Mandatory Assignment 1

Applied Optimization: Location Planning

Fall 2022

Jens Trolle, 201807850
Maria Askholm Nielsen, 201804824
Nikolaj Kjær-Sørensen, 201804933

Abstract

This paper aims to introduce methods for placing a number of drone stations in a region under different criteria. This is done using Mid Jutland in Denmark as an example. A solution approach based on solving a multi-source Weber problem using Cooper's location allocation heuristic is introduced. A method based on the p-center problem where we create a partition of p disjoint sets in each of which a one-center problem is solved is also introduced. The p-center problem is solved in two cases, one where the weights of each city is considered, and one where the weights are not considered. The results from these solution approaches are presented, and it is concluded that while they provide very different solutions, they can each be used to solve different problems relatively well. The multi-source Weber- and weighted p-center models are good when the population of a city is proportional to the probability of a drone being needed in the city, but the solutions provided by Cooper's heuristic are too far away from some cities, leading to the drones not being able to live-transmit video data. In the case with $p \geq 7$, the weighted p-center model finds good feasible solutions, which are also close to the solutions found by the multi-source Weber model. The unweighted p-center method gives solutions, where drones can be deployed quickly to the furthest cities. In cases where max-time-to-city is more important than average distance travelled, this is a good approach. A hybrid method which combines the unweighted p-center model and Cooper's heuristic is also introduced and discussed. These methods are also compared with a method from Drezner et al. 1996.

Department of Mathematics
Aarhus University

Introduction

This paper presents methods for solving a decision problem based on placing drone infrastructure in Mid Jutland in Denmark in different ways.

The paper is structured as follows. In chapter 1 we introduce the problem in detail. We then begin describing our solution approaches. In chapter 2 we introduce the single-Weber problem and a way to solve it. We then expand it to the multi-source Weber problem, which we use to model and solve our decision problem, by minimizing the weighted distances of cities to drone stations.

We introduce the p-center problem in chapter 3 and go on to present weighted and an unweighted versions of the problem as well as solution procedures for these. In this chapter, a quadratically constrained mixed integer formulation of the problem is also presented.

In chapter 4 we remark on the results obtained from the the solution methods, which were presented in chapter 1. We present maps of Mid Jutland with the resulting station placements along with plots of key values. We then present a small discussion of the solution approaches and conclude by introducing a hybrid approach, where we start by solving the multi-source Weber problem and then solve the unweighted p-center problem using the previous solution as initial locations.

Then in chapter 5 we compare our models with one developed by Drezner et al. 1996. This is done by first summarizing their article and then discussing pros and cons of our their model and ours.

Finally in chapter 6 we conclude upon our findings by noting, that the solution approaches which are presented cannot be directly compared, and solve different problems.

Contents

Chapter 1

Introduction of the Problem

In this paper we attempt to find solutions to the following problem stated by the Danish Ministry of Defense. They would like to improve control and surveillance of critical infrastructure in the country by acquiring a substantial amount of surveillance drones. These drones are equipped with real-time surveillance cameras and data gathering sensors, and can be controlled online.

These drones need a number of runways and charging stations to be installed to operate. The ministry wants the drones to operate in all 5 regions of Denmark - this paper focuses on placing the runways and charging stations in Mid Jutland.

We start by finding possible locations of the drone stations in Mid Jutland using Cooper's location-allocation heuristic from Klose 2022, p. 34-35 which uses the single Weber problem to determine the location. These methods will be elaborated in chapter 2.

Having some solutions, we note that the drones have some limitations. They can only transmit real-time video within a radius of 50 kilometres and have a maximum flight time of 110 minutes. Therefore we introduce a different location scheme such that the maximal distance to be covered by a drone is smallest. This more directly accommodates the limitations of the drones. This can be done by looking at the p-center problem, which can be solved by solving p one-center problems. These methods will be elaborated in chapter 3, where the one-center problem can be seen in an unweighted and a weighted case.

1.1 Provided Data

To solve this problem, we have been provided with a dataset consisting of all municipalities in Mid Jutland along with the populations of the cities in these municipalities. Some cities have a population of 0 in the dataset, so to avoid dividing by 0 in the applied algorithms, we have set their populations to 1. This small change in the data will not affect the results in any meaningful way. These population numbers are used as weights when solving the problem of placing the drone stations.

Using geopy[1] we are able to take cities and get their coordinates on the earth. Letting l_i^x and l_i^y, for $i = 1, \dots, m$ denote the longitude and latitude of the m cities. Without loss of generality, we let $l_0^x = \min_i l_i^x$ and $l_0^y = \min_i l_i^x$ be the origin, that is the south-west most point is the origin.

1: From https://pypi.org/project/geopy/.

We can map the longitudes and latitudes to Cartesian coordinates as follows

$$x_i = \frac{R \cdot (l_i^x - l_0^x) \cdot \pi \cdot \cos\left(\frac{\pi \cdot l_0^y}{180}\right)}{180}$$
$$y_i = \frac{R \cdot (l_i^y - l_0^y) \cdot \pi}{180}.$$

This is done, such that when we calculate the Euclidean distance, we are dealing with distances in kilometers. We note that this mapping can be reversed, which we later use for plotting.

Chapter 2

A Model Based on the Multi-Source Weber Problem

In this chapter we delve into our method of placing p drone stations in Mid Jutland, such that the total distance flown by each drone is minimized. This is done using only the weighted distances. We first describe the single-Weber problem and how we can solve it. We then introduce the multi-source Weber Problem along with Cooper's allocation-location heuristic, which we use to solve the original problem of placing the p drone stations.

2.1 The single-Weber Problem

We formulate the single-Weber problem as in Klose 2022, p. 13. Let $Y_i \in \mathbb{R}^2,\ i = 1, \dots, m$ be m given points in the Euclidean plane with weights, $w_i > 0,\ i = 1, \dots, m$. In this case, we interpret the Y_i variables as the cities and w_i as the population of city i. We seek to determine a new drone station, $X \in \mathbb{R}^2$, such that the sum of weighted distances is minimized, that is, we want to minimize the following

$$f(X) = \sum_{i=1}^{m} w_i d_i(X), \tag{2.1}$$

where $d_i(X) = \|X - Y_i\|$ denotes the Euclidean distance between the station, X, and drone, Y_i. This model makes the following assumptions, when we apply it to the placement of a depot.

First, every point in $\mathbb{R}^2$ is a feasible location. It is sufficient to consider a plane, and not the sphere, that is we do not take into account the curvature of the earth. We can measure distances using the Euclidean distance between points. The size of a facility does not matter, that is, we can model it as a point. Transportation costs are proportional to the distance travelled and the weight. We do not have any fixed costs associated with the location.

