Parameterless clustering by dynamic tree-cutting

Simon Lehmann Knudsen
simkn15@student.sdu.dk
ECTS: 10
04/09-2017 - 31/01-2018
5th semester in Computer Science

31/01-2018

Contents

Abstract

Background

Results

Conclusion

Figure 1: Examples of cluster shapes

1 Introduction

Clustering describes the unsupervised learning task of grouping similar objects together into so-called clusters. Due to the versatility, clustering is applied in almost all scientific fields, like economics, marketing, astronomy and many others [17]. In the field of bioinformatics, cluster analyses are used, e.g. analysis of microarray, cancer subtyping, protein homology detection, and many more [17]. The application fields are very different, but the researcher faces similar decisions and challenges [17]. A cluster is a group of objects that are more similar to each other than those of another cluster. Although this definition is clear, it becomes more vague when recognizing clusters in the plane. Figure 1 shows three different cluster shapes. One of the major problems of clustering is the absence of a clear definition of a *good* clustering [17]. Given a set of N objects $X = \{x_1, \ldots, x_N\}$, the following types of clustering tasks can be differentiated [25]:

- Partitional Clustering: The task of seeking a k-partition $C = \{C_1, \ldots, C_k\}$ of X, such that:
  1. $C_i \neq \emptyset, i = 1, \ldots, k$
  2. $\cup_{C_i \in C} C_i = X$
  3. $C_i \cap C_j = \emptyset \quad i, j = 1, \ldots, k, i \neq j$
- Overlapping Clustering: Follows the same principle as the partitional clustering with the difference that condition 3 does not hold. Each object can be a member of several clusters.
- Fuzzy Clustering: Assigns each object x_i a degree of membership $u_{i,j}$ to each cluster C_j of the k partitioning $C = \{C_1, \ldots, C_k\}$, such that
  1. $\sum_{i=1}^{k} u_{i,j} = 1 \quad \forall i$
  2. $0 < \sum_{i=1}^{k} u_{i.j} < N \quad \forall i$
- Hierarchical Clustering: The task of constructing a tree-like nested structure of partitions of X. Further discussion and definitions will be derived in Section 2.3.

The above definition only state what technically can be considered a clustering. They do not claim any requirements for a *good* or *bad* clustering. Putting all objects into one cluster, or setting all objects as their own cluster does not yield any insights. Judging the quality of a clustering is a non-trivial task, and is highly dependent on the situation. There exist no universally agreed-upon definition of the term cluster, due to the subjectivity of clustering [25].

Finding the proper clustering method requires a deep knowledge of the different clustering tools, as well as the domain of the dataset. The determination of the best cluster criteria can be regarded as one of the most challenging questions when performing a cluster analysis [17]. There exist no overall best-performer clustering tool, not even when limited to the same dataset type. The quality of the result if always highly dependent on the actual dataset [23]. When sorting n objects into m clusters, this totals to

$$\frac{1}{m!}\sum_{k=0}^{k-m}(-1)^{m-k}\binom{m}{k}k^n$$

different possibilities. For 25 objects and 5 clusters the number of possibilities grows to $2.4 \cdot 10^{15}$ [2]. The number of possibilities makes it absolutely infeasible to test all possible combinations to find the best clustering.

With todays technologies we are producing a vast amount of genomic data at an ever increasing pace [16]. For us to gain any benefits from the data, it not only has to be analyzed, but in an efficient and automated manner [24]. Even though there exists databases which provide information on protein family classification, e.g. SCOP^2 [7], the number of protein families are growing [15]. Resulting in an increasing importance to have reliable and automated means of classifying proteins in families. There are many different clustering tools, which all require different parameters, and can only be used efficiently with a profound understanding of the underlying algorithm [24]. As every clustering approach uses a different way of determining its optimal clustering. Meaning that every clustering has its strengths and weaknesses, such that there are no universal best performer [24]. Which approach to use, in order to gain the best clustering, is highly dependent on the dataset [24]. [24] investigated the performance of seven well-known clustering, with focus on the behavior of the tools' parameters. The results were that a good performance can only be reached by exhaustive parameter finding by comparison to a gold standard. As these golds standard are not available in practice, the parameters has to be retrieved by different means. The conclusion of the study was that state-of-the-art clustering tools have limitations capturing the diversity of protein families and requires a specific parameter for every dataset which cannot be easily provided in practice.

We want to investigate parameterless clustering by dynamic tree-cutting. Where the parameter for the individual clusters(protein families) are optimal. We want to do this by extending Transitivity Clustering(TC) into Hierarchical Clustering(HC) with a divisive approach, which we will call Transitivity-Hierarchical Clustering(THC). TC returns a clustering and a cost for the clustering. From the hierarchical clustering we make a dendrogram, which is a tree structure of the clustering. Each node(cluster) in the dendrogram is clustered with TC, and each node is a subset of its parent. For each node in the dendrogram we will randomize the similarities of the proteins in the node, and compare the actual cost vs. the random cost. Depending on the difference of cost this should indicate if the resulting clustering(split) is feasible or infeasible. If the split is determined as infeasible, the split is disregarded, and the actual node is returned as a cluster in the clustering result(Cutting the dendrogram such that the split did not occur). The resulting dendrogram will be a tree that possibly is cut on different levels. We call this dynamic tree-cutting. As mentioned above each protein family requires its own parameter. The idea with dynamic

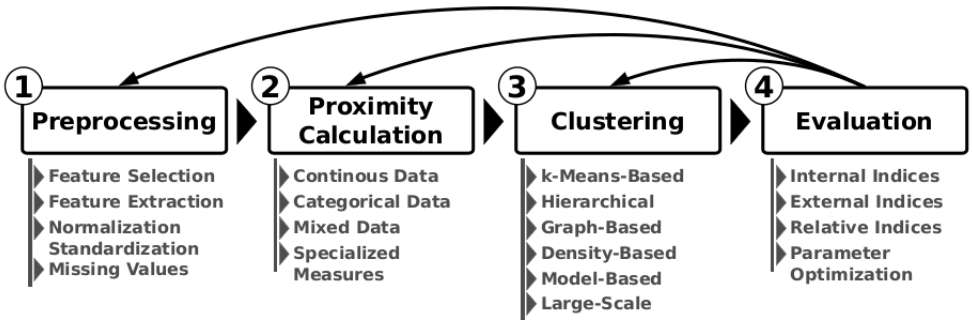

Figure 2: Clustering overview of a cluster analysis [6]

tree-cutting is that each branch will be cut when the parameter is optimal for the individual clusters. The final tree is then the optimal clustering.

1.1 Clustering Overview

A cluster analysis consists of 5 steps according to Jain et al. [13]. Figure 2 shows 4 out of the 5 steps. The first step which is "data abstraction" will not be discussed here and thus is excluded. The main source for the 'Clustering Overview' is [17]. All the steps are highly connected with each other. Meaning a suboptimal decision on the beginning possibly has severe effects for the overall clustering quality. From the overview, the researcher has to answer a series of questions [17]:

1. What are the relevant features?
2. Should the features be normalized?
3. What is the most appropriate proximity measure?
4. What is the most appropriate clustering tool?
5. How should the parameters of the clustering tool be set?
6. How can the result be evaluated?
7. How can this be done efficiently on massive datasets?

1.1.1 Preprocessing

A cluster analysis starts with the preparation of the data, which, in general, can be present in two forms [6]:

- One-mode: A matrix of size $n \cdot n$. One-mode derives from the fact that rows and columns represents the same thing, the objects. The matrix describes the between-object relationship, and may be a distance or similarity matrix.

- Two-mode: Describes a $n \cdot p$ matrix, where n is the objects and p the features that describes the objects. This is also referred to as raw-data. One of the main challenges is converting the two-mode into one-mode by means of a suitable proximity function. All clustering tools require a method of relating the objects to each other. Tools working directly with two-mode data has an internal mechanism to obtain the between-object relationship.

$$\mathtt{X} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & \dots \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & \dots & \dots & x_{np} \end{bmatrix}$$

Figure 3: Two-mode data matrix

Presented with raw-data the features may be of different types which are [17]:

- Quantitative features which can be divided into:
  1. Continuous values (e.g. fold changes of gene expressions)
  2. Discrete values (e.g. number of genes)
  3. Interval values (Timespan, 1-2 days)
- Qualitative features which can be divided into:
  1. Nominal or unordered (e.g., eye-color)
  2. Ordinal (e.g., qualitative evaluations of pain: "no pain", "some pain")
- Structural data; Repeated measurements of the same variable under different conditions, e.g., time-series data of gene expression

Selecting the proper set of features is crucial. A given feature can have missing values or even outliers which can disturb the data. Imagine a feature which is uniformly at random distributed over the entire dataset. Such a feature is useless for identifying groups. The feature actually disturbs the process, and might blur the clusters into each other. In the other end a feature can be highly correlated and not yielding any additional information, which can bias the proximity calculations. The following methods can be distinguished of deriving a suitable set of features [17]:

- Feature Selection: Describes the task of selecting the most informative features for the clustering task. The goal of feature selection is the removal of irrelevant, redundant or noisy features, which serves the following objectives: (1) improve the cluster performance, (2) reduce the dataset size, (3) learn about the importance of the features, or (4) a combination of them [9]. Usually feature selection is utilized for supervised learning. Considering the growing sizes of datasets it plays an important role for

unsupervised learning as well. Reducing the number of features can potentially reduce the computation time. One challenge for feature selection is finding a balance between reducing the dataset while maintaining the structural information in the dataset.

- Feature Extraction: Describes methods which creates entirely new features based on the original features. The goals are similar to that of feature selection, reduction of the feature space to only the most relevant features. One frequently used method is the principal component analysis (PCA) [14]. The idea is to transofrm the dataset to a reduced set of new uncorrelated features which retain most of the variation of the original dataset [14]. Another common technique is the multidimensional scaling(MDS) which attempts an embedding of the objects into a lower-dimensional space while preserving the pairwise distances between objects as best as possible. Both methods are often used for data visualization by reducing the feature space to two or three dimensions.

There exists many additional feature manipulation techniques, also methods which expands the feature space, feature expansion [10].

Normalization or standardization is another vital step to preprocessing. The different scales might have a strong influence on the similarity function. When using the Euclidean distance, a feature may have distances ranging [0, 1], which will almost be disregarded when other features are in the millions [10]. For which it might be crucial to normalize or standardize the features. The most common methods [19, 18]:

- Min-max normalization: Linear transformation of a feature to a predefined value range, normally between [0,1]

$$f' = \frac{f - \min_F}{\max_F - \min_F} \tag{1}$$

  $\min_F$ and $\max_F$ denotes the minimum and maximum of the feature F, and f represents some value of F.

