Pairwise display of high dimensional information via Eulerian tours and Hamiltonian decompositions

C.B. Hurley*and R.W. Oldford†

February 11, 2010

Abstract

A graph theoretic approach is taken to the component order problem in the layout of statistical graphics. Eulerian tours and Hamiltonian decompositions of complete graphs are used to ameliorate order effects in statistical graphics. Similar traversals of edge weighted graphs are used to amplify the visual effect of selected salient features in the data. Relevant graph theory is summarized and classic algorithms are tailored to this problem. Graphics for multiple comparisons are reviewed and a new display developed that is based on graph traversal. Improved star glyph displays of multivariate data are described. Parallel coordinate displays tailored to particular features of the data are developed. The methods and new graphical displays are made available as an R package, `PairViz`.

*Research supported in part by a Research Frontiers Grant from Science Foundation Ireland.

†Research supported in part by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada.

1 Introduction

Graphical displays frequently require an ordering of their components (e.g. matrix displays, glyphs, parallel coordinate plots, etc.). The ordering itself is an encoding of information that, if neglected, could hide or distort important information in the data.

Ordering has long been used to good effect, to reveal more about the data, to encourage data comparisons, and to make datasets coherent – in short to meet Tufte's (1987) principles of graphical excellence. Wilkinson and Friendly (2009) present a number of early examples of permuting rows and columns of various matrix displays (e.g. Brinton, 1914, and Bertin's 1967 "reorderable matrix"). Some of these sort rows and columns, some permute them to make pairwise patterns more apparent.

There are many techniques available for obtaining improved orderings for data displays. For example, Friendly and Kwan (2003) and Hurley (2004) describe methods which place similar variables adjacent to each other in displays such as scatterplot matrices and parallel coordinates, thus simplifying interpretation, while Peng et al (2004) take a clutter reduction approach. Wilkinson (2005) reviews a variety of ordering algorithms (see references therein); many of these and others are implemented in the R package `seriation` (Hahsler et al, 2008). The present paper takes a different approach to the ordering problem.

We abstract the problem to one of graph traversal, and so are able to bring mathematical results and algorithms to bear on it. In some cases, traversals can be chosen to ameliorate the order effect, rendering the display more nearly invariant to the component ordering. In other cases, some traversals are chosen over others to reinforce the desired effect of the display.

Section 2 gives an introductory example. Here we investigate the effect of ordering in star glyph displays, and suggest how the order effect may be minimized. Section 3 surveys the relevant graph theory and summarizes those mathematical results most applicable to

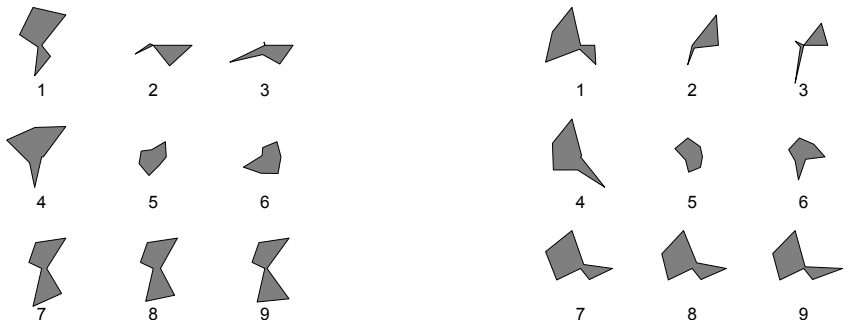
the ordering problem. The section stands on its own and is applicable to any statistical problem where order is of concern, not just ones involving data visualization. In Section 4, we work through two further examples which apply the graph theoretic framework in a statistical graphic framework. First we develop a new statistical display for multiple comparisons by recasting the problem in terms of graph traversal. Second we develop improvements on parallel coordinate displays by identifying particular graph traversals with particular coordinate arrangements. Graph edge weights allow the display to target interesting features of the data. Section 5 describes the algorithms used to produce the graph traversals used and some closing remarks are made in the last section.

2 Example – star glyphs

In a star glyph display, variables are assigned to equispaced radii, case values plotted on each radius, and lines drawn connecting them. High-dimensional features of the data are quickly explored across cases by comparing glyphs – individual radii and overall shape, an example of what Tufte (1991) describes as "small multiples". Suppressing the radial rays from the display, one focuses on comparison of shapes rather than of variable values across cases – a distinction which has been usefully described as that between integrable and separable dimensions by Wilkinson (2005, p. 269). A star glyph display invites visual clustering by shape. Figure 1 shows four star glyph displays of a subset of car models from the `mtcars` dataset[1] given in R. Each display uses the same seven variables; they differ only in the assignment order of variables to axes. Not surprisingly, the shapes of the star glyphs vary considerably from one ordering to another.

Let's attempt to use the first ordering to cluster the cars. The glyphs for models 7, 8

[1]The star glyphs use the first seven variables of the dataset and the selected models are in rows 7,28,27,31,30,1,12 and 14.



(a) H_0: default dataset order of variables

(b) order H_1

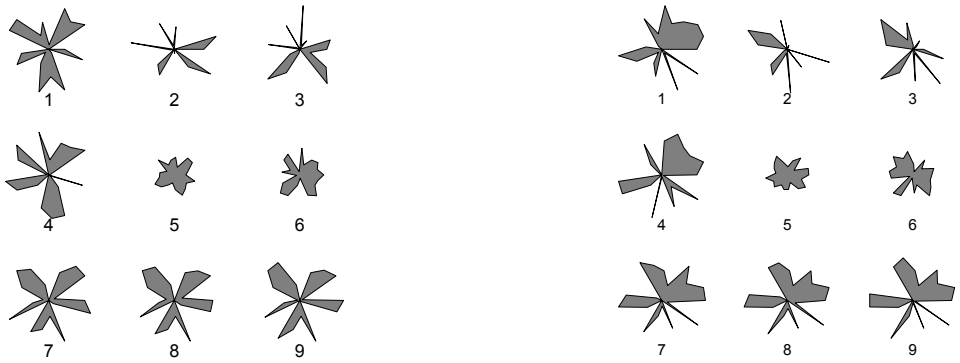
(c) order H_2

(d) order H_3

Figure 1: Star plots of 9 models from the `mtcars` data using different variable orderings.

and 9 look very similar to each other, and quite similar to the glyph for model 1. Models 5 and 6 both appear as roughly similar medium-sized blobs. The model 4 glyph looks different from all others. Other orderings tell a different story; in order H_1 model 4 looks like it belongs to the $\{1, 7, 8, 9\}$ cluster, while in H_2 and H_3 we have two separate clusters consisting of models $\{1, 4\}$ and $\{7, 8, 9\}$. Clearly visual clustering based on star glyph displays is order dependent.

Intuitively, if we replaced the sequence of variables used in the star glyph by a longer sequence where all pairs of variables appear adjacently, we should remove some of the dependence on variable order. Figure 2 shows two such sequences of variables. The first plot uses the sequence $H_1 : H_2 : H_3$, which is a concatenation of the sequences appearing

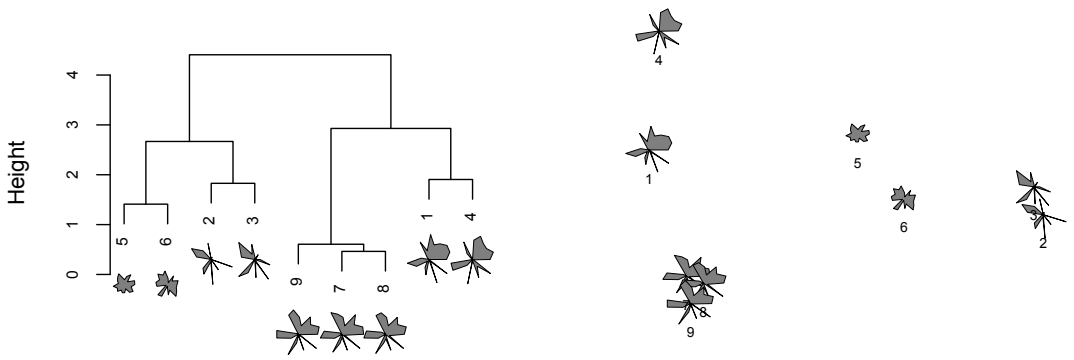


(a) Order with repeated variables (b) Correlation order with repeated variables

Figure 2: Star plots of 9 models from the `mtcars` data.

in Figure 1(b), (c) and (d), and was constructed using our algorithm WHam (see Section 5.2.3). The second plot uses a sequence constructed via our algorithm GrEul (see Section 5.1.2) which favours high correlation pairs of variables appearing early on in the sequence. We notice that the shapes of the star glyphs vary less with the sequences used in Figure 2 than with those used in Figure 1. This occurs because in Figure 1 the star vertices are rearranged in each ordering, whereas in Figure 2, it is the star edges that are rearranged.

