Eberhard Karls Universität Tübingen
Mathematisch-Naturwissenschaftliche Fakultät
Wilhelm-Schickard-Institut für Informatik

Master Thesis Machine Learning

Wasserstein t-SNE and applications to social science

Fynn Bachmann

30th September 2021

Reviewers

Philipp Hennig
philipp.hennig@uni-tuebingen.de
Methods of Machine Learning
Max Planck Institute for Intelligent Systems
University of Tübingen

Philipp Berens
philipp.berens@uni-tuebingen.de
Institute for Ophthalmic Research
W. R. Centre for Integrative Neuroscience
University of Tübingen

Bachmann, Fynn:
Wasserstein t-SNE and applications to social science
Master Thesis Machine Learning
Eberhard Karls Universität Tübingen
1st April – 30th September 2021

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbständig und nur mit den angegebenen Hilfsmitteln angefertigt habe und dass alle Stellen, die dem Wortlaut oder dem Sinne nach anderen Werken entnommen sind, durch Angaben von Quellen als Entlehnung kenntlich gemacht worden sind. Diese Masterarbeit wurde in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt. Der vollständige Code und alle Daten, die benutzt wurden, um die hierin dargestellten Abbildungen zu gestalten, sind auf GitHub verfügbar[1].

Tübingen, den 30.09.2021

_____

Fynn Bachmann

[1] `https://www.github.com/fsvbach/MScThesis`

Zusammenfassung

Umfragen und andere sozialwissenschaftliche Datensätze liegen häufig in hierarchischer Form vor, d.h. die einzelnen Teilnehmer (Stichproben) können auf einer höheren Ebene, z.B. ihrer geografischen Region, zusammengefasst werden. Ein Vergleich dieser höheren Ebenen (Einheiten) erfolgt in der Regel über den Abstand ihrer Mittelwerte. Jede Einheit kann jedoch als eine Wahrscheinlichkeitsverteilung über ihre Stichproben betrachtet werden. In dieser Arbeit entwickeln wir einen Ansatz zur Einbettung hierarchischer Datensätze in niedrige Dimensionen unter Verwendung der Wasserstein Distanz, die nicht nur die Mittelwerte, sondern auch die Formen der Verteilungen innerhalb der Einheiten berücksichtigt. Der Algorithmus *t-distributed Stochastic Neighbor Embedding* (t-SNE) wird verwendet, um 2D-Einbettungen der Matrix der paarweisen Wasserstein Distanzen zu konstruieren. Wir generieren synthetische Daten, bei denen ein Teil der Information in der Kovarianz der Stichproben kodiert ist, um die Effektivität von *Wasserstein t-SNE* zu demonstrieren. Anschließend wenden wir diese Methode auf mehrere sozialwissenschaftliche Datensätze an, insbesondere auf die Bundestagswahl und die European Values Study. Durch die Visualisierung ihrer Struktur mit Wasserstein t-SNE zeigen wir spezifische Beispiele von bedeutungsvollen Strukturen, die durch den Algorithmus aufgedeckt und hervorgehoben werden. Wir kommen zu dem Schluss, dass unsere Methode 2D-Einbettungen verbessern und eine feinere Struktur in den Daten aufdecken kann, die sonst verborgen wäre. Wir glauben, dass sie auch in anderen Anwendungsbereichen nützlich sein kann, z.B. bei der Visualisierung biologischer oder biomedizinischer Datensätze.

Abstract

Surveys and other social science datasets often come in hierarchical form, i.e. individual participants (samples) can be grouped at a higher level such as their geographical region. A comparison of these higher levels (units) is usually made by the distance between their means. However, each unit can be viewed as a probability distribution over its samples. In this work, we develop an approach for embedding hierarchical datasets in low dimensions using the Wasserstein distance metric that takes into account not only the means but also the shapes of within-unit distributions. The t-distributed Stochastic Neighbor Embedding (t-SNE) algorithm is used to construct 2D embeddings of the matrix of pairwise Wasserstein distances. We generate synthetic data, in which part of the information is encoded in the covariance of the samples, to demonstrate the effectiveness of our *Wasserstein t-SNE*. We then apply this method to several social science datasets, in particular the German Federal Election and the European Values Study. Visualizing their structure with Wasserstein t-SNE, we show specific examples of meaningful structure uncovered and highlighted by the algorithm. We conclude that our method can improve 2D embeddings and can reveal a finer structure in the data that would be otherwise hidden. We believe it can be useful in other application domains, e.g. for visualizing biological or biomedical datasets.

Acknowledgements

First of all I would like to thank Philipp Hennig for bringing up the idea for this method. I enjoyed spending my thesis on Machine Learning research with applications to social sciences. Secondly I have to thank Dmitry Kobak for the continuous exchange and all the valuable ideas for figures and experiments which pushed the project forwards. You have really been a great assistance. There are many other people who supported this work on the way, e.g. the colleagues in the office who always understand everything quickly. Also I want to thank everyone who proof-read the final thesis. Special thanks goes out to the developers of openTSNE who were flexible enough to add features upon request, such as the possibility to run t-SNE with a distance matrix as input. It is a very nice package, thank you all.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this thesis we will study the dimension reduction algorithm **t-distributed Stochastic Neighbor Embedding** [1] in a particular setting. We will develop a method to embed datapoints which are probability distributions. This requires a notion of similarity for such objects. Throughout the thesis we will compare the **Wasserstein metric** [2] with the Euclidean distance of the means. The objective is to find out whether the resulting embeddings show a different structure. This chapter gives a brief introduction to data visualization in general. Furthermore, it explains the data structure that is required for later experiments. Chapter 2 provides an overview about the mathematical concepts. We will explain different dimension reduction algorithms and define the Wasserstein metric for our setup. Subsequently, in Chapter 3, our method is applied to synthetic data as well as real-world data such as the European Values Study [3] and the German Federal election [4]. We will eventually summarize that the method is able to improve clustering but that the perfect use-case it yet to be found.

1.1 Visualizing Structure in Datasets

In the modern world data is collected on a daily basis. When collecting information about patients, companies, countries (i.e. datapoints) policy makers have to deal with complex and high dimensional feature spaces. Most of the datasets however contain low dimensional structure, which is really of interest. Before searching for information we can embed data in a visualizable space, often the 2D-plane. This procedure not only helps to get an overview of the dataset but is also able to find clusters of datapoints, which share certain features. The main goal of data visualization is therefore to simplify datasets without loosing too much structure, a task which is not optimally solvable but subject to different trade-offs and design decisions.

1.1.1 Motivational example

To illustrate the effects of data visualization we have a look at the election of Baden-Würtemberg in March 2021 [6]. This dataset is a matrix $\boldsymbol{X} \in [0,1]^{72\times 7}$

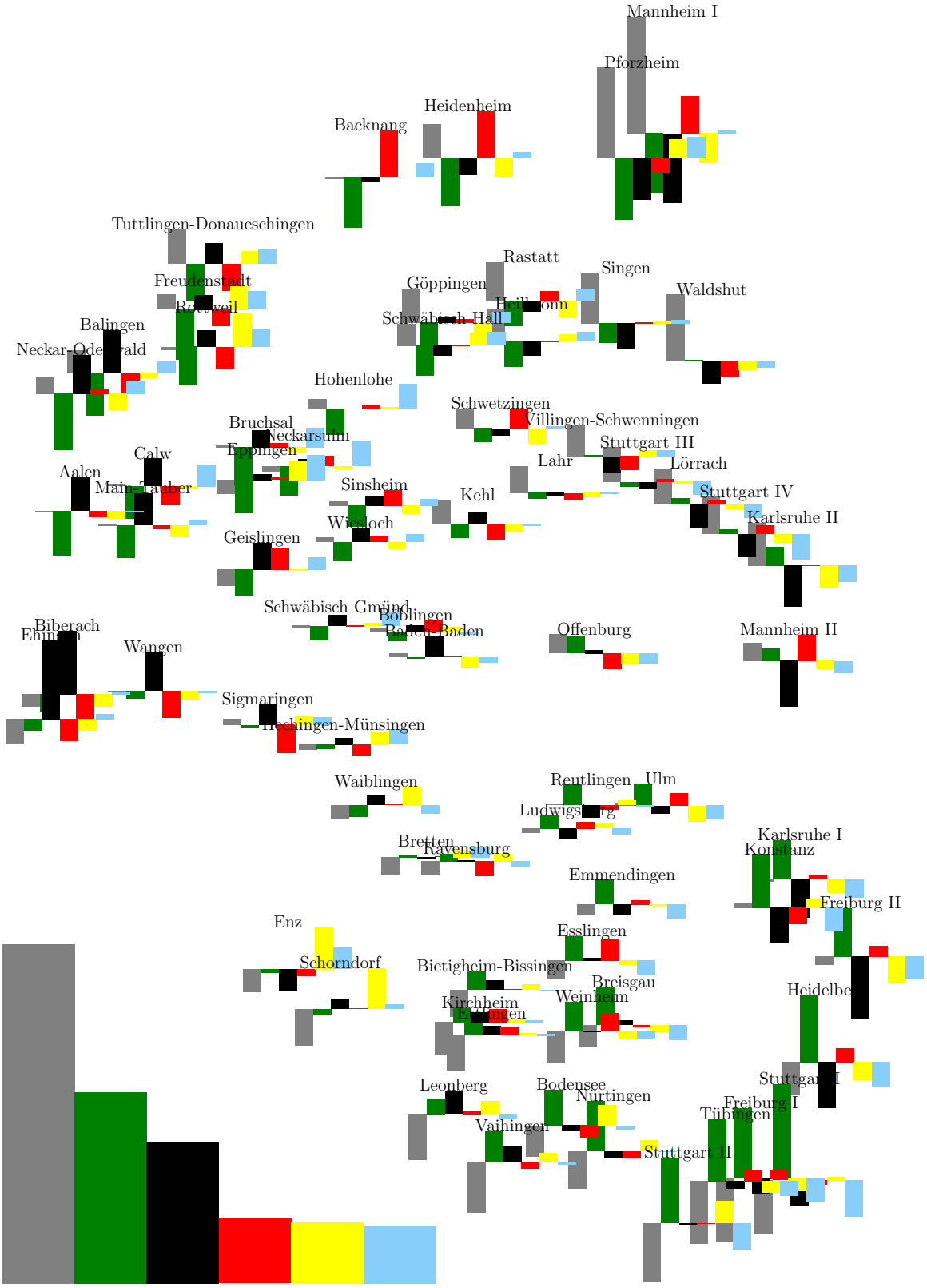

Figure 1.1: *Political Landscape of Baden-Württemberg (visualization inspired by [5]).* In the lower left corner the mean result is given. Each voting district is annotated with its deviation from the mean. The six colors represent the six dominant parties, gray shows the share of nonvoters. The t-SNE algorithm was used to embed similar voting districts nearby.

where each row X_n represents a voting district. For simplicity we consider here only the results of the six major parties: *DIE GRÜNEN*, *CDU*, *SPD*, *FDP*, *AfD*, *DIE LINKE* as well as non-voters. The percentages for each party are positive values which sum to one. To show the structure of this dataset we would need a seven-dimensional coordinate system which is impossible to visualize. A dimension reduction algorithm however is able to embed these $N = 72$ datapoints into the 2D-plane which we can see in Figure 1.1. Nearby points in the seven-dimensional space are also close in the embedding. In the lower right corner we find a cluster of university cities (e.g. *Heidelberg*, *Freiburg*) that dominantly vote for the green party. Towards the upper left corner the share of green votes decreases as we encounter more rural areas like *Balingen* and *Aalen*. Industry driven communities like *Heidenheim* or *Mannheim* are found in the upper right corner and have a larger share of non-voters. Without exploiting any prior knowledge about the voting districts this meaningful structure arises directly from the data itself. Dimension reduction can help to get an overview before doing an analysis of the dataset. However, we must keep in mind that the low dimensional representation comes with a loss of information and will almost never keep the full structure.

1.2 Hierarchical Data

In the first example we encountered a dataset where each datapoint was represented by a single vector. For the rest of this thesis however we will analyze *hierarchical data*, i.e. datasets where each datum X_n is a probability distribution which we will call *unit* from now on. Of each unit we have an arbitrary amount of samples. Throughout the thesis we will investigate whether significant information is lost by collapsing the samples into their unit mean. More precisely, we will find that structure can be present in the covariance of units and we will develop a technique to include this information into the dimension reduction algorithm.

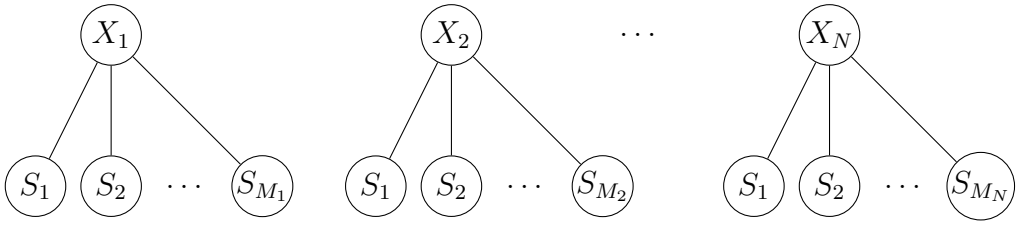

Figure 1.2: *Hierarchical Data.* Each unit X_n is a probability distribution from which M_n samples are drawn.

1.2.1 Unit means

As an example of the scheme in Figure 1.2 we can think of participants in a survey from regions around the world. If we want to visualize similarities and difference of countries, we consider them as units X_n. The participants of the respective nationality are then samples from the unit. The most natural choice to embed this hierarchical dataset is to compute the mean sample of each distribution and run the visualization algorithm on the flattened matrix. However, as stated above, significant information is lost by this procedure. Figure 1.3 illustrates answers to the European Value Study in *Germany* and *Albania*. While the mean of both countries is fairly close to each other, it reasonable to assume that the political situation in both countries is rather different. This might be encoded in the variance of the answers which can be seen in the histogram. In this thesis we will therefore use the Wasserstein metric to compare distributions. It is motivated by the question: *How much mass has to be transferred (and how far) to obtain one distribution from the other?* In the figure the distance of the countries becomes much larger when applying the Wasserstein metric. We will describe its details in Section 2.2. For now it should be clear that this approach is much more accurate than using only the distance of the means as we can also take into account bimodal structure. Later in this thesis we will see that the additional information can change cluster memberships of units in the embedding.

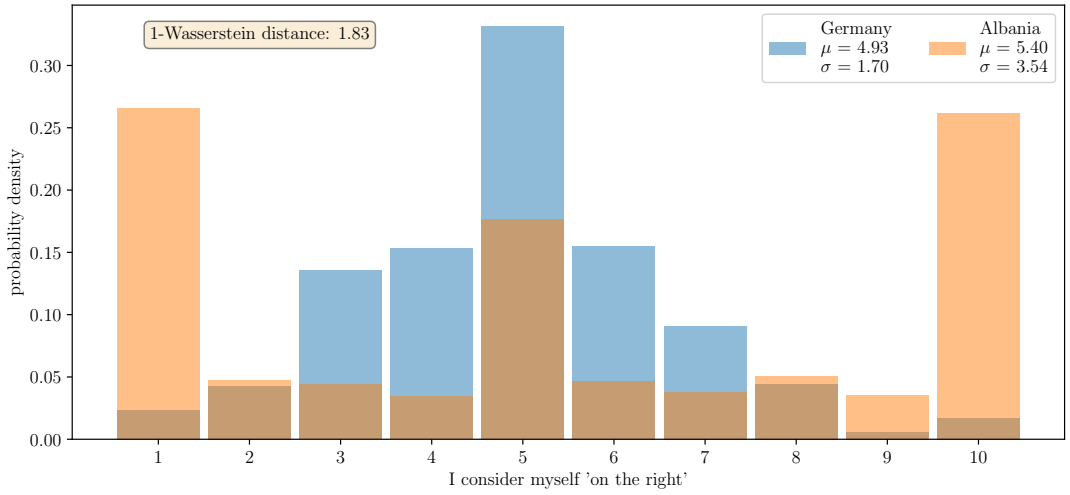

Figure 1.3: *Variance in the European Values Study.* Histogram of answers to the Question: Where do you consider yourself politically from left (1) to right (10)?

1.2.2 Labels

In Figure 1.4 we consider another hierarchical dataset in which the units are countries, i.e. the Big-Five Personality Survey [7]. On the left we can see the t-SNE embedding where we used flags to represent countries. However no clear structure is visible. The units are embedded in a donut shape with a small cluster on the top right (*Sudan*, *Morocco*, *Tunisia*, *Algeria* and *Iraq*). The cultural similarity of these countries leads to the assumption that there might be however some structure in the embedding. It can be helpful to a priori define labels of each unit – information that is not included when embedding the data, but helps afterwards when visualizing the structure.

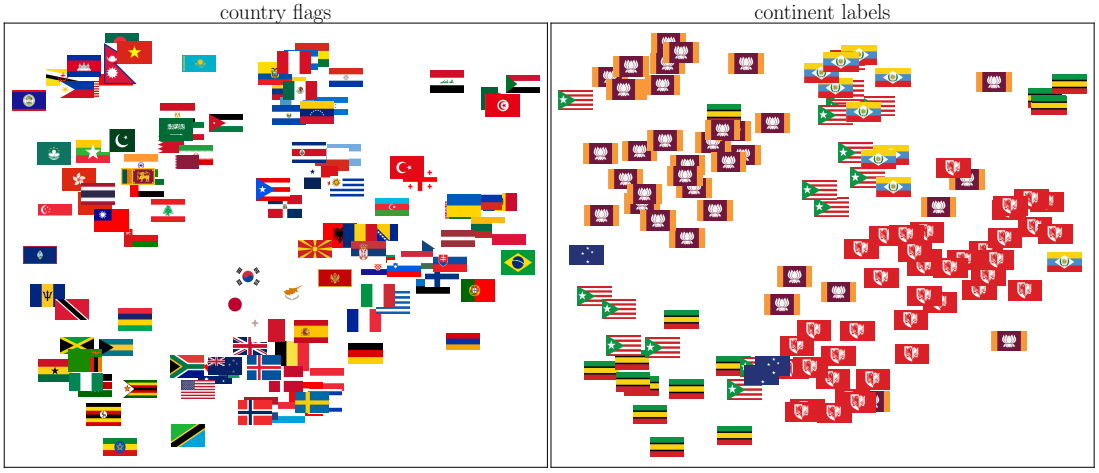

Figure 1.4: *Big-Five Personality Traits (flags from [8, 9]).* Left: t-SNE embedding of the mean personality per country using the flag of the country. Right: Each unit is visualized by its label (continent).

