Multi-objective optimisation of a commercial vehicle complex network

by

Sumarie Meintjes
28408129

Submitted in partial fulfillment of the requirements
for the degree

Bachelors of Industrial Engineering

in the

Faculty of Engineering, Built Environment and Information Technology

University of Pretoria

October 2013

Executive Summary

In this project we build on research that has been done by Joubert and Axhausen (2013), who built a commercial vehicle complex network for Gauteng. Two shortcomings are identified in the approach they followed. The first shortcoming is the approximations used to determine whether an activity formed part of a cluster. These approximations resulted in some activities to be assigned to the wrong clusters, and other activities to not be assigned to any cluster. The second shortcoming is that the completeness of the complex network was never explicitly considered when they evaluated the different combinations of input clustering parameters.

We address the first shortcoming by generating a concave hull for each cluster. The concave hull envelopes all points in the cluster, and one can accurately determine whether an activity forms part of a cluster. To generate the concave hulls, we integrate the Duckham Algorithm with the existing clustering algorithm used by Joubert and Axhausen (2013). The first step of the Duckham Algorithm is to generate the Delaunay triangulation of the cluster. For some combinations of input clustering parameters, more than 2% of the clusters were degenerate. A degenerate Delaunay triangulation occurs when three or more points in a cluster are colinear (lie on a straight line), or when four points in a cluster are cocircular (lie on the circumference of a circle). No valid Delaunay triangulations can be generated for these clusters. We suggest to deal with these degeneracies by using the weighted average of the points as a reference to the cluster, instead of simply ignoring it.

We consider the completeness of the complex network as part of a multi-objective problem: we cannot maximise completeness without making a trade-off with computational complexity. We address this multi-objective problem by conducting a multiple response surface experiment and performing multi-objective evaluation by constructing two efficient frontiers. From the multiple response surface experiment, we found that the input clustering parameters (ϵ, p_{min}) that optimises the completeness of the complex network, while minimising the computational complexity, is (1, 2). From the multi-objective evaluation, we determined that in general, using $\epsilon = 1$ will result in an efficient point.

To conclude, we use input clustering parameters (1, 2) to build a commercial vehicle complex network in the Nelson Mandela Bay Municipality, and perform various network analyses on this network.

Acknowledgements

I would like to thank Quintin van Heerden for his continued assistance regarding programming and design of experiments. Thank you for support throughout the year. I would also like to thank Wynand Breytenbach for his assistance with design of experiments.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Businesses connect to each other on a daily basis. One type of connection that is of particular interest in this project is the movement of commercial vehicles between businesses. These movements indicate which businesses are dependent on each other, and to what extent they are dependent.

Joubert and Axhausen (2011) took the first step in understanding commercial vehicle movements in Gauteng by analysing the time and spatial characteristics of disaggregated commercial vehicle activities. For this purpose, *DigiCore Fleet Management* provided raw global positioning system (GPS) data of more than 40 000 commercial vehicles travelling in South Africa over a six-month period. From this dataset, Joubert and Axhausen (2011) extracted vehicle activity chains, which they then analysed. The focus of their analysis was on the movement and activities of these vehicles. They did not, however, analyse the locations of the facilities where the activities were performed. The facilities that commercial vehicles move to and from are assumed to be firms that participate in commercial activities, such as manufacturing plants, distribution centres and shopping centres.

As a next step, Joubert and Axhausen (2013) wanted to understand the connectivity between the facilities that commercial vehicles move to and from. They used the vehicle activity chains to establish a commercial vehicle transport planning model.

To build the model, they first had to estimate the locations of the facilities where the activities are performed. From the vehicle activity chains they could estimate the location of each facility, as each activity has a GPS coordinate indicating where it was performed. However, the GPS coordinates are not accurate, and it is difficult to determine true facility location from these coordinates alone. Even though many vehicles may stop at the same facility to perform an activity, each activity's GPS coordinates may still be unique. Since multiple activities of multiple vehicles were considered over a six-month period, it resulted in a high density of activity points around areas where facilities are located. These regions of high density had to be identified and grouped so that facility locations could be estimated.

A density-based clustering algorithm was used, each cluster representing a facility at which a commercial vehicle can perform an activity. A unique identifier, called the *facility ID*, was assigned to each clustered facility to keep track of them. The size of the clusters and the number of clusters that are generated depend on parameters that must be set by the user. If more clusters are generated, more facilities will be included in the model and it will be a more complete representation of reality. Unfortunately, this also increases the computational complexity of the model.

The second part of building the model was to determine in what sequence each commercial vehicle moves between facilities. The vehicle activity chains were used, since the movement of a commercial vehicle between two facilities is represented by two consecutive activities in the vehicle's activity chain. However, only the movements between two clustered facilities were included in the model. If an activity was performed at a clustered facility, it was assigned the

clustered facility's unique *facility ID*. Therefore, if two consecutive activities both have *facility ID*s associated with them, this meant that a movement was made between two clustered facilities, and the movement was included in the model.

Borgatti and Li (2009) explain that discrete interactions between actors often imply the existence of an underlying relationship, even though the reason for the interaction between two actors is often not known. Consequently, Joubert and Axhausen (2013) assume that a direct trip made between two facilities indicates that there exists some business relationship between the two facilities. These relationships form the basis of networks of connectivity between facilities, and allowed them to establish a complex network of commercial vehicles in Gauteng.

1.1 Complex networks

Borgatti and Li (2009) explain that to take a *network perspective* means to view any system as a set of interrelated actors or nodes. So, taking a network perspective on commercial vehicle movements between businesses implies considering the businesses as actors that interact with and influence one another. This definition also implies that as many as possible, if not all, businesses in the network should be included in the model. The more businesses that are included in the network, the more accurately one can determine how they influence one another. This big-picture view of the network is very different to traditional supply chain models, where only a single firm is identified as the subject of study (Joubert and Axhausen, 2013).

Newman (2003) defines a network in its simplest form as a set of nodes, called *vertices*, with connections between them, called *edges*. A sample network has been adapted from Newman (2003) and can be seen in Figure 1.1.

The first step in building a network is to define its vertices and edges. Borgatti and Li (2009) explain that there is no incorrect way of defining a network's vertices and edges: simply define the relations that you choose to study. Joubert and Axhausen (2013) chose to define the facilities as vertices, and the commercial vehicle movements made between the facilities as edges.

Newman (2003) explains that when a network consists of different types of vertices, different types of edges, or both, it is considered to be a complex network. Based on his description of different types of vertices and edges, it is concluded that all vertices (facilities) in this network are similar. However, not all edges in this network are similar. Since a vehicle travelling from facility a to b does not imply a similar movement from facility b to a, these two movements will be represented by two separate edges pointing in opposite directions. Such a complex network is now referred to as being *directed*. When a trip is made between two facilities, it is assumed to strengthen the relationship between the facilities in the direction that the trip was made. For every direct trip made between two facilities, the weight of the directed edge is incremented by one. Therefore, the strength of the relationship between the two facilities is represented by the weight of the edge, with the direction of the edge indicating the direction of the interaction.

Joubert and Axhausen (2013) explain why it can be beneficial to transport planners and policy makers to build a commercial vehicle complex network. Network theory allows one to identify key players in the network, as well as their associated industries and facilities. By identifying the key players, policies and technologies may be implemented and deployed more rapidly through these key players. Network theory also provides a way to test how vulnerable and resilient the network is to deliberate attack on its actors and the relationships between them. A chain is only as strong as its weakest link, so identifying and addressing the weaknesses in a network will be beneficial to the network as a whole.

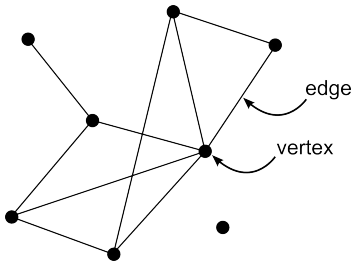


Figure 1.1: A sample network in its simplest form, adapted from Newman (2003). Networks can be seen all around the world: researchers who cite one another's academic articles, friends who make telephone calls to one another and companies that conclude business transactions with one another.

1.2 Building the commercial vehicle complex network

Many activity chains were generated for each vehicle over the six-month period. Joubert and Axhausen (2013) assigned a unique *digicoreVehicle ID* to each vehicle to differentiate between all the vehicles and their activity chains. Figure 1.2 shows how the vehicles and their activity chains were recorded. For each vehicle, one is able to determine when, where, and in what sequence activities were performed.

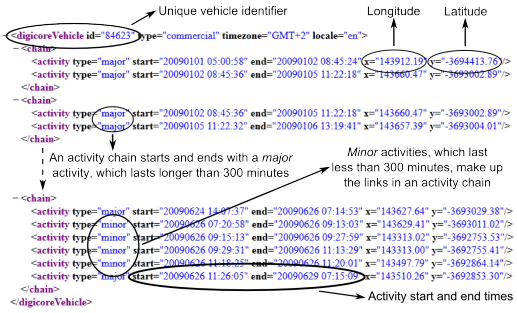


Figure 1.2: Vehicle activity chains before modification. Some of these activities could have been performed at the same facility, but since all the activities' locations are unique, there is no way of knowing at which facility an activity was performed.

1.2.1 Identifying the facilities in the complex network

Joubert and Axhausen (2013) used a density-based clustering algorithm, called the *DJ-Cluster* approach (Zhou et al., 2004) to identify facilities. To understand the *DJ-Cluster* approach, consider the clustering example in Figure 1.3.

In Figure 1.3(a) activities are superimposed on an aerial photograph of facilities where commercial activities are performed. In Figure 1.3(b) the *DJ-Cluster* approach is used to identify clusters for these activities. For each point p, the neighbourhood is calculated as all the points within a specified distance, ϵ, from point p. For a neighbourhood to be considered a cluster around p, denoted by C, it must consist of at least p_{min} points. In this example, $\epsilon = 4.8$ units and $p_{min} = 10$ points.

If no cluster C exists around point p, p is not considered to have a big enough logistics impact to be included in the complex network, and it is ignored. If there exists a cluster C around point p, p and its neighbouring points are identified as a new cluster, as long as none of the points in C are associated with an existing cluster. However, if any of p's neighbouring points are associated with an existing cluster, p and all its neighbouring points are merged with C into a new cluster.

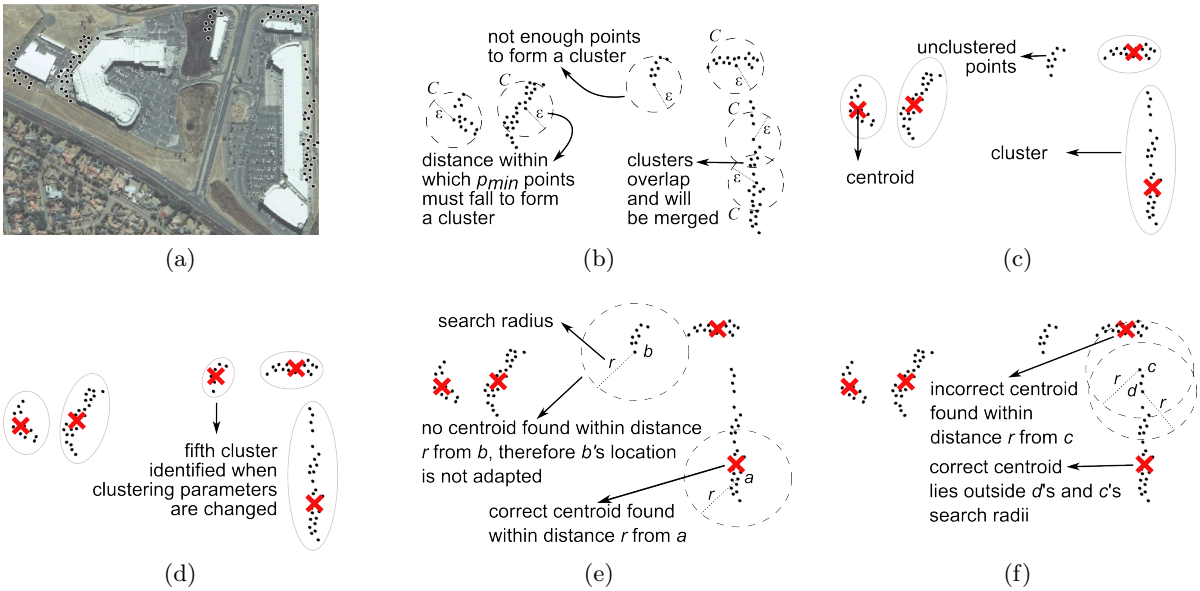


Figure 1.3: (a) Activities are superimposed on an aerial photograph. (Source: South African National Geospatial Institute (NGI) imagery available on *OpenStreetMap* at location -26.114435, 27.894554) (b) Of the six neighbourhoods that are being calculated, five are identified as clusters, each denoted by C. (c) Four clusters have been identified. Each cluster's centroid represents the location of the facility. (d) By reducing the minimum number of points, p_{min}, that make up a cluster, five clusters are identified for the same set of points. (e) Search radius, r, is used to approximate whether activities a and b form part of a cluster. (f) Activities c and d are misassociated to facilities using the search radius.

Each cluster's centroid is calculated and the facility's location is estimated to be the location of the centroid. A unique identifier, the *facility ID*, is allocated to each centroid. Figure 1.3(c) shows the four clusters and their centroids that were identified using the original clustering parameters. Millions of points were considered during the clustering step, and it would have been a computational burden to keep record of every single point that formed part of a cluster. Therefore, after the centroids were calculated, the record of the points which formed part of each cluster were discarded.

It has been mentioned that the size and the number of clusters generated are determined by the two clustering parameters, p_{min} and ϵ, which must be set by the user. Figure 1.3(d) illustrates this fact, since one more cluster has been identified by simply decreasing p_{min} to 5 points. Therefore more, or fewer, clusters can be identified by changing the combination of clustering parameter values.

A crucial part of this step was to determine which combination of clustering parameters would correctly cluster the activities. Joubert and Axhausen (2013) experimented with different combinations of clustering parameter values and visually inspected aerial photographs to determine how accurately facilities were identified and clustered. They used various metrics to evaluate the performance of the clustering parameters. However, the completeness of the resulting complex network was not one of the metrics considered in their evaluation. It is possible that the chosen clustering parameters resulted in many activities not being clustered, therefore fewer clustered facilities were identified, reducing the completeness of the complex network.

1.2.2 Adapting activity locations

If an activity formed part of a cluster, its location had to be adapted to that of the cluster's centroid, and the *facility ID* had to be assigned to it. Since there was no record of the activities

that formed part of the clusters, approximations were used to complete this step.

Figure 1.3(e) shows how these approximations were done using a search radius, r. For each point, p, the closest centroid within distance r is assumed to be the centroid of the facility where the activity was performed. Using the search radius, the correct centroid of the facility where activity a was performed, is identified. In the vehicle activity chain where activity a occurs, its location is adapted to that of the centroid, and the centroid's *facility ID* is assigned to it.

No centroid is located within distance r from activity b, therefore it is correctly assumed that activity b was not performed at a clustered facility. Its location in the activity chain remains the same, and no *facility ID* is assigned to it.

Figure 1.3(f) illustrates two cases where the search radius fails to adapt an activity's location to the correct clustered facility.

1. The incorrect centroid falls within activity c's search radius, and it is incorrectly assumed that it was performed at this facility. As a result, activity c's location is adapted to that of the wrong cluster, and the wrong *facility ID* is assigned to it.

2. Activity d was performed too far from its centroid, therefore the search radius fails to identify its cluster. It is assumed that activity d was not performed at a clustered facility, therefore its location is not adapted to that of the centroid, and no *facility ID* is assigned to it.

Executing this stage resulted in modified activity chains, where some of the activities had new locations and *facility ID*s allocated to them. Figure 1.4 shows three of the modified activity chains of *digicoreVehicle 84623*.

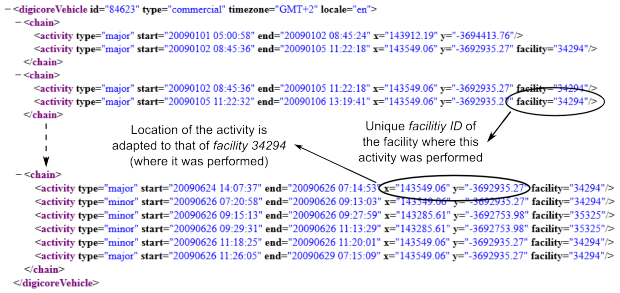


Figure 1.4: Vehicle activity chains after modification, indicating with a unique *facility ID* whether an activity was performed at a clustered facility.

