A New Flexible Dendrogram Seriation Algorithm for Data Visualisation

Denise Earle and Catherine B. Hurley [1]
National University of Ireland Maynooth
Maynooth, Co. Kildare, Ireland.
E-mails: denise.earle83@gmail.com; catherine.hurley@nuim.ie

ABSTRACT

Seriation is a data analytic tool for obtaining a permutation of a set of objects with the goal of revealing structural information within the set of objects. Seriating variables, cases or categories generally improves visualisations of statistical data, for example, by revealing hidden patterns in data or by making large datasets easier to understand. In this paper we present a new algorithm for seriation based on dendrograms. Dendrogram seriation algorithms rearrange the nodes in a dendrogram in order to obtain a permutation of the leaves (i.e. objects) that optimises a given criterion. Our algorithm is more flexible than currently available seriation algorithms because it allows the user to either choose from a variety of seriation criteria or to input their own criteria. This choice of seriation criteria is an important feature because different criteria are suitable for different visualisation settings. Common seriation criteria include measurements of the path length through a set of objects and measurements of anti-Robinson form in a symmetric matrix. We propose new seriation criteria called lazy path length and banded anti-Robinson form, and demonstrate their effectiveness in a variety of visualisation settings.

1 Introduction

Statistical graphics are commonly constructed using the order of variables, cases or categories in which they are listed in the data. As many authors have demonstrated (see, for example, Friendly and Kwan 2003 and Hurley 2004) visualisations often benefit when the data is systematically reordered. Reordered visualisations are clearer, more easily interpreted and more informative.

The systematic reordering of data is referred to as *seriation*. Seriation aims to reveal structural information within a set of objects by finding a suitable permutation of those objects.

There are many different approaches to seriation, including Travelling Salesperson heuristics (see, for example, Lawler et al. 1985), simulated annealing algorithms (see, for example, Brusco et al. 2007) and dimension reduction techniques such as principal components analysis (see, for example, Friendly and Kwan 2003). This paper focusses on one branch of seriation algorithms called "dendrogram seriation" algorithms.

The order of the leaves in a dendrogram provides a permutation of a set of n objects. However, the leaf ordering is not unique. Each of the $n-1$ nodes in a dendrogram can be rearranged resulting in a total of 2^{n-1} possible permutations of the objects. "Dendrogram seriation" algorithms rearrange the nodes in a dendrogram in order to obtain a permutation of the leaves (i.e. objects) that optimises some seriation criterion. Dendrogram seriation simplifies the problem of seriating n objects by reducing the size of the search space from $n!$ possible permutations to 2^{n-1} possible permutations. It has the added benefit of providing a clustering of the n objects.

[1]Both authors supported by a Research Frontiers Grant from Science Foundation Ireland.

Gruvaeus and Wainer (1972) first developed dendrogram seriation methods with the goal of obtaining a unique ordering of objects from a hierarchical clustering. Since then, the methodolgy has been further developed by Degerman (1982), Gale et al. (1984), Eisen et al. (1998), Alon et al. (1999), Wishart (1999), Bar-Joseph et al. (2001), Morris et al. (2003), Forina et al. (2007), Tien et al. (2008) and Wu et al. (2010). Dendrogram seriation has also been successfully applied to many visualisation settings including scatterplot matrices, parallel coordinates plots (Hurley 2004) and heatmaps (see, for example, Gale et al. 1984 and Eisen et al. 1999).

Section 2 introduces a new flexible dendrogram seriation algorithm called DendSer (Earle 2010), which generalises that of Wishart (1999). The succeeding sections define various seriation cost functions. These are used together with DendSer in various applications.

2 DendSer algorithm

The DendSer algorithm provides a seriation of n objects as follows. First calculate a dissimilarity matrix and perform a hierarchical clustering. This results in a dendrogram Δ, and provides an initial permutation π of the n leaf objects. Let F denote a seriation criterion or cost function which measures the "goodness" of a permutation. Some suitable choices for this cost function will be presented in later sections. The goal of DendSer is to find which of the 2^{n-1} possible rearrangements of Δ produces a permutation minimising F.

Even though the search space has been reduced from $n!$ to 2^{n-1}, except for small n, exploring all 2^{n-1} possibilities is generally not feasible. However in the special case of a shortest-path cost function Bar-Joseph et al. (2001) developed an $O(n^3)$ algorithm which finds the optimal rearrangement of Δ.

