CareerBuilder™ Series

A Quick & Easy SAS, Statistics and Data Analysis Course
Complete Jobs Seekers and Beginners

Agenda

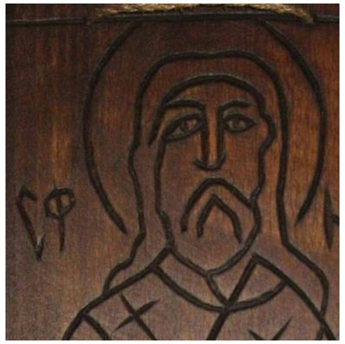
- Traditional SAS/GRAPH System
  - Introducing Traditional SAS/GRAPH System
  - Creating Scatter and Series Plot Using PROC GPLOT
  - Producing Bar and Pie Chart Using PROC GCHART
- New SAS/GRAPH System after SAS 9.2
  - Overview of New SAS/GRAPH System
  - ODS Graphics
  - Procedures for Statistical Graphics
  - Mastering Graph Template Language (GTL)

Agenda

- ❑ Traditional SAS/GRAPH System
  - ➢ Introducing Traditional SAS/GRAPH System
  - ➢ Creating Scatter and Series Plot Using PROC GPLOT
  - ➢ Producing Bar and Pie Chart Using PROC GCHART
- ❑ New SAS/GRAPH System after SAS 9.2
  - ➢ Overview of New SAS/GRAPH System
  - ➢ ODS Graphics
  - ➢ Procedures for Statistical Graphics
  - ➢ Mastering Graph Template Language (GTL)

SAS/GRAPH: Traditional Approach

Before Version 9.2



Traditional SAS Graphics Procedures

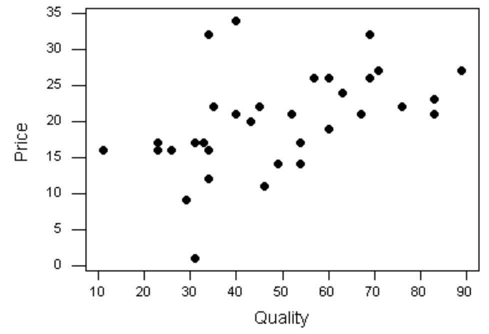
Traditional SAS/GRAPH approach:
ODS output combined with

- PROC GPLOT
- PROC GCHART
- PROC GCONTOUR
- PROC GMAP
- PROC GKPI
- …

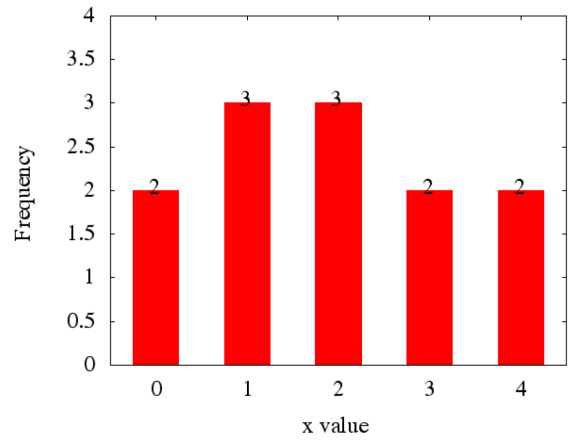
To use traditional way, you need to globally set up the following items (1) GOPTIONS (2) AXES (label, value..) (3) SYMBOLS (color, interpolation, shape, height..) (4) LEGENDS (5) ANNOTATE (6) TITLE (7) GSEG CATALOG (8) HARDWARE…

Graphs SAS Can Produce

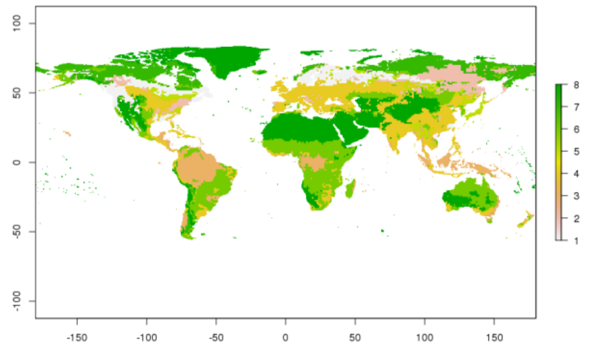
Scatter or trend series



Bar Chart



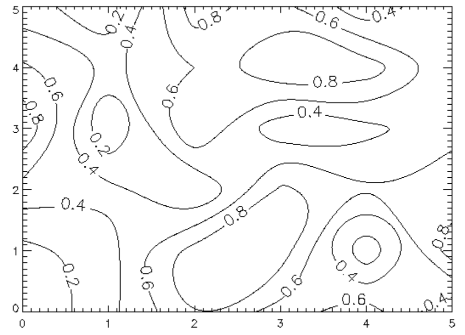
Map



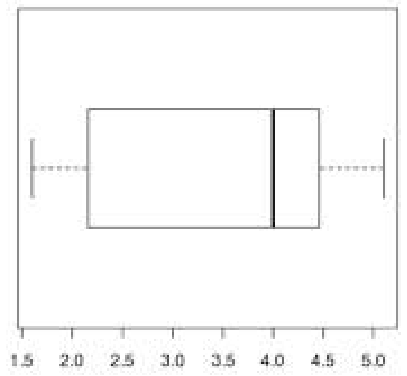
Stem Leaf

Stem	Leaf
0	4
1	0, 7, 8
2	3, 3, 4, 7, 8
3	2, 2, 2, 3, 5, 7, 7
4	0, 0, 1, 1, 3
5	6, 7

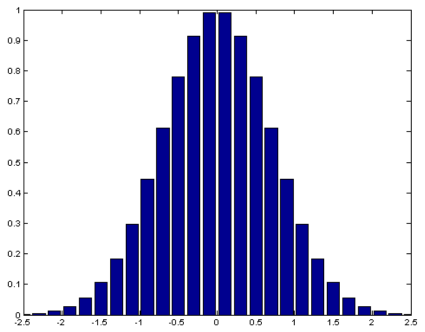
Contour



Box Plot



Histogram



Pie Chart

Setting Look of SAS/Graph (Traditional Way)

1 GOPTIONS

- Out of SAS procedures and data step.
- Setting SAS/graphics environment.
- Globally control color, font, text width..

2 OPTIONS in SAS/Graph Procedure

Function is similar to GOPTION but only effective within SAS/Graph procedure

3 Global Statement

Condoling AXIS, SYMBOL, LEGENDER, PATTERN, TITLE, FOOTNOTE,…

There are a lot of sub options in each method above!

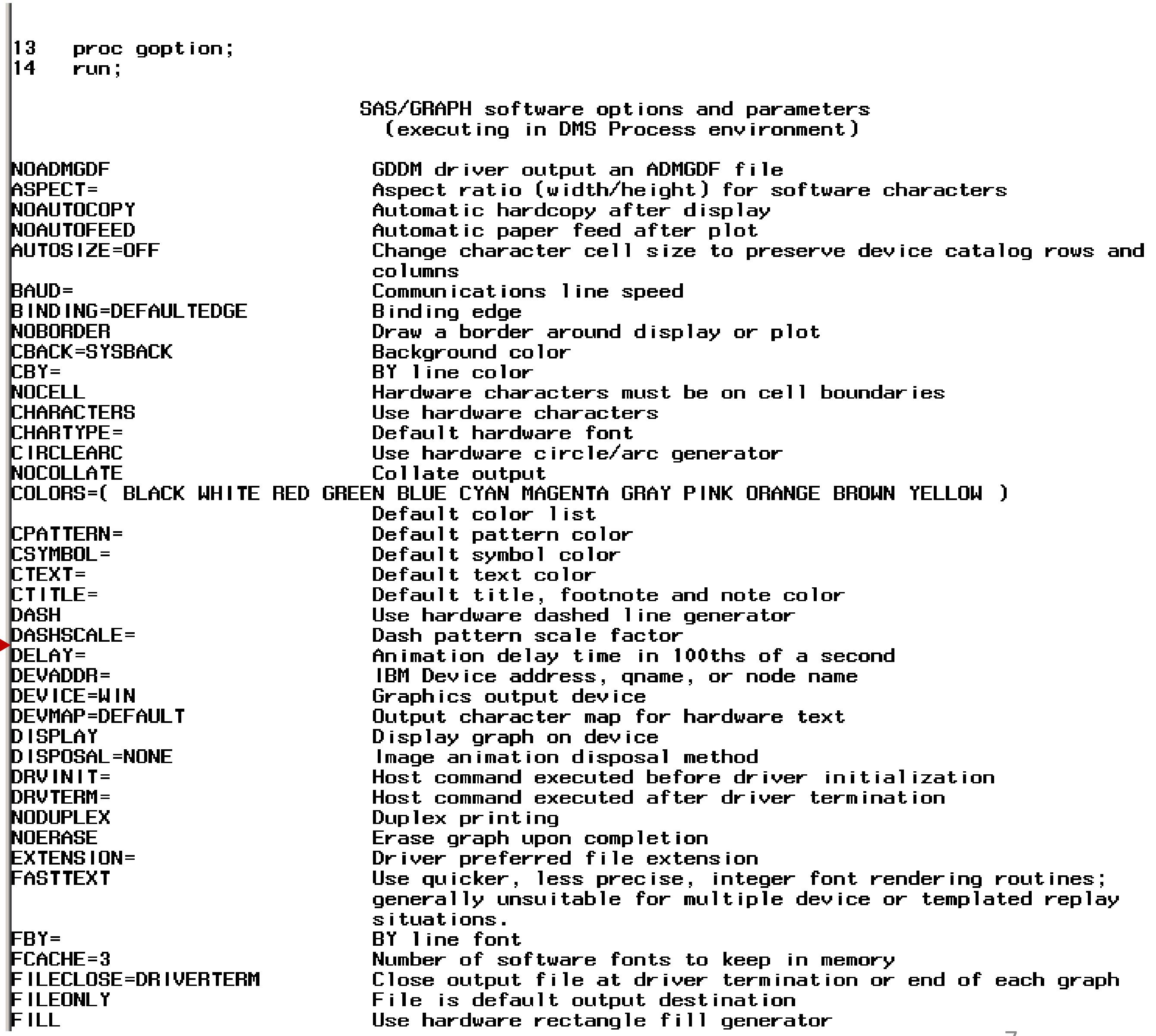
Using GOPTION

SYNTAX

GOPTIONS option-list;

There are too many options, but you can know what are the current options using:

PROC GOPTIONS;
RUN;



```
13   proc goption;
14   run;

                              SAS/GRAPH software options and parameters
                                (executing in DMS Process environment)

NOADMGDF                       GDDM driver output an ADMGDF file
ASPECT=                        Aspect ratio (width/height) for software characters
NOAUTOCOPY                     Automatic hardcopy after display
NOAUTOFEED                     Automatic paper feed after plot
AUTOSIZE=OFF                   Change character cell size to preserve device catalog rows and
                               columns
BAUD=                          Communications line speed
BINDING=DEFAULTEDGE            Binding edge
NOBORDER                       Draw a border around display or plot
CBACK=SYSBACK                  Background color
CBY=                           BY line color
NOCELL                         Hardware characters must be on cell boundaries
CHARACTERS                     Use hardware characters
CHARTYPE=                      Default hardware font
CIRCLEARC                      Use hardware circle/arc generator
NOCOLLATE                      Collate output
COLORS=( BLACK WHITE RED GREEN BLUE CYAN MAGENTA GRAY PINK ORANGE BROWN YELLOW )
                               Default color list
CPATTERN=                      Default pattern color
CSYMBOL=                       Default symbol color
CTEXT=                         Default text color
CTITLE=                        Default title, footnote and note color
DASH                           Use hardware dashed line generator
DASHSCALE=                     Dash pattern scale factor
DELAY=                         Animation delay time in 100ths of a second
DEVADDR=                        IBM Device address, qname, or node name
DEVICE=WIN                     Graphics output device
DEVMAP=DEFAULT                 Output character map for hardware text
DISPLAY                        Display graph on device
DISPOSAL=NONE                   Image animation disposal method
DRVINIT=                       Host command executed before driver initialization
DRVTERM=                       Host command executed after driver termination
NODUPLEX                       Duplex printing
NOERASE                        Erase graph upon completion
EXTENSION=                     Driver preferred file extension
FASTTEXT                       Use quicker, less precise, integer font rendering routines;
                               generally unsuitable for multiple device or templated replay
                               situations.
FBY=                           BY line font
FCACHE=3                       Number of software fonts to keep in memory
FILECLOSE=DRIVERTERM           Close output file at driver termination or end of each graph
FILEONLY                       File is default output destination
FILL                           Use hardware rectangle fill generator
```

Several Important GOPTIONS

```
GOPTIONS RESET=GOPTIONS;          /** Reset only goption to defaults**/
GOPTIONS RESET=ALL;               /** Reset ALL options to defaults**/

GOPTIONS ROTATE=LANDSCAPE;        /** Set LANDSCAPE I PORTRAIT**/

GOPTIONS DEVICE=“WIN”;            /** Set SAS/Graph device entry**/

GOPTIONS XMAX=6;                  /** Set display area width**/

GOPTIONS YMAX=5;                  /** Set display area height**/

GOPTIONS GUNIT=PCT;               /** Set unit of measurement for height
                                      PCT means % of graphics area
                                      or can be n cells,  n inches**/
```

Several Important GOPTIONS

CTITLE=RED

Set title-color

FTEXT="Arial"

Set text font

CTEXT=BLUE

Set text-color

HTEXT=3

Control text-height

HTITLE=5

Set title-height <units>

FTITLE=CENTB

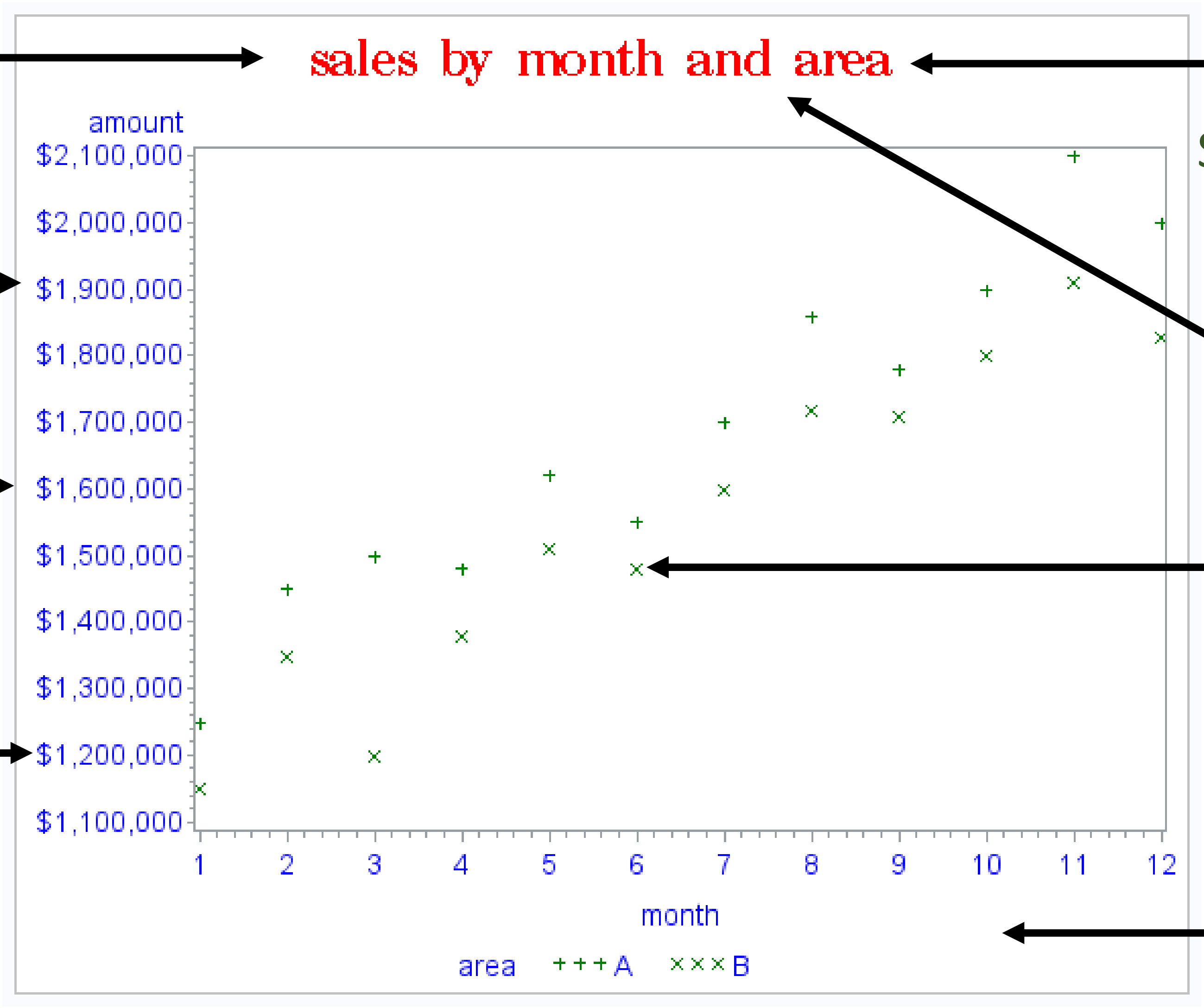
Set title-font

CSYMBOL=G

Set plot symbol-color

Set background-color

CBACK=WHITE



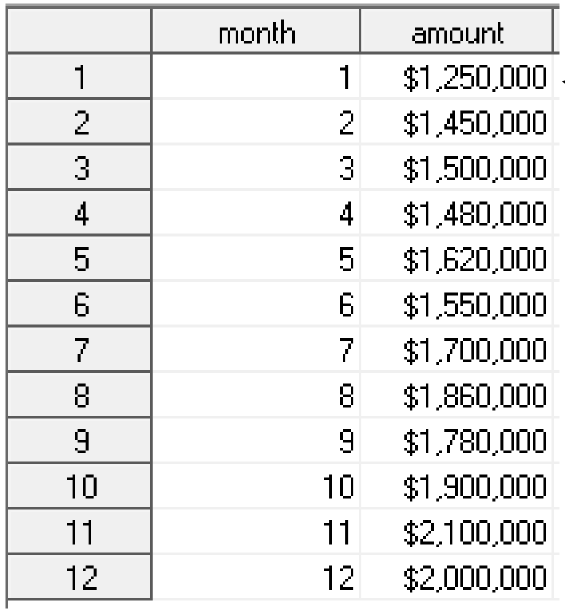
Agenda

- ❑ Traditional SAS/GRAPH System
  - ➢ Introducing Traditional SAS/GRAPH System
  - ➢ Creating Scatter and Series Plot Using PROC GPLOT
  - ➢ Producing Bar and Pie Chart Using PROC GCHART
- ❑ New SAS/GRAPH System after SAS 9.2
  - ➢ Overview of New SAS/GRAPH System
  - ➢ ODS Graphics
  - ➢ Procedures for Statistical Graphics
  - ➢ Mastering Graph Template Language (GTL)

Using PROC GPLOT

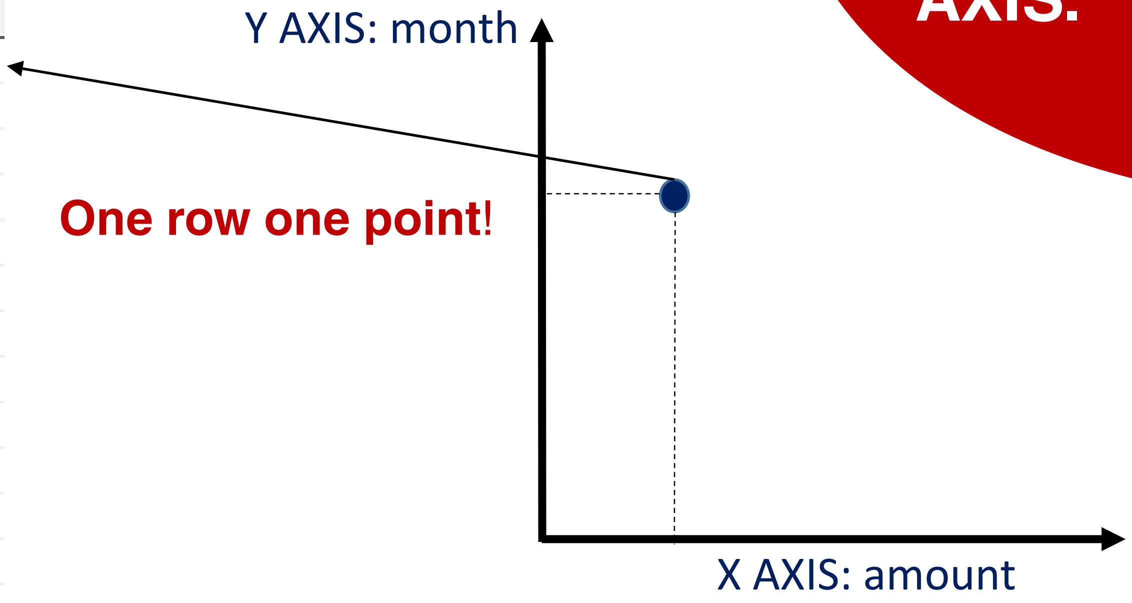
I heard 'PROC GPLOT' is the most popular graph procedure in SAS, tell me what I can do with it?

It produces graph one variable against another. That means one variable is X coordinator and another is Y AXIS.



	month	amount
1	1	$1,250,000
2	2	$1,450,000
3	3	$1,500,000
4	4	$1,480,000
5	5	$1,620,000
6	6	$1,550,000
7	7	$1,700,000
8	8	$1,860,000
9	9	$1,780,000
10	10	$1,900,000
11	11	$2,100,000
12	12	$2,000,000

One row one point!



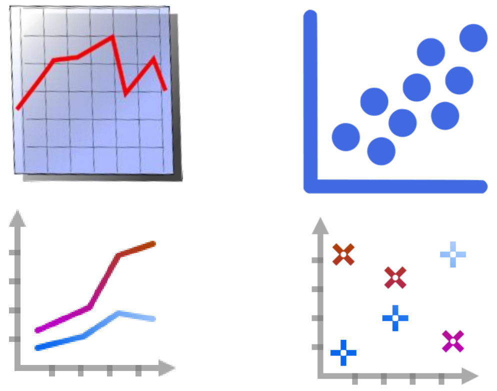
Syntax of PROC GPLOT

```
PROC GPLOT DATA=<data set name> <options>
  PLOT requests / <options>
Quit;
```

There are two forms of PLOT requests:

- PLOT X*Y; /* Single scatter or curve*/

- PLOT X*Y=Z ; /* Multi-scatter or curve*/



Z is a categorical variable (such as gender), and you want to produce multiple (Male and Female) curves (series) stratified by Z.

