A New Approach to the Parallel Coordinates Method for Large Data Sets

Norman Matloff* Yingkang Xie†

Abstract

Parallel coordinates is an exploratory method aimed at visualizing interrelations among variables. The concept is highly appealing, but the method becomes difficult or impossible to use when the number of data points n and/or the number of variables p become even moderately large. The former causes the "black screen problem," while the latter makes relations between "distant" axes difficult to discern.

Various remedies such as line density, alpha-blending and axes permutation have been proposed. In this work we present a fresh approach, again based on multivariate density estimation, but in a very different manner: We plot only lines with the highest densities, to have a few "typical" lines in the graph. This solves the large-n problem, and ameliorates the large-p problem. The user may also specify that the lines having the smallest densities be plotted, as a means of detecting outliers. Finally, the user can specify that lines with locally-maximum densities be plotted, with the goal of cluster-hunting. We also present an application to regression diagnostics.

The software uses parallel processing to speed the computation, and is available on CRAN.

Key Words: parallel coordinates, visualization, large data, multivariate density estimation

1. Overview of Parallel Coordinates

Many of today's data sets are large, either in number of observations n or numbers of variables p, so exploratory analysis of interrelationships between the variables is especially challenging. A central problem with visualization in large-p settings is that our displays are two-dimensional, thus set up only for the case $p = 2$.

Various solutions to these problems have been offered; see (Matloff, 2013) for an overview. The particular solution of interest in this paper is *parallel coordinates* (Inselberg, 2009) (Unwin *et al*, 2006). This is actually a very old method introduced back in the 19^{th} century, but not really developed until 1997. Software for the method is widely available, such as in the CRAN packages **lattice**, **MASS** and **GGally**, and now in our new package **freqparcoord**, which takes a novel approach to the problems.

The general method of parallel coordinates is quite simple. Here we draw p vertical axes, one for each variable, For each of our n data points, we draw a polygonal line from each axis to the next. An artificial example is shown in Figure 1, with the axes being height, weight and age.

However, the parallel coordinates method has its own problems. As n grows, one quickly encounters the "black screen" problem, in which large portions of the screen are filled so solidly that discerning relations between the variables is impossible.

*Dept. of Computer Science, University of California, Davis
†Dept. of Statistics, University of California, Davis

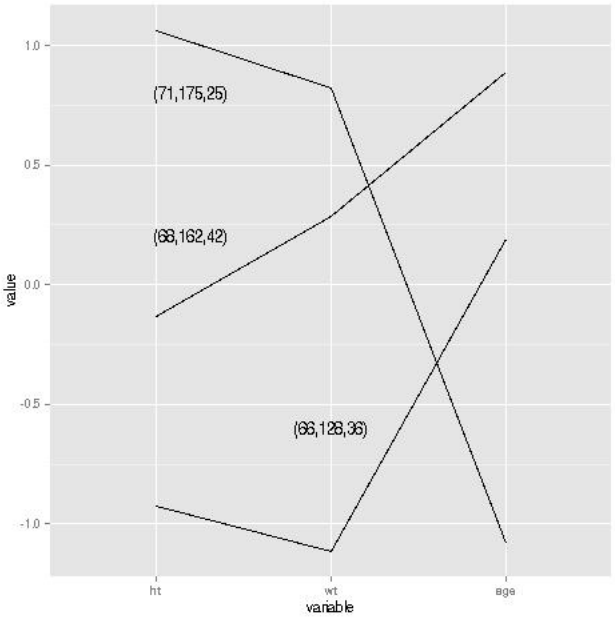

Figure 1: Simple Parallel Coordinates Example

The baseball player data shown in Figure 2[1] shows an example of this, broken down by player position. The space is filled so much by the lines that it's hard to discern much variation in pattern from one position to another.. Note that n here is only about 1200, yet the problem already arises at this small sample size..

One method used to try to avoid the black-screen problem is *alpha blending*, which in essence works by making the lines fainter. Figure 3 shows the result on the baseball data for $\alpha = 0.2$. Although there has been some thinning, the screen is still mostly filled, and thus the problem remains.

Another approach was proposed in (Wegman, 1990). Treating the polygonal lines as random objects, one estimates a "density," analogous to the multidimensional density of a random numeric vector. Related methods have been proposed, such as (Moustafa, 2009), (Alsakran *et al*, 2010) and (Bürgin and Ritschard, 2014) . Though these do not appear to have been widely adopted, in our view this was a step in the right direction, and our own method can be viewed as being related.