1.2.3 Identifying the edges in the complex network

During the final stage, only direct trips made between two clustered facilities were included in the complex network as edges. To identify these trips, consecutive activities in the modified activity chains were considered. If both activities had a *facility ID* it meant that both formed part of a clustered facility, and the direct trip had to be included in the complex network. If one, or both, of the nodes did not yet exist in the complex network, it had to be created. The directed edge was then created between the nodes. If both of the nodes already existed, and a directed edge had already been created to represent the same movement, the edge's weight was incremented by one. However, if one, or both, of the activities did not have a *facility ID*, an edge could not be created between the two facilities.

1.3 Research design

The main deliverable for this project is a complex network of commercial vehicles in the Nelson Mandela Bay Municipality (NMBM). The complex network will be built in *Java* using the open

source MATSim (Multi-Agent Transport Simulation) toolkit, released under the GNU Public License (GPL).

The same approach used by Joubert and Axhausen (2013) will be used to build the complex network, with two improvements:

1. Implement another method of adapting activity locations to improve the accuracy thereof.
2. Consider the impact that the clustering parameters have on the completeness of the complex network, while recognising that this will increase the computational complexity.

1.3.1 Increasing the accuracy of adapting activity locations

Instead of using search radius, r, a polygon should be generated for each cluster. Figure 1.5 illustrates that activities a, c and d all fall within, or on, the boundary of their cluster's polygon. Therefore they will be assigned to the correct cluster, and their locations will be adapted correctly. Activity b does not fall within, or on, any polygon's boundary and it is not assigned to any cluster.

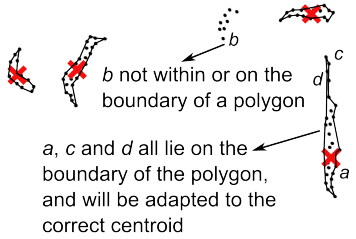


Figure 1.5: Generating a polygon for each cluster. The polygon will enclose all the points that form part of the cluster.

Galton and Duckham (2006), Duckham et al. (2008) and Park and Oh (2012) all agree that the concave hull is the polygon that accurately describes the region occupied by a set of points. Two suitable concave hull algorithms were considered for integration with the clustering algorithm, and will be compared using two criteria:

1. **The quality of the concave hull.** The concave hull must resemble the shape of the dataset.
2. **The time complexity of the concave hull algorithm.** Concave hull generation is an additional computational burden, and the chosen concave hull must have a low time complexity.

1.3.2 Considering the complex network completeness when evaluating the clustering parameters

Complex network data was generated by conducting multiple runs using different combinations of clustering parameters. For each combinations of clustering parameters, the following data was collected:

1. The number of nodes, representing the completeness of the complex network.
2. The computational time to build the complex network.
3. The file size of the resulting complex network.

The guidelines for designing an experiment, as set out by Montgomery (2009), were used to conduct a multiple response surface experiment. Response surface methodology is a collection of mathematical and statistical techniques useful for the modelling and analysis of problems in which a response of interest is influenced by several variables and the objective is to optimise the response (Montgomery, 2009). A 3-dimensional response surface was plotted for each response. This provided us with a visual representation of how the responses change with different combinations of clustering parameters. By overlaying the contour plots of all three responses, the combination of clustering parameters that maximises completeness, and minimises both time and size, were determined.

We acknowledge that this project deals with a multi-objective problem: when we try to optimise file size or computational time, we lose completeness. Therefore, a comprehensive multi-objective evaluation was performed to make a trade-off between completeness, time and file size.

1.4 Document structure

In Chapter 2, examples of activity points are used to illustrate why the concave hull best describes the distribution of these points. An intuitive concave hull algorithm proposed by Park and Oh (2012) and a time-efficient concave hull algorithm proposed by Duckham et al. (2008) are introduced. The theory of complex networks and their application in a supply chain context are also discussed in this chapter. In Chapter 3, both concave hull algorithms are implemented and tested with sample datasets. The chosen concave hull algorithm is integrated with the clustering algorithm, and the results are shown. Some problems resulting from the chosen concave hull algorithm are also discussed in this chapter, and solutions are suggested. Chapter 4 provides a guideline for designing and conducting experiments, which will be followed to conduct the multiple response surface experiment. The necessary statistical analyses are done, and the resulting response surface plots are generated and discussed. In Chapter 5, the multi-objective evaluation is conducted, and trade-offs are made by constructing two efficient frontiers. Based on the findings from Chapters 4 and 5, we recommend a combination of clustering parameters to use for building a complex network in Chapter 6. Some of the network analyses discussed in Chapter 2 are used to analyse this complex network. In Chapter 7, the project is concluded and recommendations for future research are made.

Chapter 2

Literature review on concave hulls and network theory

Four different bodies of knowledge are used in this project, namely concave hulls, network theory, response surface methodology and multi-objective optimisation. In this chapter, concave hulls and network theory are discussed in a supply chain context. The literature regarding response surface methodology and multi-objective evaluation are discussed in Chapters 4 and 5, respectively.

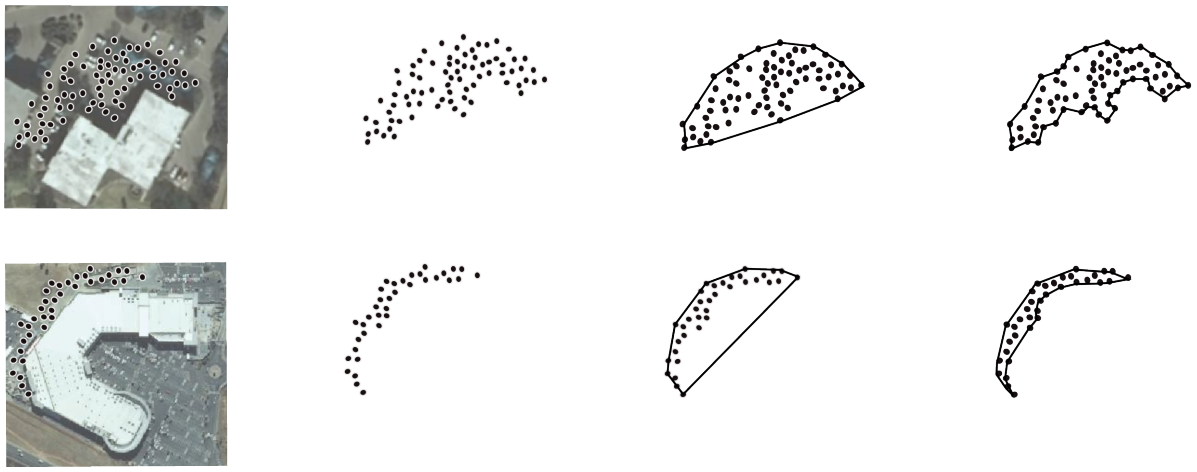
2.1 Concave hull algorithm

In this project we propose to use a polygon to describe each clustered facility. The polygon's centroid will be an estimate of the facility's location, and all the activities associated with the facility must be covered by the polygon. Galton and Duckham (2006) mention that implementing a dedicated clustering algorithm first, and then generating polygons for each cluster individually, is a potentially useful way to describe the region occupied by a set of points. Joubert and Axhausen (2013) have already implemented a density-based clustering algorithm. The second algorithm, which will generate a polygon for each cluster, will be described in this chapter.

Galton and Duckham (2006) explain that the polygon that is often used to describe a region associated with a set of points is the convex hull. The convex hull for a set of points is the smallest convex polygon that encloses the set of points, and is often used in geographic information science (GIScience), pattern recognition and image processing (Park and Oh, 2012). One advantage of the convex hull is that there exists one unique hull for a set of points. Another advantage that Duckham et al. (2008) and Park and Oh (2012) mention is that there are a great number of convex hull algorithms to choose from.

On the other hand, the convex hull fails to describe regions with prominent concave distributions, for example, C-, H-, F- and S-shaped datasets. Galton and Duckham (2006), Duckham et al. (2008) and Park and Oh (2012) all agree that the concave hull provides a much better characterisation of such datasets. Figure 2.1(a) shows activities that are superimposed on aerial photographs of two facilities where commercial activities occur: one is a courier service and the other is a shopping centre. The activities are shown without the facilities in Figure 2.1(b). The unique convex hull that is generated for each set of points is shown in Figure 2.1(c), whereas a possible concave hull for each set of points is shown in Figure 2.1(d). There are many different concave hulls (with varying degrees of concavity) that can be generated for a single set of points, and only one possible concave hull for each set of points is shown in the example. Nonetheless, one can see that the concave hull describes the distribution of points much better than the convex hull does.

"Why is that important?", one may ask. Because we would like to find a polygon that accurately describes only the points associated with it. A convex hull of one facility may overlap



(a) (b) (c) (d)

Figure 2.1: (a) Activities are superimposed on aerial photographs of facilities where commercial activities are performed (Source: South African National Geospatial Institute (NGI) imagery available on *OpenStreetMap* at location -26.005578, 28.121035 (top) and -26.114435, 27.894554 (bottom), accessed on 13 May 2013). (b) The activities are extracted in this figure. (d) The unique convex hull has been generated for each set of points. Both convex hulls seem to have the same shape, even though the distribution of points is different. (e) One possible concave hull is generated for each set of points. One can clearly see that the concave hull provides a more accurate representation of the activities.

with a convex hull of another facility, opening up a potential clash: with which facility should a point in the overlapping area be associated? Generating a concave hull for the same area will result in two separate concave hulls, one for each distinct region, without overlap.

The computation involved in generating a concave hull is more complex than that of a convex hull, therefore there are fewer concave hull algorithms to choose from. Of the few that are available, many have a high time complexity. Two concave hull algorithms have been identified as suitable candidates for implementation. Both of the algorithms are discussed in the following subsections, after which they will be compared.

2.1.1 The Park & Oh Algorithm

The Park & Oh Algorithm is a 2-dimensional concave hull algorithm that was developed as a starting point for a multi-dimensional concave hull algorithm. It has a time complexity of $O(n\log n + n)$, where n is the number of points in the cluster. This multi-dimensional concave hull algorithm is the first of its kind (Park and Oh, 2012) and it is impressive that the authors were able to develop such a simple algorithm for very complex computations. In this report, only the 2-dimensional concave hull algorithm is considered for implementation, summarised in Algorithm 1.

The first step of the Park & Oh Algorithm is to produce a convex hull for the points in the dataset, using any known convex hull algorithm. A boundary edge list is generated, called the concave list.

What the authors refer to as a *digging process* is then executed on some of the boundary edges of the hull. The digging process consists of removing a boundary edge from the concave list, and adding two new edges to the list. Consider Table 2.1 for an example of how the concave list will change after digging. Assume that there exists an edge between points a and b (edge ab) and that the algorithm has identified point c as the nearest inner point to ab. The first step of the digging process, which will be executed on edge ab towards point c, is to remove edge ab

Algorithm 1 The Park & Oh Algorithm

Input: Set of input points; threshold parameter, P
Output: List describing the edges of the concave hull

```
Generate the convex hull for the set of input points
Construct the concave list, describing the edges of the hull
Choose the threshold parameter, N
for all edges i in the concave list do
    Find the nearest inner point to edge i
    Ensure that the nearest inner point is not closer to any of i's neighbouring edges
    Calculate the length, l, of edge i
    Calculate the decision distance, d, the shortest distance from the nearest inner point to one of edge i's two vertices
    if l/d > N then
        Insert two revealed edges into the end of the concave list
        Remove edge i from the concave list
    end if
end for
return the concave list
```

from the concave list. The index of each other edge in the concave list will therefore decrement by one. Secondly, two new edges, ac and cb, are added to the end of the concave list. The digging process from ab to point c is now complete. Each edge in the concave list needs to be evaluated for digging at least once during execution of the algorithm.

Table 2.1: Concave list before and after digging takes place at edge ab towards point c

Edge	**Concave list, before**	**Concave list, after**
1	ab	bd
2	bd	de
3	de	ea
4	ea	ac
5	-	cb

A threshold parameter, P, which must be set by the user, is used in a simple calculation to determine whether the digging process will occur at a given edge or not. The threshold parameter determines whether the resulting hull is more concave or convex. Park and Oh (2012) suggest a threshold parameter between 0.0 and 5.0 to obtain valid results, and mention that a lower threshold parameter results in a more concave hull, whereas a higher threshold parameter results in a more convex hull.

Using two datasets, shaped like an A and S respectively, Park and Oh (2012) compared the performance of their algorithm to that of the Duckham Algorithm, which will be discussed in Section 2.1.2. Based on the smoothness of the boundary edges, they conclude that both algorithms perform equally well for an A-shaped dataset, and that the Duckham Algorithm performs slightly better for an S-shaped dataset. However, both algorithms use parameters to vary the concavity of the resulting hull, and it is unclear whether the authors took the difference in parameterisation into account. As Duckham et al. (2008) state, when two algorithms are being compared to each other, the difference in parameterisation must be accounted for either by finding a way to link the parameters of the two algorithms that are being compared, or by finding a way to compare all possible resulting concave hulls of the two algorithms. Park and Oh

(2012) only state which parameter values were chosen for each algorithm, but do not mention how these values were determined or why they were chosen. It is assumed that they did not take the difference in parameterisation into account when they evaluated the performance of their algorithm.

2.1.2 The Duckham Algorithm

Duckham et al. (2008) present the 2-dimensional Duckham Algorithm as a time-efficient algorithm with a time complexity of $O(n\log n)$, where n is the number of points in the cluster. It is assumed that the use of discrete mathematics in their algorithm allowed them to reach such a low time complexity. Unfortunately, it also makes the algorithm less intuitive and more complex. The Duckham Algorithm is summarised in Algorithm 2.

Algorithm 2 The Duckham Algorithm

Input: Set of input points; length parameter, l
Output: Concave hull

```
Construct the Delaunay triangulation for the set of input points
Construct the list, B, containing the set of boundary edges
Sort the list B in descending order of edge length
Initialise all vertices to false
for all edges in the triangulation do
    if the edge e is a boundary edge then
        Set its first vertex to true
        Set its second vertex to true
    end if
end for
while the boundary list is not empty do
    Set edge e ← head(B)
    Remove e from B
    if |e| > l and the resulting triangulation is regular then
        Remove edge e from the triangulation
        Insert the two new edges (that are revealed when e is removed) into B
        Set the vertices of the two new edges to true
    end if
end while
return the polygon formed by the set of boundary edges of the triangulation
```

The length parameter, l, must be set by the user and can be adjusted to change the degree of concavity of the output. It can potentially take on any non-negative real number, however, the authors suggest that l should not take on a value larger than the longest boundary edge, or a value smaller than the shortest boundary edge (Duckham et al., 2008). Larger values of l will result in a more convex polygon, while smaller values of l will result in a more concave polygon.

The first step of the algorithm is to generate the Delaunay triangulation of the set of points. The sample input points shown in Figure 2.2(a) are used to generate a Delaunay triangulation Figure 2.2(b). A Delaunay triangulation ensures that the circumcircle associated with each Delaunay triangle contains no other point in its interior

Once the Delaunay triangulation is generated, the boundary edges of the triangulation are identified and listed in descending order according to edge length. Boundary edges are removed from the triangulation as long as the edge removed is longer than the length parameter, l, and the resulting triangulation remains regular. For a triangulation to be regular, all boundary edges must have only two boundary edges associated with them. Figure 2.2(c) illustrates four concepts that determine the regularity of a Delaunay triangulation. Figure 2.2(d) shows that

if edge *ab* is removed, the triangulation remains regular, and therefore returns a *true* value when evaluated for regularity. Figure 2.2(e) shows that when edge *bc* is removed, it results in an irregular triangulation, and therefore returns a *false* value when evaluated. The algorithm continues in this fashion as long as there are more boundary edges to remove.