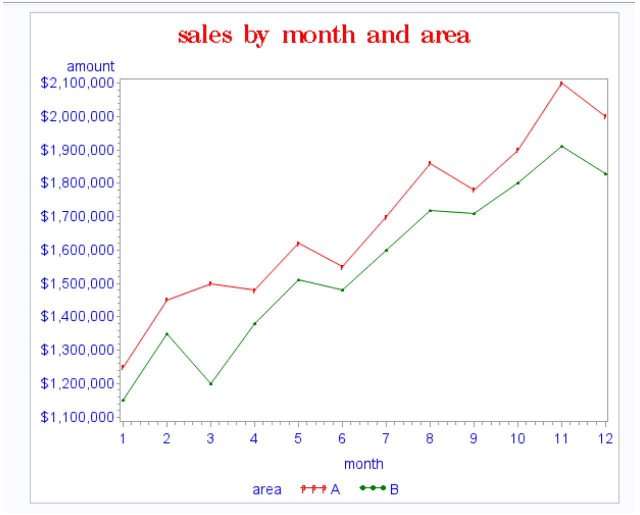
Example of PROC GPLOT

```
/* Using GOPTIONS to set global SAS/graph options */
goptions reset=goptions
         FTEXT="Arial" CTEXT=BLUE HTEXT=3
         FTITLE=CENTB  CTITLE=RED HTITLE=5
         CSYMBOL=G CBACK=WHITE CPATTERN=B
         GUNIT=pct CELL BORDER
         XMAX=6 YMAX=5
         ROTATE=LANDSCAPE;

/* Using global command to set color, symbol shape
   and connection for each curve. The setting will
   overwrite some options in GOPTIONS above*/

symbol1 c=RED v=plut i=j;
symbol2 c=G v=dot i=j;
proc gplot;
  plot amount*month=area;
  title 'sales by month and area';
run;
quit;
```

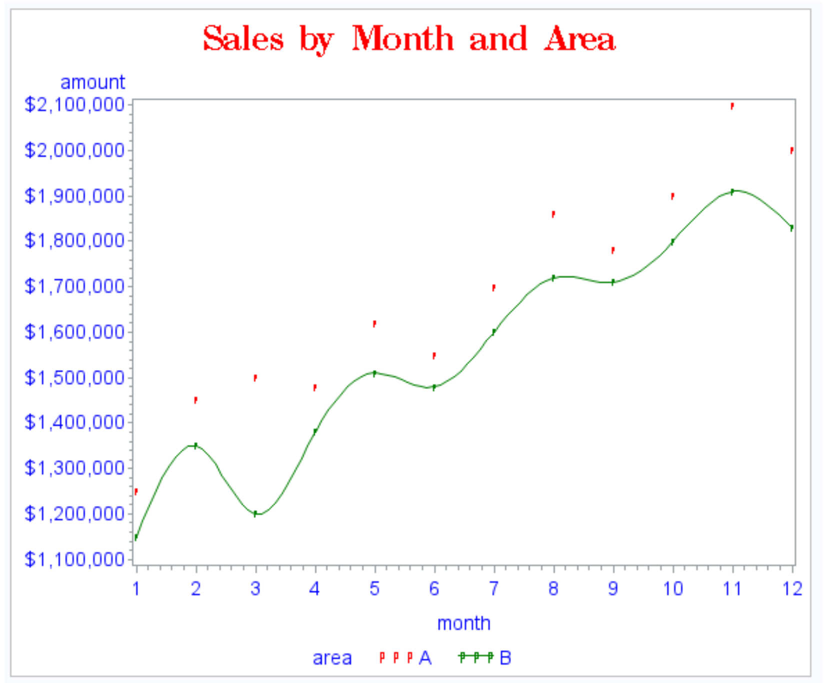
The first curve is RED, and the second one is GREEN

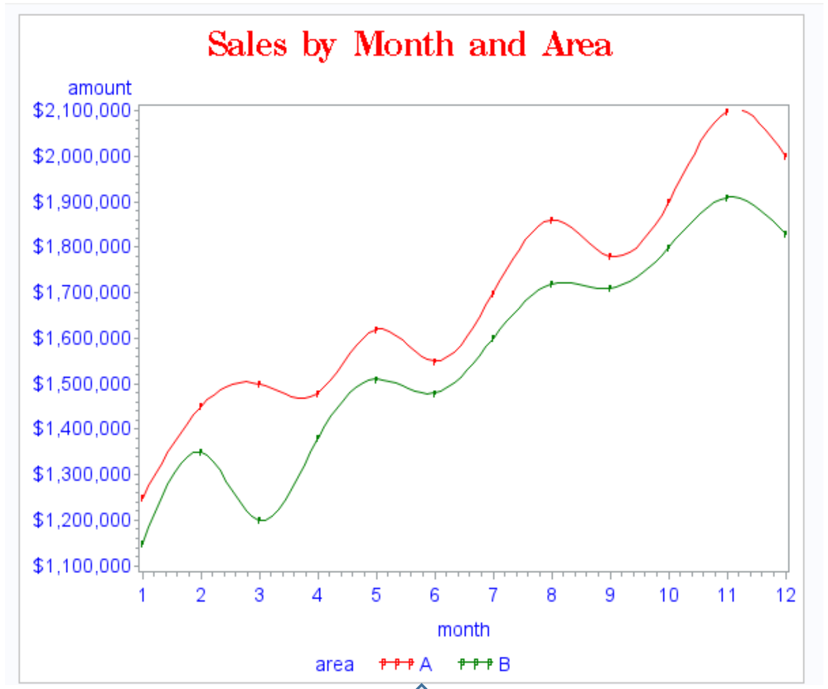
This means you connect points with 'JOIN' way. Where i=j means INTERPOL=JOIN

Here you create two curves stratified by the field area (A and B)



Example of PROC GPLOT





```
symbol1 c=RED v=dot i=NONE;
symbol2 c=G v=dot i=spline;
```

Using i=NONE to produce SCATTER plot

```
symbol1 c=RED v=dot i=spline;
symbol2 c=G v=dot i=spline;
```

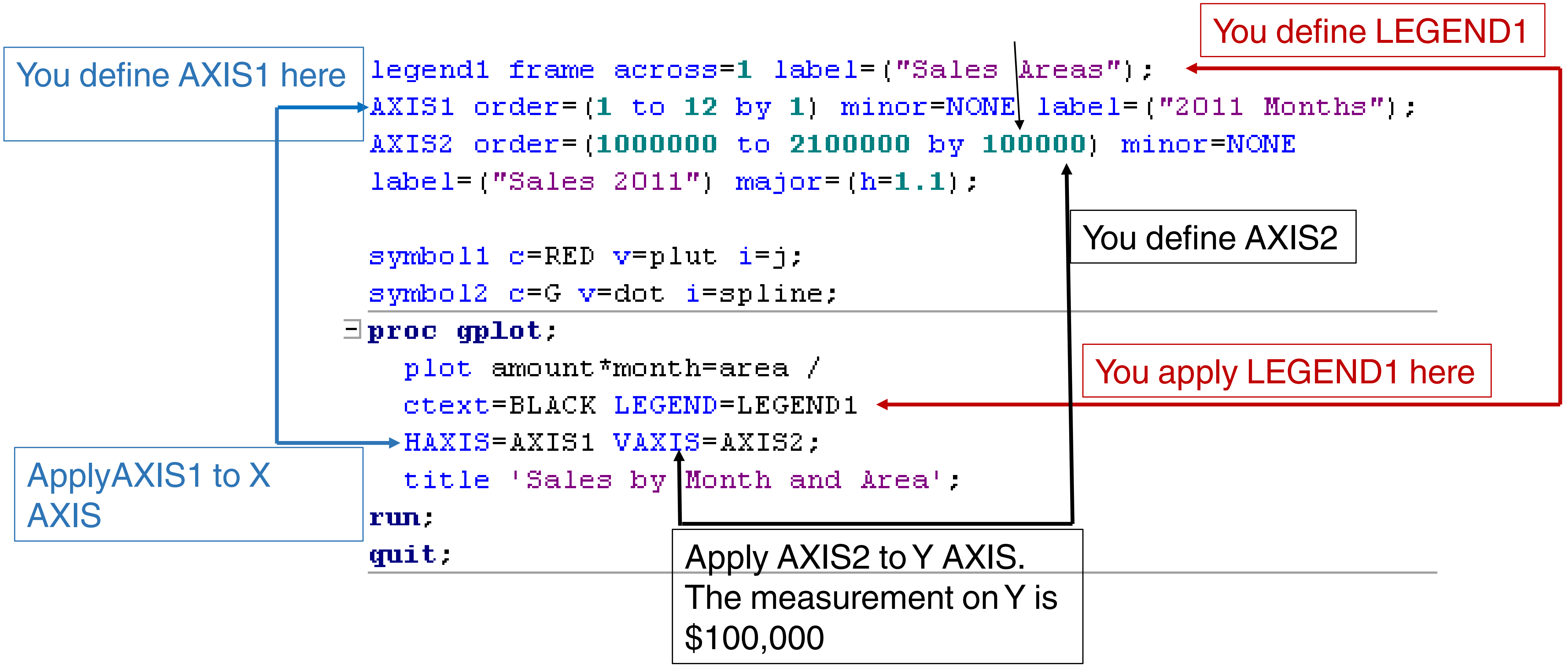
Using i=SPLINE to make curves in plot more smooth

Useful Graphics Options in PROC GPLOT

```
PROC GPLOT DATA=<data set name> <options>
  PLOT requests / <options>
Quit;
```

CTEXT=RED;	Set text color, overwrite GOPTIONS
LEGEND=<LEGENDn>;	Use LEGENDn in graph, you must first define LEGENDn using global statement outside PROC GPLOT
HAXIS=AXISn;	Use AXISn as X in graph, you must first define AXISn outside PROC GPLOT
VAXIS=AXISn;	Use AXISn as Y in graph, you must first define AXISn outside PROC GPLOT

Example of PROC GPLOT



You define AXIS1 here

You define LEGEND1

```
legend1 frame across=1 label=("Sales Areas");
AXIS1 order=(1 to 12 by 1) minor=NONE label=("2011 Months");
AXIS2 order=(1000000 to 2100000 by 100000) minor=NONE
label=("Sales 2011") major=(h=1.1);

symbol1 c=RED v=plut i=j;
symbol2 c=G v=dot i=spline;
proc gplot;
   plot amount*month=area /
   ctext=BLACK LEGEND=LEGEND1
   HAXIS=AXIS1 VAXIS=AXIS2;
   title 'Sales by Month and Area';
run;
quit;
```

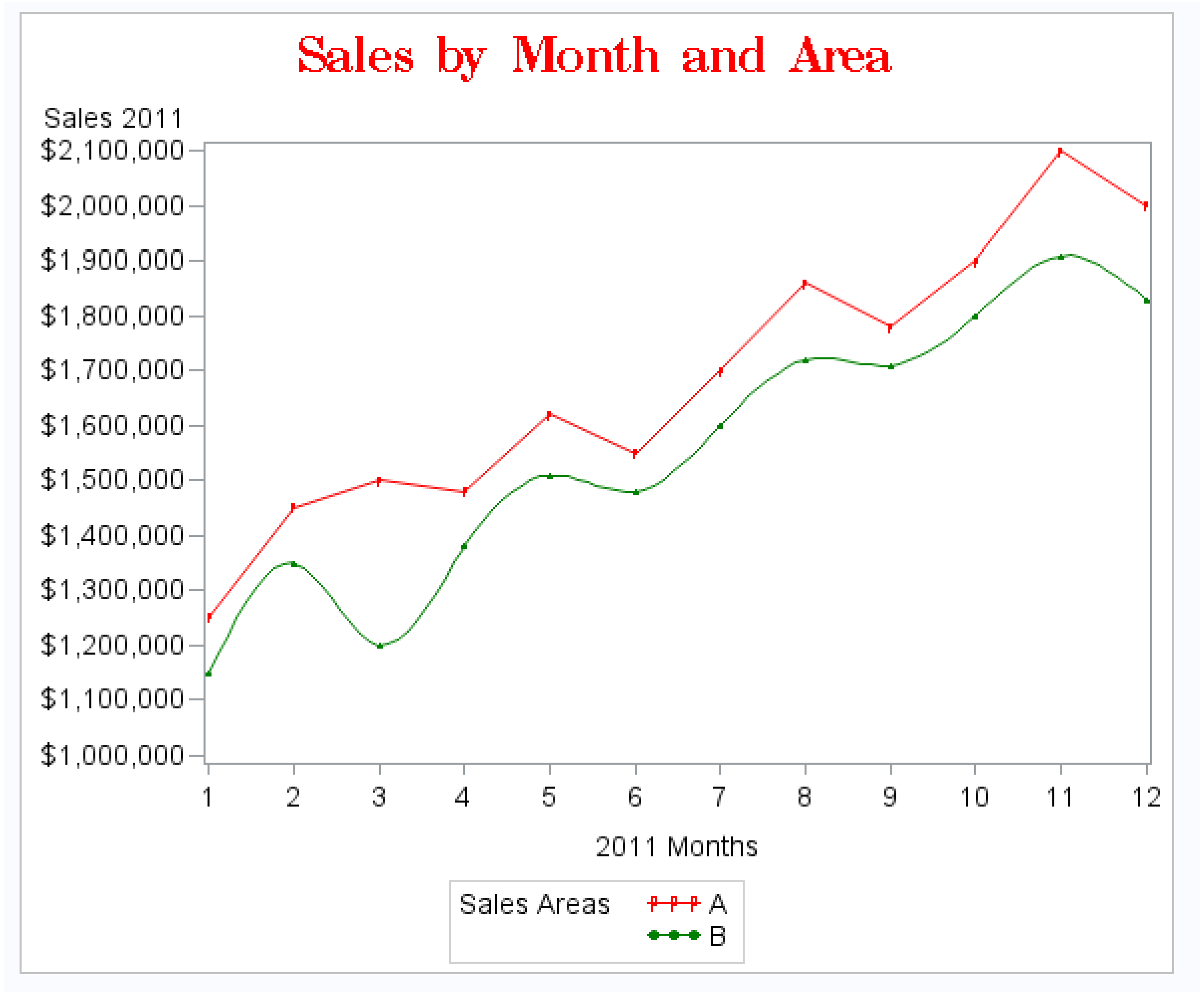
You define AXIS2

You apply LEGEND1 here

ApplyAXIS1 to X AXIS

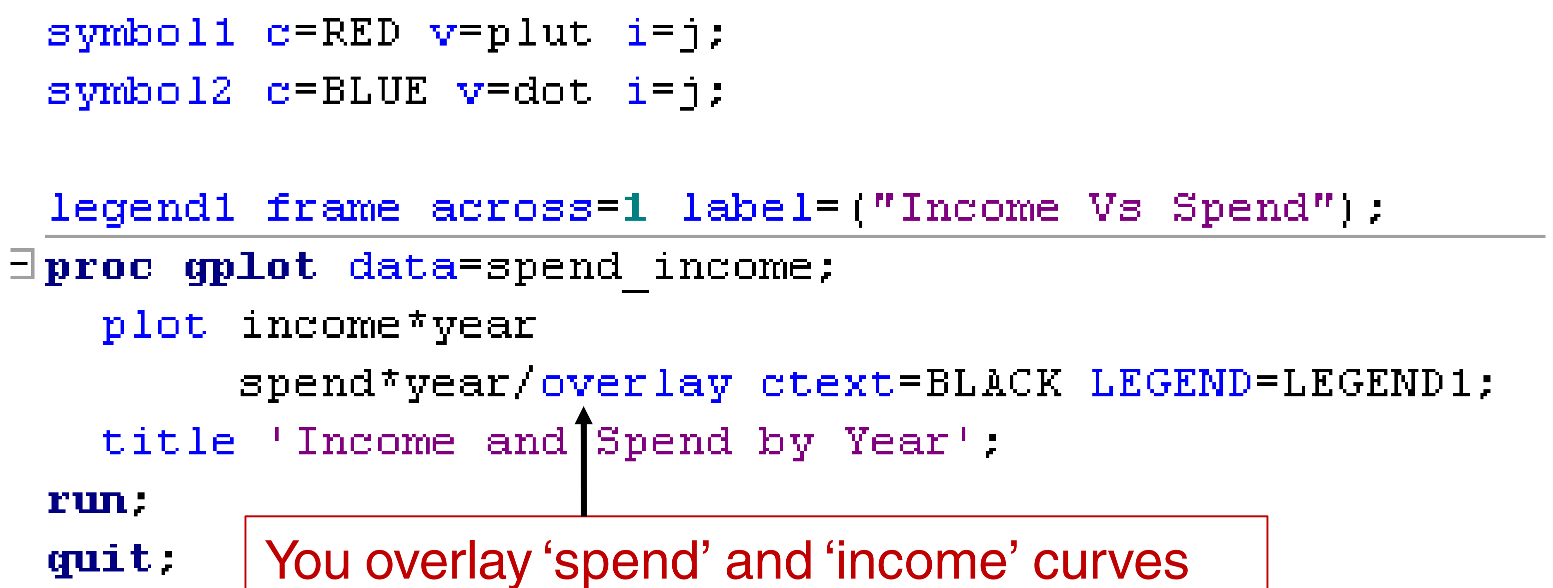
Apply AXIS2 to Y AXIS. The measurement on Y is $100,000

Example of PROC GPLOT



Overlay Curves in PROC GPLOT

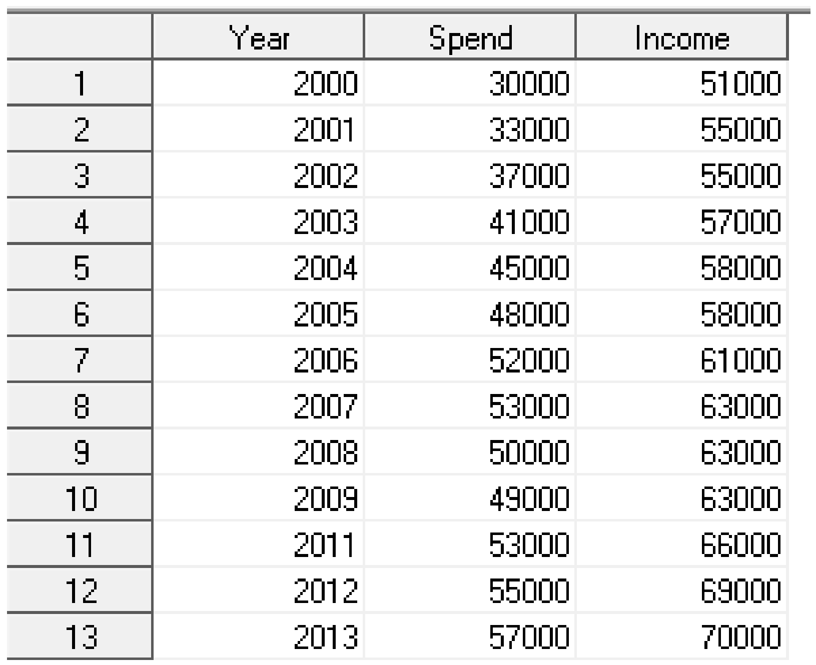
You can overlay multiple curves in PROC GPLOT. This is different from stratified curves, because curves come from two different variables in the data set.



```
symbol1 c=RED v=plut i=j;
symbol2 c=BLUE v=dot i=j;

legend1 frame across=1 label=("Income Vs Spend");
proc gplot data=spend_income;
  plot income*year
       spend*year/overlay ctext=BLACK LEGEND=LEGEND1;
  title 'Income and Spend by Year';
run;
quit;
```

You overlay 'spend' and 'income' curves using 'overlay' option. Note, they are two columns in the table 'spend_income'.

Table: spend_income



	Year	Spend	Income
1	2000	30000	51000
2	2001	33000	55000
3	2002	37000	55000
4	2003	41000	57000
5	2004	45000	58000
6	2005	48000	58000
7	2006	52000	61000
8	2007	53000	63000
9	2008	50000	63000
10	2009	49000	63000
11	2011	53000	66000
12	2012	55000	69000
13	2013	57000	70000

Agenda

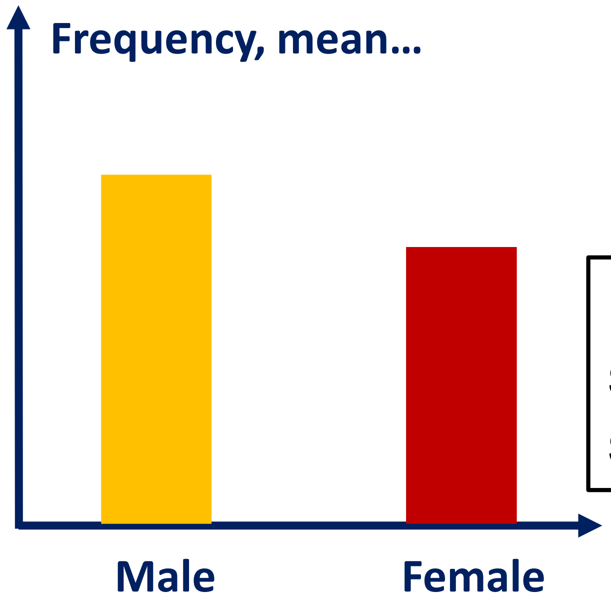
- Traditional SAS/GRAPH System
    - Introducing Traditional SAS/GRAPH System
    - Creating Scatter and Series Plot Using PROC GPLOT
    - Producing Bar and Pie Chart Using PROC GCHART
- New SAS/GRAPH System after SAS 9.2
    - Overview of New SAS/GRAPH System
    - ODS Graphics
    - Procedures for Statistical Graphics
    - Mastering Graph Template Language (GTL)

Using PROC GCHART

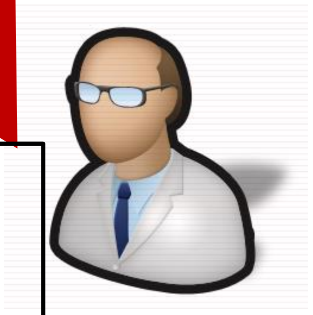
What are the things that PROC GPLOT **cannot** do but PROC GCHART **can** do?

PROC GCHART can produce 'Summarized Data' graph but PROC GPLOT creates 'Original Data' plot unless you summarize the data first.

Note, the X AXIS contains the values of a categorical (or discretized continuous) variable, but Y AXIS stands for the summarized statistics (means, sum, frequency..).

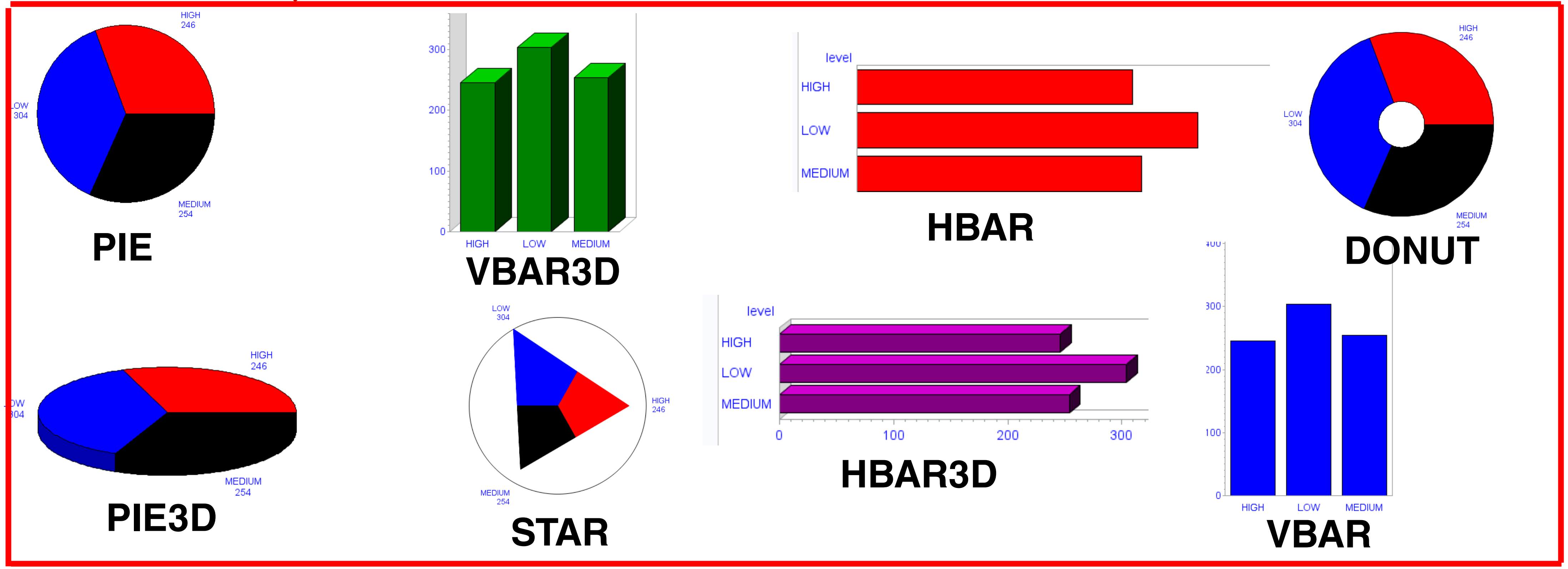


It means PROC GCHART first summarizes data (count, mean, sum..) for you, then plots.