- Autoscaling: Uses the standard deviation σ_F of the values of F and the mean $\bar{F}$ to scale the variable

$$f' = \frac{f - \bar{F}}{\sigma_F} \tag{2}$$

- Decimal scaling: Describes the shifting of the radix point such that the largest value in the dataset is smaller than 1:

$$f' = \frac{f}{10^j} \tag{3}$$

  where j is the smallest integer that that $\frac{\max_F}{10^j} < 1$.

A dataset may have missing values, which can derive from technical limitations, noise or error. Many similarity function cannot handle missing values, and removing incomplete objects or features may be an unacceptable solution. Treating missing values can be categorized as [8]:

- Prereplacing: Simply replace the missing values before usage. Methods to reconstruct can be the mean-and-mode method, nearest neighbor estimators or means of linear regression
- Embedded: Methods which attempts to handle missing values during the clustering process, thus allowing missing values in the dataset.

1.1.2 Proximity measures

Working with two-mode datasets, one important aspect of the cluster analysis is to establish a relationship between objects. The one-mode matrix is obtained via a proximity function, which either is a distance function or similarity function. Similarity function are maximal the more similar two objects are. The distance function are minimal the more similar two objects are. The choice of a suitable proximity function is highly dependent on the data types of the two-mode matrix. We will not go too much in depth with the following methods, but briefly mentioning for the sake of knowledge of existence.

- Continuous Data: The common approaches for dealing with continuous data [17]:
  - Euclidean distance: The most common distance measure, which corresponds to the distance between two points measured with a ruler. Let $u = \{u_1, \ldots, u_p\}, v = \{v_1, \ldots, v_p\} \in X$ be two objects with p features:
  $$d(u, v) = \sqrt{\sum_{i=1}^{p} (u_i - v_i)^2}$$
  - Minkowski Distance: Which is the generalization of the Euclidean distance:
  $$d(u, v) = (\sum_{i=1}^{p} |u_i - v_i|^l)^{\frac{1}{l}}$$
  with the parameter l. For $l = 2$ the Minkowski distance corresponds to the Euclidean distance. For $l = 1$ is corresponds to the Manhatten distance.
  - Correlation: This method accounts for linear relationship between features. The Pearson correlation coefficient is defined as:
  $$\Phi(u, v) = \frac{\sum_{i=1}^{p} (u_i - \bar{u})(v_i - \bar{v})}{\sqrt{\sum_{i=1}^{p} (u_i - \bar{u}^2) \cdot \sum_{i=1}^{p} (v_i - \bar{v})^2}}$$
  The values range between [-1, 1]. The Pearson correlation is highest when the two features are perfectly linearly correlated
- Categorical Data
  - Boolean Variables: When comparing two objects u and v there are in total four possible outcomes, depicted in Figure 4. a counts the number of positive matches. d the number of negative matches. b and c the number of mismatches. Most measures define the similarity

		Object v	
Object u	Feature	1	0
	1	**a**	**b**
	0	**c**	**d**

Figure 4: Outcomes for boolean variables [17]

between objects by relating the number of matching features, a and d, to the number of mismatches, b and c.

$$s(u, v) = \frac{a + d}{a + \lambda(b + c) + d}$$

For $\lambda = 1$ this corresponds to the Jaccard coefficient [12].

– General categorical variables: When dealing with categorical variables with more than two levels, different measures needs to be employed. One approach is deriving a score by counting the features for which the two objects agree:

$$s(u, v) = \sum_{i=1}^{p} \delta_k(u, v)$$

where $\delta_k(u, v)$ is the indicator function whether u and v are in the same category of feature k.

- Mixed Data Types: Describes a mixture of different data types, e.g., a mix of continuous and categorical features.
- Specialized Measures: Describes similarity functions which are specifically designed to a particular data type. In the fields of bioinformatics where a prominent one is NCBI BLAST [1].

The choice of proximity function is crucial to the cluster analysis in terms of the clustering quality. Unfortunately there are no best-performer for all cases.

1.1.3 Clustering Strategies

The following strategies will only be mentioned briefly due to the scope of the project

- k-means based: Algorithms which seek to identify an optimal k clustering.
- Hierarchical: Clustering method which creates an entire nested tree structure of the clustering. In order to receive a specific clustering the tree is cut at a certain height to produce k clusters. This method will be discussed in Section 2.3.
- Graph-based: Algorithm the represents the data internally as a graph. The objects corresponds to the nodes, and the edges to the similarities between objects. Transitivity Clustering is a graph based strategy which we will discuss in Section 2.2.

- Density-based: Approach which seeks to identify arbitrarily shapes clusters by separating high-density areas from low-density areas. Example of algorithms; DBSCAN and DENCLUE.
- Model-based: Clustering methods assume that the given dataset was generated by an underlying probabilistic method. THe aim is to maximize the model such that it best describes the observed data. Example of algorithms: Hidden-Markov-Model-based clustering, Self-Organizing Maps.
- Large-scale: Clustering methods in this category might belong to any of the above, but were developed to cope with large datasets. These are often running on distributed systems. Example of algorithms: BIRCH(Balanced Iterative Reducing and Clustering using Hierarchies), CURE(Clustering Using Representatives).

1.1.4 Clustering Properties

Whether a given tool is suitable or not cannot necessarily be derived solely based on the strategy of the clustering tool. The following properties should be considered given the compatibility of the clustering tool and the dataset [3, 25]:

- Scalability: Describes the behavior with increasing data set size in terms of runtime and memory consumptions.
- Dimensionality: The ability to handle the number of present features. Density-based clustering might be unable to discover any significant change of densities in data sets with very high dimensions, like gene expression data.
- Robustness: Describes the ability to cope with noise in the data. An algorithm which which reacts to small variations in the data with high variations in output may not be suitable. K-means is a tool which tend to react erratically in the presence of noise [26].
- Number of parameter and sensitivity: Describes the method's reliance on the user-input in tuning the algorithm. (1) The number of parameters should be as small as possible in order to avoid over-training of the algorithm. (2) It should be possible to anticipate the effect of the changing parameters, and not show near random effects on the clustering. E.g. small changes should not lead to big changes in the result. (3) The parameters should be interpretable. When the number of parameters increases it makes it difficult for a non-expert user.
- Arbitrary cluster shapes: The ability of the algorithm to identify arbitrarily shaped clusters.
- Usability and availability of the clustering tool is important. Most analyses are not carried out by computer experts, making a pure command-line tool error prone and could lead to maloperation.

1.1.5 Clustering Evaluation

One of the most critical steps in a cluster analysis is the evaluation of the clustering result. Clustering is a unsupervised learning without training data to guide, so how do we measure the quality? Evaluating two results can be difficult, if not impossible, when looking at plots of the clusterings. Thus, we need other methods to validate the results. `Cluster validity indices` are an important factor to evaluate solutions. There are three categories of validity indices [11]

- External validity indices : Evaluate the result with respect to a pre-specified structure, such as a gold standard.
- Internal validity indices : Evaluate the result with respect to information intrinsic to the data alone.
- Relative validity indices: Choosing the best clustering scheme of a set of defined schemes, according to a pre-specified criterion.

Internal measures are often based on

- Compactness: Measures how closely related the objects are in a cluster
- Separation: Measures how distinct or well-separated a cluster is from other clusters

Internal measures are e.g. Sum of Squares(SSQ), Silhouette Coefficient and Dunn Index. External measures use external information not present in the data. Normally a 'gold standard' is used. A gold standard is a clustering solution developed by experts, and are rarely available. In practice they are not available at all. Comparing against this ground truth is the way for determining the quality. External measures are e.g. F-measure and Jaccard coefficient. Generally, external validity indices evaluate a result in terms of the purity of individual clusters and the completeness of the clusters [24]. Relative will not be further discussed, but an overview can be found in [11]. There are many methods underlying these categories, but for the sake of brevity and the scope of the work, we limit ourselves to the F-measure. F-measure falls under the category of an external measure and will be discussed in Section 2.4.

Superfamily	Number of families
Amidohydrolase	29
Crotonase	16
Enolase	9
Haloacid dehalogenase	20
Vicinal oxygen chelate	17

Table 1: Overview of superfamilies and families in the Brown dataset

2 Background

2.1 The Data

We are using the Brown dataset [4], which consists of 866 proteins(which we will refer to as the 'big dataset'). The dataset consists of 5 superfamilies: Amidohydrolase, Crotonase, Enolase, Haloacid dehalohenase and Vicinal uxygen chelate. A superfamily consists of a subset of families. The 5 superfamilies can be divided into a total of 91 families. We will be looking at the dataset in two sizes. The 'small dataset' consists of the Amidohydrolase superfamily which consists of 232 proteins, distributed over 29 families. We will be using both the big dataset and the small dataset throughout the project. Table 1 shows an overview of the superfamilies and the number of families in the entire Brown dataset. The information on number of families can be useful when looking at the clustering results. Clustering the small dataset, Amidohydrolase superfamily, the number of clusters in the optimal result should the same, or close to, the number of families, 29.

2.2 Transitivity Clustering

In this project we focus on Transitivity Clustering(TC). TC is a well proven algorithm which performs well on various tasks [5], and is well suited for the plans we have. Before going into any details about TC we need some basic graph-theoretic definitions [22]:

Definition 1 (`Undirected simple graph`)**.** An `undirected simple graph` $G = (V, E)$ consists of a set of nodes V and a set of edges $E \subseteq \binom{V}{2}$, where $\binom{V}{2}$ denotes the set of two-element subsets of V. The edges are undirected and contains no self-loops or multiple edges between two nodes. uv is an unordered par $\{u, v\} \in \binom{V}{2}$.

Definition 2 (`Transitive graph`)**.** An undirected simple graph $G = (V, E)$ is called `transitive`

$$\text{if for all triples } uvw \in \binom{V}{3}, uv \in E \text{ and } vw \in E \text{ implies } uw \in E.$$

Definition 3 (`Weighted Transitive Graph Projection Problem(WTGPP)`)**.** Given a set of objects `V`, a threshold $t \in \mathbb{R}$, and a pairwise similarity function sim: $\binom{V}{2} \to \mathbb{R}$, the graph G is defined as

$$G = (V, E);\ E = \left\{ uv \in \binom{V}{2} : \text{sim}(uv) > t \right\} \tag{4}$$

The WTGPP is the determination of a transitive graph $G' = (V, E')$ such that there exist no other transitive graph $G'' = (V, E'')$ with $\text{cost}(G \to G'') < \text{cost}(G \to G')$. The modification costs are defined as

$$\text{cost}(G \to G') := \underbrace{\sum_{uv \in E \setminus E'} |\text{sim}(uv) - t|}_{\text{deletion cost}} + \underbrace{\sum_{uv \in E' \setminus E} |\text{sim}(uv) - t|}_{\text{addition cost}} \tag{5}$$

Transitivity Clustering takes one parameter, t, which is the threshold for similarities. Following is the steps in Transitivity Clustering [21]:

1. Model the given pairwise similarity, from the similarity matrix, as a similarity graph, G. The nodes corresponds to the objects, with weighted edges as the similarity values.