2. Frequency-Based Parallel Coordinates

Our CRAN package, **freqparcoord**, presents a novel method for solving the blackscreen problem:

> Display only the most "typical" lines. Here "typical" is defined to be the lines with the highest estimated density in the original (non-line) data.

Our package uses k-nearest neighbor density estimation, but any estimation procedure could be used.

[1] Data courtesy of the UCLA Statistics Department, www.socr.ucla.edu data/mlb.

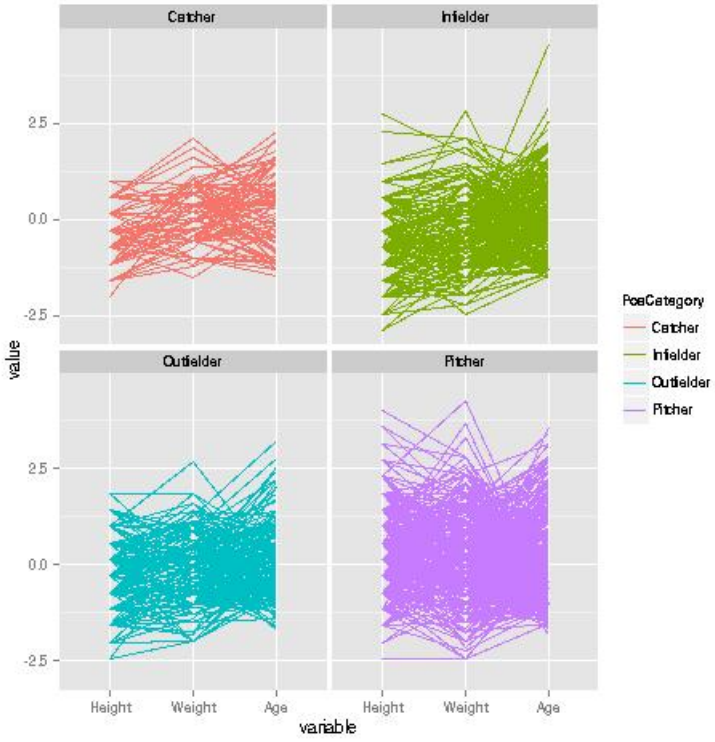

Figure 2: Baseball Player Data

In the baseball data, for instance, we estimate the trivariate density of the original (height, weight, age) data (centered and scaled), and then plot only the m lines corresponding to the m points of highest density. Figure 4 shows the results for $m = 25$.

The figure shows clear differences among the player position categories:

- Pitchers are typically tall, thin, young.
- Catchers typically are much heavier, older.
- Infielders typically shorter, thinner, younger.

As another example, consider the New York City taxi data, http://www.theatlantic.com/video/index/253385/taxi-data-visualization/. Here is a summary of the variables, in the **data** and **fare** data sets:

- **data:** passenger_count, trip_time_in_secs, trip_distance, pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude, pickuptime
- **fare:** fare_amount, surcharge, mta_tax, tip_amount, tolls_amount, total_amount, cmt, crd (paid with credit card), tippc, booltip (tip, yes or no), pickuptime, daytime

Only the first data set will be analyzed here. We used a subsample of 100000 trips.

After removing outliers (Section 3 below), we ran our method with $m = 50$, with results seen in Figure 5. The call was

```
freqparcoord(d100,50,c(8:15)
```