Syntax of PROC GCHART

```
PROC GCHART DATA=<data set name> <options>
    Graph Type  <Chart Variable> /<Options>;
Quit;
```



Pattern Statement for PROC GCHART

The global statement ‘PATTERN’ is highly related to the outcome of ‘PROC GCHART’:

BAR CHART →

```
PATTERN value=SOLID|EMPTY color=RED|BLUE…;
```

PIE or STAR CHART →

```
PATTERN1 value=PSOLID|PEMPTY color=RED;
PATTERN2 value=PSOLID|PEMPTY color=BLUE;
….
PATTERNn value=PSOLID|PEMPTY color=<COLORn>;
```

Options of PROC GCHART

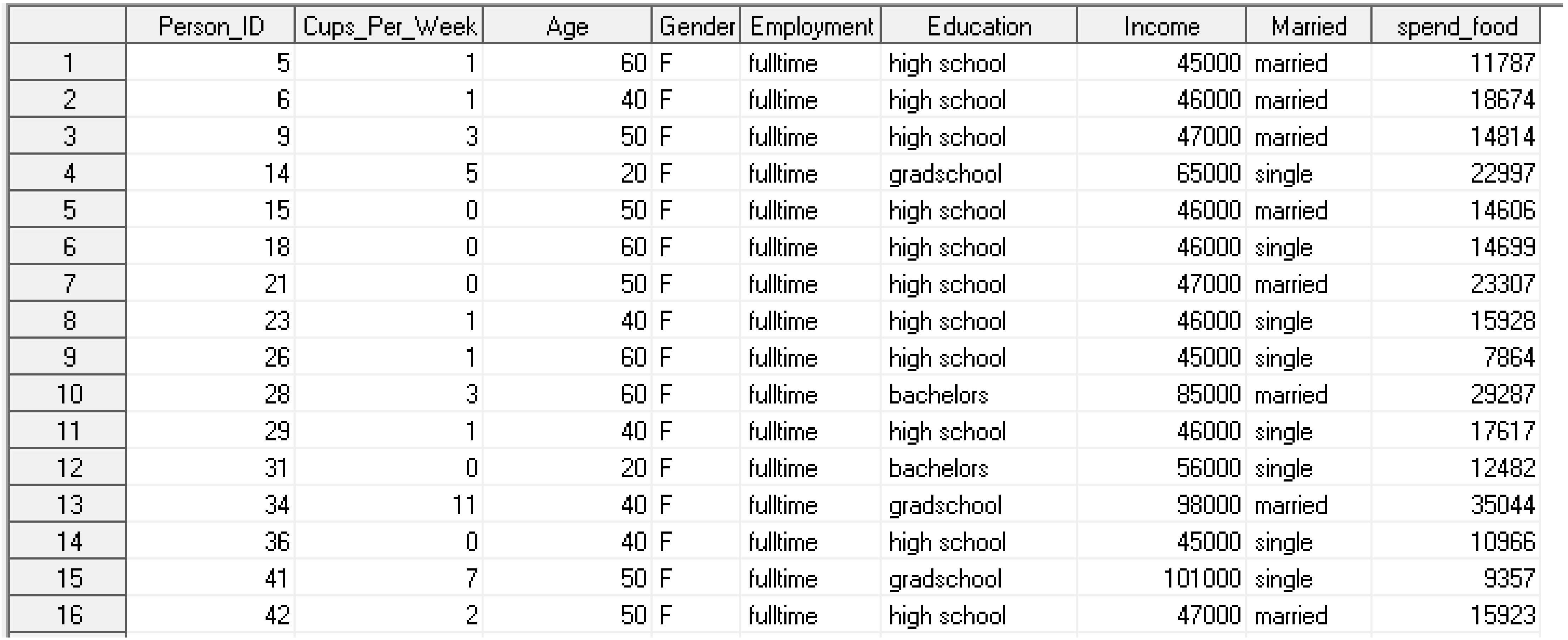
```
PROC GCHART DATA=<data set name> <options>
   Graph Type  <Chart Variable> /<Options>;
Quit;
```

SUMVAR=	NOSTATS	SUBGROUP=
TYPE=	MIDPOINTS=	
DISCRETE	NOHEADING	
CTEXT=	GROUP=	

Examples of PROC GCHART

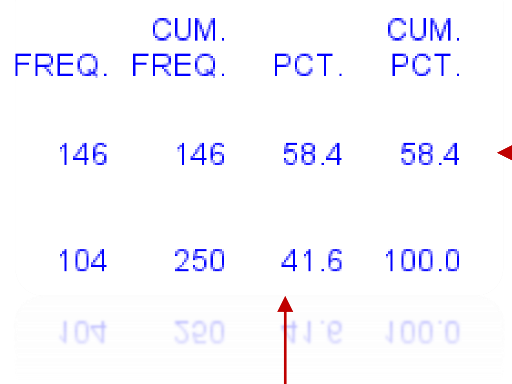
We now use the following data for the following examples

Table: survey



	Person_ID	Cups_Per_Week	Age	Gender	Employment	Education	Income	Married	spend_food
1	5	1	60	F	fulltime	high school	45000	married	11787
2	6	1	40	F	fulltime	high school	46000	married	18674
3	9	3	50	F	fulltime	high school	47000	married	14814
4	14	5	20	F	fulltime	gradschool	65000	single	22997
5	15	0	50	F	fulltime	high school	46000	married	14606
6	18	0	60	F	fulltime	high school	46000	single	14699
7	21	0	50	F	fulltime	high school	47000	married	23307
8	23	1	40	F	fulltime	high school	46000	single	15928
9	26	1	60	F	fulltime	high school	45000	single	7864
10	28	3	60	F	fulltime	bachelors	85000	married	29287
11	29	1	40	F	fulltime	high school	46000	single	17617
12	31	0	20	F	fulltime	bachelors	56000	single	12482
13	34	11	40	F	fulltime	gradschool	98000	married	35044
14	36	0	40	F	fulltime	high school	45000	single	10966
15	41	7	50	F	fulltime	gradschool	101000	single	9357
16	42	2	50	F	fulltime	high school	47000	married	15923

Examples: Simple BAR

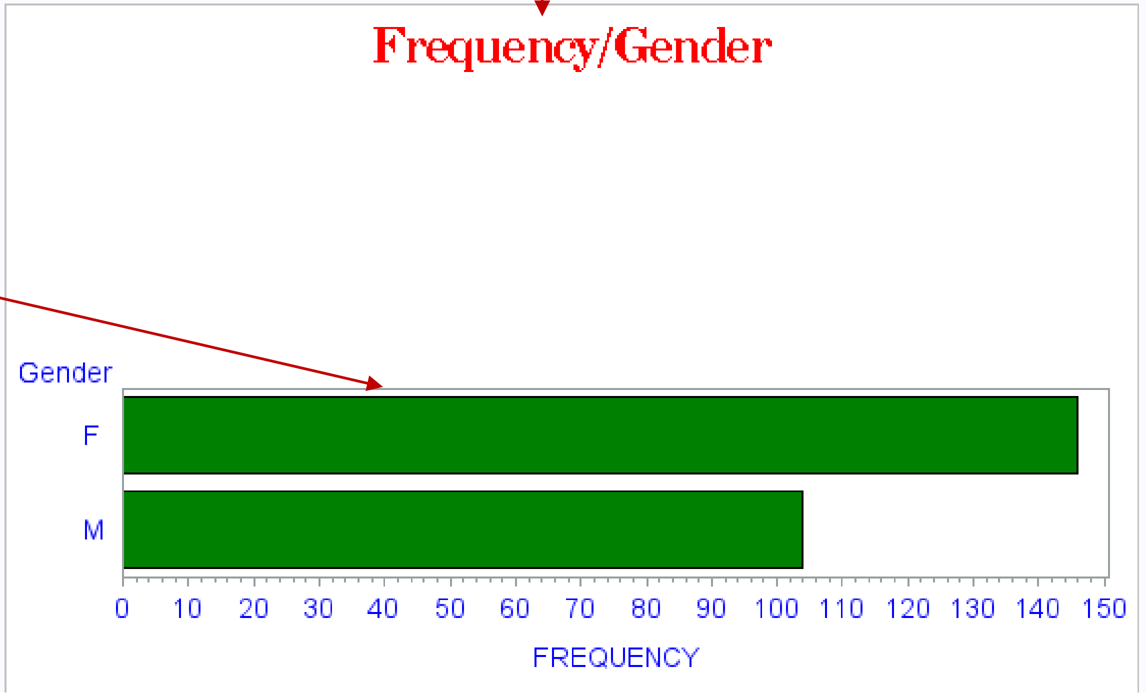


```
         CUM.             CUM.
FREQ.   FREQ.     PCT.     PCT.

  146     146     58.4     58.4


  104     250     41.6    100.0
```

Using ‘NOSTATS’ not to attach statistics beside the bar chart

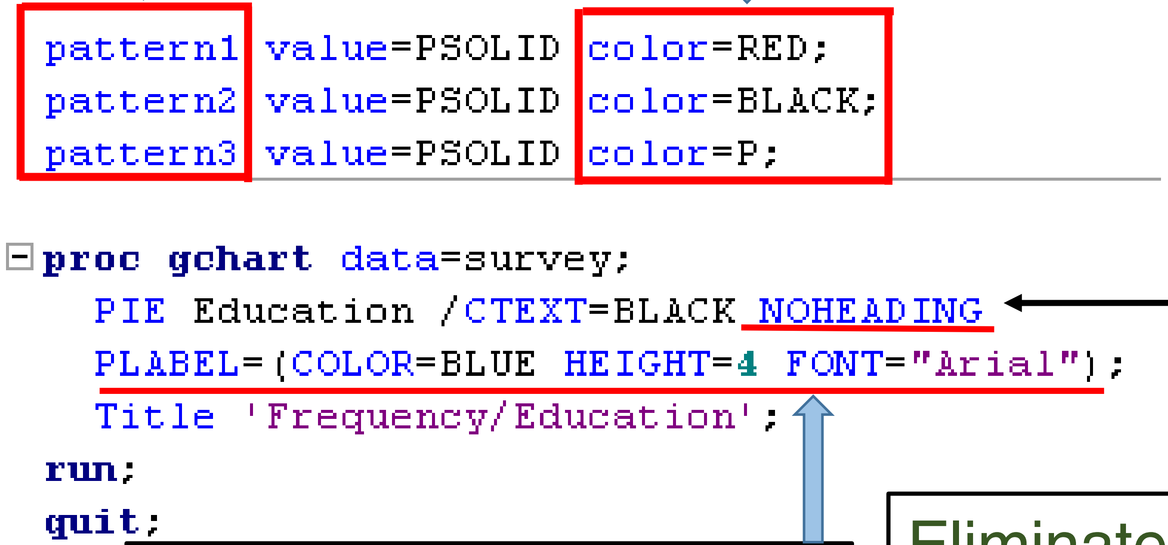
The default summarized statistics is Frequency (or count)

```
pattern value=SOLID color=GREEN;

proc gchart data=survey;
   HBAR gender /NOSTATS ;
   Title 'Frequency/Gender';
run;
quit;
```



Examples 1: Simple PIE Chart

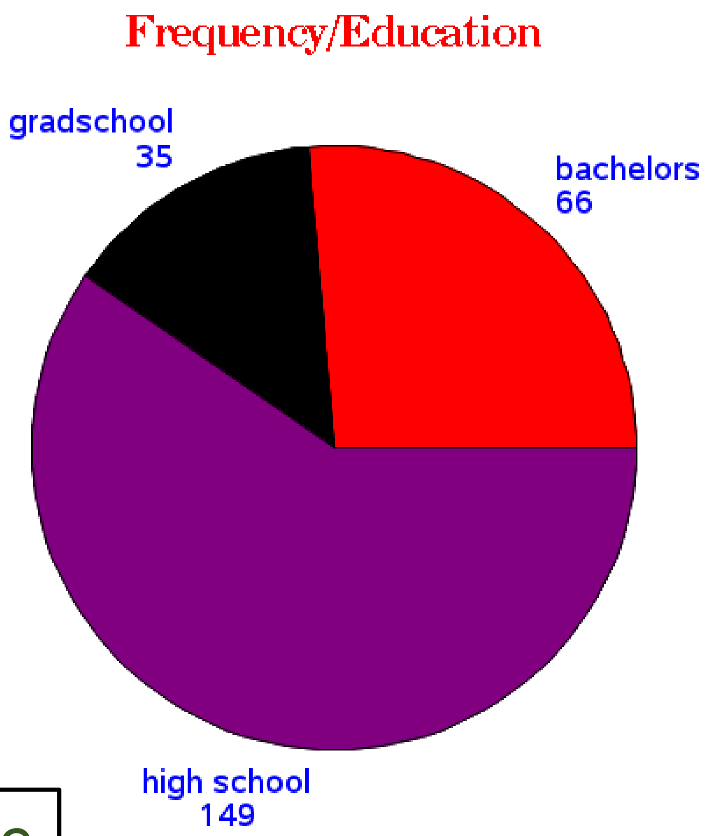
If you draw PIE chart with different look in each portion, use PATTERN1, PATTERN2,.. , and set different values.



```
pattern1 value=PSOLID color=RED;
pattern2 value=PSOLID color=BLACK;
pattern3 value=PSOLID color=P;

proc gchart data=survey;
   PIE Education /CTEXT=BLACK NOHEADING
   PLABEL=(COLOR=BLUE HEIGHT=4 FONT="Arial");
   Title 'Frequency/Education';
run;
quit;
```

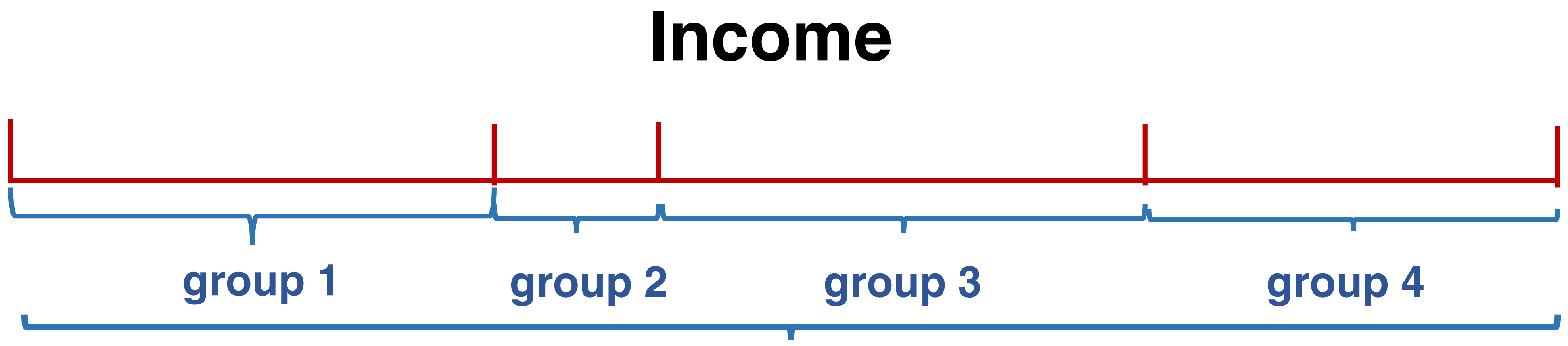
Using 'PLABEL' to set look of label in PIE chart

Eliminate header, because it is replicated with my title



Producing Chart for Continuous Variable

‘PROC GCHART’ can be used to produce BAR chart for continuous variable such as ‘income’. The variable will be automatically discretized into categorical variable by calculating middle point.



You actually use this variable to draw BAR chart.

Income_group

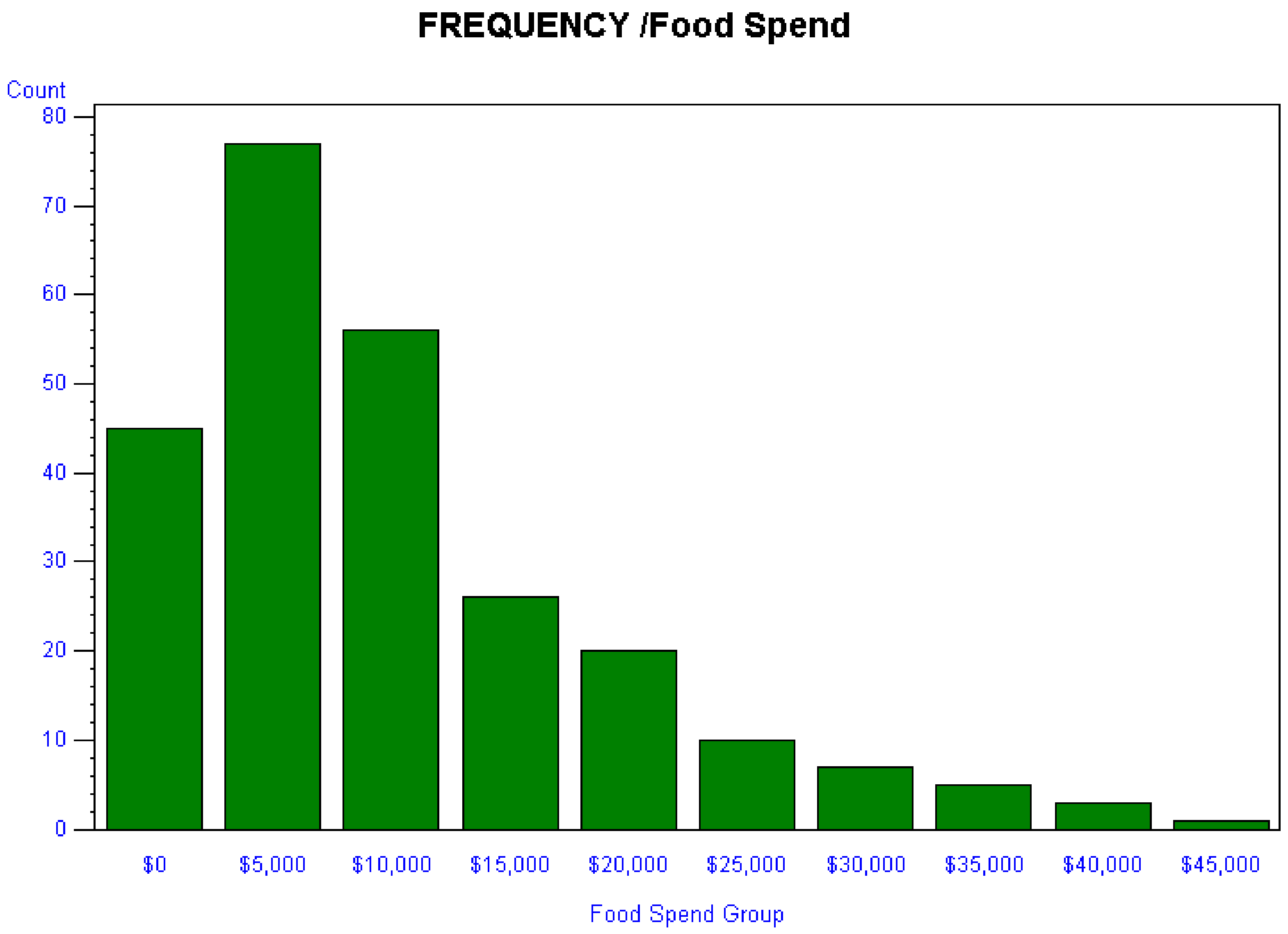
The continuous variable ‘income’ has been discretized as a categorical one!

Examples: Producing Chart for Continuous Variable

Set Midpoint and Response AXIS for BAR chart

```
goptions reset=goptions
          gunit=pct
         ROTATE=LANDSCAPE;

AXIS1 order=(0 to 45000 by 5000)  label=("Food Spend Group" HEIGHT=5);
AXIS2 order=(0 to 80 by 10) label=("Count" HEIGHT=5) major=(h=1.1);

pattern value=SOLID color=GREEN;

proc gchart data=survey;
  VBAR spend_food /CTEXT=BLUE MAXIS=AXIS1 RAXIS=AXIS2;
  format spend_food dollar11.;
  Title 'FREQUENCY /Food Spend';
run;
quit;
```

Examples: Producing Chart for Continuous Variable

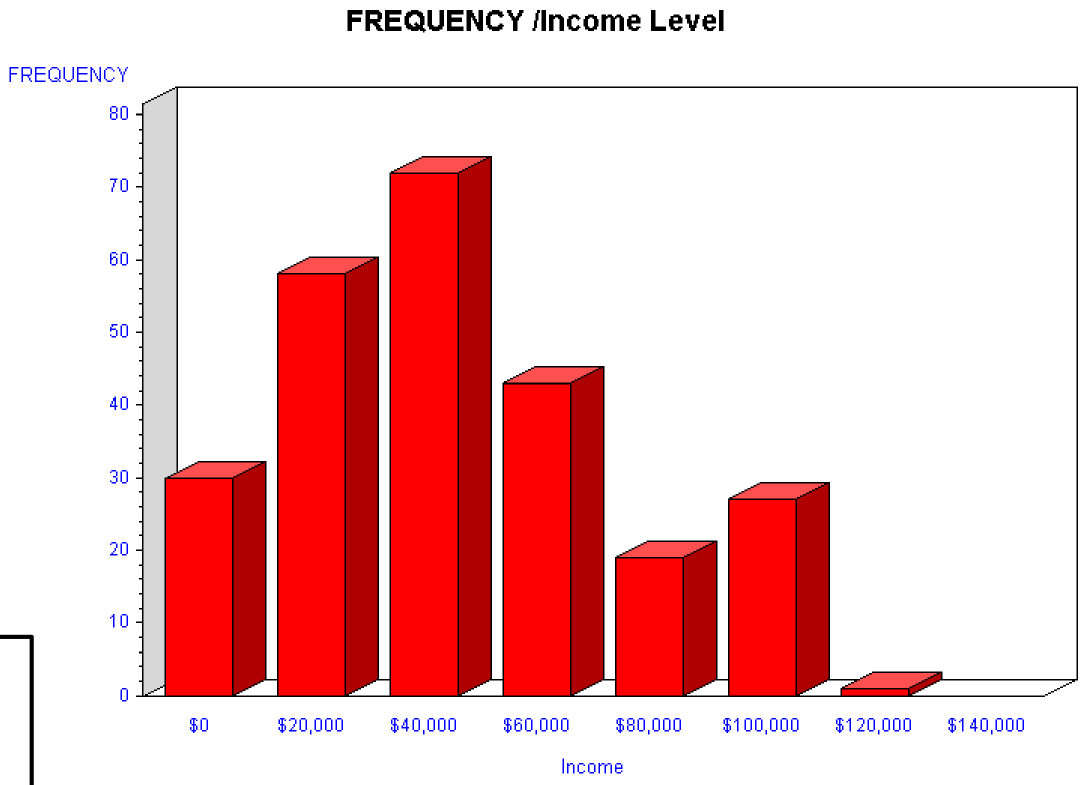


Using ‘MIDPOINTS=‘ Options

When you produce BAR chart for continuous variable ,the ‘**MIDPOINTS=**‘ Option can be applied in ‘PROC GCHART’ to rearrange the midpoints that are automatically generated by SAS.

```
pattern value=SOLID color=RED;

proc gchart data=survey;
   VBAR3D income/CTEXT=BLUE midpoints=0 to 150000 by 20000;
   format income dollar11.;
   Title 'FREQUENCY /Income Level';
run;
quit;
```

This can improve the look of your chart!