2. Transform the similarity graph, G, into another graph, G', by subtracting the threshold from the edge weights. Subsequently removing those edges with weights below zero, which is the deletion cost for equation 5.

3. Transform G' into a transitive graph, G'', with minimal cost. Thus, in this step we add all edges such that the graph is transitive, which is the addition cost for equation 5.

The resulting transitive graph, G'', is the clustering solution.

2.3 Hierarchical Clustering

Definition 4 (`Hierarchical Clustering(HC)` [17])**.** Let N be the number of objects in the dataset.
HC builds a nested structural partition of V, $H = \{H_1, \ldots, H_Q\}(Q \leq N)$, such that $C_i \in H_m, C_j \in H_l$, and $m > l$ imply $C_i \in C_j$ or $C_i \cap C_j = \emptyset$ for all $i, j \neq i, l = 1, \ldots, Q$.

There are two forms of hierarchical Clustering, `agglomerative` and `divisive`. Agglomerate starts with n clusters, where n is the number of objects in the dataset. Joining clusters until one cluster is remaining, where all n objects are members. Divisive is the opposite. Starting with one cluster with n objects. Splitting the clusters until all clusters are singletons. Thus, all n objects represents are cluster. Agglomerative is the most used, as divisive usually is a more expensive procedure. One important feature of hierarchical clustering is that a join or split of clusters are irrevocable, thus cannot be undone. Figure 5 shows an overview of agglomerative vs. divisive.

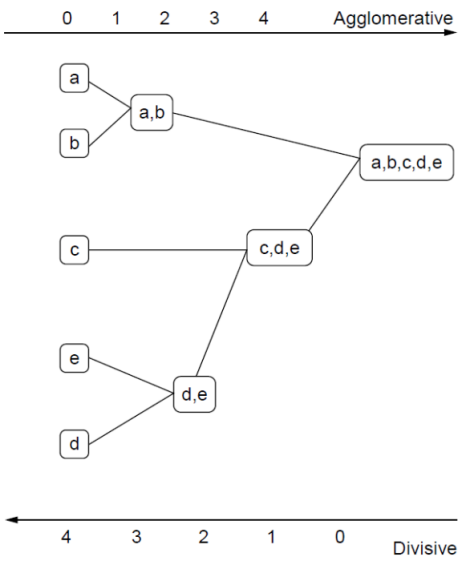

Figure 5: Agglomerative vs. Divisive: Reference Cluster Analysis, page 72

The steps of joins or splits is often showed as a `dendrogram`, which is viewed as a tree structure. The root of the tree is the cluster containing all n objects. Moving down the tree the nodes represents the clusters which was split from its parent. At the bottom of the tree we have the leafs, where the number of leafs represents all singleton clusters(the n objects). The tree can be cut a given height resulting in a clustering solution.

Figure 6 shows a `dendrogram` of the tree structure of a HC. The horizontal axis shows all the objects in the dataset. The vertical axis shows the distances between objects and/or clusters. Objects 1 and 2 are joined to a cluster at height 2. These are joined with the remaining objects at height 5, resulting in one cluster holding all objects.

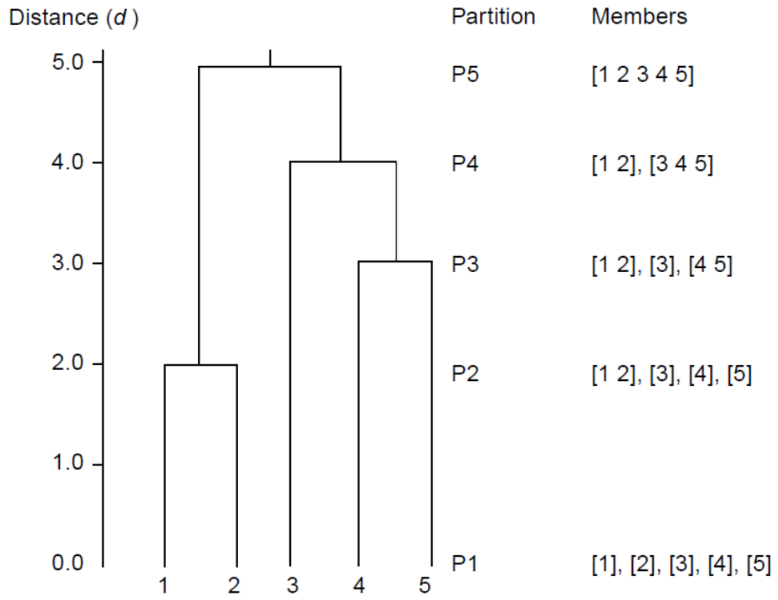

Figure 6: Dendrogram of a Hierarchical Clustering: Reference Cluster Analysis, page 75

In a typical hierarchical Clustering the size and number of clusters are given by a threshold, much like the one used in `TC`. Meaning that one iteration can possibly affect all clusters, which either increases or decreases the overall quality. In Figure 6 the tree could be cut between distance 3-4 to obtain a solution containing three clusters.

2.4 Cluster Validation

We are using the brown dataset, which has a gold standard available. The F-measure uses a gold standard to measure the quality of a clustering, and thus falls under the category of an external measure. There are multiple versions of the F-measure. We will be using the F1-measure, where the measures Recall and Precision is weighted equally. F1-measure is the quasi-standard in clustering evaluation and has been proved useful in many biomedical contexts [24]. From here on F1-measure will be stated as F-measure. Understanding the F-measure requires the following definitions [24]:
$K = (K_1, \ldots, K_m)$ = Clustering result obtained from the algorithm. K_i is the i'th cluster in K.
$G = (G_1, \ldots, G_l)$ = Gold standard clustering. G_j is the j'th cluster in G.
n = amount of objects in the dataset.
n_i = number of objects in cluster K_i.
n^j = number of objects in cluster C_j.
n_i^j = number of objects contained in $K_i \cap C_j$

Definition 5 (True Positive)**.** The number of common objects between cluster i and the compared gold standard cluster j.

$$\mathrm{TP} = |K_i \cap C_j| \tag{6}$$

Definition 6 (False Positive)**.** The number of objects in cluster i, which are not in the compared gold standard cluster j.

$$\text{FP} = |K_i \backslash C_j| \tag{7}$$

Definition 7 (False Negative)**.** The number of objects that are not in cluster i, which are in the compared gold standard cluster j

$$\text{FN} = |C_j \backslash K_i| \tag{8}$$

Definition 8 (Recall)**.**

$$\text{Recall}(i, j) = \frac{n_i^j}{n_i} = \frac{TP}{TP + FP} \tag{9}$$

Definition 9 (Precision)**.**

$$\text{Precision}(i, j) = \frac{n_i^j}{n^j} = \frac{TP}{TP + FN} \tag{10}$$

Definition 10 (F-measure for a cluster)**.** The general F-measure for cluster j and class i is given by:

$$F_\beta(i, j) = \frac{(1 + \beta^2) \cdot \text{Precision}(i, j) \cdot \text{Recall}(i, j)}{(\beta^2 \cdot \text{Precision}(i, j)) + \text{Recall}(i, j)} \tag{11}$$

As we are using the F_1-measure we can derive the following formula

$$F(i, j) = 2 \cdot \frac{\text{Precision}(i, j) \cdot \text{Recall}(i, j)}{\text{Precision}(i, j) + \text{Recall}(i, j)} \tag{12}$$

Definition 11 (F-Measure for a clustering)**.** In order to obtain the F-measure for a clustering solution, we need to find the mean F-measure. The F-measure of a cluster j is multiplied by the amount of objects in the gold standard cluster which have most in common objects. Take the sum over all clusters. Divide by total amount of objects in the dataset. Each cluster from the gold standard can only be referenced/mapped once.

$$\frac{\sum_{i=1}^{m} \text{F-measure}(K_i) \cdot n_i^j}{n} \tag{13}$$

A F-measure is between 0 and 1. A value near 1 indicates a good match with the gold standard(good clustering result). Values near 0 indicates a bad result. It is important to pay attention to the statement in Definition 11 saying that each cluster from the gold standard can only be referenced once. If, e.g. three clusters from K maps to the same clusters in C, two clusters will be neglected. The F-measure will be 0.0 for the neglected clusters, which negatively influences the quality. This scenario can happen in all clustering solutions, but the probability for multiple mappings on to the same gold standard cluster increases when $|K| > |C|$. In the opposite direction the scenario where $|K| < |C|$, also has a negative impact. With $|K| < |C|$ the clusters in K would be bigger than the cluster in C. Looking back at Definition 11, we see that the multiplication is done with the size of C_j. Therefore we can conclude that the amount of objects that K_i is bigger than C_j, $|K_i| - |C_j|$, has a smaller influence on the total F-measure than the remaining objects in K_i.

2.5 Multidimensional Scaling

Preserving proper relationships between-object similarities is difficult when randomizing data. The randomization does not take into account that the similarities have to make any sense. E.g. think of having a dataset with two clusters, c_1 and c_2. Let o_1 be the point in the center of c_1. The randomization has made the similarities between o_1 and c_1 remain the same, but similarities between o_1 and c_2 has made o_1 the center point of c_2, following the similarities, while c_1 and c_2 have a low similarity. One solution is to change the similarities making o_1 a point between the two clusters, which alters the between-object similarities for the entire dataset. One way to preserve the between-object similarity as best as possible is to increase the number of dimensions, using multidimensional scaling [24]. From the N-dimensional space of the dataset, the between-object distances can be calculated and a similarity matrix can be made. The resulting similarity matrix is the randomized version of the original data. One question that remains is what an appropriate number of dimensions would be, which will be covered in the Method section. The multidimensional scaling was done using a function in R, **isoMDS()** from the package **MASS**. The function requires as parameter a distance matrix, which is made from the similarity matrix with the Euclidean distance.

3 Method

3.1 Assessing the best clustering

Assessing the quality of both datasets a test have been run for both where we increment the threshold by one for each run. Starting threshold($t_{\min}$) was set to -1, since the lowest similarity is 0. The -1 is to ensure that the first run has all proteins in one cluster. The maximum used threshold($t_{\max}$) is highest similarity + 1. The maximum similarity is not the same for the datasets. The small dataset has $t_{\max} = 324$, the big dataset has $t_{\max} = 181$. For the small dataset a similarity file was given. For the big dataset a BLAST output was given, where the similarities could be extracted by taking the $-log_{10}()$ of the values. BLAST(Basic Local Alignment Search Tool) is the most widely used method to search for sequence similarities [1].

