**Machine Learning (HWS24)**

# Assignment 3: Singular Value Decomposition

Nursultan Mamatov, nmamatov, 1983726
Cagan Yigit Deliktas, cdelikta, 1979012

November 17, 2024

## 1 Task 1: Intuition on SVD

### 1.1 Part a

We were able to obtain the rank (number of linearly independent rows), singular vectors, and their corresponding values for the first four matrices without using NumPy. We proceeded with a trial-and-error strategy and developed a method, explained below, which worked for the first four matrices.

| Matrix | Rank | Singular Values | Left/Right Singular Vectors |
|---|---|---|---|
| $M_1$ | 1 | 1 non-zero ($\sigma = 3$) | $u = \frac{1}{\sqrt{3}} \cdot [1, 1, 1, 0, 0]$<br>$v^T = \frac{1}{\sqrt{3}} \cdot [1, 1, 1, 0, 0]$ |
| $M_2$ | 1 | 1 non-zero ($\sigma = \sqrt{27}$) | $u = \frac{1}{\sqrt{3}} \cdot [0, 1, 1, 1, 0]$<br>$v^T = \frac{1}{3} \cdot [0, 2, 1, 2, 0]$ |
| $M_3$ | 1 | 1 non-zero ($\sigma = \sqrt{12}$) | $u = \frac{1}{2} \cdot [0, 1, 1, 1, 1]$<br>$v^T = \frac{1}{\sqrt{3}} \cdot [0, 1, 1, 1]$ |
| $M_4$ | 2 | 2 non-zero ($\sigma_1 = \sqrt{9}$, $\sigma_2 = \sqrt{4}$) | $u_1 = \frac{1}{\sqrt{3}} \cdot [1, 1, 1, 0, 0]$<br>$u_2 = \frac{1}{\sqrt{2}} \cdot [0, 0, 0, 1, 1]$<br>$V_1^T = \frac{1}{\sqrt{3}} \cdot [1, 1, 1, 0, 0]$<br>$V_2^T = \frac{1}{\sqrt{2}} \cdot [0, 0, 0, 1, 1]$ |

Table 1: Summary of matrix ranks, singular values, left and right singular vectors.

**Matrices Where Manual Decomposition Worked (M1, M2, M3, M4):**

- **Clear Patterns or Block Structures:** These matrices had simple or clearly separable structures, allowing us to manually identify distinct directions (singular vectors) corresponding to the non-zero rows or columns.
  - (M1, M2, M3, M4): We considered the number of unique and nonzero rows and columns as the right and left singular vectors, respectively. We then normalized them by their norms.

**Matrices Where Manual Decomposition Did Not Work (M5, M6):**

- **More Complex, Irregular Structures:** These matrices had more mixed or complicated patterns, which made it difficult to manually identify the singular vectors or to break the matrix down into simple rank-1 components.
  - M5: The non-zero entries were spread out and didn't follow a simple, separable pattern.
  - M6: Had mixed rows with irregular non-zero entries, which made it impossible to visually decompose it into clear singular vectors.

**Reconstruction of the Matrices with the Outer Product:**

$$\hat{M} = \sum_{i=1}^{r} \sigma_i \cdot \mathbf{u}_i \otimes \mathbf{v}_i^T$$

We used the formula above to reconstruct the matrices using the left and right singular vectors along with their corresponding values. It turns out that we were able to recover the original matrices for the first four.

## 1.2 Part b

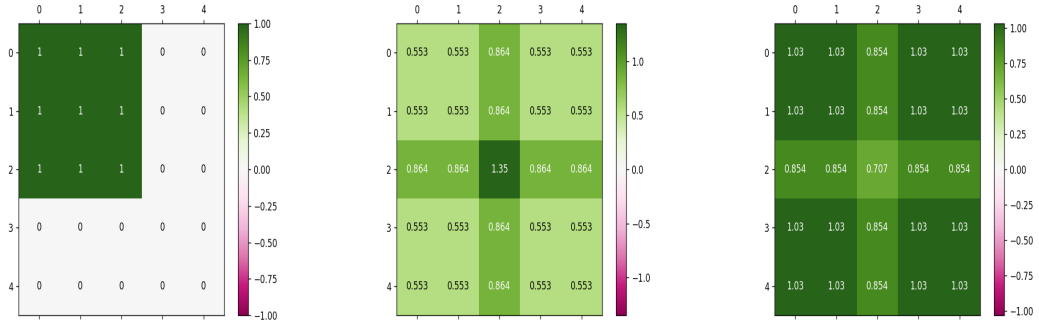
The results from NumPy's SVD computation largely correspond to our inferred compact SVD structure for each matrix (M1-4). Since we could not calculate the SVD of matrices 5 and 6 without using NumPy, we relied solely on NumPy for these two.

