Dissimilarity Plots: A Visual Exploration Tool for Partitional Clustering

Michael Hahsler\(^1\) & Kurt Hornik\(^2\)

\(^1\) Intelligent Data Analysis Group, Southern Methodist University
\(^2\) Vienna University of Economics and Business

Celebrating the 20th Anniversary of JCGS
42\(^{\text{th}}\) Symposium on the Interface
June 1–3, 2011
Table of Contents

1 Motivation

2 Visualization Techniques for Partitions

3 Seriation

4 Dissimilarity Plots

5 Examples
Assessment of Cluster Quality

Clustering assigns objects to groups (clusters) so that objects from the same cluster are more similar to each other than to objects from other clusters.

Assess the quality of a cluster solution

- Typically judged by intra and inter-cluster similarities
- Visualization for judging the quality of a clustering and to explore the cluster structure
Dendrograms (Hartigan, 1967) for hierarchical clustering:

Restriction

Dendrograms are only possible for hierarchical/nested clusterings.
Table of Contents

1 Motivation

2 Visualization Techniques for Partitions

3 Seriation

4 Dissimilarity Plots

5 Examples
Projection-based Visualization

Project objects into 2-dimensional space with dimensionality reduction techniques (e.g., PCA, MDS; Pison et al. (1999)).

Problems with dimensionality (figure to the right: MDS/32-dimensional data)
Plot Quality Metrics

Visualize metrics calculated from inter and intra-cluster similarities to judge cluster quality. For example, **silhouette width** (Kaufman and Rousseeuw, 1990).

→ Only a diagnostic tool for cluster quality
Other Visualization Methods

Several other visualization methods (e.g., based on self-organizing maps and neighborhood graphs, shadow plots, shadow-stars, stripes plots) are reviewed and introduced in Leisch (2008, 2010).

- Typically hide structure within clusters or
- are limited by the number of clusters and dimensionality of data.
Dissimilarity Matrix Shading and CLUSION

Each cell of the (dissimilarity) matrix is represented by a gray value (Sneath and Sokal, 1973; Ling, 1973; Gale et al., 1984). Initially matrix shading was used with hierarchical clustering → heatmaps.

For graph-based partitional clustering: CLUSION (Strehl and Ghosh, 2003). Uses coarse seriation such that “good” clusters from blocks around the main diagonal.

CLUSION allows to judge cluster quality but does not reveal the structure of the data

→ Dissimilarity plots

Improve matrix shading/CLUSION with (near) optimal placement of clusters and objects within clusters using seriation
Part of **combinatorial data analysis** (Arabie and Hubert, 1996)

- **Aim:** arrange objects in a linear order given available data and some loss function in order to reveal structural information.
- **Problem:** Requires to solve a discrete optimization problem → solution space grows by the order of $O(n!)$

**Techniques:**

1. Partial enumeration methods (currently solve problems with $n \leq 40$)
   - dynamic programming (Hubert *et al.*, 1987)
   - branch-and-bound (Brusco and Stahl, 2005)

2. Heuristics for larger problems
Seriation II

- Set of $n$ objects $\mathcal{O} = \{O_1, O_2, \ldots, O_n\}$
- Symmetric dissimilarity matrix $\mathbf{D} = (d_{ij})$, where $d_{ij}$ for $1 \leq i, j \leq n$ represents the dissimilarity between $O_i$ and $O_j$, and $d_{ii} = 0$ for all $i$.
- Permutation function $\Psi$ reorders the objects in $\mathbf{D}$ by simultaneously permuting rows and columns.
- A loss function $L$ to evaluate a given permutation.

Optimization problem

$$\Psi^* = \arg\min_{\Psi} L(\Psi(\mathbf{D}))$$
Perfect anti-Robinson matrix (Robinson, 1951): A symmetric matrix where the values in all rows and columns only increase when moving away from the main diagonal. Gradient conditions (Hubert et al., 1987):

within rows: \( d_{ik} \leq d_{ij} \) for \( 1 \leq i < k < j \leq n \);
within columns: \( d_{kj} \leq d_{ij} \) for \( 1 \leq i < k < j \leq n \).

The closer objects are together in the order of the matrix, the higher their similarity.

