Paper SAS4341-2016

Graph A Million with the SGPLOT Procedure

Prashant Hebbar and Sanjay Matange, SAS Institute Inc.

ABSTRACT

In today's world of torrential data, plotting large amounts of data has its own challenges. The SGPLOT procedure in SAS Output Delivery System (ODS) Graphics offers a simple yet powerful tool for your graphing needs. In this paper we present some graphs that work better for large data sets. We create heat maps, box plots, parallel coordinate plots that visualize large data.

You can make your graphs resilient to your growing data with ODS Graphics!

INTRODUCTION

The SGPLOT procedure has been a cornerstone of ODS Graphics since it was introduced in SAS® 9.2. It provides quick and simple ways to plot single-celled scatter plots, series plots, box plots and much more. It has seen many additions and improvements since its debut.

The traditional approach of directly plotting all your observations does not work well for large data sets. For example, creating a scatter plot of a million observations with two variables would take a long time and might even require modifications to bump up the memory (on the rendering side). Once the plot is rendered, it is difficult to discern any meaning unless we use transparency, which comes with its own performance cost. While transparency does give you a general feel for the density of the observations, it lacks the resolution needed for fine-grained comparisons.

This paper describes some ways to deal with this issue. Some of the examples shown here require the third maintenance release of SAS® 9.4.

See the Resources section later in this paper for links to the PDF file of this paper and the programs used here.

SCATTER PLOT

To illustrate the performance and readability issues with direct plotting, let us consider an airline on-time data set for the first quarter of 2012. This data set has 44 variables and 1,472,587 observations. Some sample observations are shown below (showing only the columns of interest):

FL_DATE	UNIQUE_CARRIER	DAY_OF_WEEK	DEP_DELAY	ARR_DELAY	ARR_DELAY_NEW	DISTANCE
17JAN2012	DL	2	0	-9	0	400
19JAN2012	AA	4	5	11	11	680
...	...					

Table 1. Sample Observations from Airline On-Time Data Set

Let us try to get a feel for how arrival delays vary by day of the week, across different airlines. One approach is to draw a scatter plot of the raw observations. We plot UNIQUE_CARRIER against DAY_OF_WEEK and color the markers by ARR_DELAY as a response, while also using data transparency to see the observation density at any given coordinate. The output is shown in Figure 1.

