Syllabus and Introduction

ECE 341: Random Processes

North Dakota State University

Instructor: Jacob Glower

note: All lecture notes, homework sets, and solutions are posted on www.BisonAcademy.com

ECE 341: Random Processes

Instructor: Jacob Glower
Office Location: at home...
Office Phone: none at present
Class Hours: noon-2:30pm Monday - Friday (also on zoom)
Office Hours: 7-8pm on Zoom
Lectures: YouTube videos
Text: Posted on Bison Academy
Other References

- Probability and Stochastic Processes, Yates ($6 used)
- Probability and Statistics, DeGroot, ($6 used) - recommended
- Principles of Statistics, Bulmer ($6 used)
- A First Course on Stochastic Processes, Karlin and Taylor ($11 used)
- Introduction to Statistics: (Univ British Columbia free!)

Grading

- Midterms: 1 unit each
- Homework & Quizzes 1 unit
- Total: Average of all above

Final Percentage:

- 100% - 90% A
- 89% - 80% B
- 79% - 70% C
- 69% - 60% D
- < 59% E

Grading will be on a straight scale to encourage working together

- Just not on tests (please)

Homework: Homework is graded as

- 80%: You attempted the problem with an organized approach I can follow.
- 20%: You got the right answer.

Homework is due on a daily basis. Solutions will be posted and we'll go over the homework the following day.

Testing:

- Tests will be posted at 8am, due 7am the following day.
- 2 hour time limit from when you start
- Each test will be different (each student generates random numbers for the test)
- All tests are open-book, open notes, calculators, Matlab all permitted
- Just not other people.
    - Providing or receiving help from others prohibited
    - Posting test on-line and/or using an on-line solution prohibited

What is a random process?

- A repeatable event
- The outcome changes each time

Not a Random Process	Random Process
Can I write a program to play poker? • Either yes or no Did it rain last April? • past event: yes or no Will the Vikings beat the Packers? • not a repeatable event	How long will it take me to write a program similar to playing poker? • varies: 10 to 1000 minutes How much will it rain in April? • varies each year

