CHAPTER 26
The PLOT Procedure

Overview 692
Procedure Syntax 694
   PROC PLOT Statement 695
   BY Statement 698
   PLOT Statement 698
Concepts 710
   RUN Groups 710
   Generating Data with Program Statements 710
   Labeling Plot Points with Values of a Variable 711
      Pointer Symbols 711
      Understanding Penalties 712
      Changing Penalties 713
      Collision States 713
      Reference Lines 714
      Hidden Label Characters 714
      Overlaying Label Plots 714
      Computational Resources Used for Label Plots 714
      Time 714
      Memory 714
Results 715
   Scale of the Axes 715
   Printed Output 715
   Missing Values 715
   Hidden Observations 715
Examples 716
   Example 1: Specifying a Plotting Symbol 716
   Example 2: Controlling the Horizontal Axis and Adding a Reference Line 717
   Example 3: Overlaying Two Plots 718
   Example 4: Producing Multiple Plots per Page 719
   Example 5: Plotting Data on a Logarithmic Scale 721
   Example 6: Plotting Date Values on an Axis 723
   Example 7: Producing a Contour Plot 725
   Example 8: Plotting BY Groups 728
   Example 9: Adding Labels to a Plot 730
   Example 10: Excluding Observations That Have Missing Values 733
   Example 11: Adjusting Labels on a Plot with the PLACEMENT= Option 735
   Example 12: Adjusting Labeling on a Plot with a Macro 739
   Example 13: Changing a Default Penalty 741
The PLOT procedure plots the values of two variables for each observation in an input SAS data set. The coordinates of each point on the plot correspond to the two variables’ values in one or more observations of the input data set.

Output 26.1 on page 692 is a simple plot of the high values of the Dow Jones Industrial Average (DJIA) between 1954 and 1994. PROC PLOT determines the plotting symbol and the scales for the axes. These are the statements that produce the output:

```sas
options nodate pageno=1 linesize=64
   pagesize=25;

proc plot data=djia;
   plot high*year;
   title 'High Values of the Dow Jones';
   title2 'Industrial Average';
   title3 'from 1954 to 1994';
run;
```

Output 26.1  A Simple Plot

You can also overlay two plots, as shown in Output 26.2 on page 692. One plot shows the high values of the DJIA; the other plot shows the low values. The plot also shows that you can specify plotting symbols and put a box around a plot. The statements that produce Output 26.2 on page 692 are shown in Example 3 on page 718.
Output 26.2  Plotting Two Sets of Values at Once

PROC PLOT can also label points on a plot with the values of a variable, as shown in Output 26.3 on page 693. The data plotted represent population density and crime rates for selected U.S. states. The SAS code that produces Output 26.3 on page 693 is shown in Example 11 on page 735.
Procedure Syntax

**Requirement:** At least one PLOT statement is required.

**Tip:** Supports RUN–group processing

**Tip:** Supports the Output Delivery System (see Chapter 2, “Fundamental Concepts for Using Base SAS Procedures”)

**Reminder:** You can use the ATTRIB, FORMAT, LABEL, and WHERE statements. See Chapter 3, "Statements with the Same Function in Multiple Procedures," for details. You can also use any global statements as well. See Chapter 2, "Fundamental Concepts for Using Base SAS Procedures," for a list.

```
PROC PLOT <option(s)>;
   BY <DESCENDING> variable-1
      <...<DESCENDING> variable-n>
      <NOTSORTED>;
   PLOT plot-request(s) </ option(s)>;
```
The PLOT Procedure

PROC PLOT Statement


**PROC PLOT <option(s)>;**

<table>
<thead>
<tr>
<th>To do this</th>
<th>Use this option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Produce a separate plot for each BY group</td>
<td>BY</td>
</tr>
<tr>
<td>Describe the plots you want</td>
<td>PLOT</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>To do this</th>
<th>Use this option</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify the input data set</td>
<td>DATA=</td>
</tr>
<tr>
<td>Control the axes</td>
<td></td>
</tr>
<tr>
<td>Include missing character variable values</td>
<td>MISSING</td>
</tr>
<tr>
<td>Exclude observations with missing values</td>
<td>NOMISS</td>
</tr>
<tr>
<td>Uniformly scale axes across BY groups</td>
<td>UNIFORM</td>
</tr>
<tr>
<td>Control the appearance of the plot</td>
<td></td>
</tr>
<tr>
<td>Specify the characters that construct the borders of the plot</td>
<td>FORMCHAR=</td>
</tr>
<tr>
<td>Suppress the legend at the top of the plot</td>
<td>NOLEGEND</td>
</tr>
<tr>
<td>Specify the aspect ratio of the characters on the output device</td>
<td>VTOH=</td>
</tr>
<tr>
<td>Control the size of the plot</td>
<td></td>
</tr>
<tr>
<td>Specify the percentage of the available horizontal space for each plot</td>
<td>HPERCENT=</td>
</tr>
<tr>
<td>Specify the percentage of the available vertical space for each plot</td>
<td>VPERCENT=</td>
</tr>
</tbody>
</table>

**Options**

**DATA=SAS-data-set**
specifies the input SAS data set.


**FORMCHAR <(position(s))>=’formatting-character(s)’**
defines the characters to use for constructing the borders of the plot.

*position(s)*
identifies the position of one or more characters in the SAS formatting-character string. A space or a comma separates the positions.
Default: Omitting (position(s)), is the same as specifying all twenty possible SAS formatting characters, in order.

Range: PROC PLOT uses formatting characters 1, 2, 3, 5, 7, 9, and 11. The following table shows the formatting characters that PROC PLOT uses.

<table>
<thead>
<tr>
<th>Position</th>
<th>Default</th>
<th>Used to draw</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>vertical separators</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>horizontal separators</td>
</tr>
<tr>
<td>3 5 9 1 1</td>
<td>-</td>
<td>corners</td>
</tr>
<tr>
<td>7</td>
<td>+</td>
<td>intersection of vertical and horizontal separators</td>
</tr>
</tbody>
</table>

formatting-character(s) lists the characters to use for the specified positions. PROC PLOT assigns characters in formatting-character(s) to position(s), in the order that they are listed. For instance, the following option assigns the asterisk (*) to the third formatting character, the pound sign (#) to the seventh character, and does not alter the remaining characters:

formchar(3,7)='*#'

Interaction: The SAS system option FORMCHAR= specifies the default formatting characters. The system option defines the entire string of formatting characters. The FORMCHAR= option in a procedure can redefine selected characters.

Tip: You can use any character in formatting-characters, including hexadecimal characters. If you use hexadecimal characters, you must put an x after the closing quote. For instance the following option assigns the hexadecimal character 2D to the third formatting character, the hexadecimal character 7C to the seventh character, and does not alter the remaining characters:

formchar(3,7)='2D7C'x

Tip: Specifying all blanks for formatting-character(s) produces plots with no borders, for example

formchar(1,2,7)=''

HPERCENT=percent(s) specifies one or more percentages of the available horizontal space to use for each plot. HPERCENT= enables you to put multiple plots on one page. PROC PLOT tries to fit as many plots as possible on a page. After using each of the percent(s), PROC PLOT cycles back to the beginning of the list. A zero in the list forces PROC PLOT to go to a new page even though it could fit the next plot on the same page.

hpercent=33
prints three plots per page horizontally, each plot is one-third of a page wide.

hpercent=50 25 25
prints three plots per page, the first is twice as wide as the other two.

hpercent=33 0
produces plots that are one-third of a page wide, each plot is on a separate page.

hpercent=300
produces plots three pages wide.
At the beginning of every BY group and after each RUN statement, PROC PLOT returns to the beginning of the percent(s) and starts printing a new page.
Alias:  HPCT=
Default:  100
Featured in:  Example 4 on page 719

MISSING
includes missing character variable values in the construction of the axes. It has no
effect on numeric variables.
Interaction:  overrides the NOMISS option for character variables

NOLEGEND
suppresses the legend at the top of each plot. The legend lists the names of the
variables being plotted and the plotting symbols used in the plot.

NOMISS
excludes observations for which either variable is missing from the calculation of the
axes. Normally, PROC PLOT draws an axis based on all the values of the variable
being plotted, including points for which the other variable is missing.
Interaction:  The HAXIS= option overrides the effect of NOMISS on the horizontal
axis. The VAXIS= option overrides the effect on the vertical axis.
Interaction:  NOMISS is overridden by MISSING for character variables.
Featured in:  Example 10 on page 733

UNIFORM
uniformly scales axes across BY groups. Uniform scaling allows you to directly
compare the plots for different values of the BY variables.
Restriction:  You cannot use PROC PLOT with the UNIFORM option with an
engine that supports concurrent access if another user is updating the data set at
the same time.

