

Charles University in Prague
Faculty of Mathematics and Physics



MASTER THESIS

Jakub Kratochvíl

Dimension Reduction Techniques in Morphometrics

Department of Software and Computer Science Education

Supervisor: RNDr. Josef Pelikán

Study programme: Information Technology

Specialization: Software systems - Computer graphics

Prague 2011

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. I understand that my work relates to the rights and obligations under the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 paragraph 1 of the Copyright Act.

In date..... signature

Název práce: Aplikace redukce dimenze v morfometrii

Autor: Jakub Kratochvíl

Katedra: Kabinet software a výuky informatiky

Vedoucí diplomové práce: RNDr. Josef Pelikán, KSVI

Abstrakt: Tato práce se zabývá aplikací metod redukce dimenze v antropologii a morfometrii. Zejména se soustřeďuje na nelineární metody redukce dimenze. Práce zavádí nový postup nazývaný multipass redukce dimenze. Ukážeme, že pomocí multipass redukce dimenze lze vylepšit výsledky klasifikace a snížit počet dimenzí nutných pro klasifikaci pomocí klastrování. Klíčová slova: Redukce dimenze, morfometrie, locally linear embedding, multidimensional scaling

Title: Dimension Reduction Techniques in Morphometrics

Author: Jakub Kratochvíl

Department: Department of Software and Computer Science Education

Supervisor: RNDr. Josef Pelikán, Department of Software and Computer Science Education

Abstract: This thesis centers around dimensionality reduction and its usage on landmark-type data which are often used in anthropology and morphometrics. In particular we focus on non-linear dimensionality reduction methods - locally linear embedding and multidimensional scaling. We introduce a new approach to dimensionality reduction called multipass dimensionality reduction and show that improves the quality of classification as well as requiring less dimensions for successful classification than the traditional singlepass methods.

Keywords: Dimensionality reduction, morphometrics, locally linear embedding, multidimensional scaling

Special thanks

I would like to thank my mother for supporting me during my studies. Furthermore, I would like to thank my supervisor Josef Pelikán for outstanding guidance. Finally, I would like to thank all members of the Morphome3cs team for excellent collaboration and team spirit.

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 7 |
| 1.1 | Previous Work | 7 |
| 1.2 | Motivation and Goal of the thesis | 10 |
| 1.3 | Outline of the Thesis | 11 |
| 2 | Dimensionality Reduction | 13 |
| 2.1 | Feature selection | 14 |
| 2.2 | Feature extraction | 14 |
| 2.3 | Principal Component Analysis | 15 |
| 2.4 | Locally Linear Embedding | 17 |
| 2.5 | Multidimensional scaling (MDS) | 20 |
| 3 | Specifics of landmark data | 23 |
| 3.1 | Generalized Procrustes Analysis (GPA) | 23 |
| 3.2 | Euclidean Distance Matrix Analysis | 25 |
| 4 | Clustering | 28 |
| 4.1 | Hard and fuzzy clustering | 28 |
| 4.2 | Cluster validity and selecting the number of clusters | 29 |
| 4.3 | Hierarchical clustering | 29 |
| 4.4 | Partitional clustering | 30 |
| 5 | Comparison of traditional algorithms | 32 |
| 5.1 | Description of the dataset | 32 |
| 5.2 | Impact of preprocessing and dimensionality reduction | 35 |
| 5.3 | Choosing the right dimension | 38 |
| 6 | Multipass dimensionality reduction | 54 |
| 6.1 | Description of the algorithm | 54 |
| 6.2 | Comparing the algorithms | 56 |
| 6.3 | Impact of the clustering algorithm | 59 |
| 6.4 | Selection of Target Dimension | 62 |

| | |
|---|-----------|
| | 6 |
| 6.5 Impact of the LLE k-parameter | 64 |
| 6.6 Summary | 70 |
| 7 Challenges and open problems | 71 |
| 8 Conclusion | 72 |
| A Morphome3cs | 81 |
| A.1 Overview and aim of the Morphome3cs project | 81 |
| A.2 Users of the Morphome3cs platform | 81 |
| A.3 Statistical computation in Morphome3cs | 82 |
| B Installing Morphome3cs | 84 |
| C Reproducing the results | 86 |
| C.1 GUI-based scripts | 86 |
| C.2 Scripts without GUI | 89 |
| D Note on the used configuration | 92 |
| E Contents of the DVD | 93 |

1 Introduction

In the 1960s, the hard disk of a mainframe would have a capacity of several megabytes. Nowadays, a personal computer' one can store several Tera bytes of data. However, simply storing data is hardly beneficial. In order to achieve progress, data must be processed into meaningful information.

Yes, we are aided in this endeavor by the powerful hardware. Nevertheless, interpretation, analysis and visualization of computations is still a challenging task. In order to simplify this task numerous methods for data mining were developed. Among the most often used ones are neural networks, dimensionality reduction, clustering and support-vector machines.

One such field, where the above mentioned methods are used, is the field of morphometry. Morphometry or morphometrics is a compound word of morpho (Latin for shape or form) and metrum (Latin for to measure). Hence morphometry focuses on study and measurement of shape and form differences between biological samples. One of the common tasks for morphometrical analysis is determining the extent of sexual dimorphism among the samples. This means establishing the differences between the male and female species from the dataset. Other possible task is classifying the members of the dataset according to their species.

This thesis focuses on usage of data mining methods in the field of morphometry. In particular, we will focus on dimensionality reduction. In the introductory part of the thesis, we will discuss previous work on dimensionality reduction and its usage in the field of morphometry. Subsequently, we will define the goal of the thesis. Based upon this goal, we will describe the structure of the thesis.

1.1 Previous Work

In this section, we will describe the previous work in field of dimensionality reduction. We first survey the linear dimensionality reduction methods and then focus on non-linear methods. We underline the impact of dimensionality reduction in the field of geometric morphometry and anthropology.

PCA and Linear Dimensionality Reduction

In this part of the introduction, we focus on Principal component analysis and its applications. We also briefly mention kernelPCA. In particular, we emphasize the role of PCA in both outline-based and landmark-based morphometry.

This section does not contain mathematical description of the PCA algorithm. This can be found in section 2.3. Detailed description of the GPA algorithm is presented in section 3.1.

Dimensionality reduction is a rapidly developing area of research. Initially the focus centered predominantly around linear methods for dimensionality reduction such as principal component analysis (PCA) which was first described in [33]. A further interest in the principal component analysis was sparked by the introduction of kernelPCA. KernelPca transforms the originally linear PCA method into a nonlinear method by computing the PCA transformation in a Hilbert space and is introduced in [39].

PCA and kernelPCA are used in various areas of research such as graph drawing, text recognition, data mining etc. Harel and Koren in [10] use dimensionality reduction in the field of graph drawing. The main advantage of this type of graph drawing algorithm is that it provides a quantifiable measure of output quality. Unlike other algorithms for graph drawing such as simulated annealing high-dimensional embedding does not converge towards local minima. It is also fast and allows for drawing of large graphs.

In the field of anthropology and morphometry, PCA is extensively used. In landmark-based analysis PCA is often used in tandem with Generalized Procrustes Analysis. In fact, Julien in [15] claims that it has become the default method of operation in morphometrical analysis. In particular, the analysis of the Kendall Shape Space is very thorough. We refer the interested reader to Kendall's article [18] on eigenshape analysis.

Other category of methods used in morphometrical analysis are methods based on outline analysis. PCA is also extensively used in combination with these methods. Various methods for outline representation were suggested. For an initial survey of outline based methods, Julien's monograph [15] on

Morphometrical methods in R is a good starting point. A common combination is using principal component analysis with Elliptic Fourier Descriptors. For example Berg et al. in [3] use an approach based on Fourier Descriptors in segmentation of femur from computer tomography images. For a detailed survey of Elliptic Fourier Descriptors (EFD), you may also refer to Hwang's and Lina's article [24] on EFD.

As the performance of computers increases the focus has of morphometrical analysis has shifted towards three-dimensional meshes and volumetric data rather than two-dimensional images. Hutton, Buxton and Hammond in [12] use principal component analysis for registration of facial scans. They use PCA in combination with thin-plate spline deformation to model the triangular meshes. The final registration of the meshes is then performed using an iterative closest point search in combination with an active shape model.

Non-Linear Dimensionality Reduction

This section centers around the description of previous work in the field of non-linear dimensionality reduction. We focus on multidimensional scaling and locally linear embedding. Furthermore, we describe the applications of these methods in the field of anthropology. This section does not focus on the mathematical of the presented methods. The interested reader should refer to section 2.4 for an explanation of the locally linear embedding algorithm. Multidimensional scaling is explained in section 2.5.

One of the oldest of non-linear dimensionality reduction methods is multidimensional scaling (MDS), introduced by Kruskal in [19]. While its use in the field of morphometry and anthropology is not as widespread as that of PCA, multidimensional scaling is commonly used in the field of morphometry. For example Chiu et al. in citeCHIU use MDS for analysis of snail shells. Multidimensional scaling is especially useful when studying shape variation. It is mostly used as a landmark based method rather than for outline processing. Christensen in [5] uses multidimensional scaling in combination with principal component analysis in order to determine the geographical variation of lodgepole pines.

Non-linear dimensionality reduction has started to gain popularity at the end of the last millennium. This can be connected with the introduction of both kernelPCA and Locally Linear Embedding (LLE) in [37]. Since then many variants of LLE were proposed such as kernelised LLE or laplacian eigenmaps (LEM). For a detailed overview of these techniques please refer to Kayo's dissertation [17] on the applications of LLE. Kayo also concludes that LLE offers the best classification rate of the non-linear methods. However, no discussion of the target dimensionality parameter is undertaken.

In the field of anthropology, usage of locally linear embedding is rather sparse. It predominantly focuses on its usage in supervised learning. This is based upon the supervised LLE algorithm suggested by de Ridder and Duin in [36]. Zhang and Chau then apply this methodology and further enhance it for classification of plant leaves in [40]. Other similar methods are proposed by Lee and Chen in Classification for Leaf Images. For a detailed explanation of this algorithm please refer to [20]. Both of those algorithms are based on outline analysis rather than landmark based approach.

Furthermore, S. Kadoury in his master thesis [16] uses traditional LLE in the area of face recognition in combination with support vector machines. However, this study is rooted in image analysis rather than the field of anthropology.

1.2 Motivation and Goal of the thesis

Motivation

Having explored the previous work on dimensionality reduction in the field of anthropology we can conclude that non-linear methods are underrepresented. Especially the usage of locally linear is rather scarce. Using locally linear embedding on landmark-type data is relatively untested.

One of the main disadvantages of non-linear dimensionality reduction methods (such as Multidimensional scaling or Locally Linear Embedding) is that the target dimension is an explicit parameter of the algorithm. As a result, the user is faced with blindly choosing and testing multiple target dimensions before a reasonable output is obtained. While the optimal choice

of the target dimension for the classification algorithm was briefly discussed in the above mentioned articles a thorough analysis was never conducted.

Goal of the thesis

Based on the above mentioned facts the goal of this thesis is thus to combine existing clustering and non-linear dimensionality reduction methods and apply the resulting methods in the field of anthropology. The main contribution of the thesis is analyzing and determining the effect of dimensionality reduction on the quality of the clustering output.

In particular, we will introduce a new approach called multipass dimensionality reduction and show its main advantages and disadvantages as a preprocessing algorithm for k-means clustering. Moreover, we will provide experimental evidence that shows that multipass dimensionality reduction decreases the number of dimensions required for successful classification.

1.3 Outline of the Thesis

In order to achieve the above stated goal, we will adopt the following approach: In the first section, we discuss the problem of dimensionality reduction. We first define the problem of dimensionality reduction and distinguish between feature extraction and feature selection. We then survey both existing linear and non-linear dimensionality reduction methods.

In the subsequent part, we discuss the specifics of anthropological data. We introduce various types of datasets (landmark-data, Fourier descriptors, trimeshes) that come up in the field of anthropology. In particular, we focus on the usage of Euclidean Distance Matrix Analysis (EDMA) and Generalized Procrustes Analysis (GPA) for registration of landmark data.

The next section centers around the problem of clustering. We distinguish between hierarchical and partitional clustering. We focus on the partitional algorithms and describe the k-center algorithm in greater detail.

Having described and analyzed the main relevant methods, we switch our attention to practical application of these methods. At first, we focus on

showing the importance of the dimensionality reduction process as a preprocessing algorithm for the clustering process. We show that dimensionality reduction improves the success rate of the clustering process. We then focus on the issue of choosing of the target dimension for the dimensionality reduction algorithm in order to maximize the success rate and reduce the variability of the clustering process.

We then introduce the process of multipass dimensionality reduction and compare its results with the basic singlepass approach from the previous chapter. We will show that this method further improves the quality of the clustering. In particular, we will provide practical evidence that the multipass method improves the performance of the dimensionality reduction algorithm at target dimension two. Therefore the user is not forced to blindly determine the target dimensionality.

This thesis builds upon the Morphome3cs project. In the appendix, we will therefore briefly discuss the design and architecture of this software. In particular, we will briefly focus on the technological issues of using the R library in C# programs.

2 Dimensionality Reduction

In this part, we introduce the problem of dimensionality reduction and provide mathematical definition for it. Furthermore, we explain the difference between feature selection and feature extraction. Then we will focus on the particular methods for dimensionality reduction. We will cover the following methods:

- Principal component analysis (PCA)
- Locally Linear Embedding (LLE)
- Multidimensional scaling (MDS)

In modern data analysis, problems with a high number of features per object are becoming increasingly more popular. These problems are especially important in bioinformatics and multimedia analysis. In the field of anthropology, specimens are usually characterized by a set of multiple landmarks. Each of those landmarks represents a two or three dimensional data point. Since the number landmarks per specimen is usually over ten we quickly arrive to over 20 features per specimen. In such situations, it is often beneficial to reduce the dimensionality of the data (describe it in using features) in order to improve the efficiency and accuracy of data analysis.

Cunningham in [6] cites the following main reasons for dimensionality reduction:

- The identification of a reduced set of features that are predictive of outcomes can be very useful from a knowledge discovery perspective.
- For many learning algorithms, the training and/or classification time increases directly with the number of features.
- Noisy or irrelevant features can have the same influence on classification as predictive features so they will impact negatively on accuracy.

On the whole, dimensionality reduction techniques facilitate the interpretation of results and allow us to visualize these results in a user-friendly way.

In mathematical terms, the problem of dimensionality reduction can be stated as follows: given the p -dimensional feature vector $x = (x_1, \dots, x_p)$, find a lower dimensional representation $s = (s_1, \dots, s_k)$, where $k < p$, that best preserves the content of the original data with respect to a certain criterion function f . For a detailed explanation refer to [7].

In the following section we will describe the main approaches to dimensionality reduction. The two main broad sets of methods revolve around:

- Feature selection
- Feature extraction

2.1 Feature selection

Feature selection attempts to find a subset of the original feature vector that best represents the original data. Expressed formally this means for a feature vector $x = (x_1, \dots, x_p)$ find $s \subset 1 \dots p, |s| = k$, such that $f(x_s)$ is maximal.

2.2 Feature extraction

Feature extraction first transforms the original feature space into a different lower dimensional target space. The two main variants of feature extraction are linear and non-linear feature extraction. Linear techniques result in each of the components of the new variable being a linear combination of the feature vector:

$$s_i = \sum_{j \in 1 \dots p} w_{i,j} x_j.$$

Or in matrix terms: $s = Wx$, where $W_{k \times p}$ is a linear transformation matrix.

Non-linear dimensionality reduction methods are methods that do not have such a representation. An example of such method is locally linear embedding (LLE), which will be described in greater detail in the following sections.

2.3 Principal Component Analysis

Principal Component Analysis (PCA) is the most commonly linear dimensionality reduction method. In this section, we show how to find the principal components and explain their mathematical significance.

Algorithm Overview

First, let us explain how to compute PCA. Let X be a dataset consisting of n m -dimensional observation.

$$X = \{x_1, \dots, x_n\}, x_i = \{x_{i1} \dots x_{im}\}$$

We begin by centering the observations. This is accomplished by computing the mean and subtracting it from the data.

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

$$\bar{X} = X - \bar{x}$$

Next, we compute the covariance matrix C of the centered dataset.¹

$$C = \frac{\bar{X} * \bar{X}^T}{n}$$

Now we are ready to compute the principal vectors. In order to do this, we perform eigen-value decomposition of the covariance matrix C .

$$V^{-1}CV = D$$

Here, D is a diagonal matrix containing the eigenvalues of C . V contains the eigenvectors that correspond to these eigenvalues. Finally, we project the dataset into the target dimension using a subset $V_q, q \subset 1 \dots m$ of the eigenvectors.

