

## Extending ODS Output by Incorporating Trellis™ Graphics from S-PLUS

Robert Treder, Ph. D., Insightful Corporation, Seattle WA  
Jagrata Minardi, Ph. D., Insightful Corporation, Seattle WA

### ABSTRACT

In the past decade many new ideas in graphical displays (Cleveland, 1993) have made their way into common usage in technical papers and reports, especially in the pharmaceutical industry. In particular, the Trellis™ graphics found in S-PLUS provide a level of detail in a multi-dimensional context that no other display can match in clarity or graphical brevity. Though these displays are not available in SAS GRAPH, they may be incorporated in standard SAS ODS output by running S-PLUS alongside SAS and incorporating S-PLUS graphical output into the ODS report. This extends the possibilities of an ODS production report in a substantial way. This paper demonstrates the simple mechanism for incorporating graphics from another non-SAS system into a standard ODS report and describes important examples using Trellis graphics in pharmaceutical studies.

### 1. INTRODUCTION TO TRELIS GRAPHICS

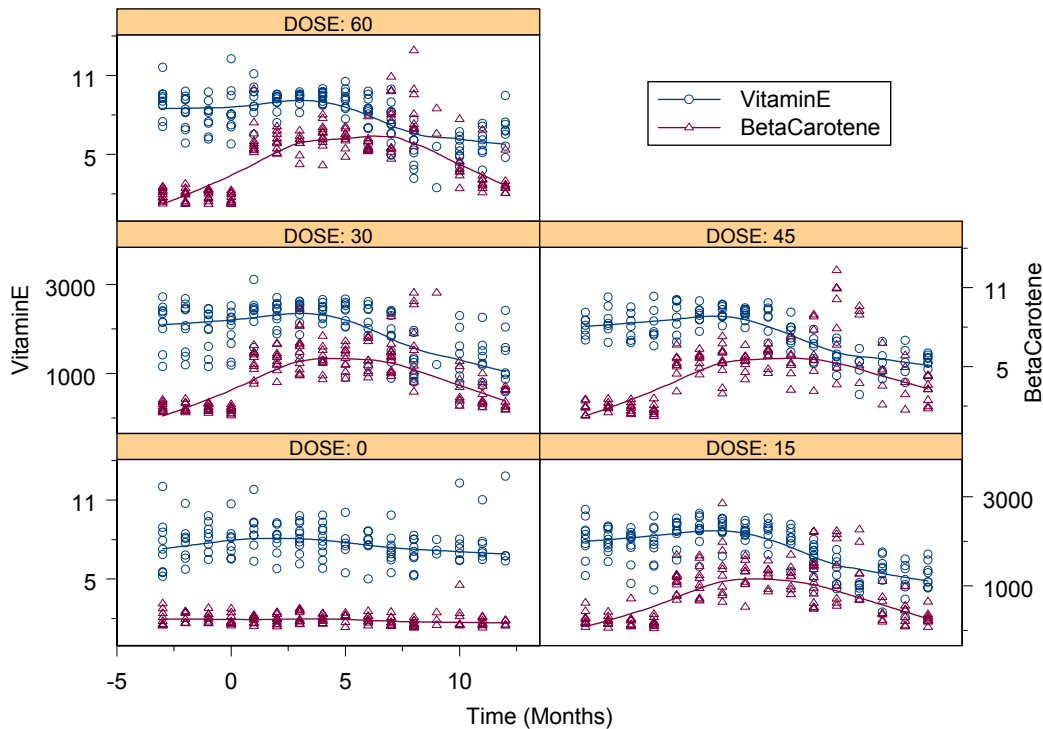
Trellis graphs let you view relationships between different variables in a data set through conditioning. A series of panels is displayed, with each panel containing a subset of the data based on subsets of values of one or more conditioning variable. For discrete-valued or categorical conditioning variables the panels contain data corresponding to each discrete value or category. For numeric conditioning variables the panels contain data corresponding to subintervals of the range of the conditioning variables.

### CASE STUDY

The graph in Figure 1 is a trellis graph of plasma beta carotene and vitamin E levels from a double-blind study to determine the effect of long-term supplementation of beta carotene on plasma vitamin E levels believed tied to neoplastic diseases. Each panel shows the data for a different dose: 0, 15, 30, 45 and 60 mg/day of oral beta carotene. Both vitamin E and beta carotene levels are plotted in each panel with separate axes for each measure. The curve overlaid in each graph and for each measure results from applying a locally weighted regression smoother, loess, (Cleveland and Devlin, 1988) which estimates the average plasma levels of the measures as a function of time.

Figure 1 clearly shows, for all treatment groups except the placebo, a gradual build up of plasma beta carotene levels and a subsequent decrease of vitamin E levels starting at around the fourth or fifth month of treatment. For the placebo group there is no change in plasma levels of either beta carotene or vitamin E.

The power of the graphic in Figure 1 is clear. All treatment groups are visually compared to the placebo group over time. By including both plasma measures in each panel we can see a build-up of beta carotene is obtained before any visible decrease in vitamin E levels occurs.



**Figure 1:** Trellis display of data from double-blind study to determine the effect of long-term supplementation of beta carotene on plasma vitamin E levels.

