

Effective Forecast Visualization With SAS/GRAPH

Samuel T. Croker, Lexington, SC

ABSTRACT

A statistical forecast is useless without sharp, attractive and informative graphics to present it. It is really much easier to produce such graphs in SAS than by exporting the data to Excel and using Excel charts. This case study demonstrates this simplicity using a few macros that make it possible to generate production quality forecast, confidence interval and other forecast-specific visualizations for appropriate SAS procedures. Methods for dealing with odd date intervals, such as five-day weeks, and usefully displaying difficult datetime axis values are explored. The annotate facility is used to help with displaying model and additional date information. A method for displaying confidence intervals as regions is shown as well as a way to produce separate regions for different types of data. Use of SYMBOL, AXIS and LEGEND statements is discussed along with details for using fonts, colors and lines to make the graphics look magazine worthy.

INTRODUCTION

SAS 9 has begun a revolution in the off the shelf graphic abilities of SAS software, but sometimes it is still easier to continue with SAS/GRAPH (particularly if one is working at a client that refuses to upgrade to SAS 9).

The following statements in SAS/GRAPH are available in PROC GPLOT to modify the appearance and structure of a time series plot:

- The SYMBOL statement can be used to make the data objects look nice
- The AXIS statement is available to modify the axes to make them presentable and to make the data easy to see
- The LEGEND statement is used to modify the standard legend to make it easier to read
- PROC GPLOT options such as OVERLAY, HREF, VREF, GRID and many others can make the display of information easier

Other options and facilities are necessary to produce quality graphics:

- The ANNOTATE facility can be used to help display information that might be confusing if put directly on an axis
- Date, Time and Datetime formats and functions are available to alter the display of and process date values
- There are many SAS graphics devices available for producing output – the ACTIVEX device is used in this example

With all of the options available, keep in mind that the goal is to make a picture that quickly and easily be understood without miss-representing the data. This usually means keeping the fluff to a minimum.

FORECAST VISUALIZATION

The primary focus of this example is de-cluttering the forecast plot by using colored regions for forecast prediction or confidence intervals and manipulation of the date/time axis. This example is only a starting point for many different approaches and is not meant to be a complete solution for every forecast.

A TRANSFER FUNCTION FORECAST EXAMPLE

SETTING THE STAGE

An adaptation of the well known Box and Jenkins Series J example will be used to demonstrate a few techniques that are helpful. First, a few details must be set up such as adding an arbitrary datetime value to the data.

```
data seriesj;
  if _n_=1 then dtm=1451692800;
  else dtm+900;
  input x y ;
  datalines;
  <SKIPPED LINES>
run;
```

The following two macro variables will set how far back from the end of the time series will PROC ARIMA start the multi-step forecasts (BACK) and how many multi-step forecasts will be generated (LEAD).

```
%let back=50;
%let lead=70;

proc arima data=seriesj;
    identify var=x nlag=21;
    estimate p=3 q=(2 3) noint;
    identify var=y crosscorr=(x) nlag=21;
    estimate p=1 q=3 input=( 5 $ (1,2)/(2) x ) plot;
    forecast out=predict id=dtm back=&back lead=&lead;
quit;
```

GATHERING INFORMATION ABOUT THE DATA AND FORECASTS

At this point, the forecasts have been made and now the preparation for plotting will begin. Start with gathering a little bit of data about the original time series so that the plot can be controlled.

```
/* Find and store the number of actual observations, along with the min
   and max. This will be used for plotting only */
proc sql noprint;
    select          count(*)
                  , min(y)
                  , max(y)
    into
                  :numrows
                  , :min_yaxis
                  , :max_yaxis
    from seriesj;
quit;
%let min_yaxis    =%sysfunc(floor(&min_yaxis));
%let max_yaxis    =%sysfunc(ceil(&max_yaxis));
%let startholdout =%eval(&numrows-&back);
%let forecastend  =%eval(&numrows+&lead-&back);
%let plotstart    =%eval(&numrows-&back*2);
```

Now a few macro variables have been set that can be used to control the plotting and to break up the confidence intervals into regions.