¹Sometimes it may be useful to normalize the data with respect to unit variance but it is not always necessary.

$$X_{red} = X * V_q$$

On the whole, the algorithm for PCA looks as follows:

- Center the dataset.
- Normalize the data to unit variance. (optional)
- Compute the covariance matrix
- Find the eigenvectors and eigenvalues of the covariance matrix
- Using the eigenvectors from previous, find a projection of the dataset to target dimension

Mathematical Significance of Principal Components

In this part, we will show that the transformation computed by PCA is a orthogonal transformation that maximizes variance.

Let X be a *centered* dataset with n observations. We will show based upon [14] that the first eigenvector w_1 of the covariance matrix XX^T is the unit vector that maximizes variance. In mathematical terms we are attempting to solve the following problem:

$$w^* = \mathbf{argmax}_{|w|=1} \mathbf{Var}\{(w^T X)\}$$

This is equal to:

$$w^* = \mathbf{argmax}_{|w|=1} \frac{\sum_{i=1}^n (w^T x_i - w^T \bar{x})^2}{n}$$

Since the dataset was centered ($\bar{x} = 0$), we can simplify the term to:

$$w^* = \mathbf{argmax}_{|w|=1} \frac{\sum_{i=1}^n w^T x_i^2}{n}$$

Or in matrix terms:

$$w^* = \mathbf{argmax}_{|w|=1} \frac{(w^T X)(w^T X)^T}{n}$$

In order to find the maximum, we use Lagrangian multipliers. The system we are trying to solve looks follows:

$$\begin{aligned} ww^T &= 1 \\ 2XX^T w^T - 2w^T \lambda &= 0 \end{aligned}$$

The second equation can be simplified to:

$$w^T(XX^T) = \lambda w^T$$

From this we see that indeed the first eigenvector of the covariance matrix XX^T is the solution of the system, while λ is equal to the first eigenvalue of XX^T . We refer the reader to [14] for additional details. The other principal components are orthogonal to the previous components and solve the following equation:

$$\begin{aligned} X_k &= X - \sum_{i=1}^{k-1} w_i w_i^T X \\ w_k &= \mathbf{argmax}_{|w|=1} \{\mathbf{Var}(w^T X_k)\} \end{aligned}$$

In other words, after removing all the variance explained by the previous principal components, the next component is a projection that maximizes variance in the residual dataset. For a more detailed explanation refer to Jolliffe's monograph [14] on PCA.

2.4 Locally Linear Embedding

In this part, we will focus on locally linear embedding (LLE). LLE is a non-linear dimensionality reduction technique. First we will describe the algorithm and subsequently we will explain the basic properties of the method. We will base our description on Saul's and Roweis's introductory articles [38] and [37]).

It should be noted that while originally used as an unsupervised technique recent research has adapted LLE for supervised usage. S. Kaudury uses LLE as a preprocessing algorithm for training of a support vector machine. For a

detailed description of the algorithm please refer to [16]. A different approach was taken by D. de Ridder and R. Duin. The authors in [36] modify the original LLE algorithm so that it includes the classification of the features. Then they used this algorithm for clustering.

However, the focus of this thesis lies predominantly on the improvement of unsupervised classification.

Algorithm

Locally linear embedding is based upon reconstruction of the original data using linear combination of each data points nearest neighbors. Let us consider a dataset X consisting of m n -dimensional feature vectors x_1, \dots, x_n .

The first step of the algorithm is to find k -nearest neighbors (other feature vectors) of each feature vector. K is a user-defined parameter of the algorithm. The metric we use to determine the nearest neighbors of the feature vectors is not critical to the algorithm and it may be interesting to explore using exotic metrics. However, this is outside of the scope of the thesis.

Let $N_{k,i} \subset 1 \dots n$ be indices of k -nearest neighbors of the feature vector x_i and let $X_{k,i}$ be the k -nearest neighbor feature vectors of x_i

The next step of the algorithm is to find a weight matrix W that minimizes the following error term:

$$\Delta(W) = \sum_i |X_i - \sum_{j \in N_{k,i}} W_{ij} X_j|^2$$

We are trying to find such a weight matrix that best represents the original data using only the neighboring vectors. Therefore we must constraint the weight matrix so that $w_{i,j} = 0$, if $j \notin N_{k,i}$. Furthermore, let $\sum_j w_{i,j} = 1$ - ie. the weights for each feature vector add to 1.

It should be noted that the weights for each feature vector are independent and so we can compute the weight vector separately for each of the vectors.

Let us begin by defining the *local covariance matrix* of the feature vector x_i : Cov_{x_i} .

$$Cov_{x_i} = (X_{k,i} - x_i)(X_{k,i} - x_i)^T$$

We compute the local covariance matrices for each of the feature vectors and solve the following equation for w_i :

$$\text{Cov}_{x_i} w_i = 1$$

We then normalize w_i so that it satisfies $\sum_j w_{i,j} = 1$ and thus obtain the elements of the weight matrix.

The last step of the algorithm is finding a projection Y of the dataset X that minimizes the following error term:

$$\Phi(Y) = \sum_i |Y_i - \sum_j W_{ij} Y_j|^2$$

In the first minimalization problem the matrix W was being minimized, while here we are minimizing the projected lower dimension coordinates with respect to a fixed weight matrix. Once again we need to impose constraints to ensure that the solution is well-defined. As explained in [38], we would like the resulting coordinates to be centered around the origin and thus let $\sum_i Y_i = 0$. To avoid degenerate solutions, we impose the following constraint:

$$\frac{1}{N} \sum Y_i Y_i^T = 1$$

To solve this problem, we compute the bottom (lowest) eigennumbers and the respective eigenvectors of the following matrix:

$$M = (W - I)(W - I)^T.$$

The first eigenvalue of M is equal to 1, and is not used for the projection. The d -lowest eigenvectors then form the representation of the dataset X in d -dimensions.[37] On the whole, the algorithm for LLE looks as follows:

1. Find the k -nearest neighbors of each feature vector
2. Compute a weight matrix W that best represents the dataset using the neighboring feature vectors
3. Find Y - the best low dimensional k -neighbor mapping with respect to the weight matrix W

Complexity

In this part, we will compare the time complexity of PCA and LLE. We will determine what the critical parts of each method are and show how the value of k in LLE influences the time complexity of the method.

One of the disadvantages of the LLE approach compared to PCA is higher computational complexity. The critical part of the LLE algorithm is computation of the weight matrix W . For each data point a $k \times k$ set of linear equations needs to be solved. Using Gaussian elimination, solving each of the systems requires $O(k^3)$ time. There are n dimensions for each of m feature vectors and therefore the total complexity of the step is $O(nmk^3)$. The main issue here is that this term rises very fast with the number of neighbors we choose in order to interpret the dataset. It should be noted that computing the nearest neighbors requires $O(m^2n)$ time. The final projection - computing the bottom eigenvectors of a $m \times m$ matrix can be performed in $O(dm^2)$ time, where d is the target dimension (number of eigenvectors we need to find).

In comparison, the critical step of the PCA analysis is the computation of the covariance matrix. Without using advanced matrix multiplication algorithms, $O(m^2n)$ operations are required. Normalization of the data can be performed in $O(nm)$ and computing the first d eigenvectors of the covariance matrix requires once again $O(dm^2)$ time.

In conclusion, for low values of k PCA is comparable with LLE in terms of time complexity. However, the complexity of LLE rises fast when we increase the number of relevant neighboring feature vectors.

2.5 Multidimensional scaling (MDS)

In this section, we will briefly introduce the multidimensional scaling method (MDS).

Let $M = \{m_1, \dots, m_n\}$ be a dataset. Let d be a distance function of the following form:

$$d : M \times M \rightarrow \mathfrak{R}_0^+$$

Furthermore, we denote the distance between two elements of M as follows:

$$d_{i,j} = d(m_i, m_j)$$

Unlike the previously discussed methods LLE and PCA, multidimensional scaling requires input data in form of a dissimilarity matrix. This matrix D contains the distances between the objects from the input dataset:

$$D = \begin{pmatrix} 0 & d_{1,2} & d_{1,3} & d_{1,4} & \dots \\ d_{2,1} & 0 & d_{2,3} & d_{2,4} & \dots \\ d_{3,1} & d_{3,2} & 0 & d_{3,4} & \dots \\ d_{4,1} & d_{4,2} & d_{4,3} & 0 & \dots \\ \cdot & \cdot & \cdot & \cdot & \ddots \end{pmatrix}$$

The goal of the MDS method is to find an optimal embedding of the D matrix to a target dimension p . An optimal embedding is one such that best preserves the distances between the objects in the lower dimensions. In other words, we attempt to find a matrix \widehat{D} so that the distances correspond as closely to the original ones. We then introduce a stress function S of the following form.

$$S : D \times \widehat{D} \rightarrow \mathfrak{R}_0^+$$

There are various definitions for the stress function. The original and the most used one is the raw-Stress version from Kruskal [19]:

$$S(\widehat{D}, D) = \frac{1}{2} \sum_{i=1}^{|M|} \sum_{j=1}^{|M|} w_{i,j} (\widehat{d}_{i,j} - d_{i,j})^2$$

For a more detailed overview of the respective stress functions see Groenen's and van den Welden's article [9] on multidimensional scaling.

In order to minimize this function, one can use several numerical methods. The most commonly used one is a procedure called SMACOF. This is based upon the concept of stress majorization. In overview stress majorization works as follows. Let $f(x)$ be a function that we are attempting

to minimize. We now introduce a so called *surrogate function* $g(x, y)$. The surrogate function must be chosen so that:

$$g(x, y) \geq f(x)$$

y is called a *supporting point* and it is fixed. In the supporting point the following is true:

$$g(y, y) = f(y)$$

The minimalization procedure works as follows:

1. Choose initial starting value $y = y_0$.
2. Find an update of $x(t)$ such that $g(x(t), y) \leq g(y, y)$.
3. Stop if $f(y) - f(x(t)) < \epsilon$, otherwise $y = x(t)$ and proceed with step 2.

For the details of the majorization procedure and other numerical approaches towards computing the lower dimensional embedding please refer to the article [21] on SMACOF in R from de Leeuw and Mair.

3 Specifics of landmark data

In this section we explain in greater detail how anthropological data differs from traditional data sources such as image data.

In image data, the basic input for all methods are typically raw pixel values. When one tries to compare two images, they mostly have the same image size and the same color scale. This is not the case with anthropological data. The images may be rotated, translated or be of different size. Therefore the input data first must be normalized.

3.1 Generalized Procrustes Analysis (GPA)

The most commonly used method for normalization of landmark data is generalized Procrustes analysis. The method was first described in [18]. This method normalizes the data with respect to:

1. Translation
2. Scale
3. Rotation
4. Alternatively with respect to reflection

We first explain how to align two objects towards each other. Subsequently, we demonstrate how to align an entire dataset consisting of multiple specimen using the Procrustes analysis.

For the following analysis, let A, B be 2-dimensional specimens represented by n landmarks ($A = \{(x_1, y_1) \dots (x_n, y_n)\}$, $B = \{(w_1, z_1) \dots (w_n, z_n)\}$). In order to normalize the data with respect to translation, one first computes the arithmetic mean for both dimensions and subtracts it from the original data:

$$\bar{x} = \frac{x_1 + \dots + x_n}{n}, \bar{y} = \frac{y_1 + \dots + y_n}{n}$$

$$\bar{A} = (x - \bar{x}), (y - \bar{y})$$

Normalization of scale is simple as well. The scale of a specimen is computed as follows:

$$s = \sqrt{\frac{(x_1 - \bar{x})^2 + (y_1 - \bar{y})^2 \dots + (x_n - \bar{x})^2 + (y_n - \bar{y})^2}{n}}$$

To normalize A in respect to translation, simply divide it member wise with the scaling factor:

$$\overline{A}_s = \frac{(x - \bar{x})}{n}, \frac{(y - \bar{y})}{n}.$$

In order to remove the rotational component, we will use B as a reference shape and rotate A around origin so that the sum of squared distances between corresponding points of A and B is minimized. Rotation of 2-dimensional A around origin by angle ϕ is a linear operation expressed in matrix form as follows:

$$A_{rot} = A_s * R,$$

where

$$R = \begin{pmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & -\cos(\phi) \end{pmatrix}$$

The term we are attempting to minimize with respect to ϕ is as follows:

$$f(\phi) = (A_{rot} - B)^2 = (x \sin \phi - y \cos \phi - w)^2 + (x \cos \phi + y \sin \phi - z)^2,$$

where $x = x_1 \dots x_n, y = y_1 \dots y_n, w = w_1 \dots w_n, z = z_1 \dots z_n$

The derivation of this term is:

$$\begin{aligned} f'(\phi) = 2(A_{rot} - B) &= 2(-x \sin \phi - y \cos \phi)(x \cos \phi - y \sin \phi - w) + \\ &+ 2(y \cos \phi - x \sin \phi)(x \cos \phi + y \sin \phi - z) \end{aligned}$$

Solving the above term for $f'(\phi) = 0$, gives the optimal orientation of A with respect to reference shape B :

$$\phi = \arctan \frac{xz - wy}{wx + zy}$$

In three dimensions removing translation and scaling is analogical to the two dimensional case. However, removing translation is more challenging. As the process is rather technical we will not describe it here. Instead, we refer the interested reader to Kendall's article on GPA [18].

Now that we have shown how to register 2 objects we will focus on how to transform the entire dataset. The simple way is to select one of the images in the dataset as a reference shape and use the above procedure for each of the other shapes in the dataset. However, the following algorithm highlighted in [18] demonstrates a more elegant solution.

Let us first define Procrustes distance. This is the distance between the registered shape A_{rot} and reference shape B given by:

$$d_{proc} = \sqrt{(A_{rot} - B)^2}$$

The algorithm works as follows:

1. Choose one of shapes in dataset as a reference shape.
2. Normalize all members of the dataset with respect to scale, translation and rotation with respect to chosen reference shape.
3. Calculate the arithmetic mean shape of the resulting shapes
4. If the Procrustes distance between the mean shape and the reference is above a threshold, let the mean shape be the new reference shape and return to step 2.

3.2 Euclidean Distance Matrix Analysis

Another way of ensuring invariance to translation and rotation is using Euclidean distance matrix analysis (EDMA). In this section, we will describe how to compute the EDMA transformation and explain its main advantages and disadvantages in morphological analysis.

Algorithm

In this part of the thesis we describe the process of EDMA computation. Let A be a dataset of the following form:

$$A = \{a_1 \dots a_n\}$$

Here $a_1 \dots a_n$ are objects. Each of these objects is described by m elements i.e:

$$a_i = (a_{i1} \dots a_{im})$$

In order to compute the EDMA matrix, first compute the conventional euclidean distances between the specimens:

$$d_{ij} = \text{dist}(a_i, a_j) = \sqrt{\sum_{k=1}^{k=m} (a_{ik} - a_{jk})^2}$$

Next, we compute the mean shape \bar{a} :

$$\bar{a} = \sum_{i=1}^n \frac{d_i}{n}$$

Furthermore, for each object compute the error vector r_i :

$$r_i = \sum_{j=1}^n \frac{(d_{ij} - \bar{a}_j)^2}{n}$$

The EDMA is a symmetric matrix that looks as follows:

$$E = \begin{pmatrix} 0 & e_{12} & \dots & e_{1n} \\ e_{12} & 0 & \dots & e_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ e_{1n} & \dots & \dots & 0 \end{pmatrix}$$

If the initial objects were two dimensional then:

$$e_{ij} = \sqrt[4]{|\bar{a}_i - r_{ij}|}$$

In three dimensions EDMA is computed as follows:

$$e_{ij} = \sqrt[4]{\left|\bar{a}_i - \frac{3r_{ij}}{2}\right|}$$

For a detailed derivation of the EDMA procedure and its statistical properties we refer the interested reader to [22].

Discussion of EDMA

In this section we will briefly describe the basic properties of the EDMA transformation and mention some of its applications.

The main advantage of the EDMA method is that it is coordinate free. It is invariant to translation and rotation. It is not invariant to scaling but this should be obvious from comparing two specimen represented in the EDMA matrix. Two identical specimen i, j only differing in size would be represented in the EDMA matrix as:

$$e_i = ke_j$$

On the other hand, as EDMA is based upon distances it is harder to visualize and interpret its results, if compared to the strictly landmark-based GPA. Displaying the results could according to researchers introduce bias. For a detailed discussion of the positives and negatives of EDMA please refer to [35].