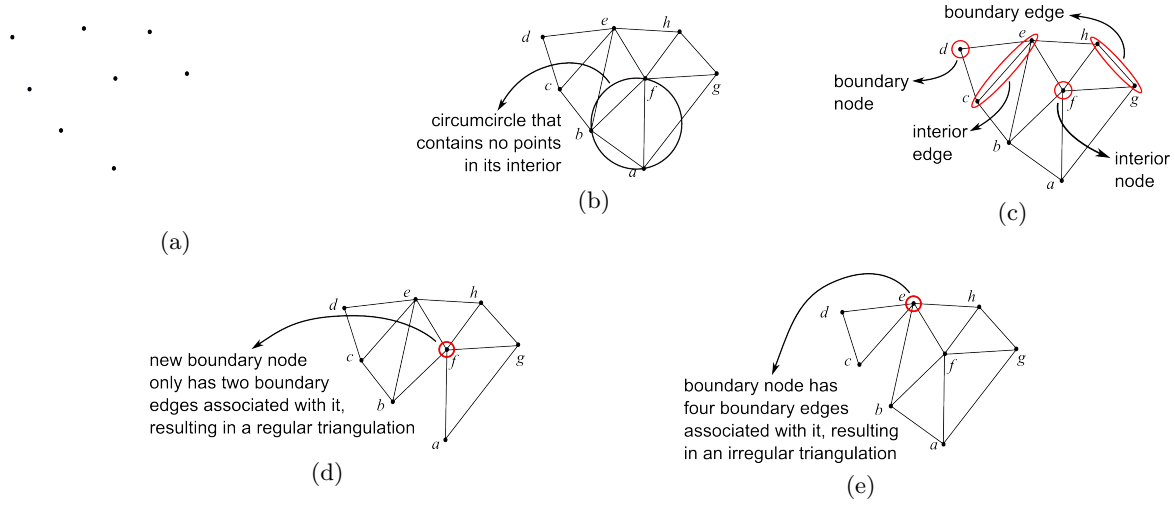


Figure 2.2: Generating the Delaunay triangulation for a set of input points, and testing for regularity using the Duckham Algorithm. Adapted from Duckham et al. (2008). (a) Input points. (b) The Delaunay triangulation for the set of input points, showing an example of the circumcircle for a Delaunay triangle. (c) Before an edge is removed from the triangulation, the algorithm tests whether the resulting triangulation will remain regular. A triangulation is irregular when a boundary node has more than two boundary edges. (d) Removing edge *ab* results in a regular triangulation, therefore it will be removed from the triangulation. (e) Removing edge *bc* results in an irregular triangulation, and the edge will remain part of the triangulation.

To objectively evaluate the performance of their algorithm, Duckham et al. (2008) generated a range of concave hulls using datasets of known distribution, such as alphabet letters and the boundary shapes of countries. The concave hulls were then evaluated to determine how well the algorithm performs. They explain that evaluating a concave hull is very subjective since there is no single correct concave hull; whether a concave hull is correct or incorrect depends largely on the application in which it is used.

The following section describes how network theory can be applied in a supply chain context, and what valuable information can be gained from a complex network.

2.2 Complex networks

Newman (2003) reviewed the structure and function of complex networks, and from this we identified applicable network analysis techniques. Borgatti and Li (2009) show why taking a complex network approach can be valuable to supply chain management. Insights from both articles are discussed in this section.

Borgatti and Li (2009) and Joubert and Axhausen (2013) explain a fundamental concept in network analysis: node centrality. It is defined as the importance of a node due to its structural position in the network as a whole. Central nodes disseminate information the fastest through a network. Depending on the type of information that needs to be released to all actors in the network, node centrality may indicate through which central nodes the information needs to be released. As an illustration of node centrality, consider the sample complex network in Figure

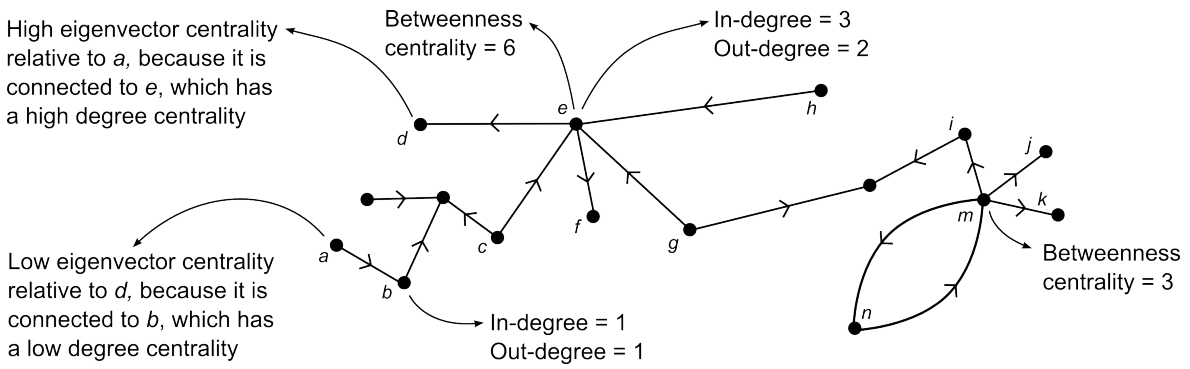


Figure 2.3: A sample complex network where the nodes represent facilities, and the edges represent commercial vehicle movement between the nodes.

2.3 and the explanations for the different types of node centralities in a supply chain context:

Degree centrality indicates how many nodes a node is connected to. Node b is connected to two other nodes. Since this is a directed complex network, the degree centrality can be divided into in-degree and out-degree. Node e has a higher degree centrality than node b, which indicates that node e has access to more information than node b. In this complex network, where directed edges indicate the movement of commercial vehicles, a high in-degree indicates that a business has many suppliers. On the other hand, a high out-degree indicates that a business has many customers.

Betweenness centrality indicates on how many shortest paths between other nodes the node occurs. Node e has a betweenness centrality = 6, because it occurs on the shortest paths between six nodes: *c-e-d*, *c-e-f*, *g-e-d*, *g-e-f*, *h-e-f*, and *h-e-d*. Node m has a lower betweenness centrality than node e. It lies on the shortest paths between three nodes: *n-m-i*, *n-m-j*, and *n-m-k*. Nodes with high betweenness centrality scores are important to the health of the entire network, because they are structurally important nodes that can control and possibly filter information flows. They can also potentially become bottlenecks that slow down the network.

Eigenvector centrality indicates whether a node is connected to well-connected nodes, and is also called eigenvalue centrality. A well-connected node in this instance means one that is connected to nodes with a high betweenness centrality. Both nodes a and d have a degree centrality = 1, however, node d has a higher eigenvector centrality relative to node a. Node d is connected to node e, which has a degree centrality = 6. Node a, on the other hand, is only connected to node b, with a degree centrality = 2.

Just as conclusions can be drawn by evaluating centrality scores individually, Joubert and Axhausen (2013) explain that certain combinations of centrality scores may also provide valuable insights in a supply chain context. Nodes with higher betweenness centrality than eigenvector centrality are considered to be *gatekeepers*, and are more likely to be well-informed in policy planning. On the other hand, nodes with higher eigenvector centrality than betweenness centrality are considered to have unique access to central actors.

Borgatti and Li (2009) mention that whole network properties have not been studied as thoroughly as node centrality has. However, some of these properties may still be valuable to this project.

Density of the network This is defined as the number of edges in the network divided by the number of pairs of nodes in the network. There is a definite lack in literature that prevents us from drawing conclusions about the density in a supply chain context, however attempts can be made to draw conclusions from the density of the network for this project.

Structural equivalence Two nodes are considered to be structurally equivalent when they have the same incoming and outgoing nodes. The implications of structurally equivalent nodes in a supply chain context is that they probably provide the same service, or sell the same product; they therefore have the same suppliers and customers. These firms will be aware of one another, use one another's performance as benchmarks and probably compete for customers. If the structurally equivalent nodes are part of the same business, it indicates redundancy within the business.

Size of the network Even though this property is not explicitly defined, the number of edges that are included in the complex network can be used as an indication of the size of the network. The larger the network, the more complete it is assumed to be.

The first step to improve the completeness of the network is to ensure that activities are associated with facilities. To achieve that, the concave hull algorithms that have been discussed in this review will be implemented in the next chapter.

Chapter 3

Implementing a concave hull algorithm

In this chapter, we compare two concave hull algorithms, and select one for integration with the clustering algorithm. In the next section we explain the methodology used to quantitavely compare the algorithms, followed by a section on each of the evaluation criteria. To conclude this chapter, we apply the chosen concave hull algorithm to Nelson Mandela Bay Municipality data.

3.1 Methodology for testing the chosen concave hull algorithms

Just as Duckham et al. (2008) evaluated the performance of their concave hull algorithm by testing it with datasets shaped like alphabet letters, the two identified concave hull algorithms are tested with C- and H-shaped datasets. These two alphabet letters were chosen due to their prominent concave distributions.

Both concave hull algorithms were implemented in *Java* to allow for easy integration with the existing complex network code. Where possible, parallelisation, also known as multi-threading, was used to reduce run time.

As Duckham et al. (2008) explain, determining which algorithm generates the "correct" concave hulls is subjective; therefore two criteria have been identified to determine which algorithm performs the best. Both criteria are considered to be equally important, and are weighted equally. Firstly, the concave hull algorithms are evaluated according to the quality of concave hulls produced. Secondly, we determine and compare the time complexities of both concave hull algorithms.

3.2 Quality of the concave hulls

This criterion measures how closely the concave hull resembles the shape of the dataset. This is determined by the degree of concavity, which is influenced by the threshold parameter. For each algorithm, multiple runs were conducted with different threshold values. C- and H-shaped datasets, each containing 1000 points, were used and can be seen in Figure 3.1. The results are deterministic, therefore only one run was conducted for each threshold value. The results for each concave hull algorithm are discussed in Sections 3.2.1 and 3.2.2, respectively.

3.2.1 The Park & Oh Algorithm

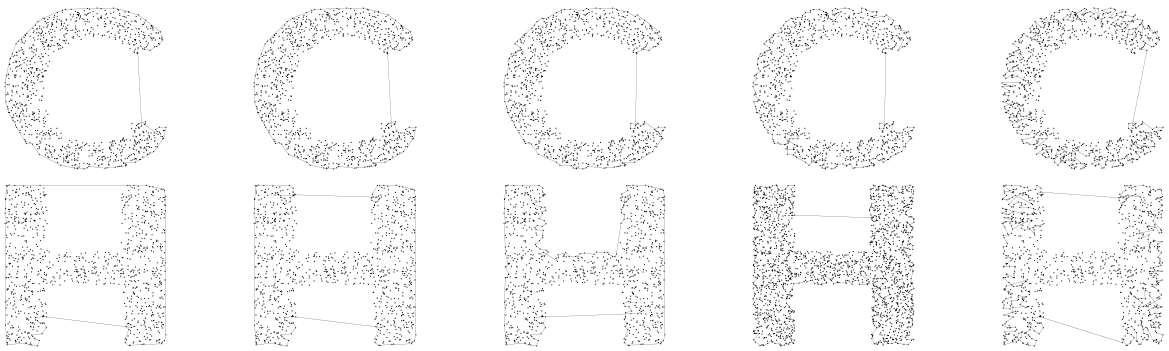
Park and Oh (2012) state that valid results are obtained when threshold parameter $P = [1.0, 5.0]$, and that a higher threshold parameter value will result in a more convex polygon. They suggest a threshold parameter of 2.0 for optimal results. For the sake of completeness it was

(a) (b)

Figure 3.1: C- and H-datasets used to test the quality of the concave hulls. Each dataset contains 1000 points.

decided to test both datasets with five different threshold parameters, $\mathbf{P} \in \{1.0, 2.0, 3.0, 4.0, 5.0\}$. These results are shown in Figure 3.2, ranging from the lowest degree of concavity to the highest degree of concavity.

The concave hulls produced by the Park & Oh Algorithm never completely resemble the shape of the datasets. The hull becomes more concave as the threshold parameter is decreased, confirming the authors' statement that the degree of concavity increases as the threshold parameter decreases. The algorithm never seems to dig into the longer edges of the concave hull: it prefers to dig on the shorter edges. The concave parts of the datasets have long edges, and digging must occur at these edges so that the resulting concave hull will resemble the shape of the dataset. It is postulated that this problem could be inherent in the algorithm, and not due to incorrect implementation.



(a) 5.0 (b) 4.0 (c) 3.0 (d) 2.0 (e) 1.0

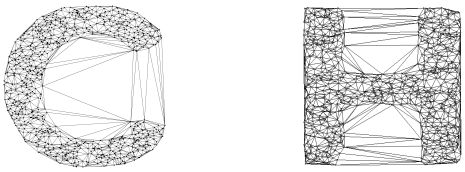
Figure 3.2: Concave hulls generated by the Park & Oh Algorithm.

3.2.2 The Duckham Algorithm

To compare the Duckham concave hulls with the Park & Oh concave hulls, the different parameterisation was taken into account (Duckham et al., 2008). Five threshold parameters were used to test the Park & Oh Algorithm, therefore five equivalent length parameters were identified to test the Duckham Algorithm.

Duckham et al. (2008) suggest a length parameter, l, between the shortest and longest boundary edges of the Delaunay triangulation. When the length parameter is longer than the longest boundary edge, no digging can occur, resulting in a convex hull. When the boundary edge is shorter than the shortest boundary edge, the degree of concavity becomes too high.

The Delaunay triangulations are shown in Figure 3.3. The shortest and longest boundary edge lengths of each Delaunay triangulation dictate the minimum and maximum length parameters that will be tested for each dataset.



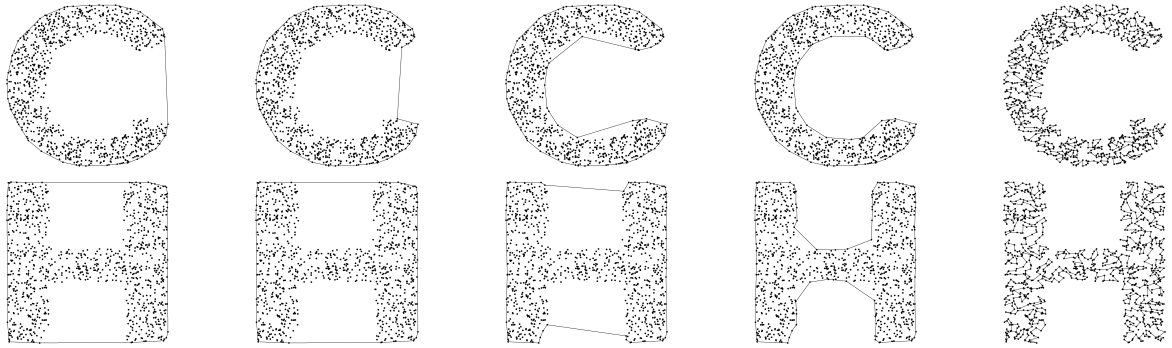
(a) (b)

Figure 3.3: Delaunay triangulations for C- and H-shaped datasets.

The shortest boundary edge of the C-shaped Delaunay triangulation is 17.8m, while the longest boundary edge is 1 020.451m. These lengths were rounded off to find the minimum length parameter, $l_C = 18$, and maximum length parameter, $l_C = 1\,020$. The three intermediate length parameters are $l_C = 268.5$, 519, 769.5.

The shortest boundary edge of the H-shaped Delaunay triangulation is 13.975m, and the longest boundary edge is 1 262.206m. Therefore we chose the minimum length parameter, $l_H = 14$, and the maximum length parameter, $l_H = 1\,262$. Based on these values, the three intermediate length parameters are $l_H = 324$, 638, 950.

The concave hulls that were produced with the sample datasets are shown in Figure 3.4. The degree of concavity increases as the threshold parameters decrease.



(a) $l_C = 1020$; $l_H = 1262$ (b) $l_C = 769.5$; $l_H = 950$ (c) $l_C = 519$; $l_H = 638$ (d) $l_C = 268.5$; $l_H = 326$ (e) $l_C = 18$; $l_H = 14$

Figure 3.4: Concave hulls generated by the Duckham Algorithm.

In Figure 3.4(a) the concave hulls produced have an extremely low degree of concavity and it seems as though these hulls are the convex hulls for the datasets. In 3.4(b) the same concave hull is produced for the H-shaped dataset, even though the length parameter is decreased to 950. The concave hulls that best resemble the shapes of the datasets are found in Figure 3.4(d), although the H-shaped concave hull can be improved by decreasing its length parameter. An extremely high degree of concavity is obtained in Figure 3.4(e), and almost every single point in the datasets are found on the boundaries of the concave hulls.

