This handout will describe a set of procedures/commands that can be used to create high quality graphics in SAS and export them in several formats. All the procedures are part of the SAS/GRAPH package. Our aim is not to cover all the procedures that appear within SAS/GRAPH; instead we will focus in the ones most frequently used.

First, we generate a dataset that we will use for most of the examples that are presented below. To do so, run the following commands:

```sas
data new;
  do _n_=1 to 150;
    x = 5 + 2*ranuni(1234);
    y = 3 + 3*ranuni(1234);
    z = 2 + 4*ranuni(1234);
    obs + 1;
  output;
  end;
run;
```

The dataset `new` consists of 150 observations on three variables, `x`, `y`, and `z` (note that the fourth variable, `obs`, is just the observation ID).

I. Creating and formatting plots of one variable: The GPLOT procedure

Creating a plot in SAS using the GPLOT procedure can be done by simply using the command:

```sas
proc gplot data=new;
  plot x*obs;
run;quit;
```

The above command will use the variables that reside in the dataset `new` (specified by the command `data=new`) to plot `x` versus the observation ID (specified by the command `plot x*obs`). The resulting graph is presented in Figure 1.
This graph merely utilizes the graphical capabilities of the GLOT procedure; it simply plots the data-points of $x$, without even connecting them. In order to produce a “better” looking graph, try the following

```symbol i=join v=circle c=blue;
proc gplot data=new;
plot x*obs;
run;
```

Figure 1: A very simple plot
In this new figure, the data-points of $x$ are represented by a circle, all points are connected, and the line color is blue. This formatting is done using the `symbol` statement just before the GPLOT procedure. Specifically, the command we used is:

```
symbol i=join v=circle c=blue;
```

- The `i=join` controls the way that the data-points will be interpolated. Using `i=join` tells SAS to produce a graph where all points are connected. If you don’t want the points to be connected, simply use `i=none`. Other interpolation methods are `i=needle` and `i=step`; try them to see their effects.
- The `v=` defines the symbol that we will use for the data-points. Using `v=circle` produces a plot where the data-points are represented by a circle. You can use `v=none, v=dot, v=star`, just to name a few.
- The `c=` defines the colour that we will use for the graph; you can use `c=blue, c=red, c=gray`, etc.

Having explained the way of formatting the plotting curve, we can now focus to commands that enhance the readability of your graph, such as adding titles or footnotes. This can be done by simply using the `title` and `footnote` commands just before the GPLOT procedure. For an illustration, try the following:

```
title1 'A simple plot of variable x';
title2 'x: Normally distributed';
footnote 'Source: Randomly generated data using ranuni';
```
There is one important feature to notice here: instead of using the command `title`, we used `title1` and `title2`. This is the only way to tell SAS that we want both titles to be printed in the graph. In addition, SAS will automatically alter the size of the fonts used for each title so that the font of `title1` is greater than the font of `title2`.

We will finish this section by presenting one more command that will enhance the readability of your graph even more. Suppose that you want to rename the vertical axis from `x` to `Regressor` and the horizontal axis from `obs` to `Observation`. To do that, use the following set of commands:

```plaintext
symbol i=join v=none c=black;
proc gplot data=new;
plot x*obs/;
run;quit;

`title1 'A simple plot of variable x';
title2 'x: Normally distributed';
title3 'x = 5 + 2*ranuni(1234)';
footnote1 'Source: Randomly generated data using ranuni';
symbol i=join v=none c=black;
axis1 label=('Regressor');
axis2 label=('Observations');
proc gplot data=new;
plot x*obs/vaxis=axis1 haxis=axis2;
run;quit;
```

The resulting graph is presented below.
By a careful look in the previous code, we can deduce that the inclusion of axes’ labels in a graph is done in two steps. First, we have to create the label; this is done using the axis1 and axis2 commands. We set the label by simply using the command \texttt{axis1 label=('Whatever name you like!')}. The second step is to place the label into the appropriate axis of the graph. This is done using the \texttt{.../vaxis=axis1 haxis=axis2} command\footnote{The commands that appear in the same line with the \texttt{plot x*obs} are called options in the “SAS” language.}. The \texttt{vaxis} stands for vertical axis and the \texttt{haxis} for the horizontal one. By specifying \texttt{vaxis=axis1}, you instruct SAS to use the information you enter in the \texttt{axis1} command for the vertical axis; similarly, you set \texttt{haxis=axis2} to modify the horizontal axis; note that, if you don’t use these last two commands, the axes will not be modified.