Visual clustering based on either of the sequences shown in Figure 2(a) and (b) gives the same results; it appears there are four clusters, made up of models $\{1, 4\}$, $\{2, 3\}$, $\{5, 6\}$ and $\{7, 8, 9\}$. These findings are verified by the dendrogram shown in Figure 3(a) obtained from average link clustering (single and complete linkage dendrograms were identical) and reinforced by plotting the symbols in the space of the first two principal components (based on the correlation matrix) as shown in Figure 3(b). This example suggests that star glyphs based on sequences where all pairs of variables appear adjacently lead to more reliable visual clustering.



(a) Dendrogram: Average linkage

(b) Glyph scatterplot on first two principal components

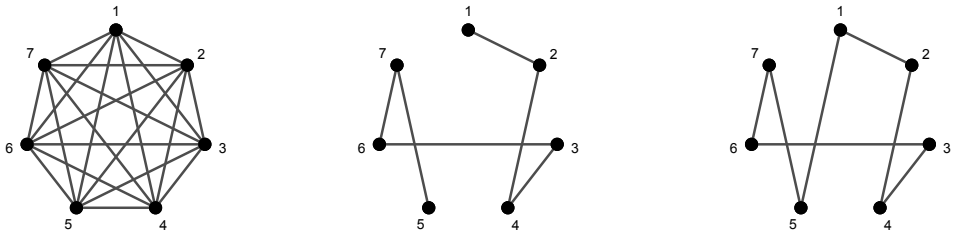
Figure 3: Clustering the cars.

3 Graph theory

In this section we present results from graph theory which are relevant to the component order problem in the layout of statistical graphics, and in particular to the construction of the repeated variable glyphs shown in Figure 2.

The complete graph on n nodes or vertices is an undirected graph, denoted K_n, with vertex set $V(K_n) = \{1, 2, \ldots, n\}$ and edge set $E(K_n) = \{e_{ij} | i, j \in V(K_n), i \neq j$ with $e_{ij} = e_{ji}\}$ (when there is no ambiguity, the edge e_{ij} might also be written ij). The cardinality of the vertex set is called the graph's *order*, here n, and that of its edge set the *size* of the graph, here $|E(K_n)| = n(n-1)/2$. A path P on K_n is an alternating sequence of vertices and edges such that each edge has as its endpoints the preceding and succeeding vertices, i.e., $P = i, e_{ij}, j, e_{jk}, \ldots, e_{st}, t$,

Figure 4(a) shows K_7. A complete graph is a convenient representation of n objects (the nodes) together with all possible pairings (the edges). Any path along edges of the graph simultaneously provides an arrangement of those objects identified with the nodes of the path and of the pairings identified with the edges of the path.



(a) K_7 (b) A Hamiltonian path (c) A Hamiltonian cycle

Figure 4: K_7, Euler tours, and Hamiltonians.

3.1 Hamiltonians, Eulerians, and Hamiltonian decompositions

A path is called a *Hamiltonian path* if it visits all vertices of a graph exactly once. The Hamiltonian path of Figure 4(b) orders the nodes as 1243675 (or the reverse), and is identified with a permutation of the nodes. The set of all Hamiltonian paths on a complete graph K_n is the set of all permutations of $1, 2, \ldots, n$. Closing a Hamiltonian path by joining its ends, as in Figure 4(c), creates a *Hamiltonian cycle* which can be identified with many permutations (each being a cyclic permutation of the original).

For example, for the star glyph display of Figure 1, imagine the variables as nodes of a complete graph. Then the order of assignment of the seven variables to the radial arms of the star is equivalent to the choice of a Hamiltonian cycle on K_7. The orderings H_0, H_1, H_2 and H_3 of Figure 1 (a)-(d) correspond to four different Hamiltonian cycles on K_7.

A graph G is *Hamiltonian* if it contains a Hamiltonian cycle. Complete graphs K_n are Hamiltonian for all n and K_n contains $(n-1)!$ distinct Hamiltonian cycles.

Equivalently, a path can be regarded as providing an ordering on the edges it contains. Figure 4(b) orders its edges as $12, 24, 43, 36, 67, 75$ to which the Hamiltonian cycle of Figure 4(c) adds the edge 51. Often interest lies in visiting (and hence ordering) all of the edges in a graph. A path which contains all of the edges of a graph, visiting each edge exactly once is called an *Eulerian path (or Eulerian trail)* and if the path is closed then the

traversal is called an *Eulerian tour*. A graph G which has an Eulerian tour is called *Eulerian*. The graph K_7 of Figure 4(a) is Eulerian. An Eulerian tour of a complete graph provides an arrangement of all possible pairings of the nodes. One such tour for K_7 is $T_0 = 12345674614271573526 31$. We have already seen an application of such tours in visualization. In fact the variable sequences of Figure 2 are two other Eulerian tours on K_7.

3.1.1 Many to choose from

As with Hamiltonians (cycles and paths), there need not be a unique Eulerian tour for a given graph. Typically there are a great many to choose from. For example, K_7 admits $129,976,320$ Eulerian tours that are not cyclic permutations of one another (first determined by Reiss, 1871-3; see McKay and Robinson, 1998) while K_{21} has more than 3.4×10^{184}. (For odd $n \leq 21$, the number is available online via Sequence A007082 of the Online Encyclopedia of Integer Sequences (Sloane 2004).)

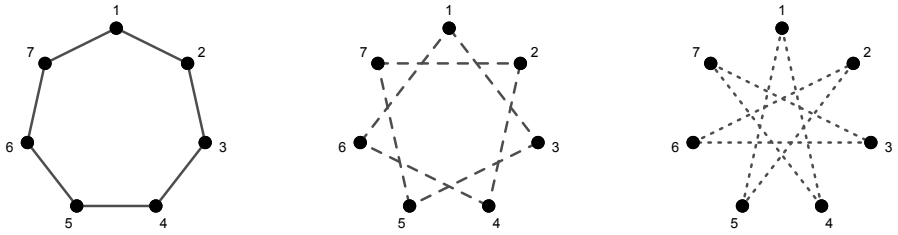
While an Eulerian tour of a complete graph produces an arrangement of all possible pairings, it may be that some Eulerian tours (arrangements) are preferred over others. With some measure of the value of each, the Eulerian tours could, in principle, be ordered and the best selected.

Supposing each edge in the graph has a weight, we might prefer Eulerians whose edge weights were by some measure as low (high) as possible in the early part of the sequence. A greedy Eulerian might be one which begins with the lowest (highest) weight edge. From there, it moves to the adjacent edge (i.e. sharing a vertex) with lowest (highest) weight, and after that follows low (high) weight unvisited edges until all edges are exhausted. For example, the Eulerian on K_7 constructed for Figure 2(b) starts at the pair of variables with the highest correlation placed at the 3 o'clock position. Moving counter-clockwise,

the Eulerian follows high correlation edges thereafter.

3.1.2 Hamiltonian decomposed Eulerian tours

One possibility is that the Eulerian tour be composed entirely of edge-distinct Hamiltonian cycles, a so-called *Hamiltonian decomposition*. Figure 5 shows a Hamiltonian decomposi-



(a) $H_{11} = 12345671$ (b) $H_{12} = 13572461$ (c) $H_{13} = 14736251$

Figure 5: A Hamiltonian decomposition $H_1 = H_{11} : H_{12} : H_{13}$ of K_7.

tion of K_7. Note that this decomposition is also a *symmetric* Hamiltonian decomposition because a node labelling exists which makes all cycles symmetric about the same node (this decomposition is in fact symmetric about every node).

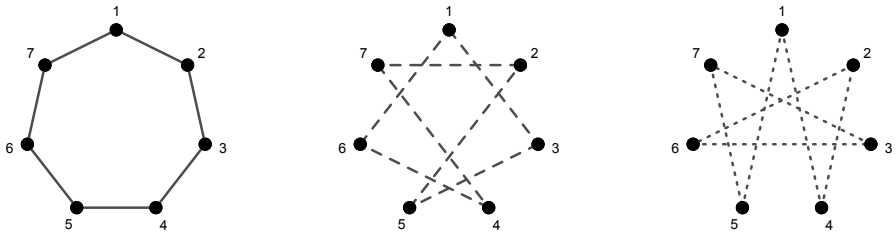
An Eulerian tour can be had by joining these Hamiltonian cycles, in any order, at the same node. For example, $T_1 = 1\ 234567\ 1\ 357246\ 1\ 473625\ 1$ is an Eulerian tour that joins the three Hamiltonians at 1 in the order of H_{11}, H_{12}, H_{13}; $T_2 = 2\ 345671\ 2\ 753164\ 2\ 514736\ 2$ joins the cycles at node 2 in the order H_{12}, H_{11}, H_{13} with the middle cycle reversed. For any Hamiltonian decomposition, an Eulerian tour can be constructed by varying the order of the Hamiltonian cycles, varying the direction in which each cycle is traversed, and varying the point of contact between the cycles. For example, Figure 2(a) shows an Eulerian tour on K_7 constructed by appending the edge-disjoint Hamiltonian cycles $H_1 : H_2 : H_3$ given in Figure 1.

