Discrete Structures

Faisal Bukhari, PhD
Associate Professor
Department of Data Science, Faculty of Computing and Information Technology, University of the Punjab

Textbooks

- **Probability & Statistics for Engineers & Scientists**, Ninth Edition, Ronald E. Walpole, Raymond H. Myer

- **Elementary Statistics: Picturing the World,** 6th Edition, Ron Larson and Betsy Farber

- **Elementary Statistics,** 13th Edition, Mario F. Triola

Reference books

- **Probability Demystified**, Allan G. Bluman
- **Schaum's Outline of Probability and Statistics**
- **MATLAB Primer**, Seventh Edition
- **MATLAB Demystified** by McMahon, David

Reference

Readings for these lecture notes:

MATLAB Demystified, David McMahon

MATLAB® Primer, Seventh Edition, Timothy A. Davis Kermit Sigmon

Elementary Statistics PICTURING THE WORLD by **Ron Larson and Betsy Farber**

https://www.blackjackinfo.com/knowledge-base/blackjack-theory-and-math/a-question-for-the-statistics-experts/

http://math.stackexchange.com/questions/598808/if-you-roll-a-fair-six-sided-die-twice-whats-the-probability-that-you-get-the

These notes contain material from the above resources.

Referencing individual entries

Individual matrix and vector entries can be referenced with indices inside parentheses. For example, A(2,3) denotes the entry in the second row, third column of matrix A.

A = [1 2 3 ; 4 5 6 ; -1 7 9]

A(2,3)

Create a column vector, x, with:

x = [3 2 1]’

or equivalently:

x = [3 ; 2 ; 1]

Relational operators

The relational operators in MATLAB are:

- < less than
- > greater than
- <= less than or equal
- >= greater than or equal
- == equal
- ~= not equal

Note that = is used in an assignment statement whereas == is a relational operator.

Logical operators:

Relational operators may be connected by **logical operators**:

- & and
- | or
- ~ not
- && short-circuit and
- || short-circuit or

fix()

fix(X) rounds the elements of X to the nearest integers towards zero.

Examples of fix()

```
>> fix(5.5)

ans =

     5

>> fix(5.9)

ans =

     5
```

Example: ***Intervals*** on the real line, defined below, appear very often in mathematics. Here ***a*** and ***b*** are real numbers *with* $a < b$.

Open interval from ***a*** to ***b*** = $(a,b) = \{x : a < x < b\}$

Closed interval from ***a*** to **b** = $[a,b] = \{x : a \leq x \leq b\}$

Open-closed interval from ***a*** to ***b*** = $(a,b] = \{x : a < x \leq b\}$

Closed-open interval from ***a*** to **b** = $[a,b) = \{x : a \leq x < b\}$

The **open-closed** and **closed-open** intervals are also called ***half*-open**

rand()

rand(): returns an n-by-n matrix containing pseudorandom values drawn from the standard uniform distribution on the **open interval** (0,1).

Example 1 of rand()

```
>> n = rand(1,10)
```

n =

Columns 1 through 3

0.4218 0.9157 0.7922

0.1576 0.9706 0.9572

Columns 4 through 6

0.4854 0.8003 0.1419

Columns 7 through 9

Column 10

0.9595

Example 2 of rand()

```
>> n = fix(10*rand(1,10))

n =

  Columns 1 through 6

    8    9    1    9    6    0

  Columns 7 through 10

    2    5    9    9
```

Simulation

- A **simulation** is the use of a **mathematical or physical model** to **reproduce the conditions** of a situation or process. Collecting data often involves the **use of computers**.
- Simulations allow you to study situations that are **impractical or even dangerous** to create in real life, and often they save time and money.
- For instance, **automobile manufacturers** use **simulations with dummies** to study the effects of **crashes on humans**.