\textit{Two technical notes}

1. As you have seen in the above examples, all the options that alter the behavior of the graph, like \texttt{title1}, \texttt{title2}, \texttt{symbol}, \texttt{footnote}, etc. are entered before the \texttt{proc gplot} command. The programs will work fine even if you include them after the \texttt{proc gplot} statement; just make sure that they will appear before you enter the \texttt{run} command. Feel free to select the way that is more convenient to you.

2. The title, footnote, and symbol options are what SAS calls global options. What this means is that if you run them once, they will appear in all subsequent graphs, even if you don’t ask for any of them. For example, try now to run the command

\begin{verbatim}
proc gplot data=new;
plot x*obs;
\end{verbatim}
run;quit;

As you can see from the above command, we did not specify any title, footnote, or symbol for the graph. However, the graph still displays the title, footnotes, etc. we set before. One way to work around this behavior is to type the following command before the proc gplot:

goptions reset=global;

This command will reset all the graph options to their default values; none of the previous titles or footnotes will be printed in the new graph. Try it by running

goptions reset=global;
proc gplot data=new;
plot x*obs;
run;quit;

Generally speaking, it is a good idea to include the goptions reset=global; before any of the graphs you create; this ensures that what is printed in a graph is only what you asked for!

II. Creating and formatting plots of two or more variables: The GPLOT procedure revisited

Suppose that you want to create a graph that plots both variables x and y. To do that enter the following

goptions reset=global;
axis1 label=('Regressors');
axis2 label=('Observations');
symbol1 i=join v=star c=black;
symbol2 i=join v=none c=red;
title1 'A simple plot of x and y';
title2 'x, y: Normally distributed';
footnote1 ' Source: Randomly generated data using ranuni';
proc gplot data=new;
plot (x y)*obs/overlay vaxis=axis1 haxis=axis2;
run;quit;

The resulting graph is presented below.
First, note that the `title1`, `title2`, `footnote`, `axis1` and `axis2` statements work in the same way as before. The are only three differences with the previous, one variable graphs, namely:

i. You have to use the command `plot (x y)*obs` in order to inform SAS that you need a graph that will plot both x and y.

ii. You have to use the `overlay` option; otherwise, SAS will produce two different graphs, one for x and one for y, and will not plot them in the same one.

iii. Since you plot two different variables, you have to use the `symbol1` and `symbol2` statements (instead of just `symbol` that we used before). The `symbol1` statement controls the appearance of the first variable that you plot (in this case x), and `symbol2` controls the one of the second i.e. y.

One of the problems when we plot several variables into one graph is to distinguish the one from the other. We can simply say that x is represented by the black line and y by the red; however, this is not convenient since we always have to look on the code to tell that. An efficient way is by using legends. This can be done using the legend command as follows:

```sas
options reset=global;
axis1 label=("Regressors");
axis2 label=("Observations");
legend1 label=none value=('X' 'Y');
symbol1 i=join v=star c=black;
```

\(^2\)If you wonder how we know which variable is considered “first” and which is considered “second” just have a look in the plot command we used above. Since we used `plot (x y)*obs`, x is the first and y is the second.
symbol2 i=join v=none c=red;
title1 'A simple plot of x and y';
title2 'x, y: Normally distributed';
footnote1 'Source: Randomly generated data using ranuni';
proc gplot data=new;
plot (x y)*obs/overlay vaxis=axis1 haxis=axis2 legend=legend1;
run;quit;
The new graph has a legend that shows clearly which variable is x and which is y. To create the legend, we followed a similar procedure as with the axis1 and axis2 statements. First, we used the legend1 label=none value=('X' 'Y'); command. This instructs SAS to create a legend for the lines we plot and give to them the names ‘X’ and ‘Y’; once again, we use the order ‘X’, ‘Y’ because the order we used in the plot command was plot (x y)*obs. Also, we asked for no label on the legend; change this to label=('My Legend') to see its effect. Second, we used the legend=legend1 option in order to place the legend we created in the graph.

**Technical note**

One thing to notice from all the examples we covered above is that, any text you want to display in a graph must be written inside the single quotes, i.e. 'Any text'. This is the case for the labels, legend, title, and footnotes. So, make sure that you will always use this rule.