- MIN_YAXIS is the minimum value of the Y axis data over the entire time series. The forecast could go outside of this range. This value will be used in scaling the Y axis.
- MAX_YAXIS is the maximum value of the y axis data over the entire time series.
- NUMROWS is the number of data points in the time series.
- STARTHOLDOUT is the actual time value that corresponds to the BACK macro variable set earlier.
- FORECASTEND is the point where all forecasting stops.
- PLOTSTART is the point where the graph will start displaying the time series + forecasts.

All of these settings can be changed to alter the display of the graph. In this case the PLOTSTART was restricted so that the forecast area of the graph is highlighted.

BUILDING THE PLOT DATA

In this example, some hold out data was generated in order to compare the forecast with the actual data. So three confidence interval regions will be generated: one for the one step ahead forecasts, one for the hold out data and

one for the forecasts themselves. Earlier the BACK and LEAD macro variables were set to control how the FORECAST statement of PROC ARIMA will generate forecasts. These values, along with the number of rows of the original data, were used to calculate the time id where hold out forecasting and complete forecasting begins. See the data step below for details:

```
data      actual(                keep=dtm x y forecast)
          onestep_low(          keep=dtm 195 rename=(195=onestep_cr ))
          onestep_high(                keep=dtm u95 rename=(u95=onestep_cr ))
          holdout_low(           keep=dtm 195 rename=(195=holdout_cr ))
          holdout_high(                keep=dtm u95 rename=(u95=holdout_cr ))
          forecast_low(           keep=dtm 195 rename=(195=forecast_cr))
          forecast_high(         keep=dtm u95 rename=(u95=forecast_cr));

set predict;
output actual;
if dtm<=&startholdout then do;
    output onestep_low;
    output onestep_high;
end;
if dtm>&startholdout and dtm<=&numrows then do;
    output holdout_low;
    output holdout_high;
end;
if dtm>&numrows then do;
    output forecast_low;
    output forecast_high;
end;
run;
```

This data step results in seven data sets that will be manipulated individually and then put back together to create the plot. Note the fact that each pair of low and high datasets will contain the same variable name for both lower and upper confidence band. The reason for this is that the INTERPOL=MS option of the SYMBOL statement will plot the values associated with it as a polygon. The lower bounds are sorted in descending order while the upper remain in the original order (should be ascending) so that the upper and lower bounds can be drawn in the same way that you would draw any polygon without removing your pen from the paper. It is a “connect the dots” scenario and if they are not sorted in this way a mess is generated.

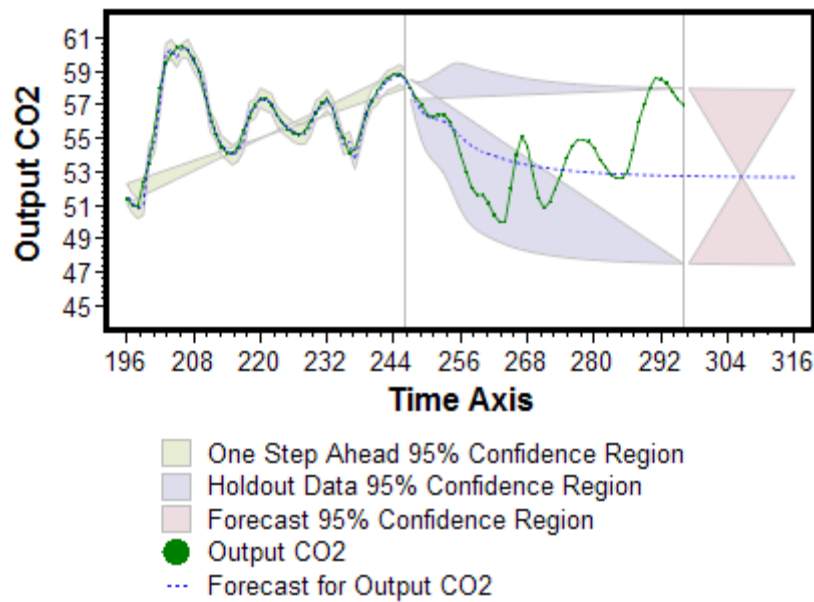


Figure 2

```
proc sort data=onestep_low;          by descending dtm; run;
proc sort data=holdout_low;         by descending dtm; run;
proc sort data=forecast_low;       by descending dtm; run;
```

