Similarity Map Add-in for LibreOffice Calc

Group 5

Karmen Gostiša
Anastasiia Klimashevskaia
Jaka Konda
Günther Moser

Institute of Interactive Systems and Data Science (ISDS),
Graz University of Technology
A-8010 Graz, Austria

30 Jun 2019

Abstract

This project paper presents the LibreOffice Calc Add-in implementation of a Similarity Map, a visualisation technique used for 2D projection of multidimensional data. Several implemented dimensionality reduction algorithms are discussed and instructions on how to install and use the Add-in are provided.

Contents

List of Figures

Chapter 1

Introduction

Nowadays datasets may contain a lot of attributes that are difficult to visualise. Because of that, it is important to reduce the number of attributes by using dimensionality reduction techniques. Using feature projection, we transform the data from high-dimensional space to low. Projecting the data in lower-dimensional space (such as 2 or 3-dimensional) is useful for creating visualisations that may be of great help to gain insight or better understand complex dataset, such as identifying clusters or outliers. A 2D visualisation technique where distances between plotted points represent the similarity, is called a Similarity Map.

This project work presents the LibreOffice Calc Add-in implementation of a Similarity Map. Add-in provides a way to create Similarity Maps using different dimensionality reduction algorithms such as PCA, MDS, Isomap and t-SNE.

The paper is organised as follows: first LibreOffice Calc is presented, describing the user interface and features. Afterwards, Similarity Map technique is explained and a theoretical overview of implemented dimensionality reduction algorithms is provided. Then, the Similarity Map Calc Add-in implementation is presented and instructions on how to install and use it are provided. In the end, a conclusion about the development of Add-in is drawn.

Chapter 2

LibreOffice Calc

LibreOffice Calc [*LibreOffice Calc* 2019] is a spreadsheet component of the office productivity suite LibreOffice [*LibreOffice* 2019]. It is freely available and open-source software, released under the Mozilla Public License. Calc can be expanded by Add-ins, external modules that implement additional functions for working with spreadsheets.

2.1 User interface

LibreOffice provides different user interface options and icon styles. The development of user interfaces is moving towards modern options such as tabbed or grouped icons [*LibreOffice Help* 2019]. The traditional standard toolbar (see Fig. 2.1) is also available.

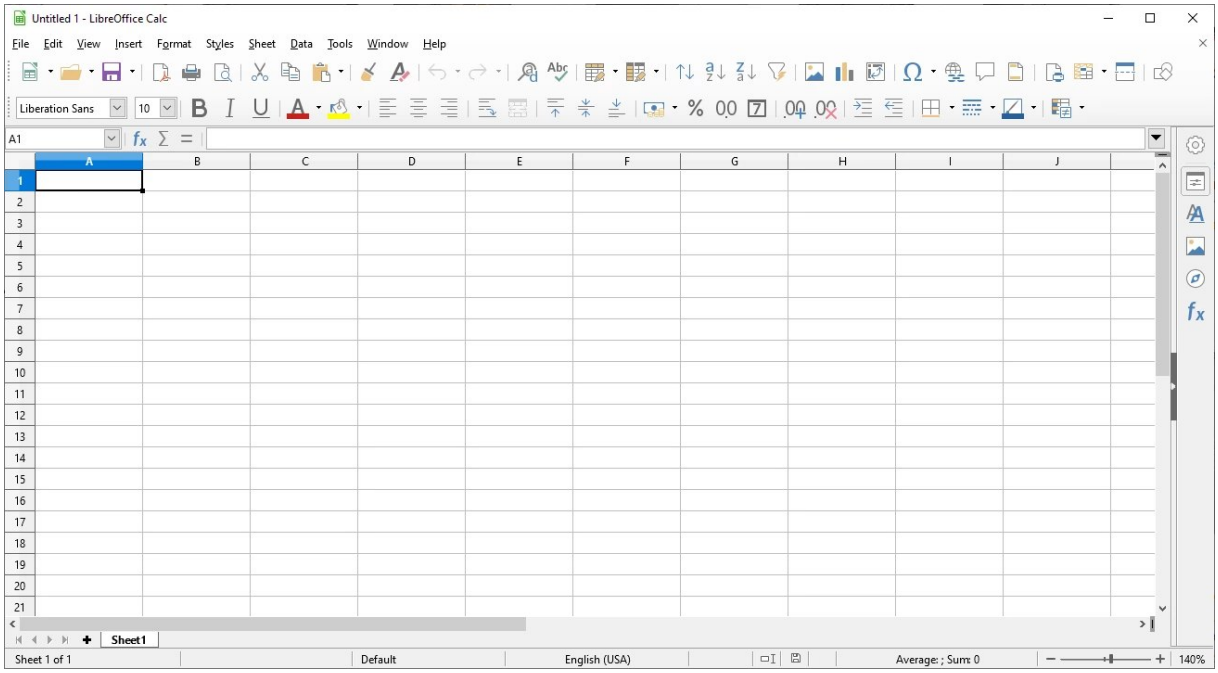


Figure 2.1: Screenshot of the LibreOffice Calc Standard Toolbar interface.

2.2 Features

Features provided by LibreOffice Calc include [*Calc Filters* 2019]:

- Functions to create formulas for complex calculations on data, including those for imaginary numbers [*Add-in Functions* 2019].
- Database functions to filter, store and arrange data.
- A wide range of dynamic 2D and 3D charts.
- Macros to automate common repetitive tasks using scripting languages such as LibreOffice Basic, JavaScript, Python and BeanShell.
- Ability to open, edit or save OpenDocument, Excel, CSV, PDF and several other formats.

Chapter 3

Similarity Map and Algorithms

This Chapter describes the Similarity Map visualisation and provides the overview of implemented algorithms used to create such visualisations.

3.1 Similarity Map

A Similarity Map is a visualisation technique used for 2D projection of multidimensional data. The distances between plotted points represent the similarity - the more similar the points are, the closer they are.

3.2 Implemented algorithms

Different algorithms can be used to create a Similarity Map. In this Section we present the ones that were implemented in this project work.

3.2.1 Principal Component Analysis (PCA)

The first and probably the most standard method used for dimensionality reduction is Principal Component Analysis (PCA). In geometrical terms, PCA reduces the dimension of a dataset by projecting it onto a lower-dimensional subspace that retains as much of the original dataset characteristics as possible. Smaller datasets are easier to analyse and visualise. PCA does not involve any randomness so it is classified as a deterministic algorithm.