For the small dataset the optimal threshold was found to be 47 with a F-measure of 0.9816176. The same F-measure was given in the threshold range of 47-56. On [5] which has the quality measure for the same dataset, the optimal threshold was found to be 48.86 with F-measure 0.986. The measure from clusteval is approximately 0.004 higher than the implemented F-measure. The testing has only been with thresholds as integers, and not decimals. Testing with threshold 48.86 did not give any better F-measure. The F-measure has been debugged several times, and there was not found any errors in the calculations. The optimal threshold for the big dataset was found to be 17 with F-measure 0.7647134. As earlier mentioned the different families in sequenced based homology protein dataset requires different thresholds for the optimum clustering, as the optimal threshold for the datasets reveals. [20] Uses F_2-measure II, thus not exactly the same F-measure we are using. For the small dataset their findings was optimal threshold as 67 for the small dataset and 56 for the big dataset.

3.2 Transitivity-Hierarchical Clustering

The Hierarchical Clustering with TC as the basis was implemented. The resulting dendrograms are similar to those returned by the hclust() method in R, thus making the implementation feasible. The first version of HC incrementally went through the range of threshold going from one cluster to all singleton clusters. An incrementation value for threshold can manually be set. An improved version was made where it is possible to enable binary search for a "good" threshold. Whenever the amount of splits at the given node are too many, it will try to find a more suitable, lower, threshold. Where the range of allowed splits, and incrementation of threshold is set manually. The binary search allows incrementation of bigger steps, while taking into account that the splits are not too many, which could lower the quality of the splits. The resulting dendrogram can be seen in Section **??**. When testing with hclust() the agglomerative approach with *complete linkage* was used. We implemented Hierarchical Clustering with the divisive approach and TC as the clustering algorithm. These differences has an impact on the dendrogram, as they work differently. Due to the scope of the project we will not explain how *complete linkage* works.

3.3 Randomization

For the randomization of the similarity matrix we need to preserve symmetry. Self-similarity is never queried, leaving the diagonal irrelevant. Let D^i be the i'th dimension in the D-dimensional space, where $D^i \in \{x, y\}$ in the 2-dimensional space. Thus D^i is the column vector for all values for the objects in dimension `i`. D^i_{min} is the minimum value, and D^i_{max} is the maximum value found in D^i. We tested four different randomization approaches. In the results for each randomization approach we reason for the given approach and why it is feasible or infeasible. The reason for trying multiple randomization approaches is closely related to the results of the former approaches. The number indicates the version of the approach, where 1 is the first we tested, and 4 is the last. The following are descriptions of all tested randomization approaches. Approach 2 through 4 are using multidimensional scaling.

Approach 1:

1. Fill the upper matrix with randomly selected values from the actual similarity matrix. Diagonal is set to 0's.
2. Make the matrix symmetric. The result is the random similarity matrix.

Approach 2:

1. Convert the actual similarity matrix into distances with the Euclidean distance.
2. Scale from 2 dimensions into N dimensions with the isoMDS() method. Returned is the points in the N dimensional space.
3. For all elements in D^i randomly select a value in range $D^i_{\text{min}} - D^i_{\text{max}}$.
4. Calculate the between-object distances.
5. Conversion from distances to a similarity matrix, which is the resulting randomized similarity matrix.

Approach 3:

1-5 Same steps as in approach 2.

6 Replace the lowest similarities in the matrix with 0's, matching the number of 0's in the actual similarity matrix.

Approach 4:

1-5 Same steps as in approach 2.

6 Make an ordering of both matrices in ascending order.

7 Replace the i'th ordered element in actual similarity matrix with the i'th ordered element in the random similarity matrix, on the respective index in the random similarity matrix. Let [5,7] be the index of the i'th ordered similarity in random matrix. The value on index [5,7] will be replaced with the i'th ordered value from the actual similarity matrix.

8 Result is a randomized similarity matrix with the exact similarity distribution as the actual matrix.

3.4 Results

3.4.1 Transitivity-Hierarchical Clustering

Figure 7 shows the dendrogram using the standard hierarchical clustering in R, hclust(). Figure 8 is the dendrogram from the implemented hierarchical clustering using TC as basis. A key note is that we are using the divisive approach where we move from one cluster to all singleton clusters. Therefore we are starting at the top of the dendrogram and moving downwards. Looking closer at both of the dendrograms we can see similar branches. For instance to the right in Figure 7 we see a large partition, this is one of the protein families with 100 proteins. In the left side of Figure 8 we notice a partition of the same size. The two partitions are in opposite sides, meaning that the ordering of the proteins on the x-axis are not the same. This does not influence the overall dendrogram as long as there are no overlapping branches. If there are any overlapping branches the ordering is suboptimal, since it makes the dendrogram more confusing. Looking more closely on both of the dendrograms we can find similar partitions, even though the splits are not on the same height. The big dataset has almost 4 times the proteins as the small dataset, making the branches of the dendrogram very dense. The dense branches makes it more difficult to analyze. The dendrograms for the big dataset are Figures 9 and 10. The extension of TC into Hierarchical Clustering looks feasible from the resulting dendrograms. The difference in the dendrogram is most likely due to the underlying algorithms. The method hclust()(the standard implementation in R) uses the agglomerative approach with *complete linkage* for the merging of objects and clusters. We are using the divisive approach with TC as the clustering method.

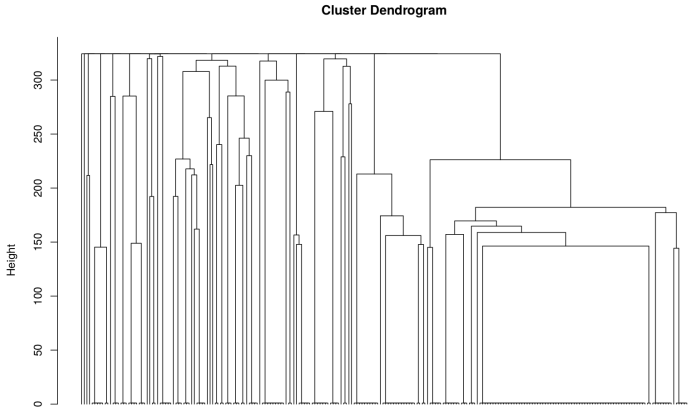

Figure 7: Resulting dendrogram with the small dataset - using the standard hclust() method in R

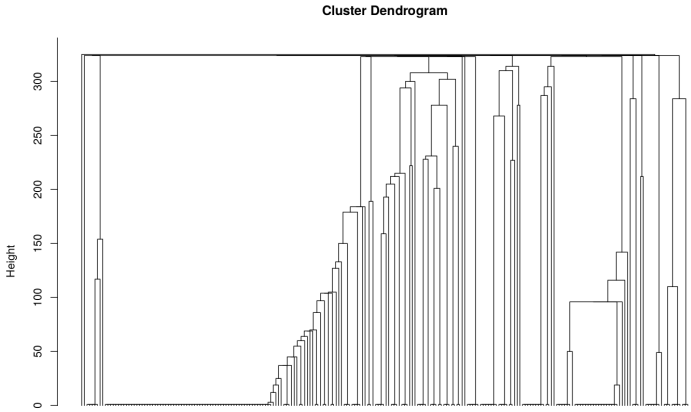

Figure 8: Resulting dendrogram with the small dataset - Implemented hierarchical clustering with TC

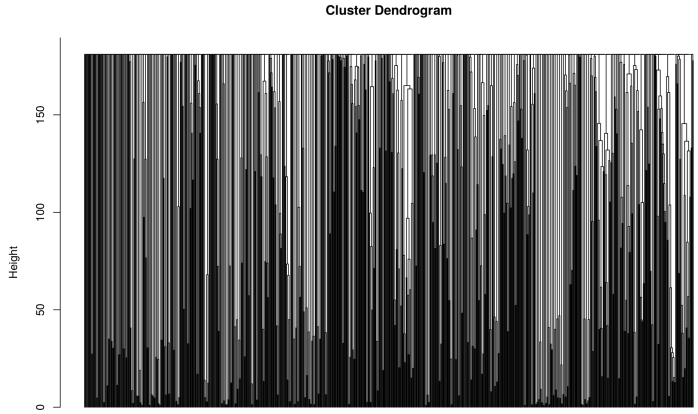

Figure 9: Resulting dendrogram with the big dataset - using the standard hclust() method in R

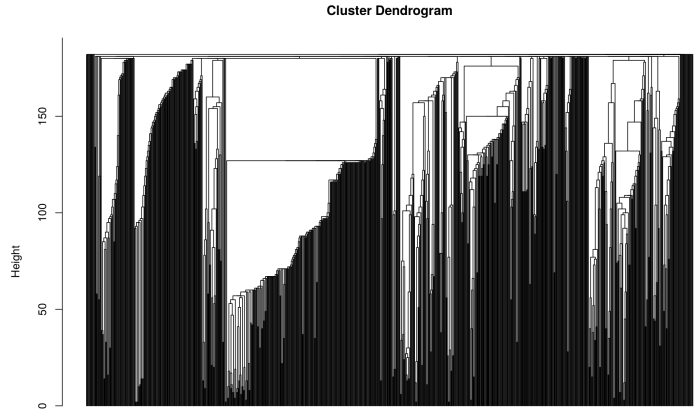

Figure 10: Resulting dendrogram with the big dataset - Implemented hierarchical clustering with TC

3.4.2 Randomization

As mentioned in Section 3.3 we have tested four different randomization approaches where the results are in the following sections. For the actual datasets the costs for the different threshold in the range of $t_{\min}$ to $t_{\max}$ is given by figures 11 and 12. As we can see the small dataset peaks at approximately 60,000. The big dataset peaks at approximately 680,000. Each dataset has a $t_{\min}$ and a $t_{\max}$. These were defined for the small and big dataset in Section3.1. For the randomized dataset they correspond to:

- $t_{\min}$: The lowest similarity in the dataset where we subtract 1. For all datasets it has been found $t_{\min} = -1$.
- $t_{\max}$: The highest similarity in the dataset where we add 1. No guarantee for this to be the same as for the small or big dataset, depending on which was randomized. Although $t_{\max}$ for the random big dataset seems to be the same as for the actual data, which the cost plots will show in the the following sections.