Once this is done, the data is put back together in a sorted form by stacking each dataset on top of each other. There will be a lot of missing values, but this does not cause any problems as they are not printed.

```
data plotdata;
  set
    actual
    onestep_low
    onestep_high
    holdout_low
    holdout_high
    forecast_low
    forecast_high
  ;
  label onestep_cr="One Step Ahead 95% Confidence Region";
  label holdout_cr="Holdout Data 95% Confidence Region";
  label Forecast_cr="Forecast 95% Confidence Region";
  label forecast="Forecast for Output CO2";
run;
```

SETTING UP THE SAS/GRAPH OPTIONS

```
goptions reset=all device=activex;
symbol1
  interpol=ms          /* map/plot pattern fill */
  cv=ywh               /* pattern fill color   */
```

```

        co=ligr          /* outline color          */
        value=none; /* type of point */
symbol2
    interpol=ms
    cv=bwh
    co=ligr
    value=none;
symbol3
    interpol=ms
    cv=pkwh
    co=ligr
    value=none;
symbol4
    interpol=join /* join the points together */
    color=green /* line and point color */
    value=dot /* type of point */
    line=1 /* line style */
    width=.5 /* line width */
    height=.25; /* size of the point */
symbol5
    interpol=join
    color=blue
    value=none
    height=.5
    line=2
    width=.5;
axis1
    order=(&plotstart to &forecastend by %sysevalf((&forecastend-&plotstart)/10))
    label=("Time Axis") /* label the x axis */
    major=(height=2) /* specify the characteristics of the major ticks */
    minor=(number=4 height=1) /* specify the characteristics of the minor ticks */
    offset=(2,2) /* put some space at either end of the axis */
    width=3; /* specify the weight of the axis line */
axis2
    order=(&min_yaxis to &max_yaxis
           by %sysfunc(ceil(%sysevalf(%eval(&max_yaxis-&min_yaxis)/10))))
    label=("Output C02" angle=90)
    major=(height=2)
    minor=(number=4 height=1)
    offset=(2,2)
    width=3;

legend1 across=1;