It comprises four steps [Holland no date]:

1. **Standardisation:** In order to prevent variables with larger ranges to dominate over those with small ranges, data must be transformed to comparable scales. Using maths, this can be done by subtracting the mean (μ) from value of each variable (x) and dividing by the standard deviation (σ):

$$z = \frac{x - \mu}{\sigma} \tag{3.1}$$

2. **Computation of Covariance Matrix:** Sometimes variables are so highly correlated they contain redundant information. A covariance matrix helps to identify relationships and correlations between variables. It is a symmetric matrix with the length of the number of dimensions and it contains the covariances associated with all possible pairs of the initial variables. An example of covariance matrix for 3-dimensional data is

$$\begin{vmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{vmatrix}$$

Covariance is commutative, so the upper and lower triangular areas are equal, and covariance of a variable with itself is its variance, so in the diagonal there are variances of each initial variable. The correlation can be determined by looking at the sign of covariance; if it is positive, then the two variables are correlated in a way they increase or decrease together; or if it is negative, variables are inversely correlated, meaning one increases when the other decreases.

3. **Identification of principal components:** To determine new variables that retain as much information of initial variables as possible, also called *principal components*, we need to compute eigenvectors and eigenvalues. The idea is to do such linear combination of the initial variables, so principal components are uncorrelated and first components hold most of information, ie. the largest variance. Dimensionality reduction can then be done by discarding components with low information. Eigenvectors of the covariance matrix are *directions* of axes with most information, that we call principal components. Eigenvalues represent eigenvectors coefficients that give the *amount* of variance in each principal component. To get principal components in order of significance, the ranking of eigenvectors in order of their eigenvalues should be done.

4. **Dimensionality reduction:** Eigenvectors of the components that we decide to keep, form a matrix called *feature vector*. In the last step, we want to reorient the data from original axes to the ones represented by the principal components. This can be done by multiplying the transpose of the feature vector (λ) by the transpose of the original data (X) set as follows:

$$X_t = \lambda^T * X^T \tag{3.2}$$

3.2.2 Multidimensional Scaling (MDS)

Multidimensional scaling is a visual representation of similarity between individual cases of a dataset. Objects that are more similar have shorter distances so they are drawn closer together than objects that are less similar (and have larger distances). Besides interpreting similarity on a graph, MDS is also used for a dimensionality reduction.

The basic steps of the method include [Buja et al. 2008]:

1. **Assigning points** to random coordinates in multidimensional space, typically 2 or 3-dimensional, as higher dimensional spaces are difficult to model.

2. **Calculating Euclidean distances** for all pairs of points and construct similarity matrix.

3. **Comparing the similarity matrix with the original input matrix** by evaluating the stress function that measures differences between predicted and actual distances. The smaller the value, the greater the correspondence.

4. **Adjusting coordinates** of each point in a direction that best minimise stress.

5. **Repeating steps** from 2 to 4 until stress stays the same.

The random number generator is used in Step 1 to initialise coordinates, making MDS a non-deterministic algorithm.

3.2.3 Isomap

Isomap is an instance of isometric mapping methods that extend MDS (see Subsection 3.2.2) by trying to preserve geodesic distances in the lower dimension. The classic MDS uses Euclidean distance to estimate the similarity between data points. This distance holds good only if neighbourhood structure can be approximated as linear, while Isomap introduces the geodesic distance as graph distance that also works well in non-linear manifolds (see Fig. 3.1). Like MDS, Isomap is also a non-deterministic algorithm. It actually differs from MDS in initial steps only [*Dimension Reduction - IsoMap* 2019].

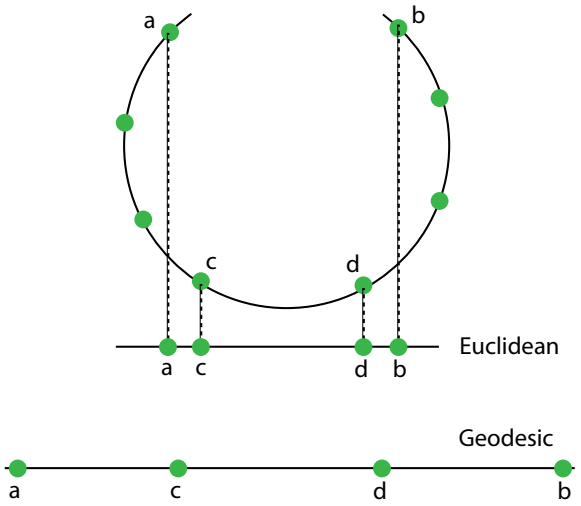


Figure 3.1: The Euclidean and geodesic distances on a 2D circular manifold. Redrawn from [*Dimension Reduction - IsoMap* 2019].

The Isomap algorithm is comprised of the following stages [Tenenbaum et al. 2000]:

1. **Determining neighbours of each point:** Neighbours can either be all points in some fixed radius or K-nearest neighbours.

2. **Constructing neighbourhood graph:** Each point is connected to other based on a criteria chosen in Step 1 using Euclidean distance as the edge length.

3. **Computing the shortest path between two nodes:** The geodesic distance is approximated as the sum of edge weights along the shortest path between two nodes (computed using Dijkstra's algorithm, for example). In this Step we obtain a similarity matrix.

4. **Computing lower-dimensional embedding:** Low-dimensional embedding is then achieved the same way as in MDS (see Step 3 onward in Subsection 3.2.2).

3.2.4 T-distributed Stochastic Neighbor Embedding (t-SNE)

Drawing visualisations, we are often interested in drawing only objects that are similar. We are less interested in objects that are dissimilar or far apart. The method that focuses on preserving only local similarities is t-SNE. It is a non-linear technique used for visualising high-dimensional data and data exploration. A t-SNE algorithm tries to optimise the two similarity measures between pairs of instances in the high and low-dimensional space using a cost function. Because it uses a gradient descent optimisation that is initiated randomly, the t-SNE is a non-deterministic algorithm, however randomness can be eliminated using manually initialisation such as output from the PCA.