Producing Chart for General Summary Statistics

What are other statistics **other than FREQUENCY** can PROC GCHART plot?

You can specify 'TYPE=' option in BAR chart procedure, also specify the 'SUMVAR' variable used for sum or mean calculation

If the 'SUMVAR=' is not used, 'TYPE=' can be one of the following:

- **FREQ: frequency (the default)**
- **CFREQ: cumulative frequency**
- **PERCENT PCT: percentage**
- **CPERCENT CPCT: cumulative percentage**

If the SUMVAR= option is used, 'TYPE=' can be one the following:

- **SUM: sum (the default)**
- **MEAN: mean**

Producing Chart for General Summary Statistics

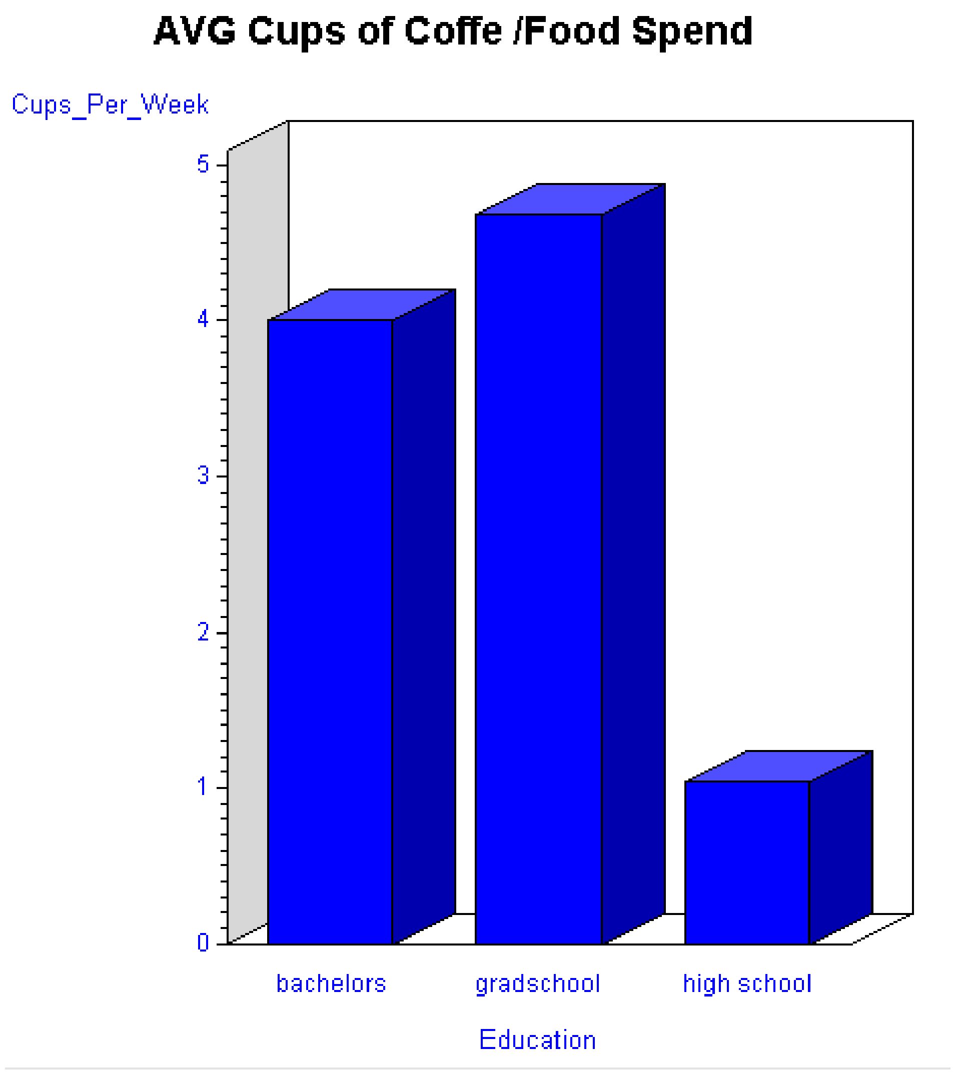
First Example

```
pattern value=SOLID color=BLUE;

proc gchart data=survey;
  VBAR3D Education
  /CTEXT=BLUE sumvar=Cups_Per_Week type=mean;
  Title 'AVG Cups of Coffe /Food Spend';
run;
quit;
```

Second Example

```
proc gchart data=survey;
   VBAR spend_food /NOSTATS sumvar=income type=mean ;
   format income spend_food dollar11.;
   Title 'AVG Income /Food Spend';
 run;
 quit;
```

Here you can use the continuous variable 'spend_food' as chart variable and 'income' as SUMVAR variable



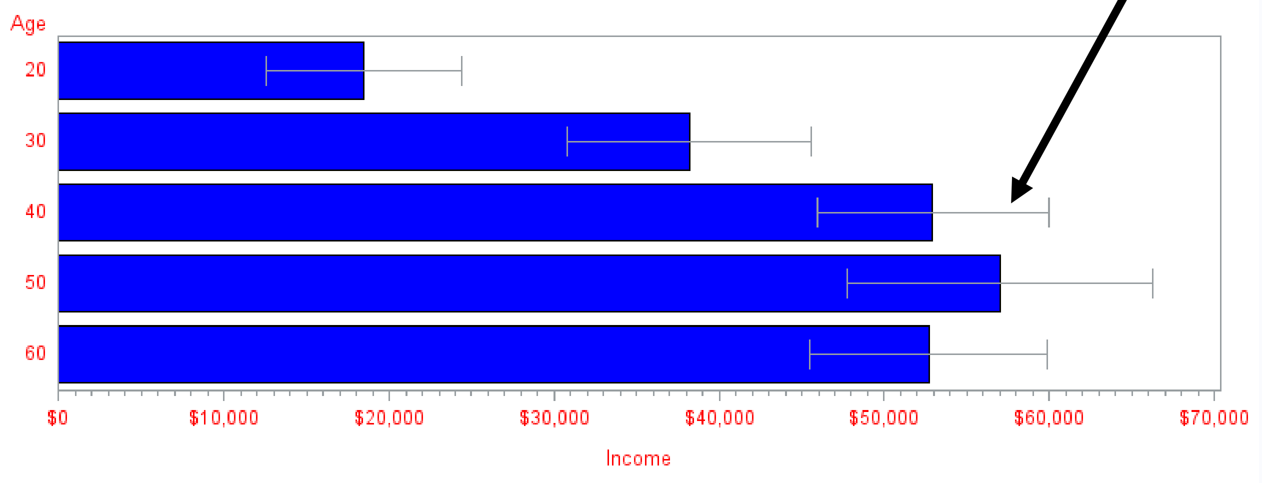
Using 'DISCRETE' Options in BAR Chart

When you produce BAR chart for a numeric variable ,the '**DISCRETE**' Option can be used in 'PROC GCHART' to treat each unique value as a category or class.

```
pattern value=SOLID color=BLUE;

proc gchart data=survey;
   HBAR age/NOSTATS CTEXT=RED sumvar=income
   DISCRETE type=mean clm=95;
   format income dollar11.;
   Title 'AVG Income /Age Group';
run;
quit;
```

If you do not use 'DISCRETE' option, SAS will group age value then plot!

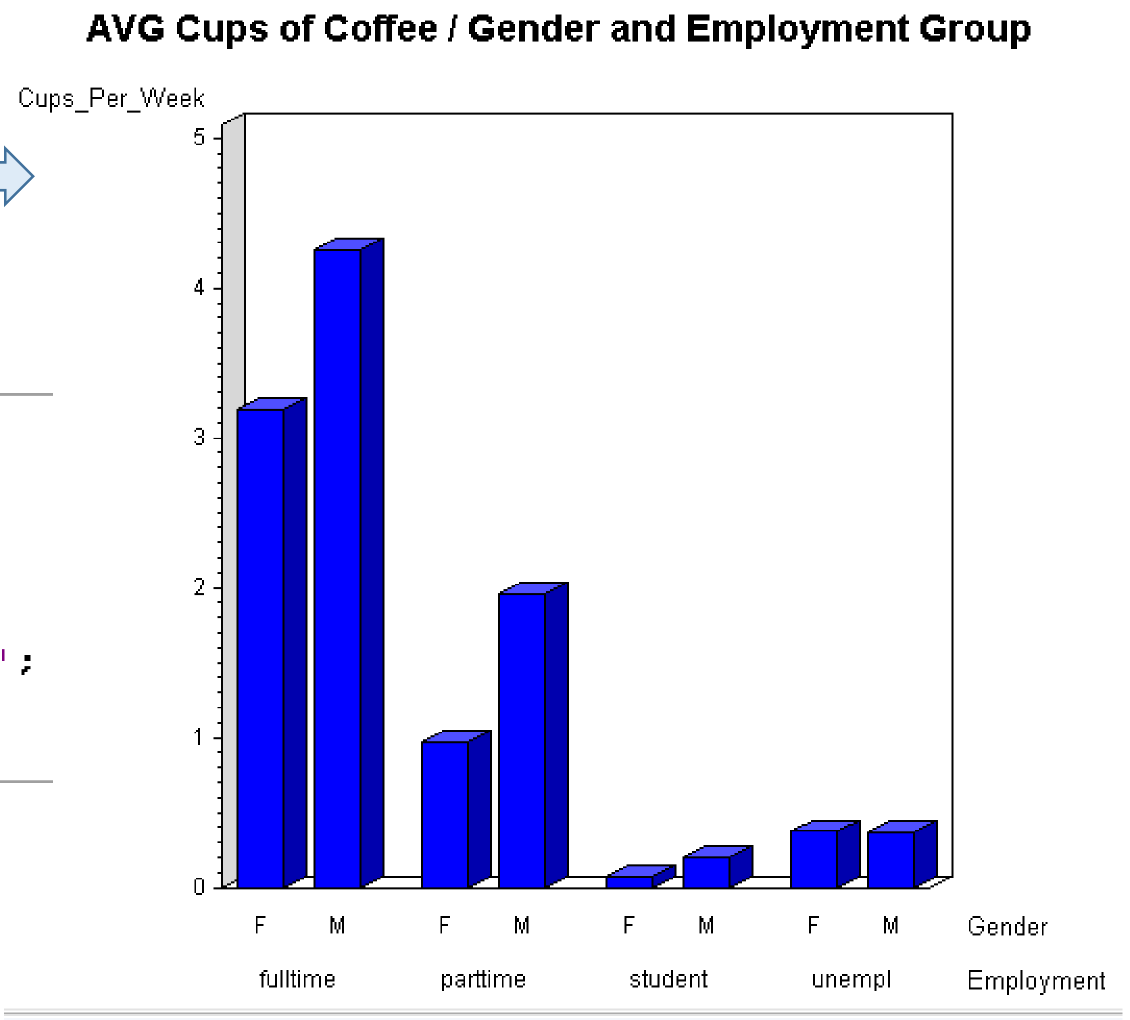
Use 95% confidence interval



Using ‘GROUP‘ Options in BAR Chart

You can add another stratus variable into BAR chart using ‘GROUP=‘ option.

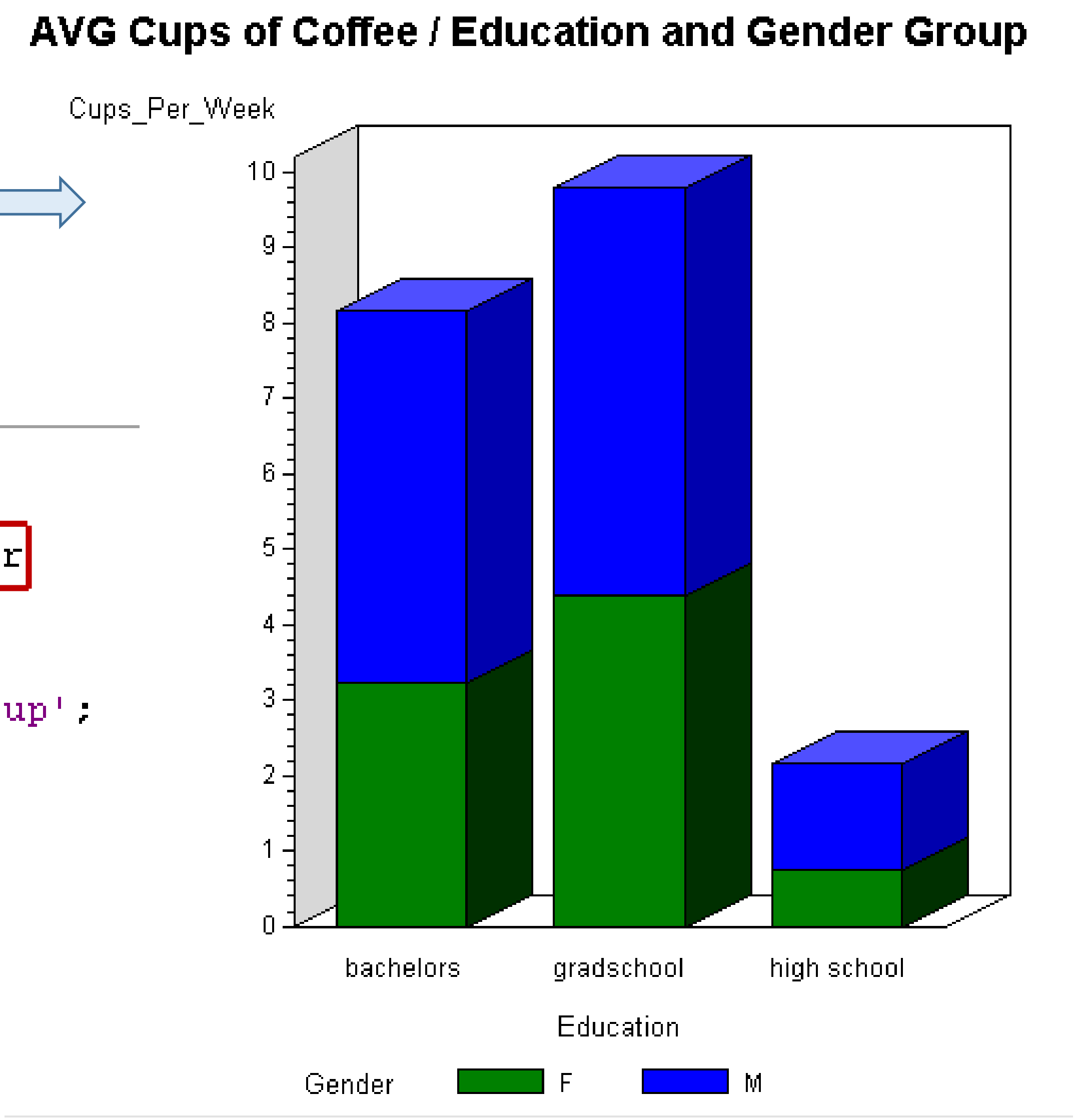
```
pattern value=SOLID color=BLUE;

proc gchart data=survey;
  VBAR3D gender /NOSTATS CTEXT=BLACK group=employment
              sumvar=Cups_Per_Week
              type=mean;
  Title 'AVG Cups of Coffee / Gender and Employment Group';
run;
quit;
```



Using ‘SUBGROUP‘ Options in BAR Chart

The other method to add another stratus variable into BAR chart is to apply ‘SUBGROUP=‘ option.

```
pattern1 value=SOLID color=GREEN;
pattern2 value=SOLID color=BLUE;

proc gchart data=survey;
   VBAR3D Education /NOSTATS CTEXT=BLACK subgroup=gender
               sumvar=Cups_Per_Week
               type=mean;
   Title 'AVG Cups of Coffee / Education and Gender Group';
run;
quit;
```



Agenda

- ❑ Traditional SAS/GRAPH System
  - ➢ Introducing Traditional SAS/GRAPH System
  - ➢ Creating Scatter and Series Plot Using PROC GPLOT
  - ➢ Producing Bar and Pie Chart Using PROC GCHART
- ❑ New SAS/GRAPH System after SAS 9.2
  - ➢ Overview of New SAS/GRAPH System
  - ➢ ODS Graphics
  - ➢ Procedures for Statistical Graphics
  - ➢ Mastering Graph Template Language (GTL)

SAS/GRAPH: New Graphics Tool

After Version 9.2

Graph Template Language (GTL).

- Using ODS Template with type 'STATGRAPH'
- Creating Graph by Rendering Data into Template
- PROC SGRENDER

Step 1:
Using 'PROC TEMPLATE' to define graph format template

Step 2:
Using 'PROC SGRENDER' to render data to the defined template

Step 3:
Get Plot !

New 'Statistical Graphics Procedures' which are also built upon GTL

Additional Graph Tools:

- ODS GRAPHICS
- PROC SGPLOT
- PROC SGPANEL
- PROC SGSCATTER
- PROC SGDESIGN

Agenda

- ❑ Traditional SAS/GRAPH System
  - ➢ Introducing Traditional SAS/GRAPH System
  - ➢ Creating Scatter and Series Plot Using PROC GPLOT
  - ➢ Producing Bar and Pie Chart Using PROC GCHART
- ❑ New SAS/GRAPH System after SAS 9.2
  - ➢ Overview of New SAS/GRAPH System
  - ➢ ODS Graphics
  - ➢ Procedures for Statistical Graphics
  - ➢ Mastering Graph Template Language (GTL)

ODS Graphics

ODS Graphics is an extension of ODS (the Output Delivery System), which delivers graph output from many SAS procedures such as ‘PROC FREQ’, ‘PROC UNIVARIATE’…

```
ODS GRAPHICS ON;
    PROC  <SAS PROCEDURE>
      ................;
    RUN;
ODS GRAPHICS OFF;
```

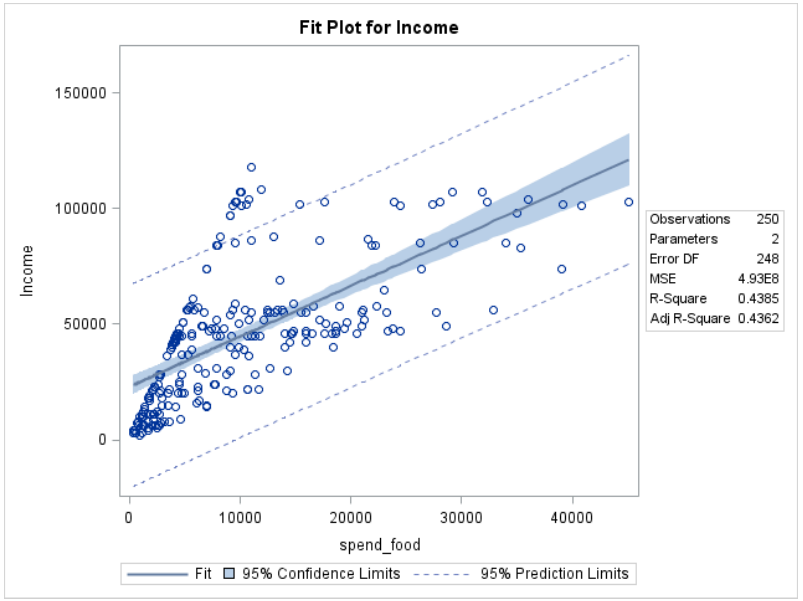
Example of ODS Graphics

Only select these two data sets to plot

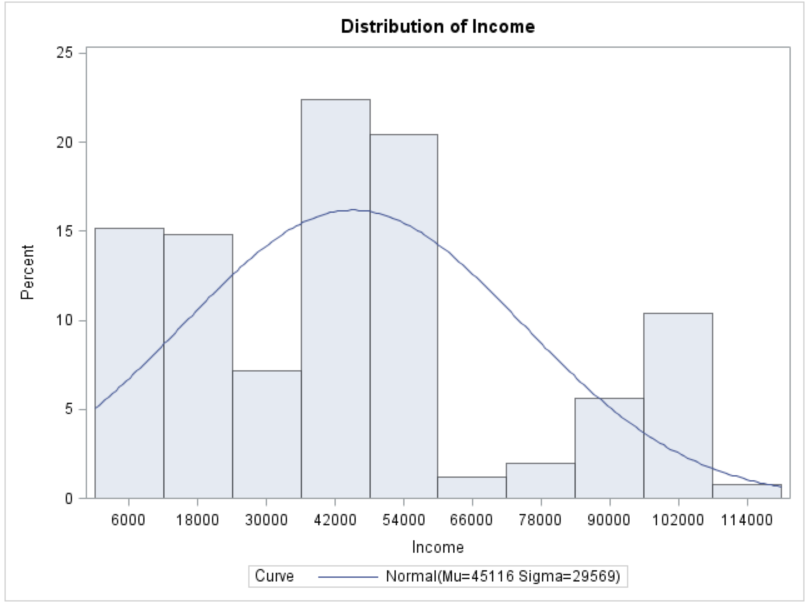
```
ods graphics on;
ods html;
ods select ParameterEstimates FitPlot;
proc reg data=survey;
model income=spend_food;
quit;
ods html close;
ods graphics off;
```

```
ods graphics on;
ods html;
proc univariate data=survey;
   var income;
   histogram income /normal;
quit;
ods html close;
ods graphics off;
```

Plot histogram output from ‘PROC Univariate’



Observations	250
Parameters	2
Error DF	248
MSE	4.93E8
R-Square	0.4385
Adj R-Square	0.4362



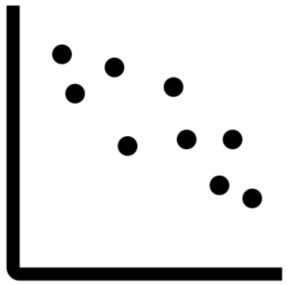
Agenda

- ❑ Traditional SAS/GRAPH System
    - ➢ Introducing Traditional SAS/GRAPH System
    - ➢ Creating Scatter and Series Plot Using PROC GPLOT
    - ➢ Producing Bar and Pie Chart Using PROC GCHART
- ❑ New SAS/GRAPH System after SAS 9.2
    - ➢ Overview of New SAS/GRAPH System
    - ➢ ODS Graphics
    - ➢ Procedures for Statistical Graphics
    - ➢ Mastering Graph Template Language (GTL)

Statistical Graphics Procedure: SGPLOT

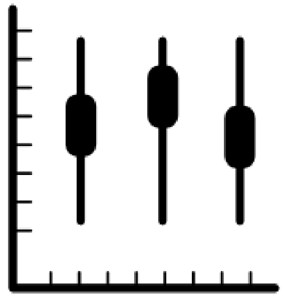
The 'PROC SGPLOT' can be applied to produce overlaid graphs on a single set of axes. It is an enhanced version of 'PROC GPLOT' because you can use it to create many types of statistical graphics beyond reach of traditional procedures.