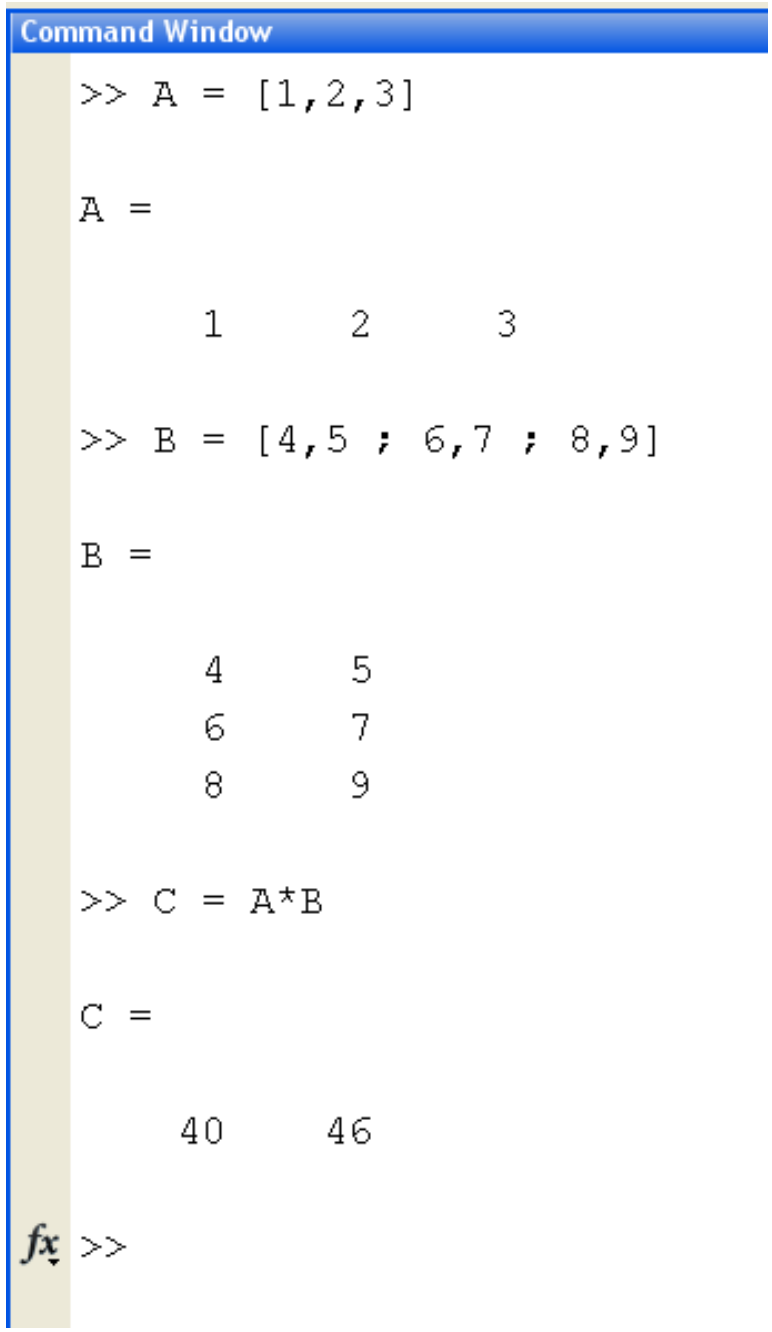
Matlab Review

Matlab can be treated like a calculator that works with matrices

```
[       start of matrix
]       end of matrix
,       next column
;       next row

+       addition
-       subtraction
*       multiplication
/       division

inv(A)  matrix inverse
```



Flow Control

For-Loop

```
for i=1:10
   t = t + dt;
   end
```

While-Loop

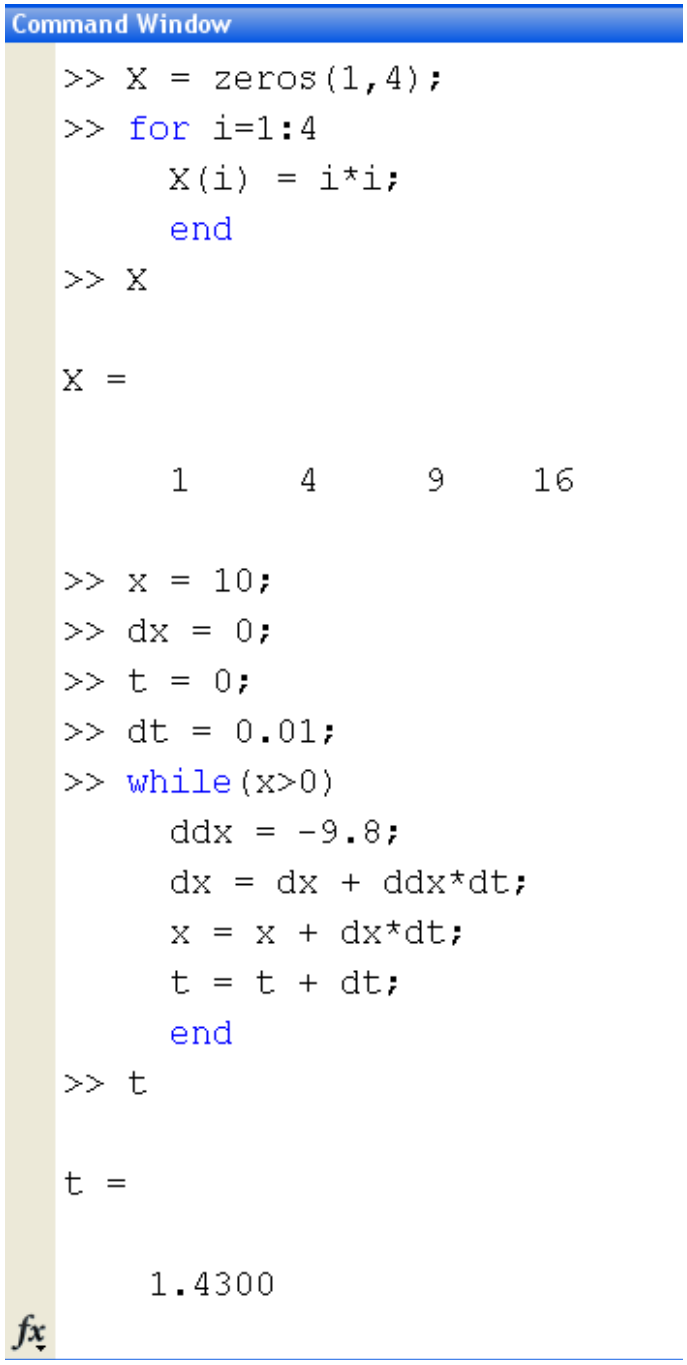
```
time = 0;
while(time < 10)
   x = x + dx*dt;
   t = t + dt;
   end
```

If

```
if(time < 10)
   x = 0;
   end
```

If-Else

```
if(x>y)
   points = points + 1;
elseif(x == y)
   points = points + 0.5;
else
   points = points + 0;
end
```



