Evaluation of Multivariate Methods for Dimensionality Reduction and Cluster Analysis in the Context of a Debate Structuring System

Master Thesis

Submitted by

José Kümin

São Paulo, SP, Brazil

06-728-976

University of Zurich

Department of Information Technology (IFI)

Zurich, Switzerland, August 23th 2012

Advisor: Prof. Dr. L. Hilty

This thesis is dedicated to my father Peter H. Kümin and to everyone who helped me during my studies. Specifically to the people that helped me with the thesis: Lorenz Hilty as the advisor, Rounak Vyas for understanding mathematical models, Cedric Jeannert for understanding multivariate methods and basic statistics, Tobias Giesbrecht for understanding statistical significance, Georgios Anthitsis for understanding linear algebra and Frank Oben and Rachel Melody Etter for corrections.

Table of contents

1. Introduction

Depending on the importance of a topic for debate, the resulting decisions based on that topic may have a large impact. If the topic is important, the decision is likely to involve many people, who, as decision-makers, formulate arguments and make their opinions known in the context of a debate. Yet collaboratively reaching a decision through a debate is no small feat. Depending on the number of arguments and the debate, the finite cognitive capacity of participants' may reach its limits, which might result loss of the overview of the debate. In addition, the more details or sub-topics that are under debate, the more different contexts a participant has to keep up with. This is due to the unstructured nature of the debate (Belnap & Withers, 2008). An adverse effect that might emerge is participants repeating their arguments to gain more awareness from other participants. This may result in, for example, participants not being able to follow the consequences of each action proposed in an argument and support bad decisions just because they have to have an opinion.

Since participants' limited cognitive capacity does not allow them to track every piece of information in a debate, the idea emerged to use a computer-based system to do this. TBDis is a system that enables collaborative debating that is independent from both time and space. The main problem targeted by this system is the loss of clarity in large debates (Kümin, 2010). During a debate, participants are encouraged to contribute their own statements to form arguments, thus adding structural information. Participants can also express their opinions by positively or negatively rating the statements of other participants, thus adding a second layer of information about the general "opinion landscape" (Hilty, 2010b). At any time during the debate, a report can be generated which provides an overview of the debate (Hilty, 2011b).

Currently, the report includes a summary of the arguments and lists the different fractions. In the latter, information about participants' ratings on statements is analysed and this is used to cluster the participants in fractions (Hilty, 2011b). The purpose of the report is to summarise the "opinion landscape" in a human readable form.

In this thesis, further methods to summarise and present the opinion landscape are proposed. A more intuitive way to understand an opinion landscape is to represent it visually in the report. Figure 1.1 is an example of a canvas with the names of different participants involved in the debate. The distances between the names gives an indication on the extent to which participants' opinions are similar or different.

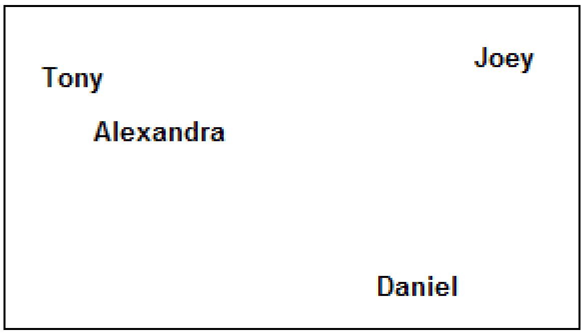


Figure 1.1: Visualized opinion landscape

The starting point for visualising an opinion landscape is the information about a participant's opinion on specific statements. A simple example is if there are only two statements that have been rated by participants, the ratings could then be plotted in two dimensions, one for each statement. Difficulties arise when there are more than two statements in a debate. For each new statement that a participant can rate, a new dimension is added. In such a case multivariate methods for dimensionality reduction can be applied on this information. This produces an output of two coordinates for each participant that is a combination of all information about a participant's opinion on statements. These coordinates can be plotted to produce a visualised opinion landscape similar to figure 1.1.

Reducing dimensionality is a process not without its faults. The problem lies with the compromise between successfully displaying the opinion landscape in two dimensions and the information that is lost by reducing the dimensionality from the total number of statements to two. A reduction to, for example, three dimensions would generally retain more information but then the opinion landscape would have to be displayed in a three-dimensional space on the report, which is not adequate. Thus in this thesis, the goal is evaluate methods for visualizing an opinion landscape in two dimensions.

Besides proposing a new way to summarise the data, the current purpose-built clustering algorithm is critiqued by comparing it to other algorithms that also fall under multivariate methods. Since no hypothesis is tested, the thesis can be classified as exploratory.

1.1 The context

In a broad sense, both the multivariate methods for dimensionality reduction and for clustering involve the analysis of data and retrieval of patterns. This process is described in the methodology for data mining (ACM SIGKDD, 2006). In fact, the content of this thesis can be classified under data mining although the exception for the TBDis system is that the data used is the rankings by the participants. Since there is no noise caused by measuring, some steps necessary in data mining are not applicable here.

The system used in this thesis is called TBDis (from To Be Discussed), which is essentially a web-based debate structuring system that facilitates the analysis of a debate. The argumentation in a debate is formed when participants interact with the system. The structuring is limited to the relationship between statements. The statements themselves can contain any desired content because the current analysis algorithms do not deal with the meaning of statements, but rather with their relation to each other.

Participants can write pro or contra arguments to statements in the debate; including to the main hypothesis that is created by the initiator of the debate. Since arguments consist of statements, a participant could also add pro and contra arguments to an existing argument, which would thus form a tree-like structure. This means that debate could continue ad infinitum. As pointed out before, participants can also rate the statements positively and negatively and indicate their certainty in their opinions within the system.

1.2 The goals

As a result of the tree-like structure with subsequent arguments provided in reference to earlier statements, ultimately all arguments refer to the main hypothesis either directly or indirectly. This technically means that the ratings on the main hypothesis reflect the overall ratings of the debate. A limitation of this understanding is that the ratings on the main hypothesis are on only a one-dimensional line. Here, there is only one statement to be rated (the main hypothesis) and all participants would fall along this line according to their positive or negative ratings and their certainty. The multivariate methods for clustering would be used to detect fractions only based on this line. By including the ratings for all statements, clustering is based on more data than simply the main hypothesis. Similarly, the methods for dimensionality reduction can then produce an opinion landscape where all statements are taken into account. Ultimately the TBDis system is the intended application for the algorithms.

The overall goal of the thesis is to find suitable algorithms for analysing and summarising the ratings of the debate supporting system. This can be broken down into two goals which are concerned with the presentation of the opinion landscape. The first goal is to find the best method for dimensionality reduction in the context of the TBDis system. The resulting two-dimensional scatter plot represents the opinion landscape. It shows the participants how similar or dissimilar their overall opinions are through the distance between the participants on the scatter plot. To achieve this most accurately, several methods are analysed and compared.

The second goal is to find the best method for analysing the ratings to cluster participants in fractions so that participants with similar opinions are listed together. To achieve this, the current clustering algorithm and three alternative algorithms are analysed and compared with each other.

1.3 The methodology and structure of the thesis

In chapter two the problem tackled by the thesis is defined in more detail. Since there was no prior knowledge about methods to visualise an opinion landscape, an extensive research was undertaken. The topic of multivariate statistics among other topics was identified as the most suitable, and it was researched in detail. Such methods were first explored in Gauch's work (1982) and are abundantly used in the field of biology. Additionally, some ways of combining the methods in order to improve the approach were discussed. As these particular methods were selected to achieve the goals of the thesis, many of the other researched topics are presented as related work. To describe the methods in enough depth, formulas were used. These were created with an online LaTeX equation editor developed by Bateman (2004).

In chapter three, the context of the thesis is presented. The current TBDis debate structuring system and its purpose-built, customised clustering algorithm are described.

In chapter four, the implementation of the methods with help of the R environment for statistical computing (R Foundation, 2012) is presented as well as ways to interface it with the webserver that hosts the TBDis system. OpenCPU (Ooms, 2011) as a surrogate server for processing R code has been found as a viable alternative to installing R in the current TBDis webserver. The evaluation methods for assessing the multivariate methods are also presented. Some features of the evaluation are the two types of analysis (quantitative and interpretative analysis), the two types of data (random and constructed data) and the two types of multivariate methods for appraisal (methods for dimensionality reduction and for clustering). In total, there are eight possible types of evaluation; however, some make more sense than the others.

In chapter five, the process of the evaluation is described for each of the following types of assessment: quantitative evaluation of methods for dimensionality reduction using random data, interpretative evaluation of those methods with constructed data as well as the interpretative evaluation of methods for cluster analysis using constructed data. The results of these evaluations are also presented and discussed in this chapter.

In chapter six, the results and the discussion are examined together and put into context so that a method for each of the two goals of the thesis can be selected. Ideas for future research are also presented in this concluding chapter.

The contribution of this thesis is seen in the design of evaluation methods that are applied to select multivariate methods for a specific context. The TBDis project also profits from this thesis through the results. Having a method to visualize the opinion landscape is assumed to improve the information content in the debate report. Additionally, the possible error found in the implementation of one of the methods in R has been reported to the R foundation.

2. Methods for analysing opinion landscapes

2.1 Problem definition

When people debate a topic they formulate arguments. More often than not, forming arguments is a way for people to express their opinions. Generally, the way in which an argument is formulated is understood by others as a person's opinion. On the other hand, if someone is playing a devil's advocate, they might raise an argument that they do not necessarily agree with but it may be valuable for bringing other points of view into the debate that might otherwise have been missed.

The outcome of this is a vast opinion landscape. In the political context, an opinion landscape describes the landscape of public opinion. The term opinion landscape is used to describe each participant's varied opinion on topics discussed in a debate where there are a manageable amount of participants. Smaller debates might present two different points of view on a topic. Typically, one is for or against the motion, and it is not difficult to identify the positions of participants. However, small debates can grow bigger, with a wider opinion landscape, if the participants start to consider more details and implications of actions that may result from a possible decision.

Depending on the depth of a discussion around the topic, there can be many implications and preconditions that require further discussion. Deeper discussions about such details can be termed sub-debates, which in aggregation make up the original debate.

This is when the opinion landscape may become too big for people to grasp. Since tracking the landscape needs cognitive effort, the following scenarios are likely:

- Participants might stop considering other's opinions on certain sub-debates because they are unsure of them.
- Participants might start repeating themselves to prove a point and make their opinions and arguments more memorable.
- The inferred opinion from arguments raised by people playing devil's advocate might be wrongly attributed to the advocate himself.

To return to the political example, these scenarios might arise as members of parliament discuss legislation for long hours. Nonetheless, the public wants to be aware of how the discussion develops and politicians' opinions on them. The media examines the final decisions and interviews politicians in order to inform the public. They visually present consolidated data in newscasts and the printed press. In some way, this is actually a way of tracking the opinion landscape.

Why is tracking the opinion landscape important? Doing so allows the participants to have a better overview over the discussion at hand and the sub-debates. This enhances the formulation of arguments that are better suited to taking discussions forward.

Yet not every debate has the benefit of people being paid to visualize and present the arguments and opinions. Debates held in online forums most likely do not receive such coverage. Therefore, there should be a way to automate the process of visualizing the opinion landscape.

In this chapter, current methods for opinion analysis are presented. Firstly, a rather broad overview of methods that are used to analyse is presented as related work. Then the landscape will be narrowed down to the methods that are more aligned with the purpose of the thesis. For instance, the semantics of arguments to extract opinions are not included in these methods.

The selected evaluation methods described in chapter 2 are based on the objective of this thesis (Orlóci & Kenkel, 1985, p188). Gauch (1982, p14) specifies the selection method for algorithms should be based on the datasets, research purposes and format of the presentation of the results.

2.2 Approaches to extracting opinion information from discourse data

There is a whole industry based on web analytics, which aims to discover what people like and use this knowledge to target them with relevant advertisements. For example, see the services provided by Viralheat (Viralheat Inc, 2012). Like Viralheat, determining who supports what is the focus of this thesis in a broad sense. In this context, support is understood as agreeing or disagreeing with an argument.

2.2.1 Discourse analysis

A discourse is a verbal interchange of ideas (Merriam-Webster, 2012). Discourse analysis is a term that encompasses several methodologies for analysing written or spoken discourse (Gee, 2005). Not only content but also rhetoric, persuasion, intonation or behaviour can be studied under the umbrella of discourse analysis. For the purpose of this thesis, discourses that have been digitized in a text form are of the most interest.

Even if sentences in a digitized discourse are well formed, they are still sentences produced by humans that are intended to be heard or read by other humans. This information content would first have to be extracted using some computational linguistics procedure before performing any type of analysis on it. The goal of this thesis to analyse and visualize the opinion landscape does not require knowing the content participants form an opinion of. For example, in order to operate, the system does not need to know that a participant has a negative opinion on the specific topic of abortion. The useful information is that he has a negative opinion on a topic X and other people also have positive or negative opinions on that same topic. The content is irrelevant for the system to function.

The value of the landscape is rather to show which opinions are close to each other, thus they are more compatible, and which opinions stand further apart. In the case of debates supported by the TBDis system, a rating system is used as a tool for participants to rate other participants' statements. This information is the basis for the opinion landscape.

Since the content of the statements themselves is not a concern here, computational linguistics' information extracting methods are not necessary. In other words, the limited scope avoids the challenge of analysing the semantic layer of the discourse. Instead, the ratings that participants give to statements in the context of the discourse are analysed. Another advantage of avoiding the semantic level is that the system is thus also not prone to be influenced by rhetoric, inequality and dominance of some participants.

Given this simplification there are still some challenges. As will be shown in section 2.4, using the discourse as a context can present difficulties. Having an overall context means the ratings are not independent from each other. As will be revealed later, some of the discussed methods of analysis call for independent variables. In other words they require the various ratings to be independent.

In conclusion, by being able to skip the information extracting methods, the starting point for the analysis is ordering the extracted information. After this step, the information will be visualized.

2.2.2 Sentiment analysis

Sentiment analysis or opinion mining is a method used to find out the attitude of a person towards a certain topic. Using the tools of natural language processing, computational linguistic methods and text analysis, the opinion from a text can be extracted and the attitude of a person can be inferred (Pang & Lee, 2008). This method can be used for long texts, such as an essay, or for shorter texts, such as text-based messages, "tweets", used in Twitter, the online social networking website.

According to Liu (2010) from the University of Illinois, there are two types of textual information: facts and opinions. While most mining efforts are concerned with facts, sentiment analysis and opinion mining are used instead to make sense of opinions. The web has vast amounts of user-generated opinionated text. Liu focuses on opinionated text citing examples from product reviews provided by the online retailer Amazon.

The author defines an opinion as a relationship between an object, the opinion holder, a time when the opinion is expressed, a feature of the object and most importantly a sentiment value. This is closely comparable to the current data structure in the TBDis system:

Bing Liu	**Current system**
Opinion holder Oh	UserID U
Object O	NodeID N
Time T	Timestamp T
Feature(O) F	
Sentiment(Oh,o,F,T) S	Opinion(U,N,T) O
	Certainty(O) C

Table 2.1: Comparison between the Liu's data structure and the TBDis system

In the current system, opinions are either "agree" or "disagree", and there are three certainty values which are used to weigh a person's opinion: "quite uncertain", "quite certain", and "very certain". They are translated in the system as numbers from zero to three. In Liu's model, Opinion and Certainty are only one variable. This indicates that it may be appropriate to consolidate both variables into an interval scale [-3,3], where -3 to -1 are three certainty values for "disagree", and 1 to 3 the certainty values for "agree". With this scale, computations can be performed. Moreover, Liu's model specifies the feature of an object upon which the opinion was given. The current system does not make this distinction of features because opinions are assigned to whole statements and not features of a statement. Statements themselves can be seen as features of an argument.

Web content mining

The domain of web content mining uses sentiment analysis amongst other methods to analyse people's discussions on the web. This field is not strictly for sentiment analysis because it includes more than just mining of the content. The other facets of web content mining are web usage mining (examine usage patterns) and web structure mining (examine hyperlinks to infer structure). In addition, the content mining facet applies not only to content in social networks but also to analysing web pages. Tasks like classifying web documents in categories are used for search engines (Schenker, Bunke, Last & Kandel, 2005).

In "Graph-Theoretic Techniques for Web Content Mining", Schenker et al. (2005) present a way to represent web documents with graphs which, like in an opinion landscape, are plotted in a scatter plot in order to show their similarities. Web content mining is essentially data mining that is applied to web documents. Data mining will be described in more detail in the next chapter.

Like discourse analysis, sentiment analysis also requires computational linguistics methods in the first step of extracting information. The TBDis system provides a way for the participants to express their opinion. Thus inference about the opinions in statements made by the participants is not necessary. Since this is the main goal of sentiment analysis, further exploration into this field can be avoided. Nonetheless, the comparison helps to confirm that the current data structure for opinions in the system is proceeding suitably.

2.2.3 Data mining

Data mining is the science of extracting useful knowledge from huge data repositories (ACM SIGKDD, 2006). The term itself is rather broad. It encompasses the entire extracting process, from beginning to end, which is broad enough to be applicable to fields from business intelligence to financial analysis. Therefore, the methods for analysing opinion landscapes can be attributed to data mining as well. This chapter provides an overview about data mining. Therefore, it helps to structure the process of selecting suitable analysis methods, and determining steps to take before and after applying the methods to the data.

A sample curriculum is available for data mining courses conceptualized by the ACM SIGKDD (The ACM Special Interest Group for Knowledge Discovery in Databases) in 2006. According to this document, data mining includes the following practices: database and data management issues, data pre-processing, choice of models and statistical inference considerations, interestingness metrics, algorithmic complexity considerations, post-processing of discovered structure, visualisation, and understandability, maintenance/updates and model life cycle considerations (Please refer to the original document for detailed descriptions of each of the components).

Given the pre-prepared data, only a subset of these components will be focused on and described:

- *Data pre-processing*, especially the questions of “What are effective methods for reducing the dimensionality of the data so the algorithms can work efficiently?” and “How are missing data items to be modelled?”, as well as the transformation of input data.

- *Choice of model and statistical inference considerations*, especially the question of choosing methods that allows the output to be evaluated. As will be shown later in section 2.3.2, evaluating the output is bound with interpretation and needs domain knowledge (knowledge about the context of the specific debate).

- *Algorithmic complexity considerations*, especially the choice of algorithms based on the size and dimensionality of data and questions about scalability of the methods for bigger data sets.

- *Visualisation and understandability*, especially “How can the output be effectively visualized in the context of or with the aid of the discovered structures from the data?”

This list seems to focus heavily on choosing the right algorithms for the targeted problem. As explained in chapter 1, the goal of the thesis is to find suitable algorithms for visualising the opinion landscape of the debate supporting system at hand, thus this list is useful for the goals of the thesis.

The aforementioned curriculum also includes methods for data cleaning before performing any methods on the data. Gauch (1982, p215) argues that cleaning the data by minimizing data noise, handling missing values and removing outliers allows for a better output. As we will be discussed later, outliers can pose challenges for analysis. However, this is assuming that the data is derived from measurements. When measuring data noise that produces strange outliers can be an issue. In contrast, the data for the opinion landscape has been carefully collected in a controlled environment. Secondly, removing outliers would result in excluding certain participants from the output graph which is not a viable option. Moreover, statements that are like no other (outliers) contain valuable information about the debate and should not be excluded from the analysis as well.

As data mining is a broad term it includes techniques for cluster analysis and dimensionality reduction which are of great interest to the purpose of this thesis (ACM SIGKDD, 2006). Depending on the source, methods such as Principal Component Analysis (PCA) are attributed to dimensionality reduction (SIGKDD, 2006) or to ordination (Gauch, 1982), although PCA is far from being the only method in this category.

Cluster analysis can be used to reduce the number objects requiring analysis by grouping similar objects together and treating them as a unity. Another use is in outlier analysis, where observations can indicate which objects do not fit into any cluster. As it is important to determine which objects or participants have similar opinions, the aim is to use cluster analysis for grouping and listing participants. In the next sections these two topics will be looked at using multivariate analysis. Techniques that are mentioned in the SIGKDD document will also be targeted in section 2.5.

2.3 Multivariate analysis for dimensionality reduction and clustering

Multivariate data is when objects are characterised by many variables. An example is a list of species and characterizing these by climate zones. For each climate zone, the researcher will sample the amount of animals by means of counting. This is typically represented as a two-dimensional matrix. Thus a Species X Samples matrix is produced. Each value in this matrix is the amount of animals of a particular species found in the climate zone. The ratings in the TBDis system have a similar matrix which is Participants X Statements. The values in the matrix are the participants' opinions about the statements. Unlike the Species X Samples matrix, in the Participants X Statements matrix, the variable is not a numeric quantity but rather a value in an interval from [-3,3] (see chapter 2.2.1**)**, which is defined by a participant's rating of a statement. Negative values represent a disagreement with three certainty levels. The inverse is valid for agreement. A zero value means that the participant has no opinion on the statement or has withdrawn their opinion. The resulting matrix has N times M values (data points). For the purpose of this thesis, columns will also be referred to as "Attributes" and the rows will be described as "Persons", which are more general terms. The value in the matrix is then the value of an attribute in relation to a person.

A simple example follows: assume we have three Persons and two Attributes. The value ranges from minus three to three. This can be represented by the following matrix (a):

```
   s1 s2
p1  3  2
p2  3  2
p3  2  0
p4  0  0
p5 -3 -3
p6 -2  0
p7  0  1
```

Figure 2.1 (a)

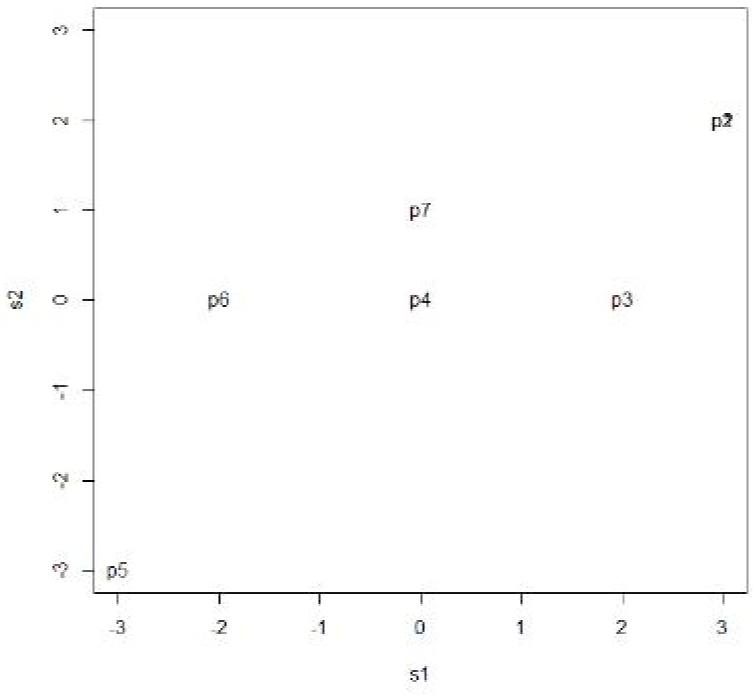


Figure 2.1 (b)

We have a 7x2 two-dimensional matrix. As discussed in chapter 1 one of the goals is to visualise the information. In order to do this, it is important to define the focus: Attributes or Persons. If the two attributes are analysed to reveal information about the relatedness between Persons, figure 2.1 (b) is produced where the axes are the attributes. This is the “Person space”. It is two-dimensional because there are two attributes. If the seven Persons were analysed to reveal information about the relatedness of the two Attributes, two points (each for one of the Attributes) would have been produced in a seven dimensional space. This would be the “Attribute space”. Both spaces are different geometric equivalents of the data matrix and thus have the same information content (Gauch, 1982, p111).

One type of measurement for the relatedness between objects is the distance between them. In this simple example, the two-dimensional scatter plot has the following vectors: p1 = (3,2), p2 = (3,2), p3 = (2,0), etc. The first number is the position on the abscissa axis (in (b) the horizontal S1 axis) and the second number is on the ordinate axis (vertical S2 axis).

In the case of ratings in the TBDis system, the aim is to analyse the statements to reveal information about the relatedness between persons or participants. There are an indefinite amount of participants and statements, however, the amount of participants is defined in the beginning of the debate, and the algorithm should be able to handle different debates which have different amount of participants. Therefore, it is impossible to determine in advance the number of dimensions in the multidimensional space. Since the goal is to visualise this information for participants, how can this information be summarised in a way that it is easy to read for the participants? This is where the multivariate analysis comes into play. Gauch (1982) explains that "Multivariate analysis helps summarizing data for better comprehension of the data and communicating results to the non-experts".

In other words, the purpose of multivariate analysis is to reduce the data complexity by reducing noise, summarizing redundancy, showing relationships and identifying outliers. It provides automatic and objective means to transform raw data into workable data (Gauch, 1982, p37).

There are several multivariate methods that can be used to show relatedness in the data. They can be used to measure the distances or cluster the data points together, and there are several ways to going about applying them.

2.3.1 Distance measures

Distances can be measured in a number of ways. The most common way is the Euclidean distance which is measured using the Pythagoras theorem. For the example above, the distances can be calculated as follows:

$$\mathrm{d}=\sqrt[2]{(x_2-x_1)^2+(y_2-y_1)^2}$$

Resulting in the distance between p1 and p3 being equal to:

$$\mathrm{d}=\sqrt[2]{(3-2)^2+(2-0)^2}=2.23607$$

There are two summands because the formula has to account for both the X-axis and the Y-axis. As a result, the formula is specific for the aforementioned case. The generalised formula for the Euclidean distance in a multidimensional space is:

$$\mathrm{d}=\sqrt[2]{\sum_{i=1}^{s}(x_{ij}-x_{ik})^2}$$

J and K represent both points for which the distance is to be calculated. S is the number of dimensions. The sum sign iterates from I = 1 to the value of S (two in this case). The formula is essentially an application of the Pythagoras theorem to every dimension and summing the outcome before calculating the square root (Pielou, 1984).

Just as there are many methods to analyse the data, there are also several methods and variations for calculating the distance between two points. For instance, one could square the result so that longer distances are emphasised. Other methods are the Canberra distance (Lance & Williams, 1967) and the interesting Manhattan distance (Krause, 1987). The latter is used to calculate distances in a grid. The name was inspired by the way in which a taxi cab covers distance from one point to another on the Manhattan Island.

Not only are there many different formulas for measuring distance but also one can chose to measure distances at different stages of the analysis. For instance, according to Orlóci and Kenkel (1985, p189), the distances for a cluster analysis can be measured either in the multidimensional space or after the dimensionality reduction. Although the latter appears to be easier, it can also contain errors, which are discussed in the next chapter.

An aggregation of distances between points can be stored in a distance matrix. Distance matrices or more generally dissimilarity matrices are required as input for several of the multivariate methods. Based on an input matrix Persons X Attributes, two types of dissimilarity (or distance) matrices can be calculated. One of them is a matrix which includes distances between Persons and the other includes the distances between Attributes. The first one is a Persons X Persons matrix. It has the same amount of rows and columns, N, which is also the dimensionality. There are N*N values in the matrix. Since it is a symmetric matrix and the diagonal is always zero (the diagonal is the distance of the Person to itself), the number of comparisons is:

$$\frac{N \times (N-1)}{2}$$

The matrix representing the Person dissimilarity space contains only information about the relatedness between Persons and no information at all about Attributes. The same is valid for the Attributes X Attributes matrix. Many of the aforementioned distance formulas square the actual distance so that it is not a negative number and higher distances are emphasised more.

It was assumed that the values in the matrix are strictly distances between points. The input of the multivariate methods does not have to be a distance matrix. Distance is just one possible way to measure dissimilarity between two points. It could be a matrix with any type of dissimilarity or similarity measures, such as percentage dissimilarities, correlation, variance or covariance. The difference between dissimilarity and similarity matrices is that in a dissimilarity matrix the greater the value, the more dissimilar are the points. The opposite is true for a similarity matrix (Kruskal & Wish, 1978). Since each data point in the multidimensional space can be characterised as a vector, one can also measure the angle between the vector of one Person and another Person in the Person space. The cosine of the angle between the vectors also has information about the similarity or dissimilarity between two Persons. This method is not used in this thesis but if it is used, it is advisable to position the centre of the axes inside the data cloud so that the values of the angles have a great spread (Gauch 1982, p 111).

The Person dissimilarity space, Attribute dissimilarity space and the original Person space (where the axes are Attributes and the data points are Persons) and Attribute space are the starting point for multivariate analysis. By applying these methods, it is hoped that structures in the data are found, as Orlóci (1978) describes.

2.3.2 Reducing dimensions

So far the different geometrical spaces that are available from the data matrix or the dissimilarity matrices have been explained. In this section, the principle of reducing dimensions from the Person space will be presented.

If we assume that we have several Persons with several Attributes we can say that the spaces (in particular the Person space) are multidimensional. This is not very helpful for visual inspection of the data or presenting the data to non-experts. In contrast, visualising two-dimensional data is relatively easy. One takes the coordinates of figure 2.2 (a) and plots them in a scatter plot as shown in (b). Three dimensions can be easily visualised with the help of tools that allow users to manipulate and rotate the three-dimensional scatter plot. Four and higher dimensions are harder to visualise using two-dimensional paper or a screen. To be able to visualise the Participant space in the report so that participants can see for themselves who has similar opinions, multivariate methods are used to reduce the dimensionality of the data to two. However As stated in the previous chapter, this process can contain errors. Next, an example is described.

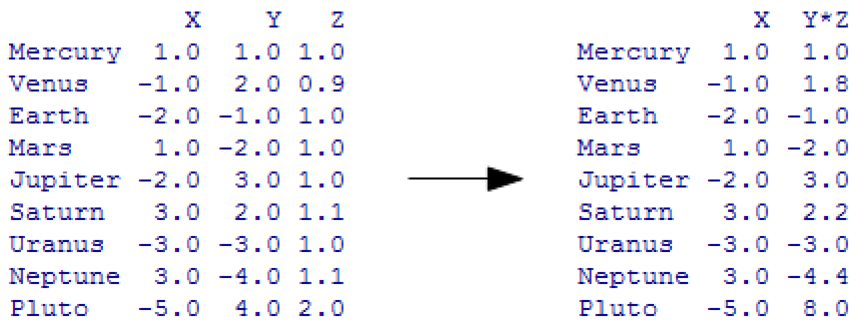
One of the methods for the reduction in dimensionality is projection (see section 2.2.3). Imagine that in a laboratory you have a static model of the solar system which represents a particular point in time (a three-dimensional space), and imagine that the position of each planet is a data point. If you were to turn on a spotlight and place it at the top of the model a shadow would appear on the ground. The shadow of three-dimensional objects is in principle what happens when a projection is made from a three-dimensional space into a two-dimensional plane. Depending on where the spotlight is placed, some planets might appear closer or further away from each other. Here a problem arises; by simply projecting to the plane (seeing the shadows on the ground), information about the third dimension is lost. The challenge is to find the best position for the spotlight so that the distances between the shadows of the planets most accurately represent their three-dimensional space. The beauty of the solar system is that the position of the spotlight does not matter too much because all planets except the dwarf planet Pluto are more or less in the same plane in space (Plait, 2002, p127). Therefore, positioning the spotlight exactly perpendicular to the plane would probably be the most accurate. In order to ensure that the position of the dwarf planet Pluto is also represented accurately in the shadow, the spotlight may have to be tilted by a few degrees. Nevertheless because not all planets are exactly on the same plane, the distances between the shadows of the planets (a two-dimensional plane) are less accurate as the distances between the planets themselves (a three-dimensional space). Although in this particular dataset the structure can be represented in two dimensions with only a small error, most datasets have points which are spread more among the dimensions.

Generally, projecting from a multidimensional space into lower dimensions can simplify the data too much, which means that information may get lost. Assuming that the data has no particular pattern in the structure, i.e. all data points in the multidimensional space were random, any type of projection would result in a large loss of information about the structure.

On the other hand, if the structure has some pattern as in the solar system example, there are different possible “angles” projections whereas some are better than others in delivering a lower dimensional space that represents the original space accurately (Gauch, 1982, p116). In the case of the solar system, a three-dimensional space was used while the structure of the data points was mostly concentrated in two dimensions. The pattern was that the variance of coordinates in one of the dimensions, the third, was not high. This allowed the right “angle” of projection to remove one dimension while still keeping most of information. In the case of multidimensional spaces, the challenge is greater because the projection removes not only one (from three to two dimensions) but several dimensions at once (from N to two dimensions).

The solar system example also indicates that it is more accurate to measure the distance in the original multidimensional space than on the projected lower dimensional space, even though the participants will only see the projection in the end. If the clustering is done in the multidimensional space and the result of a projection does not resemble heterogeneous clusters or they overlap, the scatter plot will not be usable. For this reason, it is very important to ensure accuracy in the projection. Often it is not possible to find a projection method which has the least amount of errors and where clusters do not overlap. In this case it is sensible to provide the result of a cluster analysis in plain text rather than display it in the scatter plot.

Computationally, a matrix is produced with the planets represented as rows and the dimensions (here there are three) as columns. Since one of these columns represents the values in the third dimension and has almost the same values for every planet, it is sensible to choose to simply ignore the whole column. By reducing the number of columns the number of dimensions is also reduced resulting in having two dimensions. This would suffice for the solar system example but it is not generalisable. Alternatively it is also possible to calculate ways to change the first two columns based on the values of the third. This way some of the information from the third column is still accounted for. This is shown in the following figure:



	X	Y	Z
Mercury	1.0	1.0	1.0
Venus	-1.0	2.0	0.9
Earth	-2.0	-1.0	1.0
Mars	1.0	-2.0	1.0
Jupiter	-2.0	3.0	1.0
Saturn	3.0	2.0	1.1
Uranus	-3.0	-3.0	1.0
Neptune	3.0	-4.0	1.1
Pluto	-5.0	4.0	2.0

	X	Y*Z
Mercury	1.0	1.0
Venus	-1.0	1.8
Earth	-2.0	-1.0
Mars	1.0	-2.0
Jupiter	-2.0	3.0
Saturn	3.0	2.2
Uranus	-3.0	-3.0
Neptune	3.0	-4.4
Pluto	-5.0	8.0

Figure 2.2: Simple dimensionality reduction

This example demonstrates how a dimensionality reduction can be performed. Obviously, this is not the best way to reduce dimensions as Pluto's position in relation to the other planets is much different after the projection than before. Technically this is not a projection but rather a combination of the dimensions in some way as to reduce their amount. As discussed earlier, one of the best approximations to a projection orthogonal to the solar system is simply ignoring the Z dimension.

In the context of the opinion landscape, there is also a two-dimensional data matrix; although an indefinite amount of statements are available as columns. It is difficult to reduce the number of columns while at the same time changing the values in the remaining columns to account for the information in the removed columns. However, this is exactly the purpose of techniques used in multivariate analysis.

The multivariate analysis itself is called an internal analysis. Interpreting the results of the multivariate methods in a context is an external analysis. According to Gauch (1982), while the former can be done automatically, the latter needs the trained eye of an expert in the domain. As the contribution of multivariate methods is to produce a visualisation of the data so that the participants can understand the opinion landscape, we rely on the participants as the "experts" in the context of the debate.

To perform the internal analysis, the methods follow the conceptual steps listed below before the results are visualised for the participants (Gauch, 1982):

1. Analysing relationship between data points.
2. Calculating the distances between data points in the multidimensional space using a generalised formula.
3. Projecting the data points to a visual two-dimensional plane with the least amount of information loss and the most accurate representation of the actual distances between data points.

The resulting information can then be visualised at a later stage.

2.4 Methods for multivariate analysis

As mentioned above, there are several available methods for multivariate analysis. Different scientists have categorised these methods in various ways. One way to distinguish the methods is to differentiate between ordination methods, clustering methods, and identification methods (Orlóci & Kenkel, 1985, p189ff). Gauch (1982) also adds direct gradient analysis as a discrete category. Depending on the source classification methods are categorized either under clustering methods or as their own category. As a disclaimer, it is important to mention that there are many methods within these categories and in this thesis, these will not be aggregated, but only a subset will be described. This chapter is intended to present the methods rather than discuss them in detail.

A limitation of these methods is that the experimental units are assumed to be independent (Smolkowski, 2012). To be independent, the observed values for one participant cannot have an effect on another. Although people's opinions can influence the choice of other's (French, 1980), in this thesis, it is assumed that the experimental units are independent. The outcome of the evaluation in chapter 5 is the sole criterion to determine whether a method is suitable for the TBDis system or not.

2.4.1 Classification and clustering

It is important to consider classification and cluster analysis (or just "clustering") because some scientists treat these as synonyms. The purpose of both of these methods is to order input into an output where a number of objects are "grouped" or "clustered". Clustering ("unsupervised learning") is used when only the data points in the multidimensional space are known without any kind of additional information (unlike in classification). The algorithms describe clusters based on relatedness, or similarity of the objects based on the values of the two-dimensional matrix. These clusters are not known in advance. On the other hand, in classification ("supervised learning"), all the data points are already classified or grouped before the algorithm is applied. The purpose is to order new data points into the existing classes (Han, 2012, p330). Since a cluster can be treated as a class, cluster analysis can serve as a pre-processing step for classification algorithms (Han, 2012, p445). Additionally cluster analysis can also be used for outlier detection as pointed out in section 2.6.

2.4.2 Ordination and clustering

Ordination takes the data from the two-dimensional matrix and displays continuous object variations in a low dimensional space, while classification or clustering place the data into discontinuous types (Gauch, 1986, p219). More simply, in the Person space, ordination methods are used to reduce the dimensions (reduce the amount of columns) accounting for as much information as possible, while classification or clustering methods are used to group the Persons together (reduce the amount of rows).

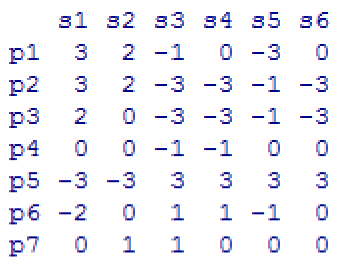
The output of ordination is the arrangement of the input matrix in a lower dimensional space so that similar objects are positioned close together while dissimilar ones are far apart. In contrast, the output of clustering is the assignment of objects into new clusters.

Williams & Gillard (1971) refer to ordination and classification/clustering as “pattern analysis” which, in contrast to other statistical analysis methods, do not start with testing a null hypothesis. The outcome of this rather exploratory analysis allows the creation of hypotheses from the structuring of data which then can be analysed with other methods. Since the TBDis system is used to try to make sense of the opinion landscape in an exploratory fashion, testing hypotheses will not be discussed.

In relation to clustering, ordination is not regarded as an alternative. As discussed in the previous section about classification, clustering and ordination are complementary methods for data analysis. Clustering orders data points into different “classes” or “clusters”, while ordination transforms the data so that similar objects are near to each other when projected into lower dimensions. A cluster analysis can be performed before or after the application of ordination on the data. Both methods contribute to the goal of making the data simpler for visualisation. According to Gauch (1982), “*a combination of ordination and classification methods applied to the same data is especially useful*”.

Ordination methods often tackle the problem by using not only ordination but also direct gradient analysis and classification methods. Most methods can be applied from the equivalent of a Person or Attribute space (Gauch, 1982, p120).

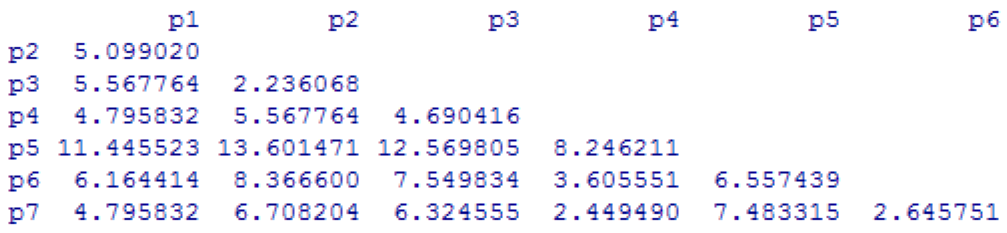
The methods will be described and illustrated with an example. The examples require a common dataset which is taken from the TBDis system. The system itself is presented in the following chapter 3, however, the short overview provided in the introduction and the introduction into the concept at beginning of this chapter should be sufficient to understand the dataset. In this particular context, the example matrix representing the opinion landscape is a Participants X Statements matrix (instead of a Persons X Attributes matrix) as follows:



	s1	s2	s3	s4	s5	s6
p1	3	2	-1	0	-3	0
p2	3	2	-3	-3	-1	-3
p3	2	0	-3	-3	-1	-3
p4	0	0	-1	-1	0	0
p5	-3	-3	3	3	3	3
p6	-2	0	1	1	-1	0
p7	0	1	1	0	0	0

Figure 2.3: Example matrix

This matrix was constructed so that there is a pattern. It is plausible to assert that statements S3 and S4 correlate. People who oppose S3 also oppose S4, and vice-versa. This is given by the designed underlying pattern in the structure. Without looking at the debate scatter plot one can suspect that S3 and S4 are statements that are probably both in agreement or both in disagreement with the main hypothesis. Also P1 and P2 seem to represent a generally like-minded opinion. One would expect P1, P2 and P3 to be close to each other and P5 to be far away in the output scatter plot. The corresponding distance matrix is as follows:



```
           p1        p2        p3        p4        p5        p6
p2   5.099020
p3   5.567764  2.236068
p4   4.795832  5.567764  4.690416
p5  11.445523 13.601471 12.569805  8.246211
p6   6.164414  8.366600  7.549834  3.605551  6.557439
p7   4.795832  6.708204  6.324555  2.449490  7.483315  2.645751
```

Figure 2.4: Distance matrix of the example matrix

The distance matrix will be used for the evaluation because, according to Hyrenbach (2011), the distance between points in ordination space should ideally be proportional to the underlying distance measures in the distance matrix. Moreover, this constructed example matrix is one of the matrices used to evaluate the methods.

2.4.3 Ordination

Pielou (1984, p133) offers a definition of ordination that describes the goal. According to Pielou, ordination is a collective term for multivariate methods that transform multidimensional data in such a way that when it is projected onto a two-dimensional plane, intrinsic patterns become apparent. Hyrenbach (2011) describes ordination as a graphical summarisation of complex relationships in which one or more dominant patterns is extracted.

To return to the solar system example, ordination is the process by which the direction and angle of the spotlight for a projection is calculated so that the shadow of the planets represents their three-dimensional arrangement in the best possible way.

2.5 Multivariate methods for dimensionality reduction

In this section, some methods of ordination will be described. The chapter most closely following the understanding of ordination outlined by Gauch (1982, p135). Green & Rao (1972) and Bennett & Bowes (1976) were used as sources for ensuring reliability and providing additional information that simplified the explanation of the methods. The lectures by Hyrenbach (2011) and Palmer (2000) are also two very good resources for understanding these methods. Gauch (1982) argues that the following methods have similar algorithms and return similar outputs. They are ordered in increasing complexity and are often based on each other.

Generally, the methods produce so called “ordination scores” for each Participant. An ordination axis is given by putting these scores in a straight line. The scores represent the position of the Participants on the ordination axis. Most of the methods generate more ordination axes. As seen before two axes are enough to get a scatter plot and show it to participants.

Weighted averages

This method proposed by Whittaker (1956) is a very simple method that delivers a one-dimensional gradient. In other words, all the Participants will be ordered in a one-dimensional ordination axis in the output. This method called “weighted averages” is used in the Principal component analysis (Hyrenbach, 2011), which will be discussed later.

Whittaker's method requires the Participants X Statements matrix and a vector containing the weights for each Statement. An ordination score O_i is calculated for every Participant “i” based on all Statements “j” and modified by the Statement’s weight W_j. The output is a vector with the ordination scores O_i for every Participant.

