**Parallel Coordinates Plot Graphical User Interface (PCP GUI) Manual**

**Acknowledgements**

Support for this project was provided by the Natural Environment Research Council [Consortium on Risk in the Environment: Diagnostics, Integration, Benchmarking, Learning and Elicitation (CREDIBLE) grant number NE/J017450/1].

**Contents**

[1 Intro 2](#_Toc365563121)

[1.1 Objectives 2](#_Toc365563122)

[1.2 Requirements 2](#_Toc365563123)

[1.4 Background; Parallel Coordinate Plots (PCPs) 2](#_Toc365563124)

[2 Functionality 3](#_Toc365563125)

[2.1 Main 3](#_Toc365563126)

[2.2 Groups 6](#_Toc365563127)

[2.3 Clusters 7](#_Toc365563128)

[2.4 Output 8](#_Toc365563129)

[2.5 States 9](#_Toc365563130)

[3 Getting Started: Countries example 10](#_Toc365563131)

[3.1 Getting the correct format for .mat files: 10](#_Toc365563132)

[3.2 Beginning the program 11](#_Toc365563133)

[4 Further Functions and Analysis: Hydrological example 12](#_Toc365563134)

[4.1 Main Functions 13](#_Toc365563135)

[4.2 Groups 14](#_Toc365563136)

[4.3 Clusters 17](#_Toc365563137)

[4.4 Density Plots 18](#_Toc365563138)

[4.5 Filtering 19](#_Toc365563139)

[4.6 Clusters (cont.) 20](#_Toc365563140)

[4.7 Further analysis 22](#_Toc365563141)

[4.8 Conclusion 22](#_Toc365563142)

[5 Further reading and possible code extension 23](#_Toc365563143)

[5.1 Further reading 23](#_Toc365563144)

[5.2 Code improvement 23](#_Toc365563145)

[5.3 Further functionality 25](#_Toc365563146)

[6 References 26](#_Toc365563147)

[6.1 Journal papers, conference proceedings and books 26](#_Toc365563148)

[6.2 Datasets 26](#_Toc365563149)

[6.3 Functions 26](#_Toc365563150)

# 1 Intro

## 1.1 Objectives

The objective of the PCP GUI is easy production and visual manipulation of parallel coordinate plots with little MATLAB knowledge.

## 1.2 Requirements

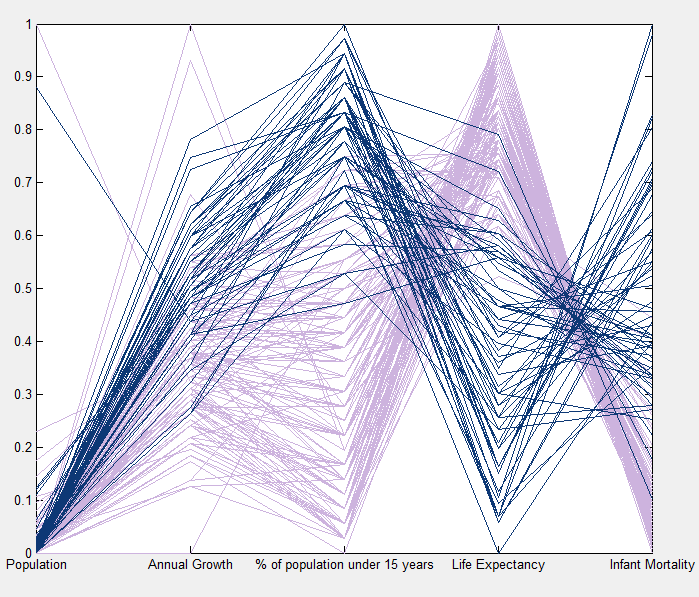
Knowledge: how to start and run a MATLAB .m file, basic understanding of the MATLAB workspace.

Toolboxes: MATLAB (7.14), Statistics Toolbox (8.0).

## 1.4 Background; Parallel Coordinate Plots (PCPs)

PCPs are a graphing method to allow the display of high dimensional datasets in one plot. This is done with an aim to quickly see global trends. Further analysis of the plots can provide useful and unanticipated insights.

Each axis on a PCP represents one variable. A member of the dataset has a value for each variable. These values are joined by a polyline to represent that member across all variables. Data are represented by multiple polylines on the plot.



**X**

Figure 1.1: Parallel Coordinate Plot (Countries dataset)

Figure 1.1 takes population, growth, age, life expectancy and mortality of countries as its variables. For example the polyline marked with an ‘**X**’ is obviously China as it has the largest population of all the countries. This line can be followed across variables to find China’s respective values for each variable. In this diagram all variables have been scaled such that the largest value in the dataset takes 1 and the smallest 0. Thus a 1 on the population axis represents about 1.3 billion people while a 1 on the life expectancy axis represents about 83 years. This scaling is common practice (Few, 2009) (unless all variables are of the same unit) and is used in the PCP GUI.

Figure 1.2 uses easily relatable scatterplots and Pearson’s r values to show how correlation between two variables is displayed by PCPs. It should be noted that PCPs are not a useful tool for putting two variables under the microscope to investigate correlation. Figure 1.3 demonstrates the need for PCPs by comparing a scatterplot matrix and a PCP displaying the same data.

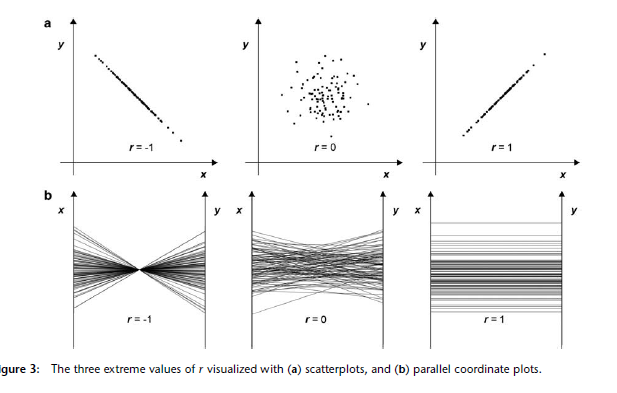
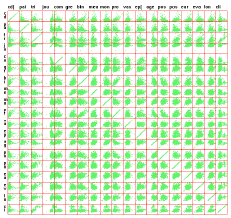
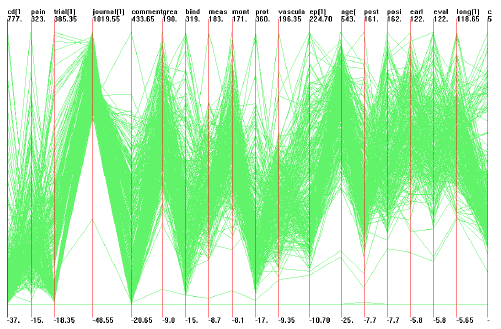


Figure 1.2: Scatterplot-PCP comparison (Li *et al*, 2010)

Figure 1.3: Scatterplot matrix-PCP comparison (Zhao and Kaufman, 2012)



# 2 Functionality

There are many various visual manipulations one may wish to perform on a PCP to better investigate or display features of the data. The following is a list of the GUI’s current functions.

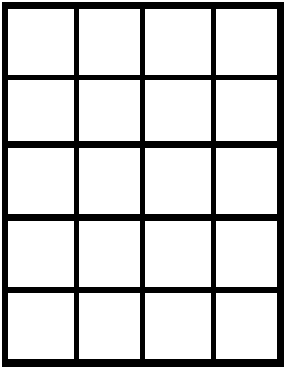
## 2.1 Main

### 2.1.1 ‘Browse for dataset’

Before use of the program starts the dataset must be loaded. The dataset should be in the format shown in Figure 2.1 (variables as columns and members as rows). Its format should be a .mat numerical array (see ‘3.1 Getting the correct format’ to create this from excel).

Figure 2.1: Dataset input format

**V1**



**V2**

Dimensions

**X1**

**X2**

**X3**

Data

NOTE:

* The current code also cannot process empty cells/cells containing string.
* The raw data will be scaled to the largest and smallest member of each dimension as described earlier.
* Currently the maximum number of dimensions is 20. This is due to limitations on space. Although this could be increased, too many dimensions will make presentation unclear.

Upon loading the dataset the program’s interface will become visible and the loaded file will be displayed. The remaining input/main panel controls will be described in (roughly) order of use.

### 2.1.2 ‘Browse for axes labels’ and ‘Browse for axes units’

The axes are given numeric labels by default, units are given blanks by default; these can be changed by clicking on the label/unit display box and typing. Alternatively the labels/units can be loaded from a .mat file. The labels/units must be stored in a cell row vector containing the labels/units as strings shown in Figure 2.2 (see ‘3.1 Getting the correct format’ to create this from excel).

If the number of members of the labels/units vector is not equal to the number of dimensions in the original dataset then the vector will not load and an error message will come up.

**N1**

**N2**

**N3**

Names

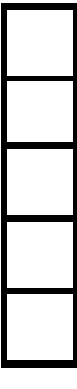


Figure 2.3: Data names input format

**L1**

**L2**

Labels/units



Figure 2.2: Axes labels and units input format

### 2.1.3 ‘Browse for data names’

Data names are used when group/cluster data is output and when using the data cursor to click on polylines. The data names will be numeric (according to position in matrix) by default. Data names can be loaded from a .mat file. The names must be stored in a cell column vector containing the names as strings as shown in Figure 2.3 (see ‘3.1 Getting the correct format’ to create this from excel).

If the number of members of the names vector is not equal to the number of members in the original dataset then the vector will not load and an error message will come up.

### 2.1.4 ‘Plot raw data’

This checkbox allows toggling of whether to plot every member in the dataset. If this is checked and groups/clusters exist then these will be plotted on top of the raw data. Note that if clusters exist and none are hidden then (as they contain every member between them) the plot raw data checkbox will make no difference.

If you wish to edit the visual properties of raw data (colour or line width and style) you can do so by using groups (see ‘2.2.2 Group Visual Toolbar’).

### 2.1.5 ‘Axes order’ and ‘Optimise order’

Changing the order of the axes can improve the presentation of a dataset (Zhao and Kaufman, 2012). A user may wish to have the more important variables at one end or group variables in some order of significance.

The GUI also offers an ‘optimise order’ function. Using random swap (Yang *et al,* 2003) an approximate order is chosen aiming to maximise the correlation between neighbouring variables.

Pressing the ‘+’ and ‘-‘ buttons allows the user to change the order in which the dimensions are plot. The dimension with order 1 will be the leftmost axis. ‘Optimise order’ tries a number of random combinations of axes, adding together the total Pearson’s r between each two neighbours for all variables. The combination with the highest total correlation is the chosen order.

NOTE:

* The ‘Optimise Order’ function will generally **not** provide the **best** order, only a close match, not every combination can be tested.
* If a user is willing to take longer for calculations in order to get a better match they should change the number of iterations equation on line 1346 in the pcpnew.m main file.

### 2.1.6 ‘Flip axis’

If an axis is flipped all data for that variable are plotted with negative values (i.e. 0 is the smallest value and -1 the largest rather than +1). This can be used to highlight a particularly important variable or trend (Moustafa, 2002).

It should be used with caution; flipping reverses and distorts the typical correlations shown in Figure 1.2, often making the data seem more correlated than it is, see Figure 2.4. Each axis with the flip axis checkbox selected will be flipped.

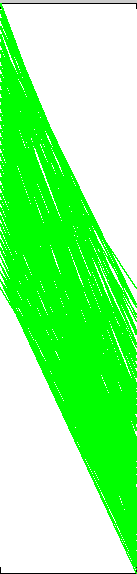


Figure 2.4: Flip-skew (Countries dataset) colours show the new and old position of polylines and their values

Flipping

Weak negative correlation

Strong positive correlation???

**r=-0.78**

### 2.1.7 ‘Display original values’

One axis may be selected to be the Y-scale when the figure is updated. This will change the numbers on the y-axis from ‘0 – 100%’ to the ‘minimum – maximum value for that variable’.

### 2.1.8 ‘Delete dimension’

If it becomes apparent that a variable is no longer necessary for investigation it can be deleted by the ‘delete dimension’ button.

NOTE:

* Once deleted a dimension cannot be reloaded.
* Deleting a dimension will remove all current clusters and groups.

### 2.1.9 ‘Display max/min values’

Toggling the display max/min values will cause the graph to display the maximum value for each dimension along the top x axis and the minimum value for each dimension along the bottom x axis.

## 2.2 Groups

### 2.2.1 ‘Create/Edit Group’

Pressing the button create/edit group will open the group specification panel. This allows you to toggle between the group types (range and rank) and set the group’s name (by default it is numeric) and specification. Once a specification has been set, the ‘save and close’ button must be pressed to continue.

Values for each variable can be specified and all members within the specified range become part of a group. For example if ‘countries with a population over 70% (1 billion people)’ was specified, the group would contain India and China. If specifications are made in multiple variables, members of the group must meet them all. Upon plotting, if no members of the dataset fit a range specification then an error message will show but the plot will continue.

A user may be searching for a certain type of datum fitting some criteria. Using the ‘Rank similarity’ group setting they can input these specifications for variables and choose the members of the group to be the nearest n polylines which match their criteria. This function will simply choose those polylines which have the shortest (scaled) distance to the initial criteria. Future additions may wish to allow weighting importance of certain variables.

### 2.2.2 Group visual toolbar

Upon creation of a group it will appear in the group panel with visual settings which can be edited for plotting. ‘Edit group’ calls a similar panel to the create group (as with created groups, for changes made in edit group the ‘save and close’ button must be pressed). ‘Remove group’ simply deletes the group.

If you wish to edit the colour and line properties of the *raw data*, simply create a group and leave all the specifications as ‘Not Specified’. This will create a group containing all dataset members but with editable visual.

Figure 2.5: Group visual toolbar, becomes available upon group creation



### 2.2.3 ‘Hide group’

This checkbox will hide the group’s members in the plot, although if the members are plotted by some other group (e.g. the plot raw data checkbox or another group) they will still appear but in that group’s format.

### 2.2.4 ‘Group colour’

Clicking the colour button of the group will bring up a standard colour palette which can be used to set the group’s colour.

### 2.2.5 ‘Envelope transparency’

By moving the envelope transparency slider off its default position (left hand side) means the group will be plot as an envelope from the highest member of the group to the lowest for each dimension rather than individual lines (currently if you want both you must create two groups).