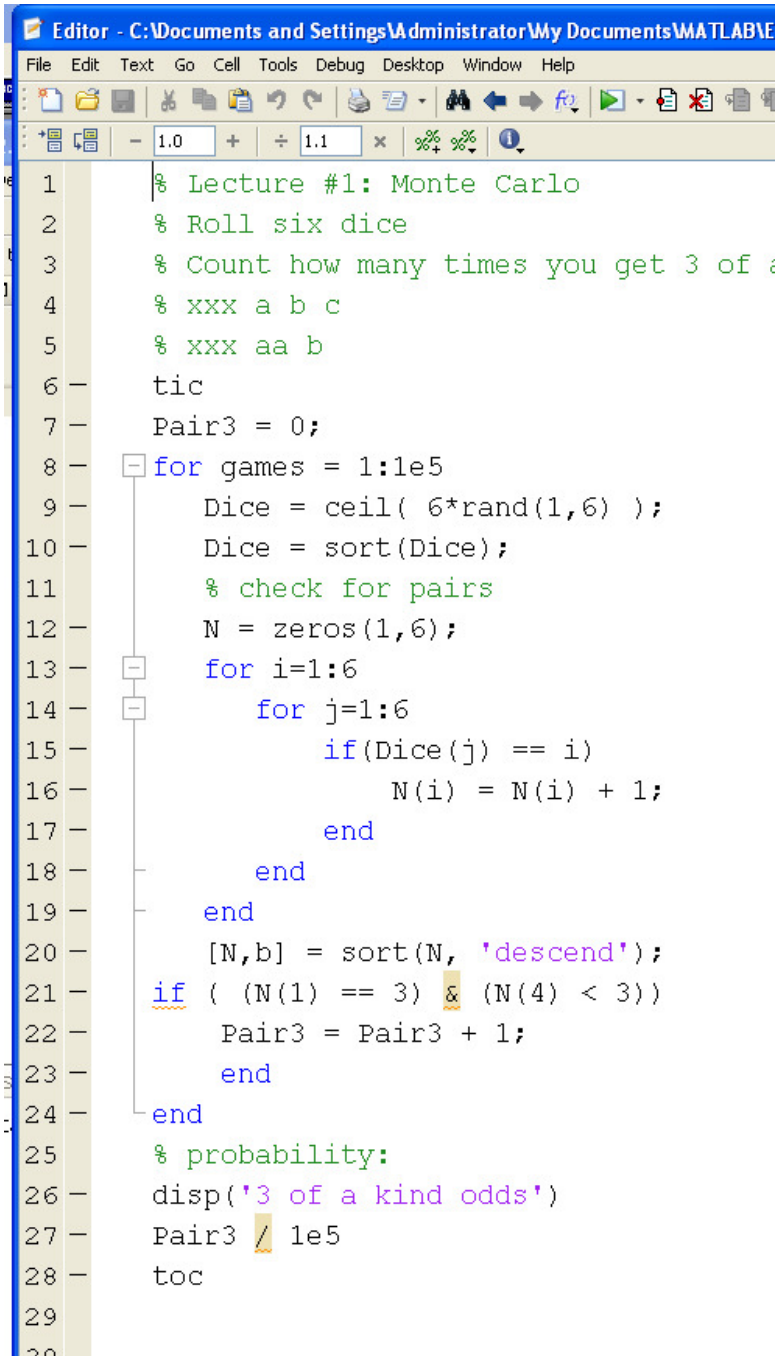
Matlab Scripts

If you're going to run the same code over and over, you can place it in a script.

Each time you execute the script, it's like pasting that code into the command window.

note: This is a convenient way to build a more complex program.