The scores are calculated as follows (Hyrenbach, 2011):

$$O_i = \frac{\sum_{j=1}^{s} v_{ij} w_j}{\sum_{j=1}^{s} v_{ij}}$$

Every value V_{ij} from the Participant’s row in the matrix is multiplied by the weight of the Statement W_j. Then the sum of the weighted values is divided by the sum of the values themselves.

The influence of the weights is large because if the weight for every Statement were the same, every Participant would have the same ordination number. The weights are selected a-priori based on a previous ordination or some other knowledge about the data (Gauch, 1986).

In the TBDis system, one could use the amount of non-zero values V_{ij} of a Statement column as the weights for each Statement. This would imply that Statements with more ratings are weighted higher making their influence greater on the output of the ordination. If each product was squared before the sum, an additional outcome would be that Participants with higher O_i scores are identified as participants that rate Statements which are popular (since they have a high number of ratings).

Another way to set the weights is to see how many steps are between the statement and the main hypothesis. The closer, the higher W_j would be, and the greater its influence on the outcome. This type of weighting can be useful independently from the ordination method.

The following figure represents the ordination scores for the Participants from the dataset presented in figure 2.3. The weights of the Statements are based on the proposal made earlier to use the amount of non-zero values of the Statements as weights. The amount of non-zero values for the

Statements in figure 2.3 are as follows: 5, 4, 7, 5, 5 and 3 (in the order of Statement one to Statement six).

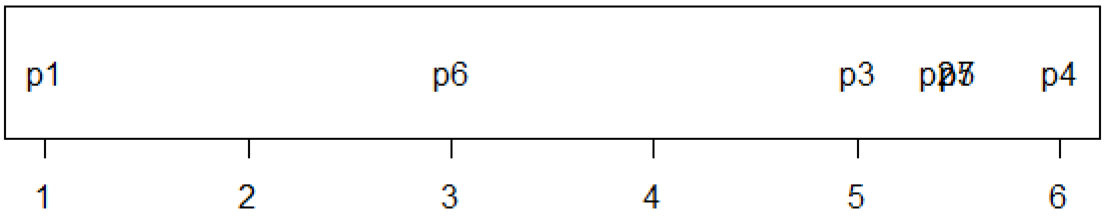


Figure 2.5: Ordination scores for Weighted Averages

The results in figure 2.5 are unexpected; the scores do not reflect the previously described constructed matrix, especially if P2 is observed. With a score of 5.4, it is too close to P5 which has a score of 5.5. According to the constructed matrix, P2 and P5 should be very far away.

Weighted averages also provides an alternative way to calculate distances between Participants. This has little in common with the dimensionality reduction itself but this digression is a viable alternative to the distance matrix. Suppose that we are looking at a fictitious Participant no. 3. All the weights of the Statements would be set in a way that mimics Participant no. 3’s ratings. For all the other Participants, it would hold true that the closer their O_i score is to Participant no. 3’s score, the closer their opinion is to Participant no. 3's and the shorter the distance.

Polar ordination

Polar ordination was developed by J. Bray and J. Curtis (1957) and it is also called Bray-Curtis ordination. Its name comes from the fact that the two farthest Participants serve as poles in an ordination axis because their distance is used as an indication of the variation in the opinion landscape.

Firstly, the calculation of a Participants X Participants distance matrix is required. Bray and Curtis used “percentage dissimilarity” as a distance measure but any of the different formulas can also be used. Secondly, two Participants must be selected as the poles. This can be done by looking for the highest number in the distance matrix, which indicates that the Participants with the most dissimilar opinions are chosen as the poles. The axis that connects the two poles is defined as the first ordination axis (Gauch, 1982). Since two poles have to be selected, this method is ideal for problems with definite endpoints (Hyrenbach, 2011).

Following the selection of poles, the ordination scores O_i for each Participant are calculated. For this, the distances between the two poles and the Participant in the multidimensional room are needed. After projecting the Participant onto the ordination axis, the ordination score is defined as the distance between the Participant and the first pole. Like with weighted averages, the output is a one-dimensional axis. Here, the first pole has an ordination score of zero and the second pole has the highest ordination score. All other Participants lie in between the two poles.

Calculating the ordination score for a participant is not too difficult. Since there are only three points that are relevant for the ordination score, one can imagine the two-dimensional triangle in a multidimensional space:

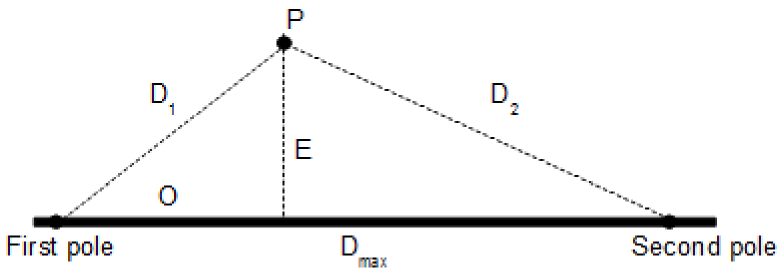


Figure 2.6: Triangle in the multidimensional space

The thicker line is the ordination axis that goes through the first and the second pole. The three different D values are already known. D_{max} is the greatest distance in the distance matrix (assuming that the poles are the two farthest Participants), D_1 and D_2 are the distances between a Participant ant the two poles. The ordination score O is calculated as follows (Hyrenbach, 2011):

$$O_i = \frac{D_{max}^2 + D_1^2 - D_2^2}{2 \times D_{max}}$$

By assigning an O_i score to each participant, a projection is performed for each of them from the two-dimensional triangle into a one-dimensional line. In other words, the multidimensional space has several of these two-dimensional triangles (one for each Participant except for the poles) and each one of them is projected onto the same ordination axis. On the axis, Participants that have similar opinions are also similarly distant from the first pole, which means that their scores are close.

The E_i value is the distance between the Participant and the ordination axis. It is defined as the information lost during the projection. Gauch (1982) argues that if E_i is around 25% to 40% of O_i, then there is too much error and much of the structure of the cloud is not captured by the ordination axis.

A solution for this is to introduce a second ordination axis which would make the space two-dimensional. In the context of the TBDis system, a second axis is required to show the two-dimensional scatter plot. The easiest way to produce the second axis is to choose the second-greatest distance between Participants which does not include either of the two original poles.

However, this is not necessarily the best way to account for more variance in the multidimensional space.

According to Palmer (2000), a better way to choose a second axis is to identify which of the many axes going through two Participants has the lowest correlation with the current ordination axis. The reasoning behind this choice is that the second ordination axis should mitigate the E_i error values as much as possible. Choosing an axis which is perpendicular to the current ordination axis would be the best way to account for more variance and minimise the error of the projection. In other words, choosing an axis which is almost parallel to the existing ordination axis does not provide much additional information, while an axis which is 90° from the original ordination axis would provide exclusively new information. However, since polar ordination relies on the triangle between three Participants for the projection, there is no way to calculate the positions of the Participants on a second ordination axis orthogonal to the first one if it does not pass through two Participants. Unless of course that by chance there are two Participants which are on exact opposite positions in relation to the first ordination axis. A second axis passing through them would be orthogonal to the first one. Thus the next best alternative is to find an ordination axis through two Participants that is the most incongruous with the first ordination axis. Gauch (1982, 131) argues that the new axis is probably not orthogonal to the original one but it is an approximation. The outcome is a two-dimensional ordination space where each Participant has a score for each axis. To produce the scatter plot for the report in the TBDis system one just has to plot the Participants are plotted using their ordination scores as coordinates.

Regarding the selection of poles, Gauch (1982) argues that in the context of ecology this process could be problematic because outliers are likely have the greatest distance in the distance matrix. As discussed in section 2.4.1 outliers can have a significant influence on the ordination. Below shows one case where this problem can occur.

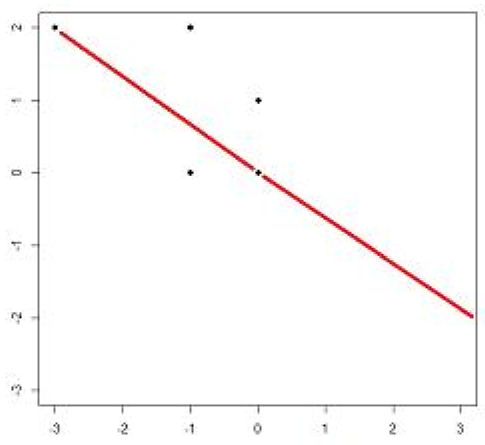


Figure 2.7 (a)

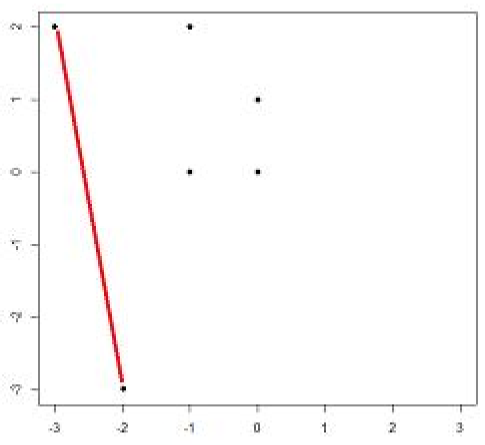


Figure 2.7 (b)

Figure 2.7 (a) shows a two-dimensional space with 5 data points. The cloud of points is concentrated in the range of [-3,0] in the first dimension and [0,2] in the second. In a polar ordination, the first axis would pass through the farthest points as depicted in figure 2.7 (a). If there is an outlier as in figure 2.7 (b), the first ordination axis would look very different. This is not the case for every outlier though. For example, an outlier with the coordinates (3,-2) would not result in a large change of the first axis. To avoid the problem, clustering algorithm could be used to identify the outlier. Alternatively, two samples could be manually selected as poles. Hyrenbach (2011) suggests the use of Beals' (1984) variance regression approach which is applied to select poles at the edge of the main cloud of points.

The bad situation shown above can be avoided by pre-selecting the poles based on criteria given by external data. For example, if there is a feature in the TBDis system where Participants of the pro and contra sides can appoint a leader or a spokesperson, these Participants can be used as the poles. Later in this chapter, another idea involving pre-selecting the poles is discussed.

To illustrate the method in an example, polar ordination is applied to the dataset presented at the beginning of this chapter. Some of the assumptions are the choice of the Euclidean distance as the distance measure and variance regression as the method for selecting poles. In addition, since polar ordination does not accept negative values in the matrix, the number three has been added to all values so that the minimum value is zero and the maximum value is six. This does not affect the distance matrix. The result is as follows:

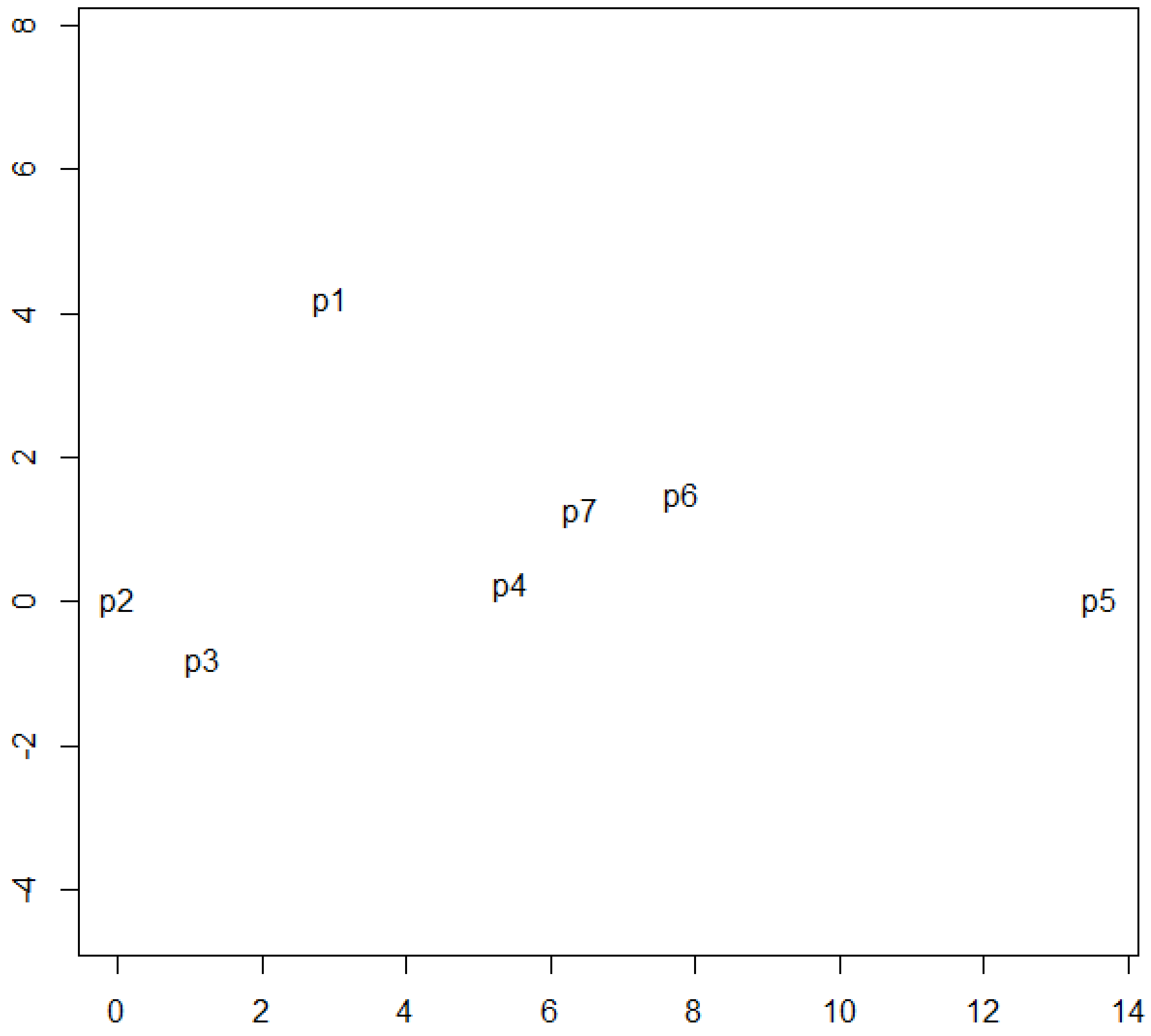


Figure 2.8: The opinion landscape from a polar ordination

It is important to point out that the scales on the X and Y axes are meaningless. In fact, both axes become meaningless after an ordination (Palmer, 2000). Thus only the relative distances (and not absolute positions) of the Participants are meaningful. Given this premise, the distances between the Participants seem to match the distance matrix quite well.

Principal component analysis

Principal component analysis (PCA) was first introduced by Pearson in 1901. The purpose is to project the multidimensional space into lower dimensions with the least amount of distortion. The "principal components" are the most important components or characteristics that describe in this case a Participant. Just like in polar ordination, the first ordination axis goes through the greatest distance between data points. The purpose is to be able to account for the most amount of variance (information) in the data points if the space were to be projected onto this axis. The difference to polar ordination is that the first axis does not have pass through any Participants. The second axis is then placed perpendicular to the first one at an angle that covers the most variance, which the first axis could not cover (Gauch, 1982). This seems counter intuitive. One can imagine the first axis as a shaft and the second axis as propeller blades of a small aircraft. The blades are always perpendicular to the shaft but they can rotate in higher dimensions (in the third dimension). This angle of "rotation" is part of what describes the position of the second axis and is chosen in a way that accounts for the most variance not covered for in the first axis.

Like in polar ordination, the ordination scores in the first axis account for a big part of the variance. The subsequent axes account for less variance than the previous, thus they have diminishing importance. Although more axes could be added to account for more variance and minimise the projection error, only the first two axes are needed for the TBDis system. This compromise is quantified in the evaluation chapter.

Unlike the first two methods, PCA has no subjective data entry like setting weights or selecting poles. It is purely objective.

The process can be best described as follows (Gauch, 1982):

1. To prepare, compute the Participants X Participants distance matrix.
2. Transform the multidimensional space so that the origin is in the middle of the cloud of Participants. This is called the centroid.
3. Set up the first PCA axis (the first principal component) similarly to polar ordination so that it goes through the centroid of the cloud. Rotate the axis to make sure that it accounts for the most variance.
4. Projecting the Participants onto the axis is not as trivial as in polar ordination. One way is to perform eigenanalysis on the distance matrix and the first PCA axis (eigenanalysis itself is not described here). PCA treats the eigenvalue as an indication for the variance explained by that PCA axis. The values in the eigenvector are then assigned to the Participants as their ordination scores. This renders a one-dimensional line where each Participant has a position.
5. Repeat steps 3 to 4 for the second PCA axis. Since each Participant has an ordination score for the first and the second axis it is possible to plot the Participants on a two-dimensional plane using the first and the second PCA axes.

The output renders not only ordination scores for Participants in multiple PCA axes but also scores for Statements (Gauch, 1982). This is done by taking the ordination scores of the Participants in a PCA axis as the Participant weights and applying the weighted averages method (Hyrenbach, 2011). Being able to calculate the scores for the Statements in the same process is an advantage of

PCA compared to previous methods. However, this is not being taken advantage of here because for the purposes of this study only the Participants need to be plotted.

The restriction of polar ordination that the axis has to pass through two Participants reduces the amount of possible ways the first ordination axis can be laid in the multidimensional room. Polar ordination takes the best way to lay the axis that it can. By not having this restriction, PCA can lay the first axis anywhere and a position may be found that accounts for more variance than the first ordination axis of a polar ordination. Another advantage of PCA is that an outlier, which would have been selected as a pole, will influence the positioning of the first ordination axis to a lesser extent.

The result of a PCA is illustrated by applying it to the dataset presented at the beginning of the chapter:

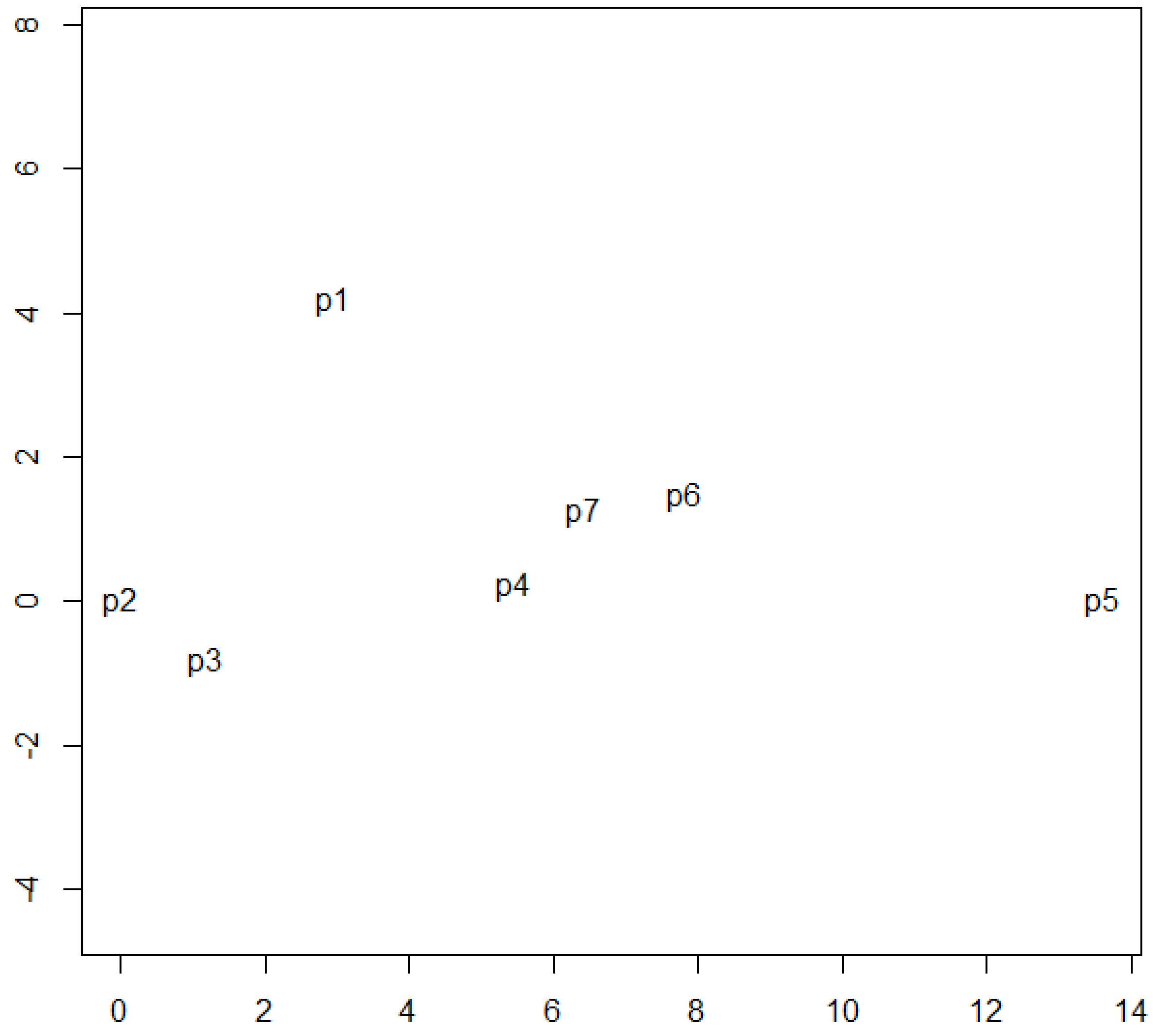


Figure 2.9: The opinion landscape from a principal component analysis

The opinion landscape looks similar to the results from the polar ordination. If the evaluation was only to encompass this dataset, either PCA or polar ordination could have been selected. According to Gauch and Whittaker (1972), polar ordination with Sorensen distance performs better on community data (where the values are amounts of species in a sample) even though it is a simpler method. Since the data from the TBDis system does not follow this pattern, both methods will be evaluated equally.

Reciprocal averaging

Another way to use weighted averages is the reciprocal averaging (RA, also called correspondence analysis). This method was introduced by Hirschfeld (1935) and it provides ordination scores not only for Participants but also for Statements. It is much more demanding in terms of computations than weighted averages but it is also provides a more objective way to set the weights.

RA is based on the weighted averages method with the addition of information from other Participants. As explained by Pielou (1984), this information is used to set the weights for the Statements, thus making the weights objective. The underlying assumption is that the importance (weight) of a Statement might be based on the ratings from the Participants.

The weights for Statements (and for Participants) are calculated in an iterative process. Starting with arbitrary weights for Statements, weighted averaging is applied for each Participant based on these arbitrary weights. The outcome is no longer defined as the Participant’s ordination score but as the Participant’s weight. A second weighted averaging is then applied for the Statements based on the Participant’s weight. This goes back and forth until the weights converge (Pielou, 1984).

The result is a set of converged weights for Participants and for Statements, and these values are influenced by each other. In contrast to the weighted average approach, there was no subjective setting of weight values. Since these weights come from a weighted averages, they are also the ordination scores.

As with weighted averages, this renders a one-dimensional ordination axis. A second axis can be extracted from the data using the same method but excluding the variance accounted by the first axis (Gauch, 1982).

The same output can be achieved by using eigenvalues, which is more similar to PCA (Hill, 1973). While PCA first projects Participants on the ordination axes and then uses their scores as weights to further calculate the scores of Statements, this alternative algorithm called Correspondence Analysis (CA) involves the Statements earlier in the process. To recap, PCA projects the Participants onto the first axis which maximises variance. For this, the method involves identifying the variance of the Participant space, but does not involve the Statement space. On the other hand, CA includes both Participants and Statements from the beginning. In CA, ordination scores are calculated so that correspondence between species scores and sample scores is maximised. The eigenvalues of the first PCA axis is related to the variance. In CA, the eigenvalue is related to the correlation coefficient between species scores and sample scores (Palmer, 2000).

It is important to note that PCA and RA/CA both produce known distortions of the data. In RA, values towards the middle of the ordination axes have a higher spread than values that lie closer to both extremes. Thus if in a fictitious example every data point were to be spread evenly, shorter distances between data points at both ends of the axis would be observed. This happens because of an artefact of RA on higher dimensions and is labelled the “arch effect”. In PCA, it is termed the “horseshoe effect” (Palmer, 2000). The effect is caused by different assumptions in the data. PCA and RA/CA assume monotonic responses, while community data usually is unimodally distributed (McCarthy, 2012). A fictional example of unimodal distributed response can be seen in a Fish species X Water temperature data matrix where the values are the number of fish in a certain water temperature. Intuitively, there is an optimal temperature for each species of fish. Fewer fish will be observed if the temperature is lower or higher. In contrast to this, a monotonic response is a gradient where, for example, the more food there is in a patch of water, the more fish can be observed. In a way, these multivariate methods are arguably not ideal for community data but they are used nonetheless. Since both effects appear because of these different assumptions and neither assumption is made for the opinion landscape, these effects will not be discussed in more detail. However, it is plausible that other distortions could arise.

The output of a RA/CA analysis is illustrated by applying it to the dataset presented in the beginning of the chapter. Since the algorithm requires all values to be positive, the number four has been added to the values of the matrix. This does not affect the distance matrix. The result is as follows:

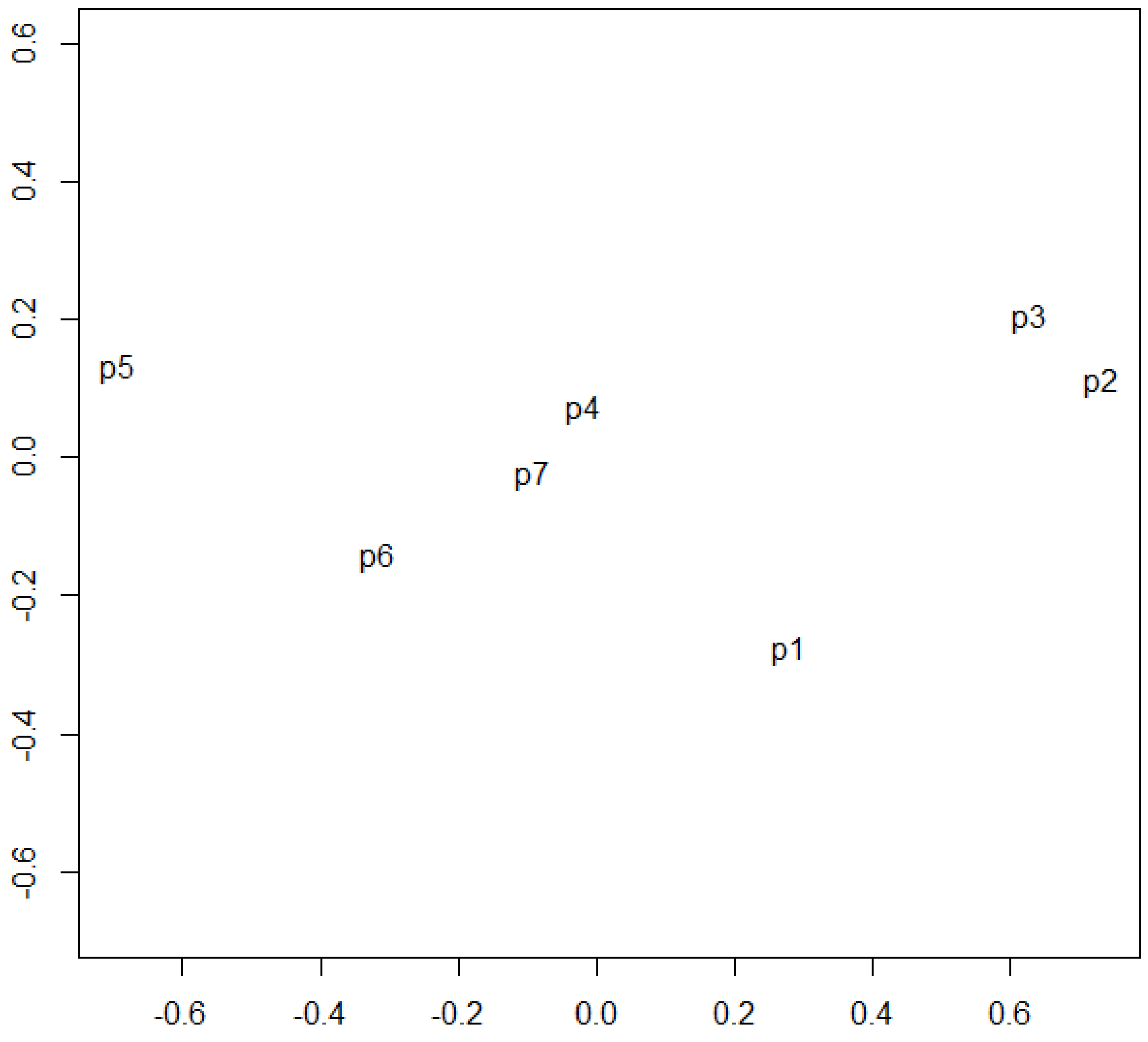


Figure 2.10: The opinion landscape from the reciprocal averaging method

According to the distance matrix, the results for this dataset seem to trend in the right direction but do not quite match the distance matrix. If the distance from P2 to P1 (5.7) and to P3 (2.2) are considered, the proportions do not seem to be correct.

Detrended correspondence analysis

As pointed out before, RA is an alternative way to reach the same output as CA. The problem with the arch effect is tackled by detrended correspondence analysis as proposed by Hill and Gauch (1980) although it is not free of faults. According to Palmer (2000) and Lindzey (2005), the procedure of detrending can skew the underlying data which may mean that the actual structures are eliminated.

There are two ways of detrending but the most common ways is by segments (Palmer, 2000). Detrending by segments means that the first ordination axis from RA/CA is divided into segments which have a couple of data points. Within each of these segments, the according ordination score for the second axis is adjusted (some value is added or subtracted) to ensure that all scores in this segment have the average of zero. Thereby, all the segments are brought down to similar levels near to the ordination axis. This technique is applied to all iterations of the RA algorithm, except for the last iteration before convergence. The output is an ordination axis that reflects the distances between the points in the higher dimension more accurately (Gauch, 1982). Detrended correspondence analysis (DCA) is thus a deliberate skewing of the data used for the algorithm to reduce the likelihood of the arch effect.

There are conflicting opinions about DCA, Lindzey (2005) suggests opinions vary from that CA is not valid without detrending or that the process of detrending could be omitted entirely. Ultimately, using this method it is possible to determine if there is any effect (arch, horseshoe or something new) that may be corrected.

The result of performing a DCA is illustrated by applying it to the dataset presented at the beginning of the chapter. Since the algorithm requires all values to be positive, the number four has been added to the values of the matrix. This does not affect the distance matrix. The result is as follows:

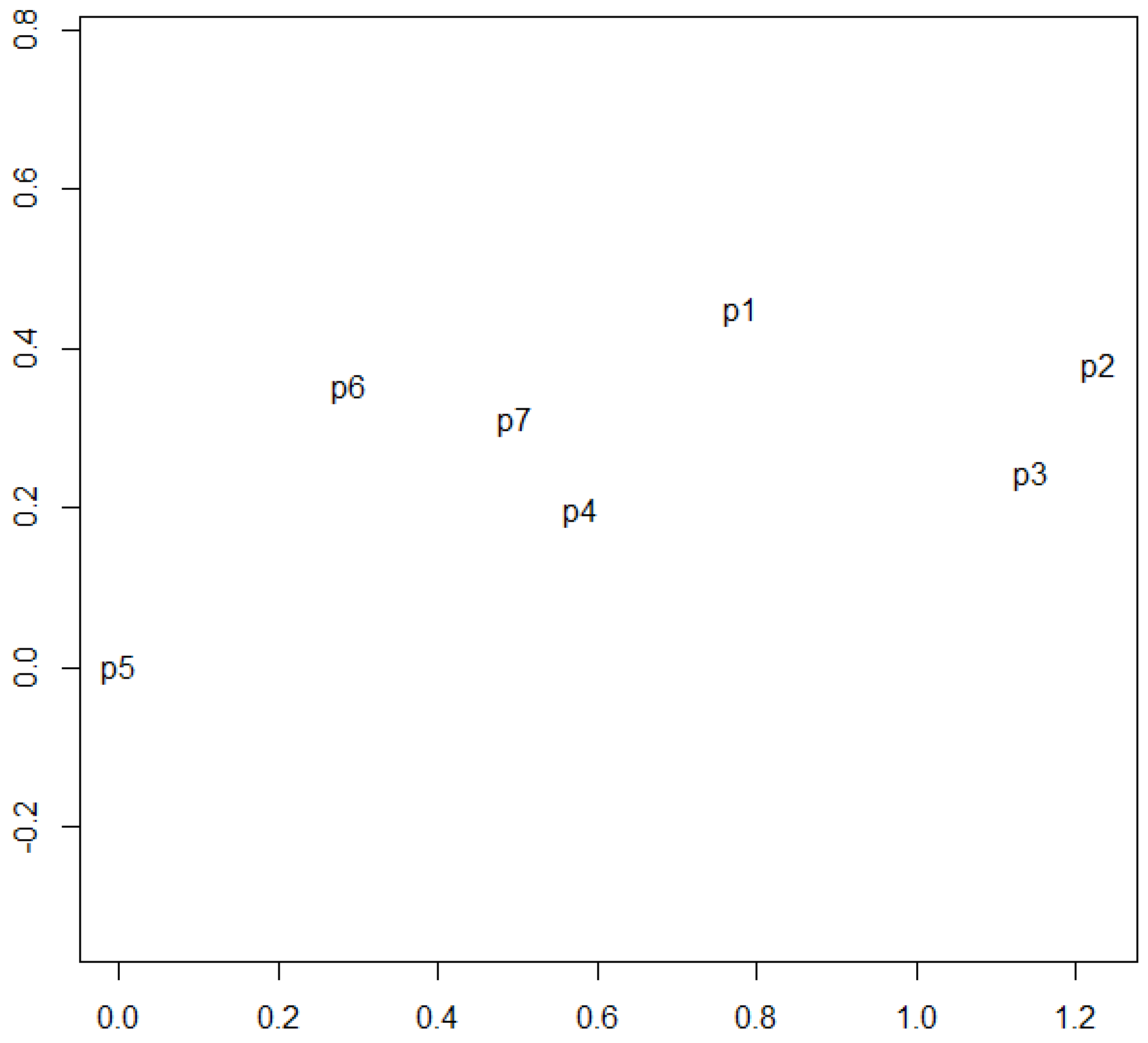


Figure 2.11: The opinion landscape from a detrended correspondence analysis

Although the result looks different from RA, the arrangement of the points has been merely flipped horizontally. According to the distance matrix, the results for this dataset also seem to trend in the right direction but the proportions do not seem to be correct for the same reasons as were discussed for RA.

2.5.1 Other multivariate approaches and methods

Multivariate regression

Multivariate regression (MVR) appeared to be a good method to explore because a basic linear regression resembles the first component of PCA. Visually, a linear regression analysis for points in a two-dimensional plane is used to find a line through the cloud of points that best fits the data.

MVR procedures require a formula from a model. For example, if one were trying to explain the ethnicity of a person given some attributes such as hair colour, skin colour, height etc. The formula would be created as follows:

$$\text{ethnicity} = \text{X} \times \textit{hairColor} + Y \times \textit{skinColor} + Z \times \textit{height}$$

Thereafter, some data is needed. 100 people were asked and observed, and their hair colour, skin colour and height were documented. They were also asked which ethnicity they are. Those non numeric characteristics are then assigned a number in a gradient: red hair is assigned as one, blonde hair is assigned as two, brown is assigned as three; Caucasian is assigned as a one, Hispanic as two, etc. Given this data from 100 people, MVR can be applied to see which of the three characteristics explain the ethnicity the most. The output is the X, Y and Z weights for the observable characteristics which make the model complete. This model can then be used for prediction. For example, if a new person is observed one could input the three characteristics into the formula and calculate the result. Depending on which range the result falls in, the new person can be categorised in a ethnicity. Thus the model allows the researcher to "predict" a new person's ethnicity without having to ask for that information. Evidently, there are other applications which do not only seek to capture missing information; this makes the model more useful than in the example.

To be able to perform MVR on the opinion landscape a formula has to be created:

$$\text{S}_1 = X \times \ S_2 + Y \times S_3 + ... + A \times S_n$$

Here it is assumed that the rating of the first Statement is explained by a weighted combination of all the other statements. This is plausible in the context of the TBDis system as the first Statement is always the main hypothesis of the debate which is explained by Statements of the Participants. The difference is that in the formula the structure of the debate is not respected. The TBDis system is constructed so that each Statement is recursively explained by their (sub-)debate. In other words, in the tree structure, S_4 might be explained by S_5 and S_6 before it is a factor to explain S_1. However, in the formula, it is as if all Statements were directly beneath the main hypothesis since the formula reflects that a concatenation of all statements explaining the main hypothesis.

In short, given this formula, MVR treats the main hypothesis differently to all other Statements. This distinction is not made in PCA and other multivariate methods. They rather try to find an optimal geometrical position for the axes to explain most of the variance or correlation.

The output of the MVR using the aforementioned formula is illustrated by applying it to the dataset presented at the beginning of this chapter. The result is as follows:

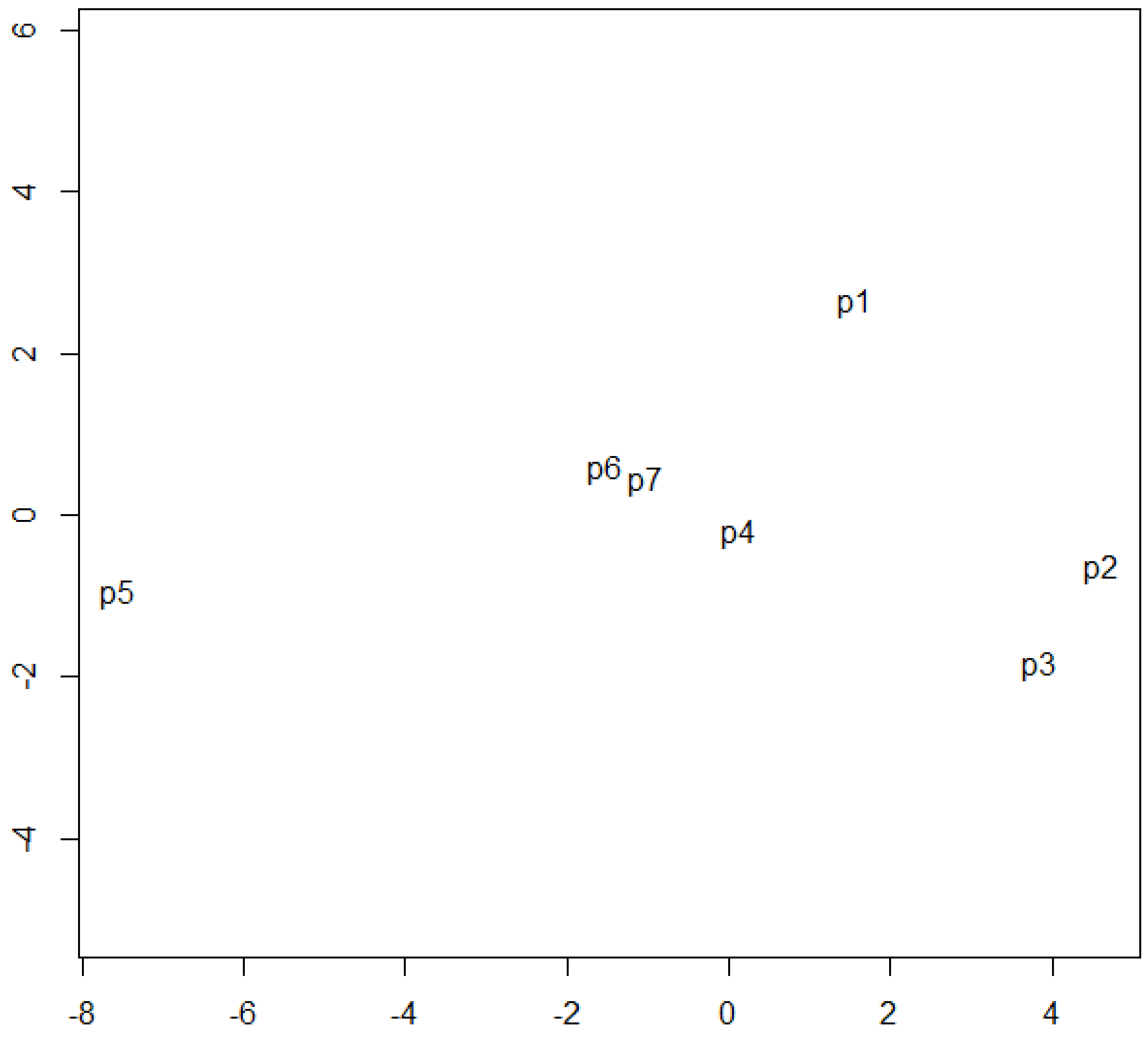


Figure 2.12: The opinion landscape from a multivariate regression

This output also goes in the right direction. A visual comparison to RA and DCA reveals that greater distances between Participants are emphasised more, which results in the positions of P6, P7 and P4 being closer together.

Multidimensional scaling

Multidimensional scaling (MDS) comes in multiple flavours. The metric version of MDS is also called Principal Coordinates Analysis (not to be confused with Principal Component Analysis). If Euclidean distances are used, the results of this method are the same as of a PCA, where the arch effect can be observed. The non-metric alternative (NMDS) employs a different process to reach a goal, which avoids the arch effect altogether. In this chapter, the second method will be described.

An application of the NMDS starts with the creation of a Participant space which is in the desired lower dimensionality. In this empty space, the Participants are assigned random coordinates. To ensure that the Participants in the new space have accurate distances between them, the distance matrices of the original space and the new space are compared. The coordinates in the new space are adjusted in order to make narrower the gap between both distance matrices. The principle is simple: the coordinates in two spaces with different dimensionality cannot be directly compared to each other but because their distance matrices have the same number of rows and columns, they are used for the comparison.

It is important to select the right amount of dimensions for the lower dimensionality space because in this method all the axes are equally as important. This is unlike PCA and other methods where the first axis has more information content as the others. In other words, the NMDS should not be used for a projection into three dimensions which is followed by the use of the first and second axis for the two-dimensional version. Instead, the NMDS should be used again for two dimensions.

The first step is to define a lower dimensional space Q' (for example, two dimensions) where all Participants have random coordinates. Since this does not reflect the real positions in the multidimensional space Q, the positions have to be adjusted. Both distance matrices D and D' are calculated. A "stress" function is used for the comparison. This function produces a value which is higher the greater the dissimilarity between the matrices is. Such a function can be represented like this (Kruskal, 1964):

$$S = \left[\frac{\sum_{i=1}^{n} \sum_{j>i}^{n} (d'_{ij} - d_{ij})^2}{\sum_{i=1}^{n} \sum_{j>i}^{n} d_{ij}^2} \right]^{1/2}$$

The formula looks complicated, but is very simple. Since the distance matrix is symmetrical and the diagonal is zero, only the values in one of the sides are used in the calculation. The values from D are subtracted from D' element-wise. In the end they are summed up. The root and squaring ensures that the distances are all positive. Just like with calculating the distance matrix, there are different versions of stress functions. This is just one example.

Based on the output of the stress function, the coordinates of the Participants in the Q' plane should be changed, the distance matrix D' recalculated and the stress function should be run again. For optimisation, the procedure is repeated until the stress is minimised. Borg & Groenen (1997) describe possible minimisation algorithms.

The coordinates of the Participants on the last Q' plane should reflect a proper projection of the original multidimensional space unless the stress value is still high. A high stress value indicates that the distances are not similar in both spaces, which means that the Q' plane is a poor representation of the original Q space. This may be due to a local optimum (Green & Rao, 1972). The rule of thumb is that anything below 0.1 is good and anything over 0.15 is not good enough. Even if the stress is rather high, the overall pattern is preserved because larger distances between Participants are less prone to have errors than shorter distances. In general, larger distances tend to be more accurate (Borgatti, 1998). It is also important to note that just like ordination methods the axes and orientation of the resulting Q' plane have no special meaning in the context of the original Q space.