3.3 Time complexity

It is assumed that larger datasets will have a higher time complexity; however, we do not know how much greater the increase in time will be for each algorithm. The chosen algorithm will have to generate concave hulls for clusters up to 140 000 points, and it is imperative that these clusters will be dealt with timeously. Figure 3.5 illustrates how the density of the cluster is affected when the number of points in the cluster is increased.

(a) 100 (b) 500 (c) 1000 (d) 5000 (e) 10000

Figure 3.5: Datasets containing varying number of points. As the number of points in the dataset increases, the density of the cluster increases, and it is assumed that the time complexity will increase.

Park and Oh (2012) state that their algorithm has a time complexity of $O(n\log n + n)$, while Duckham et al. (2008) state that their algorithm has a time complexity of $O(n\log n)$, where n is the number of points in a cluster. These theoretical time complexities may vary from the actual time complexities obtained in this project. These differences can be accounted to different approaches used in coding the algorithms.

First we determine how the theoretical time complexities compare to each other. For this purpose, the number of activities in a cluster, n, was varied and, using the time complexities stated by each author, the corresponding time complexities were calculated. The results can be seen in Figure 3.6. For each n-value, the Park & Oh Algorithm takes n milliseconds longer than the Duckham Algorithm. This is illustrated by the lines of best fit: the Park & Oh Algorithm has a slope of 5.1307, and fares slightly worse than the Duckham Algorithm, which has a slope of 4.1307.

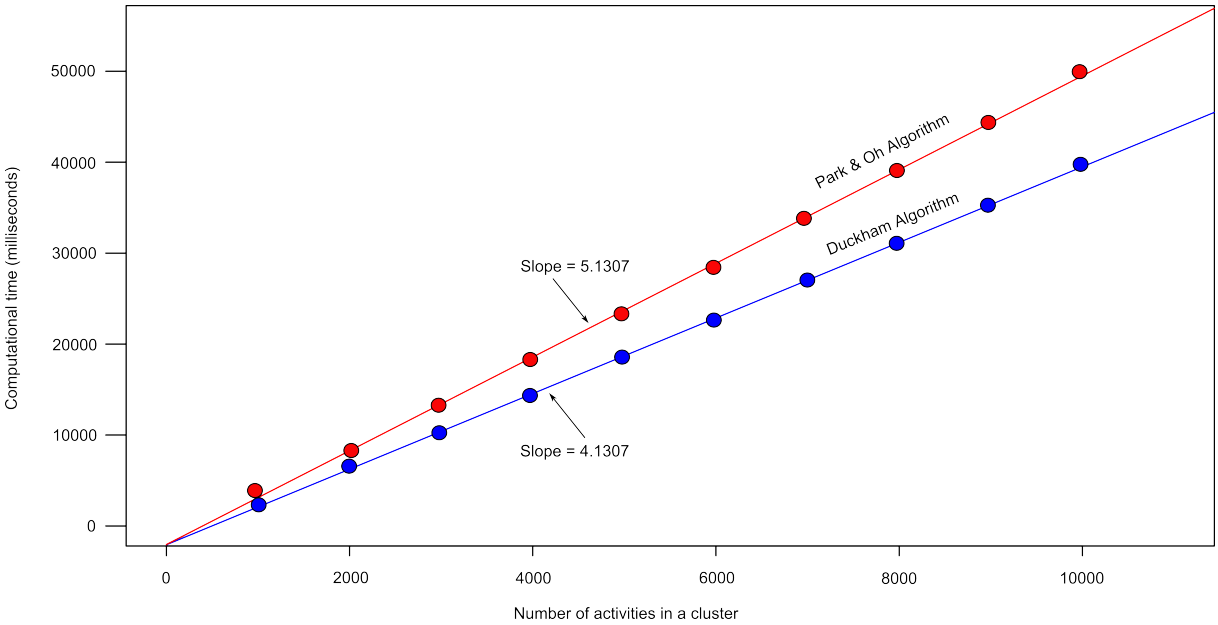


Figure 3.6: The theoretical time complexities of the Duckham Algorithm and Park & Oh Algorithm.

To determine the time complexity of the implemented algorithms, concave hulls will be generated for H-shaped datasets with the following number of points: 1 000, 2 000, 3 000, 4 000, 5 000, 6 000, 7 000, 8 000, 9 000, and 10 000. The results for this experiment are stochastic, therefore for each algorithm, 10 runs were conducted for each cluster size, and the average time for each cluster size was plotted in Figure 3.7. Both implemented algorithms have smaller slopes than those shown in Figure 3.6, indicating lower time complexities. The line of best fit of the Park & Oh Algorithm has a slope of 3.214, while the line of best fit of the Duckham Algorithm is 30 times smaller with a slope of 0.1064.

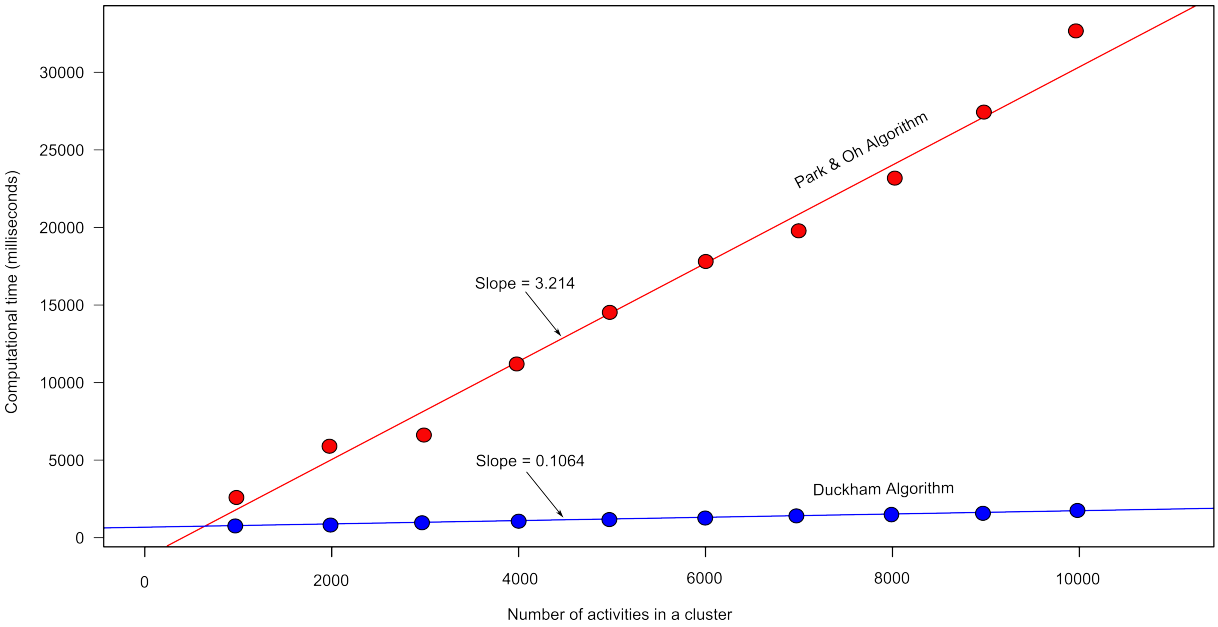


Figure 3.7: The actual time complexities of the implemented concave hull algorithms were determined using a sample H-shaped dataset with varying number of points.

3.4 Integrating the Duckham concave hull algorithm with the existing clustering algorithm

The Duckham Algorithm fares much better than the Park & Oh Algorithm on both criteria, therefore the Duckham Algorithm was implemented.

The first step of the Duckham Algorithm is to generate a Delaunay triangulation for the cluster. This implies that the cluster must contain at least three points to generate a valid Delaunay triangulation. Some clusters contain less than three points, and need to be dealt with separately. Taking this into account, Algorithm 3 is used to integrate the Duckham Algorithm with the existing clustering algorithm.

Irrespective of a cluster's size, the centroid must be determined and a *facility ID* must be assigned. If there is only one point in the cluster, that point represents the facility. When there are two points in the cluster, a line represents the facility, and the weighted average of the two points represents the centroid of the cluster. For three points, a convex hull is produced, and the weighted average of the three points is used as the centroid of the cluster. It would be redundant to pass a cluster with three points to the Duckham Algorithm, since no edges can be removed from this hull: there are three points on the boundary and no points in the interior. Only when there are four or more points in the cluster, is the cluster passed to the Duckham Algorithm. The algorithm will generate a concave hull for the cluster, after which the centroid will be determined.

Algorithm 3 Integrating the Duckham Algorithm with the clustering algorithm

Input: Set of points in cluster, C; length parameter, l
Output: A point when C contains one point; a line when C contains two points; a convex hull when C contains three points; otherwise, a concave hull

```
for all clusters generated by the clustering algorithm do
    Remove all duplicate points
    if C contains 1 point then
        the point's location is the centroid of the facility
        assign the unique facility ID to the centroid
        return the point
    else if C contains 2 points then
        calculate the centroid as the weighted average of the points
        assign the unique facility ID to the centroid
        return the line containing both points
    else if C contains 3 points then
        calculate the centroid as the weighted average of the points
        assign the unique facility ID to the centroid
        return the convex hull of the points
    elsereturn a concave hull generated by the Duckham Algorithm
    end if
end for
```

In some instances, the Duckham Algorithm could not identify a valid Delaunay triangulation for a cluster, even though there were four or more points in the cluster. Consequently, no concave hull was generated for these points. We consider these possibilities next.

3.4.1 Degenerate Delaunay triangulations

The Duckham Algorithm assumes that, for each cluster, there exists a valid Delaunay triangulation. However, for smaller clusters (fewer than 5 points), it becomes especially difficult to generate valid Delaunay triangulations.

Weatherill (1992) explains the two common degeneracies that occur in the plane:

1. **Three or more points are colinear** Colinear points occur when all the points lie on a straight line. Constructing the Delaunay triangulation for 3 or more colinear points is impossible, because the effective centre of the circumcircle(s) is at infinity.

2. **Four points lie on a circle** The Delaunay triangulation for these points is not unique, because the interior diagonal of the triangulation could be either way. Figure 3.8 shows an example of a cocircular degeneracy, where two valid Delaunay triangulations are generated for a set of points. The algorithm does not know which triangulation to choose, and thus causes an error.

In this dataset, only colinear degeneracies were identified. To deal with these degeneracies, we suggest to calculate the weighted average of the set of colinear points, and use this as the centroid for the cluster. If this solution is implemented, there will be no polygon boundary to distinguish between activities that were performed at this facility. Therefore, the location of the centroid and the *facility ID* must be assigned to the points immediately. However, this may be difficult to implement. Another solution is to simply ignore these degenerate clusters, if they are negligible.

To determine whether the degenerate Delaunay triangulations are negligible, the number of degeneracies that occurred for different combinations of clustering parameters, (ϵ, p_{min}), was

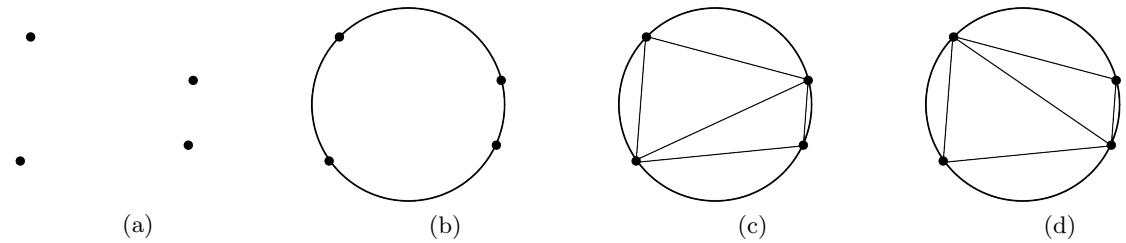


Figure 3.8: Cocircular degeneracy. (a) Input points. (b) All four points lie on the circumference of a circle. (c) The first valid Delaunay triangulation. (d) The second valid Delaunay triangulation.

determined. The bar plot in Figure 3.9 shows the percentage of clusters that are degenerate for each combination of clustering parameters. Based on these results, it is suggested that a combination's degeneracies are considered as negligible when there are less than 2% for that combination. Therefore, the only degneracies that should not be ignored occur for combinations (1, 5), (1, 10), (1, 15), (1, 20) and (1, 25).

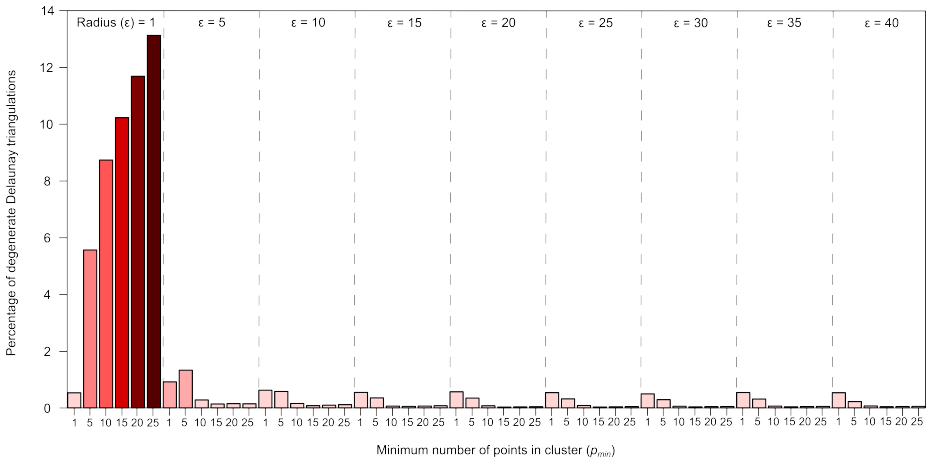


Figure 3.9: The percentage of degenerate Delaunay triangulations that occurred for each combination of clustering parameters.

3.5 Applying the Duckham Algorithm to Nelson Mandela Bay Municipality data

The Nelson Mandela Bay Municipality (NMBM) was chosen as the study area. Multiple runs were conducted for different combinations of clustering parameters: $\epsilon \in \{1, 5, 10, 15, 20, 25, 30, 35, 40\}$ and $p_{min} \in \{1, 5, 10, 15, 20, 25\}$, using a length parameter, $l = 10$.

To get a better idea of how the clusters were affected by the clustering parameters, a few facilities were plotted and are shown in Figure 3.10. More, and smaller, facilities are generated when (1, 25) is used. These facilities were merged into large facilities when (40, 25) is used. We notice that no concave hulls are generated for (1, 1), since a single point is considered to be a facility.

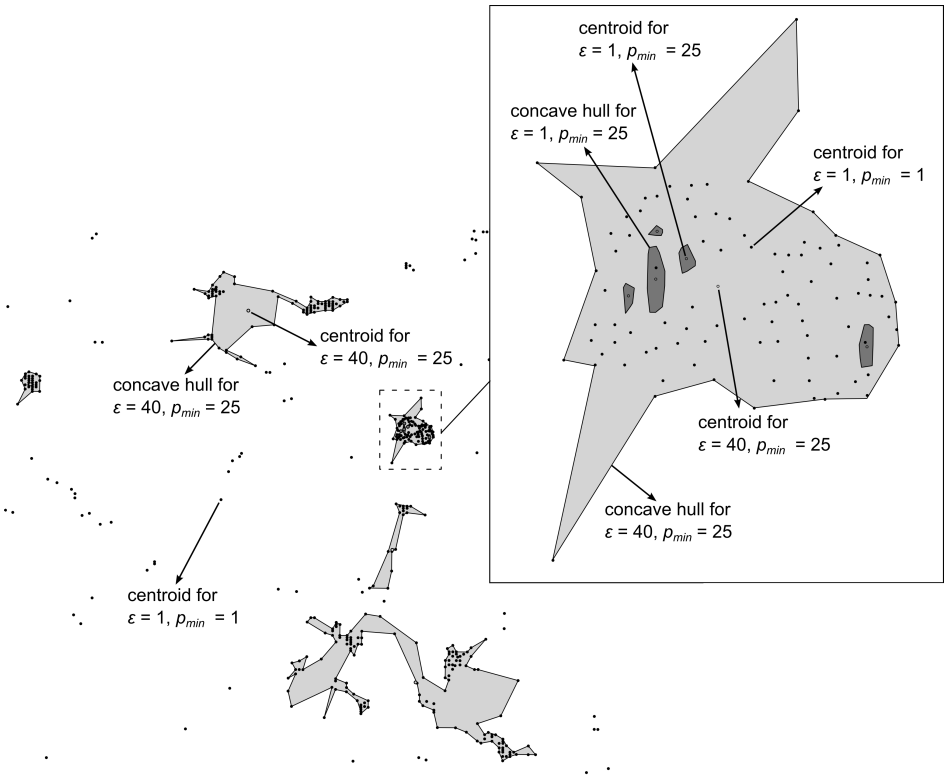


Figure 3.10: Concave hulls that are generated for activities in NMBM. This figure clearly illustrates how the size and number of concave hulls are affected by the combination of clustering parameters.