VPERCENT=percent(s)
specifies one or more percentages of the available vertical space to use for each plot.
If you use a percentage greater than 100, PROC PLOT prints sections of the plot on
successive pages.
Alias:  VPCT=
Default:  100
Featured in:  Example 4 on page 719
See also:  HPERCENT= on page 696

VTOH=aspect-ratio
specifies the aspect ratio (vertical to horizontal) of the characters on the output
device. aspect-ratio is a positive real number. If you use the VTOH= option, PROC
PLOT spaces tick marks so that the distance between horizontal tick marks is nearly
equal to the distance between vertical tick marks. For example, if characters are
twice as high as wide, specify VTOH=2.
Minimum:  0
Interaction:  VTOH= has no effect if you use the HSPACE= and the VSPACE=
options in the PLOT statement.
See also:  HAXIS= on page 702 for a way to equate axes so that the given distance
represents the same data range on both axes.
BY Statement

Produces a separate plot and starts a new page for each BY group.

Main discussion: “BY” on page 68
Featured in: Example 8 on page 728

BY <DESCENDING> variable-1
    <...<DESCENDING> variable-n>
    <NOTSORTED>;

Required Arguments

variable
    specifies the variable that the procedure uses to form BY groups. You can specify
    more than one variable. If you do not use the NOTSORTED option in the BY
    statement, the observations in the data set must either be sorted by all the variables
    that you specify, or they must be indexed appropriately. Variables in a BY statement
    are called BY variables.

Options

DESCENDING
    specifies that the observations are sorted in descending order by the variable that
    immediately follows the word DESCENDING in the BY statement.

NOTSORTED
    specifies that observations are not necessarily sorted in alphabetic or numeric order.
    The data are grouped in another way, for example, chronological order.
    The requirement for ordering or indexing observations according to the values of
    BY variables is suspended for BY-group processing when you use the NOTSORTED
    option. In fact, the procedure does not use an index if you specify NOTSORTED. The
    procedure defines a BY group as a set of contiguous observations that have the same
    values for all BY variables. If observations with the same values for the BY variables
    are not contiguous, the procedure treats each contiguous set as a separate BY group.

PLOT Statement

Requests the plots to be produced by PROC PLOT.

Tip: You can use multiple PLOT statements.

PLOT plot-request(s) <\ option(s)>;
To do this | Use this option
---|---
Control the axes | 
Specify the tick-mark values | HAXIS= and VAXIS=  
Expand the axis | HEXPAND and VEXPAND  
Specify the number of print positions | HPOS= and VPOS=  
Reverse the order of the values | HREVERSE and VREVERSE  
Specify the number of print positions between tick marks | HSPACE= and VSPACE=  
Assign a value of zero to the first tick mark | HZERO and VZERO  
Specify reference lines | 
Draw a line perpendicular to the specified values on the axis | HREF= and VREF=  
Specify a character to use to draw the reference line | HREFCHAR= and VREFCHAR=  
Put a box around the plot | BOX  
Overlay plots | OVERLAY  
Produce a contour plot | 
Draw a contour plot | CONTOUR  
Specify the plotting symbol for one contour level | Scontour-level=  
Specify the plotting symbol for multiple contour levels | SLIST=  
Label points on a plot | 
List the penalty and the placement state of the points | LIST=  
Force the labels away from the origin | OUTWARD=  
Change default penalties | PENALTIES=  
Specify locations for the placement of the labels | PLACEMENT=  
Specify a split character for the label | SPLIT=  
List all placement states in effect | STATES  

**Required Arguments**
**plot-request(s)**

specifies the variables (vertical and horizontal) to plot and the plotting symbol to use to mark the points on the plot.

Each form of `plot-request(s)` supports a label variable. A label variable is preceded by a dollar sign ($) and specifies a variable whose values label the points on the plot. For example,

```plaintext
plot y*x $ label-variable
plot y*x='*' $ label-variable
```

See “Labeling Plot Points with Values of a Variable” on page 711 for more information. In addition, see Example 9 on page 730 and all the examples that follow it.

The `plot-request(s)` can be one or more of the following:

- `vertical*horizontal <$ label-variable>`

  specifies the variable to plot on the vertical axis and the variable to plot on the horizontal axis.

  For example, the following statement requests a plot of Y by X:

  ```plaintext
  plot y*x;
  ```

  Y appears on the vertical axis, X on the horizontal axis.

  This form of the plot request uses the default method of choosing a plotting symbol to mark plot points. When a point on the plot represents the values of one observation in the data set, PROC PLOT puts the character A at that point. When a point represents the values of two observations, the character B appears. When a point represents values of three observations, the character C appears, and so on through the alphabet. The character Z is used for the occurrence of 26 or more observations at the same printing position.

- `vertical*horizontal='character' <$ label-variable>`

  specifies the variables to plot on the vertical and horizontal axes and specifies a plotting symbol to mark each point on the plot. A single character is used to represent values from one or more observations.

  For example, the following statement requests a plot of Y by X, with each point on the plot represented by a plus sign (+):

  ```plaintext
  plot y*x='+';
  ```

- `vertical*horizontal=variable <$ label-variable>`

  specifies the variables to plot on the vertical and horizontal axes and specifies a variable whose values are to mark each point on the plot. The variable can be either numeric or character. The first (left-most) nonblank character in the formatted value of the variable is used as the plotting symbol (even if more than one value starts with the same letter). When more than one observation maps to the same plotting position, the value from the first observation marks the point.

  For example, in the following statement `GENDER` is a character variable with values of `FEMALE` and `MALE`:

  ```plaintext
  plot height*weight=gender;
  ```

**Specifying Variable Lists in Plot Requests**

You can use SAS variable lists in plot requests. For example, the following are valid plot requests:
The PLOT Procedure

The PLOT Procedure

PLOT Statement

Plot request What is plotted
(a - - d) a*b a*c a*d b*c b*d c*d
(x1 - x4) x1*x2 x1*x3 x1*x4 x2*x3 x2*x4 x3*x4
(_numeric_) All combinations of numeric variables
y*(x1 - x4) y*x1 y*x2 y*x4 y*x4

If both the vertical and horizontal specifications request more than one variable and a variable appears in both lists, it will not be plotted against itself. For example, the following statement does not plot B*B and C*C:

```
plot (a b c)*(b c d);
```

**Specifying Combinations of Variables**

The operator in request is either an asterisk (*) or a colon (:). An asterisk combines the variables in the lists to produce all possible combinations of x and y variables. For example, the following plot requests are equivalent:

```
plot (y1-y2) * (x1-x2);
```

```
plot y1*x1 y1*x2 y2*x1 y2*x2;
```

A colon combines the variables pairwise. Thus, the first variables of each list combine to request a plot, as do the second, third, and so on. For example, the following plot requests are equivalent:

```
plot (y1-y2) : (x1-x2);
```

```
plot y1*x1 y2*x2;
```

**Options**

**BOX**

draws a border around the entire plot, rather than just on the left side and bottom.

**Featured in:** Example 3 on page 718

**CONTOUR<=number-of-levels>>**

draws a contour plot using plotting symbols with varying degrees of shading where number-of-levels is the number of levels for dividing the range of variable. The plot request must be of the form vertical*horizontal=variable where variable is a numeric variable in the data set. The intensity of shading is determined by the values of this variable.

When you use CONTOUR, PROC PLOT does not plot observations with missing values for variable.

Overprinting, if it is allowed by the OVP system option, is used to produce the shading. Otherwise, single characters varying in darkness are used. The CONTOUR option is most effective when the plot is dense.

**Default:** 10

**Range:** 1-10
Featured in: Example 7 on page 725

**HAXIS=axis-specification**
specifies the tick-mark values for the horizontal axis.

- For numeric values, *axis-specification* is either an explicit list of values, a BY increment, or a combination of both:
  
  \[
  n <...n> \\
  \text{BY increment} \\
  n \text{ TO } n \text{ BY increment}
  \]

  The values must be in either ascending or descending order. Use a negative value for *increment* to specify descending order. The specified values are spaced evenly along the horizontal axis even if the values are not uniformly distributed. Numeric values can be specified in the following ways:

<table>
<thead>
<tr>
<th>HAXIS= value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 to 100 by 5</td>
<td>Values appear in increments of 5, starting at 10 and ending at 100.</td>
</tr>
<tr>
<td>by 5</td>
<td>Values are incremented by 5. PROC PLOT determines the minimum and maximum values for the tick marks.</td>
</tr>
<tr>
<td>10 100 1000 10000</td>
<td>Values are not uniformly distributed. This specification produces a logarithmic plot. If PROC PLOT cannot determine the function implied by the axis specification, it uses simple linear interpolation between the points. To determine whether PROC PLOT correctly interpolates a function, you can use the DATA step to generate data that determines the function and see whether it appears linear when plotted. See Example 5 on page 721 for an example.</td>
</tr>
<tr>
<td>1 2 10 to 100 by 5</td>
<td>A combination of the previous specifications.</td>
</tr>
</tbody>
</table>

- For character variables, *axis-specification* is a list of unique values that are enclosed in quotes:

  \[
  \text{'value-1' <...'value-n'>}
  \]

  For example,

  \[
  \text{haxis='Paris' 'London' 'Tokyo'}
  \]

  The character strings are case-sensitive. If a character variable has an associated format, *axis-specification* must specify the formatted value. The values can appear in any order.

