Paper 137-2017

Making Graphs Easier to Validate - The Benefits of ODS Graphics

Philip R Holland, Holland Numerics Limited, UK

ABSTRACT

The days of comparing paper copies of graphs on light boxes are long gone, but the problems associated with validating graphical reports still remain. Many recent graphs created using SAS/GRAPH® include annotations, which complicate an already complex problem. With the restriction in ODS Graphics that only a single input data set should be used, and 'annotation' can be more easily added by overlaying an additional graph layer, it is now more practical to use that single input data set for validation, which removes all of the scaling, platform and font issues that got in the way before. This paper will guide you through the techniques to simplify validation while you are creating your perfect graph.

INTRODUCTION

The paper aims to demonstrate that even a complex graph can, if generated using multiple SAS input data sets, can be validated simply by considering the individual SAS input data sets separately. ODS Graphics procedures require only a single input data set, so all of the data used to create any graph must exist in that data set. However, as sparse data is perfectly acceptable, ODS Graphics will ignore any missing data values, so concatenating multiple data sets containing different variables can create a suitable input data set.

In SAS/GRAPH any annotations are achieved using a second data set, but, while this is also possible in ODS Graphics, it is generally unnecessary, as there are techniques available to use input data to add extra features to any graph.

BASIC INPUT DATA FOR BASIC GRAPHS

Throughout this paper I will save the input data set to a permanent library prior graph production, after completing all of the necessary data manipulation, and then use that saved version to create the graph, so there is a proven link between the data and the graph.

The other principle used will be make certain that there are only data values in the input data set that are useful for either graph production or for validation.

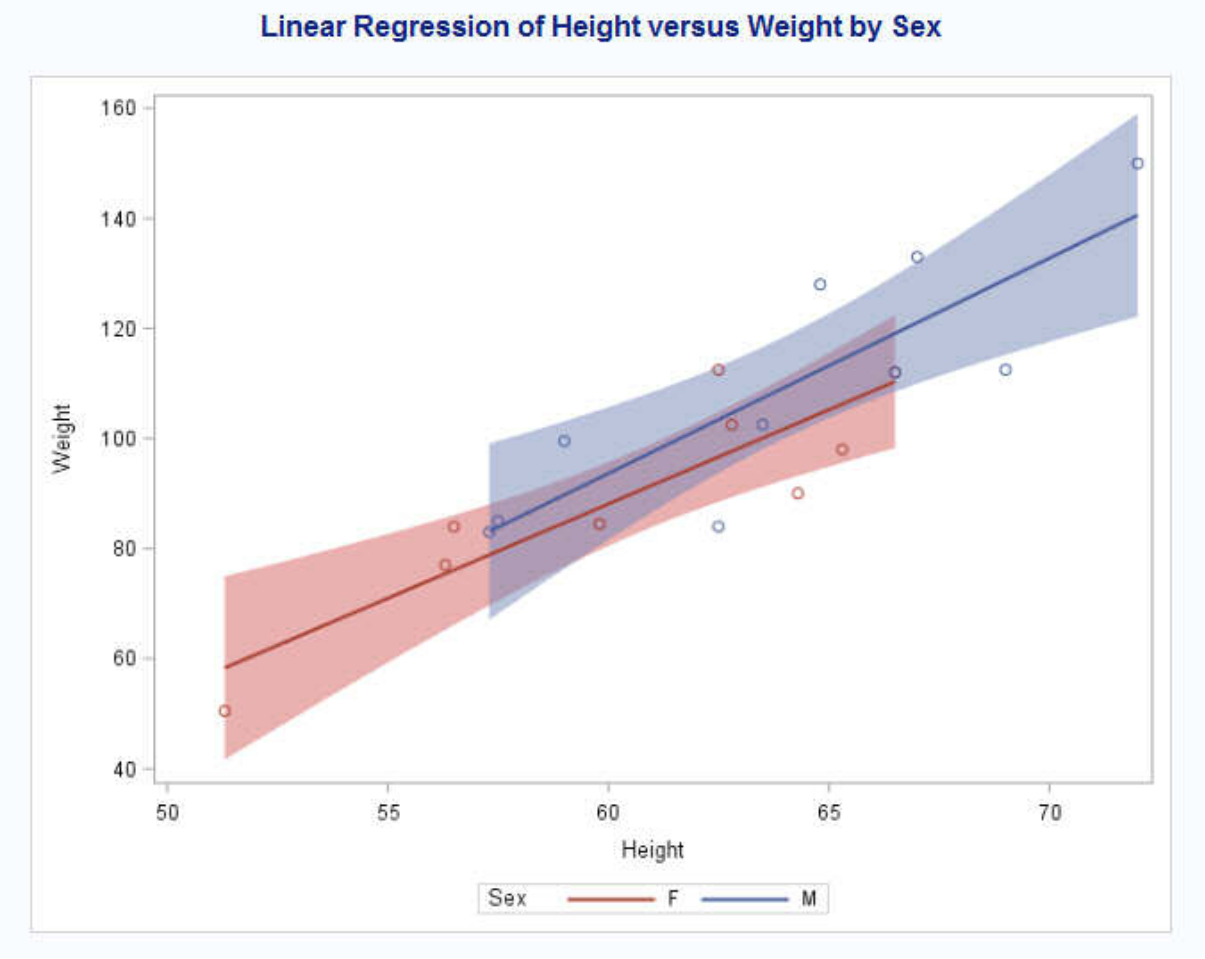
I will start to illustrate these principles using a simple linear regression plot of *height* against *weight* by *sex* from **sashelp.class**. The input data set will require the following variables:

- *sex*
- *height*
- *weight*

However, *name* and *age* will not be used in the final graph, or for validation, so they should be dropped.

```
DATA sasuser.graph1;
  SET sashelp.class;
  KEEP sex height weight;
RUN;

PROC SGPLOT DATA = sasuser.graph1;
  TITLE "Linear Regression of Height versus Weight by Sex";
  REG X = height Y = weight
      / GROUP = sex CLM CLMTRANSPARENCY = 0.5;
RUN;
```



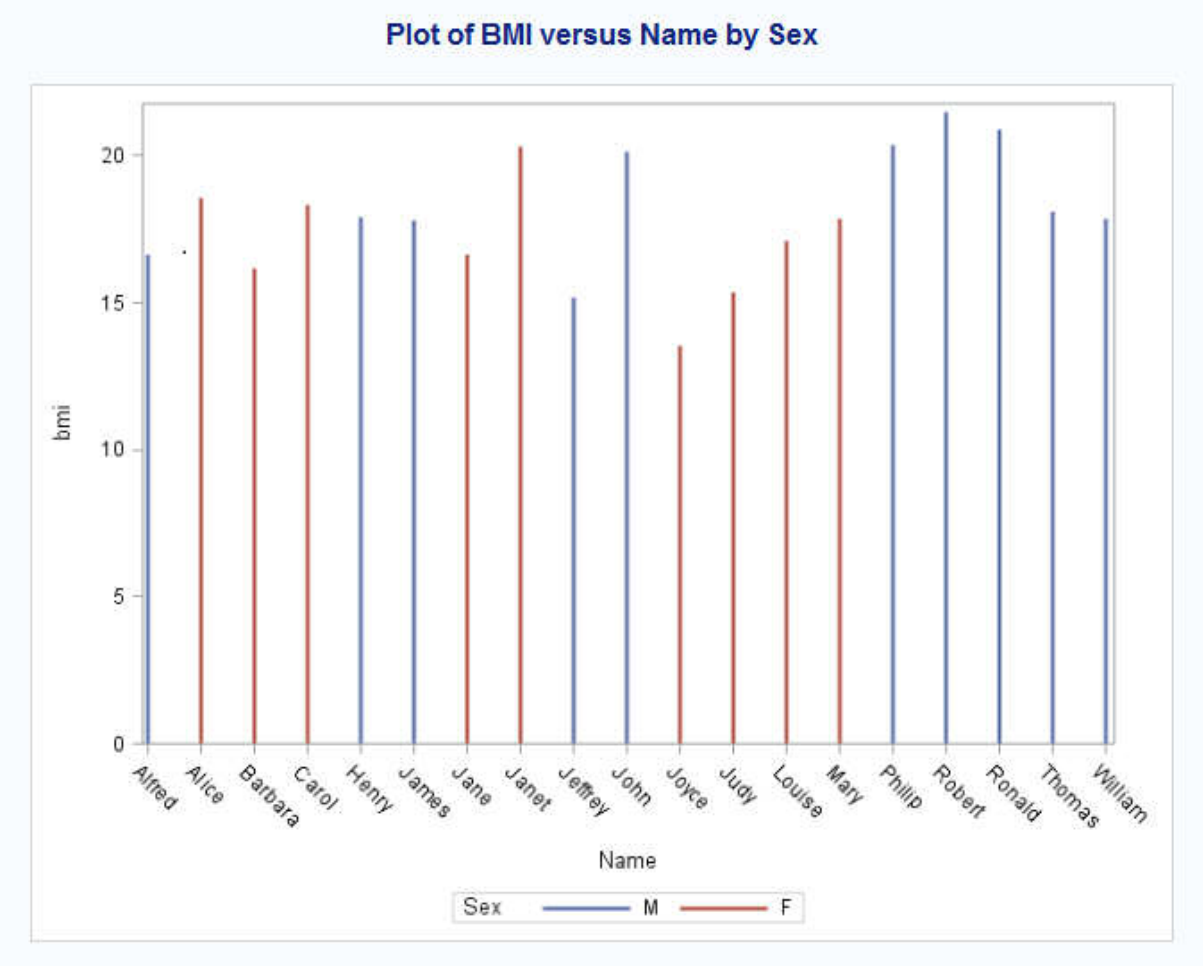
USING COMPUTED VALUES

There is also a frequent need to calculate a new value from existing variables for use in the graph. For example, BMI is calculated from height and weight, so *bmi*, *height* and *weight* variables should be kept in the input data set, so that the BMI calculation can be validated. Therefore, to create a needle plot of BMI for each of the names indicating their sex in **sashelp.class**, the input data set will require the following variables:

- *name*
- *sex*
- *height* (for validation only)
- *weight* (for validation only)
- *bmi* (computed)

```
DATA sasuser.graph2;
  SET sashelp.class;
  bmi = (weight / 2.2) / ((height * 2.54 / 100) ** 2);
  KEEP name sex height weight bmi;
RUN;
```

```
PROC SGPLOT DATA = sasuser.graph2;
  TITLE "Plot of BMI versus Name by Sex";
  NEEDLE X = name Y = bmi / GROUP = sex LINEATTRS = (THICKNESS=2);
RUN;
```



It is sometimes necessary to programmatically offset the positions of points, rather than using plot statement parameters, to fit them into the graph area when using SCATTER, SERIES, and other plot statements which are used with continuous, rather than discrete, axis values. If this has been done, then remember to also include the original variables in the input data set to simplify validation.

TEXT STATEMENTS

When using TEXT statement it is also recommended to include variables used to generate the plotted text.

```
** text box - BOTTOMRIGHT is position of text relative
   to x_text and y_text **;
TEXT X = x_text Y = y_text TEXT = text
     / ATTRID = colors GROUP = textgroup TEXTATTRS = (SIZE = 7pt)
       POSITION = BOTTOMRIGHT VCENTER = BBOX
       X2AXIS Y2AXIS;
```

This example uses four variables to add the text:

- *x_text*
- *y_text*
- *text*
- *textgroup* (for formatting selection)

There are, however, more variables used the generate the *text* values, which are saved in the **heart_data** data set, some of which are used elsewhere in the graph:

```
DATA heart_text (KEEP = subject textgroup text x_text y_text);
  SET heart_data;
  BY subject;
  LENGTH textgroup $30
         text $100
         ;
  x = 0;
  y = 100;
  lineskip = -3;
  IF NMISS(ageatstart) = 0 THEN DO;
    textgroup = 'Start';
    x_text = x;
    y_text = y;
    text = 'Age at start: ' || STRIP(PUT(ageatstart, 8.));
    OUTPUT;
    y + lineskip;
  END;
  IF NMISS(agechddiag) = 0 THEN DO;
    textgroup = 'CHD';
    x_text = x;
    y_text = y;
    text = 'Age at CHD diagnosis: '
           || STRIP(PUT(agechddiag, 8.));
    OUTPUT;
    y + lineskip;
  END;
  IF NMISS(ageatdeath) = 0 THEN DO;
    textgroup = 'Death';
    x_text = x;
    y_text = y;
    text = 'Age at death: ' || STRIP(PUT(ageatdeath, 8.));
    IF CMISS(deathcause) = 0 THEN DO;
      text = STRIP(text)
             || ', Cause: '
             || STRIP(deathcause)
             ;
    END;
    OUTPUT;
    y + lineskip;
  END;
```

```
  IF CMISS(weight, weight_status) = 0 THEN DO;
    textgroup = 'Weight';
    x_text = x;
    y_text = y;
    text = 'Weight: '
           || STRIP(PUT(weight / 2.2, 8.1))
           || ' kg, Status: '
           || STRIP(weight_status)
           ;
    OUTPUT;
    y + lineskip;
  END;
  IF CMISS(cholesterol, chol_status) = 0 THEN DO;
    textgroup = 'Cholesterol';
    x_text = x;
    y_text = y;
    text = 'Cholesterol: '
           || STRIP(PUT(cholesterol / 38.6598, 8.1))
           || ' mmol/L, Status: '
           || STRIP(chol_status)
           ;
    OUTPUT;
    y + lineskip;
  END;
  IF CMISS(diastolic, systolic, bp_status) = 0 THEN DO;
    textgroup = 'BP';
    x_text = x;
    y_text = y;
    text = 'BP: '
           || STRIP(PUT(systolic, BEST.))
           || '/'
           || STRIP(PUT(diastolic, BEST.))
           || ', Status: '
           || STRIP(bp_status)
           ;
    OUTPUT;
    y + lineskip;
  END;
RUN;
```

The following text box example shows how it could be displayed, with optional age at start, CHD diagnosis, death, weight, cholesterol and blood pressure rows:

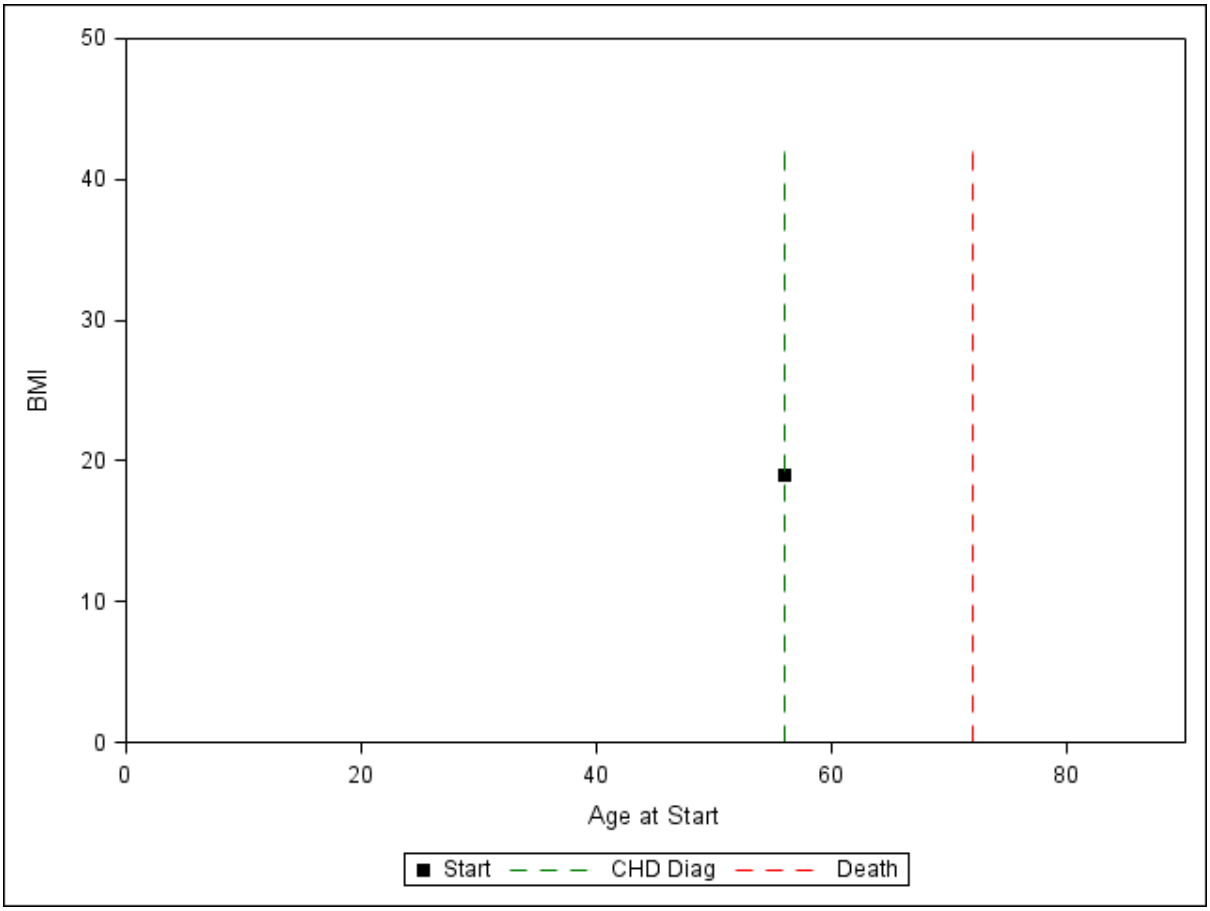
ADDING "SECTION" VARIABLES

Earlier I stated that ODS Graphics procedures expect just a single input data set. This should not be a problem as the plot statements ignore missing by default, so setting multiple data sets with different variables into a single sparse input data set works fine. However, validating a sparse data set which is really lots of data sets can be complicated. To make validation easier it may be necessary to add extra non-graph variables to identify or clarify each section of data, e.g. "section" variable. The input data set can simply be subset into individual sections, effectively reversing the construction process below, which can then be validated separately.

```
** combine heart data, text box and BMI lines **;
DATA sasuser.plot_data;
  SET heart_data (IN = a)
      heart_text (IN = b)
      bmimen_data (IN = c)
      ;
  BY subject;
  SELECT;
    WHEN (a) section = 'Health';
    WHEN (b) section = 'Text';
    WHEN (c) section = 'BMI lines';
    OTHERWISE;
  END;
RUN;
```