Note: Most matrices can only be brought into a near anti-Robinson form.
Column/Row Gradient Measures II

Loss measure (quantifies the divergence from anti-Robinson form):

\[ L(D) = \sum_{i<k<j} f(d_{ik}, d_{ij}) + \sum_{i<k<j} f(d_{kj}, d_{ij}) \]

where \( f(\cdot, \cdot) \) is a function which defines how a violation or satisfaction of a gradient condition for an object triple \((O_i, O_k, \text{ and } O_j)\) is counted. Raw number of violations minus satisfactions:

\[
\begin{align*}
    f(z, y) &= \text{sign}(y - z) = \\
    &= \begin{cases} 
        -1 & \text{if } z > y; \\
        0 & \text{if } z = y; \\
        +1 & \text{if } z < y.
    \end{cases}
\end{align*}
\]

Weight each satisfaction or violation by its magnitude (absolute difference between the values):

\[
    f(z, y) = |y - z| \text{sign}(y - z) = y - z
\]
An even simpler loss function can be created in the same way as the gradient measures above by concentrating on violations only.

\[
L(D) = \sum_{i<k<j} f(d_{ik}, d_{ij}) + \sum_{i<k<j} f(d_{kj}, d_{ij})
\]

To only count the violations we use

\[
f(z, y) = I(z, y) = \begin{cases} 
1 & \text{if } z < y \text{ and } \\
0 & \text{otherwise.}
\end{cases}
\]

\(I(\cdot)\) is an indicator function returning 1 only for violations. Chen (2002) also introduced a weighted versions of this loss function by using the absolute deviations as weights:

\[
f(z, y) = |y - z| I(z, y)
\]
Hamiltonian Path Length

- **D** is seen as a finite weighted graph \( G = (\Omega, E) \) with \( \Omega = \{O_1, O_2, \ldots, O_n\} \) and the weight \( w_{ij} \) for edge \( e_{ij} \in E \) represents \( d_{ij} \).

- An order \( \Psi \) can be seen as a **Hamiltonian path** through the graph.

- Minimizing the path length results in a seriation optimal with respect to dissimilarities between neighboring objects (Hubert, 1974; Caraux and Pinloche, 2005).

**Loss function:**

\[
L(D) = \sum_{i=1}^{n-1} d_{i,i+1}
\]

This optimization problem is related to the **traveling salesperson problem** (Gutin and Punnen, 2002) for which good solvers and efficient heuristics exist.
Table of Contents

1 Motivation

2 Visualization Techniques for Partitions

3 Seriation

4 Dissimilarity Plots

5 Examples
Creating Dissimilarity Plots

1. **Split** $D$ into clusters using the assignment function $\Gamma$ provided by the partitional clustering algorithm.

2. **Arrange objects:** Use $\Psi_1, \ldots, \Psi_k$ to show micro-structure.

3. **Arrange clusters:** $\Psi_c$ places more similar clusters together (macro-structure).
Arrange Clusters

Find $\Psi_c$ based on inter-cluster dissimilarity matrix $D_c$ which aggregates dissimilarities between all pairs of clusters given dissimilarities between all elements of the clusters in $D$.

**Hierarchical clustering:** dissimilarities between two sets of objects $\mathcal{X}$ and $\mathcal{Y}$

- complete-link: $d_c(\mathcal{X}, \mathcal{Y}) = \max\{d(x, y) : x \in \mathcal{X}, y \in \mathcal{Y}\}$
- single-link: $d_s(\mathcal{X}, \mathcal{Y}) = \min\{d(x, y) : x \in \mathcal{X}, y \in \mathcal{Y}\}$
- average-link: $d_a(\mathcal{X}, \mathcal{Y}) = \frac{1}{|\mathcal{X}| \cdot |\mathcal{Y}|} \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} d(x, y)$

**Set theory: Hausdorff metric** (Hausdorff, 2001)

$$d_H(\mathcal{X}, \mathcal{Y}) = \max\{\sup_{x \in \mathcal{X}} \inf_{y \in \mathcal{Y}} d(x, y), \sup_{y \in \mathcal{Y}} \inf_{x \in \mathcal{X}} d(x, y)\}$$

The Hausdorff metric pairs up each element from one set with the most similar element from the other set and then finds the largest dissimilarity in such element pairs.
Table of Contents

1 Motivation

2 Visualization Techniques for Partitions

3 Seriation

4 Dissimilarity Plots

5 Examples
We use the column/row gradient measure as the loss function for seriation.

1. Placement (seriation) of clusters: Average-link, row/column gradient measure using branch-and-bound to find the optimal solution

2. Placement (seriation) of objects within each cluster: row/column gradient measure uses a simulated annealing heuristic

Seriation algorithms are provided by Brusco and Stahl (2005) and are available in the R extension package seriation (Hahsler et al., 2008).
Ruspini data set (Ruspini, 1970) with 75 points in two-dimensional space with four clearly distinguishable groups.
Euclidean distances and $k$-medoids clustering algorithm (partitioning around medoids (PAM) (Kaufman and Rousseeuw, 1990)) to produce a partition with $k = 4$
Easily Distinguishable Groups II

Coarse seriation

Dissimilarity plot
Mis-specification of the Number of Clusters I

Ruspini data set with 4 groups.