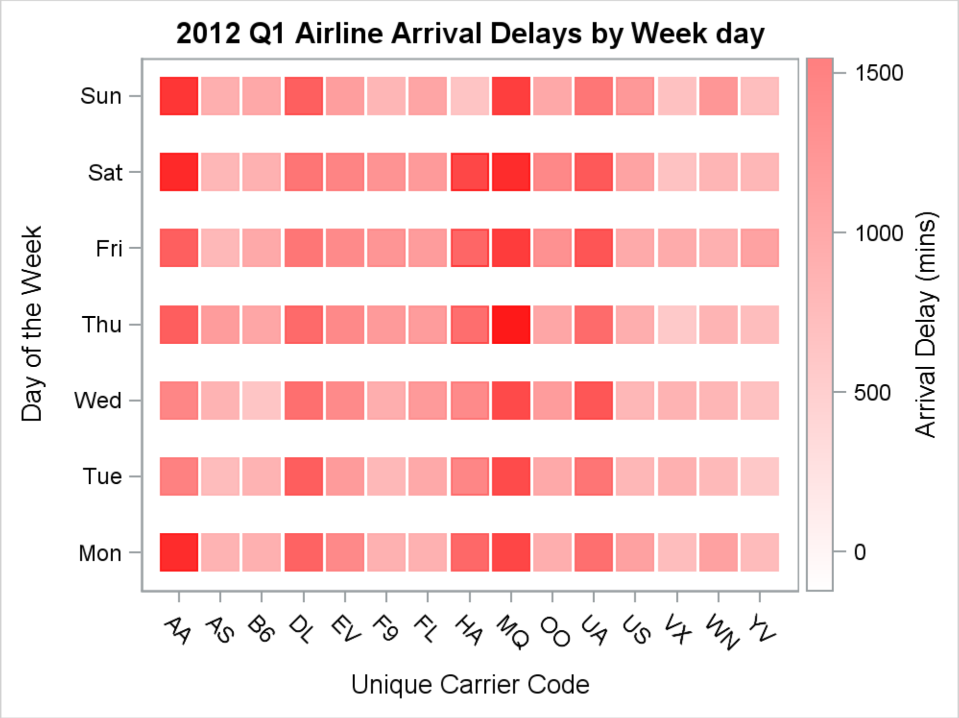


Figure 1. Scatter Plot of Airline Delay: 1,472,587 obs, 93.6 s [1]

As you can see, this graph is not very useful. It is difficult to infer the average delays by relying on the transparency effects. The color blending effects of overlaid transparent markers also make it harder to read the values off the continuous legend. The code for this graph is shown below:

```
...
proc sgplot data=scatter_vars ;
title "2012 Q1 Airline Arrival Delays by Week day";
  label unique_carrier="Unique Carrier Code"
      arr_delay="Arrival Delay (mins)" day_of_week="Day of the Week";
  format day_of_week num2downame.; /* Map 1..7 to Mon..Sun */
  scatter x=unique_carrier y=day_of_week /
    markerAttrs=(symbol=squareFilled size=20)
    colorResponse=arr_delay colorModel=(white red) transparency=0.5 ;
run;
...
```

This also takes a long time to render – proportional to the number of observations. This particular example takes about 93.6s [1] besides also needing the maximum Java VM memory setting for the renderer to be bumped up to at least 407MB! Next, let us see how we can improve the situation.

HEAT MAP

In the third maintenance release of SAS 9.4, the SGPLOT procedure supports a HEATMAP statement. This plot performs summarizations and binning over two-dimensional input data. It supports numeric as well as categorical X, Y data. It can compute frequency as the implicit response, or compute percent, sum or mean for a separate response variable.

[1] All times in this paper are from an Intel i7 3.40GHz 8-core CPU with 16GB RAM PC running Windows 7

This implies that while there is an increase in data processing times for computing the heat map ready data, the rendering time should be reduced compared to the raw scatter plot use. Let us see how we can use this new statement to visualize how arrival delays vary by day of the week, across different airlines.

Figure 2 shows a heat map with the day of the week by arrival delay, with the mean of arrival delay computed as a color response.