One way to define labels for this particular dataset is to use the continents of the respective countries. Each unit is now represented by the hypothetical flag of its continent. In Figure 1.4 on the right we then see a lot of structure within the embedding: Asian countries are clustered on the top left; African and Northern American countries are bottom left, while the European cluster is on the bottom right. For the analysis why this structure emerges from the dataset and how it could be improved by including the covariance we refer to the legend in Appendix B.1. For now this example should only serve as a quick motivation why using labels of units is necessary when arguing for or against structure in an embedding.

Chapter 2

Theory: Methods and Distances

In this thesis we develop a method to embed a high dimensional dataset $\boldsymbol{X}$, that contains hierarchical structure, i.e. where each datapoint $\boldsymbol{X}_n$ is a probability distribution from which we have an arbitrary amount of samples. These samples can be used to compute the mean, but also other moments such as covariance etc. In this chapter we give a brief description of existing dimension reduction algorithms. In Section 2.2 we discuss the Wasserstein metric which is a natural distance for probability distributions. We compare it to other distance measures, show its closed form solution for Gaussians and explore a technique to compute it exactly even in high dimensions. In Section 2.3 we introduce the Wishart distribution which we use in Chapter 3.1 to sample covariance matrices.

2.1 Dimension Reduction

A multitude of algorithms exist that embed or project high dimensional data into a low dimensional space. We will describe three particular techniques instead of giving an overview which can be found elsewhere [10, 11]. Our aim is not to compare existing algorithms, rather we explain why the chosen **t-SNE** algorithm works as well as any other to show the main result of this thesis: that structure can be found in the covariance of hierarchical data. All visualizations in this work could as well have been done using **UMAP** or similar as long as we stay consistent, since the goal is not to compare dimension reduction algorithms but the underlying metrics.

2.1.1 PCA

Principal Component Analysis (PCA) is a dimension reduction algorithm which projects a dataset to the subspace of largest variance. The basis vectors of this subspace are the eigenvectors of the data matrix which correspond to the largest singular values. As a design choice we have to fix a hyperparameter K which sets the dimension of the subspace. The higher we choose K the more accurate the projection will become but this is not the goal of dimension reduction. Usually K is set in such a way that the complete subspace can be visualized.

As an example let us consider a dataset X that contains N datapoints $X_n \in \mathbb{R}^D$. We now want to find a low dimensional projection $Y \in \mathbb{R}^{N \times K}$ that has largest variance while $K \ll D$. One way to obtain this is a Gaussian fit of the dataset. The data is then projected to the marginal distributions with largest variance.

$$\bar{x} := \frac{1}{N} \sum_{n}^{N} X_n$$
$$S := \frac{1}{N} \sum_{n}^{N} (X_n - \bar{x})(X_n - \bar{x})^T$$

A simpler way is to directly compute the Singular Value Decomposition of X and obtain the largest eigenvalues and their eigenvectors from $X = U\Lambda V^T$. Both methods yield the same unique projection axis as in Figure 2.1. There we can see datapoints being projected to the 1D-Line which corresponds to the largest eigenvector of the covariance matrix S.

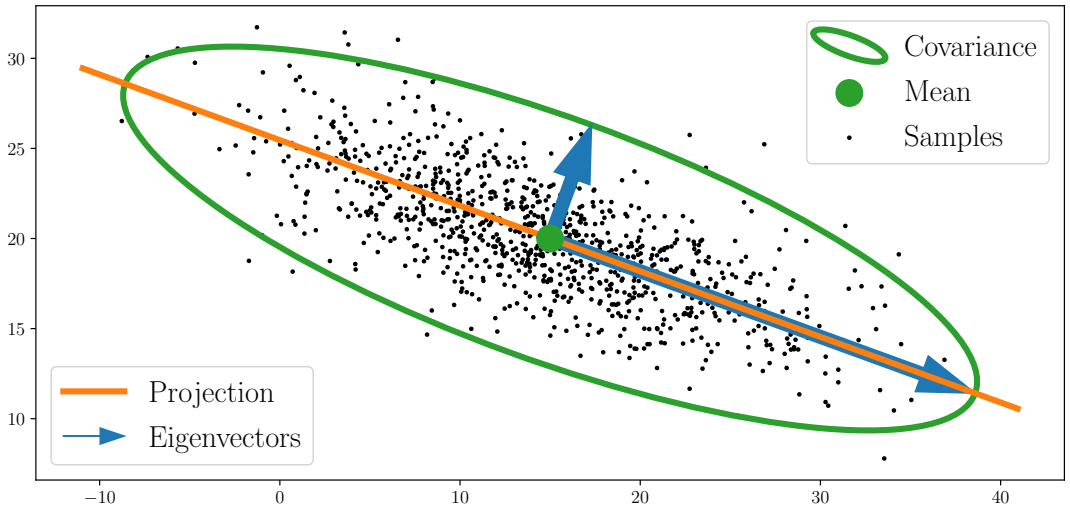

Figure 2.1: *Principal Component Analysis.* A multivariate Gaussian in projected to the axis of largest variance (orange).

The benefits of PCA are its simplicity and transparency. We can visualize the eigenvectors and interpret their components and directions. Also we obtain a generative model and can create new datapoints in the low dimensional space that will be close to their neighbors even in higher dimensions. The main drawback of PCA is its linearity. For datasets in complicated manifolds the method will not capture the structure anymore.

2.1.2 MDS

A non-linear alternative in dimension reduction is known as **Multidimensional Scaling** (MDS). This concept describes a set of algorithms for which it is not necessary to know the original data, but only a matrix $K \in S^n(\mathbb{R})$ that contains the pairwise distances (or similarities). The datapoints are then embedded in a fictional space which is usually two dimensional. The axes of that space don't have a particular interpretation, however the coordinates of the embedded data represent the distances of the input matrix. Under certain circumstances they are proven to be exact, however in most cases only an approximation is possible [12].

2.1.3 t-SNE

An extension of the previous algorithm is called **t-distributed Stochastic Neighbor Embedding** [1]. Here we also don't project but embed the data into a subspace without particular interpretation of the axes. While it was originally proposed to embed vectors, it works as well with a distance matrix as input. In this section we describe the latter approach since the units we will embed later are probability distributions which can't be represented in Euclidean space.

Definition 2.1.1 *Let P, Q be matrices of the same dimension. The Kullback-Leibler divergence is then given by*

$$KL(P\|Q) := \sum_{ij} P_{ij} \log \frac{P_{ij}}{Q_{ij}}$$

Definition 2.1.2 *Let $p(x)$ be a discrete probability density function. The perplexity of p is then given by*

$$PP(p) := 2^{H(p)} = \prod_x p(x)^{-p(x)}$$

Having defined perplexity and the Kullback-Leibler divergence, we can unbox the t-SNE algorithm. The first step is done by computing the affinity matrix P from the distances.

Definition 2.1.3 *Let $K \in S^n(\mathbb{R})$ be distance matrix of n arbitrary datapoints. The rows of the affinity matrix P will then be constructed by Gaussian kernels with bandwidth σ_i*

$$p_{j|i} = \frac{\exp(-K_{ij}^2/2\sigma_i^2)}{\sum_{k\neq i} \exp(-K_{ik}^2/2\sigma_i^2)}$$

such that all perplexities of the conditional distributions equal a predefined perplexity σ. At last the affinity matrix is symmetrized by

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

Then the embedding is initialized randomly (or with an advanced algorithm such as PCA [13]). The next step is to compute the pairwise distances of the datapoints in the embedding.

Definition 2.1.4 *Let P be an affinity matrix. The t-SNE algorithm will (locally) minimize the Kullback-Leibler divergence $KL(P\|Q)$ with respect to the low dimensional embedding $\boldsymbol{Y}$, where Q is an affinity matrix based on the t-distribution and the coordinates of the embedded vectors $\mathbf{y}_i$*

$$q_{ij} = \frac{\left(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2\right)^{-1}}$$

The Kullback-Leibler divergence of the low and high dimensional affinity matrices is then optimized with respect to the coordinates of the embedding. The points are thus moved along the gradient until convergence. This procedure results in a local minimum where no point can be slightly moved without yielding a worse embedding than before.
When interpreting t-SNE embeddings it is important to keep in mind that the choice of the t-distribution puts emphasis on close points. The opposite does not hold: points that are embedded far from each other don't necessarily need to be far away from each other in the high dimensional space.

Design choices

The t-SNE algorithm comes with certain hyperparameters and is dependent on an initialization. Its objective is non-convex, thus it has multiple local minima in which the algorithm can terminate. The t-SNE embedding is therefore not unique, i.e. it can have certain artifacts that occur with some initialization and not with others.
Another parameter that needs to be set is the perplexity σ. It defines the scale on which points are considered to be close. If the perplexity is very low, the structure will be very fine with lots of clusters since only few points are considered to be close. If the perplexity is high, all points might end up in the same cluster. Throughout this thesis we will leave the perplexity at the default value of $\sigma = 30$ and use the implementation of openTSNE [14].

2.2 Wasserstein Metric

Any embedding algorithm requires a distance measure in the high dimensional space. When dealing with hierarchical data this becomes non trivial as there is no default distance measure for probability distributions. A simple work-around is to collapse the distributions to their means and then use the Euclidean distance. But one can easily imagine that this technique can loose arbitrarily much of the information. A distance which is able to capture the structure of higher moments as well is called the **Wasserstein metric** and has been widely used in Machine Learning [15]. One particular benefit is the sensitivity to the metric space on which the distributions are defined. Another advantage is that it can compare probability densities which don't have the same support, as long as a distance measure of their support is given. The downside of the Wasserstein distance is its computation complexity. We will later see that it is related to optimal transport. However, for Gaussians a closed-form solution exists. In this section we will provide the formula and explain why its computation in the discrete case is essentially a Linear Program.

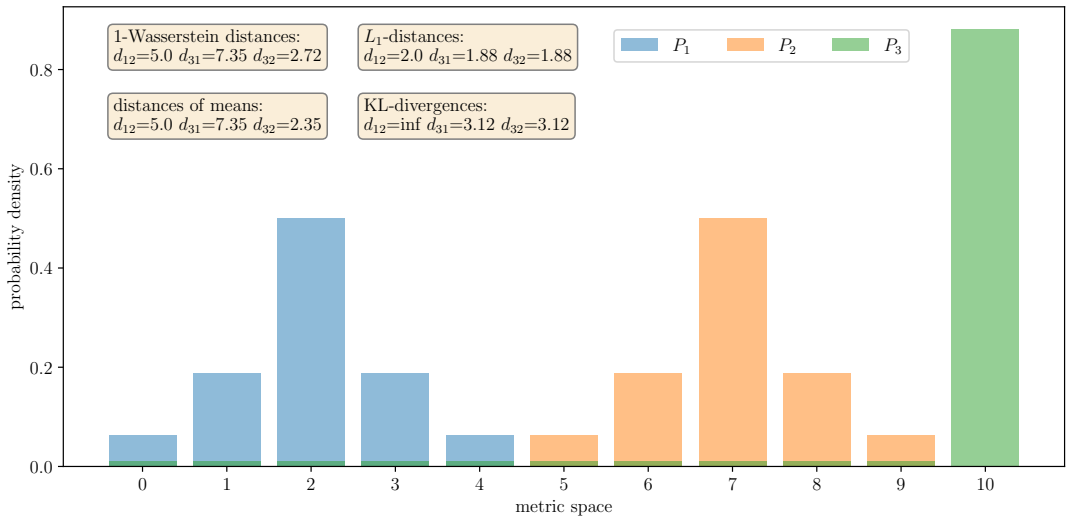

Figure 2.2: *Earth Mover's Distance.* The pairwise 1-Wasserstein distances of three probability distributions are compared to the L_1-distance and the Kullback-Leibler divergence.

2.2.1 Motivation

In Computer Science the 1-Wasserstein metric is also known as the **Earth Mover's Distance**. This name is motivated by the following thought experiment: we imagine the probability distributions as piles of earth – their distance now repre-

sents the amount of work that has to be done in order to transfer the probability mass from one distribution to the other. This definition however requires a metric space M on which the probability distributions are defined. This emerges from the fact that we have to measure how far two points are away from each other, i.e. how far the mass has to be transported.
In Figure 2.2 we can see the comparison of four different metrics. For each metric the pairwise distance of three discrete distributions is computed. While the two component-wise metrics (KL-divergence and L_1-distance) can't distinguish d_{32} from d_{31}, the Wasserstein distance mirrors our intuition that P_2 is much closer to P_3 than P_1. Another takeaway from this example is the similarity of the Wasserstein distance to the distance of the means. This effect will be important throughout the thesis.

Definition 2.2.1 *Let (M,d) be a metric space. The p-Wasserstein distance of two distributions μ and ν is then defined as*

$$W_p(\mu,\nu) := \left(\inf_{\gamma\in\Gamma(\mu,\nu)} \int_{M\times M} d(x,y)^p \mathrm{d}\gamma(x,y)\right)^{\frac{1}{p}}$$

where Γ is the set of all coupling of μ and ν.

2.2.2 Special Case: Gaussians

While the p-Wasserstein distance for continuous distributions is generally hard to compute, there exists a closed-form solution of the 2-Wasserstein metric for multivariate normals, also known as Fréchet distance [16].

Definition 2.2.2 *Let $\mathcal{N}_1, \mathcal{N}_2$ be two Gaussian distributions with $\mathcal{N}_i = (m_i, C_i)$. The 2-Wasserstein distance of these distributions is then given by:*

$$W(\mathcal{N}_1,\mathcal{N}_2)^2 := \|m_1 - m_2\|_2^2 + \mathrm{tr}\left(C_1 + C_2 - 2\left(C_2^{1/2} C_1 C_2^{1/2}\right)^{1/2}\right)$$

Note, that the left part of the sum is just the Euclidean distance of the means. The Wasserstein metric can therefore be seen as an extension of the Euclidean distance. It was also shown [16] that the right part of the sum defines a proper metric on the space of covariances. We can also write it differently as

$$\begin{aligned} d(C_1,C_2)^2 &:= \mathrm{tr}\left(C_1 + C_2 - 2\left(C_2^{1/2} C_1 C_2^{1/2}\right)^{1/2}\right) \\ &= \mathrm{tr}\left(C_1 + C_2 - 2\left(C_2 C_1\right)^{1/2}\right). \end{aligned}$$

Convex Interpolation Method

By introducing a hyperparameter $\lambda \in [0, 1]$ we can put emphasis either on means or covariances. The convex generalization of the 2-Wasserstein distance for Gaussians can therefore be written as

$$W(\mathcal{N}_1, \mathcal{N}_2)^2 := (1-\lambda) \cdot \|m_1 - m_2\|_2^2 + \lambda \cdot \operatorname{tr}\left(C_1 + C_2 - 2\left(C_2^{1/2} C_1 C_2^{1/2}\right)^{1/2}\right)$$

which yields both the Euclidean and the Covariance metric for $\lambda = 0$ and $\lambda = 1$ respectively. As the t-SNE algorithm is indifferent to scaling the distances of datapoints linearly, the additional factors won't change the embeddings. We will use this notation in the analysis to reconcile all three cases into one method. Also it can be interesting to see the evolution of the embeddings for growing λ as in the Appendix A.8.

Computation Complexity

The computation of the closed-form solution for the 2-Wasserstein distance of Gaussians is very fast. However, we have to compute a matrix multiplication as well as an eigenvalue decomposition to take the square root of the product which can be both considered to be of $\mathcal{O}(d^3)$ where d is the number of dimensions. This has to be done for each distance computation, so if we have N Gaussians, their pairwise distance matrix requires $\mathcal{O}(N^2)$ eigenvalue decompositions.
Contrary to what had originally been written on the German Wikipedia page[1] about the Wasserstein metric for Gaussians, the following equation does **only** hold, iff the covariances commute

$$\operatorname{tr}\left((C_2^{1/2} C_1 C_2^{1/2})^{1/2}\right) \neq \left\| C_1^{1/2} - C_2^{1/2} \right\|_F^2 .$$

However, this assumption would simplify the calculation even more: we could vectorize the computation in a package like `numpy` to reduce the complexity to $\mathcal{O}(N)$ eigenvalue decompositions. We did not use this approximation but a quick comparison indicated that the differences are not very large.

2.2.3 Linear Programming

So far we have only looked at the Wasserstein distance of Gaussians. In this section we will compute the exact Wasserstein distance, which we have earlier claimed to be hard for continuous distributions. However, most social science datasets contain discrete samples. That makes the computation easier. In fact,

[1] we edited the article on 16:45, 15$^{\text{th}}$ Sep. 2021

the Wasserstein distance of two discrete distributions boils down to the following Linear Program [17]:

$$W_p(\mu,\nu)^p := \min_{\mathbf{x}\in\Gamma(\mu,\nu)} \sum_{M\times M} d(m_i,m_j)^p \cdot \mathbf{x}(m_i,m_j)$$

primal form :				**dual form** :			
minimize	z	$=$	$\mathbf{c}^T\mathbf{x}$,	maximize	$\tilde{z}$	$=$	$\mathbf{b}^T\mathbf{y}$,
so that	$\mathbf{Ax}$	$=$	$\mathbf{b}$	so that	$\mathbf{A}^T\mathbf{y}$	$\leq$	$\mathbf{c}$
and	$\mathbf{x}$	$\geq$	$\mathbf{0}$				

Here, the vectorized matrix c defines the transport cost, i.e. $c_{ij} = d(m_i, m_j)^p$ represents the L_p-distance of the points m_i and m_j (M is the discrete metric space on which the probability distributions are defined). The optimization variable x represents the transport plan as in Figure 2.3. Each entry of x must be non-negative. The constraint $A\mathbf{x} = \mathbf{b}$ is satisfied, if the marginals of x equal the distributions μ, ν. Note that only the primal problem yields an explicit transport plan while the dual form has less variables and is much faster. Due to the strong duality of a Linear Program the resulting solution however is the same. In practice we therefore use the dual form to compute Wasserstein distances.