Coarse seriation, $k = 3$

Dissimilarity plot, $k = 3$
Mis-specification of the Number of Clusters II

Coarse seriation, $k = 7$

Dissimilarity plot, $k = 7$
Random data for 250 objects in $\mathbb{R}^5$: $X_1, X_2, \ldots, X_5 \sim N(0, 1)$

Euclidean distance and PAM with $k = 10$

These two components explain 45.85% of the point variability.

Average silhouette width: 0.13

Silhouette width $s_i$

-0.2 0.0 0.2 0.4 0.6 0.8 1.0

10 clusters $C_j$

<table>
<thead>
<tr>
<th>$j$</th>
<th>$n_j$</th>
<th>$\text{ave}_{c \in C_j} s_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>27</td>
<td>0.08</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>0.11</td>
</tr>
<tr>
<td>3</td>
<td>28</td>
<td>0.29</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>0.12</td>
</tr>
<tr>
<td>5</td>
<td>21</td>
<td>0.01</td>
</tr>
<tr>
<td>6</td>
<td>26</td>
<td>0.07</td>
</tr>
<tr>
<td>7</td>
<td>22</td>
<td>0.15</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
<td>0.11</td>
</tr>
<tr>
<td>9</td>
<td>19</td>
<td>0.25</td>
</tr>
<tr>
<td>10</td>
<td>22</td>
<td>0.06</td>
</tr>
</tbody>
</table>

$n = 250$
No Structure II

Coarse seriation

Dissimilarity plot
High-dimensional Data I

**Votes** data set (UCI Repository of Machine Learning Databases (Blake and Merz, 1998)). Votes for each of the U.S. House of Representatives congressmen on the 16 key votes during the second session of 1984.

- **Coding**: 2 variables per vote (in favor/againsts)
  - Each congressman is represented by a vector in \( \{0, 1\}^{32} \)

- **Dissimilarity measure**: Jaccard dissimilarity (Sneath and Sokal, 1973) between congressmen. Let \( S_i \) and \( S_j \) be the sets of votes two congressmen voted for in favor. Then the Jaccard dissimilarity

\[
d_{ij} = 1 - \frac{|S_i \cap S_j|}{|S_i \cup S_j|}.
\]

- **Cluster algorithm**: PAM with \( k = 12 \)
  (the first bump of average silhouette for \( k = 2, 3, \ldots, 30 \))
These two components explain 40.59% of the point variability.

Average silhouette width: 0.14

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Size</th>
<th>Silhouette Width</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>35</td>
<td>0.34</td>
</tr>
<tr>
<td>2</td>
<td>48</td>
<td>0.15</td>
</tr>
<tr>
<td>3</td>
<td>45</td>
<td>-0.01</td>
</tr>
<tr>
<td>4</td>
<td>36</td>
<td>0.08</td>
</tr>
<tr>
<td>5</td>
<td>38</td>
<td>0.05</td>
</tr>
<tr>
<td>6</td>
<td>32</td>
<td>0.08</td>
</tr>
<tr>
<td>7</td>
<td>43</td>
<td>0.22</td>
</tr>
<tr>
<td>8</td>
<td>37</td>
<td>0.27</td>
</tr>
<tr>
<td>9</td>
<td>18</td>
<td>0.09</td>
</tr>
<tr>
<td>10</td>
<td>52</td>
<td>0.07</td>
</tr>
<tr>
<td>11</td>
<td>20</td>
<td>0.33</td>
</tr>
<tr>
<td>12</td>
<td>31</td>
<td>0.05</td>
</tr>
</tbody>
</table>
High-dimensional Data III

Coarse seriation, threshold=0.7

Dissimilarity plot, threshold=0.7
Cluster composition (clusters reordered by dissimilarity plot)
Conclusion

Advantages of dissimilarity plots

- Scales well with dimensionality of data (visualizes dissimilarities)
- Shows cluster quality (block structure)
- Visual analysis of cluster structure (placement of clusters)
- Visual analysis of micro-structure (placement of objects)
- Makes misspecification of number of clusters apparent (placement of clusters/objects)

Enhancements for large number of objects/clusters

- **Object sampling:** Reduces the size of the dissimilarity matrix, however, details are sacrificed.
- **Image downsampling:** pixel skipping, pixel averaging, 2D discrete wavelet transformation
- **Interactive plot:** Plot with only average between-cluster similarities and then separate plot for each cluster (inter-cluster structures).
Further Reading


Code

Dissimilarity plot and seriation methods are implemented in the R extension package seriation (Hahsler et al., 2008) and are freely available via the Comprehensive R Archive Network at

[http://CRAN.R-project.org](http://CRAN.R-project.org)
References