Chapter 4

Determining the relationship between the input clustering parameters and the resulting complex network

In this chapter, we conducted an experiment to determine how the clustering parameters affect the complex network. Montgomery (2009) defines an experiment as a test or series of tests in which purposeful changes are made to the input variables of a process so that we may observe and identify the reasons for changes that may be observed in the output response. There are many different strategies to use when conducting experiments, but it is suggested to use a factorial experiment when there is more than one input variable. In this project, there are two input variables, namely ϵ and p_{min}. In a factorial experiment, the input variables are varied together, as opposed to one at a time. By doing this, it is possible to not only determine how each variable individually affects the output response, but also how the *combination* of input variables affect the output response. To design the factorial experiment, we followed the design of experiments guidelines, set out by Montgomery (2009).

4.1 Recognise and formulate the problem statement

Formulating the problem statement seems like an obvious first step to follow, however it can be very difficult to formulate a statement that encapsulates the entire problem. Therefore it is suggested to formulate a list of specific questions that the experiment should address. In this way, the experimenter may get a clearer understanding of what will be studied when the experiment is conducted.

The results of this experiment need to answer the following question:

> *How does the combination of clustering parameters (ϵ, p_{min}) affect the complex network in terms of completeness and computational complexity?*

4.2 Select the response variables

As the name suggests, a response variable is an output variable that responds to changes that are made to input variables. It is important to ensure that the response variables selected provide valuable information about the system being studied.

From the problem statement, the completeness of the complex network was identified as a response variable. Computational complexity is another response variable. However, it can be defined in a number of ways, for example the time it takes to build a complex network, and

the resulting file size of the complex network. Therefore three response variables were studied: completeness, time and file size.

4.3 Choose the factors, levels and ranges

Factors refer to the input variables that may influence the performance of the system, namely ϵ and p_{min}. Process knowledge was used to choose the range and the levels over which to vary the factors. The range of ϵ is $1 \leq \epsilon \leq 40$, and was chosen based on the actual sizes of facilities where commercial activities are performed: it is assumed that activities can be performed up to 40m from each other at the same facility. Of course there may be facilities where activities are performed further away from each other, but the clustering algorithm will possibly merge all the smaller clusters that occur at one facility. The specific levels are $\epsilon \in \{1, 5, 10, 15, 20, 25, 30, 35, 40\}$. The p_{min} range is $1 \leq p_{min} \leq 25$. We want to include as many facilities as possible, therefore even facilities with only one point will be considered as a clustered facility when $p_{min} = 1$. The specific levels are $p_{min} \in \{1, 5, 10, 15, 20, 25\}$.

4.4 Choose the experimental design

Based on steps one to three, the experimenter will be able to choose the experimental design that is most suited to his or her system. This includes consideration of sample size, selection of a suitable run order for the different combinations of factors, and determining whether restrictions such as blocking, replication or randomisation are required.

We decided to run each combination of clustering parameters twice. This will enable us to estimate the experimental error and determine whether the data is statistically different. Randomisation of the runs will not occur, since we assume that there are no external factors that may influence the results.

One would normally use a two-factor-a-level design to experiment with systems that have one response and two factors that are varied over a levels. Since there are three responses to be analysed, the two-factor-a-level design is insufficient. To analyse multiple responses, multiple response surface methodology should be used. Montgomery (2009) explains that response surface methodology is a collection of mathematical and statistical techniques useful for the modelling and analysis of problems in which responses of interest are influenced by several variables, and the objective is to optimise the responses. For each response, a 3-dimensional surface was obtained, illustrating how it is affected by the factors. Simultaneously considering the responses involves building an appropriate response surface model for each response, checking for model adequacy, and trying to find a set of operating conditions that in some sense optimises all responses (or at least keeps them in desired ranges).

4.5 Perform the experiment and statistically analyse the data

It is extremely important to perform the experiment exactly as it was planned. Any errors in the experimental procedure may destroy the validity of the experiment, and a lot of resources may be wasted as a result.

For each response, we followed the same method.

1. Determine whether the data is normally distributed. If it is not, we transform the data using an appropriate transformation.

2. Choose an appropriate polynomial to fit to the data.

3. Test the adequacy of the chosen model using analysis of variance (ANOVA).

4. If the model is adequate, we generate the response surface.

To optimise the responses, we overlaid the contour plots of all three response surfaces. This enabled us to determine which combination of clustering parameters results in the most desirable complex network. This experiment was performed using Design-Expert® software. Please refer to Appendix A for the complete computer output for the model.

4.5.1 Completeness of the complex network

This response is measured by the number of nodes. For clustering parameters (1, 25) the least complete network was obtained: it contained 1 708 nodes. It makes sense that this combination resulted in the least complete network. The clustering parameters require that a minimum of 25 points fall within a 1 m radius to form a cluster. The probability of this occurring is low, as reflected by the small number of nodes in the complex network. Using clustering parameters (1, 1) resulted in the most complete complex network, which contained 382 317 nodes. The ratio of minimum to maximum completeness is 223.839. When the ratio is greater than 10, it is suggested to use a transformation to normalise the data. Data transformation is often needed to meet assumptions that make the ANOVA valid.

To choose the appropriate transformation, we applied the Box-Cox transformation method, as explained by Osborne (2010). Box-Cox provides a family of transformations that will optimally normalise a particular variable, eliminating the need to randomly try different transformations to determine the best option. Most data transformations can be described by the power function, $\sigma = \mu^{\alpha}$, where σ is the standard deviation, μ is the mean, and α is the power. The Box-Cox coefficient, λ, is calculated as $1 - \alpha$ and is used to determine which transformation is required. In Figure 4.1(a) we plot ln(residuals) against possible λ values. The lowest point on the graph corresponds to $\lambda = 0$, and the Box-Cox method recommends a natural log transformation for this λ.

Since we only have a limited number of data points, we need to interpolate between them to generate the response surface. The sequential model sum of squares is used to predict which model should be fitted to the data. We want to choose the highest order polynomial that has an F-probability < 0.05. When the F-probability satisfies this condition, it suggests that the additional terms that have been added to the model are significant (based on a confidence interval of 95%). Based on the F-values generated for different polynomials, we used a fifth-order polynomial.

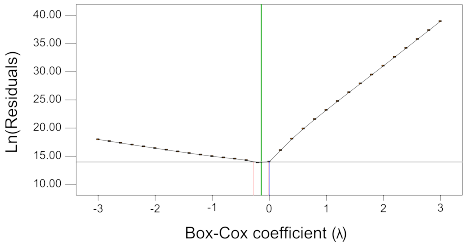
ANOVA gives an indication as to whether or not the selected fifth-order model is statistically significant. ANOVA assumes that the errors are normally and independently distributed with mean zero and constant variance, σ^2. For this response, the model F-value is 2 470.93 and the p-value is < 0.0001. This means that there is only a 0.01% chance that an F-value this large could occur due to noise. Therefore we conclude that the model is significant and can be used to describe the completeness response surface.

R^2 is the proportion of variability that is explained by the ANOVA model, and adjusted R^2 values reflects the number of factors in the model (Montgomery, 2009). For the completeness response, the R^2 and adjusted R^2 values are 0.9993 and 0.9989, respectively. The predicted R^2 is a measure of how well the model predicts a response value and is equal to 0.9968, which is in reasonable agreement with adjusted R^2. Therefore, ANOVA can be used to explain the variability in this model.

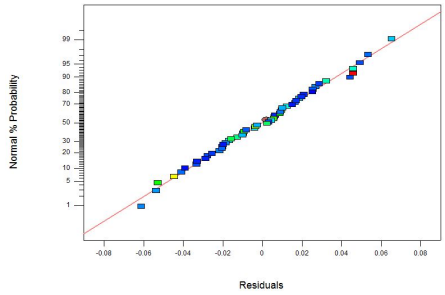
To test the validity of the assumptions made in the ANOVA, a normal probability plot of residuals is used to determine whether the underlying error distribution is normal. If this is the case, the plot will resemble a straight line, indicating that, for each combination of clustering parameters, the completeness of the complex network does not deviate from the mean. Figure 4.1(b) shows the normal probability plot for completeness. The points follow a linear trend, and there are no outliers. Therefore we can conclude that the underlying error distribution is

normal.

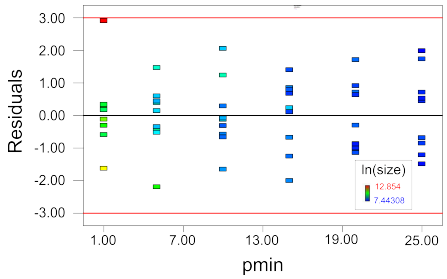
The residuals plot in Figures 4.1(c) and 4.1(d) should be completely structureless, indicating that there is no external factor influencing the data.



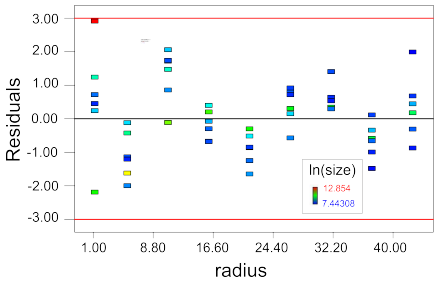
(a) Box-Cox plot for completeness



(b) Normal probability plot of the residuals.



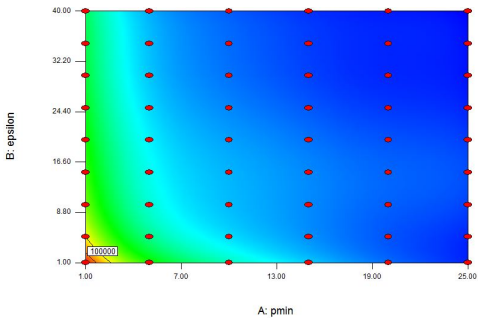
(c) Residuals for the design points are plotted against the p_{min} values.



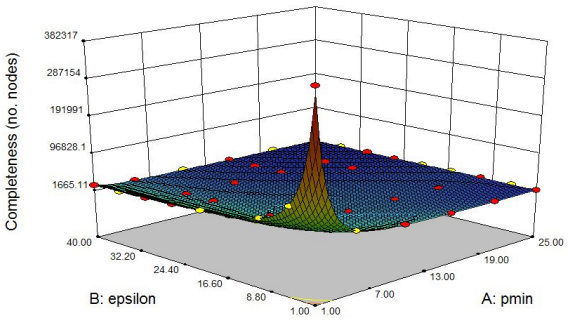
(d) Residuals for the design poi nts are plotted against the ϵ values.

Figure 4.1: Determining model adequacy for the completeness response surface model.

The response surface for completeness of the complex network was generated using the suggested fifth-order polynomial. The 2-dimensional contour plot and the 3-dimensional response surface are shown in Figure 4.2. The blue area of the plot indicates less complete complex networks. The most complete complex networks are represented by the red area of the response surface. For most of the clustering parameters, the response surface is blue. If we want to optimise the completeness of the complex network, we should avoid all combinations of clustering parameters that correspond to these areas. The combination of clustering parameters that results in the most complete complex network is (1, 1), as even the most isolated single node is considered a cluster.



(a) Two-dimensional contour plot.



(b) Three-dimensional response surface.

Figure 4.2: Response surface for completeness of the complex network.

4.5.2 Build time of the complex network

The longest build time occurred with clustering parameters (20, 40), and was 1 442.58 seconds. This makes sense because more computations are needed with large clustering parameters: many clusters need to be merged, and this takes time. The complex network that had the shortest build time only took 792.78 seconds, and was obtained with clustering parameters (1, 40). This build time is low because there exists very few clusters that contain a minimum of 40 points within a 1m radius. This results in a minimum to maximum ratio of 1.81965. Since the ratio is below 10, we assume that the data is normally distributed and that no transformation is required.

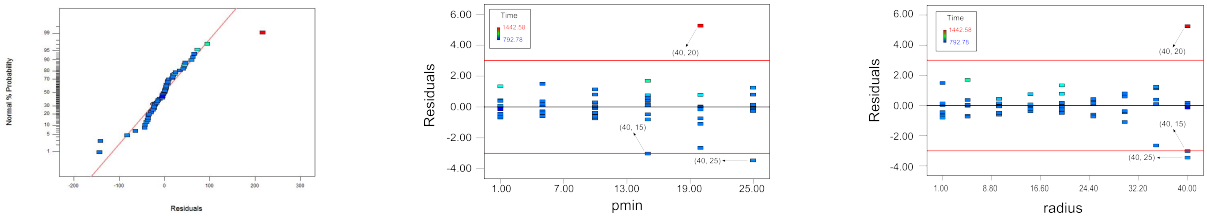
Similar to the completeness response model, we need to identify a suitable polynomial that can be used to interpolate the time data points. The highest order polynomial that has an F-probability < 0.05 is the fifth-order polynomial. The F-value is 0.0481, which indicates that the additional terms that are added to the model are significant.

The ANOVA for time gives an indication as to whether or not the selected model is statistically significant, based on a confidence interval of 95%. For the fifth-order model, the F-value is 2.27 and the p-value is 0.0179. This means that there is only a 1.79% chance that an F-value this large could occur due to noise, indicating that the model is significant.

R^2, which is the proportion of variability explained by the ANOVA model, is equal to 0.5789. Adjusted R^2 reflects the number of factors in the model: the more factors in the model, the more this value deviates from R^2. For this response, adjusted $R^2 = 0.3237$. This is quite low, and indicates that only 32.37% of the variation is explained by the model. The predicted R^2, which is a measure of how well the model predicts a response value, is equal to -2.0157. The negative value indicates that the overall mean is a better predictor of the response values than the current model.

The normal probability plot of residuals is shown in Figure 4.3(a). The points follow a linear trend, with the exception of three outliers. These outliers were generated for these combinations: (40, 20), (40, 15) and (40, 25). These points all have large clustering parameters, for which more computations are needed. When more computations need to be done, the computational time is increased, causing these points to vary greatly from the mean time. Usually, outliers should be removed from the design space and a new model should be built for the response. However, when these outliers are removed, the ANOVA indicates that the model is insignificant. Therefore it was decided not to remove any of the outliers.

Figures 4.3(b) and 4.3(c) show the residuals plot for each factor. Residuals should lie within the red confidence intervals and should be unstructured. This is indeed the case, indicating that there is no external factor affecting the data.



(a) Normal probability plot of residuals for the time response

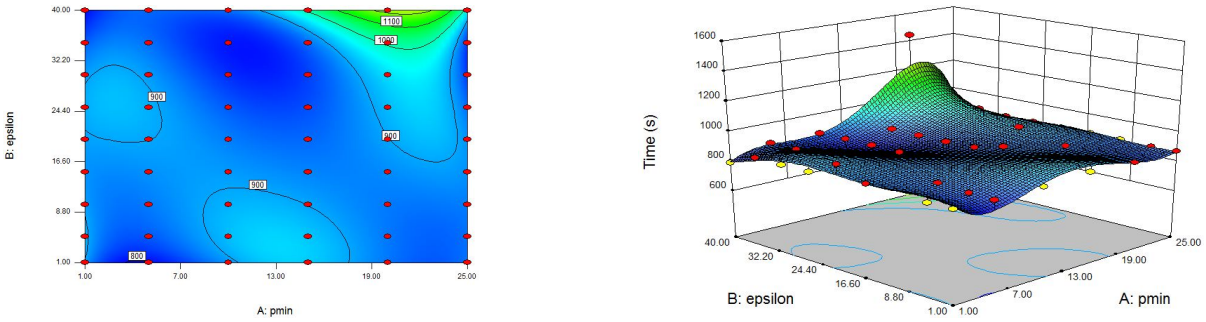
(b) Residuals for the design points are plotted against the p_{min} values.

(c) Residuals for the design points are plotted against the ϵ values.

Figure 4.3: Determining model adequacy for the time response surface model.