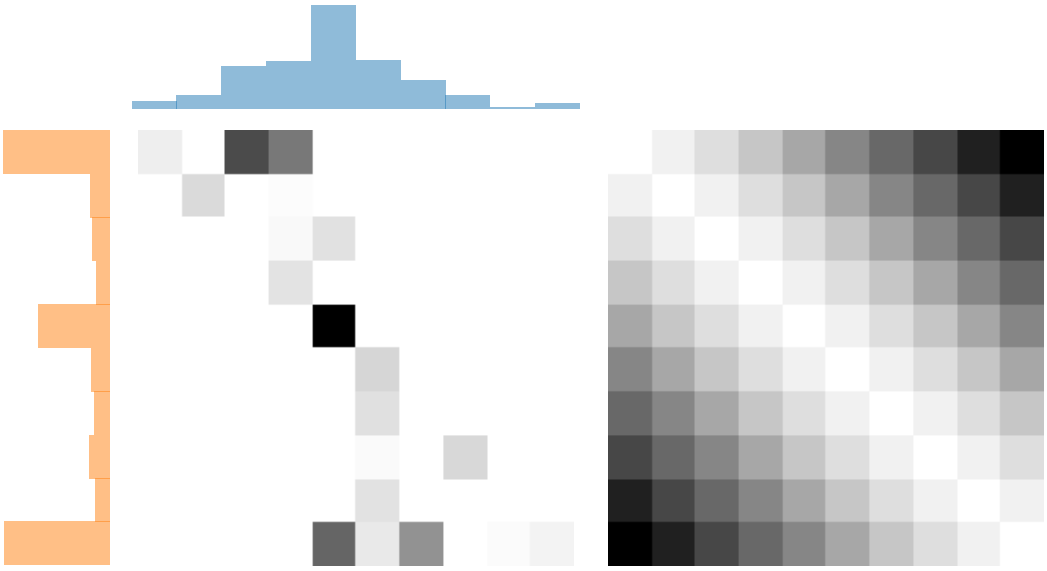

Figure 2.3: *Wasserstein distance as Linear Program (visualization inspired by [18]).* The optimal transport map X of two probability distributions ν (orange) and μ (blue) is shown in the center. The heatmap on the right represents the cost matrix C.

Scalability

A Linear Program can solve the exact Wasserstein distance with unique optimal solution. However the complexity increases with the number of variables in $\mathbf{x}$. If we add a second dimension to the histogram in Figure 2.3 each marginal distribution gets squared in the number of bins, thus the variables in $\mathbf{x}$ increase biquadratically. However, in high dimensions most of the bins will have zero probability mass, that is, no sample will be observed at that point. So the complexity of the Linear Program is upper bounded by the number of samples in each distribution. In Figure 2.4 we show the same Linear Program but just exploit the fact that each row or column which should sum to zero can directly be left out in the formulation. The cost matrix is then just given by the pairwise L_p-distance of all samples which can be vectorized and computed efficiently.

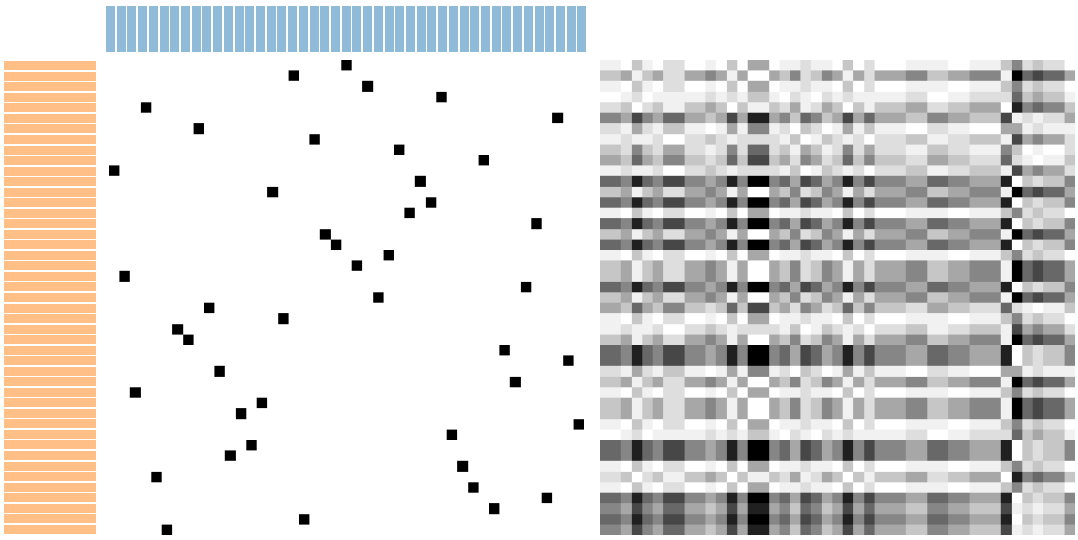

Figure 2.4: *Scalability of Wasserstein distance.* The marginals are considered to be uniformly distributed over their sample space. The cost matrix on the right contains as entries the pairwise distance of the samples. In the center the optimal transport map is shown.

Performance

The approach described in the last section is indifferent to the number of features, since we can compute the pairwise Euclidean distance of the samples regardless their dimension. In particular this means, that they no longer need to be samples from a discrete distribution. Even when we deal with continuous data, such as samples from a Gaussian, we can consider these samples to be discrete and set up the Linear Program as in Figure 2.4. Thus we can run an experiment to find out how accurate the exact Wasserstein distance computation becomes for growing

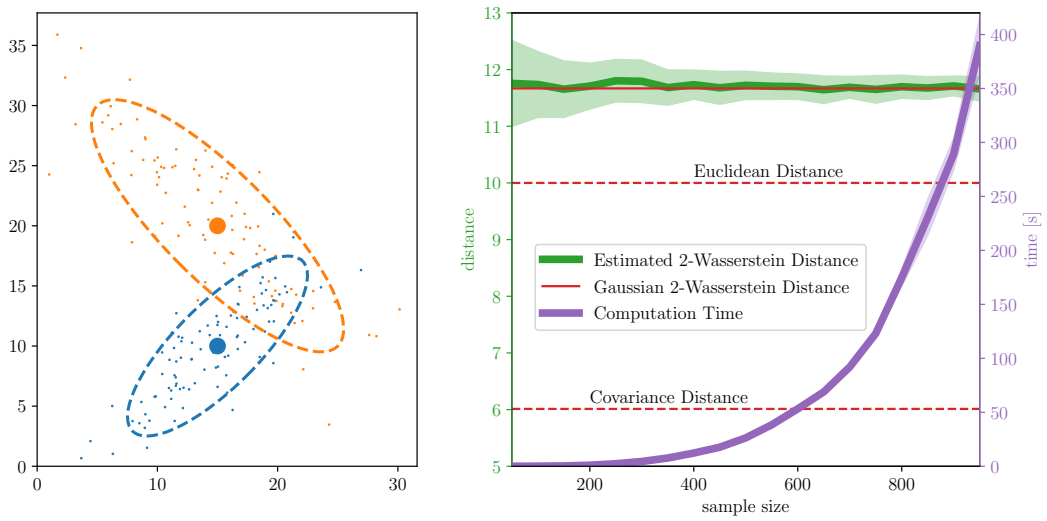

Figure 2.5: *Computation Time and Accuracy.* The Wasserstein distance of two Gaussians is computed using a different number of samples. The red lines show the theoretical distances of the Gaussians. In purple and green the computation time and estimates are shown with their standard deviation in 50 trials.

sample sizes. Figure 2.5 shows the convergence towards the ground truth calculated with the closed-form solution. The dashed red lines represent the parts of the sum in the formula. We can see again, that the Euclidean distance plays a bigger role even though the orientation of the covariance is rotated by $\frac{\pi}{2}$. When looking at the computation time we can remark that it grows almost exponentially. This may be an artefact of the setup of the experiment. Since the sample size is equal for both Gaussians, also both of them have the same probability distribution (both uniform with $\frac{1}{S}$ on a different support). Thus the problem becomes an Integer Linear Problem which has exponential complexity [19].

2.3 Wishart Distribution

In the next chapter we will generate random Gaussian distributions. This requires a generative model to draw their means and covariances from, i.e. μ and Σ in

$$\mathcal{N}(\mathbf{x}; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d \, |\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right).$$

While it is straight forward to generate random means, sampling covariance matrices is not as easy. One way to do so is to draw them from a **Wishart distribution**, which can be understood as the result of the following procedure: First draw ν samples from the multivariate normal with zero mean and a scale covariance, then estimate the covariance matrix of these samples. A formal way to define this random variable is shown in Definition 2.3.1.

Definition 2.3.1 *Let Σ be a $d \times d$ positive-definite, symmetric scale matrix and $\nu \geq d$ a positive integer. The probability density function of the Wishart distribution is then given by*

$$\mathcal{W}(\mathbf{\Lambda}; \nu, \Sigma) = \frac{|\mathbf{\Lambda}|^{(\nu-d-1)/2} \, \mathrm{e}^{-(1/2)\,\mathrm{tr}(\mathbf{\Sigma}^{-1}\mathbf{\Lambda})}}{2^{\nu d/2} \, |\mathbf{\Lambda}|^{\nu/2} \, \Gamma_d\left(\frac{\nu}{2}\right)}$$

Figure 2.6 visualizes covariance matrices drawn from the Wishart distribution. As a reference we show in the top row samples which are generated by another method: the length of both eigenvectors as well as their angle is sampled uniformly from an interval. This gives a uniquely defined covariance matrix. We observe that this method yields less homogeneous covariance matrices, i.e. they have larger pairwise distances. The Wishart samples only show this effect for minimal $\nu = 2$ in the middle row.

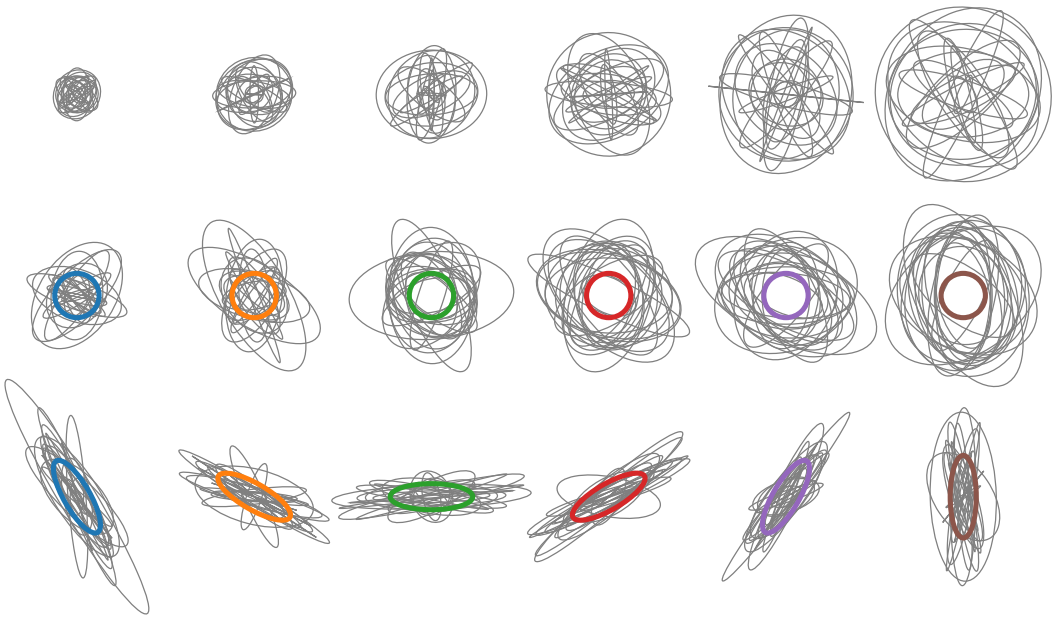

Figure 2.6: *Sampling Covariance matrices.* Top row: Uniformly sampled covariance matrices with increasing interval. Middle row: Wishart samples with identity scale matrix and increasing $\nu \leq 7$. Bottom row: Wishart samples with rotated scale matrix and fixed $\nu = 2$.

Chapter 3

Analysis: Finding Structure in Covariance

The objective of the following analysis is to find use-cases for the method, i.e. hierarchical datasets where information is stored in the unit covariances as previously explained. We will start this chapter by creating synthetic data to design a proof of concept. From Section 3.2 on we will apply the method to real-world data to show that in practice, too, new insights can be won by the inclusion of covariance.

3.1 Synthetic Data

To model the hierarchical setup of later experiments we define a **Hierarchical Gaussian Mixture** (HGM). By creating synthetic data from this model we can encode information in the covariances. Thus we will be able to show that Wasserstein t-SNE is actually able to separate units that share the same mean but have different covariances. To measure an accuracy true labels are necessary. The HGM will therefore consist of multiple classes from which the units are drawn. Of course, the information about its class is not given to the embedding method Wasserstein t-SNE, so it can't use the label for clustering.

Proposition 1 *Let $\mathcal{U}, \mathcal{N}, \mathcal{W}$ be Uniform, Normal and Wishart distributions respectively. A Hierarchical Gaussian Mixture is then defined by the number of classes (K), the number of units per class (N_k), the number of samples per unit (M_{nk}) and their dimension F, where*

- *each class is defined by a Gaussian and a Wishart distribution $C_k = [\mathcal{N}_k, \mathcal{W}_k] = [\mathcal{N}(\nu_k, \Gamma_k), \mathcal{W}(s_k, \Lambda_k)]$ with $\nu_k \in \mathbb{R}^F$, $\Gamma_k \in \mathbb{R}^{F \times F}$, $s_k \in \mathbb{R}$ and $\Lambda_k \in \mathbb{R}^{F \times F}$*
- *each unit is a Gaussian distribution $X_n = \mathcal{N}(\mu_n, \Sigma_n)$. Their means are samples from the Class-Gaussian $\mu_n \sim \mathcal{N}(\nu_k, \Gamma_k)$ and their covariances are samples from the Class-Wishart $\Sigma_n \sim \mathcal{W}(s_k, \Lambda_k)$*
- *the samples of each unit are given by $S_m \sim \mathcal{N}(\mu_n, \Sigma_n)$*

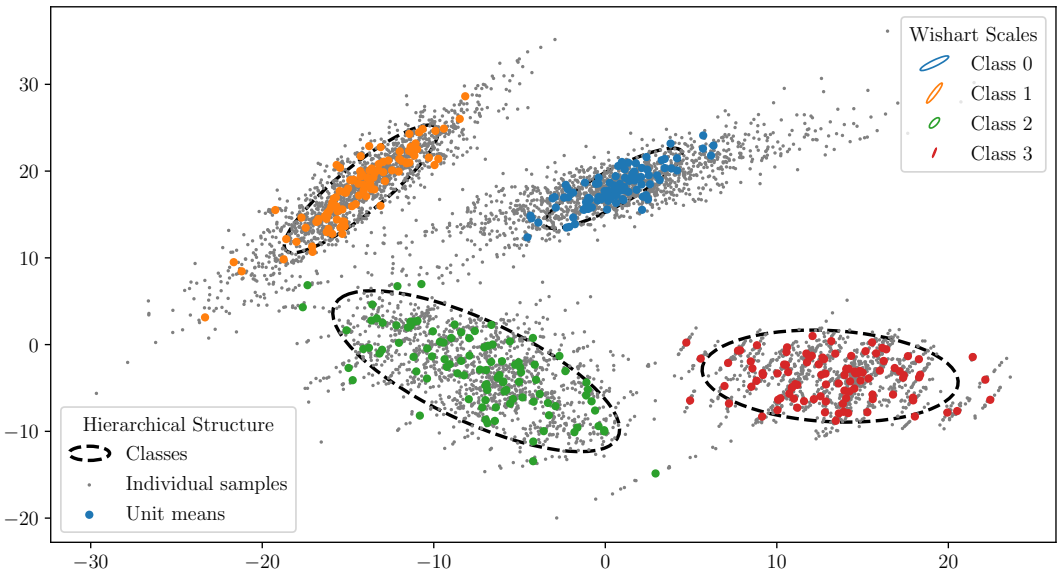

Figure 3.1: *Hierarchical Gaussian Mixture.* The dataset contains $N = 100$ units with $M = 20$ samples each. The units belong to $K = 5$ classes with $a = 20$ and $b = 5$ (default parameters).

3.1.1 Example

To illustrate a 2-dimensional HGM we create a dataset with random parameters in Figure 3.1. This works by defining two additional parameters a, b that set the space from which the class parameters are drawn.

Definition 3.1.1 *For $F, K, N, M \in \mathbb{N}_+$ and $a, b \in \mathbb{R}_+$ the random HGM is given by the sampling procedure $s_k \sim \mathcal{U}([F, 2F])$, $\nu_k \sim \mathcal{U}([-a, a]^F)$, $\Gamma_k \sim \mathcal{W}(F, b \cdot \mathbf{1}_F)$ and $\Lambda_k \sim \mathcal{W}(F, \mathbf{1}_F)$. It has Class-Mean-Distance a and Class-Scale-Variance b.*

For the random HGM in Figure 3.1 we choose to draw the means of the Class-Gaussians from a large area so that the classes are very distinct from each other. The further away the classes are, the easier it will be for an algorithm to cluster the units correctly. Similarly the dashed black curves indicate the covariance of the Class-Gaussians. The larger the variance within a class, the more outliers will add noise to the dataset. The same holds for the Class-Wisharts and its covariance matrices. We can see that the green class on the lower left has larger variation: some of its units show a different shape in their samples than others. However, all covariances within a class are samples from the same Wishart, so their distance should be smaller compared to the covariances of units in other classes.

3.1.2 Proof of Concept

Now that we have properly defined a HGM, we can set the parameters in such a way that the information about the unit means is not enough to cluster the dataset correctly, while the Wasserstein t-SNE approach would successfully do so. In Figure 3.2 we can see a HGM that consists of $K = 4$ classes, two of which share the same Gaussian while two of them share the same Wishart. On the right the respective embeddings are shown. The Euclidean embedding expectedly doesn't capture the structure of the dataset. Its two clusters have multiple labels in it (orange-blue and red-green). Pure information about covariance isn't enough neither, so the Covariance embedding at the bottom only finds two clusters as well (blue-green and orange-red). The convex combination of both, the Wasserstein embedding, however separates all four classes from each other and can therefore be considered superior in this setting.