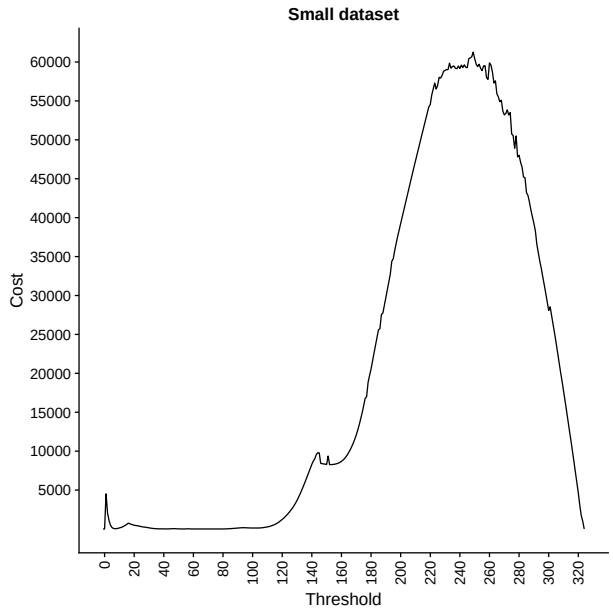

Figure 11: Clustering costs returned by TC

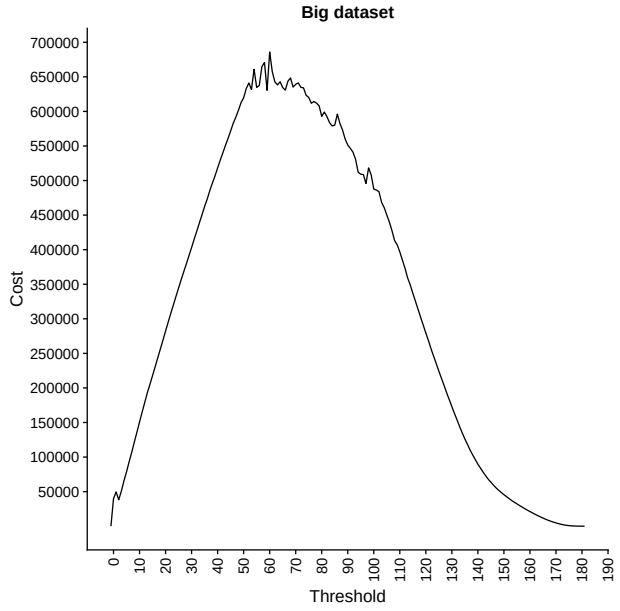

Figure 12: Clustering costs returned by TC

3.4.3 Randomization Approach 1

Requiring the cost for the randomized dataset the range $t_{\min}$ to $t_{\max}$ was tested with incrementation of 1. This approach turned out as misleading, as the costs for the random data peaked at 1,000,000, which is far from the actual data of 60,000. Multiple randomizations was tested, leaving the same high costs above 1,000,000. Reasons for the high costs are due to the between-object similarities, as discussed in the section Multidimensional Scaling. There is no structure in the random data, which is the cause of the high costs. As the results was misleading for the small dataset the big dataset was not tested.

3.4.4 Randomization Approach 2

The costs for approach 1 was tremendously high compared to the actual data, which lead to a randomization utilizing multidimensional scaling. Multidimensional scaling was used to increase the number of dimensions. In this higher dimensional space, the points was randomized, where we derived similarities. This approach was tested with dimensions in the range of 5-100 with steps of 5. In order to obtain the costs for the randomized data TC was run with threshold in the range of $t_{\min}$ to $t_{\max}$. For the small dataset we randomized once for each dimension, giving a sample size of 20 runs. The costs are peaking in the range 190,000-320,000 across the range of dimensions, see Figure 13. As this is an improvement compared to approach 1, it was tested with the big dataset. For the big dataset we randomized four times for each dimension, resulting in a sample size of 80 runs. The costs for the small dataset has decreased significantly. Here the peaks of costs are in the range of 2,000,000-3,100,000, shown in Figure 14. For both datasets the costs of the random dataset are in the range of 3-6 times higher than the actual data. For all the tested dimensions the biggest gap is where the cost for the random data is peaking, or very close to the peak. This is shown in figures 15 and 16. The influencer of the biggest gap is simply where the cost is peaking in the randomized data. Therefore we cannot derive any useful information with this approach. Utilizing multidimensional scaling has not improved the randomization well enough in terms of the structure of the data.

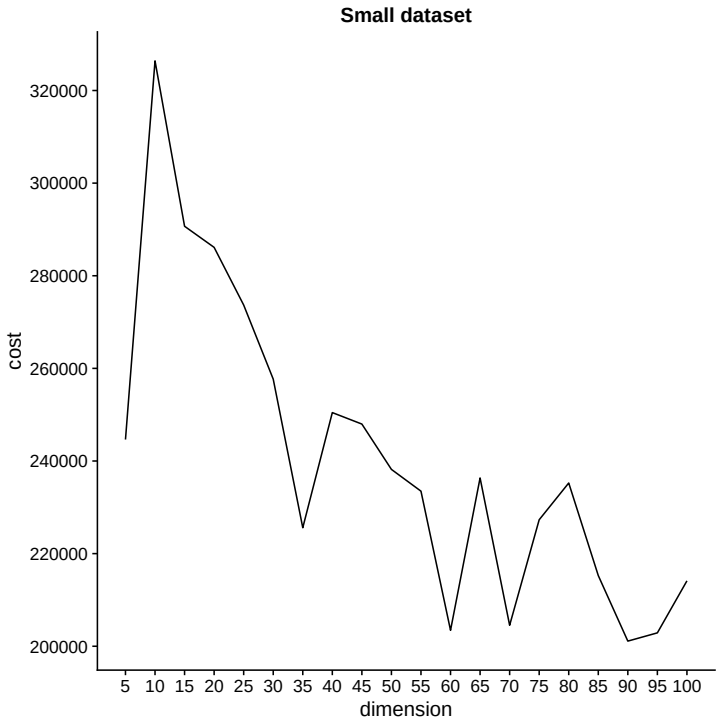

Figure 13: Randomization approach 2: Max cost distribution across dimensions. The costs returned by TC with threshold in the range of $t_{\min}$ to $t_{\max}$. The threshold was incremented by 1 for each run.

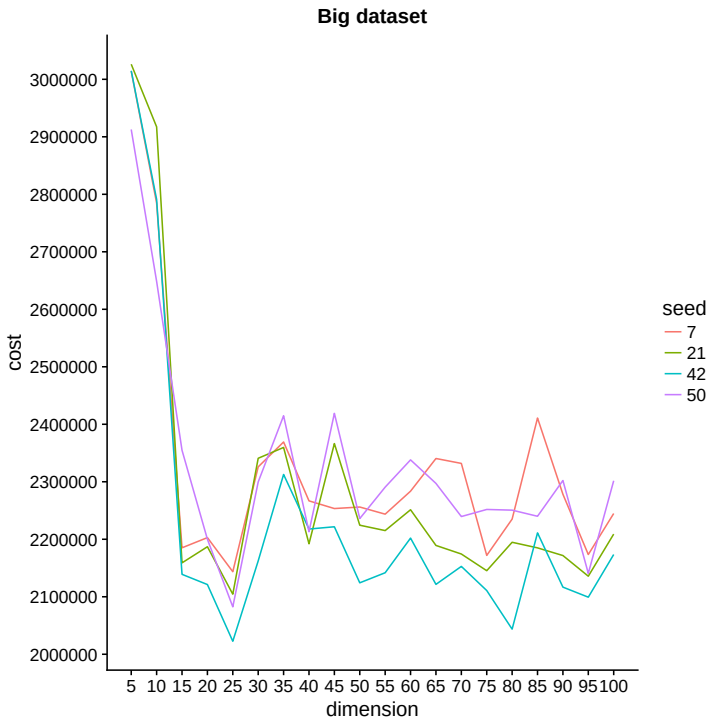

Figure 14: Randomization approach 2: Max cost distribution across dimensions. The costs returned by TC with threshold in the range of $t_{\min}$ to $t_{\max}$. The threshold was incremented by 1 for each run.

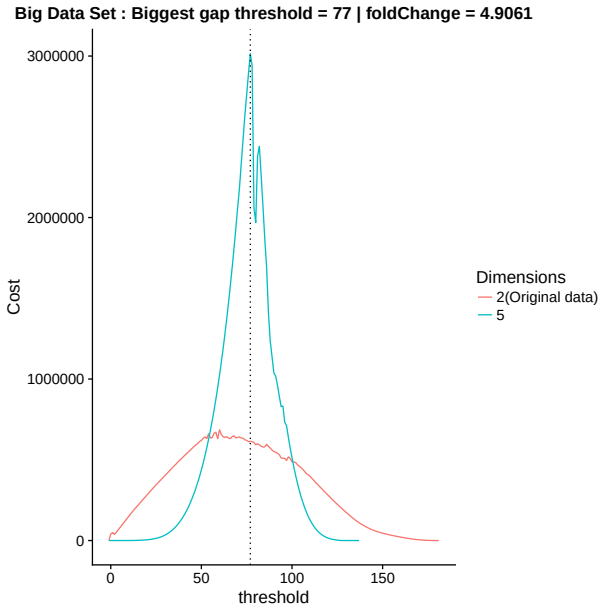

Figure 15: Max gap cost

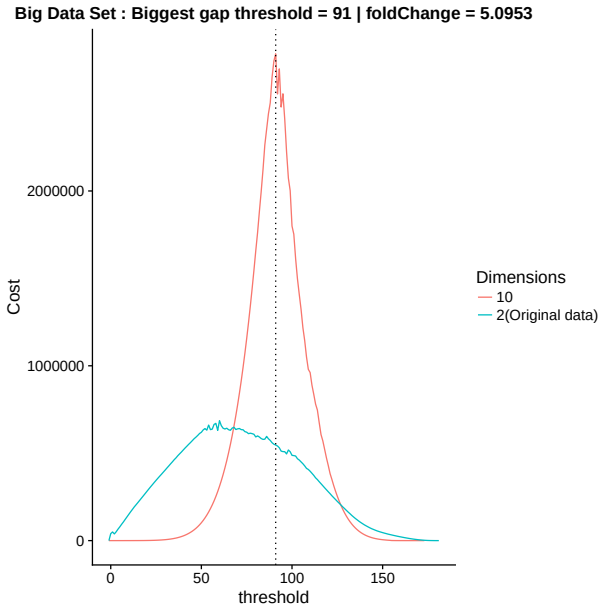

Figure 16: Max gap cost

3.4.5 Randomization Approach 3

The costs for the randomized data was obtained the same way as we did with approach 4. From the results of testing many different dimensions with approach 2, we concluded it was enough to test the dimensions 2, 3, 4, 5 and 10 from here on. Each dimension was randomized four times, giving a sample size of

20 runs. Figure 17 shows the distribution of the highest cost across the tested dimensions. The distribution looks better than the one in approach 2, Figure 14. The cost is starting in the lower end with the lower dimensions. From dimension 30 the cost stays within a range of 500,000. Looking at Figure 18 it shows similar results as approach 2, where the biggest gap is at the peak of the random data. In 16/20 the biggest gap is at the highest peak of the random data. The remaining 4 tests has the biggest gap outside of the peak of the random data. Figure 19 shows a promising result. The max gap is at threshold 17, which is the exact same threshold which gave the best clustering quality according to the F-measure. One thing to notice is that this result is from a random dataset with 2 dimensions. For the remaining 3 that has the biggest gap outside the peak, the gap is at thresholds 9(2 dimensions), 10(5 dimensions) and 14(2 dimensions). Although this approach looks looks promising, we wanted to try another approach. Hitting the biggest way from the peaking cost in 4/20 of the tests is not by any means an impressive statistic. If this statistic would hold, we need to randomize and run TC on a given dataset 5 times. Including the run for the actual data, thus a total of 6 clusterings.