### 2.2.6 ‘Line style/width’

Standard MATLAB line styles and widths can be set by the dropdown menu.

## 2.3 Clusters

To split the data into clusters simply select a number of clusters from the dropdown menu. This will automatically use a k-means method to create selected ‘k’ clusters. This iterative method seeks to minimise distances of cluster members to the centroid of the cluster. The optimal number of clusters button will show you a graph (similar to figure 2.6) of iterations by the k-means method. From this graph you can see the benefits of additional clusters. Choose as few clusters as possible with a small average distance to centroids (in the case of Figure 2.6, probably 3).

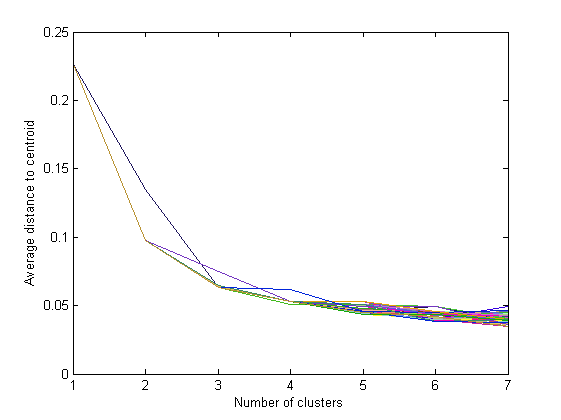


Figure 2.6: Optimal number of clusters (Countries dataset)

Alternatively if your dataset contains binary or categorical dimensions it is suggested that the k-medoid (Chen, 2010) radio button is selected. This function should not be used unless necessary as the clusters it produces generally have larger average distances to centroids.

NOTE:

* As the optimal number of clusters function iteratively loops a k-means function it may take some time, progress will be updated in the display panel and can be cancelled.

### 2.3.1 Clusters visual toolbar/’noise reduction’

Controls on the clusters toolbar behave the same as the group toolbar with the addition of the ‘noise reduction’ checkbox.

Checking noise reduction will plot in bold (or enveloped if selected) the upper and lower quartiles for the group and the members which are within these quartiles for EVERY dimension (in this respect for small datasets the noise reduction button can be restrictive).

## 2.4 Output

### 2.4.1 ‘Create/Update figure’

To plot the current settings a MATLAB figure created by the GUI must exist. If no such figure is open then an error will be returned. If a figure has already been created and plotted upon then another may still be created, allowing comparison of figures. Only the most recently created figure may be updated.

### 2.4.2 ‘Filter/return original data’

Pressing ‘filter data’ will allow the user to set a range specification of members to be eliminated from the dataset. The scale will change upon plotting if the highest or lowest member of a dimension has been removed. Return original data reloads the starting dataset.

This is a good method for the user to eliminate outliers which skew the visual of a certain variable as the scale is readjusted to the new dataset.

NOTE:

* Currently if data is filtered and filtered again the previous dataset (rather than the original) will be reloaded. (This can be changed by setting the backed up data in ‘browse\_data\_callback’ rather than ‘filter.m’)
* Both buttons remove all groups and clusters from the program.

### 2.4.3 ‘Output group/cluster data’

This button allows the user to select a file name and location to save data in an Excel or text file (Pfister, 2009). The first Excel sheet or text file (C1) will contain the names of members of each cluster. The second Excel sheet or text file (C2) contains the standard deviation, average distance to centroid and centroid location of each cluster. The third Excel sheet or text file (G1) contains the members of each group.

### 2.4.4 Density plots

If there are so many data in the set that patterns cannot be distinguished as the appearance of the polylines becomes homogenous a density plot can be used. The ‘XRes’ allows the amount of pixels between two dimensions to be chosen and ‘YRes’ the total pixels on the Y axis. Selecting checkbox log scale will increase sensitivity.

NOTE:

* The default settings are a high resolution log plot as shown in Figure 2.7; this may take a few minutes to plot. Progress will be updated in the display panel. For better performance decrease the resolution.
* This function cannot yet be combined with any groups or clusters.

## 2.5 States

States can be loaded and saved using the load and save state button. Although all the interface/groups/clusters will be the same upon reloading, any open figures will not be reloaded. Figures can be saved separately on the figure toolbar as per usual in MATLAB.

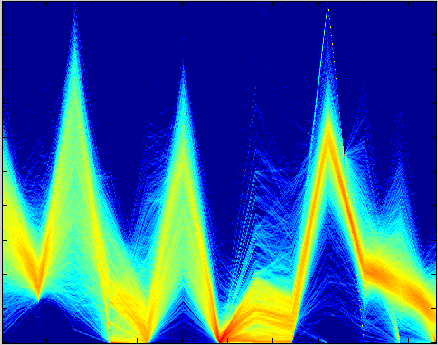
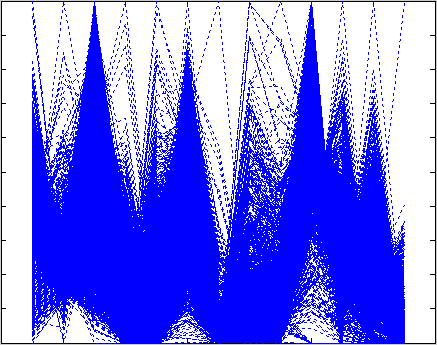


Figure 2.7: Normal plot to density plot (Hydrological dataset)

**Examples**

# 3 Getting Started: Countries example

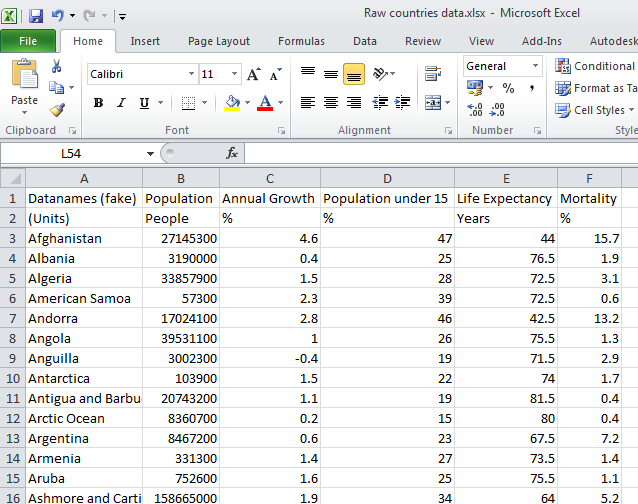
This example introduces the basics for using the PCP tool using the Countries datasets:

* Inserting raw excel data into MATLAB in the correct format.
* Starting the program and loading in data.
* Plotting data.

## 3.1 Getting the correct format for .mat files:

In the ‘Countries’ section of the ‘Examples’ folder, open the excel file ‘Raw countries data’. It should look like Figure 3.1. (Please note that the values for this dataset are out of date and the country names assigned to each row of data are incorrect).

Figure 3.1: Excel format of Countries dataset



From this raw data, four .mat files must be created. Using the function xlsread is the easiest way to do this and should be compatible with at least MATLAB 2008a. Copying and pasting the code in Figure 3.2 will let you browse for the excel file and import the data easily.

[fileName pathname]=uigetfile({'\*.xlsx';'\*.xls'},'Select Excel file');

fullname=fullfile(pathname,fileName);

[num,txt]=xlsread(fullname);

Figure 3.2: Import from excel table code

### 3.1.1 Dataset

As specified earlier, the dataset requires rows to be members and columns to be variables: the format of this excel dataset. (If you are importing an Excel file which has members as columns and rows as variables you should use the code: >>num=num'; to transpose the matrix).

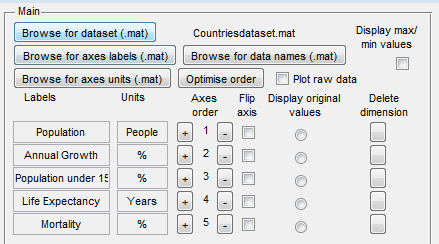
The ‘num’ matrix is the correct format for the dataset so can simply be saved from the workspace to the Countries (or wherever) folder.

Note that in this example you should check that the matrix is a <198x5 double>.

### 3.1.2 Other data information

The ‘txt’ matrix is the correct format for the remaining data matrices, but their ranges must be specified. Type the following commands to specify each:

Figure 3.3: The ‘Main’ panel



>> labels=txt(1,2:6);

>> units=txt(2,2:6);

>> names=txt(3:200,1);

Each of these matrices must be saved from the workspace with appropriate names.

Note that in this example the labels/units matrix should be a <1x5 cell> and the names matrix should be a <198x1 cell>.

## 3.2 Beginning the program

To run the program: open the ‘pcpnew.m’ file and press run or ‘F5’. This will open a GUI where ‘Browse for dataset (.mat)’ is the only available uicontrol. Click this and browse in ‘Examples’ and ‘Countries’ for the ‘Countriesdataset.mat’ you saved earlier, this will turn on the remaining main interface.

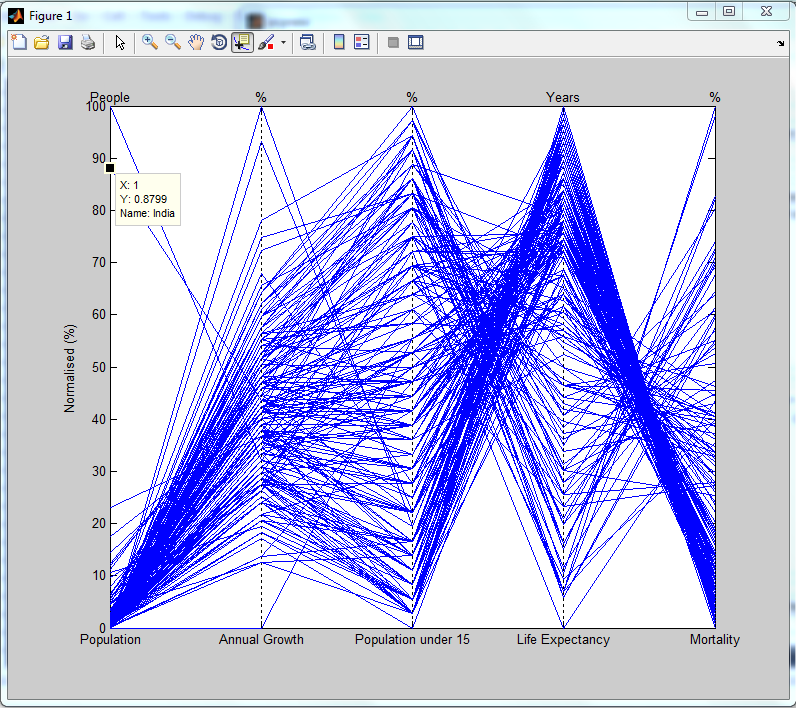
Use the other ‘Browse for..’ buttons to browse for the other respective .mat files. You should end up with a Main panel looking like Figure 3.3.

First, check the ‘Plot raw data’ checkbox. Create **then** update a new figure on the ‘Figures part of the ‘Output’ panel.

If everything is working then a plot should appear looking like Figure 3.4. All the basic MATLAB figure options apply. Additionally the datacursor option will display the associated member name from the browsed data names, as pictured.

For additional functionality and how to perform analysis please see ‘4 Further Functions and Analysis’.

Figure 3.4: A PCP diagram of the Countries dataset

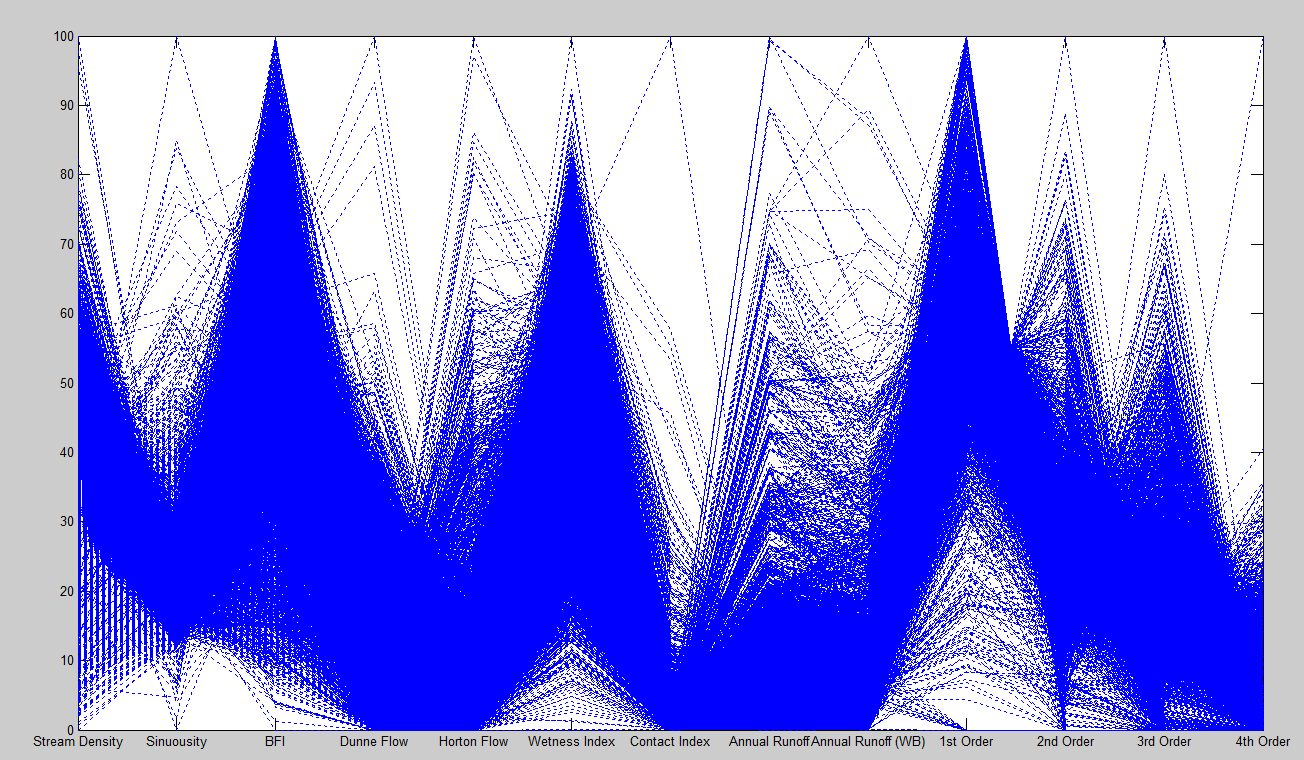


# 4 Further Functions and Analysis: Hydrological example

This example uses the remaining PCP tool functions and gives an example of how to use parallel coordinate plots for analysing data.