The algorithm includes following steps [Maaten and Hinton 2008]:

1. **Creating a probability distribution** that describes relationships between neighbouring points. For each point in the high-dimensional space, a Gaussian distribution is centered on that point and the probability density of all points under that Gaussian distribution is measured. This is done pairwise and similar data points will have more similar values under this Gaussian circle.

2. **Recreating a low-dimensional space** that best expresses the probability distribution from Step 1. This Step is similar to the previous, except is uses a Student t-distribution with one degree of freedom that is also called the Cauchy distribution. Another set of probabilities is obtained, but this time for the low-dimensional space.

3. **Optimising the low-dimensional distribution** in the way the two map structures are as similar as possible. The difference between probability distributions of the two-dimensional spaces is measured using Kullback-Liebler (KL) divergence. The optimisation is done using gradient descent minimising the KL cost function.

Chapter 4

Expanding LibreOffice Calc with Similarity Map Add-in

In this Chapter, the most important parts of the developed Add-in are described.

4.1 Required Add-in structure

Each LibreOffice Add-in must follow a specific structure in order to be successfully installed. Typically, the structure includes following files:

- **description**: a folder, containing files with Add-in description;
- **META-INF**: a folder, containing the *manifest.xml* file that specifies the location of the source code and *Addons.xcu* file;
- **images**: a folder that contains Add-in graphics;
- **src**: a folder with all Add-in source code;
- **Addons.xcu**: a file, specifying toolbar items and menu entries;
- **description.xml**: a file that provides paths to license and description;
- **LICENSE.txt**.

The required files are *manifest.xml* and *description.xml*.

The OXT is OpenOffice extension file used to represent extensions for the Office suite. The OXT file format follows the ZIP specification, therefore the final package of Add-in is made by zipping the root folder and renaming the file extension to .oxt.

4.1.1 Programming Languages

LibreOffice extension can be developed using several programming languages, such as C++, Python, Java, Javascript and LibreOffice Basic. It is possible to contribute extensions on the official web page for Libre Office extensions [*LibreOffice Extensions* 2019]. Most of extensions there are under Attribution-ShareAlike 3.0 Unported (CC BY-SA 3.0) license.

4.1.2 Extension description

For a valid extension only one file is needed, namely *description.xml*. A minimal version of it is shown in Listing 4.1.

As the minimal version provides little information, it can be extended using properties as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<description xmlns="http://openoffice.org/extensions/description 2006"
  xmlns:d="http://openoffice.org/extensions/description 2006"
  xmlns:l="http://libreoffice.org/extensions/description/2011"
  xmlns:xlink="http://www.w3.org/1999/xlink">
</description>
```

Listing 4.1: Minimal content of the *description.xml* file.

- **Version:** the current version of extension;
- **Identifier:** the unique name for the LibreOffice Extension Manager;
- **Platform:** a list of supported operating systems;
- **Display Name:** the Add-in name;
- **Publisher:** a list of authors.

The final file *description.xml* of Similarity Map Add-in is shown in Listing 4.2.

4.1.3 Manifest

The *manifest.xml* file must be in the META-INF directory. Absence of this file prevents Extension Manager to remove the extension properly. It also contains full paths to the source code and the .xcu files defining the menu buttons. The manifest of Similarity Map Add-in is seen in Listing 4.3.

4.1.4 Menu and Toolbar

Button can be created using a .xcu file, that is basically an XML file. Buttons are defined within the root node AddonUI. Custom button can be defined as shown in Listing 4.4.

One of the most important parameters is the context value. In this example it defines operations that will be performed over currently opened spreadsheet document. The URL value defines a service that gets called when user invokes the button. In our case, the service is defined in *SIMMAP.py* file.

The base class is located in *SIMMAP.py* file and when someone clicks on the button, the function execute gets called. In the constructor, forms are created and click listeners are registered to receive user actions.

The file *CalcWindowsState.xcu* [4.6] creates a toolbar button and then connects it to the *Addons.xcu* file that in turn executes the service.

4.2 Connecting to LibreOffice

All of the communication between LibreOffice and Python is done using the UNO model. It offers two approaches. The first is interactive that allows developer to connect to the LibreOffice through an interactive Python shell. The other is intended for developed Add-ins, where the context is retrieved from the LibreOffice Extension Manager.

To connect using an interactive approach, first a LibreOffice server must be started by navigating to the LibreOffice installation folder and then to *program* subfolder. Next, a user can connect to this server using an interactive Python shell. This is performed by two shell commands as listed in 4.7

In the shell, we first need to import required packages and then establish a connection to the service. Once connected, a service manager is opened, obtaining the program context named *desktop* that handles

```
<?xml version="1.0" encoding="UTF-8"?>
<description xmlns="http://openoffice.org/extensions/description/2006"
  xmlns:d="http://openoffice.org/extensions/description/2006"
  xmlns:l="http://libreoffice.org/extensions/description/2011"
  xmlns:xlink="http://www.w3.org/1999/xlink">
  <version value="1.0.0"/>
  <identifier value="javahelps.simmap"/>
  <icon>
    <default xlink:href="images/logo.png" />
  </icon>
  <display-name>
    <name>Similarity Map</name>
    <name lang="en">Similarity Map</name>
  </display-name>
  <extension-description>
    <src xlink:href="pkg-desc/desc_en.txt"/>
    <src xlink:href="pkg-desc/desc_en.txt" lang="en" />
  </extension-description>
  <dependencies>
    <l:LibreOffice-minimal-version value="4.0"
    d:name="LibreOffice.org or Open Office 4.0"/>
  </dependencies>
  <registration>
    <simple-license accept-by="admin">
      <license-text xlink:href="LICENSE.txt" />
      <license-text xlink:href="LICENSE.txt" lang="en-US" />
    </simple-license>
  </registration>
</description>
```

Listing 4.2: The *description.xml* file of the Similarity Map Add-In.

```
<manifest:manifest xmlns:manifest="http://openoffice.org/2001/manifest">
<manifest:file-entry manifest:full-path="Addons.xcu"
  manifest:media-type="application/vnd.sun.star.configuration-data"/>
<manifest:file-entry manifest:full-path="Office/UI/CalcWindowState.xcu"
  manifest:media-type="application/vnd.sun.star.configuration-data"/>
<manifest:file-entry manifest:full-path="SIMMAP.py"
  manifest:media-type="application/vnd.sun.star.uno-component;type=Python"/>
</manifest:manifest>
```

Listing 4.3: The manifest file of Similarity Map Add-in.