DendSer examines each node N of Δ in turn, starting at the first node formed by the hierarchical clustering process and ending at the root node. At each node N, it evaluates whether a rearrangement of N results in a permutation that reduces F. Let $T(N;\Delta)$ denote the set of permutations of the n leaf objects offered by a possible rearrangement of N. DendSer compares F evaluated on the current permutation π to permutations in $T(N;\Delta)$. If any of the permutations in $T(N;\Delta)$ reduce F, this permutation is retained and Δ and therefore π are updated. One iteration is complete when the algorithm has examined all $n-1$ nodes. At this stage the algorithm moves to the next iteration, repeating the process of examining all $n-1$ nodes. The algorithm stops when a full iteration fails to improve the permutation of the leaves or the maximum number of iterations is reached.

The DendSer algorithm results in a rearrangement of the dendrogram Δ which provides an improved permutation π of the leaf objects. The flexibility of DendSer over existing seriation algorithms is that it is not limited to one cost function F and one method of node rearrangement.

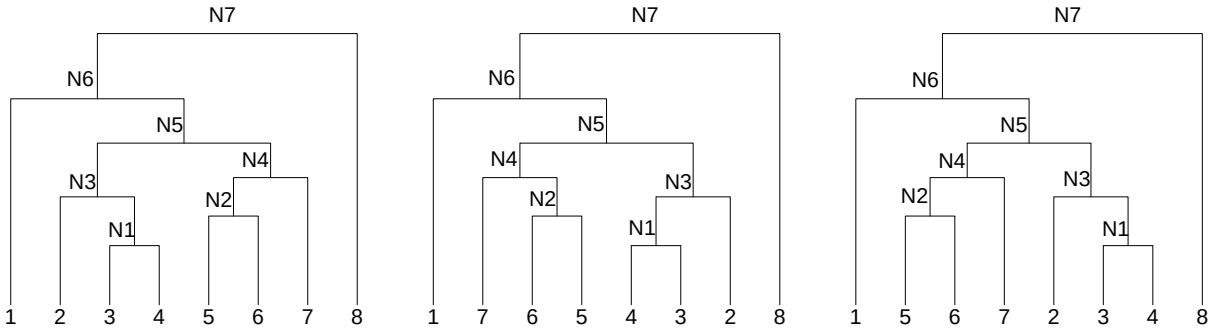
Next we describe various methods of node rearrangement. Succeeding sections present some choices of F that are suited to visualisation applications and give some examples.

At each visit to a node N, the DendSer algorithm evaluates various rearrangements of that node. These rearrangements are termed *node operations*. Node operations typically involve reflection or translation.

Definition 2.1. *The reflection of a node N in a dendrogram reverses the order of the leaves in N.*

Definition 2.2. *Let N_l and N_r be the left and right sub-nodes of a node N. The translation of N swaps the positions of N_l and N_r but does not change the order of the leaves in N_l and N_r.*

For example, Figure 1 illustrates the effect of reflection and translation on a dendrogram node.



(a) Δ: Initial dendrogram (b) Reflect N_5 (c) Translate N_5

Figure 1: The node N_5 in the dendrogram in (a) is reflected in the dendrogram in (b) and translated in the dendrogram in (c).

Note that reflecting or translating a node N does not change the hierarchy represented by N, only the order of the leaves in N.

For a dendrogram Δ and node N with left and right sub-nodes N_l and N_r, we define the following node operations.

$$\begin{aligned} R_0(N;\Delta) &= \{\text{permutation of leaves in } \Delta \text{ after reflecting a node } N\}. \\ T_0(N;\Delta) &= \{\text{permutation of leaves in } \Delta \text{ after translating a node } N\}. \\ C_0(N;\Delta) &= R_0(N;\Delta) \cup T_0(N;\Delta). \end{aligned}$$

The node operations R_0 and T_0 each offer a choice of one new permutation at each node N, while C_0 offers a choice of two new permutations (except in the trivial case where N has just two leaves). Previous dendrogram seriation algorithms used either node reflection (Wishart 1999, Morris et al. 2003), or node translation (Degerman 1982, Gale et al. 1984, Eisen et al. 1998 and Tien et al. 2008). However Earle (2010) demonstrates that dendrogram seriation using both node reflection and translation leads to improved results.