- For axis variables that contain date-time values, *axis-specification* is either an explicit list of values or a starting and an ending value with an increment specified:

  \[
  \text{'date-time-value'i <...'date-time-value'i>}
  \]
'date-time-value'i TO <...'date-time-value'i>  
<BY increment>

'date-time-value'i
any SAS date, time, or datetime value described for the SAS functions  
INTCK and INTNX. The suffix i is one of the following:

- D date
- T time
- DT datetime

increment
one of the valid arguments for the INTCK or INTNX functions: For dates,  
increment can be one of the following:

- DAY
- WEEK
- MONTH
- QTR
- YEAR

For datetimes, increment can be one of the following:

- DTDAY
- DTWEEK
- DTMONTH
- DTQTR
- DTYEAR

For times, increment can be one of the following:

- HOUR
- MINUTE
- SECOND

For example,

```plaintext
haxis='01JAN95'd to '01JAN96'd  
by month
```

```plaintext
haxis='01JAN95'd to '01JAN96'd  
by qtr
```

**Note:** You must use a FORMAT statement to print the tick-mark values  
in an understandable form.

**Interaction:** You can use the HAXIS= and VAXIS= options with the VTOH= option  
to equate axes. If your data are suitable, use HAXIS=BY n and VAXIS=BY n with  
the same value for n and specify a value for the VTOH= option. The number of  
columns separating the horizontal tick marks is nearly equal to the number of  
lines separating the vertical tick marks times the value of the VTOH= option. In  
some cases, PROC PLOT cannot simultaneously use all three values and changes  
one or more of the values.

**Featured in:** Example 2 on page 717, Example 5 on page 721, and Example 6 on  
page 723

**HEXPAND**
expands the horizontal axis to minimize the margins at the sides of the plot and to  
maximize the distance between tick marks, if possible.

HEXPAND causes PROC PLOT to ignore information about the spacing of the  
data. Plots produced with this option waste less space but may obscure the nature of  
the relationship between the variables.
HPOS=axis-length
specifies the number of print positions on the horizontal axis. The maximum value of axis-length that allows a plot to fit on one page is three positions less than the value of the LINESIZE= system option because there must be space for the procedure to print information next to the vertical axis. The exact maximum depends on the number of characters in the vertical variable’s values. If axis-length is too large to fit on a line, PROC PLOT ignores the option.

HREF=value-specification
draws lines on the plot perpendicular to the specified values on the horizontal axis. PROC PLOT includes the values you specify with the HREF= option on the horizontal axis unless you specify otherwise with the HAXIS= option. For the syntax for value-specification, see HAXIS= on page 702.

Featured in: Example 8 on page 728

HREFCHAR='character'
specifies the character to use to draw the horizontal reference line.
Default: vertical bar (|)
See also: FORMCHAR= option on page 695 and HREF= on page 704

HREVERSE
reverses the order of the values on the horizontal axis.

HSPACE=n
specifies that a tick mark will occur on the horizontal axis at every nth print position, where n is the value of HSPACE=.

HZERO
assigns a value of zero to the first tick mark on the horizontal axis.
Interaction: PROC PLOT ignores HZERO if the horizontal variable has negative values or if the HAXIS= option specifies a range that does not begin with zero.

LIST<=penalty-value>
lists the horizontal and vertical axis values, the penalty, and the placement state of all points plotted with a penalty greater than or equal to penalty-value.
Tip: LIST is equivalent to LIST=0.
See also: “Understanding Penalties” on page 712
Featured in: Example 11 on page 735

OUTWARD='character'
tries to force the point labels outward, away from the origin of the plot, by protecting positions next to symbols that match character that are in the direction of the origin (0,0). The algorithm tries to avoid putting the labels in the protected positions, so they usually move outward.
Tip: This option is useful only when you are labeling points with the values of a variable.

OVERLAY
overlays all plots specified in the PLOT statement on one set of axes. The variable names, or variable labels if they exist, from the first plot are used to label the axes. Unless you use the HAXIS= or the VAXIS= option, PROC PLOT automatically scales the axes in the way that best fits all the variables.

When the SAS system option OVP is in effect and overprinting is allowed, the plots are superimposed; otherwise, when NOOVP is in effect, PROC PLOT uses the plotting symbol from the first plot to represent points appearing in more than one plot. In such a case, the output includes a message telling you how many observations are hidden.
PENALTIES<(index-list)>=penalty-list
changes the default penalties. The index-list provides the positions of the penalties in the list of penalties. The penalty-list contains the values you are specifying for the penalties indicated in the index-list. The index-list and the penalty-list can contain one or more integers. In addition, both index-list and penalty-list can contain

value TO value

See also: “Understanding Penalties” on page 712
Featured in: Example 13 on page 741

PLACEMENT=(expression(s))
controls the placement of labels by specifying possible locations of the labels relative to their coordinates. Each expression consists of a list of one or more suboptions (H=, L=, S=, or V=) that are joined by an asterisk or a colon. PROC PLOT uses the asterisk and colon to expand each expression into combinations of values for the four possible suboptions. The asterisk creates every possible combination of values in the expression list. A colon creates only pairwise combinations. The colon takes precedence over the asterisk. With the colon, if one list is shorter than the other, the values in the shorter list are reused as necessary.

Use the following suboptions to control the placement:

H=integer(s)
specifies the number of horizontal spaces (columns) to shift the label relative to the starting position. Both positive and negative integers are valid. Positive integers shift the label to the right; negative integers shift it to the left. For example, you can use the H= suboption in the following way:

place=(h=0 1 -1 2 -2)

You can use the keywords BY ALT in this list. BY ALT produces a series of numbers whose signs alternate between positive and negative and whose absolute values change by one after each pair. For instance, the following PLACE= specifications are equivalent:

place=(h=0 -1 to -3 by alt)

place=(h=0 -1 1 -2 2 -3 3)

If the series includes zero, the zero appears twice. For example, the following PLACE= options are equivalent:

place=(h= 0 to 2 by alt)

place=(h=0 0 1 -1 2 -2)

Default: H=0
Range: −500 to 500

L=integer(s)
specifies the number of lines onto which the label may be split.

Default: L=1
Range: 1-200

S=start-position(s)
specifies where to start printing the label. The value for start-position can be one or more of the following
CENTER
the procedure centers the label around the plotting symbol.

RIGHT
the label starts at the plotting symbol location and continues to the right.

LEFT
the label starts to the left of the plotting symbol and ends at the plotting symbol location.

Default: CENTER

V=integer(s)
specifies the number of vertical spaces (lines) to shift the label relative to the starting position. V= behaves the same as the H= suboption, described earlier.

A new expression begins when a suboption is not preceded by an operator. Parentheses around each expression are optional. They make it easier to recognize individual expressions in the list. However, the entire expression list must be in parentheses, as shown in the following example. Table 26.1 on page 707 shows how this expression is expanded and describes each placement state.

place=((v=1)
   (s=right left : h=2 -2)
   (v=-1)
   (h=0 1 to 2 by alt * v=1 -1)
   (l=1 to 3 * v=1 to 2 by alt *
   h=0 1 to 2 by alt))

Each combination of values is a placement state. The procedure uses the placement states in the order in which they appear in the placement states list, so specify your most preferred placements first. For each label, the procedure tries all states, then uses the first state that places the label with minimum penalty. Once all labels are initially placed, the procedure cycles through the plot multiple times, systematically refining the placements. The refinement step tries to both minimize the penalties and to use placements nearer to the beginning of the states list. However, PROC PLOT uses a heuristic approach for placements, so the procedure does not always find the best set of placements.

Alias: PLACE=

Defaults: There are two defaults for the PLACE= option. If you are using a blank as your plotting symbol, the default placement state is PLACE=(S=CENTER : V=0 : H=0 : L=1), which centers the label. If you are using anything other than a blank, the default is PLACE=((S=RIGHT LEFT : H=2 -2) (V=1 -1 * H=0 1 -1 2 -2)). The default for labels placed with symbols includes multiple positions around the plotting symbol so the procedure has flexibility when placing labels on a crowded plot.

Tip: Use the STATES option to print a list of placement states.