```

PLOTTING THE FORECAST

The graphic options should be reset every time any SAS/GRAPH procedure is used. See the SAS documentation for GOPTIONS for more selections and information. The SYMBOL statement is used to define each different type of plot object that is representing a data object. In this example, symbols 1-3 are the confidence interval regions and symbols 4 and 5 will show the actual values and forecasts respectively.

```

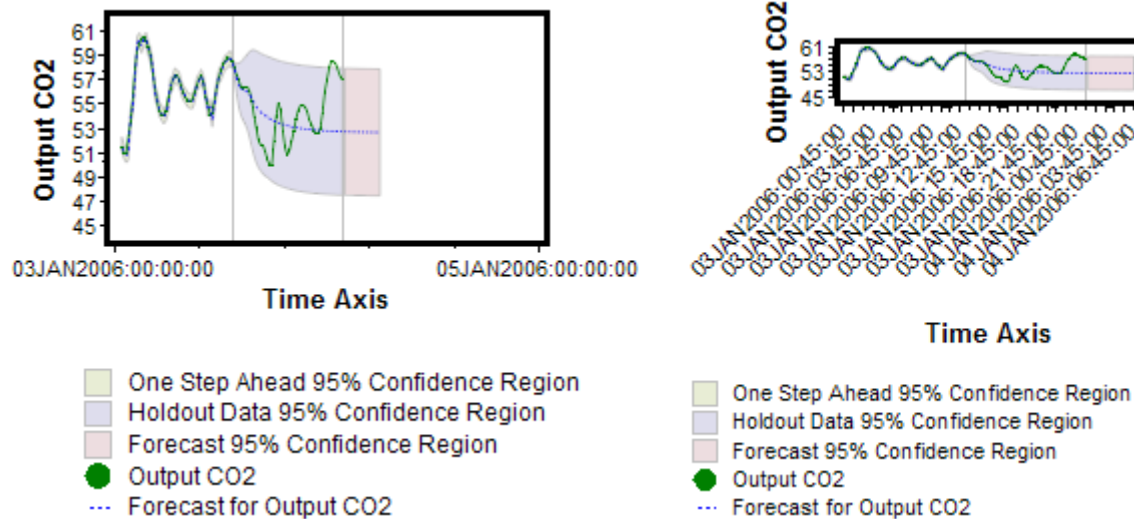
proc gplot data=plotdata;
  plot
    onestep_cr*dtm=1      /* plot the one step ahead region with symbol1 */
    holdout_cr*dtm=2      /* plot the holdout region with symbol2 */
    forecast_cr*dtm=3     /* plot the forecast region with symbol3 */