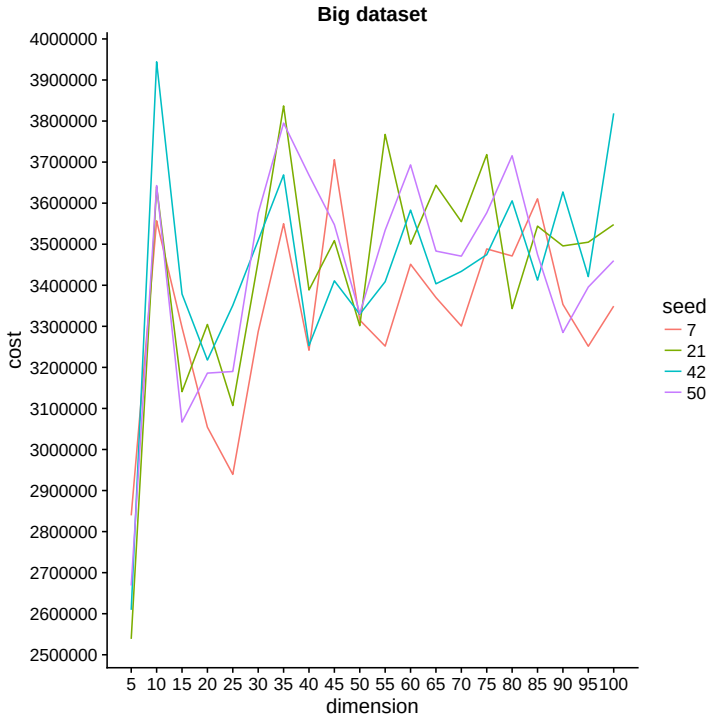

Figure 17: Randomization approach 3: Max cost distribution across dimensions

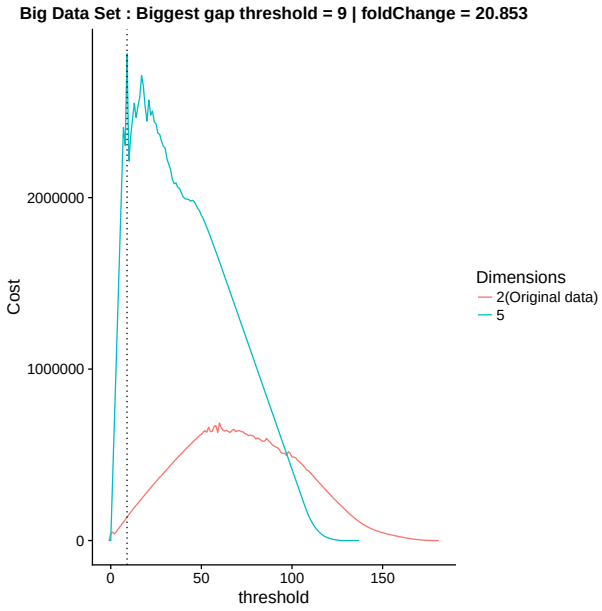

Figure 18: Max gap cost

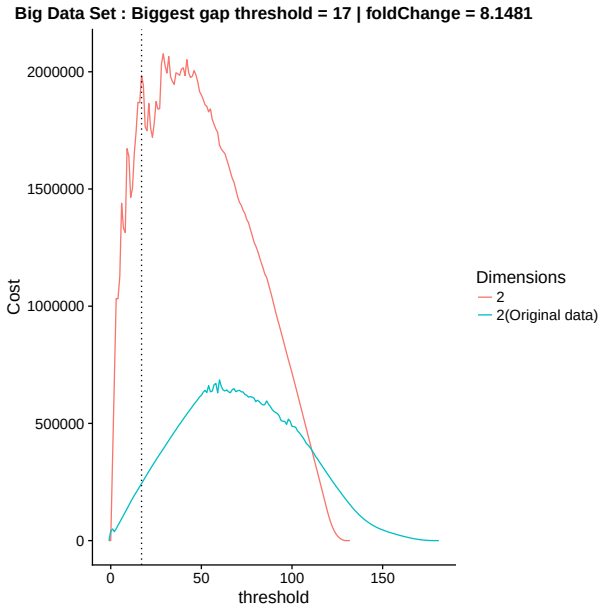

Figure 19: Max gap cost

3.4.6 Randomization Approach 4

This approach was tested with same number of randomizations and dimensions as approach 3. The max gap was found on dimensions 2, 3, 4, 5 and 10, which each was randomized 4 times. Figure 20 shows the distribution of costs across the dimensions, and it looks more promising than approach 3. This is kind of

expected as this randomization was made from the exact same similarities as the actual data, in a random ordering. Figures 21 and 22 shows the most promising results. The thresholds that has the biggest gap are 20 and 15, which is close the the optimum of 17. Noticing the dimensions are 2 and 3. These trials has the opposite statistics as approach 3. Approach 3 had 16/20 where the gap was found at the peak of cost, and 4/20 the gap was found away from the peak. With approach 4 we have 16/20 where the biggest gap is found away from the peak, and 4/20 where the gap is directly on the peak of cost. Although we do not hit the exact same threshold, 17, which had the overall best clustering quality, this approach seems promising.

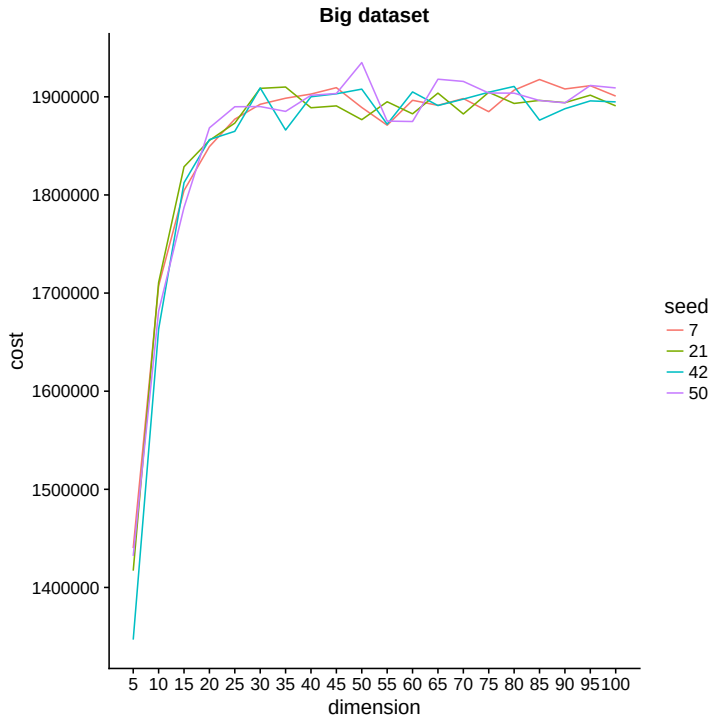

Figure 20: Randomization approach 4: Max cost distribution across dimensions

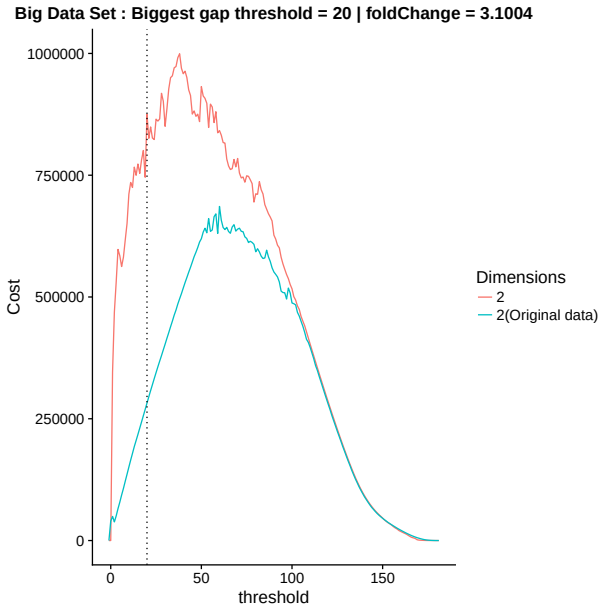

Figure 21: Max gap cost

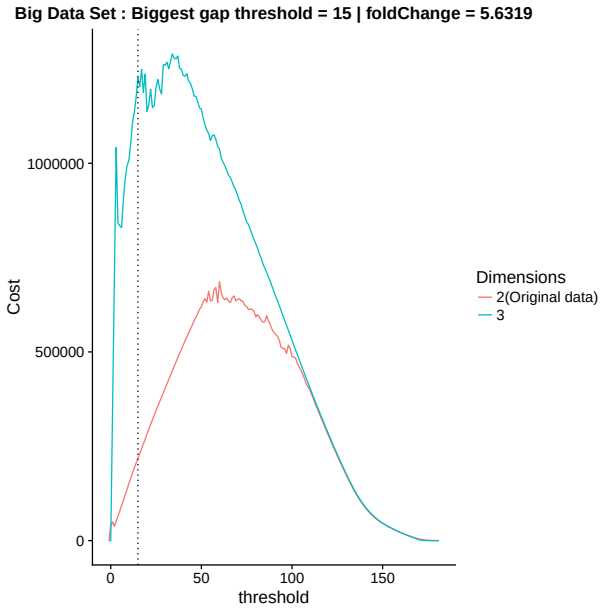

Figure 22: Max gap cost

From the 4 tested randomization approaches both 3 and 4 looks promising. Approach 3 has a result which hits the exact same threshold as the optimal clustering result with threshold 17. Approach 4 comes close to this threshold. The closest for approach 4 was threshold 15 and 20. One thing to notice is the number of dimensions on these results which was 2 and 3. At this point it is difficult to say which approach is better than the other. Approach 3 and 4 was

each tested 4 times with 5 different dimensions, which is a small sample size. It requires a bigger sample size in order to determine consistency and properly compare the randomization approaches. Considering the statistics for the given trials where the peak was away from the peaking cost, approach 4 seems to be the more promising. Approach 3 and 4 are viewed as feasible randomization. Approach 1 and 2 are not considered as feasible randomizations at this point.