With the data loaded and checked the graph in Figure 1 took about 5 minutes to create in S-PLUS using the GUI. The code to recreate the graph is easily captured (based on the GUI operations) in a script which can be run as a batch process to create the graphic and output it through a SAS ODS reporting procedure.

We now describe several more examples of Trellis graphs and then describe the procedure for generating a report including the S-PLUS graphics using SAS ODS.

### MULTIPANEL DISPLAYS

Suppose you have a data set containing measurements on several variables, and you want to see how plots of two variables change in relation to a third *conditioning* variable. A Trellis graph is a series of panels in which each panel contains a subset of the original data. The subsets of the plotted data are based on subsets of values of the conditioning variable. In the example in Figure 1, the conditioning variable is dose and the value of dose for each panel corresponds to the unique doses in the study. When a conditioning variable is categorical, S-PLUS automatically generates plots for each level. When a conditioning variable is numeric, conditioning is automatically based on the sorted unique values. However, when the values of a numeric variable are more than just a few, we typically create interval subsets for conditioning. That is, we turn the numeric variable into a categorical variable and then condition on the category levels.

## 2. EXAMPLES

We illustrate the main options for multi-panel conditioning using the beta carotene study data. It is stored in a data set named `BetaCarotene`. The variables are

- TIME:** Time in months.
- DOSE:** Daily dose of beta carotene in milligrams.
- BC:** Blood plasma levels of beta carotene.
- VitaminE:** Blood plasma levels of vitamin E.

### CONDITIONAL SCATTERPLOT

Figure 2 uses multi-panel conditioning to display the `BetaCarotene` data. Each panel displays the plasma concentrations of vitamin E over time for one dose level. `VitaminE` is graphed along the vertical scale and `TIME` is graphed along the horizontal scale. For example, the lower left panel displays values of `VitaminE` and `TIME` for the 0-dose group. The *panel variables* are `VitaminE` and `TIME`, and the *conditioning variable* is `DOSE`.

The specification for this plot is done with a call to the `xyplot` function in S-PLUS.

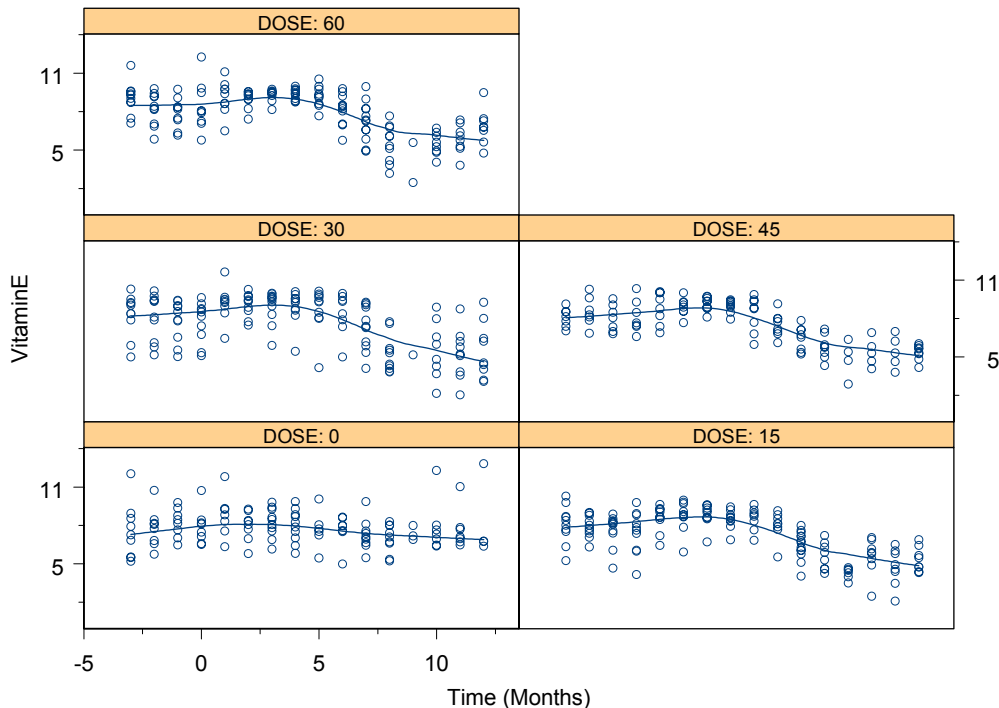
```
xyplot(VitaminE ~ TIME | DOSE, data = BetaCarotene,  
       panel = panel.smooth)
```

The `xyplot` function creates conditional scatterplots. The function takes a *formula* specifying three variables, the response (*y*-axis), the main covariate (*x*-axis) and the conditioning variable (for creating the subset in each panel). In this example the formula is

```
VitaminE ~ TIME | DOSE
```

We read the formula from left to right saying, "Plot `VitaminE` vs `TIME` conditional on `DOSE`." The *y*-axis variable is to the left of the tilde (`~`) and the *x*-axis variable is to the right. The conditional variable is to the right of the vertical bar (`|`).