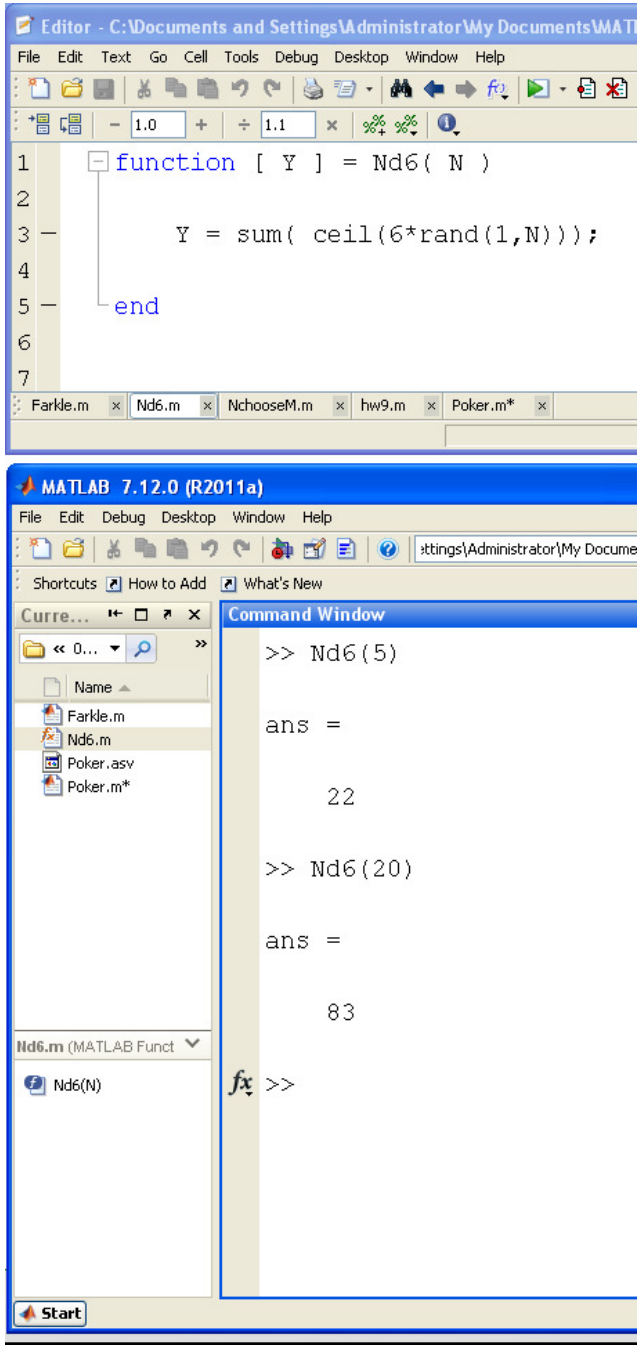
- Display the data when writing the program to see what is happening
- You can fix errors along the way



Matlab Functions

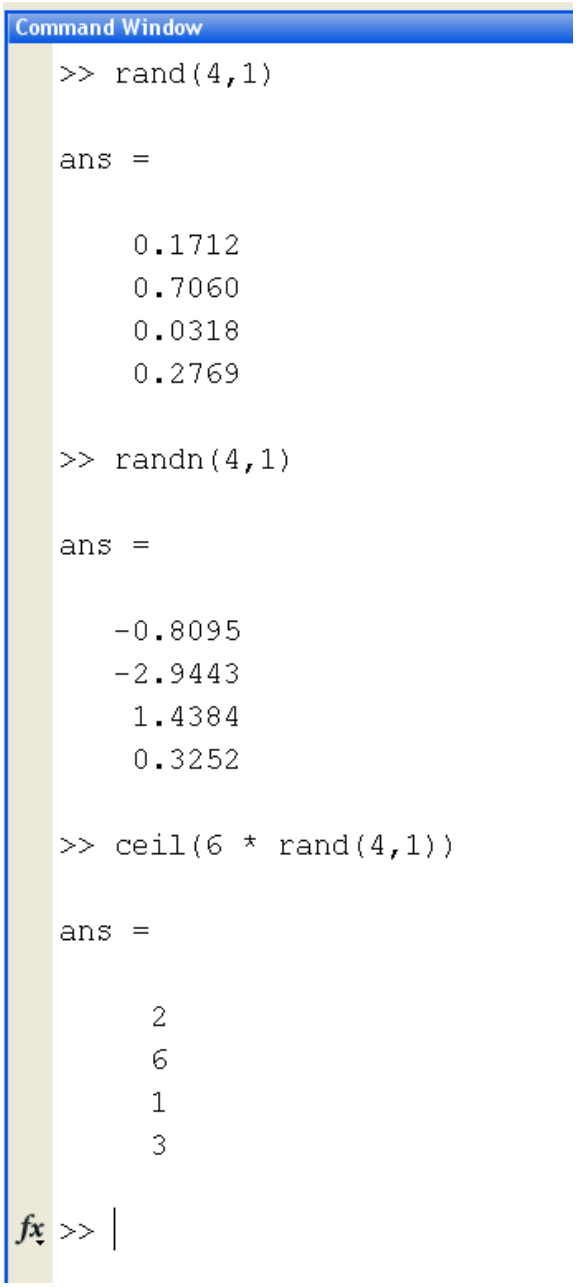
Part of what makes Matlab so powerful is you can create your own functions

- These functions become Matlab commands that other functions can use.
- As companies build up their library of Matlab functions, they get better and better at designing their product.
- The Matlab functions become company proprietary information (design secrets).