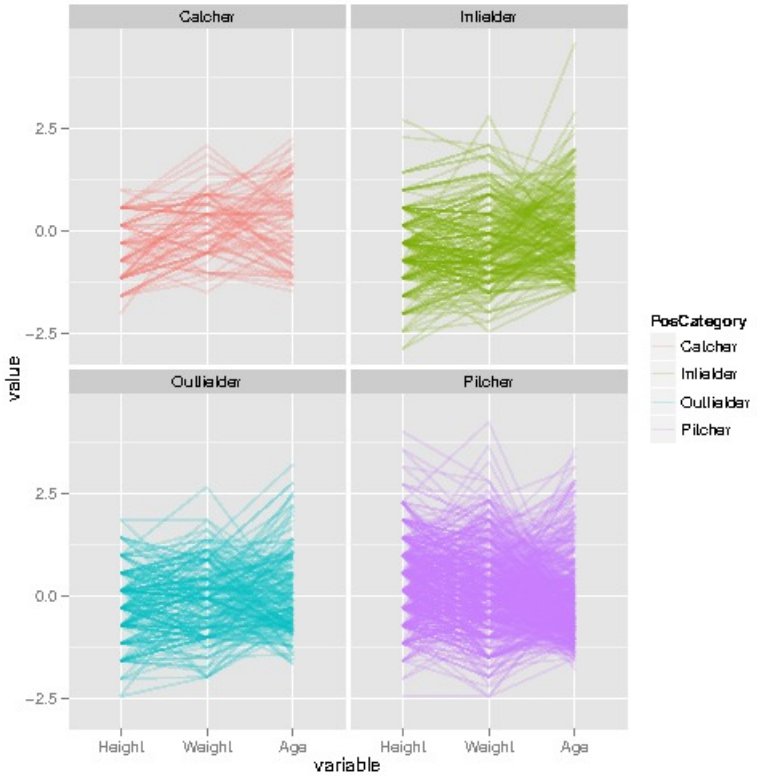

Figure 3: Baseball Player Data, $\alpha = 0.2$

specifying the data frame **d100**, $m = 50$, plotting columns 8 through 15 of the data.

We immediately see at least two strong clusters, largely differing on pickup/dropoff location and time of day.

We also see some more subtle patterns. For instance, there is much more variation in trip time than in trip distance, presumably due to variation in traffic.

As seen with the baseball data, we can break the analysis down by groups. Figure 6 shows a breakdown by number of passengers per trip. Here we see that:

- The 1-passenger trips tend to be early in the day or in the evening.
- Trips with 2-4 passengers tend to be later in the day.
- The 5-6 passengers trips are more diverse in time. These may be families.

3. Outlier Detection

Here, instead of determining the *most* typical lines, we find the *least* typical ones—the outliers. Here is code to find 1 outlier in each of the position categories for the baseball data:

```
p <- freqparcoord(mlb,-1,4:6,7,keepidxs=4)
```

The negative nature of the second argument specifies that we want the points with the smallest density values. Here we are grouping by column 7, and the last argument indicates

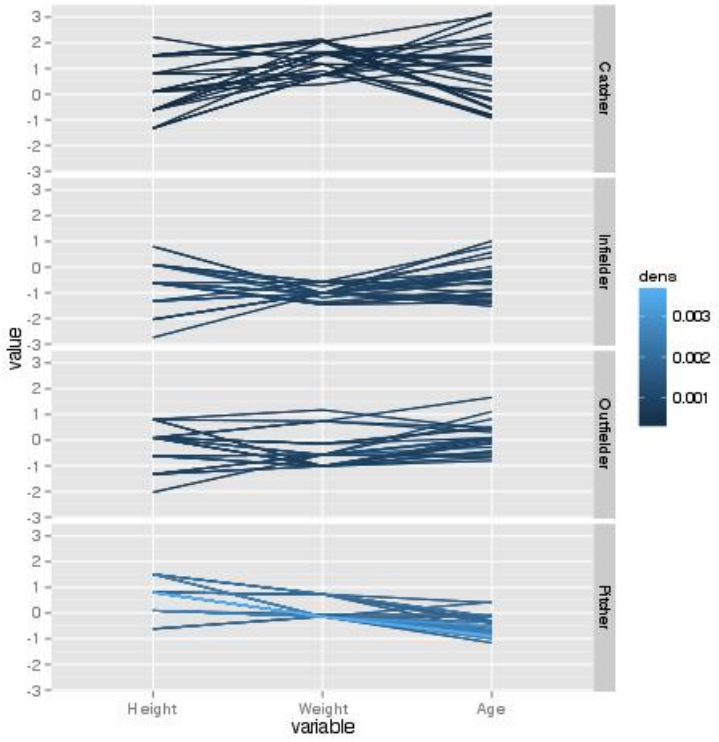

Figure 4: Baseball Player Data, freqparcoord, m = 25

that we wish to retain the row indices of the selected data points (sorting them on column index 4). We can then display the graph by printing **p** and **p$xdisp**. The output of the latter is

Name	Team	Position	Ht	Wt	Age	Pos
Julio_Franco	NYM	First_Baseman	73	188	48.52	Infielder
Barry_Bonds	SF	Outfielder	74	228	42.60	Outfielder
A._Pierzynski	CWS	Catcher	75	245	30.17	Catcher
Randy_Johnson	ARZ	Start_Pitcher	82	231	43.47	Pitcher

Franco is an interesting case, 48 years old!