The function also takes the dataset name, `BetaCarotene`, and a function specifying the action in each panel. The `panel.smooth` function used in this case plots the (*x*, *y*) pairs in each panel and overlays a loess smooth to estimate the relation



**Figure 2:** Trellis display of data from double-blind study to determine the effect of long-term supplementation of beta carotene on plasma vitamin E levels. Each panel contains the data for a given dose level.

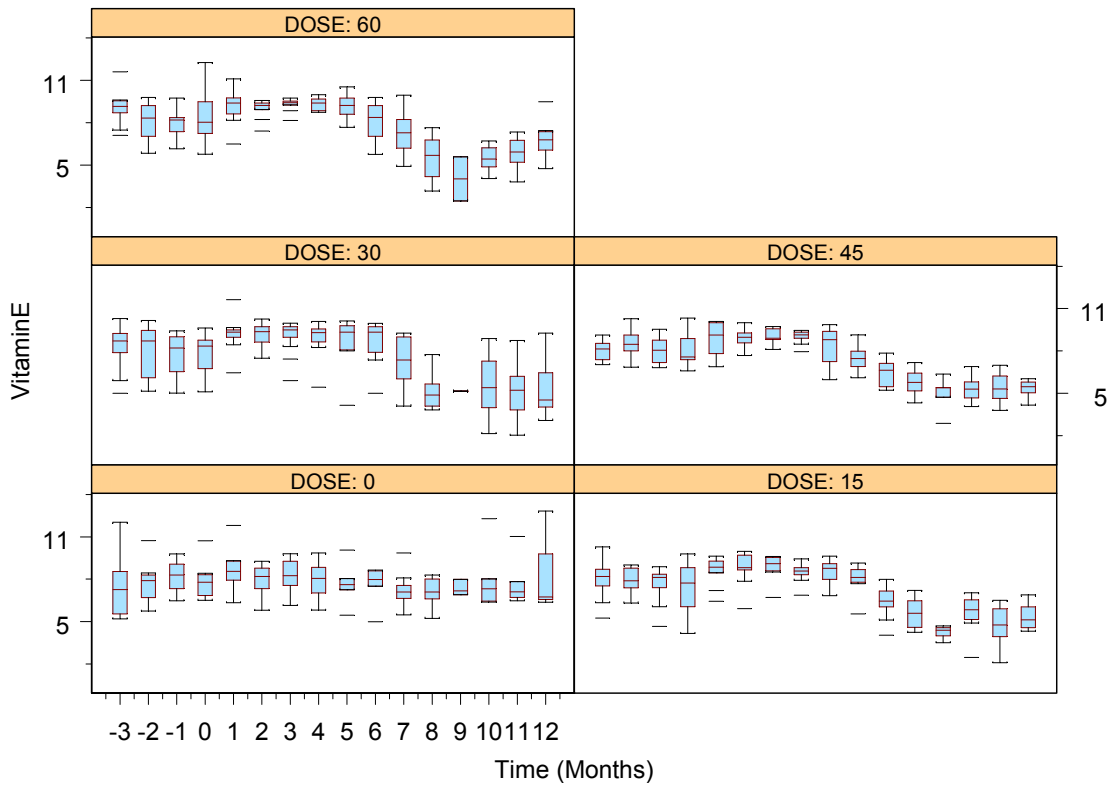
### CONDITIONAL BOX AND WHISKER PLOT

Instead of a scatterplot we can create conditional box and whisker plots to get a better sense of the distribution of points at each time and for each dose. These plots can be created through either the GUI or the command line as we have seen for scatterplots.

The syntax is slightly different than for `xyp1ot` when creating the boxplot with the command line. We call a different function, `bwplot`, an abbreviation for box and whisker plot and the boxes are arranged horizontally rather than vertically so **TIME** becomes the y-axis variable.