Random Numbers in Matlab

- rand(4,1): generate a 4x1 matrix of random numbers in the range of (0,1)
- randn(4,1) generate a 2x3 matrix of normally distributed random numbers
- ceil(6 * rand(4,1)): generate four 6-sided dice (4d6)



Problem:

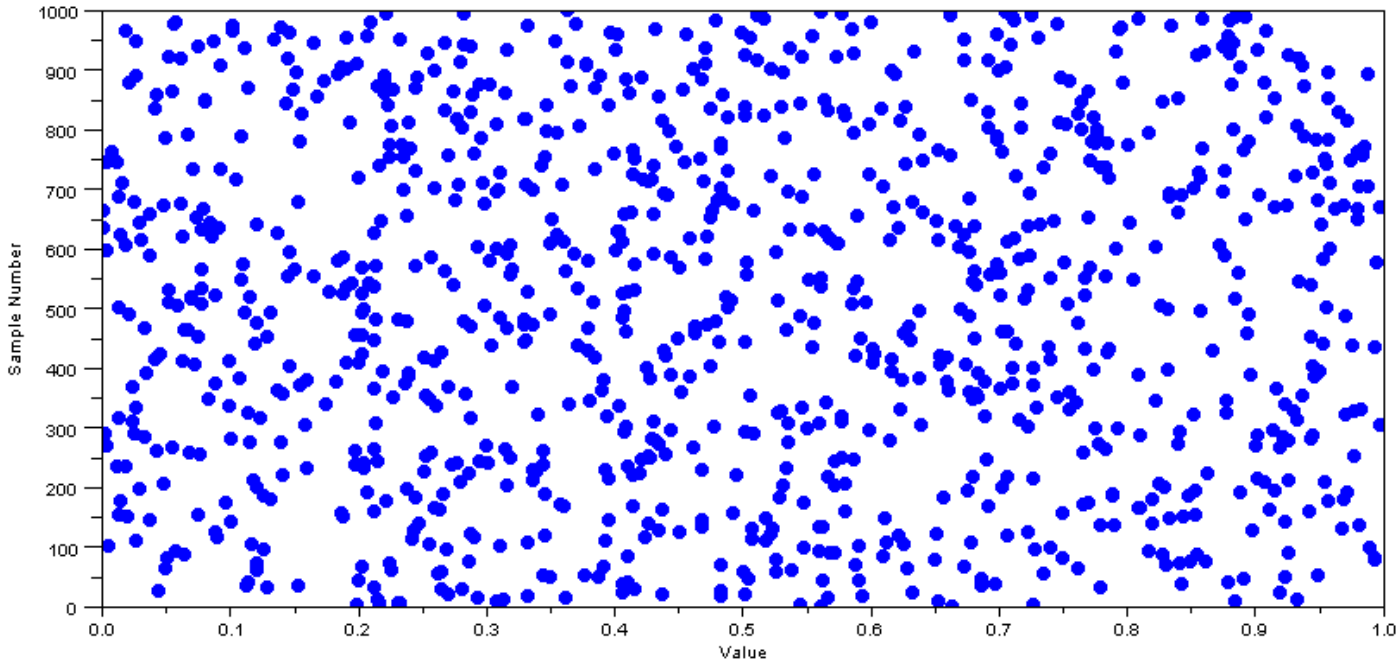
- How would you mathematically describe the rand() function in MATLAB?
- How would you mathematically describe any random process?

Monte Carlo simulation

- Take a large number of samples
- As the sample size go to infinity, you fully describe the random process.
- Ex: take 1000 samples:

```
X = rand(1000,1);
Die = ceil( 6 * rand(1000,1) );
```