As $f(X)$ is continuous and differentiable, at least for points $X \neq Y_i,\ i = 1, \dots, m$, a necessary and sufficient optimality condition is

$$\nabla f(X) = 0 \quad \iff \quad X = \sum_{i=1}^{m} \alpha_i(X) Y_i, \tag{2.2}$$

where

$$\alpha_i(X) := \frac{1}{s(X)} \cdot \frac{w_i}{d_i(X)} \quad \text{and} \quad s(X) = \sum_{k=1}^{m} \frac{w_k}{d_k(X)} \tag{2.3}$$

as described in Klose 2022, p. 14. To calculate this value of X one can use Weiszfeld's procedure, which one can build upon using Ostresh's procedure.

2.1.1 Weiszfeld and Ostresh's procedures

Weiszfeld's procedure is a fixed point method for solving eq. (2.2). The procedure works as follows. We input the cities, Y_i and the weights, w_i and the algorithm returns the optimal location, X.

Algorithm 1 Weiszfeld's Procedure

Initialize by starting with a point in $\text{conv}(Y_1, \dots, Y_m)$ or simply the centre of gravity, that is: $X^0 = \frac{1}{W} \sum_{i=1}^{m} w_i Y_i, \quad$ where $W = \sum_{i=1}^{m} w_i$.
Set $X^1 = \sum_{i=1}^{m} \alpha_i(X^0) Y_i$.
$t \leftarrow 1$
while $\|X^t - X^{t-1}\|$ **do**
    $t \leftarrow t + 1$
    $X^t = \sum_{i=1}^{m} \alpha_i(X^{t-1}) Y_i$
end while
return X

By Proposition 2.1 in Klose 2022, we get that this is a gradient procedure, where we perform the following step in each iteration

$$X^{t+1} = X^t - \frac{1}{s(X^t)} \nabla f(X^t),$$

when $X^t \neq Y_i, \ \forall i$.

To expand upon this procedure, we look to Section 2.8 in Klose 2022, where Ostresh's modification to the procedure is described. Ostresh does this by letting

$$s_k(X^t) = \sum_{i \in I_k} \frac{w_i}{d_i(X^t)} \tag{2.4}$$

and by instead taking the step

$$X^{t+1} = X^t - \frac{\lambda}{s(X^t)} \nabla f(X^t) \tag{2.5}$$

if we are not in a customer point, and

$$X^{t+1} = X^t - \frac{\lambda}{s_k(X^t)} \left(1 - \frac{w_k}{\|\nabla f_k(X^t)\|}\right) \nabla f_k(X^t) \tag{2.6}$$

if we are in a customer point and $w_k < \|\nabla f_k(X^t)\|$. We have, that this procedure converges to optimality for $\lambda \in [1, 2]$.

This leaves us with a way to solve the single-Weber problem to optimality.

2.2 The multi-source Weber Problem

We now look to the multi-source Weber Problem, a generalization of the single-Weber Problem for $p > 1$. That is, we are trying to place p drone stations in the Euclidean plane, such that the sum of weighted distances from a drone to the nearest drone station is minimized. Letting

$$d_i(X_j) = \|X_j - Y_i\| = \sqrt{\left(x_{j1} - y_{i1}\right)^2 + \left(x_{j2} - y_{i2}\right)^2} \tag{2.7}$$

denote the distance from drone i to drone station j, we can write the problem as a non-linear program

$$\min_{X_1,\dots,X_p \in \mathbb{R}^2} f(X_1, \dots, X_p), \quad \text{where } f(X_1, \dots, X_p) = \sum_{i=1}^{m} w_i \min_{j=1,\dots,p} d_i(X_j). \tag{2.8}$$

From Klose 2022, p. 34 we note, that this problem is strictly NP-hard, and as such is very difficult to solve optimally. Note in particular, that the problem is non-convex, as we are taking the minimum of convex functions, which is not convex. This leads us to looking for heuristic solution methods.

2.2.1 Cooper's location-allocation Heuristic

When solving multi-source Weber Problems, one has less trouble if either the assignment of customers to locations or if the locations of the facilities are fixed. In the first case, one can simply solve a single Weber problem for each subset of customers, which are assigned the same facility. The second case is solved by assigning customers to their nearest facility. Cooper suggests an approximate solution method by simply switching between a location and an allocation phase. The algorithm is described in Algorithm 2.

In our case, we choose the initial locations of the drone stations randomly in the convex hull of all cities. Following that, we go into step 2 of the process. We partition the set I into disjoint subsets by simply assigning each city to their nearest drone station. There is a risk, of having empty partitions, but we get around this, by relocating the relevant location, to a random point in the convex hull. Now having p disjoint subsets of the index set, we solve the single Weber problem using Ostresh's modified procedure.

Effectively, we are solving p different single Weber problems, for which we have a pretty robust solution framework, and checking if the resulting solutions are good enough.

Algorithm 2 Cooper's location-allocation algorithm

Step 1: Initialization:
Choose initial positions, $X_1^0, \dots, X_p^0 \in \mathbb{R}^2$ for each facility.
$t \leftarrow 0$.
Step 2: Allocation:
Partition the index set I into p disjoint subsets such that for all $i \in I$ and $j = 1, \dots, p$:

$$i \in I_j \implies d_i(X_j^t) = \min_{k=1,\dots,p} d_k(X_j^t).$$

Step 3: Location:
for $j = 1, \dots, p$ **do**
Solve the single Weber problem

$$X_j^{t+1} = \arg\min_{X_j \in \mathbb{R}^2} \sum_{i \in I_j} w_i d_i(X_j)$$

end for
if $\|X_j^{t+1} - X_j^t\| \leq \epsilon$, for all $j = 1, \dots p$ **then**
return X^{t+1}
end if
$t \leftarrow t + 1$
Go to Step 2.

Chapter 3

Two Models Based on the p-Center Problem

In this chapter we discuss how we solve the problem of placing p drone stations in Mid Jutland with the goal of minimising the largest distance from a city to a drone-station. This is done both with weighted distances an unweighted distances.

The problem can be stated as a quadratically constrained mixed integer program. We let Y_i denote city i and w_i is the associated weight for $i = 1, ..., m$. Furthermore we let X_j denote the location of the j'th drone station for $j = 1, ..., p$. Note that for the unweighted case, we simply use $w_i = 1$ for $i = 1, ..., m$.

$$
\begin{aligned}
\min_{X_1,...,X_p,z_{ij}} \quad & \theta, \\
\text{s.t.} \quad & w_i \cdot d_{ij} \cdot z_{ij} \leq \theta, \quad \text{for } i = 1, ..., m \text{ and } j = 1, ..., p, \\
& \|X_j - Y_i\| \leq d_{ij}, \quad \text{for } i = 1, ..., m \text{ and } j = 1, ..., p, \\
& z_{ij} \in \{0, 1\}, \quad \text{for } i = 1, ..., m \text{ and } j = 1, ..., p, \\
& \sum_{j=1}^{p} z_{ij} = 1, \quad \text{for } i = 1, ..., m.
\end{aligned}
$$

While we don't explicitly have a quadratic constraint, note that

$$
\|X_j - Y_i\| \leq d_{ij} \iff \|X_j - Y_i\|^2 \leq d_{ij}^2 \iff (X_j - Y_i)^T (X_j - Y_i) \leq d_{ij}^2,
$$

which is quadratic.
This problem is difficult to solve directly, so instead we use a heuristic to find good solutions.