Simulation of coin tosses [1]

Question: Simulate the outcomes of 1000 biased coin tosses with **p[Head] = 0.3**

Solution:

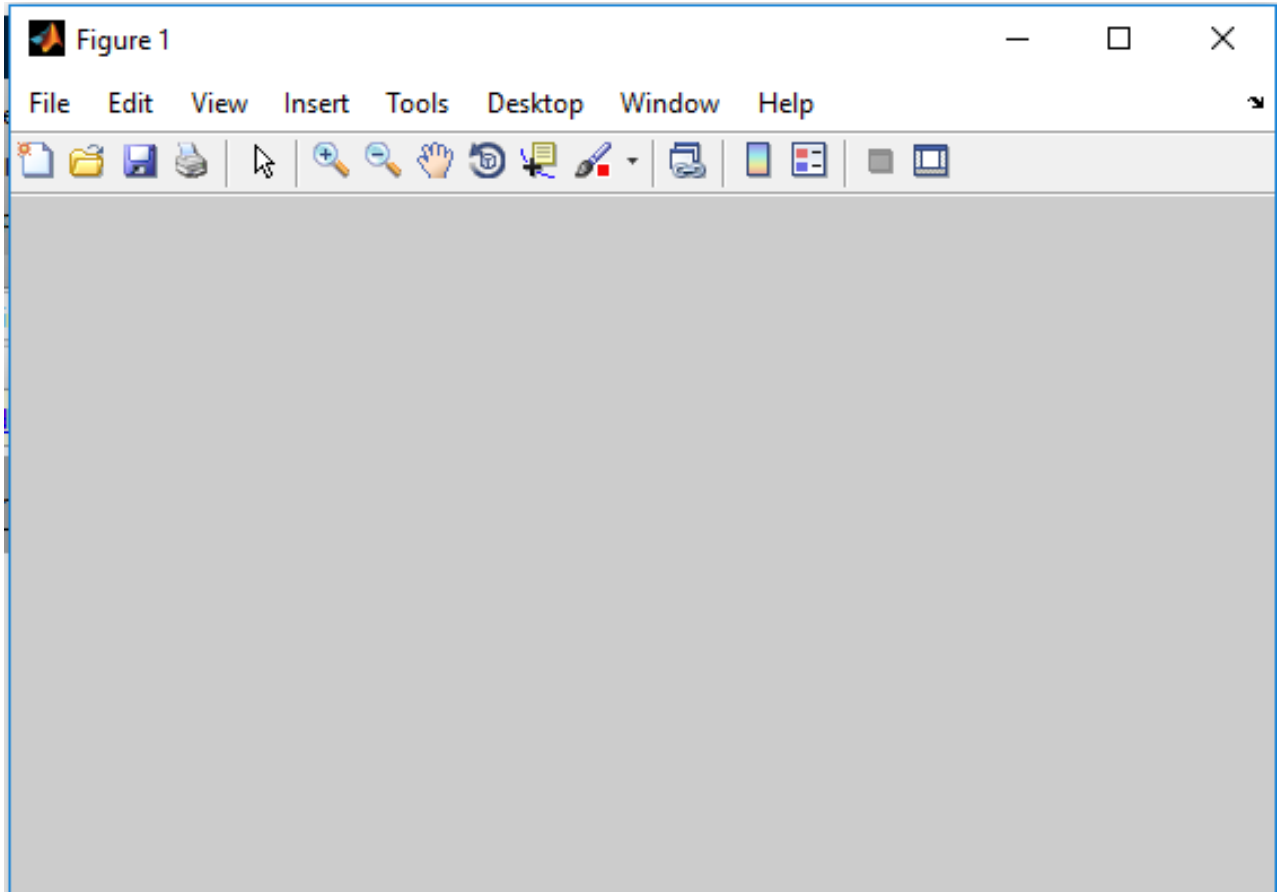
```
randomNumber = rand(1000,1);
% Generate 1000 random numbers stored  in column vector

heads = randomNumber <= 0.3;
% Number of heads which meet the above condition

totalNumberOfHeads = sum(heads);
% Total number of heads

probabilityOfHeads = totalNumberOfHeads /1000;
% Probability of heads
```