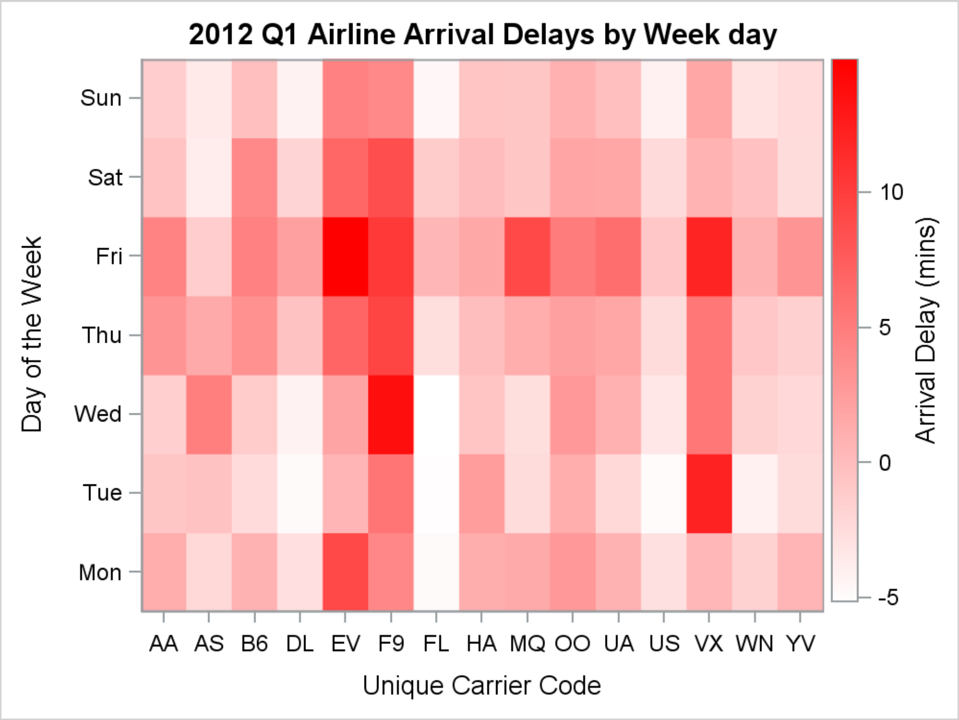


Figure 2. Heat Map of Airline Delay: 1,472,587 obs, 3.1 s

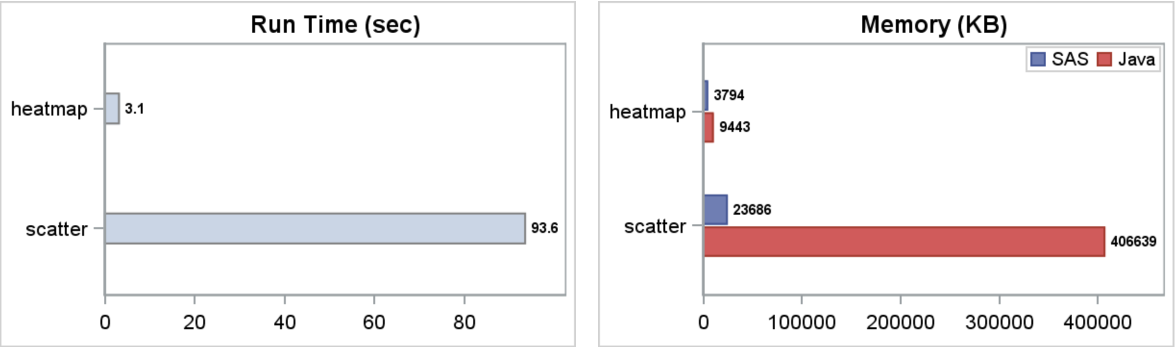
Now we can see some sort of a pattern – Fridays seem to be a little worse than other days as indicated by the prevalence of bright red tiles. Also, a couple of the airlines seem to have less delay compared to the rest. In this graph, the color of the tiles also maps correctly to the colors in the continuous legend.

The program snippet for this is shown below:

```
...
proc sgplot data=heatmap_vars ;
title "2012 Q1 Airline Arrival Delays by Week day";
  label unique_carrier="Unique Carrier Code"
        arr_delay="Arrival Delay (mins)"
        day_of_week="Day of the Week";
  format day_of_week num2dowName.; /* map 1..7 to Mon..Sun */
  heatmap x=unique_carrier y=day_of_week / name="heatmap"
        colorResponse=arr_delay colorStat=mean discreteY
        colorModel=(white red);
run;
...
```

Note the DISCRETEY flag in the HEATMAP statement. Since the DAY_OF_WEEK variable in this data set is an integer from 1 to 7, this flag lets the system treat it as a discrete variable and avoid binning it as a numeric variable.

The run times and memory usage (for SAS as well as the Java side renderer) for the previous scatter plot and the heat map are shown below:



As you can see, using the heat map for large data is definitely a huge improvement over raw scatter plots.

HEAT MAP FOR NUMERIC VARIABLES

The HEATMAP statement also supports binning of numeric input variables. To illustrate this, let us create a heat map of departure time by distance with the response being the mean of arrival delay. Figure 3 shows the SGPLOT output for this graph.