The general approach to solving the p-center problem at hand, is to first partition the cities into p disjoint subsets and then consider each subset individually. Thus we only need to solve the one-center problem for each subset. After we have solved the p one-center problems and thus moved the p drone stations to new positions, we reassign the cities to the closest drone station. This is repeated until the positions of the drone-stations no longer change.

We make the initial partitioning of the cities by placing our p drone stations randomly i p different cities. We then assign each city to the closest drone station, thus obtaining a partitioning into p disjoint subsets of the cities. Alternatively, we considered using a K-means algorithm to make the initial partitioning. This would likely provide us with a good initial solution for our problem, as the drone stations would be spread out quite evenly. This would mean, that the algorithm might converge in relatively few iterations. This will however also mean, that it is very likely that we start each run of the algorithm from the same initial solution, and since the solution found be the algorithm depends on the initial solution, we get the same solution every time. By choosing the initial solution randomly, we ensure that we get different solutions by running the algorithm multiple times, and we can then look at the best solution found overall.
By placing the drone stations on p different cities, we also avoid having drone stations with no "associated" cities in our initial solution. When we say that a city is associated to a drone station, we mean that this drone station is the closest drone station to the city. Should we somehow end up in a situation where we have a drone station with no associated cities, this drone station is relocated to a random new position in the convex hull of the cities such that we keep all p drone stations at all times.

We have solved the problem considering both the weighted distances as well as the unweighted distances. Each approach will be discussed in the following two sections.

3.1 Solving the one-center problem: Unweighted distances

For solving the one-center problem in the unweighted case we iteratively find larger circles, such that in the end we find a minimum circle covering all points. The algorithm used to find the circles is based on a few important observations (Klose 2022, p. 51). These are:

OBS 1: A minimum covering circle of m points depends only on two or three points.

OBS 2: Covering two points, Y_1, Y_2, is simply achieved by placing the center of the covering circle at $X = \frac{Y_1+Y_2}{2}$ and choosing the radius $r = \|X - Y_1\| = \|X - Y_2\|$.

OBS 3: The problem of covering three points is split into to cases:

- The triangle spanned by the three points is acute. In this case, the center of the minimum covering circle is found as the intersection of the three perpendicular bisectors of the lines making the triangle. Note, that only two of the bisectors will be needed however.
- The triangle spanned by the three points has an obtuse or right angle. Naming the points A, B and C, we may assume that the angle at A is obtuse or right. In this case, only the two points B and C are required to determine the minimum covering circle.

OBS 4: Say we have a circle defined by three points. Now introduce a fourth point D not inside the current circle. Name the furthest of the three points A and draw a line from A through the center of the current circle. Now, this line will split the plane into two, where the two remaining points of the original triangle will lie on separate sides of the line. Let B be the name of the point not on the same side of the line as D, and name the last point C.

The minimum covering circle of the four points is then the minimum covering circle of the three points A, B, and D. Note that depending on the angles in the triangle given by A, B and D, we may only need two of the points.

We can use the sign of the inner product of two vectors to determine if the angle between the two vectors is acute or obtuse. Since for two vectors a, b with angle v we have

$$\cos(v) = \frac{a^T b}{||a|| \cdot ||b||}.$$

We then know that if the angle is acute, $\cos(v) > 0$, then the inner product $a^T b > 0$. Similarly we know that if an angle is obtuse we must have $\cos(v) < 0$, which implies $a^T b < 0$. If the angle we must have $\cos(v) = 0$ and $a^T b = 0$.

Based on these observations we can now summarize a procedure for solving the unweighted one-center problem in the following steps (Klose 2022, p. 52).

Step 1 (Initialization) Choose two points, $A, B \in \{Y_1, Y_2, ..., Y_m\}$, such that the distance between the points is as large as possible. Go to **Step 1**.

Step 2 (Two points) From the two points A, B, construct a minimum covering circle. This is done as described in OBS 2. If this current circle contains all m points, stop. Otherwise let C denote the point furthest from the center of the current circle and proceed to **Step 3**.

Step 3 (Three points) As noted in OBS 3, we now need to begin by checking if the triangle defined by the points A, B, C has an obtuse angle, and then follow the procedure described in OBS 3 to obtain the center of the new circle containing all three points. If the triangle has an obtuse angle, name the two points in the corners with acute angles A, B and return to **Step 2**. Otherwise, continue to **Step 2**.

Step 4 (Extend the circle) First, we check if the current circle covers all the points. If this is the case, we stop. Otherwise we choose a point D that is furthest from the center of the current circle and then follow the procedure described in OBS 4. This will give us three points that we will name A, B, C, and we will then return to **Step 3**. Note that if only two of the points are required to define the minimum covering circle, **Step 3** will send us back to **Step 2**.

This algorithm has not been independently implemented in our solution, but rather it has been treated as a special case of the weighted distances algorithm. This is done, as large parts of the code would be repeated in the weighted and unweighted case. Our implementation differs from the above description in two minor ways.
Firstly, **Step 4** above is swapped for **Step 4** and **Step 5** in section 3.2, meaning that we try all relevant combinations of points instead of using the procedure described in OBS 4.
Secondly, in OBS 3 we described how we found the center in the case of an acute triangle by finding the intersection of two of the perpendicular bisectors. We however found it easier to solve the problem more directly by solving the following 3 equations:

$$(Y_{11} - X_1)^2 + (Y_{12} - X_2)^2 = r^2 \tag{3.1}$$
$$(Y_{21} - X_1)^2 + (Y_{22} - X_2)^2 = r^2 \tag{3.2}$$
$$(Y_{31} - X_1)^2 + (Y_{32} - X_2)^2 = r^2 \tag{3.3}$$

where

$$Y_1 = \begin{pmatrix} Y_{11} \\ Y_{12} \end{pmatrix}, \quad Y_2 = \begin{pmatrix} Y_{21} \\ Y_{22} \end{pmatrix}, \quad Y_3 = \begin{pmatrix} Y_{31} \\ Y_{32} \end{pmatrix},$$

denotes the three points, $X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}$ denotes the center and finally r denotes the radius of the circle. These ensure that each of the three points have the same distance to the center. Since we are only really interested in the center and not the radius, we can reduce the system by subtracting (3.1) from (3.2) and from (3.3):

$$(Y_{21} - X_1)^2 + (Y_{22} - X_2)^2 - (Y_{11} - X_1)^2 - (Y_{12} - X_2)^2 = 0 \tag{3.4}$$
$$(Y_{31} - X_1)^2 + (Y_{32} - X_2)^2 - (Y_{11} - X_1)^2 - (Y_{12} - X_2)^2 = 0. \tag{3.5}$$

We solve these two equation with the python package sympy. This runs slower than it would if we solved the problem analytically, but since the problem isn't too large, this is not an issue.