```
<node oor:name="AddonMenu">
    <node oor:name="SimMap.AddonMenu.M1" oor:op="replace">
        <prop oor:name="Context" oor:type="xs:string">
            <value>com.sun.star.sheet.SpreadsheetDocument</value>
        </prop>
        <prop oor:name="URL" oor:type="xs:string">
            <value>service:vnd.simmap?simmap</value>
        </prop>
        <prop oor:name="Title" oor:type="xs:string">
            <value>Similarity Map</value>
        </prop>
        <prop oor:name="Target" oor:type="xs:string">
            <value>_self</value>
        </prop>
    </node>
</node>
```

Listing 4.4: XML definition for menu action in Similarity Map Add-in.

```
g_ImplementationHelper = unohelper.ImplementationHelper()
g_ImplementationHelper.addImplementation(SIMMAP, 'vnd.simmap', ('com.sun.star.task.Job',),)
```

Listing 4.5: An example of a UNO service.

```
<node oor:name="UIElements">
      <node oor:name="States">
            <node oor:name="private:resource/toolbar/addon_SimMap.OfficeToolBar.T1"
            oor:op="replace">
                  <prop oor:name="UIName" oor:type="xs:string">
                        <value xml:lang="en">Similarity Map</value>
                  </prop>
                  </node>
            </node>
    </node>
```

Listing 4.6: Definition of the toolbar button.

```
./soffice --calc \
  --accept="socket,host=localhost,port=2002;urp;StarOffice.ServiceManager"
./python
```

Listing 4.7: Starting LibreOffice server and interactive Python shell.

```
import socket
import uno

localContext = uno.getComponentContext()
resolver = localContext.ServiceManager.createInstanceWithContext(
                'com.sun.star.bridge.UnoUrlResolver', localContext)

ctx = resolver.resolve(
    'uno:socket,host=localhost,port=2002;urp;StarOffice.ComponentContext')
smgr = ctx.ServiceManager

desktop = smgr.createInstanceWithContext('com.sun.star.frame.Desktop', ctx)

# access the current calc document
document = desktop.getCurrentComponent()
```

Listing 4.8: Retrieving document context in Python from shell.

```
document = XSCRIPTCONTEXT.getCurrentComponent()
```

Listing 4.9: Retrieving document context in Python from Add-in.

work with the documents. Finally with the `getCurrentComponent()` method the active document is retrieved.

With the installed Add-in, the LibreOffice script manager provides a global `XSCRIPTCONTEXT` variable that already holds a reference to the desktop context. Therefore only the last step is required as follows:

4.3 User Interface

The user interface for the add-in is aimed to be extremely simple, but functional. After selecting the data to work with, the user can click the add-in button in "Tools" section and will see a dialog window with different algorithms for building the similarity map - as in 4.1. Selecting the appropriate method and parameters for it, the user invokes the add-in to create a new sheet with the calculated coordinates for the similarity map plotting. A plot is also created automatically, showing the data samples scattered on a plane according to their similarities. The user is able then to export the plot using the built-in export functions of LibreOffice Calc with the desired extension.

4.4 Head Detection

The selected data might or might not contain the header column to indicate attributes names. Two conditions were therefore introduced:

- Whenever top left cell is empty, it is assumed that the name column does not contain header name. An example can be seen in Fig. 4.2.
- Data types between first two rows are matched. If in any of the rows fails to match, it is assumed that this is due to string data type because of the header name and numerical part differ.

If any of the conditions match, first row is assumed as a header row and removed before running any of the projection algorithms.

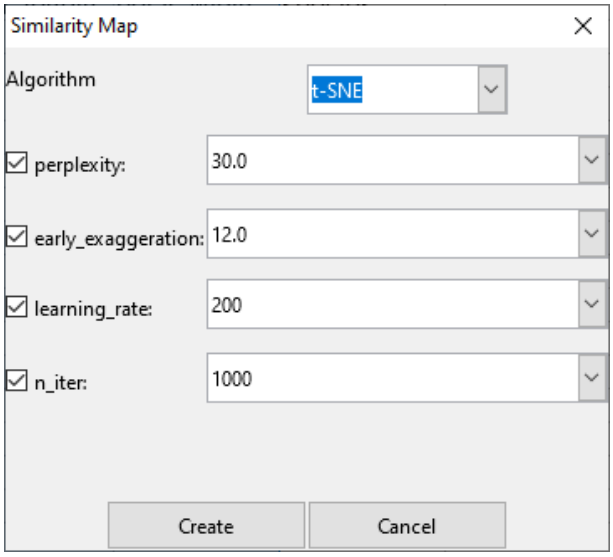


Figure 4.1: Dialog window to select a similarity map algorithm.

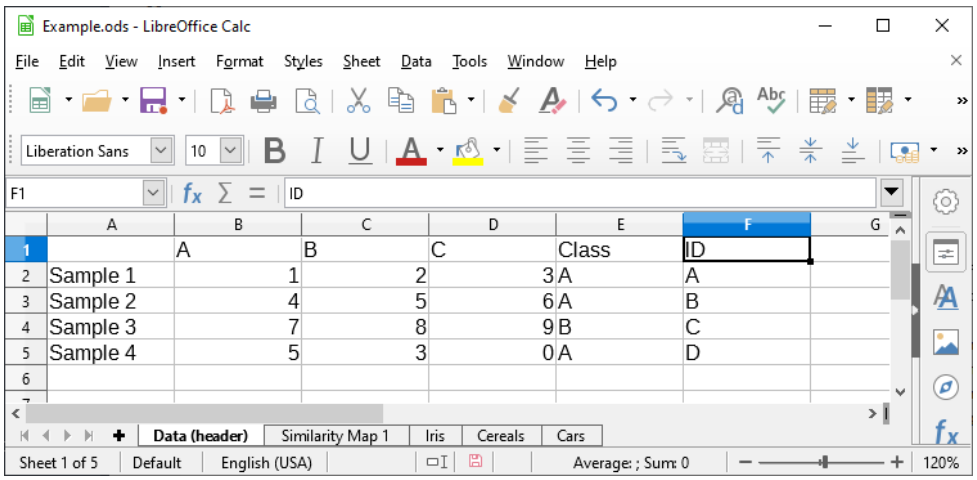


Figure 4.2: An example of a data table with empty top left cell.

4.5 Algorithms

All dimensionality reduction algorithms were implemented using the scikit-learn [Pedregosa et al. 2011] Python library, a widely used Machine Learning library that supports many supervised and unsupervised algorithms. It introduces a common interface for all of the implemented methods resulting in consistent usage. The example of PCA is as follows in 4.10.

The Similarity Map Add-in implements following methods from the scikit-learn library: PCA, MDS, Isomap and t-SNE; each previously described in detail in Chapter 3. In the following subsections, implementation details of the algorithms are described.

PCA

Scikit-learn implements three different algorithms for PCA decomposition, based on the size of the dataset for computational reasons. By default, for datasets smaller that 500×500, a Singular Value Decomposition (SVD) is performed and only the requested components are kept, the rest are truncated. If the dataset is larger, a randomised algorithm by Halko et al. [Halko et al. 2011] is used to speed up the computation. The algorithm was limited only to SVD solver, as it is not expected to run projections with