Before illustrating the result of a NMDS, it is again important to point out that given a random initial condition and the possibility to converge to local optimum, the results of this method are non-deterministic. Running the same method on the same data might deliver different results. Some results are as follows:

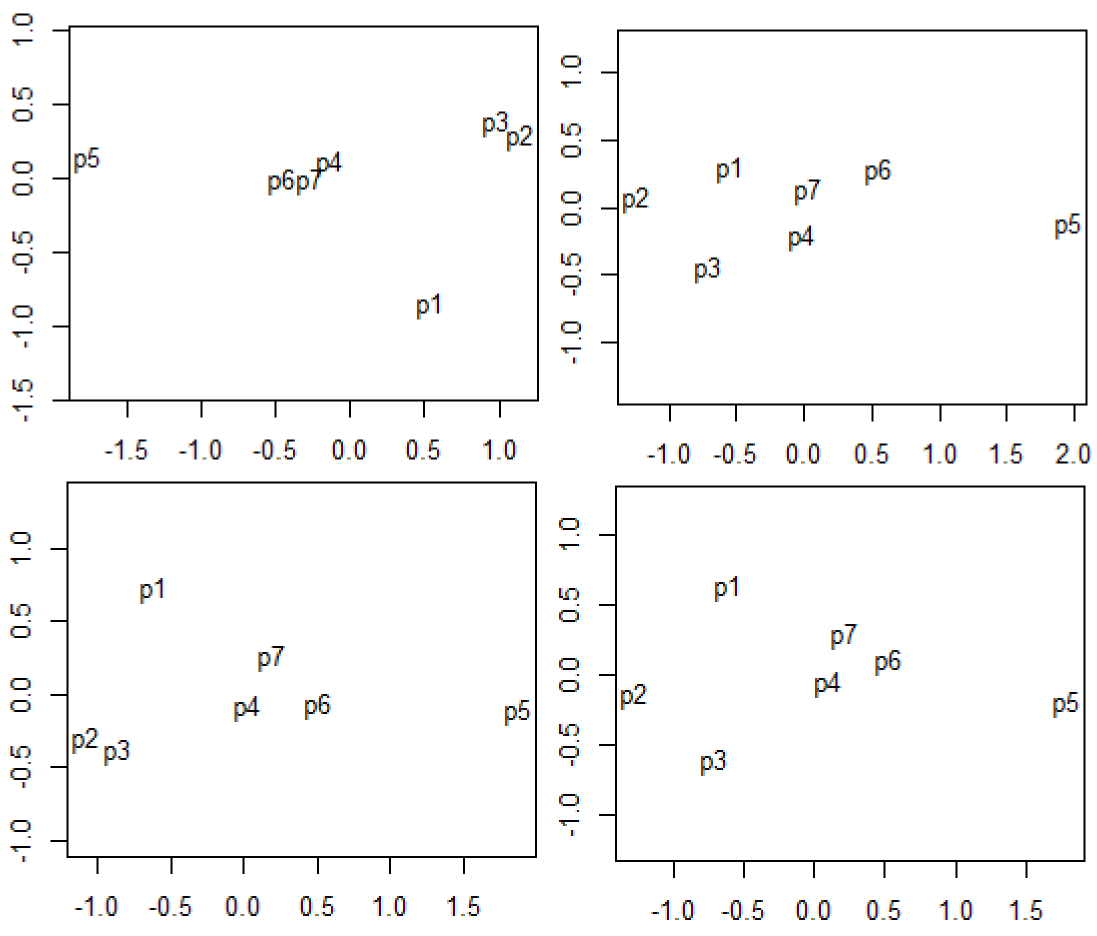


Figure 2.13: Four different representations of the opinion landscapes from the non-metric multidimensional scaling method

These are four possible results from a NMDS. At first glance, all four seem to have the same pattern as the previous methods. The closeness of P7, P4 and P6, P2 and P3 and the distance between P2 and P5 seem to be correct. Since the scatter plot is made to show to Participants and positions do not have to be perfect, this method seems to be suitable. Nonetheless, it is probable that clustering algorithms will render different results for the upper right scatter plot because of the closeness between the Participants on the left side.

Dimension clustering

This thesis has two goals. One is to be able to reduce the dimensionality of the opinion landscape so that it can be visualized for the participants. The second aim is to cluster the participants into fractions. Looking at both goals in the sense of the example matrix (figure 2.3), the first goal is about reducing the number of columns (Statements) from six to two while calculating new values in order to plot the coordinates. The second goal is about reducing the number of rows (Participants) from seven to a number based on the correlation between the rows while maintaining a record of which rows are the most similar to each other. In other words, the tool for the latter (cluster analysis) also reduces the amount of objects. Thus it is plausible that there is a way to use it for the first goal as well.

However, the aforementioned restrictions have to be taken into account: the requirement of the second goal to keep a record of who gets clustered with whom is no longer important. Instead, the existence of representative values for each cluster to ensure that the coordinates can be plotted is more important. In a two-dimensional Participant space, one way of clustering may be as follows:

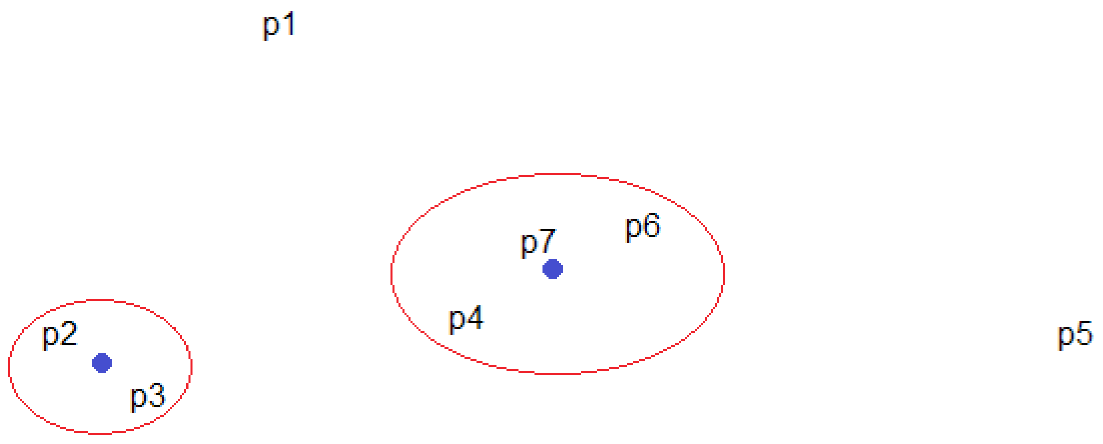


Figure 2.14: Identifying the centroid of the clusters

In this example, there are two clusters with two and three Participants and two outliers. One way to have representative values for the clusters is to determine the centroid as outlined in the discussion of PCA (see blue dots on figure 2.14). These dots should not be considered real Participants but are used for the purposes of clustering Statements for dimensionality reduction. The coordinates of the dots are a combination of the coordinates of the Participants in that cluster and thus they can represent the position of multiple Participants. Clustering statements and taking the cluster's centroid as the new position allows for dimensionality reduction.

As will be discussed in the next chapter, there are different methods for clustering. Most methods produce different amounts of clusters when analysing datasets. Since for the visualisation of the opinion landscape, exactly two Statements (two dimensions) are desired, a method has to be used that consistently produce two clusters. This method is k-means. Just like with NMDS, one of the parameters is the amount of dimensions/clusters that the output should have. The method will be described in more detail in the next chapter.

K-means is used to reduce the amount of Statements of the dataset that was presented at the beginning of this chapter to two clusters, and using their centroid renders a matrix with seven Participants and two Statement clusters. Plotting the coordinates using both centroids as X and Y values renders the following results:

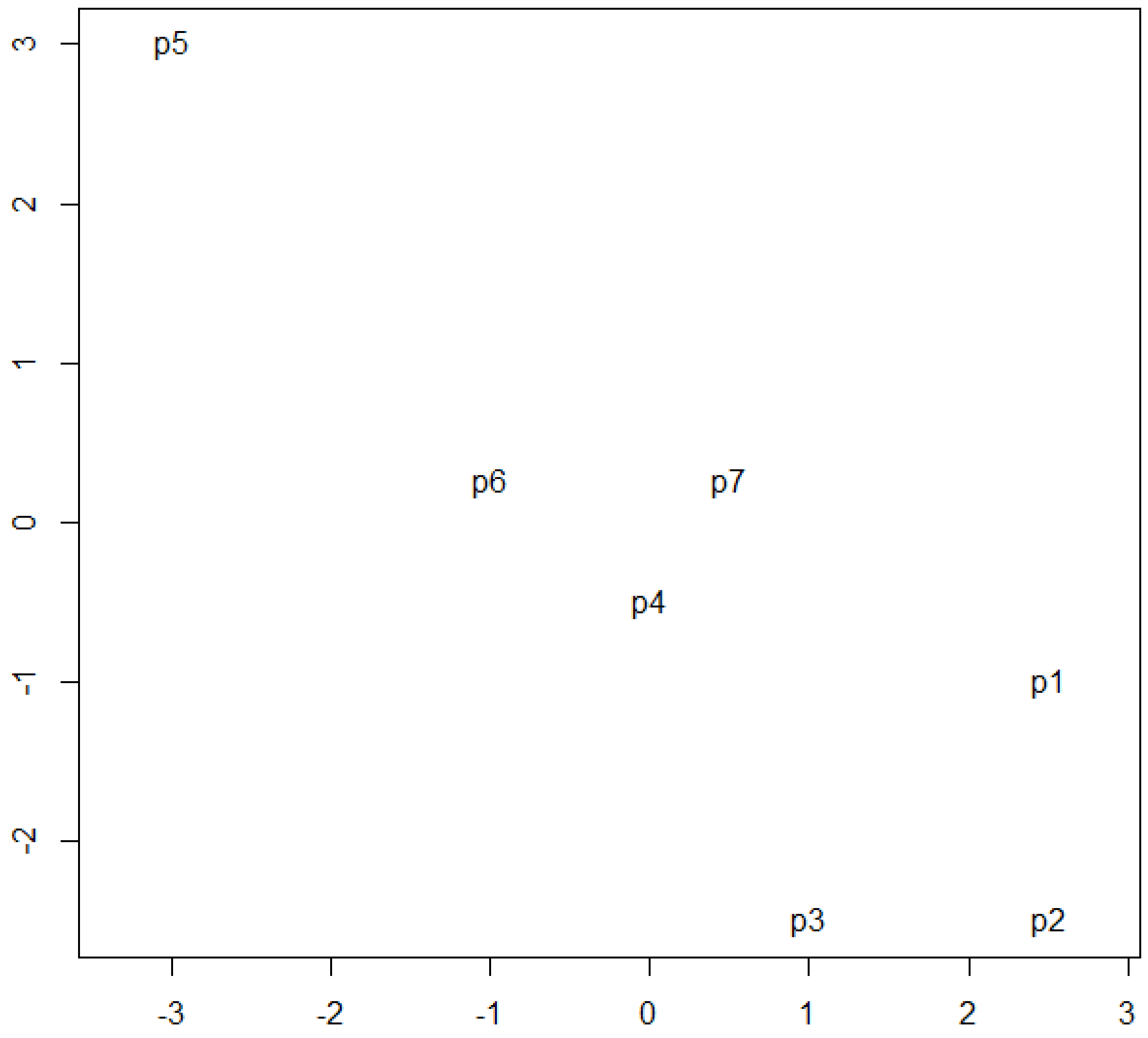


Figure 2.15: The opinion landscape from a k-means dimension clustering

This output looks slightly different from the output of the ordination methods. For example, the triangle between P1, P2 and P3 does not seem to have the correct proportions. Nonetheless, the result is surprisingly good for such a simple method.

2.6 Multivariate methods for clustering similar data

To reach the second goal of forming fractions, clustering methods are a suitable tool. The premise is that given the coordinates of data points in a multidimensional space, it is possible to explain which data points are closer to each other.

An array of different clustering methods can be used for highlighting which data points are similar to each other. In a cluster analysis, the data points are grouped according to their relative distance to each other. Generally, data points in the same group have coordinates which are more similar to each other than to data points in different groups. This may seem intuitive but it is not necessarily always the case (see density-based clustering). An implication of a cluster analysis is that elements inside clusters should be rather similar (homogeneous) and elements in different clusters should be rather distinct (heterogeneous). Cluster analysis can thus help classify observations into groups which are unknown prior to analysis (Gauch, 1982).

It may appear trivial to cluster data points together. However, this is not the case as will be shown in the following example. Assume we have the following output of an ordination that has reduced the dimensionality of a multidimensional Participant Space to a two-dimensional plane:

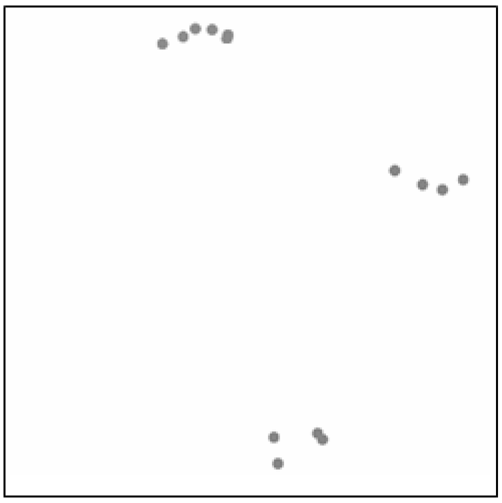


Figure 2.16: Example data points

In this example, most people would identify three clusters because there are three groups that are obviously far apart. Yet if the computer focuses on the group on the bottom of the figure it can will suggest that there are actually two clusters in the bottom part. So does the figure as a whole have three or four clusters? The highest degree of detail would have every data point as the sole point in a “cluster”. On the other hand, all data points together form only one cluster which is marked by the black square.

To solve this problem, the analysis should have a criterion that describes what a cluster is. More specifically, a threshold for the grade of detail must be defined. Most algorithms have either a stopping criterion or another variable for the amount of iterations to be done by the algorithm.

One can also note that clustering methods are used to reduce the amount of data by producing few data points. Following such a process, one representative data point for a whole cluster is used in further calculations.

Clustering can also be used for detecting outliers. One way would be to cluster data points until there is only one left. This one should be the outlier. Clustering is regarded to be more efficient than ordination for identifying outliers (Gauch, 1980).

In different clustering methods the relationship between data points is considered differently. For example, the following two images in figure 2.17 depict the output of two different algorithms. It is impossible to determine which one of the two clustering methods is correct because it depends on the context.

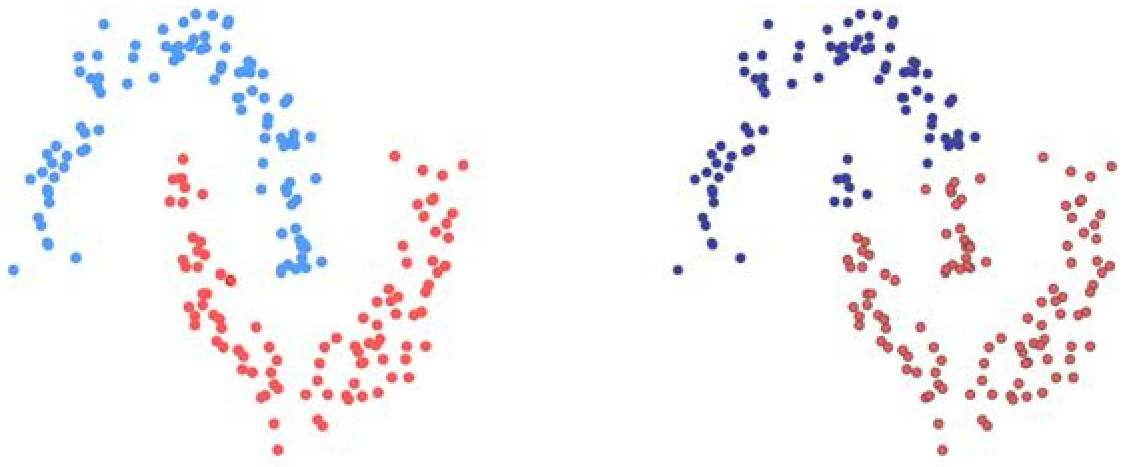


Figure 2.17 (a) Figure 2.17 (b)

So far, an example in clustering data points in a two-dimensional space has been discussed. However, as we have seen before, the two-dimensional space as output of a projection from a multidimensional space can contain errors because it simplifies the dataset. Using the original coordinates in the multidimensional space rather than the error-prone coordinates would yield a higher fidelity for the output of the clustering algorithm. This is independent of the clustering method used.

Based on this knowledge, clustering algorithms could be used to evaluate the quality of the projections from the ordination algorithms. Firstly, the clustering of the multidimensional dataset (higher fidelity) would be performed and each point would be marked with an indication of which cluster it belongs to. After the dimensionality reduction, thereby reaching a lower dimension, a second cluster analysis using the same method would be performed. The difference between both analyses (I.e. how many data points are in different clusters) could be used to grade the method for dimensionality reduction algorithm. For a subjective evaluation, the output of the ordination could be plotted and the data points could be colour-coded according to the first cluster analysis. If the projection is of good quality the plot should show neat clusters.

Hierarchical clustering

One of the most basic methods for cluster analysis is hierarchical clustering. The agglomerative method starts from the bottom up; the data points with shortest distance to each other are continually grouped, starting with the shortest one. Distances can be extracted from the distance matrix used for ordination. The output is a so-called dendogram which shows the groupings done in the iterations. The top-down approach is called divisive clustering and starts with all data points in a single cluster and splits them based on distances. This also results in a dendogram (Blei, 2008).

For both methods, a stopping criterion has to be applied or else all data points form one cluster or every data point forms a cluster by itself. One idea for a stopping criterion can be to stop if the distance between the centre of the current cluster and the centre of the nearest cluster is greater than the diameter of the current cluster.

Another method of selecting the level of detail is looking at the p-value. This value indicates how strong the cluster is supported by data. In the work of Suzuki and Shimodaira (2011), two types are identified; AU (Approximately Unbiased) and BP (Bootstrap Probability) values, and the authors suggest using AU.

Applying hierarchical clustering to the data presented at the beginning of this chapter renders the result in figure 2.18.

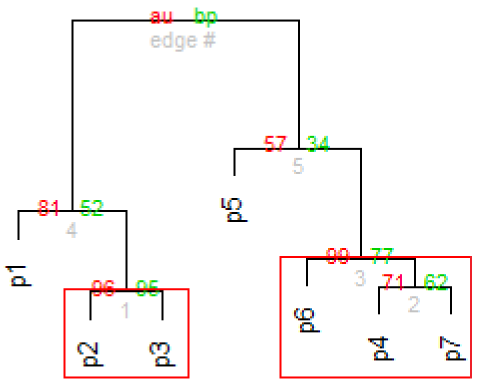


Figure 2.18 (a): Hierarchical clustering dendogram

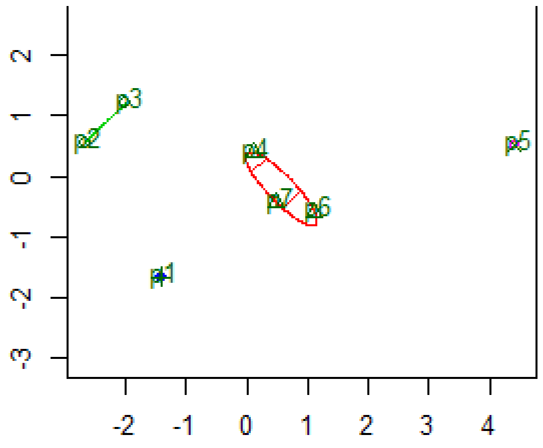


Figure 2.18 (b): Hierarchical clustering on PCA

Figure 2.18 (a) is the dendogram representation of the cluster analysis. Here it can be observed that P2 and P3 form the first cluster, then P4 and P7, and after that, P6 joins the second cluster. According to Suzuki and Shimodaira, clusters with an AU P-value of more than 95% are strongly supported by data, thus this value was selected as a stopping criterion. All clusters with a higher value than 95% are highlighted in figure 2.18 (a) and on the scatter plot in figure 2.18 (b). The latter is a representation of the clusters in a two-dimensional plane achieved by a PCA.

K-means clustering

As discussed in section 2.5.1, k-means requires a pre-defined number of clusters (K). This method is non-hierarchical and does not have a dendogram as an output.

The process is as follows (Schenker, Bunke, Last & Kandel, 2005):

1. Initialise K number of clusters and assigning random coordinates to their centroids.
2. Assign each Participant to the closest cluster centroid.
3. Based on the coordinates of the Participants in a cluster, calculate the centre of the data cloud from this cluster and relocate the cluster centroid to that position.
4. Repeat steps two and three until no Participant is assigned to a different cluster in step two.

The greatest drawback is that one has to a-priori set a number of clusters. A possible solution is to use an iterative version of k-means, introduced by Boecker et al. (2005), which is hierarchical and also iterative. Firstly, a regular k=2-means clustering is performed. Then, recursively for each cluster, a k=2-means clustering is performed until a stopping criterion is reached. This could be when the distance between the farthest data point in a cluster goes below a certain threshold, which means that clusters are created as long as data points are enough far apart.

A second method is to perform several k-means clustering from k=1 to the number of Participants. Then the quality of each clustering is assessed in order to select the best one. Here, a low sum of squares of the distances within the groups is the aim (Everitt and Hothorn, 2009). A lower value means that the cluster is relatively compact. Certainly, the lowest value is reached when every Participant is in an own cluster. The curve for the data presented at the beginning of this chapter can be seen in figure 2.19 (a). Since it is not desired that each participant is in an own cluster, the criteria is rather having as many clusters as necessary so that the next greater amount of clusters (for example, comparing a k-means with k=4 to k=5) does not add much more information. This is certainly a subjective matter. In this case, the threshold is set when the slope between two amounts of clusters rises above -10. This is reached between k=4 and k=5. Therefore, the amount of clusters selected in relatively objective way is four.

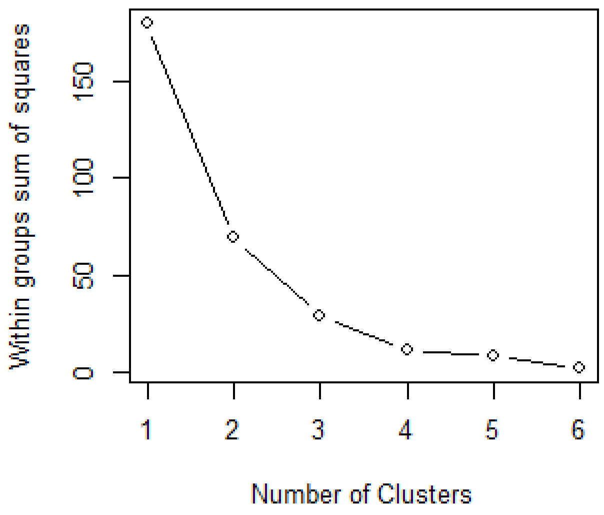


Figure 2.19 (a): sum of squares for different K values

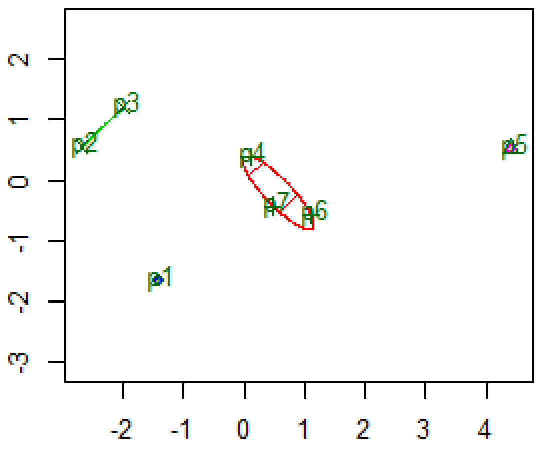


Figure 2.19 (b): K=4-means clustering on PCA

The result of a k=4-means clustering can be seen in figure 2.19 (b). Given the data from the beginning of the chapter, the clustering is identical to a hierarchical clustering.

Density-based clustering

Using density-based clustering, the decision if a data point belongs to a cluster is not based on the distance between two data points but rather on the amount of data points within a short distance. The density is defined as the number of points within a specified radius (Ester, Kriegel et al., 1996).

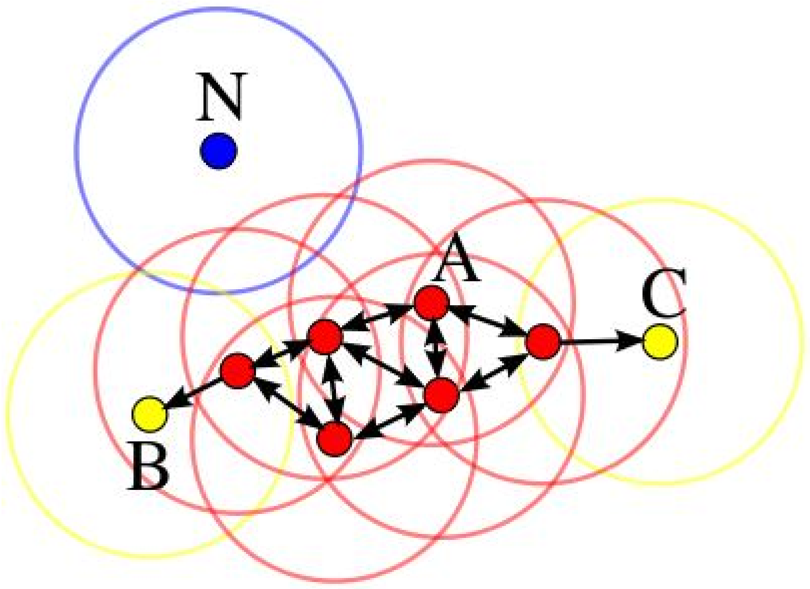


Figure 2.20: Classification of points in dbscan

Dbscan by Ester, Kriegel et al. (1996) is a popular algorithm for this method. It connects points in a cluster, which are positioned at distances to each other that are below a threshold (the radius) and satisfy a density function (the minimal amount of points). These two conditions are determined for every point. If the point satisfies both conditions (a red point in figure 2.20), a cluster is formed. It is then marked as a core point. Points that are in the radius of core points but do not satisfy both conditions are mere border points and belong to the cluster of the core point. Other points are outliers. The two conditions are parameters that are set a-priori by the researcher. Dbscan is usually used for datasets with many points. Often the minimal amount of points is set to 20. Since debates can have a fairly low amount of participants the value one (“1”) is used for the minimum amount of points. The mean of all the distances from the distance matrix is a suitable value for the radius because the mean itself is a value that somewhat reflects the density. For example if there are two identical configurations of participants in a multidimensional space, the mean is the same. If there is an outlier in one of these two configurations, the density of the configuration overall has sunk and the mean has risen.

Due to the features of this method, the shape of the clusters can be arbitrary and a centre of a cluster can be quite far away from any of the data points (i.e. if the cluster forms a concave half-circle).

The result of performing dbscan of the data from the beginning of the chapter is again identical:

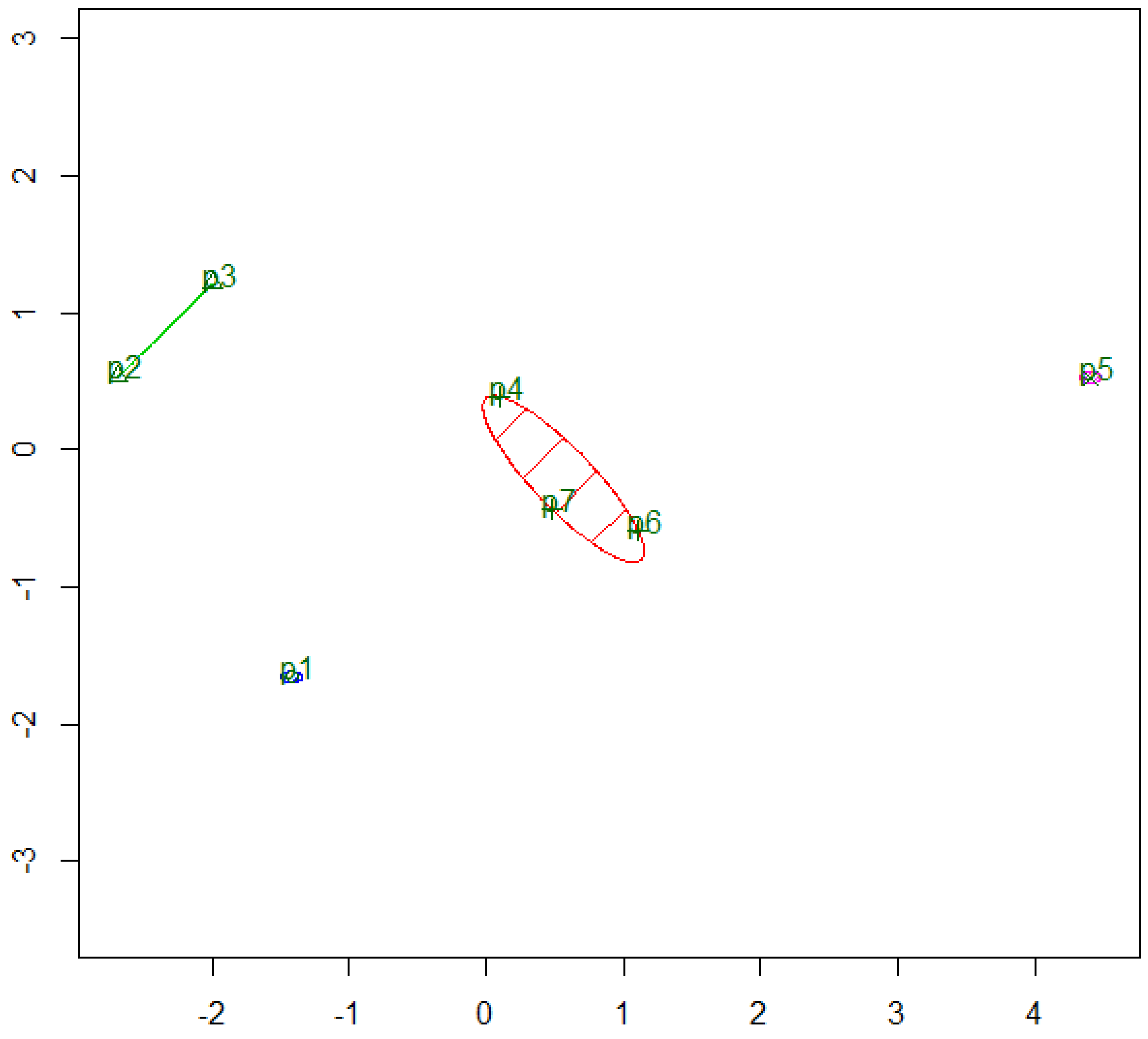


Figure 2.21: Dbscan clustering on PCA

2.7 Discussion

Gauch (1982) provides several recommendations on how to prepare the data prior to using multivariate methods. For instance, it is recommended that the values in the species-by-sample matrix are logarithmic. This is because one often finds low numbers of species and then there is a species with an absurdly high number. The recommendation of using logarithmic numbers is to focus on the smaller numbers. This does not apply to the Participants X Statements matrix because participants are not counted per Statements but rather by their opinions. It is also plausible that given a rising number of statements in a debate, participants might not have the time or motivation to rate all of them. This renders zero values for ratings in the Participants X Statements matrix. Gauch (1982, p213) suggests that in a Species X Sample matrix, species with zero values should be either erased or their values changed by an algorithm that tries to guess correct values.

Moreover, by using some multivariate techniques (mostly ordination techniques) such rare species are perceived as outliers who can distort the output scatter plot. In this case, Gauch (1982, p215) argues that outliers should be a-priori identified and removed if an ordination technique is to be used. To identify the outliers, a hierarchical clustering technique can be used. If the results are to be analysed for research purposes, removing outliers would be suitable in order to focus on the differences between the opinions of core Participants (according to Gauch). However, if the scatter plot is to be displayed in a debate report, removing the outliers would result in some Participants being missing from the scatter plot. Furthermore, outliers in the Statement space are Statements that contain significant information content. If two Statements are similar, not much information content is added by the fact that there are two Statements. Removing outliers in the Statement space would affect the Participant space negatively since it is likely that the difference between Participants is given by the outlier Statements. Thus outliers in the Statement space and Participant space provide valuable information for the opinion landscape. The recommendation depends on the application of the analysis and neither seems to apply to the TBDis system.

2.8 Design ideas for application of the methods

There are many methods for multivariate analysis which as a whole allow for a great degree of freedom for the researcher. The methods themselves are also very flexible. For example, weights in the “weighted averages” method can be based on whatever is deemed important in the context of the analysis. Moreover, if methods using distance matrices are to be used, a selection can be made from many different formulas in order to calculate the distances and dissimilarities.

As stated in section 2.4.2, methods of ordination, clustering and classification can be used in conjunction. Thus yet another degree of flexibility is given, for example, by being able to apply one type of ordination and then another type based on the output of the first one.

For instance, some ordination methods use the Participants X Statements data matrix while other methods use the dissimilarity matrix. It is conceivable to develop an approach in which firstly a method based on the data matrix is applied, followed by a calculation of the dissimilarity, which then is the basis for another ordination method.

Another approach would be to selectively apply some methods to particular data. For example, performing a cluster analysis to cluster only Participants with more than [number of Statements]/2 non-zero values. These are the Participants that have given more opinions than other Participants indicating a possible special role. A classification method could then be used for classifying the rest of the Participants into the existing clusters. Or using only the first kind of participants and above, counting the number of clusters and doing a k-means clustering for everybody where K equals the number of clusters found in the first step.

Moreover, for polar ordination, the poles could be pre-selected for the first ordination axis instead. As discussed before, Hyrenbach (2011) points out that this method is ideal for problems with definite endpoints. The pro and contra nature of the TBDis system is a strong indication that there are definite endpoints. One way to pre-select poles would then be to somehow identify the opinion leaders for the biggest fractions (requiring a cluster analysis to identify the fractions) and select those Participants as poles. This would combine a clustering method with an ordination method.

Another idea is to combine NMDS with other methods. If no specific initial condition is specified, random coordinates are assigned to Participants in the NMDS method. Yet this method can also begin with non-random coordinates, such as the result of a PCA or a MVR. Feeding in the result of a PCA or MVR as input to the NMDS nudges the algorithm in a particular direction rather than it reaching a local optimum. NMDS also turns deterministic. In other words, the combination of these methods would improve the output of a NMDS. The combination also helps to solve another problem: since details of PCA and MVR have been treated as a black box in the description of the methods, these methods may render unsatisfactory results given certain unknown conditions. In such a worst-case scenario, giving an unsatisfactory input to NMDS is no worse than the alternative of starting with random coordinates. Given that even in the case of random initial coordinates for Participants, NMDS renders a satisfactory result, a possible unsatisfactory output of a PCA or MVR must be systematically worse than random coordinates for the combination to render worse results.

In this chapter, it becomes apparent that there are many approaches to applying the methods of multivariate analysis. However, such a broad range of methods can have some adverse effects. Using an overwhelming number of approaches can be really confusing when trying to understand how and why they are used. Explaining and justifying the selection of approaches is thus critical.

3. Current debate structuring system

This chapter has a dual purpose. First the TBDis system will be presented. As the system provides the platform for debating and rating statements it is the overall context of the multivariate analysis. The context of the debate's content is important for the participants because it is a factor in deciding on which algorithm to use (Gauch, 1982). The context of the system itself is also important since it provides the structure that defines when and how one can rate a statement. Finding out how the system developed over time is the first topic. The second purpose is to describe and explain the concept of opinion landscape for the analysis is presented. In the end the clustering algorithm which is tailor made for TBDis is explained.

3.1 Introduction

The debate structuring system TBDis ("To Be Discussed"; formerly known as CANDis) is part of the motivation for this thesis. It was developed and described in 2010 by Kümin and is based on the CANDis ("Computer-Aided Normative Discourse") theory by Hilty (2010a) which will be described later in this chapter. The system is currently in continuous development by the Informatics and Sustainability Research group (ISR) at the University of Zurich. One major focus area of this project is the "Decision Support and Collective Intelligence" which explores the possibilities of using collective intelligence for sustainability projects (Hilty, 2011a).

The main problem that the system tackles is the lack of clarity in large debates about a topic. The assumption is that typically, a debate that deals with the pro and contra standpoints of a subject matter splits into different sub-debates around many of the pro and contra arguments. In a chronological linear debate as in a forum or in face-to-face meetings spreading into several days or weeks, this can cause misunderstandings. For example a participant in a discussion might forget other people's arguments and commences a topic out of attention which had already been discussed thus relating a new argument to a topic that had already been closed. Usually the arguments that participants remember are the more recent ones. This is due to the unstructured nature of debates (Belnap & Withers, 2008).

The motivation for the TBDis system is to target the aforementioned clarity problem in debates and harness other advantages of web based debates. By being on the web the debates are not temporally or geographically bound. Also there is no need to write minutes of what has being said since this is implicitly done when the participants enter their statements into the system. The debate structuring functionality gives a scaffold to participants as in the definition by Collins, Brown and Newman (1989). It provides guidance.

The TBDis system can be characterized as a Group Decision Support System whose goal is to eliminate noise and uncertainty from a discussion (Ito & Shintani, 1997). More specifically one of the main goals is to maintain the traceability to all key arguments which in the end are the basis of the final decision upon the subject matter. This traceability has to be granted to every participant and is achieved by structuring the debate. Moreover at any point during the debate participants have the option to compile a report. This report shows the fractions (clusters) of participants that have been detected by the system. The algorithm is described at the end of the chapter. In case there is no consensus about the topic of the debate, the report also shows which participants would have to change which parts of their opinions for a potential consensus to be reached with a minimum shift of opinion. The motivation for this feature is that even if the debate did not result in a consensus it still should have a valuable result. The report is a way to see the different opinions and find out why no consensus was reached (Hilty, 2011b). The output of the multivariate analysis of the opinion landscape is also supposed to be displayed in the report. If it performs better, it will replace the current version of the fraction analysis.

Mark Klein had his paper about a similar concept called the ‘Collaboratorium’ published in 2007. His system also enables on-line debates around complex topics. Some of his insights were that a visual argument map allows participants to see spots in the debates where argumentation or further research into the topics is needed. Moreover the pro and contra type of argumentation conveys the scientific way evidence-based arguments are formulated. This is implicit in the TBDis system as participants are not explicitly asked for sources for statements but it is assumed that they have a motivation to reach a consensus (Kümin, 2010). Another insight is that relatively more active users in a forum gain a lot of attention and talk time. In such a system as the ‘Collaboratorium’ and also TBDis, a participant that is relatively more active would positively contribute to the subject matter but not be able to raise his influence on the result of the debate by just getting more attention. Moreover, TBDis offers the option to be used in an anonymous mode, such that any potential influence of personal relationships or individual behaviour (other than formulating arguments that convince other participants) on the debate is suppressed.

3.1.1 Principles and core features

The broader goal of the system is deciding upon a subject matter with a certain circle of participants with ideally minimal effort. Maintaining traceability as a goal is one way to supports this vision. The theory also gives some principles that ease the approximation to an ideal debate (Hilty, 2010a):

- Autonomy: Every participant can freely choose the content and time of any type of contributions to the debate. The system has a to-do-list feature that gives recommendations on what to focus on but does not interfere otherwise.

- Equality: Anonymous debating should be possible if at least one user requests it. Anonymity reduces group thinking. Group thinking is a phenomenon when participants adapt their own opinion to a subgroup’s opinion and rejects external input (Philosophical Dictionary, 2009).

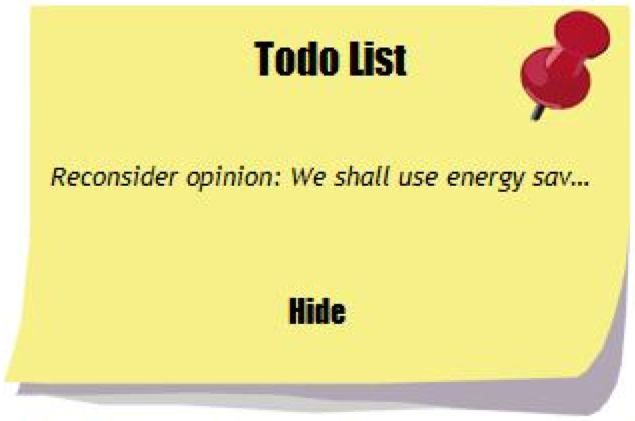
- Coherence: Revoking contributions cannot disrupt the structure of the debate. In that sense, removing a statement must also remove the whole sub-debate around that statement.

According to the autonomy principle, only the author of a statement can decide to remove it. Others have the opportunity to “adopt” the statement by becoming the new author if they want to keep that part of the debate.

- Transparency: The status of the debate should be visible to every participant. It is given by the navigable tree structure.
- Efficiency: The group should be able to reach a result with little effort. The feature of individual to-do-lists created by the system is the main driver for efficiency. Recommendations are a way to focus a participant on parts of the debate that have to be improved on because his or her opinion has the potential to make a difference.

The TBDis system was developed according to the above principles. By sufficing those and other minor requirements, the system became feature rich. For example instead of just allowing the author of a statement to edit their statements, there is also the possibility for other participants to send edit requests to the author. And due to constant synchronization with the server he would see the change request instantly without having to reload the page or leave and re-enter the graph. The synchronous collaboration is a feature that tackles another problem of complexity unstructured debates. Since the debate can span several days or weeks and the scope can grow indefinitely, it is plausible that participants stay on the website with the graph open for several minutes at a time while they looking for alternatives and reflecting or researching a topic (Althaus, 1996). All changes made by other participants are meanwhile synchronized to the clients such that everybody sees the same status of the debate. Multi-language support is also a feature.

The main feature besides the structuring of the debate is the moderation algorithm. Although the name hints at an automatic way of directly governing the debate, the algorithm is not meant to do that. It has two components: the to-do-list and the moderation itself. The to-do-list (See figure 3.1) is the front-end of the moderator that the participants see. Everyone has his own list; therefore it could be seen more as a personal assistant. Basically the moderation component monitors the current state of the participant’s contributions to the debate in relation to state of the debate as a whole and gives pointers on what to do next, i.e. it adds entries to the to-do-list prompting formal inconsistencies between the participant’s contributions or if a specific action would make the debate run more efficiently. For example a contradiction arises if a participant agrees to a pro and to a contra argument of a particular statement. The to-do-list is updated and actions for the resolution of this contradiction are suggested. Or if somebody has added a new contra argument to the participant’s statement he will be notified of that. He should then either oppose the contra argument or revise his opinion on the original statement. However, all such suggestions are pure recommendations and can in no way restrict the autonomy of the participant. The moderation algorithm is not trying to replace the decision maker but rather support him.



Reconsider opinion: We shall use energy sav...

Figure 3.1: The todo-list with recommended actions

Additionally to that the moderation component monitors the global status of the debate such as checking whether the debate has ended. It is important to note that the moderation algorithm does not analyse the content of the statements but rather the participants' ratings and the structure of the debate. There is no semantic analysis and each statement is seen as a black box. Another feature is the report generator which has been discussed earlier in this chapter. The topic of this thesis, the multivariate analysis, is a tool for improving the report generator. Similarly to the moderation algorithm, the ratings are analysed. Instead of generating recommendations for the to-do-list the analysis' output is to be presented in the report generated.

So far there have been stress tests and bug tests with 20+ participants but there have been no empirical studies otherwise.

3.2 The TBDis project

In this chapter the TBDis system will be presented in the context of the TBDis project.

3.2.1 Theoretical background

The system was built in the context of the TBDis project and was formerly known as CANDis; Computer Aided Normative Discourse. The goal was to create a theory for normative discourses and validate it through the development of a web based system.