**Matrices M1, M2, M3:** We were able to reconstruct the original matrices $M_1$, $M_2$, and $M_3$ since they are all rank-1 matrices. Thus, all the information was captured with the rank-1 approximation of the SVD.

### Matrices M4, M5, M6 (Rank-2 or Rank-3 - More Complex)

- M4: This matrix has two distinct row and column patterns. The rank-1 approximation captures only one of these patterns, resulting in a loss of information since the second pattern is ignored. This leads to an incomplete representation of the original structure. (Figure 1a).
- M5: This matrix has three distinct row and column patterns, making it rank-3. A rank-1 approximation captures only the largest pattern, losing the complexity of the remaining two patterns, leading to a poor approximation. (Figure 1b).

- M6: This matrix has a rank of 2 but is complex, with intricate relationships between rows and columns. A rank-1 approximation misses important details, making it an oversimplified representation of the original matrix. (Figure 1c).




(a) M4 Reconstructed (b) M5 Reconstructed (c) M6 Reconstructed

Figure 1: M4-M6 Reconstructed

### 1.3 Pard d

Matrix M6 has a rank of 2, which implies it has 2 **non-zero singular values** that **meaningfully contribute** to the matrix structure. NumPy returns 5 singular values, with the last three being extremely close to zero: $9.95090019 \times 10^{-17}$, $2.18529703 \times 10^{-17}$, and $5.31822283 \times 10^{-50}$. These values are so small that they can be considered **numerically negligible**. Thus, we can treat these very small singular values as zero (we rounded them to 5 decimal places).

**Singular values given by NumPy (Rounded):**

$$= \begin{bmatrix} 4.82843 & 0.82843 & 0. & 0. & 0. \end{bmatrix}$$

**Singular values given by NumPy (Original):**

$$= \begin{bmatrix} 4.82 & 8.28 \times 10^{-1} & 9.95 \times 10^{-17} & 2.18 \times 10^{-17} & 5.31 \times 10^{-50} \end{bmatrix}$$

## 2 Task 2: The SVD on Weather Data

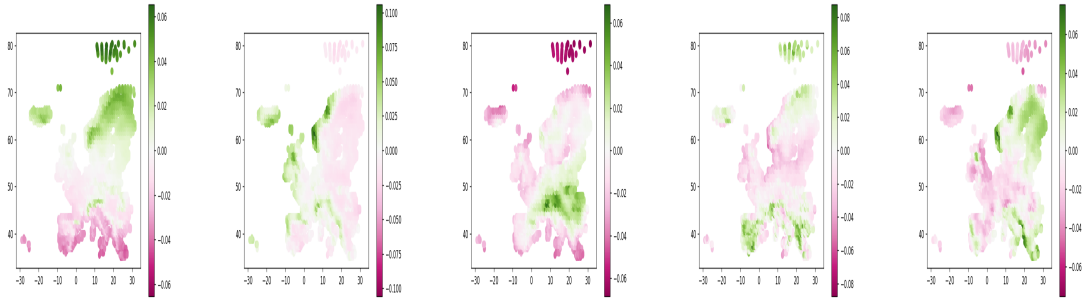
### 2.1 Part a: Normalize the data

We applied z-score normalization since temperature and rainfall measurements differ widely in their scales and units, normalization helps make each variable comparable by scaling them equally.

## 2.2 Part b: Compute the SVD

We computed the SVD of the normalized data using `numpy.linalg.svd()`. It returned the left and right singular vectors, along with 48 non-zero singular values, which means the rank of the matrix is 48. We also verified this using `numpy.linalg.matrix_rank()`.