```
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_r = pca.fit_transform(X)
```

Listing 4.10: PCA using scikit-learn.

more than 500 attributes from within LibreOffice Calc and to ensure deterministic output of the method.

MDS

In the official documentation it is not stated what implementation is used and there are two main options available - metric and nonmetric. The Add-in was limited to metric option using the Euclidean distance, creating an MDS implementation known as absolute MDS [Torgerson 1952]. The algorithm uses random initial placement, making it non-deterministic.

Isomap

The Isomap algorithm serves as a preprocessing step for the MDS algorithm described in section 3.2.2. The parameters are left default as described in the documentation [*Scikit-learn Isomap* 2019] they are usually most suited default parameters. The most important, number of Neighbours used for graph generation, is set to 5. The shortest path calculation is performed either by Dijkstra's algorithm or by Floyd-Warshall algorithm, determined automatically based on the data. To guarantee consistent results, the Add-in manually sets Dijkstra's that also provides faster execution.

t-SNE

Scikit-learn implements the original version of t-SNE algorithm by Maaten et al. [LJPvd and Hinton 2008]. The original algorithm is expanded by using Barnes-Hut [Barnes and Hut 1986] gradient descent calculation as proposed in 2013 [Van Der Maaten 2013] by the original author of t-SNE algorithm. The improvement decreases computation time from $O(n^2)$ to $O(nlog(n))$. The parameters are left by default values as described in the official documentation [*Scikit-learn t-SNE* 2019].

Initialisation is performed as recommended, using PCA to reduce dimensionality of the data and speed up the computation. Another positive side effect of such initialisation is that it removes the random initialisation component, making the algorithm deterministic.

4.6 Creating plots

LibreOffice Calc has various built-in plots for visualising the data. The task was to select the most suitable one and invoke the plot creation using PyUNO library. Scatter plot with coordinates for each point representing a data sample serves the task well enough, with X- and Y-axis removed for clarity, as the coordinates of the points themselves are not carrying information of any importance to the user.

According to the object system of Calc, the plot is created within a shape, namely, a Rectangle in this case - there are various Shape classes implemented in the object library. The size of the shape can be set here also.

Afterwards the data range has to be set for the plot to draw the information from. As the dataset may vary, the data range has to be set dynamically, so the shape of the data frame plotted is passed as a parameter to the function. The Chart object then can be created using this range, for example, as shown in 4.12.

Where XYDiagram - is a scatter plot object. A chart is initially created as an empty object, and then the exact diagram type is set. Afterwards some adjustments are necessary: turn off the line connecting

```
from com.sun.star.awt import Rectangle
rect = Rectangle()
rect.Width, rect.Height, rect.X, rect.Y = 22000, 12000, 1000, 9200
```

Listing 4.11: Shape creation in PyUNO.

```
from com.sun.star.table import CellRangeAddress
range_address = []
range_address.append(CellRangeAddress())

range_address[0].Sheet = sheet.RangeAddress.Sheet
range_address[0].StartColumn = 1
range_address[0].EndColumn = 3
range_address[0].StartRow = 0
range_address[0].EndRow = shape[0] + 1

charts.addNewByName("SimilarityMap", rect, tuple(range_address), False, False)
chart = charts.getByName("SimilarityMap").getEmbeddedObject()
chart.createInstance("com.sun.star.chart.XYDiagram")
```

Listing 4.12: Data Range setting and chart creation.

the points by default, hide axes, legend and grids, add numbering to the samples for clarity - for example see 4.13. Some properties are belonging to the Chart object, the others are properties of the Diagram, what makes the whole construction a bit confusing for a programmer not familiar with the structure.

```
from com.sun.star.chart.ChartDataCaption import VALUE
from com.sun.star.chart.ChartDataCaption import TEXT

chart.HasMainTitle = True
chart.HasLegend = False
chart.Title.String = "Similarity Map"
chart.Title.CharHeight = 24
chart.HasSubTitle = False
chart.HasLegend = False

diagram = chart.getDiagram()
diagram.Lines = False
diagram.HasYAxis = False
diagram.HasXAxis = False
diagram.DataCaption = VALUE
diagram.DataCaption = TEXT
```

Listing 4.13: Adjusting the plot to the needs of the task.

The biggest issue with this part of the task is that the documentation for PyUNO is very scarce and there are almost no examples of some existing code like this. At this point code writing becomes mostly a guessing game with trying out all the different fields and properties trying to achieve the desired result.

Chapter 5

User guide

This Chapter provides the instructions on how to install and use developed Add-in for the Similarity Map visualisation technique.

5.1 Installation

LibreOffice Calc comes in a suite with LibreOffice and its own core Python version, that includes Universal Network Objects (UNO) [*UNO* 2019], a component model used for developing extensions, namely for communication between processes and/or languages that LibreOffice uses.

Due to the shipped Python core version, some of the required libraries are missing, including PIP [*Python pip* 2019], a package manager for installing third party Python libraries. The easiest way to obtain PIP is to download the *get-pip.py* script from the Python Packaging Authority [1]. To install it, first navigate to the LibreOffice installation folder and then into subfolder named *program*. The *get-pip.py* script should then be run exactly from this folder (*program*) by following Python command:

```
./python get-pip.py
```

Listing 5.1: PIP installation.

Finally, the remaining required packages can be installed by using the following command:

With this, all the prerequisites for running the Add-in should be installed. It can be tested by starting the LibreOffice's Python installation and running and importing one of the installed modules, see 5.3.

In the latest version as of writing (6.2.4), there appears to be a problem with the library multiprocessing that is pre-installed. Pool handling is missing and is required for Isomap nad t-SNE algorithms. It can be resolved by copying file pool.py from a full installation directory into the program/python-core-3.5.7/lib/multiprocessing. For convince, this file is provided in the documents folder that came with the report.

[1] https://bootstrap.pypa.io/get-pip.py

```
./python -m pip install setuptools pandas numpy sklearn scipy --user
>>> from sklearn.decomposition import PCA
```

Listing 5.2: A python command to install required Python dependencies.