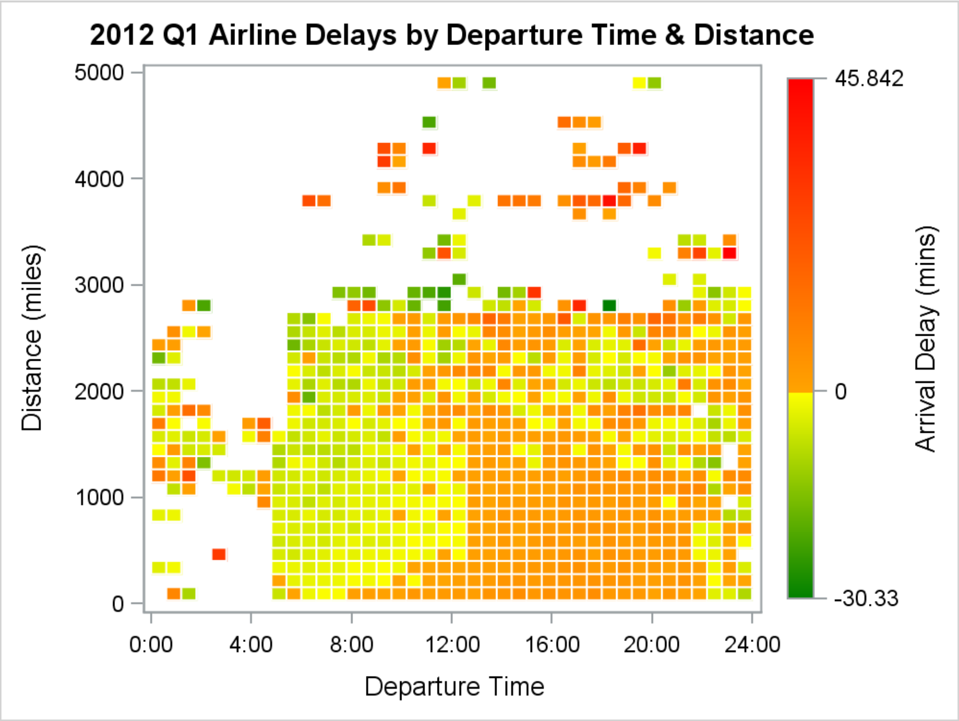


Figure 3. Heat Map of Two Numeric Variables, 1,472,587 obs, 3.2 s

As you can see in the graph, the delays seem to start creeping in during the afternoon and last until late at night. The white areas in the heat map are due to missing data. We have specified forty bins in each dimension in this example to keep the tiles legible.

The code snippet for this graph is shown below:

```
proc sgplot data=heatmap_num rAttrMap=rangeMapData;
title "2012 Q1 Airline Delays by Departure Time & Distance";
  label dep_time="Departure Time"
      arr_delay="Arrival Delay (mins)"
      distance="Distance (miles)";
  heatmap x=dep_time y=distance / name="heatmap"
    colorResponse=arr_delay rAttrId=myid colorstat=mean
```

```
      nXBins=40 nYBins=40
      outline outlineAttrs=(color=white)
      ;
    run;
```

Notice that this example uses a custom color model for the response via a range attribute map data set, which gives you the power to map the color response values as you choose. This heat map's range attribute map data set contains separate piecewise linear mappings for the negative and positive ranges. It was created with the following code:

```
  data rangeMapData;
    length min $5 max $5 colorModel1 $12 colorModel2 $12;
    input min $ max $ colorModel1 $ colorModel2 $;
    retain id "myid";

    datalines;
    _min_   0     green  yellow
     0     _max_ orange red
    ;
  run;
```

This lets you clearly see the delayed (positive) and the ahead of time (negative) arrivals

A TWEAKED BOX PLOT

Box plots are fairly standard and nothing new for the SGPLOT procedure. However, they don't give you much information about how the observations are distributed along the range. The distribution information could be more important, offering additional insight for larger data sets.

You could pair the boxes with a distribution plot, or overlay scatter plots with jittered markers toward this purpose. The support for heat maps enables one more solution: you can now overlay box plots over heat maps!

Let us graph arrival delays for each airline in our data set using box plots. Let us consider the delays in the [-30, 60] range to limit the outliers in this illustration. We will also plot a heat map of these observations, using their frequency as the color response, below the box plot layer. The result is shown in Figure 4.