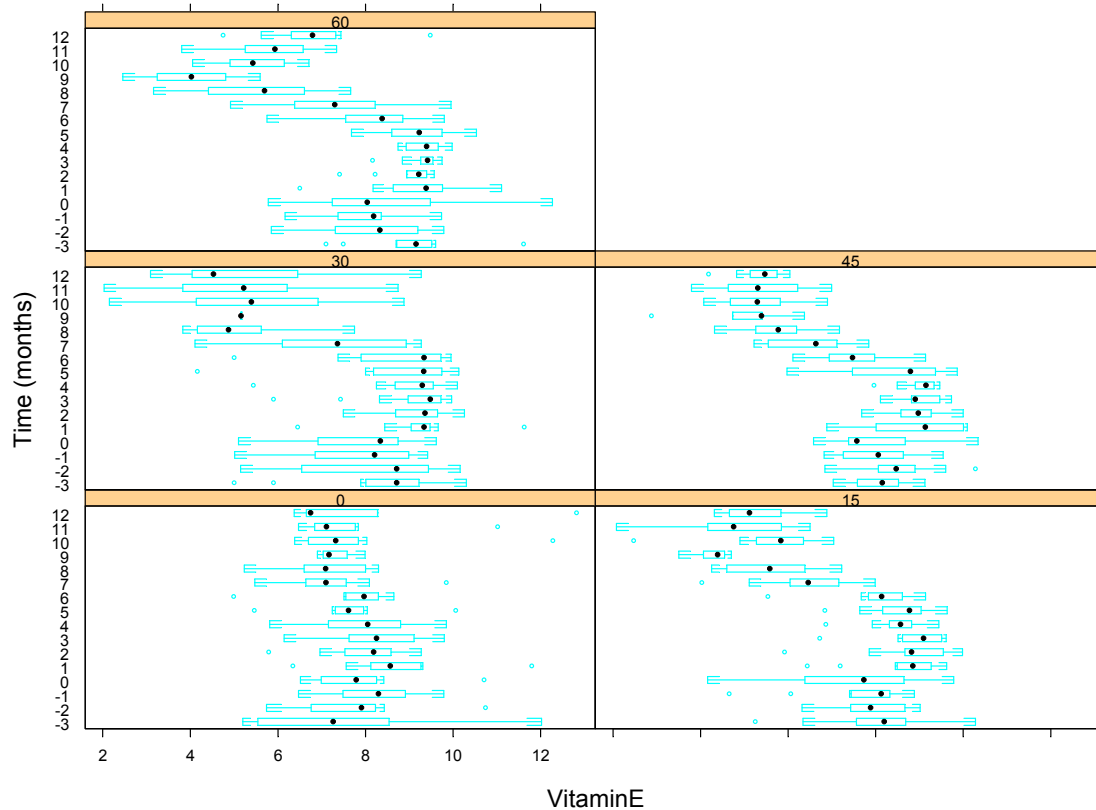
```
bwplot(TIME ~ VitamineE | DOSE, data = BetaCarotene)
```

The conditional boxplots created through the GUI are displayed in Figure 3. The conditional boxplots created through the command line are displayed in Figure 4. Both have options for controlling layout, fonts, and box shading color, plus many other options.



**Figure 3:** Conditional boxplots of plasma vitamin E levels for the beta carotene study created with the S-PLUS GUI.

Note that in Figure 4, since the time axis is vertical, time increases going from the bottom to the top of the graph. It is as clear with the boxplot display as it is with the scatterplots that plasma vitamin E decreases over time for all the treatments with non-zero dose.



**Figure 4:** Conditional boxplots of plasma vitamin E levels for the beta carotene study, created with the S-PLUS command line.

#### CONDITIONAL DOTPLOT

We can look at the same data with yet another display, the dotplot. This plot requires having a single value for each dose and time combination. We can create a dotplot of means by first computing the mean vitamin E concentration for dose and time combination. In S-PLUS we do that using the `tapply` function as follows:

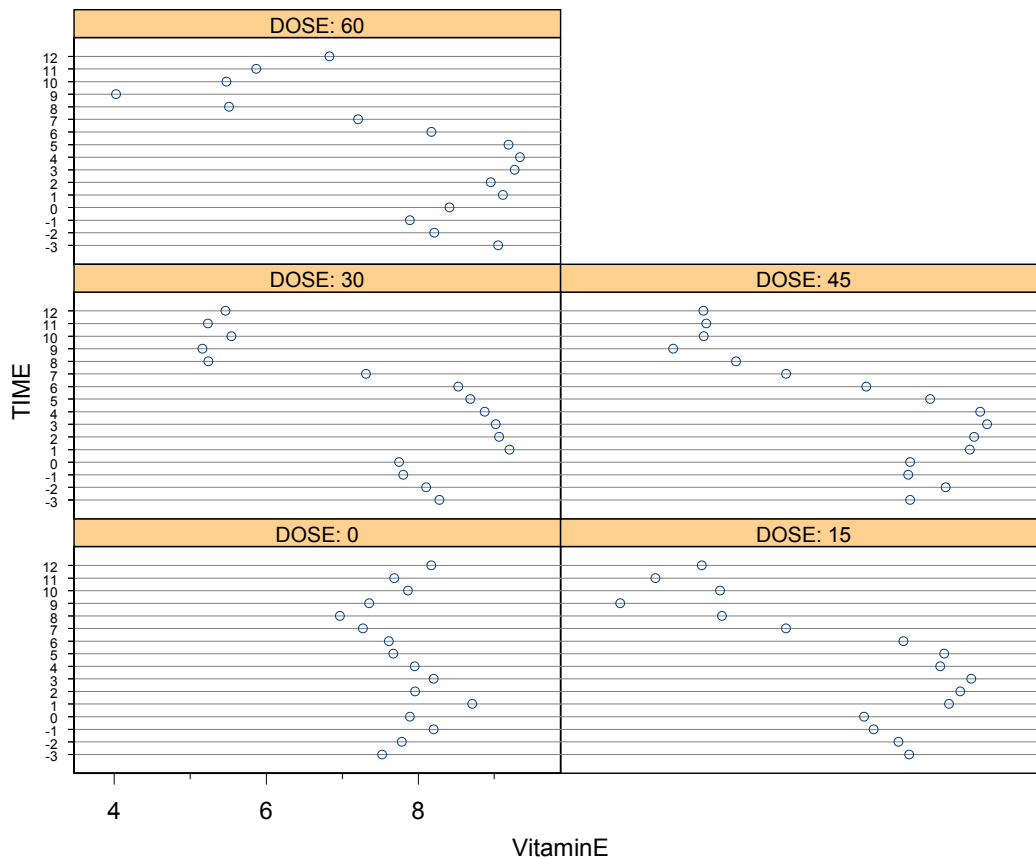
```
tapply(VitaminE, list(TIME, DOSE), mean, na.rm = T)
```

`tapply` takes a vector of values, `VitaminE`, and creates a two-dimensional table filled with the mean vitamin E concentration for each `TIME` and `DOSE` combination. The `na.rm = T` argument removes missing values when computing the mean for each cell.