Gruvaeus and Wainer (1972) used node reflection, but on a node's sub-nodes rather than on a node itself. For this we use the notation R_1, where the subscript "1" refers to reflection at a depth of one. Define

$$R_1(N;\Delta) = R_0(N_l;\Delta) \cup R_0(N_r;\Delta) \cup R_0(N_l, N_r;\Delta),$$

where N_l and N_r are the left and right sub-nodes of N. $R_1(N;\Delta)$ is a set of (up to) three permutations of the leaves in Δ: one corresponding to the reflection of N_l, one corresponding to the reflection of N_r and one corresponding to the reflection of both N_l and N_r. Earle (2010) similarly defined the node operation T_1 which uses translation of the left and right sub-nodes. She also considered node operations which use reflections (translations) of the parent node combined with reflections (translations) of each of the child nodes.

The DendSer algorithm is not limited to a single, fixed choice of node operation. Indeed, the ideal choice depends on the cost function F. For example, as we see in Section 3 the leaf sorting algorithm requires the use of T_0. Earle (2010) found that the new node operation C_0 is generally effective and efficient in minimising various cost functions.

In the following sections we propose a number of cost functions and give some examples.

3 Leaf sorting

A number of authors (Degerman 1982, Gale et al. 1984, Eisen et al. 1998 and Tien et al. 2008) described the following "leaf sorting" algorithm. Consider a dendrogram where each leaf object has a weight. For each node N, compute the mean weight of the leaves for the left and right sub-nodes of N. Denote these weights by $\bar{w}_L$ and $\bar{w}_R$ respectively. If $\bar{w}_R < \bar{w}_L$ then N is translated, otherwise N remains unchanged. The end result of this "leaf sorting" algorithm is a rearrangement of the dendrogram, where the leaf weights generally increase as one reads from left to right.

It is easy to see (Earle 2010) that the leaf sorting algorithm is a special case of DendSer with one iteration, node operation T_0 and the leaf sort cost function (LS) defined as

$$\mathrm{LS}(\pi) = -\sum_{i=1}^{n} i w_{\pi(i)}.$$

Here π is a permutation of the n leaf objects, whose weights are w_i, for $1 \leq i \leq n$. Next we explore an application of leaf sorting.

Figure 2(a) shows the result of the `hclust` R function (R Development Core Team 2010), using average linkage on the (standardised) pottery data. This data from Tubb et al. (1980) contains nine chemical measurements on 45 pieces of pottery found at five locations. The clustering of the 45 pieces of pottery is based on the chemical measurements only.

We now use DendSer with the LS cost function to rearrange the dendrogram, where the leaf weights are the scores on the first principal component; the result appears in Figure 2(b). This should arrange the leaves and clusters in a systematic way and assist in the interpretation of the clusters found.

In Figure 2(a) three clusters are evident. In the rearranged dendrogram of Figure 2(b), the scores on the first principal component are shown in the associated barchart, where the bars are coloured by cluster. As the scores have a clear increasing pattern, the three clusters separate along the first principal component. This leads to a simple description of the chemical composition of the three clusters.

Furthermore, it turns out that the blue cluster contains the pottery pieces found at two locations in Hampshire, the green cluster contains the pottery pieces found at Gloucester and the orange cluster contains the pottery pieces found at two locations in Wales.