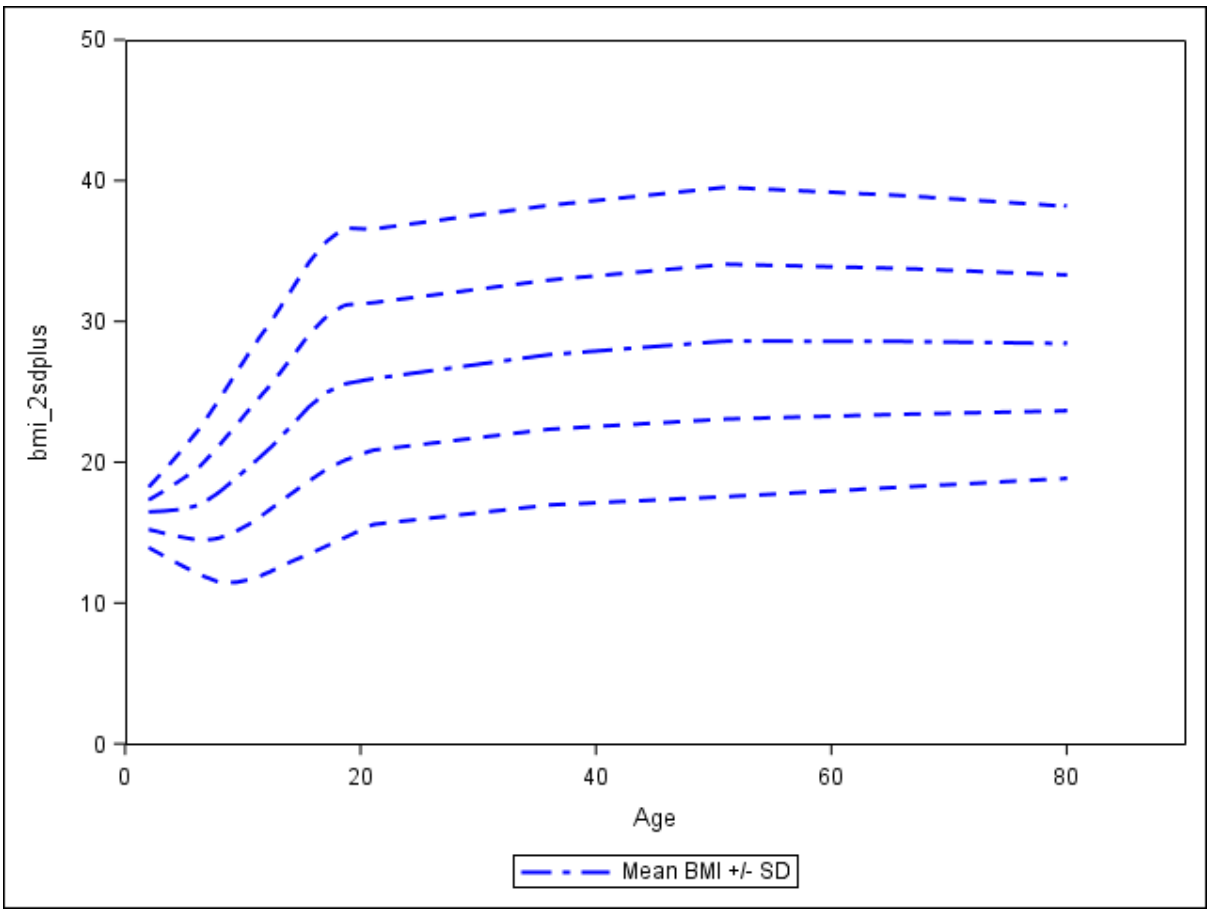
The Health data is used to generate the plot directly relevant to the selected subject:

```
** subject BMI it start **;
SCATTER X = ageatstart Y = bmi
        / MARKERATTRS = (COLOR = BLACK SYMBOL = SQUAREFILLED)
          NAME = 'Start' LEGENDLABEL = 'Start';
** CHD diagnosis and death markers **;
DROPLINE X = agechddiag Y = 42
         / LINEATTRS = (COLOR = GREEN PATTERN = 20) NAME = 'CHD Diag';
DROPLINE X = ageatdeath Y = 42
         / LINEATTRS = (COLOR = RED PATTERN = 20) NAME = 'Death';
```



The Text data is used to generate the text box shown in the previous section, and the BMI data is used to generate the BMI lines:

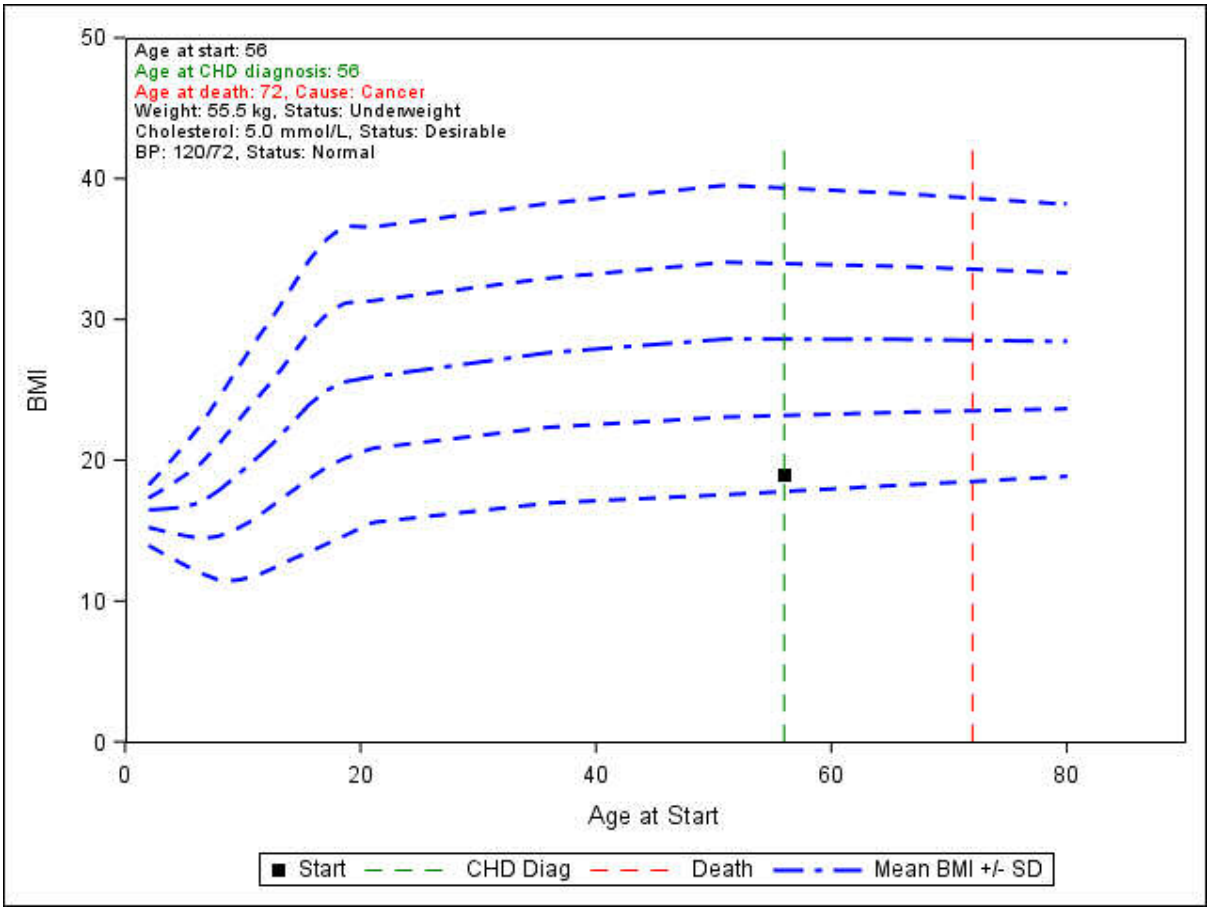
```
** BMI lines **;
LOESS X = age Y = bmi_2sdplus
      / NOMARKERS LINEATTRS = (COLOR = BLUE PATTERN = 20);
LOESS X = age Y = bmi_sdplus
      / NOMARKERS LINEATTRS = (COLOR = BLUE PATTERN = 20);
LOESS X = age Y = bmi_mean
      / NOMARKERS LINEATTRS = (COLOR = BLUE)
        NAME = 'BMI' LEGENDLABEL = 'Mean BMI +/- SD';
LOESS X = age Y = bmi_sdless
      / NOMARKERS  LINEATTRS = (COLOR = BLUE PATTERN = 20);
LOESS X = age Y = bmi_2sdless
      / NOMARKERS LINEATTRS = (COLOR = BLUE PATTERN = 20);
```



These three plots are combined to create the following complex graph, but, as the data can be separated out into discrete data sets, it can be easily validated in its constituent parts using the techniques described in this paper. The full code to generate this graph can be found in the Appendix. This graph uses two SAS-supplied data sets which are available in SAS 9 installations: **sashelp.heart** and **sashelp.bmimen**:

```
** legend **;
KEYLEGEND 'Start' 'CHD Diag' 'Death' 'BMI';
** primary axes **;
XAXIS MIN = 0 MAX = 90 OFFSETMIN = 0 OFFSETMAX = 0;
YAXIS MIN = 0 MAX = 50 OFFSETMIN = 0 OFFSETMAX = 0;
```

```
** secondary axes, used to position the text box **;
X2AXIS MIN = 0 VALUES = (0 TO 1000 BY 1000) DISPLAY = NONE
       OFFSETMIN = 0.01 OFFSETMAX = 0;
Y2AXIS MIN = 0 VALUES = (0 TO 100 BY 100) DISPLAY = NONE
       OFFSETMAX = 0;
```



VALIDATION OF FONTS, LINE PATTERNS AND COLOURS

So far I have deliberately avoided discussing fonts, line patterns and colours, because I firmly believe that they are outside the scope of validation. ODS Graphics even includes options to also create editable graphics files (*.sge), which can be edited using the ODS Graphics Editor to change fonts, line patterns and colours, but not the data displayed in the graph. Therefore I avoid any validation of fonts, line patterns and colours, preferring to focus on those features in the graphs that cannot be altered later.