Having constructed a data set `meanVite` containing the mean vitamin E concentrations along with the corresponding measurement times and doses, we can create the dotplot as follows:

```
dotplot(TIME ~ VitaminE | DOSE, data = meanVite)
```

There is little difference between the dotplot created through the command line and the one created through the GUI. Figure 5 shows the resulting plot.



**Figure 5:** Conditional dotplots of plasma vitamin E levels for the beta carotene study created with the S-PLUS command line.

### GROUPED DATA PLOT

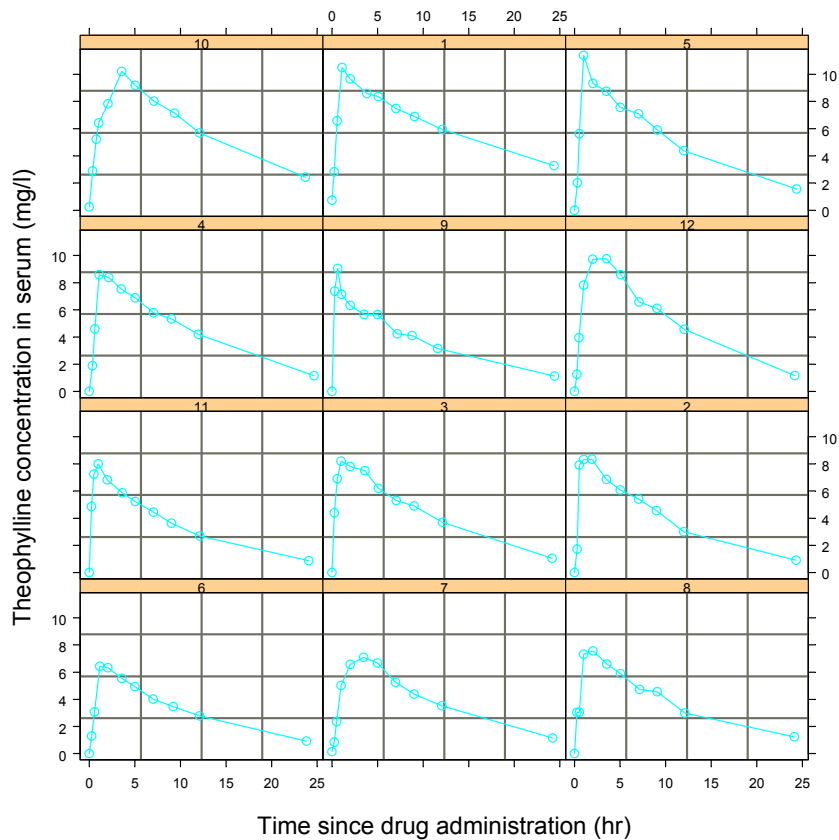
Trellis graphics are useful not only for performing exploratory data analysis but also for displaying model results. A number of modeling functions make use of Trellis displays because the data structure includes grouping variable(s). One example is data used in fitting mixed effects models resulting from taking repeated measurements on the same subjects.

The data we will consider is from a study of the kinetics of the anti-asthmatic drug Theophylline. Concentration levels over time were studied on 12 subjects. In S-PLUS we create a groupedData object with a formula like the ones we have seen for creating Trellis graphs, a data set, and labels for graphs and other output. The formula is of the form

```
resp ~ cov | group
```

where **resp** is the response variable, **cov** is the main covariate for plotting and model fitting and **group** is the grouping variable. Once the groupedData object is set we can easily generate exploratory plots, models and plots of model predictions with functions that take advantage of the grouping structure supplied by the formula. For example a plot of the original data with observations connected by a straight line for each subject (Figure 6) is created with the simple plot call. This simple command is possible because there is a plot method specific to grouped data objects.

```
plot(Theoph)
```