### III. Creating 3D plots: The G3D procedure

In this section we will describe the g3d procedure that is used to create 3D plots. The commands used are very similar to the ones we have seen already. For illustration purposes, let us generate a dataset from the Cobb–Douglas production function.

```sas
data new3d;
  do x=1 to 50 by 0.5;
    do y=1 to 50 by 0.5;
      /* Code for generating data */
    end;
  end;
run;
```
z=(x**0.3)*(y**0.7);
output;
end;
end;
run;

A 3D plot of the production function can be generated as follows

goptions reset=global;
title1 'The Cobb-Douglas Production Function';
title2 'Constant Returns to Scale';
proc g3d data=new3d;
plot y*x=z/ROTATE = 15 TILT=75
CTOP=BLUE CBOTTOM=GRAY
CTEXT=BLACK CAXIS=BLACK;
label y='Input 1' x ='Input 2' z = 'Output';
run;
quit;
Figure 6: The Cobb-Douglas production function

Note that the title commands are used in the same way as before (the same holds for the footnote command). Concerning the rest of the commands, they are used in the following way:

- The plot y*x=z instructs SAS to create a 3D plot. The y*x let SAS know which variables form the xy plane; the variable that appears after the = sign, variable z in our case, is the one that is plotted on the vertical axis. For example, suppose that u=f(x1, x2); then, the plot command will be plot x1*x2=u or plot x2*x1=u.
- The ROTATE=15 command instructs SAS to rotate the xy plane about the perpendicular z axis by 15 degrees.
- The TILT=75 instructs SAS to tilt the graph towards you by 75 degrees. The easiest way to understand the use of the ROTATE and TILT commands is by simply changing their numerical values to see how they affect the resulting graph.
- Use CTOP, CBOTTOM, CTEXT or CAXIS to control: the colour that appears on the surface of your plot, the colour that appears under the surface you plot, the text-colour, and the axis-colour, respectively.
- Finally, the label command is used to alter the label of the x, y, and z axis.

IV. Using graph templates: The GREPLAY procedure

SAS has the ability to display multiple plots on the same page. To accomplish that, you have to use the GREPLAY procedure which allows you to put different numbers of plots in different layouts on
the same page. SAS has already some layouts, called templates, but you can also create your own. Here we will describe the built-in SAS templates and their use. The built-in SAS templates are:

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>H2</td>
<td>1 BOX LEFT, 1 BOX RIGHT</td>
</tr>
<tr>
<td>H2S</td>
<td>1 BOX LEFT, 1 BOX RIGHT (WITH SPACE)</td>
</tr>
<tr>
<td>H3</td>
<td>3 BOXES ACROSS (HORizontally)</td>
</tr>
<tr>
<td>H3S</td>
<td>3 BOXES ACROSS (WITH SPACE)</td>
</tr>
<tr>
<td>H4</td>
<td>4 BOXES ACROSS (HORizontally)</td>
</tr>
<tr>
<td>H4S</td>
<td>4 BOXES ACROSS (WITH SPACE)</td>
</tr>
<tr>
<td>L1R2</td>
<td>1 BOX LEFT, 2 BOXES RIGHT</td>
</tr>
<tr>
<td>L1R2S</td>
<td>1 BOX LEFT, 2 BOXES RIGHT (WITH SPACE)</td>
</tr>
<tr>
<td>L2R1</td>
<td>2 BOXES LEFT, 1 BOX RIGHT</td>
</tr>
<tr>
<td>L2R1S</td>
<td>2 BOXES LEFT, 1 BOX RIGHT (WITH SPACE)</td>
</tr>
<tr>
<td>L2R2</td>
<td>2 BOXES LEFT, 2 BOXES RIGHT</td>
</tr>
<tr>
<td>L2R2S</td>
<td>2 BOXES LEFT, 2 BOXES RIGHT (WITH SPACE)</td>
</tr>
<tr>
<td>U1D2</td>
<td>1 BOX UP, 2 BOXES DOWN</td>
</tr>
<tr>
<td>U1D2S</td>
<td>1 BOX UP, 2 BOXES DOWN (WITH SPACE)</td>
</tr>
<tr>
<td>U2D1</td>
<td>2 BOXES UP, 1 BOX DOWN</td>
</tr>
<tr>
<td>U2D1S</td>
<td>2 BOXES UP, 1 BOX DOWN (WITH SPACE)</td>
</tr>
<tr>
<td>V2</td>
<td>1 BOX UP, 1 BOX DOWN</td>
</tr>
<tr>
<td>V2S</td>
<td>1 BOX UP, 1 BOX DOWN (WITH SPACE)</td>
</tr>
<tr>
<td>V3</td>
<td>3 BOXES STACKED VERTICALLY</td>
</tr>
<tr>
<td>V3S</td>
<td>3 BOXES STACKED VERTICALLY (WITH SPACE)</td>
</tr>
<tr>
<td>WHOLE</td>
<td>ENTIRE SCREEN TEMPLATE</td>
</tr>
</tbody>
</table>