One of the interesting applications of EDMA is studying sexual dimorphism of species. This is highlighted in the original article on EDMA [22] as well as in [23]. For other uses of EDMA and its variants, please refer to Julien's book on morphometry [15].

4 Clustering

Having described dimensionality reduction and the specifics of landmark data we will now focus our attention on clustering. We will first define the clustering problem and differentiate between hard and fuzzy clustering. We will briefly mention the issue of cluster validity. Subsequently, we will describe hierarchical and partitional clustering. In particular, we will focus on the k-means algorithm.

4.1 Hard and fuzzy clustering

First, let us define the clustering problem: Let n be a (possibly user-defined) number of clusters. Let $D = \{d_1, \dots, d_m\}$ be a dataset. Then a *hard clustering algorithm* produces the following mapping:

$$F(D) \rightarrow 1 \dots n$$

In comparison, in fuzzy clustering the resulting mapping only assigns a probability for each pattern and group pair:

$$F(d_i, k) \in \langle 0; 1 \rangle$$

$$\sum_{k=1 \dots n} F(d_i, k) = 1, \forall i \in 1 \dots m$$

The clustering process can be divided broadly into following steps:

- Data acquisition and representation of the data features
- Defining a measure of proximity
- The actual clustering process
- Evaluation and interpretation of the clustering output

Further details on clustering are provided in [13].

4.2 Cluster validity and selecting the number of clusters

We can see that the only required supervisor input in a traditional clustering algorithm is the number of clusters. As the actual relevance of the result is heavily dependent on the number of clusters, a measure of the output clustering quality should be defined. The process of defining and implementing this measure is described as *clustering validity*. For a detailed description of this problem please refer to Jain's and Dubes's monograph on clustering algorithms [13].

4.3 Hierarchical clustering

The two main approaches to clustering are:

- top-down or partitional
- bottom-up or hierarchical

We will briefly describe the hierarchical approach, however, this thesis focuses predominantly on usage of partitional clustering.

In this section, we will describe the process of hierarchical clustering and highlight its usage in anthropology and morphometry. At the start of hierarchical clustering each exemplar forms a single element cluster with itself. We then select a pair of clusters which are closest together and join them to form a new cluster. We repeat this procedure until we have reached the desired number of clusters.

In a more formal language the hierarchical procedure is as follows:

- Let $Q = \{q_1 \dots q_m\}$ be a dataset
- Let n be the desired number of clusters,
- Let $C = D$ be the clustering set, let $d : c \times c \rightarrow \mathfrak{R}^+$ be a distance function
- If $|C| \leq n$ terminate

- Find c_i and c_j such that $d(c_i, c_j)$ is minimal
- Let $C = C - c_i - c_j + (c_i \cup c_j)$
- Go to 4

Usage in anthropology

In anthropology hierarchical methods are predominantly used in order to produce dendrograms. Unlike partitional methods hierarchical methods output the time at which a certain element was joined to a cluster. This is especially useful when trying to study the evolution of species. As these images can help in deciding at which time a certain species was separated from its original one.

4.4 Partitional clustering

While there exist other forms of partitional clustering we will describe the k-means (or k-center) clustering algorithm in this section.

The user once again selects the number of clusters. First, the algorithm assigns the clusters randomly. Typically elements of the dataset are used as starting centers. In the next step, the algorithm assigns each specimen in the dataset to its nearest center. Then we recalculate the centers. If the position of the centers did not change sufficiently, we end the algorithm. Otherwise, we recompute the clusters.

The k-center algorithm is described in pseudocode here:

```

\\INPUTS:
no_clusters=user defined
dataset = {d[1] .. d[m]}
\\ASSIGN STARTING CLUSTERS RANDOMLY:
for(i in 1 .. no_clusters)
    center[i]= d[random]//should not repeat centers
\\THE MAIN LOOP:
while( True )

```

```
{
//find the nearest center for each element in the dataset:
for( i in dataset)
    {
        cluster[i]=mindist_index(center,d[i])
    }
//recompute the centers:
for( i in centers)
    {
        //choose the members of the respective cluster
        //and recalculate its position:
        newcenter[i]=calc_center(dataset[cluster==i])
    }
//the centers are stable - end:
if(energy(center,new_center)< eps)
    return {center,cluster};
//the centers are not stable - recalculate them:
else:
    center=new_center;
}
```

5 Comparison of traditional algorithms

In the previous sections, we have described the specifics of landmark data. Furthermore, we have explained the basic approaches towards dimensionality reduction and clustering. Thus we have all the tools required for a practical application of these techniques. Our goal will be to determine the importance of dimensionality reduction as a preprocessing algorithm for the clustering algorithm. Furthermore, we will compare the performance of MDS and LLE as preprocessing algorithms.

This section centers around providing experimental evidence for two main hypothesis. First, we would like to show that for high dimensional datasets dimensionality reduction improves the quality of classification if compared to direct clustering. The second task is to determine optimal target dimensionality to maximize the clustering quality for each of the dimensionality reduction methods.

In order to fulfill these goals, we will adopt the following structure.

First we describe the nature of the dataset and the process of data acquisition in greater detail. The main focus will be at attempting to analyze the role of dimensionality reduction algorithms in the clustering process. To examine the first hypothesis, we will compare the results of the dimensionality reduction based methods at target dimensionality 2 with direct clustering. In order to examine the second part of the conjecture, we will analyze the behavior of the dimensionality reduction algorithms at differing target dimensions.

5.1 Description of the dataset

In this section we will describe the dataset that will be used in the testing process. Furthermore, we will discuss the suitability of this dataset in regards to our goal.

The dataset that will be used consists of 24 three dimensional facial scans of students and employees of the faculty of Mathematics and Physics. On each of the facial scans, 13 landmarks were placed:

- 2 landmarks on each eye
- 3 landmarks on the nose
- 4 landmarks on the mouth
- 2 landmarks in the kin region

The landmark configuration is depicted on the following image 1.

The goal of the algorithms will be to classify the exemplars based on gender. In the dataset there are 7 female samples and 17 male ones. As we are interested in unsupervised learning, the classification algorithm will receive no initial training set or expert information apart from the landmarks provided.

We should note that the sample size is relatively low. However, in order to provide evidence in respect to the above stated conjectures we require a high-dimensional dataset. Computations with large high-dimensional datasets are time-consuming. The described dataset provides a comparatively high number of dimensions (39). At the same time it allows for relatively fast testing of the algorithms in question. We will attempt to partially mitigate the small sample size by running each experiment multiple times on the same dataset.

It should be noted that these landmarks were not placed by an expert biologist and hence suffer from imprecision. This is, however, not a flaw for the purposes of our testing. It allows us to study the behavior of the algorithms in less than optimal conditions. Finally, we must emphasize that while we use landmarks on face scans as our testing dataset, our goal was not to develop a face recognition algorithm. We merely use these landmarks as a testing dataset. The methods presented should be transferable to other similar datasets in both two and three dimensions or even landmarks in volumetric data such as CT scans or magnetic resonance images.

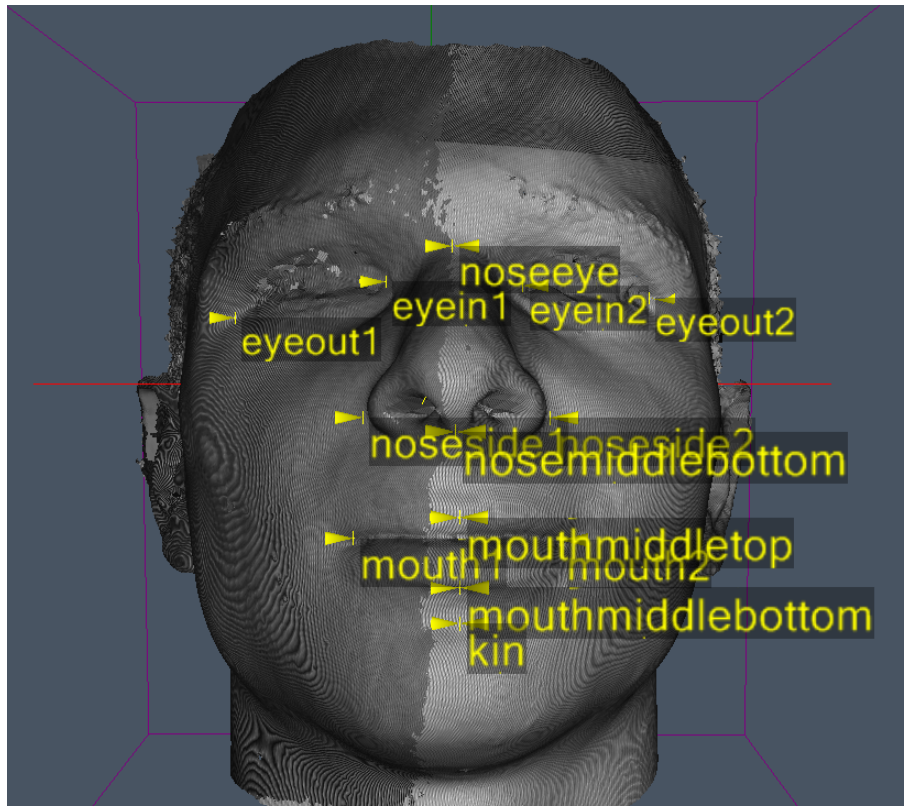


Figure 1: Landmark placements

Source:

- Landmarks placed by author in Morphome3cs Landmark Editor.
- Data provided by the Faculty of Science of the Charles university.

5.2 Impact of preprocessing and dimensionality reduction

Goal of the experiment

In this section, we will attempt to determine the impact of the preprocessing and dimensionality reduction algorithms on the quality of clustering. Our main hypothesis is as follows:

For high dimensional data (such as are common in anthropology) dimensionality reduction improves the clustering quality.

We will now describe the exact nature of the measurement. Subsequently the results of the measurements will be described and interpreted with respect to the above stated hypothesis.

Details of the measurement

In order to proof the above stated hypothesis, we will compare the performance of dimensionality reduction methods with pure clustering. From the dimensionality reduction methods we have selected the following three:

- LLE with GPA (including scaling and reflection) as preprocessor
- LLE with EDMA as input
- MDS with EDMA as input

Finally, as a baseline for our measurements we need to run a pure clustering algorithm on the same dataset without any preprocessing. We will use the Hartigan-Wong algorithm both as the final part of the reduction dimensionality methods and as the baseline algorithm. We will keep the number of clusters to two.

An interesting consideration is setting the target dimension of the dimensionality reduction algorithm. This will be discussed in sections 5.3 and 6.4. Our goal here is not to find a certain magic number to show that in some corner case with a precisely defined input data we can show that dimensionality reduction has impact. Instead we are trying to show that there is no need

for such magic numbers. In fact, a palpable improvement can be obtained regardless of the choice of numeric parameters.

The logical choice in this case is therefore setting the target dimension to 2. This leaves maximum amount of work to the dimensionality reduction algorithm and thus offers the best comparison to the pure method. For LLE the other critical parameter is the number of neighbors that are to be accounted for in the algorithm. This essentially distinguishes between local and global approach to dimensionality reduction. For this part of the experiment, we kept this parameter to 8. This represents a reasonable compromise between local and global dimensionality reduction. Once again this will be discussed in the later course of the thesis. We will focus on the impact of this parameter in the section 6.5.

It should be stressed here that partitional clustering and k-means clustering in particular is dependent on the choice of starting configuration. Therefore randomization is part of every clustering algorithm. Therefore it is not sufficient to perform a single measurement. We will run each method five times. In addition to the average clustering quality, we will record maximal and minimal values as well as the median result.

Results and their interpretation

The results of the above outlined measurements are outlined in the Table 1. We can now answer that preprocessing and dimensionality reduction has an impact on the quality of clustering. While the pure Hartigan-Wong algorithm struggled with a mean success rate of 52.4 percent, the reduction dimensionality techniques improved on this result by an average of almost 8.0 percent.

Among the dimensionality reduction methods multidimensional scaling recorded the highest average clustering quality of 61.6 percent. It is closely followed (61.2) by the combination of locally linear embedding with EDMA as the input preprocessor. Given the popularity of generalized Procrustes analysis in the field of anthropology if compared to the EDMA approach it is surprising that it only recorded a success rate of 58.3 percent.

Table 1: Impact of dimensionality reduction - success rate in percent

| Method Name | Mean | Median | Max | Min |
|---------------|------|--------|------|------|
| Hartigan-Wong | 52.4 | 54.1 | 54.1 | 50 |
| GPA+LLE(2,8) | 58.3 | 58.3 | 62.5 | 50 |
| EDMA+LLE(2,8) | 61.2 | 66.6 | 66.6 | 58.3 |
| EDMA+MDS(2) | 61.6 | 62.5 | 79.1 | 50 |

Source: Author.

This is where we need to abandon the mean results and analyze the results in greater detail. While the original success rate of the GPA based method seems low, it should be noted that this is significantly impacted by one measurement. In the fourth run the GPA method only achieved the minimal value of 50 percent. This has significantly impacted the average value. If this measurement is removed the average rises to 60.4 percent. While this is still somewhat lower than the more successful methods the difference is within a single percent point.

Similar approach must be taken when considering the results of the multidimensional scaling procedure. If we disregard the single positive outlier (79.1 percent), the success rate drops to only 57.3 percent. This constitutes a drop of 4.1 percent points. It is even lower than the original unadjusted value for LLE with Procrustes analysis as preprocessing filter. On the other hand, this is still significantly above the pure clustering results.

One could in similar fashion as with GPA+LLE disregard the minimal value for the multidimensional scaling procedure. While this would further reduce the number of conducted measurements, it may give us an indication of the behavior of the multidimensional scaling in average case. If we follow this approach, we achieve an average success rate value of 59.7 percent. This is comparable with other methods but to rank the methods, obviously more measurements need to be conducted.

The difference between the reliable LLE+EDMA approach and the more variable procedure of MDS+EDMA is further highlighted on the following graph. This graph depicts all the measurements for both methods ranked in

descending order of success rate.

Summary

In conclusion, we have confirmed that for high dimensional datasets that are common in the field of anthropology, dimensionality reduction improves the quality of clustering. While the pure clustering approach was only able to barely cross the 50 percent mark at our dataset, the performance of dimensionality reduction methods hovered around 60 percent success rate. The most successful of the dimensionality reduction methods were the ones based on EDMA.

The combination of EDMA and multidimensional scaling achieved the highest average success rate. This method was also responsible for the highest individual score of nearly 80 percent. It was closely followed by the combination of LLE and EDMA. If compared to the previous method, this method achieved more stable results. It also had by far the highest median score of 66.6 percent.

Surprisingly GPA-based LLE did not perform as well on the dataset in question. Its measurements could have been negatively impacted by one single measurement. If we subtracted this measurement, we would acquire a respectable result of just over 60 percent.

One could argue here that the number of measurements is insufficient. Indeed it is on the low side. But it is sufficient to demonstrate the necessity of the dimensionality reduction methods. We will conduct further experiments in order to compare and contrast the methods and determine their optimal parameters.