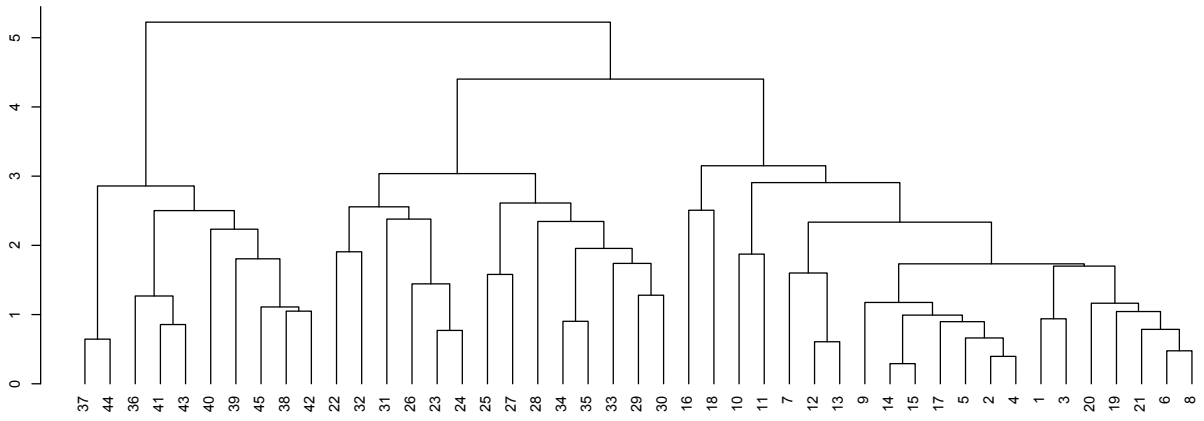
4 Path length and extensions

The shortest path problem is a variation of the Travelling Salesperson Problem (TSP). Given n cities, the goal of the TSP is to find the shortest tour that starts from a chosen city, visits each city once and returns to the starting city. For the shortest path problem, there is no return to the starting city.

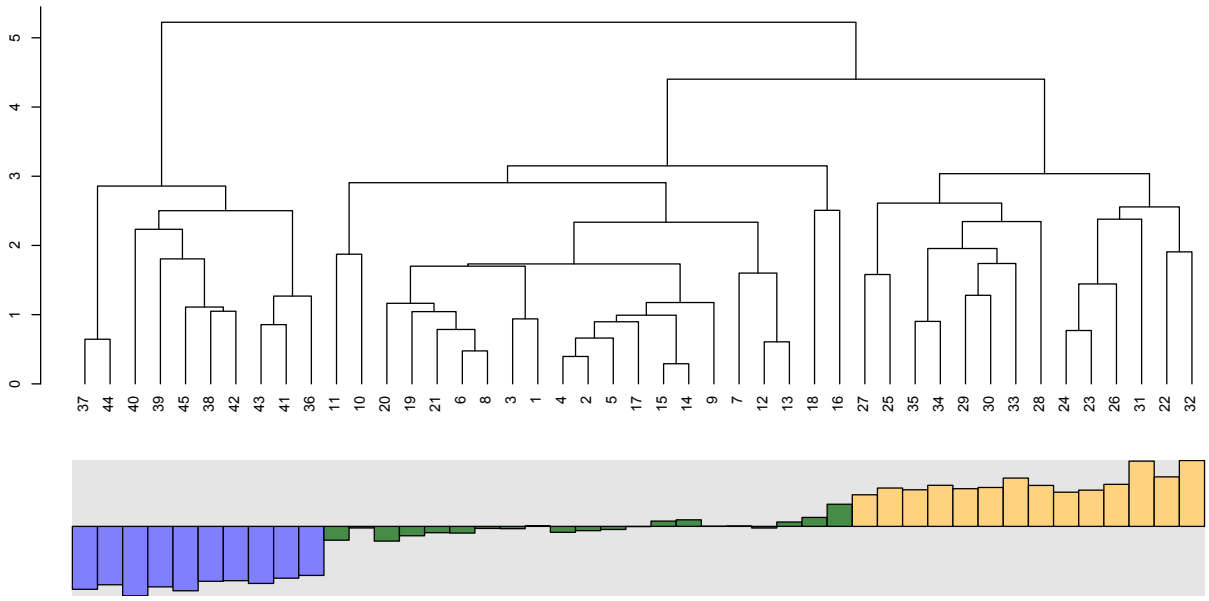
A permutation minimising the path length criterion aims to place similar objects adjacently, which makes minimising the path length of a permutation a suitable goal for seriation. Consider a set of n objects, where $d_{i,j}$ is the dissimilarity value between objects i and j. For a permutation π of the objects, the path length cost function is defined as:

$$\mathrm{PL}(\pi) = \sum_{i=1}^{n-1} d_{\pi(i),\pi(i+1)}.$$

The path length criterion has been successfully applied to a variety of statistical visualisations. Hurley (2004) described why the path length criterion is suitable when seriating variables in a par-



(a) Standard dendrogram



(b) Leaf sorted dendrogram

Figure 2: Hierarchical clustering of pottery data. The weights on the leaves in (b) are the scores on the first principal component. The lower barchart gives the weights, coloured by cluster.

allel coordinates plot, while Bar-Joseph et al. (2001) showed that minimising the path length of a permutation helps to reveal biological structure in heatmaps of gene expression data.

