PROGRAMMING WITH MATLAB

WEEK 10

BASIC STATISTICS

BASIC STATISTICS

- In general, we can write a data set of k values as follows:
$$x = \{x_1, x_2, x_3, \dots x_k\}$$
- In MATLAB, this set is represented by a row vector x.
- Statistics can be used to identify certain properties of this data.
- Take, for example, a set of exam grades: $\{27, 55, 63, 72, 72, 77, 79, 83, 95, 97, 100\}$
- We may want to know the average value (74.5455), the middle value (77), or the most frequently occurring value (72) to make various assessments over the exam grades.
- In addition, it is useful to know how the data values in a data set are spread.

BASIC STATISTICS

- MATLAB has many built-in functions for statistics.
- In simplest terms, to find the minimum or maximum value in a data set:

```
>> x = [13 11 -5 3 39 17 7 27 19 23];
>> min(x)
ans =
   -5
>> max(x)
ans =
   39
```

These functions also return the index of the largest or smallest value:

```
>> [maxVal, maxInds] = max(x)
maxVal =
   39
maxInds =
    5
```

BASIC STATISTICS

- In the case of matrices, these functions operate columnwise by default:

```
>> A = [1 11 5; 7 3 23; 17 19 13]
A =
     1    11     5
     7     3    23
    17    19    13
>> [minVal, minInds] = min(A)
minVal =
     1     3     5

minInds =
     1     2     1
```

BASIC STATISTICS

- To find the minimum value of each row, the dimension of 2 is specified as a third argument:

```
>> min(A, [ ], 2)

ans =

     1

     3

    13
```

Mean: The arithmetic mean of a data set, in other words the average of the values, is obtained by dividing the sum of the values by the number of values in the data set.

$$\frac{\sum_{i=1}^{k} x_i}{k}$$

BASIC STATISTICS

- The calculation of a mean would be programmed by adding elements of a vector together using a loop, and then dividing this sum by the number of elements:

```
>> x = [13    11    -5     3    39    17     7    27    19    23];
>> sumVal = 0;
for i=1:length(x)
sumVal = sumVal + x(i);
end
>> meanVal = sumVal/length(x)
meanVal =
   15.4000
```

or we simply use the mean function:

```
>> mean(x)
ans =
   15.4000
```

BASIC STATISTICS

- Sometimes a value that is much larger or smaller (which is called an outlier) than the rest of the data may cause our interpretations based on 'mean' to be misleading:

```
>> x = [2 11 5 3 121 9 13 7 8 15 1];
>> mean(x)
ans =
   17.7273
```

In this case, the average can be calculated after discarding the minimum or maximum values from the data set:

```
>> xnew = x(x<max(x))
xnew =
     2    11     5     3     9    13     7     8    15     1
>> mean(xnew)
ans =
    7.4000
```

BASIC STATISTICS

- Variance:

$$var(x) = \frac{\sum_{i=1}^{k}(x_i - mean(x))}{k-1}$$

Syntax: for vectors, y = var(x) returns the variance of the values in x. For matrices, y is a row vector containing the variance of each column of x.

```
>> x = [2 11 5 3 21 9 13 7 8 15 1];
>> var(x)
ans =
   36.8545
```

The standard deviation is the square root of the variance.

```
>> std(x)
ans =
    6.0708
```

BASIC STATISTICS

- Mode: the most frequent value in a data set

Syntax: m=mode(x) for vector x computes m as the sample mode, or most frequently occurring value in x. For a matrix x, m is a row vector containing the mode of each column.

```
>> x = [27, 55, 63, 72, 72, 77, 79, 83, 95, 97, 100];
>> mode(x)
ans =
    72
```

If no value is seen more often than any other, the smallest value will be the mode of that vector.

```
>> x = [1 3 5 7];
>> mode(x)
ans =
     1
```

BASIC STATISTICS

- Median: The median is defined only for a data set that has been sorted first.

Syntax: For vectors, median(x) is the median value of the elements in x. For matrices, median(x) is a row vector containing the median value of each.

```
>> x = [27, 55, 63, 72, 72, 77, 79, 83, 95, 97, 100];
>> median(x)
ans =
   77
```

If the vector is not already sorted, the median function will automatically sort and give the correct result.