5.3 Choosing the right dimension

Rationale

In the previous section we have demonstrated that dimensionality reduction techniques are helpful when attempting to cluster high-dimensional landmark-type objects. What we have not discussed is the actual role of

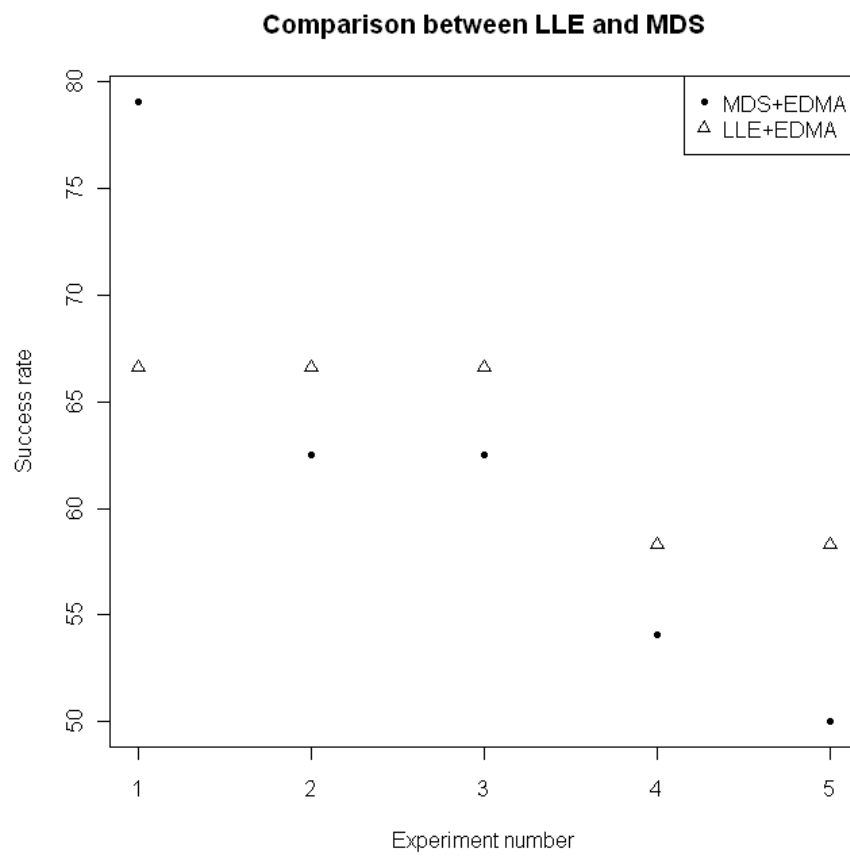


Figure 2: Comparison between behavior of LLE and MDS
Source: Author.

the dimensionality reduction algorithm in the clustering process.

Essentially there are three possible approaches for this, which we will examine in this section. These are:

- Reduce the dimension as far as possible
- Only transform the data while keeping their original dimension
- A compromise between the two approaches

Our initial hypothesis is that for high-dimensional data the best approach is to reduce the dimensionality as far as possible and leave as little work for the clustering algorithm. In addition, we would like to show that improvement does not necessarily depend on selecting the *correct* target dimension. After all, one of the main advantages of the dimensionality reduction methods is that they do not require expert user input. Thus forcing the user to optimize a certain parameter to a specific value would be counterproductive at best. In this case, one may wish to choose one of many supervised methods such as Linear Discriminant Analysis or Support Vector Machines.

We will briefly discuss, how this problem can be handled while using principal component analysis. Then we will conduct a series of measurements to proof our hypothesis for LLE and MDS-based methods.

PCA

This is one area, where using the traditional principal component analysis, simplifies the problem. Typically the PCA outputs an eigen-vector of the

following form:

$$L = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \cdot \\ \cdot \\ \lambda_i \\ 0 \\ \cdot \\ \cdot \\ 0 \end{pmatrix}$$

Here $\lambda_1 \dots \lambda_i$ are the eigennumbers of the respective covariance matrix. The number of non-zero eigenvectors of the covariance matrix is given by i . Usually the vector does not have full rank as several of the eigenvalues are zero. For our question, an interesting observation is that if we square the eigenvalues and add them together we receive the amount of variability that is explained by the selected eigenvalues. Formally, this can be written as follows:

$$\phi(\Lambda) = \frac{\sum_{k \in \Lambda} \lambda_k^2}{\sum_{j=0}^i \lambda_j^2}, f\Lambda \subseteq 1 \dots i$$

While we cannot directly state the exact optimal value for maximum clustering quality while using PCA, selecting Λ such that $\phi(\Lambda) > 0.95$ and $|\Lambda|$ minimal is a good starting point for most applications. For a detailed explanation refer to [14] and [15].

Unfortunately, there is no such measure for MDS or LLE. Both of these methods have the target dimension as part of their algorithms. Therefore we need to conduct an experiment to determine the optimal settings.

Description of the Measurements

We will measure the performance of the successful algorithms from the previous section ie. :

- GPA+LLE
- EDMA+LLE
- EDMA+MDS

Unlike in the previous section, we will not keep the target dimension constant. For each of the methods we will measure their behavior for target dimensions 2-15. As the default clustering algorithm once again the Hartigan-Wong method will be used.

Regarding the number of neighbors for the LLE-based methods, we will adopt the following approach: The number of neighbors will be 6 + the current target dimension. This should be a reasonable compromise between local and global approach to the LLE process. We will analyze this issue in greater detail in the subsequent parts of the thesis.

In order to alleviate the impact of clustering randomization, this time we will run each method seven times for each parameter value.

Multidimensional scaling

The results for the multidimensional scaling method are illustrated on the following figure 3.

First of all, we can notice that for dimensions greater than 12 the dimensionality reduction method can hardly outperform the theoretical baseline of pure clustering. While we have not conducted exhaustive search for higher dimensions, the tendency is apparent from the behavior for dimensions greater than 12. All in all, in these cases the dimensionality reduction algorithm works just as a data transformation algorithm. The lower dimensions are usually unnecessary for the interpretation and representation of the original dataset. As a result the clustering algorithm is unable to separate the datasets. In short, there is too much unnecessary information which impedes successful clustering.

It should be noted that most results for dimensions greater than 10 exhibit similar behavior. In one or two runs the clustering algorithm chooses a

suitable initial configuration and converges towards a good clustering. However, there is an overwhelming tendency towards reaching unsuitable local minima with success rates barely over 50 percent.

However, the results are slightly surprising. The most successful dimension for the presented dataset was 8 closely followed by 9. With the target dimension 8 the clustering algorithm performed admirably and recorded a success rate of 65.4 percent. With 9 dimensions the algorithm was only marginally less successful - it reached 63.8 percent success rate. This is still 3.2 percent point more than the success rate for two dimensions. On the other hand, the highest individual result from a single measurement still belongs to the two dimension variant. As mentioned before it reached 79.1 in one of the runs. In comparison, the next best result of 75 percent was achieved by several dimensions (4, 7, 8, 9, 11, 12, 13).

What also needs to be addressed is the behavior of the algorithm between the dimensions 3 to 7. For the examined data, it is obvious that the performance at these dimensions is poor in comparison to both target dimension two and dimensions 8 and 9. On average, in this segment the mean result was just 54,7 percent. This is only slightly more than the baseline for pure clustering. It is more than 10 percent points less than the performance for 8 dimensions. If we subtract the relatively successful dimension four from the equation, the average drops by further 1.2 percent point.

The question arises why this is the case. One would naturally expect that the function would exhibit monotonic behavior - decreasing as the number of dimensions increases. While we do not have a definitive answer, we can use the comparison with the PCA analysis. Once again the more dimensions we keep the more information value is preserved. On the other hand, the performance of clustering algorithm improves with the decreasing number of dimensions of the clustering dataset. We can therefore conclude that these dimensions are significant for the separation of the dataset and by removing these dimensions the information loss is greater than the gain from a slightly improved clustering performance.

We have already briefly touched on the variability issue in the previous section as we were explaining the behavior of the MDS algorithm. Here we

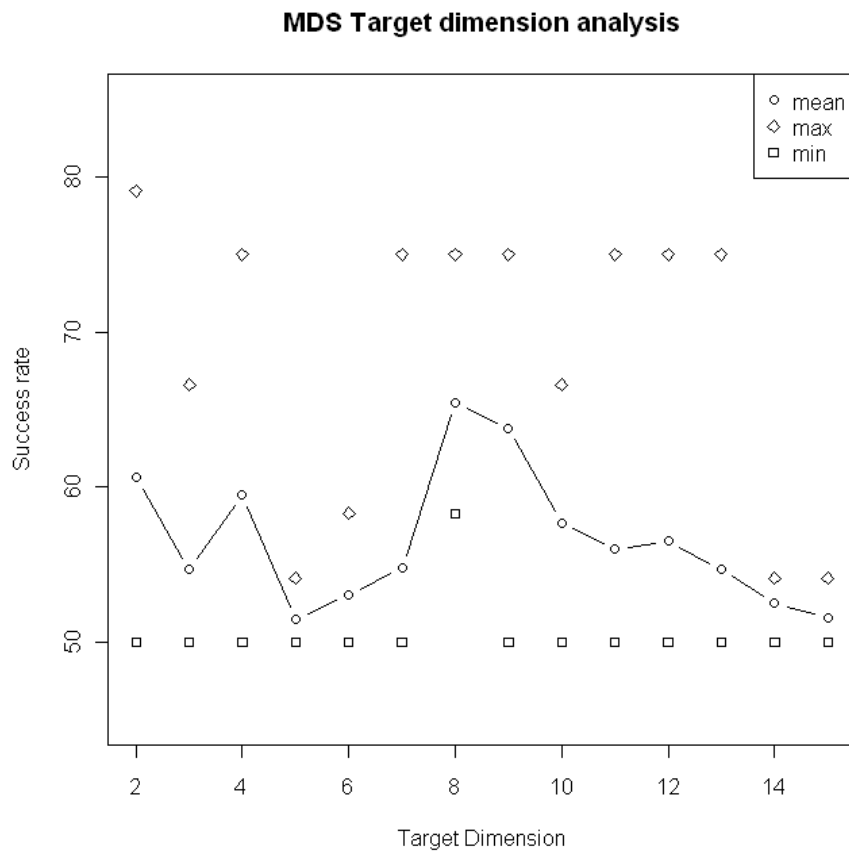


Figure 3: Multidimensional scaling - selecting the optimal target dimension
Source: Author.

Table 2: Variability of the MDS measurements

| Dimension | Mean | Variability |
|-----------|------|-------------|
| 2 | 60.6 | 91.6 |
| 4 | 59.5 | 148.9 |
| 8 | 65.4 | 56.4 |
| 9 | 63.8 | 125.8 |

Source: Author.

will further analyze the variability issue. The variability for the selected dimension is recorded along with the mean values in the following table 2. We have selected only the most promising dimensions for the variability discussion. There is little value in discussing the variability of the less successful methods.

We can see that not only does the algorithm in our experiments achieves the maximal success rate at dimension 8, its variability is also the lowest from the observed group. This means that results for dimension 8 should be the most reliable ones, in addition to allowing for the best possible clustering. On the other hand, the algorithm performs poorly for dimension 4 with both highest variance as well as the lowest success rate.

In between the two extremes, there are dimensions 9 and 2. While 2 does achieve a marginally lower success rate, its reliability is much greater than for dimension 9. This is to be expected as the higher number of dimensions may once again lead to a faulty local minima convergence. While when the algorithm converges properly it may achieve higher success rate as more information is preserved in the higher dimensional model.

Summary

It seems that on the analyzed dataset the MDS algorithm performs best for the dimensions 8 and 9. For dimension 2 the success rate is better than average. However, dimensions 8 and 9 recorded a success rate that was greater by more than 4 percent points. The method was even more stable in respect to its variability for dimension 8 than for target dimension 2. Based

on these results we must slightly change our initial hypothesis. It appears that the best usage of the dimensionality reduction algorithm so far seems to be in dividing the labor with the clustering process. We will now focus on the LLE based methods to see if we can find similar behavior for them as well.

LLE with GPA

In this part we will discuss the results of the locally linear embedding with generalized Procrustes analysis as a preprocessor. The results for this combination of methods are listed in the table 3.

While the results for dimensions greater than 13 are not listed they exhibit similar behavior to dimensions 10 to 13. The average there hovers around 58 to 61 percent. They exhibit the same type of variance as was demonstrated with the MDS results. Although while with MDS the results were typically skewed by one single favorable clustering, here the favorable clusterings occur more often and the median value is also typically a little higher (50 for the high dimensional MDS method, 58.3 for the high dimensional LLE/GPA combination).

The first observation about the GPA/LLE combination, for the examined dataset, is that the average success rate is much higher than for the MDS-based clustering. Here the success rate for dimensions two through thirteen is 61.2 which is almost as high as the success rate for the four best-performing dimensions in the MDS method (62.3). While only for three dimensions in the MDS method the 60 percent mark was crossed, here only for dimensions two, three and eleven the algorithm failed to reach this psychological border. Even the variance between the dimensions is a pleasant surprise. The standard deviation here is 3.95. For the MDS method the standard deviation is 4.3 with a much lower average. It should be noted that the standard deviation for LLE/GPA is mainly driven by the performance of the algorithm at dimension 6.

At dimension six the algorithm achieved an average success rate of 72 percent. This constitutes an improvement of more than six percent compared

to the best success rate for the MDS algorithm. Moreover, the algorithm never recorded a run with worse than a 58.3 success rate. Finally, it also recorded a 83.3 percent clustering.

However, it can be said that this is an isolated peak of the algorithm. While the MDS algorithm had a two dimensional optimum at eight and nine dimensions, the LLE/GPA approach only peaked at exactly 6 dimensions. Localizing such a peak is obviously data dependent. In order to accelerate finding of this peak, one could use the similarity between LLE and PCA. One would use the dimensionality specified by the PCA eigenvectors as a baseline in the further computation. Subsequently one would attempt to optimize the LLE clustering result in the close region specified by the PCA approach.

The highest single clustering performance was recorded by the algorithm at 11 dimensions. Apart from this single run (87.5) the algorithm at eleven dimensions did not perform remarkably well. In none of the following run did it manage to cross the 60 percent baseline. Thus the maximal value should be regarded as an outlier.

LLE with EDMA as preprocessor

In this section we will focus on the LLE-EDMA combination. The results of this method for dimensions 2 through 13 are displayed on the following figure 4.

The average success rate is 61.0 percent. In comparison, the LLE/GPA recorded an average success of 61.3. If we, however, disregard the performance of the LLE/GPA algorithm at dimension 6 (72.0 percent), its mean success rate drops by 1 percent to 60.3 percent. This would place the LLE EDMA method ahead of the LLE GPA combination.

Overall the algorithm appears to be the most stable of the three discussed. Here with stability we mean indifference towards the selection of dimension. The standard deviation across the dimensions is 2.50. This is much lower than both LLE/GPA (3.95) and the multidimensional scaling variation (4.34).

As we can see the behavior of the algorithm closely resembles the one

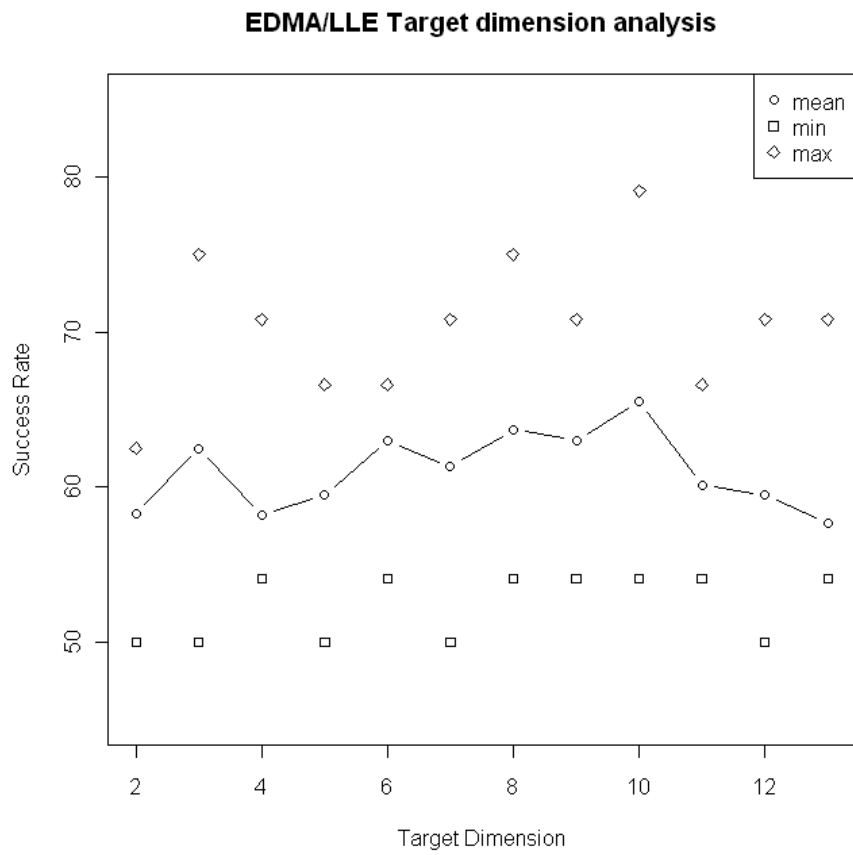


Figure 4: LLE/EDMA - selecting the optimal target dimension
Source: Author.

of the MDS algorithm. Once again the success rate drops significantly after dimension 10. While the average success rate for dimensions 6-10 is 63.3 percent, the average for dimensions 11 through 13 is only 59.1 percent. This is demonstrated in the table 4. On the other hand, the standard deviation for dimensions eleven through thirteen is 17.9 percent less than the average for the dimensions 6 through 10. If compared to the behavior for dimensions 2 through 5, the algorithm performed comparably at dimensions 11 through 13. The average success rate is lower by 1.3 percent point. On the other hand, it appears that the algorithm is much more stable at higher dimensions with the standard deviation being 0.76 points lower than for the dimensions 2 through 5.