3.2 Solving the one-center problem: Weighted distances

Just as for the unweighted case, the weighted one-center problem is solved by following a series of steps generally saying what to do when we try to cover respectively two, three or four points. However, since we now consider the weighted distances, these steps are a little different. In the weighted case, the steps can be stated as seen below (Klose 2022, p.55-56). Note that exactly how to find the center of a number of points will be elaborated on after the steps has been stated. Note also, that we let

$$d_i(X^*) = ||X^* - Y_i|| \quad \forall i.$$

That is, $d_i(X^*)$ denotes the distance from the current center X^* to a point Y_i. Also, w_i denotes the weight of point Y_i.

Step 1 (Initialisation) Select two points Y_1, Y_2 randomly. Determine the center X^* of these points, set $n = 2$ and let $\underline{R}(X^*) = w_1 \cdot d_1(X^*) = w_2 \cdot d_2(X^*)$.

Step 2 (Uncovered third point) Let Y_3 denote the point with the largest weighted distance to the center X^* and let $R(X^*)$ denote this weighted distance. That is, $R(X^*) = w_3 \cdot d_3(X^*)$. If $R(X^*) = \underline{R}(X^*)$, stop the algorithm with optimal center given by X^*. Otherwise, proceed to **Step 3**.

Step 3 (Three points) Construct the center X^* of the minimum covering circle of the three points Y_1, Y_2, Y_3, considering of course the weighted distances. If this solution is given by two points, we let $n = 2$, rename the two points Y_1 and Y_2, and update $\underline{R}(X^*) = w_1 \cdot d_1(X^*) = w_2 \cdot d_2(X^*)$. Then we return to **Step 2**. If the solution is given by three points, we let $n = 3$ and update $\underline{R}(X^*) = w_1 \cdot d_1(X^*) = w_2 \cdot d_2(X^*) = w_3 \cdot d_3(X^*)$. Then we proceed to **Step 4**.

Step 4 (Uncovered fourth point) Let Y_4 denote the point with the largest weighted distance to the center X^* and let $R(X^*)$ denote this weighted distance. That is, $R(X^*) = w_4 \cdot d_4(X^*)$. If $R(X^*) = \underline{R}(X^*)$, stop the algorithm with optimal center given by X^*. Otherwise, proceed to **Step 5**.

Step 5 (Four points) Construct the center X^* of the minimum covering circle of the four points Y_1, Y_2, Y_3, Y_4, considering of course the weighted distances. If X^* is given from two of the points, we rename those points as Y_1 and Y_2. We let $n = 2$, update $\underline{R}(X^*) = w_1 \cdot d_1(X^*) = w_2 \cdot d_2(X^*)$ and return to **Step 2**. If X^* is given from three of the points, we rename those points as Y_1, Y_2 and Y_3. We let $n = 2$, update $\underline{R}(X^*) = w_1 \cdot d_1(X^*) = w_2 \cdot d_2(X^*) = w_3 \cdot d_3(X^*)$ and return to **Step 4**.

To fully understand these steps, we need to discuss how to find the minimum covering circle of three points in the weighted case as one has to in **Step 3**, or how to find the minimum covering circle of four points in the weighted case as one has to in **Step 5**. We will begin by explaining how to find X^* in two separate cases: The center is given by two of the points or the center is given by three of the points.

3.2.1 Finding the center when the center is given by two points

From Klose 2022, p. 53 we know that if the center is given by two points, it must be given as the weighted average of the two points. That is

$$X^* = \frac{Y_1 \cdot w_1 + Y_2 \cdot w_2}{w_1 + w_2}. \tag{3.6}$$

Since this works both when $w_1 = w_2$ and when $w_1 \neq w_2$, we do not need to distinguish between these two cases.

3.2.2 Finding the center when the center is given by three points

When trying to find the center while the center is given by three points, we do however need to consider, whether some of the weights are equal. Specifically we need to distinguish between the case where all weights are equal, where exactly two weights are equal and where none of the weights are equal.

When all weights are equal, that is $w_1 = w_2 = w_3$, this is the same as the unweighted case. Thus the procedure for finding the center of the three points is explained in section 3.1.

For the case when exactly two weights are equal, we assume without loss of generality that $w_1 = w_2 \neq w_3$. The points with equal weighted distance to the points Y_1 and Y_2, is in this case all the points on the perpendicular bisector of the line between these two points. Meanwhile, the points with equal weighted distance to the points Y_2 and Y_3 will lie on a circle. Thus we can find the points with equal distance to all the points, Y_1, Y_2 and Y_3, by finding the intersection points between the this circle and the aforementioned perpendicular bisector. This will generally provide two solutions, and we should then choose the solution which lies in the convex hull of the points Y_1, Y_2 and Y_3 (Klose 2022, p. 54).
Assuming $w_3 > w_2$, we let $r = \frac{w_2}{w_3}$. The circle of points with equal distance to Y_2 and Y_3 has a center C and radius ρ given by (Klose 2022, p. 53)

$$C = \frac{Y_3 - r^2 Y_2}{1 - r^2} \quad \text{and} \quad \rho = \frac{||Y_2 - Y_3||r}{1 - r^2}.$$

If we had $w_3 < w_2$ we would reverse the roles of Y_2 and Y_3 in these equations and let $r = \frac{w_2}{w_3}$ instead.
This means that we can describe the points (x_1, x_2) on the circle by

$$(x_1 - c_1)^2 + (x_2 - c_2)^2 - \rho^2 = 0, \tag{3.7}$$

where $C = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$.
Meanwhile we can describe the points on the line given by the perpendicular bisector of the line from Y_1 to Y_2 with a parameterisation of the line. First of all, we know that the midway point between Y_1 and Y_2 will lie on this line. Letting

$$Y_1 = \begin{pmatrix} Y_{11} \\ Y_{12} \end{pmatrix} \quad \text{and} \quad Y_2 = \begin{pmatrix} Y_{21} \\ Y_{22} \end{pmatrix},$$

we can write this midway point as

$$\begin{pmatrix} \frac{Y_{11}+Y_{21}}{2} \\ \frac{Y_{12}+Y_{22}}{2} \end{pmatrix}.$$