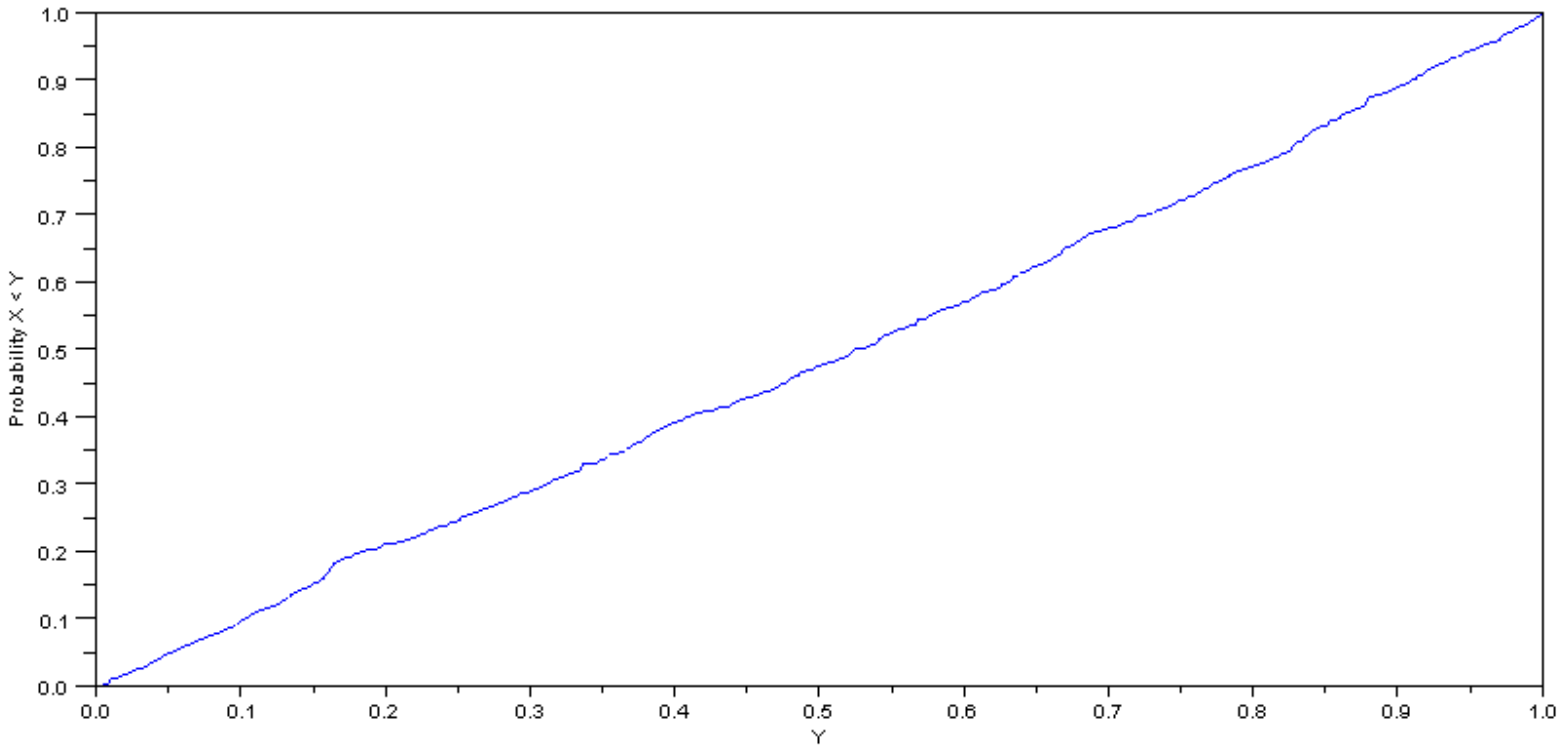
List 1000 numbers isn't very useful. Instead, lets plot the data:



Again, this doesn't tell you much.