The 3-dimensional response surface for time and its corresponding contour plot is shown in Figure 4.4. The blue areas of the plot indicate the combinations of clustering parameters parameters that result in short build times. Since we would like to minimise the time it takes

to build a complex network, we should choose clustering parameters that coincide with the blue parts of the plot. The green areas of the plot indicate long build times, and should preferably be avoided.



(a) Two-dimensional contour plot.

(b) Three-dimensional response surface plot.

Figure 4.4: Response surface for build time.

4.5.3 File size of the resulting complex network

This response is measured in kilobytes[1]. The largest file size obtained was 32 212 kB, and was created using clustering parameters (1, 1). This combination of clustering parameters also resulted in the most complete complex network, containing many nodes and edges that are written to file. The smallest file size obtained was 472 kB, and was created using clustering parameters (1, 25). These clustering parameters require that a minimum of 25 points need to be performed within a radius of 1m. The probability of this occurring is low, therefore fewer clusters are created, resulting in a less complete complex network and a smaller file size. The ratio of minimum to maximum file size is 66.0976. The ratio is larger than 10, indicating that the data points are not normally distributed.

Again we use the Box-Cox transformation method to determine which transformation to use. In Figure 4.5(a) we plot ln(residuals) against possible λ values. The lowest point on the graph corresponds to $\lambda = -0.5$. Based on the Box-Cox transformations, an inverse square root transformation should be used for this λ.

As we did with the previous two responses, it is necessary to identify an appropriate polynomial to which the file size data can be fitted. The sequential model sum of squares indicates that a fifth-order polynomial has an F-probability equal to 1.553×10^{-8}, suggesting that the additional terms are significant.

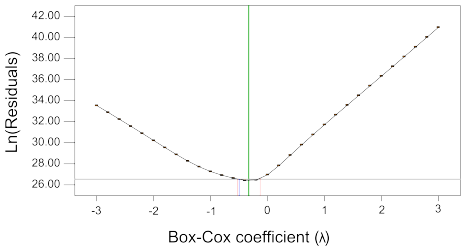
The ANOVA gives an indication as to whether or not the selected fifth-order polynomial is statistically significant, based on a confidence interval of 95%. The model F-value is 591.05 and the p-value is < 0.0001. This means that there is only a 0.01% chance that an F-value this large could occur due to noise. Therefore the model is significant.

The R^2 and adjusted R^2 values for this model are 0.9972 and 0.9955, respectively. The predicted $R^2 = 0.9930$, which is in reasonable agreement with the adjusted R^2. Based on these values, we can conclude that 99.55% of variability in the model is explained by the ANOVA.

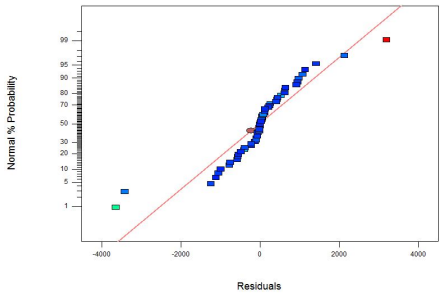
The adequacy of the ANOVA is tested using a normal probability plot of the residuals. This plot is shown in Figure 4.5(b). The points follow a linear trend, and there are two outliers. These outliers were generated for combinations (10, 25) and (10, 20), indicating that the sizes of these files deviate from the mean file size. Usually, outliers should be removed from the design space and a new model should be built for the response. However, when these outliers are removed, the ANOVA indicates the model to be insignificant. Therefore it was decided not to remove the outliers.

[1]Where 1 kilobyte = 1024 bytes.

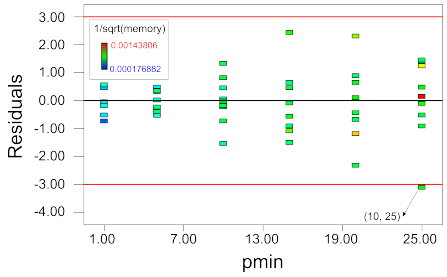
The residuals for each factor are shown in Figures 4.5(c) and 4.5(d). Residuals should lie within the red confidence intervals and should be unstructured. All points do, with the exception of one outlier: (10, 25).



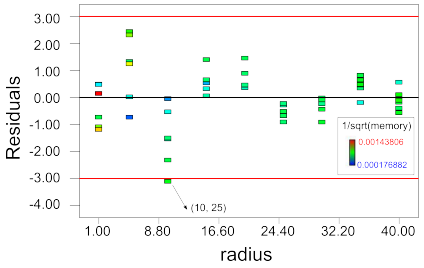
(a) Box-Cox plot for file size



(b) Normal probability plot of file size residuals.



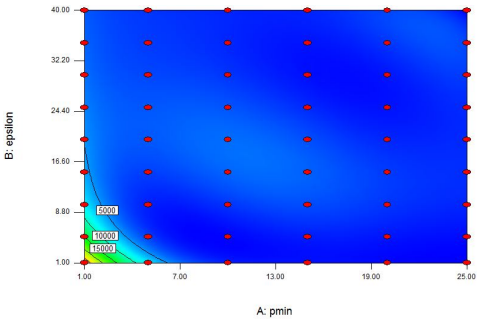
(c) Residuals for file size data are plotted against the p_{min} values.



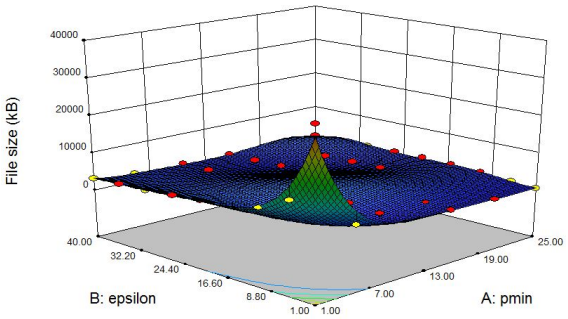
(d) Residuals for file size data are plotted against the ϵ values.

Figure 4.5: Determining model adequacy for the file size response surface model.

The response surface for file size, and its corresponding contour plot, are shown in Figure 4.6. The green areas of the plot indicate that a reasonable file size was generated. The objective for file size is to minimise it, so the red area of the surface should preferably be avoided.



(a) Two-dimensional contour plot.



(b) Three-dimensional plot.

Figure 4.6: Response surface for file size.

4.5.4 Optimising the responses

Optimising all three the responses simultaneously was done by overlaying the response surface plots, and finding the combination of clustering parameters that satisfies the objectives of the responses: maximise completeness, minimise time and minimise file size. In conjunction with overlaying the contour plots, desirability functions were used to determine which solution is the most desirable. A desirability function for a response varies between 0 and 1, and indicates to what extent the responses are optimised (Montgomery, 2009). A desirability function of

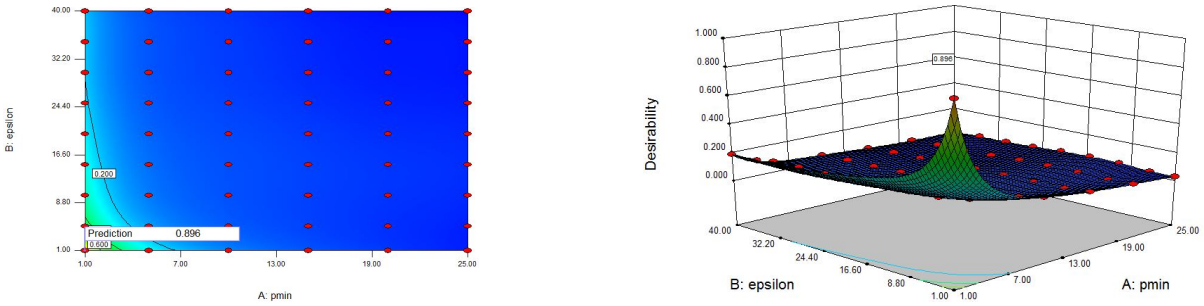
1 indicates that the response has reached its target value, while a desirability function of 0 indicates that the response is outside the acceptable region.

For each factor and response, the limits within which the solution should lie, should be selected. There is no way of knowing where the optimal solution lies because this is the first round of optimisation. Therefore the factor limits are chosen to be the same as those that were used when designing the experiment, and the limits of the responses are simply the limits of the response surface plots. If more rounds of optimisation are done, these limits could be adjusted to find a solution with a higher desirability.

Six solutions were found that optimise the responses simultaneously, and are shown in Table 4.1. Solution 1 is the most desirable, with a desirability value of 0.594. This is not a very high desirability, indicating that the optimal combination of clustering parameters may lie outside the limits that have been tested. The contour overlays are shown in Figure 4.7.

Table 4.1: Solutions generated when overlaying the contour plots

Solution	ϵ (m)	p_{min}	Completeness	Time (s)	File size (kb)	Desirability
1	1	1.8	178 665	848.455	16 048	0.594
2	1	1.69	194 986	853.754	17 285	0.593
3	2.62	1	196 587	906.908	17 201	0.577
4	17.34	1	31 082.1	890.85	58 570	0.378
5	24.87	1	22 808.2	905.231	44 072	0.342
6	36.36	1	17 433	836.766	34 417	0.326



(a) Two-dimensional overlay.

(b) Three-dimensional overlay.

Figure 4.7: Overlaying the contour plots to simultaneously optimise all three responses.

4.6 Conclude with recommendations

By overlaying the contour plots, the optimal combination of clustering parameters was identified: $\epsilon = 1$, $p_{min} = 1.8$. This solution was chosen as the best based on its desirability value of 0.594. However, this means that only 59.4% of the multiple objectives are satisfied. Even though this solution has the highest desirability, there may be an issue with the suggested p_{min} value. In the clustering algorithm, p_{min} is measured in number of points, which is a natural number. To overcome this, it is suggested to use clustering parameters $\epsilon = 1$, $p_{min} = 2$.

Chapter 5

Multi-objective optimisation of size, time and file size

In multi-objective optimisation, there is not one optimal solution, but a range of efficient solutions, called the efficient frontier. Rardin (1998) explains that an efficient solution cannot be the best if there are other solutions that are equally good or better. He defines an efficient point as a feasible solution to a multi-objective optimisation model if no other feasible solution scores at least as well in all objective functions and strictly better in one. Efficient points are also called Pareto optimal and nondominated points. Graphically, efficient points are those feasible points for which no distinct feasible point lies in the region bounded by the contours of the objective functions through the point, which contains every solution with equal or superior value of all objectives. The collection of efficient points for any multi-objective optimisation model is the efficient frontier for that model.

There are many ways to evaluate multi-objective problems. Rardin (1998) explains that one of these methods is to determine the efficient frontier by parametrically varying specified levels of all but one objective, while optimising the others.

Alternatively, one could use goal programming to solve a multi-objective problem. Rardin (1998) explains that goal programming models are constructed in terms of goals to achieve, rather than quantities to be minimised or maximised. Stewart (2007) states that this is a powerful tool when there is a need to model conflicting objectives.

Since there is data available about the complex networks, it is unnecessary to formulate a multi-objective problem and solve it using a linear programming package. Instead, we evaluate this multi-objective problem by constructing the efficient frontier.

5.1 Constructing the efficient frontier

In this project there are three objectives to be optimised, and these are split into two separate formulations, each with two objectives. For the first formulation, the completeness objective is optimised and the file size objective is varied over specified levels as a constraint. In the second formulation, the completeness objective is optimised, and the time objective is varied over specified levels as a constraint.

For the first formulation, we begin by optimising the completeness of the complex network, without regard for file size. The largest complex network is built using clustering parameters, (ϵ, p_{min}), (1, 1), and is seen in the first entry (382 317, 31 212) in Table 5.1. Next, file size is optimised without any regard for size. This solution is found when (1, 25) is used, and can be seen in the last entry (2 070, 482) in Table 5.1. The first and last entries in the table specify the constraints of the multi-objective problem: ϵ can only be equal to 1, and p_{min} can be varied between 1 and 25. The file size was varied over the levels specified in the "File size objective"

column, and the corresponding completeness, that satisfies the efficient point constraints, was determined. The efficient points are shown in Figure 5.1(a).

Table 5.1: Efficient frontier of the complex network for the completeness versus file size

Completeness objective	File size objective (kB)	Efficient point	
		ϵ	p_{min}
382 317	31 212	1	1
27 917	4 882	1	5
9 879	1 953	1	10
5 129	1 464	1	15
3 113	976	1	20
2 070	**472**	1	25

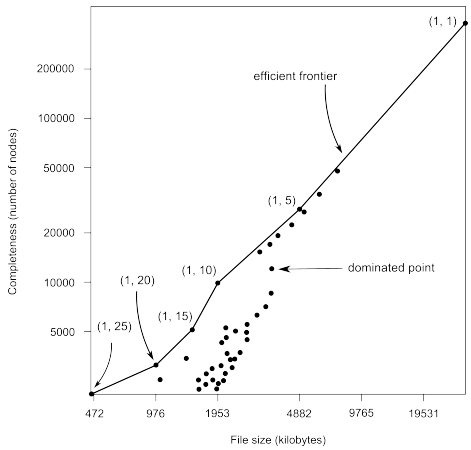
For the second formulation, we begin by optimising the completeness of the complex network, without regard for time. The largest complex network is built using clustering parameters (1, 1), and is the first entry (382 317, 876.22) in Table 5.2. Next, time is optimised without any regard for completeness. This solution is found when (40, 1) is used, and is the last entry (15 675, 792.78) in the table. The first and last entries in the table specify the constraints of the multi-objective problem: ϵ can be varied between 1 and 40, and p_{min} can only be equal to 1. To determine the efficient points, time was varied over the levels specified in the "Time objective" column, and the corresponding completeness, that satisfies the efficient point constraints, was determined. The efficient points are shown in Figure 5.1(b).

Table 5.2: Efficient frontier of the complex network for completeness versus time

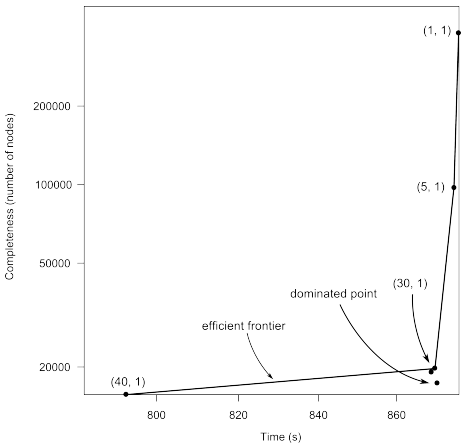
Completeness objective	Time objective (s)	Efficient point	
		ϵ	p_{min}
382 317	876.22	1	1
98 009	875	5	1
19 728	870	30	1
15 675	**792.78**	40	1

The efficient points shown in Tables 5.1 and 5.2 were plotted in Figure 5.1. All points that lie on the efficient frontier are efficient solutions to the multi-objective problem. In Chapter 4, the combination of clustering parameters, (ϵ, p_{min}), that is suggested as the optimal solution, is (1, 1.8). However, the efficient frontiers show that there are a range of clustering parameters, all of which satisfy the multiple objectives.

In Chapter 6, one combination of clustering parameters is chosen, and network analysis is performed on the resulting complex network.



(a) Completeness is maximised and file size is minimised.



(b) Completeness is maximised and time is minimised.

Figure 5.1: The efficient frontiers for completeness, time and file size.

Chapter 6

Network analysis

Based on the recommendation made in Chapter 4, clustering parameters (1, 2) were used to build a complex network in NMBM and perform network analyses. A total of 3 590 vehicles performed at least 60% of their activities in NMBM, and these vehicle's activity chains were used to build the complex network. The resulting complex network has 14 116 nodes, and 38 034 directed edges, resulting in a density of 0.018%. The maximum edge weight was 432, relating to approximately 2.4 trips per day between the two facilities over the six-month period.

6.1 Degree distributions

It was earlier stated that the degree of a node in a network is the number of edges connected to it (Newman, 2003). The degree distributions of nodes often follow a power law function (Joubert and Axhausen, 2013; Newman, 2003). To determine whether this is the case, the same approach used by Joubert and Axhausen (2013) was followed. The in-degree distributions of the NMBM complex network were fitted against a power law function in Figure 6.1. The power law R^2 = 0.902, which indicates that 90.2% of the in-degree distribution can be fitted to a power law function. The out-degree results were very similar.

