

## List of useful functions

There are four libraries imported for this analysis: `matplotlib`, `numpy`, `pandas`.

- `numpy` is used for performing mathematical operations on arrays
- `matplotlib` allows graphs to be plotted
- `pandas` provides an interface for manipulating datasets

The list below gives some functions that you will find useful, although you can use other functions from the libraries given. Clicking on the heading of each function will take you to the function's main documentation page.

### numpy

#### `numpy.sqrt(n)`

Description	Example
Return the square root of <code>n</code>	<pre>&gt; a = numpy.array([1, 4, 9]) &gt; numpy.sqrt(a) numpy.array([1, 2, 3])</pre>

#### `numpy.mean(data)`

Description	Example
Return the mean of <code>data</code>	<pre>&gt; a = numpy.array([1, 4, 9, 3]) &gt; numpy.mean(a) 4.25</pre>

#### `numpy.sum(data)`

Description	Example
Sum all elements in <code>data</code>	<pre>&gt; a = numpy.array([1, 4, 9]) &gt; numpy.sum(a) 14</pre>

#### `numpy.minimum(data1, data2)`

Description	Example
Compare two datasets and returns an array containing the element-wise minima	<pre>&gt; a = numpy.array([1, 4, 9]) &gt; b = numpy.array([9, 4, 1]) &gt; numpy.minimum(a, b) numpy.array([1, 4, 1])</pre>

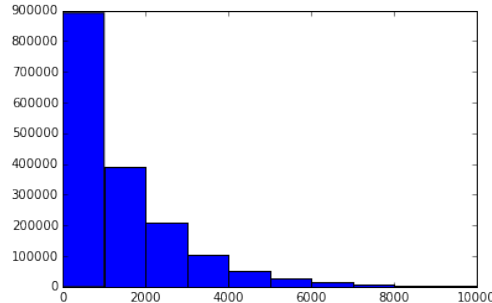
```
numpy.maximum(data1, data2)
```

Description	Example
Compare two datasets and returns an array containing the element-wise maxima	<pre>&gt; a = numpy.array([1, 4, 9]) &gt; b = numpy.array([9, 4, 1]) &gt; numpy.maximum(a, b) numpy.array([9, 4, 9])</pre>

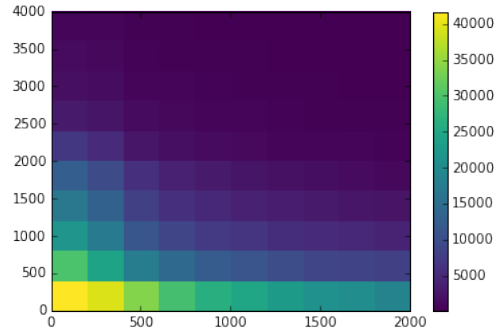
## matplotlib

Functions in `matplotlib` are accessed through `pyplot`.

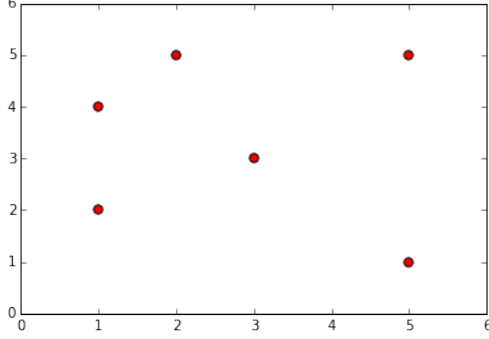
```
pyplot.hist(data, n, [x1, x2])
```

Description	Example
Plots a histogram of given data in <code>n</code> equally spaced bins over the range <code>x1</code> to <code>x2</code>	<pre>&gt; pyplot.hist(     data.H1_PX, 10, [0, 10000] )</pre> 

```
pyplot.hist2d(data1, data2, n, [[x1, x2], [y1, y2]])
```

Description	Example
Plot a 2D histogram from two datasets, with $n^2$ bins equally spaced between <code>x1</code> and <code>x2</code> in <code>x</code> and <code>y1</code> and <code>y2</code> in <code>y</code>	<pre>&gt; pyplot.hist2d(     data.H1_PX, data.H2_PX,     10, [[0, 2000], [0, 4000]] ) &gt; pyplot.colorbar()</pre> 

`pyplot.scatter(x, y, size, color)`

Description	Example
Plot a scatter plot of <b>x</b> vs <b>y</b> where each point has area <b>size</b> and colour <b>color</b>	<pre>&gt; pyplot.scatter(     [1, 2, 5, 1, 3, 5],     [2, 5, 1, 4, 3, 5],     40, "red") )</pre> 

## pandas

The following pandas functions have to be applied to an existing **DataFrame** object, see the example analysis for a demonstration.