4. Cluster Hunting

This is specified in **freqparcoord** by setting the **locmax** option, with the name alluding to the fact that the algorithm searches for points whose estimated density is largest among its **klm** neighbors (optional argument).

To see how this works, here is a simulation from a distribution with known clusters:

```
# generate simulated bivariate data with clusters at
# (0,0), (1,2), (3,3), see whether "locmax" (clustering)
# picks them up
cv <- 0.5*diag(2)
# rmixmvnorm() included with freqparcoord pkg
x <- rmixmvnorm(10000,2,3,list(c(0,0),c(1,2),c(3,3)),list(cv,cv,cv))
p <- freqparcoord(x,m=1,method="locmax",keepidxs=1,k=50,klm=800)
```

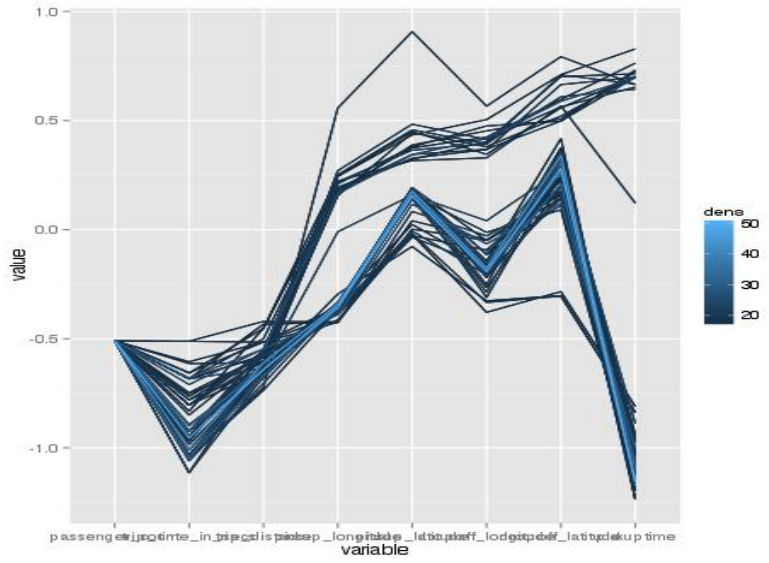

Figure 5: Taxi Data, freqparcoord, m = 50

```
p$xdisp
             [,1]      [,2]
[1,] -0.169502  0.2464784
[2,]  1.138110  2.2674989
[3,]  3.102322  2.8854155
```

The method worked well here, picking up the correct clusters.

Real data, of course, is messier, and we should expect less-crisp results—but still useful ones. For instance, the graph produced by the call

```
freqparcoord(d100,1,c(8:15),method="locmax",klm=1000,
    cls=cl4,keepidxs=15)
```

on the taxi data is shown in Figure 7. (The meaning of the **cls** argument will be explained in Section 6.)

Here the clusters seem to form on the basis of short vs. long trip distance, pickup/dropoff location, and time of day. These might be expected *a priori*, but a surprising pattern is seen in the two peaks in the top-center of the graph. After investigating the particulars of those lines (via inspection of the **xdisp** component of the output), one finds that in essence, those two lines reflect a "changing of the guard."

- Around 1:45 p.m., there are a number of trips from mid-Manhattan to La Guardia Airport.
- Around 7:30 p.m., the reverse occurs, with many passengers traveling from La Guardia Airport to mid-Manhattan.

This is a good example of the principle that variables should be investigated jointly, not just one at a time.