Moreover, the Hamiltonians in Figure 5 are presented in canonical form (in terms of node labelling as given in Colbourn, 1982), so permuting the node numbers on Figure 5(a)

and carrying that assignment across the Hamiltonians of Figure 5(b) and (c), can produce a different Hamiltonian decomposition and consequently many more Eulerian tours.

By construction, these different decompositions will be isomorphic to one another (two Hamiltonian decompositions H and H' are isomorphic if there is a one-to-one mapping of the nodes of the graph onto themselves which maps each Hamiltonian cycle of H onto a Hamiltonian cycle of H') and will sometimes be identical (e.g. the decomposition produced by mapping the nodes 1234567 of Figure 5 to 2715436 is identical to that of mapping 1234567 to 4675321). In this way the Hamiltonian decomposition of Figure 5 generates a class of decompositions. It does not, however, generate all Hamiltonian decompositions of K_7.

There is only one other set of isomorphic Hamiltonian decompositions of K_7 which is not isomorphic to that of H_1 from Figure 5. The canonical form for this set is H_2 of Figure 6 (see Colbourn, 1982).



(a) $H_{21} = 12345671$ (b) $H_{22} = 13527461$ (c) $H_{23} = 14263751$

Figure 6: $H_2 = H_{21} : H_{22} : H_{23}$ is the canonical form of the second (and only other) set of Hamiltonian decompositions of K_7.

H_2 is also a symmetric decomposition, though one with many fewer symmetries than H_1 (i.e. only about node 1 in H_2). The fewer symmetries result in a smaller group order of the automorphisms (6 for H_2 vs. 42 for H_1) and consequently many more distinct (though isomorphic) decompositions (viz. $7!/42 = 120$ for H_1, $7!/6 = 840$ for H_2).

As before, the cycles of each distinct decomposition can be arranged in many ways to

produce different Eulerian tours. Using H_1, there will be thousands of distinct Hamiltonian decomposed Eulerian tours for K_7; using H_2 there will be seven times as many to choose from.

For larger orders of complete graphs, the number of non-isomorphic classes of Hamiltonian decompositions is huge. There are 122 non-isomorphic decompositions of K_9 and more than $45,000$ for K_{11} (Colbourn, 1982, stopped computing more after finding this many). While in principle it is possible to order the Hamiltonian decompositions according to some preference, it is rarely practicable. Even choosing the single Hamiltonian having the smallest total edge weight (i.e. the travelling salesman problem) is NP hard.

3.2 General results for complete graphs

If G is a connected graph, G is Eulerian if and only if it is an even graph (i.e. every vertex has an even number of edges), or equivalently if and only if G has a cycle decomposition. Since complete graphs where the order n is odd are connected and even, Eulerian tours and Hamiltonian decompositions exist.

The same notions can be extended to the complete graphs of even order through the following well known results which have been attributed to Walecki (by Lucas, 1892; e.g. see Alspach, et al 1990):

Decomposition of complete graphs. K_n can be decomposed as follows:

For $n = 2m + 1$, into either

m Hamiltonian cycles, or

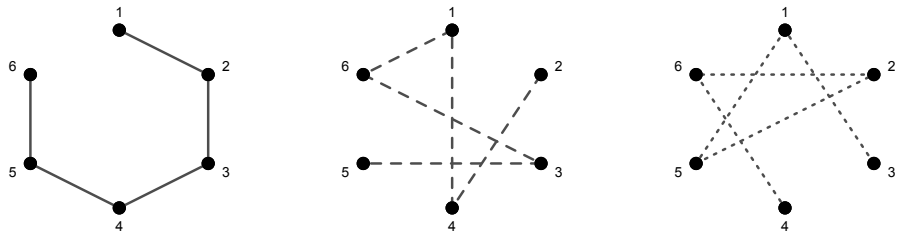
m Hamiltonian paths and an almost-one factor.

For $n = 2m$, into either

m Hamiltonian paths, or

$m - 1$ Hamiltonian cycles and a 1-factor (or perfect matching).

The Hamiltonian cycle decomposition for the case of odd n has already been illustrated. When n is even, the analogous decomposition of K_{2m} is into Hamiltonian paths rather than cycles. Figure 7 shows one such decomposition for K_6. This was had directly from the

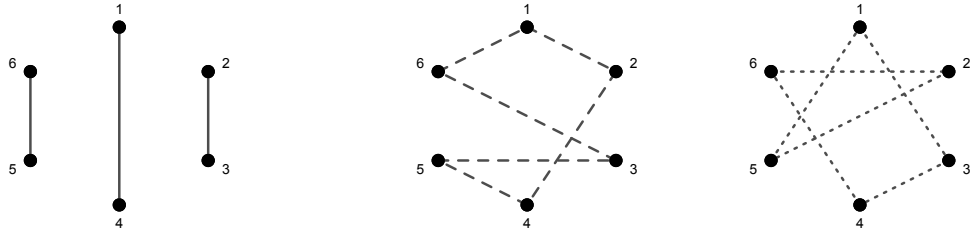


(a) $h_{31} = 123456$ (b) $h_{32} = 241635$ (c) $h_{33} = 315264$

Figure 7: $H_3 = h_{31} : h_{32} : h_{33}$ is a Hamiltonian *path* decomposition of K_6.

Hamiltonian cycle decomposition of Figure 6 by deleting node 1 and relabelling nodes $2-7$ as $1-6$; one might just as easily have used Figure 5.

Alternatively K_6 can be decomposed into a 1-factor (or perfect matching) and two Hamiltonian cycles as shown in Figure 8. Similarly, K_{2m+1} is decomposable into m Hamil-



(a) 1-factor (or perfect matching) (b) 1245361 (c) 1346251

Figure 8: K_6 decomposed into a 1-factor and two Hamiltonian cycles.

tonian paths and an "almost 1-factor" (i.e. a 1-factor perfectly matching $2m$ points plus a single isolated vertex).

Although K_{2m} is not even, and hence not Eulerian, m edges can be added to produce a graph that will be Eulerian and will have Hamiltonian cycle decompositions. For example,

simply close the Hamiltonian paths of Figure 7 to produce double edges 16, 25, and 34. If only an Eulerian path is required, only $m-1$ edges need be added – an Eulerian path exists for any connected graph having exactly two nodes of odd degree, so the $m-1$ edges added must be such as to satisfy this condition. For example, in Figure 7, add only extra edges 25, and 34; then an Eulerian path will begin and end at the endpoints of the remaining Hamiltonian path h_{31} of 7(a).

Alternatively, one could start with the Hamiltonian cycle decomposition of K_{2m} (e.g. $m = 3$ in Figure 8) and add $m-1$, or m, edges to the perfect matching to create a Hamiltonian path, or cycle, respectively that will in turn permit an Eulerian path, or tour (cycle) on the augmented graph.

Because complete graphs of even order can always be augmented to achieve Eulerian paths, etc., it will be convenient to have a single notation for both K_{2m+1} and the $m-1$ edge augmented graph of K_{2m}. Denote by

$$K_n^e = \begin{cases} K_{2m+1} & \text{if } n = 2m+1 \\ K_{2m} + G(K_{2m}) & \text{if } n = 2m \end{cases}$$

where $G(K_{2m})$ is a subgraph of K_{2m} having $m-1$ edges chosen so that the graph resulting from the sum has exactly two odd nodes.

It will also be convenient to refer to an *Eulerian of* K_n^e to mean an Eulerian *tour* of K_n^e when $n = 2m+1$ and an Eulerian *path* of K_n^e when $n = 2m$. Similarly a *Hamiltonian decomposition of* K_n^e will refer to a Hamiltonian *cycle* decomposition when $n = 2m+1$ and a decomposition into $m-1$ Hamiltonian cycles plus one Hamiltonian path when $n = 2m$.

There will of course be many Hamiltonian decompositions, and many more Eulerians, of K_n^e to choose from.

4 Applications to statistical graphics

Section 2 showed how a relatively straightforward use of Eulerians and Hamiltonians could immediately produce more reliable visual clustering by star glyphs. This is but one example where known graph theory can be usefully applied to statistical graphics. In this section we provide two more.

The first is a new display for the classic statistical problem of multiple comparisons. By recasting the problem as a graph theoretic one within a statistical context, an entirely new display is developed. An Eulerian tour in the context of the pairwise comparison of treatment groups provides a simple but powerful approach to this classic problem. Section 4.1 reviews some well known displays for this problem, develops the new construction, and illustrates it on a dataset from Cameron and Pauling (1978).