Earle (2010) proposed a new criterion called *lazy path length* which is a variation of the path length criterion. Given a set of objects with dissimilarities between them, the goal is to find a permutation of the objects that has a short path length and where dissimilarities between adjacent objects generally increase. The lazy path length cost function is defined as

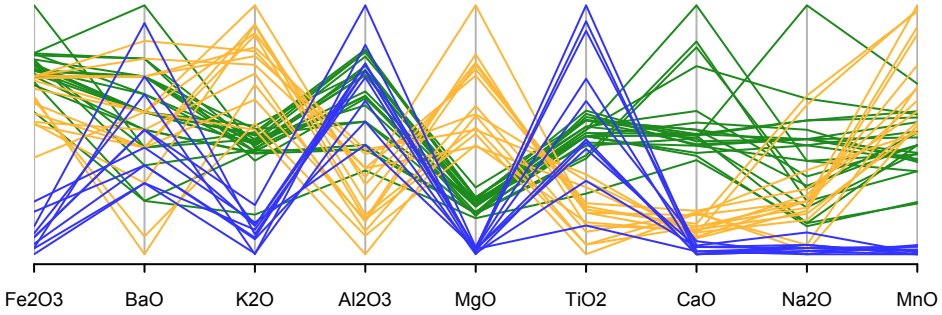
$$\mathrm{LPL}(\pi) = \sum_{i=1}^{n-1} (n-i) d_{\pi(i),\pi(i+1)}.$$

The LPL cost function is a weighted measure of the path length of a permutation, where dissimilarities at the beginning of the permutation are given more weight than dissimilarities near the end.

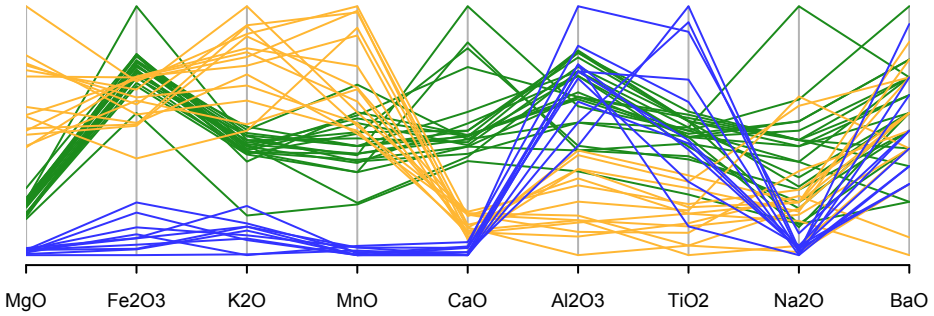
The DendSer minimisation procedure starts with a dendrogram Δ, and then attempts to find

which of the 2^{n-1} possible rearrangements of Δ produces a permutation minimising the PL or LPL cost function. DendSer is of course a greedy algorithm and is not guaranteed to find the overall minimum. However, for the PL cost function Bar-Joseph et al. (2001) developed an $O(n^3)$ algorithm which finds the optimal rearrangement of Δ. For both cost functions PL and LPL, Earle (2010) reports that the node operation C_0 is an effective choice, and this is used in the following example.

We return to the pottery data of the previous section and use LPL in conjunction with parallel coordinate plots (PCP) to explore the chemical composition of the pottery pieces from the three clusters (or equivalently regions) of Figure 2. In Figure 3(a) the variables are ordered arbitrarily and the lines



(a) Arbitrary ordering



(b) DendSer + LPL

Figure 3: The variables in the PCP in (a) are arbitrarily ordered. The variables in the PCP in (b) are ordered according to the permutation returned by using DendSer with the LPL cost function. The lines are coloured according to the three regions.

are coloured according to the three regions. The lines in this PCP are quite "zig-zaggy" and the panels show poor separation of the three regions.

For exploring the region differences, an informative permutation of the variables is one that conveys the differences between the regions. As a measure of separation between the clusters, for a panel showing variables i and j, we define

$$m_{i,j} = \sum_{k_1 \neq k_2} ||c_{k_1,ij} - c_{k_2,ij}||$$

where $c_{k,ij}$ is the centroid vector for the kth cluster for variables i and j, and $||\cdot||$ denotes Euclidean distance. Taking $d_{i,j} = m - m_{i,j}$ where $m = \max_{i \neq j} m_{i,j}$, we obtain Δ using average linkage and find a suitable permutation of the variables using DendSer and the LPL cost function.