## 2.3 Part c: Plot each of the first 5 columns of U



(a) 1 (b) 1-2 (c) 1-3 (d) 1-4 (e) 1-5

Figure 2: First five columns of U

The results in Figure 2 reveals Europe's climate structure in a progressive way. The first components capture broad, intuitive climate patterns, such as north-south or maritime-continental gradients, reflecting dominant, large-scale trends. Later components, like the fourth and fifth, bring out finer, localized climate features and subtle transitions that are less immediately visible. For example, in 2a, green areas (positive first singular vector values) might represent the climate type of Northern Europe, where winters are cold and summers are mild, whereas pink areas (negative first singular vector values) could represent the climate type of Southern Europe, where winters are mild and summers are hot.

## 2.4 Part d

The observation suggests that different pairs of columns in U reveal distinct patterns in climate data related to latitude:

- Early Pairs (e.g, U[:,0] vs U[:,1]): These show a clear north-south (latitude) gradient, with blue on the left (south) and red on the right (north). This suggests that the first component of U strongly reflects broad climate differences by latitude.
- Later Pairs (e.g., U[:,1] vs U[:,2]): The gradient reverses here, with red shifting to the left and blue to the right, indicating that these later components represent other climate patterns that don't follow a simple north-south pattern.

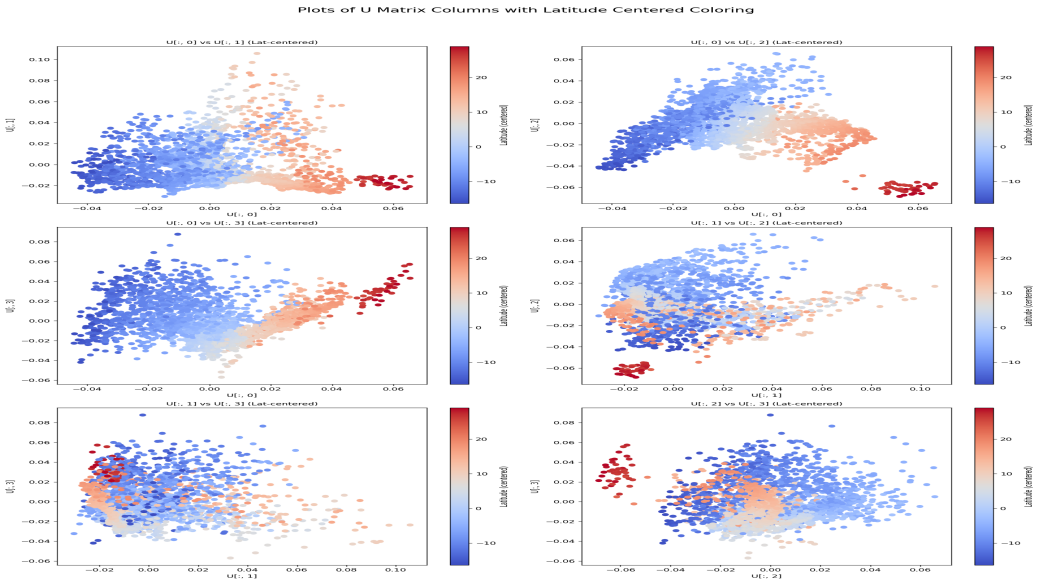



Figure 3: Plots of U Matrix Columns with Latitude Centered Coloring

For the second part of the code, which uses longitude-centered coloring, here's what the observations likely mean:

- Mostly Yellow Points: Since yellow represents positive values, this suggests that many of the points in the data have a positive deviation from the mean longitude. This likely reflects areas in the eastern part of Europe.
- Green and Dark Areas: The green color indicates values close to the mean longitude (0 deviation), while dark points represent negative values, showing areas west of the average longitude.
- Lower values of $u[:, 1]$ correspond to higher longitudes (further east, represented by yellow in the legend). As the values of $u[:, 1]$ increase, the locations correspond to lower longitudes (further west, represented by green-blue in the legend).

## 2.5 Part e

- **Guttman–Kaiser Criterion:** The criterion suggests retaining all singular values greater than 1, which, in this case, means using the first 37 singular values.
- **90% of Squared Frobenius Norm:** According to this method, only the first singular value captures 90% of the data's total variance, so it suggests using a rank of 1.
- **Scree Test:** By examining the scree plot in Figure 4, we observe that the slope starts to flatten after the sixth singular value, indicating that six components could be a good choice.

- **Entropy-Based Method:** The entropy of the singular values is calculated as 0.2752, with the method recommending a rank of 1.
- **Random Flipping of Signs:** This method, which evaluates the stability of the rank by flipping signs in the data, suggests a rank of 7 for a stable representation.