Begin the program and browse for the .mat files as described in section 3.2 except within the ‘Hydrological’ example folder instead of ‘Countries’. Plot the raw data to check the dataset has correctly loaded, see Figure 4.1. Note you may have to stretch the figure to display the labels correctly. Performance for this dataset will be decreased as it has over 6000 members; **wait for visible confirmation that your control action has been updated before clicking another.**

Figure 4.1: A PCP diagram of the Hydrological dataset



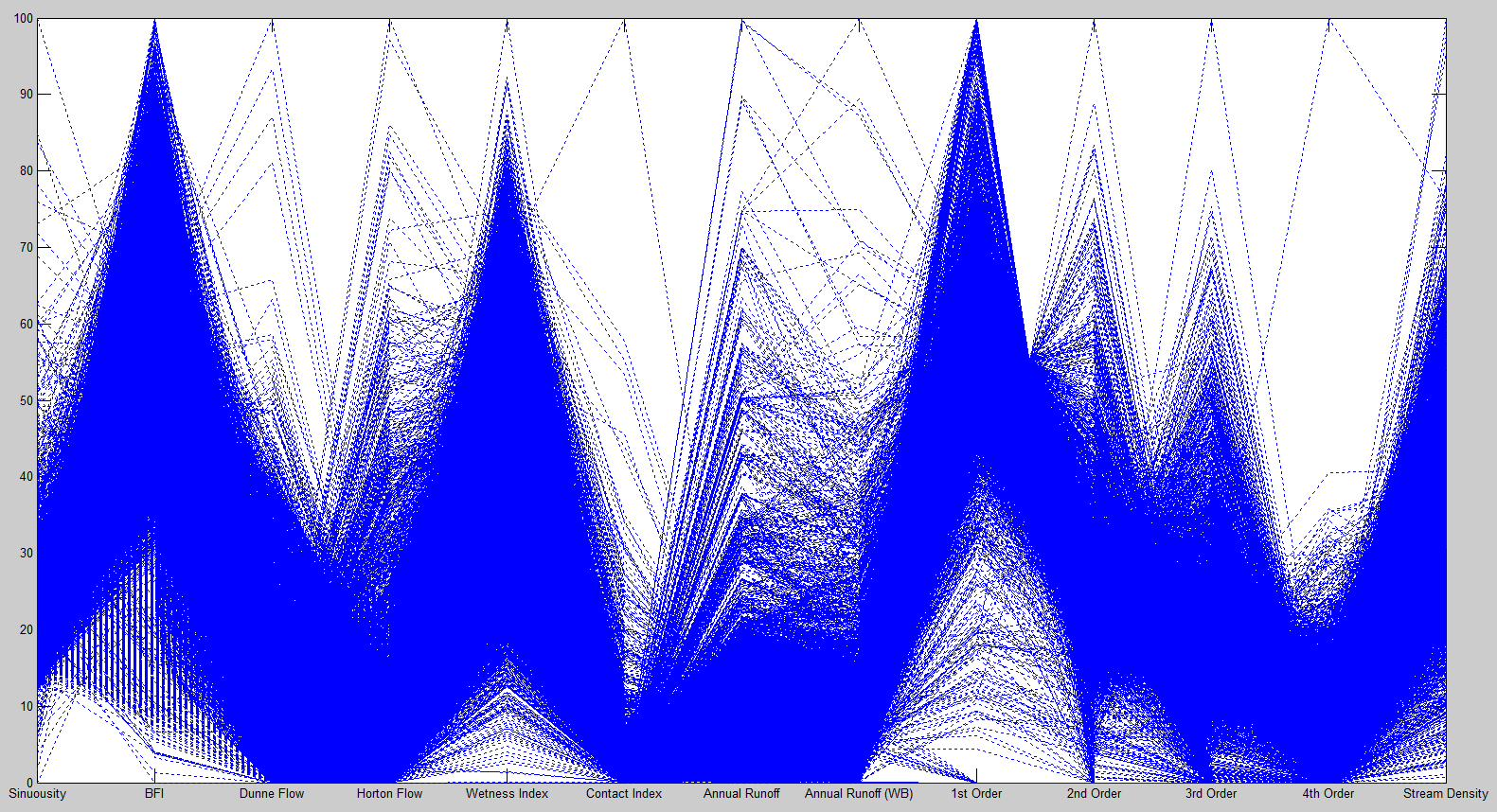
## 4.1 Main Functions

Use the ‘+’ and ‘-‘ controls on ‘axes order’ to move the ‘Stream Density’ axis to the end of the plot. Try updating the new figure to see the effects, Figure 4.2. Axes order can easily be controlled in this manner.

The ‘optimise order’ button will automatically select an order which has high similarities between neighbouring axes. For smaller datasets, such as the Countries Example, the order is likely to be identical (or reversed) every time the button is pressed. For larger datasets, such as Hydrological, you are only likely to see repeated pairs (such as 11-1 and 3-5) in the same order.

For information on flipping and displaying original/max/min values read their sections in 2.1.6, 2.1.7 and 2.1.8.

Figure 4.2: Axes can be moved simply using the axes order buttons



## 4.2 Groups

For analytical purposes, especially with such large datasets as the Hydrological Example, it can be difficult to see anything obvious from the raw data. Thus it may be useful to check reasonable assertions. For example: ‘The less runoff in a catchment the fewer high order streams there are’.

To start, we would like to create a group specifying low runoff. Press the ‘create new group’ button in the group panel. This will open a group GUI. To start with we will simply find the lowest runoff catchment. Click the ‘rank similarity’ button to enter the ranking interface, this will let us specify a criteria and find the best matched members, as shown in Figure 4.3.

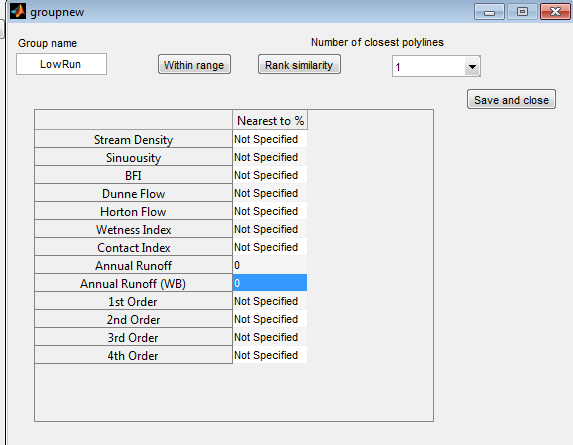
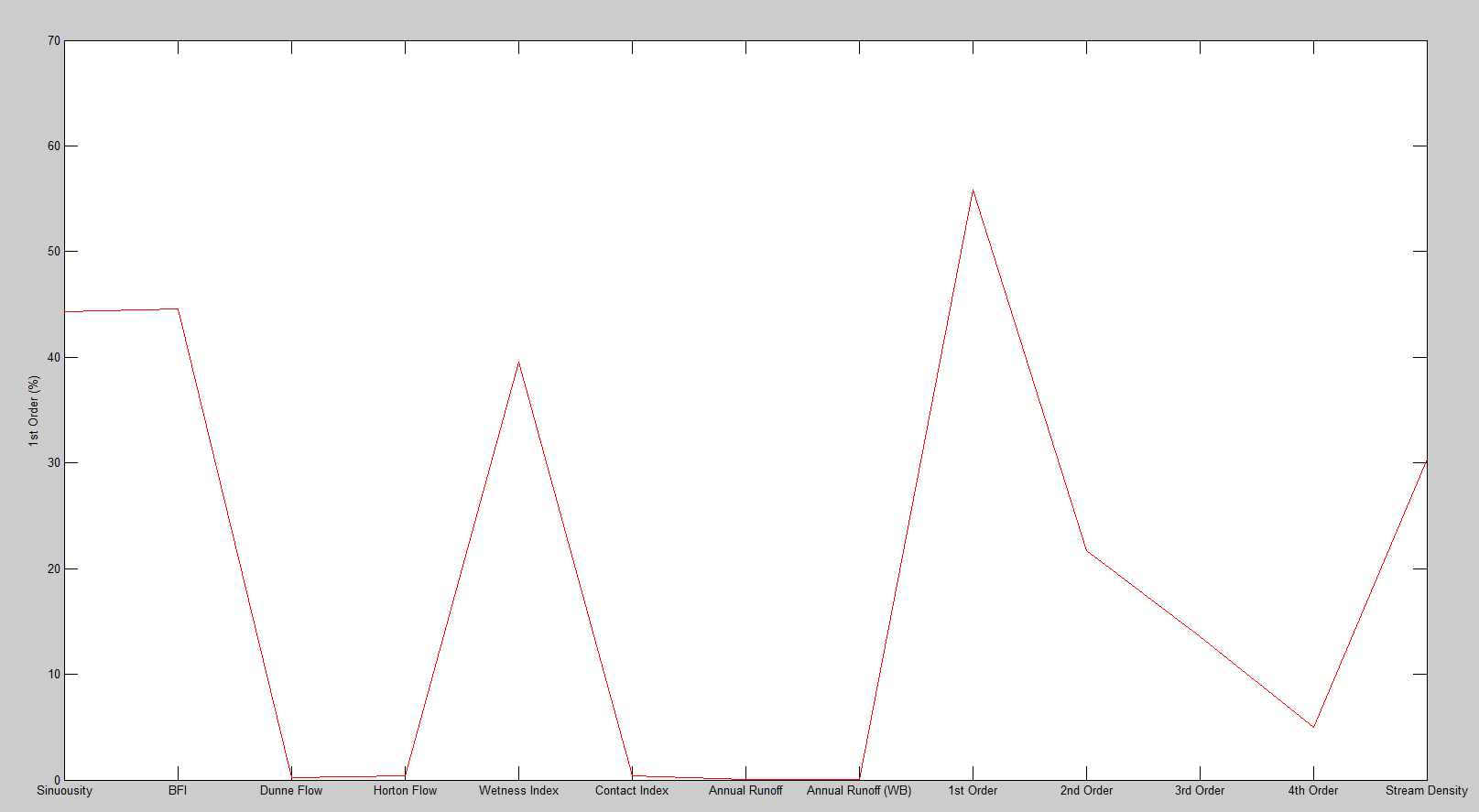


Figure 4.3: The ‘Rank similarity’ group interface

0% Annual Runoff and Annual Runoff (WB) have been specified and we have selected to choose 1 closest match. Click ‘Save and close’ to create the group.

Creating a new figure and updating it (Figure 4.4) shows us that, as suggested, the lowest runoff catchment has a high % 1st order streams and less high order streams, decreasing with order.

Figure 4.4: The polyline of the lowest runoff member in the Hydrological dataset



However we must check that this is not an anomaly, edit the low runoff group by clicking the ‘edit group’ button and change it into a range by clicking the ‘within range’ button. Select this range as less than 1% for both runoffs as shown in Figure 4.5.

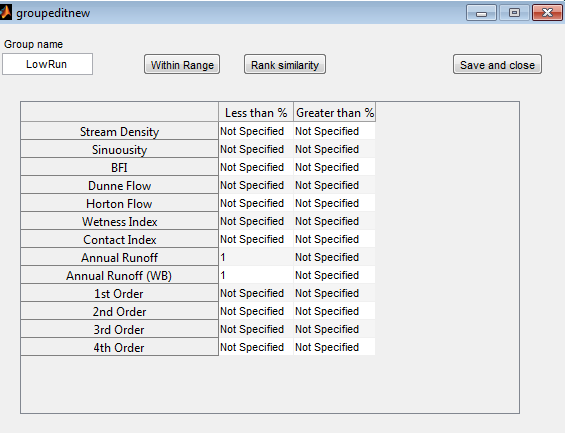


Figure 4.5: The ‘Within range’ group interface

Updating the plot shows us that the overall trend is there but it is much less distinct, Figure 4.6a. Checking the ‘plot raw data’ checkbox in the main panel allows us to compare the group with the rest of the data. By inspection the trend looks even less convincing, Figure 4.6b!