See also: “Labeling Plot Points with Values of a Variable” on page 711

Featured in: Example 11 on page 735 and Example 12 on page 739
### Table 26.1 Expanding an Expression List into Placement States

<table>
<thead>
<tr>
<th>Expression</th>
<th>Placement state</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(V=1)</td>
<td>S=CENTRE L=1 H=0 V=1</td>
<td>Center the label, relative to the point, on the line above the point. Use one line for the label.</td>
</tr>
<tr>
<td>(S=RIGHT LEFT : H=2 -2)</td>
<td>S=RIGHT L=1 H=2 V=0</td>
<td>Begin the label in the second column to the right of the point. Use one line for the label.</td>
</tr>
<tr>
<td></td>
<td>S=LEFT L=1 H=2 V=0</td>
<td>End the label in the second column to the left of the point. Use one line for the label.</td>
</tr>
<tr>
<td>(V=-1)</td>
<td>S=CENTRE L=1 H=0 V=1</td>
<td>Center the label, relative to the point, on the line below the point. Use one line for the label.</td>
</tr>
<tr>
<td>(H=0 1 to 2 BY ALT * V=1 -1)</td>
<td>S=CENTRE L=1 H=0 V=1</td>
<td>Center the label, relative to the point, on the line above the point.</td>
</tr>
<tr>
<td></td>
<td>S=CENTRE L=1 H=0 V=-1</td>
<td>Center the label, relative to the point, on the line below the point.</td>
</tr>
<tr>
<td></td>
<td>S=CENTRE L=1 H=1 V=1</td>
<td>From center, shift the label one column to the right on the line above the point.</td>
</tr>
<tr>
<td></td>
<td>S=CENTRE L=1 H=1 V=-1</td>
<td>From center, shift the label one column to the right on the line below the point.</td>
</tr>
<tr>
<td></td>
<td>S=CENTRE L=1 H=-1 V=1</td>
<td>From center, shift the label one column to the left on the line above the point.</td>
</tr>
<tr>
<td></td>
<td>S=CENTRE L=1 H=-1 V=-1</td>
<td>From center, shift the label one column to the left on the line below the point.</td>
</tr>
<tr>
<td></td>
<td>S=CENTRE L=1 H=2 V=1</td>
<td>From center, shift the labels two columns to the right, first on the line above the point, then on the line below.</td>
</tr>
<tr>
<td></td>
<td>S=CENTRE L=1 H=2 V=-1</td>
<td>From center, shift the labels two columns to the left, first on the line above the point, then on the line below.</td>
</tr>
<tr>
<td>(L=1 to 3 * V=1 to 2 BY ALT * H=0 1 to 2 BY ALT)</td>
<td>S=CENTRE L=1 H=0 V=1</td>
<td>Center the label, relative to the point, on the line above the point. Use one line for the label.</td>
</tr>
</tbody>
</table>
### PLOT Statement

Chapter 26

<table>
<thead>
<tr>
<th>Expression</th>
<th>Placement state</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>S=CENTER L=1 H=1 V=1</td>
<td>From center, shift the label one or two columns to the right or left on the line above the point. Use one line for the label.</td>
<td></td>
</tr>
<tr>
<td>S=CENTER L=1 H=0 V=-1</td>
<td>Center the label, relative to the point, on the line below the point. Use one line for the label.</td>
<td></td>
</tr>
<tr>
<td>S=CENTER L=1 H=1 V=-1</td>
<td>From center, shift the label one or two columns to the right and the left on the line below the point.</td>
<td></td>
</tr>
<tr>
<td>S=CENTER L=1 H=-2 V=-1</td>
<td>Use the same horizontal shifts on the line two lines above the point and on the line two lines below the point.</td>
<td></td>
</tr>
<tr>
<td>S=CENTER L=1 H=-2 V=-2</td>
<td>Repeat the whole process splitting the label over two lines. Then repeat it splitting the label over three lines.</td>
<td></td>
</tr>
<tr>
<td>S=CENTER L=2 H=0 V=1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S=CENTER L=3 H=-2 V=-2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Scontour-level=’character-list’**

specifies the plotting symbol to use for a single contour level. When PROC PLOT produces contour plots, it automatically chooses the symbols to use for each level of intensity. You can use the S= option to override these symbols and specify your own. You can include up to three characters in `character-list`. If overprinting is not allowed, PROC PLOT uses only the first character.

For example, to specify three levels of shading for the Z variable, use the following statement:

```plaintext
plot y*x=z / contour=3 s1='A' s2='+' s3='X0A';
```

You can also specify the plotting symbols as hexadecimal constants:

```plaintext
plot y*x=z / contour=3 s1='7A'x s2='7F'x s3='A6'x;
```

This feature was designed especially for printers where the hex constants can represent grey-scale fill characters.

**Range:** 1 to the highest contour level (determined by the CONTOUR option).

**See also:** SLIST= and CONTOUR
The PLOT Procedure △ PLOT Statement 709

SLIST='character-list-1' <...'character-list-n'>
specifies plotting symbols for multiple contour levels. Each character-list specifies the plotting symbol for one contour level: the first character-list for the first level, the second character-list for the second level, and so on. For example:

```plaintext
plot y*x=z /
    contour=5 slist='.' ':' '! '=' '+' 'O';
```

**Default:** If you omit a plotting symbol for each contour level, PROC PLOT uses the default symbols:

```plaintext
slist='.' ',' '-' '=' '+' 'O' 'X'
```

**Restriction:** If you use the SLIST= option, it must be listed last in the PLOT statement.

**See also:** Scontour-level= and CONTOUR=

SPLIT='split-character'
when labeling plot points, specifies where to split the label when the label spans two or more lines. The label is split onto the number of lines specified in the L= suboption to the PLACEMENT= option. If you specify a split character, the procedure always splits the label on each occurrence of that character, even if it cannot find a suitable placement. If you specify L=2 or more but do not specify a split character, the procedure tries to split the label on blanks or punctuation but will split words if necessary.

PROC PLOT shifts split labels as a block, not as individual fragments (a fragment is the part of the split label that is contained on one line). For example, to force This is a label to split after the a, change it to This is a*label and specify SPLIT=’*’.

**See also:** “Labeling Plot Points with Values of a Variable” on page 711

STATES
lists all the placement states in effect. STATES prints the placement states in the order that you specify them in the PLACE= option.

VAXIS=axis-specification
specifies tick mark values for the vertical axis. VAXIS= follows the same rules as theHAXIS= option on page 702.

**Featured in:** Example 7 on page 725 and Example 12 on page 739

VEXPAND
expands the vertical axis to minimize the margins above and below the plot and to maximize the space between vertical tick marks, if possible.

**See also:** HEXPAND on page 703

VPOS=axis-length
specifies the number of print positions on the vertical axis. The maximum value for axis-length that allows a plot to fit on one page is 8 lines less than the value of the SAS system option PAGESIZE= because you must allow room for the procedure to print information under the horizontal axis. The exact maximum depends on the titles used, whether or not plots are overlayed, and whether or not CONTOUR is specified. If the value of axis-length specifies a plot that cannot fit on one page, the plot spans multiple pages.

**See also:** HPOS= on page 704

VREF=value-specification
draws lines on the plot perpendicular to the specified values on the vertical axis.
PROC PLOT includes the values you specify with the VREF= option on the vertical
axis unless you specify otherwise with the VAXIS= option. For the syntax for value-specification, see HAXIS= on page 702.

**Featured in:** Example 2 on page 717

**VREFCHAR='character'**
specifies the character to use to draw the vertical reference lines.

- **Default:** horizontal bar (–)
- **See also:** FORMCHAR= option on page 695, HREFCHAR= on page 704, and VREF= on page 709

**VREVERSE**
reverses the order of the values on the vertical axis.

**VSPACE=n**
specifies that a tick mark will occur on the vertical axis at every $n$th print position, where $n$ is the value of VSPACE=.

**VZERO**
assigns a value of zero to the first tick mark on the vertical axis.

**Interaction:** PROC PLOT ignores the VZERO option if the vertical variable has negative values or if the VAXIS= option specifies a range that does not begin with zero.

---

**Concepts**

---

**RUN Groups**

PROC PLOT is an interactive procedure. It remains active after a RUN statement is executed. Usually, SAS terminates a procedure after executing a RUN statement. Once you start the procedure, you can continue to submit any valid statements without resubmitting the PROC PLOT statement. Thus, you can easily experiment with changing labels, values of tick marks, and so forth. Any options submitted in the PROC PLOT statement remain in effect until you submit another PROC PLOT statement.

When you submit a RUN statement, PROC PLOT executes all the statements submitted since the last PROC PLOT or RUN statement. Each group of statements is called a *RUN group*. With each RUN group, PROC PLOT begins a new page and begins with the first item in the VPERCENT= and HPERCENT= lists, if any.

To terminate the procedure, submit a QUIT statement, a DATA statement, or a PROC statement. Like the RUN statement, each of these statements completes a RUN group. If you do not want to execute the statements in the RUN group, use the RUN CANCEL statement, which terminates the procedure immediately.

You can use the BY statement interactively. The BY statement remains in effect until you submit another BY statement or terminate the procedure.

See Example 11 on page 735 for an example of using RUN group processing with PROC PLOT.

---

**Generating Data with Program Statements**

When you generate data to be plotted, a good rule is to generate fewer observations than the number of positions on the horizontal axis. PROC PLOT then uses the increment of the horizontal variable as the interval between tick marks.
Because PROC PLOT prints one character for each observation, using SAS program statements to generate the data set for PROC PLOT can enhance the effectiveness of continuous plots. For example, suppose that you want to generate data in order to plot the following equation, for $x$ ranging from 0 to 100:

$$y = 2.54 + 3.83x$$

You can submit these statements:

```sas
options linesize=80;
data generate;
do x=0 to 100 by 2;
y=2.54+3.83*x;
output;
end;
runk;
proc plot data=generate;
plot y*x;
runk;
```

If the plot is printed with a LINESIZE= value of 80, about 75 positions are available on the horizontal axis for the X values. Thus, 2 is a good increment: 51 observations are generated, which is fewer than the 75 available positions on the horizontal axis.