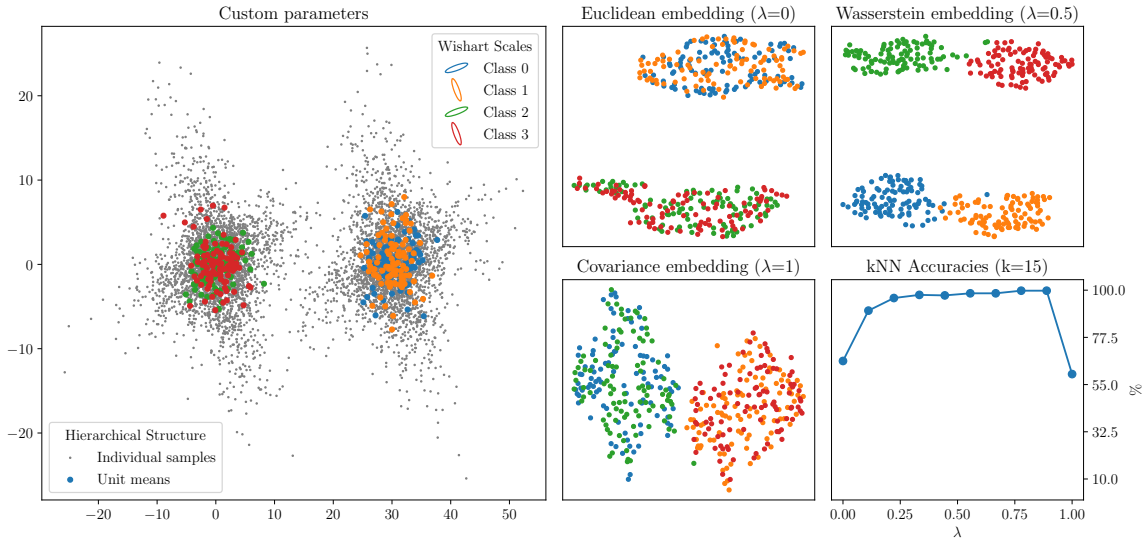

Figure 3.2: *Proof of Concept.* On the left a HGM is defined with $K = 4$ classes. On the right the respective embeddings are shown. On the lower right corner the nearest neighbor accuracy is plotted for 10 different values of λ.

3.1.3 Interpretation of λ

While it is obvious that in our method $\lambda \in \{0, 0.5, 1\}$ represents the Euclidean, Wasserstein and Covariance distance, it is not intuitive what the values for λ in between should represent. In a way λ puts emphasis on either the means or the covariance. We saw in Chapter 2 why this might be important. The problem we will face later is that even for large variations of unit correlations, the means will dominate the Wasserstein distance. A HGM which shows this effect is given in

Figure 3.3. This random HGM has large variation in its Class-Wisharts, however the Class-Mean-Distance is very small so all classes are on top of each other. As one can see on the lower right, the kNN accuracies increase until $\lambda = 0.6$ which might be an indication that the distance in covariance is a little more important than the distance of the means. In the Wasserstein embedding on the top right both these distances are weighed equally. However, this effect will become clearer when we look at real-world data, which is the topic for the next two sections.

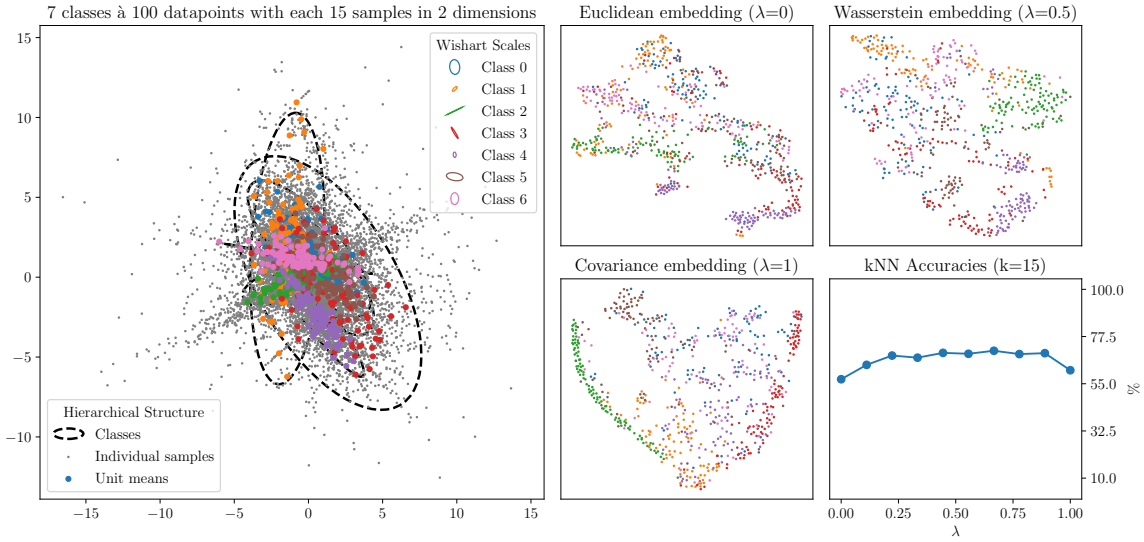

Figure 3.3: *Random HGM with embeddings.* On the left a HGM is defined with $K = 7$ classes. On the right the respective embeddings are shown. On the lower right corner the nearest neighbor accuracy is plotted for 10 different values of λ.

3.2 European Value Study 2017-2020

The first example of real-world data analyzed in this thesis is the **European Values Study** (EVS). From 2017 to 2020 this 40 year old study has been updated for each European country. It is available at the database of GESIS [3]. Its topics cover relevant demographic aspects such as views on morality, politics, economy etc. Moreover the NUTS-2 region of each participant is encoded in the questionnaire, so this dataset suits the hierarchical setting. In our analysis a NUTS-2 region will represent a unit from which we have as samples the participants of that region. In the first steps we will fit Gaussians to each unit and do the Gaussian Wasserstein analysis. Later however, we will be able to compute the exact pairwise Wasserstein distance for all units and construct the exact embedding. In Figure 3.4 we can see as a first example the histograms for *Poland* and *Switzerland* in two features.

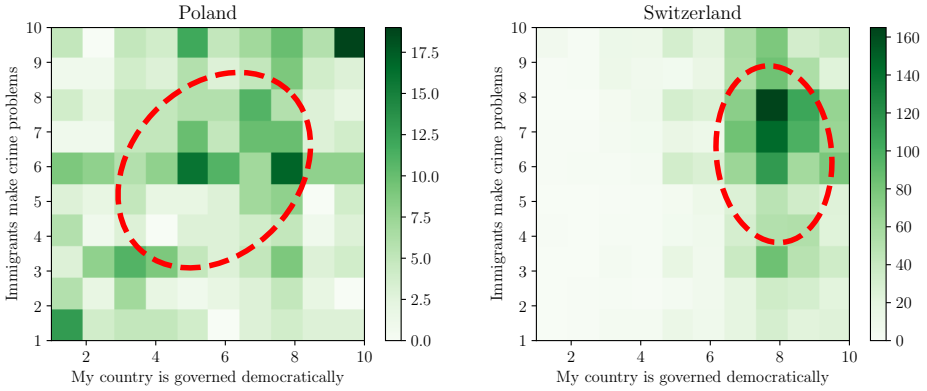

Figure 3.4: *Correlation in the European Values Study.* Histogram of survey results for two countries and two features. Answer options range from 1 (disagree) to 10 (agree). A Gaussian fit is given by the dashed red curve.

3.2.1 Preprocession

The EVS contains 107 questions, of which only a third offer numeric values as answers. We will focus on these as the others are difficult to compare numerically. In particular we chose 34 questions which are given in the Appendix A.1. Moreover we stretched all of the possible answers to an interval $[1, 10]$, since some questions had a smaller range.

Secondly we noted that in the dataset no NUTS-2 regions for Germany are given. In the documentation we found that they were omitted since otherwise it would have been possible to track individuals from their NUTS-2 region. As a workaround we used the higher NUTS-1 level for Germany instead, which corresponds to the 16 states.

An important question when dealing with survey data is how to use incomplete samples. For example, some participants answered only part of the questions. For this analysis we threw away all incomplete questionnaires. In that way we can calculate covariances without having to impute missing data. Of course a more sophisticated solution to this problem is possible, such as nearest neighbor imputation. Lastly we omitted all regions which had less that 40 participants, since it would not have been very accurate to compute covariances otherwise.

Logit transformation

We will later see that the Gaussian approximation of the units is not very accurate. To model bimodal distributions as well, we considered to transform the data with a logit function and then fit Gaussians in logit-space. However, we eventually decided to leave out this transformation as it didn't improve the embeddings significantly. For completeness we put the results in the Appendix B.2.

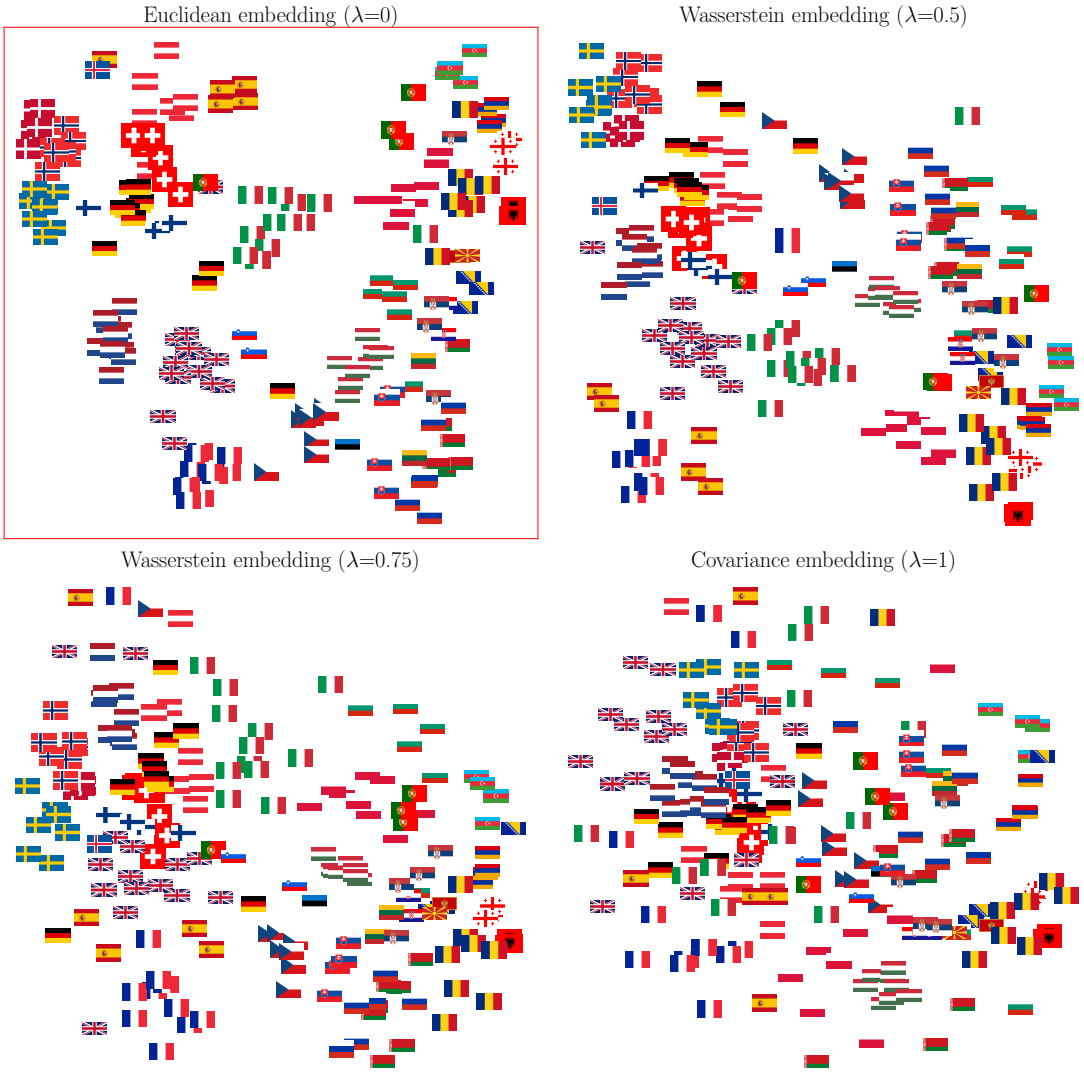

Figure 3.5: *EVS Gaussian Wasserstein Embeddings.* For four values of λ the resulting embedding is shown. The upper left embedding corresponds to taking the means only, whereas the lower right embedding only considers distance of covariances. The embeddings in between are using the interpolated distances.

3.2.2 Gaussian Wasserstein Embeddings

To get an overview of the dataset we run t-SNE with three different parameters $\lambda \in \{0, 0.5, 1\}$ and obtained the embeddings in Figure 3.5. The Euclidean embedding on the left shows meaningful structure: Western and Eastern Europe is separated with *Italian* and *Czech* regions bridging the two clusters. On the right the same structure is visible in the Covariance embedding, however not as sharp. Moreover the substructure of both the Western and Eastern cluster is lost. The Wasserstein embeddings in the middle, which are a combination of both, resemble the Euclidean embedding. This is not surprising since not much new information

is contributed by the Covariance embedding. A legend to the figure is provided in the Appendix A.3.

Feature Analysis

To find out from which features the structure of the embeddings emerge, we can encode the unit means by size of the flags. From Figure 3.6 it is obvious that religion plays a dominant role in the structure, but also other questions have a clear gradient in the Eastern-Western direction. This indicates that the unit means alone are able to capture the structure of the dataset. In that sense, the covariances might be additional noise to the unit means and do not contribute much to a more structured embedding.

Figure 3.6: *EVS Feature analysis.* The size of the label flag indicates the mean response to the question in that region. The Euclidean distance is used to embed the units.

Accuracy of Gaussian Approximation

Another reason why the Euclidean embedding seems better than the Wasserstein embedding could be that the Gaussian approximation of the units is inaccurate. It was easy to see in Figure 3.4 that the Gaussian fit is far from accurate. Another more drastic example was shown in Figure 1.3 in Chapter 1 where *Albania* had a bimodal distribution in one feature. Due to the Gaussian approximation this information might me weakened. In the next section we will therefore compute the exact Wasserstein distance to check whether it can strengthen the contribution of the higher moments.

3.2.3 Exact Wasserstein Embedding

Using the Linear Program described in Section 2.2 we computed the pairwise Wasserstein distance of all $N = 193$ NUTS-2 units. This procedure took 17 hours

on a desktop computer and was only possible at all if we reduced the samples size to maximally $S_n \leq 1000$. However only few units had significantly more participants (e.g. *Iceland*). In Figure 3.7 we compare the distance matrices of the Gaussian approximation and the exact Wasserstein distance. The differences are obvious: the Gaussian approximation systematically underestimates the distance, which might be due to the fact that multi-modal distributions collapse when being fitted by a Gaussian as in Figure 1.3. On the right the ratio of the absolute differences is given with most ratios being higher than $r \geq 0.4$. Despite the differences being quite large, the structure of both matrices looks similar. Due to the adaptive perplexity in the t-SNE algorithm it is understandable why in the resulting embeddings both methods show a similar structure. The respective figure is found in the Appendix 3.8.

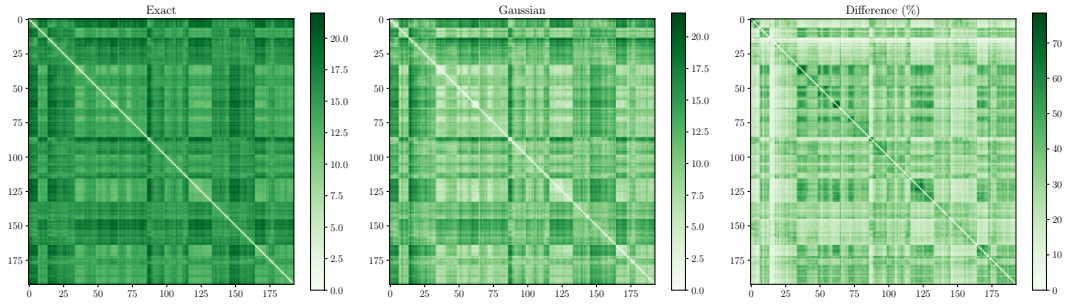

Figure 3.7: *EVS Distance matrices.* The exact Wasserstein distances are compared to the Gaussian approximations. On the right their relative difference is visualized.

Comparison to Euclidean Embedding

The ambition of the analysis was to find out whether Wasserstein t-SNE could improve a standard embedding, i.e. the Euclidean embedding. In the case of the Gaussian approximation this seemed not to be true. Now we want to inquire whether the exact Wasserstein distance embedding shows a qualitatively different structure. In Figure 3.8 we highlighted the unit *Île de France* (Paris). While this region had been assigned to the French cluster in the Euclidean embedding, it now stands alone in a different part of the embedding. Is this an artefact of the t-SNE initialization? Or does it represent a variation in the distance matrix?
One way to find out is to calculate the nearest neighbors of *Île de France*. In Figure 3.9 we see that its eight nearest neighbors are all French units when using the Euclidean distance. The result for the Wasserstein distance is rather different. Only two of the top-30 neighbors are French. This indicates that the structure in the Wasserstein embedding is not a t-SNE artifact. It could be a meaningful

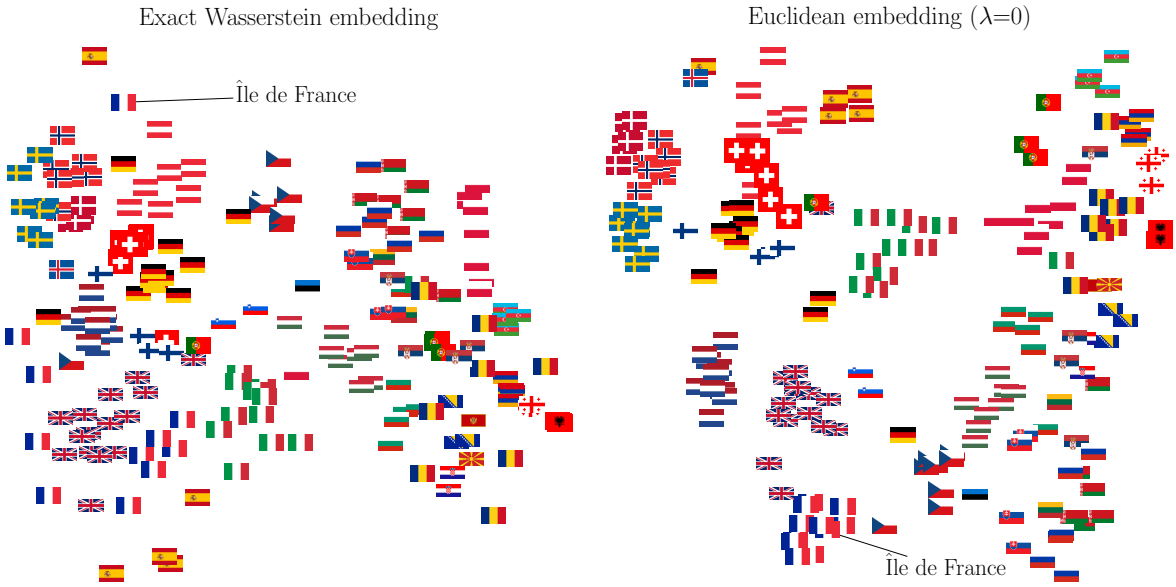

Figure 3.8: *EVS Wasserstein t-SNE embedding.* Left: Exact Wasserstein distance was used. Right: Euclidean distance was used. In both embeddings the unit *Île de France* is highlighted.

variation from which arises the hypothesis that in Paris, the capital, different features correlate. However, we have to take into account that all Wasserstein distances are very close to each other. After computing the affinity matrix the aversion of the unit against other French units might not remain significant. We will discuss later that it might be a general disadvantage of our method that the Wasserstein distance introduces a lot of noise to the distances. This emerges from the noise in the covariances.

