# Data Visualization and Graphical Analysis

Graphical analysis is a method of presenting and analyzing data using graphs, charts, diagrams, or other visual formats. Graphical analysis can help to simplify complex data, reveal patterns and trends, compare different categories, and communicate information effectively. Some of the advantages of graphical analysis are:

- It helps us understand the data or information even when we have no prior knowledge about it.
- It saves time and space, as graphs can convey a lot of information in a concise and clear manner.
- It allows us to relate and compare the data for different time periods or different kinds, and to identify outliers and anomalies.
- It is used in statistics to determine the mean, median and mode for different data, as well as in the interpolation and extrapolation of data.
- It can support making predictions and informed decisions based on the data trends and patterns.

However, graphical analysis also has some limitations, such as:

- It can be costly and time-consuming to create and maintain graphical representations of data, especially if they involve images, colors, and paints.
- It can be misleading or inaccurate if the graphical representations are not designed properly, such as using inappropriate scales, axes, labels, or legends.
- It can be difficult to interpret or compare graphical representations if they are too complex, cluttered, or inconsistent.
- It can restrict further data analysis, as diagrams do not allow the user to perform calculations or statistical tests on the data.
- It can portray only a limited number of characteristics, as diagrams cannot show all the details and nuances of the data.

Therefore, graphical analysis should be used with caution and in conjunction with other methods of data presentation and analysis, such as tables, text, and numerical summaries.

# GNUPlot

Gnuplot is a free and open source software that can be used to create and display graphs of various types, such as line plots, scatter plots, bar charts, histograms, pie charts, and more. Gnuplot can also plot mathematical functions and data points in two or three dimensions, and perform curve fitting and statistical analysis on the data. Gnuplot can be run interactively from the command line, or scripted using files that contain commands and data. Gnuplot can output the graphs in different formats, such as PNG, JPEG, PDF, SVG, and LaTeX.

To use gnuplot, you need to have it installed on your system. You can check if you have gnuplot by typing gnuplot --version in the terminal. If you don't have gnuplot, you can install it from the official website or using your package manager. For example, on Ubuntu, you can install gnuplot by typing sudo apt install gnuplot in the terminal.

To start gnuplot, you can simply type gnuplot in the terminal. You will see a prompt that looks like this:

```
gnuplot>
```

You can type commands at the prompt to plot graphs, set options, and perform calculations. For example, to plot a simple sine function, you can type:

```
gnuplot> plot sin(x)
```

This will open a window that shows the graph of the sine function. You can also save the graph to a file by specifying the output format and the file name. For example, to save the graph as a PNG file, you can type:

```
gnuplot> set terminal png
gnuplot> set output 'sine.png'
gnuplot> plot sin(x)
gnuplot> set output
```

The last command closes the output file. You can also plot data from a file by using the plot command with the file name as an argument. For example, if you have a file called data.txt that contains two columns of numbers, you can plot them as points by typing:

```
gnuplot> plot 'data.txt'
```

You can also specify the columns to use for the x and y axes, the style of the points, the title of the graph, and the labels of the axes. For example, to plot the second column versus the first column, using circles as points, with a title and labels, you can type:

```
gnuplot> plot 'data.txt' using 1:2 with points point type 7 title 'Data' xlabel 'X' ylabel 'Y'
```

You can also plot multiple graphs on the same window by using the comma operator. For example, to plot the sine and cosine functions together, you can type:

```
gnuplot> plot sin(x), cos(x)
```

You can also fit a function to the data points by using the fit command. For example, to fit a linear function to the data in data.txt, you can type:

```
gnuplot> f(x) = a + b*x
gnuplot> fit f(x) 'data.txt' via a,b
gnuplot> plot 'data.txt', f(x)
```

This will print the values of the parameters a and b, and the statistics of the fit, and plot the data points and the fitted function. You can use any function that is defined properly, and any number of parameters.

## Basic commands in Gnuplot

Some of the basic commands in gnuplot are:

- plot: This command is used to plot 2-d functions and data.The syntax is plot [range] [options] expression [options]. For example, plot sin(x) will plot the sine function in the default range and style. plot [0:2*pi] sin(x) with lines will plot the sine function in the specified range and with lines as the style. plot 'data.txt' using 1:2 will plot the data from the file data.txt using the first and second columns as the x and y coordinates.

- splot: This command is used to plot 3-d surfaces and data. The syntax is similar to plot, except that it can take two ranges for the x and y axes, and an expression or a file name for the z coordinate. For example, splot x*y will plot the surface z = x*y in the default ranges. splot [0:1] [0:1] x*y with pm3d will plot the same surface in the specified ranges and with a color map as the style. splot 'data.txt' using 1:2:3 will plot the data from the file data.txt using the first, second, and third columns as the x, y, and z coordinates.

- set: This command is used to set various options and parameters for the graphs, such as the terminal, the output, the title, the labels, the ranges, the scales, the styles, the legends, and more. For example, set terminal png will set the output format to PNG. set output 'graph.png' will set the output file name to graph.png. set title 'My Graph' will set the title of the graph to My Graph. set xlabel 'X' will set the label of the x axis to X. set xrange [0:10] will set the range of the x axis to [0, 10]. set logscale y will set the scale of the y axis to logarithmic. set style data points will set the default style for data plots to points. set key off will turn off the legend.

- show: This command is used to show the current settings and parameters for the graphs, such as the terminal, the output, the title, the labels, the ranges, the scales, the styles, the legends, and more. For example, show terminal will show the current output format. show output will show the current output file name. show title will show the current title of the graph. show xlabel will show the current label of the x axis. show xrange will show the current range of the x axis. show logscale will show the current scales of the axes. show style will show the current styles for the plots. show key will show the current legend settings.

- fit: This command is used to fit a function to the data points by using a non-linear least-squares algorithm. The syntax is fit [options] function file via parameters. For example, fit f(x) 'data.txt' via a,b will fit the function f(x) to the data from the file data.txt using the parameters a and b. The function and the parameters must be defined before using the fit command. The fit command will print the values of the parameters and the statistics of the fit, and store them in internal variables for later use.

- help: This command is used to get help on the syntax and usage of other commands and topics. The syntax is help [topic]. For example, help plot will show the help on the plot command. help functions will show the help on the built-in functions. help operators will show the help on the operators. help without any argument will show the general help.