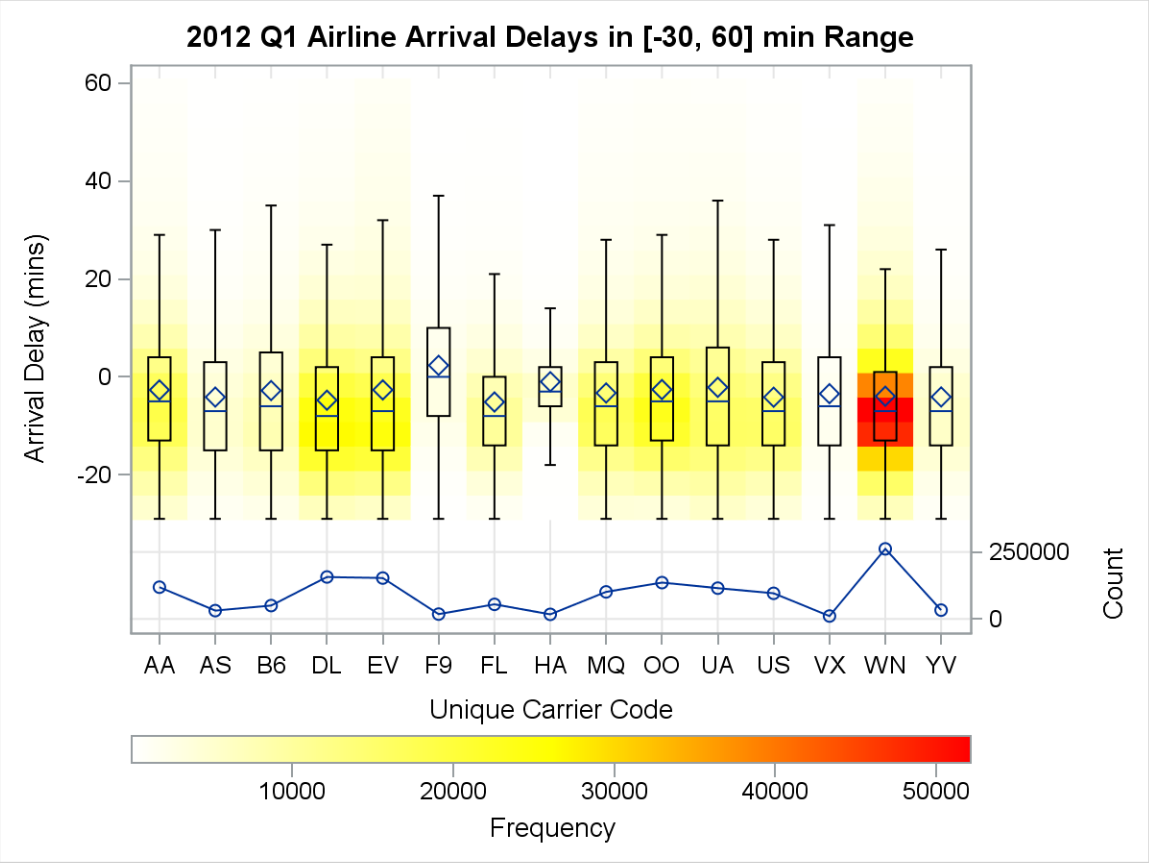


Figure 4. Box Plots over Heat Map of Arrival Delay: 1,352,185 obs, 17.02 s.

This graph clearly shows the difference in the distributions of observations between each airline. Carrier 'WN' has the most observations and they are concentrated near the median. We have also plotted the count for each airline below the boxes for reference. The code snippet for this graph is shown below:

```
...
proc sgplot data=box_heat_final noAutoLegend;
title "2012 Q1 Airline Arrival Delays in [-30, 60] min Range";
  label unique_carrier="Unique Carrier Code"
        arr_delay="Arrival Delay (mins)"
        _freq_="Count";
  heatmap x=unique_carrier y=arr_delay / name="heatmap"
        yBinSize=5 colorModel=(white yellow red);
  vbox arr_delay / category=unique_carrier nofill nooutliers
        whiskerAttrs=(color=black);
  series x=ucc y=_freq_ / y2axis markers;
  xaxis grid;
  yaxis offsetMin=0.2;
  y2Axis offsetMax=0.85 grid labelpos=dataCenter;
  gradLegend "heatmap" / position=bottom;
run;
...
```

We have suppressed the outliers in the above example for simplicity, but you could still use this overlay technique with outlier display. Also note that the single cell has been effectively split into two by using the "axis splitting" technique using offsets on the Y and Y2 axes, allowing the series plot to be drawn in a separate data space aligned with the Y2 axis.