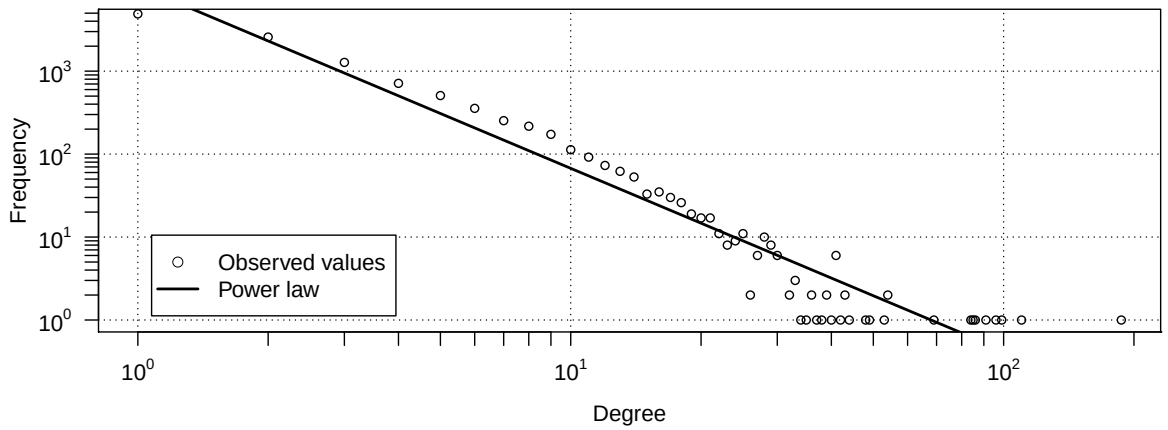


Figure 6.1: Degree distribution power law fit.

6.2 Identifying key players in the complex network

Three centrality scores were considered for the network analysis: weighted degree centrality (both in- and out-degrees), eigenvector centrality and betweenness centrality. Summary statistics

of these centrality scores can be seen in Table 6.1.

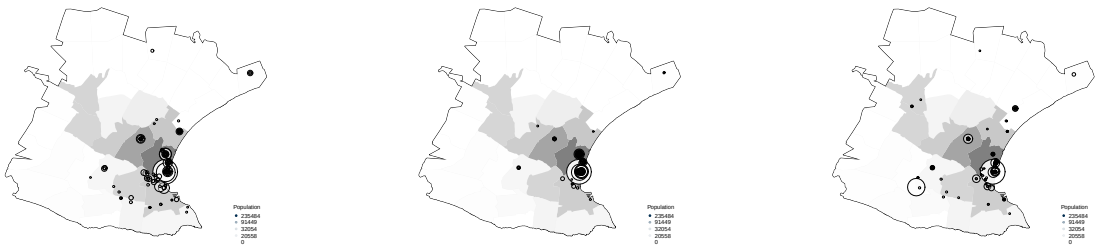
Table 6.1: Network statistics

Centrality	Mean	Mode	Std. dev.	Min.	Percentile			Max.
					25^{th}	50^{th}	75^{th}	
Degree[1]	6.753	1	24.350	1	1.000	3.000	6.000	1668
Degree[2]	5.074	1	8.333	1	1.000	3.000	5.000	338
Betweenness	40558	0	207 465	0	0.000	42.000	17 786	10.6×10^6
Eigenvector	0.012	0	0.029	0	0.001	0.002	0.010	1

Following the approach of Joubert and Axhausen (2013), the facilities with the top 200 centrality scores were plotted on a map of NMBM in Figure 6.2. The population of NMBM is represented by the shaded regions: the darker the region, the higher the population. The size and transparency of the markers relate to the centrality score: the larger and darker the marker, the higher the score.

For all three centralities, the distribution of key players coincides with areas where the population density is the highest. This is similar to the results found by Joubert and Axhausen (2013): key players in their Gauteng complex network were situated in the centre of Gauteng, where the population density is the highest. Key players are often situated on the periphery of urban areas, but due to the unique socio-economic landscape of South Africa, this is not the case here. During the Apartheid years, rural areas were created around the periphery of urban areas. As a result, businesses were established in the centres of urban areas. The high density of key players in the centres of urban areas increases the movement of commercial vehicles in these areas, and greatly contributes to congestion.

Some key players scored highest on more than one centrality score. An overnight truck stop, for example, scored especially high on all three centralities. Another key player that scored high on all three centrality scores is a refueling station. As Joubert and Axhausen (2013) argue, one might not regard a refueling station as an important facility to consider in network analysis, but these facilities can be considered when implementing weigh bridges or radio-frequency (RF) readers.



(a) Betweenness centrality (b) Eigenvector centrality (c) Weighted degree centrality

Figure 6.2: Spatial distribution of the key players for different centrality scores, adapted from Joubert and Axhausen (2013).

[1]weighted
[2]not weighted

6.2.1 Relationships between centrality scores

Theoretically, the eigenvector centrality should be an approximately linear function of the betweenness centrality (Joubert and Axhausen, 2013). Any facilities that do not follow this trend are considered to be outliers, and are of particular interest in this network analysis. In Figure 6.3(a) the eigenvector and betweenness centrality scores of all the facilities are plotted. The darker the marker, the greater the deviation from the linear distribution, and the higher the absolute value of the residual.

The facilities with a higher eigenvector than betweenness centrality are considered to have *unique access* to key players in the network. When key players cannot be reached through other methods, the facilities with *unique access* should be targeted to reach the key players in the network. The facilities with higher eigenvector than betweenness centrality are considered to be *gatekeepers*, since they are connected to many other facilities.

The *facility ID*s of the ten facilities with the highest residual values were identified and are shown in Figure 6.3(a). These facilities' geographic locations were determined and plotted over a map of NMBM in Figure 6.3(b). Some of these facilities were situated extremely close to one another due to the small ϵ value that was used in the clustering parameters.

The facilities were identified, and it was determined that the three distinct groups of facilities formed part of three businesses. Therefore the activities of one business were clustered into separate facilities in the complex network. We could argue that this does not matter because the separate facilities in the complex network may represent separate functions of a business, such as shipping and receiving. Since we do not know the purpose of the movements between facilities, it is difficult to determine whether clustering separate facilities for each business is correct. By analysing the exact locations of the facilities, it was determined that they are found within a few metres from each other, and not on two opposite ends of the business (as one might expect from shipping and receiving activities). Even though the completeness of the complex network is very high, the network does not logically represent reality.

(a) (b)

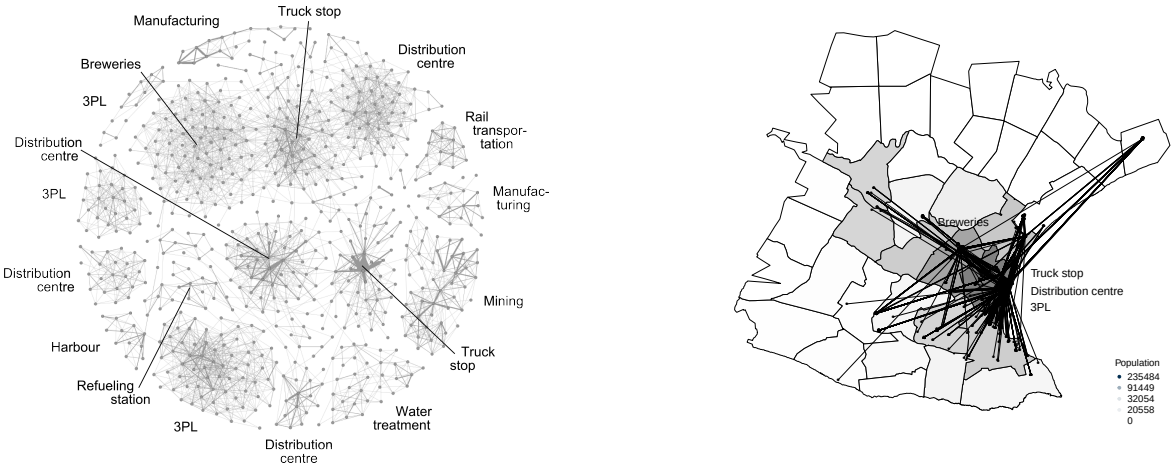
Figure 6.3: Identifying key players using linear residuals of eigenvector and betweenness centralities.

6.2.2 Cohesive subgroups

To filter the complex network, the 95^{th} percentile of the edge degree was used. This relates to a weighted degree centrality of 22 or more. Of the directed edges that remained, some started and ended at the same node. This occurs when a commercial vehicle moves between two points

at a large facility. Consequently, two consecutive activities in the commercial vehicle's activity chain, have the same *facility ID*. To prevent this from occurring, these activities should have been merged before building the complex network.

Boccaletti et al. (2006) defines a cohesive subgroup as a group of nodes such that there is a higher density of edges within groups than between them. This is the case in Figure 6.4(a), where 17 subgroups are shown. We notice that two truck stops and a refueling station has been identified as a subgroup. Even though this is not classified as a business where commercial activities take place, we cannot simply ignore the importance of these facilities.



(a) (b)

Figure 6.4: (a) Cohesive subgroups are extracted from the complex network based on their weighted degree. (b) Four of these subgroups were located and plotted over a map of NMBM.

We extracted the four subgroups with the highest densities, and located them on a map of NMBM in Figure 6.4(b). We did not include the edge's weights or directions, since we are only interested in the locations of the subgroups. Three of the four subgroups are situated in the most densely populated area of NMBM. This may have a number of disadvantages to the population in the area, such as damage to roads and congestion during peak times. However, the locations provide advantages to the businesses in terms of accessibility: they are located close to the harbour and major roads. We do not know whether these businesses make use of the harbour for importing and exporting purposes, but if they do, their locations allow them to save money on transport to and from the harbour.

Chapter 7

Conclusion and recommendations

In this project we identified two shortcomings of the clustering approach used by Joubert and Axhausen (2013).

7.1 Increasing the accuracy of adapting activity locations

Firstly, a search radius was used to approximate at which clustered facility an activity was performed. In some instances the search radius either assigned an activity to a wrong cluster, or it could not assign the activity to a cluster at all.

We proposed to overcome this shortcoming by generating a concave hull for each cluster. Each concave hull forms the boundary of a cluster, increasing the accuracy of assigning activities to clusters: one only needs to determine whether an activity falls within, or on, the boundary of a concave hull. The Duckham Algorithm (Duckham et al., 2008) and the Park & Oh Algorithm (Park and Oh, 2012) were compared using two criteria: the quality of the concave hulls generated, and the time complexity of the algorithm. The Duckham Algorithm outperformed the Park & Oh Algorithm in both resembling the desired output and computationally.

The Duckham Algorithm assumes that all datasets have valid Delaunay triangulations. For the data used in this project, this was not always the case. Degeneracies occurred when three or more points in a cluster were colinear, and consequently no valid Delaunay triangulations could be generated for these clusters. The degeneracies were assumed negligible when there were less than 2% degeneracies for a combination of clustering parameters. All combinations with $\epsilon = 1$m had more than 2% degeneracies; the rest of the combinations of clustering parameters produced a negligible amount of degeneracies.

7.2 Considering completeness of the complex network

The second shortcoming of the approach followed by Joubert and Axhausen (2013) was their lack of consideration of the completeness of the complex network when they evaluated the clustering parameters.

We recognised that by addressing this shortcoming, we were faced with a multi-objective problem: we could not maximise the completeness of the complex network without making a trade-off with computational complexity. To address this problem, multiple response surface methodology and multi-objective evaluation was used. By overlaying the response surface plots from the multiple response surface experiment, we found the combinations of clustering parameters that optimise the complex network in terms of completeness, time and file size. For the multi-objective evaluation we constructed two efficient frontiers, which illustrated that there are multiple combinations of clustering parameters that optimise the objectives.

Based on the results, we chose clustering parameters (1, 2) to use for building a commercial vehicle complex network in the Nelson Mandela Bay Municipality. We found that the in-degree

distributions of the complex network follows a power law distribution. Similar to Joubert and Axhausen (2013), we found that the majority of key players in the complex network are located in the most densely populated areas. We attributed this to South Africa's unique socio-economic landscape.

7.3 Recommended research

In order to gain more insight into the commercial vehicle complex network in the Nelson Mandela Bay Municipality, we recommend that more network analyses be done on this network.

Further research needs to be conducted to determine whether a more complete complex network is a more accurate representation of reality. An important factor considered in this project was the completeness of a complex network. If more clusters were formed, more concave hulls were generated, and this increased the completeness of the complex network. However, complex networks with more concave hulls were, in some cases, the result of actual facilities being clustered as separate concave hulls. One could argue that it does not matter that multiple concave hulls are produced for one facility: at least all of the activities of the facility are accounted for somewhere in the complex network. If a complex network were to be used to identify key players in a supply chain, it is important to model a facility as one node in the complex network. If multiple nodes represent the facility, it will be less central in the network because the facility's centrality scores are divided among multiple nodes.

Bibliography

Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., and Hwang, D.-U. (2006). Complex networks: Structure and dynamics. *Physics Reports*, 424:175–308.

Borgatti, S. P. and Li, X. (2009). On social network analysis in a supply chain context. *Journal of Supply Chain Management*, 45(2):5–22.

Duckham, M., Kulik, L., Worboys, M., and Galton, A. (2008). Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recognition*, 41:3224–3236.

Galton, A. and Duckham, M. (2006). What is the region occupied by a set of points? In Raubal, M., Miller, H., Frank, A., and Goodchild, M., editors, *Geographic Information Science: 4th International Conference*, volume 4197 of *Lecture notes in Computer Science*, pages 81–98, Berlin Heidelberg. Springer-Verlag.

Joubert, J. W. and Axhausen, K. W. (2011). Inferring commercial vehicle activities in Gauteng, South Africa. *Journal of Transport Geography*, 19:115–124.

Joubert, J. W. and Axhausen, K. W. (2013). A complex network approach to understand commercial vehicle movement. *Transportation*, 40(3):729–750. doi: 10.1007/s11116-012-9439-0.

Montgomery, D. C. (2009). Design and analysis of experiments. 7th edition.

Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM (Society for Industrial and Applied Mathematics) Review*, 45:167–256.

Osborne, J. W. (2010). Improving your data transformations: Applying the Box-Cox transformation. *Practical Assessment, Research & Evaluation*, 15. ISSN 1531-7714.

Park, J. and Oh, S. (2012). A new concave hull algorithm and concaveness measure for n-dimensional datasets. *Journal of Information Science and Engineering*, 28:587–600.

Rardin, R. L. (1998). *Optimization in operations research*. Prentice Hall, Upper Saddle River, N. J. ISBN: 0023984155.

Stewart, T. J. (2007). The essential multiobjectivity of linear programming. *Orion*, 23(1):1–15.

Weatherill, N. P. (1992). Delaunay triangulation in cocomputation fluid dynamics. *Computers Mathematical Application*, 24(5/6):129–150.

Zhou, C., Frankowski, D., Ludford, P., Shekhar, S., and Terveen, L. (2004). Discovering personal gazetteers; an interactive clustering approach. In *Proceedings of the 12th annual ACM international workshop on Geographic Information Systems*, pages 266–273, Washington DC, USA. ACM.