    y*dtm=4               /* plot the data with symbol4 */
    forecast*dtm=5        /* plot the forecasts with symbol5 */
  / overlay               /* plot all of the above on the same graph */
  haxis=axis1             /* use axis1 for the horizontal axis */
  legend=legend1          /* put a legend on the graph */
  vaxis=axis2             /* use axis2 for the vertical axis */
  href=(&startholdout &numrows) /* print vertical lines at the holdout and the
                                forecast start */
  chref=ligr              /* color the vertical reference lines light grey */
  ;
run;
quit;

```

ADDITIONAL CONCERNS

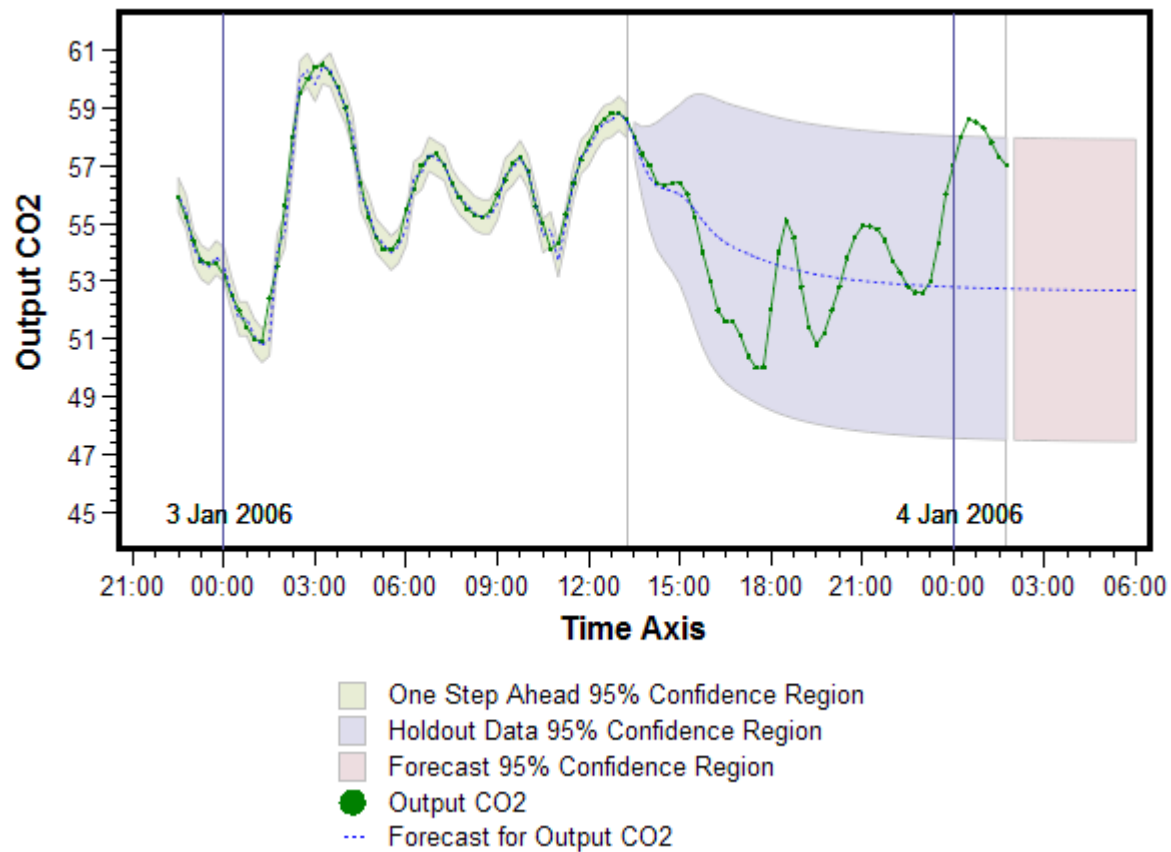
DEALING WITH COMPLICATED DATETIME INTERVALS