However, if the plot is printed with a LINESIZE= value of 132, an increment of 2 produces a plot with a space between each plotting symbol. For a smoother line, a better increment is 1, since 101 observations are generated.

### Labeling Plot Points with Values of a Variable

#### Pointer Symbols

When you are using a label variable and do not specify a plotting symbol or if the value of the variable you use as the plotting symbol is null ("00"), PROC PLOT uses pointer symbols as plotting symbols. Pointer symbols associate a point with its label by pointing in the general direction of the label placement. PROC PLOT uses four different pointer symbols based on the value of the S= and V= suboptions in the PLACEMENT= option. The table below shows the pointer symbols:

<table>
<thead>
<tr>
<th>S=</th>
<th>V=</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEFT</td>
<td>any</td>
<td>&lt;</td>
</tr>
<tr>
<td>RIGHT</td>
<td>any</td>
<td>&gt;</td>
</tr>
<tr>
<td>CENTER</td>
<td>&gt;0</td>
<td>^</td>
</tr>
<tr>
<td>CENTER</td>
<td>&lt;=0</td>
<td>v</td>
</tr>
</tbody>
</table>

If you are using pointer symbols and multiple points coincide, PROC PLOT uses the number of points as the plotting symbol if it is between 2 and 9. If it is more than 9, the procedure uses an asterisk.
Note: Because of character set differences among operating environments, the pointer symbol for S=CENTER and V>0 may differ from the one shown here.

**Understanding Penalties**

PROC PLOT assesses the quality of placements with penalties. If all labels are plotted with zero penalty, no labels collide and all labels are near their symbols. When it is not possible to place all labels with zero penalty, PROC PLOT tries to minimize the total penalty. Table 26.2 on page 712 gives a description of the penalty, the default value of the penalty, the index you use to reference the penalty, and the range of values you can specify if you change the penalties. Each penalty is described in more detail in Table 26.3 on page 712.

<table>
<thead>
<tr>
<th>Penalty</th>
<th>Default penalty</th>
<th>Index</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>not placing a blank</td>
<td>1</td>
<td>1</td>
<td>0-500</td>
</tr>
<tr>
<td>bad split, no split character specified</td>
<td>1</td>
<td>2</td>
<td>0-500</td>
</tr>
<tr>
<td>bad split with split character</td>
<td>50</td>
<td>3</td>
<td>0-500</td>
</tr>
<tr>
<td>free horizontal shift, fnh</td>
<td>2</td>
<td>4</td>
<td>0-500</td>
</tr>
<tr>
<td>free vertical shift, fnv</td>
<td>1</td>
<td>5</td>
<td>0-500</td>
</tr>
<tr>
<td>vertical shift weight, vsw</td>
<td>2</td>
<td>6</td>
<td>0-500</td>
</tr>
<tr>
<td>vertical/horizontal shift denominator, vhkd</td>
<td>5</td>
<td>7</td>
<td>1-500</td>
</tr>
<tr>
<td>collision state</td>
<td>500</td>
<td>8</td>
<td>0-10,000</td>
</tr>
<tr>
<td>(reserved for future use)</td>
<td></td>
<td>9-14</td>
<td></td>
</tr>
<tr>
<td>not placing the first character</td>
<td>11</td>
<td>15</td>
<td>0-500</td>
</tr>
<tr>
<td>not placing the second character</td>
<td>10</td>
<td>16</td>
<td>0-500</td>
</tr>
<tr>
<td>not placing the third character</td>
<td>8</td>
<td>17</td>
<td>0-500</td>
</tr>
<tr>
<td>not placing the fourth character</td>
<td>5</td>
<td>18</td>
<td>0-500</td>
</tr>
<tr>
<td>not placing the fifth through 200th character</td>
<td>2</td>
<td>19-214</td>
<td>0-500</td>
</tr>
</tbody>
</table>

Table 26.3 on page 712 contains the index values from Table 26.2 on page 712 with a description of the corresponding penalty.

<table>
<thead>
<tr>
<th>Index</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a nonblank character in the plot collides with an embedded blank in a label, or there is not a blank or a plot boundary before or after each label fragment.</td>
</tr>
<tr>
<td>2</td>
<td>a split occurs on a nonblank or nonpunctuation character when you do not specify a split character.</td>
</tr>
<tr>
<td>3</td>
<td>a label is placed with a different number of lines than the L= suboption specifies, when you specify a split character.</td>
</tr>
</tbody>
</table>
4-7 a label is placed far away from the corresponding point. PROC PLOT calculates the penalty according to this (integer arithmetic) formula:

\[
\text{penalty} = \left[ \max(|H| - fhs, 0) + vsw \times \max(|V| - (L + fhs + (V > 0))/2, 0) \right] / vhsd
\]

Notice that penalties 4 through 7 are actually just components of the formula used to determine the penalty. Changing the penalty for a free horizontal or free vertical shift to a large value such as 500 has the effect of removing any penalty for a large horizontal or vertical shift. Example 6 on page 723 illustrates a case in which removing the horizontal shift penalty is useful.

8 a label may collide with its own plotting symbol. If the plotting symbol is blank, a collision state cannot occur. See “Collision States” on page 713 for more information.

15-214 a label character does not appear in the plot. By default, the penalty for not printing the first character is greater than the penalty for not printing the second character, and so on. By default, the penalty for not printing the fifth and subsequent characters is the same.

**Note:** Labels can share characters without penalty.

### Changing Penalties

You can change the default penalties with the PENALTIES= option in the PLOT statement. Because PROC PLOT considers penalties when it places labels, changing the default penalties can change the placement of the labels. For example, if you have labels that all begin with the same two-letter prefix, you might want to increase the default penalty for not printing the third, fourth, and fifth characters to 11, 10, and 8 and decrease the penalties for not printing the first and second characters to 2. The following PENALTIES= option accomplishes this change:

```plaintext
penalties(15 to 20)=2 2 11 10 8 2
```

This example extends the penalty list. The twentieth penalty of 2 is the penalty for not printing the sixth through 200th character. When the last index \( i \) is greater than 18, the last penalty is used for the \((i - 14)\)th character and beyond.

You can also extend the penalty list by just specifying the starting index. For example, the following PENALTIES= option is equivalent to the one above:

```plaintext
penalties(15)=2 2 11 10 8 2
```

### Collision States

Collision states are placement states that may cause a label to collide with its own plotting symbol. PROC PLOT usually avoids using collision states because of the large default penalty of 500 that is associated with them. PROC PLOT does not consider the actual length or splitting of any particular label when determining if a placement state is a collision state. The following are the rules that PROC PLOT uses to determine collision states:

- **When S=**CENTER, placement states that do not shift the label up or down sufficiently so that all of the label is shifted onto completely different lines from the symbol are collision states.
- **When S=**RIGHT, placement states that shift the label zero or more positions to the left without first shifting the label up or down onto completely different lines from the symbol are collision states.
- **When S=**LEFT, placement states that shift the label zero or more positions to the right without first shifting the label up or down onto completely different lines from the symbol are collision states.
Note: A collision state cannot occur if you do not use a plotting symbol.

Reference Lines

PROC PLOT places labels and computes penalties before placing reference lines on a plot. The procedure does not attempt to avoid rows and columns that contain reference lines.

Hidden Label Characters

In addition to the number of hidden observations and hidden plotting symbols, PROC PLOT prints the number of hidden label characters. Label characters can be hidden by plotting symbols or other label characters.

Overlaying Label Plots

When you overlay a label plot and a nonlabel plot, PROC PLOT tries to avoid collisions between the labels and the characters of the nonlabel plot. When a label character collides with a character in a nonlabel plot, PROC PLOT adds the usual penalty to the penalty sum.

When you overlay two or more label plots, all label plots are treated as a single plot in avoiding collisions and computing hidden character counts. Labels of different plots never overprint, even with the OVP system option in effect.

Computational Resources Used for Label Plots

This section uses the following variables to discuss how much time and memory PROC PLOT uses to construct label plots:

\[ n \]  
number of points with labels

\[ len \]  
constant length of labels

\[ s \]  
number of label pieces, or fragments

\[ p \]  
number of placement states specified in the PLACE= option.

Time

For a given plot size, the time required to construct the plot is roughly proportional to \( n \times len \). The amount of time required to split the labels is roughly proportional to \( n.s^2 \). Generally, the more placement states you specify, the more time that PROC PLOT needs to place the labels. However, increasing the number of horizontal and vertical shifts gives PROC PLOT more flexibility to avoid collisions, often resulting in less time used to place labels.