Note that in this setting, the $d_{i,j}$ values measure similarity rather than dissimilarity between the cluster centroids. While this may seem counter-intuitive, we are simply using dendrogram seriation to

find a permutation minimising a cost function, in this instance LPL. The only requirement for the cost function is that $d_{i,j} = d_{j,i}$.

The PCP in Figure 3(b) shows the optimal permutation and contains panels that show better separation of the regions than does the PCP of Figure 3(a). This makes it easier to extract information about the differences between the chemical composition of the pottery pieces. Panels showing high separation of the regions appear at the beginning of the PCP and panels showing low separation are at the end. Given that people generally read from left to right, it follows that people are also likely to examine a PCP from left to right (or top to bottom, depending on the orientation of the PCP). Therefore, placing the most "interesting" panels at the beginning of the PCP allows the analyst to immediately see features of interest in the data.

5 Robinson form and extensions

Consider a symmetric matrix where the values in the matrix are non-increasing as one moves away from the diagonal. A matrix with this pattern is said to have "Robinson" form after the statistician W.S. Robinson, who first described this pattern in Robinson (1951). Similarly, a matrix where the values are non-decreasing as one moves away from the diagonal is said to have "anti-Robinson" form.

Definition 5.1. *A symmetric matrix $D = [d_{i,j}]$, for $1 \leq i, j \leq n$, has anti-Robinson form if $d_{i,k} \leq d_{i,j}$ and $d_{k,j} \leq d_{i,j}$, for $i < k < j$.*

Anti-Robinson (AR) form is a natural concept for seriation and visualisation. If a dissimilarity matrix has anti-Robinson form, then the smallest dissimilarity values are close to the main diagonal and the largest values are far away from the main diagonal. This means that objects with low dissimilarity are close together in the corresponding permutation and objects with high dissimilarity are far apart.

Many functions are available for measuring how close a symmetric matrix is to anti-Robinson form. See, for example, Hubert et al. (2006) and Chen (2002). In this paper we use the following cost function adapted from a measure described in Hubert et al. (2006):

$$\mathrm{ARc}(\pi) = \sum_{|i-j| \leq n-1} (n - |i-j|) d_{\pi(i),\pi(j)}.$$

Optimising AR form in a dissimilarity matrix aims to fit *every* element of the matrix into a specific pattern. However, this pattern is quite strict and may be too rigid for some dissimilarity matrices. *Banded anti-Robinson form* (Earle 2010) "relaxes" AR form and may be described as a hybrid of the path length and anti-Robinson criteria.

Definition 5.2. *A symmetric matrix $D = [d_{i,j}]$, for $1 \leq i, j \leq n$, has banded anti-Robinson form if for a band-width w with $w < n$:*

- *$d_{i,k} \leq d_{i,j}$ and $d_{k,j} \leq d_{i,j}$, for $i < k < j$ and $|i-j| \leq w$.*
- *$d_{i,j} \leq d_{i',j'}$, for $|i-j| \leq w$ and $|i'-j'| > w$.*

A matrix following banded AR form has small values close to the main diagonal, but only the values within a band of width w around the main diagonal satisfy AR form. The banded anti-Robinson cost function is defined as:

$$\mathrm{BAR}(\pi) = \sum_{|i-j| \leq w} (w + 1 - |i-j|) d_{\pi(i),\pi(j)}.$$

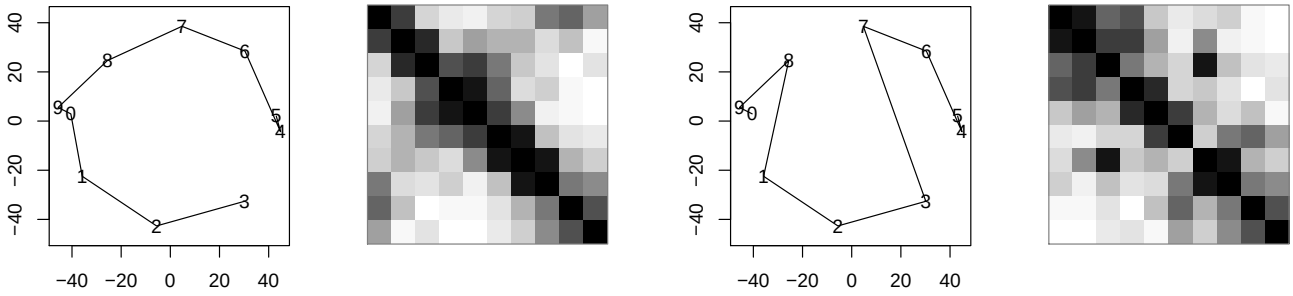
Note that when $w = 1$, the BAR cost function is identical to the PL cost function and when $w = n-1$, the BAR cost function is identical to the ARc cost function. The default choice of w is $n/5$, which we have found works well.