SAS DATETIME formats are very difficult to display concisely on an axis. The length of the string representation is too long, and using major and minor tick marks can also make the graph too cluttered, or present too little information for use. The axis values can also be angled, but the result is most often far from satisfactory.



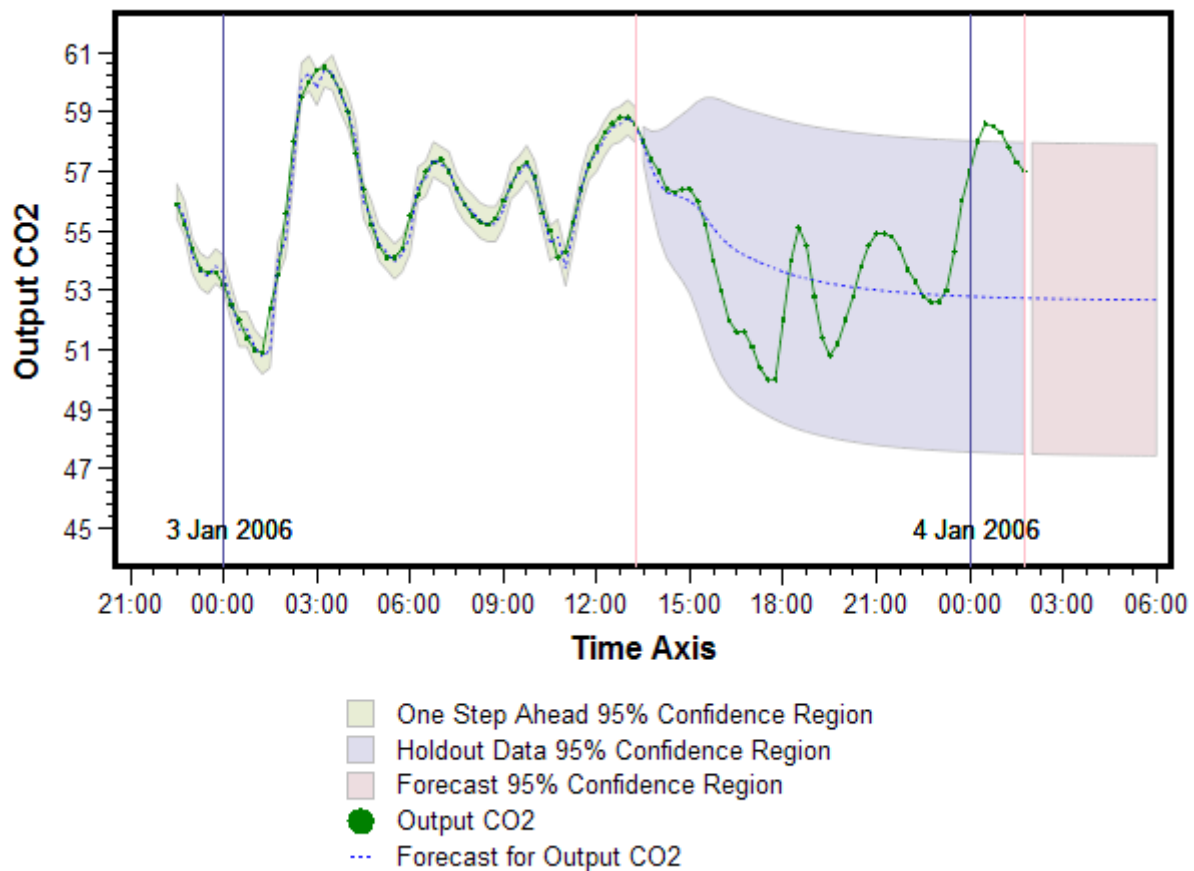
One way to deal with this is to present only the time values on the X-axis and annotate the graph to show the date values. This annotation is done using the ANNOTATE facility. The code used to annotate this graph is shown below:

```
data DayLines; set plotdata(keep=dtm );
  length color function $8 text $14;
  retain xsys '2' ysys '1' when 'a';
  if hour(dtm)=0 and minute(dtm)=0 and dtm>=&plotstart then do;
    wdate=put(datepart(dtm),worddatx12.);
    function='move'; x=dtm; y=0; output;
    function='draw'; x=dtm; y=100; color='lib'; size=1; output;
    function='label'; x=dtm; y=5; size=1;
  position='2';angle=90;color='black'; text=wdate; output;
  end;
run;
```