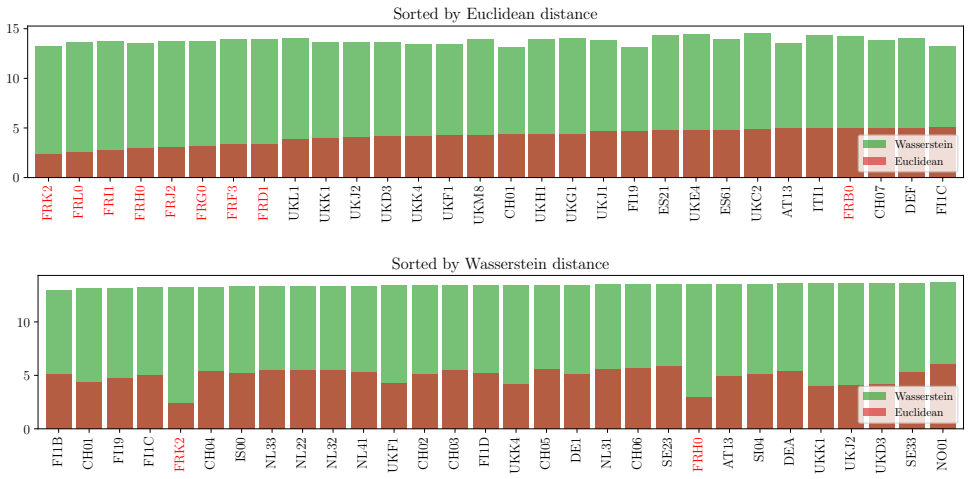

Figure 3.9: *Nearest Neighbors of Île de France.* The Wasserstein metric is an extension of the Euclidean distance so its values are strictly larger.

3.3 German Federal Election 2017

The German Federal Election (GER) is divided into 299 voting districts, each of which consists of roughly 150-850 poll stations. For simplicity we exclude voting by mail. In this analysis one voting district will be considered as a probability distribution over poll stations, i.e. the poll stations are samples from the voting district as in Figure 3.10. We shall find that certain parties correlate differently within all over the country. We thus find structure in the covariance of the units and observe different clusters for different λ.

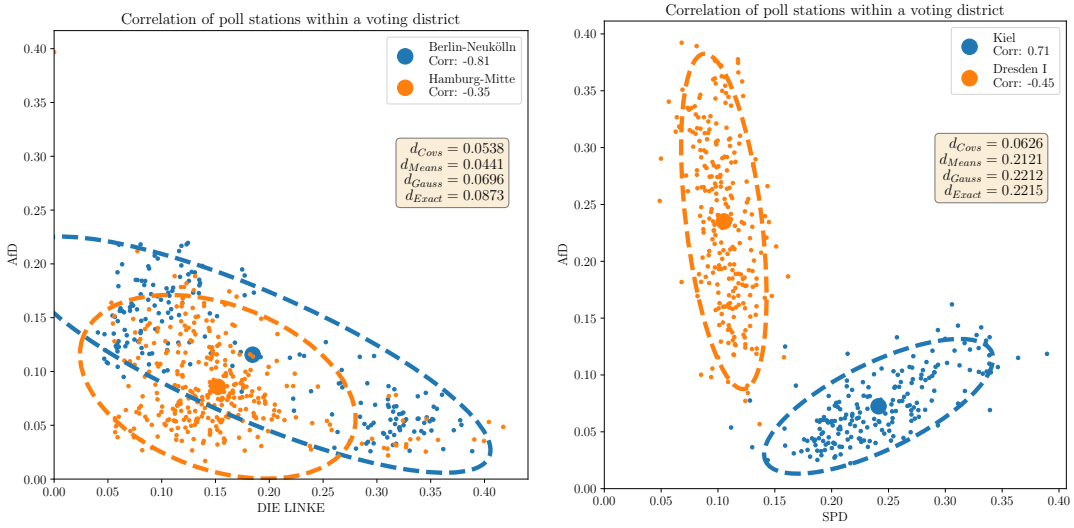

Figure 3.10: *German Federal Election 2017.* Left: Two urban voting districts of the same type with their distribution of poll stations. Right: Two voting districts with different types.

3.3.1 Preprocession

The GER dataset is available online [4] but comes in absolute numbers, i.e. for each poll station the absolute number of votes per party are given. Voting by mail is registered with a completely different scheme so we have to omit these votes. As a first step we excluded any party that isn't active in all states, in particular this means that we only consider the main six parties, where we merged the Bavarian *CSU* into the national *CDU*. This is relevant, because otherwise spatial information could be directly inferred from the correlation of (non-active) parties. Secondly we computed the percentages from the absolute votes, that is, we divide by the number of voters per poll station. This erases information about the size of the poll station and is important because cities have a different poll station density than rural areas.

Thirdly we define labels for each voting district. For this analysis we chose the corresponding state. We consider this choice reasonable since different states may show different voting behavior. However it is far from obvious that the structure in the dataset should be able to separate the datapoints into well distinguished clusters of different labels.
Finally, since all percentages of the parties lie in the interval $[0, 1]$ we observed that the variance of a party result is dependent on the mean, e.g. that a party with mean zero can't have any variance. In the Appendix we show a brief analysis of this mean-variance correlation and apply a variance stabilizing transformation. However, for this section we keep the percentages as they are since it doesn't make any difference for the embeddings.

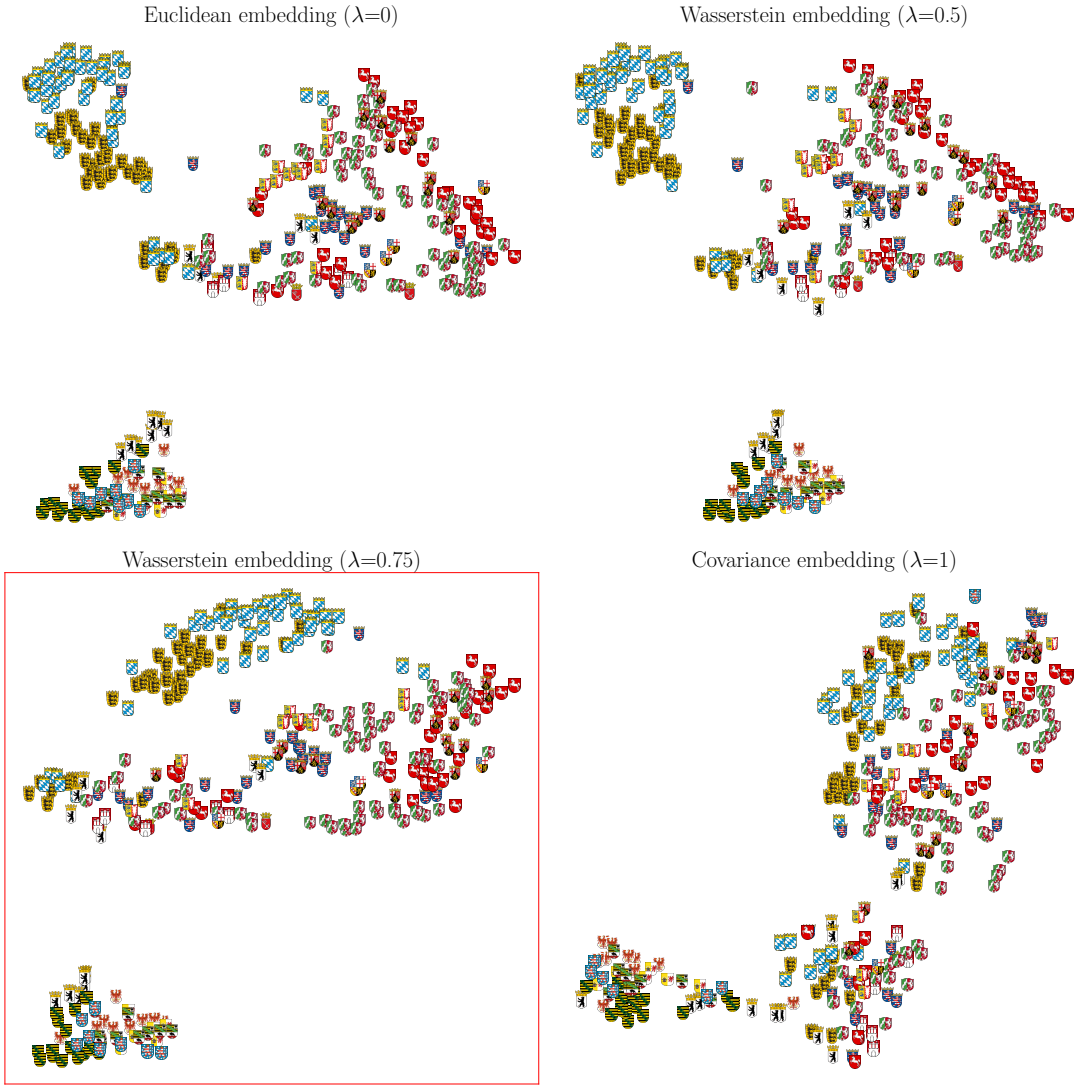

Figure 3.11: *GER Gaussian Wasserstein Embeddings.* For four values of λ the resulting embedding is shown. The upper left embedding corresponds to taking the means only, whereas the lower right embedding only considers distance of covariances. The embeddings in between are using the interpolated distances.

3.3.2 Gaussian Wasserstein Embeddings

Using the Gaussian approximation we can again compute the pairwise distances of the units and embed the voting districts as in Figure 3.11. In the Euclidean embedding we see three clusters. The legend in the Appendix A.5 shows that these clusters could be labeled as *southern Germany*, *western Germany* and *eastern Germany*. The Covariance embedding on the right however shows a different structure. There are again three clusters but with different units assigned to them: the cluster in the middle contains mostly cities. This cluster wasn't as isolated in the Euclidean embedding. The Wasserstein embedding, since it is an interpolation with $\lambda = 0.75$, combines both structures. Interestingly we can see all four clusters here. This specification qualifies the dataset for deeper analysis. It seems likely that we get a finer structure in the embedding if we as well consider the covariance of districts into the distance.

Features

Before we start analyzing where the difference in correlation emerges from, we can take a look at the mean structure. Which features are responsible for the embedding? From feature embeddings in Appendix A.6 it becomes clear that the *AfD* is mainly responsible for the separation of Eastern Germany, while *DIE GRÜNEN* and *CDU* yield finer gradients in the big cluster at the top. A more detailed analysis of the mean features as well as a legend with all names of the districts is given in the Appendix A.5.

Correlations

To understand the structure in covariance that was present in Figure 3.11 we can visualize the pairwise correlation of the parties for each district as in Figure 3.12. We used the Gaussian Wasserstein embedding which showed four distinct clusters, so that we can easily distinguish different areas and their interpretation. One can see that certain features (e.g. *AfD* and *SPD*) correlate in cities, but anti-correlate in eastern Germany. What causes this different correlation of parties? This would be an interesting topic of research for political scientists, we however can only speculate. A very general interpretation could be that both *SPD* and *AfD* are considered to be working class parties in western Germany. In the east however workers on the left traditionally vote *DIE LINKE*, so the *SPD* is rather an option for the middle class. The spatial separation of working and middle class is then reflected in the voting results of different poll stations.
In the Appendix we show more such examples. Moreover we find that the corre-

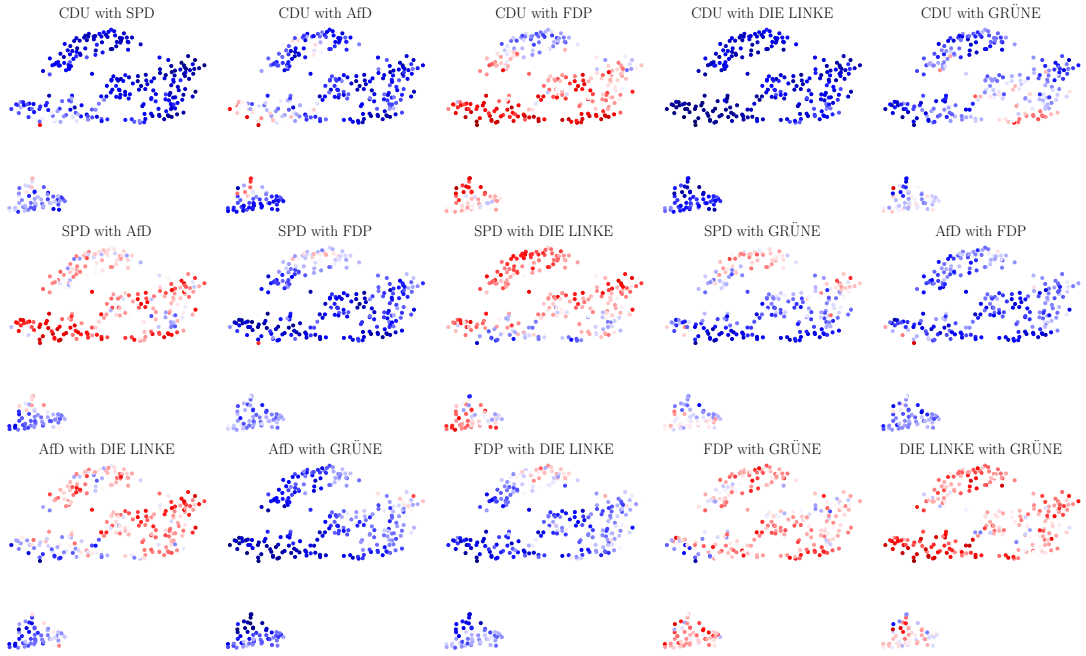

Figure 3.12: *Visualization of Party Correlations.* The Gaussian Wasserstein embedding with $\lambda = 0.75$ is used to cluster the units. The colors represent the correlation of the respective parties in that voting district, red being high (1) and blue being anti-correlation (-1).

lation is not independent from the means. This is indicated by Figure A.7 which shows structure in the correlation without using this information for the embedding. While one can imagine that the variance of a random variable depends on its mean if it is sampled from a closed interval, it is unclear why the correlation should be dependent on the mean. We did experiments to apply a variance stabilizing transformation to the data. The results of these experiments can be found in the Appendix B.3.

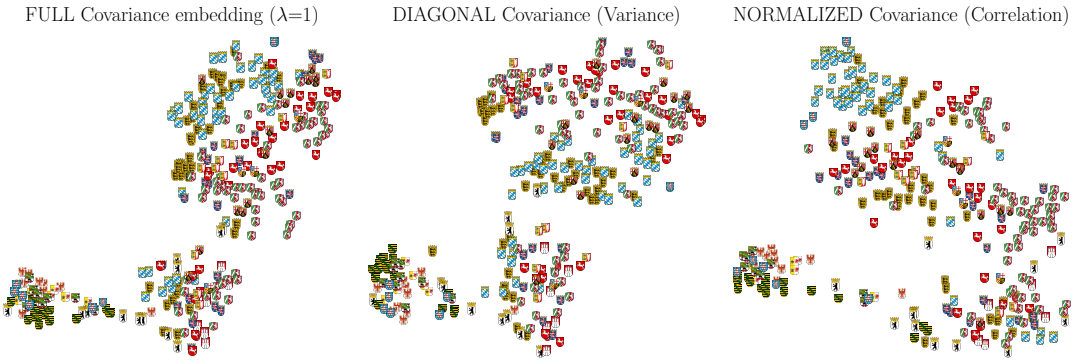

Figure 3.13: *GER Covariance analysis.* On the left only the marginal variances were included into the distance measure, while on the right the covariance was normalized to correlation.

Covariances

The Covariance embedding in Figure 3.11 shows a clear structure. Where does this structure emerge from? Covariance can be seen as a combination of correlation and marginal variance. In Figure 3.13 we show these two different aspects with their embeddings. The Variance embedding in the center suggests that the structure in covariance is rather due to marginal variance of each individual feature as the structure is similar to the embedding on the left. However we can observe from the right embedding, that also correlation shows structure and is therefore not negligible.

3.3.3 Exact Wasserstein Embedding

As explained in Section 2.2 we can use a Linear Program to calculate the Wasserstein distance of two sampled probability distributions. We did this for all $\frac{299 \times 298}{2} = 44551$ combinations to compare whether the exact Wasserstein distance was different from the Gaussian approximation. In Figure 3.14 we can see that indeed the results are quite similar. This indicates that the Gaussian approximation was appropriate.

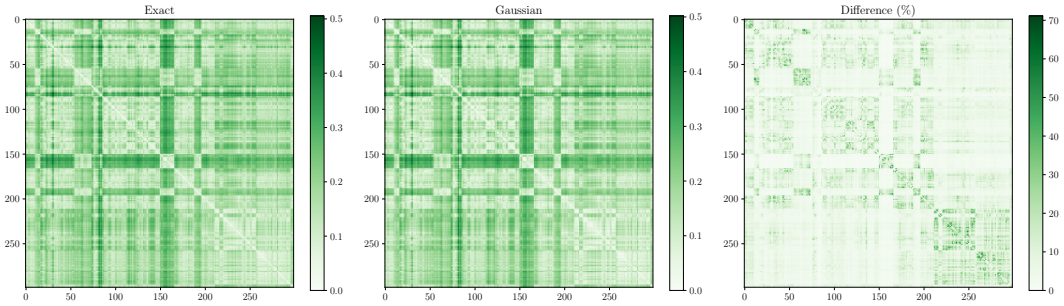

Figure 3.14: *GER Distance matrices.* The exact Wasserstein distances are compared to the Gaussian approximations. On the right their relative difference is visualized.