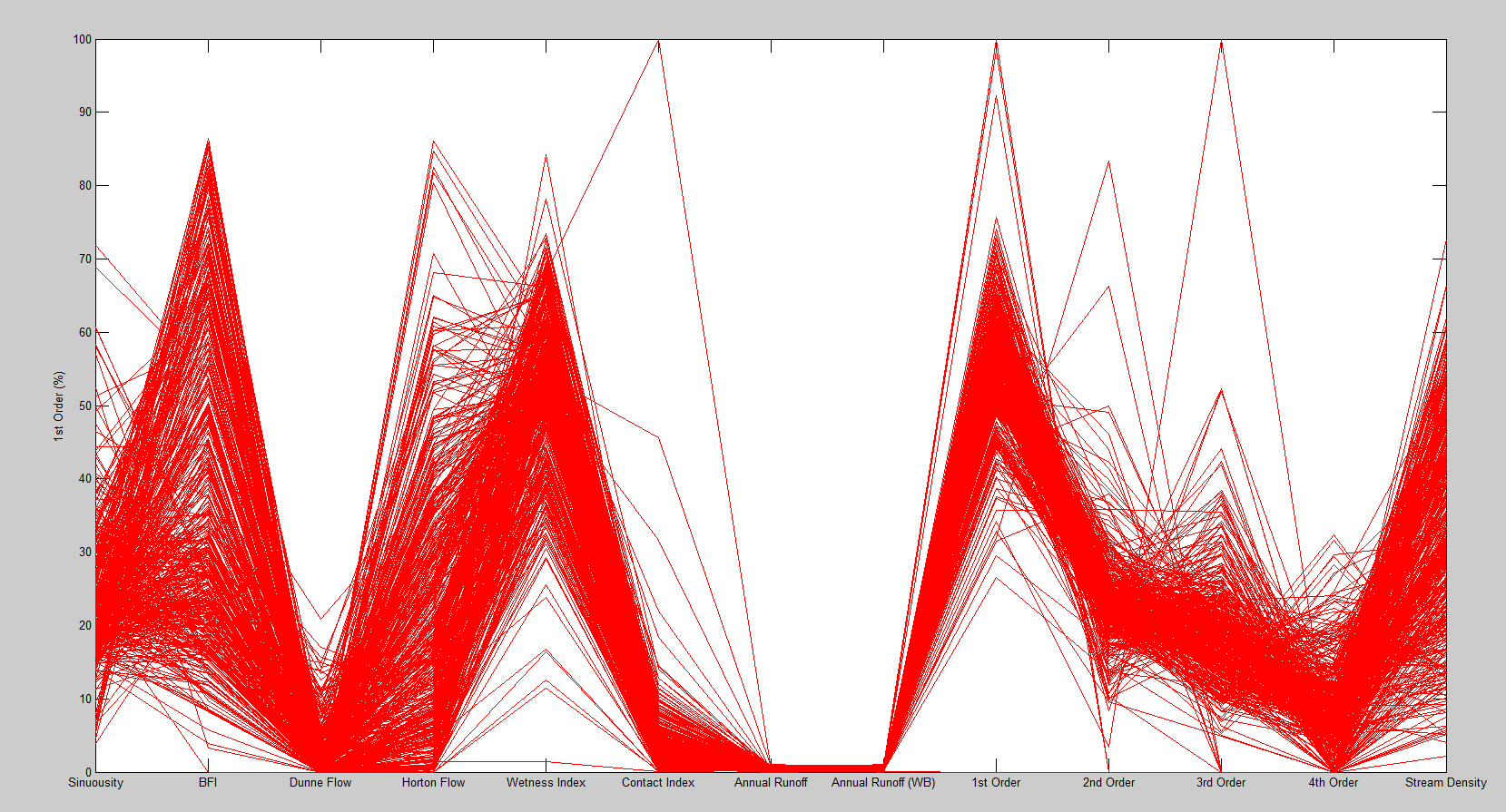
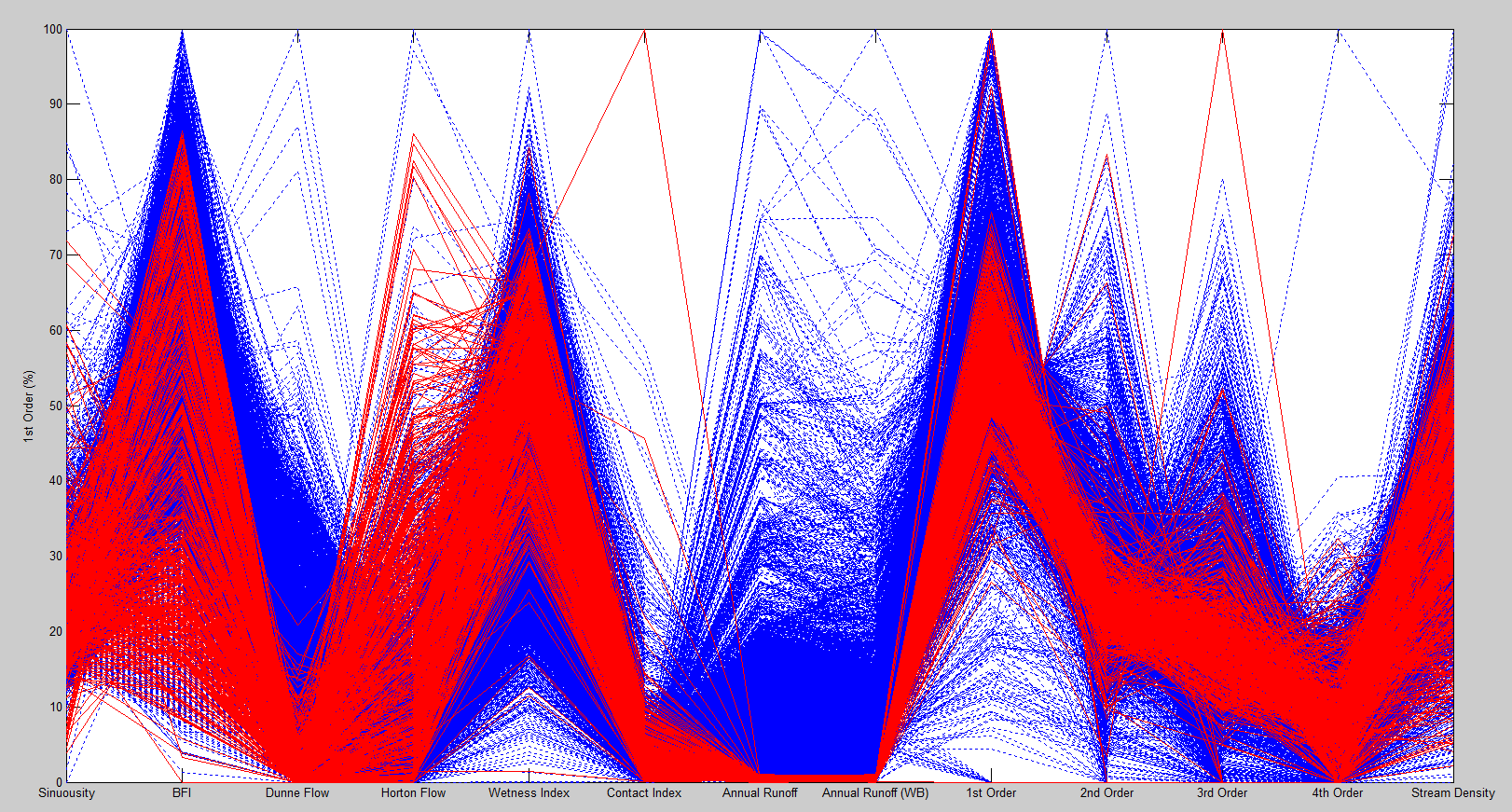


Figure 4.6a (left): Less than 1% runoff members, 4.6b (right): Comparison with raw data

To be sure of this ‘non trend’, let us create another group for high runoff catchments. We can see that the vast majority of runoffs are below 30% so another ‘Within Range’ group specifying groups with greater than 30% runoff should be created.

To differentiate between the two groups, press the ‘Group colour’ button for the second group to change its colour and update the figure.

Figure 4.7: The ‘Rank similarity’ group interface

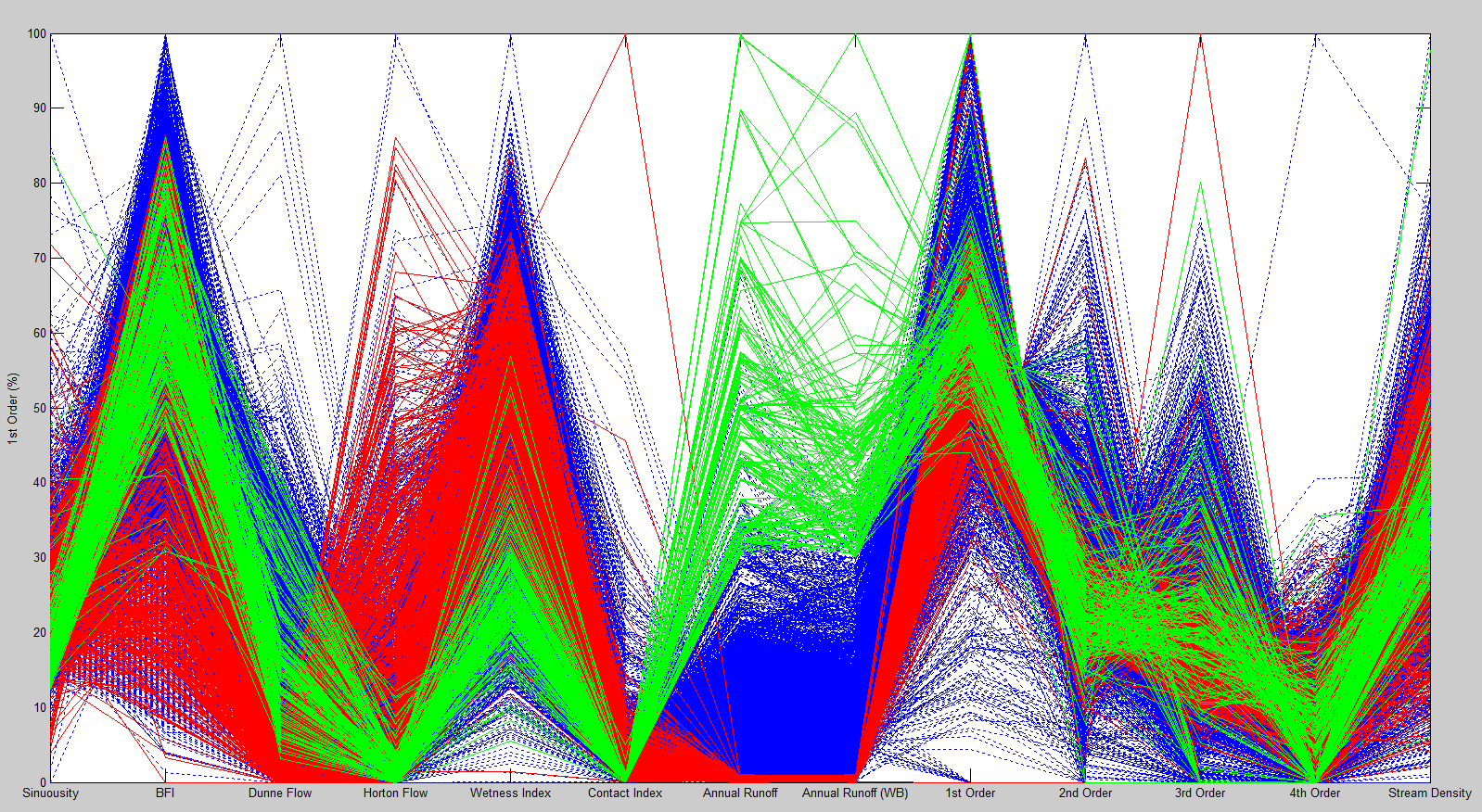
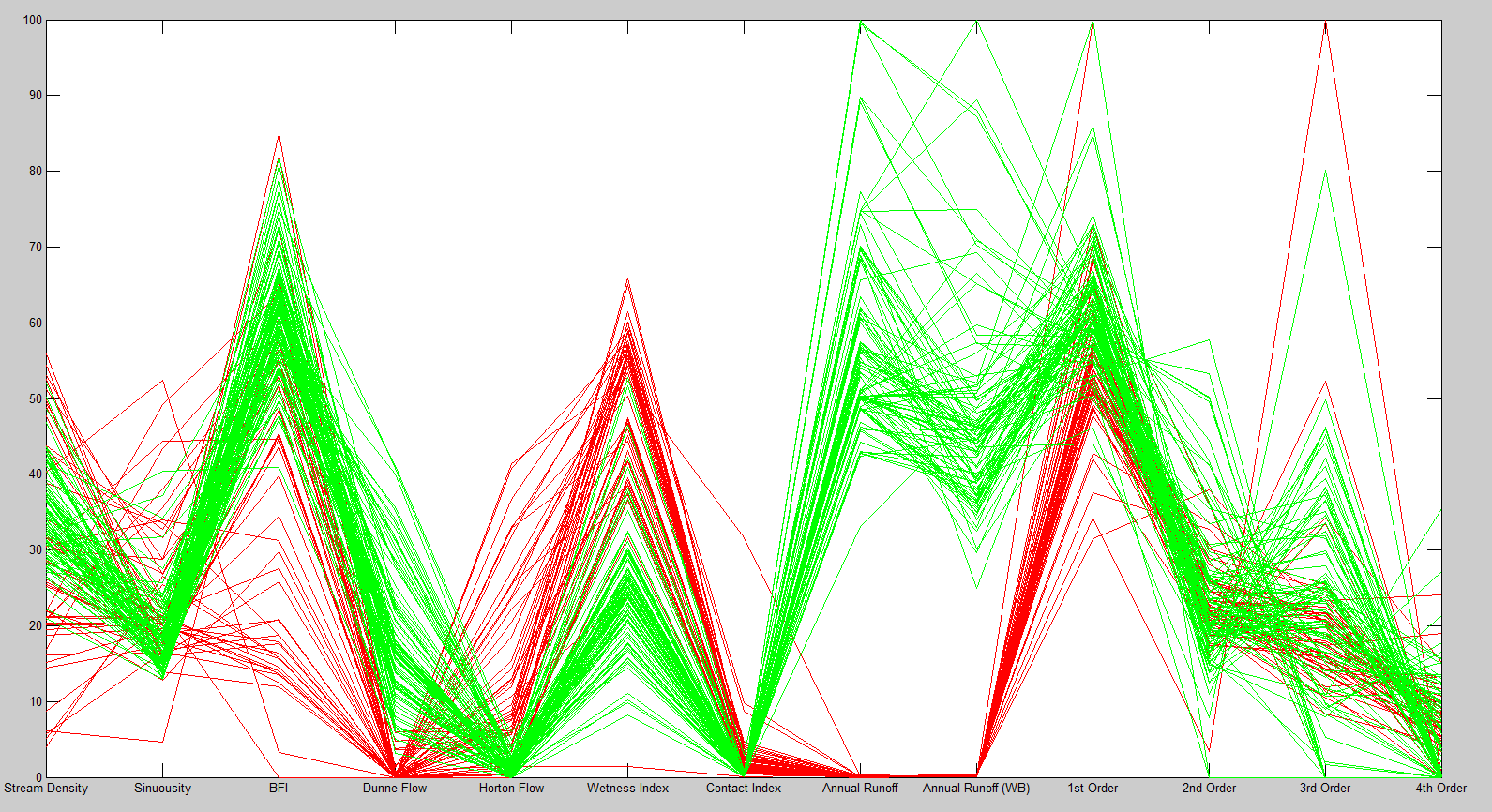


Figure 4.7 shows clearly that there is little or no distinction between high and low runoff groups when it comes to stream orders. We do, however, observe that the lower runoff groups appear to have on average a lower BFI, a lower Dunne flow, a higher Morton flow and a higher wetness index than higher runoff groups.

Would we expect these relationships or are there just too much data confusing our perception?

Try editing the groups into ranks and specifying the 50 nearest polylines each to the max and minimum runoff as this will contain less members and thus be easier to interpret.