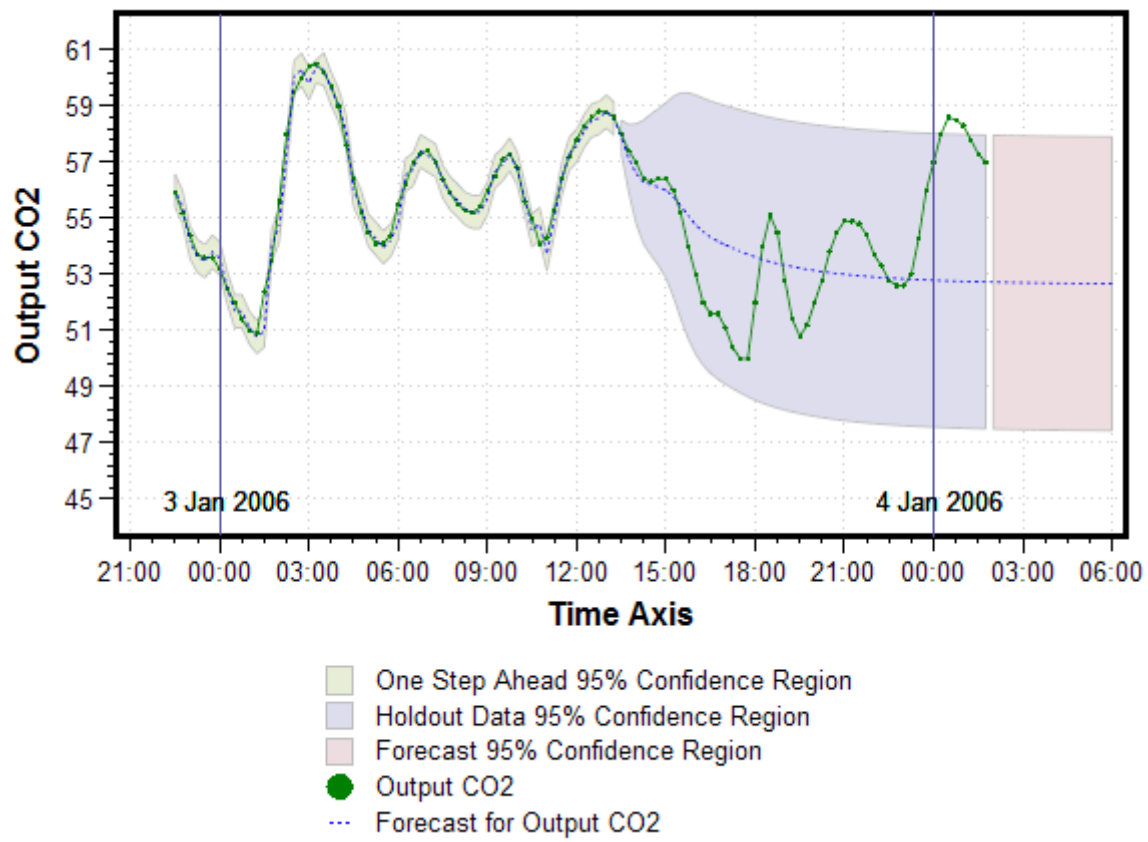
The same could be done with other sub-intervals such as hour-minute-second issues. The same annotate data set can also be used to draw other data driven objects on the graph. The following annotate data set could be used to replace the HREF statement in the PLOT command of PROC GPLOT:

```
data DayLines; set plotdata(keep=dtm );
  length color function $8 text $25;
  retain xsys '2' ysys '1' when 'a';
  if hour(dtm)=0 and minute(dtm)=0 and dtm>=%eval(&plotstart-8100) then do;
    wdate=put(datepart(dtm),worddatx12.);
    function='move'; x=dtm; y=0; output;
    function='draw'; x=dtm; y=100; color='lib'; size=1; output;
    function='label'; x=dtm; y=5; size=1;
    position='2';angle=90;color='black'; text=wdate; output;
  end;
  if dtm=&startholdout or dtm=&numrows then do;
    function='move'; x=dtm; y=0; output;
    function='draw'; x=dtm; y=100; color='pink'; size=1; output;
    output;
  end;
run;
```