```
./python
>>> from sklearn.manifold import TSNE, MDS
```

Listing 5.3: Test of the installed Python dependencies.

LibreOffice also provides a way to package dependencies with Add-in, but this option is only available for dependencies written in pure Python code. Those containing native code must be installed separately due to differences in operating systems and platforms.

After the dependencies are installed, the Add-in can be installed by running the OXT file. It is a registered extension so the LibreOffice will automatically open it, showing a license agreement. After accepting it, the Add-in will be installed and the restart is needed to make it available for use.

5.2 Step-by-step guide

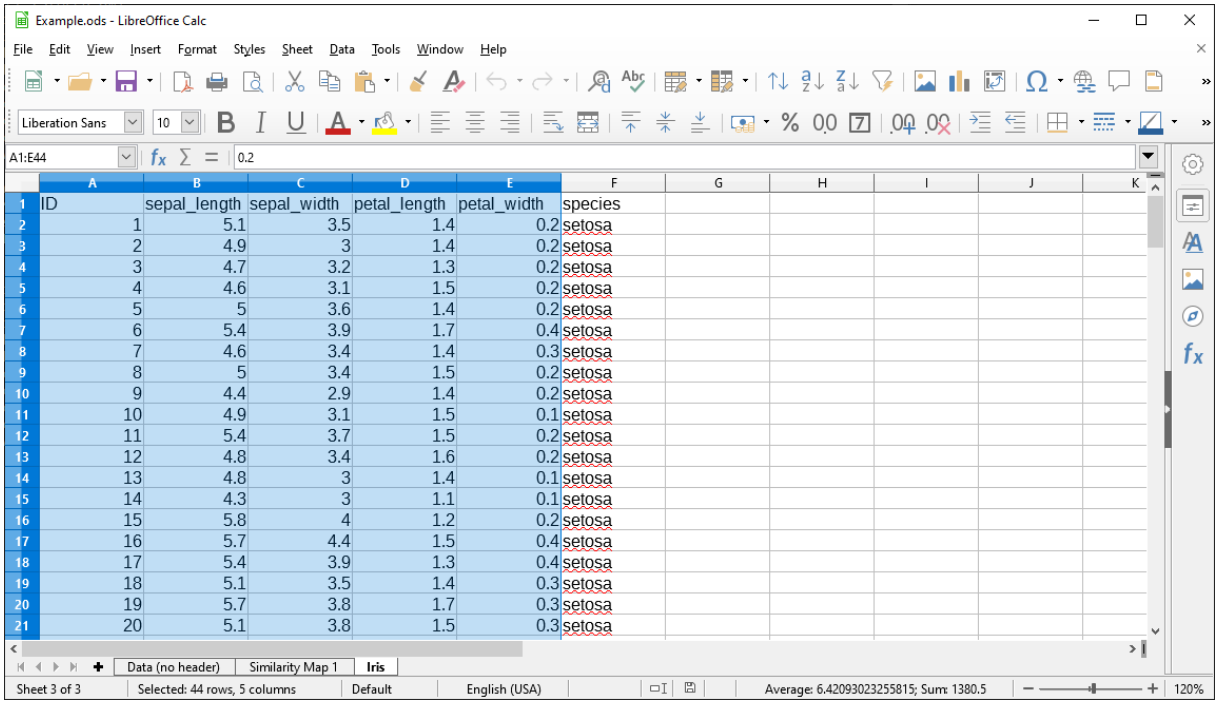
The action algorithm is the following:

1. Upload the dataset in the Calc spreadsheet and select all the cells with the data as shown in 5.1. The first selected column should represent record names, while the others attributes.

2. Select the "Similarity Map" tool in the "Tools" dropdown menu on top - see 5.2

3. Pressing the button will invoke a dialog window with all algorithm options and parameters for it - as shown in 5.3. The user can adjust them to their needs and press "Create" to calculate the similarities.

4. As the result the user will see a new sheet with the calculated data in it and the scatter plot representing the similarity map - see 5.4. If needed, it is also possible to export the chart created using ordinary built-in export functions of LibreOffice Calc by right clicking on the plot and selecting "Export Image".



	A	B	C	D	E	F
1	ID	sepal_length	sepal_width	petal_length	petal_width	species
2	1	5.1	3.5	1.4	0.2	setosa
3	2	4.9	3	1.4	0.2	setosa
4	3	4.7	3.2	1.3	0.2	setosa
5	4	4.6	3.1	1.5	0.2	setosa
6	5	5	3.6	1.4	0.2	setosa
7	6	5.4	3.9	1.7	0.4	setosa
8	7	4.6	3.4	1.4	0.3	setosa
9	8	5	3.4	1.5	0.2	setosa
10	9	4.4	2.9	1.4	0.2	setosa
11	10	4.9	3.1	1.5	0.1	setosa
12	11	5.4	3.7	1.5	0.2	setosa
13	12	4.8	3.4	1.6	0.2	setosa
14	13	4.8	3	1.4	0.1	setosa
15	14	4.3	3	1.1	0.1	setosa
16	15	5.8	4	1.2	0.2	setosa
17	16	5.7	4.4	1.5	0.4	setosa
18	17	5.4	3.9	1.3	0.4	setosa
19	18	5.1	3.5	1.4	0.3	setosa
20	19	5.7	3.8	1.7	0.3	setosa
21	20	5.1	3.8	1.5	0.3	setosa

Figure 5.1: Data selection to create a similarity map.

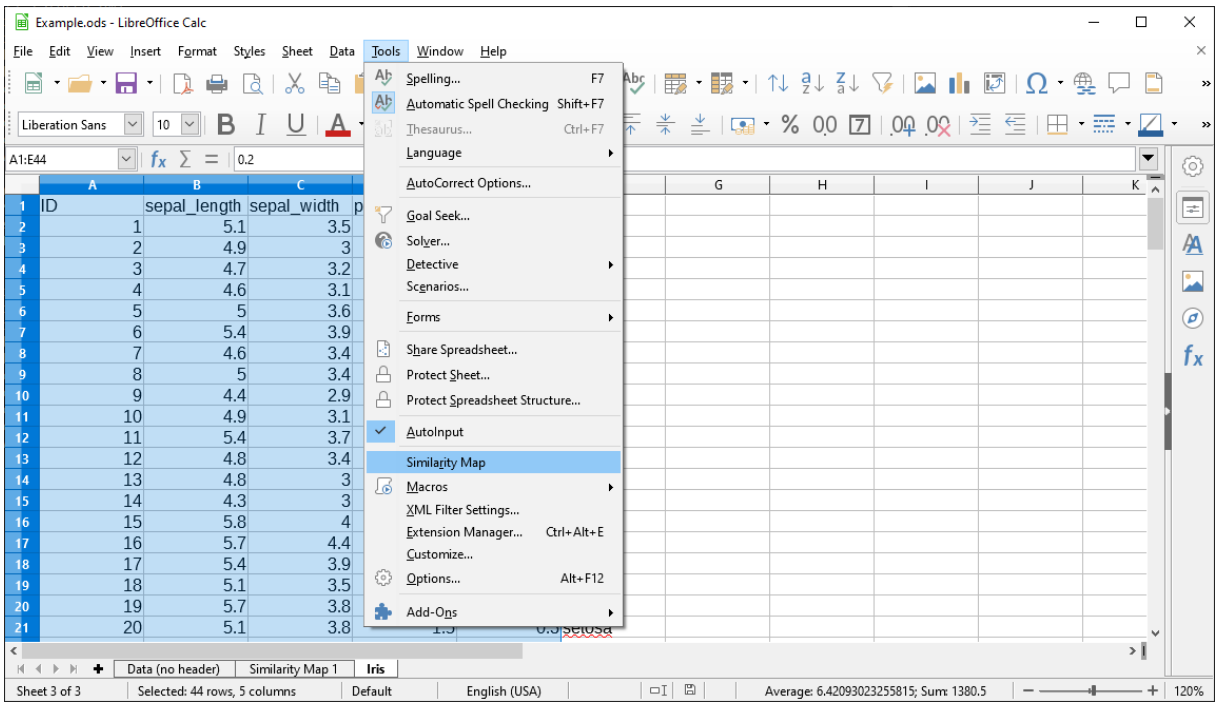