figure

figure: opens up a new figure window

Example of figure command

>> figure



hold on vs. hold off

hold on: holds current plot

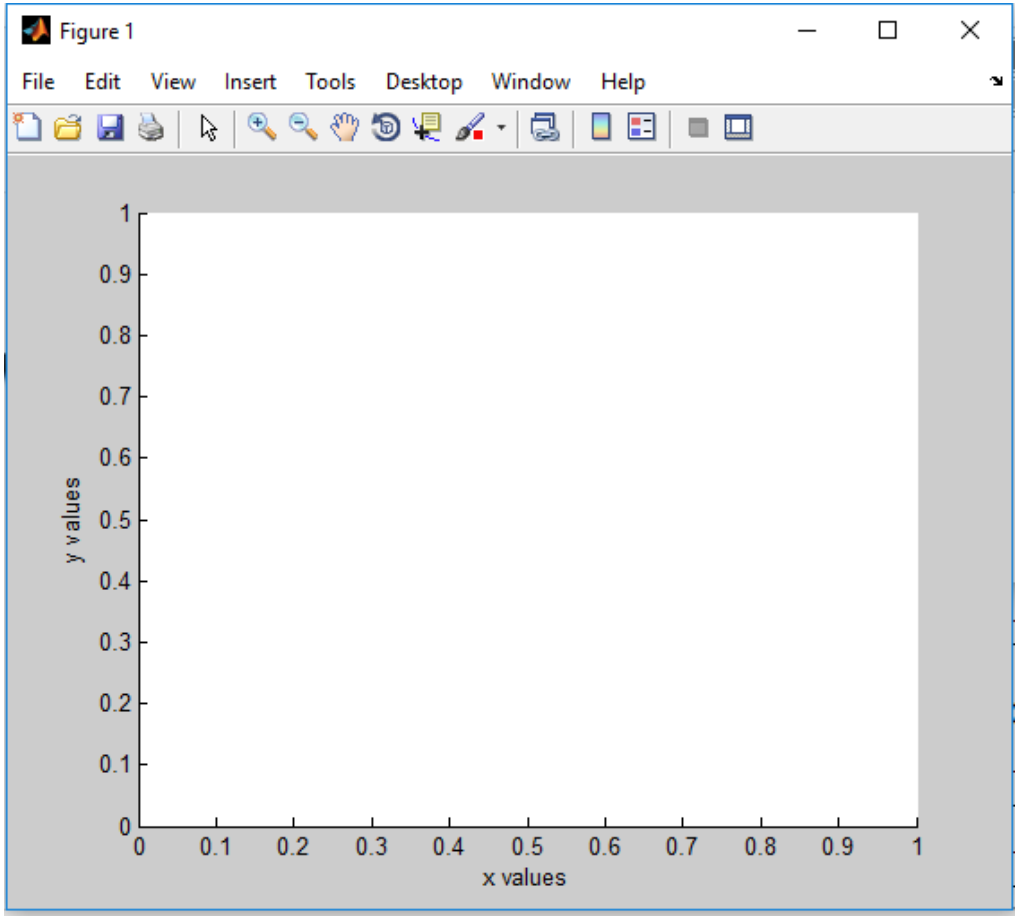
hold off: releases current plot

xlabel vs. ylabel

xlabel: Labels the x-axis

ylabel: Labels the y-axis

```
>> figure
>> hold on
>> xlabel('x values')
>> ylabel('y values')
```