**Figure 6:** Plot of grouped data automatically creates a Trellis plot with separate panels for each group member.

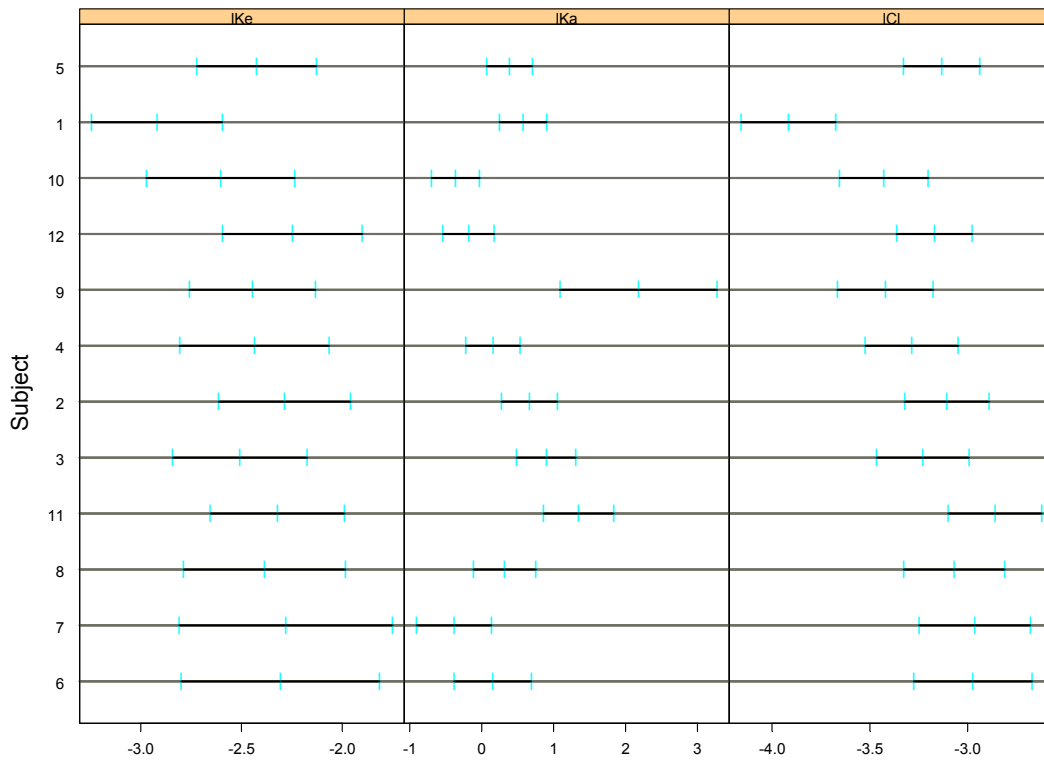
We can quickly fit a first order compartment model to all fixed effects with separate coefficient estimates for each subject, using the `nlsList` function. We specify a formula indicating the response, the first order compartment modeling function (`SSfo1`) and the covariates.

```
fmTheo.lis <- nlsList( conc ~ SSfo1(Dose, Time, lKe, lKa, lCl),
  data = Theoph )
```

With the fit object we can generate a plot of the confidence intervals (Figure 7) for all the estimated coefficients for each subject by simply doing

```
plot(intervals(fmTheo.lis))
```

The simplicity of the call is due to a plot method for the `intervals` function. Figure 7 shows the resulting plot.



**Figure 7:** Plot of confidence intervals for fixed effects estimates for each subject.

Now we can fit a mixed effects model starting from the saturated fixed effects model specifying block diagonal covariance structure.

```
fmTheo.nlme <- nlme(fmTheo.lis,
                    random = pdDiag(lKe + lCl ~ 1) )
```

The plot of the resulting fit is obtained by the call

```
plot(augPred(fm3Theo.nlme, level = 0:1))
```

The `augPred` function generates fixed and random effects predictions for mixed effects models. The plot method overlays the predicted cures in separate panels for each subject along with the observations.