MULTI-DIMENSIONAL DATA

One of the ways large data presents itself is in data that is multi-dimensional (that is, the data contains a large number of variables). Here is a sample of one such data set — the Australian Weather data set.

Date	Location	MinTemp	MaxTemp	Temp9am	Cloud9AM	Rainfall	WindSpeed9am	...
2012-06-22	Adelaide	6.1	10.8	8.4	1	27.6	13	
2012-07-09	Albury	8	10.9	8.6	8	13.2	9	
	...	...						

Table 2. Australian Weather Multi-dimensional Data Set

When you want to get a quick overview of such a data set without using compute-intensive techniques such as principal components, a parallel coordinates plot is a good start. Let us try to analyze six of the variables in the above data set: MinTemp, MaxTemp, Temp9am, Cloud9am, Rainfall, and WindSpeed9am by Location. Figure 5 shows such a plot produced using the SGPLOT procedure. While this is not using a million observations, we are representing 142,800 values here (23,800 observations with 6 variables each).

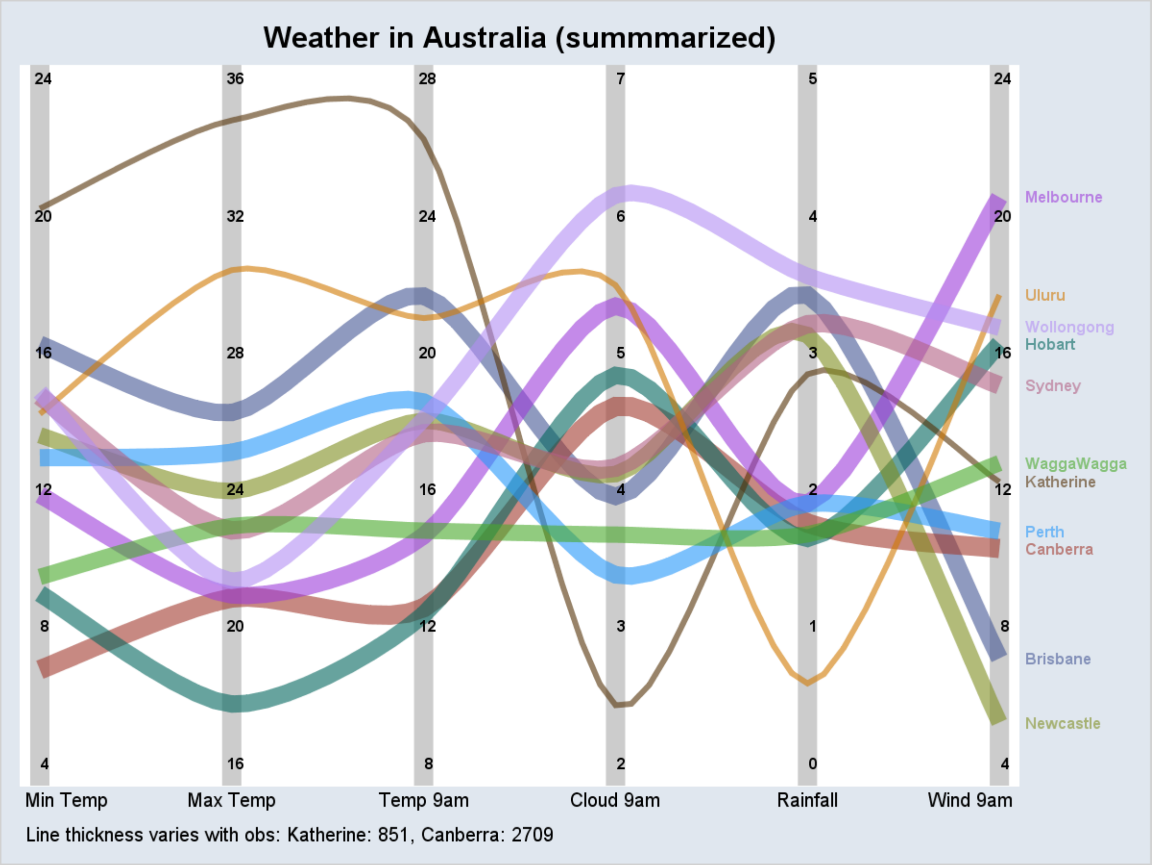


Figure 5: Parallel Coordinate Plot: 6 vars x 23,800 obs, 0.22 s.

We have summarized the variables of interest by their location before plotting them. This helps to reduce the amount of data going into the rendering system.

Generally, parallel lines between two parallel axes suggest a positive relationship between their corresponding dimensions, whereas crossing lines suggest a negative relationship. This graph suggests that MaxTemp and Temp9am are positively related, while Rainfall and WindSpeed9am are negatively related.

Recall that we computed the means of these six variables by location. We have represented their relative frequencies using the thickness response (new for the third maintenance release of SAS® 9.4) for the series plot. We have then simulated the axes with reference lines and labeled the ticks using a text plot (new for the second maintenance release of SAS® 9.4).

Let us look at the code for this graph:

```
...
proc sgplot data=par_axis_final noBorder;
title 'Weather in Australia (summmarized)';

  styleAttrs backColor=cxE0E7EF;

  refLine pos / axis=x lineAttrs=(color=grey thickness=10)
      transparency=0.6 label=label labelPos=min labelAttrs=(size=2.2%)