We might choose a rank of 6 or 7 as a balanced approach. This rank captures a substantial amount of structure without overfitting, aligning with both the Scree Test and Random Flipping's recommendations for stability. However, if we prioritize a very compact representation, we could consider a rank of 1, as suggested by the entropy-based and Frobenius norm methods, though this would likely oversimplify the data. Ultimately, we can opt for $k = 1$, as it captures enough information from the data while adhering to Occam's razor.

## 2.6 Part f

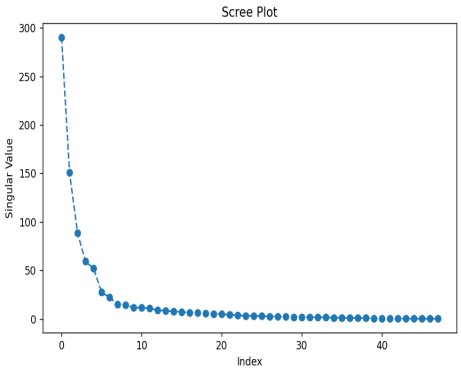



Figure 4: Scree test method

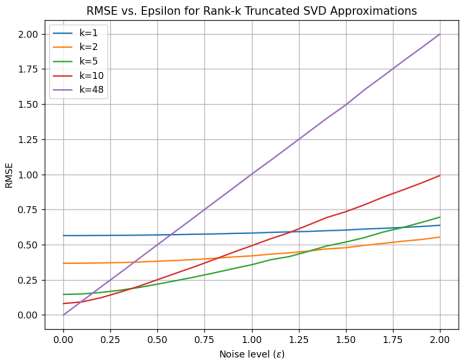



Figure 5: RMSE vs. Epsilon

**Optimal Rank Selection:** According to the results in Figure 5, for noisy data, choosing a lower rank (e.g., k=2) provides a more stable and robust approximation, as it captures essential structure while minimizing sensitivity to noise.
**Model Complexity vs. Noise Sensitivity:** Higher ranks (like k=10 or k=48) capture more detail in the data, which is beneficial for clean data but results in higher RMSE under noise due to their sensitivity to small, noisy features.
**Ultimate Choice of k:** Based on the results, k=2 appears to be an ideal choice for this data under increasing noise, as it consistently minimizes RMSE across different noise levels.

# 3 Task 3: SVD and Clustering

## 3.1 Part a

Clusters represent different climate zones and types across Europe based on the temperature and rainfall data. Longitude is on the X axis, and latitude is on the y axis (Figure 6).

**Dark Green:** The dark green parts represent the northeastern Europe where the weather is cold and rainfall amount is high. These areas might generally experience long winters, and cool summers.

**Light Blue:** It represents the coastal areas located in parts of central and western Europe. Average temperature might be a bit higher compared to dark green area.

**Dark Blue:** This area likely experiences a continental climate, being located farther from the coast. Temperature variation here could be greater, with colder winters and warmer summers with low rainfall.

**Light Green and Pink:** This area likely has a Mediterranean climate, with hot summers and mild, wet, rainy winters.

## 3.2 Part b

**Separation** The first singular vector explains a larger portion of the variance in the data, as data points extend further along the x-axis than along the y-axis. Clusters appear to be well separated from each other, but at the borders, there is some overlap (mixed colors) between clusters, suggesting gradual transitions between climate zones rather than sharply distinct types. It seems that clusters 1 to 4 can be separated using only the first left singular vector (which provides a vertical separation), while the second left singular vector is needed to identify cluster 0 (providing a horizontal separation) as we can see in Figure 7.

**Outliers** Some data points from Cluster 0 (purple) with high values along the second left singular vector may be considered outliers, as they extend well above the main group. A similar pattern can be observed in Cluster 3 (light green), where some points extend far to the right, beyond the center of the cluster as shown in Figure 7. These points might represent areas with unique climatic conditions, distinct from the majority in the dataset.