Comparison to Euclidean Embedding

To understand if the method includes new information to the visualization, the exact Wasserstein t-SNE embedding is compared to the Euclidean embedding in Figure 3.15. We see that the structure is rather similar. But, however, a few datapoints are embedded differently by the two approaches. *Berlin-Neukölln* for example is put to the cluster of cities by the Wasserstein approach, but to eastern Germany in the Euclidean embedding. This is again due to the nearest neighbors of the district: while in the Euclidean distance matrix *Berlin-Neukölln* has as

top-3 nearest neighbors *Berlin-Pankow*, *Potsdam* and *Berlin-Mitte* (all eastern Germany), the Wasserstein distance yields nearest neighbors from the cluster of cities.

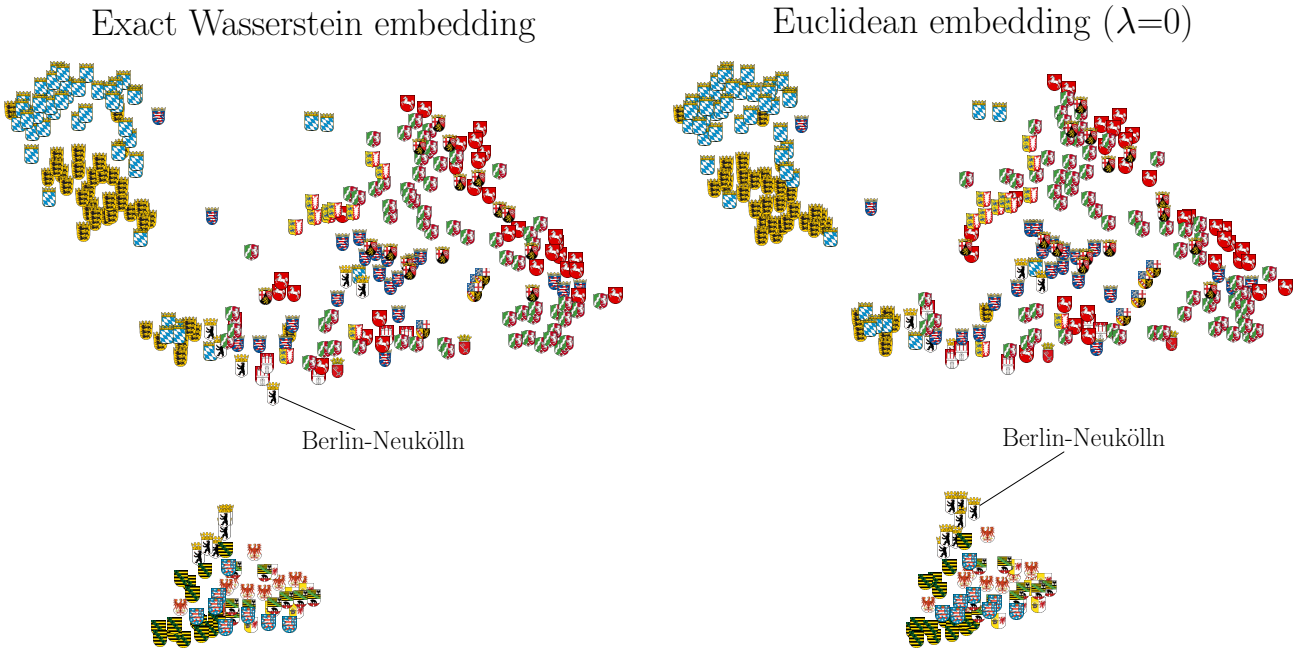

Figure 3.15: *GER Wasserstein t-SNE embedding.* Left: The exact Wasserstein distance was used to embed the dataset. Right: As a comparison the Euclidean embedding is shown. The voting district *Berlin-Neukölln* is highlighted in both embeddings.

A visualization of the nearest neighbors for *Berlin-Neukölln* is given in the Appendix A.9. While the Wasserstein metric adds distance to all of the units, those from eastern Germany have the strongest increase. More of these examples can be found in other areas of the embedding. The question remains whether these novelties are meaningful. When we look again at Figure 3.10 we see that *Berlin-Neukölln* has in fact an interesting structure of the poll stations, which is almost bimodal. Units like these make Wasserstein t-SNE an interesting method to experiment with.

Chapter 4

Discussion and Outlook

In this thesis we introduced Wasserstein t-SNE to visualize hierarchical datasets. The method uses a more sophisticated distance measure than collapsing each unit to its mean. We first looked at synthetic data, i.e. we designed the data in a way that the unit covariances contained information, and then showed that the method is able to visualize the structure in the embedding. Finally we applied Wasserstein t-SNE to real-world data and investigated whether there, too, we could find structure in the covariances.

4.1 Summary

While it had been very straight forward to design a proof of concept in Section 3.1.2, it turned out to be difficult to find a real-world datasets on which our method significantly changed the outcome. We tried the European Values Study in the hope, that the bimodal division of today's political landscape would yield units that share the same mean but have different covariances. We could find those examples, e.g. in Figure 1.3, but their effect was not strong enough to be meaningfully visible in the Wasserstein t-SNE embedding. The analysis in Section 3.2 showed that information about the means is sufficient to find roughly the same structure that our method finds with full information. However, small differences in the embeddings such as *Île de France* indicate that on a small scale the method could yield an improvement for other datasets.
The second dataset that we analyzed was the German Federal Election. Again we found small variations in the embedding, e.g. the position of *Berlin-Neukölln* changed. This special voting district is politically divided as we saw in Figure 3.10. The unit consists of samples that either vote largely for the *AfD* or *DIE LINKE*. That indicated that the voters for the respective parties live spatially separated, while in other units voters for these parties live in the same neighborhood. The exact Wasserstein distance, using the Linear Program solver, could take this bimodal structure of the unit into account. However there were only few such cases. Mostly the means dominated the covariances. A particular benefit of the λ-interpolation was then, that we were able to obtain finer structure, i.e. the interpolated distance with $\lambda = 0.75$ yielded an embedding which

showed four clusters instead of three. The feature to put more weight on the covariance is not possible with the exact Wasserstein distance but only with the convex generalization of the Wasserstein distance for Gaussians.
A third dataset, the Big-Five Personality Survey, was also analyzed. Since the results didn't vary much from the results of the previously mentioned datasets, we put that section in the Appendix B.1.

4.2 Future Work

In the multiverse of datasets we will continue to look for promising candidates to show the benefits of our method. But also other experiments could be done which are related to the topics of this thesis. In this section we give a small overview of possible extensions without going much into detail. These are sketches of projects which could help to understand the methods further or even improves it.

4.2.1 Distances of Covariances

As we have seen in the analysis, our intuition about the distance of Gaussian distributions is slightly off. Figure 4.1 indicates that a rotation of the covariance matrix doesn't change much of the distance which is in fact dominated by the means. However, these experiments were only in low dimensions. It could be interesting to design experiments that investigate the behavior of covariance distances in higher dimensions. As the parameters of a covariance matrix increase quadratically with the dimension and the parameters of the mean linearly, it seems reasonable to observe an effect here.

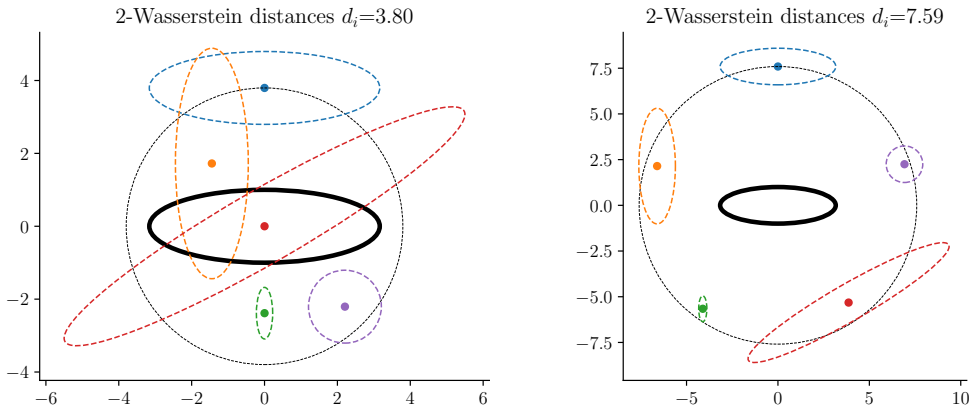

Figure 4.1: *Distance of Covariance matrices.* Five colored Gaussian distributions have equal 2-Wasserstein distance d_i to the reference Gaussian in the center.

4.2.2 Improving the Runtime

Another problem that we came across was the computation time of the exact Wasserstein distance. We noted that for higher dimensions the histogram of the sample space will be almost uniform, since it will become very unlikely that two individual share the same feature vector for a large feature space. In that case the Linear Program could become an Integer Problem which is known to be NP-complete [19]. That may be the reason why the computation time we measured in Figure 2.5 almost increased exponentially. There could consequentially be more efficient ways to compute the Wasserstein distance for this special case. Another possibility could be to estimate it from a sample subset as we have seen that also for small sample size the distance is reasonably accurate.

4.2.3 Finding more Use-Cases

The most urgent improvement however is the discovery of a clear use case, i.e. a hierarchical dataset where a large part of the information is encoded by covariance. This dataset is difficult to find due to two reasons: First, the dataset needs to come in hierarchical form with many units so that an embedding actually makes sense. Secondly, the information in the means must not be sufficient. The units should either be bimodal or show even more complicated distributions of their samples so that the Gaussian fits and moreover the means lack significant information. In the following we will describe possible datasets that we could image to work well, but which we haven't found yet.

Medical Data

The medical dataset that we have in mind is a study among patients. The units correspond to individuals and the samples are multiple measurements of features about the patient. For example, $N = 1000$ patients have 100-200 cells each. Each cell is defined by a set of features (e.g. shape, size). We want to embed the patients in 2D to find structure within the dataset. Rather than taking the mean of the cells as feature vector for each patient, we also take into account their feature covariances. The embedding could then reveal that certain correlations lead to a phenotype (e.g. sickness) of the patient which we would be using as labels.

Time Series

An interesting hierarchical dataset could as well involve time series. An example for such data could be the daily weight of individuals. We infer the Gaussian

Process for the time series and compute the Wasserstein distance of them, since a closed-form solution of the 2-Wasserstein distance for Gaussian Processes has recently been proposed [20]. Similarly, we could use multiple measurements of the same time series and compute the point-wise variances. A problem that arises in this scenario is that the temporal structure might be lost in the distance measure.

Topic Model

Another idea that regularly comes up is a dataset based on bag-of-words representations. While the units in such a dataset are intuitive (e.g. books, TV-shows) the problem we encounter here is the lack of a metric space. A bag of words doesn't let us easily define a distance other than component-wise metrics such as the Kullback-Leibler divergence. This however quickly diverges from the general design of the experiment in this thesis and would then lead to a different kind of model.

4.2.4 Inferring a HGM

The final extension that we thought about is connected to the HGM in Section 3.1. Similar to the Expectation-Maximization algorithm for Gaussian Mixture Models [21] one could think about inferring the parameters of the HGM from a given hierarchical dataset. The properties of the Wishart and Gaussian distributions might qualify the model for a deeper analysis. The benefit of an algorithm to infer the parameters of a HGM would be that it would yield a generative model from which we could sample new units. Furthermore we would have a quantification of how good the fits are and thus be able to compare different settings with respect to their likelihood. The derivation of that algorithm might get fairly complicated however and was therefore left for future research.

Appendix A

Supplementary Material

Table A.1: *Questions in the EVS dataset.* Each row corresponds to a question asked in the questionnaire with answers ranging from 1 to 10.

I have complete control over my life	Someone like me can do much for environment
I am satisfied with my life	There are more important things than environment
God is important in my life	Others should start to protect the environment
I consider myself 'on the right'	Environmental threats are exaggerated
Everyone is responsible for him/herself	Do you justify: claiming state benefits
The unemployed should take any job	Do you justify: avoiding a fare on public transport
Competition is good	Do you justify: cheating on tax
Incomes should be made equal	Do you justify: accepting a bribe
Private ownership should be increased	Do you justify: homosexuality
My country is governed democratically	Do you justify: prostitution
I am satisfied with the government	Do you justify: abortion
Immigrants take jobs away	Do you justify: divorce
Immigrants make crime problems	Do you justify: euthanasia
Immigration is a strain on welfare system	Do you justify: suicide
Immigrants should maintain their traditions	Do you justify: having casual sex
I would give money for the environment	Do you justify: death penalty

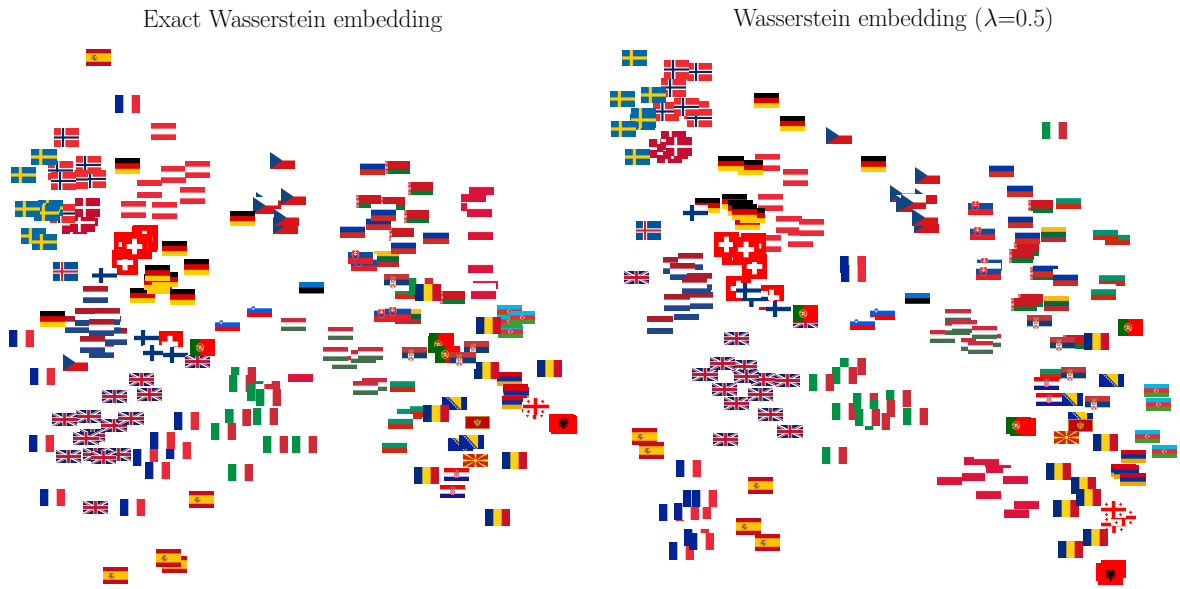

Figure A.1: *Goodness of Gaussian Approximation in EVS.* Left: The Wasserstein t-SNE embedding of the EVS is shown. Right: The Gaussian distances with $\lambda = 0.5$ were used to embed the comparison.

Figure A.2: *Legend of the Wasserstein EVS embedding.* The names of the respective NUTS-2 regions are given as annotations to the country flags.

Figure A.3: *Legend of the Euclidean EVS embedding.* The names of the respective NUTS-2 regions are given as annotations to the country flags.

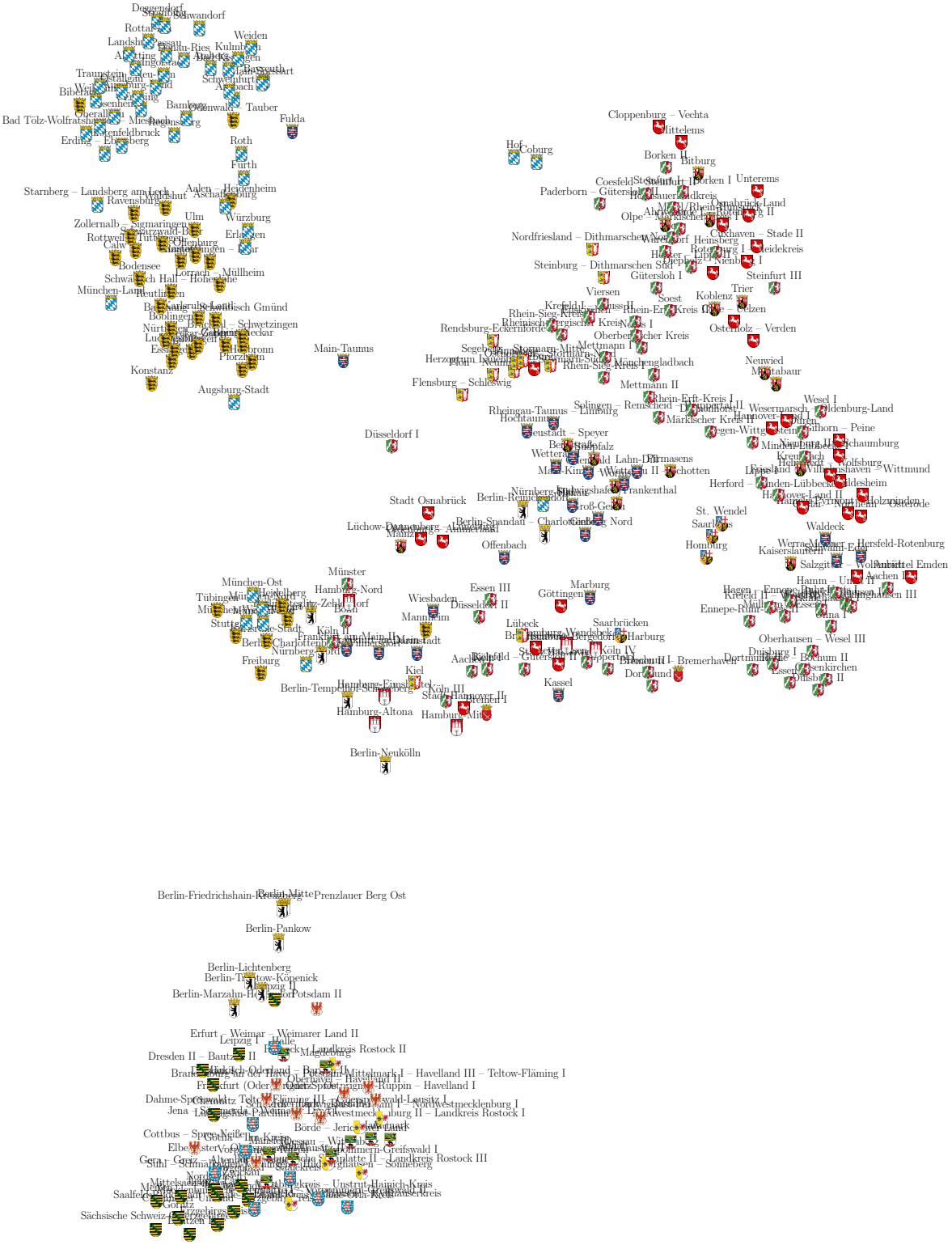

Figure A.4: *Legend of the Wasserstein GER embedding.* Names of the voting districts are given as annotations to the emblem of its state.