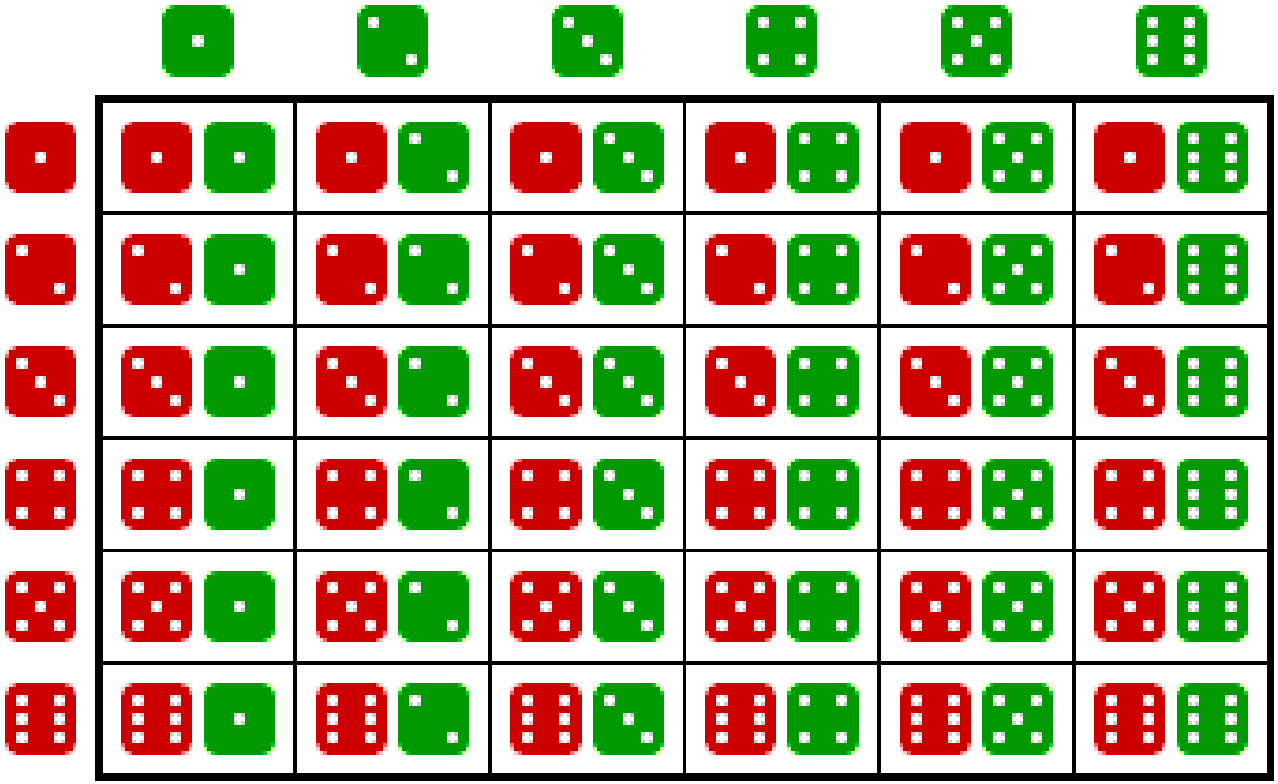


bar()

bar: Bar graph.

bar(X,Y) draws the columns of the M-by-N matrix Y as M groups of N vertical bars. The vector X must not have duplicate values.

Simulation of the sum of two fair dice [1]



Simulation of the sum of two fair dice [2]

Simulate the sum of outcome of two dice, when two are rolled 10, 000 times.