## 3.3 Part c

Z scores can be found:

$$\mathbf{Z} = \mathbf{U_k} \cdot \mathbf{S_k}$$

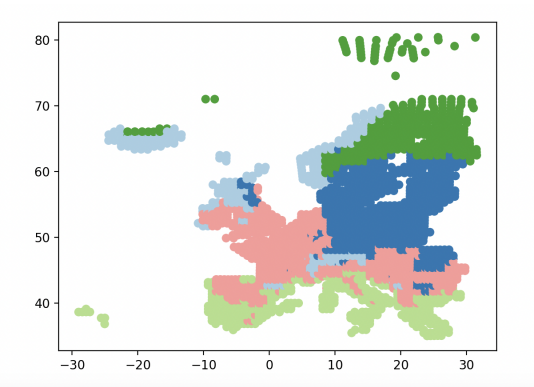



Figure 6: Longitude vs Latitude Colored with Clusters

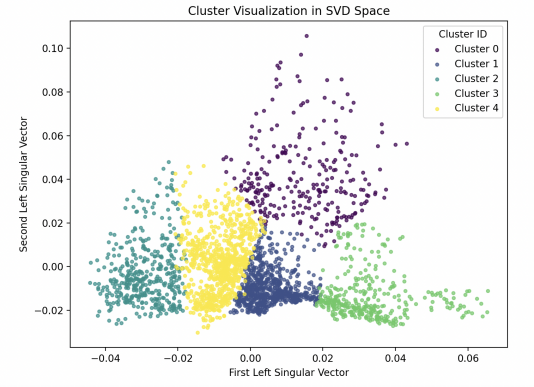


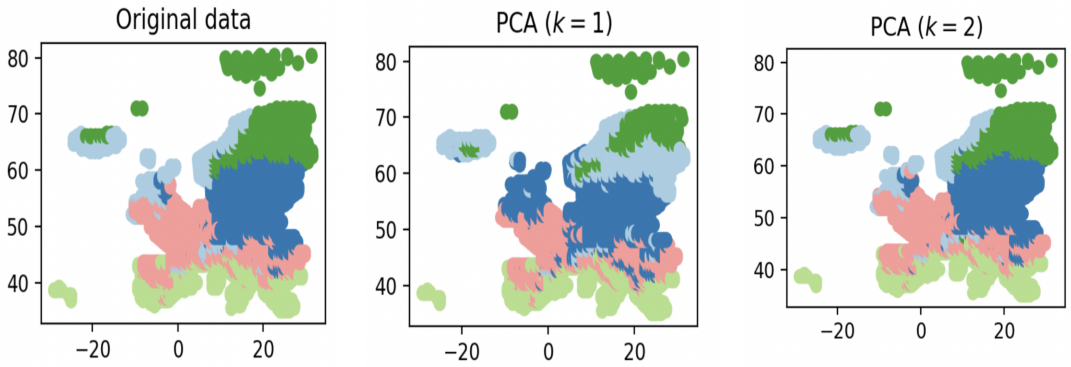

Figure 7: First vs Second Left Singular Vector Colored with Clusters

This represents the matrix multiplication of the first $k$ left singular vectors and the first $k$ singular values from $\mathbf{s}$.

**PCA with k=1:** We can clearly see that even with just the first principal component, we can almost capture every detail of the original data (Figure 8a), as shown in Figure 8b. There are only subtle differences in the reconstructed data.

**PCA with k=2:** This time we used the second component too, the reconstructed data seems exactly like the original data as we can see in Figure 8c.

**Power of PCA:** Since the first principal component is the most significant, explaining the largest portion of the variation in the data, it helps us reconstruct nearly the same data. Even reducing to just one or two components (i.e., $k = 1$ or $k = 2$) still allows the clustering algorithm to differentiate the clusters in a similar way as the full dataset.




(a) Original Data (b) PCA with k=1 (c) PCA with k=2

Figure 8: Original Data and Data Constructed with PCA

## References

[1] Gemulla, R. (2024). *IE 675b: Machine Learning* [Lecture slides]. University of Mannheim.

[2] Murphy, K. P. (2022). *Probabilistic machine learning: An introduction*. The MIT Press. https://probml.github.io/pml-book/book1.html

## Ehrenwörtliche Erklärung

Ich versichere, dass ich die beiliegende Bachelor-, Master-, Seminar-, oder Projektarbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Quellen und in der untenstehenden Tabelle angegebenen Hilfsmittel angefertigt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

**Declaration of Used AI Tools**

| Tool | Purpose | Where? | Useful? |
|---|---|---|---|
| ChatGPT | Rephrasing | Throughout | + |
| ChatGPT | Reinforcing mathematical understanding | Q1 a, Q2 e | + |

Unterschrift
Mannheim, den 17. November 2024