Memory

PROC PLOT uses 24p bytes of memory for the internal placement state list. PROC PLOT uses \( n(84 + 5/len + 4s(1 + 1.5(s + 1))) \) bytes for the internal list of labels. PROC PLOT builds all plots in memory; each printing position uses one byte of memory. If you run out of memory, request fewer plots in each PLOT statement and put a RUN statement after each PLOT statement.
Results

Scale of the Axes

Normally, PROC PLOT looks at the minimum difference between each pair of the five lowest ordered values of each variable (the \textit{delta}) and ensures that there is no more than one of these intervals per print position on the final scaled axis, if possible. If there is not enough room to do this, and if PROC PLOT guesses that the data were artificially generated, it puts a fixed number of deltas in each print position. Otherwise, it ignores the value.

Printed Output

Each plot uses one full page unless the plot's size is changed by the \texttt{VPOS=} and \texttt{HPOS=} options in the PLOT statement, the \texttt{VPERCENT=} or \texttt{HPERCENT=} options in the PROC PLOT statement, or the \texttt{PAGESIZE=} and \texttt{LINESIZE=} system options. Titles, legends, and variable labels are printed at the top of each page. Each axis is labeled with the variable's name or, if it exists, the variable's label.

Normally, PROC PLOT begins a new plot on a new page. However, the \texttt{VPERCENT=} and \texttt{HPERCENT=} options enable you to print more than one plot on a page. \texttt{VPERCENT=} and \texttt{HPERCENT=} are described earlier in “PROC PLOT Statement” on page 695.

PROC PLOT always begins a new page after a RUN statement and at the beginning of a BY group.

Missing Values

If values of either of the plotting variables are missing, PROC PLOT does not include the observation in the plot. However, in a plot of \texttt{Y*X}, values of \texttt{X} with corresponding missing values of \texttt{Y} are included in scaling the \texttt{X} axis, unless the \texttt{NOMISS} option is specified in the PROC PLOT statement.

Hidden Observations

By default, PROC PLOT uses different plotting symbols (A, B, C, and so on) to represent observations whose values coincide on a plot. However, if you specify your own plotting symbol or if you use the \texttt{OVERLAY} option, you may not be able to recognize coinciding values.

If you specify a plotting symbol, PROC PLOT uses the same symbol regardless of the number of observations whose values coincide. If you use the \texttt{OVERLAY} option and overprinting is not in effect, PROC PLOT uses the symbol from the first plot request. In both cases, the output includes a message telling you how many observations are hidden.
Example 1: Specifying a Plotting Symbol

Procedure features:
- PLOT statement
- plotting symbol in plot request

This example expands on Output 26.1 on page 692 by specifying a different plotting symbol.

Program

```plaintext
options nodate number pageno=1 linesize=80 pagesize=35;

data djia;
  input Year @7 HighDate date7. High @24 LowDate date7. Low;
  format highdate lowdate date7.;
  datalines;
  1954 31DEC54 404.39 11JAN54 279.87
  1955 30DEC55 488.40 17JAN55 388.20
  ...more data lines...
  1993 29DEC93 3794.33 20JAN93 3241.95
  1994 31JAN94 3978.36 04APR94 3593.35
;

proc plot data=djia;
  plot high*year='*';
  title 'High Values of the Dow Jones Industrial Average';
  title2 'from 1954 to 1994';
run;
```

The data set DJIA contains the high and low closing marks for the Dow Jones Industrial Average from 1954 to 1994. A DATA step on page 1499 creates this data set.

The plot request plots the values of High on the vertical axis and the values of Year on the horizontal axis. It also specifies an asterisk as the plotting symbol.
Output

PROC PLOT determines the tick marks and the scale of both axes.

Example 2: Controlling the Horizontal Axis and Adding a Reference Line

Procedure features:

PLOT statement options:
   HAXIS=
   VREF=

Data set: DJIA on page 716

This example specifies values for the horizontal axis and draws a reference line from the vertical axis.

Program

options nodate pageno=1 linesize=80 pagesize=35;
The plot request plots the values of High on the vertical axis and the values of Year on the horizontal axis. It also specifies an asterisk as the plotting symbol. HAXIS= specifies that the horizontal axis will show the values 1950 to 1995 in five-year increments. VREF= draws a reference line that extends from the value 3000 on the vertical axis.

```
proc plot data=djia;
  plot high*year='*' / haxis=1950 to 1995 by 5 vref=3000;
  title 'High Values of Dow Jones Industrial Average';
  title2 'from 1954 to 1994';
run;
```

Output

Example 3: Overlaying Two Plots

**Procedure features:**

- PLOT statement options
  - BOX
  - OVERLAY

**Data set:** DJIA on page 716
This example overlays two plots and puts a box around the plot.

**Program**

```plaintext
options nodate pageno=1 linesize=64 pagesize=30;

proc plot data=djia;
  plot high*year='*' low*year='o' / overlay box;
  title 'Plot of Highs and Lows';
  title2 'for the Dow Jones Industrial Average';
run;
```

**Output**

![Output diagram]

---

**Procedure features:**
PROC PLOT statement options

HPERCENT=  
VPERCENT=

Data set:  DJIA on page 716

This example puts three plots on one page of output.

**Program**

options nodate pageno=1 linesize=120 pagesize=60;

VPERCENT= specifies that 50% of the vertical space on the page of output is used for each plot.
HPERCENT= specifies that 50% of the horizontal space is used for each plot.

proc plot data=djia vpercent=50 hpercent=50;

This plot request plots the values of High on the vertical axis and the values of Year on the horizontal axis. It also specifies an asterisk as the plotting symbol.

plot high*year='*';

This plot request plots the values of Low on the vertical axis and the values of Year on the horizontal axis. It also specifies an asterisk as the plotting symbol.

plot low*year='o';

The first plot request plots High on the vertical axis, plots Year on the horizontal axis, and specifies an asterisk as a plotting symbol. The second plot request plots Low on the vertical axis, plots Year on the horizontal axis, and specifies an 'o' as a plotting symbol. OVERLAY superimposes the second plot onto the first. BOX draws a box around the plot. OVERLAY and BOX apply to both plot requests.

plot high*year='*' low*year='o' / overlay box;
title 'Plots of the Dow Jones Industrial Average';
title2 'from 1954 to 1994';
run;
Output

Plots of the Dow Jones Industrial Average from 1954 to 1994

Plot of High\*Year. Symbol used is '*'.
Plot of Low\*Year. Symbol used is 'o'.

NOTE: 7 obs hidden.

Example 5: Plotting Data on a Logarithmic Scale

Procedure features:
PLOT statement option

HAXIS=

This example uses a DATA step to generate data. The PROC PLOT step shows two plots of the same data — one plot without a horizontal axis specification and one plot with a logarithmic scale specified for the horizontal axis.

**Program**

```plaintext
options nodate pageno=1 linesize=80 pagesize=40;

The DATA step generates the values of X and Y. The values of X result from using specified values of Y as an exponent.

data equa;
  do Y=1 to 3 by .1;
    X=10**y;
    output;
  end;
run;

HPERCENT= makes room for two plots side-by-side by specifying that 50% of the horizontal space is used for each plot.

proc plot data=equa hpercent=50;

The plot requests plot Y on the vertical axis and X on the horizontal axis. HAXIS= specifies a logarithmic scale for the horizontal axis for the second plot.

plot y*x;
plot y*x / haxis=10 100 1000;
title 'Two Plots with Different';
title2 'Horizontal Axis Specifications';
run;
```
Output

Two Plots with Different Horizontal Axis Specifications

Plot of Y*X. A=1, B=2, etc. Plot of Y*X. A=1, B=2, etc.

<table>
<thead>
<tr>
<th>Y</th>
<th></th>
<th>Y</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0 + A</td>
<td>3.0 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.9 + A</td>
<td>2.9 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.8 + A</td>
<td>2.8 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.7 + A</td>
<td>2.7 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.6 + A</td>
<td>2.6 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.5 + A</td>
<td>2.5 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4 + A</td>
<td>2.4 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3 + A</td>
<td>2.3 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2 + A</td>
<td>2.2 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1 + A</td>
<td>2.1 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.0 + A</td>
<td>2.0 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.9 + A</td>
<td>1.9 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.8 + A</td>
<td>1.8 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.7 + A</td>
<td>1.7 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.6 + A</td>
<td>1.6 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5 + A</td>
<td>1.5 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4 + A</td>
<td>1.4 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3 + A</td>
<td>1.3 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2 + A</td>
<td>1.2 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1 + A</td>
<td>1.1 + A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0 + A</td>
<td>1.0 + A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

Example 6: Plotting Date Values on an Axis

**Procedure features:**

- **PLOT statement option**
  - `HAXIS=`

This example shows how you can specify date values on an axis.