The second example uses graph theory to improve parallel coordinate plots (e.g. see Inselberg 1985, 2009, Wegman 1990). It draws on essentially all of the theory reviewed above for complete graphs. Casting the layout of the parallel axes as a graph traversal problem encourages us to add information to the graph, statistical information to produce traversals designed to reveal targetted structure of the data. Hamiltonian paths, Eulerian tours, and Hamiltonian decompositions are all used to construct different parallel coordinate displays, each suited to a different purpose.

There are numerous other opportunities to apply the above complete graph framework to data visualization, improving existing graphics and developing new ones. Hurley and Oldford (2008a) describe some and Hurley and Oldford (2008b) show how graphs can be applied to explore high dimensional space through dynamic scatterplots. Key to any of these applications is abstraction of the visual layout problem to one of nodes connected by meaningfully weighted edges.

4.1 A new display for pairwise comparisons

In the classic one-way anova situation, several conditions are compared at once for differences in some outcome. Also of interest are all pairwise comparisons, with correction for the problem of multiple comparisons.

To be concrete, we take data on the survival times of terminal patients with different types of cancer – namely Breast, Bronchus, Colon, Ovary, or Stomach, who have been treated with vitamin C (Cameron and Pauling, 1978). The square root of the survival times are used to better approximate normality. Figure 9 shows the 95% simultaneous

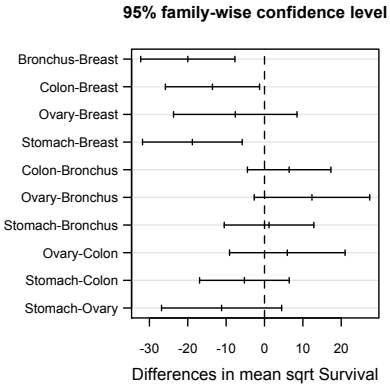


Figure 9: A standard layout of 95% confidence intervals for differences of mean survival times (square root scale), corrected for multiple comparisons.

confidence intervals for the pairwise difference in means, using "Tukey's honest significant differences". This tidy little layout is fairly standard for multiple comparisons (e.g. it is the default plot method for `TukeyHSD`, Bates, 1997+).

Comparisons whose intervals do not overlap the vertical zero line are statistically significant (e.g. Bronchus-Breast) at a simultaneous 5% level and those which do overlap are not statistically significant (e.g. Ovary-Breast). Each interval estimates the magnitude of the corresponding difference (at a 95% simultaneous confidence level). The magnitude of

the individual means is absent from this display.

This and other multiple comparison plots are critically examined by Hsu and Peruggia (1994) who also introduce an interactive "mean-mean" plot designed to address the shortcomings of the existing plots. Figure 10 shows the Heiberger and Holland (2006a)

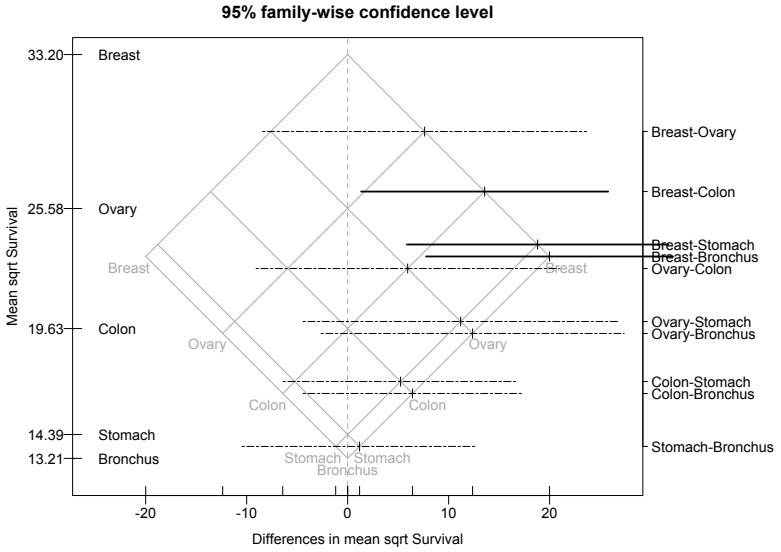


Figure 10: Mean-mean multiple comparison plot: 95% simultaneous confidence intervals, from the R package `HH` (Heiberger and Holland, 2006b). Significant differences are shown as a thick black interval.

static version of the mean-mean plot. Treatment means and pairwise confidence intervals appear on the same display, against the background of a rotated grid. Drawbacks include the possibility of overstriking intervals or grid lines or both whenever pairwise averages or original averages or both are identical (or nearly so). Neither is it clear that the information added by the background grid (originally used to motivate and construct the plot) merits the amount of ink it is given. An important feature is that it shows the sample means themselves in addition to their differences. In most applications, having identified the significant differences one is interested in the actual size of each effect being compared.

Indeed, a comparison of the entire distribution of each group, not just their group means, is highly desirable.

4.1.1 Boxplots with pairwise testing

Sample distributions can be displayed as histograms, boxplots, density estimates, and so on, which in turn can be compared along a common scale in a variety of ways: possibly overlaid (e.g. densities), or placed back to back (e.g. histograms, densities) or simply laid out side by side (e.g. boxplots, histograms, density estimates). Here we will use a boxplot for the distribution of each group and lay them out side by side to facilitate their pairwise comparison.

Figure 11 shows variable width boxplots of the (square-root transformed) survival times

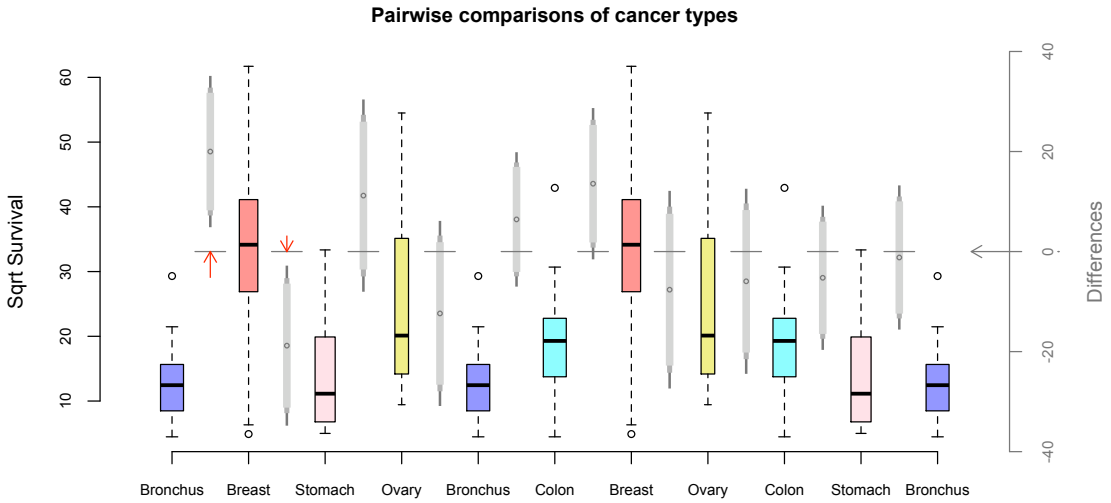


Figure 11: Boxplots and pairwise comparisons of vitamin-C treated cancer patients. The left axis and boxplots refer to square-root transformed survival times, the right axis and gray scale vertical bars refer to confidence intervals for pairwise differences of means. Red arrows indicate comparisons significantly different from 0.

for each cancer type. The cancer types (with their boxplots) are repeated along the hori-

zontal axis in such a way that every cancer type appears directly beside every other cancer type exactly once. From left to right, this is an Eulerian tour of K_5, where each cancer type corresponds to a node. Because every pair appears together, a fairly rich comparison of survival distributions can be made via the boxplots. Each cancer type's boxplot is uniquely coloured to facilitate directed comparisons. For example if interest lies primarily in comparing the survival distribution of Ovary cancer to that of the others, simply look for each occurrence of Ovary's thin yellow boxplot and compare it with the boxplots on either side.

Between each pair of boxplots is a grey vertical strip. Each strip is a Tukey HSD confidence interval for the difference in means between the distributions on either side of it, each circle indicating the point estimate of that difference. All confidence interval values can be read from the vertical axis of differences at the right of the plot. Just as the boxplots are the nodes of K_5, the ten grey confidence intervals between them are the edges of K_5. Moving from left to right, from boxplot to confidence strip to boxplot to confidence strip and so on, is an Eulerian tour traversal of K_5 from node to edge to node to edge, respectively.

Design features of this plot are chosen to help the user switch visual focus between the mutely coloured boxplots and the gray confidence intervals as need be. This is much like the "layering" of information, simple examples of which have been described by Tufte (1991). For example the right axis is for the confidence intervals and a grey arrow from this axis anchors a horizontal dashed grey line across the plot from its zero. The dashes of this line appear only across the space between boxplots which is reserved for the confidence intervals – the line never interferes with the boxplots themselves.