3.4.7 Randomization and Transitivity-Hierarchical Clustering

As both approach 3 and 4 looks promising at this point, these are used to assess the max gap cost during the hierarchical clustering. Looking at the max gap cost at each node in the dendrogram should somehow reveal if the split will increase of decrease the quality of the clustering. If the indication is that the quality will decrease if the given node is split, the branch should be cut and returning this node as a cluster in the clustering result. Let $n_{i,j}$ be node i at height j in the dendrogram. If the returned clustering cost from TC is higher than the randomized version of the same node, it indicates the actual node has a worse structure than the randomized. The structure of the randomized node is worse than the actual node, if the cost of the randomized is higher. How to interpret this in terms of a dynamic tree-cutting approach and finding a pattern in which the cuts can be read is unknown at this point. Approach 3 and 4 was each tested 9 times on the small dataset, and 9 times on the big dataset. The tests includes both binary search and non-binary search of the hierarchical clustering. The non-binary search was tested 3 times with step as 1 and 3. The binary search test was tested with step 30, and range of allowed splits was 2-10. The same parameters was used on the big dataset. On the following plots the cost axis is made with logarithmic scale as the variables (lines) are scaled differently. The cost at a certain threshold does not necessarily only involve a single cluster/node. If three different nodes are clustered at, e.g., threshold 10, the clustering costs is the sum of costs for all three nodes. Remind that for each node the cost of the randomized will be found as well. Meaning the difference in cost at a given threshold is the difference between the clustering cost of the three nodes and the clustering cost of the three randomized nodes. On the following plots the difference is scaled into 4 different variables. 'actual' is the direct cost value difference, as as the absolute value. 'avg' is the total difference averaged over the number of involved proteins. 'avgPairs' is the difference over the amount of pairs for the involved proteins. 'foldChange' is the fold change between the actual cost and the randomized cost. As the plots are using logarithmic scale it was not possible to show negative values, thus taking the absolute value. The conversions from a negative to positive fold change is indicated by black points. If the fold change was negative then the randomized data had a clustering cost lower than the actual clustering cost. Often one of the clustering costs are 0, which would set the fold change to ∞, where the fold change is manually set to 0.001.

Finding the difference in cost for each node in the dendrogram, THC was tested with the settings in Table 2.

Figures 23 through 28 shows the difference in cost for approach 3 and 4 for the small and big dataset. What to interpret from these plots, or even if they show any valuable information at all is the next step in the project. One way to

Rand. approach	Dimensions	Seed	Steps	Binary Search	Allowed splits
3	5	7, 21, 42	1, 5	No	-
3	5	7, 21, 42	30	Yes	2-10
4	5	7, 21, 42	1, 5	No	-
4	5	7, 21, 42	30	Yes	2-10

Table 2: Settings for finding the biggest cap at each node in the dendrogram with THC and randomizations. The randomizations scaled the data up to 5 dimensions. Each seed counts as a randomization of the data, $2 \cdot 3$ randomizations for each approach on the given data. This was done on both the small dataset and the big dataset. Steps is the incrementation of threshold for every iteration. Binary Search states if the test was run with binary search on the threshold, until number of splits are in the range of allowed splits.

improve the plots would be taking the average for each approach, where these plots are for a single run. Due to the scope of the project it was not possible to gain any further knowledge. It is needed to analyze the plots, deriving any patterns indicating a cut. Altering the hierarchical clustering algorithm making it possible to make the cuts on branches, and returning the resulting clustering. To determine the quality the F-measure is calculated and the clustering can be validated. The procedure of analyzing the plots and finding any valuable patterns, or information, in terms of the cost difference is the on going process. With the proper information or pattern the dynamic cut of the tree can be derived returning the optimal clustering. If there is no valuable information to gain from the plots, another approach to analyzing the difference in cost needs to be considered, or maybe another randomization approach. Thus the next steps in the project would be:

- Analyze the plots for the difference in cost at each node in the dendrogram. Goal is to find a pattern or anything that indicates where to cut the branches in the dendrograms, in order to obtain a tree which is dynamically cut on different levels. The resulting tree is the optimal clustering.
- Alter the algorithm for THC, such that we can cut the branches at different levels. Depending on the difference in costs it makes a decision for the cut of branches, and a clustering can be returned.
- The clustering result from THC will be compared to the gold standard to gain a quality measure of the clustering.
- Analyzing the plots and try different approaches on where to cut is trial-and-error. This step goes on until we find this approach to parameterless clustering by dynamic tree-cutting either feasible or infeasible.

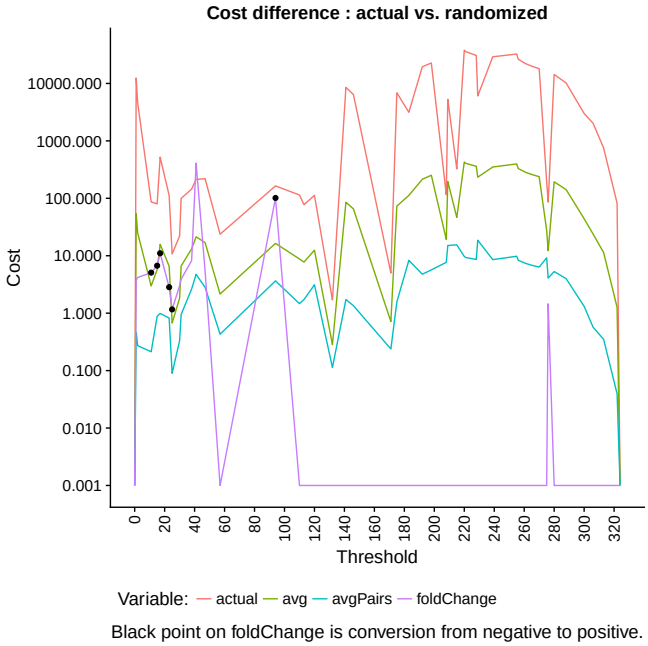

Figure 23: Small data, approach 3, step = 1, non-binary search

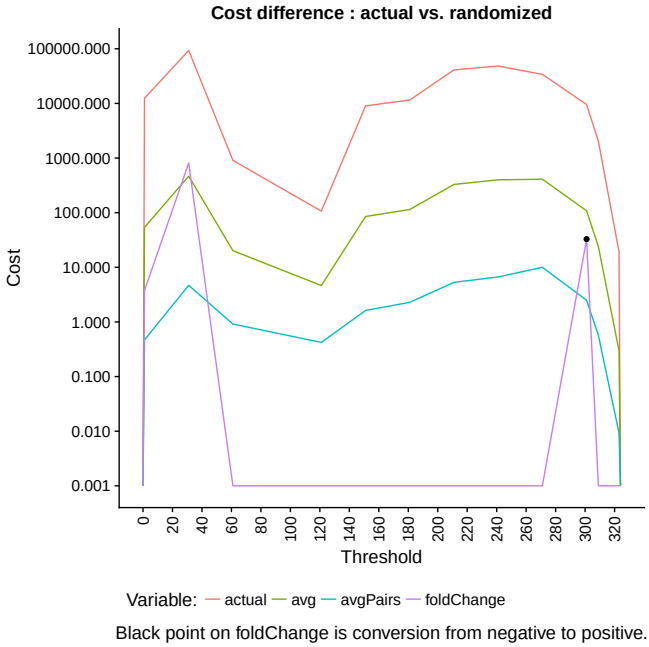

Figure 24: Small data, approach 3, step = 30, binary search, allowed range of splits 2-10

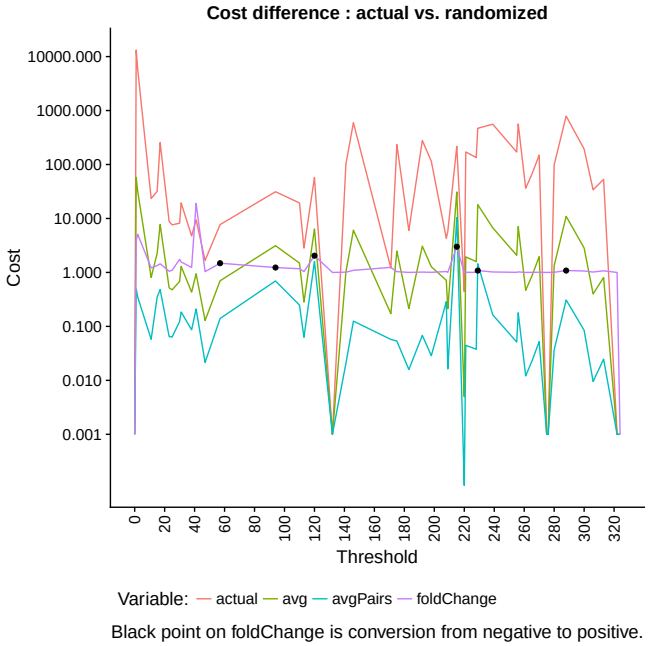

Figure 25: Small data, approach 4, step = 1, non-binary search

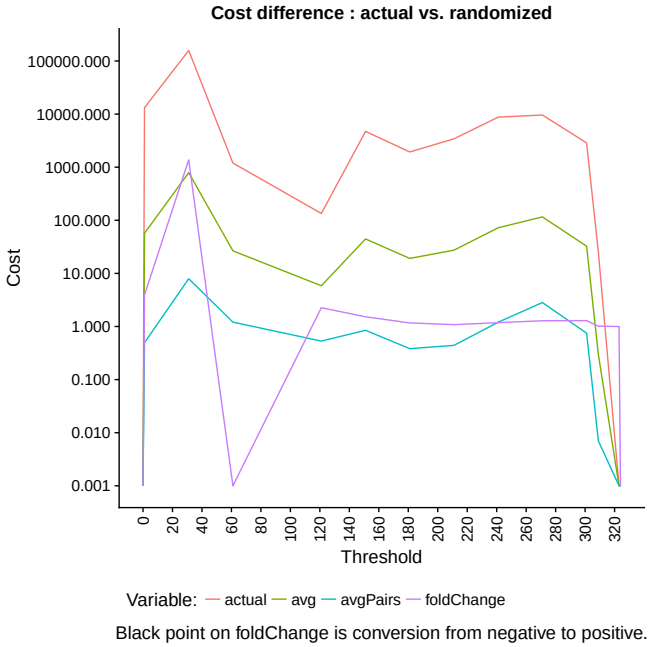

Figure 26: Small data, approach 4, step = 30, binary search, allowed range of splits 2-10