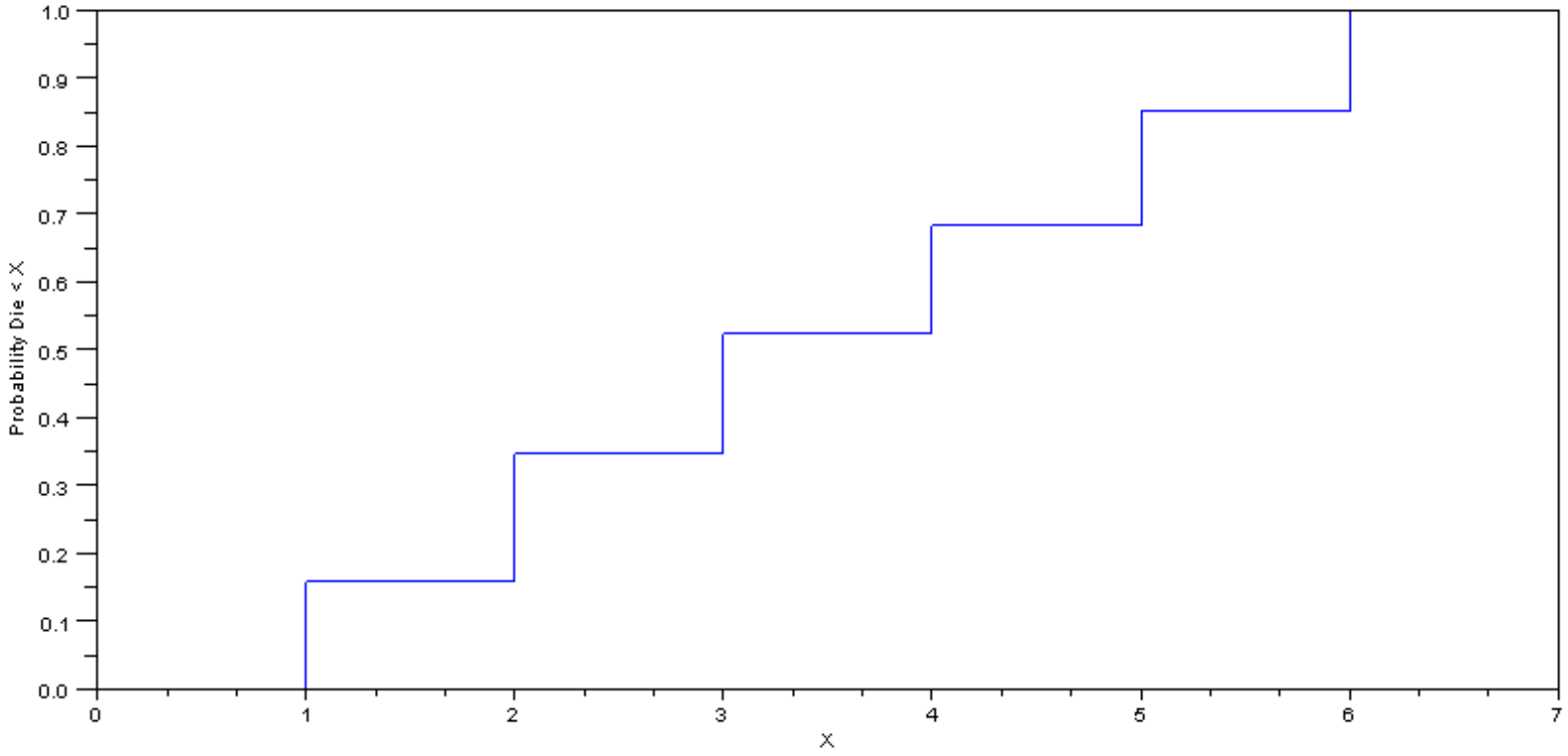
Sort the data and plot (cumulative distribution function or cdf):

```
Xs = sort(X);
p = [1:length(Xs)]' / length(Xs);
plot(Xs, p)
xlabel('Y');
ylabel('Probability X < Y')
```



Discrete CDF (dice)

```
d6 = ceil( 6*rand(100,1) );
d6 = sort(d6);
p = [1:length(d6)]' / length(d6);
plot(d6, p)xlabel('X');
ylabel('Probability Die < X');
```