Scatter Plot



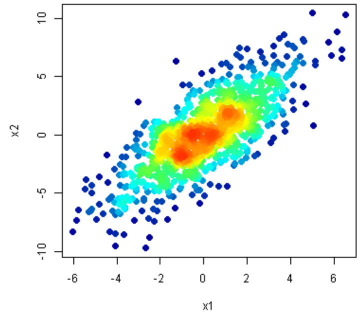
Series Plot

Box Plot

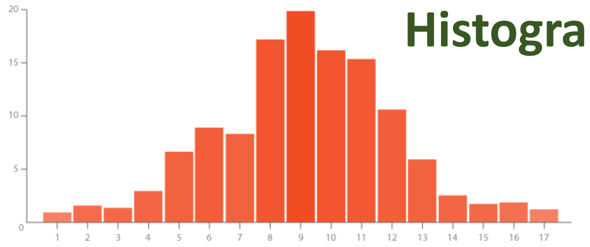


Bat Chart

Ellipse Plot

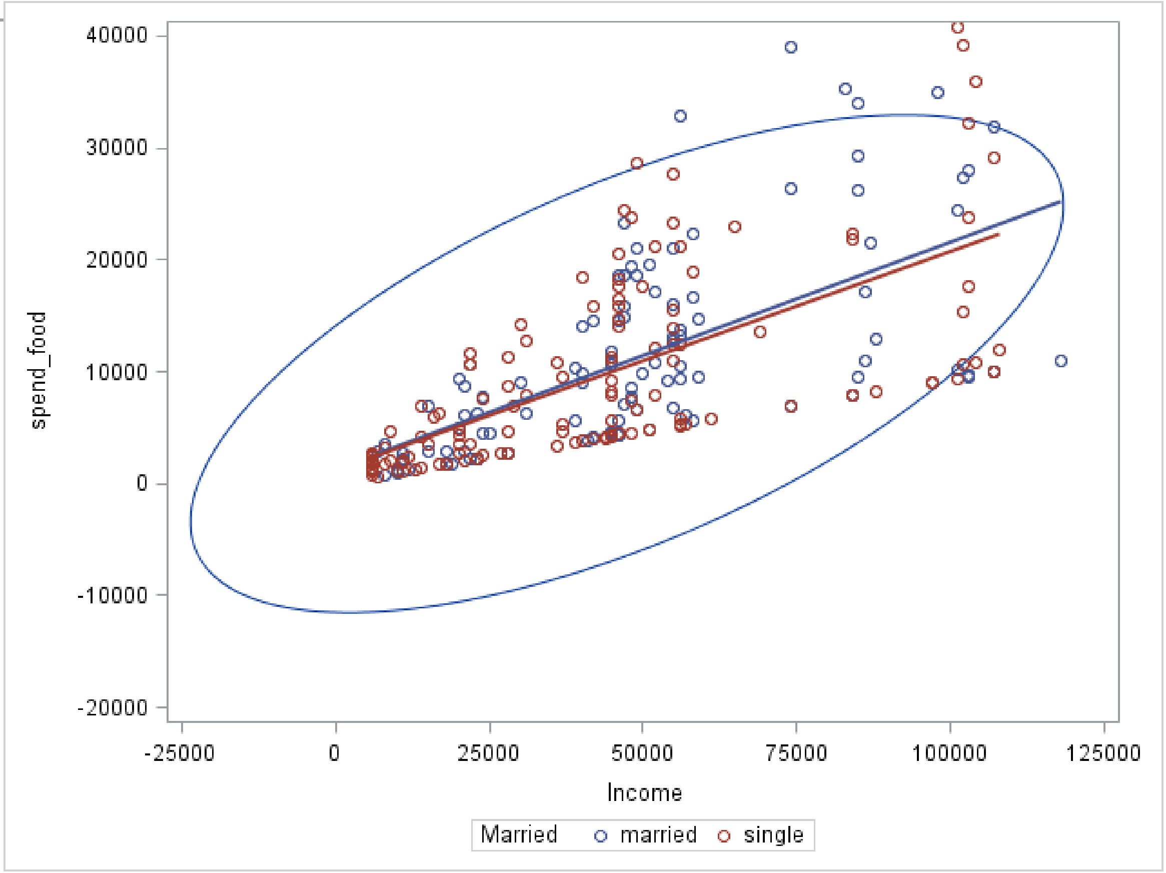


Histogram



Example of 'PROC SGPLOT': Scatter and Ellipse Plot

```
proc sort data=survey out=survey_married;
 by married;
run;

proc sgplot data=survey_married;
  /***Data restriction for creating graph******/
  where (income>spend_food and 5000<income<120000);
  /***Define the maximum and minimum vaues of
  X AXIS and Y AXIS******/
  XAXIS max=120000 min=-20000;
  YAXIS max=40000 min=-20000;;
  /***produce scatter plot for income and spend_food
    stratified by married indicator group**/
  scatter x=income y=spend_food /group=married;
  /**Produce ellipse plot for income and spend_food **/
  ellipse x=income y=spend_food;
   /**Draw regression line for income and spend_food
      stratified by married indicator group**/
  reg y=spend_food x=income / group=married;
run;
quit;
```

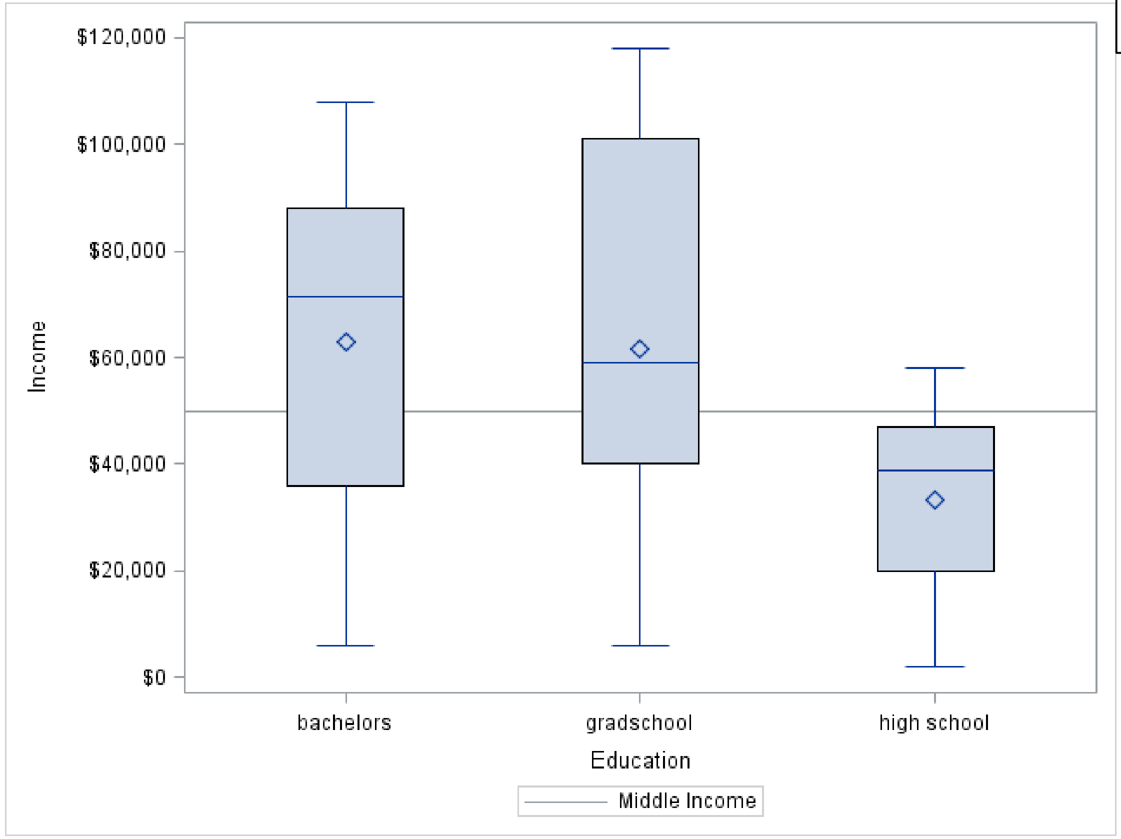


Example of ‘PROC SGPLOT’: BOX Plot

```
proc sgplot data=survey;
    refline 50000 /axis=y name='middle' legendlabel='Middle Income';
    keylegend "middle" /location=OUTSIDE position=BOTTOM;
    format income dollar11.;
    vbox income / category=education;
run;
```

Define legend for the reference line using the name ‘middle’.

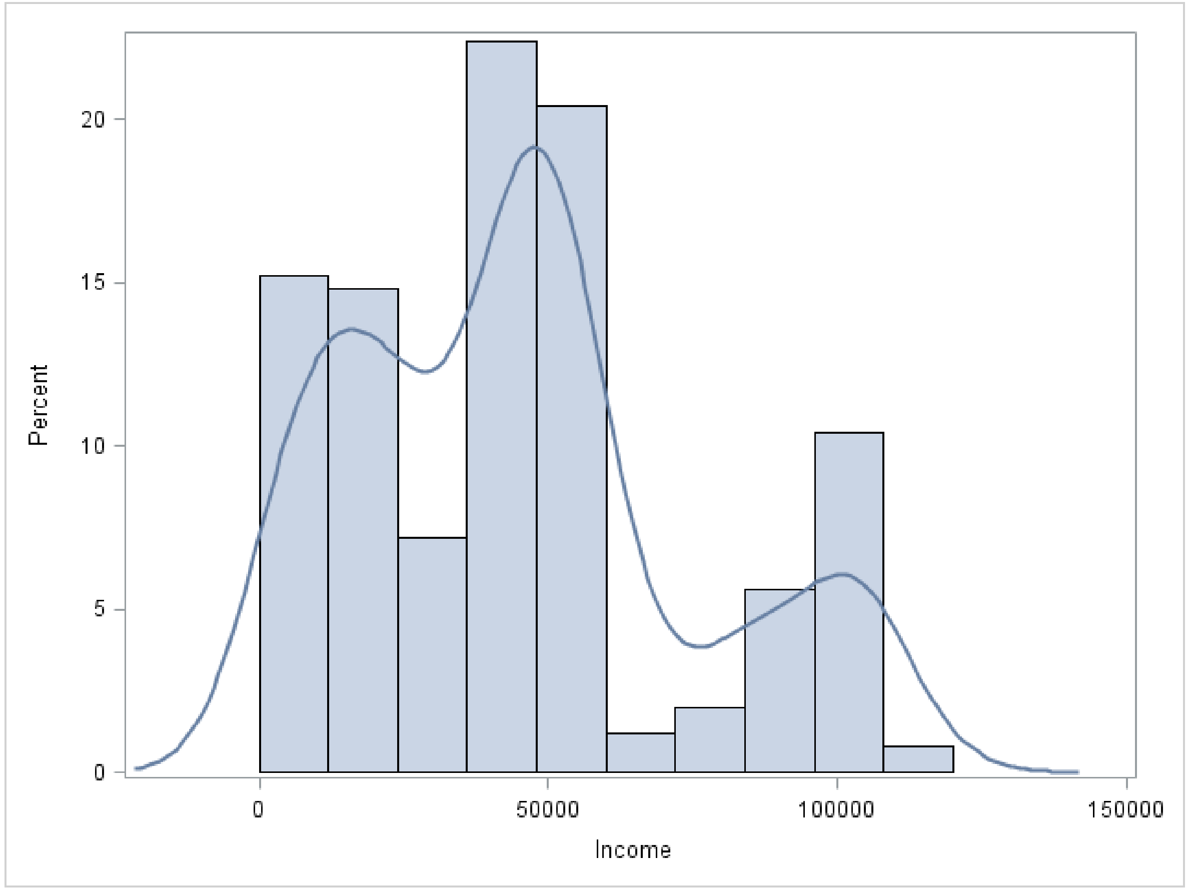
Create reference line for the box plot. The name of the plot is ‘middle’.



Example of ‘PROC SGPLOT’: Histogram and Density

```
proc sgplot data=survey;
   histogram income;
   density income/type=kernal;
   keylegend "middle" /location=OUTSIDE position=TOPLEFT;
run;
```

Overlay histogram and density plot



Statistical Graphics Procedure: SGPANEL

The 'PROC SGPANEL' creates multi-cells graph which is used to require a lot of work.

```
PROC SGPANEL;
    PANELBY variable(s) </options>;
    PLOT STATEMENT;
RUN;
```

For example, if ‘PANELBY’ variable is gender, then the graph for male and female would be created in two cells respectively.

Statistical Graphics Procedure: SGPANEL

Several Important ‘PANELBY’ options:

LAYOUT=PANEL|LATTICE : If you choose ‘LATTICE’ and have two classification variable, then the cells are arranged such that the value of the first variable are columns and the values of the second variable are rows. If you choose ‘PANEL’ (default) then cells are arranged by the settings of ‘COLUMNS’ and ‘ROWS’ (see below).

COLUMNS=n : Specify the number of columns in the panel.

ROWS=n : Specify the number of rows in the panel.

If you do not set numbers of columns and rows, they are automatically defined based on classifier's values and layout.

NOVARNAME : Remove the variable name and the ‘=‘ symbol from cell heading.

ONEPANEL : Place the whole panel into a single output.

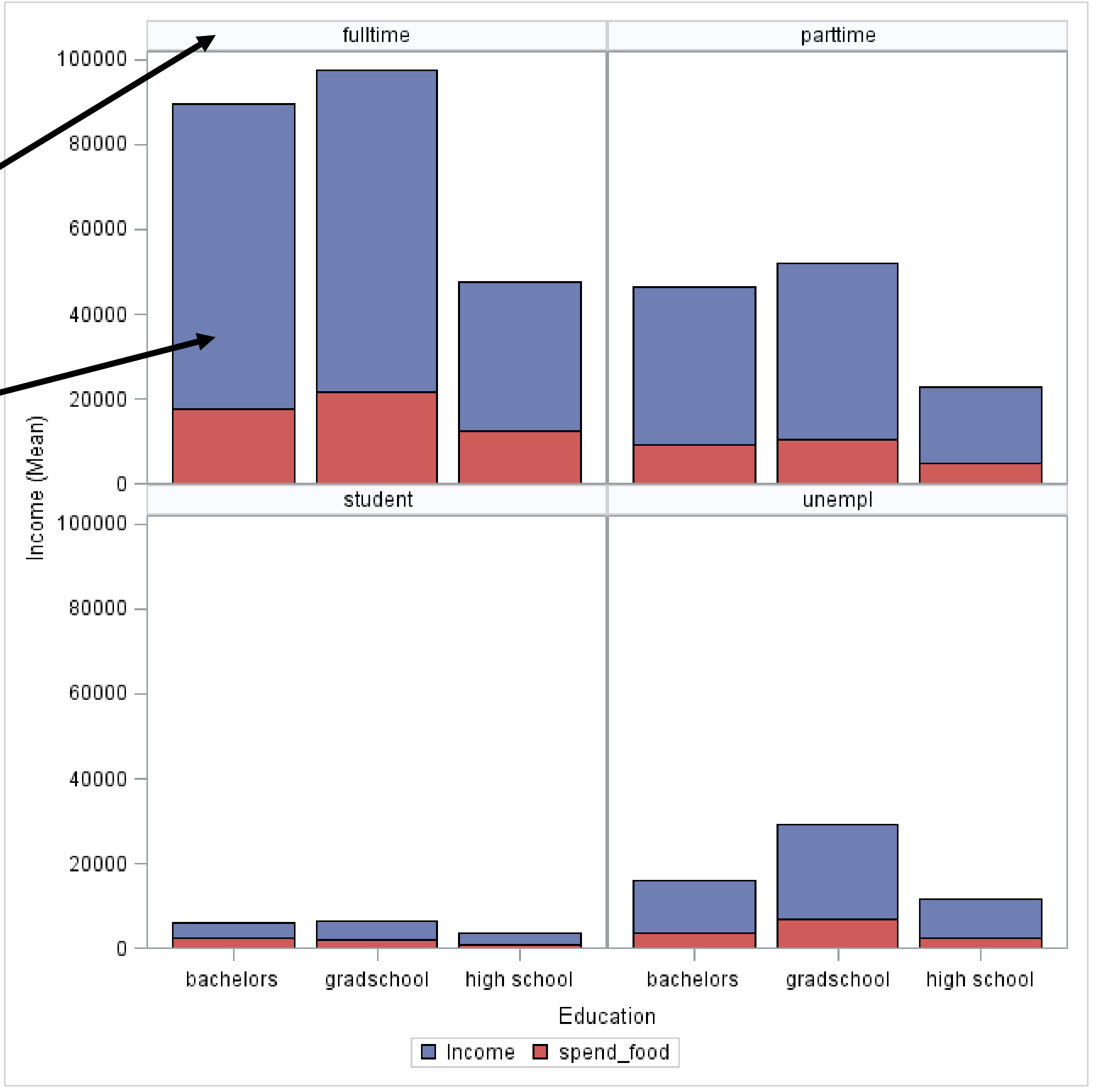
BORDER|NOBORDER : Add or remove the border around each cell.

Examples of ‘PROC SGPANEL’: Bar Chart

‘Education’ bar chart stratified by ‘area.

```
proc sgpanel data=survey;
   panelby employment / novarname;
   vbar education /response=income stat=mean;
   vbar education /response=spend_food stat=mean;
run;
```

Each panel includes two overlaid bar charts (response variables are income and spend_food respectively).

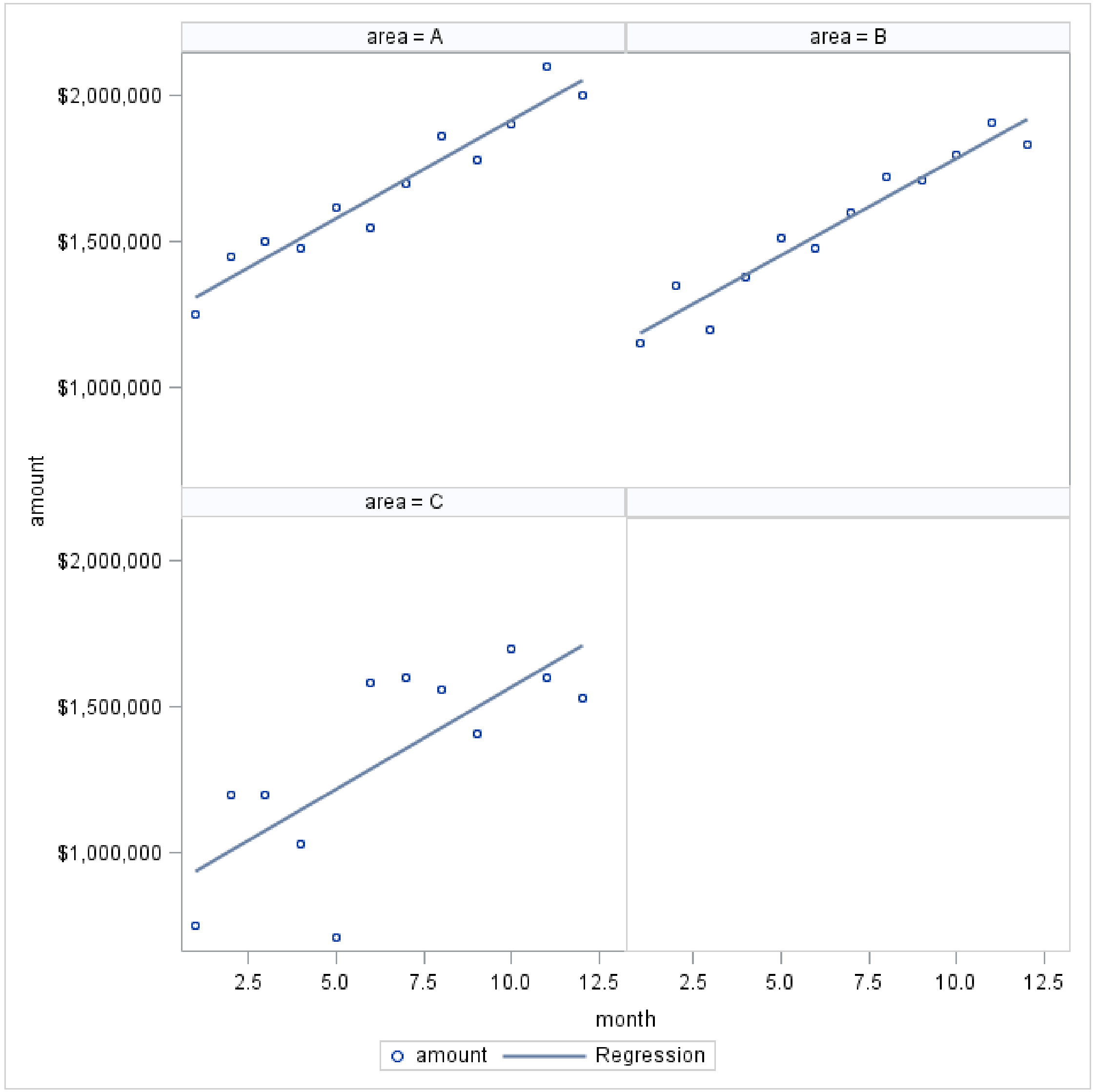


Examples of 'PROC SGPANEL': Scatter Plot

Multi-cell chart stratified by 'area.

```
proc sgpanel data=Sales_3;
   panelby area / noborder;
   scatter x=month y=amount;
   reg   x=month y=amount;
run;
```

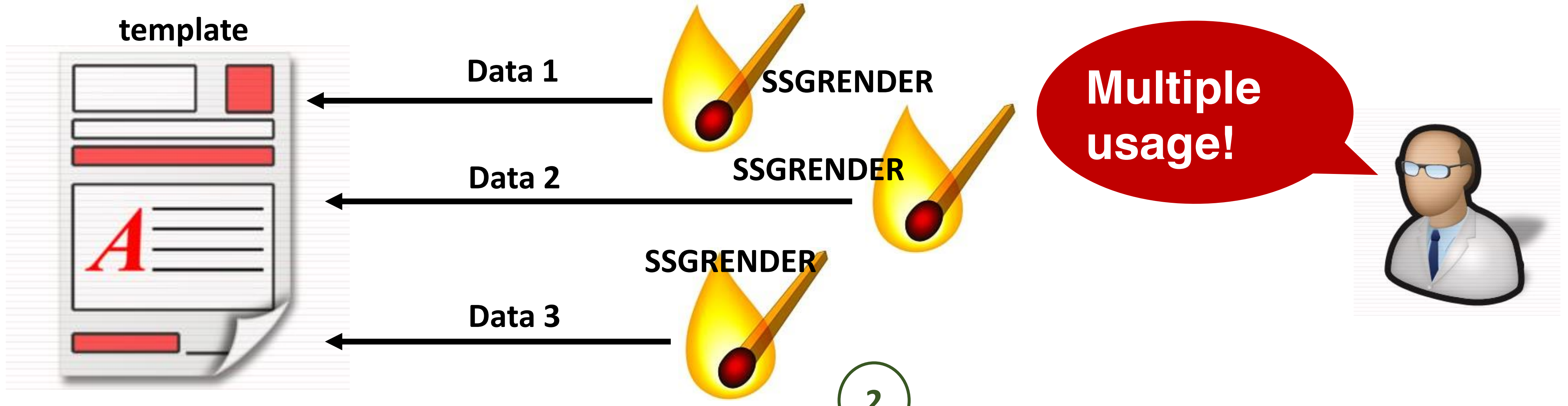
Overlay scatter plot and regression line: X AXIS is month and Y AXIS is amount.



Agenda

- ❑ Traditional SAS/GRAPH System
  - ➢ Introducing Traditional SAS/GRAPH System
  - ➢ Creating Scatter and Series Plot Using PROC GPLOT
  - ➢ Producing Bar and Pie Chart Using PROC GCHART
- ❑ New SAS/GRAPH System after SAS 9.2
  - ➢ Overview of New SAS/GRAPH System
  - ➢ ODS Graphics
  - ➢ Procedures for Statistical Graphics
  - ➢ Mastering Graph Template Language (GTL)

Overview of Graph Template Language (GTL)



1

Define ‘**STATGRAPH**’ template using the GTL syntax. The template contains generic instruction for generating graph.