CONCLUSIONS

Take full advantage of the ODS Graphics requirement for a single input data set:

- Include SAS variables that are required to create the graph.
- Include any variables necessary to calculate computed values in the graph, particularly for those variable used to generate TEXT statement values.
- Do not include any variables that are not needed for the graph production or for validation.

CONTACT DETAILS

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

Name:	Philip R Holland
Organization:	Holland Numerics Limited
Address:	94 Green Drift
City, State ZIP:	Royston, Hertfordshire, SG8 5BT, United Kingdom
Work Phone:	+44-7714-279085
Email:	phil@hollandnumerics.org.uk
Web:	blog.hollandnumerics.org.uk
sasCommunity.org:	www.sascommunity.org/wiki/User:Prholland

APPENDIX - FULL CODE

```
** extract male data from sashelp.heart **;
** add subject numbers using _N_ **;
** calculate BMI **;
** save subject count into &n **;
DATA heart_data (KEEP = subject
                        agechddiag ageatstart ageatdeath deathcause
                        height weight bmi weight_status
                        diastolic systolic bp_status
                        cholesterol chol_status
                        );
  LENGTH subject 8;
  SET sashelp.heart (WHERE = (sex = 'Male')) END = eof;
  subject = _N_;
  bmi = ((weight / 2.2) / ((height * 2.54 / 100) ** 2));
  IF eof THEN CALL SYMPUT('n', STRIP(PUT(_N_, 8.)));
RUN;

** summarise sashelp.bmimen to get mean and SD **;
PROC SUMMARY DATA = sashelp.bmimen NWAY MISSING;
  CLASS age;
  VAR bmi;
  OUTPUT OUT = bmimen_stats MEAN = bmi_mean STDDEV = bmi_sd;
RUN;

** generate text for summary box in top-left of graph **;
DATA heart_text (KEEP = subject textgroup text x_text y_text);
  SET heart_data;
  BY subject;
  LENGTH textgroup $30
         text $100
         ;
  x = 0;
  y = 100;
  lineskip = -3;
  ** Age, if non-missing **;
  IF NMISS(ageatstart) = 0 THEN DO;
    textgroup = 'Start';
    x_text = x;
    y_text = y;
    text = 'Age at start: ' || STRIP(PUT(ageatstart, 8.));
    OUTPUT;
    y + lineskip;
  END;
  ** Age at CHD diagnosis, if non-missing **;
  IF NMISS(agechddiag) = 0 THEN DO;
    textgroup = 'CHD';
    x_text = x;
    y_text = y;
    text = 'Age at CHD diagnosis: '
           || STRIP(PUT(agechddiag, 8.));
    OUTPUT;
    y + lineskip;
  END;
```

```
** Age at death and cause, if non-missing **;
IF NMISS(ageatdeath) = 0 THEN DO;
  textgroup = 'Death';
  x_text = x;
  y_text = y;
  text = 'Age at death: ' || STRIP(PUT(ageatdeath, 8.));
  IF CMISS(deathcause) = 0 THEN DO;
    text = STRIP(text)
           || ', Cause: '
           || STRIP(deathcause)
           ;
  END;
  OUTPUT;
  y + lineskip;
END;
** Weight in kg and weight status, if both non-missing **;
IF CMISS(weight, weight_status) = 0 THEN DO;
  textgroup = 'Weight';
  x_text = x;
  y_text = y;
  text = 'Weight: '
         || STRIP(PUT(weight / 2.2, 8.1))
         || ' kg, Status: '
         || STRIP(weight_status)
         ;
  OUTPUT;
  y + lineskip;
END;
** Cholesterol in mmol/L and cholesterol status,
    if both non-missing **;
IF CMISS(cholesterol, chol_status) = 0 THEN DO;
  textgroup = 'Cholesterol';
  x_text = x;
  y_text = y;
  text = 'Cholesterol: '
         || STRIP(PUT(cholesterol / 38.6598, 8.1))
         || ' mmol/L, Status: '
         || STRIP(chol_status)
         ;
  OUTPUT;
  y + lineskip;
END;
```

```
  ** Blood pressure and blood pressure status,
      if all non-missing **;
  IF CMISS(diastolic, systolic, bp_status) = 0 THEN DO;
    textgroup = 'BP';