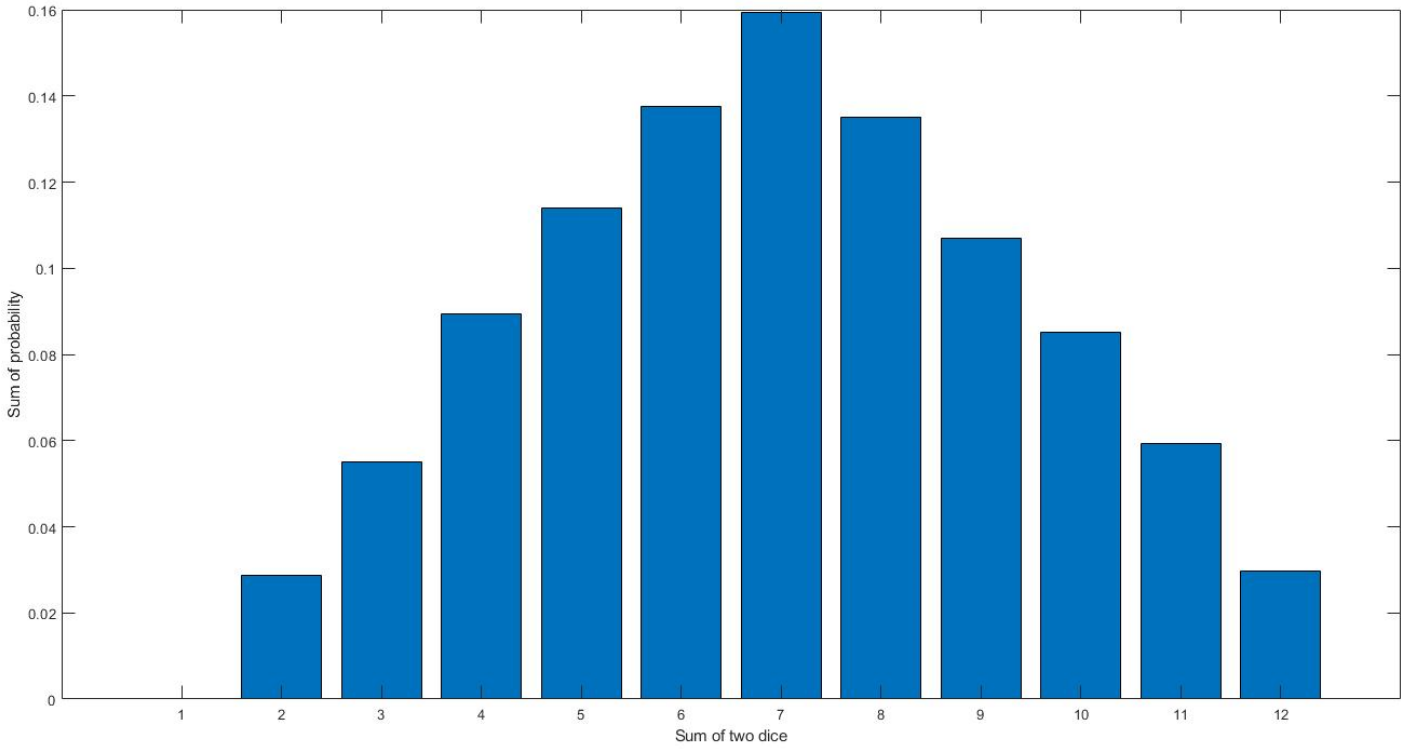
		White Die					
		1	2	3	4	5	6
Red Die	1	(1,1)	(2,1)	(3,1)	(4,1)	(5,1)	(6,1)
	2	(1,2)	(2,2)	(3,2)	(4,2)	(5,2)	(6,2)
	3	(1,3)	(2,3)	(3,3)	(4,3)	(5,3)	(6,3)
	4	(1,4)	(2,4)	(3,4)	(4,4)	(5,4)	(6,4)
	5	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)	(6,5)
	6	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)

```
% create die 1
Die1 = floor(6 * rand(10000, 1) + 1 );
% create die 2
Die2 = floor(6 * rand(10000, 1) + 1);
% sum of 2 dice
SumOfDice = Die1 + Die2;
% check if sum is 2
D2 = SumOfDice == 2;
% compute probability of 2
probD2 = sum(D2)/10000;
D3 = SumOfDice == 3;
probD3 = sum(D3) /10000;
```

```
D4 = SumOfDice == 4;

probD4 = sum(D4) /10000;

D5 = SumOfDice == 5;

probD5 = sum(D5) /10000;

D6 = SumOfDice == 6;

probD6 = sum(D6) /10000;

D7 = SumOfDice == 7;

probD7 = sum(D7) /10000;
```

```
D8 = SumOfDice == 8;

probD8 = sum(D8) /10000;

D9 = SumOfDice == 9;

probD9 = sum(D9) /10000;

D10 = SumOfDice == 10;

probD10 = sum(D10) /10000;

D11 = SumOfDice == 11;

probD11 = sum(D11) /10000;
```

```
D12 = SumOfDice == 12;

probD12 = sum(D12) /10000;

probD1 = 0;
p = [probD1 , probD2, probD3, probD4,
  probD5,    probD6, probD7,  probD8,
  probD9, probD10, probD11, probD12 ]';

bar(p)
hold on
xlabel('Sum of two dice')
ylabel('Probability')
```


0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
Sum of probability
1
2
3
4
5
6
7
8
9
10
11
12
Sum of two dice