As with Figures 9 and 10, inference is had by determining whether the zero line cuts across a confidence interval. If it does (e.g. between Stomach and Bronchus at the right)

that difference is not found to be statistically significantly different from zero. Conversely, if it fails to cut through a confidence interval (e.g. between Bronchus and Breast at the left side of the plot) then the difference is significantly different from zero. When this occurs, a vertical red arrow is drawn pointing towards the confidence interval (and on the opposite side of the horizontal line) to draw attention to the interval. Moreover, the greater the length of the arrow is, the greater is its significance (i.e. the smaller its "p-value").

Note that unlike the previous plots (Figures 9 and 10) only two differences are seen to be significantly different from zero in Figure 11, namely (Breast - Bronchus) and (Stomach - Bronchus). The reason (Colon - Breast) does not show up here is that this plot shows confidence intervals for several levels simultaneously and the largest value here is 99% not 95% as in the other plots.

Careful examination of the vertical confidence intervals of Figure 11 will reveal that they progressively narrow and darken at the ends. In the figure each interval has three widths and three shades of gray corresponding to three confidence levels: 90%, 95% and 99%. A close look at the confidence interval between the Colon and Breast boxplots shows that the horizontal zero line cuts through the 99% confidence interval, but not the 95%. The difference is significant at the 5% level just as in the other plots but, as this plot indicates, is not significant at the 1% level.

Note also that significant differences seem to appear mostly on the left side of this plot. This is had by attaching a weight to each edge of K_m^e (here K_5) and applying a greedy algorithm which selects the lowest weight edge from those available at each step (algorithm GrEul of Section 5.1.2). To produce Figure 11, graph edges are weighted by the appropriate significance level. A different choice of weights could produce a different Eulerian tour and hence ordering.

In summary, the graph traversal approach has motivated the design of a relatively

simple yet highly informative plot for multiple comparisons.

4.2 Improving parallel coordinate plots

Parallel coordinate displays (Inselberg 1985, 2009, Wegman 1990) are multivariate data displays where n variables are assigned to parallel, equispaced axes, observations are plotted on each axis and lines are drawn connecting observations belonging to each case. These displays are useful for detecting clusters, outliers and correlation between pairs of variables. Once again, choosing a variable ordering amounts to selecting a Hamiltonian path on the complete graph with the variables as nodes. However, as demonstrated by Wegman(1990), there are strong reasons for displaying all pairwise variable relationships in a parallel coordinate display, not just the $n-1$ pairwise relationships determined by a particular choice of Hamiltonian.

Here we use parallel coordinate displays to revisit the `mtcars` data. Figure 12 shows

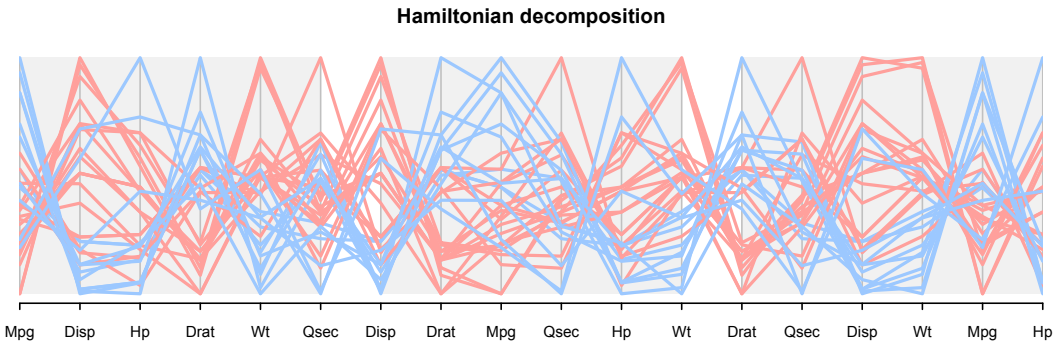


Figure 12: Parallel coordinate plots of the `mtcars` data. This shows a Hamiltonian decomposition, grey sections distinguish the three Hamiltonian paths. Line colour distinguish transmission type.

a parallel coordinate display using six performance measures and all 32 car models. The display has three sections, each highlighting a different Hamiltonian path, which together constitute a Hamiltonian decomposition on K_6^e (constructed using the method of Section 5.2.1). The first six axes (leftmost grey section) show the variables in the order in which

they are listed in the dataset, corresponding to the Hamiltonian 123456. Here we see that the first two variables, Mpg and Disp, are negatively correlated, but the association between the first and third variables, Mpg and Hp is not so obvious until we look at the last grey section and discover that they are also negatively correlated. The dataset has a cluster of unusually heavy cars which, we discover from the Wt-Disp panel in the last grey section, also have high displacement.

The main argument against all-pairs parallel coordinate displays is that the number of panels (i.e., the number of edges in the Eulerian on K_n^e) is $O(n^2)$. (From the discussion in Section 3.2, the number of edges is $\binom{n}{2}$ when n is odd, and $\binom{n}{2} + (n-2)/2$ when n is even.) Figure 13 shows an all-pairs parallel coordinate display for the `sleep` data which has

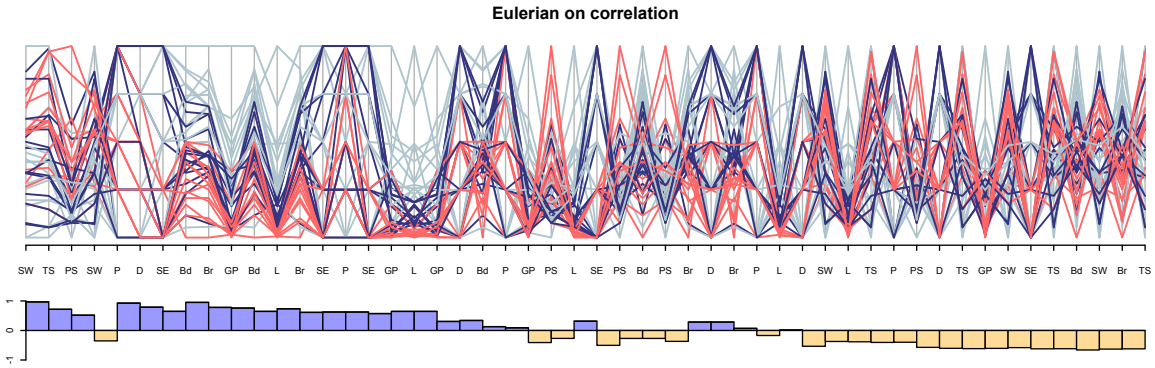


Figure 13: Parallel coordinate plot of the `sleep` data with panels ordered by correlation. Line colours are assigned using the life expectancy variable "L" – lowest third red, second third navy, last third light blue. The barchart show correlations for each panel.

$n = 10$ measurements on 62 mammal species (Allison and Cicchetti, 1976). The Eulerian has 49 edges and it may be difficult to see patterns on a standard computer screen or page. To ameliorate this, we use the GrEul algorithm of Section 5.1.2 to construct an Eulerian where panels exhibiting high positive correlation appear early on in the sequence. This correlation-driven Eulerian is presented in Figure 13, and the associated barchart

shows the correlation of each successive pair of variables. For the most part, panels in the first half of the sequence have positive correlation and thus less tangling of line segments. Panels in the second half of the sequence have negative correlation and their highly tangled, zig-zagging line segments obscures other patterns. Zooming in on the first 25 panels (not shown) produces a more interpretable display focusing on positive correlation.

Any "interestingness" measure could be used in place of correlation to structure the parallel coordinate display. Here we explore the use of so-called scagnostic indices, originally proposed by Tukey and Tukey (1985), revisited by Wilkinson et al (2005) and recently implemented in the R `scagnostics` package (Wilkinson and Anand, 2009). Scagnostics evaluate different features of a bivariate scatterplot, such as presence of outliers, striation and sparsity in the point pattern. Using scagnostics in the construction of greedy Eulerians or for identifying interesting Hamiltonians or even several from a Hamiltonian decomposition yields displays which focus on particular features of the data.

For example, consider the use of scagnostics to find outliers and clusters of data points in the `sleep` data. The top left display of Figure 14 shows the best Hamiltonian for the "outlying" index, that is maximizes the sum of index values along its path. The result is a parallel coordinate plot tailored for outlier discovery. Two outliers are particularly evident in the L-GP panel, and not surprisingly panels involving discrete variables (P, SE and D) score zero on this index. The second display of Figure 14 portrays the best Hamiltonian for the "clumpy" index. The index levels are generally low here and it seems this dataset does not exhibit much in the way of clumpiness, at least for variable pairs. Striated data is a particular form of clustering and the bottom left display of Figure 14 shows the best Hamiltonian for "striated" + "sparse". As we might expect, this index rewards discrete variables.