A preliminary version of the theory already provides a debate structure based on the AND/OR tree structure. This structure is then expanded by a XOR connection due to the pro and contra nature of the debate structure described in the theory. Generally it defines what a statement is and how it relates to other concepts.

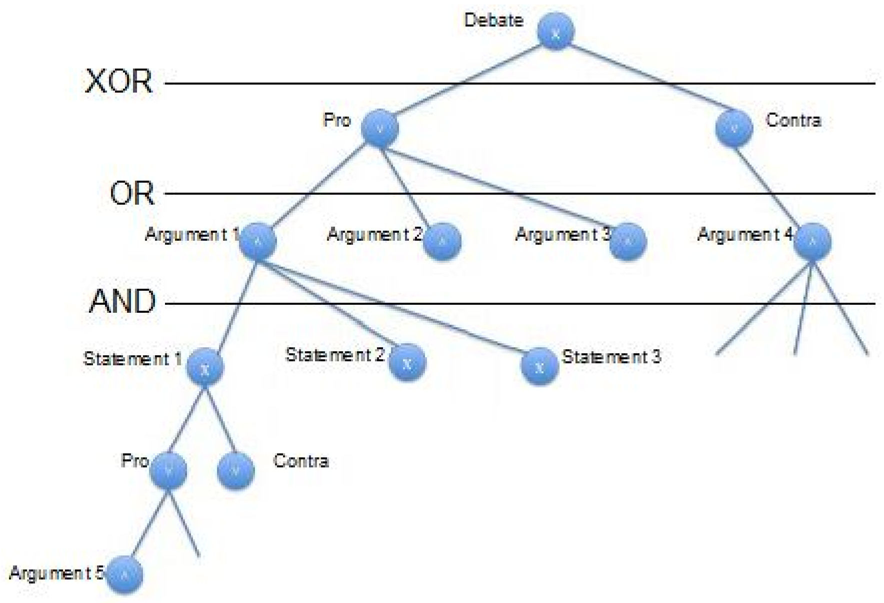


Figure 3.2: CANDis' theory structure

Figure 3.2 depicts the structure and relations between arguments and statements. The main hypothesis is a special statement because it is the root of the debate. The XOR division indicates that a consensus, an agreement, on the main hypothesis can only be reached if either pro or contra is supported. To support the Pro side, at least one pro argument has to be supported, thus the OR-connection. The same is valid for the Contra side. A pro or contra argument consists of statements of which all have to be supported thereby supporting the argument too (AND-connection). As such the statements may also be the root of a debate just like the main hypothesis. This recursive structure enables participants to debate on the arguments. The tree structure branches out or spans out infinitely and can grow to potentially infinite depth and breadth.

Detailed concepts of the theory will not be discussed here. One of the reasons is that the theory is still a work in progress and will be completed in the light of empirical tests using the prototype of the TBDis system. This fact also had an impact on the TBDis system as the implementation phase had volatile requirements. In some sense the theory and the system co-evolved over time influencing each other's maturity level.

3.2.2 Translation into the system

Although the theory treats statements as an atomic concept, there are two different types of statements in the TBDis system: Lead statements and additional conditions. Since it does not make sense to add an empty argument without any statements, the participant is prompted to enter a text when creating an argument. This first type of statement is known as the lead statement. It is created at the same time as the argument. Later on, the creator or any other participant can add more statements to the argument. These statements add information to the argument and are typically constraining the validity of the argument to certain conditions. For example, a contra argument for the statement "*We shall use energy saving lamps*" might be "*energy saving lamps have mercury inside.*". This is the lead statement of the argument. One could add pro/contra arguments to that and start a sub debate. An additional condition to this argument could be "*Mercury can leak into the environment*" (See figure 3.3). By explicitly stating it, participants can now add pro and contra arguments to it for example debating about the packaging and which regulatory institutions have approved it. Structure-wise, we now have two statements in the same argument- the lead statement and a condition. To support the whole argument, a participant would have to support both statements due to the 'AND' connection given by the TBDis theory. A further additional condition of this argument could be "*mercury is dangerous to humans*". However, if nobody introduces it explicitly, this indicates that there is an implicit agreement about the truth of this condition and no debate is necessary.

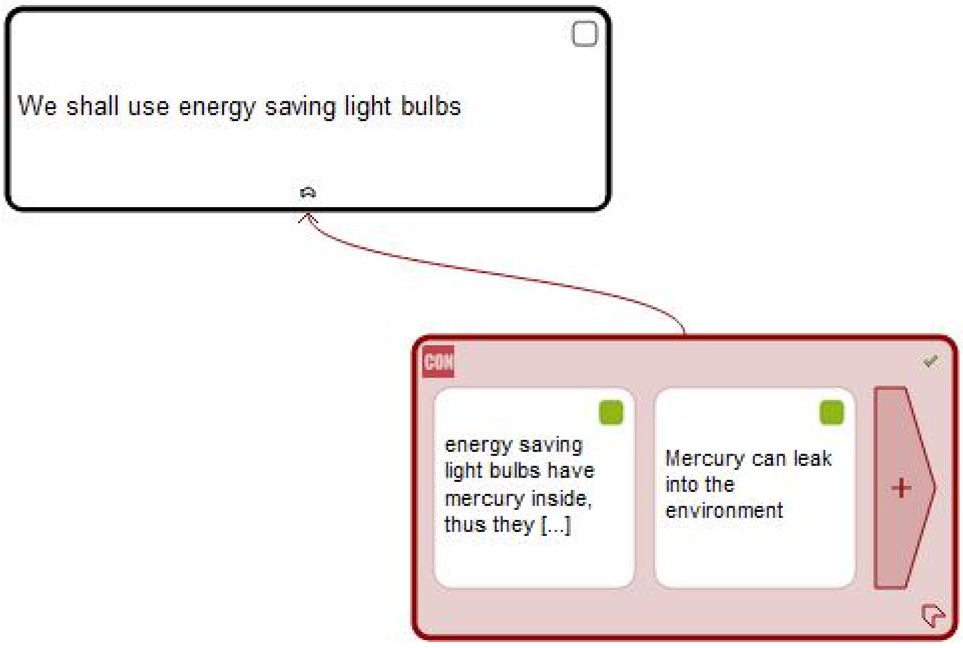

We shall use energy saving light bulbs
CON
energy saving light bulbs have mercury inside, thus they [...]
Mercury can leak into the environment
+

Figure 3.3: A contra argument with two statements

3.2.3 Interface

The user interface architecture of the TBDis system mainly consists of a website and the debate graph. The website has information about the system, functions for registering users, profile settings, creating a debate and showing the debates in which persons are currently participating in or simply put, showing active debates. The graph is shown when a debate is opened. Statements and arguments are shown as in figure 3.3. Statements being represented by the squares with the text of the statement as content and arguments being visualized as container of multiple statements. The graphical symbols are floating on a blank canvas. The arguments as well as the canvas itself can be dragged around in a method inspired by Google maps. They can be recursively collapsed and expanded by clicking on the arrows. There is also an alternative tree view for the graph in the form of a text based list. The report generator as a software component is separate from the component managing and displaying the graph and the website but reports based on the same data as the graph.

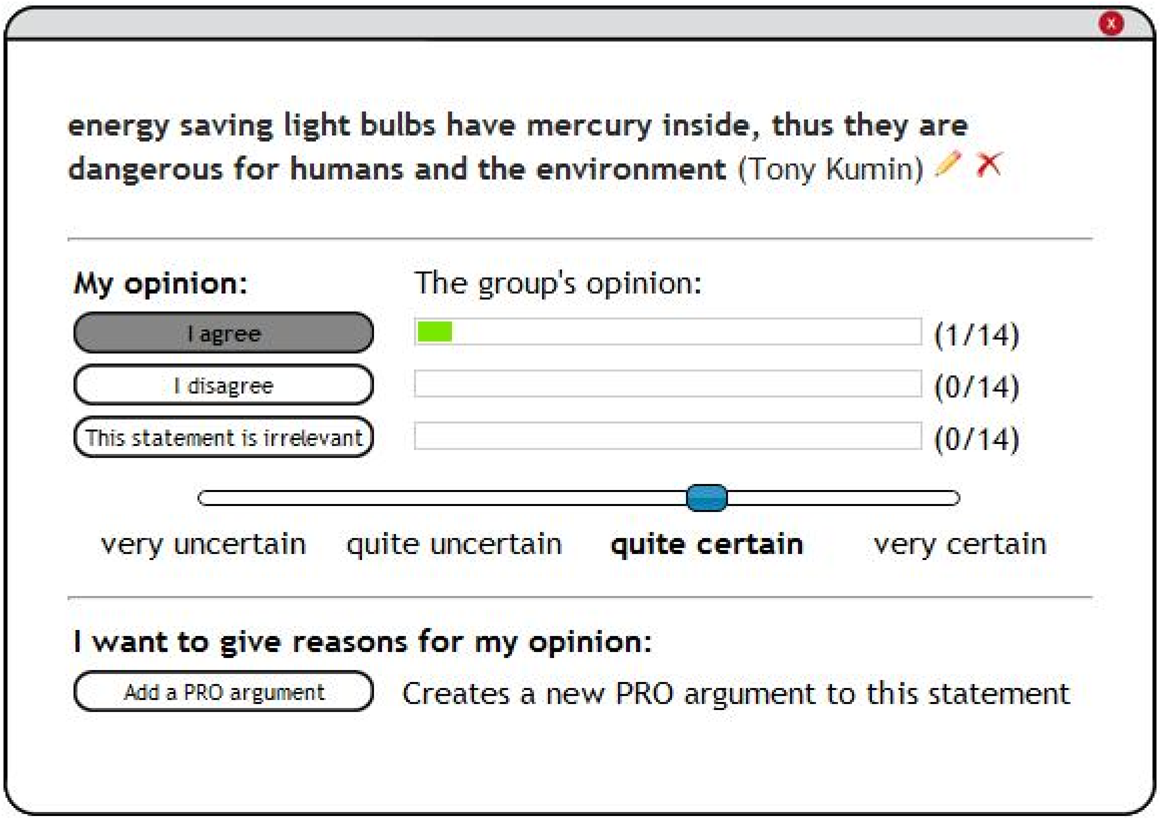

energy saving light bulbs have mercury inside, thus they are dangerous for humans and the environment (Tony Kumin)

Figure 3.4: The statement window

In the debate graph a statement window is opened by clicking on the statement (see figure 3.4). There, one can rate the statement and add arguments to give reasons for the rating. This is because although the system grants the participant many freedoms, he is encouraged not to create a contradictory argument to his opinion. For example if the participant is the author of the statement, he can delete it. This function is not unconditional; it depends on the debate about the statement to be deleted. If anybody has rated anything or added new arguments, then the statement cannot just be deleted. It is marked as “for adoption” such that someone can take over and be the future author of the statement. Rephrasing works similarly but additionally to that, anyone can rephrase a statement. In case the participant is not the author of the statement, a message with a rephrasing request is sent to the author instead.

As described before the to-do-list is the interface between the user and the moderation algorithm where users get recommendations on actions that would potentially speed up reaching consensus. This window is also floating on the screen but can be minimized if it is taking up too much space.

3.2.4 Software Architecture

For the front-end of the TBDis system the Dracula framework (Strathausen, 2010) was used. It consists of the two libraries Raphael (Baranovskiy, 2008) and GraphJS (Hellesoy & Hoover, 2006) which are both JavaScript libraries. Raphael is a library which simplifies the drawing of arbitrary shapes in a canvas. The Graffle plugin for Raphael was also used. GraphJS is a library that works with Raphael and Graffle to draw circles and edges as connections. Since its purpose is to draw graphs, there is also an algorithm for making sure the shapes don't overlap. Ultimately GraphJS was not used as a library but rather as a template. The whole code was adapted to fit its new purpose. The system was later refactored to work with jQuery (The jQuery Foundation, 2012). The back-end is in PHP and MySQL was used as database.

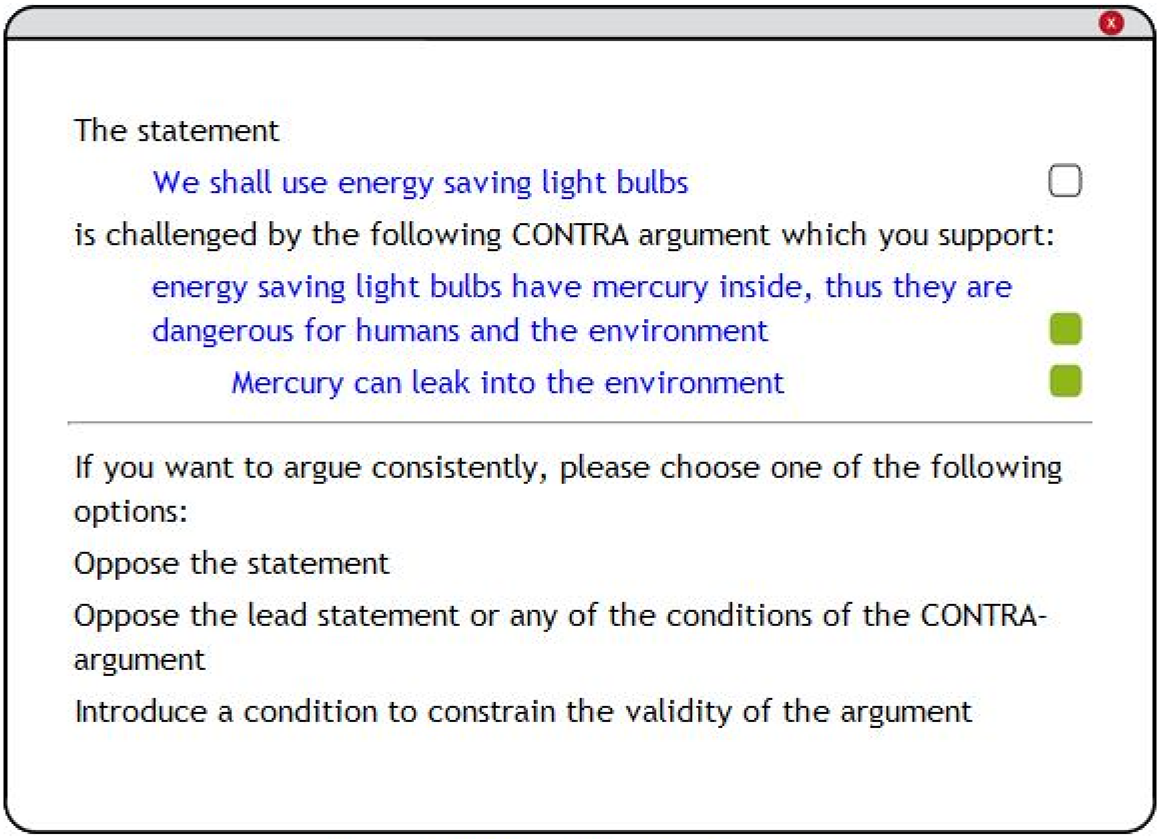
Even though debates are led in an asynchronous mode and can persist over days or weeks, it is plausible or even normal that participants work at the same time on a debate. Due to this collaborative nature, it is important to regularly synchronize the client's state with the server's state. The Ajax techniques provided by jQuery were used for such synchronisations. The asynchronous data retrieval enables the client's browser to ask for information from the server without having to reload the website (Garrett, 2005). It is thus possible to stay on the website showing the graph, click on a statement and a floating container is created. Meanwhile an asynchronous connection to the server is opened in the background to get the content of the statement window which is then shown in the container. Since the whole page is retrieved and the client is able to continue working while the page is downloaded, the connection is defined as asynchronous. Another application of Ajax is getting specific information through a synchronous connection. For example, if some JavaScript code needs to get metadata about the debate for further processing, it should send a query to the database just in case the information has changed since the information was last retrieved. In this case, the Javascript requests from the server the output of a PHP page which in turn queries the database. The specific information is then printed in JSON notation (Crockford, 2008) such that the original JavaScript code can continue processing. Since the processing was blocked during the download of the information the connection is defined as synchronous. Retrieving data in the JSON format is more elegant than dynamically running JavaScript code to set variables.

3.3 Working with the opinion landscape

The previous section introduced the concepts of statements (main hypothesis, lead statements and conditions) and arguments and the relation between them. This knowledge is important when interpreting the results of the multivariate analysis. The rest of this chapter will discuss the rating system, how it is supported and influenced by an automatic to-do-list and the transformation into the opinion landscape which is the input of the multivariate analysis.

3.3.1 Supporting statements to show opinion

Support is always shown by rating the statement positively and rating all statements on an argument positively shows support for that argument. In figure 3.3 the participant has shown support for the whole argument which is also indicated by the green tick mark on the upper right corner of the argument. But he doesn't have an opinion on the main hypothesis yet. As the participant supports the argument, it would make sense to oppose the main hypothesis by negatively rating it. In this case a to-do-list would prompt the participant to oppose the main hypothesis (see figure 3.1). Details are shown upon clicking on the recommendation (see figure 3.5)



The statement

We shall use energy saving light bulbs ☐

is challenged by the following CONTRA argument which you support:

energy saving light bulbs have mercury inside, thus they are dangerous for humans and the environment 🟩

Mercury can leak into the environment 🟩

If you want to argue consistently, please choose one of the following options:

Oppose the statement

Oppose the lead statement or any of the conditions of the CONTRA-argument

Introduce a condition to constrain the validity of the argument

Figure 3.5: A todo-list recommendation

If one thinks of the opinion landscape as it will be described in the next section, another effect of the to-do-list is to help fill in the gaps in the matrix where no rating was given. The more data in the matrix (the more the system knows about the participant's opinions), the better the report will be.

3.3.2 Data structure

This section describes the data structure of the debate and the ratings in the database. It then presents the data structure as an input of the multivariate analysis.

Although statements and arguments look differently in the graph, they are treated equally in the database. In fact, statements and arguments are nodes in the debate tree and are stored in the same table as rows. To maintain the connections from statements to arguments and arguments to their parent statements one of the attributes of the node is its parent node. Ratings are stored in another table. Each rating has attributes storing information about who rated what, how and the certainty of the opinion (the certainty is the slider in figure 3.4).

Additionally to the MySQL database there is also an XML format for exporting the debates. The process starts by downloading the list of nodes and ratings. The nodes do not keep their node ID. Instead they get IDs from zero to the amount of nodes. The list of participants is also downloaded but here the original user IDs are kept. When importing the debate, all the nodes have new IDs based on how many nodes are already in the database. This technique also makes it possible to clone debates for testing purposes. They would be identical to the participants.

To perform the multivariate analysis on the data, it is retrieved and stored in a PHP array. From this step onwards it will be referred to as the opinion landscape. As described in chapter 2 one can imagine this array as being a matrix with Participants as rows and Statements as columns. Depending on the method used, this array is either directly processed or sent to an external service for processing (see chapter 4.2). Either ways, the opinion landscape is the input for the dimensionality reduction as well as the input for the cluster analysis. The outputs are a two-dimensional plane and a mapping of which participant belongs to which cluster. Most of the algorithms discussed in this thesis are taken from the literature.

Among other features, the report generator of the TBDis system also identifies potential fractions in a debate. It has a component for clustering participants given the opinion landscape. As the goal of the thesis is to select the most suitable algorithm for both dimensionality reduction and for clustering, the current algorithm will also be presented.

3.3.3 The TBDis fraction analysis

In this chapter current clustering algorithm (Hilty, 2011b) for TBDis is described. This algorithm is already implemented and is run whenever a participant is using the report generator. It has been created as an ad-hoc solution for the TBDis prototype and will be replaced by the solution that is systematically developed as part of this thesis if it turns out to be better.

Method

Since this algorithm for identifying fractions is basically a cluster analysis it is built similarly to the ones from the literature. There is an iterative process and a stopping condition such that not all participants get clustered into just one group. In the context of this cluster analysis, fractions are defined as being a group of at least two people. Participants with unique opinions are not clustered into a fraction (Hilty, 2011b).

The process begins by trying to identify some fractions. If less than half of all participants are in fractions, the granularity of the analysis has to be adjusted. Otherwise the cluster analysis is considered over. By removing some statements from the scope of the analysis it is more likely that fractions are found. Removing statements is based on how important they are. Additionally it is considered better when the higher the amounts of statements are considered when looking for fractions. When statements have been removed the process restarts.

Algorithm

Due to the fact that the algorithm was developed specifically for the TBDis system it has some special requirements. Unlike the multivariate methods from the literature, this algorithm requires the columns of the the opinion matrix to be sorted in level-order according to the level of the statement node in the debate tree. This fact is classified as a requirement because it can only be done with the help of the structural information of the tree. If an unsorted opinion matrix would be provided as input, the algorithm would have no way of sorting the nodes in level-order. Also the algorithm works with only three types of ratings: positive, negative or neutral, rather than with a range from -3 to +3. Additionally the level of every node is a criteria used in the algorithm. This information is also given to the fractions function.

The matrix is implemented as an array of arrays. Each index represents a participant's opinion. The opinion is stored as an array. In this opinion array, each index represents a statement and the values are ratings given by the participant according to the statement.

First the data is prepared. Step one involves removing all the non-voters (people with zero values as their opinion array) from the matrix. This is done because there is no information content there. The values are prepared as well by transforming the positive ratings into +1, negative ratings into -1 and missing ratings into a zero.

The iterative part begins directly by trying to identify fractions. Basically the matrix rows are checked for duplicates. Participants that have the same opinion array are considered as a fraction. A fraction must consist of at least 2 participants.

If at least half of the participants are in any fraction then the clustering process is halted. This is known as the stopping condition. But since there are potentially many statements in a debate, it is likely that all participants have unique opinions. Therefore the granularity of the analysis has to be adapted. This is done by removing less important statements from the matrix and comparing the participants' opinion again. For example: if all statements but the main hypothesis would be removed before doing the analysis again, the output would be two fractions with all the positively-rating-participants in one and the negatively-rating-participants in the other fraction. This output is valuable but it is also already shown to the participants when they view the statement window of the main hypothesis. The goal here is to have clusters based not only on the main hypothesis but on all the statements; or at least based on the most important ones.

Before removing statements, the algorithm has to choose which statements are to be removed. Choosing which statements to remove is what makes this algorithm unique compared to the other clustering algorithm because it uses the level of the node in the tree. This information comes from the tree and thus cannot be retrieved from the opinion landscape. Two key requirements: (1) Remove the least amount of statements needed such that a new fraction among the participants with unique opinion can be built and (2) prioritize removing statements that are the farthest from the main hypothesis, assuming that they are less important for the debate.

To know which statements should be removed, the algorithm calculates the minimal amount of statements to be removed such that a new fraction can be created. For this, the participants' with unique opinion will have their opinion array compared to each other.

For each comparison a so called "difference array" is calculated. This array contains all the statements to which two participants have had different opinions on, thus the nomenclature. For example if the participants' opinion arrays differ by two indices (two statements whose participants' opinions are different) then the difference array will contain two statements. The outputs of this part are many difference arrays with different amount of statements in them. Removing the statements in one of these difference arrays would have the effect that both participants will be in the same fraction. Following the requirement of removing the minimal amount of statements possible, the algorithm gathers all difference arrays with the minimal amount of statements in them. If there is just one, the process is trivial: all the statements in that difference array are removed from further analysis.

If there is more than one minimal difference array the algorithm calculates which of them has the least important statements and removes them from further analysis (second requirement). This is done by looking into all difference arrays and for each array adding up the level of the statements from a level-sort. For example, we have two difference arrays: The first has a statement from the third level and one from the fifth. The second array has a statement from the third level and one from the seventh. The second one has a higher sum of levels (10 > 8), thus these are the statements that will be removed. If the sum is equal the decision is random.

By removing statements in the difference array the algorithm ensures that by now there has been at least one statement removed from the scope of the analysis. Also this means that when doing the fraction analysis again, at least one more fraction should be identified. The granularity is adapted again as long as less than half of the participants are in fractions.

Additionally to the clustering of participants in the heterogeneous groups there has also been a function to determine whether a participant is "compatible" with a certain fraction. This was only used for participants who after the clustering still had unique opinions to see where they would fit best. The method used was pairwise multiplication of the statement values from a unique opinion array and the residue of a fraction array which has some statements excluded. If all products were non-negative, the unique opinion was compatible with the fraction. The interpretation is that a non-negative product means that both factors are either positive or negative or zero. This implies that both opinion arrays point in the same direction.

The result is a list of fractions with the names associated with the user IDs. Also the statements that all participants of this fraction agree, disagree and are neutral upon are listed. The algorithm works fairly well and the results in the report are plausible. But it is not perfect: (1) when removing statements such that a new fraction can be built; only participants with unique opinions are compared to each other. This means that the next fraction that is built will constitute of at least two participants that have had unique opinions. What is not considered is that in theory it is also possible that a participant with unique opinion could be assigned to an existing fraction (rather than a new one) by removing fewer statements. (2) The transformation of ratings into +1 and -1 does not have any actual effect since there is no calculation with the ratings themselves. Only the differences in ratings are accounted for. This means that the certainties of the opinions are not accounted for in this method of clustering. (3) Also the method of adapting the granularity needs external information from the tree. This makes the algorithm not generalizable unless some other method for calculating a weight is used.

4. Implementation and evaluation methods

In this chapter, the possibilities of implementing the algorithms, introduced in chapter 2, with prior known programming languages are explored. The goal is to apply the algorithms and test them with the same input data to ensure that there is a common basis. This is a prerequisite for comparing the outputs so that the methods to be used in conjunction with each other in the TBDis system can be selected. The data structure is described. Ultimately, the language R was chosen for the implementation due to its simplicity and straightforward integration into the current system.

Researching the implementation of algorithms in previous research was challenging because a large portion of the literature describes the algorithms in a human readable form rather than in code. Nevertheless, the provided references to software such as DECORANA (Hill, 1979) lead the way. It turns out that these tools would be ideal in a research setting where a great degree of the work is somewhat manual in contrast to the current setting of the TBDis system where input and output are required to be fed in and reutilised in an automatic way. In the case of TBDis, the input is in the MySQL database. It would be more difficult to determine how to interface the database with software tools for the calculations that is triggered by a webserver command and feeds the output back into the page.

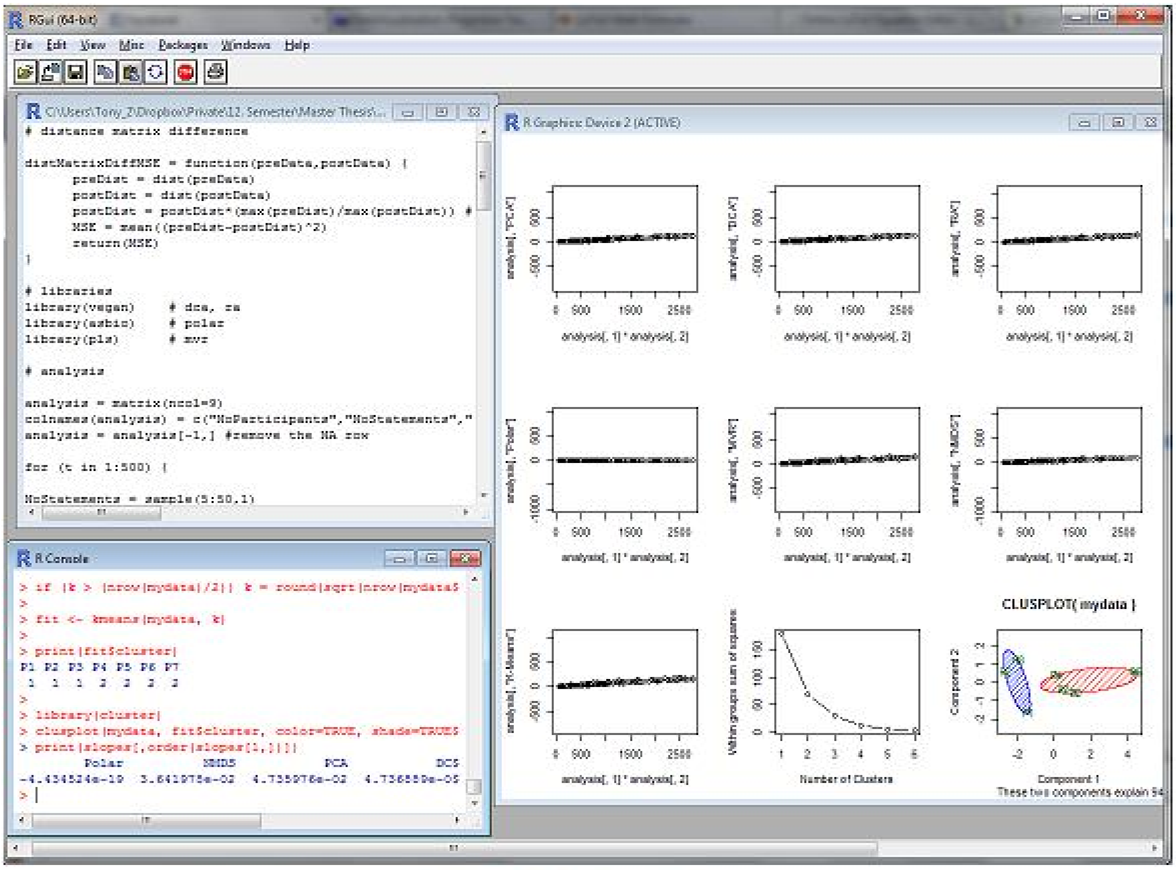
The TBDis system was developed in Javascript and PHP/SQL, which makes these the more obvious choices for programming languages for the algorithms. Given this premise, research was conducted on whether there are already PHP or Javascript (or any other related language) implementations of the algorithms in online communities. As explanations of algorithms in the literature consist mostly of mathematical equations, it would have been necessary to install a package for basic mathematical operations, for example from Pear (The PHP Group, 2012). However, during the research, some references to the R language were identified (R Foundation, 2012). This language is specific for statistical analysis and offers some functions for performing multivariate analysis on data. As a result, an interface for the R language was chosen over a PHP implementation and there is also an easy way to interface it with the current system.

4.1 The R language

Originally written by Robert Gentleman and Ross Ihaka, the R language and program are part of free software for statistical computing. Currently, it is maintained by the R Foundation with several contributors. It is similar to the S statistical language, but it is open source.

As best described by the R Foundation (2012) itself, "*R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering* *etc.) and graphical techniques, and is highly extensible.”*. It is popular amongst not only statisticians but also other types of researchers and scientists (Ooms, 2011).

The R language features if-then-else conditions, loops, recursiveness in functions and a wide array of built-in functions for data analysis and graphical representations of data (R Foundation, 2012). The R software provided by the R Foundation is an interface for the R language with a console (see figure 4.1).


RGui (64-bit)
File Edit View Misc Packages Windows Help
C:\Users\Tony_Z\Dropbox\Private\12. Semester\Master Thesis\...
R Graphics: Device 2 (ACTIVE)
R Console
CLUSPLOT(mydata)
Component 1
Component 2
These two components explain 9
Number of Clusters
Within groups sum of squares
analysis[, 1] * analysis[, 2]

```
# distance matrix difference

distMatrixDiffMSE = function(preData,postData) {
      preDist = dist(preData)
      postDist = dist(postData)
      postDist = postDist*(max(preDist)/max(postDist)) #
      MSE = mean((preDist-postDist)^2)
      return(MSE)
}

# libraries
library(vegan)     # dca, ca
library(ssbio)     # polar
library(pls)       # mvr

# analysis

analysis = matrix(ncol=9)
colnames(analysis) = c("NoParticipants","NoStatements","
analysis = analysis[-1,] #remove the NA row

for (t in 1:500) {

NoStatements = sample(5:50,1)
```

```
> if (k > (nrow(mydata)/2)) k = round(sqrt(nrow(mydata$
>
> fit <- kmeans(mydata, k)
>
> print(fit$cluster)
P1 P2 P3 P4 P5 P6 P7
 1  1  1  2  2  2  2
>
> library(cluster)
> clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE$
> print(slopes[,order(slopes[1,])])
        Polar          NMDS            PCA            DCS
-4.434524e-19  3.641975e-02  4.735976e-02  4.736550e-05
> |
```

Figure 4.1: The R Gui

There are alternatives to the R software, notably RStudio which is an Eclipse style, fully-fledged integrated developing environment that adds some R specific functionality (RStudio, Inc., 2012). Some other interfaces for the web such as R-Php by Pontillo and Mineo (2005) also exist and were considered.

4.2 Interface with PHP

R is a powerful tool that has been used here to reach the goals of this thesis. Since the context of the thesis is the TBDis system, an interface is necessary that can be used in the TBDis system to run R code and present the output to the participants.

Two possible methods are firstly, installing R in the server where the TBDis resides and connecting to the system, or secondly, using interfacing with a server or service that can receive R code and send back output. A great disadvantage of installing R in the server is that the server provided to the TBDis system is a webserver. This means that there are only web-related services provided to the system. Additionally to the Apache webserver, the server computer needs to have R and other packages installed such as a package to generate the output images (Pontillo & Mineo, 2005). Jockers (2008) describes a way to use PHP to pre-process the data, place it in a text file and execute an R command to run the R code. This command creates a JPG file for the output scatter plot.

The web interface provided by Pontillo & Mineo (2005) has been deemed as unsuitable here, since their implementation of R-Php does not provide a way to output scatter plots or to use external R packages. Another provider of a web interface is openCPU (Ooms, 2011). Although the name indicates a general purpose service, it is essentially a remote computing service for requests in the R language. It is "*a framework for scalable, embedded scientific computing, based on R and Latex*" (Ooms, 2011). It provides free cloud service for statistical computing and sharing code with the aim to embed R functionality in web applications without requiring installation of R (Ooms, 2010). It is a RESTful application. As defined by Fielding and Taylor (2002-05), Representational State Transfer (REST) is a software architecture that allows the browser (client) to send a request to a RESTful service (server).

4.3 Implementing in R

The language R and the R program offer functions for the methods described in chapter 2. This means that it is not necessary to implement any of the methods by hand. Several of the methods require some type of pre-processing which will be described in this section. Since the R program has a console, all of the code snippets in this section can be run by copying and pasting in the console.

The Participants X Statements matrix is the input dataset for most methods. The data structure for this matrix must be in the format of a "data frame". A data frame is a container in R that stores the matrix and allows for assigning row and column names to organise the data better. A data frame can be created by importing data as follows:

```
mydata <- read.delim("c:\\some path\\folder\\7.csv", sep=";", header = TRUE,
row.names=1)
```

Where the values in the loaded file are separated by a semicolon (as indicated by the "sep" parameter). The "header" and "row.names" parameters indicate that the file has a header and the row names are in the first column. This method is used to load pre-constructed datasets into the R program.

If a matrix is to be constructed, it is also possible to prepare the matrix and create a data frame:

```
mydata = c(3,2,-1,0,-3,0,3,2,-3,-3,-1,-3,2,0,-3,-3,-1,-3,0,0,-1,-1,0,0,-3,-
3,3,3,3,3,-2,0,1,1,-1,0,0,1,1,0,0,0)
dim(mydata) = c(6,7)
mydata = t(mydata) #transpose
rownames(mydata) <- c("P1","P2","P3","P4","P5","P6","P7")
colnames(mydata) = c("S1","S2","S3","S4","S5","S6")
mydata =
data.frame(S1=mydata[,1],S2=mydata[,2],S3=mydata[,3],S4=mydata[,4],S5=mydata[,5]
,S6=mydata[,6])
```

This method is used when creating random datasets, which will be the case for the evaluation later in this section. It is also used for the sample codes in order to perform the different types of analysis.

As a follow-on from chapter 2 where the methods were introduced and a sample input and output was presented, this section focussed on using the R program to apply these methods. Before performing the analysis, the packages necessary for the methods must be installed. To do this in the R program, “Packages” can be selected in the menu, followed by “Install package(s)…”. The results of the analysis methods are often objects and the information is stored in variables. To identify which variables from an object are available, the `attributes(myObject)` function can be used. To access a variable from an object the $-sign is used. For example, the scores of a polar ordination are accessible by inputting `polarOrdOutputObject$Scores` in the console. It is also important to point out that the variable names are case-sensitive. If one is unsure about a function, inputting `?functionName` (e.g. `?mvr`) in the console opens the help file; provided that the package for the function has been loaded.

The analysis methods themselves are built-in functions of the aforementioned packages. The weighted averages method by itself does not provide a second axis, thus it was omitted Most of these functions are introduced in Everitt and Hothorn’s book (2009):

- `polar.ord()` is the function for polar ordination. To work, the function requires the installation of the “asbio” package. The parameters and their values used in the context of the thesis are the “index” (the distance measure; set to “euclidean”) and the “endpoint” (determining poles; set to “var.reg”). The meanings of these values are discussed in chapter 2. Additionally, the input dataset has to be adapted for this method. As it does not accept negative values, the number three is added to all the values so that the range is now [0,6] rather than [-3,3]. This change does not affect the values in the distance matrix. The resulting scores are stored in the `$Scores` (uppercase “S”) variable of the output object.
- `prcomp()` is for principal component analysis (PCA). No special parameters or packages are needed for this function. The resulting scores are stored in the `$x` variable of the output object.

- `decorana()` is the function for two methods: reciprocal averaging (RA) and detrended correspondence analysis (DCA). As described in chapter 2, DCA is based in RA thus it comes as no surprise that the same package (called “vegan”) and function is used for both. The function requires all values in the input dataset to be positive, non-zero values. Thus the number four is added to all the values, which means that the new range is [1,7]. The distinction between RA and DCA is given by the “ira” parameter. A RA analysis and a DCA are performed by setting the parameter to one or zero respectively. The resulting scores are stored in the `$rproj` variable of the output object.
- `mvr()` is the function for multivariate regression (MVR). The function is part of the “pls” package. The input parameter is different from the previous functions. Here, a formula for a linear model is desired and not the dataset. As discussed in chapter 2 the selected formula shows that the first statement (the main hypothesis) is explained by a linear combination of all other statements. Rather than having the `mydata` variable as the input, the formula is given: `S1~S2+S3+…+SN`. The S1 to SN represent the N-th column in the `mydata` variable. Also provided is an additional function for dynamically adapting the formula to the amount of statements. The resulting scores are stored in the `$scores` (lowercase “S”) variable of the output object.
- `monoMDS()` is the function for non-metric multidimensional scaling (NMDS). Unlike the other methods, this function requires the distance matrix as the input. This allows the researcher to choose the distance measure prior to applying the method on the dataset. The distance matrix is calculated by the `dist(mydata)` function. The “model” parameter is set to “global” in order for the regular non-metric multidimensional scaling algorithm to be used. The resulting coordinates are stored in the `$points` variable of the output object.
- `kmeans()` is the function for k-means clustering, which is the method selected for clustering dimensions. This method was prioritised based on that the number of resulting clusters is a-priori set by the researcher and it was provided to the function as a parameter. Since plotting coordinates on a two-dimensional scatter plot requires the coordinates to be two-dimensional, the number two can be set as the parameter for a k-means clustering of dimensions. This results in the reduction of dimensionality from N to two. This is more convenient than trying to ensure that other clustering methods output exactly two clusters with no outliers. Additionally, the matrix has to be transposed to ensure that the columns (i.e. statements, dimensions) rather than the rows (i.e. participants) are clustered. The resulting coordinates are stored in the `$centers` variable of the output object because the coordinates of the centre of the clusters are used to represent the clustered dimensions.

These multivariate methods for dimensionality reduction were selected to meet the one of the main goal of the thesis: visualise the opinion landscape on a scatter plot. The output objects contain the information about the coordinates. In the case of NMDS and k-means, the output is a matrix with two columns and as many rows as there are participants. The values are used as the coordinates of the participants. In the result object of polar ordination, PCA, RA, DCA and MVR, the variable for the ordination scores is usually a matrix with more than two columns. This is because the built-in methods in R calculate all ordination axes. However, because the first two explain the most variance (see chapter 2), they are used as the coordinates for the participants. The first two columns are accessed and stored vectors using the following pattern:

```
x <- polarOrdOutputObject$Scores[,1]
y <- polarOrdOutputObject$Scores[,2]
```

To visualise the coordinates, the following functions are used:

```
plot(x,y, type="n", asp=1)
text(x,y, labels=rownames(mydata))
```

The `plot()` function displays a scatter plot with hidden points (due to `the` **`type="n"`** parameter) and an aspect ratio between the X and Y axes of one to one. The `text()` function inserts the row names of the dataset in the corresponding X and Y coordinates. This has the effect that the names of the participants rather than unnamed points are displayed in the scatter plot. Essentially, the scatter plot produced is the opinion landscape as presented in each section on methods found in chapter 2.

As described in chapter 2, NMDS has a random initial condition and optimises the coordinates based on the distance matrix. As the method has more than one outcome given the same input, it is non-deterministic. This is important to note because the method can result in different local optima arrangements of the participants on the scatter plot. It is thus plausible that if the report of the participants' debate is opened multiple times, the opinion landscape may appear different which can confuse the participants. This is a clear disadvantage of using NMDS.

The built-in R functions for cluster analysis methods used in this thesis are described as follows:

- `pvclust()` is the function for hierarchical clustering. To work, the function requires the installation of the "pvclust" library. The parameter for the distance measure is "method.dist", and "euclidean" is the chosen value. The parameter for selecting the variation of the hierarchical clustering algorithm is "hclust.method", and the value "complete" was chosen. When the output object of the cluster analysis is plotted, the dendogram is shown with different p-values for each "merging" (see chapter 2). Unlike the other two methods, the output does not show a definite clustering of the participants. However, it does provide an indication of which participants will be clustered together under different levels of detail. Thus the level of detail (more generally, the stopping condition) has to be selected and applied to the output of the hierarchical clustering. As discussed in chapter 2, one of the criteria for the level of detail is using p-values. It describes how confident the algorithm is that a cluster is supported by data. This is the reason for chosing `pvclust()` rather than the regular `hclust()` function for hierarchical clustering. Pvclust is a wrapper around hclust that provides the desired p-values. They are needed for deciding which "merges" in the dendogram translate to clusters where the output is a clustering of participants. The threshold for the p-value is set to 95% confidence. In short, the output of a hierarchical clustering has to be post-processed with the help of p-values to ensure the clustering of participants.

- `kmeans()` is the function for k-means clustering. Besides the data matrix, a second parameter is the amount of clusters that k-means should produce. The required "K" value is calculated in a pre-processing stage that based on the within-group sum of squares as described in chapter 2. The resulting clustering configuration is stored in the `$cluster` variable of the output object.
- `dbscan()` is the function for density based clustering. The library "fpc" is required by the function. The density is *a-priori* defined by the researcher through setting the "eps" and "minPts" parameters. As described in chapter 2, a participant is part of the core of a cluster if it has "minPts" other participants within "eps" distance. Since debates can include a fairly low number of participants, the value "one" is used for "minPts". The mean of all the distances from the distance matrix is a suitable value for "eps" as discussed in chapter 2.

All three functions have some type of pre- or post-processing which add a subjective component to them. This is also valid for some functions in dimensionality reduction which require other input parameters apart from the data matrix.

The dendogram shown in chapter 2 is one type of visualisation for hierarchical clustering. As participants are not likely to instantly understand a dendogram, a common visualisation function for all three methods for cluster analysis is used (as seen in chapter 2):

```
clusplot(mydata, kmeansOutput$cluster, color=TRUE, shade=TRUE, labels=3,
lines=0, asp=1)
```

This function presents the participants in a scatter plot with circles to indicate clusters. This requires the installation of the "cluster" library. Since it is very likely that the data provided for the function is multidimensional, the function first performs a PCA to reduce the dimensionality and then the clusters are shown. It is possible that clusters overlap. Of course, the clusters do not overlap in the multidimensional space, but they might overlap after the PCA.

The `plot()` and `clusplot()` functions for multivariate methods are key for an evaluation because they visually interpreting the results. This and another type of evaluation are described in the next section.