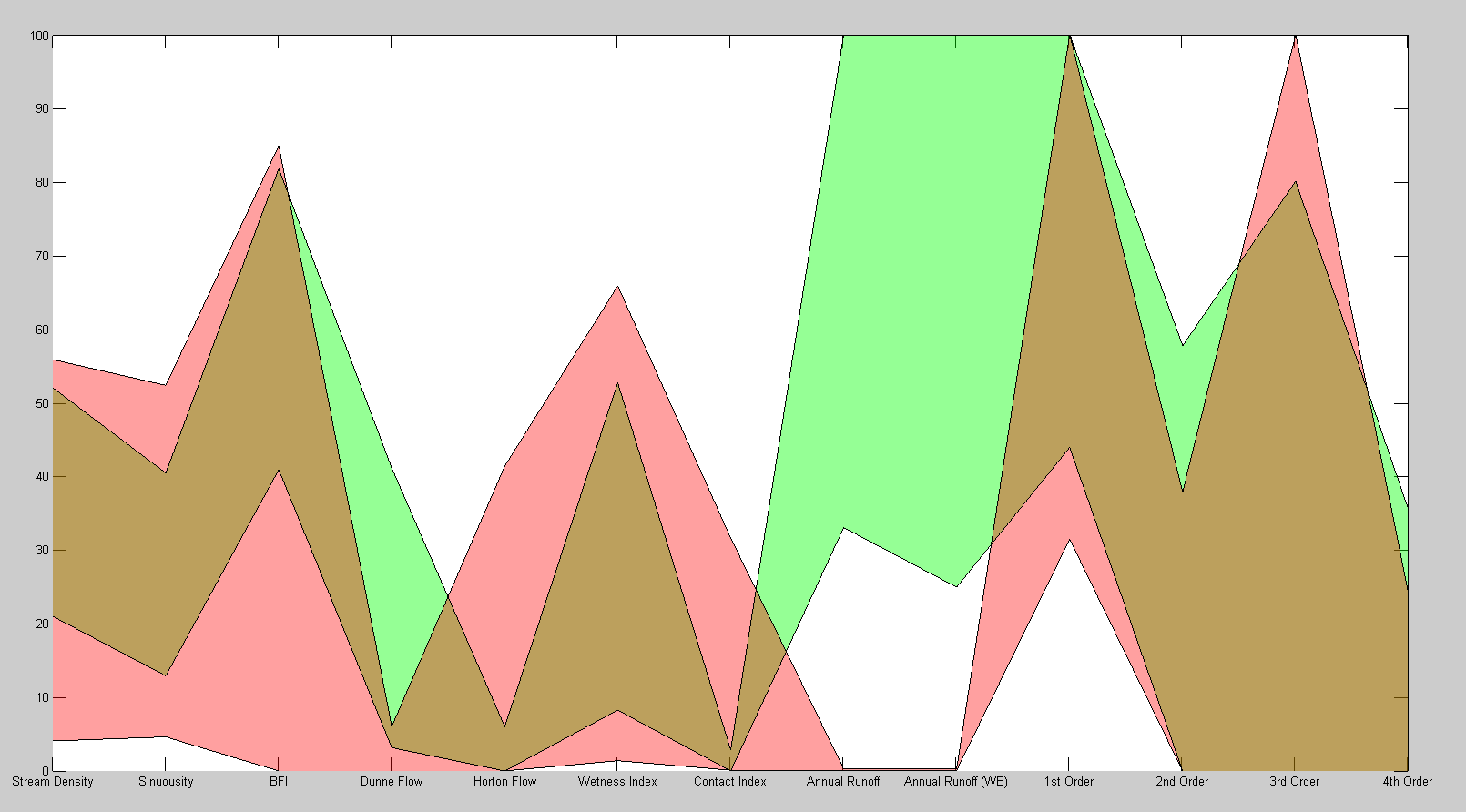
Figure 4.8: Using rankings rather than ranges can help reduce data density



We can see from figure 4.8 that few high runoff catchments have below 10% Dunne Flow and below 30% wetness index but are consistently lower than 10% for Horton flow.The opposite appears to be true for lower runoff catchments with a particularly small band in Dunne Flow.

An envelope can be placed over the highest and lowest value for each dimension within each group by using the ‘Envelope transparency’ slider. This helps show that although the statement: low runoff is correlated to low BFI may not be true, we can see that the spread of BFI values is much greater for low runoff while high runoff BFI’s are more concentrated at a higher value.

Figure 4.9: The slider creates an envelope around the max and min values for each variable



## 4.3 Clusters

While actively hunting for patterns as in the groups section above may work for some datasets, we have seen that it can be a long and iterative process. Using a k-means clustering algorithm to split the data into clusters automatically may be more useful. Try creating two clusters using the ‘Select number of clusters’ dropdown menu. Update the figure to see these effects, Figure 4.10 (note that the colours may be inverted in your plot, this is due to the random nature of k-means).

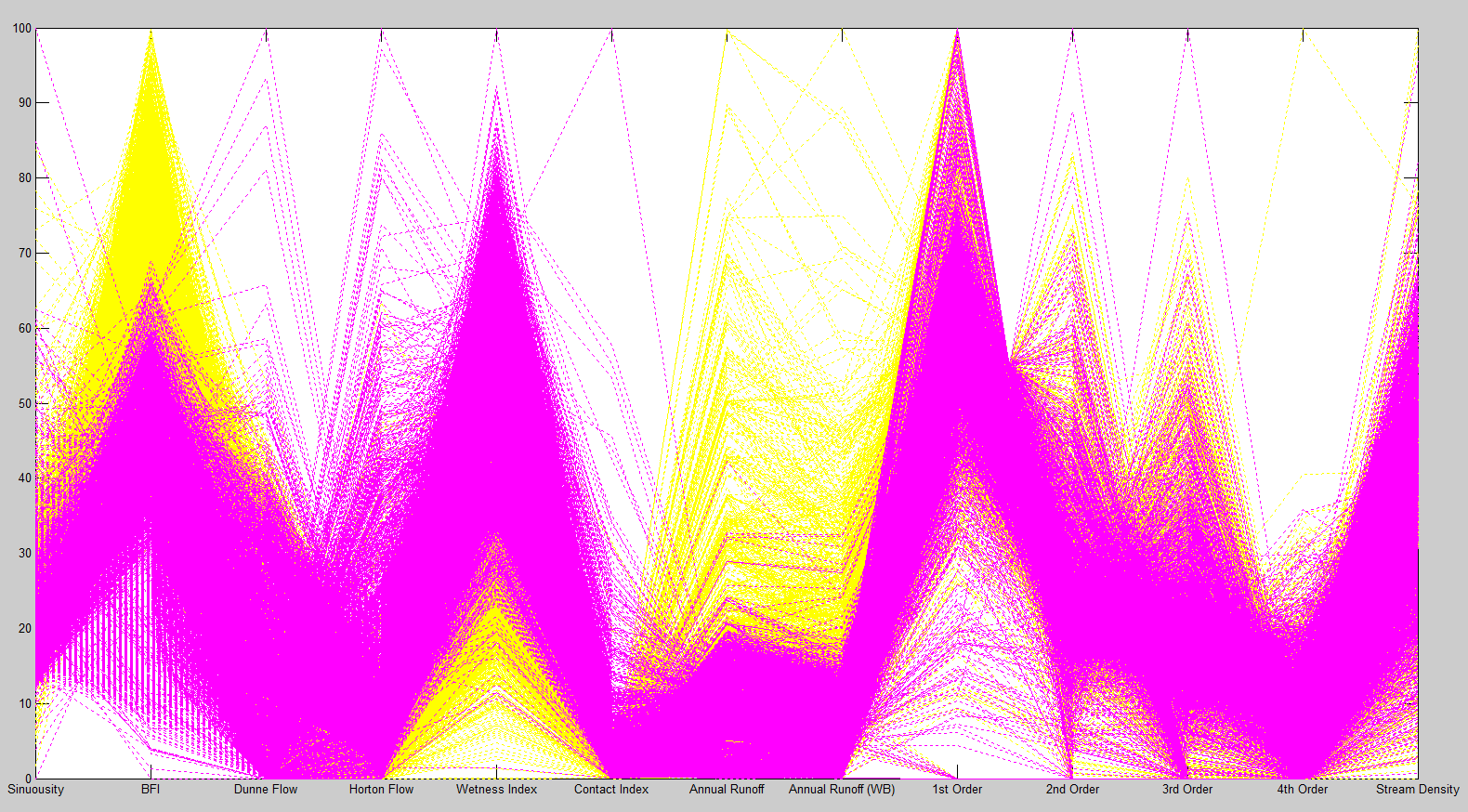
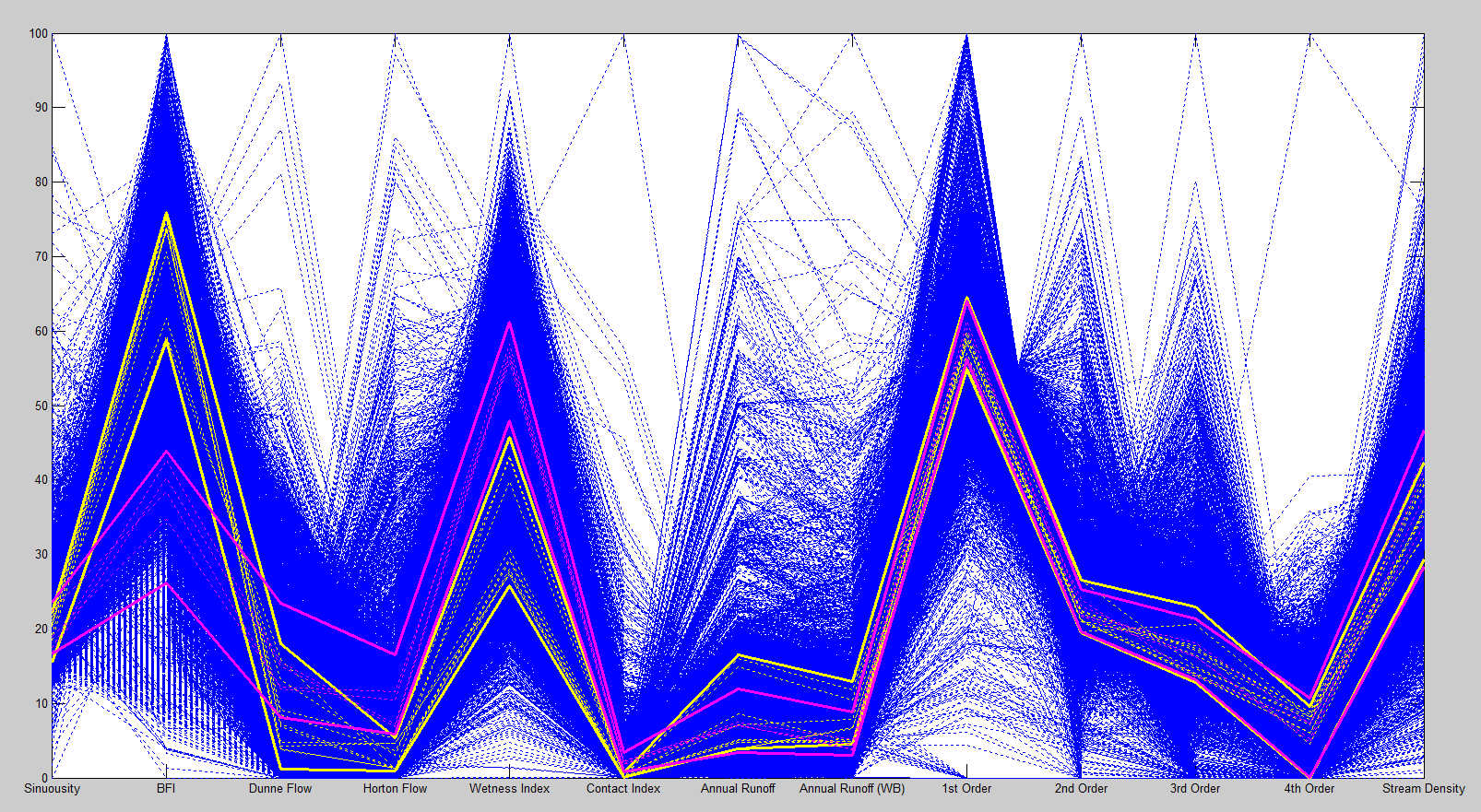
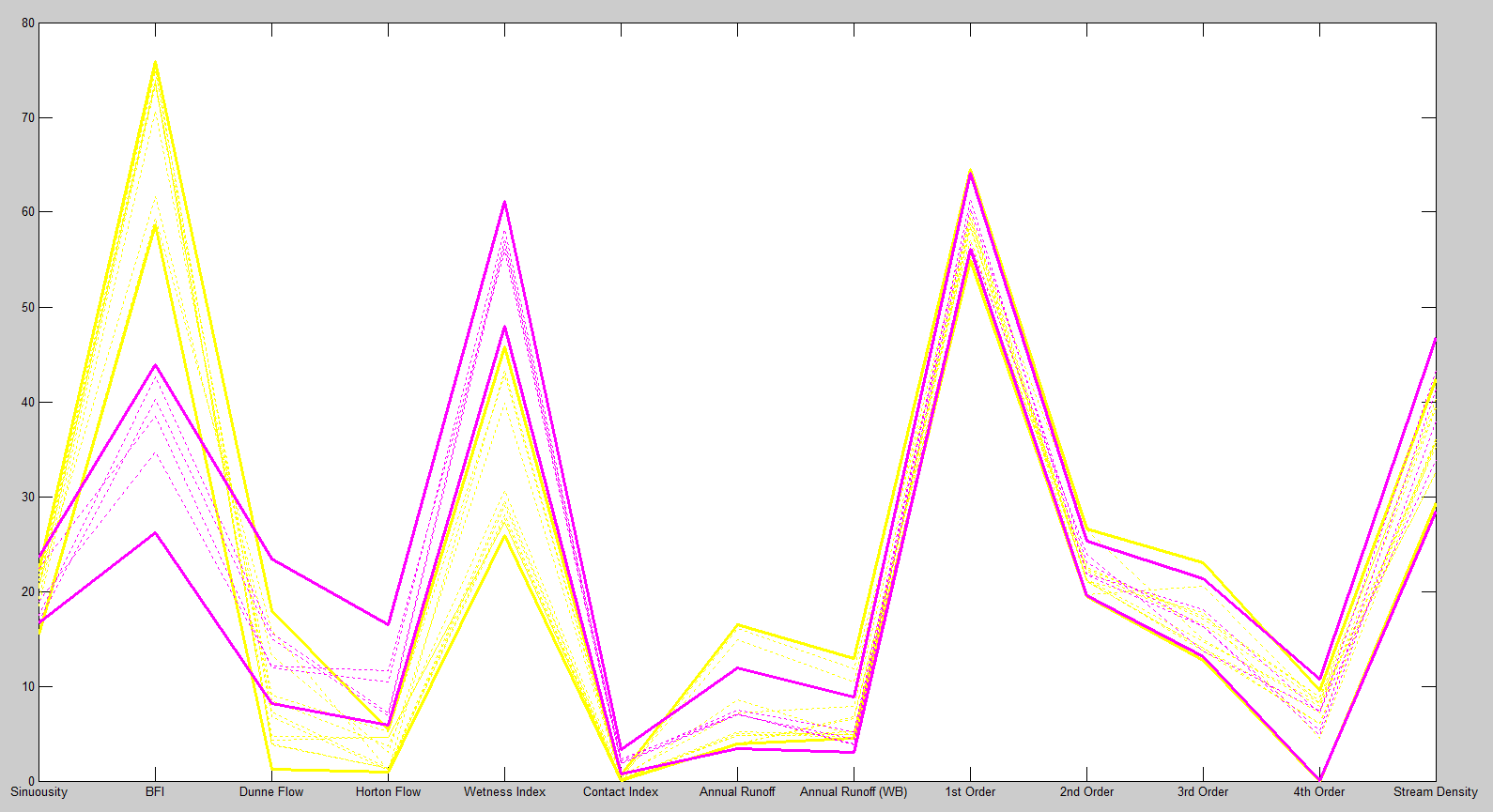