We also know that the direction of the line will be given by the vector from Y_2 to Y_1, $Y_1 - Y_2$, rotated 90 degrees. That is, we have the directional vector

$$\begin{pmatrix} Y_{22} - Y_{12} \\ Y_{11} - Y_{21} \end{pmatrix}.$$

Putting this together, we get the following parameterisation of the line for $t \in \mathbb{R}$:

$$X = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \frac{Y_{11}+Y_{21}}{2} \\ \frac{Y_{12}+Y_{22}}{2} \end{pmatrix} + t \cdot \begin{pmatrix} Y_{22} - Y_{12} \\ Y_{11} - Y_{21} \end{pmatrix}. \tag{3.8}$$

Put another way, we have the two expressions:

$$x_1 = \frac{Y_{11} + Y_{21}}{2} + t \cdot (Y_{22} - Y_{12}), \tag{3.9}$$

$$x_2 = \frac{Y_{12} + Y_{22}}{2} + t \cdot (Y_{11} - Y_{21}). \tag{3.10}$$

Inserting these into eq. (3.7) gives us

$$\begin{aligned}
0 &= \left(\frac{Y_{11} + Y_{21}}{2} + t \cdot (Y_{22} - Y_{12}) - c_1 \right)^2 + \left(\frac{Y_{12} + Y_{22}}{2} + t \cdot (Y_{11} - Y_{21}) - c_2 \right)^2 - \rho^2 \\
&= \left(\frac{Y_{11} + Y_{21}}{2} - c_1 \right)^2 + t^2 \cdot (Y_{22} - Y_{12})^2 + 2 \cdot \left(\frac{Y_{11} + Y_{21}}{2} - c_1 \right) \cdot (Y_{22} - Y_{12}) \cdot t \\
&\quad + \left(\frac{Y_{12} + Y_{22}}{2} - c_2 \right)^2 + t^2 \cdot (Y_{11} - Y_{21})^2 + 2 \cdot \left(\frac{Y_{12} + Y_{22}}{2} - c_2 \right) \cdot (Y_{11} - Y_{21}) \cdot t - \rho^2 \\
&= t^2 \cdot \left((Y_{22} - Y_{12})^2 + (Y_{11} - Y_{21})^2 \right) \\
&\quad + t \cdot ((Y_{11} + Y_{21} - 2 \cdot c_1) \cdot (Y_{22} - Y_{12}) + (Y_{12} + Y_{22} - 2 \cdot c_2) \cdot (Y_{11} - Y_{21})) \\
&\quad + \left(\frac{Y_{11} + Y_{21}}{2} - c_1 \right)^2 + \left(\frac{Y_{12} + Y_{22}}{2} - c_2 \right)^2 - \rho^2.
\end{aligned} \tag{3.11}$$

Now we let

$$
\begin{aligned}
a &= (Y_{22} - Y_{12})^2 + (Y_{11} - Y_{21})^2 \\
b &= (Y_{11} + Y_{21} - 2 \cdot c_1) \cdot (Y_{22} - Y_{12}) + (Y_{12} + Y_{22} - 2 \cdot c_2) \cdot (Y_{11} - Y_{21}) \\
c &= \left(\frac{Y_{11} + Y_{21}}{2} - c_1 \right)^2 + \left(\frac{Y_{12} + Y_{22}}{2} - c_2 \right)^2 - \rho^2 \\
d &= b^2 - 4 \cdot a \cdot c.
\end{aligned}
$$

Finally, this means that when solving eq. (3.11) for t, we get the two values

$$
t_+ = \frac{-b + \sqrt{d}}{2a} \quad \text{and} \quad t_- = \frac{-b - \sqrt{d}}{2a}.
$$

We then obtain two solutions by inserting t_+ and t_- in eq. (3.8):

$$
\begin{aligned}
X_+ &= \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}_+ = \begin{pmatrix} \frac{Y_{11}+Y_{21}}{2} \\ \frac{Y_{12}+Y_{22}}{2} \end{pmatrix} + t_+ \cdot \begin{pmatrix} Y_{22} - Y_{12} \\ Y_{11} - Y_{21} \end{pmatrix} \\
X_- &= \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}_- = \begin{pmatrix} \frac{Y_{11}+Y_{21}}{2} \\ \frac{Y_{12}+Y_{22}}{2} \end{pmatrix} + t_- \cdot \begin{pmatrix} Y_{22} - Y_{12} \\ Y_{11} - Y_{21} \end{pmatrix}.
\end{aligned}
$$

As mentioned, we use the solution that lies in the convex hull of Y_1, Y_2 and Y_3. To determine which one of the two points lies in the convex hull, we use the python package scipy to find the convex hull of Y_1, Y_2 and Y_3, and also to find the convex hull of Y_1, Y_2, Y_3 and X_+. Since the convex hull is defined by its extreme points, we check if these two convex hulls have the same extreme points. If they do, we must have that X_+ lies in the convex hull of Y_1, Y_2 and Y_3, and we therefore use X_+ as the center. Otherwise we use X_- as the center.

Finally, for the case where all the weights are different, without loss of generality we assume that $w_1 < w_2 < w_3$. We can then find two circles. One describing the points of equal weighted distance to Y_1 and Y_2, and another describing the points of equal weighted distance to Y_2 and Y_3. As already mentioned, we can find the centers and radii of these circles in the following way.

$$
C_1 = \frac{Y_2 - r_1^2 Y_1}{1 - r_1^2} \quad \text{and} \quad \rho_1 = \frac{||Y_1 - Y_2|| r_1}{1 - r_1^2},
$$

and

$$
C_2 = \frac{Y_3 - r_2^2 Y_2}{1 - r_2^2} \quad \text{and} \quad \rho_2 = \frac{||Y_2 - Y_3|| r_2}{1 - r_2^2},
$$

where $r_1 = \frac{w_1}{w_2}$ and $r_2 = \frac{w_2}{w_3}$.
The center defined by our three points can then be found by finding the intersections of the two circles. This gives two solutions, and we then choose the solution that lies in the convex hull of the points Y_1, Y_2 and Y_3. The intersection of these two circles are found as described in Klose 2022, p. 62. Letting

$$
C_1 = \begin{pmatrix} c_{11} \\ c_{12} \end{pmatrix}, \quad C_1 = \begin{pmatrix} c_{21} \\ c_{22} \end{pmatrix} \quad \text{and} \quad d = ||C_1 - C_2||,
$$

we first calculate

$$
K = \frac{1}{4} \sqrt{((\rho_1 + \rho_2)^2 - d^2) \cdot (d^2 - (\rho_1 - \rho_2)^2)}.
$$

Using this, we can now find the two intersection points by calculating

$$x_1 = \frac{1}{2}(c_{11} + c_{21}) + \frac{1}{2}\frac{c_{21} - c_{11}}{d^2}(\rho_1^2 - \rho_2^2) \pm 2(c_{22} - c_{12})\frac{K}{d^2}$$
$$x_2 = \frac{1}{2}(c_{12} + c_{22}) + \frac{1}{2}\frac{c_{22} - c_{12}}{d^2}(\rho_1^2 - \rho_2^2) \mp 2(c_{21} - c_{11})\frac{K}{d^2}.$$

Once we have these two solutions, we check which one is in the convex hull of Y_1, Y_2 and Y_3 in the same way, as we checked it in the case where exactly two of the weights are equal.