The DendSer minimisation procedure starts with a dendrogram Δ, and then attempts to find which of the 2^{n-1} possible rearrangements of Δ produces a permutation minimising the ARc or BAR cost function. Earle (2010) recommends the node operation T_0 for minimising ARc and C_0 for minimising BAR, and we follow these recommendations in the next example.

Rothkopf (1957) described an experiment, where inexperienced subjects listened to pairs of morse codes and then decided whether a pair of codes were identical. The data used in this example contain the results for the ten single digits, where the ij^{th} entry in the data is the percentage of subjects who said codes i and j were identical after hearing code i first and then code j.

The data is an asymmetric similarity matrix (denoted by S) because the response to hearing code i first and then code j may not be the same as the reponse to hearing the codes in reverse order. S is symmetrised by averaging the corresponding pairs of off-diagonal elements and is then converted into a dissimilarity matrix using the transformation $D = 100 - S$.

Ignoring the path through the points, the scatterplot in Figure 4(a) shows the two dimensional



(a) DendSer + BAR (b) DendSer + ARc

Figure 4: The path through the points in the scatterplot in (a) corresponds to the permutation of digits from using DendSer with BAR. The same permutation also orders the rows/columns in the corresponding heatmap of the dissimilarity matrix of the digits. In (b) the digits are ordered using DendSer with ARc.

classical multidimensional scaling of D. This plot visualises a two dimensional representation of the dissimilarities between the codes: similar codes are close together and dissimilar codes are far apart.

The ten codes form a circle and if the codes are ordered along this circle, then the corresponding dissimilarity matrix follows a "circumplex" pattern (see, for example, Wilkinson 2005, §16.5). A dissimilarity matrix follows a circumplex pattern if when moving away from the main diagonal in the matrix, the dissimilarities begin low, then increase to a point and then decrease becoming low again. The heatmap in Figure 4(a) shows the circumplex pattern in the dissimilarity matrix for the Morse code data, where black to white represents low to high dissimilarities.

Starting with an average linkage dendrogram of D, DendSer is used to obtain permutations with each of the cost functions BAR and ARc. These permutations are shown by the paths through the points in the scatterplots of Figures 4(a) and (b). The rows/columns in the heatmaps of the adjacent dissimilarity matrices are also ordered according to these permutations.

DendSer with ARc fails to recover the circular ordering of the morse codes for the following reason. If a dissimilarity matrix follows anti-Robinson form, then the values in this matrix increase when moving away from the main diagonal. However, in a circumplex dissimilarity matrix, the values increase and then decrease when moving away from the main diagonal. It is the region of low values

in the north-east and south-west corners of a circumplex dissimilarity matrix (see, for example, the heatmap in Figure 4(a)) that messes up the anti-Robinson pattern. DendSer with BAR recovers the circular ordering of the morse codes because BAR is unaffected by the region of low values in the north-east and south-west corners of a circumplex dissimilarity matrix.

The morse code data is one example showing that anti-Robinson form may be too rigid a structure for some dissimilarity matrices, whereas the more relaxed banded anti-Robinson form is more flexible in revealing patterns in dissimilarity matrices.

6 Concluding Remarks

DendSer is a flexible dendrogram seriation algorithm that allows the user to choose from many seriation criteria and node operations. This is in contrast to other dendrogram seriation algorithms, which focus on one criterion and one node operation only. The preceding sections gave some examples of seriation criteria; Hahsler et al. (2010) list some other possibilities. By combining the choice of both the node operation and seriation criterion, DendSer provides a general framework for performing many of the currently available dendrogram seriation techniques and some new ones besides.

Included in the choice of seriation criteria for DendSer are two new criteria called banded anti-Robinson form and lazy path length. These criteria were illustrated in the examples of the preceding sections and both have applications to a variety of visualisation settings. In particular, we believe that banded anti-Robinson form provides a convenient compromise between anti-Robinson form, which may enforce too *global* a structure, and path length, which may look for too *local* a structure.

