There are four libraries imported for this analysis: MatplotLib, numpy, pandas and root_numpy. Root is a framework created by CERN for large scale data analysis such as the one you are about to attempt (and hopefully complete!) and pandas also contains functions related to data manipulation and analysis. Numpy is python's "fundamental package for scientific computing" and MatplotLib allows graphs to be drawn (using the same functions as in MATLAB). The list below gives some functions that you will find useful, although you can use other functions from the libraries given, common arguments used in the functions are given. Note that you should type them as library.function(arguments).

numpy

sqrt(n) - Suprisingly enough returns the square root of n.

mean(data) - Returns the mean of given data.

sum(data == 1) - Returns the number of occurences of 1 in the data. (Like COUNTIF in Excel)

maximum(data1,data2) - Compares two datasets and returns an array containing the element-wise maxima. http://docs.scipy.org/doc/numpy-1.10.0/reference/generated/numpy.maximum.html

minimum(data1,data2) - Compares two datasets and returns an array containing the element-wise minima. http://docs.scipy.org/doc/numpy-1.10.1/reference/generated/numpy.minimum.html

root_numpy

root2array(filepath, selection) - Takes files from root format to a usable array in python. https://rootpy.github.io/root_numpy/reference/generated/root_numpy.root2array.html

MatplotLib

hist(data, bins = n, range = [x,y]) - Plots a histogram of given data in *n* equally spaced bins over the range *x* to y. e.g. hist(data.H1_PX, bins = 40, range = [-100000,100000]).

hist2d(data1,data2, bins = n,) - Plot a 2D histogram from two datasets, with n^2 bins.

scatter(x,y,c,alpha) - Plot a scatter plot of x vs y, with colour 'c' and alpha blending value (between 0-1). http://matplotlib.org/api/pyplot_api.html#matplotlib.pyplot.scatter

Pandas

DataFrame(data) - The primary pandas data structure. Use on initial data set to create a variable which other panda functions can be applied too. e.g. pandaData = DataFrame(data), pandaData.head() shows the first few rows of your data, formatted in a nice table.

http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html

Note: The rest of these functions apply to a panda data structure!!!

DataFrame.head(n) - When applied to a panda data structure produces a table of the first n rows of data in the strucutre.

DataFrame.eval(expression) - Evaluate an expression in the context of the DataFrame. E.G. data['M'] = data.eval('E^2 - P^2')

DataFrame.min() - Returns the minimum value for each axis in a dataframe (an axis is equivelant to a column)

DataFrame.max() - Returns the maximum value for each axis in a dataframe

http://pandas.pydata.org/pandas-docs/stable/generated/pandas.eval.html

DataFrame.query(expression) - Allows selection from a DataFrame variable using an expression of logical rules. e.g. for a = [1,2,3], b = [2,4,0], c = [22,5,-4] using DataFrame.query(a<b<c) returns a = 1, b = 2, c = 22 and a =2, b=4, c=5 but not the third possibility.