2

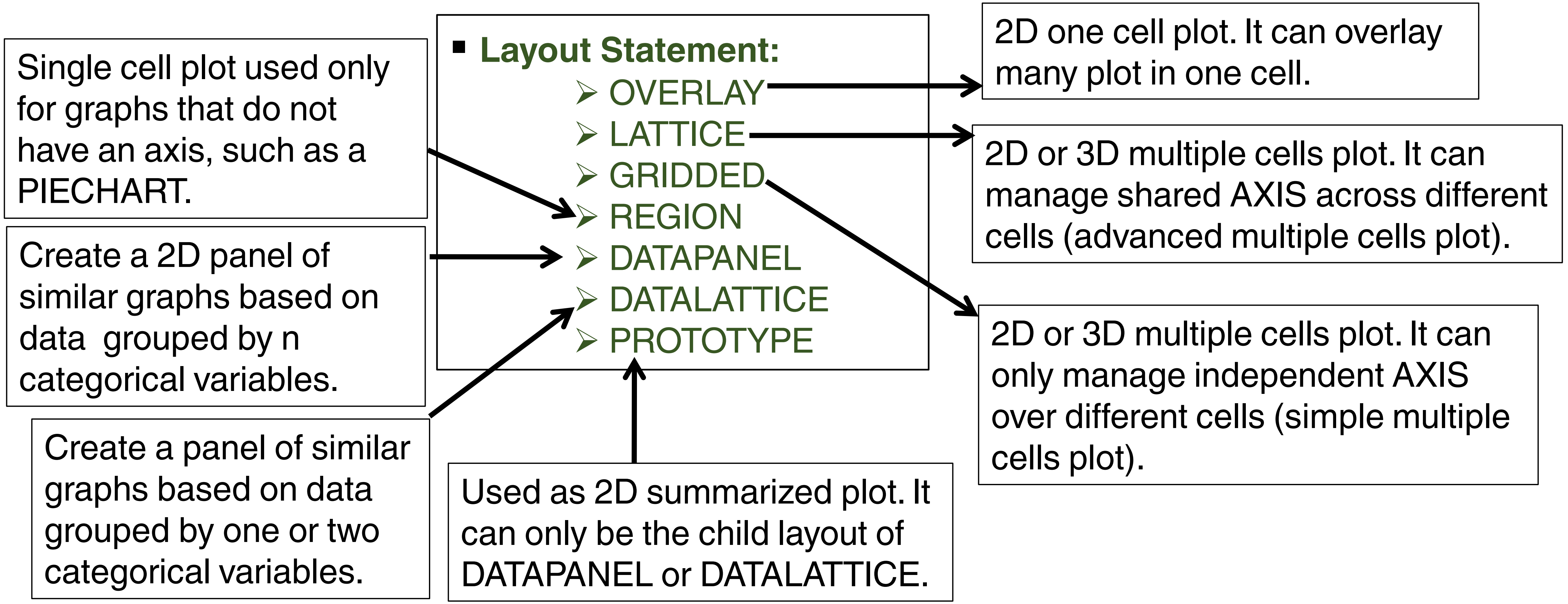
You can produce different types of graph by executing the SGRENDER procedure to assign specific data to the ‘STATGRAPH’ template.

Basic Elements in STATGRAPH Template

```
PROC TEMPLATE;
Define STATGRAPH <template name>;
BeginGraph;
EntryTitle <"title">;
Layout <layout name>;
Plot (XAXIS, YAXIS options… ) Statements
Legend (DiscreteLegend, ContinuousLegend)
EntryFOOTNOTE <footnote>;
EndGraph;
END;
```

Graphical Area

Layout Statement (1)



- **Layout Statement:**
  - ➢ OVERLAY
  - ➢ LATTICE
  - ➢ GRIDDED
  - ➢ REGION
  - ➢ DATAPANEL
  - ➢ DATALATTICE
  - ➢ PROTOTYPE

2D one cell plot. It can overlay many plot in one cell.

2D or 3D multiple cells plot. It can manage shared AXIS across different cells (advanced multiple cells plot).

2D or 3D multiple cells plot. It can only manage independent AXIS over different cells (simple multiple cells plot).

Single cell plot used only for graphs that do not have an axis, such as a PIECHART.

Create a 2D panel of similar graphs based on data grouped by n categorical variables.

Create a panel of similar graphs based on data grouped by one or two categorical variables.

Used as 2D summarized plot. It can only be the child layout of DATAPANEL or DATALATTICE.

Plot Statement (2)

- **Plot Statement:**
  - **Non Summarized**
    - SERIESPLOT
    - SCATTERPLOT
    - BLOCKPLOT
    - BANDPLOT
    - BOXPLOTPARM
  - **Summarized**
    - BARCHART
    - PIECHART
    - HISTOGRAM
    - DENSITYPLOT
    - BUBBLEPLOT
    - REGRESSIONPLOT

These are the popularly used graphs.

Examples

```
ScatterPlot X=month Y=amount / GROUP=area NAME="Sales";
```

```
SeriesPlot X=month Y=amount / GROUP=area NAME="Area";
```

```
regressionplot x=month y=amount;
```

```
barchartparm x=month y=amount;
```

Legend Statement (3)

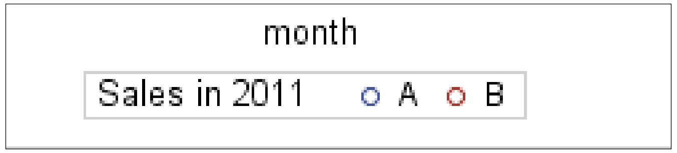
- Legend Statement:
  - ➢ DiscreteLegend
  - ➢ ContinuousLegend

Each entry consists of a graphical item

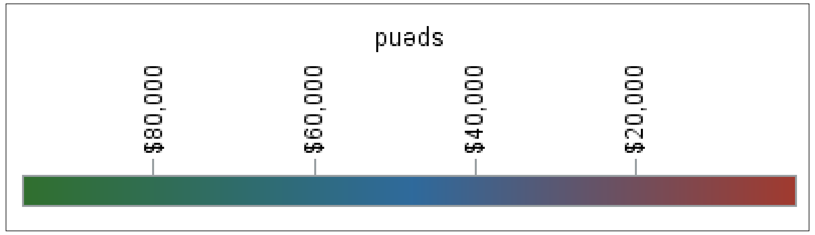
legend that maps a color gradient to response values

Examples

```
ScatterPlot X=month Y=amount / GROUP=area NAME="Sales";
DiscreteLegend "Sales" / ACROSS=3 TITLE="Sales in 2011";
```



```
scatterplot x=salary y=tax / name="sp"
markercolorgradient=spend
markerattrs=(symbol=circlefilled);

ContinuousLegend "sp" / title='spend';
```



Example 1: Series Plot

```
PROC TEMPLATE;
  DEFINE STATGRAPH series;
    BeginGraph;
      EntryTitle "Sales 2011";
      Layout overlay;
        SeriesPlot
          X=month Y=amount / GROUP=area NAME="Area";
          DiscreteLegend "Area" / TITLE="Areas"
        valueattrs=(size=11pt) autoitemsize=true;
      EndLayout;
    EndGraph;
  END;
RUN;

proc sgrender data=sales TEMPLATE=series;
run;
quit;
```

To use GTL, you first define '**STATGRAPH**' type template called 'series'

Define the title of the graph

You create a 'SeriesPlot' graph, which is actually the jointed scatter plot in traditional SAS/GRAPH system

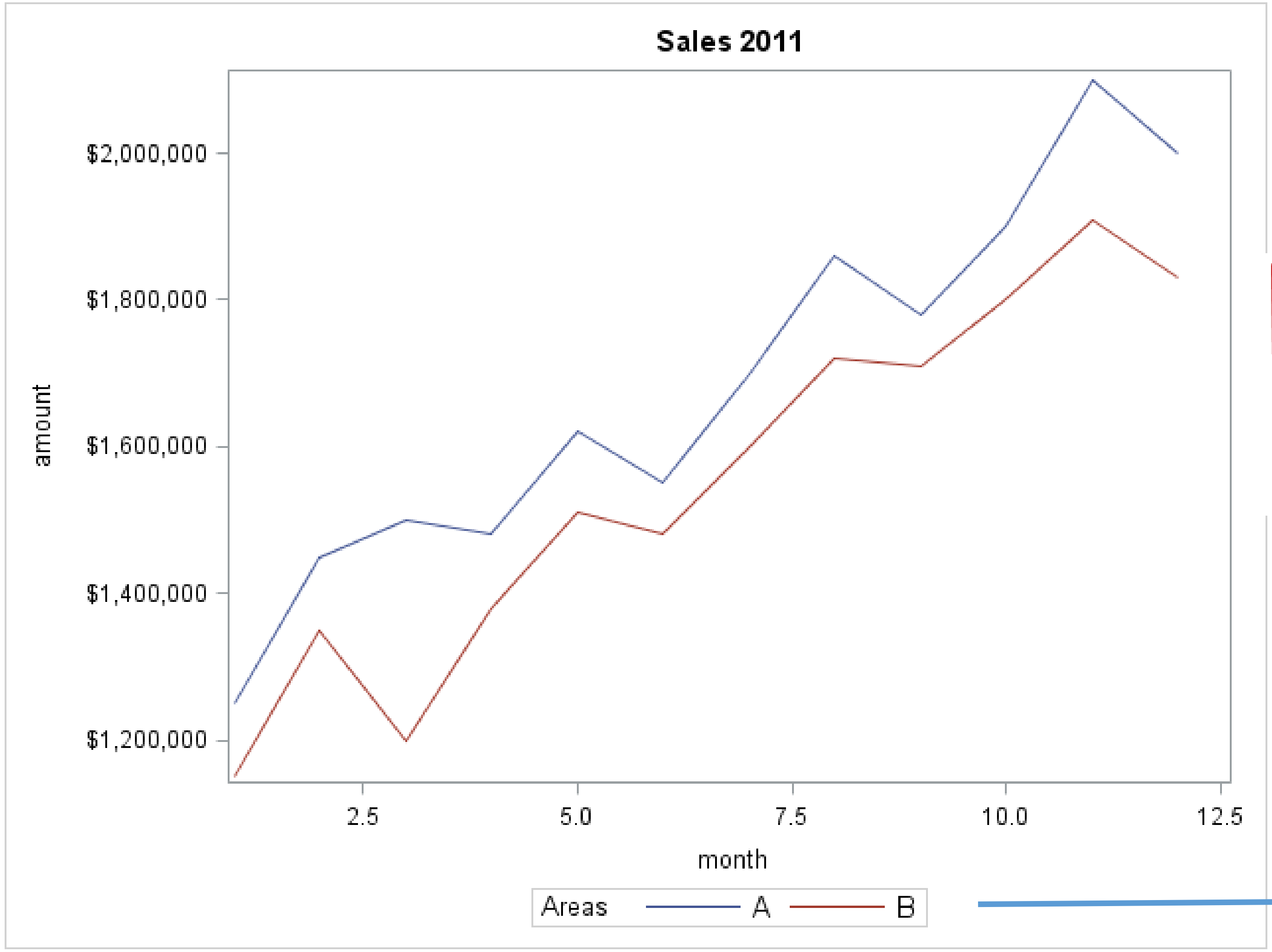
This is the graph area

Setting legend containing one or more legend entries (area names)

DEFINE a single cell layout using the key word 'overlay'. So you can also overlay the results of multiple graph statements.

You render the data into the template and produce plot

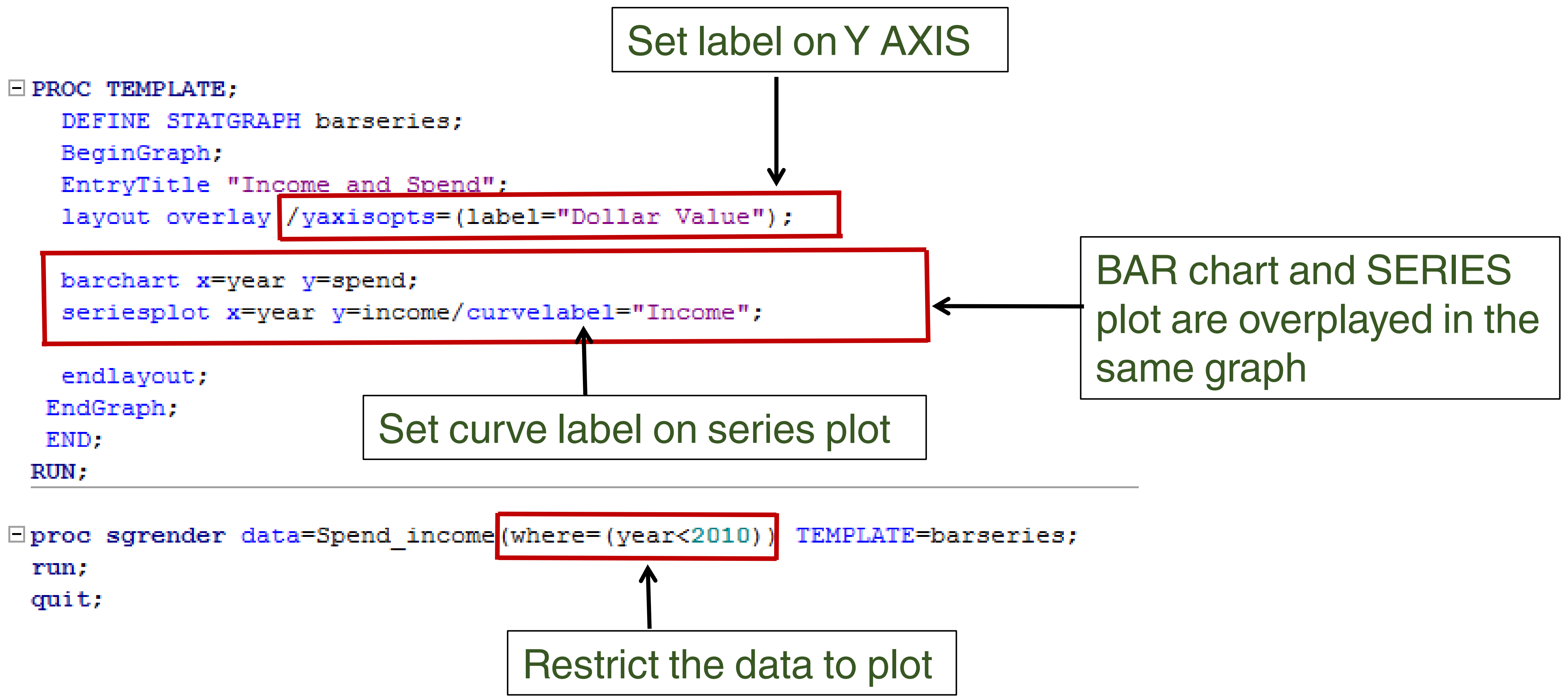
Example 1: Outcome



```
Layout overlay;
  SeriesPlot
    X=month Y=amount / GROUP=area NAME="Area";
    DiscreteLegend "Area" / TITLE="Areas"
   valueattrs=(size=11pt) autoitemsize=true;
EndLayout;
```

Use 'VALUEATTRS' option in 'DiscreteLegend' statement to set label size

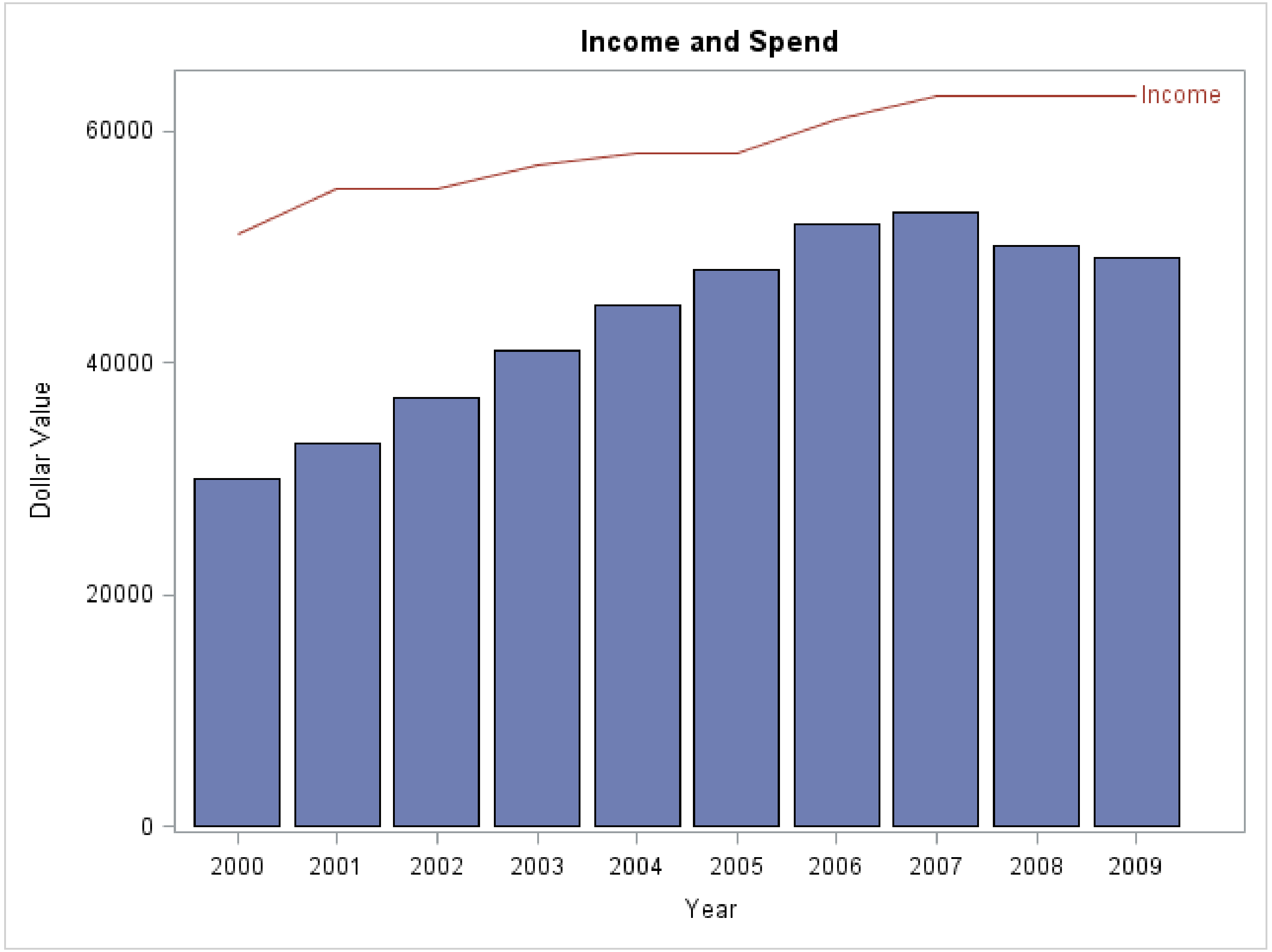
Example 2: Overlay Series and Bar Plots



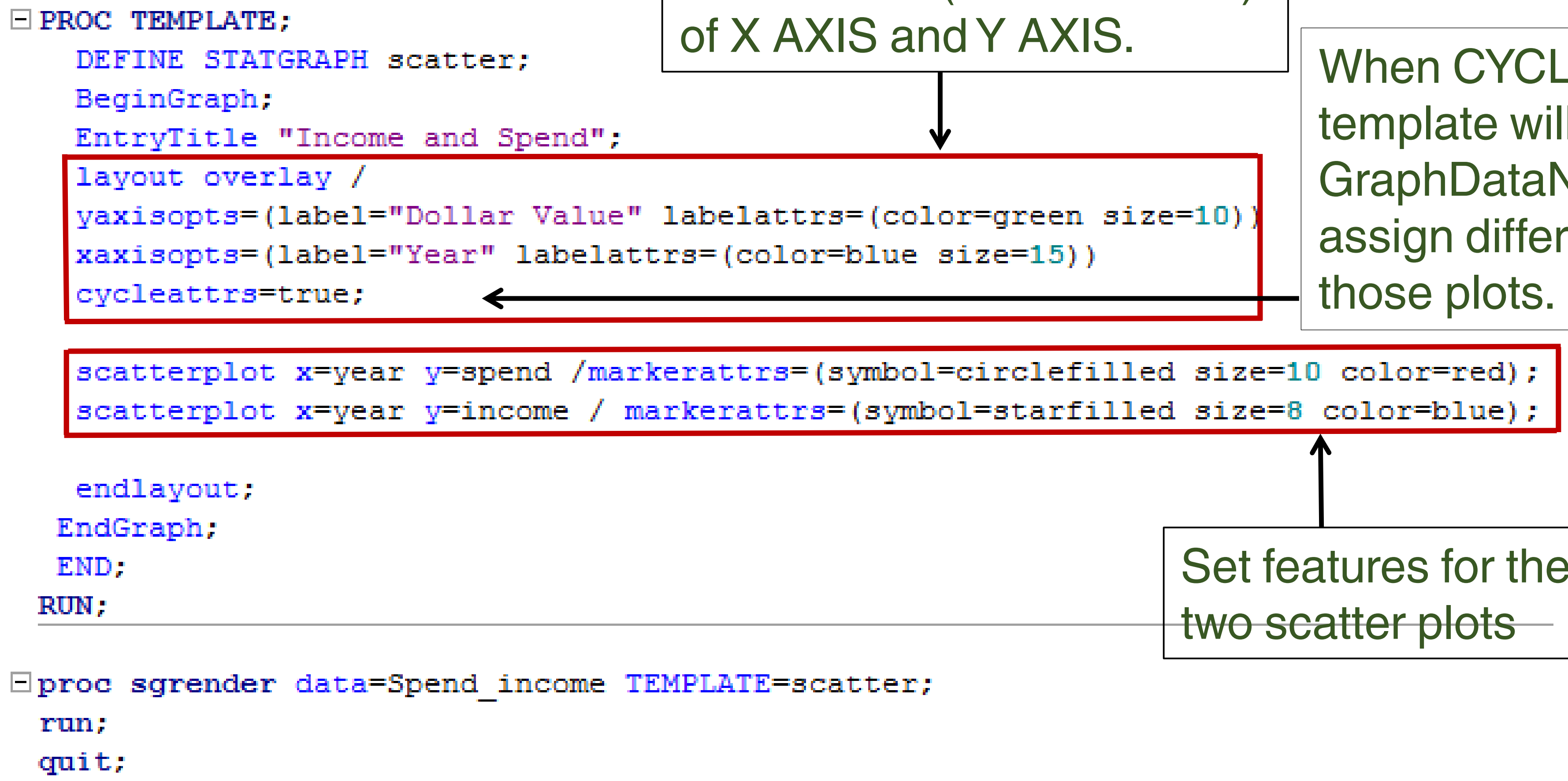
```
PROC TEMPLATE;
  DEFINE STATGRAPH barseries;
  BeginGraph;
  EntryTitle "Income and Spend";
  layout overlay /yaxisopts=(label="Dollar Value");

  barchart x=year y=spend;
  seriesplot x=year y=income/curvelabel="Income";

  endlayout;
 EndGraph;
 END;
RUN;

proc sgrender data=Spend_income(where=(year<2010)) TEMPLATE=barseries;
run;
quit;
```

Set label on Y AXIS
BAR chart and SERIES plot are overplayed in the same graph
Set curve label on series plot
Restrict the data to plot

Example 2: Outcome



Example 3: Scatter Plot



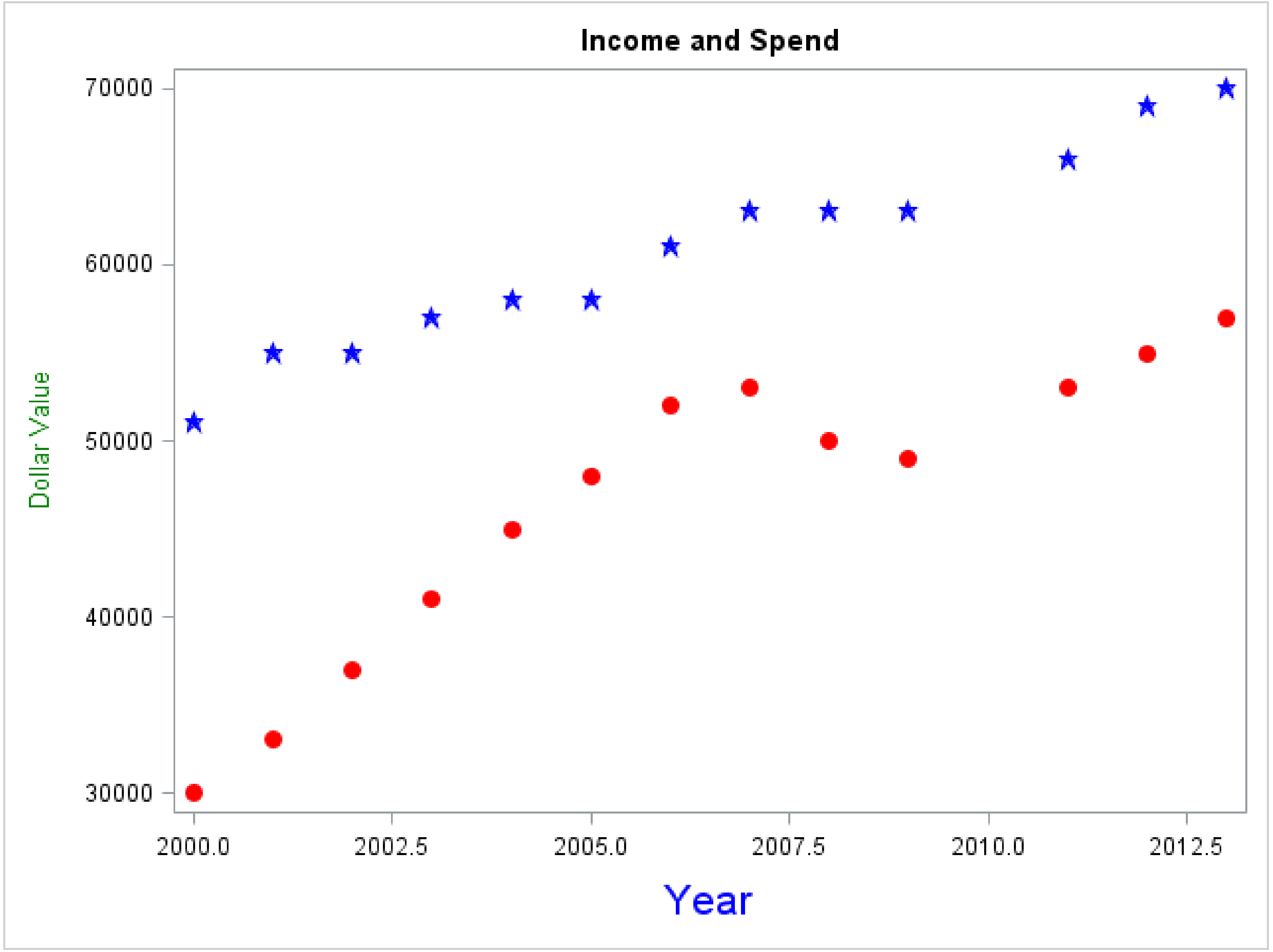
```
PROC TEMPLATE;
  DEFINE STATGRAPH scatter;
  BeginGraph;
  EntryTitle "Income and Spend";
  layout overlay /
  yaxisopts=(label="Dollar Value" labelattrs=(color=green size=10))
  xaxisopts=(label="Year" labelattrs=(color=blue size=15))
  cycleattrs=true;

  scatterplot x=year y=spend /markerattrs=(symbol=circlefilled size=10 color=red);
  scatterplot x=year y=income / markerattrs=(symbol=starfilled size=8 color=blue);

  endlayout;
 EndGraph;
 END;
RUN;

proc sgrender data=Spend_income TEMPLATE=scatter;
run;
quit;
```

Set features (color and size) of X AXIS and Y AXIS.