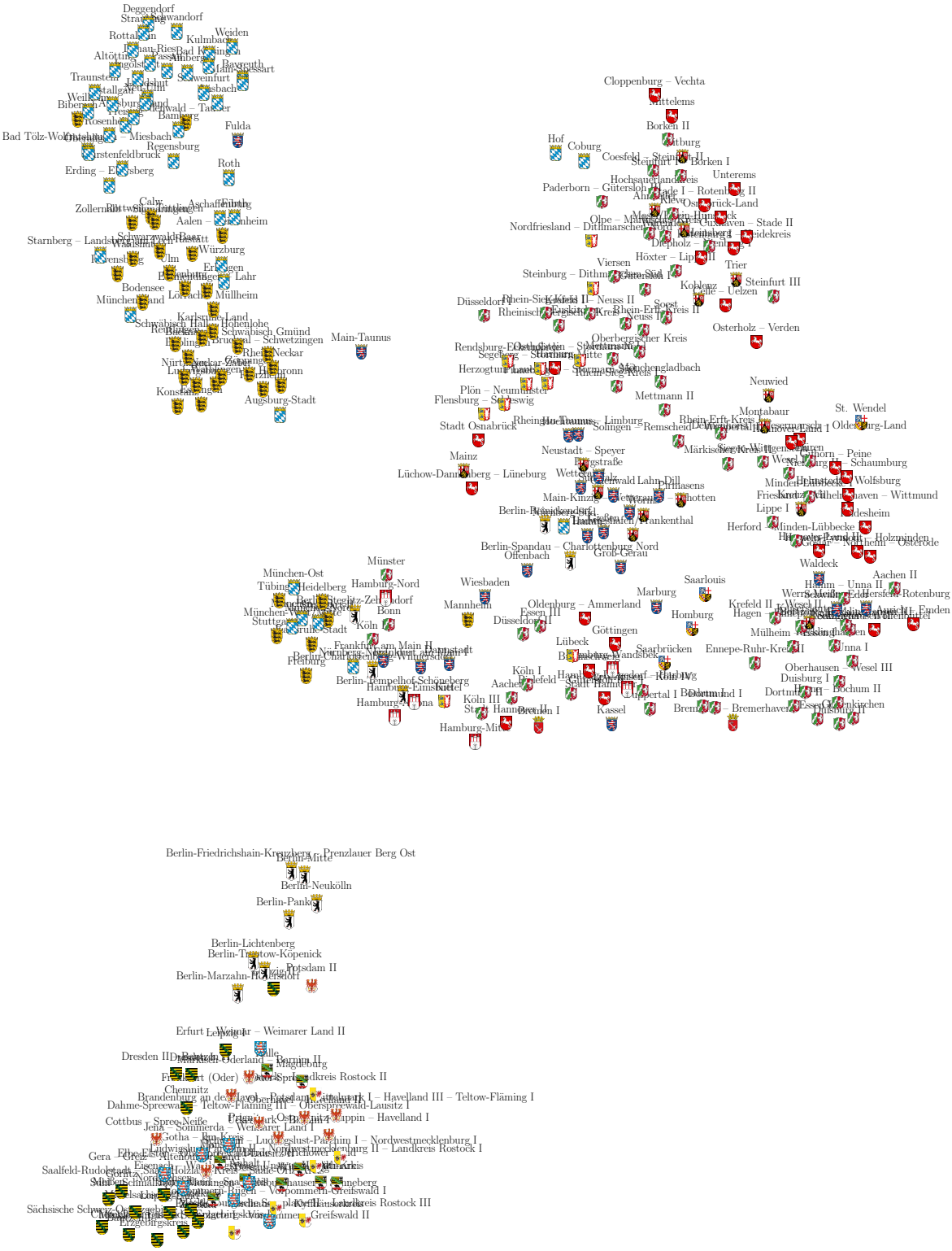

Figure A.5: *Legend of the Euclidean GER embedding.* Names of the voting districts are given as annotations to the emblem of its state.

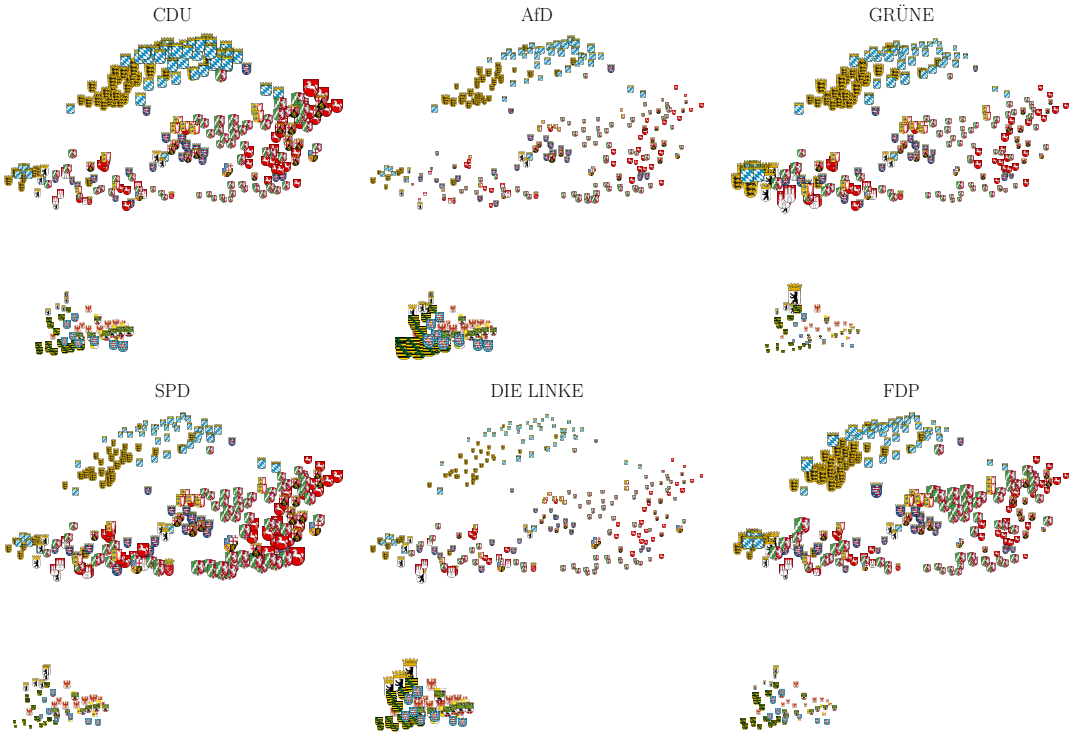

Figure A.6: *GER Feature analysis.* For six different parties the mean percentage of votes is encoded in the size of the emblem. The coordinates of each unit are given by the Wasserstein embedding.

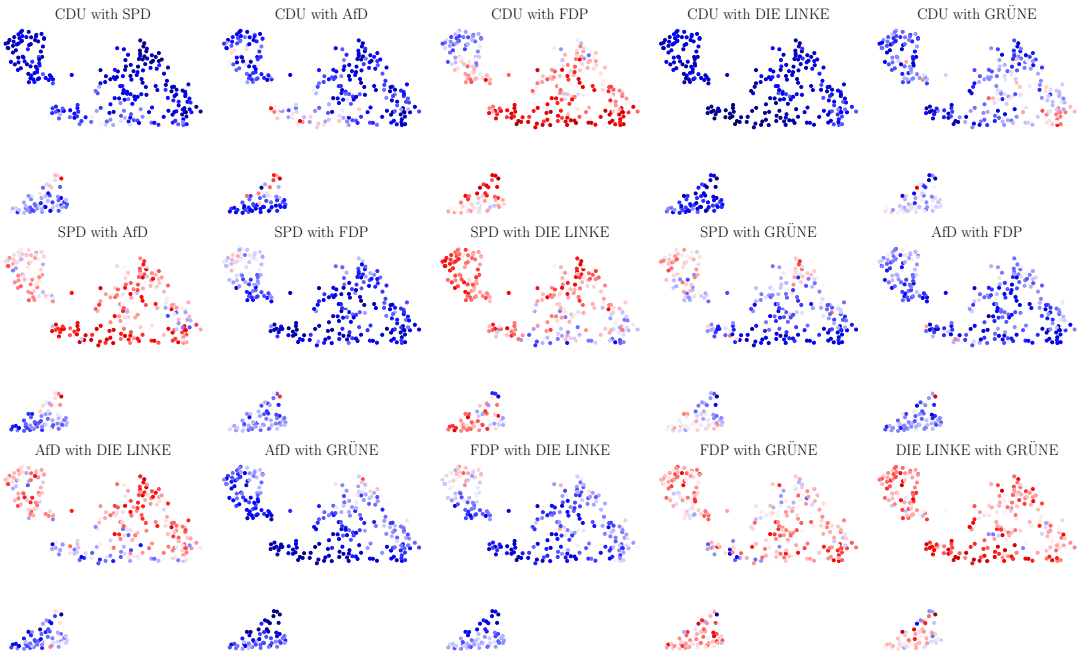

Figure A.7: *Party Correlations in Euclidean embedding.* The Euclidean embedding is used to cluster the units. The colors represent the correlation of the respective parties in that district, red being high (1) and blue being anti-correlation (-1).

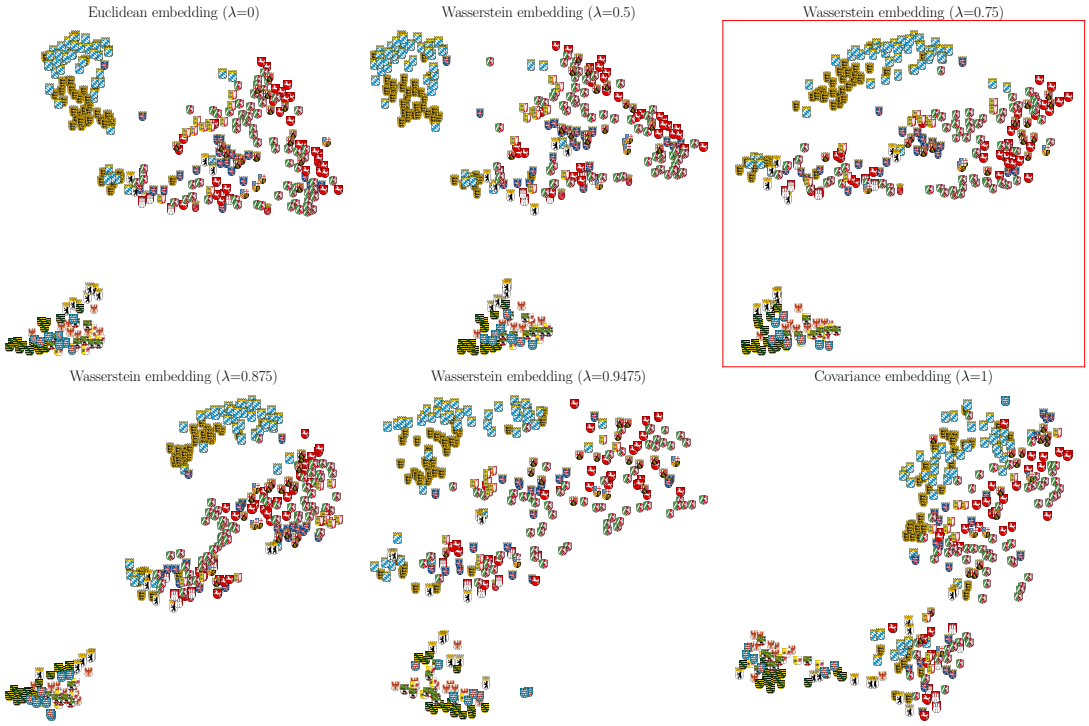

Figure A.8: *Evolution of embeddings with increasing* λ. For six values of λ the resulting embedding is shown. The upper left embedding corresponds to taking the means only, whereas the lower right embedding only considers distance of covariances. The embeddings in between are using the interpolated distance.

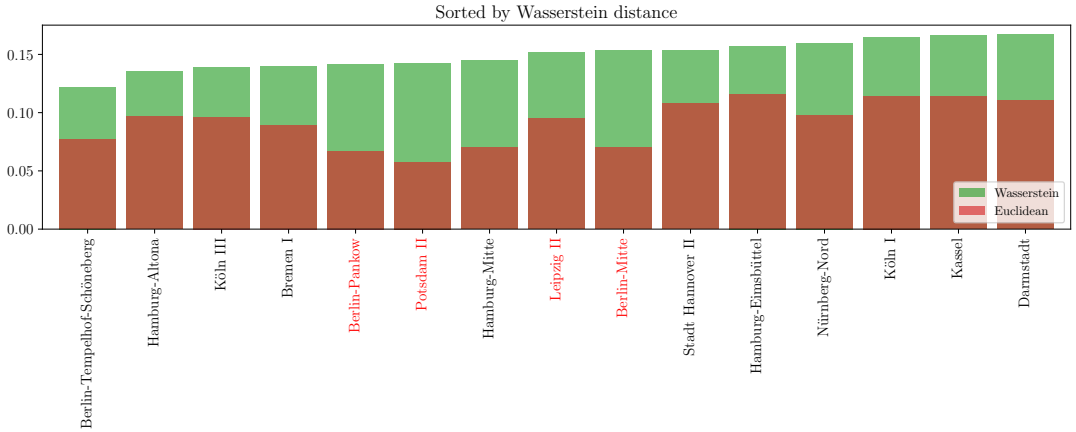

Figure A.9: *Nearest Neighbors of Berlin-Neukölln.* The red districts are from the cluster of eastern Germany, the districts in black are among the cluster of cities. The 2-Wasserstein metric is an extension of the Euclidean distance so its values are strictly larger.

Appendix B

Supplementary Analysis

Throughout this thesis we mentioned some experiments, which we did but didn't include in the analysis. In this chapter we will show them for the sake of completeness. The following results are additional work and can be considered completely independent from the rest of the project.

B.1 Big-Five Personality Traits

A third dataset that we analyzed was the Big-Five Personality Online Survey 2016 [7] (BIG5). It contains the personality traits of over two million participants and is hosted on Kaggle. After an extensive preprocession we used a clean subset of the data where it is insured that every participant is only included once. We particularly chose samples where the completion time is neither too short nor too long and furthermore constrained the number of participants per country to range from 60 to 12000. Thus 120 countries remained with a total of 219584 participants.

Table B.1: *Big-Five Personality Traits.* The rows of this matrix corresponds to individual personality traits of citizens of that country. The hierarchical form of the dataset is due to multiple participants per country.

	AGR	**CSN**	**EST**	**EXT**	**OPN**
United Arab Emirates	4.8	4.6	1.9	1.8	4.7
United Arab Emirates	2.1	4.3	2.5	1.5	4.7
United Arab Emirates	4.5	2.1	3.2	1.9	3.5
United Arab Emirates	3.9	2.4	2.0	1.4	3.9
United Arab Emirates	3.9	2.4	1.6	3.0	4.6
...	...	...	...	...	...
Zimbabwe	3.5	4.6	2.4	2.1	4.9
Zimbabwe	3.1	3.4	2.0	2.8	4.7
Zimbabwe	2.9	2.2	2.9	4.8	4.4
Zimbabwe	4.6	3.6	1.9	1.7	4.6
Zimbabwe	4.3	3.5	3.5	1.5	4.6

[219584 rows x 5 columns]

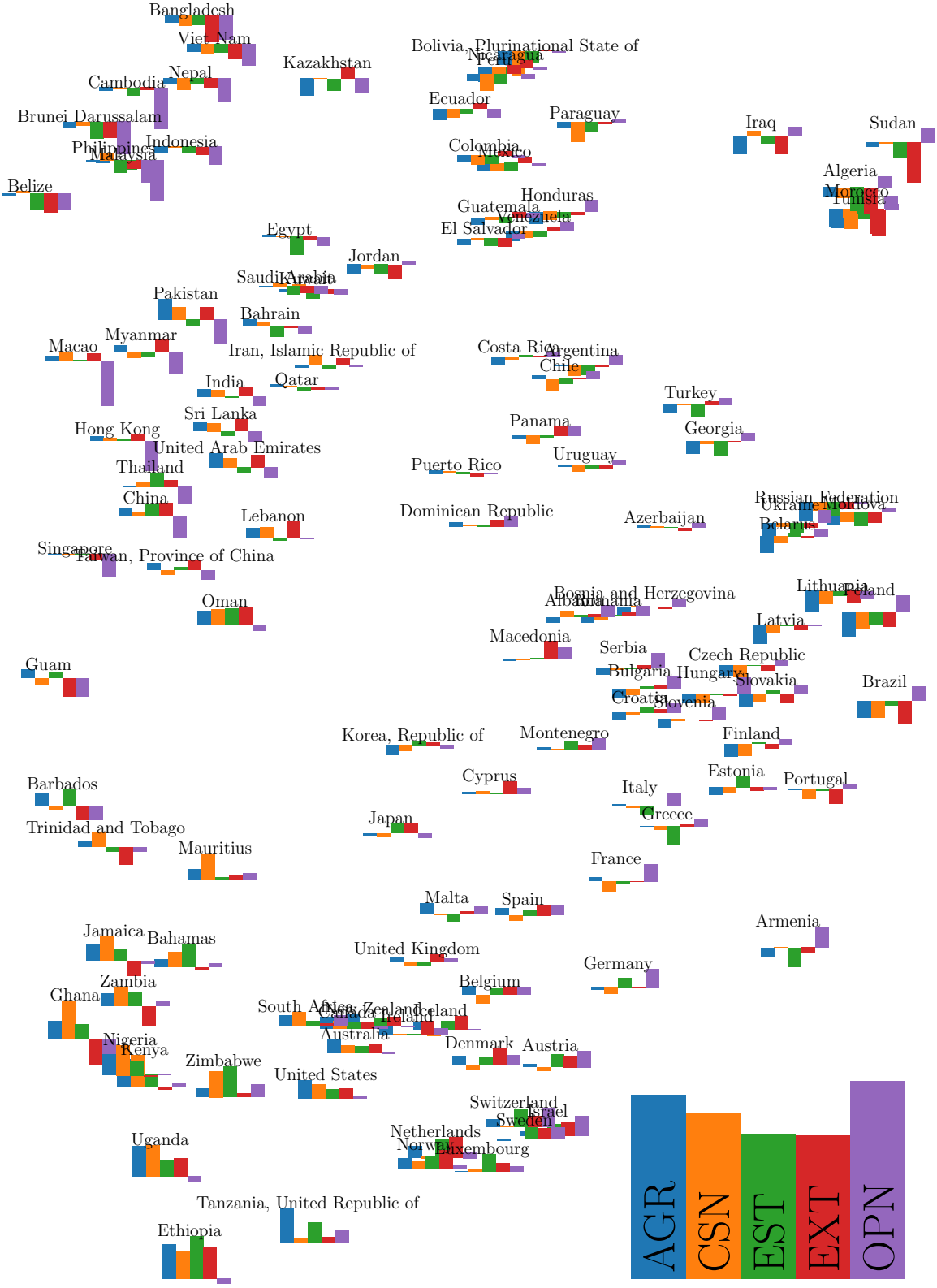

Figure B.1: *Legend of the Euclidean BIG5 embedding.* This embedding was created with the t-SNE algorithm. The five colors represent the five personality traits. In the lower left corner the mean personality is given. Each unit is annotated by its deviation from the mean.