The algorithm performed at its best for dimensions 6 through 10. The highest average success rate was recorded for 10 dimensions (65.5). On the other hand, this is still 7.5 percent point behind the maximal performance of the GPA based algorithm at six dimensions. The LLE/EDMA algorithm also recorded its highest percentage clustering at 8 dimensions (79.1 percent). The algorithm also recorded a couple of runs with 75 percent success rate. This namely occurred at dimensions three and eight.

The next best performances of the algorithm occurred at target dimensions six, eight and nine. The algorithm reached 63.7 at eight dimensions. For dimensions six and nine the mean success rate was 63.0. Unlike the GPA algorithm the behavior in the optimal region appears to be more stable. While the GPA base algorithm does have the best performance of 72.0 percent, it only cross 62 percent at one other dimension. Meanwhile the EDMA based algorithm crossed the 62 percent line five times.

Outside of dimensions 6 through 10 only the behavior at dimension 3 is noteworthy. The algorithm here reached 62.5 percent success rate. This is far better than the behavior of the competing algorithms for dimension 3. It is exactly 6.0 percent point better than the LLE/GPA combination at 3 dimensions. The MDS algorithm failed to cross the 55 percent barrier at target dimension three altogether. We can generalize that for dimensions two through five the LLE/GPA was the best performing algorithm. This is further demonstrated in the following table 5.

As we can see it recorded an average success rate of 60.5 percent. None of the other algorithms managed to cross the 60 percent mark. This was mainly due to the behavior of the LLE/EDMA algorithm at dimensions two and three. On the other hand, the GPA based algorithm outperformed the LLE/EDMA combination at target dimensions four and five. At dimension 4 even the MDS approach was more successful than the EDMA-based locally linear embedding.

Last but not least, we will focus on the variability of the measurements of the EDMA/LLE algorithm. The variability and standard deviation was computed for target dimensions 3, 8, 9 and 10. The results are recorded in the table 6.

If compared to MDS algorithm the EDMA/LLE method appears to be more stable. While the variability of the MDS method ranged from 56.4 to 148.9, the variability of the selected EDMA runs never crossed 70. On the whole, the lowest standard deviation was recorded for dimension nine (6.58). The highest success rate for dimension eight (63.7 percent), was not caused by positive outliers as we can see from the table. The variance for dimension 8 was comparable to the best result of dimension nine. Target dimension three was the least successful one both in terms of mean success rate as well as in terms of variability from the selected dimensions.

Summary

In conclusion, the LLE with GPA algorithm recorded the best overall result for the examined dataset at dimension 6 with a success rate of 72.0 percent. The highest average across all dimensions was recorded by the LLE with EDMA algorithm. It was also the most stable algorithm.

On the dataset in question, all three algorithms performed best in the situation when they shared their work evenly with the clustering algorithm. The best performing dimension for LLE with GPA was dimension 6, while MDS and LLE/EDMA recorded the highest averages at dimension 8 through 10. For dimensions two through five the algorithms rarely crossed the 60 percent mark. While the average score for dimensions six through ten was

well over the 60 percent mark, we should note that there was a tendency to lower performance as the dimension increased over ten. This is caused by the fact that the information contained in these dimension has very little relevance for the clustering process and only misleads the clustering process.

It appears that for our dataset, the dimensionality should not be reduced as far as possible as this leads to loss of information for the clustering process. Given the relatively small sample size, we must be careful in assessing the relevance of this observation. Nevertheless, the fact that the same behaviour was observed for all three algorithms, solidifies this claim.

Thus the question arises, whether we can alter the process so that we can adopt the original approach - ie. reducing the dimension as far as possible. We will discuss our approach that uses multipass dimensionality reduction in the following section.

Table 3: Success rate for the GPA/LLE measurements

| Dimension | Mean | Max | Min |
|-----------|------|------|------|
| 2 | 58.3 | 62.5 | 50 |
| 3 | 56.5 | 70.8 | 50 |
| 4 | 61.9 | 66.7 | 54.1 |
| 5 | 61.3 | 75 | 50 |
| 6 | 72.0 | 83.3 | 58.3 |
| 7 | 61.8 | 75 | 50 |
| 8 | 62.4 | 75 | 54.1 |
| 9 | 61.2 | 75 | 50 |
| 10 | 60.6 | 75 | 50 |
| 11 | 58.8 | 87.5 | 54.1 |
| 12 | 60.1 | 75 | 50 |
| 13 | 61.2 | 75 | 50 |

Source: Author.

Table 4: Success rate of the EDMA/LLE algorithm

| Dimensions | Mean | Std. Dev. |
|------------|------|-----------|
| 2-5 | 60.4 | 2.00 |
| 6-10 | 63.3 | 1.51 |
| 11-13 | 59.1 | 1.24 |

Source: Author.

Table 5: Success rate of the algorithms for target dimensions 2 through 5

| Dimension | MDS | GPA/LLE | EDMA/LLE |
|------------|------|---------|----------|
| 2 | 60.6 | 58.3 | 61.6 |
| 3 | 54.8 | 56.5 | 62.5 |
| 4 | 59.5 | 61.9 | 58.2 |
| 5 | 52.4 | 61.3 | 59.5 |
| Mean (2-5) | 56.9 | 59.5 | 60.4 |

Source: Author.

Table 6: Variability of the MDS measurements

| Dimension | Mean | Variability | Std. Dev. |
|-----------|------|-------------|-----------|
| 3 | 62.5 | 69.44 | 8.33 |
| 8 | 63.7 | 44.98 | 6.70 |
| 9 | 63.0 | 43.30 | 6.58 |
| 10 | 63.0 | 50.14 | 7.08 |

Source: Author.

6 Multipass dimensionality reduction

In the previous section we have discussed the impact of the target dimension on the result of the classification process. We have shown in the previous section that it appears that the optimal division of labor between the clustering and dimensionality reduction algorithm is an even division partition. While there exist guidelines as to how to select optimal dimensionality for PCA, dimensionality is an explicit parameter of both MDS and LLE. Thus the user is forced to determine the optimal dimension by trial and error.

Therefore our goal is to focus on lowering the number of dimensions required for successful classification. In particular, we will focus on the performance of the algorithms at target dimensionality 2. Success in this task would allow the user to quickly find a reasonable clustering output without being forced to iterate through the entire target dimension space.

In order to achieve this goal, we will introduce a new multipass approach towards dimensionality reduction. First and foremost, we describe the algorithm for multipass dimensionality reduction. Subsequently, we compare the performance of the multipass algorithms with the traditional approach at dimensionality two. Furthermore, we concentrate at selecting the optimal k-means algorithm on the classification process.

We will also study the behavior of the multipass algorithm at higher dimensions and once again discuss its advantages and disadvantages compared to the singlepass methods. Finally we will focus on the impact of the number of neighbors parameter both on the singlepass and multipass LLE algorithms.

6.1 Description of the algorithm

In this section, we will describe the multipass dimensionality reduction algorithm.

At start, the user selects the target dimension for the algorithm and also assists in defining the step function. We begin the process by applying the preprocessing algorithm on the dataset. The preprocessing algorithm in our case will be either EDMA or GPA. Next, we determine the next

target dimension by applying the user defined step function. Then we apply the dimensionality reduction algorithm (ie. LLE or MDS) with the target dimension determined by the step function. We repeat these two steps until the desired dimensionality (typically 2 or 3) is reached. In this thesis, we will limit ourselves to using simple division as the step function s :

$$s(d_{trg}, d_{cur}, n) = \max(d_{curdivn}, d_{trg})$$

We finish the process by using a clustering algorithm on the resulting data. In a more formal way, the algorithms looks as follows:

1. INPUT:
 - Dataset
 - Target dimension
 - Step function or step parameter
 - Dimensionality reduction algorithm (MDS, LLE)
 - Preprocessing algorithm (GPA, EDMA)
 - Clustering algorithm (Hartigan-Wong, Lloyd)
2. Apply the preprocessing algorithm on the dataset. Let the current dataset be the dataset returned by the preprocessing algorithm.
3. Calculate the next dimension using the step function.
4. Apply the dimensionality reduction algorithm on the current dataset. Let the current dataset be the dataset returned by the dimensionality reduction algorithm.
5. If the next dimension is greater than the target dimension, go to 3, otherwise go to 6.
6. Use the clustering algorithm on the current dataset.
7. OUTPUT: Classification provided by the clustering algorithm in the previous step.

A similar concept of iterative dimensionality reduction is introduced in [27]. There the author uses gradient descent to improve the quality of the dimensionality reduction, but they do not alter the target dimension. They also predominantly focus on PCA and kernelPCA. Our focus will be on the improvement of LLE and MDS performance.

6.2 Comparing the algorithms

Once again we begin the testing process by comparing the algorithms at target dimensionality two. Our starting hypothesis is that **multipass dimensionality reduction improves the clustering quality at low dimensionality if compared with the singlepass approach**. Furthermore, the multipass algorithm should lead towards more stable clustering results.

In order to proof this hypothesis, we will compare the results from the section 5.2 with the results of the multipass algorithm at dimensionality two. In order to proof the second part of the hypothesis we will compare the variability of the success rates between the two approaches. We will execute each of the following algorithms seven times on the same dataset:

- Multipass EDMA/MDS
- Multipass EDMA/LLE
- Multipass GPA/LLE

Once again the final clustering algorithm will be in all cases Hartigan-Wong clustering. The number of clusters will be kept to two clusters. We will focus on the impact of both of these variables in the following parts of the thesis. For the LLE based algorithms, the number of closest neighbors was kept at eight. This means that in the initial stages the algorithm performs the LLE algorithm with the number of neighbors equal to the target dimension plus one. Once the dimensionality drops below eight, the number of neighbors is left at eight.

The overview of the results is presented in the table 7.

Table 7: Multipass dimensionality reduction

| Algorithm | Mean | Median | Max | Min |
|-----------|------|--------|------|------|
| MDS+EDMA | 76.7 | 79.1 | 79.1 | 70.8 |
| LLE+EDMA | 58.3 | 58.3 | 66.7 | 54.1 |
| LLE+GPA | 67.2 | 58.3 | 79.1 | 58.3 |

Source: Author.

Unfortunately, our hypothesis is only partly confirmed by these results. On the whole, the multipass-based algorithms with target dimension 2 recorded a mean success rate of 67.4 percent. We can see that this is far higher (7.3 percent points) than the average of the singlepass algorithms at dimensionality two. However, from the table it is obvious that the algorithms benefited differently from the multipass approach.

Especially, the multipass MDS algorithm performed extremely well. Not only did it outperform the standard MDS algorithm at two dimensions. In fact, it was the most successful of the methods used so far. It recorded an average success rate of 76.7 percent. This is 4.7 percent point higher than the previously best result for the singlepass GPA+LLE combination at target dimensionality 8. Moreover, the clustering results appear to be stable. This is suggested by the very high minimal clustering value of 70.8 percent. All of the approaches handled so far, recorded a minimal clustering value of less than sixty percent. The second best minimal clustering of 58.3 percent was shared by multipass LLE+GPA at target dimensionality 2 and the singlepass LLE+GPA algorithm at dimensionality 6. Further evidence to the stability of the MDS multipass algorithm is provided in the table 8.

From this table, we can discern that not only did it the MDS achieve the highest success rate on the examined dataset. Moreover, if compared to the three other well performing methods, multipass MDS method reached a standard deviation of 4.04, while the nearest competitor recorded a standard deviation of 8.80.

This table also suggests the potential shortcoming of the multipass LLE+GPA method. On the one hand, it recorded a relatively high success rate of 67.2 percent. The minimal clustering success rate of 58.3 can be interpreted as

Table 8: Variability of results

| Algorithm | Target dimension | Mean | Var | Std. Dev. |
|--------------------|------------------|------|--------|-----------|
| Multipass MDS+EDMA | 2 | 76.7 | 16.04 | 4.04 |
| Multipass LLE+GPA | 2 | 67.2 | 123.61 | 11.11 |
| Singlepass LLE+GPA | 6 | 72.0 | 77.57 | 8.80 |

Source: Author.

a relative success. Only the multipass MDS algorithm recorded a better minimal clustering success.

On the other hand, the GPA-based multipass LLE suffers from great instability. All of the measurements performed by us ended with a result of either 79.1 percent or 58.3. In the end, this resulted in a very high variability value of 123.61. This means that the quality of the clustering is largely dependent on the initial center choice selected by the k-center algorithm as well as the exact mechanics of the k-center algorithm. One could attempt to fix this flaw by careful selection of clustering algorithm. Another possibility would be to run the clustering algorithm multiple times and average the classification results over multiple runs.

The results for the LLE+EDMA combination are disappointing. Indeed, its mean success rate is 2.9 percent points lower than the success rate for its singlepass variant at dimensionality two. None of the runs exceeded 70 percent clustering success rate. Overall, the best result was just 66.7 percent. All the other runs achieved a success rate of less than 60 percent.

In conclusion, we may state that on the presented dataset multipass MDS achieved the best performance. Multipass dimensionality reduction also improved the performance of the LLE+GPA combination. Unfortunately, the performance of the LLE+EDMA combination was poor. Therefore in the further course of the thesis we will focus on the behavior of the more promising methods: LLE+GPA and MDS+EDMA.

6.3 Impact of the clustering algorithm

So far we have neglected the importance of the clustering algorithm in the process. In all the previous test cases, we have used the Hartigan-Wong algorithm. On the other hand, it appears that especially in the case of multipass LLE we could improve the performance of the method by careful selection of the clustering process.

Therefore our goal in this section is to test several clustering algorithms and compare their results. We will compare the following k-means-based algorithms:

- Hartigan-Wong [11]
- Lloyd [25]
- Mac Queen [26]
- Forgy [8]

It should be noted that the above described algorithms are implemented in the R library. As a result they can be used in the Morphome3cs software. For a detailed information on the R binding please refer to section A.3.

MDS

We will start determining the impact of the clustering algorithm by studying the performance of the multipass MDS at dimensionality 2. The results of this experiment are presented in the table 9.

Overall we can see that the success rate of the algorithm heavily depends on the choice of the clustering approach. the difference between the most successful We can see that the Hartigan-Wong algorithm was the most successful one both in terms of average performance as well as clustering stability. This is no surprise as this algorithm is more sophisticated than its competitors.

Both the Mac Queen and Forgy approaches achieved similar results. While the Forgy method achieved a marginally higher success rate (0.6 percent point higher than the Mac Queen variant), the Forgy algorithm was less

Table 9: Clustering Algorithm Performance - multipass MDS

| Algorithm | Mean | Var | Std. Dev. |
|---------------|------|--------|-----------|
| Hartigan-Wong | 76.5 | 16.04 | 4.04 |
| Forgy | 69.6 | 133.91 | 11.44 |
| Lloyd | 60.0 | 120.33 | 10.96 |
| Mac Queen | 69.0 | 114.76 | 10.70 |

Source: Author.

stable than the Mac Queen approach with a 0.74 higher standard variance. In the testing process, the Forgy method recorded a minimal clustering success rate of 50 percent. In this respect, it was outperformed by the Mac Queen algorithm which achieved a minimal value of 54.1 percent. Both of the examined algorithms achieved a maximal clustering with 79.1 success rate.

Finally, the Lloyd algorithm performed poorly. It barely crossed the 60 percent success rate. Even its variance was not comparable with the Hartigan-Wong approach.

Multipass LLE+GPA

In the section 6.2, we have stated that while the multipass algorithm improves the results of the LLE+GPA method at target dimension two, it suffers from great instability if combined with the Hartigan-Wong clustering. In this section, we will therefore focus on improving the stability of the LLE+GPA method by using other clustering method. Unfortunately the results for multidimensional scaling suggest that this might not be possible as the Hartigan-Wong algorithm outperformed the other k-center variants.

The results of the experiment are presented in the table 10.