Now that we have covered how to find the center, both in case the center is defined by two points and in case the center is defined by three points, we can take a closer look at **Step 3** and **Step 5** from earlier in section 3.2.
In **Step 3** we are told to "Construct the center X^* of the minimum covering circle of the three points Y_1, Y_2 and Y_3 considering of course the weighted distances". More precisely, we look at different combinations of the points one at a time. We know, that we cannot define the circle from Y_1 and Y_2 alone, as Y_3 has been chosen such that Y_3 is uncovered by the circle created from Y_1 and Y_2, so we consider the combinations $(Y_1, Y_3), (Y_2, Y_3)$ and finally (Y_1, Y_2, Y_3). We start with the combinations of two points, find the center X^* from these two points as described in section 3.2.1 and see if the weighted distance from the center to the excluded point is smaller than the weighted distance to either of the points in the current combination. E.g. for the combination (Y_1, Y_3) we find a solution X^*, then we check if $w_2 \cdot d_2(X^*) \leq w_1 \cdot d_1(X^*)$. If this is the case, the center is defined by the two points Y_1 and Y_3 and we stop looking for other solutions. Otherwise we continue with the next combination, and if no combination of two points gave a covering circle, we use all three points and find the center in the way described in section 3.2.2.
In **Step 5** we do a very similar thing. For the same reasons as before, we know that Y_4 is uncovered by the current circle, so it must be included in all combinations we check. Hence we have to check the combinations: $(Y_1, Y_4), (Y_2, Y_4), (Y_3, Y_4), (Y_1, Y_2, Y_4), (Y_1, Y_3, Y_4)$ and (Y_2, Y_3, Y_4). For each of these combinations we then have to check if the points left out are covered by the circle resulting from the points in the combination. Again, this circle is defined by its center, which is found as described in section 3.2.1 or section 3.2.2 depending on the number of points in the combination. E.g. if we consider (Y_1, Y_4), this gives us a solution X^*, and we must then have that $w_2 \cdot d_2(X^*) \leq w_1 \cdot d_1(X^*)$ and $w_3 \cdot d_3(X^*) \leq w_1 \cdot d_1(X^*)$ for this to be a feasible solution. Otherwise we try the next combination. One important thing to note is that if we again consider the combination (Y_1, Y_4) giving us the center X^* and have $w_2 \cdot d_2(X^*) \leq w_1 \cdot d_1(X^*)$ but $w_3 \cdot d_3(X^*) > w_1 \cdot d_1(X^*)$ this is of course infeasible. In this situation we may not be able to construct a solution when we reach the combination (Y_1, Y_2, Y_4), as we are not guaranteed to have an intersection of the two circles we construct. This is because we may find, that any point that has equal weighted distance to Y_1 and to Y_4, may have a strictly smaller weighted distance to Y_2. Thus we must exclude this combination from consideration, or the code will crash.

Chapter 4

Numerical Results and Further Discussion

We have tested our implementations of each of the described algorithms on the cities of Mid Jutland, placing both 5, 6, 7, 8, 9 and 10 drone stations. Since each of the algorithms contain some randomness that affect the final solution, we ran each algorithm 50 times and kept the best solution found. An example of a solution can be seen in figure 4.1. The small blue dots are

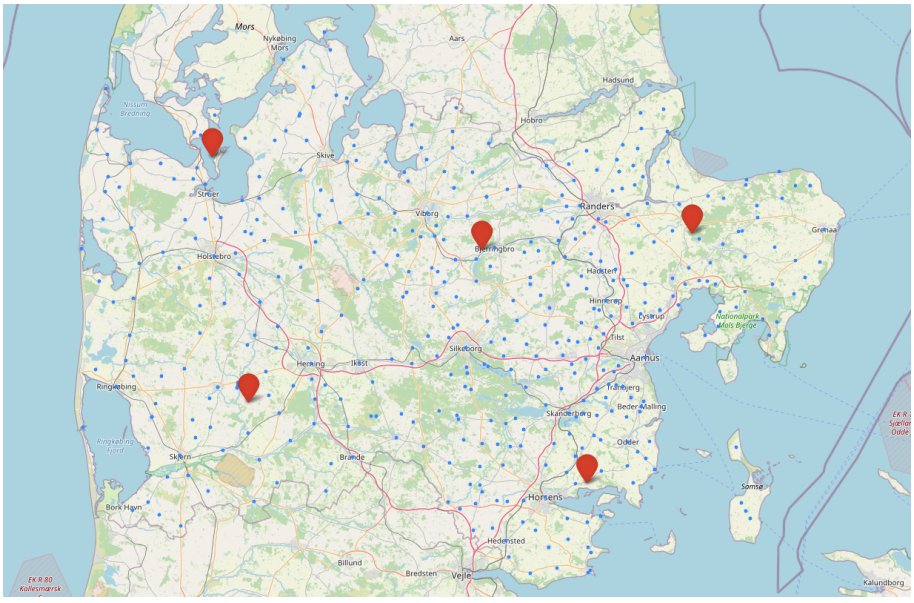

Figure 4.1: Solution to unweighted p-center problem with p=5.

the cities in our data set and the large red markers mark the locations of the drone stations. More plots like this for the other solution approaches and other values of p can be found at our GitHub page[1]. Some key-values for each of our solutions can be seen in table 4.1. The values from the table can also be seen plotted in figure 4.2. While there is no point in comparing the

1: https://github.com/jenstrolle/loc_handins.