For each of the scagnostic indices of Figure 14(a-c), we could explore further "next

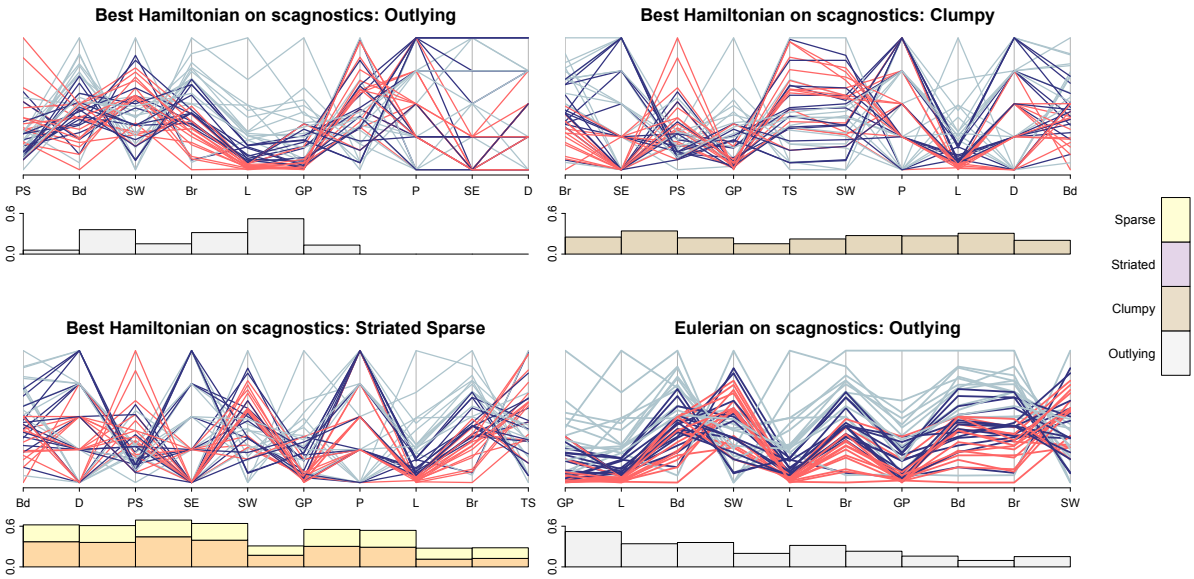


Figure 14: Hamiltonian parallel coordinate plots of the `sleep` data. Line colours are assigned using the life expectancy variable "L" – lowest third red, second third navy, last third light blue. The barcharts show scagnostic index levels for each panel, legend is on the right hand side.

best" Hamiltonians, as provided by the WHam algorithm of Section 5.2.3. This algorithm selects further Hamiltonians from the decomposition where the Hamiltonians are ordered by the sum of their index values. Another approach uses the GrEul algorithm to focus on high-index panels. The bottom right display of Figure 14 shows the first 10 axes from the Eulerian following outliers. Note that, by contrast with the Hamiltonian of Figure 14(a), the truncated Eulerian visits only 5 of the 10 variables. In this instance, the Hamiltonian captures all of the high index panels.

In summary, Figure 14 demonstrates two methods of zooming in on selected interesting subsets of all-pairs parallel coordinate plots which allow the analyst to focus on different features of the data.

5 Graph traversal algorithms

In this section, we describe algorithms for constructing various graph traversals which were used in the applications to statistical graphics of Sections 2 and 4. First we recall the standard algorithm for constructing Eulerian paths (Hierholzer 1873) and then modify it for weighted graphs. We then move on to constructions for Hamiltonian decompositions, specifically on complete graphs. Finally, we present a new algorithm which is useful for building Hamiltonians on complete graphs that are weighted.

5.1 Constructing Eulerian paths

5.1.1 Hierholzer's algorithm

Algorithm 1 (Hierholzer 1873) constructs Eulerian tours; another well known algorithm is

Algorithm 1 Hierholzer 1873 (adapted to find an Eulerian tour or open Eulerian path)

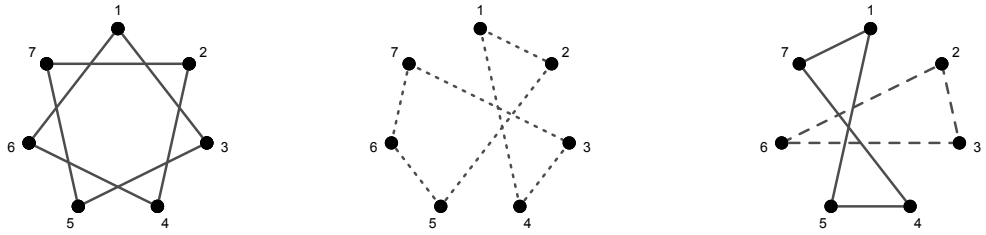
Require: A connected graph G that is even or that has exactly two odd vertices.

1: Choose a vertex v. If G is even, v can be any vertex, otherwise v is one of the two odd vertices.
2: Starting at v construct a path T in G, stopping when a vertex is reached without an unused edge.
3: **while** there are edges of G not already in path T **do**
4: Choose *any* vertex w in T that is incident on an unused edge.
5: Starting at w, construct a path D of unused edges stopping when a node is reached without any unused edges.
6: Enlarge T by splicing path D into T at vertex w.
7: **end while**
8: **return** T

due to Fleury (1883). Recall from Section 3.2 that Eulerian tours exist for even graphs, but with a minor adaptation Hierholzer's algorithm constructs an open Eulerian path or trail for graphs with exactly two odd nodes. Fleury's algorithm is essentially the same (e.g. see Fabràga and Fiol, 2004) and could be adapted analogously.

Hierholzer's method has many arbitrary choices – the choice of the vertex v in line 1 and at each step of the path constructed in lines 2 and 5, the choice of w in line 4, and if w appears more than once in T at step 6, the choice of which occurrence of w in T to use to splice path D into T (though the most recent is suggested).

Figure 15 shows how an application of Hierholzer's method might create an Eulerian



(a) First Hamiltonian cycle (b) Second Hamiltonian cycle (c) Two non-Hamiltonian cycles

Figure 15: An application of Hierholzer's method to K_7 which happens to follow one Hamiltonian cycle after another.

tour for K_7. Starting at node 1 the selection of edges is such as to produce the Hamiltonian cycle 13572461 of Figure 15(a), followed by a second Hamiltonian cycle 12567341 of Figure 15(b), and finally by the short cycle 15471 of Figure 15(c). At this point, node 1 has no further unused edges and $T = 1357246\ 1\ 256734\ 1\ 547\ 1$. Path D (of Algorithm 1 line 5) is the dashed cycle 2362 of Figure 15(c), which line 6 of the algorithm allows to be spliced into T at node 2. The resulting Eulerian tour can be either 1357 2362 46125673415471 or 13572461 2362 5673415471.

Hierholzer's method applies to the graph K_n^e for all n. When $n = 2m + 1$, $K_{2m+1}^e = K_{2m+1}$ is even and it yields an Eulerian tour. When $n = 2m$, K_{2m}^e is an augmented version of K_{2m} with exactly two odd nodes, and the result is an open Eulerian path.

5.1.2 Eulerians on weighted graphs

If the graph G is a weighted graph, we might prefer an ordered Eulerian T with low weight edges occuring early in the sequence and with weights tending to increase as the sequence progresses. As the discussion in Section 3.2.1 illustrates, the number of distinct Eulerian tours is typically immense, and finding the overall "best" tour is not a practical option. However, a greedy algorithm which attempts this is easily had by exploiting the arbitrary choices available in Hierholzer's method. The necessary minor modifications of Algorithm 1 are given below as the greedy Eulerian or GrEul of Algorithm 2. Note that the choice

Algorithm 2 GrEul: Greedy Eulerian.

Require: A connected graph G that is even or that has exactly two odd vertices.

1: *Choose a starting vertex v from one of the odd vertices connected by the lowest weight edge, using the next lowest weight edge in their vertex sets to decide between them.*
2: Starting at v construct a path T in G, *always moving to the lowest weight unused edge*, stopping when a vertex is reached without an unused edge.
3: **while** there are edges of G not already in path T **do**
4: Choose *the last* vertex w in T that is incident on an unused edge.
5: Starting at w, construct a path D of unused edges, *always moving to the lowest weight unused edge* and stopping when a node is reached without any unused edges.
6: Enlarge T by splicing path D into T at vertex w.
7: **end while**
8: **return** T

of starting vertex is limited to the two odd vertices when constructing Eulerian paths, but when constructing K^e_{2m} one can always ensure that a particular start vertex v has odd degree.