Figure 4.10: Clustering is a much easier way to spot trends

There’s so much data we can barely see anything, conclusions made from this would be uncertain. Try checking the ‘noise reduction’ checkbox for each cluster. This creates in bold (or as an envelope if envelope transparency has been selected) a line for the upper and lower quartile (UQ/LQ) for each dimension of a cluster. The polylines plotted between the UQ and LQ are the members which lie between the UQ/LQ for *every* dimension, see Figure 4.11.

Figure 4.11: Noise reduction gives a good idea of the shape of the bulk of the cluster



(For smaller datasets, there is a chance that there will be no members in a cluster which meet this criteria, for example see the green/pink clusters in Figure 4.18.)

This provides a much clearer picture: the groups appear to have been sorted by their well-defined distinction in dimensions BFI, Dunne Flow, Horton Flow and wetness index, with very little differentiation between annual runoff. This explains why the investigation into low and high runoffs earlier yielded more uncertainty.

An interesting point, particularly with the dimension ‘Contact Index’ is outliers. The thinness of our cluster quartiles shows that the majority of activity is, by our scale, occurring over a very small area. Could these outliers be skewing our investigation? (Clusters section continues on XX)

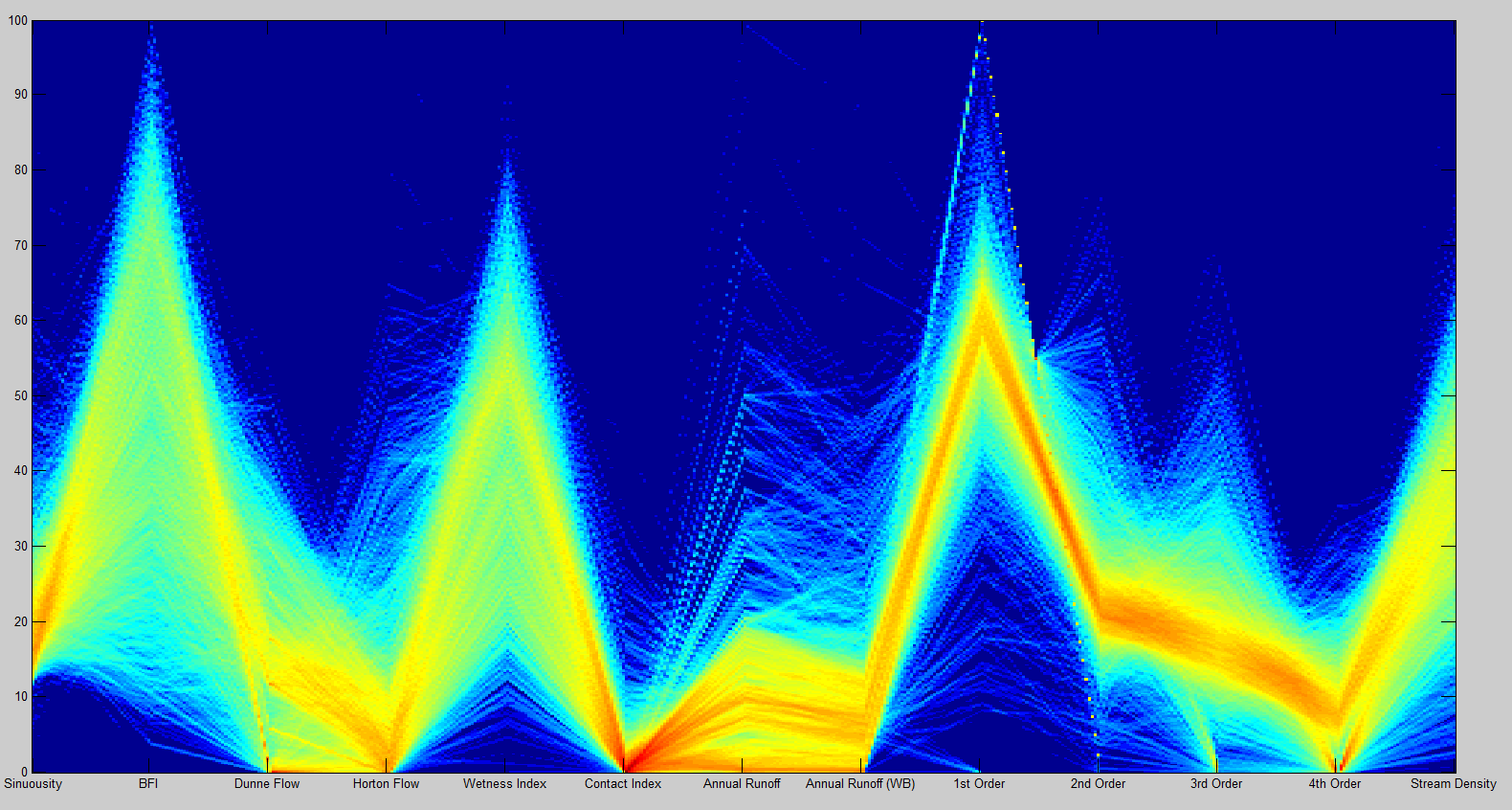
## 4.4 Density Plots

In order to focus on the bulk of the data we must first find where this occurs. Looking at the raw data (Figure 4.1) just blurs the polylines into one homogenous block of colour. We want to look into those blocks and see where the majority of members lie.

In the ‘Density plot’ panel, select an ‘XRes’ (X axis resolution: the amount of pixels in between a pair of dimensions) and a ‘YRes’ (Y axis resolution: the amount of pixels on the Y axis). Their defaults of 40 and 500 respectively will provide a high resolution picture but it may take a few minutes to calculate. Checking the log scale box is useful for greater variation in densities (i.e. larger datasets, so leave it checked).

Click ‘Plot to new figure’ to create density plot (Figure 4.12). Note that this function is currently only compatible with the ‘Main’ panel functions (ordering, flipping and scaling).

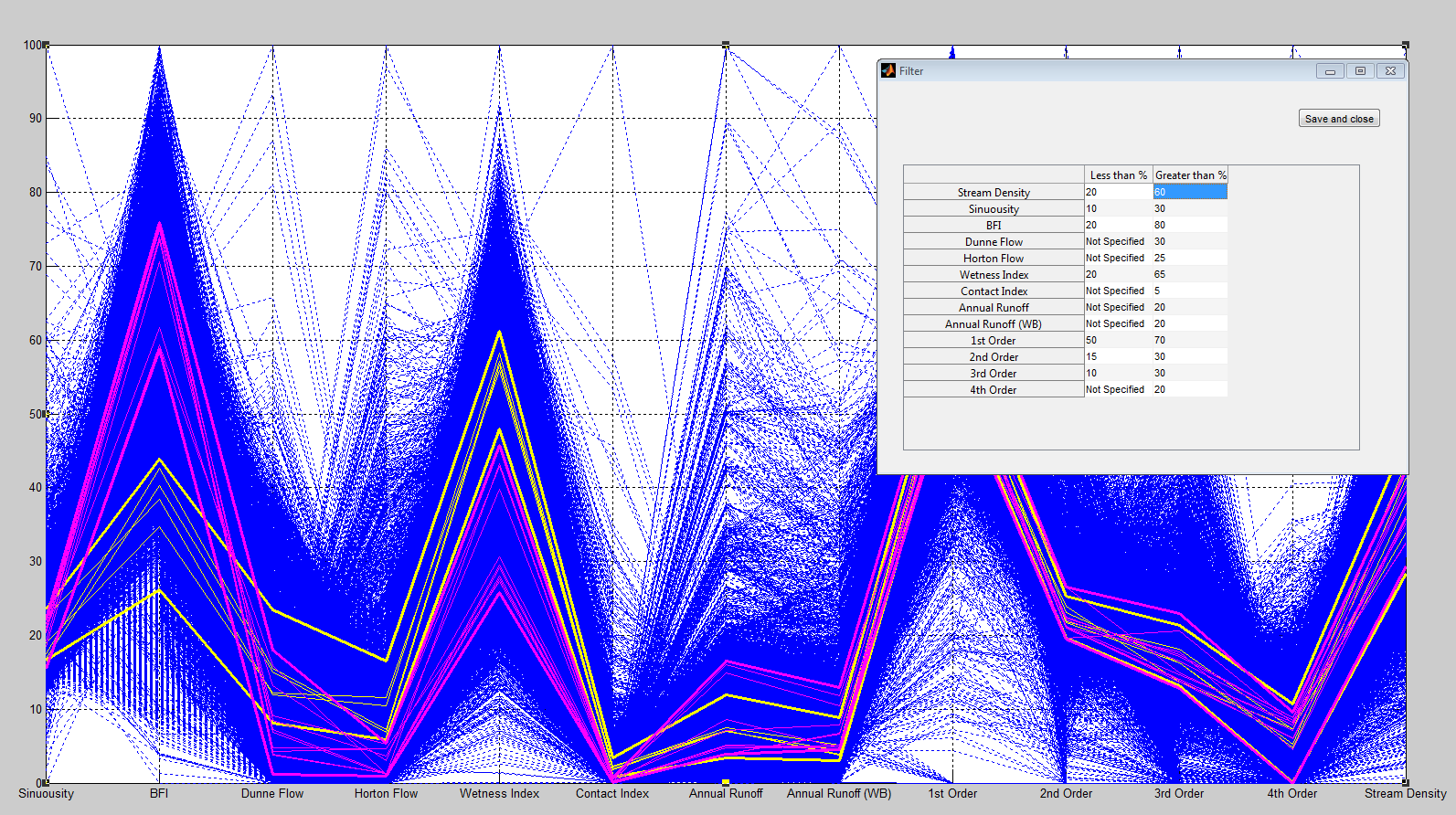
Figure 4.12: Using rankings rather than ranges can help reduce data density