Table 4.1: Table showing key-figures from results. Distances are measured in km.

p	**5**	**6**	**7**	**8**	**9**	**10**
Cooper: Sum of weighted distances	15451074.0	12921800.3	10930881.5	9907800.5	9230691.0	8268505.7
Cooper: Largest distances	62.45	56.95	53.42	53.42	53.42	53.42
Weighted p-center: Largest weighted distance	941790.2	705950.9	606990.2	501460.3	302533.7	291768.2
Weighted p-center: Largest distance	64.78	64.78	48.83	47.66	48.45	48.45
Unweighted p-center: Largest weighted distance	8691828.1	7851793.8	7908552.0	6140123.1	4946918.7	6076248.1
Unweighted p-center: Largest distance	38.83	33.94	32.03	29.89	27.69	25.91

sum of largest weighted distances found in the Cooper algorithm to any other key figure, it is still worth noting that the decrease in the sum of the weighted distances is notable with every increase of p. The largest improvement happens when going from 5 to 6 drone stations, but still when we go from 9 to 10 drone stations, the improvement is quite large. However, in order to properly say what number of drone stations should be chosen, we need crucial information such as the cost of building a drone station. Also, when we however look at the largest distance from a city to the nearest drone station, we see that for no $p \in \{5, 6, 7, 8, 9, 10\}$ is this distance lower than the 50 km range of the drone signal. Thus for the problem at hand, all the found solutions are infeasible. A final thing to note is that if one looks at the aforementioned plots of solutions on maps, one will see that Cooper's algorithm generally places the drone stations in the largest cities such as Århus, Randers, Horsens and so on, or somewhere reasonably between two or three large cities.

When considering the solutions to the weighted p-center problem, we observe very large improvements in the largest weighted distance when increasing p until $p = 9$. The improvement from nine to 10 drone stations does however seem small, and it would be difficult to argue that the 10'th drone station should be built. Here we do however see that when p is larger than seven, the drones will always stay within signal reach of the drone stations thus making the solutions feasible, but only just. Finally we note that when looking at the aforementioned plots of solutions on maps, we observe that the algorithm applied to solve the p-center problem with weighted distances generally places the drone stations close to the larger cities, but it is noticeable that the furthest cities also pull on the locations of the drone stations.

For the unweighted p-center problem, we find that the drone stations are quite evenly spread out. This is also reflected by the largest distances from a city to the nearest drone station being almost half the distance found by the two other algorithms for each p. This also means that we have a feasible solution for $p = 5$, but the largest distance decreases steadily for larger values of p. Looking at the largest weighted distances however makes it clear that the drone stations are now placed far from the largest cities. The largest weighted distance is around 10 times as large for each p as when the solutions were found by solving the weighted p-center problem.

4.1 Discussion of the different solution approaches

Looking at the results and solutions found by the different solution approaches, there is no immediate best approach. The drone station locations found by considering the unweighted p-center problem would give the smallest reaction time for a drone to reach the furthest cities, but if the drone very rarely has to fly there because very few people live there, this doesn't seem very important.
So if the number of people that live in a city is proportional to the probability that a drone has to fly to the city, we should rather choose a solution found by Cooper's algorithm or the algorithm for the weighted p-center problem. Specifically, if the probability of needing to go to city i is given by

$$p(i) = \frac{w_i}{\sum_{j=1}^{m} w_j}, \quad \forall i,$$

then we have that the objective value in the single Weber problem is

$$\sum_{i=1}^{m} w_i d_i(X) = \sum_{j=1}^{m} w_j \cdot \sum_{i=1}^{m} \frac{w_i}{\sum_{j=1}^{m} w_j} d_i(X) = \mathbb{E}[d_i(X)] \cdot \sum_{j=1}^{m} w_j.$$

This means, that if we wish to minimise the expected value of the distance travelled, we should probably choose one of the solutions found by the Cooper algorithm. But we already mentioned that we find no solutions with the Cooper algorithm that has largest distance to a city smaller than 50 km. Also, seeing as these solutions are generally placed in the centres of the large cities, it may not be very easy to place the required runways in these locations, depending on the size of these, and possibly the noise such a station would generate.
This makes the solutions to the weighted p-center problem seem the most attractive, since these take the weights into account and they are feasible for $p \geq 7$. Also, when looking at the maps, we see that the solutions found by solving the weighted p-center problem are often not very different from the solutions found by Cooper's algorithm.

We have also thought of a hybrid approach that might also lead to decent solutions. Assuming that the probability of sending a drone to a certain city is proportional to the number of people living in the city, we want to first run the Cooper algorithm to find solutions that more or less minimise the expected value of the flight times/distances of the drones. Our approach is summarised in the following steps:

Step 1 Let X_j^C for $j = 1, ..., p$ denote the solutions found with Cooper's algorithm. Let $k = 1$ and $X_j^0 := X_j^C$ for $j = 1, ..., p$. Proceed to **Step 2**.

Step 2 Run a single iteration of the algorithm used to solve the unweighted p-center problem while using the solutions X_j^{k-1} as initial solutions. Let X_j^U denote the solutions found this way for $j = 1, ..., p$. Proceed to **Step 3**.

Step 3 Set

$$X_j^k = (1 - \alpha)X_j^{k-1} + \alpha X_j^U \quad \text{for } j = 1, ..., p$$

for a well-chosen $\alpha \in (0, 1)$. Proceed to **Step 4**.

Step 4 If the largest distance from a facility to a city is less than 50 km or if $X_j^k = X_j^{k-1}$ for $j = 1, ..., p$, then stop. Otherwise let $k = k + 1$ and return to **Step 2**.

The idea with this approach is to stay as close to the solutions found by the Cooper algorithm while carefully moving towards feasible solutions. This will likely end in a feasible solution, if such a solution can be found by the algorithm for solving the unweighted p-center problem.