Appendix A

Tables in appendix

A.1 Computer output of the multiple response experiment

Table A.1: Acronyms and abbreviations used in the computer output

Acronym/Abbreviation	Meaning
2FI	Two factor interaction
DF	Degrees of freedom
SS	Sum of squares
Std. dev.	Standard devaition
PRESS	Predicted residual sum of squares
CV	Coefficient of variation
CI	Confidence interval
CE	Coefficient estimate
SE	Standard error
VIF	Variance inflation factor

Table A.2: Summary statistics for completeness

Model	p-value	Adjusted R^2	Predicted R^2
Linear	0.4678	-0.0087	-0.1337
2FI	0.0833	0.0316	-0.1469
Quadratic	0.5470	0.0163	-0.2555
Cubic	0.0472	0.1336	-0.4153
Quartic	0.2251	0.1767	-1.0434
Fifth	0.0481	0.3237	-2.0157
Sixth	0.0102	0.5375	-3.7574

Table A.3: Sequential model sum of squares for completeness

Model	SS	DF	Mean Square	F-value	p-value
Mean vs Total	4.261E+007	1	4.261E+007		
Linear vs Mean	10 526.78	2	5 263.39	0.77	0.4678
2FI vs Linear	20 458.79	1	20 458.79	3.12	0.0833
Quadratic vs 2FI	8 133.47	2	4 066.74	0.61	0.5470
Cubic vs Quadratic	61 560.51	4	15 390.13	2.63	0.0472
Quartic vs Cubic	40 657.79	5	8 131.56	1.46	0.2251
Fifth vs Quartic	66 260.42	6	11 043.40	2.41	0.0481
Sixth vs Fifth	66 505.61	6	11084.27	3.54	0.0102
Residual	84 493.33	27	3 129.38		
Total	4.297E+007	54	7.958E+005		

Table A.4: Model summary for completeness

Model	Std. Dev.	R^2	Adjusted R^2	Predicted R^2	PRESS
Linear	82.61	0.0294	-0.0087	-0.1337	4.066E+005
2FI	80.95	0.0864	0.0316	-0.1469	4.113E+005
Quadratic	81.58	0.1091	0.0163	-0.2555	4.502E+005
Cubic	76.56	0.2808	0.1336	-0.4153	5.075E+005
Quartic	74.64	0.3941	0.1767	-1.0434	7.328E+005
Fifth	67.64	0.5789 0.3237	-2.0157	1.081E+006	
Sixth	55.94	0.7644	0.5375	-3.7574	1.706E+006

Table A.5: Model summary statistics for completeness

Statistic	Value
Std. Dev.	67.64
R^2	0.5789
Mean	888.34
Adjusted R^2	0.3237
CV %	7.61
Predicted R^2	-2.0157
PRESS	1.081E+006
Adequate Precision	10.175

Table A.6: ANOVA for fifth order completeness response surface model

Factor	**SS**	**DF**	**Mean Square**	F**-Value**	p**-value**
Model	2.076E+005	20	10 379.89	2.27	0.0179
A-p_{min}	276.54	1	276.54	0.060	0.8073
B-ϵ	5 069.12	1	5 069.12	1.11	0.3002
AB	4 676.29	1	4 676.29	1.02	0.3194
A^2	9 797.97	1	9 797.97 2.14	0.1528	
B^2	184.86	1	184.86	0.040	0.8419
A^2B	38 455.61	1	38 455.61	8.40	0.0066
AB^2	2 264.43	1	2 264.43 0.49	0.4867	
A^3	455.99	1	455.99	0.100	0.7542
B^3	54.89	1	54.89	0.012	0.9134
A^2B^2	7 040.02	1	7 040.02	1.54	0.2236
A^3B	15 851.97	1	15 851.97	3.46	0.0716
AB^3	9 268.29	1	9 268.29	2.03	0.1641
A^4	8 212.48	1	8 212.48	1.79	0.1895
B^4	4 233.01	1	4 233.01	0.93	0.3431
A^3B^2	13 919.82	1	13 919.82	3.04	0.0904
A^2B^3	2 820.11	1	2 820.11	0.62	0.4380
A^4B	42 734.00	1	42 734.00	9.34	0.0044
AB^4	4 355.23	1	4 355.23	0.95	0.3364
A^5	557.82	1	557.82	0.12	0.7292
B^5	1 873.45	1	1873.45	0.41	0.5267
Residual	1.510E+005	33	4575.73		
Cor Total	3.586E+005	53			

Table A.7: Coefficients for the completeness response model

Factor	CE	DF	SE	95% CI Low	95% CI High	VIF
Intercept	858.62	1	27.89	801.88	915.35	
A-p_{min}	-24.92	1	101.37	-231.17	181.32	58.02
B-ϵ	-91.18	1	86.63	-267.42	85.07	37.80
AB	78.36	1	77.51	-79.34	236.06	14.52
A^2	163.01	1	111.40	-63.63	389.66	22.95
B^2	-20.72	1	103.07	-230.42	188.98	16.88
A^2B	530.57	1	183.02	158.22	902.93	65.22
AB^2	121.32	1	172.45	-229.54	472.18	53.29
A^3	110.17	1	348.99	-599.86	820.20	509.18
B^3	-28.21	1	257.61	-552.33	495.90	213.85
A^2B^2	-78.68	1	63.43	-207.73	50.37	3.83
A^3B	-126.52	1	67.98	-264.82	11.78	8.26
AB^3	97.36	1	68.41	-41.82	236.54	7.23
A^4	-130.40	1	97.34	-328.43	67.63	21.56
B^4	89.44	1	92.99	-99.75	278.62	15.39
A^3B^2	-211.00	1	120.98	-457.13	35.13	19.41
A^2B^3	-93.82	1	119.51	-336.96	149.32	17.79
A^4B	-440.93	1	144.28	-734.47	-147.39	35.89
AB^4	131.01	1	134.28	-142.19	404.20	25.62
A^5	-91.79	1	262.90	-626.68	443.09	274.78
B^5	122.99	1	192.20	-268.06	514.03	104.34

Table A.8: Summary statistics for file size

Model	p-value	Adjusted R^2	Predicted R^2
Linear	< 0.0001	0.5170	0.4537
2FI	< 0.0001	0.7021	0.6469
Quadratic	< 0.0001	0.8292	0.7685
Cubic	< 0.0001	0.9676	0.9463
Quartic	< 0.0001	0.9919	0.9866
Fifth	0.0002	0.9955	0.9930
Sixth	0.0043	0.9972	0.9940

Table A.9: Sequential model sum of squares for file size

Model	Sum of Squares	DF	Mean Square	F-Value	p-value
Mean vs Total	2.508E-005	1	2.508E-005		
Linear vs Mean	1.191E-006	2	5.954E-007	29.37	< 0.0001
2FI vs Linear	4.086E-007	1	4.086E-007	32.67	< 0.0001
Quadratic vs 2FI	2.811E-007	2	1.406E-007	19.61	< 0.0001
Cubic vs Quadratic	2.842E-007	4	7.106E-008	52.19	< 0.0001
Quartic vs Cubic	4.673E-008	5	9.346E-009	27.64	< 0.0001
Fifth vs Quartic	6.994E-009	6	1.166E-009	6.21	0.0002
Sixth vs Fifth	2.978E-009	6	4.963E-010	4.17	0.0043
Residual	3.216E-009	27	1.191E-010		
Total	2.730E-005	54	5.056E-007		

Table A.10: Model summary statistics for file size

Model	**Std. Dev.**	R^2	R^2 **Adjusted**	R^2 **Predicted**	**PRESS**
Linear	1.424E-004	0.5353	0.5170	0.4537	1.215E-006
2FI	1.118E-004	0.7189	0.7021	0.6469	7.856E-007
Quadratic	8.468E-005	0.8453	0.8292	0.7685	5.150E-007
Cubic	3.690E-005	0.9731	0.9676	0.9463	1.194E-007
Quartic	1.839E-005	0.9941	0.9919	0.9866	2.987E-008
Fifth	1.370E-005	0.9972	0.9955	0.9930	1.553E-008
Sixth	1.091E-005	0.9986	0.9972	0.9940	1.327E-008

Table A.11: ANOVA for fifth order file size response surface model

Factor	**SS**	**DF**	**Mean Square**	F**-Value**	p**-value**
Model	2.219E-006	20 1.109E-007	591.05	< 0.0001	
A-p_{min}	2.038E-009	1	2.038E-009	10.86	0.0024
B-ϵ	1.127E-008	1	1.127E-008	60.05	< 0.0001
AB	5.248E-010	1	5.248E-010	2.80	0.1040
A^2	6.131E-013	1	6.131E-013	3.267E-003	0.9548
B^2	1.271E-008	1	1.271E-008	67.73	< 0.0001
A^2B	2.661E-011	1	2.661E-011	0.14	0.7089
AB^2	2.226E-009	1	2.226E-009	11.86	0.0016
A^3	1.170E-011	1	1.170E-011	0.062	0.8044
B^3	7.434E-009	1	7.434E-009	39.61	< 0.0001
A^2B^2	1.007E-011	1	1.007E-011	0.054	0.8183
A^3B	3.907E-011	1	3.907E-011	0.21	0.6512
AB^3	4.402E-008	1	4.402E-008	234.53	< 0.0001
A^4	2.484E-009	1	2.484E-009	13.23	0.0009
B^4	6.952E-011	1	6.952E-011	0.37	0.5469
A^3B^2	2.677E-011	1	2.677E-011	0.14	0.7081
A^2B^3	1.419E-011	1	1.419E-011	0.076	0.7850
A^4B	1.977E-011	1	1.977E-011	0.11	0.7476
AB^4	4.089E-009	1	4.089E-009	21.79	< 0.0001
A^5	1.351E-010	1	1.351E-010	0.72	0.4023
B^5	2.709E-009	1	2.709E-009	14.44	0.0006
Residual	6.194E-009	33	1.877E-010		
Cor Total	2.225E-006	53			

Table A.12: Model summary for file size

Statistic	Value
Std. Dev.	1.370E-005
R^2	0.9972
Mean	6.815E-004
Adjusted R^2	0.9955
CV	% 2.01
Predicted R^2	0.9930
PRESS	1.553E-008
Adequate Precision	147.869

Table A.13: Coefficients for file size response surface model

Factor	CE	DF	SE	95% CI Low	95% CI High	VIF
Intercept	6.368E-004	1	5.648E-006	6.253E-004	6.483E-004	
A-p_{min}	6.765E-005 1	2.053E-005	2.588E-005	1.094E-004	58.02	
B-ϵ	1.360E-004	1	1.754E-005	1.003E-004	1.716E-004	37.80
AB	-2.625E-005	1	1.570E-005	-5.819E-005	5.689E-006	14.52
A^2	-1.290E-006	1	2.256E-005	-4.719E-005	4.461E-005	22.95
B^2	1.718E-004	1	2.087E-005	1.293E-004	2.143E-004	16.88
A^2B	-1.396E-005	1	3.707E-005	-8.937E-005	6.145E-005	65.22
AB^2	1.203E-004	1	3.493E-005	4.922E-005	1.913E-004	53.29
A^3	1.764E-005	1	7.068E-005	-1.262E-004	1.614E-004	509.18
B^3	-3.284E-004	1	5.217E-005	-4.345E-004	-2.222E-004	213.85
A^2B^2	2.975E-006	1	1.285E-005	-2.316E-005	2.911E-005	3.83
A^3B	-6.281E-006	1	1.377E-005	-3.429E-005	2.173E-005	8.26
AB^3	-2.122E-004	1	1.385E-005	-2.404E-004	-1.840E-004	7.23
A^4	-7.171E-005	1	1.971E-005	-1.118E-004	-3.160E-005	21.56
B^4	1.146E-005	1	1.883E-005	-2.685E-005	4.978E-005	15.39
A^3B^2	9.253E-006	1	2.450E-005	-4.059E-005	5.910E-005	19.41
A^2B^3	-6.656E-006	1	2.420E-005	-5.590E-005	4.259E-005	17.79
A^4B	9.483E-006	1	2.922E-005	-4.997E-005	6.893E-005	35.89
AB^4	1.269E-004	1	2.720E-005	7.161E-005	1.823E-004	25.62
A^5	4.517E-005	1	5.324E-005	-6.316E-005	1.535E-004	274.78
B^5	1.479E-004	1	3.893E-005	6.871E-005	2.271E-004	104.34

Table A.14: Summary statistics for completeness

Model	p-value	Adjusted R^2	Predicted R^2
Linear	< 0.0001	0.7210	0.6876
2FI	0.0019	0.7659	0.7194
Quadratic	< 0.0001	0.9086	0.8807
Cubic	< 0.0001	0.9748	0.9580
Quartic	< 0.0001	0.9947	0.9877
Fifth	< 0.0001	0.9989	0.9968
Sixth	< 0.0001	0.9997	0.9990

Table A.15: Sequential model sum of squares for completeness

Model	Sum of Squares	DF	Mean Square	F-Value	p-value
Mean vs Total	3 957.17	1	3 957.17		
Linear vs Mean	48.64	2	24.32	69.47	< 0.0001
2FI vs Linear	3.17	1	3.17	10.80	0.0019
Quadratic vs 2FI	9.18	2	4.59	40.05	< 0.0001
Cubic vs Quadratic	4.11	4	1.03	32.53	< 0.0001
Quartic vs Cubic	1.13	5	0.23	34.16	< 0.0001
Fifth vs Quartic	0.21	6	0.036	26.54	< 0.0001
Sixth vs Fifth	0.035	6	5.858E-003	17.14	< 0.0001
Residual	9.227E-003	27	3.417E-004		
Total	4 023.66	54	74.51		

Table A.16: Model summary statistics for completeness

Model	**Std. Dev.**	R^2	R^2 **Adjusted**	R^2 **Predicted**	**PRESS**
Linear	0.59	0.7315	0.7210	0.6876	20.77
2FI	0.54	0.7792	0.7659	0.7194	18.66
Quadratic	0.34	0.9173	0.9086	0.8807	7.94
Cubic	0.18	0.9791	0.9748	0.9580	2.79
Quartic	0.081	0.9961	0.9947	0.9877	0.82
Fifth	0.037	0.9993	0.9989	0.9968	0.21
Sixth	0.018	0.9999	0.9997	0.9990	0.063

Table A.17: ANOVA for the fifth order completeness response model

Factor	**SS**	**DF**	**Mean Square**	F**-Value**	p**-value**
Model	66.45	20	3.32	2 470.93	< 0.0001
A-p_{min}	0.15	1	0.15	113.90	< 0.0001
B-ϵ	0.19	1	0.19	141.38	< 0.0001
AB	0.015	1	0.015	11.12	0.0021
A^2	1.572E-003	1	1.572E-003	1.17	0.2874
B^2	0.053	1	0.053	39.65	< 0.0001
A^2B	5.553E-003	1	5.553E-003	4.13	0.0503
AB^2	3.896E-004	1	3.896E-004	0.29	0.5940
A^3	1.529E-003	1	1.529E-003	1.14	0.2939
B^3	0.051	1	0.051	38.24	< 0.0001
A^2B^2	0.081	1	0.081	60.59	< 0.0001
A^3B	0.081	1	0.081	59.97	< 0.0001
AB^3	0.34	1	0.34	252.84	< 0.0001
A^4	0.39	1	0.39	287.30	< 0.0001
B^4	0.12	1	0.12	91.00	< 0.0001
A^3B^2	0.012	1	0.012	9.22	0.0046
A^2B^3	0.018	1	0.018	13.11	0.0010
A^4B	0.022	1	0.022	16.12	0.0003
AB^4	0.037	1	0.037	27.17	< 0.0001
A^5	0.047	1	0.047	34.83	< 0.0001
B^5	0.079	1	0.079	58.77	< 0.0001
Residual	0.044	33	1.345E-003		
Cor Total	66.49	53			

Table A.18: Model summary for completeness

Statistic	Value
Std. Dev.	0.037
R^2	0.9993
Mean	8.56
Adjusted R^2	0.9989
CV %	0.43
Predicted R^2	0.9968
PRESS	0.21
Adequate precision	235.730

Table A.19: Coefficients for completeness resposne surface model

Factor	**CE**	**DF**	**SE**	**95% CI Low**	**95% CI High**	**VIF**
Intercept	8.15	1	0.015	8.12	8.18	
A-p_{min}	-0.59	1	0.055	-0.70	-0.47	58.02
B-ϵ	-0.56	1	0.047	-0.65	-0.46	37.80
AB	-0.14	1	0.042	-0.23	-0.055	14.52
A^2	-0.065	1	0.060	-0.19	0.058	22.95
B^2	-0.35	1	0.056	-0.47	-0.24	16.88
A^2B	0.20	1	0.099	-2.355E-004	0.40	65.22
AB^2	-0.050	1	0.093	-0.24	0.14	53.29
A^3	0.20	1	0.19	-0.18	0.59	509.18
B^3	0.86	1	0.14	0.58	1.15	213.85
A^2B^2	0.27	1	0.034	0.20	0.34	3.83
A^3B	0.29	1	0.037	0.21	0.36	8.26
AB^3	0.59	1	0.037	0.51	0.67	7.23
A^4	0.89	1	0.053	0.79	1.00	21.56
B^4	0.48	1	0.050	0.38	0.58	15.39
A^3B^2	-0.20	1	0.066	-0.33	-0.066	19.41
A^2B^3	-0.23	1	0.065	-0.37	-0.10	17.79
A^4B	-0.31	1	0.078	-0.47	-0.15	35.89
AB^4	-0.38	1	0.073	-0.53	-0.23	25.62
A^5	-0.84	1	0.14	-1.13	-0.55	274.78
B^5	-0.80	1	0.10	-1.01	-0.59	104.34