Figure 5.2: "Similarity Map" tool button.

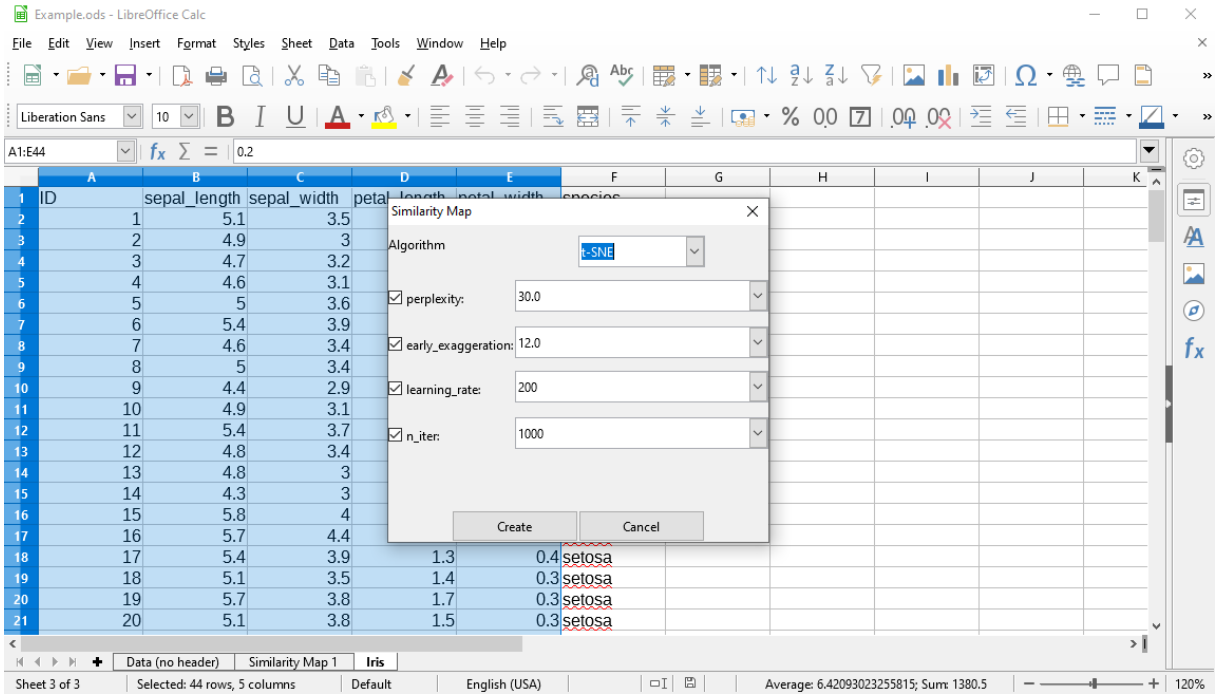


Figure 5.3: Dialog window with algorithm options.

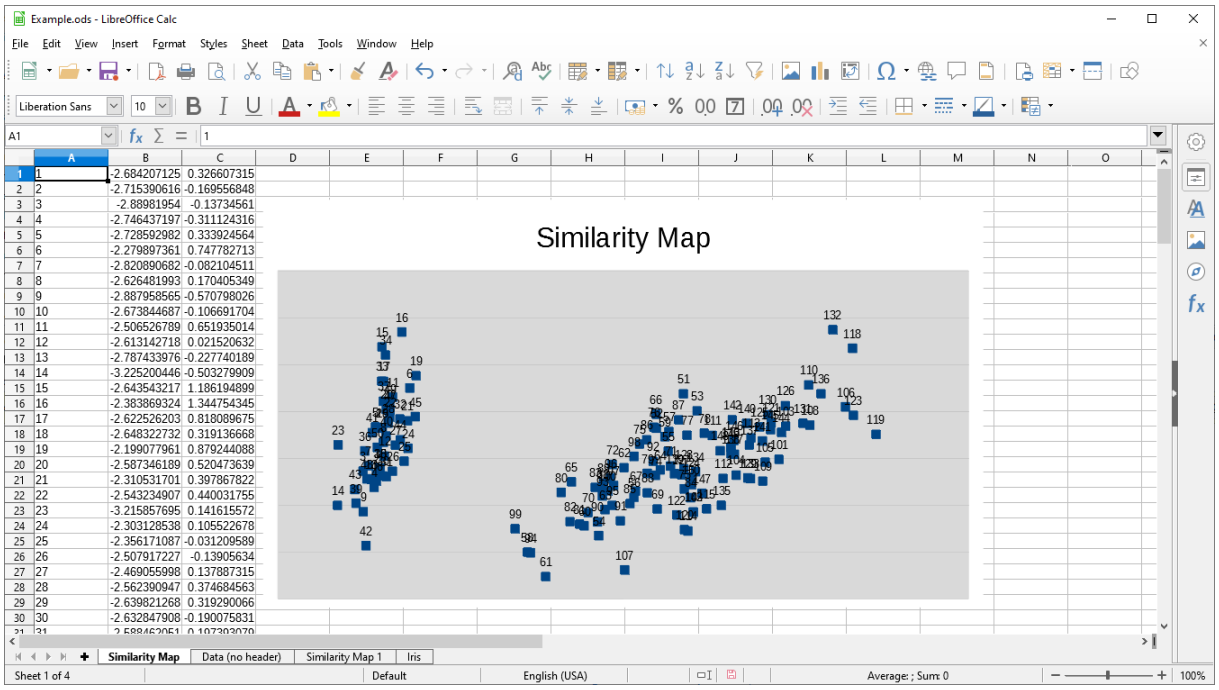


Figure 5.4: Similarity map plotting result.

Chapter 6

Conclusion

In this project work, a Similarity Map Add-in for LibreOffice Calc was developed. The LibreOffice Calc and several dimensionality reduction algorithms were presented, along with the detailed description of Add-in implementation and instructions on how to install and use it.

The Similarity Map Add-in was successfully developed, however, there are still many possibilities for improvement. The greatest weakness comes with the complexity of the installation as libraries must be manually installed. LibreOffice ships only with Python core version that lacks the required libraries used in this Add-in. Complex algorithms and the majority of the speed critical code are written in native code. This is compiled to a platform dependant binary, that can not be shipped together with a cross platform Add-in, introducing the difficult installation.

The user interface is also not user friendly and intuitive as desired due to a severe lack of documentation and examples. Checkboxes might be completely removed with parameters hidden based on the selected algorithms and with input controls more appropriate such as number boxes that contain input validation. Most of the existing functionality also allow to use predefined named tables, which SimilarityMap Add-In does not.

Overall, the developed cross-platform Add-in provides an easy way to create a Similarity Map, a visualisation that greatly helps explaining complex datasets or to obtain insights from the data such as detection of clusters and outliers without the need to code.

Bibliography

Add-in Functions [2019]. https://help.libreoffice.org/Calc/Add-in_Functions,_List_of_Analysis_Functions_Part_Two (cited on page 4).

Barnes, Josh and Piet Hut [1986]. *A hierarchical O (N log N) force-calculation algorithm*. nature 324.6096 (1986), page 446 (cited on page 15).

Buja, Andreas, Deborah F Swayne, Michael L Littman, Nathaniel Dean, Heike Hofmann, and Lisha Chen [2008]. *Data visualization with multidimensional scaling*. Journal of Computational and Graphical Statistics 17.2 (2008), pages 444–472 (cited on page 6).