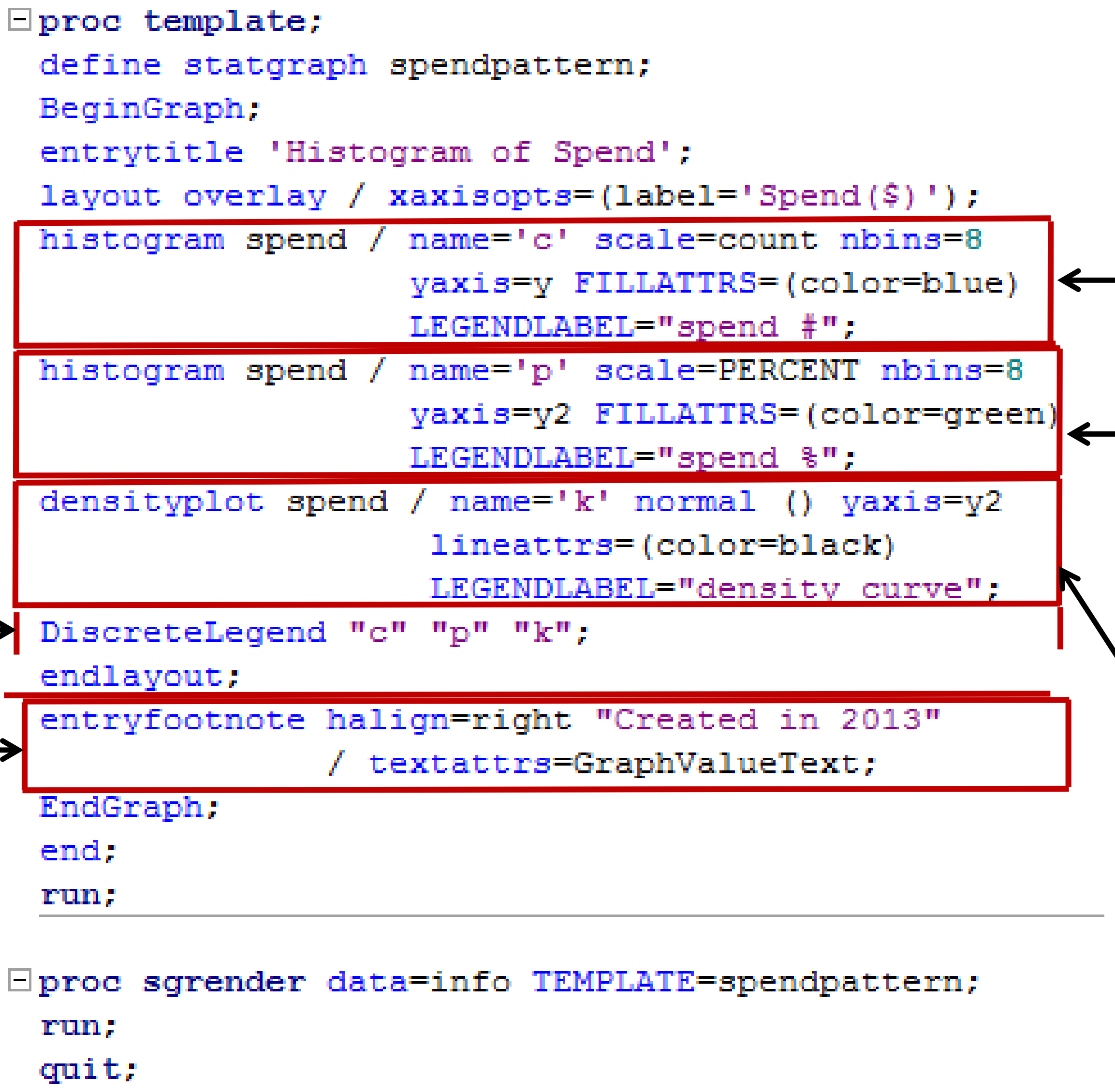
When CYCLEATTRS=TRUE, the template will use the GraphData1–GraphDataN style elements to assign different visual properties to those plots.

Set features for the markers of two scatter plots

Example 3: Outcome



Example 4: Histogram Plot



```
proc template;
 define statgraph spendpattern;
 BeginGraph;
 entrytitle 'Histogram of Spend';
 layout overlay / xaxisopts=(label='Spend($)');
 histogram spend / name='c' scale=count nbins=8
                   yaxis=y FILLATTRS=(color=blue)
                   LEGENDLABEL="spend #";
 histogram spend / name='p' scale=PERCENT nbins=8
                   yaxis=y2 FILLATTRS=(color=green)
                   LEGENDLABEL="spend %";
 densityplot spend / name='k' normal () yaxis=y2
                    lineattrs=(color=black)
                    LEGENDLABEL="density curve";
 DiscreteLegend "c" "p" "k";
 endlayout;
 entryfootnote halign=right "Created in 2013"
              / textattrs=GraphValueText;
 EndGraph;
 end;
 run;

proc sgrender data=info TEMPLATE=spendpattern;
 run;
 quit;
```

Set first histogram with Y AXIS's scale being 'count'. The label of legend is 'spend #'. The Y AXIS is on the left side (Y).

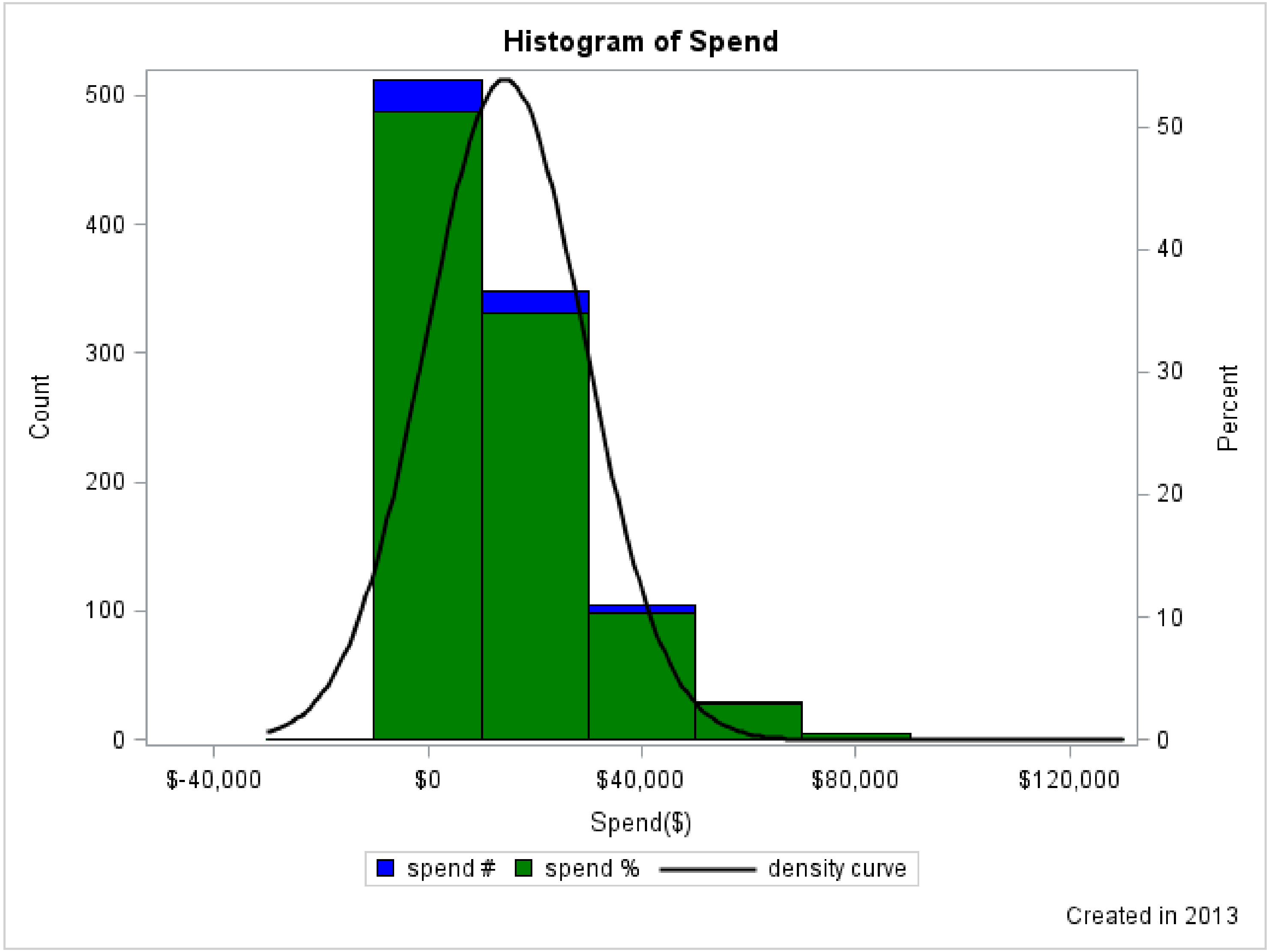
Set second histogram with Y AXIS's scale being 'percent'. The label of legend is 'spend %'. The Y AXIS is on the right side (Y2).

Draw density curve, overlaid with histogram.

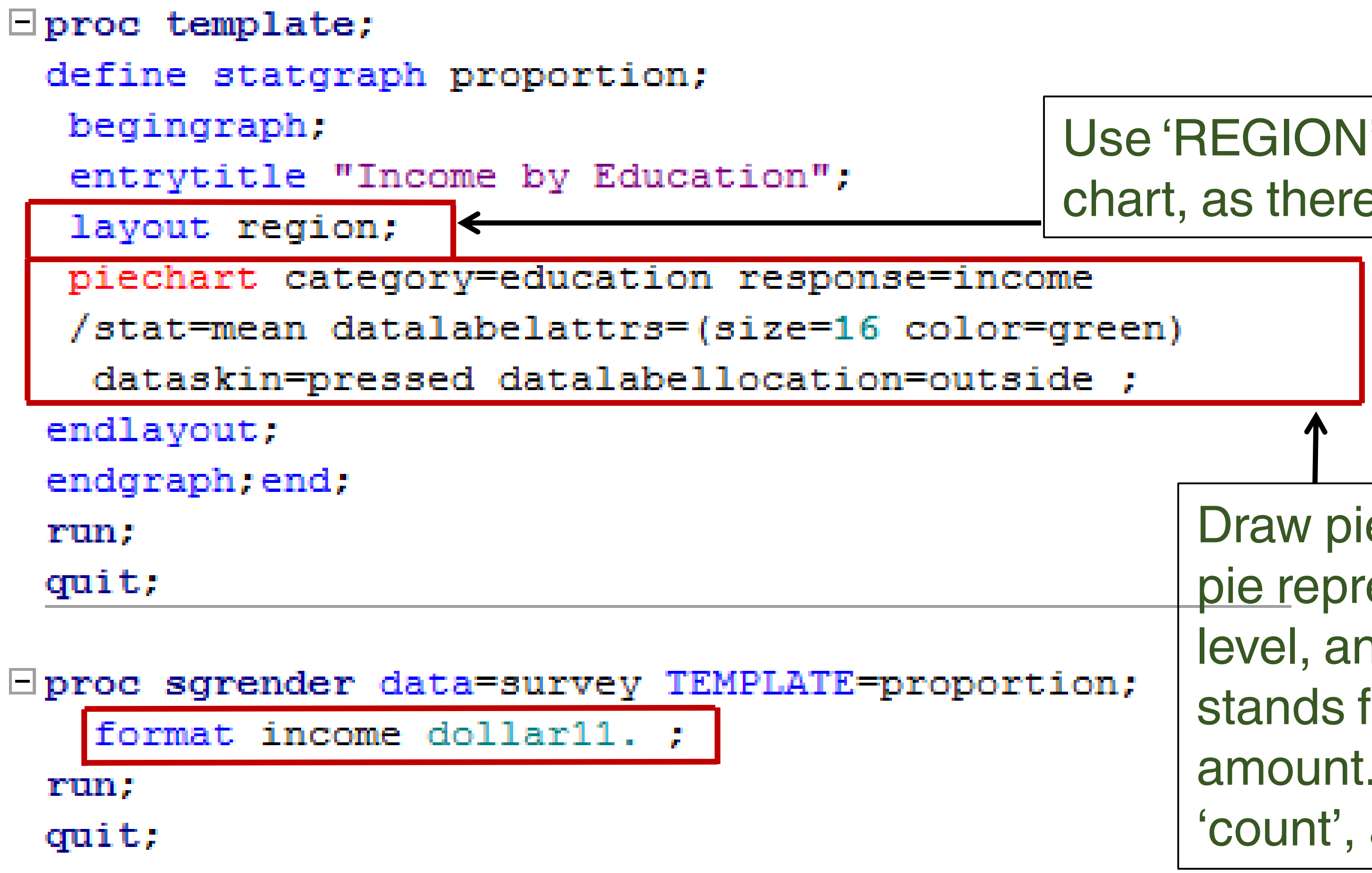
Legend for three plots

Footnote of the graph

Example 4: Outcome



Example 5: Pie Chart

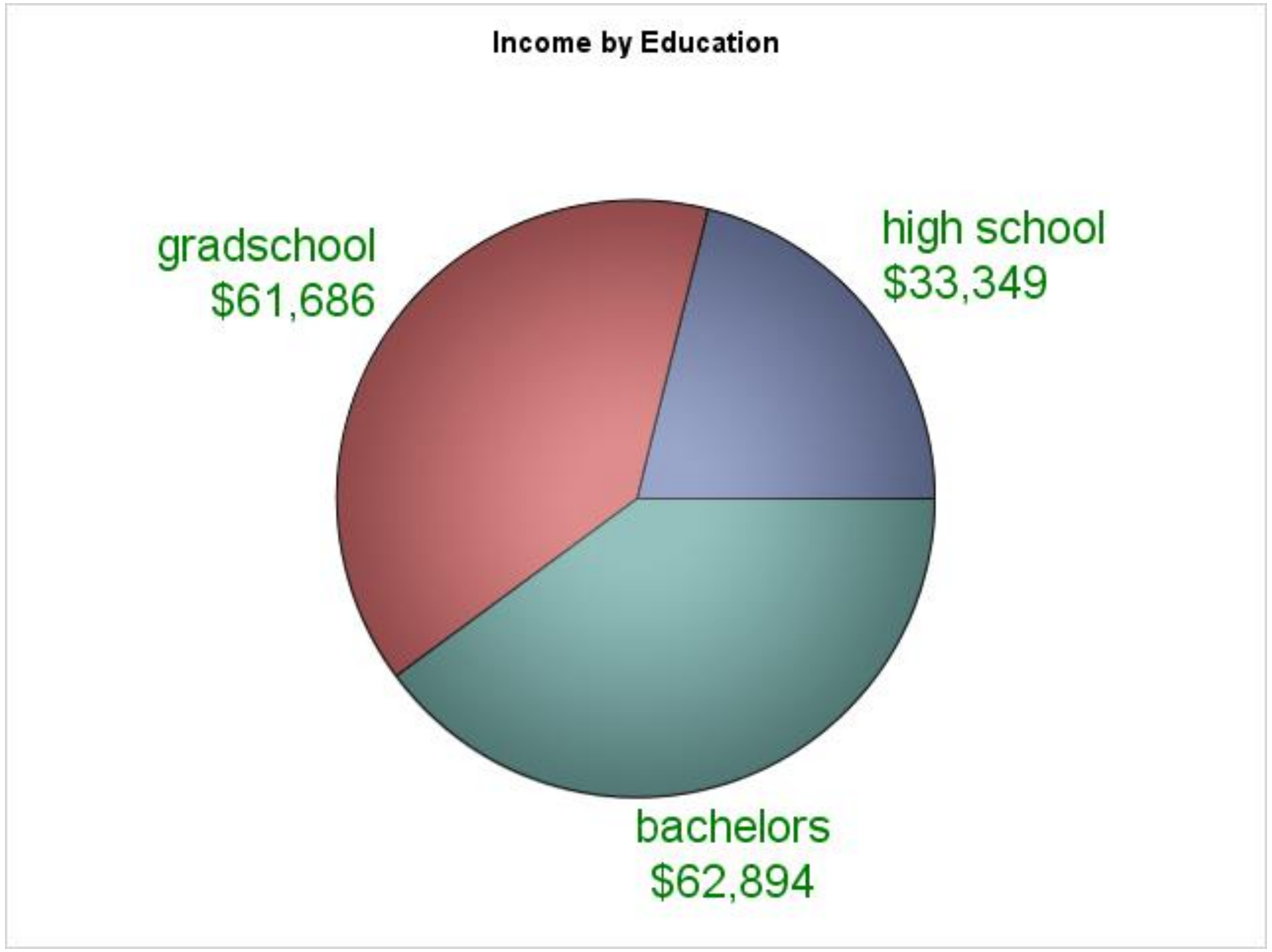


```
proc template;
 define statgraph proportion;
  begingraph;
  entrytitle "Income by Education";
  layout region;
  piechart category=education response=income
  /stat=mean datalabelattrs=(size=16 color=green)
   dataskin=pressed datalabellocation=outside ;
 endlayout;
 endgraph;end;
 run;
 quit;

proc sgrender data=survey TEMPLATE=proportion;
   format income dollar11. ;
 run;
 quit;
```

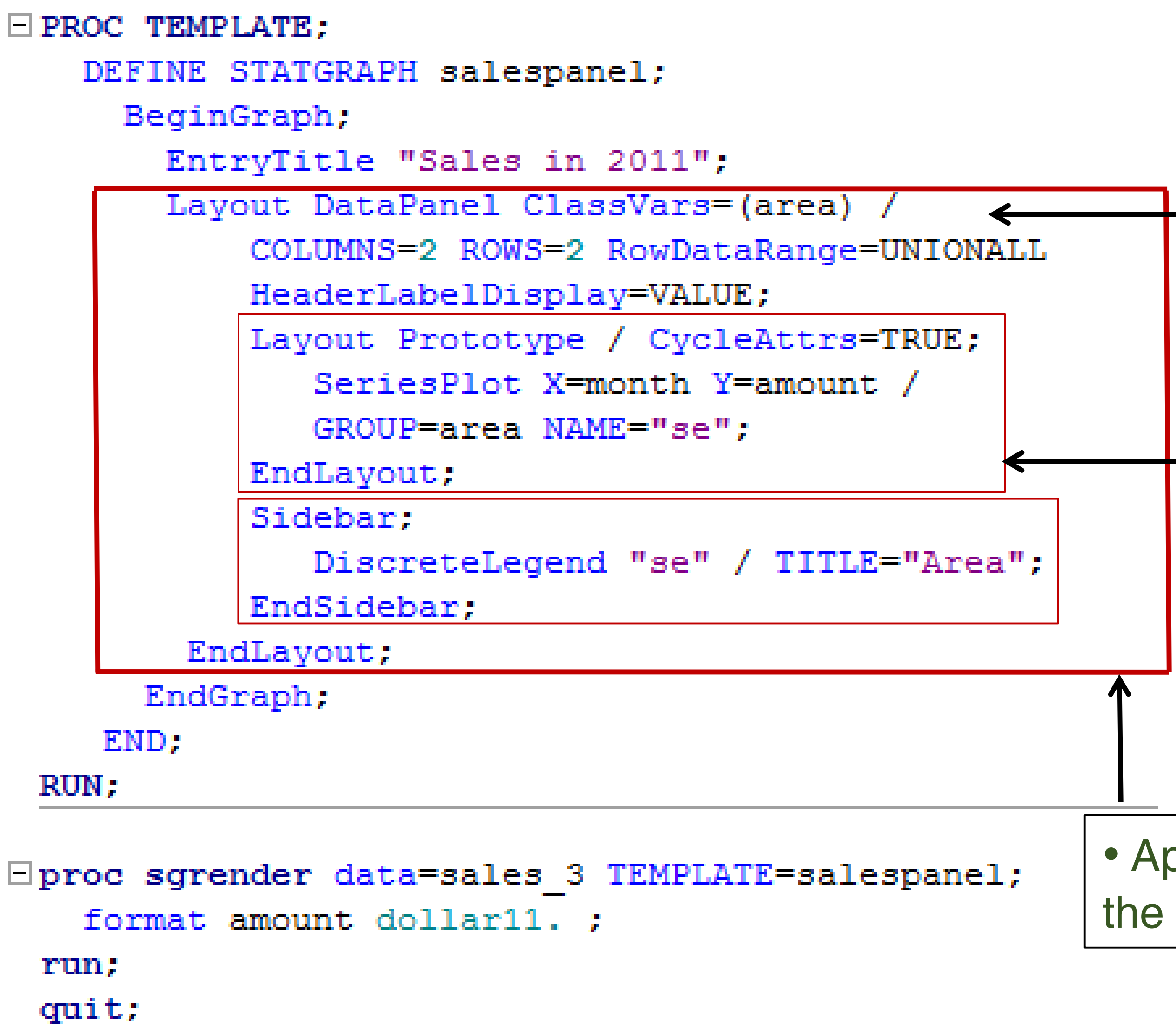
Use ‘REGION’ layout for PIE chart, as there is no AXIS

Draw pie chart. Each part of the pie represents each educational level, and area for each part stands for the mean income amount. The default ‘response’ is ‘count’, and default ‘stat’ is ‘sum’.