4.4 Evaluation methods

When evaluating the multivariate methods the two goals have to be considered. To recapitulate, the first goal is to reduce the dimensionality of the opinion landscape so that the opinion landscapes can be shown to the participants in the report. The second goal is to cluster participants into fractions that can then be listed in the report. Both goals serve to reduce the complexity for the participants. This section is thus concerned with evaluating methods for dimensionality reduction as well as methods for clustering data.

To evaluate the multivariate methods, two different approaches have been adopted. Firstly, the more objective approach is to apply the multivariate methods to a high amount of random datasets and calculating statistics. Thereafter, the result is compared to the results of other multivariate methods. The second approach is more subjective, which includes applying the multivariate methods to selected constructed data and interpreting the results. Gauch (1982, p166) points out that methods should be evaluated with simulated data that has known structures. The outcome can then be compared against expected results.

The interpretative evaluation is arguably more important than the quantitative evaluation because no matter how much better in theory the method is, the method should be scrapped if the results do not reflect the underlying structure of the data that the participants are interested in. The quantitative evaluation serves to increase confidence that the method is consistently better performing than another method. A balanced evaluation is achieved by using many random datasets and some carefully constructed ones (Gauch, 1982).

As there are two categories of multivariate methods (clustering and dimensionality reduction) to evaluate, two types of data (random and constructed) and two types of analysis methods (interpretation and quantitative), eight possible evaluations exist. Some of the evaluations are more appropriate than others. As pointed out before, interpreting clustering or dimensionality reduction using random data is not suitable since random data has no patterns. Generally, subjective interpretation of random data is inadvisable.

More interesting is the interpretation of clustering or dimensionality reduction using constructed data. Since the constructed datasets in this thesis have underlying structures, being able to show these structures to the participants is a criterion for both types of multivariate methods. As pointed out before it is the strongest criterion in the thesis. It can be divided into two more criteria: the method being able to show the underlying structures and the output being unambiguous and not confusing for the participants.

The quantitative evaluation of multivariate methods using random data is also interesting. Calculating statistics to reveal the performance of different algorithms on an average of several datasets gives an indication of how robust the algorithms are.

Hyrenbach (2011) mentions some general barriers that may hinder a good evaluation. Firstly, an evaluation is not valuable if there is measurement or sampling error. As the data is taken directly from the TBDis system, this does not apply to the data studied here. Another, more relevant challenge is termed methodological artefacts. That is, even if the methods of evaluation are followed perfectly, there is still some subjectivity on the selection of the evaluation methods.

An evaluation of a method means that the method is assessed against other methods in a way that makes them comparable. There are different possible scales or criteria used to evaluate the methods. For dimensionality reduction, a criterion that can be used according to Moore et al. (1970) and Hyrenbach (2011) is whether or not the underlying structure is accurately represented after performing the method. This can be evaluated by constructing datasets with underlying patterns and determining if they are accurately represented. A more quantitative criterion presented by Hyrenbach is the proportion of total variance explained. In other words, how much of the variance in the multidimensional space is still visible in the two-dimensional scatter plot. Also, scalability can be a criterion. A general criterion for both types of multivariate methods is robustness or consistency in delivering useful results given several different datasets (Gauch, 1982).

A criterion for immediate exclusion of methods from this evaluation is the underlying assumptions made about their input dataset. As described in chapter 2, several methods for dimensionality reduction work better with two different types of data: monotonic and unimodal. Naively applying a method with different assumptions to data can lead to poor results. One can argue that this is the case for the methods applied in this thesis; however, it is currently not possible to exclude a method based solely on this criterion as it is not known if the opinion landscape matrix of a debate is monotonic or unimodal or anything else. This is because there have are an insufficient number of tests to determine if there are monotonic or unimodal patterns. Furthermore, even scientists seem to use the methods without sufficing these assumptions. For example, Gauch (1982) explains that community ecologists use these methods with conflicting assumptions about the data. The resulting arch and horseshoe effects are dealt with in a not too elegant matter. As described in chapter 2, the arch effect from RA is removed by detrending the output data.

4.4.1 Quantitative analysis

The aim of quantitative evaluation of multivariate methods is to obtain some type of calculated score that indicates how well the method performed. By having a score for each method that is applied to the same input dataset, the methods become comparable on a scale.

Methods for dimensionality reduction

Most multivariate methods for dimensionality reduction provide their own scores for evaluations. The output object of a polar ordination and a PCA contains a variable with values about the variance explained by each ordination axis. MVR provides values about the residuals of the projections. K-means provides values for the sum of squares of distance between the points and the centre of their cluster, and NMDS provides the stress value that was reached in the local optimum. These values can be used to compare the datasets with the same method. The problem is that these values are not comparable to each other across methods. For this reason, an evaluation method was developed that does not depend on any of these values.

Since a good method is a method that keeps most of the structure intact after the dimensionality reduction, one way to evaluate the method is to compare the structure of the dataset before and after the method is applied.

The double cluster analysis evaluation method is proposed in chapter 2. The method is used to try to find out the extent of the difference between the output of a cluster analysis of the original data and the data after the dimensionality reduction. To use this method, both datasets must be clustered and the errors are counted. Errors occur when participants are in different clusters in both datasets. As described by Manning, et al. (2008), the purity of the clusters in the lower dimensional space is computed. The rand index (Hubert & Arabie, 1985) gives an indication of how similar two outcomes of clustering algorithm are and is used as the score for the quality of the dimensionality reduction.

The distance matrix comparison evaluation method is also suitable. The underlying assumption is that the structure of the dataset is represented in the distance matrix. If it is changed, the structure will be changed as well. In essence, this method is used to calculate the distance matrix before and after the dimensionality reduction. Both matrices are compared and the mean square error is calculated (MSE). The formula for MSE is as follows:

$$\mathrm{MSE} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} (d'_{ij} - d_{ij})^2}{n^2}$$

The “n” is the amount of dimensions of the distance matrices. Both distance matrices are symmetrical and have N number of participants as columns and N participants as rows. Each distance value is subtracted and the result is squared. Thereafter, the sum of all squares is divided by the amount of values and thereby, the mean of the sum is calculated. This is comparable to the stress function in NMDS.

As R is a programming language for statistical computing, applying this method is straightforward:

```
MSE = mean((preDist-postDist)^2)
```

Before the MSE is calculated, the data is normalised by equating the maximal values of both matrices and multiplying the remaining values by the proportional factor. The MSE value is used as the score for the dimensionality reduction.

Figure 4.2 illustrates the difference between two outputs with low and high MSE values. The dataset that is used is a constructed dataset where P11's opinion is inconsistent. The structure is given by having P11 agree to all statements, whether they are part of a pro or a contra argument. Figure 4.2 (a) shows the output of a PCA where P11 is an outlier. Figure 4.2 (b) shows the same output as (a) with the difference that the coordinates for P11 have been manipulated so that P11 is closer to the data cloud. Thus it is expected that this structure will reflect the underlying structure of the multidimensional space more poorly than in (a). In fact, the MSE values are different. The value for (a) is 1.90 while it is 9.19 for (b). The value rose dramatically because all the distance involving P11 had changed. A comparison of the distance matrix in (b) with the original one (a) results in a high mean square error.

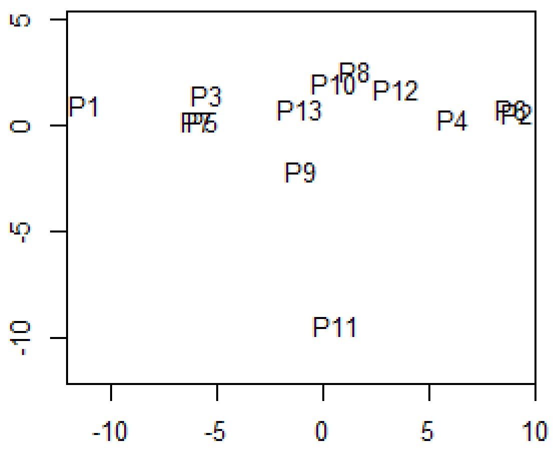


Figure 4.2 (a): Output of a PCA

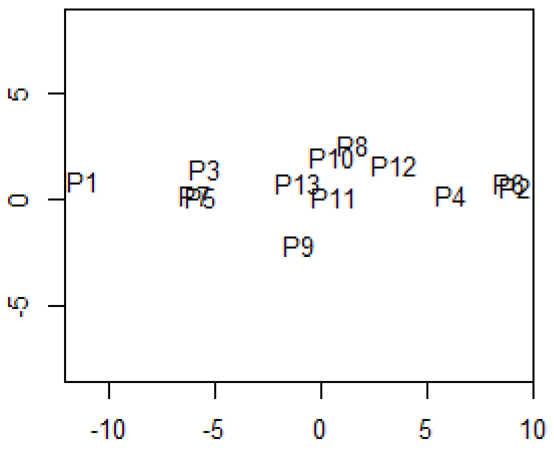


Figure 4.2 (b): Modified output

Although the evaluation method can be applied automatically, it is not fully objective. This is because the decision to choose the method of comparing distance matrices and using the MSE as a score or criterion is still subjective.

Calculating the MSE to quantify the performance of several methods that are applied to the same dataset allows the different methods to be compared. It is important to note that this comparison is only based on one dataset and the rank order might not be the same for other datasets are used.

Methods for cluster analysis

Typically, the goal of a cluster analysis is to attain high within-cluster similarity and a low between-cluster similarity (Manning, et al., 2008). In other words, the criterion is to have clusters which are in themselves homogeneous, but heterogeneous when compared with other clusters. However, it is only valuable to compare these indicators when the clustering methods output the same amount of clusters.

A reference to start a quantitative evaluation is the R function `cluster.stats()`. It provides statistics about the output of a cluster analysis such as the within-group sum of squares. If a second, different output is provided, the rand index is also calculated. The rand index introduced by Hubert and Arabie (1985) is a method that is used to describe the extent of the similarity between two outputs of cluster analyses. It can be applied to compare two outputs but it cannot be used to put them in rank order. To make use of this evaluation method, there should be a reference of what a good output actually is.

Some criteria for the quantitative evaluation of clustering methods as described by Steinhausen and Langer (1977) are maximal homogeneity of the clusters and minimum methodological artefacts. Nonetheless, in density based clustering, the homogeneity of the clusters is not an outcome of the analysis but rather the parameters that are required for the analysis. The two parameters, reachability and minimum number of points, define how homogeneous the clusters are. As seen in chapter 2, density based clustering is able to produce a cluster that can stretch around another cluster. In the context of TBDis, this has no meaning since the opinions from the two sides of the cluster could be totally different. An alternative criterion presented by Steinhausen and Langer (1977) is the optimal amount of clusters. The k-means clustering method requires an a-priori definition of how many clusters should be created. In these two examples, the quality of the method using this criterion is defined by input parameters which are selected by the researcher. However, Manning, et al. (2008) also point out that good scores on internal criterion (i.e., quantitative analysis) do not necessarily translate into effectiveness in the practical use of methods. An external criterion decided by the researcher based on the context of the analysis is thus proposed by the authors.

Steinhausen and Langer (1977) argue that the selection of an exclusively mathematical way to evaluate methods for cluster analysis is not possible because of the approximative characteristic of the results. The context (given by structured data) plays a large role. In addition, a visual evaluation, as proposed in the next chapter, should be conducted. The authors explain that in order to visualise clusters in multidimensional data, a projection onto the first two principal components should be done. Essentially, Steinhausen and Langer recommend performing an interpretative analysis on constructed data.

4.4.2 Interpretative analysis

An analysis with quantitative methods provides a rather objective and direct way to compare methods. However, as mentioned earlier, if the visualisation is ambiguous or confusing for the participants, the method should not be used. This strong criterion can be assessed by an interpretative analysis. An alternative term is “face validity”, which means that the output is good enough if the visual output is plausible and looks like what is expected. In the context of the thesis, this approach consists of looking into the underlying ratings matrix (the opinion landscape), identifying structures, and evaluating whether these structures are visible in the output.

It is also important to point out that in this thesis the interpretative analysis is conducted by the researcher. A better approach is if the interpretative analysis takes place in an experimental setting with people as debate participants.

```
   S1 S2 S3 S4 S5 S6 S7 S8
P1  2  1  0  0 -1  2  0  0
P2 -2  0  1  2  1 -3 -3  2
P3  3 -1  3  2 -3 -2  3  2
P4 -2 -1 -2  3  3 -3 -2  2
P5  2  2  3 -3  3  3  2 -3
```

```
          P1        P2        P3        P4
P2  8.000000
P3  7.141428  9.110434
P4  9.055385  4.000000 10.630146
P5  7.000000 11.789826 11.045361 12.767145
```

Figure 4.3: A random opinion landscape with distance matrix

Figure 4.3 shows an example of an opinion landscape with random ratings and the corresponding distance matrix. Figure 4.4 shows two different outputs of methods for dimensionality reduction. It should be noted that the axes in these outputs have no meaning as discussed in chapter 2.

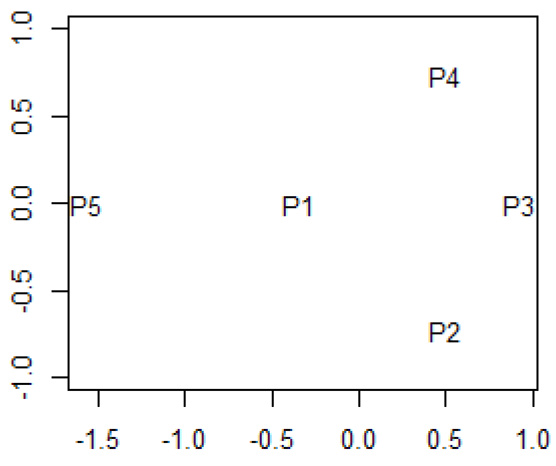


Figure 4.4 (a)

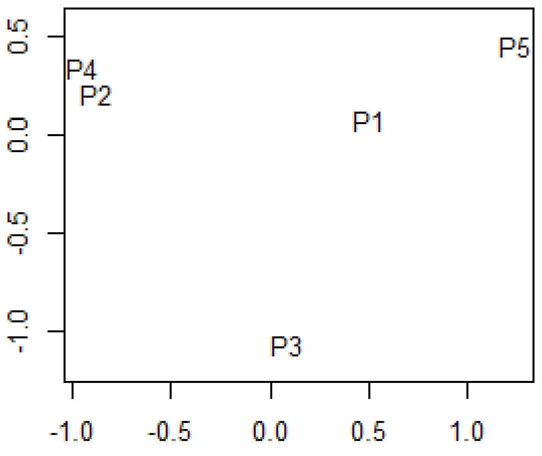


Figure 4.4 (b)

When performing an interpretative analysis, the opinion landscape should be observed to identify structures. According to the values in the distance matrix it seems that P4 and P2 have similar opinion vectors (similar ratings for each statement). This is indicated by a fairly low distance between both participants in the distance matrix. Bearing this information in mind, the scatter plots are examined next. However, on observation, the scatter plot in figure 4.4 (a) is not sufficient because P4 and P2 are too far away to build a cluster. Figure 4.4 (b), on the other hand, seems to be suitable.

This analysis method can only be used with deterministic multivariate methods. Interpreting an output of non-metric multidimensional scaling is somewhat meaningless since, given the same input dataset, the output may look different. In fact, both scatter plots in figure 4.4 are two possible outputs of NMDS although they are different.

Since cognitive effort is used in this analysis method it does not make much sense to interpret several random datasets. The step of identifying structures can be easily skipped by using constructed data where there are deliberate structures in the opinion landscape. Moreover, the plausibility of the output is bound to a context such that constructed data from this context should be used in this analysis. In this research, the context of the constructed data is the TBDis system, which is also the overall context of the evaluation.

4.4.3 Datasets for evaluation

Two types of data are discussed. This chapter is not intended to select one of the two types, but rather present the advantages and disadvantages of the data and discuss where each type is more suitable.

Constructed data

The advantage of a constructed opinion landscape is that underlying structures can be designed in the dataset. By using constructed data, it is possible to specifically evaluate whether a multivariate method is able to produce a scatter plot where the intended underlying structure is still visible. Moreover, the structures are from the context of TBDis. For example, the data is designed in such a way that participants supporting a statement would oppose a contra argument to this statement. This type of context is not guaranteed in random datasets. This shows that random data is fundamentally different from constructed data for the evaluation.

Constructed data is primarily meant for the interpretative analysis although the application of quantitative analysis on constructed data is not excluded. Interpreting the output of a multivariate method on constructed data can indicate whether or not the output is sufficient in the context of the TBDis system. It is also possible that some intended structures were wrongly implemented in the dataset. In this case, every multivariate method should fail to find these structures.

Ten datasets for opinion landscapes have been constructed in the context of this thesis. Generally, the datasets were constructed in so that there is no inconsistency in a participant's opinion. This is based on expected user behaviour because, as pointed out before, more often than not participants are consistently agreeing to pro or consistently agreeing to contra arguments to the main hypothesis. In addition, the higher the statement number is, the more zero values there are in its column. This is also based on the dynamics of the debate because a statement with a high number implies that it was expressed later in time. The shorter the time since the statement was created, the less people are likely to have seen the statement and rated it. It is expected that statements with higher numbers have a lower influence in the multivariate methods but this is not analysed. These datasets can be seen as criteria for the score given to each method in the interpretative evaluation.

The first dataset is a very basic one with seven participants and three statements. It is designed in so that P1 and P2 form a group in a cluster analysis while in a dimensionality reduction the cluster should be the farthest apart from P5. The second dataset is based on the first one. It adds three more statements and is designed so that there is a smaller distance between P2 and P3 than between these and P1. The second dataset is the one used for the illustrations in chapter 2. The third dataset is the output of one of the field tests of the TBDis system. The system was then used with 12 student participants.

The remaining seven datasets were designed with the following general patterns in mind:

1. The first participant (P1) has the most extreme pro-oriented point of view while the second participant (P2) has an directly contrasting, opposite opinion. The expected result for a dimensionality reduction is that P1 and P2 are very far away from each other in the scatter plot. In a cluster analysis, they should never be in the same cluster unless the optimal amount of clusters is one.
2. Participants with odd numbers (P3, P5, etc.) are more likely to rate statements related to pro arguments positively and contra arguments negatively. The opposite is valid for participants with even numbers (P4, P6, etc.). The expected result in a dimensionality reduction is that all odd numbered participants should be closer to P1 than to P2 in the gradient from P1 to P2, and the opposite is true for even numbered participants. In a cluster analysis, participants with odd numbers are more likely to be clustered with other participants with odd numbers, and the same is expected for even numbered participants.
3. The higher the participant number, the more zero values there will be in the participant's opinion vector. In other words, participants with higher numbers are the ones that have expressed their opinions less often. The expected result in a dimensionality reduction is that the higher the number of ratings a participant has given, the more likely they are to lie closer to the centre of the scatter plot.
4. The last participant in the dataset has not given an opinion. This participant's opinion vector has a value of zero for every statement. The expected result in a dimensionality reduction is that this participant is more likely to be the closest to the centre of the data cloud. It is plausible that this participant is clustered with participants with even or odd numbers.

In particular, the fourth and the ninth dataset are meant to include only these general patterns in order to evaluate whether or not the patterns emerge as expected. For the remaining datasets, some particular underlying structures were designed into them. The expected results from dimensionality reduction and cluster analysis are described below:

In the fifth dataset, P1 and P3 should be close to each other and they are likely to form a cluster. P2 and P5 should behave similarly but on the other side of the scatter plot.

The sixth dataset is characterised by the closeness of P2 and P6 to each other and P3 and P7 being less close.

The seventh dataset expands the sixth dataset as it retains the same structure while having more statements and participants. P11’s new role is to be inconsistent in his opinion by rating all statements positively. This dataset was used in the explanation of the mean square error.

The output of the eighth dataset should position P3, P5 and P7 fairly close to each other so that they are likely to be grouped in a cluster analysis. The same holds true for P4 and P6.

The tenth dataset is based on the ninth with a structure retains the positions of P1 to P7. P8 to P17 are new and should form a gradient from the left and right side of the graph towards the centre. Additionally, P7 and P17 should be fairly close to each other.

Randomised data

Creating random opinion landscapes allows for an analysis that averages the output of multivariate methods over a high number of cases. In contrast to that, by solely using constructed opinion landscapes, it is not certain that the outputs of the methods are robust. Random data can be generated in R in a simple manner. The following code generates a random dataset:

```
NoStatements = sample(5:50,1)
NoParticipants = sample(NoStatements:55,1)
Mydata = array(sample(-3:3,(NoParticipants*NoStatements),
replace=T),dim=c(NoParticipants,NoStatements))
rownames(mydata) <- paste(rep("P",NoParticipants),1:NoParticipants, sep="")
colnames(mydata) = paste(rep("S",NoStatements),1:NoStatements, sep="")
mydata = data.frame(mydata)
preData = mydata
```

Firstly, one random number for the amount of statements is generated. This number lies between five and fifty. The same is done for the number of participants. The matrix with the ratings is created by generating as many random numbers between [-3,3] as necessary to fill a matrix with the dimensions `NoParticipants` and `NoStatements`. The dimensionality of this matrix is adjusted accordingly. The row and column names are both a vector with strings from P1/S1 to N, where N is the amount of participants/statements. This information is later used for displaying the scatter plots.

It is important to note that the datasets include the numbers [-3,3] in a completely random configuration. This means that likely structures given by the structuring functionality of TBDis do not arise.

5. Evaluation and discussion

In the previous chapter, the implementation of the algorithms and evaluation methods was presented. In this chapter, the outputs of the methods, based on different criteria, will be discussed. At the end of each section, a rank order of the methods is provided.

5.1 Evaluation and discussion of methods for dimensionality reduction

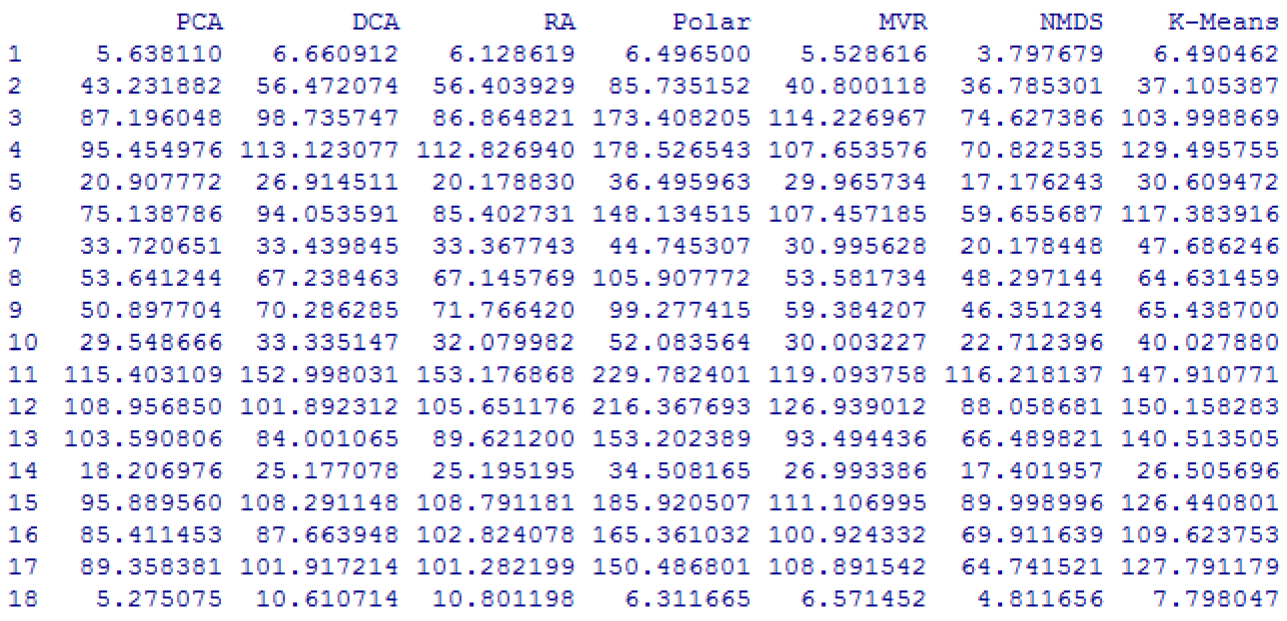
Firstly, the methods for dimensionality reduction are evaluated based on a quantitative analysis. The criteria of having a low mean square error (MSE) value and scalability are evaluated and the results are discussed.

The method of weighted averages is not part of the evaluation because this method does not render a second ordination axis. Without such an axis, the opinion landscape cannot be displayed in two dimensions. Nonetheless, weighted averages plays a key role in RA (see chapter 2).

5.1.1 Quantitative analysis of the mean square error

As described in the quantitative analysis, there are several methods to evaluate multivariate methods. However, an evaluation only is valid if the rank order of the methods is consistent over many datasets. The advantage of random data in contrast to constructed data is that it is possible to generate many datasets. The advantage of quantitative methods, on the other hand, is that it can be used to measure the quality of the method by comparing the states of the data both before and after the method has been applied, whereas the dataset itself is not of too much importance. Consequently, it makes sense to perform a quantitative analysis with random data.

A quantitative analysis of multivariate methods with several random datasets with a calculation of their means allows the methods sorted by their rank order to be presented. A quantitative analysis using the distance matrix comparison evaluation method for dimensionality reduction with random data results in the following figure:



```
           PCA        DCA         RA      Polar        MVR       NMDS    K-Means
1     5.638110   6.660912   6.128619   6.496500   5.528616   3.797679   6.490462
2    43.231882  56.472074  56.403929  85.735152  40.800118  36.785301  37.105387
3    87.196048  98.735747  86.864821 173.408205 114.226967  74.627386 103.998869
4    95.454976 113.123077 112.826940 178.526543 107.653576  70.822535 129.495755
5    20.907772  26.914511  20.178830  36.495963  29.965734  17.176243  30.609472
6    75.138786  94.053591  85.402731 148.134515 107.457185  59.655687 117.383916
7    33.720651  33.439845  33.367743  44.745307  30.995628  20.178448  47.686246
8    53.641244  67.238463  67.145769 105.907772  53.581734  48.297144  64.631459
9    50.897704  70.286285  71.766420  99.277415  59.384207  46.351234  65.438700
10   29.548666  33.335147  32.079982  52.083564  30.003227  22.712396  40.027880
11  115.403109 152.998031 153.176868 229.782401 119.093758 116.218137 147.910771
12  108.956850 101.892312 105.651176 216.367693 126.939012  88.058681 150.158283
13  103.590806  84.001065  89.621200 153.202389  93.494436  66.489821 140.513505
14   18.206976  25.177078  25.195195  34.508165  26.993386  17.401957  26.505696
15   95.889560 108.291148 108.791181 185.920507 111.106995  89.998996 126.440801
16   85.411453  87.663948 102.824078 165.361032 100.924332  69.911639 109.623753
17   89.358381 101.917214 101.282199 150.486801 108.891542  64.741521 127.791179
18    5.275075  10.610714  10.801198   6.311665   6.571452   4.811656   7.798047
```

Figure 5.1: A sample output of an analysis

Each row is a sample randomly generated dataset (18 samples out of 500 are shown). The columns represent the different multivariate methods for dimensionality reduction. The values are the corresponding MSE scores for a method on a particular sample. The mean MSE for each method for all samples are as follows:

NMDS	PCA	DCA	RA	MVR	k-means	Polar
49.89351	63.88359	66.5715	67.68388	69.81908	83.48715	115.0058

By observing this rank order, it is apparent that NMDS is the method with the lowest mean MSE score, followed by PCA, DCA, etc. This shows that for 500 random datasets with no underlying structure NMDS provided on average the least amount of errors. 500 samples were generated because there is then enough random debates from small to larger sizes.

This result is not definitive because it is not known if the rank order will stay the same when the analysis is performed over and over again with other 500 random samples. To be confident about the result, a t-test of significance should be conducted. This test reveals whether or not there are significant differences between the values. If the p-value that is produced by the `t.test()` function is below 0.05, one can be 95% confident that the values are significantly different. 0.05 is the commonly agreed threshold for a t-test. The value for the t-test between NMDS and PCA has a p-value of 5.8333E-11 which is far smaller than 0.05. The value for the t-test between PCA and DCA is 0.2602. This means that MSE value for NMDS is significantly different from PCA but PCA is not significantly different from DCA. Since NMDS has a lower mean MSE and is significantly different from PCA, NMDS should be used for the TBDis system.

The non-deterministic characteristic of non-metric multidimensional scaling leads to a second consideration about whether or not to include NMDS in the evaluation. The problem is that with a random initial condition the output may be a local optimum. Thus there is more than one possible solution. Apart from the fact that seeing different scatter plots may be confusing for the participants if the report is opened multiple times, it is also harder to evaluate how well NMDS performs in specific cases. Nonetheless, on average NMDS performed consistently better than any other method in a pre-evaluation with several datasets. Thus it was not excluded.

To counteract the non-deterministic characteristic of NMDS, the idea of combining it with another method for dimensionality reduction, proposed in chapter 2, is implemented in the evaluation. By providing the output of any of the remaining methods as the input for NMDS (rather than having a random initial condition), the method tries to optimise the positions of an initial condition that is by itself already a rather good solution. It is hoped that given a good initial condition, NMDS will reach a global optimum rather than a local one. As a result of using a non-random initial condition, the combinations the methods with NMDS are deterministic.

Since the methods have been described in chapter 2 in a rather simple way, the underlying mechanics of the algorithms were omitted. For example, neither calculating the correlation between two axes in polar ordination nor how PCA is used to calculate the ordination scores with eigenanalysis were discussed in detail. This means that the methods were used although not all the aspects of the method have been understood to their fullest (it would not be economical to do that). It is plausible, yet not probable, that given a certain dataset the method might render very poor results and the reason for this may be contained within its underlying mechanics. This can be compared to the falsifiability of a theory, as described by Karl Popper (1959), because it does not matter how many experiments with positive results are presented, it only needs one experiment with a negative result to falsify a theory. Or to use his example, if one only sees white swans in the world the hypothesis that "there are only white swans in the world" cannot be verified until all the swans in the world are looked at. As soon as a black swan is found, however, the hypothesis is falsified. To return to the context of the dimensionality reduction methods, only calculating all possible outcomes of an analysis with a certain method X would verify the hypothesis that "method X delivers suitable results". Because assuring this would be impossible, it is a limitation of this thesis, regardless which method is ultimately selected as the best one.

As pointed out earlier, by combining methods with NMDS the process can be made to be deterministic and it has performed well in a pre-evaluation by itself. Combining the methods can improve the result, since NMDS would optimise a result that is already close to the best solution. Additionally, the aforementioned limitation is weakened by the combination because if a "black swan" result is provided as the input for the NMDS, the algorithm will use this input to at least reach a local optimum. This behaviour also eases poor results of methods that may be caused by not satisfying the underlying assumptions of monotonic and unimodal input data.

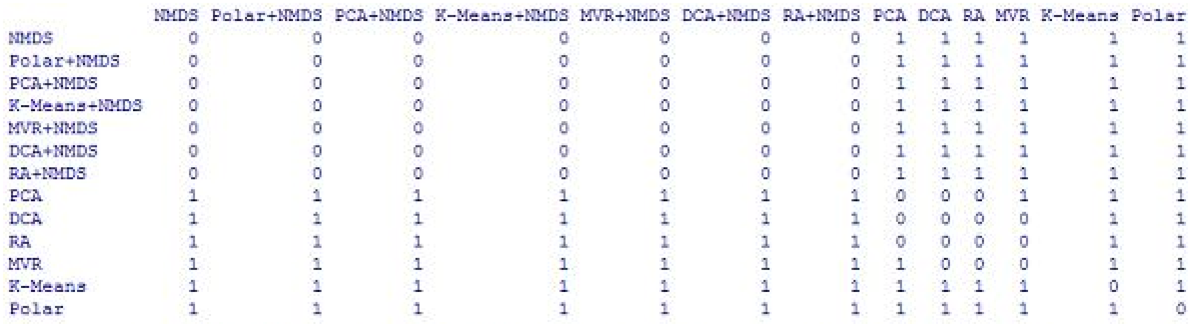
Thus, in addition to evaluating all methods including the non-deterministic NMDS, the combinations of NMDS with all the other methods is also part of a new evaluation.

By adding the new combination of methods and running the analysis again, the following new rank order of MSE means is gotten:

NMDS	Polar+NMDS	PCA+NMDS	k-means+NMDS	MVR+NMDS	DCA+NMDS
47.36745	47.39967	47.62144	47.64163	47.65388	47.67547

RA+NMDS	PCA	DCA	RA	MVR	k-means	Polar
47.77992	60.19809	63.24389	64.33315	66.58581	78.37201	108.41116

Instead of individually reporting the p-value of a t-test for each pair of means, a matrix has been constructed. The matrix in figure 5.2 does not show the p-values themselves but rather whether the difference between values is significant or not; it is significant when it is below 0.05. A one (“1”) indicates significance. It is visible that the difference of the means between a NMDS and any other method is significant. In fact, the means of MSE values for all the NMDS combinations are insignificantly different compared to each other. This indicates that according to this evaluation it does not matter which of these methods is used. This conclusion is expected given that the MSE means all lay very close to each other in relation to non-NMDS based methods.



	NMDS	Polar+NMDS	PCA+NMDS	K-Means+NMDS	MVR+NMDS	DCA+NMDS	RA+NMDS	PCA	DCA	RA	MVR	K-Means	Polar
NMDS	0	0	0	0	0	0	0	1	1	1	1	1	1
Polar+NMDS	0	0	0	0	0	0	0	1	1	1	1	1	1
PCA+NMDS	0	0	0	0	0	0	0	1	1	1	1	1	1
K-Means+NMDS	0	0	0	0	0	0	0	1	1	1	1	1	1
MVR+NMDS	0	0	0	0	0	0	0	1	1	1	1	1	1
DCA+NMDS	0	0	0	0	0	0	0	1	1	1	1	1	1
RA+NMDS	0	0	0	0	0	0	0	1	1	1	1	1	1
PCA	1	1	1	1	1	1	1	0	0	0	1	1	1
DCA	1	1	1	1	1	1	1	0	0	0	0	1	1
RA	1	1	1	1	1	1	1	0	0	0	0	1	1
MVR	1	1	1	1	1	1	1	1	0	0	0	1	1
K-Means	1	1	1	1	1	1	1	1	1	1	1	0	1
Polar	1	1	1	1	1	1	1	1	1	1	1	1	0

Figure 5.2: Significance of the difference between dimensionality reduction methods in terms of their mean square error (MSE) values

The same is valid for PCA, DCA and RA since their values are also not significantly different. MVR would belong to this group but there is a significant difference between it and PCA, which means that MVR is worse in this case. K-means and polar ordination are both ranked by themselves as the worst methods. The rank order is then adjusted for the groups:

NMDS-based methods	PCA, DCA and RA	MVR	k-means	Polar

5.1.2 Quantitative analysis of scalability

As there have not been any large scale tests with the TBDis system, it is not yet known how large a debate can be on average. How many participants will typically debate on the same main hypothesis? Or what is the average number of statements per participant? These are questions that currently cannot be answered. It is thus not known how scalable the method should be, which means that the importance of the scalability criterion is debatable. Nonetheless, performing an analysis of scalability is a good idea. This analysis also falls under a quantitative analysis with random data because a large dataset is needed and is used for calculations.

In case none of the methods scale linearly, an alternative way to reduce the amount of statements in the opinion landscape was thought of. It involves using a smaller dataset as input for the multivariate methods, and including only the most important statements in the opinion landscape could be a solution. The criterion for the decision about the importance of each statement could be taken from the context of the matrix. For example, the closer to the main hypothesis the statement is, the more important it is. Or the more ratings a statement has, the more important it is. It may also be possible to combine both methods by giving each statement a score that is based on both criteria, ranking them in order and using only the ones with higher scores for the opinion landscape. However, this would contradict the aim to involve all statements in the analysis.

Previously, the rank order of the methods for dimensionality reduction was created on the basis of the mean of the MSE values for 500 random datasets. To analyse the scalability of methods, the MSE is also used. Unlike in the previous analysis where the mean of the MSE values for each method are compared, in the scalability evaluation, the MSEs are compared over the size of the random datasets. For each method of dimensionality reduction, the MSE value is then plotted against the amount of ratings (`NoParticipants` times `NoStatements`):

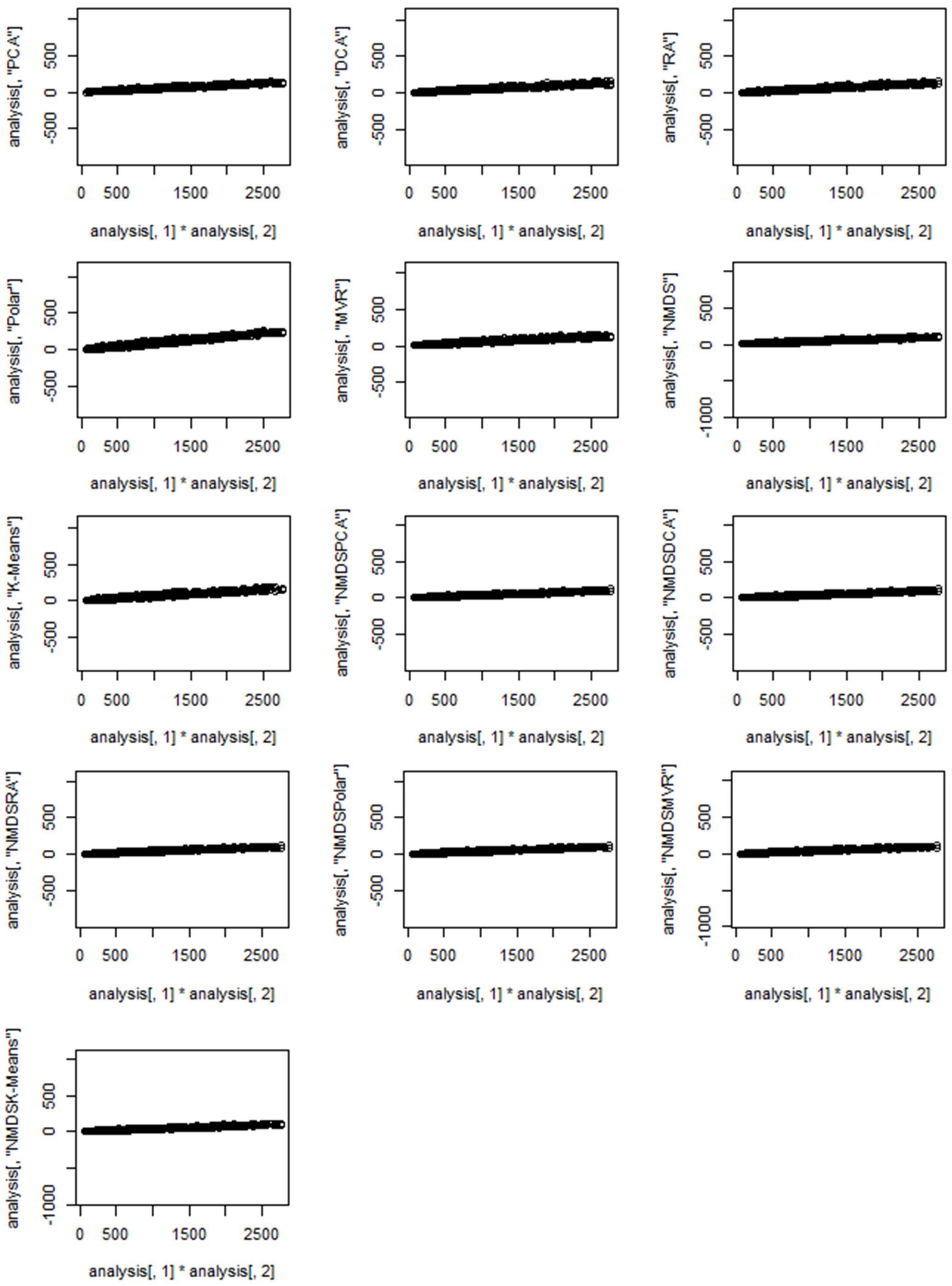


Figure 5.3: MSE values plottet against the number of ratings for each dataset

It is then visible for each scatter plot how much larger the MSE gets the bigger the dataset. The criterion for scalability itself is how fast the MSE value rises. The faster the value rises, the less scalable the method is. Since the values seem to rise linearly as opposed to exponentially, quantifying the slope can be done by performing a simple regression analysis. The linear model for a method uses its MSE (stored in, for example, `analysis[,”PCA”]`) as the dependent variable and the amount of ratings (calculated by `analysis[,1] * analysis[,2]`). The linear model is as follows:

$$\mathrm{MSE}_i = x * (N_{participants} * N_{statements})_i + c \quad i = 1, \ldots, 500$$

The output is the X factor and the C constant that form a straight line through the data cloud. The X factor is the slope of the line. The calculation of the slope is implemented in R as follows:

```
linearModelPCA = lm(analysis[,”PCA”]~x)
slope = linearModelPCA$coefficients["x"]
```

Performing a regression analysis for a method to calculate the slope over 500 datasets gives an indication of how scalable the method is. The rank order for the slopes of each method is as follows:

NMDS	MVR+NMDS	Polar+NMDS	k-means+NMDS	PCA+NMDS	DCA+NMDS
0.03615658	0.03644072	0.03649889	0.03667487	0.03680011	0.03681509

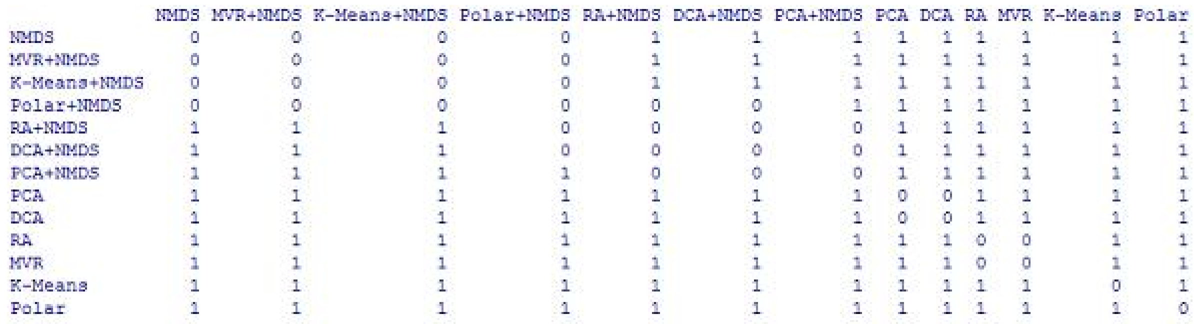
RA+NMDS	PCA	DCA	RA	MVR	k-means	Polar
0.03700819	0.04750052	0.04808712	0.04912571	0.04978748	0.05921730	0.08919249

This rank order is similar to the rank order of the means of MSE values; although this is only one sample of regression slopes. It is comparable to just one of the rows in figure 5.1. The next step is to calculate more regression slopes to determine if, on average, NMDS is still more scalable than the other methods. Thus the analysis of 500 random datasets is performed ten times to produce ten scalability evaluations. The rank order of the means after ten iterations is as follows:

NMDS	MVR+NMDS	k-means+NMDS	Polar+NMDS	RA+NMDS	DCA+NMDS
0.03686137	0.03693122	0.03696915	0.03703584	0.03728291	0.03731739

PCA+NMDS	PCA	DCA	RA	MVR	k-means	Polar
0.03735832	0.04831647	0.04852488	0.04968499	0.05008463	0.06032849	0.08977895

This rank order indicates that for an average of 10 samples, NMDS is the most scalable method. To see if this result is significant, several t-tests are performed. They are performed in the same way as with the evaluation of MSE values. The result is a matrix with the result of the t-tests for every possible comparison. Again, the matrix in figure 5.4 does not show the p-value itself, but rather if the difference in the rank order is significant (a one ("1") if the p-value is below 0.05), or not (a zero). While NMDS is the most scalable method according to the rank order, the difference is not significant. This time, the NMDS-based method combinations are split in two groups. Again, the insignificance of difference within the groups means that it does not matter which one of the methods are used because they are equally scalable.



```
             NMDS MVR+NMDS K-Means+NMDS Polar+NMDS RA+NMDS DCA+NMDS PCA+NMDS PCA DCA RA MVR K-Means Polar
NMDS            0        0            0          0       1        1        1   1   1  1   1       1     1
MVR+NMDS        0        0            0          0       1        1        1   1   1  1   1       1     1
K-Means+NMDS    0        0            0          0       1        1        1   1   1  1   1       1     1
Polar+NMDS      0        0            0          0       0        0        1   1   1  1   1       1     1
RA+NMDS         1        1            1          0       0        0        0   1   1  1   1       1     1
DCA+NMDS        1        1            1          0       0        0        0   1   1  1   1       1     1
PCA+NMDS        1        1            1          1       0        0        0   1   1  1   1       1     1
PCA             1        1            1          1       1        1        1   0   0  1   1       1     1
DCA             1        1            1          1       1        1        1   0   0  1   1       1     1
RA              1        1            1          1       1        1        1   1   1  0   0       1     1
MVR             1        1            1          1       1        1        1   1   1  0   0       1     1
K-Means         1        1            1          1       1        1        1   1   1  1   1       0     1
Polar           1        1            1          1       1        1        1   1   1  1   1       1     0
```

Figure 5.4: Significance of the difference between dimensionality reduction methods in terms of scalability

RA combined with NMDS and DCA with NMDS are not in the first group because they are only not significantly different from one of the methods of that group. The remaining groups are PCA and DCA, RA and MVR, and both k-means and Polar ordination are ranked by themselves. Taking the significantly different groups into account, the rank order of scalability is as follows:

NMDS and its combinations with MVR, Polar and k-means	NMDS combined with RA, DCA and PCA	PCA and DCA	RA and MVR	k-means	Polar

In this section, two criteria for methods of dimensionality reduction have been analysed. The first result is that any NMDS-based methods perform better as their MSE values are fairly low. A low value means that the dimensionality reduction has fewer errors. These methods are better in assuring that the distances between participants in the multidimensional space are portrayed accurately in the two-dimensional scatter plot. The second result is that some of these NMDS-based methods are more scalable than others. Particularly pure NMDS and the combinations of NMDS with the output of MVR, Polar ordination or k-means are the most scalable. Since these four methods are the most scalable and are also the better performing in terms of their MSE values, either method would be a good choice for the TBDis system.