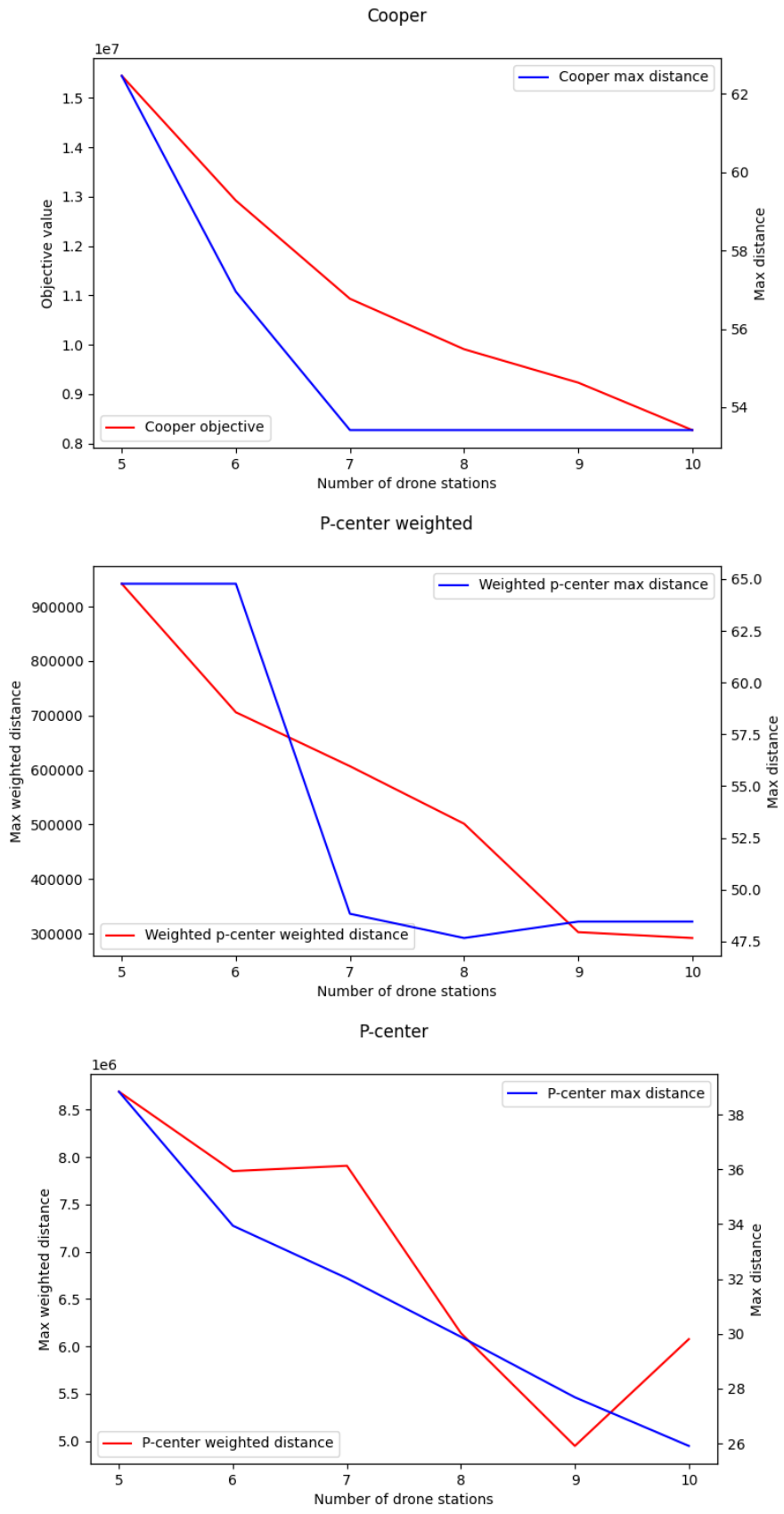

FIGURE 4.2: Values from table 4.1 turned into plots.

Chapter 5

Comparison with the Literature

A method for solving the p-center problem with continuous demand is presented in Drezner et al. 1996. This is equivalent to covering a area in the plane with p identical circles with the smallest possible radius. They argue, that continuous demand is more realistic. An example they present is mobile-network covering, where people are not still at all times.

The described approach in the paper can be applied to any area in the plane although areas with the shape of a convex polygon are much easier to deal with. More concrete the paper has its focus on a square area.

The approach in the paper is a Voronoi-based solution procedure which is an extension of Cooper's ideas and then a special "finishing up" procedure formulated.

The Voronoi-based heuristic approach in the paper can be summarized in 5 steps:

Step 1 Generate p centers randomly in the given area.

Step 2 Use the p centers to construct the Voronoi diagram.

Step 3 Solve the one-center problem in the Voronoi polygon and relocate the p centers.

Step 4 If the improvement from one iteration to another is less than some small ε or the maximum number of iterations is exceeded, stop.

Step 5 Otherwise go to **Step 2**.

When the Voronoi procedure terminates there will be p radii of various lengths. The intuitively idea of the "finishing up" procedure is, for these p radii there will be a maximum- and a minimum radius. Then if there exist a radius smaller than the maximum radius, it would be plausible that this radius could be increased and simultaneously decrease the maximum radius while the covering of the area is still fulfilled.

The "finishing up" procedure is only applied when the Voronoi procedure has terminated, and once the radii of the Voronoi polygons from different iterations is close to each other the procedure has a slow final convergence which can result in a long run time before the "finishing up" procedure can be applied.

5.1 Comparison between Drezner et al. 1996 and our Models

As Drezner et al. 1996 are only looking to solve unweighted p-center problems, we will not compare their model with our proposed multi-source Weber model.

In Drezner et al. 1996, one hopes to solve problems dealing with convex shapes, while in our case, we are not limited by this. The reason for them dealing with only convex shapes is to more easily control the solutions they find. Using a finishing procedure like in Drezner et al. 1996 to make final improvements on solutions seems a very good idea. Doing this also decreases their computation time.

Our models ran quite quickly, but doing something similar to their finishing up algorithm might have lead to even better solutions. We note, that we use an equation solving library in python, which makes up the bulk of the computation time in the unweighted p-center problem.

Both of our models assume that we are able to place locations in a continuous area. Intuitively, one would assume that in our models, locations would be placed in the convex hull of all cities. This is also observed in testing. In our models it is very clear which places drone stations cannot be placed i.e. lakes and the sea. This is not at all taken into account in Drezner et al. 1996. A natural extension of this paper would be to limit feasible location placements to be on land.

In Drezner et al. 1996 the model tries to cover all of some area, while we are trying to "hit" every city in the dataset. This distinction is important, as it means our solution criteria are fundamentally different. The methods can find solutions to the respective problems, but they might differ in quality. One would be able to compare these solutions if both were applied to the same problem.

Chapter 6

Conclusion

In this project we have examined the problem of placing five to 10 drone stations in Mid Jutland. We introduced three different ways of looking at the problem:

Obj. 1 Minimizing the sum of the weighted distances from each city to its closest facility (Multi-Weber problem).

Obj. 2 Minimizing the largest weighted distance from a city to its closest facility (Weighted p-center problem).

Obj. 3 Minimizing the largest distance from a city to its closes facility (p-center problem).

We then covered in detail how each of these problems were solved using heuristic algorithms. The general approach is to split the cities into p regions and then solve the single-Weber, the weighted one-center problem or the unweighted one-center problem respectively for each region, before updating the regions. This process is repeated until convergence.

We then examine the results from solving the problem in the three different ways. We note that if the only goal is to minimise the expected distance flown by a drone and the probability of sending a drone to a city is proportional to the number of people in the city, then **Obj. 1** is the most intuitive, and **Obj. 3** seem the least relevant, as we here ignore the number of people living in each city. However when focusing on **Obj. 1** the drones have to fly further than they can transmit the live video feed, which makes **Obj. 2** seem the best to focus on. This chapter is rounded off with a short description of a hybrid approach that tries to find the feasible solutions closest to the solutions achieved by focusing on **Obj. 1**, however this approach is not tested.

We end the project with a comparison to Drezner et al. 1996. While they consider a problem with continuous demand where we considered discrete demand, the two approaches taken to the problem are generally alike.

Bibliography

Drezner, Z. and Suzuki, A. (1996). “The p-center location problem in an area”. In: *Location Science* 4.1, pp. 69–82.

Klose, A. (2022). *Lecture Notes on Location Planning*.