      labelAttrs=(size=2.2%);

  series x=x y=y_pct / group=location transparency=0.4
      thickResp=_freq_ smoothConnect curveLabel curveLabelLoc=outside
      curveLabelAttrs=(weight=bold size=2%);

  text x=axis_x y=axis_y text=tvalue / textAttrs=(size=6 weight=bold);

  xAxis display=none offsetMin=0.02 offsetMax=0.02;
  yAxis display=none offsetMin=0.03 offsetMax=0.02;

footnote j=l height=7pt
     'Line thickness varies with obs: Katherine: 851, Canberra: 2709';
run;
...
```

We have normalized all the six variables of interest as Y_PCT in the range [0, 1] with each variable corresponding to an X value from 1 to 6 in a prior data step. The resulting X and Y_PCT are then drawn as series plots. The series lines have been drawn with the SMOOTHCONNECT option, which reduces sharp transitions at the points in the series plot.

CONCLUSION

To review the main techniques in this paper:

- Take advantage of plots that do summarization, such as heat maps and box plots.
- For a quick overview, pre-summarize and visualize multi-dimensional data as parallel coordinate plots.
- Lastly, when graphing the same large data multiple times, create an intermediate data set containing only the variables needed for the graph for better performance.

The need to visualize large data sets is only going to increase in the future. But as this paper has shown, designing your graphs to use heat maps and parallel coordinate plots with summarization can help toward the analysis of such data.

RESOURCES

The PDF file of this paper and the SAS code for all the programs is available at:
Paper: http://support.sas.com/resources/papers/proceedings16/SAS4341-2016.pdf
Code: http://support.sas.com/resources/papers/proceedings16/SAS4341-2016.zip

REFERENCES

Unwin, A., Theus, M. and Hofmann, H. 2006. *Graphics of Large Datasets*. New York: Springer

RECOMMENDED READING

- *SAS® 9.4 ODS Graphics: Procedures Guide*

ACKNOWLEDGMENTS

Many thanks to Lingxiao Li for his constructive comments.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Prashant Hebbar
SAS Institute, Inc.
prashant.hebbar@sas.com

Sanjay Matange
SAS Institute, Inc.
sanjay.matange@sas.com