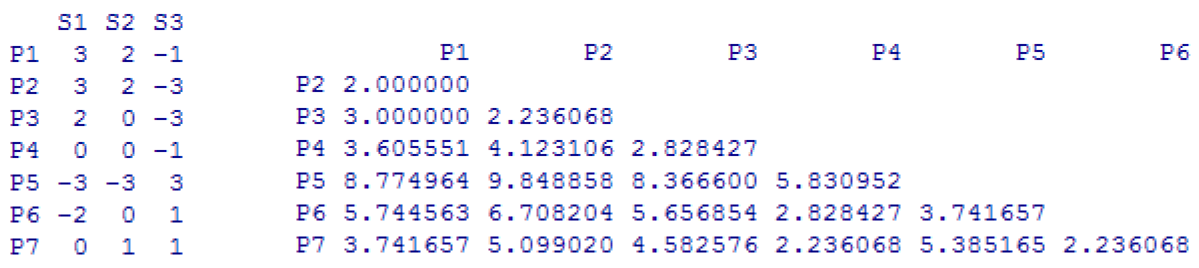
5.1.3 Interpretative analysis using constructed data

In the previous section, methods for dimensionality reduction were evaluated based on performance values in terms of two quantitative criteria. In this section, the methods undergo an interpretative analysis with constructed data. The data used in the interpretative analysis is grounded in the context of the TBDis system and it is described in chapter 4. The goal is to evaluate whether or not the methods are suitable in context. One could compare this evaluation method with giving or withholding a recommendation for a candidate for a political position.

According to Gauch (1982), applying the methods is just one side of the coin. The external analysis, i.e. the interpretative analysis, is just as important. The importance of this analysis is even greater than the previous analyses because even if a method is scalable and in theory renders the scatter plots with the least errors, it should not be selected if it does poorly with data in the context of TBDis. Moore et al. (1970) point out the importance of using constructed data by explaining that the efficiency of a method can be judged by how understandable the structural complexity is to the viewer.

Firstly, an example of how such an analysis is performed will be described. Thereafter, the results of the interpretation are presented. It is again pointed out that these are results of face validity which is an interpretation by the researcher. In a future work, an analysis by means of face validity should be done in a lab experiment with more people.

As an example, a multivariate regression (MVR) of constructed data for dimensionality regression is interpreted. The dataset used in this example is the first constructed dataset:



```
   S1 S2 S3
P1  3  2 -1
P2  3  2 -3
P3  2  0 -3
P4  0  0 -1
P5 -3 -3  3
P6 -2  0  1
P7  0  1  1
```

```
         P1       P2       P3       P4       P5       P6
P2 2.000000
P3 3.000000 2.236068
P4 3.605551 4.123106 2.828427
P5 8.774964 9.848858 8.366600 5.830952
P6 5.744563 6.708204 5.656854 2.828427 3.741657
P7 3.741657 5.099020 4.582576 2.236068 5.385165 2.236068
```

Figure 5.5: A constructed opinion landscape (dataset Nr1) with distance matrix

As with all constructed datasets, the opinion landscape in this dataset has a particular structure. In this case, P1 and P2 should be close enough to form a group in a cluster analysis, while in a dimensionality reduction the cluster should be the farthest apart from P5.

The output of the MVR is the visualisation used for this evaluation (see figure 5.6 (a)). In any case, it is important that the aspect ratio is set to one so that there are no distortions based on the aspect ratio.

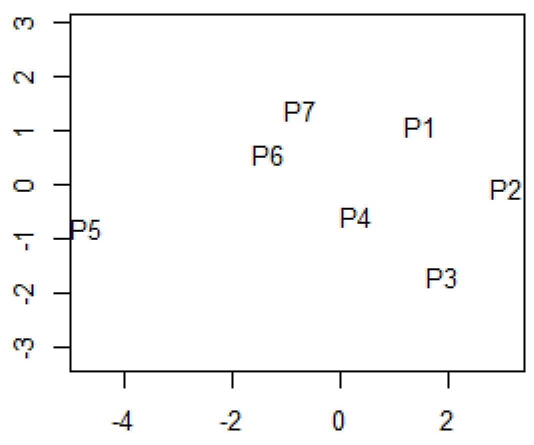


Figure 5.6 (a)

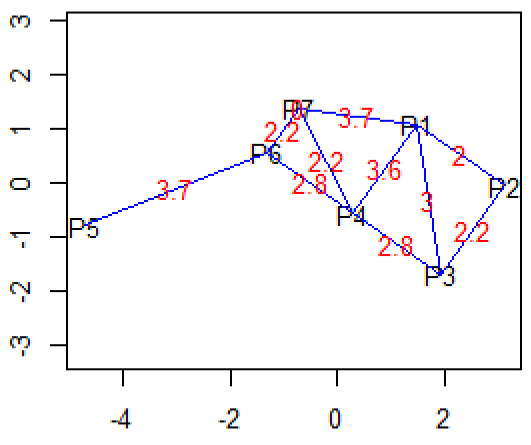


Figure 5.6 (b)

To begin with, the method is evaluated in terms of whether the designed structure is visible or not. Indeed P2 and P5 are very far apart in figure 5.6 (a). This aspect of the intended underlying structure is reflected in the output, but the closeness of P1 and P2 is not reflected.

Secondly, an evaluation is conducted based on the four general patterns for constructed data that were described in chapter 4. These patterns have been designed in datasets 4 to 10 so that they do not show up in figure 5.6. Nonetheless, participants with less strong opinions are expected to be found towards the centre. This is the case, as P4, P6 and P7 are in the middle of the scatter plot.

Thirdly, some distances between the participants in the scatter plot are compared to the distance matrix (see figure 5.6 (a)). As mentioned before, the axes have no meaning and the effective distance from the distance matrix has to be compared proportionally. For instance, the distance between P7 and P6 is roughly 2.2 which is the same distance as between P7 and P4. Visually, both distances between the participants are roughly in the proportion of 2:1. In other words, the distance between P7 and P6 seems to be double the distance between P7 and P4. However, the distance matrix shows a proportion of 1:1 (the same distance). This is a result of middle quality. It is not perfect but it would be much worse if, for example, P4 were much farther away than P6.

A tool to help the evaluation of the latter criteria is a custom code snipped draws line between participants whose distance is shorter than the mean of all distances in the distance matrix. For larger datasets, the condition was changed to half of the mean distance. Additionally, the distance value is displayed on the line (see figure 5.6 (b)):

```
distM = as.matrix(dist(mydata))
for(i in 1:ncol(distM))
        for (j in (i+1):nrow(distM))
                if (round(distM[i,j],1) < mean(distM)) {
                        segments(xf[i],yf[i],xf[j],yf[j],col="blue")
                        text(((xf[i]+xf[j])/2),((yf[i]+yf[j])/2),labels=c(round(distM[
                i,j],1)), col="red")
                }
```

The comparison of the distance matrix with the distance in the scatter plot is based on the same principle as quantitatively calculating the mean square error (MSE). Here, the principle is applied to constructed data in the context of the TBDis system rather than on randomly constructed datasets.

The final score for this method on the first dataset is low. Although the proportions seem good in some cases, the fact that the strong proximity between P1 and P2 cannot be reflected is enough to lower the score.

Here, the results of the interpretative analysis of multivariate methods with constructed data are presented. The following table gives an overview of the results of the first five datasets:

Methods for dimensionality reduction	Constructed datasets									
	Dataset 1		Dataset 2		Dataset 3		Dataset 4		Dataset 5	
PCA	A	0.02	A	0.2	B	8.19	A	0.3	A	1.6E-10
DCA	-	0.99	B	1.42	A	4.64	-	1.95	B	2.03
RA	-	1.12	B	1.48	A	4.56	A	1.21	B	2
Polar	A	0.03	A	0.23	-	14.8	B	0.63	-	ERROR
MVR	B	0.6	-	0.12	B	12.3	A	0.66	A	0.04
NMDS	-	-	-	-	-	-	-	-	-	-
k-means	A	0.14	-	0.98	-	15.6	-	0.89	A	0.007
PCA+NMDS	A	0.02	A	0.18	B	9.27	-	0.47	A	1.6E-10
DCA+NMDS	B	0.21	B	0.31	B	9.3	B	0.55	B	1.09
RA+NMDS	B	0.13	B	0.16	B	9.3	-	0.48	B	1.13
Polar+NMDS	A	0.03	A	0.26	B	9.98	A	0.31	-	-
MVR+NMDS	B	0.56	A	0.24	B	9.29	-	0.49	A	3.5E-4
k-means+NMDS	A	0.06	A	0.44	B	9.26	-	0.7	A	7.1E-3

Table 5.1: Results of the interpretative analysis with comparison to each MSE

For each dataset and method a score was given. An “A” or a “B” indicates that the method passed the face validity criteria for this dataset. Methods with an “A” performed better than methods given a “B”. A dash indicates that the method did not meet the criteria. NDMS was not evaluated here because the output is non-deterministic.

As discussed, the interpretative analysis gives an indication about whether or not the method is suitable in the context of the TBDis system. Interesting results were observed, for example, the performance of MVR on the second dataset. Although this method received the lowest mean square error (MSE) value, it is not recommended because in the output of the method P6 and P7 are too close. This was not intended in the design of the data matrix. Thus the method did not meet the face validity criterion for this dataset. Another case where the results of the quantitative approach and face validity differ is in dataset four. Here, some NMDS-based methods are not recommended despite their rather low MSE values because in their output P7 and P9 are in exactly the same spot. This is not possible as they both have different opinion vectors.

While evaluating the methods using the fifth dataset, there was an error related to polar ordination. The error read “`Error in if (A < B) { : missing value where TRUE/FALSE needed`”. This is unexpected because polar ordination worked well for the many iterations over random datasets. The function also worked on all other constructed dataset and all other methods worked on this particular set. So far, the features of the context for the error have not been narrowed down. It is important to note that the combination of polar ordination and NMDS is also not recommended because in the case of an error polar ordination does not provide any coordinates as input for the NMDS. Since there is then no input for the NMDS, the algorithm assumes a random initial condition and behaves as a regular NMDS.

The following table provides an overview of the results of the last five datasets.

Methods for dimensionality reduction	Constructed datasets									
	Dataset 6		Dataset 7		Dataset 8		Dataset 9		Dataset 10	
PCA	A	0.3	B	1.9	B	0.47	A	0.034	B	0.33
DCA	B	3.05	-	9.2	-	1.57	-	2.65	-	0.62
RA	B	3.05	-	9.9	-	1.49	-	1.89	-	0.58
Polar	A	0.93	B	2.57	B	0.61	A	0.05	B	0.36
MVR	-	0.54	-	2	B	0.67	A	0.12	B	0.39
NMDS	-	-	-	-	-	-	-	-	-	-
k-means	-	1.67	B	2.7	-	0.76	B	0.62	B	1.46
PCA+NMDS	A	0.28	A	1.2	A	0.25	A	0.03	A	0.28
DCA+NMDS	A	0.28	B	1.74	A	0.31	A	0.13	A	0.31
RA+NMDS	A	0.28	B	1.73	A	0.25	A	0.07	A	0.29
Polar+NMDS	A	0.28	A	1.25	A	0.25	A	0.04	A	0.3
MVR+NMDS	A	0.28	A	1.26	A	0.22	A	0.04	A	0.28
k-means+NMDS	-	0.51	A	1.26	A	0.33	A	0.28	A	0.38

Table 5.2: Results of the interpretative analysis with comparison to each MSE (cont.)

Interesting patterns can also be found in the sixth to tenth datasets. For instance, in the sixth dataset, MVR and the combination of k-means with NMDS are not recommended because some participants would be incorrectly clustered together. In the seventh dataset, DCA and RA have a quite poor score and MSE value because, for some reason, P11 (a participant with inconsistent opinions, agreeing to all statements) is displayed in the middle of the cloud while in all other methods the participant is correctly an outlier.

Some other observations were documented during the evaluation:

- The MSE values for polar ordination using constructed data seldom lie towards the end of each individual rank order. This is in contrast to the method’s MSE values using randomised data.

- More often than not, the output of a DCA or RA fails to display the participant with zero values in the matrix in the middle of the scatter plot exactly between P1 and P2.
- The dataset with the output of a TBDis test with participants that was not carefully constructed has much higher MSE values than all the other datasets.
- In the same way that PCA was presented as being treated as a black box and NMDS as able to correct poor results, one of the results of polar ordination seems to have been a black swan. The error that occurred in a polar ordination hints to the problems of treating the methods as a black box. Perhaps the error is also caused by a faulty implementation of the method in R, which was relied upon for correctness.

To quantify the results of face validity and put the methods in rank order, the scores for each method and dataset are weighted and added to produce a final score. The weighting for an “A” score is two, for a “B” score is one and for a dash “-” is zero. The rank order is as follows:

PCA+NMDS and Polar+NMDS	PCA and MVR+NMDS	k-means+NMDS	DCA+NMDS
17	16	15	14

RA+NMDS	Polar	MVR	RA and k-means	DCA	NMDS
13	12	10	7	5	0

In this section an evaluation based on the face validity criterion for methods of dimensionality reduction has been performed.

Although in the resulting rank order the method combining polar ordination with NMDS had the highest score, it is not recommended because the implementation for a dataset threw an error. It is assumed that this is not the problem of treating polar ordination as a black box but rather an implementation problem. In conclusion, if polar ordination is not reliable whether in its implementation or algorithm, the method is not recommended. This leaves the combination of PCA with NMDS as the best performing method given the criterion. However, even for this method, there was a dataset where the method did not meet the criteria of face validity.

5.2 Evaluation and discussion of methods for cluster analysis

In this section, the methods for cluster analysis are evaluated. Just like the evaluation in the previous section, this evaluation is also based on the subjective criteria of face validity. Moreover, three of the methods require setting some parameters before performing them: the threshold for the WSS optimisation in k-means, the p-value in hierarchical clustering and the density defining parameters for density-based clustering. This adds another layer of subjectivity.

5.2.1 A disclaimer about quantitative analysis using random data

As discussed in chapter 4, no evaluation method has been found that treats every clustering method equally. When analysing the criteria of high within-cluster similarity and low between-cluster similarity, there is a particular way to set ESP and MINPTS parameters so that density based clustering is the most suitable according to this criterion. When analysing the optimal amount of clusters, there is an optimal threshold for the optimisation algorithm in the pre-processing of a k-means clustering so that k-means is the most suitable according to this criteria.

5.2.2 Interpretative analysis using constructed data

In this section, four cluster methods are compared. In addition to the three methods presented in chapter 2, the current clustering algorithm in the TBDis system is added to the evaluation.

As described in chapter 3, the TBDis clustering algorithm requires statements to be sorted in level order prior to the analysis. Moreover, the algorithm makes use of structural information about the debate. For each statement, the level of a statement in a level-sort-order the statement must be known. In this analysis this information is provided to the algorithm.

The evaluation is similar to the interpretative analysis of methods for dimensionality reduction with the exception that the raw output of the clustering methods is also displayed in table 5.3 and 5.4. The raw outputs of hierarchical clustering, k-means and density-based clustering are a series of numbers. The first number indicates the cluster that P1 is in; the second is for P2, etc. The numbers themselves have no meaning but being assigned the same number results in two participants being placed in same cluster. For each dataset, the expected clusters are displayed in the table as well. They are expected according to the designed structures described in chapter 4.

	Expected clusters	Hierarchical clustering		k-means	
1	P1,P2	8 9 10 11 12 13 14	-	3 3 3 2 1 2 2	B
2	P2,P3	8 1 1 2 12 2 2	A	2 2 2 1 1 1 1	B
3	-	1 14 15 16 17 1 19 20 21 1 23 24	-	2 1 2 2 2 2 2 2 2 2 2 2	-
4	-	11 12 13 14 15 16 1 18 1 20	-	2 1 2 3 2 3 4 3 4 4	-
5	P1,P3 and P2,P4	6 7 8 9 10	-	2 1 2 1 1	B
6	P2,P6	6 8 6 8 6 8 6 8 8 8 8	B	2 1 3 1 2 1 3 4 5 4 4	B
7	P2,P6	14 1 3 17 18 1 3 2 22 2 24 25 26	A	2 3 2 3 2 3 2 3 1 3 1 3 1	B
8	P3,P5,P7 and P4,P6	13 14 2 1 2 1 2 20 21 22 23 24	A	2 3 1 3 1 3 1 3 1 4 4 4	B
9	-	9 10 11 1 13 1 15 16	-	2 1 2 1 2 1 2 2	-
10	P1,P7	19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36	-	1 5 1 4 3 4 3 2 1 2 1 4 1 4 3 4 3 3	-

Table 5.3: Outputs of the cluster analysis methods and scores given by the researcher

	Expected clusters	Density-based		TBDis algorithm	
1	P1,P2	1 1 1 2 3 2 2	B	P1,P2 and P4,P7	A
2	P2,P3	1 2 2 3 4 3 3	A	P2,P3 and P1,P7	A
3	-	1 2 3 4 5 1 6 1 7 1 8 9	-	P3,P4,P6,P7,P8 ,P11	-
4	-	1 2 1 3 1 3 4 3 4 4	-	All in one cluster apart from P10	-
5	P1,P3 and P2,P4	1 2 1 3 3	B	P1,P3	B
6	P2,P6	1 2 3 2 4 2 3 5 6 5 5	B	P2,P6 and P3,P7 and P8,P10	A
7	P2,P6	1 2 3 4 3 2 3 5 6 5 7 5 5	A	P2,P6 and P8,P10 and P1,P11	A
8	P3,P5,P7 and P4,P6	1 2 3 2 3 2 3 2 4 4 4 4	B	P1,P3,P5 and P2,P4,P6	-
9	-	1 2 3 4 3 4 3 3	-	P3,P5 and P4,P6,P7	-
10	P1,P7	1 2 3 4 3 4 3 2 3 2 3 4 3 4 3 4 3 3	-	P3,P9 and P8,P10 and P5,P11 and P4,P12 and P6,P14 and P7,P15,P17	-

Table 5.4: Outputs of the cluster analysis methods and scores given by the researcher (cont.)

An interesting observation is that k-means and density-based clustering have the same output for the first, fourth and fifth datasets. The outputs of the sixth and ninth datasets are almost identical.

For the fifth dataset, the TBDis algorithm actually has the same output. It is not shown as such because the algorithm operates under different assumptions than the other methods. To clarify, in the TBDis algorithm, the participants with zero values for all statements (i.e., “non-voters”) are filtered out of the cluster analysis. Thus P4 and P5 do not build a cluster. This implies that when comparing one of the three new methods to the output of the TBDis algorithm, the last participant from each dataset has to be ignored because according to chapter 4 it is a non-voter.

The scoring system is slightly different than interpreting scatter plots of dimensionality reduction methods. An “A” means that the method managed to create a cluster that encompasses exactly the participants that were expected to be clustered. Further clusters that the method provides are ignored. A “B” is if there are more than just the expected participants in the cluster. A dash means that the method failed to detect the expected clusters. For example, if a dataset was designed to have P1 and P2 in a cluster, an “A” would be attributed to the method that has a cluster that only encompasses these two participants. A “B” is attributed to a method that has P1 and P2 in a cluster together with one or more other participant(s). A “-” would be attributed to a method that does not cluster P1 with P2 in any way. The rank order is as follows:

TBDis algorithm	Density-based clustering	Hierarchical clustering	k-means
9	8	7	6

Given the ten datasets (seven of which contained structures for evaluation), the TBDis algorithm performs better than the other methods. Since the algorithm is built into the TBDis system and even makes use of structural data, an above average performance was expected. In addition, this is the outcome of a subjective interpretative analysis performed by the researcher rather than in an experimental setting, which is more ideal.

6. Conclusions

In this thesis, one facet to reduce the complexity of information for people has been tackled. In the field of collaborative technologies, more specifically online debating, participants have the possibility to argue about a particular topic. Participants in a debate can input arguments into the system to help the group decide about the topic. The TBDis system was developed with the goal of structuring these arguments. Additionally, on this structural layer, participants can express their opinion on other participant's arguments by rating the statements of these arguments. This adds a second layer of information to the debate. When debating using the TBDis system, participants can only see the ratings per statement. In this thesis, a way of displaying an overall summary for all statements is proposed. The information from the methods is based on the ratings and is presented to the participants in a report about the debate.

The two goals of this thesis are concerned with the presentation of this information layer described as the opinion landscape. The first goal is to find the best method to reduce the dimensionality of the matrix that stores the ratings. The resulting scatter plot is a two-dimensional representation of the opinion landscape where participants can identify their position in relation to their peers. The ordination methods evaluated are polar ordination, principal component analysis (PCA), reciprocal averaging (RA) and detrended correspondence analysis (DCA). Other multivariate methods in the evaluation are multivariate regression (MVR), non-metric multidimensional scaling (NMDS) and dimension clustering using k-means clustering.

The second goal is to analyse the rating so that clusters can be formed. The resulting list depicts which participants have such similar opinions that they can be categorised into a fraction. To achieve this goal, four clustering methods are described and evaluated, one of which is the current fraction detecting algorithm of the TBDis system. The other clustering methods evaluated are hierarchical clustering, density-based clustering and k-means clustering.

To evaluate the multivariate methods, both quantitative and interpretative analyses have been performed. As applicable, the analysis has been performed with either random or constructed data in the context of the TBDis system. The results of each analysis are as follows:

1. A quantitative analysis of methods for dimensionality reduction with random data resulted in NMDS being ranked as the best performing method. Even if on average the method had the lowest mean square errors (MSE), it is not suitable as the method to be used to calculate the result presented to the participants because it is non-deterministic. A significance test shows that NMDS is not significantly better than any other NMDS-based methods. Either one of these would be suitable.

2. The same analysis was conducted to assess scalability, which resulted in NMDS being ranked the most scalable method. A significance test shows that four methods are not

significantly different from each other. Any of the combinations of NMDS with polar ordination, MVR or k-means are also suitable.

3. An interpretative analysis of methods for dimensionality reduction with constructed data resulted in the combinations of NMDS and either PCA or polar ordination as the most accurate in representing the designed underlying structures in the context of the TBDis system.
4. An interpretative analysis of clustering methods with constructed data showed that the built-in TBDis algorithm produces the most accurate representation. It is followed by density-based clustering, hierarchical clustering and finally k-means as the poorest method.

Other types of analysis such as interpreting random data were not performed as they were not appropriate here.

In terms of methods for dimensionality reduction, NMDS consistently performed the best in the quantitative analysis. NMDS-based methods also consistently performed on top of all others in the quantitative and interpretative analysis. The only non-NMDS-based method that performed well was PCA in the interpretative analysis. Within the NMDS-based methods, the combinations with MVR, polar ordination or k-means were amongst the best in reducing dimensions and at the same time, they were the most scalable. In the interpretative analysis, the combination of NMDS with either polar ordination or PCA was the most accurate.

This narrows down the selection of methods to the combination of NMDS with polar ordination as the most suitable choice for the TBDis algorithm. Nonetheless, polar ordination is severely limited since it is unable to handle one of the constructed datasets (see chapter 6.1). Whenever this is the case, the non-deterministic result of a pure NMDS is produced. Just as NMDS itself cannot be recommended because the outcome is not guaranteed, this limitation means that the combination of NMDS and polar ordination should not be recommended.

The next two methods in the general rank order are the combinations of NMDS with either PCA or MVR. Both performed the best in the first analysis (i.e. quantitative analysis in terms of having a low MSE value). NMDS with MVR was significantly more scalable than NMDS with PCA, while the latter resulted in more accurate outputs in the interpretative evaluation with constructed data. The decision between the methods depends on how important the interpretative analysis is in this thesis. Indeed, the interpretative analysis has a higher weighting in such a case because it has been defined as the strongest criterion for exclusion. The second factor for the decision is that the quantitative analysis in terms of scalability is based on the results of the first quantitative analysis. Since it is also based on the MSE, there is some redundancy between the quantitative analyses.

Thus although not as scalable, the combination of NMDS with PCA is the final best recommendation for the TBDis system.

Nonetheless, it would be interesting to analyse the decision between these two methods when the scores for the methods in an interpretative analysis are given as a result of a field experiment. It is important to point out that the evaluation, in its current configuration, might be biased towards NMDS-based methods. As described in chapter 2, NMDS works by minimizing the difference between the original distance matrix and the distance matrix in the iterations of the algorithm using the stress function as the measure of quality. MSE, as chosen as the measure of quality of the dimensionality reduction, works in a similar way by measuring the difference between the original distance matrix and the distance matrix after the dimensionality reduction. Because the NMDS method work in a similar manner as the evaluation method, there might be a bias towards NMDS-based methods for dimensionality reduction.

The outcome of the interpretative analysis for clustering methods suggests that the TBDis algorithm is the most suitable for the TBDis system. The algorithm might be advantageous because structural information about the debate is considered within it. An advantage of keeping this algorithm is that it does not require parameters with values that can be subjectively defined.

In short, there is a high degree of confidence that the goals of the thesis can be reached by using the recommended methods. The combination of NMDS and PCA is one of the best performing and most accurate methods with constructed data. The TBDis algorithm, although not perfect, renders a good result in the context of the TBDis system.

This thesis also faces some limitations, some of which have already been discussed. The limitations are as follows:

1. The interpretative analysis has been conducted by the researcher. This means only one person has looked at the output of the methods and given them a score. Granted, there is a list of criteria and patterns (also subjectively defined) that give a higher degree of formalisation, but it would be better to perform the analysis in an experimental setting with more people.
2. Cluster algorithms need subjectively set parameters. Parameters, like the threshold for the WSS optimization in k-means, the p-value in hierarchical clustering and the density defining parameters for density-based clustering, are subjectively determined by the researcher.
3. The criterion for the quantitative analysis of methods for dimensionality reduction is subjective. Choosing to use the difference between the distance matrix before and after

applying the method on data is subjective. The method of quantifying the difference between the distance matrices, the MSE value, is also a subjectively chosen measure.

4. The fact that the underlying mechanism for many of the methods is not known raises the likelihood of an unexpected and unsuitable output. Additionally, not knowing whether the opinion landscapes in the TBDis system are monotonic or unimodal might have hindered better results. Piping the outputs of the methods as the input of a NMDS eases both problems because even if a poor input is provided, NMDS excels in its performance (on average). In fact, NMDS performs on average the best with random inputs. A poor input is assumed to be no worse than random.

5. The general reliance on the implementation of the multivariate methods in R can be problematic. For instance, polar ordination had an error with the fifth constructed dataset. A drill down into the open source code revealed that the error was in the calculations of matrices in a function for the variance regression method of selecting the poles. This error is unexpected and the context of the error is not known since the method had not experienced problems with any other dataset and all the other methods produced no errors when analysing the fifth dataset.

6. The performance of the methods in terms of computation time has not been analysed and weighted for the evaluation. This is important because the participants should not wait for a website to load. For instance, the pvclust() function applied on constructed datasets takes several seconds before the output is presented.

6.1 Future work

In this section, some ideas for future work are presented. The ideas range from tweaking some parameters of the evaluation to proposing alternative subjects of evaluation.

As pointed out in this chapter, the TBDis clustering algorithm was the most accurate in reflecting the intended structures in the datasets. The algorithm has an advantage because it takes information about the structure of the debate into account. An interesting comparison would be to observe how the TBDis algorithm performs without the structural information. It would be interesting to see the extent to which the effect is reduced without this information, if at all.

On the other hand, determining a way to feed structural information into the other three algorithms might improve them. Furthermore, an analysis of the same datasets with and without structural information may indicate how important structural information is for both types of multivariate methods.

The choice to use MSE as the measure of quality should be questioned because element-wise subtracting one distance matrix from another might be biased towards larger distances. The reason for this is that subtracting two large values from each other has a greater influence on the resulting MSE than subtracting two small values. For example, imagine that there are four participants, which means three values are found in the distance matrix: 4, 104 and 200. After the dimensionality reduction, the distance matrix has the values 4, 103 and 200. The differences are 4-4 = 0, 104-103 = 1 and 200-200=0. The MSE equates to 1/3. The difference between both matrices is not large. One of the distances is only 1/104 shorter. However, if the difference matrix after the reduction has the values 3, 104 and 200, the MSE value is the same although one of the distances is 25% (or 26/104) shorter. Although the difference in the scatter plot in the second case is greater, the MSE value is not greater than in the first case. It is thus influenced more by larger values in the distance matrix. A better method might be to calculate the percentage difference between the values rather than subtracting them. It may also be argued that this does not matter because all methods are evaluated by the same measurement of quality.

Furthermore, as discussed in the limitations, the scores in an interpretative evaluation should be given as a result of a field experiment.

The next step is to implement the new methods for multivariate analysis as a new feature in the TBDis system. The report that is produced would be used to show the current opinion landscape to the participant. Yet why should it end there? Another interesting piece of information for the participants might be to know how the opinion landscape changes over time. If the system has knowledge of the appearance of the current landscape and the landscape from one week earlier, a scatter plot could be used to represent the movements of the participants, which will indicate how participants changed their opinions over time. It might also be interesting to monitor the responses over time after a particularly strong argument is entered into the system.

Bibliography

ACM SIGKDD (2006). Data Mining Curriculum: A Proposal (Version 1.0). Retrieved on 05.04.2012 from http://www.sigkdd.org/curriculum/CURMay06.pdf

Althaus, S. (1996). Computer-mediated communication in the university classroom: An experiment with on-line discussions. *Paper presented at the annual meeting of the American Political Science Association*, San Francisco, CA.

Baranovskiy D. (2008). Raphaël Javascript Library. Retrieved on 20.07.2012 from http://raphaeljs.com/reference.html

Bateman, W. (2004). Online LaTeX Equation Editor - create, integrate and download. Retrieved on 20.03.2012 from http://www.codecogs.com/latex/eqneditor.php

Beals, E. W. (1984). Bray-Curtis ordination: an effective strategy for analysis of multivariate ecological data. *Advances in Ecological Research* 14: 1-55

Belnap, J. K. & Withers, M. G. (2008). Discourse analysis: The problematic analysis of unstructured/unfacilitated group discussions. Manuscript submitted for publication, Brigham Young University.

Bennett, S. & Bowes, D. (1976). An introduction to multivariate techniques for social and behavioral sciences. New York: Wiley.

Blei, D. M. (2008). Hierarchical clustering. Lecture in Interacting with Data. Retrieved on 27.07.2012 from http://www.cs.princeton.edu/courses/archive/spr08/cos424/syllabus.html

Boecker et al. (2005). A Hierarchical Clustering Approach for Large Compound Libraries. *J. Chem. Inf. Model.*, 45 (4), pp 807–815

Borg, I. & Groenen, P. (1997). Modern Multidimensional Scaling, Theory and Applications. New York, Springer-Verlag.

Borgatti, S. (1998). Multidimensional Scaling. Retrieved on 12.07.2012 from http://www.analytictech.com/networks/mds.htm

Bray, J. R. & Curtis, J. T., (1957). An ordination of the upland forest communities of southern Wisconsin. *Ecological Monographs*, 27, 325-49.

Carr, C. S. (2001). Computer-Supported Collaborative Argumentation: Supporting Problem-based Learning in Legal Education. *In Proc. of the Computer support for Collaborative Learning Conference (CSCL),* Maastricht University, Holland.

Chen, Y., Dong, G., Han J., Wah B. W. & Wang J. (2002). Multi-dimensional regression analysis of time-series data streams. *In Proceedings of the 28th international conference on Very Large Data Bases (VLDB '02)*. VLDB Endowment 323-334.

Collins, A., Brown, J. S. & Newman, S.E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. *In L.B. Resnick (Ed.), Knowing, learning, and instruction: Essays in honor of Robert Glaser (pp. 453–494).* Hillsdale, NJ: Lawrence Erlbaum Associates.

Crockford, D. (2008). JSON in JavaScript. Retrieved on 26.07.2012 from http://www.json.org/js.html

Ester, M., Kriegel, H. P. et al. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. *In Proc. 1996 Int. Conf. On Knowledge Discovery and Data Mining (KDD'96)*, pp. 226-231, Portland, Oregon.

Everitt, B. & Hothorn, T. (2011). An Introduction to Applied Multivariate Analysis with R. Springer.

Everitt, B. S. & Hothorn, T. (2009). A Handbook of Statistical Analyses Using R, Second Edition, CRC Press.

Fielding, R. T., Taylor, R. N. (2002-05), Principled Design of the Modern Web Architecture. *ACM Transactions on Internet Technology (TOIT).* New York: Association for Computing Machinery. 2 (2): 115–150

French, S. (1980). Updating of Belief in the Light of Someone Else's Opinion. Journal of the Royal Statistical Society, 143, Part 1, pp 43-48

Galagan, J. (2007). Classification and Clustering of Microarray Data. Lecture in Introduction to Computational Genomics for Infectious Disease Retrieved on 10.05.2012 from http://www.broadinstitute.org/annotation/winter_course_2006/

Garrett, J. J. (2005) Ajax: A New Approach to Web Applications. Retrieved on 26.07.2012 from http://adaptivepath.com/ideas/essays/archives/000385.php

Gauch, H. G. & Whittaker, R. H. (1972). Comparison of ordination techniques. *Journal of ecology* 53, 868–875.

Gauch, H. G. (1980). Rapid initial clustering of large data sets. *Vegetatio*, 42, 103-11

Gauch, H. G. (1982). Multivariate Analysis in Community Ecology. Cambridge University Press, Cambridge, England.

Gauch, H. G., Whittaker, R. H. & Wentworth, T.R. (1977). A comparative study of reciprocal averaging and other ordination techniques. *Journal of ecology*, 65, 157-74.

Green, P. E. & Rao, V. R. (1972). Applied Multidimensional Scaling: A Comparison of Approaches and Algorithms. New York: Holt, Rinehart and Winston.

Han J. (2012) Data Mining: concepts and techniques, Morgan Kaufman, Waltman, MA.

Hellesoy, A., Hoover, D. (2006). Graph JavaScript framework, version 0.0.1 - JavaScript – Social Snippet Repository. Retrieved on 20.07.2012 from http://snipplr.com/view/1950/graph-javascriptframework-version-001/

Hill, M. O. & Gauch, H. G. (1980). Detrended Correspondence Analysis: An Improved Ordination Technique. *Vegetatio* 42, 47–58.

Hill, M. O. (1973). Reciprocal averaging: An eigenvector method of ordination. Journal of Ecology, 61, 237-49.

Hill, M. O. (1979). *DECORANA — A FORTRAN program for Detrended Correspondence Analysis and Reciprocal Averaging*. Section of Ecology and Systematics, Cornell University, Ithaca, New York.

Hilty, L. M. (2010a). CANDis-Moderationsalgorithmus. Unpublished document.

Hilty, L. M. (2010b). Projekt CANDis – Computer-Aided Normative Discourse. Unpublished document.

Hilty, L. M. (2011a). UZH - Informatics and Sustainability Research - Decision Support and Collective Intelligence. Retrieved on 26.07.2012 from http://www.ifi.uzh.ch/isr/research/dsaci.html

Hilty, L. M. (2011b). Spezifikation des Reportgenerators. Unpublished document.

Hirschfeld, H. O. (1935). A connection between correlation and contingency. *Proceedings of the Cambridge Philosophical Society*. 31, 520-4.

Hubert, L. & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2, 193–218.

Hyrenbach, D. (2011). Multivariate Applications in Marine Science. Lecture in Advanced Biometry. Hawaii Pacific University. Retrieved on 21.05.2012 from http://www.pelagicos.net/classes_advanced_biometry_sp11.htm

Ito, T. & Shintani. T. (1997). Persuasion among agents: An approach to implementing a group decision support system based on multi-agent negotiation. *In Proceedings of the 5th International Joint Conference on Artificial Intelligence (IJCAI'97).*

Janssen, T. & Sage, A. P. (1996). Group decision support using Toulmin argument structures. *IEEE International Conference on Systems, Man, and Cybernetics*, 4, 2704-2709.

Jockers, M. L. (2008). Executing R in Php. Retrieved on 09.04.2012 from http://www.matthewjockers.net/2008/11/11/executing-r-in-php/

Johnson, D. W., Johnson, R. T. (2000). Civil Political Discourse in a Democracy: The Contribution Of Psychology. *Journal of Peace Psychology*, Vol 6(4), 291-317.

Kellis, M. (2008). Clustering basics, gene expression, sequence clustering. Lecture in Computational Biology: Genomes, Networks, Evolution. Massachusetts Institute of Technology. Retrieved on 05.04.2012 from http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-047-computational-biology-genomes-networks-evolution-fall-2008/

Klein, M. (2007). The MIT Collaboratorium: Enabling Effective Large-Scale Deliberation for Complex Problems. MIT Sloan Research Paper No. 4679-08.

Krause, E. F. (1987). *Taxicab Geometry: An Adventure in Non-Euclidean Geometry*. Courier Dover Publications.

Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika, 29, 1-27, 115-129.

Kruskal, J. B. and Wish, M. (1978). *Multidimensional Scaling*. Number 07-011 in Sage University Paper Series on Quantitative Applications in the Social Sciences. Sage Publications, Newbury Park.

Kümin, J. A. (2010). *Prototyp einer Webapplikation zur Unterstützung der Strukturierung von Debatten*. Bachelor Thesis, Faculty of Economics, University of Zurich

Lance, G. N. & Williams, W. T. (1967). Mixed-Data Classificatory Programs I - Agglomerative Systems. *Aust. Comput. J.*, 1(1):15–20.

Lindzey, S. (2005). Concepts of Correspondence Analysis, Retrieved on 06.07.2012 from http://userwww.sfsu.edu/~efc/classes/biol710/ordination/CA.htm

Liu, B. (2010). “Opinion Mining and Sentiment Analysis: NLP Meets Social Sciences”. Talk given at the *Invited Workshop on Social Theory and Social Computing*, Honolulu, Hawaii. Retrieved 05/04/2012 from http://www.cs.uic.edu/~liub/FBS/Liu-Opinion-Mining-STSC.ppt

Manning, C. D., Raghavan, P. & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.

Marinescu R., Dechter R. (2004). AND/OR tree search for constraint optimization. *In CP’04 – Workshop on Pref. and Soft Constraints*, 2004.

McCarthy, B. C. (2012). Correspondence Analysis and Canonical Analysis. Lecture in Biostatistics-II, Multivariate Methods. Ohio University. Retrieved on 06.07.2012 from http://www.ohio.edu/plantbio/staff/mccarthy/multivariate/multivariate.htm

Merriam-Webster (2012). “Discourse”. Retrieved 05/04/2012 from http://www.merriam-webster.com/dictionary/discourse

Moore, J. J., Fitzsimons, P., Lambe E. and White J. (1970). A Comparison and Evaluation of Some Phytosociological Techniques. *Vegetatio,* 20, 1-20.

Ooms, J. (2010). Jeroen Ooms' personal homepage. Retrieved on 09.04.2012 from http://www.stat.ucla.edu/~jeroen/

Ooms, J. (2011). OpenCPU: producing and reproducing results. Retrieved on 09.04.2012 from https://public.opencpu.org/pages/

Orlóci, L. & N. C. Kenkel, (1985). Introduction to data analysis. International Co-operative Publishing House, Fairland MD.

Orlóci, L. (1978). Multivariate analysis in vegetation research. Junk, Den Haag

Palmer, M. W. (2000). Ordination Methods - an overview. Retrieved on 12.03.2012 from http://ordination.okstate.edu/overview.htm

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine, Sixth Series*, 2, 559-72.

Philosophical Dictionary (2009). Groupthinking. Last edited: 12 December 2011. Retrieved on 20.04.2012 from http://www.xs4all.nl/~maartens/philosophy/Dictionary/G/Groupthinking.htm

Pielou, E. C. (1984). The Interpretation of Ecological Data: A Primer on Classification and Ordination. Wiley, New York

Plait, P. C. (2002). Bad Astronomy: Misconceptions and Misuses Revealed, from Astrology to the Moon Landing "Hoax". ISBN: 978-0-471-40976-2

Pontillo, A. & Mineo, A. (2005). R-php. Retrieved on 09.04.2012 from http://dssm.unipa.it/R-php/

Popper, K. (1959). The Logic of Scientific Discovery. Basic Books, New York, NY

R Foundation (2012). The R Project for Statistical Computing. Retrieved on 09.04.2012 from http://www.r-project.org/index.html

RStudio, Inc. (2012). RStudio. Retrieved on 09.04.2012 from http://rstudio.org/

Schenker, A., Bunke, H., Last, M. & Kandel, A. (2005) Graph-Theoretic Techniques for Web Content Mining, World Scientific Publishing, Singapore.

Smolkowski, K. (2012). The Esxperimental Unit. Retrieved on 06.08.2012 from http://www.ori.org/~keiths/Files/Tips/Stats_Unit.html

Steinhausen, D. & Langer, K. (1977). Clusteranalyse. Einführung in Methoden und Verfahrender automatischen Klassifikation. Walter de Gruyter, Berlin.

Strathausen (2010). A graph layouting and drawing framework for the browser. Retrieved on 20.07.2012 from http://blog.ameisenbar.de/en/2010/03/02/dracula/ and http://graphdracula.net

Suzuki, R. & Shimodaira, H. (2011). pvclust - Shimodaira Lab. Retrieved on 27.07.2012 from http://en.shimolab.com/pvclust

The jQuery Foundation (2012). jQuery: The Write Less, Do More, JavaScript Library. Retrieved on 20.07.2012 from http://jquery.com/

The PHP Group (2012). Math :: PEAR Packages. Retrieved on 09.04.2012 from http://pear.php.net/packages.php?catpid=15&catname=Math

Toulmin, S. (1958). *The Uses of Argument*, Cambridge University Press, London.