Unlike with multipass MDS algorithm, here we can see that the Hartigan-Wong algorithm was outperformed by both the Lloyd's and Mac Queen's methods. The highest average score was achieved by the Lloyd algorithm. This average score of 69.6 is the same as the performance of the Forgy's algorithm with multipass MDS. In fact, LLE+Lloyd was even more stable

Table 10: Clustering Algorithm Performance - multipass LLE+GPA

| Algorithm | Mean | Var | Std. Dev. |
|---------------|------|--------|-----------|
| Hartigan-Wong | 67.2 | 123.61 | 11.11 |
| Forgy | 60.1 | 45.70 | 6.70 |
| Lloyd | 69.6 | 119.88 | 10.94 |
| Mac Queen | 68.4 | 91.86 | 9.58 |

Source: Author.

than the Forgy's algorithm in terms of variability. Moreover, LLE+Lloyd even recorded the highest individual run by a multipass clustering algorithm so far - 83.3 percent.

While the Mac Queen's algorithm achieved a slightly lower average success rate (1.2 percent point) than the Forgy variant, it still outperformed the Hartigan-Wong algorithm. It should be stressed that in this experiment Mac Queen's algorithm outperformed both Hartigan-Wong and Forgy in terms of variability. Four of its runs were above seventy five percent success rate. This is the same number as for the Lloyd's method and one more than for Hartigan-Wong.

The variability of the Forgy's algorithm was exceptional (45.70). Unfortunately, its mean success rate was very low. All but one of its run were below sixty percent. Overall this combination does not look very promising.

In conclusion, we can state that while the Hartigan-Wong algorithm combined well with the multidimensional scaling, it suffered from high degree of variability when combined with locally linear embedding. On the other hand, by using the less complex Lloyd's and Mac Queen's algorithms we have managed to improve the stability of the LLE method. Moreover, this change also resulted in a 2.4 percent point success rate increase. Finally we have achieved the highest value for a multipass algorithm run by using Lloyd's clustering in combination with LLE.

6.4 Selection of Target Dimension

In this section, we will study the impact of the target dimensionality parameter on the multipass algorithms. We have already stated that our main purpose in designing the multipass approach was to improve the performance of dimensionality reduction at target dimension 2. Now we will analyze the behavior of the multipass algorithm and compare it with the performance of the singlepass algorithm at low dimensions.

It should be noted that at dimensions 9 and higher the impact of the multipass approach is relatively low as the number of iterations is only two. In these cases, the multipass algorithm converges towards the singlepass version and we have therefore not focused on these dimensions in our analysis.

MDS

As the MDS algorithm seemed the most promising in our initial tests, we will focus on the behavior of the multipass MDS algorithm for dimensions two through six. The results of the experiment are presented in the table 11.

The results for dimensions three through six are not as persuasive as for target dimensionality 2. Overall the average success rate for dimensions three through six is just 59.6. On the other hand, we must note that clusterings with maximal value greater than 75 percent were recorded for all dimensions but dimension 4. This indicates that the output from multidimensional scaling does separate the two genders, however, due to the extra dimensions the clustering algorithm is unable to find a correct clustering. This effect is not present at dimensionality two resulting in a high mean clustering value.

If we compare the results for dimensions two through six with the performance of the singlepass algorithm, we determine that for all dimensions but target dimension 4 the multipass algorithm outperformed the singlepass variant. The comparison between the performances of the two algorithms for dimensions 2 through 6 is presented on the figure 5.

On average the singlepass method at dimensions 2 through 6 reached a success rate of 56.1. In comparison, the multipass method performed on average 6.3 percent point better. Similarly in terms of maximal performance,

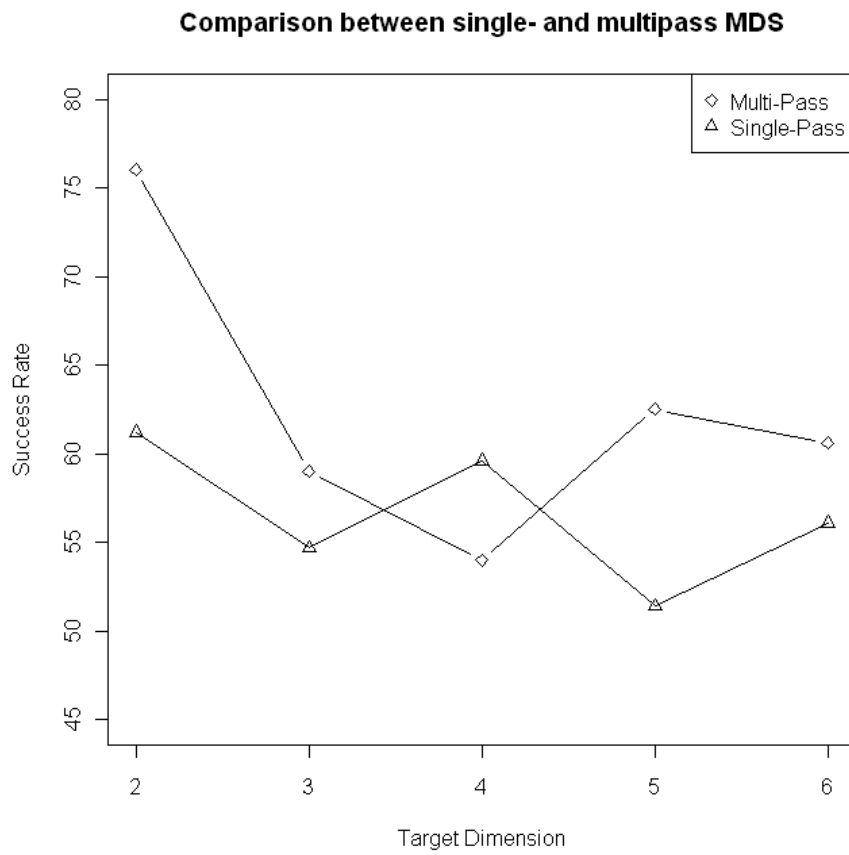


Figure 5: Comparison between singlepass and multipass MDS
Source: Author.

Table 11: Multipass MDS - target dimension

| Dimension | Mean | Max | Min |
|-----------|------|------|------|
| 2 | 76.0 | 79.1 | 70.8 |
| 3 | 59.2 | 79.1 | 50.0 |
| 4 | 54.1 | 54.1 | 54.1 |
| 5 | 62.5 | 75.0 | 50.0 |
| 6 | 60.6 | 79.1 | 50.0 |
| 7 | 62.5 | 79.1 | 50.0 |
| 8 | 61.6 | 75.0 | 54.1 |

Source: Author.

the singlepass MDS algorithm only reached over 70 percent clustering on two occasions. The multipass algorithms achieved this on 15 occasions.

6.5 Impact of the LLE k-parameter

So far we have not discussed the impact of the number of neighbors on either the singlepass or multipass locally linear embedding. Therefore we will focus on this parameter in this section. First, we will discuss the impact of this factor on the singlepass LLE algorithm and then we will focus on its impact on the multipass approach.

Singlepass LLE

The best performance of the singlepass LLE algorithm with GPA as preprocessor was recorded at target dimensionality 6 and therefore we have decided to focus on the k-parameter at this dimensionality. We have kept the target dimensionality constant and varied the number of neighbors used for the LLE algorithm from 7 to 15.

The results of the experiment are documented in the figure 6.

We can see that the algorithm produced the best result with 11 neighbors - 73.2 percent. This is only 2.8 percent behind the best performance of the multipass MDS algorithm at target dimensionality 2. Overall the algorithm

achieved a mean success rate of 62.7 percent. The standard deviation with respect to the neighbor parameter was 5.58.

We have grouped the results of this experiment into three groups:

- Low (7-10)
- Medium (11-13)
- High (14-16)

Table 12: Aggregate results - singlepass LLE+GPA

| Number of neighbors | Mean | Max | Min |
|---------------------|------|------|------|
| 7-10 | 58.7 | 61.3 | 53.5 |
| 11-13 | 68.3 | 73.2 | 65.5 |
| 14-16 | 62.0 | 63.4 | 60.7 |

Source: Author.

The aggregate results for these neighbor counts are shown in the table 12. As is obvious from the table, the algorithm performed best at dimensions 11 through 13. With this number of neighbors, the algorithm recorded an average success rate of 68.3 percent.

It was expected that the performance of the algorithm would be less impressive at dimensions 7 through 10. Here the algorithm recorded a mean success rate of 58.8 percent. This is almost ten percent points less than in the above mentioned optimal range. Furthermore, in two instances the algorithm failed to cross the 60 percent mark. Especially the result at dimensionality

For $dim = k + 1$ the LLE algorithm degenerates as the dataset tends to fragment into closed sets of neighboring specimens. As a result the main advantage of LLE - its non-linearity - is more or less lost. For further examples of this effect, see for instance [37] and [40].

Multipass LLE

Having studied the behavior of the single pass algorithm in respect to the number of neighbors chosen, we will now focus on the behavior of the multipass LLE algorithm. We have already mentioned that the algorithm here

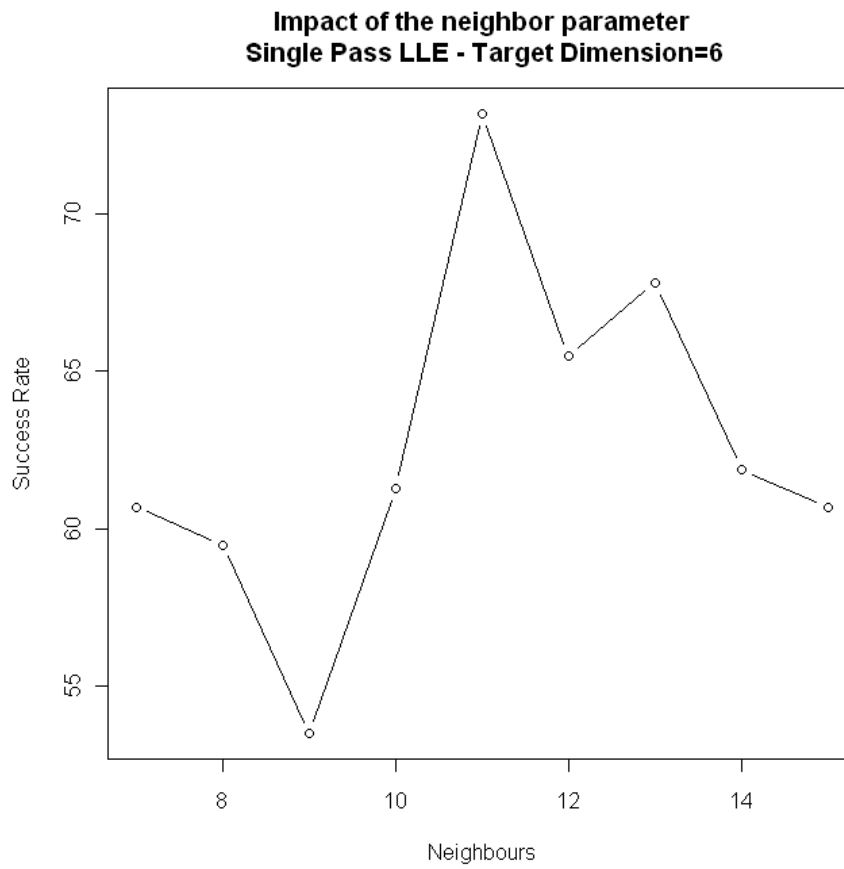


Figure 6: Singlepass LLE+GPA - Impact of the neighbor factor
Source: Author.

behaves slightly differently. As long as the next dimensionality is higher than the maximal number of neighbors the multipass algorithm uses the current number of dimensions plus 1. Only after the dimensionality drops below the specified number we start using the user defined k-parameter.

In this experiment we will use the multipass LLE algorithm with GPA as a preprocessor. Target dimensionality will be kept at 2. For fair comparison reason, we have decided to use Hartigan-Wong clustering even though it appears to be less beneficial for the LLE algorithm. For each maximal neighbor count we have run the test seven times. We have decided to focus on neighbor counts 2 through 12.

The results of the above described experiment are documented in the figure 7.

On our dataset, the algorithm performed best with the maximal neighbor count of 9. Here it recorded an average success rate of 75.0 percent. Only 1.2 percent point worse was the performance of the algorithm with maximal neighbor count 7. Compared to the singlepass algorithm, the multipass algorithm managed to cross the mean success rate of 70 percent for three neighbor count choices. The last occurrence was with neighbor count 3. The singlepass algorithm only achieved this feat once for eleven neighbors. Only once did the algorithm fail to cross the 60 percent barrier. This happened with five neighbors. The singlepass approach failed to cross the sixty percent mark for neighbor counts eight and nine.

Even in terms of the mean performance across neighbor count did the multipass algorithm outperform the single pass version. The mean success rate for the multipass method was 65.7. This is 3.0 percent point higher than the success rate of the singlepass algorithm. In terms of variance between the performance of the method among different neighbor counts, both the singlepass and multipass recorded similar results. The slight advantage was once again held by the multipass algorithm which recorded a standard deviation of 5.41 which is 0.14 lower than the standard deviation of the singlepass algorithm.

Once again we will group the results based on the number of neighbors used into three categories:

- Low (2-5)
- Medium (6-9)
- High (10-12)

The summary of the aggregate results is presented in table 13.

Table 13: Aggregate results - multipass LLE+GPA

| Number of neighbors | Mean | Max | Min |
|---------------------|------|------|------|
| 2-5 | 63.9 | 70.8 | 58.3 |
| 6-9 | 69.9 | 75.0 | 64.8 |
| 10-12 | 62.4 | 64.8 | 60.4 |

Source: Author.

All in all, we can observe that both the multipass and singlepass methods exhibit similar properties in respect to the number of neighbors chosen. Both perform best in the medium range. Here both the algorithms reach a an average success above 65 percent. While the performance of both method in the low and high neighbor ranges oscillates between 55 and 65 percent.

The main drawback of the multipass algorithm appears to be a high difference between adjacent neighbor counts. While the algorithm performs extremely well for target neighbor count 7 and 9, it performs less than optimally for neighbor count 8. Unfortunately, we have little idea why this is the case.

On the analyzed dataset, once again the multipass algorithm outperformed the singlepass approach in all three categories. The highest difference was in the low category. This gap (5.2 percent points) was predominantly caused by the performance of the multipass algorithm with 3 neighbors and the comparatively poor performance of the singlepass algorithm with maximum neighbor count of 9. Even without these outliers the singlepass algorithm would still lag behind the multipass variant but the difference would be within two percent points. The smallest gap of 0.4 percent point was in the high neighbor count category. The difference in the medium category was 1.6 percent point.

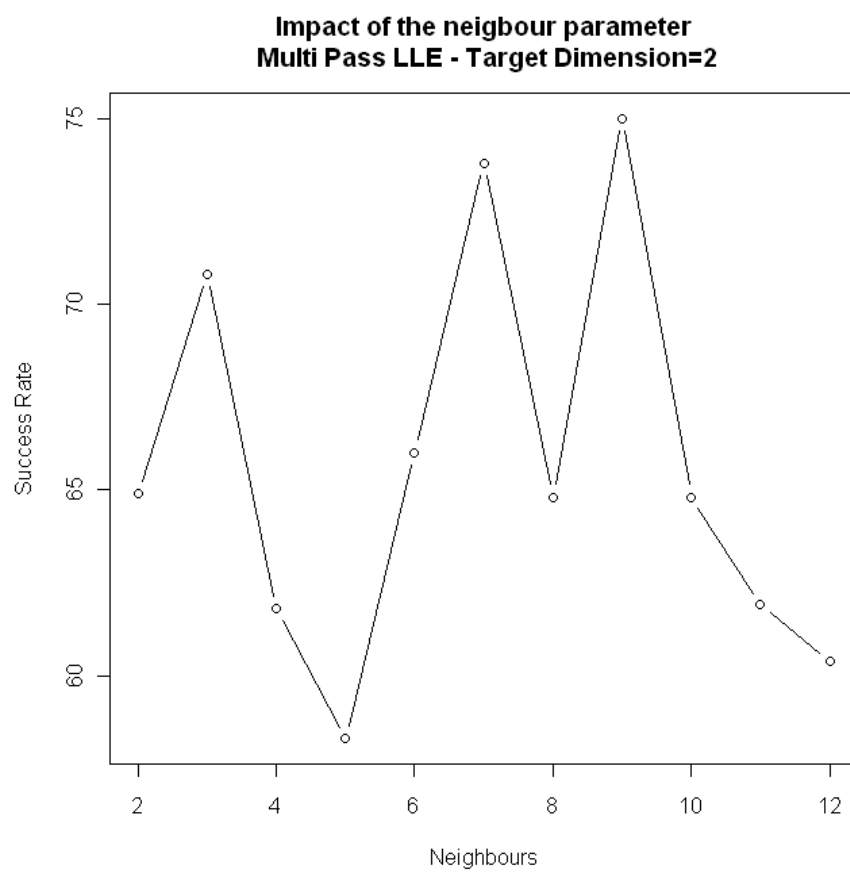


Figure 7: Multipass LLE+GPA - Impact of the neighbor factor
Source: Author.