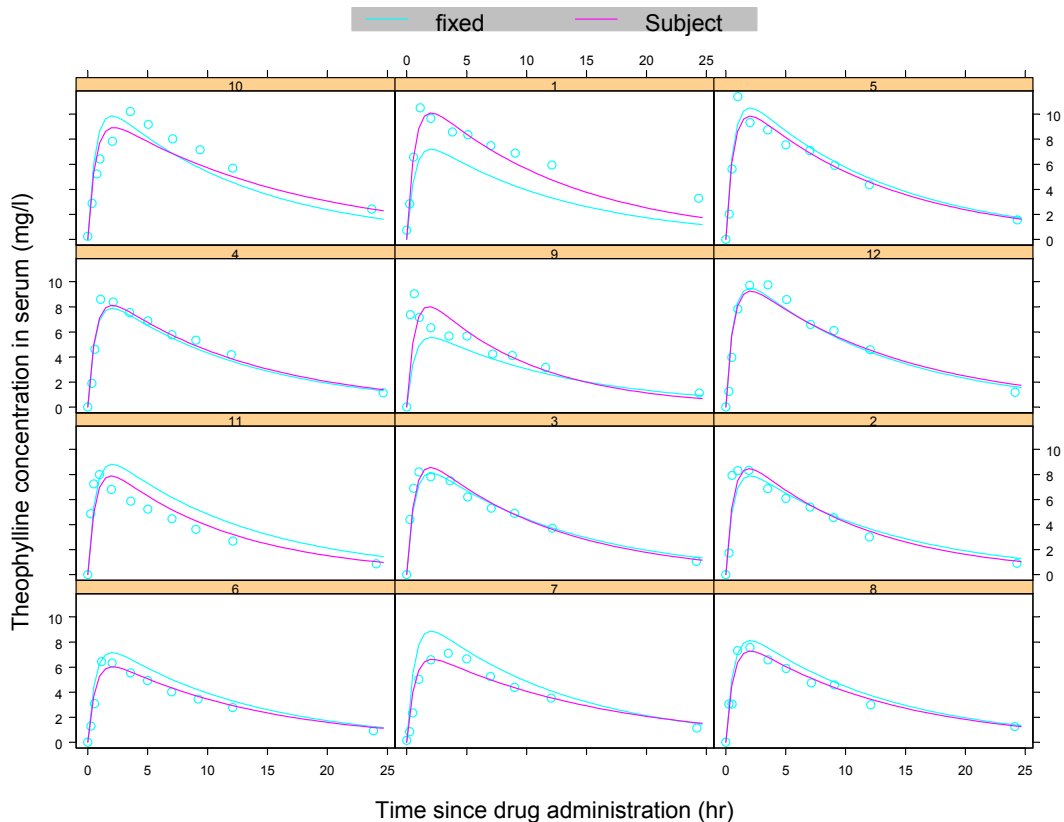


Figure 8: Trellis plot of average fixed and random effects fits for each subject in the Theopylline study.

### 3. INCORPORATING S-PLUS GRAPHICS IN AN ODS REPORT

Incorporating S-PLUS graphics into an ODS report is straightforward once you learn the basic structure of the commands. The critical pieces are

1. Create an ODS template defining a pre or postimage filename used for capturing the S-PLUS graphs.
2. Create an S-PLUS script file with the commands to create the graphics. This script file, run in batch, will need to write the output graphics to the file specified in the ODS template.
3. Create a system call in your SAS code that will
  - a. Define an environment variable with the dataset name that will be used in the S-PLUS script to locate the SAS data set.
  - b. Run the S-PLUS script in batch mode.
4. Create an ODS report using the template defined in Step 1.

#### ADDING AN IMAGE BEFORE OR AFTER A TABLE IN AN RTF DOCUMENT

In this first example we add a figure after a table of means. The critical specification is the style definition in **proc template**. In this example we set **postimage** to **bcplot.jpg**, the file name containing the graphic that will be generated by S-PLUS. Note also that **dataname** is set as an environment variable to specify the prefix of the data file name "betacarotene" which S-PLUS uses to locate the data file for reading.

```
libname ph '.';

OPTIONS ORIENTATION = landscape nonumber nodate ;

proc template;
  define style styles.test;
  parent=styles.rtf;
  style table from table /
    postimage='bcplot.jpg';
```

```

end;
run;

%LET dataname = 'betacarotene';

X "setenv sasdata &dataname; Splus BATCH bcplot.ssc /dev/null -j";

ods rtf file='pharmSUG2.rtf' style=styles.test;

proc tabulate data=ph.BetaCarotene;
  class DOSE TIME;
  var BC;
  format TIME 3.;
  table DOSE, TIME*MEAN*BC;
  title 'Mean of BetaCarotene by Dose and Time';
run;

ods rtf close;

```

The S-PLUS batch job is:

```

bcdata <- importData(paste(getenv('sasdata'),'sas7bdat',
  sep='.'))

java.graph(file="bcplot.jpg",format='JPG')

xyplot(BC ~ TIME | DOSE, data=bcdata, panel = panel.smooth)

dev.off()

```

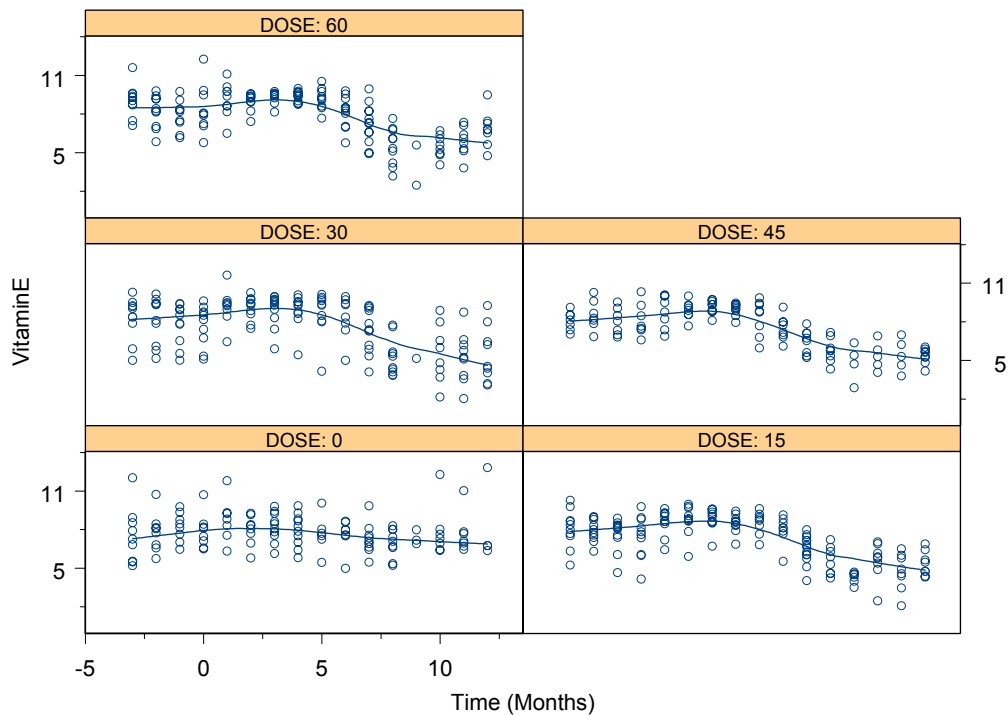
All the work is done in the call to the **xyplot** function. The first argument to xyplot,

```
BC ~ TIME | DOSE
```

specifies to plot BC as a function of TIME conditional on DOSE.