For the following vector, the mean of 63 and 72 in the middle will be returned

```
>> x = [27, 55, 63, 72, 72, 77];
>> median(x)
ans =
  67.5000
```

BASIC STATISTICS

- Sorting: Sorting in ascending or descending order.

Syntax: For vectors, sort(x) sorts the elements of x in ascending order. For matrices, sort(x) sorts each column of x in ascending order. Sort order can be specified.

```
>> x = [17, 1, 7, 11, 3, 13, 19];
>> sort(x)
ans =
    1    3    7   11   13   17   19
>> sort(x, 'descend')
ans =
   19   17   13   11    7    3    1
```

BASIC STATISTICS

- Random Number Functions: Random numbers are useful for a variety of purposes, such as generating data encryption keys, simulating and modeling processes involving randomness, and selecting random samples from larger data sets. Also, if a program is being written and the data is not yet available, it is useful to test the program first by initializing the data to random numbers. There are several built-in functions for generating random numbers in MATLAB.

rand: Uniformly distributed pseudorandom numbers.

Syntax: r= rand(n) returns an nxn matrix containing pseudorandom values drawn from the standard uniform distribution on the open interval(0,1). rand(m,n) returns an mxn matrix.

```
>> rand

ans =

   0.8147

>> rand(3)

ans =

   0.9058    0.6324    0.5469

   0.1270    0.0975    0.9575

   0.9134    0.2785    0.9649
```

BASIC STATISTICS

randn: Normally distributed pseudorandom numbers.

Syntax: r= randn(n) returns an nxn matrix containing pseudorandom values drawn from the standard normal distribution. randn(m,n) returns an mxn matrix.

We can also generate random numbers with a different mean and standard deviation.

For example, to generate a vector x containing 10 random numbers normally distributed with a mean of 7 and a standard deviation of 3:

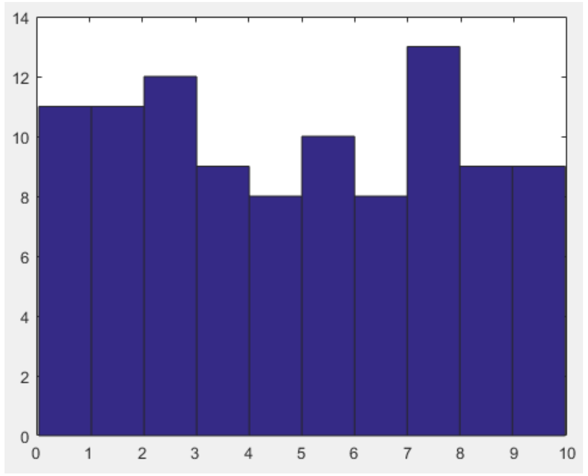
```
>> x = 3*randn(1,10) + 7
x =
  Columns 1 through 8

    2.9503   16.1048    9.1762    6.8108    9.1442    6.3851    6.6276   11.4691

  Columns 9 through 10

   11.2271   11.2516
```

BASIC STATISTICS

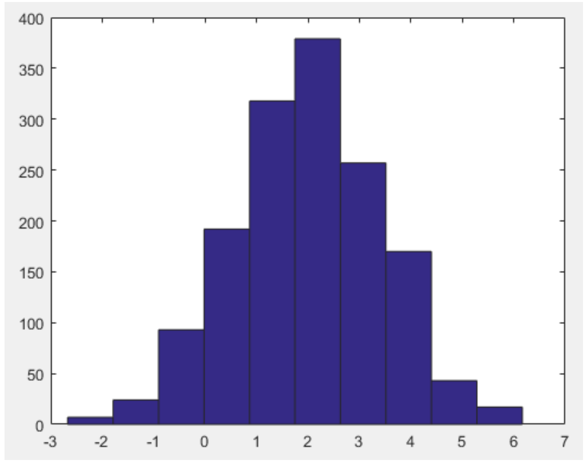
- Histogram: A histogram is a particular type of bar chart that shows the frequency of occurrence of values within a vector. Histograms use what are called bins to collect values that are in given ranges.

Syntax: n = hist(x) bins the elements of x into 10 equally spaced containers and returns the number of elements in each container. If x is a matrix, hist works down the columns.

```
>> x = 10*rand(1,100);
>> hist(x)
```



```
>> x = 1.5*randn(1,1500) + 2;
>> hist(x)
```



0
50
100
150
200
250
300
350
400
-3
-2
-1
0
1
2
3
4
5
6
7