# MATLAB Reference 3: Descriptive Statistics & Histograms

## Descriptive Statistics

| Command & syntax | Purpose |
|---|---|
| `min(A)` | If A is an array, returns the **minimum value** in the array A. If A is a matrix, returns the minimum value from each column. |
| `max(A)` | If A is an array, returns the **maximum value** in the array A. If A is a matrix, returns the minimum value from each column. |
| `range(A)` | If A is an array, returns the **difference between the minimum and maximum value**s in the array A.<br>If A is a matrix, returns the differences between the minimum and maximum values in each column. |
| `length(A)` | Returns the **number of elements** in the longest dimension of A. |
| `size(A)` | Returns the **number of elements** in each dimension of A. |

### Measures of central tendency of the data, or what is typical of the distribution of the data

| Command & syntax | Purpose |
|---|---|
| `mean(A)` | If A is an array, returns the **average or arithmetic mean**.<br>If A is a matrix, returns the average or arithmetic mean of each column. |
| `median(A)` | If A is an array, returns the **median value**.<br>If A is a matrix, returns the median of each column. |
| `mode(A)` | If A is an array, **most frequently occurring value**.<br>If A is a matrix, returns the most frequently occurring value of each column. |

### Measures of spread of the data

| Command & syntax | Purpose |
|---|---|
| `var(A)` | Returns the sample **variance** of all the values in A<br>**Note**: The variance computed by MATLAB's `var(A)` is different from the variance of Python's `numpy` or `scipy` modules. Read the note below for further information. |

| | |
|---|---|
| `std(A)` | Returns the sample **standard deviation** of all the values in A<br>**Note**: The standard deviation computed by MATLAB's `std(A)` is different from the standard deviation of Python's `numpy` or `scipy` modules. |

**Difference of variance and standard deviation formulas in MATLAB, Python, Excel**: MATLAB's `var(A)` computes the ***sample*** variance: $\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$, where $n$ is the number of samples, $x_i$ is the i-th data point and $\bar{x}$ is the arithmetic mean of the sample. Python's `numpy` and `scipy` modules compute by

`import numpy; numpy.var(A)` or

`import scipy; scipy.var(A)` the ***population*** variance defined by $\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$, where $N$ is the number of all data points in the population and $x_i$ and $\bar{x}$ are as before. Note that the standard deviation is the (positive) square root of the variance. Use `std` instead of `var` in the above code to compute the respective standard deviation.

**Why is there a difference?** The difference comes from the Bessel correction [https://en.wikipedia.org/wiki/Bessel%27s_correction]. In short, if you use the population formula, your result is inaccurate if you look at only a sample. If you e.g. compute the standard deviation for 30 exam scores and there are only 30 students in the class, then you must use the population formulas. If you know there are 30 students in the class, but you only have the grades from 14, then you must use the sample formulas or your result is inaccurate.

**How to compute the sample standard deviation in Python?**

```
import numpy; numpy.std(A,ddof=1) # ddof means degrees of freedom
```

**How to compute the population standard deviation in MATLAB?**

```
std(A,1) % the 1 tells MATLAB to compute the population statistic
```

**How to do it in Excel?**

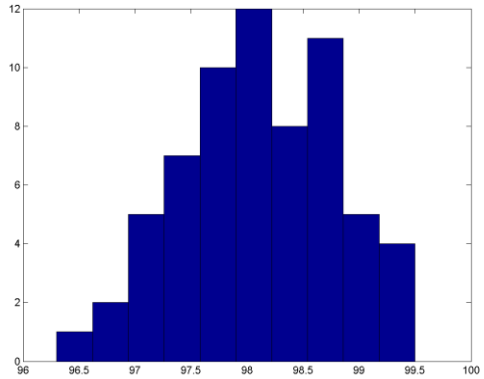
| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 4 | 5 | | | |
| 2 | **=STDEV.S(A1:E1)** | | | | **1.5811** | sample standard deviation | | |
| 3 | **=STDEV.P(A1:E1)** | | | | **1.4142** | population standard devitation | | |
| 4 | **=VAR.S(A1:E1)** | | | | **2.5** | sample variance | | |
| 5 | **=VAR.P(A1:E1)** | | | | **2** | population variance | | |

## Histogram Commands

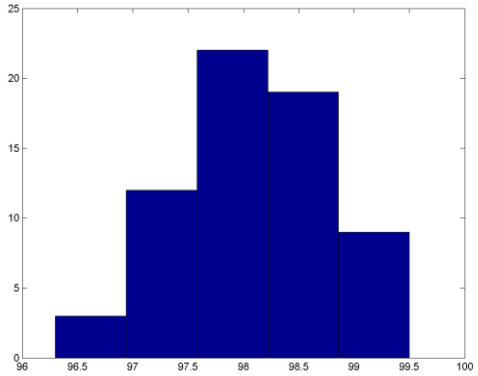
| Command & syntax | Purpose |
|---|---|
| `hist(temps)` | Returns a histogram of the dataset “temps” with 10 bins (default) |
| `hist(temps, X)`, where X is a scalar | Where X is a scalar defining the number of bins, returns a histogram of the dataset “temps” |
| `hist(temps, Y)`, where Y is a vector | Where Y is a vector that defines bin centers, returns a histogram of the dataset “temps” and distributes the data points among bins with centers specified in vector Y |
| `freq = hist(temps)` | Returns the number of data points in each bin for a default of 10 bins |
| `freq = hist(temps, X)` | Where X is the number of bins, returns the number of data points in each bin |
| `freq = hist(temps, Y)` | Where Y is a vector that defines bin centers, returns the number of data points in each bin |
| `[freq, centers] = hist(temps, X)` | Where X is the number of bins, returns the number of data points in each bin (freq)<br>AND<br>Returns the values of the bin centers (centers) |
| `centers_manual = [start: increment: end]`<br>`hist(male_temps, centers_manual)` | Manually define bin centers |

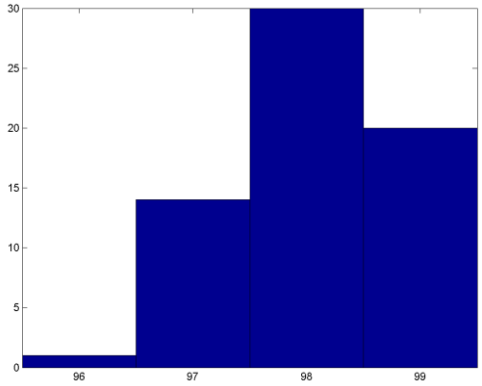
**Example Histograms and Commands**

```
hist(data)      % generate default histogram
```




```
nbr_bins=5          % specify number of bins
hist(data,nbr_bins) % histogram with a specified number of bins
```



| | |
|---|---|
| `centers =[96,97,98,99]  % specify bin centers`<br>`hist(data,centers)  % histogram with a specified bin`<br>`% centers` |  |
| `freq=hist(data,nbr_bins)  % determine # of data points in each`<br>`% bin with number of bins specified` | `freq=[3  12  22  19  9]` |
| `freq=hist(data,centers)  % determine # of data points in each`<br>`% bin with bin centers specified` | `freq=[1  14  30  20]` |
| `[freq, centers] = hist(data, nbr_bins)  % determine # of data`<br>`% points in each bin and`<br>`% the bin centers with`<br>`% the number of bins`<br>`% specified` | `centers=`<br>`[96.62 97.26 97.9 98.54 99.18]`<br>`freq=[3 12 22 19 9]` |