We note that constructing Eulerian trails is an $O(|E|)$ task, where $|E|$ is the size of the graph, and so the algorithm given above constructs trails on K^e_n in $O(n^2)$ time. The cost associated with constructing an ordered Eulerian must include the cost of an edge sort at each vertex, and so has overall order on K^e_n of $O(n^2 \log n)$.

5.2 Lucas-Walecki Hamiltonian decompositions for K_n^e

While the adapted Hierholzer method will produce an Eulerian for any K_n^e, the Eulerian need not be Hamiltonian decomposable. Even if node choices were restricted so that the algorithm first constructed one Hamiltonian followed by another, the result need not be a Hamiltonian decomposition. Figure 15 shows just such a situation.

Fortunately, the special structure of K_n^e can be exploited to write down explicit formulas for Eulerians and Hierholzer's method need not be used. This method has the added advantage that for $n = 2m$, the Eulerian is composed of m Hamiltonian paths, while for odd $n = 2m+1$, it is composed of m Hamiltonian cycles. The constructions given here have been attributed to Walecki by Lucas (1892), and are sometimes described as *Lucas-Walecki* constructions (Bailey et al, 2003). A disadvantage is that these necessarily generate only a single class of isomorphic decompositions which, though potentially huge, cannot include those produced from possibly thousands of other Hamiltonian decomposition classes that are non-isomorphic to this one.

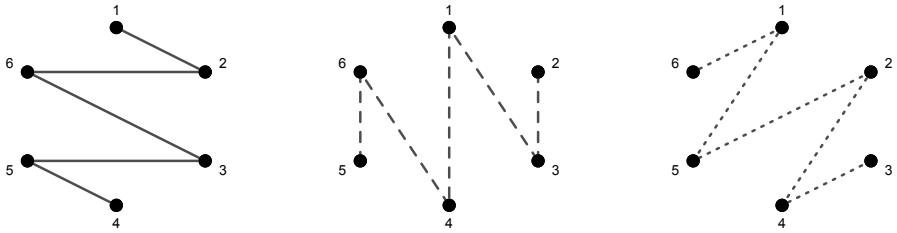
5.2.1 Hamiltonian decompositions, n even

As before, let $n = 2m$ and define

$$\begin{aligned} H[1,1] &= 0 \\ H[1,j] &= H[1,j-1] + (-1)^j(j-1) \pmod n, \quad j = 2,\ldots,n, \\ H[k,j] &= H[k-1,j] + 1 \pmod n, \quad k = 2,\ldots,m \text{ and } \quad j = 2,\ldots,n. \end{aligned}$$

Finally, increase each element of H by 1, and form T by listing the elements of H row-wise. The resulting path T is an Eulerian trail on K_{2m}^e.

When the vertices of K_{2m} are arranged clockwise around a circle, the first row of H visits all vertices in a zig-zag pattern. This is shown for K_6 in Figure 16(a). Each successive



(a) $h_{41} = 126354$ (b) $h_{42} = 231465$ (c) $h_{43} = 342516$

Figure 16: A Hamiltonian path decomposition $H_4 = h_{41} : h_{42} : h_{43}$ of K_6.

row of H follows another zig-zag starting one position clockwise away from the start of the previous row.

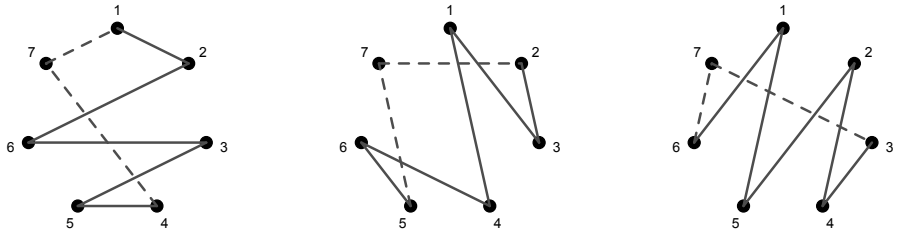
The rows of H form a Hamiltonian path decomposition of K_{2m} and consequently every pair of vertices appears consecutively in exactly one of the rows. When the rows are glued together to form the T-sequence, the edge pairs contributed by $H[i, n]$ and $H[i + 1, 1]$ are duplicates. These are the edges j $(j+m-1)$(i.e. $e_{j,(j+m-1)}$) for $j = 2, \ldots, m$. The resulting T-sequence, having duplicate edges, is a decomposition of K^e_{2m} into m Hamiltonian cycles, where in this case K^e_{2m} is formed from K_{2m} by adding the additional edges between nodes j and $j + m - 1$ for all $j = 2, \ldots, m$.

Figure 16 illustrates the process for K^e_6. Each panel shows a Hamiltonian path from a row of H, and these paths are joined up to give $T = 126354\ 231465\ 342516$. In this sequence the edges 4 2 (or 24) and 5 3 (or 35) are duplicates. Note also that this decomposition is isomorphic to the decomposition H_3 for K_6 given in Figure 7.

Wegman (1990) used this Lucas-Walecki construction to list m different permutations of $2m$ variables where each pair of variable adjacencies appears exactly once. Following Wegman (1990), we will use the more evocative name, "zig-zag method", to refer to this construction. For $n = 2m + 1$ the zig-zag method lists m permutations of variables, where each pair of variables appears adjacently at least once, but with some pairs appearing twice. The result will obviously not be a Hamiltonian decomposition.

5.2.2 Hamiltonian decompositions, n odd

An easy way of generating a Hamiltonian decomposition for $n = 2m + 1$ uses a minor modification of the zig-zag method just described. Start with the H matrix used in the construction of the path for $2m$ vertices, and create the augmented matrix H^* by prepending a column of n's to H. Row-wise listing the elements of H^* and adding a final n produces an Eulerian tour for K_{2m+1}. Each row of H^* has the form $n, j, \ldots, (m + j)$, so we have inserted the required edges nj at the beginning of each row and $(m + j)n$ at the end, for $j = 1 \ldots, m$. The extra n at the end of the T-sequence contributes the edge $(n - 1)n$. For example when $n = 7$, we transform the $n = 6$ sequence of 126354 231465 342516 to 7 126354 7 231465 7 342516 as illustrated in Figure 17. Note that this decomposition is



(a) $H_{41} = 1263547$ (b) $H_{42} = 1327564$ (c) $H_{43} = 1524376$

Figure 17: A Hamiltonian decomposition $H_4 = H_{41} : H_{42} : H_{43}$ of K_7. For K_{2m+1}, the paths for K_{2m} are constructed and then the point $2m + 1$ joined to the ends to complete the cycle.

isomorphic to the decomposition H_2 for K_7 given in Figure 6.

In general, each row of H^* is a Hamiltonian path, and since each row begins with n, we have formed a decomposition of K_{2m+1} into m Hamiltonian cycles.

Interestingly, constructions of Hamiltonian decompositions on K_{2m+1} have applications in experimental design. Bailey et al (2003) call these decompositions *round-dance neighbour designs* They relate them to Latin and Tuscan squares and give a number of other constructions.

5.2.3 Hamiltonian decompositions on weighted graphs

For weighted graphs our goal is an ordered Eulerian T where weights tend to increase as the sequence progresses. Here we will build such paths out of Hamiltonians. Note that Figure 2(a) of Section 2 gave an application of these paths to star glyph displays, while in Section 4.2 we discussed their application to parallel coordinate plots.

For a given Hamiltonian (path or cycle) decomposition $H = H_1 : H_2 : \cdots : H_m$, it is clear from the discussion of Section 3.1 that since the labelling of vertices is arbitrary, any sequence of vertices can be chosen as the first (or any other) Hamiltonian in the Hamiltonian decomposition, but then the other Hamiltonians in the path must follow the same labelling scheme. The order in which the Hamiltonians appear in constructing the Eulerian can be permuted and each component path or cycle H_i can be oriented arbitrarily to form the Eulerian composed of these Hamiltonians.

These operations open up a huge number of possible paths, far too many to attempt to find the overall winner based on some merit measure using edge weights. However some preferences can be made algorithmically. For example, given an Eulerian T composed of a Hamiltonian decomposition $H = H_1 : H_2 : \cdots : H_m$ (e.g. arrived at by applying the zig-zag algorithm) we could choose to order the Hamiltonians within the decomposition so that those with smaller total edge weight precede those with larger total edge weight. Then within each Hamiltonian we could choose to orient the path (or cycle) so that smaller weights tended to appear earlier in the path (cycle) than larger weights (a strict ordering will not likely be possible). If no decomposition is in hand, we could first choose a Hamiltonian with smallest total weight out of all possible Hamiltonians. It would be nice to think that we could do this recursively, always getting the next best Hamiltonian from the remaining graph, but as the example of 8 shows, it is possible to produce several disjoint Hamiltonians in sequence without arriving finally at a full decomposition. So recursing

in this way will only be useful for some number of Hamiltonians. If a full decomposition is desired, then we will choose only the first Hamiltonian to be 'best' and then apply the zig-zag algorithm from this starting point to ensure that a full decomposition results.

