M. Buchem, N. Megow, A. Lindermayr, Z. Liu **Sommer 2024**

Hands-on Tutorial on Optimization

Exercise Sheet: Cheese Coup (25.09.2024)

Exercise 1

It is the year 1903. The Swiss Baron Chaese von Due, a respected millionaire and cheese connoisseur spends his nights going about his secret second profession: stealing cheese. In private, von Due is proud to claim that he is not only the world's most adroit but also most dreaded thief in this particular field.

His professional ambition is particularly attracted by the old earl M. N. Taler, who is well-known both for his greed and his enormous collection of rare cheeses. Instead of savouring every mouthful of his delicious cheeses, however, Taler just stores them in his gigantic cellar and uses every occasion to brag about his valuable collection. To Baron von Due, this despicable abuse of delicious cheese cannot go unpunished! Help him to relieve M. N. Taler of his collection of cheeses.

(a) Von Due has bribed a servant of Taler to obtain information about the secret entrance and the contents of one of Talers cellars. The compartment is said to contain the following:

type	pieces	weight	value
Brie	1	15kg	1800 Gulden
Camembert	1	9kg	1200 Gulden
Reblochon	1	10kg	1300 Gulden

Von Due cannot carry more than 30kg in his bag. He wants to steal cheese with the highest total value possible and he does not mind cutting of some fraction of a cheese to be able to fit it into his bag.

Help von Due by modeling his problem as an LP and solve it using Python and Pulp. Keep in mind that von Due has great plans beyond this particular cellar. Thus, you should build your model flexible enough to be able to deal with other data. Your Python function that implements and solves the model should take the following arguments and be able to solve all (feasible) inputs:

- A list `cheese` containing the names of the cheeses in Von Due's cellar.
- Dictionaries `pieces`, `weight` and `value` that map the different cheeses to the available number of pieces, their weight and their piece value, respectively.

(b) Shortly after arriving in the cellar, von Due notices a hidden passage that leads to a second compartment containing a number of exquisite pieces of Handkäse, as well as a cake of Roquefort:

type	pieces	weight	value
Handkäse	30	.4kg/piece	50 Gulden/piece
Roquefort	1	10kg	2200 Gulden

Both the Handkäse and the Roquefort however, valuable as they are, are a very dangerous haul: If cut, they exude a very discernible smell that would make it easy for M. N. Taler's guards to track down von Due. Therefore, von Due can only pack entire pieces of these two types of cheese.

Extend your model from (a) by making sure that certain types of cheese cannot be cut in the optimal solution. Your Python function should now take an additional argument `cuttable` containing the names of all cheeses that can be cut. All cheeses that are contained in `cheese` but not in `cuttable` can *not* be cut.

(c) Von Due's final coup will be to infiltrate M. N. Taler's famous cheese museum. The museum is heavily guarded at night, so von Due has to make his entry during regular opening hours. To get through security checks, he plans to fill k small pouches with cheese and drop them into the museum's litter bins. After they have been emptied at night, he will retrieve them from the rubbish in the museum's back yard.

In order for the pouches to slip through, they cannot exceed a certain weight. The total weight of all pouches, on the other hand, is no longer relevant as von Due can use a carriage to take home the loot.

Adapt your model such that several small pouches can be filled. The corresponding template file already contains all necessary information.

(d) Von Due is satisfied with the total value of stolen cheese in your solution. However, he complains that some pouches are relatively heavy, whereas others are barely filled at all. A more even distribution of weight would greatly increase the secrecy of his coups. He is even willing to accept a small reduction in the value of the loot in exchange.

He explains to you the following measure of unevenness of a solution:

$$\text{unevenness} := \sum_{i=1}^{k} \left| \text{weight } i\text{-th pouch} - \frac{\sum_{j=1}^{k} \text{weight } j\text{-th pouch}}{k} \right|$$

Modify your model in order to compute, among all solutions that that achieve at least 90% of the maximal attainable value and that minimize the above measure of unevenness, the one that maximizes the total value of stolen cheese.

Hint: You do not have to solve this within one model. You should first maximize the total value of all pouches, then add the constraint

$$\text{total value} \geq 0.9 \cdot \text{maximal attainable value},$$

minimize the unevenness, add the constraint

$$\text{unevenness} \leq \text{minimal unevenness},$$

and finally maximize the total value of stolen cheese. Since these different models depend on each other (the output of one model is part of the input for another), you should write a Python function that iteratively solves them and passes the necessary information between them.