The resulting report contains the table from proc tabulate and the graphic from S-PLUS.

	TIME															
	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12
	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean	Mean
DOSE	BC	BC	BC	BC	BC	BC	BC	BC	BC	BC	BC	BC	BC	BC	BC	BC
0	272.11	265.00	283.25	245.25	235.25	227.75	294.30	258.40	267.83	256.43	196.11	150.44	246.33	312.57	179.14	165.25
15	257.90	205.80	232.80	172.20	1080.80	1116.10	1388.70	1218.11	1177.50	1192.25	1167.86	1345.13	1360.00	500.63	394.38	292.38
30	228.50	229.20	242.80	176.90	1242.80	1309.40	1505.54	1324.20	1348.20	1339.67	1370.67	1662.10	2811.00	689.38	518.33	429.67
45	235.14	255.88	228.63	194.50	1187.38	1302.88	1272.27	1269.22	1387.75	1260.63	1529.63	2167.13	1836.80	967.00	811.40	599.13
60	232.80	225.70	236.67	188.22	1556.33	1445.89	1397.78	1587.27	1509.78	1385.64	1857.17	2078.10	1897.00	917.13	648.13	509.63



#### ADDING AN IMAGE AT THE BEGINNING OF AN RTF DOCUMENT

In SAS 8.2 ODS RTF, you can add an image to the beginning of the file by using in-line formatting. The idea, from SAS Technical Support, is to add an image into the TITLE1. For subsequent procedures or DATA steps, remove TITLE1 so the image only appears once. A sketch of the macro code is shown below.

```
ods escapechar='^';
ods rtf file='pharmSUG2.rtf' style=styles.test;

/* call S-PLUS as before */
X "setenv sasdata &dataname; Splus BATCH bcplot.ssc /dev/null -j";

proc print data=ph.BetaCarotene;
  title j=1 '^S={preimage="bcplot.jpg"} ';
run;

/* invoke SAS procedures */

ods rtf close;
```

#### ADDING AN IMAGE TO AN HTML DOCUMENT

Adding an image to an HTML document is straightforward. You can create a link to a file that contains an S-PLUS JPEG or GIF graphic, or even a dynamic S-PLUS java graphlet. Create S-PLUS graphic file from within the SAS macro exactly as shown in the first example of this section. A sketch of the macro code is shown below.

```
/* do not write the closing tags */
ODS HTML BODY=report (no_bottom_matter);

/* invoke SAS procedures here . . .*/
ODS HTML CLOSE;
```

```

/* call S-PLUS */
X "setenv sasdata &dataname; Splus BATCH bcplot.ssc /dev/null -j";

/* append to HTML file */
FILENAME report 'pharmSUG2.html' MOD;

DATA _NULL_;
  FILE report;
  PUT '<hr>';
  PUT '';
RUN;

/* write the closing tags */
ODS HTML FILE=report (no_top_matter);
ODS HTML CLOSE;

```

#### 4. CONCLUSIONS

Use of S-PLUS to enhance SAS ODS reports is straightforward using **proc template** to create a **style** which incorporates graphics from S-PLUS or other programs for that matter. One constraint we discovered is that the graphic formats that SAS ODS allows from third-party software does not include Windows Metafiles (wmf). Windows Metafiles are vector-based graphic descriptions which result in displays with higher resolution. Except for this shortcoming, the mechanism described provides a convenient procedure for incorporating S-PLUS graphics in ODS output.

#### 5. REFERENCES

William S. Cleveland. (1993) *Visualizing Data*. Hobart Press, Summit, NJ. 360 pp.

Cleveland, W.S., and Devlin, S.J., (1988) Locally-weighted Regression: An Approach to Regression Analysis by Local Fitting. *J. Am. Statist. Assoc.*, Vol. 83, pp 596-610.

Insightful Corporation, (2001) *S-PLUS 6 For Windows Programmer's Guide*. 986 pp.

SAS Institute, Inc. (1999) *Base SAS Software*.

#### CONTACT INFORMATION

**Author Name:** Robert Treder  
**Company:** Insightful Corporation  
**Address:** 1700 Westlake Ave. N, Suite 500  
**City/State/Zip:** Seattle WA 98109  
**Work Phone:** (206) 802-2231  
**Fax:** (206) 283-8691  
**E-mail:** bob@insightful.com  
**Web:** [www.insightful.com](http://www.insightful.com)

**Author Name:** Jagrata Minardi  
**Company:** Insightful Corporation  
**Address:** 1700 Westlake Ave. N, Suite 500  
**City/State/Zip:** Seattle WA 98109  
**Work Phone:** (206) 802-2236  
**Fax:** (206) 283-8691  
**E-mail:** jminardi@insightful.com  
**Web:** [www.insightful.com](http://www.insightful.com)