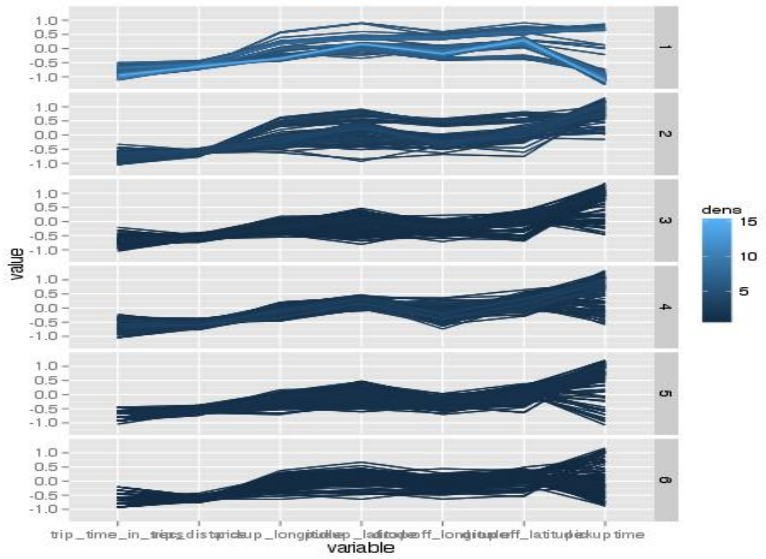

Figure 6: Taxi Data, Grouped by Number of Passengers

5. Regression Diagnostics

As an application of our method, we present an approach to regression diagnostics.

Here the vertical axes represent the predictor variables, with one additional axis representing the *divergences*, defined by

divergences = fitted parametric model - fitted nonparametric model

where again our package uses k-nearest neighbors (regression instead of density here) to calculate the nonparametric fit.[2] A function **regdiag()** included with the package looks at typical values among the most negative and most positive divergences. In other words: **regdiag()** asks, "In what region[s] of predictor space is the fit poorer?"

For example, here is an analysis of salaries of programmers and engineers, Silicon Valley, in the 2000 Census.[3]

```
> data(prgeng)  # built-in data set
> pg1 <- prgeng
> pg1$ms <- as.integer(pg1$educ == 14)  # MS
> pg1$phd <- as.integer(pg1$educ == 16)  # PhD
> pg1$se <- as.integer(pg1$occ==102)  # software engineer.
> l1 <- lm(wageinc ~ age+ms+phd+se+sex,data=pg1)
# look at 40% most neg., 40% most pos. divergences
> p <- regdiag(l1,tail=0.40)
> p  # display graph
> p$paramr2  # parametric adj. R2
[1] 0.07027561
> p$nonparamr2  # nonparamr2 R2
[1] 0.1286746
```

[2]Note that the divergences are not the residuals.

[3]5% PUMS file.

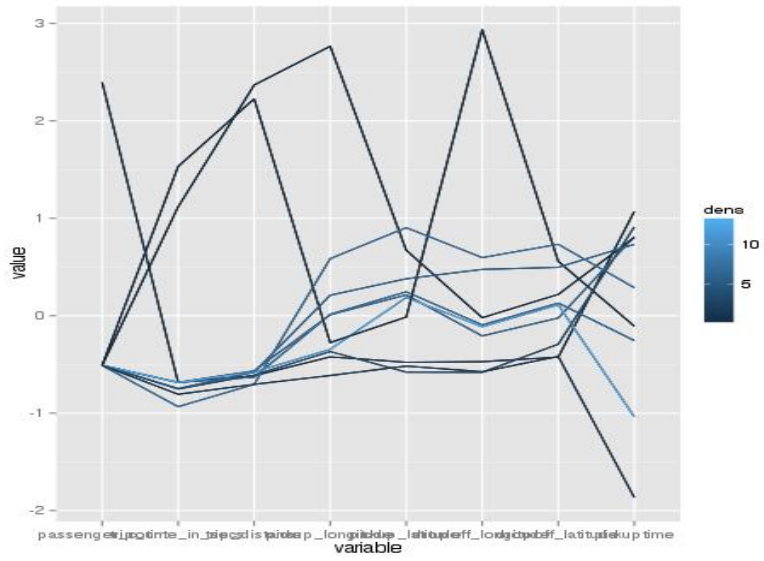

Figure 7: Taxi Data Clusters

The graph is shown in Figure 8. Before commenting on it, we first call the reader's attention to the printed output above. The adjusted R^2 valued is 0.07. This is low, but the point here is that the corresponding value for the nonparametric fit is nearly double, 0.13. This indicates that the parametric model is insufficient.

And indeed, Figure 8 shows that the most negative divergences, i.e. the worst cases of underprediction, tend to occur for the older workers, while the younger workers tend to be overpredicted. This suggests adding an Age^2 term to the parametric model. After doing so (not shown here), the adjusted R^2 matched that of the nonparametric fit.