Viralheat Inc. (2012). Viralheat | Social Media Simplified. Retrieved on 01.04.2012 from https://viralheat.com/

Whittaker, R. H. (1956). Vegetation of the Great Smoky Mountains. Ecol. Monogr. 26: 1-80.

Williams, W. T. & Gillard, P. (1971). Pattern Analysis of a Grazing Experiment. *Australian Journal of Agricultural Research*, 22: 245-60.

Gee, J. P. (2005). An Introduction to Discourse Analysis: Theory and Method. London: Routledge.

Pang, B. & Lee, L. (2008). Subjectivity Detection and Opinion Identification. *Opinion Mining and Sentiment Analysis*. Now Publishers Inc.

Tables, figures and formulas

Tables

Table 2.1: Comparison between the Liu's data structure and the TBDis system

Table 5.1: Results of the interpretative analysis with comparison to each MSE

Table 5.2: Results of the interpretative analysis with comparison to each MSE (cont.)

Table 5.3: Outputs of the cluster analysis methods and scores given by the researcher

Table 5.4: Outputs of the cluster analysis methods and scores given by the researcher (cont.)

Figures

Figures 2.16 & 2.17: Galagan, J. (2007). Classification and Clustering of Microarray Data. Lecture in Introduction to Computational Genomics for Infectious Disease Retrieved on 10.05.2012 from http://www.broadinstitute.org/annotation/winter_course_2006/

Figure 2.20: Wikimedia Commons (2011). File:DBSCAN-Illustration.svg. Retrieved on 20.05.2012 from http://en.wikipedia.org/wiki/File:DBSCAN-Illustration.svg

All other figures are either outputs from R or created by the researcher.

Formulas

Formulas created with: Bateman, W. (2004). Online LaTeX Equation Editor - create, integrate and download. Retrieved on 20.03.2012 from http://www.codecogs.com/latex/eqneditor.php

Appendix I: Datasets

Dataset 1

```
;S1;S2;S3
P1;3;2;-1
P2;3;2;-3
P3;2;0;-3
P4;0;0;-1
P5;-3;-3;3
P6;-2;0;1
P7;0;1;1
```

Dataset 2

```
;S1;S2;S3;S4;S5;S6
P1;3;2;-1;0;-3;0
P2;3;2;-3;-3;-1;-3
P3;2;0;-3;-3;-1;-3
P4;0;0;-1;-1;0;0
P5;-3;-3;3;3;3;3
P6;-2;0;1;1;-1;0
P7;0;1;1;0;0;0
```

Dataset 3

```
;S1;S2;S3;S4;S5;S6;S7;S8;S9;S10;S11;S12;S13;S14;S15;S16;S17;S18;S19;S20;S21;S22;S23;S24;
S25;S26;S27
P1;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
P2;3;2;2;-2;2;2;2;2;-3;-3;1;2;3;0;2;2;-2;-2;2;3;2;0;0;2;0;2;-2
P3;-3;2;2;-3;1;0;0;0;-2;3;0;0;0;-2;0;0;2;0;0;0;0;0;0;0;0;0;0
P4;-2;3;0;-3;0;0;0;2;0;2;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
P5;-2;2;0;-2;0;2;0;0;0;2;2;-2;-2;2;0;-2;-2;2;-2;0;2;2;2;0;0;-2;2
P6;-2;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
P7;2;3;0;-2;0;0;0;0;0;0;0;0;2;2;0;0;0;2;0;0;0;0;0;0;0;0;0
P8;-3;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;2;0;0;-2;2;0;0
P9;-3;0;0;3;0;0;0;0;0;0;2;0;0;0;0;0;0;0;0;0;3;0;0;0;0;0
P10;0;0;0;0;0;0;2;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
P11;3;0;0;2;2;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
P12;-2;-2;-2;-2;2;2;0;2;2;2;2;0;-3;2;2;-2;-2;2;0;-2;2;2;0;-2;2;0;0
```

Dataset 4

```
;S1;S2;S3;S4;S5;S6;S7;S8
P1;3;3;-3;3;3;-3;-3;3
P2;-3;-3;3;-3;-3;3;3;-3
P3;3;2;-3;2;2;-1;-3;2
P4;-2;-2;3;-1;-1;3;1;-1
```

```
P5;3;2;-2;2;2;-1;-3;0
P6;-2;-1;0;-1;0;1;2;0
P7;1;2;-1;1;0;-1;0;0
P8;-2;-2;2;-2;0;0;0;0
P9;2;2;-2;0;0;0;0;0
P10;0;0;0;0;0;0;0;0
```

Dataset 5

```
;S1;S2;S3;S4;S5;S6;S7;S8;S9;S10;S11
P1;3;3;3;-3;-3;3;3;3;-3;3;-3
P2;-3;-3;-3;3;3;-3;-3;-3;3;-3;3
P3;2;2;2;-2;-2;2;2;2;-2;2;-3
P4;-1;-1;-1;1;1;-1;-1;-1;1;-1;1
P5;0;0;0;0;0;0;0;0;0;0;0
```

Dataset 6

```
;S1;S2;S3;S4;S5;S6;S7
P1;3;3;-3;-3;-3;-3;-3
P2;-3;-3;3;3;3;3;3
P3;3;1;-3;0;0;-3;-3
P4;-2;-2;2;2;1;3;3
P5;2;3;-3;-3;-2;0;0
P6;-3;-3;2;3;3;3;3
P7;2;2;-2;0;0;-3;-3
P8;-1;-2;2;0;0;0;0
P9;1;1;-1;3;3;0;0
P10;-1;-1;1;0;0;0;0
P11;0;0;0;0;0;0;0
```

Dataset 7

```
;S1;S2;S3;S4;S5;S6;S7;S8;S9;S10;S11;S12
P1;3;3;-3;-3;-3;-3;-3;3;3;-3;-3;3
P2;-3;-3;3;3;3;3;3;-3;-3;3;3;-3
P3;3;1;-3;0;0;-3;-3;0;0;0;0;0
P4;-2;-2;2;2;1;3;3;-3;-2;2;2;0
P5;2;3;-3;-3;-2;0;0;2;0;0;0;0
P6;-3;-3;2;3;3;3;3;-3;-3;3;3;-3
P7;2;2;-2;0;0;-3;-3;2;1;0;0;0
P8;-1;-2;2;0;0;0;0;-2;-1;0;0;0
P9;1;1;-1;3;3;0;0;1;1;0;0;0
P10;-1;-1;1;0;0;0;0;-1;-1;0;0;0
P11;3;3;3;3;3;3;3;3;3;3;3;3
P12;-3;-2;3;2;0;2;0;-1;-1;0;0;0
P13;0;0;0;0;0;0;0;0;0;0;0;0
```

Dataset 8

```
;S1;S2;S3;S4;S5;S6;S7;S8;S9
P1;3;3;3;3;-3;3;3;-3;-3
P2;-3;-3;-3;-3;3;-3;-3;3;3
P3;3;2;1;1;-3;2;1;-1;-2
P4;-3;-2;-2;-2;3;-2;-1;3;2
P5;3;2;2;2;-3;2;1;-2;-1
P6;-3;-2;-1;-1;3;-2;-1;3;2
P7;2;3;2;2;-3;1;1;-2;-1
P8;-2;-2;-2;0;3;-2;-3;0;3
P9;2;2;1;0;-2;1;0;0;0
P10;-1;-1;0;0;2;-1;0;0;0
P11;1;1;0;0;-2;0;0;0;0
P12;0;0;0;0;0;0;0;0;0
```

Dataset 9

```
;S1;S2;S3;S4;S5;S6;S7;S8;S9
P1;3;3;-3;-3;-3;-3;-3;-3;3
P2;-3;-3;3;3;3;3;3;3;-3
P3;3;3;-3;-2;-2;-2;-2;-1;0
P4;-3;-2;3;1;3;2;2;0;0
P5;2;2;-2;-1;-1;-1;0;-1;0
P6;-2;-2;2;0;2;1;0;0;0
P7;1;1;-1;0;0;-1;0;0;0
P8;0;0;0;0;0;0;0;0;0
```

Dataset 10

```
;S1;S2;S3;S4;S5;S6;S7;S8;S9;S10;S11;S12;S13;S14;S15
P1;3;3;3;3;-3;-3;-3;-3;-3;-3;-3;-3;3;3;3
P2;-3;-3;-3;-3;3;3;3;3;3;3;3;3;-3;-3;-3
P3;3;3;3;2;-3;-2;-2;-2;-2;-2;-2;-1;0;0;0
P4;-3;-2;-2;-2;3;1;1;3;0;2;2;0;0;0;0
P5;2;2;2;2;-2;-1;-1;-1;0;-1;0;-1;0;0;0
P6;-2;-2;-2;-1;2;0;0;2;0;1;0;0;0;0;0
P7;1;1;1;1;-1;0;0;0;0;-1;0;0;0;0;0
P8;-3;-3;-3;-3;3;3;2;3;3;3;2;3;-1;-2;-2
P9;3;2;3;2;-3;-2;-2;-3;-3;-3;-2;-3;1;2;2
P10;-3;-3;-3;-3;2;2;2;2;3;3;2;3;-1;-1;-1
P11;3;2;2;2;-2;-1;-1;-2;-3;-2;-2;-3;1;1;1
P12;-2;-1;-2;-1;2;2;1;2;2;2;1;2;-1;0;0
P13;2;2;3;0;-2;-2;0;-2;-2;-2;-1;-2;1;0;0
P14;-2;-2;-2;0;2;1;0;1;1;1;0;1;0;0;0
P15;1;1;1;0;-2;0;0;-1;-1;-1;0;-1;0;0;0
P16;-1;-1;-1;0;1;0;0;1;1;1;0;0;0;0;0
P17;1;1;1;0;-1;0;0;0;0;0;0;0;0;0;0
P18;0;0;0;0;0;0;0;0;0;0;0;0;0;0;0
```

PHP Versions of the datasets for using with the TBDis fraction analysis algorithm:

Dataset 1

```
$matrix = array(
array(3,2,-1),
array(3,2,-3),
array(2,0,-3),
array(0,0,-1),
array(-3,-3,3),
array(-2,0,1),
array(0,1,1)
);

$arrayOfNodeIDLevels = array();//information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
```

Dataset2

```
$matrix = array(
array(3,2,-1,0,-3,0),
array(3,2,-3,-3,-1,-3),
array(2,0,-3,-3,-1,-3),
array(0,0,-1,-1,0,0),
array(-3,-3,3,3,3,3),
array(-2,0,1,1,-1,0),
array(0,1,1,0,0,0)
);

$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 4;
$arrayOfNodeIDLevels[4] = 4;
$arrayOfNodeIDLevels[5] = 4;
```

Dataset 3

```
$matrix = array(
array(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
array(3,2,2,-2,2,2,2,2,-3,-3,1,2,3,0,2,2,-2,-2,2,3,2,0,0,2,0,2,-2),
array(-3,2,2,-3,1,0,0,0,-2,3,0,0,0,-2,0,0,2,0,0,0,0,0,0,0,0,0,0),
array(-2,3,0,-3,0,0,0,2,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
array(-2,2,0,-2,0,2,0,0,0,2,2,-2,-2,2,0,-2,-2,2,-2,0,2,2,2,0,0,-2,2),
array(-2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
array(2,3,0,-2,0,0,0,0,0,0,0,0,2,2,0,0,0,2,0,0,0,0,0,0,0,0,0),
array(-3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,-2,2,0,0),
array(-3,0,0,3,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0,3,0,0,0,0,0),
array(0,0,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
array(3,0,0,2,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0),
array(-2,-2,-2,-2,2,2,0,2,2,2,2,0,-3,2,2,-2,-2,2,0,-2,2,2,0,-2,2,0,0)
);

$arrayOfNodeIDLevels = array(); //information about the structure;
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 2;
$arrayOfNodeIDLevels[4] = 2;
$arrayOfNodeIDLevels[5] = 2;
$arrayOfNodeIDLevels[6] = 2;
$arrayOfNodeIDLevels[7] = 2;
$arrayOfNodeIDLevels[8] = 2;
$arrayOfNodeIDLevels[9] = 2;
$arrayOfNodeIDLevels[10] = 2;
$arrayOfNodeIDLevels[11] = 4;
$arrayOfNodeIDLevels[12] = 4;
$arrayOfNodeIDLevels[13] = 4;
$arrayOfNodeIDLevels[14] = 4;
$arrayOfNodeIDLevels[15] = 4;
$arrayOfNodeIDLevels[16] = 4;
$arrayOfNodeIDLevels[17] = 4;
$arrayOfNodeIDLevels[18] = 4;
$arrayOfNodeIDLevels[19] = 6;
$arrayOfNodeIDLevels[20] = 6;
$arrayOfNodeIDLevels[21] = 6;
$arrayOfNodeIDLevels[22] = 6;
$arrayOfNodeIDLevels[23] = 8;
$arrayOfNodeIDLevels[24] = 8;
$arrayOfNodeIDLevels[25] = 8;
$arrayOfNodeIDLevels[26] = 10;
```

Dataset 4

```
$matrix = array(
array(3,3,-3,3,3,-3,-3,3),
array(-3,-3,3,-3,-3,3,3,-3),
array(3,2,-3,2,2,-1,-3,2),
array(-2,-2,3,-1,-1,3,1,-1),
array(3,2,-2,2,2,-1,-3,0),
array(-2,-1,0,-1,0,1,2,0),
array(1,2,-1,1,0,-1,0,0),
array(-2,-2,2,-2,0,0,0,0),
array(2,2,-2,0,0,0,0,0),
array(0,0,0,0,0,0,0,0)
);
```

```
$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 4;
$arrayOfNodeIDLevels[4] = 4;
$arrayOfNodeIDLevels[5] = 4;
$arrayOfNodeIDLevels[6] = 6;
$arrayOfNodeIDLevels[7] = 8;
```

Dataset 5

```
$matrix = array(
array(3,3,-3,3,3,-3,-3,3),
array(-3,-3,3,-3,-3,3,3,-3),
array(3,2,-3,2,2,-1,-3,2),
array(-2,-2,3,-1,-1,3,1,-1),
array(3,2,-2,2,2,-1,-3,0),
array(-2,-1,0,-1,0,1,2,0),
array(1,2,-1,1,0,-1,0,0),
array(-2,-2,2,-2,0,0,0,0),
array(2,2,-2,0,0,0,0,0),
array(0,0,0,0,0,0,0,0)
);

$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 4;
$arrayOfNodeIDLevels[4] = 4;
$arrayOfNodeIDLevels[5] = 4;
$arrayOfNodeIDLevels[6] = 6;
$arrayOfNodeIDLevels[7] = 8;
```

Dataset 6

```
$matrix = array(
array(3,3,-3,3,3,-3,-3,3),
array(-3,-3,3,-3,-3,3,3,-3),
array(3,2,-3,2,2,-1,-3,2),
array(-2,-2,3,-1,-1,3,1,-1),
array(3,2,-2,2,2,-1,-3,0),
array(-2,-1,0,-1,0,1,2,0),
array(1,2,-1,1,0,-1,0,0),
array(-2,-2,2,-2,0,0,0,0),
array(2,2,-2,0,0,0,0,0),
array(0,0,0,0,0,0,0,0)
);
```

```
$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 4;
$arrayOfNodeIDLevels[4] = 4;
$arrayOfNodeIDLevels[5] = 4;
$arrayOfNodeIDLevels[6] = 6;
$arrayOfNodeIDLevels[7] = 8;
```

Dataset 7

```
$matrix = array(
array(3,3,-3,-3,-3,-3,-3,3,3,-3,-3,3),
array(-3,-3,3,3,3,3,3,-3,-3,3,3,-3),
array(3,1,-3,0,0,-3,-3,0,0,0,0,0),
array(-2,-2,2,2,1,3,3,-3,-2,2,2,0),
array(2,3,-3,-3,-2,0,0,2,0,0,0,0),
array(-3,-3,2,3,3,3,3,-3,-3,3,3,-3),
array(2,2,-2,0,0,-3,-3,2,1,0,0,0),
array(-1,-2,2,0,0,0,0,-2,-1,0,0,0),
array(1,1,-1,3,3,0,0,1,1,0,0,0),
array(-1,-1,1,0,0,0,0,-1,-1,0,0,0),
array(3,3,3,3,3,3,3,3,3,3,3,3),
array(-3,-2,3,2,0,2,0,-1,-1,0,0,0),
array(0,0,0,0,0,0,0,0,0,0,0,0)
);

$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 4;
$arrayOfNodeIDLevels[4] = 4;
$arrayOfNodeIDLevels[5] = 4;
$arrayOfNodeIDLevels[6] = 4;
$arrayOfNodeIDLevels[7] = 6;
$arrayOfNodeIDLevels[8] = 6;
$arrayOfNodeIDLevels[9] = 8;
$arrayOfNodeIDLevels[10] = 8;
$arrayOfNodeIDLevels[11] = 10;
```

Dataset 8

```
$matrix = array(
array(3,3,-3,-3,-3,-3,-3,3,3,-3,-3,3),
array(-3,-3,3,3,3,3,3,-3,-3,3,3,-3),
array(3,1,-3,0,0,-3,-3,0,0,0,0,0),
array(-2,-2,2,2,1,3,3,-3,-2,2,2,0),
array(2,3,-3,-3,-2,0,0,2,0,0,0,0),
array(-3,-3,2,3,3,3,3,-3,-3,3,3,-3),
array(2,2,-2,0,0,-3,-3,2,1,0,0,0),
array(-1,-2,2,0,0,0,0,-2,-1,0,0,0),
array(1,1,-1,3,3,0,0,1,1,0,0,0),
array(-1,-1,1,0,0,0,0,-1,-1,0,0,0),
array(3,3,3,3,3,3,3,3,3,3,3,3),
array(-3,-2,3,2,0,2,0,-1,-1,0,0,0),
array(0,0,0,0,0,0,0,0,0,0,0,0)
);

$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 4;
$arrayOfNodeIDLevels[4] = 4;
$arrayOfNodeIDLevels[5] = 4;
$arrayOfNodeIDLevels[6] = 4;
$arrayOfNodeIDLevels[7] = 6;
$arrayOfNodeIDLevels[8] = 6;
$arrayOfNodeIDLevels[9] = 8;
$arrayOfNodeIDLevels[10] = 8;
$arrayOfNodeIDLevels[11] = 10;
```

Dataset 9

```
$matrix = array(
array(3,3,-3,-3,-3,-3,-3,3,3,-3,-3,3),
array(-3,-3,3,3,3,3,3,-3,-3,3,3,-3),
array(3,1,-3,0,0,-3,-3,0,0,0,0,0),
array(-2,-2,2,2,1,3,3,-3,-2,2,2,0),
array(2,3,-3,-3,-2,0,0,2,0,0,0,0),
array(-3,-3,2,3,3,3,3,-3,-3,3,3,-3),
array(2,2,-2,0,0,-3,-3,2,1,0,0,0),
array(-1,-2,2,0,0,0,0,-2,-1,0,0,0),
array(1,1,-1,3,3,0,0,1,1,0,0,0),
array(-1,-1,1,0,0,0,0,-1,-1,0,0,0),
array(3,3,3,3,3,3,3,3,3,3,3,3),
array(-3,-2,3,2,0,2,0,-1,-1,0,0,0),
array(0,0,0,0,0,0,0,0,0,0,0,0)
);

$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 4;
$arrayOfNodeIDLevels[4] = 4;
$arrayOfNodeIDLevels[5] = 4;
$arrayOfNodeIDLevels[6] = 4;
$arrayOfNodeIDLevels[7] = 6;
```

```
$arrayOfNodeIDLevels[8] = 6;
$arrayOfNodeIDLevels[9] = 8;
$arrayOfNodeIDLevels[10] = 8;
$arrayOfNodeIDLevels[11] = 10;
```

Dataset 10

```
$matrix = array(
array(3,3,-3,-3,-3,-3,-3,3,3,-3,-3,3),
array(-3,-3,3,3,3,3,3,-3,-3,3,3,-3),
array(3,1,-3,0,0,-3,-3,0,0,0,0,0),
array(-2,-2,2,2,1,3,3,-3,-2,2,2,0),
array(2,3,-3,-3,-2,0,0,2,0,0,0,0),
array(-3,-3,2,3,3,3,3,-3,-3,3,3,-3),
array(2,2,-2,0,0,-3,-3,2,1,0,0,0),
array(-1,-2,2,0,0,0,0,-2,-1,0,0,0),
array(1,1,-1,3,3,0,0,1,1,0,0,0),
array(-1,-1,1,0,0,0,0,-1,-1,0,0,0),
array(3,3,3,3,3,3,3,3,3,3,3,3),
array(-3,-2,3,2,0,2,0,-1,-1,0,0,0),
array(0,0,0,0,0,0,0,0,0,0,0,0)
);

$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 4;
$arrayOfNodeIDLevels[4] = 4;
$arrayOfNodeIDLevels[5] = 4;
$arrayOfNodeIDLevels[6] = 4;
$arrayOfNodeIDLevels[7] = 6;
$arrayOfNodeIDLevels[8] = 6;
$arrayOfNodeIDLevels[9] = 8;
$arrayOfNodeIDLevels[10] = 8;
$arrayOfNodeIDLevels[11] = 10;
```

Appendix II: Codes for multivariate methods

Table of contents

```
# These codes were used for the figures in chapter two
```

Example data

```
# example data

mydata = c(3,2,-1,0,-3,0,3,2,-3,-3,-1,-3,2,0,-3,-3,-1,-3,0,0,-1,-1,0,0,-3,-3,3,3,3,3,-2,0,1,1,-
1,0,0,1,1,0,0,0)
dim(mydata) = c(6,7)
mydata = t(mydata) #transpose
rownames(mydata) <- c("P1","P2","P3","P4","P5","P6","P7")
colnames(mydata) = c("S1","S2","S3","S4","S5","S6")

mydata =
data.frame(S1=mydata[,1],S2=mydata[,2],S3=mydata[,3],S4=mydata[,4],S5=mydata[,5],S6=mydat
a[,6])
attach(mydata)
```

Codes for methods for dimensionality reduction

```
# weightes averages algorithm

weights = c()
Oscores = c()
for (i in 1:ncol(mydata)) { weights[i] = nrow(mydata)-length(which(mydata[,i]==0)) }
for (i in 1:nrow(mydata)) { Oscores[i] = sum(mydata[i,]*weights)/sum(mydata[i,]) }
print(Oscores)
stripchart(Oscores,pch="")
text(Oscores,1,rownames(mydata))
```

```
# Polar ordination algorithm

library(asbio)
po.orig <-polar.ord(mydata+3,index="euclidean",endpoint="var.reg")
plot(po.orig$Scores[,1],po.orig$Scores[,2], type="n", asp=1)
text(po.orig$Scores[,1],po.orig$Scores[,2], rownames(mydata))



# PCA algorithm

f <- prcomp(mydata) #also: prcomp/princomp
xf <- f$x[,1]
yf <- f$x[,2]
plot(xf,yf, type="n", asp=1)
text(xf,yf,labels=rownames(mydata))


# RA algorithm

library(vegan)
vare.ra <- decorana(mydata+4, ira=1)
plot(vare.ra$rproj[,1],vare.ra$rproj[,2], type="n", asp=1)
text(vare.ra$rproj[,1],vare.ra$rproj[,2], rownames(mydata))


# DCA algorithm

library(vegan)
vare.dca <- decorana(mydata+4, ira=0)
plot(vare.dca$rproj[,1],vare.dca$rproj[,2], type="n", asp=1)
text(vare.dca$rproj[,1],vare.dca$rproj[,2], rownames(mydata))



#MVR algorithm

library(pls)
fit = mvr(S1~S2+S3+S4+S5+S6)
plot(fit$scores, type="n", asp=1)
text(fit$scores, rownames(mydata))

#more general way:
#listoffactors = c() # dynamically building the formula by string manipulation
#for (m in 2:ncol(mydata)) listoffactors =
append(listoffactors,paste(paste("mydata[,'",colnames(mydata)[m],sep=""),"']",sep=""))
#fit = mvr(as.formula(paste("mydata[,'S1']~",paste(listoffactors,collapse="+"),sep="")))


# NMDS algorithm

par(mfrow=c(2,2))
```

```
dis <- dist(mydata)

m <- monoMDS(dis, k=2, model = "global")
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))

m <- monoMDS(dis, k=2, model = "global")
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))

m <- monoMDS(dis, k=2, model = "global")
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))

m <- monoMDS(dis, k=2, model = "global")
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))

# Dimension clustering algorithm (k-means)

km <- kmeans(t(mydata),2)
plot(km$centers[1,],km$centers[2,], type="n", asp=1)
text(km$centers[1,],km$centers[2,], rownames(mydata))
```

Codes for methods for cluster analysis

```
# Hierarchical clustering algorithm

library(pvclust)
fit <- pvclust(t(mydata), method.hclust="complete", method.dist="euclidean")

par(mfrow=c(2,2))
plot(fit) # dendogram with p values
# add rectangles around groups highly supported by the data
pvrect(fit, alpha=.95)


recursiveGetParticipant <- function(paramVector){
        returnVector = c()
        for (i in 1:length(paramVector)) {
                if (paramVector[i] < 0) returnVector = append(returnVector,abs(paramVector[i]))
                else returnVector = 
append(returnVector,recursiveGetParticipant(fit$hclust$merge[paramVector[i],]))
        }
        return(returnVector)
}

clusters2 = c()
for (i in 1:nrow(mydata)) clusters2[rownames(mydata)[i]] = nrow(mydata)+i
```

```
clusters = list()
for (i in 1:length(fit$msfit)) {
	if ((fit$msfit[[i]]$p["au"] > 0.95) && (fit$msfit[[i]]$p["au"] < 1)) {
		newCluster = recursiveGetParticipant(fit$hclust$merge[i,])
		clusters[[length(clusters)+1]] = newCluster
		for (j in 1:length(newCluster)) { clusters2[paste("P",newCluster[j],sep="")] =
length(clusters) }
	}
}
print(clusters)
print(clusters2)
clusplot(mydata, clusters2, color=TRUE, shade=TRUE, labels=3, lines=0, asp=1)


# K-means clustering algorithm

par(mfrow=c(2,2))
k=0
wss=c()

wss <- nrow(mydata)*sum(apply(mydata,2,var))
for (i in 2:(nrow(mydata)-1)) { wss[i] <- sum(kmeans(mydata, centers=i)$withinss) }
plot(1:(nrow(mydata)-1), wss, type="b", xlab="Number of Clusters",  ylab="Within groups sum of
squares")


for (i in 1:(length(wss)-1)) {
	y = c(wss[i],wss[i+1])
	x = c(i,i+1)
	lm = lm(y~x)
	if (lm$coefficients["x"] > -10) {
		k = i
		break;
	}
}
if (k > (nrow(mydata)/2)) k = round(sqrt(nrow(mydata)/2)) # rule of thumb

fit <- kmeans(mydata, k)

print(fit$cluster)

library(cluster)
clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=3, lines=0, asp=1)


# Density-based clustering algorithm

library(fpc)
db = dbscan(dist(mydata), eps=mean(dist(mydata)), MinPts=1)
clusplot(mydata, db$cluster, color=TRUE, shade=TRUE, labels=3, lines=0, asp=1)
```

Appendix III: Codes for analysis

Table of contents

R

Importing data into R

```
#to read data use:
mydata <- read.delim("c:\\some path\\folder\\7.csv", sep=";", header = TRUE, row.names=1)
print(mydata)


#for example:
mydata <- read.delim("C:\\Users\\Tony_2\\Dropbox\\Private\\12. Semester\\Master
Thesis\\chapters\\evaluation data\\7.csv", sep=";", header = TRUE, row.names=1)
```

Quantitative analysis of methods for dimensionality reduction using random data

```
# distance matrix difference

distMatrixDiffMSE = function(preData,postData) {
	preDist = dist(preData)
	postDist = dist(postData)
	postDist = postDist*(max(preDist)/max(postDist)) # normalizing. making both max()
distances be the same
	MSE = mean((preDist-postDist)^2)
	return(MSE)
}

# libraries
library(vegan) # dca, ra
library(asbio)  # polar
library(pls)      # mvr

# analysis
analysis = matrix(ncol=15)
colnames(analysis) =
c("NoParticipants","NoStatements","PCA","DCA","RA","Polar","MVR","NMDS","K-
Means","NMDSPCA","NMDSDCA","NMDSRA","NMDSPolar","NMDSMVR","NMDSK-
Means")
analysis = analysis[-1,] #remove the NA row

for (t in 1:500) {
NoStatements = sample(5:50,1)
NoParticipants = sample(NoStatements:55,1)
mydata = array(sample(-3:3,(NoParticipants*NoStatements), replace=T),
dim=c(NoParticipants,NoStatements))
rownames(mydata) <- paste(rep("P",NoParticipants),1:NoParticipants, sep="")
colnames(mydata) = paste(rep("S",NoStatements),1:NoStatements, sep="")
mydata = data.frame(mydata)
preData = mydata

# PCA
f <- prcomp(preData) #also: prcomp/princomp
postDataPCA = data.frame(cbind(f$x[,1],f$x[,2]))
MSEPCA = distMatrixDiffMSE(preData,postDataPCA)

# DCA
vare.dca <- decorana(preData+4, ira=0)
postDataDCA = data.frame(cbind(vare.dca$rproj[,1],vare.dca$rproj[,2]))
MSEDCA = distMatrixDiffMSE(preData,postDataDCA)

# RA
vare.ra <- decorana(preData+4, ira=1)
postDataRA = data.frame(cbind(vare.ra$rproj[,1],vare.ra$rproj[,2]))
MSERA = distMatrixDiffMSE(preData,postDataRA)

# polar
```

```
bypass = 0
if (bypass == 0) {
po.orig <-polar.ord(preData+3,index="euclidean",endpoint="var.reg")
postDataPO = data.frame(cbind(po.orig$Scores[,1],po.orig$Scores[,2]))
MSEPO = distMatrixDiffMSE(preData,postDataPO)
} else MSEPO = 1

# MVR
listoffactors = c() # dynamically building the formula by string manipulation
for (m in 2:ncol(preData)) listoffactors =
append(listoffactors,paste(paste("preData[,'",colnames(preData)[m],sep=""),"']",sep=""))
fit = mvr(as.formula(paste("preData[,'S1']~",paste(listoffactors,collapse="+"),sep="")))
postDataMVR = data.frame(cbind(fit$scores[,1],fit$scores[,2]))
MSEMVR = distMatrixDiffMSE(preData,postDataMVR)

#k-means dimension clustering
km <- kmeans(t(preData),2)
postDataKM = data.frame(cbind(km$centers[1,],km$centers[2,]))
MSEKM = distMatrixDiffMSE(preData,postDataKM)

# NMDS 20 mal. nicht sagen schwankung ist klein. weil "in relation was?" (in relation zu den MSE
der anderen) sondern zeigen dass NMDS konstant besser ist als alle andere
m <- monoMDS(dist(preData), k=2, model = "global")
postDataNMDS = data.frame(cbind(m$points[,1],m$points[,2]))
MSENMDS = distMatrixDiffMSE(preData,postDataNMDS)

# NMDSPCA
m <- monoMDS(dist(preData), y=cbind(f$x[,1],f$x[,2]), k=2, model = "global")
postDataNMDSPCA = data.frame(cbind(m$points[,1],m$points[,2]))
MSENMDSPCA = distMatrixDiffMSE(preData,postDataNMDSPCA)

# NMDSDCA
m <- monoMDS(dist(preData), y=cbind(vare.dca$rproj[,1],vare.dca$rproj[,2]), k=2, model =
"global")
postDataNMDSDCA = data.frame(cbind(m$points[,1],m$points[,2]))
MSENMDSDCA = distMatrixDiffMSE(preData,postDataNMDSDCA)

# NMDSRA
m <- monoMDS(dist(preData), y=cbind(vare.ra$rproj[,1],vare.ra$rproj[,2]), k=2, model = "global")
postDataNMDSRA = data.frame(cbind(m$points[,1],m$points[,2]))
MSENMDSRA = distMatrixDiffMSE(preData,postDataNMDSRA)

# NMDSPolar
if (bypass == 0) {
m <- monoMDS(dist(preData), y=cbind(po.orig$Scores[,1],po.orig$Scores[,2]), k=2, model =
"global")
postDataNMDSPolar = data.frame(cbind(m$points[,1],m$points[,2]))
MSENMDSPO = distMatrixDiffMSE(preData,postDataNMDSPolar)
} else MSENMDSPO = 1

# NMDSMVR
m <- monoMDS(dist(preData), y=cbind(fit$scores[,1],fit$scores[,2]), k=2, model = "global")
```

```
postDataNMDSMVR = data.frame(cbind(m$points[,1],m$points[,2]))
MSENMDSMVR = distMatrixDiffMSE(preData,postDataNMDSMVR)

# NMDSK-Means
m <- monoMDS(dist(preData), y=cbind(km$centers[1,],km$centers[2,]), k=2, model = "global")
postDataNMDSKM = data.frame(cbind(m$points[,1],m$points[,2]))
MSENMDSKM = distMatrixDiffMSE(preData,postDataNMDSKM)

analysis =
rbind(analysis,c(NoParticipants,NoStatements,MSEPCA,MSEDCA,MSERA,MSEPO,MSEMVR,
MSENMDS,MSEKM,MSENMDSPCA,MSENMDSDCA,MSENMDSRA,MSENMDSPO,MSEN
MDSMVR,MSENMDSKM))
rownames(analysis)[nrow(analysis)] = nrow(analysis)
}
print(analysis)

par(mfrow=c(3,3))
#plot(analysis[,1],rep(1,nrow(analysis)))
#plot(analysis[,2],rep(1,nrow(analysis)))
plot(analysis[,1]*analysis[,2],analysis[,"PCA"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"DCA"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"RA"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"Polar"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"MVR"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"NMDS"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"K-Means"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"NMDSPCA"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"NMDSDCA"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"NMDSRA"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"NMDSPolar"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"NMDSMVR"],asp=1)
plot(analysis[,1]*analysis[,2],analysis[,"NMDSK-Means"],asp=1)

means = matrix(ncol=(ncol(analysis)-2))
colnames(means) = colnames(analysis)[3:ncol(analysis)]
means = means[-1,] #remove the NA row

means =
rbind(means,c(mean(analysis[,"PCA"]),mean(analysis[,"DCA"]),mean(analysis[,"RA"]),mean(anal
ysis[,"Polar"]),mean(analysis[,"MVR"]),mean(analysis[,"NMDS"]),mean(analysis[,"K-
Means"]),mean(analysis[,"NMDSPCA"]),mean(analysis[,"NMDSDCA"]),mean(analysis[,"NMDS
RA"]),mean(analysis[,"NMDSPolar"]),mean(analysis[,"NMDSMVR"]),mean(analysis[,"NMDSK-
Means"])))
orderedMeans = means[,order(means[1,])]
print(orderedMeans)

slopes = matrix(ncol=(ncol(analysis)-2))
colnames(slopes) = colnames(analysis)[3:ncol(analysis)]
slopes = slopes[-1,] #remove the NA row

x = analysis[,1]*analysis[,2]
lmVector= matrix(ncol=(ncol(analysis)-2))
```

```
colnames(lmVector) = colnames(analysis)[3:ncol(analysis)]
lmVector= lmVector[-1,] #remove the NA row
for (i in 3:ncol(analysis)) {
	lmVector[colnames(analysis)[i]] =
lm(as.formula(paste(paste("analysis[,'",colnames(analysis)[i],sep=""),"']~x",sep="")))$coefficients[
"x"]
}
slopes = rbind(slopes,lmVector)
print(slopes[,order(slopes[1,])])
```

```
# to reproduce results:
# 1) load dataset. eg: mydata <- read.delim("C:\\some folder\\Master Thesis\\chapters\\evaluation
data\\7.csv", sep=";", header = TRUE, row.names=1)
# 2) select the lines of the method and hit control+r to run those lines in the console
# 3) if selecting MVR, adapt the formula
# 4) for a visual evaluation of distances, select the lines around "show distance" and hit control+r to
show the distances or write dist(mydata) in the console
# -> you might want to adapt the threshold to (mean(distM)/2) or mean(distM)

distM = as.matrix(dist(mydata))
for(i in 1:ncol(distM))
        for (j in (i+1):nrow(distM))
                if ((j <= nrow(distM)) && (round(distM[i,j],1) < (mean(distM)/2))) {
segments(xf[i],yf[i],xf[j],yf[j],col="blue");
text(((xf[i]+xf[j])/2),((yf[i]+yf[j])/2),labels=c(round(distM[i,j],1)), col="red") }

# MSE score
print(distMatrixDiffMSE(mydata,cbind(xf,yf)))
#par(mfrow=c(2,2))

# PCA
f <- prcomp(mydata) #also: prcomp/princomp
xf <- f$x[,1]; yf <- f$x[,2]
plot(f$x[,1],f$x[,2], type="n", asp=1)
text(f$x[,1],f$x[,2],labels=rownames(mydata))


# DCA
library(vegan)
vare.dca <- decorana(mydata+4, ira=0)
xf = vare.dca$rproj[,1]; yf = vare.dca$rproj[,2]
plot(vare.dca$rproj[,1],vare.dca$rproj[,2], type="n", asp=1)
text(vare.dca$rproj[,1],vare.dca$rproj[,2], rownames(mydata))


# RA
library(vegan)
vare.ra <- decorana(mydata+4, ira=1)
xf = vare.ra$rproj[,1]; yf = vare.ra$rproj[,2]
plot(vare.ra$rproj[,1],vare.ra$rproj[,2], type="n", asp=1)
text(vare.ra$rproj[,1],vare.ra$rproj[,2], rownames(mydata))


# Polar
library(asbio)
po.orig <-polar.ord(mydata+3,index="euclidean",endpoint="var.reg")
xf = po.orig$Scores[,1]; yf = po.orig$Scores[,2]
plot(po.orig$Scores[,1],po.orig$Scores[,2], type="n", asp=1)
text(po.orig$Scores[,1],po.orig$Scores[,2], rownames(mydata))
```

```
# MVR
library(pls)
attach(mydata)
fit = mvr(S1~S2+S3+S4+S5+S6+S7+S8+S9+S10+S11+S12+S13+S14+S15) # <- has to be adapted
xf = fit$scores[,1]; yf = fit$scores[,2]
plot(fit$scores, type="n", asp=1)
text(fit$scores, rownames(mydata))
#S1~S2+S3+S4+S5+S6+S7+S8+S9+S10+S11+S12+S13+S14+S15+S16+S17+S18+S19+S20+S21
+S22+S23+S24+S25+S26+S27


# K-Means
km <- kmeans(t(mydata),2)
xf = km$centers[1,]; yf = km$centers[2,]
plot(km$centers[1,],km$centers[2,], type="n", asp=1)
text(km$centers[1,],km$centers[2,], rownames(mydata))

# PCA+NMDS
f <- prcomp(mydata) #also: prcomp/princomp
m <- monoMDS(dist(mydata), y=cbind(f$x[,1],f$x[,2]), k=2, model = "global")
xf <- m$points[,1]; yf <-m$points[,2]
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))


# DCA+NMDS
library(vegan)
vare.dca <- decorana(mydata+4, ira=0)
m <- monoMDS(dist(mydata), y=cbind(vare.dca$rproj[,1],vare.dca$rproj[,2]), k=2, model =
"global")
xf <- m$points[,1]; yf <-m$points[,2]
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))


# RA+NMDS
library(vegan)
vare.ra <- decorana(mydata+4, ira=1)
m <- monoMDS(dist(mydata), y=cbind(vare.ra$rproj[,1],vare.ra$rproj[,2]), k=2, model = "global")
xf <- m$points[,1]; yf <-m$points[,2]
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))


# Polar+NMDS
library(asbio)
po.orig <-polar.ord(mydata+3,index="euclidean",endpoint="var.reg")
m <- monoMDS(dist(mydata), y=cbind(po.orig$Scores[,1],po.orig$Scores[,2]), k=2, model =
"global")
xf <- m$points[,1]; yf <-m$points[,2]
```

```
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))


# MVR+NMDS
library(pls)
attach(mydata)
fit = mvr(S1~S2+S3+S4+S5+S6+S7+S8+S9+S10+S11+S12+S13+S14+S15) # <- has to be adapted
m <- monoMDS(dist(mydata), y=cbind(fit$scores[,1],fit$scores[,2]), k=2, model = "global")
xf <- m$points[,1]; yf <-m$points[,2]
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))
#S1~S2+S3+S4+S5+S6+S7+S8+S9+S10+S11+S12+S13+S14+S15+S16+S17+S18+S19+S20+S21
+S22+S23+S24+S25+S26+S27


# K-Means+NMDS
km <- kmeans(t(mydata),2)
m <- monoMDS(dist(mydata), y=cbind(km$centers[1,],km$centers[2,]), k=2, model = "global")
xf <- m$points[,1]; yf <-m$points[,2]
plot(m$points[,1],m$points[,2], type="n", asp=1)
text(m$points[,1],m$points[,2], rownames(mydata))
```

Interpretative analysis of methods for cluster analysis using constructed data

```
#Hierarchical clustering

library(pvclust)
fit <- pvclust(t(mydata), method.hclust="complete", method.dist="euclidean")

par(mfrow=c(2,2))
plot(fit) # dendogram with p values
# add rectangles around groups highly supported by the data
pvrect(fit, alpha=.95)


recursiveGetParticipant <- function(paramVector){
	returnVector = c()
	for (i in 1:length(paramVector)) {
		if (paramVector[i] < 0) returnVector = append(returnVector,abs(paramVector[i]))
		else                                    returnVector                                    =
append(returnVector,recursiveGetParticipant(fit$hclust$merge[paramVector[i],]))
	}
	return(returnVector)
}

clusters2 = c()
for (i in 1:nrow(mydata)) clusters2[rownames(mydata)[i]] = nrow(mydata)+i
clusters = list()
for (i in 1:length(fit$msfit)) {
	if ((fit$msfit[[i]]$p["au"] > 0.95) && (fit$msfit[[i]]$p["au"] < 1)) {
		newCluster = recursiveGetParticipant(fit$hclust$merge[i,])
		clusters[[length(clusters)+1]] = newCluster
		for   (j   in   1:length(newCluster))   {   clusters2[paste("P",newCluster[j],sep="")]   =
length(clusters) }
	}
}
hcClustering = clusters2

print(clusters)
print(clusters2)
clusplot(mydata, clusters2, color=TRUE, shade=TRUE, labels=3, lines=0, asp=1)


# density based clustering

library(fpc)
db = dbscan(dist(mydata), eps=mean(dist(mydata)), MinPts=1)
print(db$cluster)
clusplot(mydata, db$cluster, color=TRUE, shade=TRUE, labels=3, lines=0, asp=1)
dbClustering = db$cluster

# k-means clustering

#algorithm
```

```
#par(mfrow=c(2,2))
k=0
wss=c()

wss <- nrow(mydata)*sum(apply(mydata,2,var))
for (i in 2:(nrow(mydata)-1)) { wss[i] <- sum(kmeans(mydata, centers=i)$withinss) }
#plot(1:(nrow(mydata)-1), wss, type="b", xlab="Number of Clusters",  ylab="Within groups sum of
squares")


for (i in 1:(length(wss)-1)) {
        y = c(wss[i],wss[i+1])
        x = c(i,i+1)
        lm = lm(y~x)
        if (lm$coefficients["x"] > -10) {
                k = i
                break;
        }
}
if (k > (nrow(mydata)/2)) k = round(sqrt(nrow(mydata)/2)) # rule of thumb

fit <- kmeans(mydata, k)

print(fit$cluster)

library(cluster)
clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=3, lines=0, asp=1)

kmClustering = fit$cluster

print(as.integer(hcClustering))
print(dbClustering)
print(as.integer(kmClustering))
```

PHP

Code for the fraction analysis of the TBDis system

The code has been adapted in such a way that hardcoded data can be used. The original code is in comments.