Example 5: Outcome



Example 5: Multiple Plots Using DataPanel



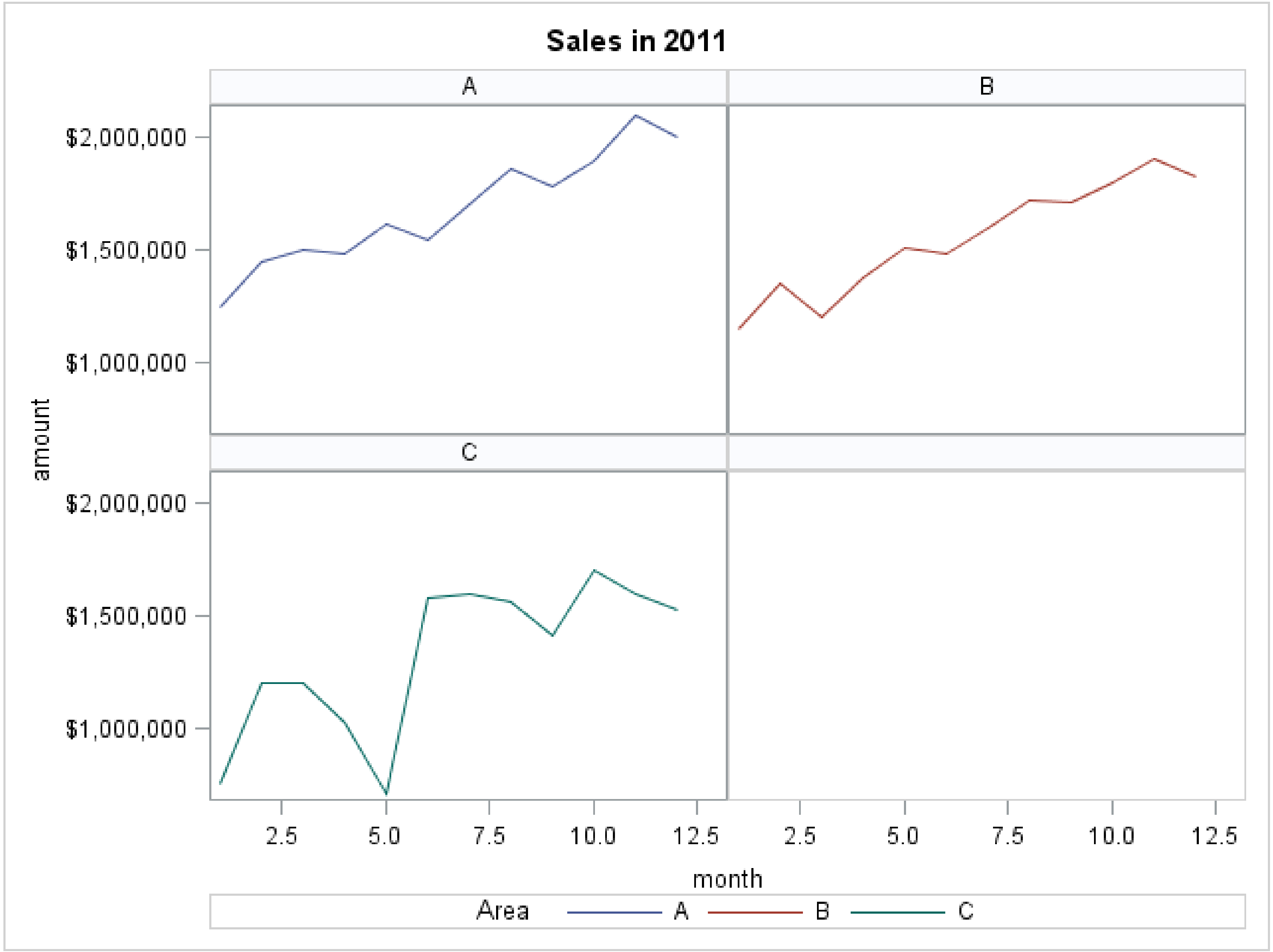
```
PROC TEMPLATE;
  DEFINE STATGRAPH salespanel;
    BeginGraph;
      EntryTitle "Sales in 2011";
      Layout DataPanel ClassVars=(area) /
          COLUMNS=2 ROWS=2 RowDataRange=UNIONALL
          HeaderLabelDisplay=VALUE;
          Layout Prototype / CycleAttrs=TRUE;
             SeriesPlot X=month Y=amount /
             GROUP=area NAME="se";
          EndLayout;
          Sidebar;
             DiscreteLegend "se" / TITLE="Area";
          EndSidebar;
       EndLayout;
     EndGraph;
   END;
RUN;

proc sgrender data=sales_3 TEMPLATE=salespanel;
   format amount dollar11. ;
run;
quit;
```

- Apply the 'DATAPANEL' layout, which requests 2 X 2 panels. The classification variable is 'area'.
- Each panel holds a series plot corresponding to the value of the categorical variable 'area'.
- The 'HeaderLabelDisplay=VALUE' means you are using the value of 'area' as the header of each panel.
- The 'RowDataRange=UNIONALL' means the same AXIS range is used in each panel.

- Use the layout 'PROTOTYPE' to set up each child layout (the parent's layout is 'DataPanel') containing plot statement.
- The layout statement will repeat each cell so that all graphs are produced.

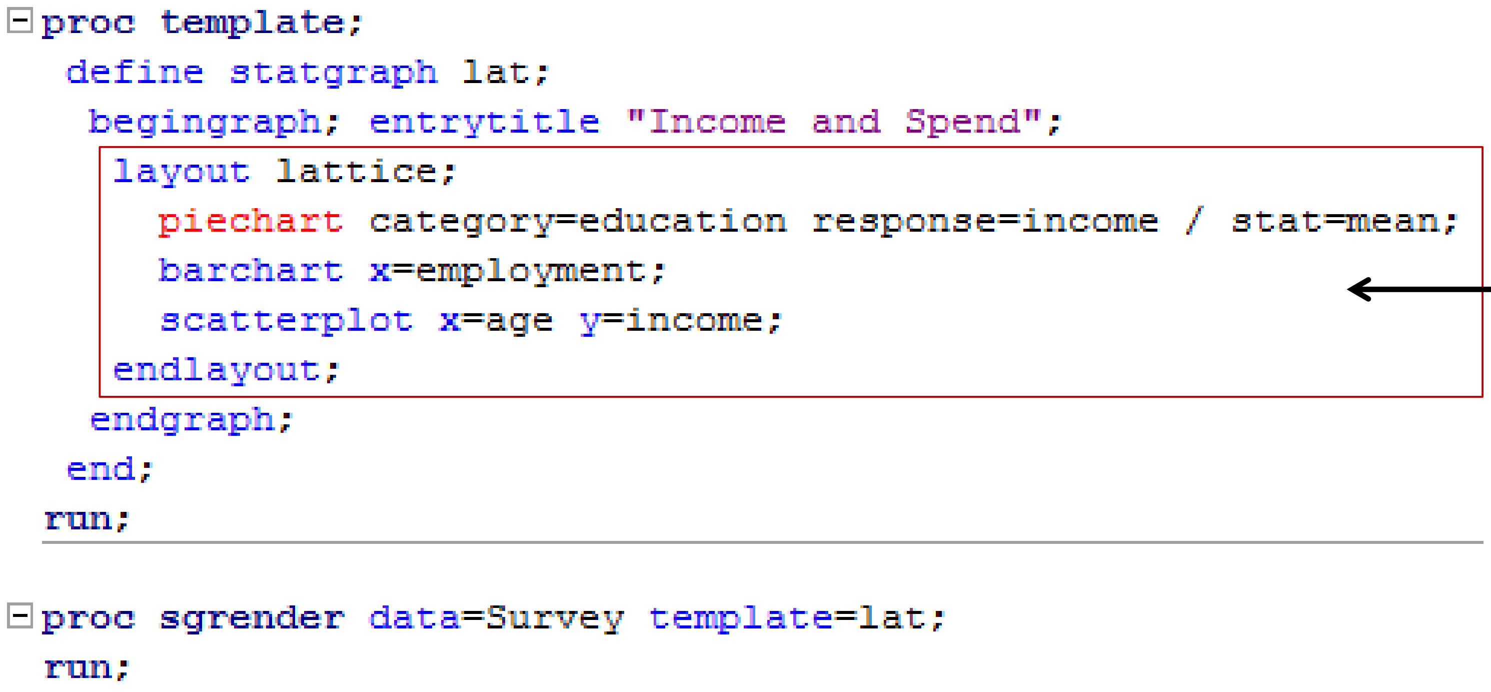
- Apply SIDEBAR statement to define the legend for the whole DataPanel

Example 5: Outcome



Example 6: Using LAYOUT LATTICE

The 'LATTICE' is more advanced layout to build graph in multiple panels. It can automatically adjust plot areas and tick areas. Therefore you do not have to predefine the number of panels (you must do so in 'DATAPANEL' layout.



```
proc template;
 define statgraph lat;
  begingraph; entrytitle "Income and Spend";
   layout lattice;
     piechart category=education response=income / stat=mean;
     barchart x=employment;
     scatterplot x=age y=income;
   endlayout;
  endgraph;
 end;
run;

proc sgrender data=Survey template=lat;
run;
```

- Use the layout 'LATTICE' to build multi-cell grade of graph.

- Produce two independent graphs (piechart, barcharts and scatter plot) separately in each cell.

Example 6: Setting Grid Dimensions

You can also define the grid dimension in 'LATTICE' layout, even though it has a default option (**order=rowmajor**)

```
layout lattice / order=rowmajor;
   /* plot definitions */
endlayout;
```

=

Default Option

```
layout lattice;
   /* plot definitions */
endlayout;
```

→ Put all plots in one column

```
layout lattice / columns=3 rows=2;
/* plot definitions */
endlayout;
```

→ To make explicit declaration about grid dimension

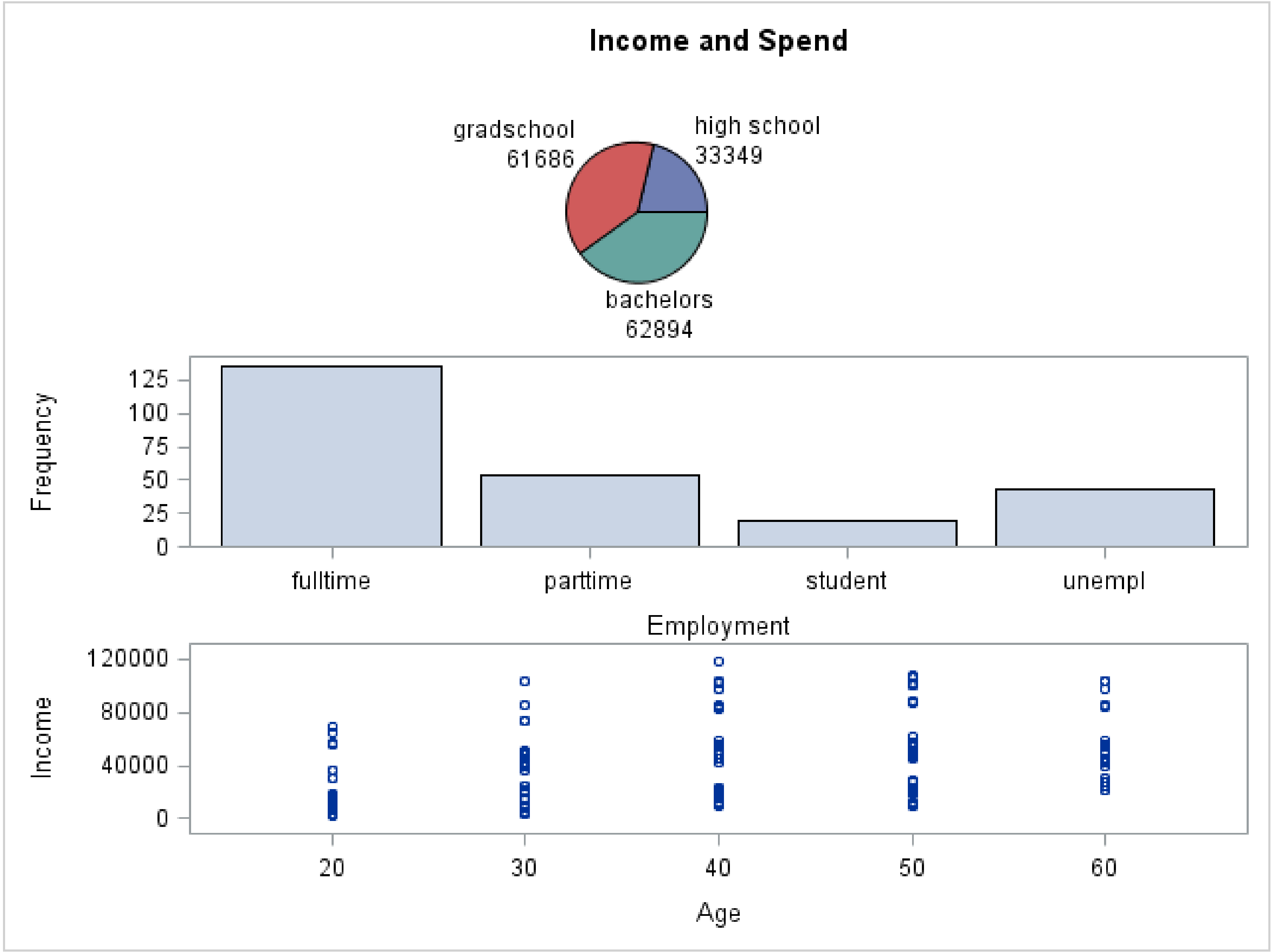
```
layout lattice / order=columnmajor;
/* plot definitions */
endlayout;
```

→ Put all plots in one row

```
layout lattice / columns=3;
/* plot definitions */
endlayout;
```

→ Set a maximum # (3 for here) of plots in each row, additional rows will be provided if you want to place more than 3

Example 6: Outcome

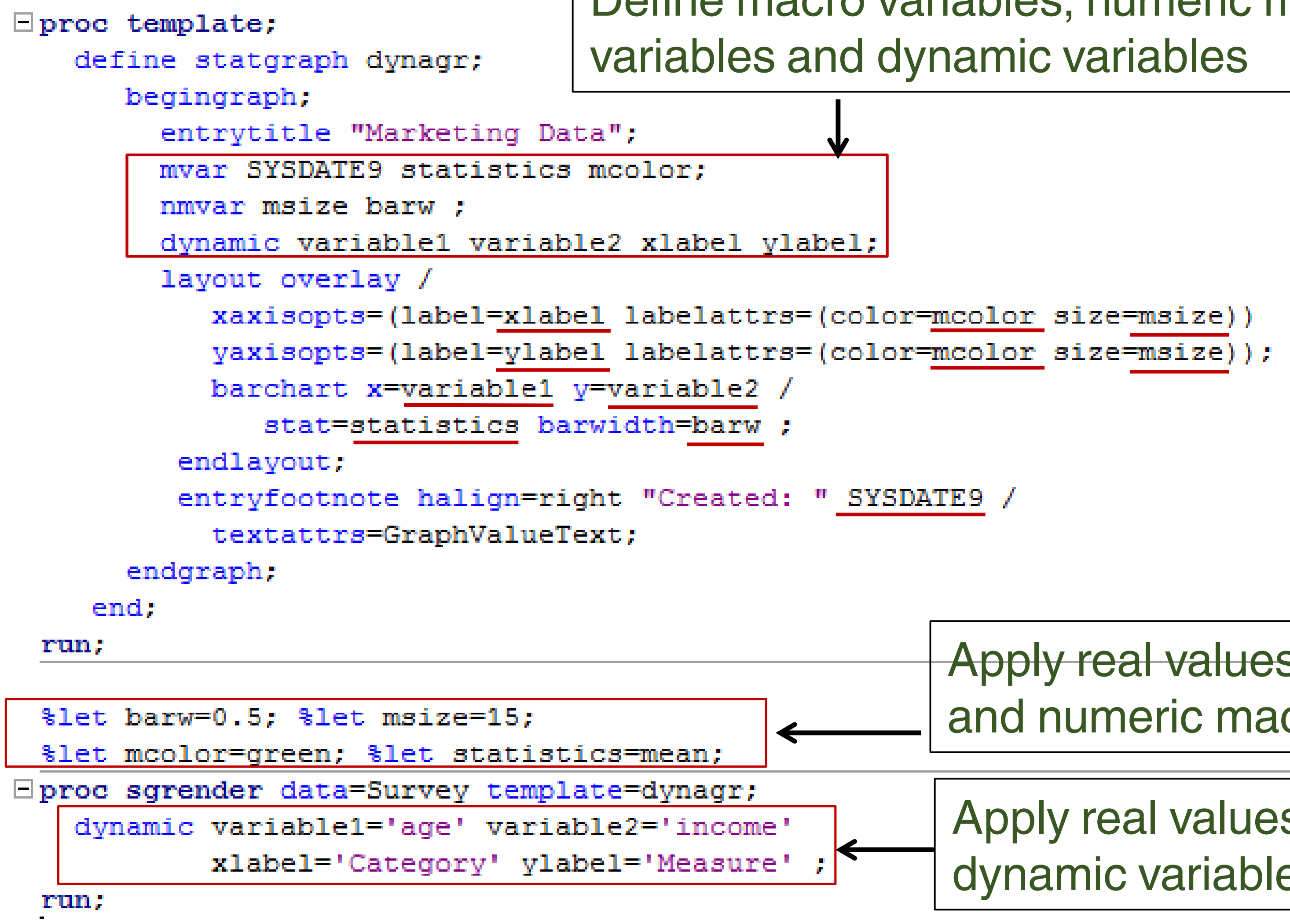


Dynamic Template

The dynamic template make your template more flexible by providing required arguments and option values.

Statement	Function
DYNAMIC	Set up dynamic template
MVAR	Define Macro Variables
NMVAR	Define Macro Variables resolving to a number

Example 7: Using Dynamic Template



```
proc template;
  define statgraph dynagr;
    begingraph;
      entrytitle "Marketing Data";
      mvar SYSDATE9 statistics mcolor;
      nmvar msize barw ;
      dynamic variable1 variable2 xlabel ylabel;
      layout overlay /
         xaxisopts=(label=xlabel labelattrs=(color=mcolor size=msize))
         yaxisopts=(label=ylabel labelattrs=(color=mcolor size=msize));
         barchart x=variable1 y=variable2 /
            stat=statistics barwidth=barw ;
       endlayout;
       entryfootnote halign=right "Created: " SYSDATE9 /
         textattrs=GraphValueText;
    endgraph;
   end;
run;

%let barw=0.5; %let msize=15;
%let mcolor=green; %let statistics=mean;
proc sgrender data=Survey template=dynagr;
  dynamic variable1='age' variable2='income'
          xlabel='Category' ylabel='Measure' ;
run;
```

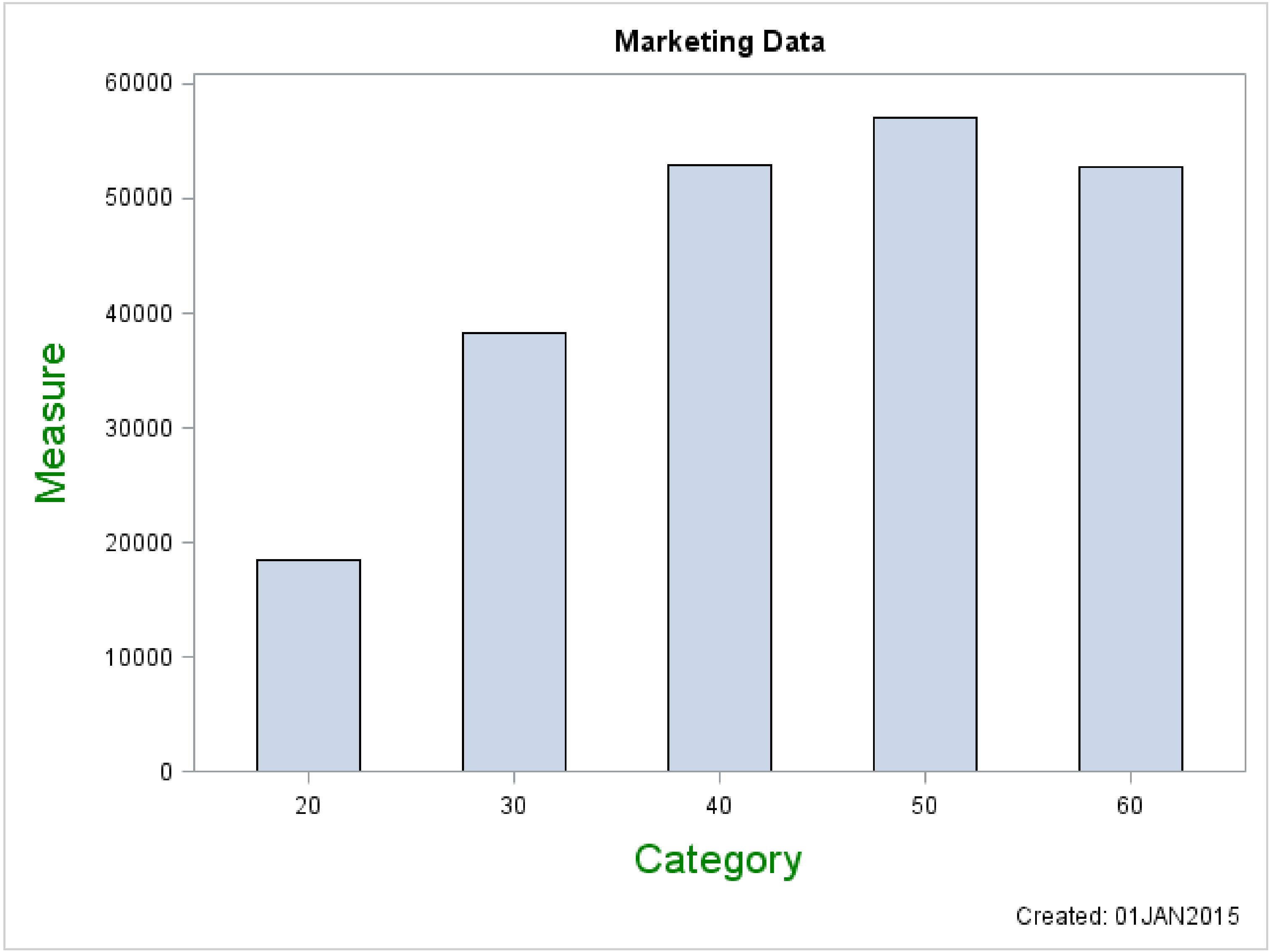
Define macro variables, numeric macro variables and dynamic variables

The difference between dynamics and macro variables is that they are initialized differently. The dynamic variables are initialized using DYNAMIC statement inside 'PROC SGRENDER'.

Apply real values to macro variable and numeric macro variables.

Apply real values to those dynamic variables.

Example 7: Outcome


Marketing Data
Measure
Category
Created: 01JAN2015