6.6 Summary

Overall, we may conclude that multipass methods do improve the performance of the clustering algorithm on the analyzed dataset. The greatest improvement was recorded for multi-pass MDS at target dimensionality 2. We also improved the performance of the LLE+GPA by applying the multipass approach.

Furthermore, by using the multipass method we have managed to decrease the number of dimensions and neighbors required for successful classification of our dataset. While the singlepass method required from 6 to 10 dimensions to achieve optimal performance, on our dataset multipass MDS peaked at dimensionality 2. Similarly, in terms of number neighbors for the LLE+GPA algorithm the singlepass approach required between 11 and 13. The multipass algorithm performed best for neighbor counts 6 through 9.

Finally, we have determined that for multipass LLE+GPA the best performing clustering methods on our dataset were Lloyd's and Mac Queen's. On the other hand, Hartigan-Wong clustering was the best performing clustering algorithm for multipass MDS.

7 Challenges and open problems

In this thesis, we have restricted ourselves to applying multipass dimensionality reduction to locally linear embedding and multidimensional scaling. Our goal was to improve the performance of dimensionality reduction in combination with clustering algorithms.

We are aware that more testing of the multipass methods should be conducted. We would like to verify the multipass methods on several datasets with greater number of clustering groups and higher number of specimen.

However, multipass dimensionality reduction can be used in conjunction with other dimensionality reduction methods such as ISOMAP, PCA and Laplacian Eigenmaps. Furthermore, there are opportunities in combining multipass dimensionality reduction with the variants of the base algorithms such as the kernel or Hessian versions of LLE. Another possibility is using our method with non-metric multidimensional scaling. The interested reader may find further information on non-metric MDS in [1].

Even though we have not focused on supervised learning, multipass dimensionality reduction could be used in this area as well. It would be especially interesting to implement a k-NN classifier based on multipass supervised LLE and compare the results with the single pass Zhang's approach as described in [40].

In order to refine the presented results, a calculation of Spearman and Procrustes measures on the clustering output should be undertaken. For a detailed description of these methods, we refer the interested reader to Kayo's dissertation [17].

8 Conclusion

Our first goal was to analyze previous research on dimensionality reduction in the fields of morphometry and anthropology. We have determined that linear dimensionality reduction methods are prevalent. Especially principal component analysis is well studied and used both in outline and landmark analysis. On the other hand, comparatively less work is dedicated towards landmark based locally linear embedding. Furthermore, most of the work on LLE has been dedicated towards supervised locally linear embedding.

In the theoretical part of the thesis, we have described the main algorithms for dimensionality reduction and clustering. We have also discussed approaches towards landmark registration.

Having assembled the theoretical apparatus for practical analysis, we first focused on traditional non-linear dimensionality reduction techniques. In particular, we have focused on locally linear embedding and multidimensional scaling in combination with EDMA and GPA. We have tested these methods using a dataset consisting of landmarks on 3D facial scans.

We speculated that the optimal dimension for the non-linear methods would be 2 or 3. Based upon our experiments, we were forced to abandon this hypothesis. It appears that for our dataset the EDMA-based algorithms peaked at target dimensionality 8 through 10. On the other hand, the combination between GPA and LLE achieved the best preliminary result at target dimensionality 6. We concluded that finding optimal target dimensionality is a complex challenge for the user. This led us towards creating the multipass dimensionality reduction.

Our main goal was to improve the performance of the dimensionality reduction in respect to the success rate of the clustering output. Furthermore, our goal was to focus reducing the number of dimensions required for successful classification so that the user is not forced to blindly search through the dimension space in order to improve the clustering quality. We have tested three variants of the multipass algorithm based upon the singlepass methods developed in the previous chapter.

On the presented dataset, the multipass MDS at dimensionality 2 two

was the best performing algorithm overall. Moreover, we have compared the results between the multi- and singlepass LLE algorithms based on the number of neighbors used for the LLE algorithm. We have determined that the multipass algorithm outperformed the singlepass variant both in terms of maximal and average performance.

In terms of clustering algorithm choice, we have determined that for the MDS algorithm Hartigan-Wong clustering performed best. In comparison, the less complicated Forgy and Mac Queen clustering methods were more stable for the multipass LLE GPA combination. Both of these algorithms outperformed the Hartigan-Wong clustering in terms of mean success rate and variance. While the Forgy algorithm recorded a slightly higher average performance, the Mac Queen algorithm was more stable in terms of standard deviation.

For the discussed dataset, we can conclude that the multipass approach not only improved the quality of clustering but allowed to reduce the number of dimensions further than the singlepass algorithm. Finally, we have listed ideas for further research and application of the multipass dimensionality reduction algorithms.

References

- [1] S. Agarwal et al., *Generalized Non-metric Multidimensional Scaling*, *Engineering*, Vol. 1, No. 98105, 2007, pp. 11-17.
- [2] T. Baier and E. Neuwirth, *R and Friends*, University of Vienna, 2010, accessible from <http://rcom.univie.ac.at/>.
- [3] E. Berg et al. , *Fourier Descriptor-Based Deformable Models for Segmentation of the Distal Femur in CT*, *Information Processing and Security Systems*, 2005, accessible from <http://www.springerlink.com/content/u7811j05vt5h4450/>.
- [4] Y. W. Chiu, *Morphometric Analysis of Shell and Operculum Variations in the Viviparid Snail, Cipangopaludina chinensis (Mollusca: Gastropoda), in Taiwan*, *Zoological Studies*, Vol. 3, No. 41, 2002.
- [5] K. I. Christensen, *A morphometric study of the geographic variation in Pinus contorta (Pinaceae)*, *Nordic Journal of Botany*, Vol. 1, No. 23, 2005.
- [6] P. Cunningham, *Dimension Reduction*, University College Dublin, 2007, accessible from <http://www.csi.ucd.ie/files/UCD-COI-2007-7.pdf>.
- [7] I. K. Fodor, *A Survey of Dimension Reduction Techniques*, U.S. Department of Energy, 2002, accessible from <https://e-reports-ext.llnl.gov/pdf/240921.pdf>.
- [8] E. W. Forgy, *Cluster analysis of multivariate data: efficiency vs interpretability of classifications*, *Biometrics*, Vol. 21, No. 1, 1965, pp. 768–769.

- [9] P. J. F. Groenen and M. van den Welden, Multidimensional Scaling, Erasmus University Rotterdam, Econometric Institute, 2004, accessible from <http://publishing.eur.nl/ir/repub/asset/1274/ei200415.pdf>.
- [10] D. Harel and Y. Koren, *Graph Drawing by High-Dimensional Embedding*, *Journal of Graph Algorithms and Applications*, Vol. 8, No. 2, 2004.
- [11] J. A. Hartigan and M. A. Wong, *A K-means clustering algorithm*, *Applied Statistics*, Vol. 1, No. 28, 1979, pp. 100–108.
- [12] T. J. Hutton , B. F. Buxton , P. Hammond, *Automated registration of 3d faces using dense surface models*, British Machine Vision Conference, 2003, accessible from <http://www.bmva.ac.uk/bmvc/2003/papers/75/paper075>.
- [13] A. K. Jain, R. C. Dubes, *Algorithms for clustering data*, New Jersey: Prentice Hall, 1988, ISBN:0-13-022278-X.
- [14] I. T. Jolliffe, *Principal Component Analysis*, New York: Springer-Verlag, 2002, ISBN 0-387-95442-2.
- [15] C. Julien, *Morphometrics with R*, New York: Springer-Verlag, 2008, ISBN 978-0-387-77789-4.
- [16] S. Kadoury, *Face Detection Using Locally Linear Embedding*, Department of Electrical and Computer Engineering McGill University, Montreal, Canada, 2005, accessible from http://www.cim.mcgill.ca/~levine/thesis_SamuelKadoury.pdf.
- [17] O. Kayo, *Locally Linear Embedding Algorithm: Extensions and Applications*, Acta Universitatis Oulensis, 2006, accessible from <http://herkules.oulu.fi/isbn9514280415/isbn9514280415.pdf>.

- [18] D. G. Kendall, *A Survey of the Statistical Theory of Shape*, Statistical Science, Vol. 4, No. 2, 1989.
- [19] J. B. Kruskal, *Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis*, Psychometrika, Vol. 1, No. 29, 1965.
- [20] C. L. Lee and S. Y. Chen, Classification for Leaf Images, Conference on Computer Vision, Graphics and Image Processing, 2003, accessible from <http://www.csie.mcu.edu.tw/~yklee/CVGIP03/CD/Paper/PR/PR-19.pdf>.
- [21] J. de Leeuw and P. Mair, *Multidimensional Scaling Using Majorization: SMACOF in R*, UCLA, 2008, accessible from <http://preprints.stat.ucla.edu/537/smacof.pdf>.
- [22] S. Lele and J. T. Richtsmeier, *Euclidean Distance Matrix Analysis: A Coordinate-Free Approach for Comparing Biological Shapes Using Landmark Data*, American Journal of Physical Anthropology, No. 86, 1991, pp. 415-427.
- [23] S. Lele and J. T. Richtsmeier, *Euclidean Distance Matrix Analysis: Confidence Intervals for Form and Growth*, American Journal of Physical Anthropology, No. 98, 1995, pp. 73-86.
- [24] C. S. Lina and C. L. Hwang, New forms of shape invariants from elliptic Fourier descriptors, Pattern Recognition Vol. 20, No. 5, 1987.
- [25] S. P. Lloyd, *Least squares quantization in PCM*, IEEE Transactions on Information Theory, Vol. 1, No. 28, 1957, pp. 128-137.
- [26] J. Mac Queen, Some methods for classification and analysis of multivariate observations, Proceedings of the

- Fifth Berkeley Symposium on Mathematical Statistics and Probability, Vol. 1, pp. 281–297.
- [27] R. Memisevic and G. Hinton, *Improving dimensionality reduction with spectral gradient descent*, Neural Networks, No. 18, 2005, accessible from <http://learning.cs.toronto.edu/~rfm/pubs/sg.pdf>.
- [28] Microsoft, *.NET Framework 3.5 SP1*, 2010, accessible from <http://www.microsoft.com/downloads/cs-cz/details.aspx?FamilyID=333325fd-ae52-4e35-b531-508d977d32a6>.
- [29] Morphometrics team, *Morphometrics I Programmer's Reference*, 2010, accessible from <http://cgg.mff.cuni.cz/trac/morpho/attachment/wiki/WikiStart/ProgrammersReferenceGuide.pdf>.
- [30] Morphometrics team, *Morphometrics I User Manual*, 2010, accessible from <http://cgg.mff.cuni.cz/trac/morpho/attachment/wiki/WikiStart/UserManual.pdf>.
- [31] Morphometrics team, *Morphometrics II Developer Manual*, 2011, accessible from <http://cgg.mff.cuni.cz/trac/morpho/attachment/wiki/WikiStart/3-developer-manual.pdf>.
- [32] Morphometrics team, *Morphometrics II User Manual*, 2011, accessible from <http://cgg.mff.cuni.cz/trac/morpho/attachment/wiki/WikiStart/2-user-manual.pdf>.
- [33] K. Pearson, *On Lines and Planes of Closest Fit to Systems of Points in Space*, Philosophical Magazine, Vol. 2 No. 6, 1901, pp. 559–572.

- [34] R Development Core Team, *R: A language and environment for statistical computing, reference index version 2.13.1*, R Foundation for Statistical Computing, Vienna, Austria, 2011, accessible from <http://www.R-project.org>.
- [35] F. J. Rohlf and L. F. Marcus, *A Revolution in Morphometrics*, TREE, Vol. 8, No.4, 1993, pp. 129-132.
- [36] D. de Ridder and R. Duin, *Locally linear embedding for classification*, Faculty of Applied Science, Delft University of Technology, The Netherlands, 2002, accessible from <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.110.7308&rep=rep1&type=pdf>.
- [37] L. K. Saul and S. T. Roweis, *Nonlinear dimensionality reduction by locally linear embedding*, Science, Vol. 290, No.5500, 2000.
- [38] L. K. Saul and S. T. Roweis, *An introduction to Locally Linear Embedding*, AT&T Labs, 2000, accessible from <http://www.cs.nyu.edu/roweis/lle/papers/lleintro.pdf>.
- [39] B. Scholkopf, A. Smola and K. R. Muller, *Nonlinear Component Analysis as a Kernel Eigenvalue Problem*, Max-Planck Institute for Cybernetics, Tübingen, 1996, accessible from http://www.face-rec.org/algorithms/Kernel/kernelPCA_scholkopf.pdf.
- [40] S. Zhang, K. W. Chau, *Dimension reduction using semi-supervised locally linear embedding for plant leaf classification*, Lecture Notes in Computer Science, 5754 LNCS, 2009, pp. 948-955.

List of Tables

| | | |
|----|--|----|
| 1 | Impact of dimensionality reduction - success rate in percent | 37 |
| 2 | Variability of the MDS measurements | 45 |
| 3 | Success rate for the GPA/LLE measurements | 52 |
| 4 | Success rate of the EDMA/LLE algorithm | 52 |
| 5 | Success rate of the algorithms for target dimensions 2 through 5 | 52 |
| 6 | Variability of the MDS measurements | 53 |
| 7 | Multipass dimensionality reduction | 57 |
| 8 | Variability of results | 58 |
| 9 | Clustering Algorithm Performance - multipass MDS | 60 |
| 10 | Clustering Algorithm Performance - multipass LLE+GPA | 61 |
| 11 | Multipass MDS - target dimension | 64 |
| 12 | Aggregate results - singlepass LLE+GPA | 65 |
| 13 | Aggregate results - multipass LLE+GPA | 68 |

List of Figures

| | | |
|----|---|----|
| 1 | Landmark placements | 34 |
| 2 | Comparison between behavior of LLE and MDS | 39 |
| 3 | Multidimensional scaling - selecting the optimal target dimension | 44 |
| 4 | LLE/EDMA - selecting the optimal target dimension | 48 |
| 5 | Comparison between singlepass and multipass MDS | 63 |
| 6 | Singlepass LLE+GPA - Impact of the neighbor factor | 66 |
| 7 | Multipass LLE+GPA - Impact of the neighbor factor | 69 |
| 8 | Installation of R packages | 85 |
| 9 | Selecting the install folder for Morphome3cs | 85 |
| 10 | Selecting the Single Run script from Morphome3cs menu | 87 |
| 11 | GPA Tab | 87 |
| 12 | Dimensionality Reduction Tab | 88 |
| 13 | Clustering Tab | 89 |
| 14 | Specimen Editor | 90 |
| 15 | Selection of the Classification Attribute | 90 |

A Morphome3cs

A.1 Overview and aim of the Morphome3cs project

In this part we will describe the basic aim of the Morphometrics project as well as demonstrate a need for a unifying platform for morphometric research.

The Morphome3cs project was proposed as a Software project at the Faculty of Mathematics and Physics of Charles University in Prague. The specification was based on the requirements of researchers from the Faculty of Science, especially from the Department of Anthropology and Human Genetics.

When development was started these researchers were forced to use numerous applications in order to perform a single experiment. One application was used to acquire the landmarks, in the next program the was modified and finally a third program was used for statistical analysis. Often these applications were poorly maintained and documented. As a result the aim of the Morphome3cs projects was to offer an open and modifiable platform that would integrate most of the methods used in morphometric studies. See also [29].

The main goal was to allow for both a smooth data acquisition as well as usage of mathematical and statistical methods for users without deep mathematical knowledge. The framework was also developed for science students as an educational tool in morphometrical analysis. Furthermore, the framework was successfully used for several bachelor and diploma thesis on both faculties as well as a tool in academic research papers. For further information, refer to [31] and [29].

A.2 Users of the Morphome3cs platform

In this section we will describe the two main user groups of the Morphome3cs framework.

The users in the first group are those who mainly use Morphome3cs for research or study purposes. They use the prepared morphometric methods without requiring to understand the inner mechanism of both the framework

as well as the morphometric methods. As such no programming or mathematical knowledge is required for these users.