While the applications presented here involve small numbers of objects n, we have used DendSer successfully with 10,000 objects, and its efficiency compares well with competitor algorithms. See Earle (2010) for detailed comparisons with competing algorithms (Hahsler et al. 2010) and more discussion of node operations, cost functions and extensive examples.

REFERENCES

Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D. & Levine, A.J. (1999), "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays", *Proc. Natl. Acad. Sci. USA*, 96, 6745-6750.

Bar-Joseph, Z., Gifford, D.K. & Jaakkola, T.S. (2001), "Fast optimal leaf ordering for hierarchical clustering", *Bioinformatics*, 17(1), 22-29.

Brusco, M.J., Kohn, H. & Stahl, S. (2007), "Heuristic implementation of dynamic programming for matrix permutation problems in combinatorial data analysis", *Psychometrika*, 73(3), 503-522.

Chen, C-H., (2002), "Generalized association plots: information visualization via iteratively generated correlation matrices", *Statistica Sinica*, 12, 7-29.

Degerman, R. (1982), "Ordered Binary Trees constructed through an application of Kendall's Tau", *Psychometrika*, 47, 523-527.

Earle, D. (2010), "Dendrogram seriation in data visualisation: algorithms and applications". PhD thesis, National University of Ireland Maynooth.

Eisen, M.B., Spellman P.T., Brown, P.O., & Botstein, D. (1998), "Cluster analysis and display of genome-wide expression patterns". *Proceedings of the National Academy of Science of the United States*, 95(25), 14863-14868.

Forina, M., Lanteri, S., Casale, M. & Concepción Cerrato Oliveros, M. (2007), "A new algorithm for seriation and its use in similarity dendrograms", *Chemometrics and Intelligent Laboratory*

Systems, 87(2), 262-274.

Friendly, F. & Kwan, E. (2003), "Effect ordering for data displays", *Computational Statistics & Data Analysis*, 43, 509-539.

Gale, N., Halperin, W.C. & Costanzo, C.M. (1984), "Unclassed matrix shading and optimal ordering in hierarchical cluster analysis", *Journal of Classification*, 1, 75-92.

Gruvaeus, G. & Wainer, H. (1972), "Two additions to hierarchical cluster analysis", *British Journal of Mathematical and Statistical Psychology*, 25, 200-206.

Hahsler, M., Hornik, K. & Buchta, C. (2010), "seriation: Infrastructure for seriation". R package version 1.0-2.

Hubert, L., Arabie, P. & Meulman, J. (2006), *The structural representation of proximity matrices with MATLAB*, Philadelphia: SIAM.

Hurley, C. (2004), "Clustering visualizations of multidimensional data", *Journal of Computational & Graphical Statistics*, 13(4), 788-806.

Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G. and Shmoys, D.B. (1985), *The Traveling Salesman Problem*. John-Wiley & Sons, Ltd.

Morris, S.A., Asnake, B. & Yen, G.G. (2003), "Optimal dendrogram seriation using simulated annealing", *Information Visualisation*, 2, 95-104.

R Development Core Team (2010), R: A language and environment for statistical computing, R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org.

Robinson, W.S. (1951), "A method for chronologically ordering archaeological deposits", *American Antiquity*, 16, 293-301.

Rothkopf, E.Z. (1957), "A measure of stimulus similarity and errors in some paired associate learning tasks", *Journal of Experimental Psychology*, 53, 94-101.

Tien, Y-J., Lee, Y-S., Wu, H-M. & Chen, C-H. (2008), "Methods for simultaneously identifying coherent local clusters with smooth global patterns in gene expression profiles", *BMC Bioinformatics*, 9(155).

Tubb, A., Parker, A.J. & Nickless, G. (1980), "The analysis of Romano-British pottery by atomic absorption spectrophotometry", *Archaeometry*, 22(2), 153-171.

Wishart, D. (1999), "ClustanGraphics 3 Interactive graphics for cluster analysis". In: Gaul W, Locarek-Junge H (Eds). *Classification in the Information Age*, 268-275, Springer.

Wilkinson, L. (2005), *The Grammar of Graphics (2nd Ed.)*, Springer-Verlag, New York.

Wu, H-M., Tien Y-J., & Chen, C-H., (2010), "GAP: A graphical environment for matrix visualization and cluster analysis", *Computational Statistics & Data Analysis*, 54(3), 767-778.