6. Parallel Computation

The k-nearest neighbor computation can be voluminous. This is partly addressed by using the **FNN** package on CRAN for fast nearest-neighbor computation, and further addressed by giving the user the option of using the built-in R package **parallel**. With the latter, the data is divided into chunks, and the nonparametric fit is done only within chunks (Matloff, 2014), resulting in strong speedups. The argument **cls** is a cluster built under the **parallel** package.

7. Discussion

Our method here has tuning parameters that must be chosen by the user:

- the number of "typical" lines **m**
- the number of nearest neighbors **k**
- in the clustering case, **klm**, the number of points used to determine a local maximum

Many statistical methods involve tuning parameters, but it is important to distinguish between those that are inferential or at least devoted to finding single point estimates and those

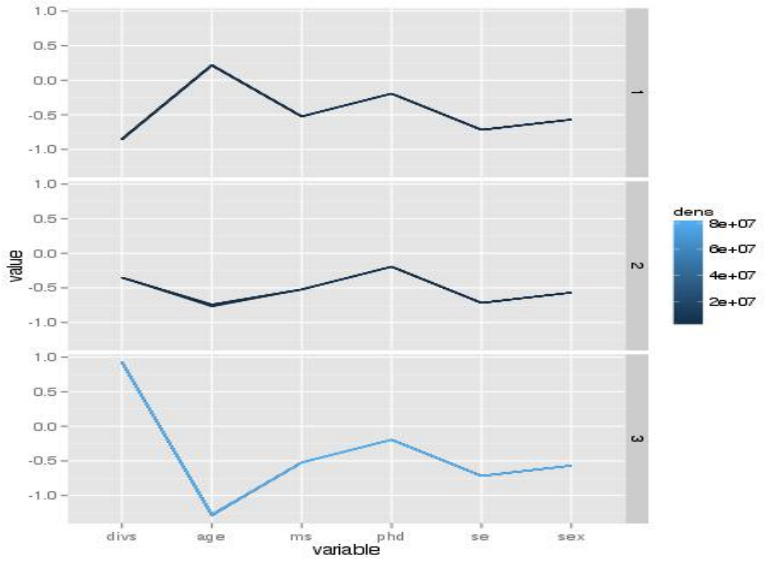

Figure 8: PUMS Regression Diagnostics

that are exploratory in nature. Our method, and of course parallel coordinates methods in general, are in this latter category.

We recommend that the user try various values for these parameters on any given data set.

In Section 1, we cited two problems with the parallel coordinates method for multivariate visualization. The first problem involves the "black screen" problem, caused by excessive clutter and overplotting that arise with even moderately large values of n . Our approach to that problem is to plot only the most "typical" lines (or in the otulier-hunting case, the least typical ones). In this way, we avoid clutter, no matter how large n is.

The second problem we cited is that it is visually difficult to correlate axes that are separated widely on the screen. As mentioned, this is often handled by allowing the user to interactively permute the order of the axes. We add that because our method plots only a few lines to begin with, visual correlation of the various axes is much easier. Permutation and other methods can still be used, but by plotting only a few lines, the problem is already ameliorated.

REFERENCES

Alsakran, J, Zhao, Y. and Zhao, X. (2010). " Tile-Based Parallel Coordinates and Its Application in Financial Visualization," *VDA 2010*.

Bürgin, R. and Ritschard, G. (2014). "A Decorated Parallel Coordinates Plot for Categorical Longitudinal Data," *The American Statistician*, 68, 2, 98-103.

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

Matloff, N. (2013). "Long Live (Big Data-fied) Statistics!" (invited paper), *Proceedings of JSM 2013*, 98-108.

Matloff, N. (2014). *Software Alchemy: Turning Complex Statistical Computations into Embarassingly-Parallel Ones*, http://arxiv.org/abs/1409.5827.

Moustafa, R. (2009). "QGPCP: Quantized Generalized Parallel Coordinate Plots for Large Multivariate Data Visualization," *Journal of Computational and Graphical Statistics*.

Unwin, A., Theus, M. and Hofmann, H. (2006). *Graphics of Large Datasets: Visualizing a Million*, Springer, 2006.

Wegman, E. (1990). "Hyperdimensional Data Analysis Using Parallel Coordinates," *JASA*, 85, 411, 664-675.