.25	86.91	96.93	101.51	94.90
	d_l_i	1	0	1	0	2	3	2	3	7	3
3	d_h_i	99.80	103.58	99.83	93.70	99.26	103.34	95.00	99.30	91.27	92.70
	d_l_i	1	1	2	2	2	4	3	2	3	5
4	d_h_i	98.34	97.64	95.57	102.72	97.90	93.53	92.46	93.54	89.41	88.35
	d_l_i	2	3	1	1	4	0	3	5	8	5
5	d_h_i	102.33	103.18	94.98	96.67	97.03	98.08	92.12	94.56	94.34	97.43
	d_l_i	4	0	1	1	4	1	1	6	7	6
6	d_h_i	101.81	90.57	99.16	99.51	105.30	93.20	98.72	98.91	99.07	82.67
	d_l_i	0	1	4	1	2	2	5	3	4	3
7	d_h_i	104.23	103.14	96.53	94.16	96.71	94.18	97.72	90.15	91.14	96.42
	d_l_i	1	1	0	2	4	7	7	5	7	5
8	d_h_i	100.52	98.40	100.72	96.61	96.58	98.51	94.09	99.67	89.01	94.80
	d_l_i	0	1	4	1	3	3	2	4	6	6
9	d_h_i	100.54	103.52	107.49	98.88	89.79	95.45	91.13	95.36	89.56	87.49
	d_l_i	4	0	2	3	5	1	3	5	6	5
10	d_h_i	105.91	96.12	101.10	100.30	103.13	88.81	96.60	95.49	92.61	85.26
	d_l_i	0	1	0	1	2	3	2	9	8	8