`df.head(n)`

Description	Example																
Produces a table of the first <b>n</b> rows of data in the structure	<div>&gt; df.head(3)</div> <table><tr><th></th><th>H1_PX</th><th>H1_PY</th><th>H1_PZ</th></tr><tr><td>0</td><td>1038.634354</td><td>4933.332660</td><td>164858.932313</td></tr><tr><td>1</td><td>-318.157696</td><td>-6407.683029</td><td>152900.152771</td></tr><tr><td>2</td><td>-97.802248</td><td>199.043666</td><td>4381.611081</td></tr></table>		H1_PX	H1_PY	H1_PZ	0	1038.634354	4933.332660	164858.932313	1	-318.157696	-6407.683029	152900.152771	2	-97.802248	199.043666	4381.611081
	H1_PX	H1_PY	H1_PZ														
0	1038.634354	4933.332660	164858.932313														
1	-318.157696	-6407.683029	152900.152771														
2	-97.802248	199.043666	4381.611081														

`df.eval(expression)`

Description	Example																
Evaluate an expression in the context of the DataFrame	<pre>&gt; df['H1_PT'] = data.eval(     'H1_PT = sqrt(H1_PX**2 +     H1_PY**2)' ) df.head(3)</pre> <table><tr><th></th><th>H1_PT</th><th>H1_PX</th><th>H1_PY</th></tr><tr><td>0</td><td>5041.481177</td><td>1038.634354</td><td>4933.332660</td></tr><tr><td>1</td><td>6415.576835</td><td>-318.157696</td><td>-6407.683029</td></tr><tr><td>2</td><td>221.773895</td><td>-97.802248</td><td>199.043666</td></tr></table>		H1_PT	H1_PX	H1_PY	0	5041.481177	1038.634354	4933.332660	1	6415.576835	-318.157696	-6407.683029	2	221.773895	-97.802248	199.043666
	H1_PT	H1_PX	H1_PY														
0	5041.481177	1038.634354	4933.332660														
1	6415.576835	-318.157696	-6407.683029														
2	221.773895	-97.802248	199.043666														

Description	Example
-------------	---------

`df.min()`

Description	Example
Return the minimum value for each column in a dataframe	<pre>&gt; df.min() H1_PX      -122475.373002 H1_PY      -613485.901808 H1_PZ           1420.725768 dtype: float64</pre>

`df.max()`

Description	Example
Return the maximum value for each column in a dataframe	<pre>&gt; df.max() H1_PX      2.411849e+05 H1_PY      1.748288e+05 H1_PZ      1.998913e+07 dtype: float64</pre>

`df.query(expression)`

Description	Example																
Select part of a DataFrame provided expression evaluates to true	<pre>&gt; df_2 = df.query("H1_PX &gt; 0") df_2.head(3)</pre> <table><tr><th></th><th>H1_PT</th><th>H1_PX</th><th>H1_PY</th></tr><tr><td><b>0</b></td><td>5041.481177</td><td>1038.634354</td><td>4933.332660</td></tr><tr><td><b>10</b></td><td>624.982042</td><td>583.983691</td><td>222.633334</td></tr><tr><td><b>11</b></td><td>1789.442985</td><td>1784.378885</td><td>134.529515</td></tr></table>		H1_PT	H1_PX	H1_PY	<b>0</b>	5041.481177	1038.634354	4933.332660	<b>10</b>	624.982042	583.983691	222.633334	<b>11</b>	1789.442985	1784.378885	134.529515
	H1_PT	H1_PX	H1_PY														
<b>0</b>	5041.481177	1038.634354	4933.332660														
<b>10</b>	624.982042	583.983691	222.633334														
<b>11</b>	1789.442985	1784.378885	134.529515														