The Big-Five model is used to describe personality traits by the main components: *Agreeableness* (AGR), *Conscientiousness* (CSN), *Neuroticism* (EST), *Extraversion*(EXT) and *Openness to Experience* (OPN) [22]. The BIG5 survey measured each trait by 10 questions which can be answered from one to five. If we average the answers per trait we obtain Table B.1. However, to find structure in the covariance we kept the full information. In any case, Figure B.2 shows that the questions related to the same trait correlate with each other, but there is also some other structure in the correlation. *China* for example shows clean squares while for other countries such as *Germany* or *Russia* the second and third trait anti-correlate. This corresponds to the independence of the personality traits. The Big-Five model was developed in the USA so it is reasonable to assume that it is particularly adapted. This could lead to an interesting structure in the correlation.

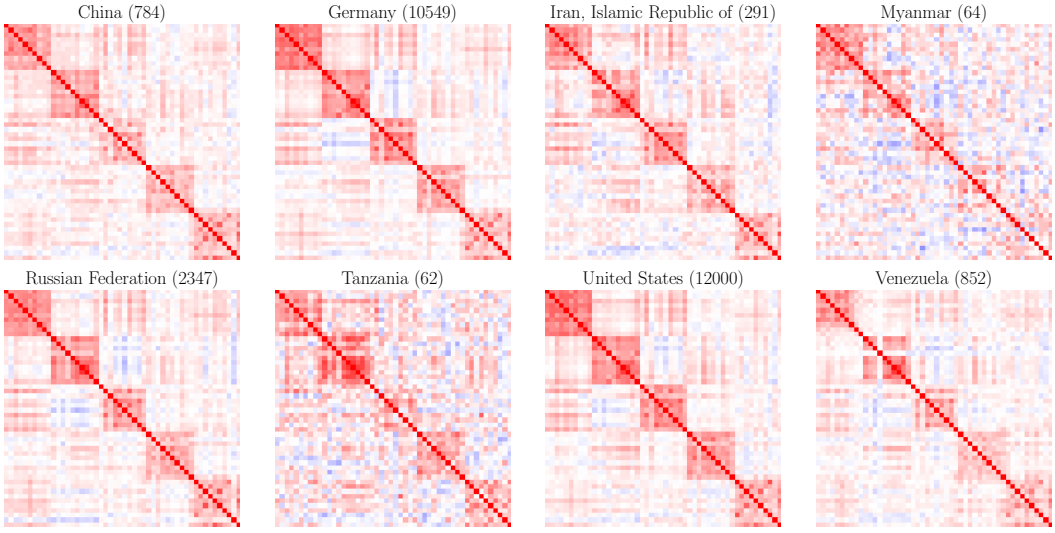

Figure B.2: *Correlation of questions related to the same trait.* For eight countries the correlation of participant's answers to the 50 questions is indicated by red (1) and blue (-1).

While we have shown the Euclidean embedding of the BIG5 dataset in the introduction, we now want to compare it to its Wasserstein embedding. Does the correlation in Figure B.2 have an effect? In order to compute the exact Wasserstein distances we decreased the sample size of some units to a maximum value of 500. Otherwise the computation would have been too long. In Figure B.3 we show the exact Wasserstein embedding of the dataset. We observe that the structure is still present, however not as sharp as in Figure 1.4.
The reason why the distinct clusters are lost in the exact Wasserstein embedding might again be the addition of noisy information. For the small range of possible answers from one to five the inclusion of covariance might not be very meaningful.

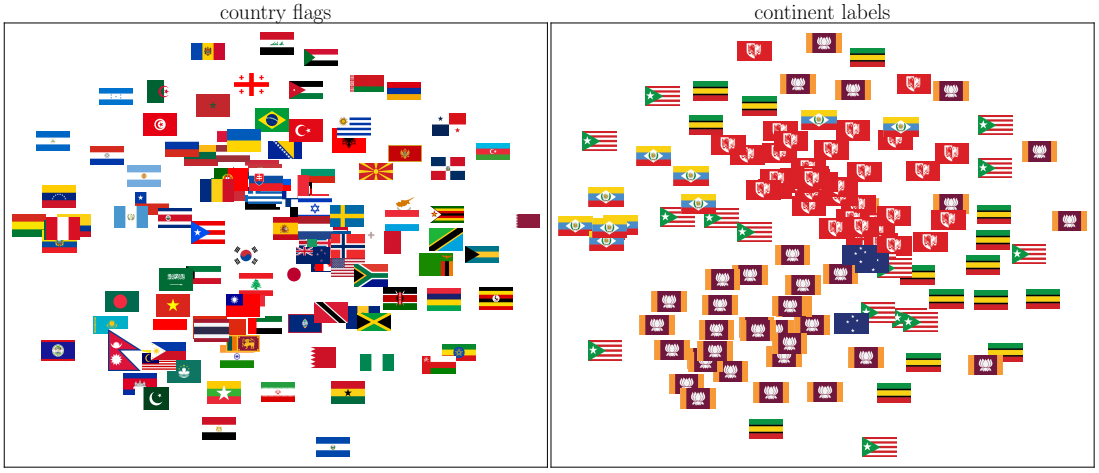

Figure B.3: *BIG5 Wasserstein embedding (flags from [8, 9]).* Left: t-SNE embedding of the mean personality per country using the flag of the country. Right: Each unit is visualized by its label (continent).

B.2 Logit Transformation

We have stated in Chapter 3.2 that the EVS data is not normally distributed for some units. The full information with the Wasserstein metric however added too much noise. A compromise would be to apply a logit transformation to the data and fit the Gaussians in logit-space.

Definition B.2.1 *Let $\sigma(x) = 1/(1 + e^{-x})$ the logistic (sigmoid) function. Its inverse is called the logit function and is defined as*

$$\text{logit}(p) = \sigma^{-1}(p) = \ln\left(\frac{p}{1-p}\right) \quad \textit{for} \quad p \in (0, 1)$$

Before we can apply the transformation properly to the data we add noise on top of the samples in order to get small derivations around the bins. Otherwise the transformation would use only ten bin-values as well and could not improve the shape of the fit. Also we have to squeeze the data into the interval $(0, 1)$ which is done linearly. The process of the transformation can be seen in Figure B.4. The results of the embedding however did not change much.

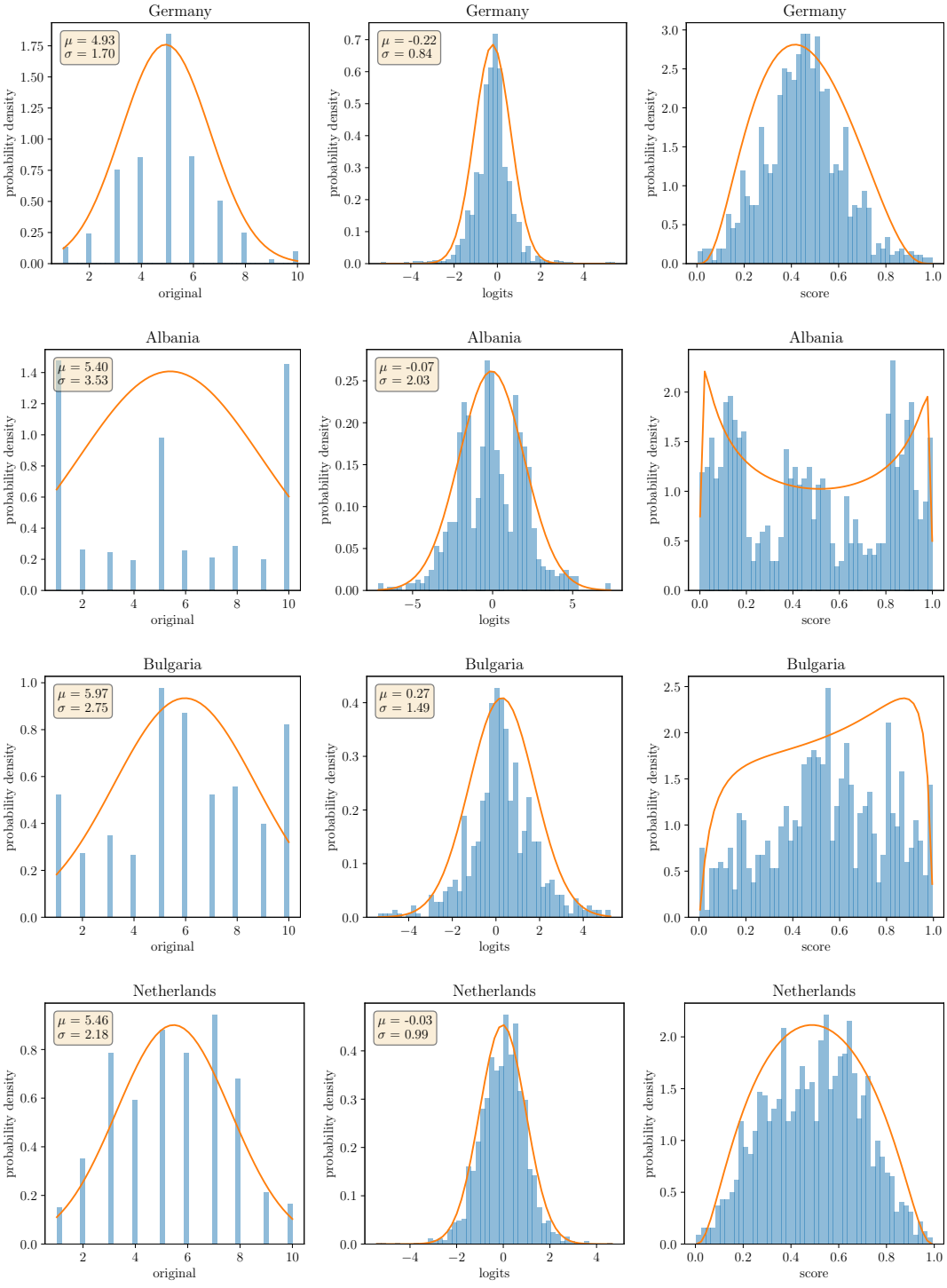

Figure B.4: *Logit Transformation of EVS units.* For different countries the original data is shown as well as the transformation to logit-space and transformation of the Gaussian fit backwards (score).

B.3 Variance Stabilization

We observed another problem in the analysis of the German Election, namely that there seems to be a correlation of the means and the variances. This is not surprising if we look at the space from which the samples are drawn. Percentages naturally lie in the interval $[0, 1]$. A unit with low mean must also have low variance, e.g. for $\mu = 0$ and $\mu = 1$ we must have $\sigma = 0$. The closer the units means are to $\mu = 0.5$, the higher standard deviation σ is possible. Therefore the quantities correlate. A way to disentangle mean and variance is given by the Variance Stabilization Transformation [23] which reads for binomial distributions

$$T(Y) = \sqrt{n} \arcsin(\sqrt{\frac{Y}{n}}).$$

Figure B.5 show the steps of this transformation on the GER dataset, where we compare for each unit and feature the standard deviation and mean. Interestingly, the last factor of $\sqrt{n}$ of the transformation yields correlation again. This is an indication that information is encoded in the size of a poll station as well, e.g. that cities have less poll stations per voting district and thus larger number of votes per station. Instead of introducing this prior knowledge into the analysis we rather left out the transformation.

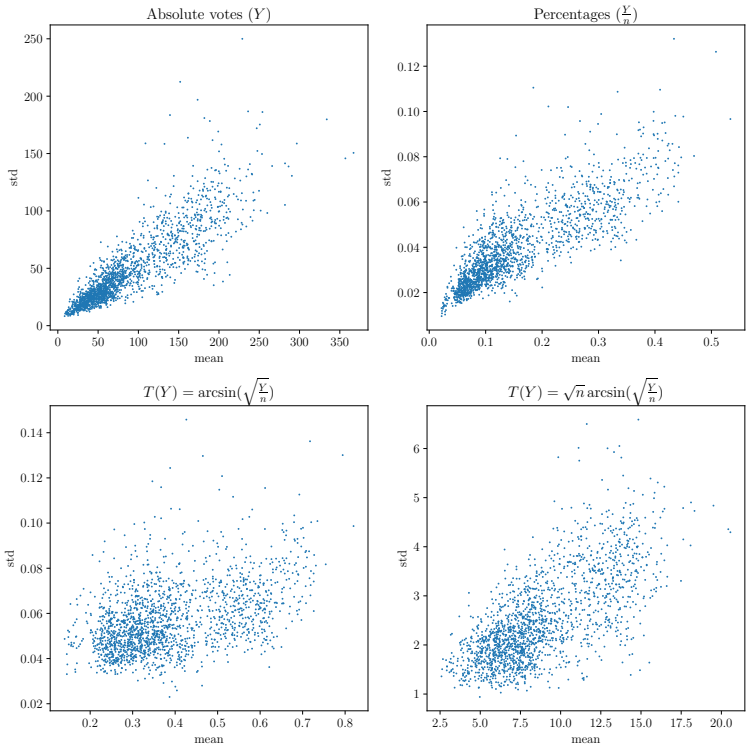

Figure B.5: *Variance Stabilizing Transformation in GER.* Four steps of the variance stabilization are shown.

References

[1] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE." *Journal of Machine Learning Research*, vol. 9, no. 11, 2008.

[2] L. V. Kantorovich, "The mathematical method of production planning and organization," *Management Science*, vol. 6, no. 4, pp. 363–422, 1939.

[3] EVS, "European Values Study 2017: Integrated Dataset (EVS 2017)," GESIS Data Archive, Cologne. ZA7500 Data file Version 4.0.0, https://doi.org/10.4232/1.13560, 2020.

[4] "Ergebnisse nach Wahlbezirken," Der Bundeswahlleiter, May 2021. [Online]. Available: https://bundeswahlleiter.de/dam/jcr/a2eef6bd-0225-447c-9943-7af0f46c94d1/btw17_wbz.zip

[5] D. Haitz. (2021, Mar.) Politische Landkarte von Baden-Würtemberg. [Online]. Available: https://twitter.com/d_haitz/status/1371534617291935747/photo/1

[6] "Landtagswahl Baden-Würtemberg 2021," Land Baden-Würtemberg, Statistisches Bundesamt, Mar. 2021. [Online]. Available: https://www.statistik-bw.de/Wahlen/Landtag/Kreise.csv

[7] "Big-Five Personality Test," Open Psychometrics, Apr. 2021. [Online]. Available: https://openpsychometrics.org/tests/IPIP-BFFM/

[8] flagpedia. (2021, Aug.) Flags of the world. [Online]. Available: https://flagpedia.net

[9] forteefly. (2021, Sep.) Flags of the continents. [Online]. Available: https://reddit.com/r/vexillology/comments/bmsp0n/flags_of_the_continents/

[10] C. J. Burges, *Dimension reduction: A guided tour*. Now Publishers Inc, 2010.

[11] A. Agrawal, A. Ali, and S. Boyd, "Minimum-Distortion Embedding," *arXiv preprint arXiv:2103.02559*, 2021.

[12] I. Borg and P. J. Groenen, *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

[13] D. Kobak and P. Berens, "The art of using t-SNE for single-cell transcriptomics," *Nature Communications*, vol. 10, no. 1, pp. 1–14, 2019.

[14] P. G. Poličar, M. Stražar, and B. Zupan, "openTSNE: a modular Python library for t-SNE dimensionality reduction and embedding," *bioRxiv*, 2019. [Online]. Available: https://www.biorxiv.org/content/early/2019/08/13/731877

[15] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," 2017.

[16] D. Dowson and B. Landau, "The Fréchet distance between multivariate normal distributions," *Journal of Multivariate Analysis*, vol. 12, no. 3, pp. 450–455, 1982.

[17] F. Mémoli, "Gromov–wasserstein distances and the metric approach to object matching," *Foundations of Computational Mathematics*, vol. 11, no. 4, pp. 417–487, 2011.

[18] V. Herrmann. (2021, Jul.) Wasserstein GAN and the Kantorovich-Rubinstein Duality. [Online]. Available: https://vincentherrmann.github.io/blog/wasserstein/

[19] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of computer computations*. Springer, 1972, pp. 85–103.

[20] A. Mallasto and A. Feragen, "Learning from uncertain curves: The 2-wasserstein metric for gaussian processes," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 5665–5674.

[21] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[22] L. R. Goldberg, "The development of markers for the Big-Five factor structure." *Psychological assessment*, vol. 4, no. 1, p. 26, 1992.

[23] G. Yu, "Variance stabilizing transformations of Poisson, binomial and negative binomial distributions," *Statistics & Probability Letters*, vol. 79, no. 14, pp. 1621–1629, 2009.