**Program**

```plaintext
options nodate pageno=1 linesize=120 pagesize=40;
```

**EMERGENCY_CALLS** contains the number of phone calls to an emergency help line.
data emergency_calls;
  input Date : date7. Calls @@;
  label calls='Number of Calls';
datalines;
1APR94 134 11APR94 384 13FEB94 488
2MAR94 289 21MAR94 201 14MAR94 460
3JUN94 184 13JUN94 152 30APR94 356
4JAN94 179 14JAN94 128 16JUN94 480
5APR94 360 15APR94 350 24JUL94 388
6MAY94 245 15DEC94 150 17NOV94 328
7JUL94 280 16MAY94 240 25AUG94 280
8AUG94 494 17JUL94 499 26SEP94 394
9SEP94 309 18AUG94 248 23NOV94 590
19SEP94 356 24FEB94 201 29JUL94 330
10OCT94 222 25MAR94 183 30AUG94 321
11NOV94 294 26APR94 412 2DEC94 511
27MAY94 294 22DEC94 413 28JUN94 309
;

The plot request plots Calls on the vertical axis and Date on the horizontal axis. HAXIS= uses a monthly time for the horizontal axis. The notation ‘1JAN94’d is a date constant. The value ‘1JAN95’d ensures that the axis will have enough room for observations from December.

proc plot data=emergency_calls;
  plot calls*date / haxis='1JAN94’d to '1JAN95’d by month;

The FORMAT statement assigns the DATE7. format to Date.

format date date7.;
title 'Calls to City Emergency Services Number';
title2 'Sample of Days for 1994';
run;
The PLOT Procedure △ Example 7: Producing a Contour Plot 725

Output

PROC PLOT uses the variables' labels on the axes.

Example 7: Producing a Contour Plot

Procedure features:

PLOT statement option

CONTOUR=

This example shows how to represent the values of three variables with a two-dimensional plot by setting one of the variables as the CONTOUR variable. The variables X and Y appear on the axes, and Z is the contour variable. Program statements are used to generate the observations for the plot, and the following equation describes the contour surface:

\[ z = 46.2 + 0.9x - 0.0005x^2 + 0.1y - 0.0005y^2 + 0.004xy \]
options nodate pageno=1 linesize=64 pagesize=25;

The DATA step creates the CONTOURS data set.

data contours;
  format Z 5.1;
  do X=0 to 400 by 5;
    do Y=0 to 350 by 10;
      z=46.2+.09*x-.0005*x**2+.1*y-.0005*y**2+.0004*x*y;
      output;
    end;
  end;
run;

PROC PRINT prints the CONTOURS data set. The OBS= data set option limits the printing to only the first 5 observations.

proc print data=contours(obs=5) noobs;
  title 'CONTOURS Data Set';
  title2 'First 5 Observations Only';
run;

CONTOURS contains observations with values of X ranging from 0 to 400 by 5 and with values of Y ranging from 0 to 350 by 10.

<table>
<thead>
<tr>
<th>Z</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>46.2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>47.2</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>48.0</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>48.8</td>
<td>0</td>
<td>30</td>
</tr>
<tr>
<td>49.4</td>
<td>0</td>
<td>40</td>
</tr>
</tbody>
</table>

NOOVP ensures that overprinting is not used in the plot.

options nodate pageno=1 linesize=120 pagesize=60 noovp;

The plot request plots Y on the vertical axis, plots X on the horizontal axis, and specifies Z as the contour variable. CONTOUR=10 specifies that the plot will divide the values of Z into ten increments, and each increment will have a different plotting symbol.
The PLOT Procedure

```
proc plot data=contours;
plot y*x=z / contour=10;
   title 'A Contour Plot';
run;
```

**Output**

The shadings associated with the values of Z appear at the bottom of the plot. The plotting symbol # shows where high values of Z occur.
Example 8: Plotting BY Groups

Procedure features:
- PLOT statement option
  HREF=

Other features:
- BY statement

This example shows BY group processing in PROC PLOT.

Program

options nodate pageno=1 linesize=80 pagesize=35;

EDUCATION contains educational data about some U.S. states. DropoutRate is the percentage of high school dropouts. Expenditures is the dollar amount the state spends on each pupil. MathScore is the score of 8th graders on a standardized math test. Not all states participated in the math test. A DATA step on page 1500 creates this data set.

data education;
  input State $14. +1 Code $ DropoutRate Expenditures MathScore Region $;
  label dropout='Dropout Percentage - 1989'
    expend='Expenditure Per Pupil - 1989'
    math='8th Grade Math Exam - 1990';
  datalines;
   Alabama AL 22.3 3197 252 SE
   Alaska AK 35.8 7716 . W
  ...more data lines...
   New York NY 35.0 . 261 NE
   North Carolina NC 31.2 3874 250 SE
   North Dakota ND 12.1 3952 281 MW
   Ohio OH 24.4 4649 264 MW
;

PROC SORT sorts EDUCATION by Region so that Region can be used as the BY variable in PROC PLOT.

proc sort data=education;
  by region;
run;

The BY statement creates a separate plot for each value of Region.

* Data are from the U.S. Department of Education.
proc plot data=education;
  by region;

The plot request plots Expenditures on the vertical axis, plots DropoutRate on the horizontal axis, and specifies an asterisk as the plotting symbol. HREF= draws a reference line extending from 28.6 on the horizontal axis. The reference line represents the national average.

    plot expenditures*dropoutrate='*' / href=28.6;
    title 'Plot of Dropout Rate and Expenditure Per Pupil';
    run;

Output

PROC PLOT produces a plot for each BY group. Only the plots for Midwest and Northeast are shown.
Example 9: Adding Labels to a Plot

**Procedure features:**

- PLOT statement
  - label variable in plot request

**Data set:** EDUCATION on page 728

This example shows how to modify the plot request to label points on the plot with the values of variables. This example adds labels to the plot shown in Example 8 on page 728.

**Program**

```plaintext
options nodate pageno=1 linesize=80 pagesize=35;

PROC SORT sorts EDUCATION by Region so that Region can be used as the BY variable in PROC PLOT.
```
The BY statement creates a separate plot for each value of Region.

```sas
proc sort data=education;
   by region;
run;
```

The plot request plots Expenditures on the vertical axis, plots DropoutRate on the horizontal axis, and specifies an asterisk as the plotting symbol. The label variable specification ( $ state) in the plot request labels each point on the plot with the name of the corresponding state. HREF= draws a reference line extending from 28.6 on the horizontal axis. The reference line represents the national average.

```sas
proc plot data=education;
   by region;
   plot expenditures*dropoutrate='*' $ state / href=28.6;
   title 'Plot of Dropout Rate and Expenditure Per Pupil';
run;
```
PROC PLOT produces a plot for each BY group. Only the plots for Midwest and Northeast are shown.

---

Plot of Dropout Rate and Expenditure Per Pupil

---

Region=MW

---

Plot of Expenditures*DropoutRate$State. Symbol used is ‘*’.

---

Expenditures

| 5500 +
|-------------------
| Michigan *
| 5000 +
| * Illinois
| * Minnesota
| * Ohio
| 4500 +
| * Nebraska * Kansas
| Iowa ** Indiana * Missouri
| 4000 +
| * North Dakota
| 3500 +

---

10 15 20 25 30

---

Dropout Percentage = 1989
Example 10: Excluding Observations That Have Missing Values

Procedure features:
PROC PLOT statement option
NOMISS

Data set: EDUCATION on page 728

This example shows how missing values affect the calculation of the axes.

Program

options nodate pageno=1 linesize=80 pagesize=35;

PROC SORT sorts EDUCATION by Region so that Region can be used as the BY variable in PROC PLOT.

proc sort data=education;
  by region;
NOMISS excludes observations that have a missing value for either of the axis variables.

```plaintext
proc plot data=education nomiss;
```

The BY statement creates a separate plot for each value of Region.

```plaintext
by region;
```

The plot request plots Expenditures on the vertical axis, plots DropoutRate on the horizontal axis, and specifies an asterisk as the plotting symbol. The label variable specification (`$ state`) in the plot request labels each point on the plot with the name of the corresponding state. HREF= draws a reference line extending from 28.6 on the horizontal axis. The reference line represents the national average.

```plaintext
plot expenditures*dropoutrate='*' $ state / href=28.6;
   title 'Plot of Dropout Rate and Expenditure Per Pupil';
run;
```
Output

PROC PLOT produces a plot for each BY group. Only the plot for the Northeast is shown. Because New York has a missing value for Expenditures, the observation is excluded and PROC PLOT does not use the value 35 for DropoutRate to calculate the horizontal axis. Compare the horizontal axis in this output with the horizontal axis in the plot for Northeast in Example 9 on page 730.

Example 11: Adjusting Labels on a Plot with the PLACEMENT= Option

Procedure features:
- PLOT statement options
  - label variable in plot request
  - LIST=
  - PLACEMENT=

Other features:
- RUN group processing

This example illustrates the default placement of labels and how to adjust the placement of labels on a crowded plot. The labels are values of variable in the data set.
This example also shows RUN group processing in PROC PLOT.