In reality, these reference lines are not necessary since the regions address the start of holdout and forecast data. It is probably therefore better to leave them out. Also, a reference grid can be added in the options section of the PLOT section of PROC GPLOT:

```
proc gplot data=plotdata annotate=daylines;
  plot
    onestep_cr*dtm=1
    holdout_cr*dtm=2
    forecast_cr*dtm=3
    y*dtm=4
    forecast*dtm=5
  /
    overlay
    haxis=axis1
    legend=legend1
    vaxis=axis2
    GRID
  ;
  format dtm tod5.;
run;
quit;
```

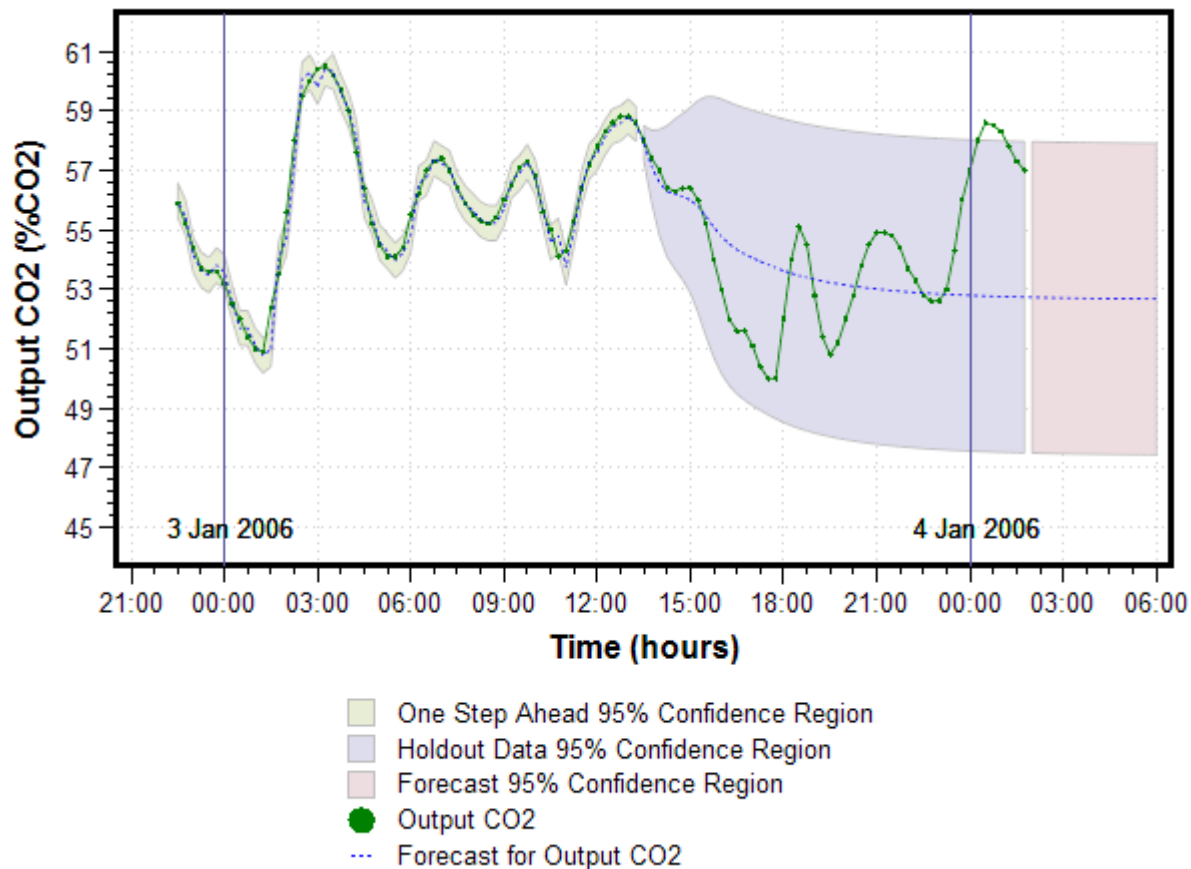


TITLES AND LABELS

There are many different ideas about how to title and label graph elements. It is better to keep everything as simple as possible but it needs to be understandable. A graph should be able to stand by itself without much in the way of narrative so the units of the x and y axis must be clear.

Forecast Visualization Demonstration

Box-Jenkins Series J with Surrogate Time Axis



CONCLUSION

Effective forecast visualization is a difficult task primarily due to the fact that the audience is often a non-technical group. It is very important for any visualization to be very accessible to a non-technical audience so it is necessary to keep things as simple as possible. This example used colored confidence regions to show the upper and lower bounds on the one-step ahead, holdout forecast and full forecast, but the same technique can be extended to many different types of graphs. The alternative is to plot these bounds as more lines or points on the graph that make the result more cluttered and difficult to interpret.

REFERENCES

Box, G. E. P, Jenkins, G. M., and Reinsel, G. C. (1994), *Time Series Analysis: Forecasting and Control*, Englewood Cliffs, NJ: Prentice Hall, Inc.

Brocklebank, John C., and Dickey, David A. (1994), *SAS for Forecasting Time Series*, Cary, NC: SAS Institute/John Wiley and Sons;

ACKNOWLEDGMENTS

Thanks to Shane Hornibrook for encouraging me to document this process.

RECOMMENDED READING (HEADER 1)

SAS/Graph Colors <http://www.devenezia.com/docs/SAS/sas-colors.html>

SAS Sample <http://support.sas.com/ctx/samples/index.jsp?sid=1151&tab=about>

CONTACT INFORMATION

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

Samuel T. Croker
252 Kyzer Road
Lexington, SC 29073
Work Phone: 803-240-2805
Email: scoyote@scoyote.net
Web: www.scoyote.net

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.