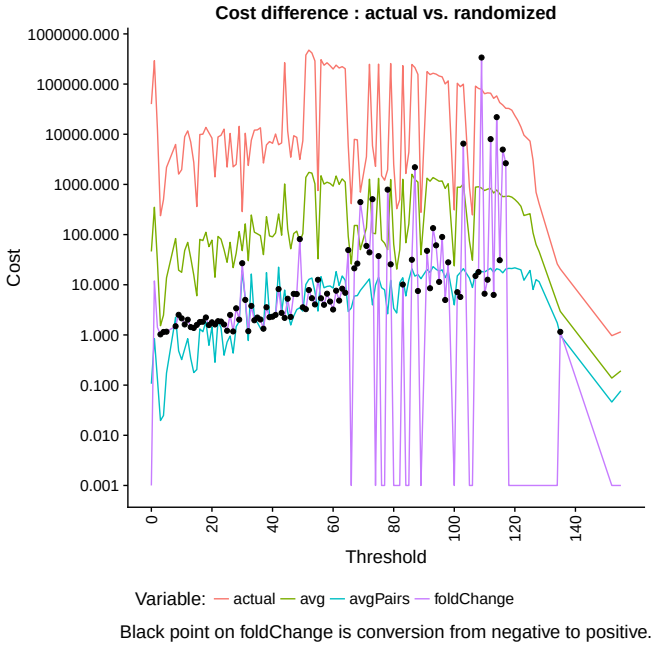

Figure 27: Big data, approach 3, step = 1, non-binary search

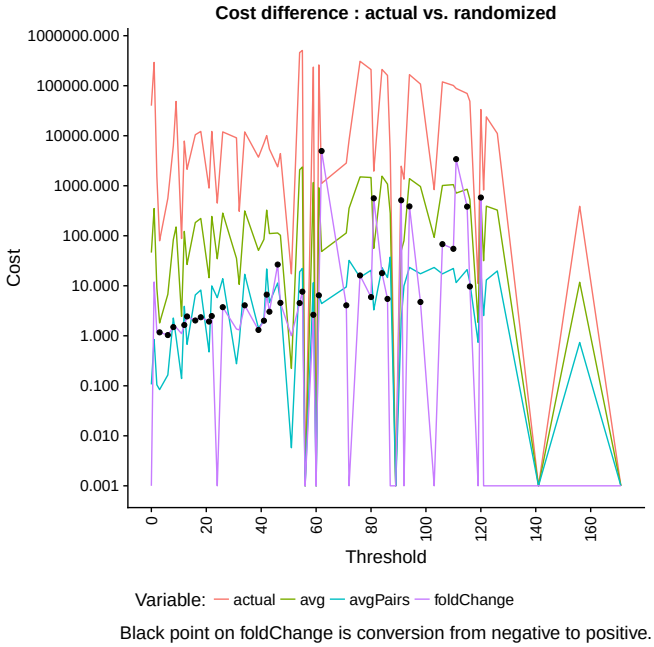

Figure 28: Big data, approach 3, step = 30, binary search, allowed range of splits 2-10

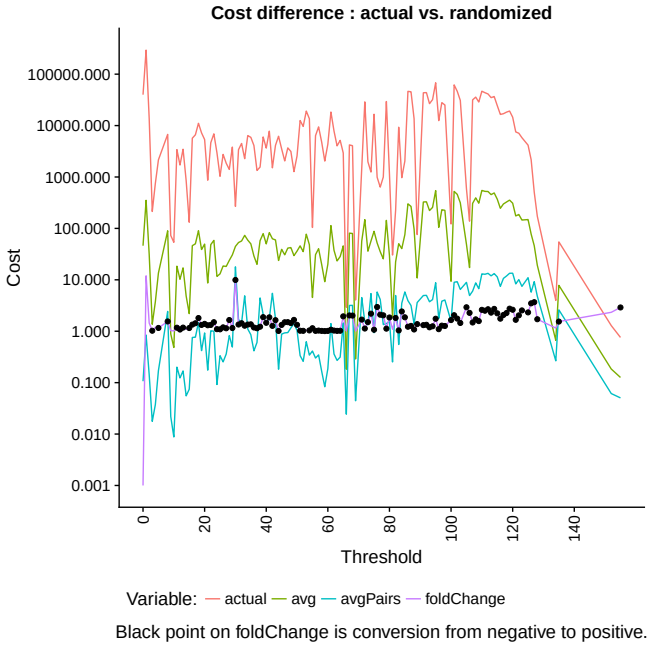

Figure 29: Big data, approach 4, step = 1, non-binary search

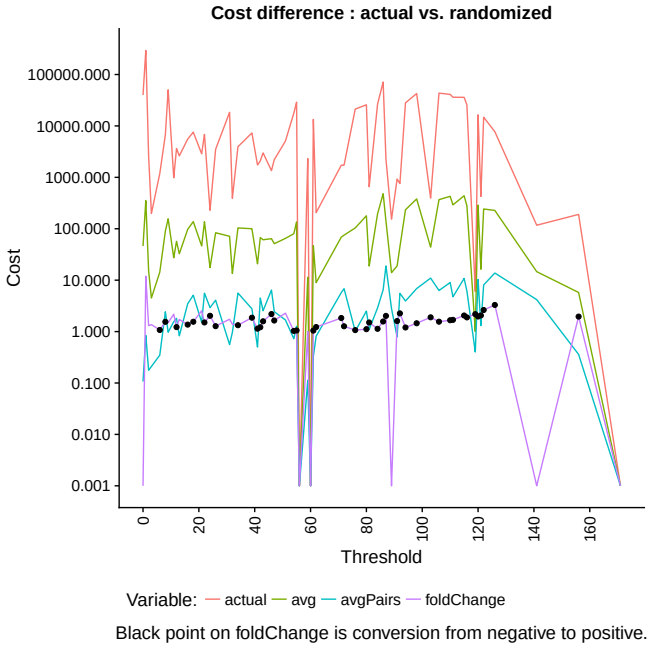

Figure 30: Big data, approach 4, step = 30, binary search, allowed range of splits 2-10

4 Implementation

R was the chosen language for the project as it eases the work around matrices, also has rather easy customization for producing plots. RStudio was the used IDE. R has a lot of built-in methods which eased the coding process, as not everything had to be invented from scratch. Also, it is possible to install different package for the suiting of your needs. Transitivity Clustering was given as an R package. Hierarchical Clustering is already implemented in R, meaning that we could use the plotting features for the dendrograms. From the package MASS we used the isoMDS() method for the multidimensional scaling. From the package 'stats' the method cmdscale() was used in correlation with isoMDS(). All coding and testing has been done on a Linux Mint environment, with Intel Core I7 and 32GB of RAM.

5 Conclusion

References

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403-410, 1990.

[2] M. R. Anderberg. *CLuster analysis for applications: probability and mathematical statistics: a series of monographs and textbooks*, volume 19. Academic press, 2014.

[3] B. Andreopoulos, A. An, X. Wang, and M. Schroeder. A roadmap of clustgering algorithms: finding a match for a biomedical application. *Briefings in Bioinformatics*, 10(3):297-314, 2009.

[4] S. D. Brown, J. A. Gerlt, J. L. Seffernick, and P. C. Babbitt. A gold standard set of mechanistically diverse enzyme superfamilies. *Genome Biology*, 7:R8, 2006.

[5] W. Christian, R. Röttger, and J. BaumBach. Comparing the performance of biomedical clustering methods. *Nature Methods*, 2015.

[6] B. S. Everitt and S. Landau. *Cluster Analysis*, volume 5th edition. Wiley, 2011.

[7] R. D. Finn, P. Coggill, R. Y. Eberhardt, S. R. Eddy, J. Mistry, A. L. Mitchell, S. C. Potter, M. Punta, M. Qureshi, A. Sangrador-Vegas, and et al. The pfam protein families database: towards a more sustainable future. *Nucleic Acids Research*, 44, 2016.

[8] G. Gan, C. Ma, and J. Wu. *Data clustering, theory, algorithms, and applications*, volume 20. Siam, 2007.

[9] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3, 2003.

[10] I. Guyon and A. Elisseeff. *Feature extraction*. Springer, 2006.

[11] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *Journal of Intelligent Information Systems*, 17:107-145, 2001.

[12] P. Jaccard. Nouvelles recherches sur la distribution floral. *Bull. Soc. Vard. Sci. Nat*, 44:223-270, 1908.

[13] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM computing surveys*, 31:3, 1999.

[14] I. Jolliffe. Principal component analysis. *Wiley Online Library*, 2002.

[15] V. Kunin, I. Cases, A. J. Enright, V. d. Lorenzo, and C. A. Ouzounis. Myriads of protein families, and still counting. *Genome biology*, 4, 2003.

[16] M. L. Metzker. Sequencing technologies - the next genetation. *Nature reviews genetics*, 11:31, 2010.

[17] R. Röttger. Clustering of biological datasets in the era of big data. *Integrative Bioinformatics*, 13(1):300, 2016.

[18] L. A. Shalabi, Z. Shaaban, and B. Kasasbeh. Data mining: A preprocessing engine. *Journal of Computer Science*, 2(9):735-739, 2006.

[19] K. Thangavel and N. K. Visalakshi. Impact of normalization in distributed k-means clustering. *International Journal of Soft Computing*, 4(4):168-172, 2009.

[20] T. Wittkop, D. Emig, S. Lange, S. Rahmann, M. Albrecht, J. H. Morris, B. Sebastian, J. Stoye, and J. Baumbach. Partitioning biological data with transitivity clustering. *Nature methods*, 7, 2010.

[21] T. Wittkop, D. Emig, A. Truss, M. Albrecht, S. Böcker, and J. Baumbach. Comprehensive cluster analysis with transitivity clustering. *Nature Protocols*, 6:3, 2011.

[22] T. Wittkop, S. Rahmann, R. Röttger, S. Böcker, and J. Baumbach. Extension and robustness of transitivity clustering for protein-protein interaction network analysis. *Internet Mathematics*, 7:4, 2011.

[23] C. Wiwie, J. Baumbach, and R. Röttger. Comparing the performance of biomedical clustering methods. *Nature methods*, 12(11):1033-1038, 2015.

[24] C. Wiwie and R. Röttger. On the power and limits of sequence similarity based clustering of proteins into families. *Pacific Symposium on Biocomputing*, 2017.

[25] R. Xu and D. Wunsch. Clustering algorithms in biomedical research: a review. *IEEE Rev Biomed Eng*, 16(3):645-678, 2005.

[26] K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzzo. Model-based clustering and data transformations for gene expression data. *Bioinformatics*, 17(10):977-987, 2001.