    x_text = x;
    y_text = y;
    text = 'BP: '
           || STRIP(PUT(systolic, BEST.))
           || '/'
           || STRIP(PUT(diastolic, BEST.))
           || ', Status: '
           || STRIP(bp_status)
           ;
    OUTPUT;
    y + lineskip;
  END;
RUN;

** calculate BMI, BMI +- SD and BMI +- 2*SD lines **;
DATA bmimen_data (KEEP = subject
                         age bmi_mean
                         bmi_2sdplus bmi_sdplus bmi_sdless bmi_2sdless
                         );
  SET bmimen_stats;
  bmi_2sdplus = bmi_mean + (bmi_sd * 2);
  bmi_sdplus = bmi_mean + bmi_sd;
  bmi_sdless = bmi_mean - bmi_sd;
  bmi_2sdless = bmi_mean - (bmi_sd * 2);
  ** include copy of lines for each subject **;
  DO subject = 1 TO &n.;
    OUTPUT;
  END;
RUN;

PROC SORT DATA = bmimen_data;
  BY subject age;
RUN;

** combine heart data, text box and BMI lines **;
DATA sasuser.plot_data;
  SET heart_data (IN = a)
      heart_text (IN = b)
      bmimen_data (IN = c)
      ;
  BY subject;
  SELECT;
    WHEN (a) section = 'Health';
    WHEN (b) section = 'Text';
    WHEN (c) section = 'BMI lines';
    OTHERWISE;
  END;
  LABEL subject = 'Subject'
        age = 'Age'
        bmi = 'BMI'
        ;
RUN;
```

```
** create text colour map **;
DATA plot_attrmap (KEEP = id value textcolor);
  LENGTH value $30
         id textcolor $50
         ;
  id = 'colors';
  textcolor = 'green';
  value = 'CHD';
  OUTPUT;
  textcolor = 'red';
  value = 'Death';
  OUTPUT;
  textcolor = 'black';
  value = 'Start';
  OUTPUT;
  value = 'Weight';
  OUTPUT;
  value = 'Cholesterol';
  OUTPUT;
  value = 'BP';
  OUTPUT;
RUN;

** Generate plot per subject of BMI lines,
  subject BMI at start,
  droplines to mark CHD diagnosis and death,
  and top-left text box **;
OPTIONS NOBYLINE;
TITLE "Subject: #BYVAL(subject)";

PROC SGPLOT DATA = sasuser.plot_data DATTRMAP = plot_attrmap;
  WHERE subject = 11;
  BY subject;
  ** BMI lines **;
  LOESS X = age Y = bmi_2sdplus
        / NOMARKERS LINEATTRS = (COLOR = BLUE PATTERN = 20);
  LOESS X = age Y = bmi_sdplus
        / NOMARKERS LINEATTRS = (COLOR = BLUE PATTERN = 20);
  LOESS X = age Y = bmi_mean
        / NOMARKERS LINEATTRS = (COLOR = BLUE)
          NAME = 'BMI' LEGENDLABEL = 'Mean BMI +/- SD';
  LOESS X = age Y = bmi_sdless
        / NOMARKERS LINEATTRS = (COLOR = BLUE PATTERN = 20);
  LOESS X = age Y = bmi_2sdless
        / NOMARKERS LINEATTRS = (COLOR = BLUE PATTERN = 20);
  ** subject BMI it start **;
  SCATTER X = ageatstart Y = bmi
          / MARKERATTRS = (COLOR = BLACK SYMBOL = SQUAREFILLED)
            NAME = 'Start' LEGENDLABEL = 'Start';
  ** CHD diagnosis and death markers **;
  DROPLINE X = agechddiag Y = 42
          / LINEATTRS = (COLOR = GREEN PATTERN = 20) NAME = 'CHD Diag';
  DROPLINE X = ageatdeath Y = 42
          / LINEATTRS = (COLOR = RED PATTERN = 20) NAME = 'Death';
```

```
  ** text box - BOTTOMRIGHT is position of text relative
      to x_text and y_text **;
  TEXT X = x_text Y = y_text TEXT = text
      / ATTRID = colors GROUP = textgroup TEXTATTRS = (SIZE = 7pt)
        POSITION = BOTTOMRIGHT VCENTER = BBOX
        X2AXIS Y2AXIS;
  ** legend **;
  KEYLEGEND 'Start' 'CHD Diag' 'Death' 'BMI';
  ** primary axes **;
  XAXIS MIN = 0 MAX = 90 OFFSETMIN = 0 OFFSETMAX = 0;
  YAXIS MIN = 0 MAX = 50 OFFSETMIN = 0 OFFSETMAX = 0;
  ** secondary axes, used to position the text box **;
  X2AXIS MIN = 0 VALUES = (0 TO 1000 BY 1000) DISPLAY = NONE
        OFFSETMIN = 0.01 OFFSETMAX = 0;
  Y2AXIS MIN = 0 VALUES = (0 TO 100 BY 100) DISPLAY = NONE
        OFFSETMAX = 0;
RUN;
```