## 4.5 Filtering

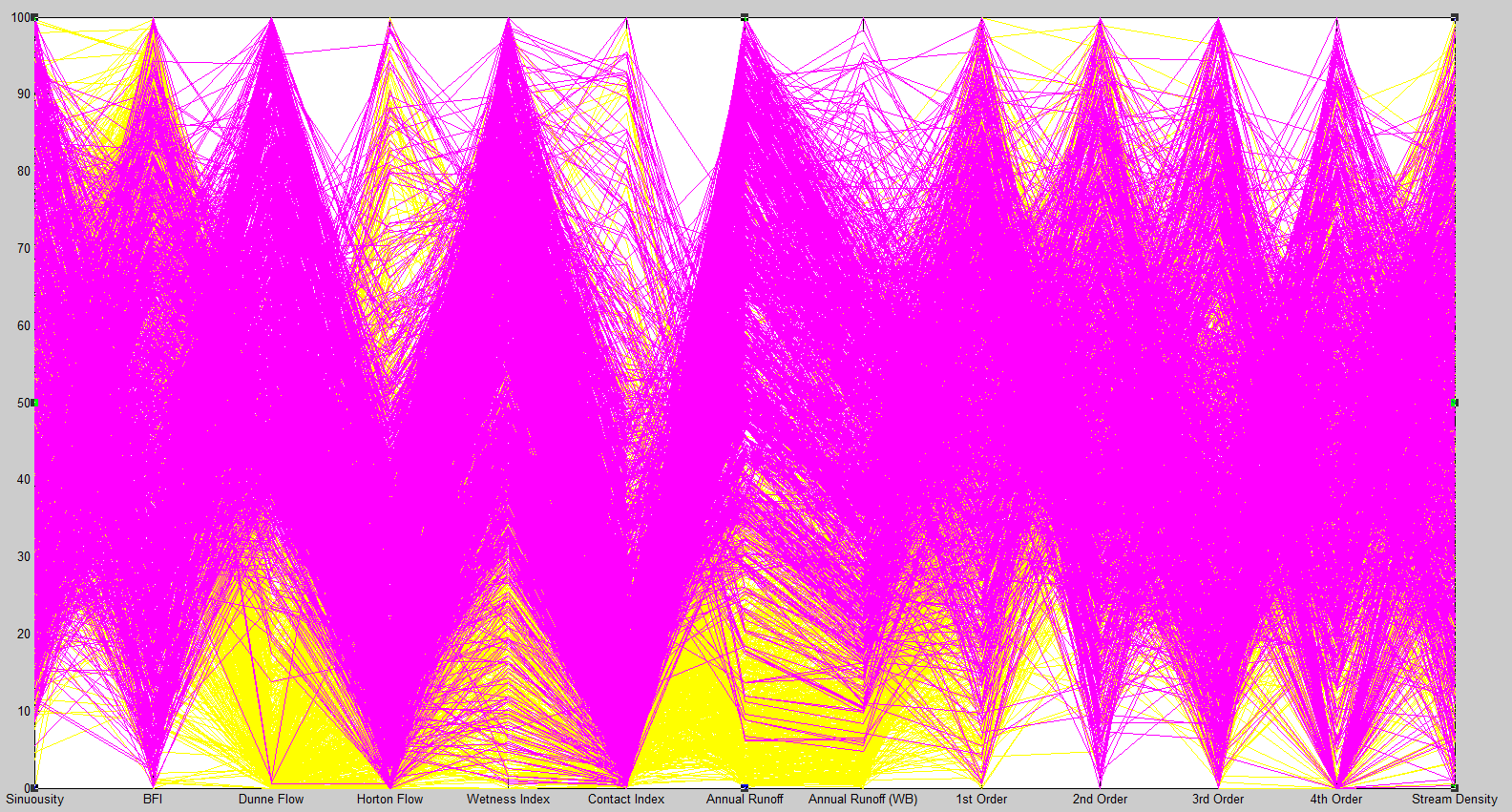
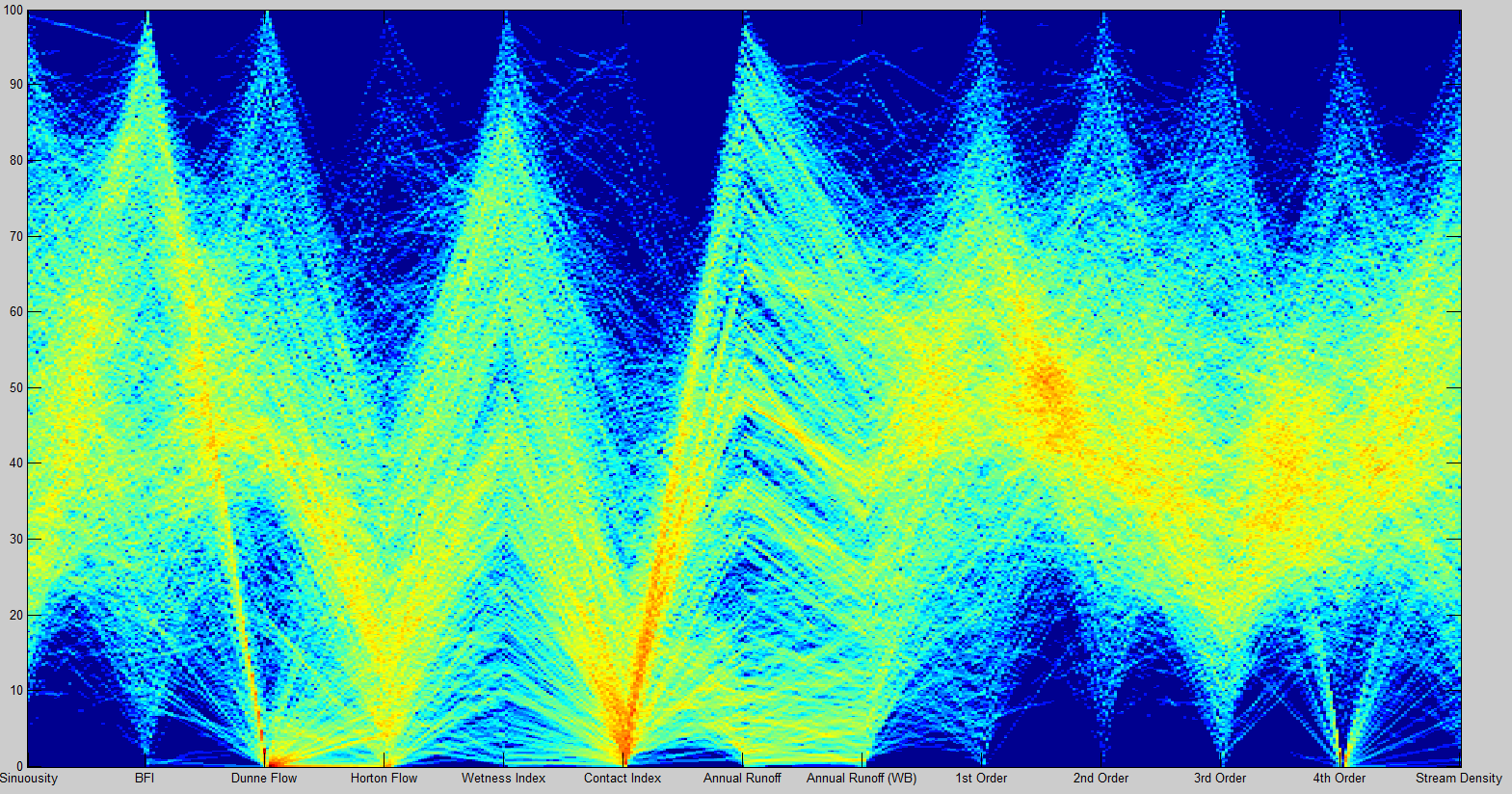
Press ‘Filter data’ in the ‘Filtering’ panel and use the density plot to decide a specification to filter out ‘noisy’ polylines. *This will erase all current groups and clusters (as will returning the original data)!*

Figure 4.13: Filtering is the best way to ‘zoom’ into a dimension



Using two clusters we can see that the data is still very dense (Figure 4.14a) but there appears to be fewer outliers. A new density plot (Figure 4.14b) confirms that the spread is much more even (with the possible exception of contact index and percentage first order streams).

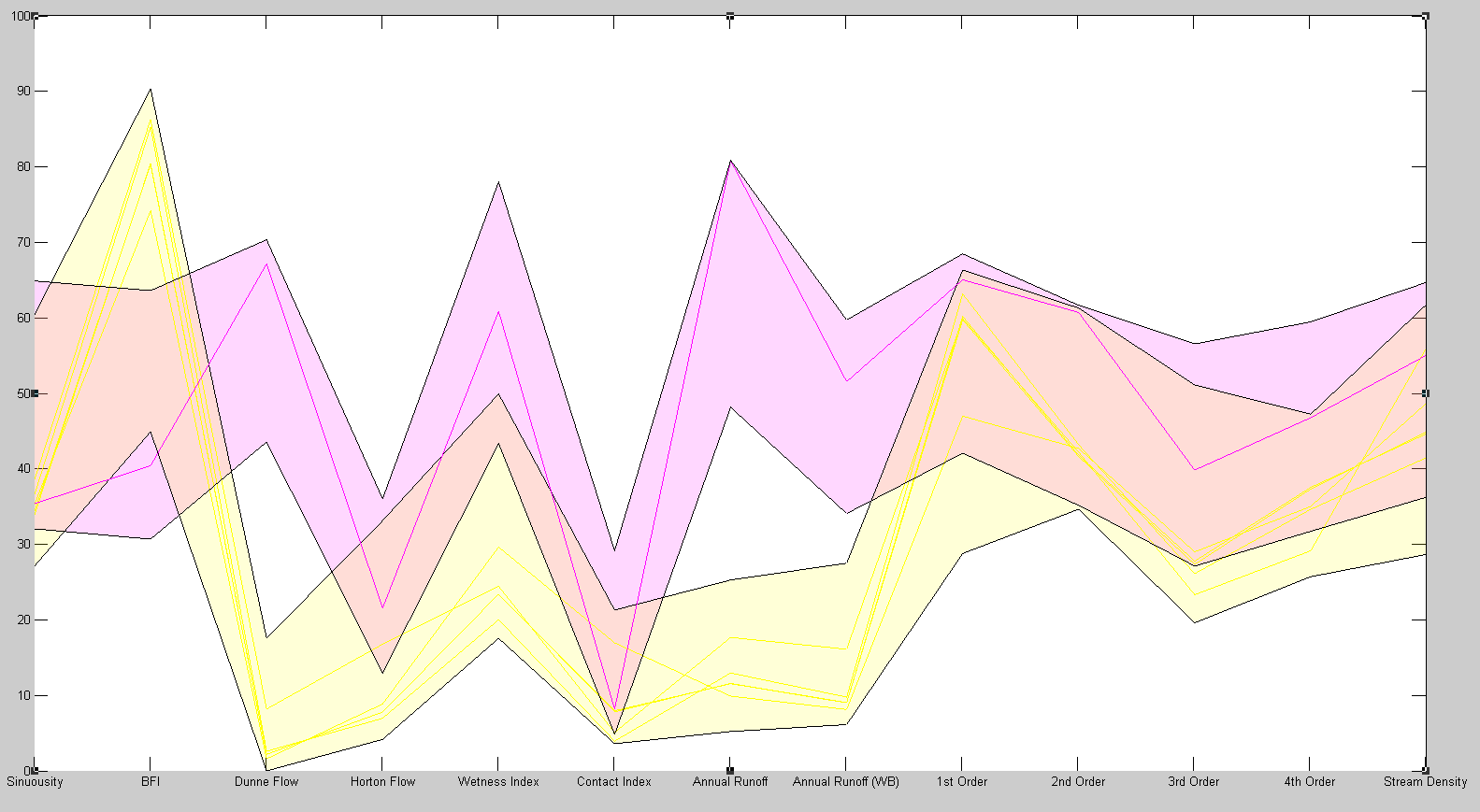
Figure 4.14a (left), 4.14b (right): Filtering can achieve a more even spread of members



## 4.6 Clusters (cont.)

Using noise reduction envelopes (by trial and error these seemed to me to provide the best presentation) re-plot the clusters, Figure 4.15. While the distinctions between Dunne Flow, Horton Flow and wetness index remain, a very large gap between runoffs has appeared and the previously strong distinction in BFI has lessened.

Figure 4.15: Different methods of presentation suit different plots, experiment to find the best



We can make sense of this by referring to the filter specification (Figure 4.13): BFI was the least severely filtered dimensions (a specification reduction of 40%), thus its impact on the clustering algorithm has not changed much. Compared to runoffs (a specification reduction of 80%) we can understand why runoffs are taking a much stronger role in clustering. I.e. initially runoffs did not play such a large clustering role as expected because of their being skewed by outliers.

### 4.6.1 How many clusters?

Reselecting the number of clusters as three, Figure 4.16, and re-plotting has not altered the first two very much but simply added a new (high wetness index, medium runoff) cluster. If you try to increase the number of clusters to four, Figure 4.17, you will see that the pink group is simply split into two smaller clusters.

Figure 4.17: These four clusters cover similar patterns as Figure 4.16

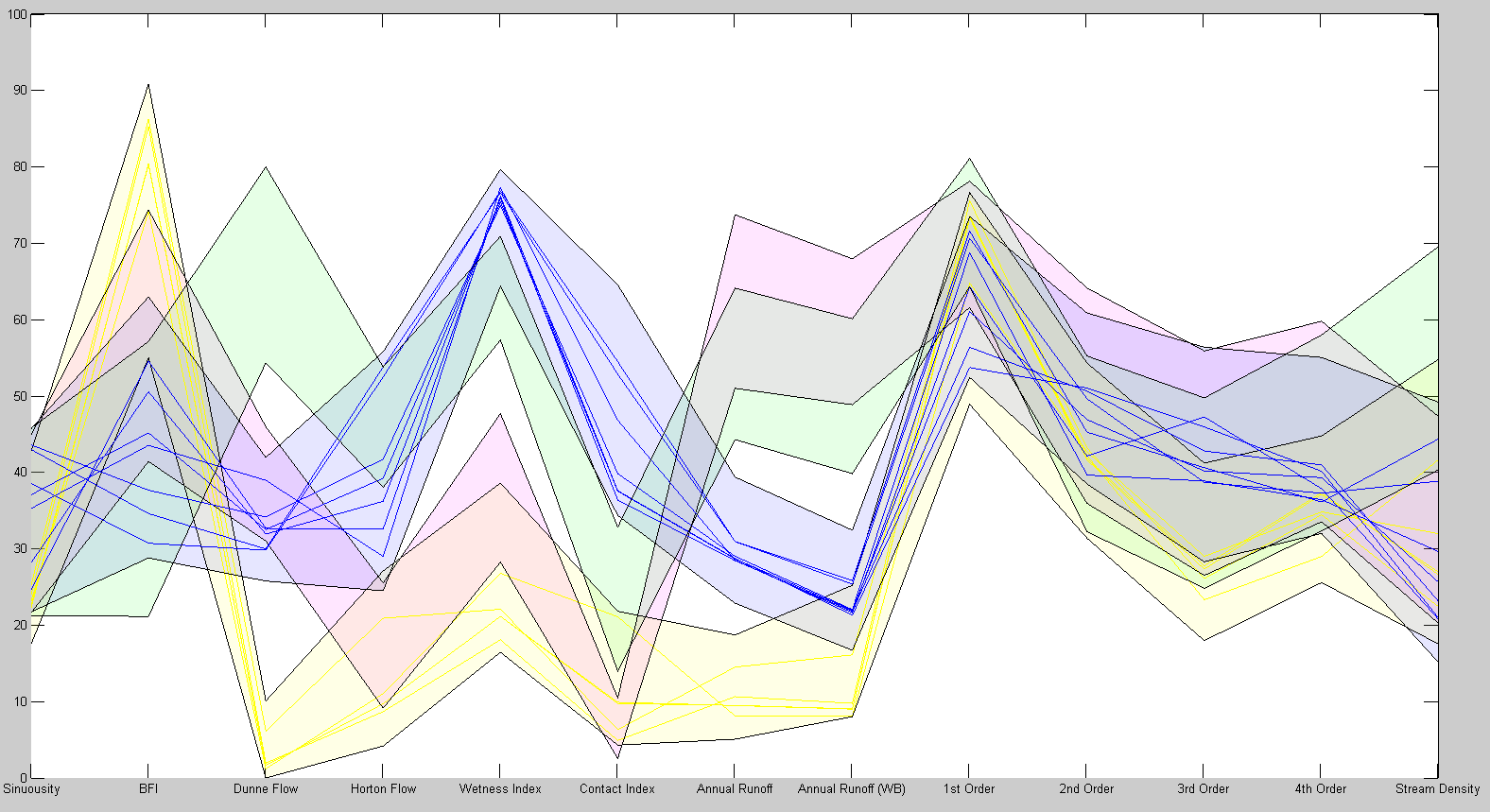
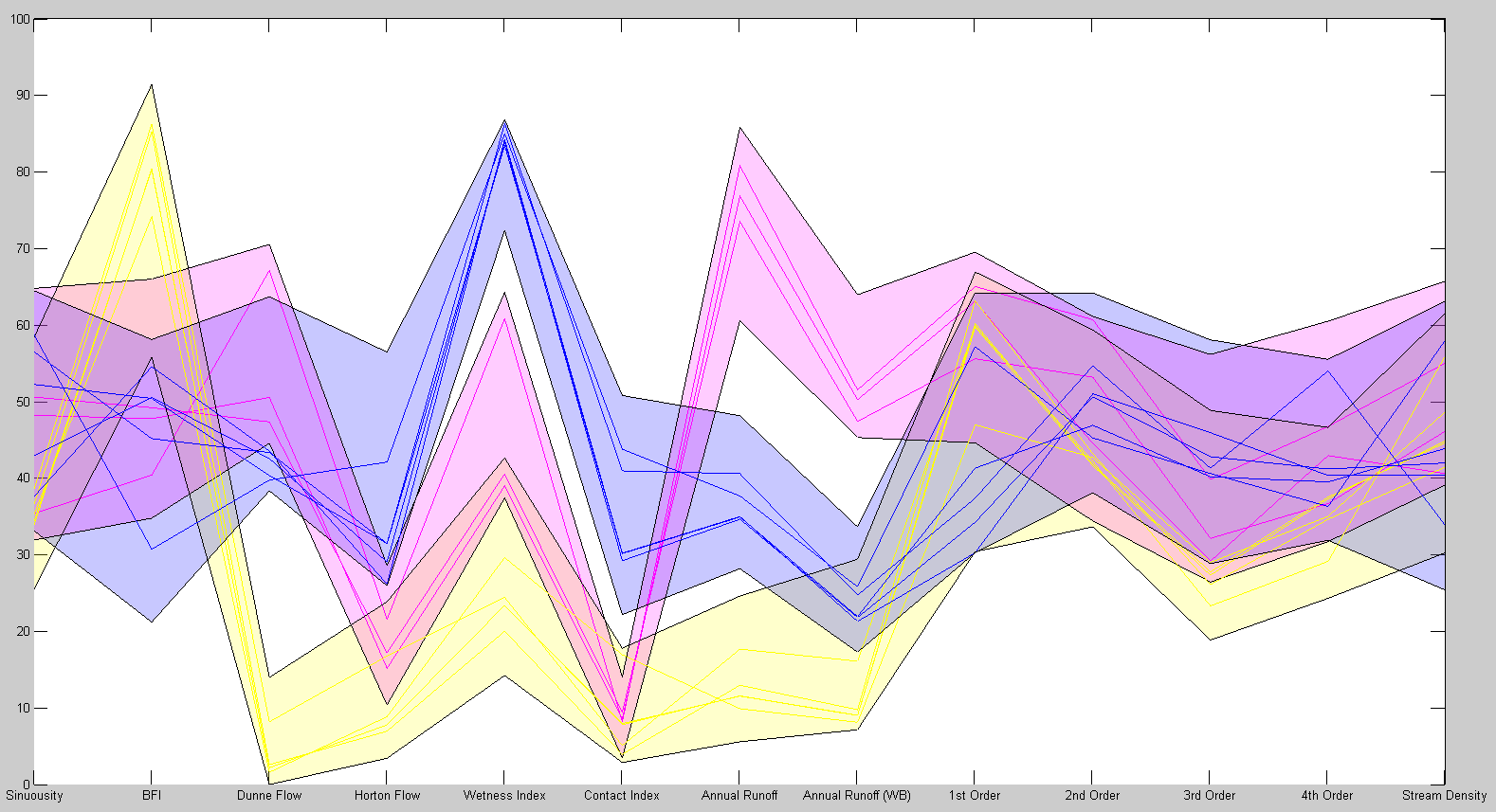