The most important steps in using the GREPLAY is to:

i. Select the template you want
This step is quite straightforward. For example, if you want to put 4 graphs in one page you can use the L2R2 or L2R2s template; if you want to put 3 graphs, you can use V3, H3, U2D1, etc. Don’t worry for the exact command that you have to use; we will study this in a while. At this moment, just pick the name of the template you want to use.

ii. Select the graphs you want to include in the template
Note that the graphs must already be created in SAS. So, suppose that you have already created some graphs and you want to display some, or even all, of them in the template you selected above. The question that arises now is the following: Do these graphs have a specific name that SAS can understand? The answer is yes! To find this name, just display the graph that you want in the SAS graphs-window. In the top of the graph-window you will see a name that will look like the following: GRAPH1 WORK.GSEG.GPLOT or GRAPH1 WORK.GSEG.GPLOT’a positive number’ for example GRAPH1 WORK.GSEG.GPLOT11. The name of your graph is the last part of the above name, in our example, GPLOT11. The figure below shows where exactly to find this name.
Having selected the template and the graphs you will display in it, you just enter the following command in SAS:

```
proc greplay igout=gseg nofs tc=sashelp.templt template=L2R2;
treplay 1:gplot2
    2:gplot3
    3:gplot4
    4:g3d;
run;
quit;
```

In the above code, you only need to set the `template=` and the names of the plots that appear after the `treplay` command. For the `template=`, just use the template you want (e.g. `template=V3S` or `template=U1D2`, etc.). After the `treplay` command, just list the plots you want to use (make sure that you will use 1:plotname 2:plotname, etc. Entering just the name of the plot will not work). Your output will look like the following
IV. Exporting graphs in several formats

The graphs that we created with GPlot, G3D and GREPLAY can be exported in many different formats that can be used in another applications. Saving a graph is quite straightforward. First, make sure that the graph you want to export is currently the one displayed in the graph-window. Click on File→Export as image… In the export window that will open, first select the folder on which you will save the graph (you can navigate through all folders of your PC by clicking on the Save in drop down menu); then enter a name for your graph in the File name box; finally select the format you prefer from the Save as type drop down menu. There are many different options in this last menu. The ones that will be the most useful are BMP, GIF, JPEG, TIFF, PS, and EPSI.

The first four types are readable from all Microsoft Office products; hence, if you want to use the graph in a report made by Word, Excel or in a PowerPoint presentation, you can use one of them. These four formats differ in the size (in terms of kilobytes) of the created file and in its quality. Try all of them to figure out which one fits you the best (usually, the JPEG format is considered the best).

The last two types, PS and EPSI, are very useful when you are writing your thesis, a paper for a journal, etc. The reason is that the vast majority of technical documents are written in a program called LaTeX. In order to include graphs in LaTeX, they must be saved in PS or EPS form; this is where the popularity of the last two formats comes from.
Exercises:

1. Use the GPLOT procedure to create a plot of $x$, $y$, and $z$ (these variables can be found in the dataset `new` we created above). Alter the graph symbols so that $x$, $y$, and $z$ are represented by a blue, red, and yellow line, respectively. Make sure that no titles or footnotes appear in the graph.

2. Insert the following title in the above graph:

   A simple plot of X, Y and Z
   
   $X \sim N(5, 4)$
   
   $Y \sim N(3, 9)$
   
   $Z \sim N(2, 16)$

   and rename the vertical axis to *Regressors* and the horizontal to *Observations*.

3. Instead of having all the information to appear in the title, modify the above graph as follows:

   Create the following title: A simple plot of X, Y and Z
   
   Add legends and modify their labels so that the legend of X is $X \sim N(5, 4)$, the one of Y is $Y \sim N(3, 9)$, and similarly for Z
   
   Add the following footnote: Randomly generated data.

4. Generate a dataset that resembles the Leontief production function. To do that, use the same code with the one we had for the dataset `new3d`; just alter the definition of z so that it now reads $z = \min(x, y)$. Create a simple 3D-plot of the Leontief production function (no titles or footnotes).

5. Modify the above graph as follows: Use the title: The Leontief Production Function

   Change the label of the x-axis to Personal Computer, the label of the y-axis to Typists, and the label of the z-axis to Pages/minute.

   Change the colour that appears on top of the graphs’ surface to red, and the one that appears below it to gray.

6. Change the perspective of the graph you created above so that the inner part of the graphs' surface is visible.

7. Create a template that will use the graphs of exercises 3, 4, 5 and the Cobb-Douglas production function we created earlier.