The second user group are responsible for administering applications, managing computation environments and preparing computation schemes/guides for the users from the first group. In short, they prepare methods for the first group of users, but they can also utilize the system for their own research. This scripting is performed in Python and therefore basic programming skills are required. Finally, this user-programmer should have both understanding of the inner mechanisms of Morphome3cs platform as well as knowledge of the mathematical methods. Additional information are provided in [29].

A.3 Statistical computation in Morphome3cs

In order to facilitate implementation of statistical method, Morphome3cs relies on the open-source statistical library R available from [34]. While it would be possible to directly execute R commands from C#, access to the R functionality is facilitated using the rcom library. This library is available from [2].

The main functionalities of the binding are as follows:

- Transfer data between C# code and R
- Allow C# code access to statistical methods of R
- Allow C# code access to R plotting methods

At the start of the Morphome3cs application several R processes are initialized. Access to these process can then be acquired via a manager. Access to the interpreter is exclusive and no other thread can interfere with the acquired R interpreter. Using the interpreter the C# thread can execute R commands and transfer data to and from R as described in [29].

Once the interpreter was released there is no guarantee that data and variables will be preserved within the R interpreter. While there could be an argument for persistence (see the function call transfer issues) this would complicate the implementation and usage severely in most cases. There

would mainly exist no guarantee as to which variables are used and can safely be accessed. Moreover, data transfer between C# and R is not time critical and thus using shared memory for the interpreters or persistent variables between acquire calls was rejected.

Limitations of the binding

Unfortunately the binding system does not currently support creation of external graphical devices of R or its other interactive features. However, the native R plotting devices can still be used and their output stored to various image formats.

Finally, R objects containing function calls cannot currently be completely transferred to C# (one example is the `lda` object). If the function call is not required to perform further R operations this can be neglected. This has however no impact on further computation within C#. A possible workaround is to perform all R-related computation within a single C# filter without releasing the R interpreter.

B Installing Morphome3cs

Here we will describe how to install Morphome3cs. The installation procedure can be divided in two parts. First the prerequisites of Morphome3cs are installed, then Morphome3cs proper is installed. The prerequisites of Morphome3cs are:

- Windows Installer 3.1
- .NET Framework v. 3.5
- RAndFriends v. 2.13.1

These are provided on the installation DVD. If you run the setup.exe program from the DVD distribution these programs will be automatically installed if they are not present on your PC. Alternatively you can download these programs yourself and manually install them. RAndFriends is available from [2]. The current version of .NET framework is available at [28].

After installing the prerequisites of Morphome3cs, you are ready to install the main application. This is a straightforward process. In the first screen 8 you are asked whether or not you want to install the necessary R packages. You should generally allow the packages to be installed if you wish to use the R-based scripts. You can download them anytime by running the RInstallPackages.exe script from the installation directory. Finally, you are asked to select the installation folder in the screen 9.

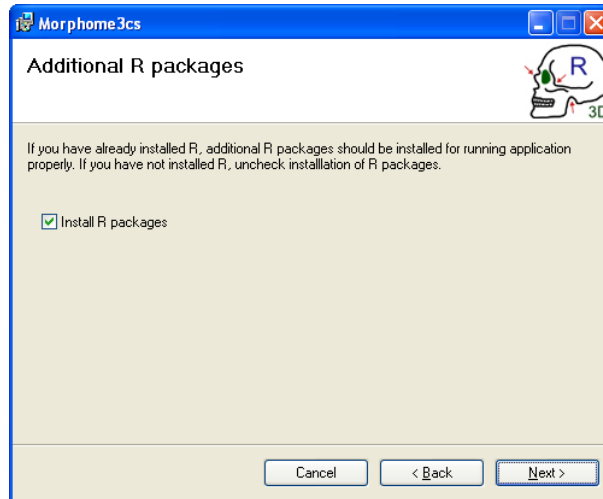


Figure 8: Installation of R packages
Source: Author.

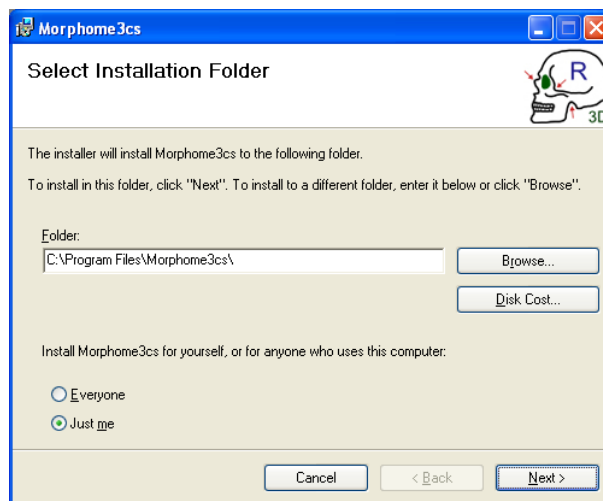


Figure 9: Selecting the install folder for Morphome3cs
Source: Author.

C Reproducing the results

In this section, we will describe how to reproduce the results of the thesis using the Morphometr3cs software.

C.1 GUI-based scripts

The first way to test the described methods is by experimenting with their settings interactively in a GUI-based Morphome3cs workflow. There are six such workflows:

- Multipass MDS
- Multipass LLE+GPA
- Multipass LLE+EDMA
- Singlepass MDS
- Singlepass LLE+GPA
- Singlepass LLE+EDMA

These workflows can be run from the Morphome3cs Workflow menu. First, the method(MDS, LLE+GPA, LLE+EDMA) is selected. Then you select whether to use single- or multipass algorithm. Finally, you choose the option Single Run from the menu. This is shown on the figure 10.

Once the filter was executed several tabs appear on the screen. In the first tab, you can select the dataset to be used in the experiment. This is typically not needed since you get to select it anyway in the specimen editor. For the GPA-based filters the tab 11 is displayed. In this tab, you can select whether to normalize the dataset in respect to reflection and scale. For the default dataset, we chose not to normalize in respect to scale and reflection since all the images were already scaled and oriented in respect to reflection during the data acquisition phase.

In the LLE tab you can select the parameters of the dimensionality reduction algorithm. The figure 12 shows the options for the multipass LLE

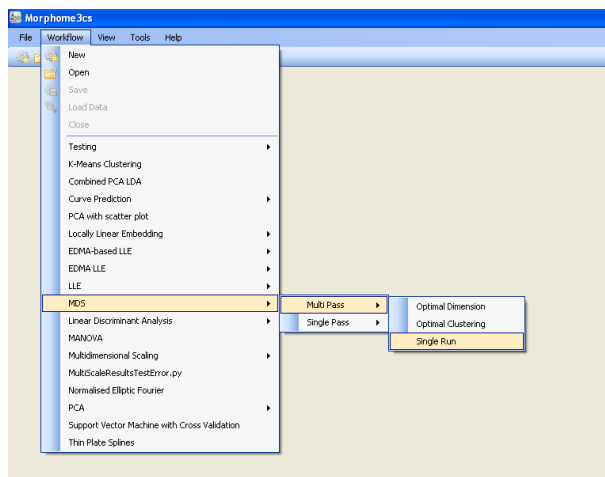


Figure 10: Selecting the Single Run script from Morphome3cs menu
Source: Author.

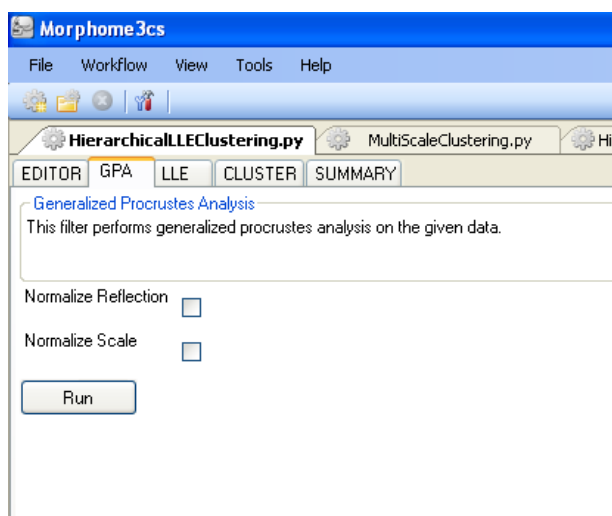


Figure 11: GPA Tab
Source: Author.

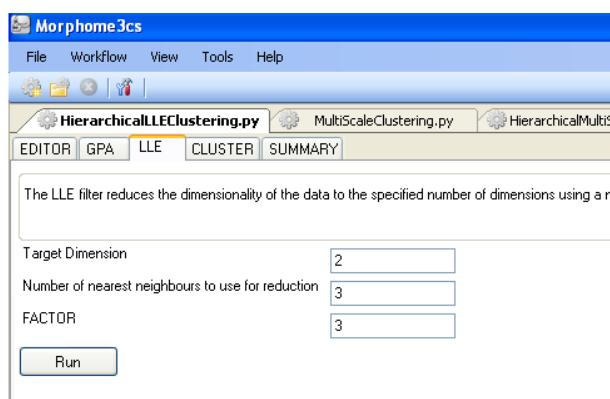


Figure 12: Dimensionality Reduction Tab

Source: Author.

variant. Here you can select the target dimensionality and the number of neighbors. For MDS, the number of neighbors is not present. Finally, the reduction factor field determines the speed with which dimensions are reduced in the multipass algorithm. For example, if you increase this number to 4, every time the target dimension for the next step of the algorithm will be divided by 4. This option is only present for multipass methods.

The clustering tab allows you to select the number of clusters to be used in the clustering algorithm and the appropriate clustering algorithm. You can see it on the figure 13. Finally, on the summary card you should select the file to which you want to write the report. This report contains the success rate and the classification of the specimens. In addition to it, the centers of the clusters are also given. Having select the report path, the next step is pressing the run button on the summary tab.

After pressing the Run button the specimen editor will appear. In this powerful tool you can among others add specimen to the dataset, change their properties and locate landmarks. For details refer to the Morphome3cs user manuals [30] and citeMORPHO2USER. Here we will restrict ourselves to merely selecting the provided dataset. To perform this action, select Open Specimens from the Editor menu and open the provided sample. Wait for the program to load the specimen table. The state of the specimen editor

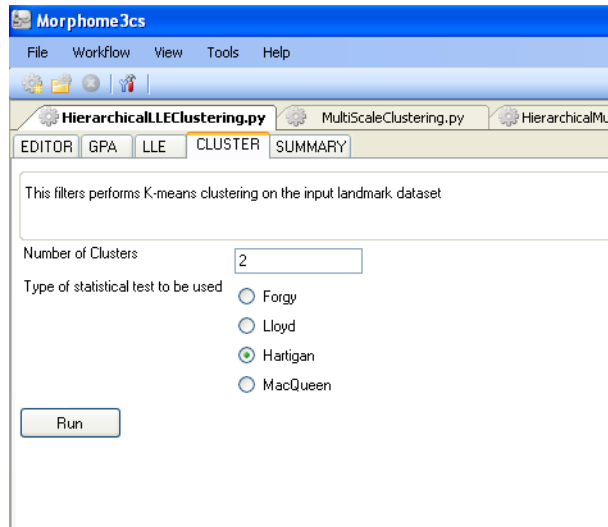


Figure 13: Clustering Tab

Source: Author.

should then resemble figure 14. Afterwards close the specimen editor.

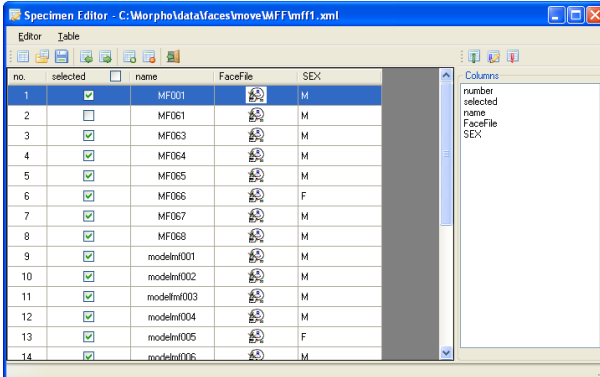
A window(as depicted in figure 15) will pop up asking you to select the attribute from the specimen table that determines the clusters in the dataset. This is only used for calculating the success rate in the report. Select SEX and press OK. Finally, the above described report will be produced.

C.2 Scripts without GUI

The other way of reproducing the results of this thesis is running the non-interactive python scripts in Morphome3cs. For each of the algorithms several methods are provided:

- Finding optimal dimension
- Finding optimal clustering method
- Finding optimal optimal neighbor count (LLE only)

The file for each of the scripts can be invoked from the Morphome3cs Workflow menu. These scripts only ask you to select the dataset and the clas-



The screenshot shows a window titled "Specimen Editor - C:\Morpho\data\faces\move\WFF\mf1.xml". The window contains a table with the following columns: "no.", "selected", "name", "FaceFile", and "SEX". The table lists 14 specimens, with the first 8 being MF001 through MF008 and the last 6 being modeln001 through modeln006. The "SEX" column contains "M" for most specimens and "F" for MF006 and modeln005. A "Columns" panel on the right lists "number", "selected", "name", "FaceFile", and "SEX".

| no. | selected | name | FaceFile | SEX |
|-----|-------------------------------------|-----------|----------|-----|
| 1 | <input checked="" type="checkbox"/> | MF001 | | M |
| 2 | <input type="checkbox"/> | MF001 | | M |
| 3 | <input checked="" type="checkbox"/> | MF003 | | M |
| 4 | <input checked="" type="checkbox"/> | MF004 | | M |
| 5 | <input checked="" type="checkbox"/> | MF005 | | M |
| 6 | <input checked="" type="checkbox"/> | MF006 | | F |
| 7 | <input checked="" type="checkbox"/> | MF007 | | M |
| 8 | <input checked="" type="checkbox"/> | MF008 | | M |
| 9 | <input checked="" type="checkbox"/> | modeln001 | | M |
| 10 | <input checked="" type="checkbox"/> | modeln002 | | M |
| 11 | <input checked="" type="checkbox"/> | modeln003 | | M |
| 12 | <input checked="" type="checkbox"/> | modeln004 | | M |
| 13 | <input checked="" type="checkbox"/> | modeln005 | | F |
| 14 | <input checked="" type="checkbox"/> | modeln006 | | M |

Figure 14: Specimen Editor
Source: Author.

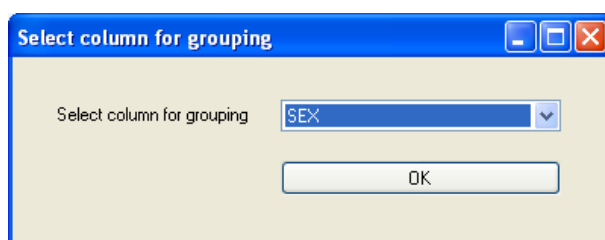


Figure 15: Selection of the Classification Attribute
Source: Author.

sification parameter as was discussed in the previous section. The output is then written to the Results directory in the Morphome3cs installation folder. You may change the directory to which the output will be written in the Morphome3cs configuration. The application must be restarted before these changes take place. Alternatively you can directly edit the scripts. For details see Morphometrics programmer's guides [29] and [31]. The scripts are located in the PythonScripts directory in the Morphome3cs installation directory.

It should be noted that these scripts are targeted towards user-programmer rather than the user-student group. These scripts may require some constants such as maximal dimensionality to be edited in the Python script so that they function optimally on datasets other than the one provided.

D Note on the used configuration

For all the measurements in this thesis, the following configuration was used:

- CPU: AMD Athlon64 3500+
- RAM: 1024 MB
- GPU: ATI Radeon 1900
- OS: MS Windows XP SP2
- MS Visual Studio 2008 SP 3.5
- RAndFriends version 2.9.2

It should be noted that this is far from modern hardware and that all the methods run smoothly. Especially the processor is single-core. While Morphome3cs allows for multi threading the methods were not optimized for multi-threaded environment. Similarly, the RAndFriends version is not the most recent one but Morphome3cs should run without problems on versions greater than 2.8.1. Nevertheless, the current version 2.13.1 is recommended.

E Contents of the DVD

- *setup.exe*, *Morphome3cs.msi* - the Morphome3cs installer
- *thesis.pdf* - this file
- *src.zip* - zipped source code of Morphome3cs
- *manuals* directory - contains the user and developer guides for Morphome3cs 1 and 2
- *data* directory - contains the test dataset and the specimen editor file *test.xml*
- *DotNetFx* directory - .NET Framework installer
- *WindowsInstaller3_1* directory - Installer setup
- *RAndFriends* directory - RAndFriends installer