The datasets have to be pasted where the matrix is defined by using ‘$matrix = ’.

```
<?php
// require_once("mysql_connect.php");
require_once("functions.php");

/* FRAKTIONSANALYSE */

// function fraktionsanalyse ($treeID) {
/*	$con = $GLOBALS['con'];
	$matrix = array();
	$nodeLevel = array();
	$arrayOfParticipantIDs = getInfo_debateParticipants($treeID,'ArrayOfIDs');

	$query = "select *, t.nodeID as tNodeID, o.userID as opinionUserID, c.content from tree t
join content c on t.nodeid = c.nodeid left join opinion o on o.nodeid = t.nodeid where t.treeid =
".$treeID." order by t.nodeID ";
	$result = mysql_queryWithLog($query,$con,'getting whole tree to put in objects for report
generator');
	$processedNodes = array();
	while($row = mysql_fetch_array($result)) {
		if ($row['isDeleted'] == 0) {
			if (!in_array($row['tNodeID'],$processedNodes)) {
				$row['nodeID'] = $row['tNodeID']; //to make sure that $nodeObj-
>nodeID is the id from the node, not from the opinion (which does not exist for arguments)

				$thisNode =  new node($row);

				$parentObj = node::getInstanceFirst(array('nodeID' =>
$row['parent']));

				if (!is_object($parentObj)) $thisNode->level = 0;
				else $thisNode->level = $parentObj->level + 1;

				$nodeLevel[$thisNode->level][] = $thisNode->nodeID;

				$thisNode->opinion = array();
				$thisNode->opinionUserID = null;
				$thisNode->certainty = null;
				$thisNode->opinionStatistics = null;
				$thisNode->visited = null;
```

```
                                        $children = $parentObj->children; $children[] = $thisNode;
$parentObj->children = $children;

                                        $processedNodes[] = $row['tNodeID'];
                              }
                              if (!is_object($thisNode)) $thisNode = node::getInstanceFirst(array('nodeID'
=> $row['nodeID']));

                              $thisNode->opinion[$row['opinionUserID']] = array('opinion' =>
$row['opinion'], 'certainty' => $row['certainty']);
                    }
          }


          // $rootNodeInfo = getInfo_debateRootNode_withContent($treeID);
          $rootNodeObj = node::getInstanceFirst(array('nodeID' => $rootNodeInfo['nodeID']));
          $arrayOfNodeIDs = parseLevelOrder($rootNodeObj);
          for ($i = 0; $i < count($arrayOfNodeIDs); $i++) { //go trough columns
                    //if (!is_array($matrix[])) $matrix[] = array();
                    $nodeObj = node::getInstanceFirst(array('nodeID' => $arrayOfNodeIDs[$i]));
                    for ($j = 0; $j < count($arrayOfParticipantIDs); $j++) { //go through lines
                              $opinion = $nodeObj->opinion[$arrayOfParticipantIDs[$j]]['opinion'];
                              if ($opinion == "trifftzu") $opinion = 1;
                              elseif ($opinion == "trifftnichtzu") $opinion = -1;
                              else $opinion = 0;
                              $matrix[$j][$i] = $opinion;
                    }
          }
          $originalMatrix = $matrix;

          $arrayOfNodeIDLevels = array();
          foreach ($arrayOfNodeIDs as $nodeID) { //put all the levels for each node in another array
with the same index as the node itself
                    $nodeObj = node::getInstanceFirst(array('nodeID' => $nodeID));
                    $arrayOfNodeIDLevels[] = $nodeObj->level;
          } */



          $matrix = array(); // $matrix[participantID,statementID] -> $matrix = array(participant1 =
array(statement1,statement2),participant2 = array (statement1, statement2))



$matrix = array(
array(3,3,3,3,-3,-3,-3,-3,-3,-3,-3,-3,3,3,3,3),
array(-3,-3,-3,-3,3,3,3,3,3,3,3,3,-3,-3,-3),
array(3,3,3,2,-3,-2,-2,-2,-2,-2,-2,-1,0,0,0),
array(-3,-2,-2,-2,3,1,1,3,0,2,2,0,0,0,0),
array(2,2,2,2,-2,-1,-1,-1,0,-1,0,-1,0,0,0),
array(-2,-2,-2,-1,2,0,0,2,0,1,0,0,0,0,0),
array(1,1,1,1,-1,0,0,0,0,-1,0,0,0,0,0),
array(-3,-3,-3,-3,3,3,2,3,3,3,2,3,-1,-2,-2),
array(3,2,3,2,-3,-2,-2,-3,-3,-3,-2,-3,1,2,2),
array(-3,-3,-3,-3,2,2,2,2,3,3,2,3,-1,-1,-1),
array(3,2,2,2,-2,-1,-1,-2,-3,-2,-2,-3,1,1,1),
array(-2,-1,-2,-1,2,2,1,2,2,2,1,2,-1,0,0),
array(2,2,3,0,-2,-2,0,-2,-2,-2,-1,-2,1,0,0),
array(-2,-2,-2,0,2,1,0,1,1,1,0,1,0,0,0),
array(1,1,1,0,-2,0,0,-1,-1,-1,0,-1,0,0,0),
array(-1,-1,-1,0,1,0,0,1,1,1,0,0,0,0,0),
array(1,1,1,0,-1,0,0,0,0,0,0,0,0,0,0),
array(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
);

$arrayOfNodeIDLevels = array(); //information about the structure
$arrayOfNodeIDLevels[0] = 0;
$arrayOfNodeIDLevels[1] = 2;
$arrayOfNodeIDLevels[2] = 2;
$arrayOfNodeIDLevels[3] = 2;
$arrayOfNodeIDLevels[4] = 2;
$arrayOfNodeIDLevels[5] = 2;
$arrayOfNodeIDLevels[6] = 2;
$arrayOfNodeIDLevels[7] = 4;
$arrayOfNodeIDLevels[8] = 4;
$arrayOfNodeIDLevels[9] = 4;
$arrayOfNodeIDLevels[10] = 4;
$arrayOfNodeIDLevels[11] = 4;
$arrayOfNodeIDLevels[12] = 4;
$arrayOfNodeIDLevels[13] = 6;
$arrayOfNodeIDLevels[14] = 6;





        echo "nodes: ".count($matrix[0])."<br>Participants: ".count($matrix)."<br><br>";

        $arrayOfNodeIDs = array(); //levelorder sort
        for ($i = 0; $i < count($matrix[0]); $i++) $arrayOfNodeIDs[$i] = $i+1; //id zero is 1, id one
is 2...

        $arrayOfParticipantIDs = array();
        for ($i = 0; $i < count($matrix); $i++) $arrayOfParticipantIDs[$i] = $i+1; //id zero is 1, id
one is 2...

        // $arrayOfNodeIDLevels = array(); //put all the levels for each node in another array with
the same index as the node itself
```

```
	//schritt 1.5: nichtwähler identifizieren und aus der matrix raus nehmen. auch aus der
arrayOfParticipantIDs raus nehmen - 02.01.2012
	$nonVotersUserIDs = array(); //an array for all the UserIDs of non voters
	for ($j = 0; $j < count($matrix); $j++) { //go through lines / participants (for each
participant)
		$opinionArray = $matrix[$j];
		if (round(str_replace("-","",implode("",$opinionArray))) == 0) {
			array_push($nonVotersUserIDs,$arrayOfParticipantIDs[$j]);
			unset($matrix[$j]);
			unset($arrayOfParticipantIDs[$j]);
		}
	}
	$arrayOfParticipantIDs = array_merge($arrayOfParticipantIDs); //truncate the arrays (they
were with keys 0,1,2,3,4,5 and now maybe 0,1,2,4,5. change to 0,1,2,3,4)
	$matrix = array_merge($matrix);

	//schritt 2, fraktionsanalyse
	$analysis = fractionUniqueAnalysis($matrix,$arrayOfParticipantIDs);
	$fractions = $analysis['fractions'];
	$uniques = $analysis['uniques'];

	//schritt 3, fraktionsgranularität anpassen
	$allRemovedNodeIDs = array();
	$counterr = 0;
	while (count($uniques) > (count($arrayOfParticipantIDs) / 2)) { //as long as there are more
than half of the people having a unique opinion about the whole graph, cut the leaf nodes (aka look
at less nodes for their opinions)
		$counterr++;
		//echo"<br>".$counterr."<br>";

		//echo count($uniques)." > ".(count($arrayOfParticipantIDs) / 2)."<br>";

		$differenceVectors = array();
		$smallestDifferenceVectorSize = count($uniques[0]['opinion']); //obergrenze
		//echo $smallestDifferenceVectorSize." ";
		//from now on working with the KEY representing the node, not with the node itself
		for ($i = 0; $i < count($uniques); $i++) { //run over all unique uservectors
	WARNING: the $i and $j from "unique" vector CANNOT be translated to userIDs after the
first fraction has been made. because the keys from "uniques" array are for JUST THE UNIQUES
not for all
			for ($j = $i+1; $j < count($uniques); $j++) { //run over all unique uservectors
				//calculate difference vector
				$thisDifferenceVector = array();
				for ($d = 0; $d < count($uniques[$i]['opinion']); $d++) { //go through
all the node KEY ids. could use $uniques[$j]['opinion'] as well
					if ($uniques[$i]['opinion'][$d] != $uniques[$j]['opinion'][$d])
array_push($thisDifferenceVector,$d); //puts the KEY id in the difference vector because for this
node, there are different opinions
				}
```

```
					if (count($thisDifferenceVector) == 0) { echo "ERROR user keys
".$i." ".$j."<br>"; echo "user ".$arrayOfParticipantIDs[$i]; preint_r($uniques[$i]['opinion']); echo
"user ".$arrayOfParticipantIDs[$j]; preint_r($uniques[$j]['opinion']); }
					if ((count($thisDifferenceVector) > 0) &&
(count($thisDifferenceVector) <= $smallestDifferenceVectorSize)) { //only put the ones with small
sizes in the differenceVectors
						$smallestDifferenceVectorSize =
count($thisDifferenceVector);
						array_push($differenceVectors,$thisDifferenceVector);
					}
				}
			}
			if (count($differenceVectors) == 0) break;

			//check which are the smallest difference vectors.
			for ($i = 0; $i < count($differenceVectors); $i++) {
				if (count($differenceVectors[$i]) > $smallestDifferenceVectorSize) {
					unset($differenceVectors[$i]);
				}
			}
			$smallestDifferenceVectors = array_merge($differenceVectors);

			//preint_r($differenceVectors);

			//falls $smallestDifferenceVectors mehr als 1 vektor drin hat:
			$highestVectorLevelSum = array("sum" => 0, "vector" => array());
			for ($i = 0; $i < count($smallestDifferenceVectors); $i++) { //a vector of multiple
vectors with KEY ids of nodes
				$thisVectorLevelSum = 0;
				for ($j = 0; $j < count($smallestDifferenceVectors[$i]); $j++) {
					$thisNodeLevel =
$arrayOfNodeIDLevels[$smallestDifferenceVectors[$i][$j]];
					$thisNodeID =
$arrayOfNodeIDs[$smallestDifferenceVectors[$i][$j]]; $thisNodeObj =
node::getInstancesMatch(array("nodeID" => $thisNodeID));
					$thisVectorLevelSum = $thisVectorLevelSum + $thisNodeLevel;
					if ($thisNodeObj->nodeType == "leitaussage") $thisVectorLevelSum
= $thisVectorLevelSum - 0.5;
				}
				if ($thisVectorLevelSum >= $highestVectorLevelSum['sum']) {
					$highestVectorLevelSum['sum'] = $thisVectorLevelSum;
					$highestVectorLevelSum['vector'] = $smallestDifferenceVectors[$i];
				}
			}

			$nodeKeysToBeRemoved = $highestVectorLevelSum['vector'];

			for ($i = 0; $i < count($matrix); $i++) {
				for ($j = 0; $j < count($nodeKeysToBeRemoved); $j++) {
unset($matrix[$i][$nodeKeysToBeRemoved[$j]]); }
				$matrix[$i] = array_merge($matrix[$i]);
			}
```

```
                for ($j = 0; $j < count($nodeKeysToBeRemoved); $j++) {

        array_push($allRemovedNodeIDs,$arrayOfNodeIDs[$nodeKeysToBeRemoved[$j]]);
//$allRemovedNodeIDs sind ALLE removed nodes von allen interationen
                        unset($arrayOfNodeIDLevels[$nodeKeysToBeRemoved[$j]]);
                        unset($arrayOfNodeIDs[$nodeKeysToBeRemoved[$j]]);
                }
                $arrayOfNodeIDLevels = array_merge($arrayOfNodeIDLevels);
                $arrayOfNodeIDs = array_merge($arrayOfNodeIDs);

                $analysis = fractionUniqueAnalysis($matrix,$arrayOfParticipantIDs);
                $fractions = $analysis['fractions'];
                $uniques = $analysis['uniques'];

                //echo count($uniques)." > ".(count($arrayOfParticipantIDs) / 2)."<br>";
                //echo "<br><br><br><br><br>";
                /*levels wegkicken: start
                $keysWithHighestValue =
array_keys($arrayOfNodeIDLevels,max($arrayOfNodeIDLevels));
                for ($i = 0; $i < count($matrix); $i++) {
                        foreach ($keysWithHighestValue as $key) { unset($matrix[$i][$key]);
unset($arrayOfNodeIDLevels[$key]); unset($arrayOfNodeIDs[$key]); }
                }
                //levels wegkicken: end */
        }
        echo "fractions:";
        preint_r($fractions);
        echo "uniques (not in fractions):";
        preint_r($uniques);
        echo "nonVoters as far as granularity level (also not in fractions):";
        preint_r($nonVotersUserIDs);

/*      //schritt 4, kompatibilitätsprüfung
        for ($i = 0; $i < count($uniques); $i++) {
                for ($j = 0; $j < count($fractions); $j++) {
                        $dot_product = array_map('bcmul', $uniques[$i]['opinion'],
$fractions[$j]['opinion']);
                        //$dot_product = array_map(function($a,$b) { return $a*$b; },
$uniques[$i]['opinion'], $fractions[$j]['opinion']); //evtl funktioniert net:
array_map(create_function('$a, $b', 'return $a * $b;'), $array1, $array2) oder array_map('bcmul',
$array1, $array2)
                        $allNegativeValues = array_filter($dot_product, 'isNegative');
                        if (count($allNegativeValues) == 0) { //einzelmeinung von $uniques ist
kompatibel mit dieser Fraktion
                                $uniques[$i]['compatibleFractions'][] = $fractions[$j];
                                $fractions[$j]['compatibleParticipants'][] = $uniques[$i]; //zero
because it's only one
                        }
                }
        }

        //schritt 5, resultate
```

```
	$anzahlEinzelmeinungen = count($uniques);
	$anzahlLevels = count(array_unique($arrayOfNodeIDLevels));

	$anzahlInkompatibleEinzelmeinungen = 0;
	for ($i = 0; $i < count($uniques); $i++) if ((!is_array($uniques[$i]['compatibleFractions'])) ||
(count($uniques[$i]['compatibleFractions']) == 0)) $anzahlInkompatibleEinzelmeinungen++;

	$allRemovedNodeContents = array();
	for ($i = 0; $i < count($allRemovedNodeIDs); $i++) { $nodeObj =
node::getInstanceFirst(array('nodeID' => $allRemovedNodeIDs[$i]));
array_push($allRemovedNodeContents,$nodeObj->content); }

	return array( 'originalMatrix' => $originalMatrix, 'matrix' => $matrix, 'nodeIDs' =>
$arrayOfNodeIDs, 'fractions' => $fractions, 'uniques' => $uniques, 'nonVotersUserIDs' =>
$nonVotersUserIDs, 'count' => array('einzelmeinungen' => $anzahlEinzelmeinungen, 'levels' =>
$anzahlLevels, 'inkompatibleEinzelmeinungen' => $anzahlInkompatibleEinzelmeinungen),
'allRemovedNodeContents' => $allRemovedNodeContents, 'allRemovedNodeIDs' =>
$allRemovedNodeIDs);
 */
 // }

function parseLevelOrder($nodeObj) {
	$return = array();
	$queue = array();
	$queue[] = $nodeObj;
	$return[] = $nodeObj->nodeID;
	while (count($queue) > 0) {
		$nodeObj = array_shift($queue);
		$nodeObj->visited = true;
		if ((($nodeObj->nodeType == "leitaussage") || ($nodeObj->nodeType ==
"zusatzbedingung")) && ($nodeObj->isDeleted == 0) && ($nodeObj->isOrphan == 0)) $return[] =
$nodeObj->nodeID;
		$children = $nodeObj->children;
		if (!is_array($children)) continue;
		usort($children, 'levelSort');
		for ($i = 0; $i < count($children); $i++) {
			$queue[] = $children[$i];
		}
	}
	return $return;
}
function levelSort($a, $b) {
	if (($a->nodeType == "pro") && ($b->nodeType != "pro")) return -1;
	if (($b->nodeType == "pro") && ($a->nodeType != "pro")) return +1;
	if (($a->nodeType == "contra") && ($b->nodeType != "contra")) return -1;
	if (($b->nodeType == "contra") && ($a->nodeType != "contra")) return +1;
	if (($a->nodeType == "leitaussage") && ($b->nodeType != "leitaussage")) return -1;
	if (($b->nodeType == "leitaussage") && ($a->nodeType != "leitaussage")) return +1;
	return ($a->nodeID > $b->nodeID?+1:($a->nodeID < $b->nodeID?-1:0));
}

function fractionUniqueAnalysis($array,$arrayOfParticipantIDs) {
	$arrayUnique = arrayUnique( $array ); //output is die matrix mit nur einzelmeinunen. alle
duplikate sind weg
	$duplicates = arrayUnique(array_diff_assoc( $array, $arrayUnique )); //output sind die
duplikate
	$arrayUnique = array_merge($arrayUnique); //indices korrigieren

	$einzelmeinungen = array();
	for ($i = 0; $i < count($arrayUnique); $i++) { //array_diff($arrayUniqu,$duplicates); does
not seem to work
		if (!in_array($arrayUnique[$i],$duplicates)) $einzelmeinungen[] = array('participant'
=>
array_merge(array_intersect_key($arrayOfParticipantIDs,array_flip(array_keys($array,$arrayUniqu
e[$i],true)))),

								'opinion' => $arrayUnique[$i]);
	}

	$fractions = array();
	foreach ($duplicates as $value) {
	//for ($i = 0; $i < count($duplicates); $i++) {
		$fractions[] = array('participants' =>
array_merge(array_intersect_key($arrayOfParticipantIDs,array_flip(array_keys($array,$value,true)
))),
								'opinion' => $value);
	}

	// preint_r($fractions);
	// echo("a");
	return array("fractions" => $fractions,"uniques" => $einzelmeinungen);
}


/* URSACHENANALYSE er,ittlung der menge von Leafnodes, über die Teilnehmer sich einig
werden müssten, sodass konsens über haupthypothese erreicht wird. es gan mehr als nur eine menge
von blätter geben. ziel sind es die kleinsten zu identifizeren */




/* findreasons erzeugt alle gründe für einen dissens, und sortiert sie um die kleinsten zu wählen */


function findReasons ($treeID) {
	$rootNodeInfo = getInfo_debateRootNode_withContent($treeID);
	$rootNodeObj = node::getInstanceFirst(array('nodeID' => $rootNodeInfo['nodeID']));
//wichtig: wenn fraktionsanalyse nicht gemacht wurde, gibts die objekte nicht

	/* behandlung von erste ebene: begin */
	/*$P = 0; $C = 0;
	for ($i = 0; $i < count($rootNodeObj->children); $i++) {
		//echo " i:".$i."/count:".count($rootNodeObj->children)."/".$rootNodeObj-
>children[$i]->nodeID." ";
```

```
                if (($rootNodeObj->children[$i]->nodeType == "pro") &&
(posconsArg($rootNodeObj->children[$i]))) $P++;
                if (($rootNodeObj->children[$i]->nodeType == "contra") &&
(posconsArg($rootNodeObj->children[$i]))) { $C++; } //echo "POSCONSARG ZU:";
preint_r($rootNodeObj->children[$i]);
        }

        if (($P == 0) && ($C == 0)) {
                for ($i = 0; $i < count($rootNodeObj->children); $i++) {
                        if (($rootNodeObj->children[$i]->nodeType == "pro") || ($rootNodeObj-
>children[$i]->nodeType == "contra")) {
                                $thisArgument = $rootNodeObj->children[$i];
                                $thisReason = new reason(array("pos" => array($thisArgument),
"neg" => array(), "posNT" => array(),  "negNT" => array()));
                                $inverseNodeType = ($thisArgument->nodeType ==
"pro"?"contra":"pro");
                                $allInverseArguments = node::getInstancesMatch(array("nodeType"
=> $inverseNodeType, "parent" => $rootNodeObj->nodeID));
                                for ($j = 0; $j < count($allInverseArguments); $j++) {
                                        $thisReason->neg[] = $allInverseArguments[$j];
                                        //echo "reasonID: ".$thisReason->id." findreasons1<br>";
                                }
                        }
                }
        }

        elseif (($P > 0) && ($C == 0)) {
                $thisReason = new reason(array("pos" => array(), "neg" => array(), "posNT" =>
array(),  "negNT" => array()));
                $allContraArguments = node::getInstancesMatch(array("nodeType" => "contra",
"parent" => $rootNodeObj->nodeID));
                for ($j = 0; $j < count($allContraArguments); $j++) {
                        $thisReason->neg[] = $allContraArguments[$j];
                        //echo "reasonID: ".$thisReason->id." findreasons2<br>";
                }
        }

        elseif (($P == 0) && ($C > 0)) {
                $thisReason = new reason(array("pos" => array(), "neg" => array(), "posNT" =>
array(),  "negNT" => array()));
                $allProArguments = node::getInstancesMatch(array("nodeType" => "pro", "parent"
=> $rootNodeObj->nodeID));
                for ($j = 0; $j < count($allProArguments); $j++) {
                        $thisReason->neg[] = $allProArguments[$j];
                        //echo "reasonID: ".$thisReason->id." findreasons33<br>";
                }
        }

        elseif (($P > 0) && ($C > 0)) {
                $thisReason = new reason(array("pos" => array(), "neg" => array(), "posNT" =>
array(),  "negNT" => array()));
```

```
                $allProArguments = node::getInstancesMatch(array("nodeType" => "pro", "parent"
=> $rootNodeObj->nodeID));
                for ($j = 0; $j < count($allProArguments); $j++) {
                        $thisReason->neg[] = $allProArguments[$j];
                        //echo "reasonID: ".$thisReason->id." findreasons4<br>";
                }
                $thisReason = new reason(array("pos" => array(), "neg" => array(), "posNT" =>
array(),  "negNT" => array()));
                $allContraArguments = node::getInstancesMatch(array("nodeType" => "contra",
"parent" => $rootNodeObj->nodeID));
                for ($j = 0; $j < count($allContraArguments); $j++) {
                        $thisReason->neg[] = $allContraArguments[$j];
                        //echo "reasonID: ".$thisReason->id." findreasons5<br>";
                }
        }*/
        /* behandlung von erste ebene: end */

        $posReason = new reason(array("pos" => array($rootNodeObj), "neg" => array(), "posNT"
=> array(),  "negNT" => array()));
        $negReason = new reason(array("pos" => array(), "neg" => array($rootNodeObj), "posNT"
=> array(),  "negNT" => array()));

        $remainingReasons = reason::getInstancesMatch(array('terminal' => null)); //these are the
reasons that have not been seen as terminal (in the beginning: all)
        while (count($remainingReasons) > 0) {
                for ($i = 0; $i < count($remainingReasons); $i++) {
                        if (nonTerminal($remainingReasons[$i])) { /*echo "nonterminal ".$i;*/
refineReason($remainingReasons[$i]); }
                        else { /*echo "isterminal ".$i;*/ $remainingReasons[$i]->terminal = true; }
                }
                $remainingReasons = reason::getInstancesMatch(array('terminal' => null));
        }

        $reasons = reason::getInstances();
        $smallestSize = PHP_INT_MAX;
        for ($i = 0; $i < count($reasons); $i++) { $thisSize = calcSize($reasons[$i]); if ($thisSize <
$smallestSize) $smallestSize = $thisSize; }

        //return reason::getInstancesMatch(array("size" => $smallestSize));
        return reason::getInstances();

}

/* hilfsfunktionen für findreasons */

function posconsArg($argumentObj) { //returns false if false, true if true
        $allParticipants = user::getInstances();
        for ($i = 0; $i < count($allParticipants); $i++) if (agree2arg($allParticipants[$i]-
>userID,$argumentObj->nodeID) == false) { return false; }
        return true;
}
```

```
function negconsArg($argumentObj) { //returns false if false, true if true
	$allParticipants = user::getInstances();
	for ($i = 0; $i < count($allParticipants); $i++) if (disagree2arg($allParticipants[$i]-
>userID,$argumentObj->nodeID) == false) { return false; }
	return true;
}

function posconsStatement($statementObj) { //returns false if false, true if true
	fillOpinionStatistics($statementObj); $totalUsers = count(user::getInstances());
	if ($statementObj->nodeType == "zusatzbedingung") { if (($statementObj-
>opinionStatistics['trifftzu'] + $statementObj->opinionStatistics['entfernen']) == $totalUsers) return
true; }
	elseif ($statementObj->nodeType == "leitaussage") { if ($statementObj-
>opinionStatistics['trifftzu'] == $totalUsers) return true; }
	return false;
}

function negconsStatement($statementObj) { //returns false if false, true if true
	fillOpinionStatistics($statementObj); $totalUsers = count(user::getInstances());
	if ($statementObj->nodeType == "zusatzbedingung") { if ($statementObj-
>opinionStatistics['trifftnichtzu'] == $totalUsers) return true; }
	elseif ($statementObj->nodeType == "leitaussage") { if (($statementObj-
>opinionStatistics['trifftnichtzu'] + $statementObj->opinionStatistics['entfernen']) == $totalUsers)
return true; }
	return false;
}

function potentialPosconsStatement($statementObj) { //returns array() if false, array("an
	if ($statementObj->opinionStatistics['trifftnichtzu'] == 0) return
array("anzahlUnentschlossene" => ($totalUsers - $statementObj->opinionStatistics['trifftzu']));
//equals to return true
	return array(); //equals to return false
}

function potentialNegconsStatement($statementObj) { //returns array() if false,
array("anzahlUnentschlossene" => X) if true
	fillOpinionStatistics($statementObj); $totalUsers = count(user::getInstances());
	if ($statementObj->opinionStatistics['trifftzu'] == 0) return array("anzahlUnentschlossene"
=> ($totalUsers - $statementObj->opinionStatistics['trifftnichtzu'])); //equals to return true
	return array(); //equals to return false
}

function disagree2arg($userID, $nodeID) { //nodeID is an argument (implicit. not checked)
	if ((!isset($userID)) || (!isset($nodeID))) {
		echo("disagree2arg: one of the variables has no value. userID: ".$userID." or
nodeID: ".$nodeID);
		return false;
	}
	$childrenArray = getInfo_DirectChildrenOfNode($nodeID);

	for ($i = 0; $i < count($childrenArray); $i++) {
		$thisChild = $childrenArray[$i];
		if ($thisChild['nodeType'] == 'leitaussage') { // wenn leitaussage oder zb mit
trifftnichtzu: true
			if ((getOpinion($userID, $thisChild['nodeID']) == 'trifftnichtzu') ||
(getOpinion($userID, $thisChild['nodeID']) == 'entfernen')) return true;
		}
		elseif (getOpinion($userID, $thisChild['nodeID']) == 'trifftnichtzu') return true;
	}
	return false;
}

function fillOpinionStatistics($nodeObj) {
	if (count($nodeObj->opinionStatistics) == 0) {
		$nodeObj->opinionStatistics = array("trifftzu" => 0, "trifftnichtzu" => 0,
"keinemeinung" => 0, "entfernen" => 0);
		$nodeObj->opinionStatistics = array_extend($nodeObj-
>opinionStatistics,getInfo_nodeOpinionStatistic($nodeObj->nodeID));
	}
}

function report_opinionStatistics($nodeObj) {
	return "<span class='opinionStats trifftzuBG'>".$nodeObj-
>opinionStatistics['trifftzu']."</span></span><span class='opinionStats
trifftnichtzuBG'>".$nodeObj->opinionStatistics['trifftnichtzu']."</span><span class='opinionStats
keinemeinungBG'>".$nodeObj->opinionStatistics['keinemeinung']."</span><span
class='opinionStats entfernenBG'>".$nodeObj->opinionStatistics['entfernen']."</span>";
}

function nonTerminal ($reasonObj) {
	return (terminal($reasonObj)?false:true);
}

function terminal ($reasonObj) {
	for ($j = 0; $j < count($reasonObj->pos); $j++) if ((is_array($reasonObj->pos[$j]-
>children)) && (is_object($reasonObj->pos[$j]->children[0]))) return false; //at least one node is
not a leaf node
	for ($j = 0; $j < count($reasonObj->neg); $j++) if ((is_array($reasonObj->neg[$j]-
>children)) && (is_object($reasonObj->neg[$j]->children[0]))) return false;
	return true;
}

function refineReason (&$reasonObj) {
	if ((count($reasonObj->pos) == 0) && (count($reasonObj->neg) == 0)) return false;
	$posOk = false; $negOk = false;
	while ((!$posOk) || (!$negOk)) {

		$ersterPosElementDasKeinBlattIst = null;
		for ($w = 0; $w < count($reasonObj->pos); $w++) {
			if ((is_array($reasonObj->pos[$w]->children)) && (is_object($reasonObj-
>pos[$w]->children[0]))) { //falls kein blattknoten
				$ersterPosElementDasKeinBlattIst = array("element" => $reasonObj-
>pos[$w], "index" => $w);
				break;
```

```
                    }
                }
                if (is_array($ersterPosElementDasKeinBlattIst)) {
                        $nodeObj = $ersterPosElementDasKeinBlattIst['element'];
                        unset($reasonObj->pos[$ersterPosElementDasKeinBlattIst['index']]);
                        $reasonObj->pos = array_merge($reasonObj->pos);
                        //array_splice($reasonObj-
>pos,$ersterPosElementDasKeinBlattIst['index'],1);
                        prove($nodeObj,$reasonObj);
                }
                else $posOk = true;

                $ersterNegElementDasKeinBlattIst = null;
                for ($w = 0; $w < count($reasonObj->neg); $w++) {
                    if ((is_array($reasonObj->neg[$w]->children)) && (is_object($reasonObj-
>neg[$w]->children[0]))) {
                        $ersterNegElementDasKeinBlattIst = array("element" =>
$reasonObj->neg[$w], "index" => $w);
                        break;
                    }
                }
                if (is_array($ersterNegElementDasKeinBlattIst)) {
                        $nodeObj = $ersterNegElementDasKeinBlattIst['element'];
                        //echo "reasonID: ".$reasonObj->id." count ".count($reasonObj-
>neg);
                        unset($reasonObj->neg[$ersterNegElementDasKeinBlattIst['index']]);
                        $reasonObj->neg = array_merge($reasonObj->neg);
                        //array_splice($reasonObj-
>neg,$ersterNegElementDasKeinBlattIst['index'],1);
                        //echo " to ".count($reasonObj->neg)."<br>";
                        disprove($nodeObj,$reasonObj);

                }
                else $negOk = true;
        }
        //preint_r(reason::getInstances());
        //return $reasonObj;
}

function prove ($nodeObj,&$reasonObj) {
        if (($nodeObj->nodeType == "leitaussage") || ($nodeObj->nodeType ==
"zusatzbedingung")) proveStatement($nodeObj,$reasonObj);
        else proveArg($nodeObj,$reasonObj);
}

function proveArg ($nodeObj,&$reasonObj) { //alle in gleicher pos liste: "AND"
        if (($nodeObj->nodeType == "leitaussage") || ($nodeObj->nodeType ==
"zusatzbedingung")) return false;
        for ($i = 0; $i < count($nodeObj->children); $i++) {
                $thisStatement = $nodeObj->children[$i];
                if (!posconsStatement($thisStatement)) { $reasonObj->pos[] = $thisStatement;
}//echo "proveArg"; preint_r($thisStatement); }
```

```
        }
        return $reasonObj;

        //$childrenWithNegativeOpinion =
node::getInstancesMatchAndHasOpinionType(array("parent" => $nodeObj-
>nodeID),array("trifftnichtzu" => 1)); // "trifftnichtzu" => 1 heisst "mindestens ein trifftnichtzu"
        //for ($i = 0; $i <= count($childrenWithNegativeOpinion); $i++) $reasonObj->pos[] =
$childrenWithNegativeOpinion[$i];
        //echo(count($reasonObj->pos));
        //return $reasonObj;
}

function proveStatement ($nodeObj,&$reasonObj) { //je in eigener pos liste: "OR"
        if (($nodeObj->nodeType != "leitaussage") && ($nodeObj->nodeType !=
"zusatzbedingung")) return false;
        $thereAreProArgsWithPosCons = false;
        for ($i = 0; $i < count($nodeObj->children); $i++) {
                $thisArgObj = $nodeObj->children[$i];
                if (($thisArgObj->nodeType == "contra") && (!negconsArg($thisArgObj))) {
$reasonObj->neg[] = $thisArgObj; /*echo "reasonID: ".$reasonObj->id." provestatement<br>";*/ }
//Füge alle Contra-Argumente zu el der neg-Liste von r hinzu
                //if (($thisArgObj->nodeType == "pro") && (posconsArg($thisArgObj)))
$thereAreProArgsWithPosCons = true; //if there is at least one pro with posconsArg, dont continue
        }
        if (!posconsStatement($nodeObj)) $reasonObj->posNT[] = $nodeObj;
        /*if ($thereAreProArgsWithPosCons == false) {
                $allProArgsToNode = node::getInstancesMatch(array("parent" => $nodeObj-
>nodeID, "nodeType" => "pro"));
                for ($i = 1; $i < count($allProArgsToNode); $i++) {
                        $thisReason = new reason(array("pos" => $reasonObj->pos, "neg" =>
$reasonObj->neg, "posNT" => array(),  "negNT" => array()));
                        $thisReason->pos[] = $allProArgsToNode[$i]; //no need to check whether
posconsArg, because of $thereAreProArgsWithPosCons == false
                         //echo "proveStatement"; preint_r($allProArgsToNode[$i]);
                }
                if (count($allProArgsToNode) > 0) $reasonObj->pos[] = $allProArgsToNode[0];
        }*/
        return $reasonObj;
}


function disprove ($nodeObj,&$reasonObj) {
        if (($nodeObj->nodeType == "leitaussage") || ($nodeObj->nodeType ==
"zusatzbedingung")) disproveStatement($nodeObj,$reasonObj);
        else disproveArg($nodeObj,$reasonObj);
}

function disproveArg ($nodeObj,&$reasonObj) {
        if (($nodeObj->nodeType == "leitaussage") || ($nodeObj->nodeType ==
"zusatzbedingung")) return false;
        $thereAreStatementsWithNegCons = false;
        for ($i = 0; $i < count($nodeObj->children); $i++) {
                $thisStatement = $nodeObj->children[$i];
                if (negconsStatement($thisStatement)) $thereAreStatementsWithNegCons = true;
        }
        if ($thereAreStatementsWithNegCons == false) {
                for ($i = 1; $i < count($nodeObj->children); $i++) { //copy original reason fpr all
statements from 1 to n
                        $thisStatement = $nodeObj->children[$i];
                        $thisReason = new reason(array("pos" => $reasonObj->pos, "neg" =>
$reasonObj->neg, "posNT" => array(),  "negNT" => array()));
                        $thisReason->neg[] = $thisStatement;
                        //echo "reasonID: ".$thisReason->id." disprovearg<br>";
                }
                if (count($nodeObj->children) > 0) { $reasonObj->neg[] = $nodeObj->children[0];
/*echo "reasonID: ".$reasonObj->id." disprovearg<br>";*/ } //put the first statement in the original
reason object of there is no negative consens
        }
        //$childrenWithPositiveOpinion =
node::getInstancesMatchAndHasOpinionType(array("parent" => $nodeObj-
>nodeID),array("trifftzu" => 1));
        return $reasonObj;
}

function disproveStatement ($nodeObj,&$reasonObj) {
        if (($nodeObj->nodeType != "leitaussage") && ($nodeObj->nodeType !=
"zusatzbedingung")) return false;
        $thereAreContraArgsWithPosCons = false;
        for ($i = 0; $i < count($nodeObj->children); $i++) {
                $thisArgObj = $nodeObj->children[$i];
                if (($thisArgObj->nodeType == "pro") && (!negconsArg($thisArgObj))) {
$reasonObj->neg[] = $thisArgObj; /*echo "reasonID: ".$reasonObj->id."
disprovestatement<br>";*/ }
                //if (($thisArgObj->nodeType == "contra") && (posconsArg($thisArgObj)))
$thereAreContraArgsWithPosCons = true;
        }
        if (!negconsStatement($nodeObj)) $reasonObj->negNT[] = $nodeObj;
        /*if ($thereAreContraArgsWithPosCons == false) {
                $allContraArgsToNode = node::getInstancesMatch(array("parent" => $nodeObj-
>nodeID, "nodeType" => "contra"));
                for ($i = 1; $i < count($allContraArgsToNode); $i++) {
                        $thisArgument = $allContraArgsToNode[$i];
                        $thisReason = new reason(array("pos" => $reasonObj->pos, "neg" =>
$reasonObj->neg, "posNT" => array(),  "negNT" => array()));
                        $thisReason->pos[] = $thisArgument;
                         //echo "disproveStatement"; preint_r($allContraArgsToNode[$i]);
                }
                if (count($allContraArgsToNode) > 0) $reasonObj->pos[] =
$allContraArgsToNode[0];
        }*/
        return $reasonObj;
}
```

```
function calcSize (&$reasonObj) {
	$size = 0;
	for ($i = 0; $i < count($reasonObj->pos); $i++) {
		$thisElement = $reasonObj->pos[$i];
		if (is_array($thisElement->opinion)) {
			$opinionArray = array_merge($thisElement->opinion);
			for ($j = 0; $j < count($opinionArray); $j++) {
				if ($opinionArray[$j]['opinion'] == "trifftnichtzu") $size = $size +
$opinionArray[$j]['certainty'];
			}
		}
	}
	for ($i = 0; $i < count($reasonObj->neg); $i++) {
		$thisElement = $reasonObj->neg[$i];
		if (is_array($thisElement->opinion)) {
			$opinionArray = array_merge($thisElement->opinion);
			for ($j = 0; $j < count($opinionArray); $j++) {
				if ($opinionArray[$j]['opinion'] == "trifftzu") $size = $size +
$opinionArray[$j]['certainty'];
			}
		}
	}
	$reasonObj->size = $size;
	return $size;
}


class reason {
	public static $instances = Array();
	public static $id = 0;

	function reason ($param) {
		if (is_array($param))  self::constructor_array($param);
		if (is_object($param)) self::constructor_obj($param);
		array_push(self::$instances,$this);
		$this->id = self::$id;
		self::$id = self::$id + 1;
		return $this;
	}

	//secondary constructors depending on type of parameter
	function constructor_obj($obj) { //copies the parameters in Obj into thisObj
		$array = $obj->getVars();
		foreach ($array as $key => $value) $this->$key = $value;
	}
	function constructor_array($array) { foreach ($array as $key => $value) $this->$key =
$value; }

	//generic object functions
	public function getVars() {
		return get_object_vars($this);
	}
	public function matchesThisObj($needle) {
		$matches = 0;
		foreach ($needle as $needleKey => $needleValue) if ($this->$needleKey ==
$needleValue) $matches++;
		if ($matches == count($needle)) return true; else return false;
	}
	public function appendInfo($param) {
		if (is_object($param)) {
			$array = $param->getVars(); foreach ($array as $key => $value) $this->$key
= $value;
		} elseif (is_array($param)) {
			foreach ($param as $key => $value) $this->$key = $value;
		}
	}

	//static functions
	public static function getInstances() {
		return self::$instances;
	}
	public static function getInstanceFirst($needle) {
		foreach (self::$instances as $instance) if ($instance->matchesThisObj($needle))
return $instance;
		return false;
	}
	public static function getInstancesMatch($needle) {
		$returnArray = Array();
		foreach (self::$instances as $instance) if ($instance->matchesThisObj($needle))
$returnArray[] = $instance;
		return $returnArray;
	}
	public static function destroyAllInstances() {
		foreach (self::$instances as $instance) unset($instance);
		return true;
	}

} ?>
```

Supporting functions and classes for the fraction analysis

```
<?php
$funcincluded = 1;

function arrayUnique($myArray) {
   foreach ($myArray as &$myvalue) $myvalue=serialize($myvalue);
   $myArray=array_unique($myArray);
   foreach ($myArray as &$myvalue) $myvalue=unserialize($myvalue);
   return $myArray;
}

class node {
        public static $instances = Array();

        function node ($param) {
                if (is_array($param))  self::constructor_array($param);
                if (is_object($param)) self::constructor_obj($param);
                array_push(self::$instances,$this);
                return $this;
        }

        //secondary constructors depending on type of parameter
        function constructor_obj($obj) { //copies the parameters in Obj into thisObj
                $array = $obj->getVars();
                foreach ($array as $key => $value) $this->$key = $value;
        }
        function constructor_array($array) { foreach ($array as $key => $value) $this->$key =
$value; }

        //generic object functions
        public function getVars() {
                return get_object_vars($this);
        }
        public function matchesThisObj($needle) {
                $matches = 0;
                foreach ($needle as $needleKey => $needleValue) if ($this->$needleKey ==
$needleValue) $matches++;
                if ($matches == count($needle)) return true; else return false;
        }
        public function appendInfo($param) {
                if (is_object($param)) {
                        $array = $param->getVars(); foreach ($array as $key => $value) $this->$key
= $value;
                } elseif (is_array($param)) {
                        foreach ($param as $key => $value) $this->$key = $value;
                }
        }

        //object functions for class NODE
        public function hasOpinionType($typeArray) {
                if (!is_array($this->opinion)) return false;
```

```
		$opinionArray = array_merge($this->opinion);
		$thisTypeArray = array();
		for ($i = 0; $i < count($opinionArray); $i++)
$thisTypeArray[$opinionArray[$i]['opinion']]++;
		foreach ($typeArray as $opinion => $minValue) if ($thisTypeArray[$opinion] <
$minValue) return false;
		return true;
	}

	//static functions
	public static function getInstances() {
		return self::$instances;
	}
	public static function getInstanceFirst($needle) {
		foreach (self::$instances as $instance) if ($instance->matchesThisObj($needle))
return $instance;
		return false;
	}
	public static function getInstancesMatch($needle) {
		$returnArray = Array();
		foreach (self::$instances as $instance) if ($instance->matchesThisObj($needle))
$returnArray[] = $instance;
		return $returnArray;
	}
	public static function destroyAllInstances() {
		foreach (self::$instances as $instance) unset($instance);
		return true;
	}

	//static functions for class NODE
	public static function getInstancesHasOpinionType($typeArray,$inverse = false) {
//typearray is like: array("trifftzu" => some_minimal_value_here);
		$returnArray = Array();
		foreach (self::$instances as $instance) {
			//echo $instance->nodeID." ".($instance-
>hasOpinionType($typeArray)?"true":"false")."<br>";
			if ($instance->hasOpinionType($typeArray) == !$inverse) $returnArray[] =
$instance;
		}
		return $returnArray;
	}


	public static function getInstancesMatchAndHasOpinionType($needle,$typeArray) {
		$returnArray = Array();
		foreach (self::$instances as $instance) if (($instance->hasOpinionType($typeArray))
&& ($instance->matchesThisObj($needle))) $returnArray[] = $instance;
		return $returnArray;
	}
}

function preint_r($var) {
	echo "<pre>";
	print_r($var);
	echo "</pre>";
}
?>
```