These ideas are put together as the WHam (or weighted Hamiltonian) algorithm outlined below as Algorithm 3.

Algorithm 3 WHam: Weighted Hamiltonian Ordered

Require: A weighted K_n^e.

1: For H_1, find the Hamiltonian (path for even n, cycle for odd n) with the smallest total weight.
2: Let $C(P)$ be a measure of the tendency for the edge weights in a path P to decrease.
3: Using the criterion C, pick the best starting point and path orientation for H_1. (For open paths, there are only two possible starts, for cycles there are n).
4: Apply this node labelling to the other Hamiltonians $H_2, \ldots, H_m$ in the sequence.
5: Use criterion C again to find the best orientation for each of $H_2, \ldots, H_m$.
6: Permute $H_2, \ldots, H_m$ in order of increasing total weight, and relabel the Hamiltonians.
7: **return** $T = H_1 : H_2 : \cdots : H_m$.

Note that line 1 of Algorithm 3 is essentially the "Travelling Salesman Problem" or TSP. While finding the optimal solution is NP-hard, there are many approximate solutions that work well in practice.

6 Concluding remarks

The appearance and resulting interpretation of many data visualizations depends on the ordering of their components. Our goal is to identify good orderings which reveal the data, make large datasets coherent, encourage data comparisons and so promote graphical excellence (Tufte 1987). We approached the ordering problem using graph traversals, and presented algorithms for constructing Hamiltonian decompositions and Eulerians, which enumerate all pairwise comparisons in a systematic way.

In Sections 2 and 4 we explored applications of these methods in data visualization,

devised a new multiple comparisons display which facilitates easy comparison of treatment groups, constructed improved star glyph displays for better visual clustering, and modified parallel coordinate displays to reveal more data patterns. We have also investigated applications to profile glyphs (with results similar to that for star glyphs) and to interaction plots (being essentially a special case of parallel coordinates). More generally, our methods are applicable to any statistical technique or visualization that relies on a particular sequencing of variables, cases or factor levels.

The main drawback to using Hamiltonian decompositions and Eulerians in constructing data visualizations is that the length of the decomposition or Eulerian path is roughly $n^2/2$. Here n is the number of nodes in the graph and in our applications corresponds to the number of variables, or treatments. For some problems, n can be reduced through appropriate dimension reduction methods. In all cases, our solution is to construct Hamiltonians and Eulerians on weighted graphs and in Section 5 we presented new algorithms designed for this purpose. The resulting visualization can then give prominence to relevant features of the data. In an interactive setting, the user could select an interesting data feature or features and immediately zoom in on a subsequence of the associated weighted Eulerian.

In this paper we focused on complete graphs, as these are widely applicable in visualization problems. But we also envisage applications of incomplete graphs. For example consider the canonical correlation setting where there are two groups of variables, $\{X_i, i = 1, \ldots, a\}$ and $\{Y_j, j = 1, \ldots, b\}$ and we wish to construct a parallel coordinate display where X and Y variables appear adjacently. Here we construct a bipartite graph where edges connect X and Y variables only. If a and b are both even an Eulerian exists, and our modified Hierholzer (Algorithm 1) or GrEul (Algorithm 2 for weighted graphs) give a construction. (If a and b are not both even, extra edges must be added to the graph

so that only two vertices are odd.) See Hurley and Oldford (2008a, 2008b) for further examples.

Finally, the algorithms and new graphical displays introduced here are available as the contributed R package `PairViz` (Hurley and Oldford 2008c).

References

Allison, T. and Cicchetti, D. (1976). "Sleep in Mammals: Ecological and Constitutional Correlates", *Science*, 194, pp. 732-734.

Alspach, B, J.-C. Bermond, and D. Sotteau (1990), "Decomposition into cycles I: Hamilton decompositions", in *Cycles and Rays* (eds. G. Hahn, G, Sabidussi, and R.E. Woodrow), Kluwer Academic Publishers, Boston.

Bailey, R.A., M.A. Ollis, and D.A. Preece (2003), "Round-dance neighbour designs from terraces", *Discrete Mathematics*, 266, pp. 9-86.

Bertin, J. (1967), *Sémiologie Graphique*, Édition Gauthier-Villars, Paris.

Bates, D. (1997+), "`TukeyHSD`: Tukey's Honest Significant Difference", *The R Project*, http://www.r-project.org

Brinton, W.C. (1914), *Graphic Methods for Presenting Facts*, Engineering Magazine Co., New York.

Cameron, E. and L. Pauling (1978),"Supplemental ascorbate in the supportive treatment of cancer: Re-evaluation of prolongation og survival times in terminal human cancer", *Proc. Nat. Acad. Sci., USA*. 75, No. 9, pp. 4538-4542.

Colbourn, C.J. (1982), "Hamiltonian decompositions of complete graphs", *Ars Combinatoria*, 14, pp. 261-269.

Fabràga, J. and M.A. Fiol (2004),"Connectivity and Traversability", Chapter 4, pp. 193-339 of *Handbook of Graph Theory* (eds. J.L. Gross and J. Yellen), CRC Press, Boca

Raton.

Fleury (1883), "Deux problèmes de géométrie de situation", *Journal de mathématiques élémentaires*, pp. 257-261.

Friendly, M. and E. Kwan (2003), "Effect Ordering for Data Displays", *Computational Statistics and Data Analysis*, 43, 509-539.

Hahsler, M, Hornik, K and C. Buchta (2008), "Getting Things in Order: An introduction to the R package seriation", *Journal of statistical software*, vol. 25, (3).

Heiberger, R.M., and P. Holland (2006a), "Mean-mean multiple comparison displays for families of linear contrasts", *Jour. of Comp. and Graphical Statistics*, 15, pp. 937-955.

Heiberger, R.M., and P. Holland (2006b), "The `HH` package", http://www.r-project.org.

Hierholzer, C. (1873), "Über die Möglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechung zu umfahren". *Math. Annalen*, VI, pp. 30-32.

Hsu, J. and M. Peruggia (1994), "Graphical representation of Tukey's Multiple Comparison Method", *Journal of Computational and Graphical Statistics*, 3, pp. 143-161.

Hurley, C. (2004), "Clustering Visualizations of Multidimensional Data", *Journal of Computational and Graphical Statistics*, vol. 13, (4), pp 788-806.

Hurley, C. and R.W. Oldford (2008a), "Eulerian tour algorithms for data visualization and the `PairViz` package", (submitted to *Computational Statistics*).

Hurley, C. and R.W. Oldford (2008b), "Graphs as navigational infrastructure for high dimensional data spaces", (submitted to *Computational Statistics*).

Hurley, C. and R.W. Oldford (2008c), "The `PairViz` package", http://www.r-project.org.

Inselberg, A. (1985), "The plane with parallel coordinates", *The Visual Computer*, 1, pp. 69-91.

Inselberg, A. (2009), *Parallel Coordinates: Visual Multidimensional Geometry and its Applications*, Springer, New York.

Lucas, D.E. (1892), *Recréations Mathématiques, Vol. II*, Gauthier Villars, Paris.

McKay, B. D. and R.W. Robinson (1998), "Asymptotic enumeration of Eulerian circuits in the complete graph", *Combinatorics, Probability and Computing*, 7, pp. 437-449.

Peng, W., Ward, M.O. and E.A. Rundensteiner (2004), "Clutter reduction in multi-dimensional data visualization using dimension reordering", IEEE Infovis, pp 89–96.

Reiss, M. (1871-3), "Evaluation du nombre de combinaisons desquelle les 28 dés d'un jeu du domino sont susceptibles d'après la règle de ce jeu", *Ann. Mat. Pura. Appl.*,5, pp. 63-120.

Sloane, N.J.A. (2004), "Sequence A007082" from the *Online Encyclopedia of Integer Sequences*, http://www.research.att.com/∼njas/sequences/.

Tufte, E.R. (1987), *The Visual Display of Quantitative Information*, Graphics Press, CT.

Tufte, E.R. (1991), *Envisioning Information*, Graphics Press, Cheshire, CT.

Wegman, E.J. (1990), "Hyperdimensional data analysis using parallel coordinates", *Journal of the American Statistical Association*, 85, pp. 664-675.

Wilkinson, L. (2005), *The Grammar of Graphics (Second Edition)*, Springer, New York.

Wilkinson, L., Anand, A. and Grossman, R. (2005), "Graph-theoretic scagnostics", *Proceedings of the IEEE Information Visualization 2005*, pp. 157-164.

Wilkinson, L. and M. Friendly (2009), "The History of the Cluster Heat Map", *The American Statistician*, 63(2), pp. 179-184.

Wilkinson, L. and A. Anand (2009), "The `scagnostics` package", http://www.r-project.org.