**Program**

```plaintext
options nodate pageno=1 linesize=120 pagesize=37;
```

CENSUS contains the variables CrimeRate and Density for selected states. CrimeRate is the number of crimes per 100,000 people. Density is the population density per square mile in the 1980 census. A DATA step on page 1493 creates this data set.

```plaintext
data census;
    input Density CrimeRate State $ 14-27 PostalCode $ 29-30;
    datalines;
263.3 4575.3 Ohio OH
62.1 7017.1 Washington WA
...more data lines...
111.6 4665.6 Tennessee TN
120.4 4649.9 North Carolina NC
;
```

The plot request plots Density on the vertical axis, CrimeRate on the horizontal axis, and uses the first letter of the value of State as the plotting symbol. This makes it easier to match the symbol with its label. The label variable specification (`$ state`) in the plot request labels each point with the corresponding state name. BOX draws a box around the plot. LIST= lists the labels that have penalties greater than or equal to 1. HAXIS= and VAXIS= specify increments only. PROC PLOT uses the data to determine the range for the axes.

```plaintext
proc plot data=census;
    plot density*crimerate=state $ state / box list=1
        haxis=by 1000 vaxis=by 250;
    title 'A Plot of Population Density and Crime Rates';
run;
```

* The data are from the U.S. Bureau of the Census and the 1987 Uniform Crime Reports, FBI.
The labels **Tennessee, South Carolina, Arkansas, Minnesota, and South Dakota** have penalties. The default placement states do not provide enough possibilities for PROC PLOT to avoid penalties given the proximity of the points. Seven label characters are hidden.

A Plot of Population Density and Crime Rates

Plot of Density*CrimeRate$State. Symbol is value of State.

---+------------+------------+------------+------------+------------+------------+------------+---

Density | |
500 + +
| |
| M Maryland

D Delaware

| F Pennsylvania | O Ohio |
250 + +
| I Illinois |
| |
| |
| |
| |
| |
| |
| N New Hampshire T S South Carolina
| W West Virginia A Alabama
| Mississipi N Vermont V M Missouri Washington W T Texas
| MinneArKMsas O Oklahoma
| North Dakota I Idaho O Oregon
| O + S South Dakota N Nevada +

---+------------+------------+------------+------------+------------+------------+------------+---

2000 3000 4000 5000 6000 7000 8000 9000

CrimeRate

NOTE: 7 label characters hidden.

A Plot of Population Density and Crime Rates

List of Point Locations, Penalties, and Placement States

<table>
<thead>
<tr>
<th>Label</th>
<th>Vertical Axis</th>
<th>Horizontal Axis</th>
<th>Penalty</th>
<th>Starting Position</th>
<th>Vertical Lines</th>
<th>Vertical Shift</th>
<th>Horizontal Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tennessee</td>
<td>111.60</td>
<td>4665.6</td>
<td>2</td>
<td>Center</td>
<td>1</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>South Carolina</td>
<td>103.40</td>
<td>5161.9</td>
<td>2</td>
<td>Right</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Arkansas</td>
<td>43.90</td>
<td>4245.2</td>
<td>6</td>
<td>Right</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Minnesota</td>
<td>51.20</td>
<td>4615.8</td>
<td>7</td>
<td>Left</td>
<td>1</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>South Dakota</td>
<td>9.10</td>
<td>2678.0</td>
<td>11</td>
<td>Right</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

Because PROC PLOT is interactive, the procedure is still running at this point in the program. It is not necessary to restart the procedure to submit another plot request. LIST=1 produces no output because there are no penalties of 1 or greater.
plot density*crimerate=state $ state / box list=1
    haxis=by 1000 vaxis=by 250

PLACEMENT= gives PROC PLOT more placement states to use to place the labels. PLACEMENT= contains three expressions. The first expression specifies the preferred positions for the label. The first expression resolves to placement states centered above the plotting symbol, with the label on one or two lines. The second and third expressions resolve to placement states that enable PROC PLOT to place the label in multiple positions around the plotting symbol.

placement=((v=2 1 : l=2 1)
    ((l=2 2 1 : v=0 1 0) * (s=right left : h=2 -2))
    (s=center right left * l=2 1 * v=0 1 -1 2 *
        h=0 1 to 5 by alt));

    title 'A Plot of Population Density and Crime Rates';
    run;

Output
No collisions occur in the plot.

Example 12: Adjusting Labeling on a Plot with a Macro

Procedure features:
- PLOT statement options
  - label variable in plot request
- PLACEMENT=

Data set:  CENSUS on page 736

This example illustrates the default placement of labels and uses a macro to adjust the placement of labels. The labels are values of a variable in the data set.

Program

```
options nodate pageno=1 linesize=120 pagesize=37;
```
The %PLACE macro provides an alternative to using the PLACEMENT= option. The higher the value of n, the more freedom PROC PLOT has to place labels.

```plaintext
%macro place(n);
  %if &n > 13 %then %let n = 13;
  placement=
    %if &n <= 0 %then (s=center); %else (h=2 -2 : s=right left);
    %if &n = 1 %then (v=1 * h=0 -1 to -2 by alt);
    %else %if &n = 2 %then (v=1 -1 * h=0 -1 to -5 by alt);
    %else %if &n > 2 %then (v=1 to 2 by alt * h=0 -1 to -10 by alt);
    %if &n > 3 %then
      (s=center right left * v=0 1 to %eval(&n - 2) by alt *
       h=0 -1 to %eval(-3 * (&n - 2)) by alt *
       l=1 to %eval(2 + (10 * &n - 35) / 30));
  %if &n > 4 %then penalty(?)=%eval((3 * &n) / 2);
%mend;

proc plot data=census;
  plot density*crimerate=state $ state / box list=1
      haxis=by 1000 vaxis=by 250 %place(4);
  title 'A Plot of Population Density and Crime Rates';
run;
```

Output
No collisions occur in the plot.

---

**Example 13: Changing a Default Penalty**

**Procedure features:**

- **PLOT statement option**
  - PENALTIES=

**Data set:**  CENSUS  on page 736

This example demonstrates how changing a default penalty affects the placement of labels. The goal is to produce a plot that has labels that do not detract from how the points are scattered.

**Program**

```plaintext
options nodate pageno=1 linesize=120 pagesize=37;
```
The plot request plots Density on the vertical axis, CrimeRate on the horizontal axis, and uses the first letter of the value of State as the plotting symbol. The label variable specification (\$ state) in the plot request labels each point with the corresponding state name.

```
proc plot data=census;
  plot density*crimerate=state \$ state /
placement=(h=100 to 10 by alt * s=left right)
penalties(4)=500 list=0
haxis=0 to 13000 by 1000 vaxis=by 100;
title 'A Plot of Population Density and Crime Rates';
run;
```
A Plot of Population Density and Crime Rates

Plot of Density*CrimeRate*State. Symbol is value of State.

NOTE: 1 obs hidden.
### A Plot of Population Density and Crime Rates

#### List of Point Locations, Penalties, and Placement States

<table>
<thead>
<tr>
<th>Label</th>
<th>Vertical Axis</th>
<th>Horizontal Axis</th>
<th>Penalty</th>
<th>Starting Position</th>
<th>Vertical Lines</th>
<th>Horizontal Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maryland</td>
<td>428.70</td>
<td>5477.6</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Delaware</td>
<td>307.60</td>
<td>4938.8</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Pennsylvania</td>
<td>264.30</td>
<td>3163.2</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Ohio</td>
<td>263.30</td>
<td>4575.3</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Illinois</td>
<td>205.30</td>
<td>5416.5</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Florida</td>
<td>180.00</td>
<td>8503.2</td>
<td>0</td>
<td>Left</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>California</td>
<td>151.40</td>
<td>6506.4</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Tennessee</td>
<td>111.60</td>
<td>4665.6</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>North Carolina</td>
<td>120.40</td>
<td>4649.9</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>New Hampshire</td>
<td>102.40</td>
<td>3371.7</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>South Carolina</td>
<td>103.40</td>
<td>5161.9</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Georgia</td>
<td>94.10</td>
<td>5792.0</td>
<td>0</td>
<td>Left</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>West Virginia</td>
<td>80.80</td>
<td>2190.7</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Alabama</td>
<td>76.60</td>
<td>4451.4</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Missouri</td>
<td>71.20</td>
<td>4707.5</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Mississippi</td>
<td>53.40</td>
<td>3438.6</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Vermont</td>
<td>55.20</td>
<td>4271.2</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Minnesota</td>
<td>51.20</td>
<td>4615.8</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Washington</td>
<td>62.10</td>
<td>7017.1</td>
<td>0</td>
<td>Left</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Texas</td>
<td>54.30</td>
<td>7722.4</td>
<td>0</td>
<td>Left</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Arkansas</td>
<td>43.90</td>
<td>4245.2</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Oklahoma</td>
<td>44.10</td>
<td>6025.6</td>
<td>0</td>
<td>Left</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Idaho</td>
<td>11.50</td>
<td>4156.3</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Oregon</td>
<td>27.40</td>
<td>6969.9</td>
<td>0</td>
<td>Left</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>South Dakota</td>
<td>9.10</td>
<td>2678.0</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>North Dakota</td>
<td>9.40</td>
<td>2833.0</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Nevada</td>
<td>7.30</td>
<td>6371.4</td>
<td>0</td>
<td>Right</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>