Figure 4.16: A whole new pattern has emerged compared to Figure 4.15



This is reflected by pressing the ‘optimal number’ button in the clusters panel. We can see in Figure 4.18 that the jump from 2 to 3 clusters reduces the average spread of clusters significantly more than 3 to 4. This button is useful if you don’t want to waste time trying out different amounts of clusters.

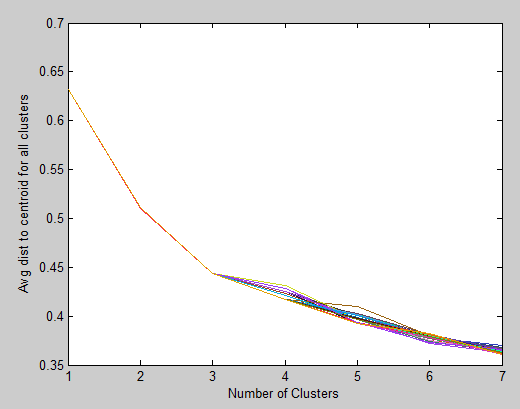


Figure 4.18: Using the optimal number of clusters button iteratively shows the when additional clusters become less useful

## 4.7 Further analysis

Referring to Figure 4.16 we can gain some insight into where to look next. By looking at how the clusters have been sorted we may suggest that the most relevant dimensions appear to be: Dunne Flow, Horton Flow, Wetness index, Contact index and Annual runoffs. We may also decide that BFI plays some role. By deleting the other dimensions we can focus the clustering algorithm.

## 4.8 Conclusion

Using the Figure 4.16 we can make a few conclusions:

* Low annual runoff predicts low wetness index, Horton Flow, Dunne Flow. It also (although less strongly) predicts high BFI.
* Medium annual runoff predicts high wetness index, Horton Flow and contact index.
* Contrastingly high annual runoff seems to predict medium wetness index and Horton flow and medium-low contact index.
* This would suggest that simple positive linear behaviour at low runoffs becomes more complicated (negative linear correlation!?) at higher runoffs.
* Runoff seems to have little to no impact on distribution of streams order, density or sinuosity. (This may be because all of these variables are closely related??).

For continued analysis or for better visualization you may wish to delete useless dimensions (e.g. the stream orders); simply press the ‘delete dimension’ button.

To display the maximum and minimum value of each dimension check the ‘Display max/min values’. These may not fit in the current setup, for more information on changing font sizes, see 5.\*\*

NOTE:

* The ‘delete dimension’ button will remove all existing groups and clusters.

### 4.8.1 Output data

If you wish to analyse groups and clusters in more detail it may be useful to output their members and details to an Excel or text file. Using the button ‘Output group and cluster data’ will allow you to specify a file location to print the data to.

NOTE:

* The button cannot currently override/replace existing files so you must save it with a new, unique filename.

### 4.8.2 States

To return your current setup at a later time; press the ‘save state’ button. This will save all the appdata and uicontrol settings to a handle structure in the location you select. Using ‘load state’ when the program is open will then reload these settings.

# 5 Further reading and possible code extension

## 5.1 Further reading

If the user desires other real life examples of applying PCP, a suggested starting point is with Inselberg, 1997 and Yong *et al*., 2009.

## 5.2 Code improvement

### 5.2.1 Speed improvement

The code currently can often lag considerably.

Most taxing (besides functions like ‘uicontrol’ or ‘figure’) is the frequent use of ‘getappdata’, ‘setappdata’ and ‘findobj’.

Speed tests suggest using ‘evalin’ in place of ‘getappdata’ and ‘assignin’ in place of ‘setappdata’ could significantly improve performance, however this may not be consistent with best practice as variables would be stored in the less secure workspace.

Passing all variables and object handles in one structure through appdata, rather than using ‘findobj’ and ‘getappdata’ repeatedly, gave worse performance in small test scenarios. This technique should be retested for the whole program as ‘findobj’ is used very often.

Passing variables and handles through ‘guidata’ has not been tested and should be considered, however ‘guidata’ calls appdata so it seems unlikely that it would be quicker.

‘gui\_mainfcn’ is the most frequently called function when profiling. As this is inherent with use of GUIDE it is hard to avoid. Rewriting the code without GUIDE would be wise to improve performance. (The code for this has been started and can be found in the ‘Without GUIDE (unfinished)’ folder)

### 5.2.3 More expansive error messages

Error messages are shown for some user errors but not others. Consideration should be given as to their appropriate display: currently they appear for 5 seconds and then disappear. Maybe they should stay until corrected or until another error occurs?

The ‘please wait’ message appears with an updating percentage for some longer functions, without an updating percentage for shorter functions and not at all for other functions. Consideration should be given as to whether this is too inconsistent for user interaction.

### 5.2.4 Output data

Currently the output data saves to text or excel formats, possibly the option of .mat files should be given also. This would not be a difficult addition to make as the data is already manipulated into cell arrays in the code.

More detail about groups may be appreciated by users, such as their mean/standard deviation as with clusters.

No additional code has been given to output k-medoid clusters. Modifying the k-medoid function to output centroid locations and average distance to centroids may be useful.

### 5.2.5 Plot to figure priority

Currently the priority to plotting ‘goes down the page’. i.e. the last group/cluster will be plotted over the first group/cluster and the clusters on top of the groups. Additional functionality to choose some kind of plotting priority would surely be useful.

### 5.2.6 >6 Groups/clusters? >20 Dimensions?

Currently the limit for clusters is 6. Any more clusters than this will either confuse the plot or be very clear cut (and thus be easy to define as groups).

Technically more than 6 groups can be created, however their interface goes outside the groups panel.

### 5.2.7 More than 20 dimensions are likely to confuse the plot.

If a user wishes for more dimensions/clusters/groups then it is suggested that they change the panels to three columns of adjustable length.

### 5.2.8 Font sizes

­If the user wishes to make the X tick labels a font size to suite the amount of dimensions then they should replace the code found in Figure 5.1 with the code found in Figure 5.2. The code is located around line 740 in the ‘updatef\_callback’ function (in pcpnew.m).

The code shown in Figure 5.2 generally fits for labels/units when the ‘Display max/min values’ box is **not** checked, but the ‘FontSize’ property could easily be altered. The X labels only are changed, if the user wishes to also change the Y labels then they should delete the 'YTickLabel',[]from the misc\_axes.

set(h.axes,'XTick',1:1:dimnum);

set(h.axes,'XTickLabel',labels','XAxisLocation','bottom','XGrid','on');

misc\_axes=copyobj(h.axes,h.f);

set(misc\_axes,'XAxisLocation','top','XTicklabel',unitm,'YTickLabel',[]);

set(misc\_axes, 'HandleVisibility', 'off','tag','misc\_axes');

Figure 5.1 (above), Figure 5.2 (below): original and additional code to set tick labels.

set(h.axes,'XTick',1:1:dimnum);

set(h.axes,'XTickLabel',[],'XGrid','on');

misc\_axes=copyobj(h.axes,h.f); set(misc\_axes,'XAxisLocation','top','XTicklabel',unitm,...

'FontSize',30/(dimnum^0.5),'YTickLabel',[]);

set(misc\_axes, 'HandleVisibility', 'off','tag','misc\_axes');

misc\_axes=copyobj(h.axes,h.f); set(misc\_axes,'XAxisLocation','bottom','XTicklabel',labels',...

'FontSize',30/(dimnum^0.5),'YTickLabel',[]);

set(misc\_axes, 'HandleVisibility', 'off','tag','misc\_axes');

## 5.3 Further functionality

### 5.3.1 Automated dimension filter

Currently a user must manually limit their dataset to 20 or less variables. It may be impractical to manually filter out dimensions by importance if there are over 100. Yang, 2003 proposes an automated method to do this which could form a pre-processing tool for conventional PCP analysis.

An alternative solution to avoid too much automation may be the ability to add dimensions from the interface. Combined with the existing ‘delete dimension’ button a user would be able to filter dimensions with more control and certainty.

### 5.3.2 Dimension spacing

Spacing dimensions based on the correlation between neighbours or the importance of variables has also been suggested, Yang, 2003. This may be useful to draw attention to stronger patterns and away from less significant dimensions.

### 5.3.3 Clustering algorithms

Zhao, 2012 suggest stronger algorithms to perform clustering, such as spectral clustering (based on the eigenvectors of an adjacency matrix). The PCP toolbox uses the k-means and k-medoids (Chen, 2010) algorithm only because a MATLAB function currently exists for it.

### 5.3.4 Automated loading

Using the code given in Figure 3.2, it would not be too difficult to create an automated import from excel into the PCP GUI, this should be given consideration.

# 6 References

## 6.1 Journal papers, conference proceedings­­­ and books

Few S. 2009. Now You See It: Simple Visualization Techniques for Quantitative Analysis (p288-294). Analytics Press. Oakland. US.

Li J., Martens J-B., Wijk J J v. 2010. Judging correlation from scatterplots and parallel coordinate plots. Information Visualization, Volume 9 (p13-30). Macmillan Publishers Ltd.

Zhao, X., Kaufman, A. 2012. Structure revealing techniques based on parallel coordinates plot. The Visual Computer, Volume 28 (p541-551). Springer.

Yang, J., Peng, W., Ward, M O., Rundensteiner, E A. 2003. Interactive Hierarchical Dimension Ordering, Spacing and filtering for Exploration of High Dimensional Datasets. Information Visualization IEEE Symposium (p105-112). Seattle. US.

Inselberg, A. 1997. Multidimensional Detective. Information Visualization IEEE Symposium (p100-107). Phoenix. US.

Moustafa, Rida EA, and Edward J. Wegman. 2002. On some generalizations of parallel coordinate plots. Seeing a Million–A Data Visualization Workshop, Rain am Lech, Germany. Volume 2. No. 4.

Yong, G., Sanping, Li., Lakhan, C., Lucieer, A. 2009. Exploring uncertainty in remotely sensed data with parallel coordinate plots. Information Journal of Applied Earth Observation and Geoinformation (p413-422). Elsevier.

## 6.2 Datasets

Countries dataset:

Johansson, A. CENG30001\_2012\_13: Physical and Human Systems Modelling 3 12\_13.

Hydrological dataset:

Falcone, J., A., Carlisle, D., M., Wolock, D., M., Meador, M., R. 2010. GAGES: A stream gage database for evaluating natural and altered flow conditions in the conterminous United States. Ecology, Volume 91, Number 2 (p621). Ecological Society of America.

## 6.3 Functions

‘dlmcell’ writes cell array to text files.

Last updated 12.10.2010

© Roland Pfister

‘kmedoids’ sorts dataset into clusters.

Version: 15.02.2012

© Mo Chen