Calc Filters [2019]. https://help.libreoffice.org/Common/About_Import_and_Export_Filters (cited on page 4).

Dimension Reduction - IsoMap [2019]. https://blog.paperspace.com/dimension-reduction-with-isomap/ (cited on pages 6–7).

Halko, Nathan, Per-Gunnar Martinsson, and Joel A Tropp [2011]. *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*. SIAM review 53.2 (2011), pages 217–288 (cited on page 14).

Holland, Steven M. [no date]. *Principal components analysis (PCA)*. Department of Geology, University of Georgia, Athens, GA (), pages 30602–2501 (cited on page 5).

LibreOffice [2019]. https://www.libreoffice.org/discover/libreoffice/ (cited on page 3).

LibreOffice Calc [2019]. https://www.libreoffice.org/discover/calc/ (cited on page 3).

LibreOffice Extensions [2019]. https://extensions.libreoffice.org (cited on page 9).

LibreOffice Help [2019]. https://libreofficehelp.com/change-libreoffice-default-look-and-feel/ (cited on page 3).

LJPvd, Maaten and GE Hinton [2008]. *Visualizing high-dimensional data using t-SNE*. Journal of Machine Learning Research 9 (2008), pages 2579–605 (cited on page 15).

Maaten, Laurens van der and Geoffrey Hinton [2008]. *Visualizing data using t-SNE*. Journal of machine learning research 9.Nov (2008), pages 2579–2605 (cited on page 7).

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay [2011]. *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research 12 (2011), pages 2825–2830 (cited on page 14).

Python pip [2019]. https://pypi.org/project/pip/ (cited on page 17).

Scikit-learn Isomap [2019]. https://scikit-learn.org/stable/modules/generated/sklearn.manifold.Isomap.html (cited on page 15).

Scikit-learn t-SNE [2019]. https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html (cited on page 15).

Tenenbaum, Joshua B, Vin De Silva, and John C Langford [2000]. *A global geometric framework for nonlinear dimensionality reduction*. science 290.5500 (2000), pages 2319–2323 (cited on page 7).

Torgerson, Warren S [1952]. *Multidimensional scaling: I. Theory and method*. Psychometrika 17.4 (1952), pages 401–419 (cited on page 15).

UNO [2019]. http://www.openoffice.org/udk/ (cited on page 17).

Van Der Maaten, Laurens [2013]. *Barnes-hut-sne*. arXiv preprint arXiv:1301.3342 (2013) (cited on page 15).