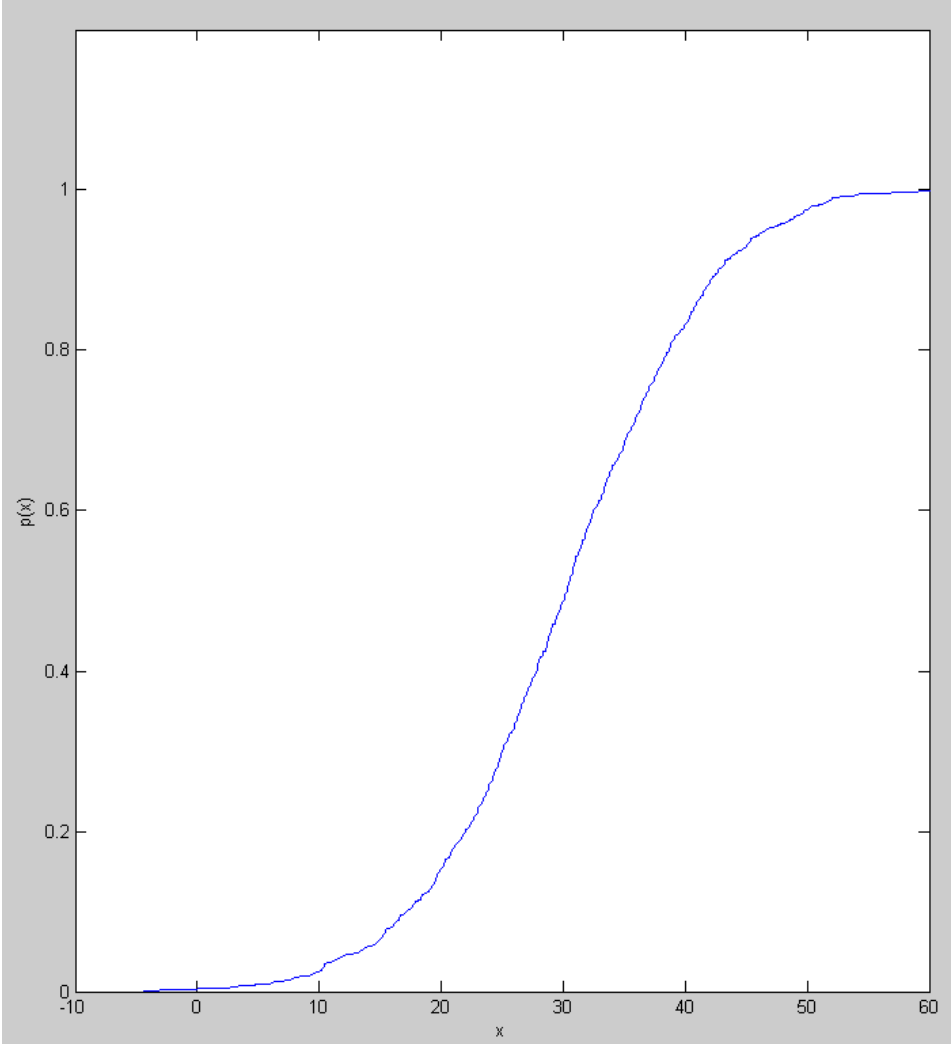


Cumulative Distribution Function

$$f(x) = p(y < x)$$

Note that for a CDF,

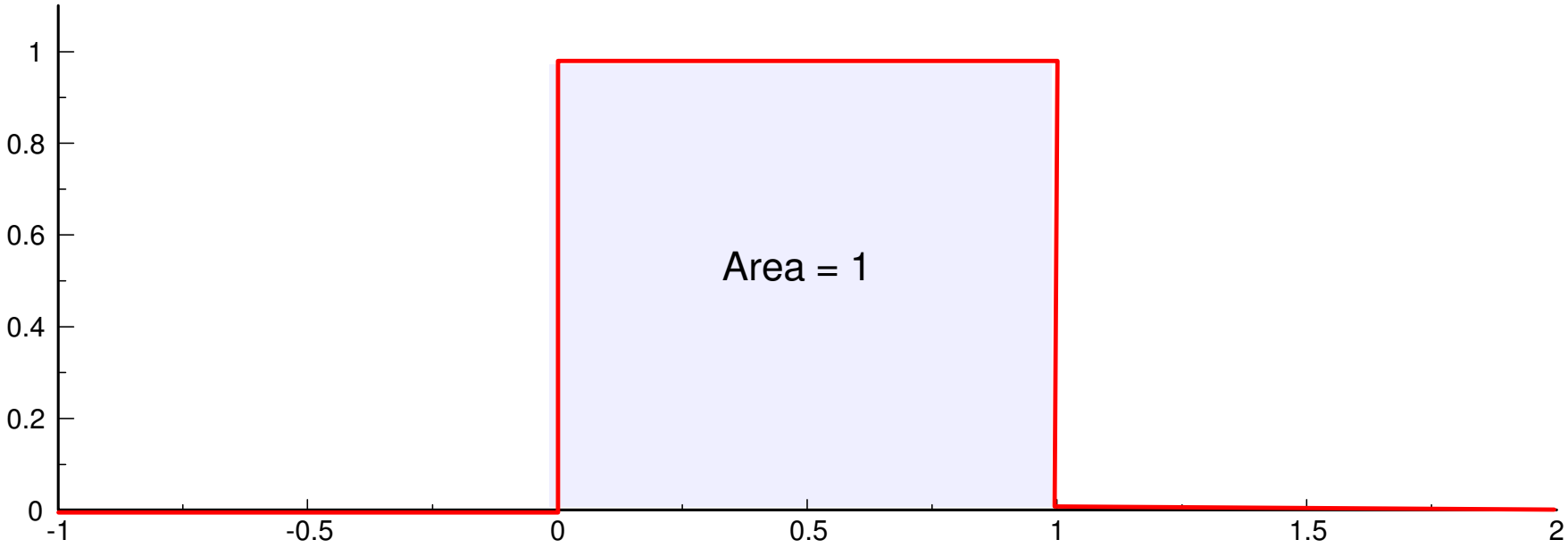
- It starts at zero (the probability that the result is less than minus-infinity is zero. It has to be something.)
- It goes to one (the probability that some number results is one.)



Probability Density Function (pdf)

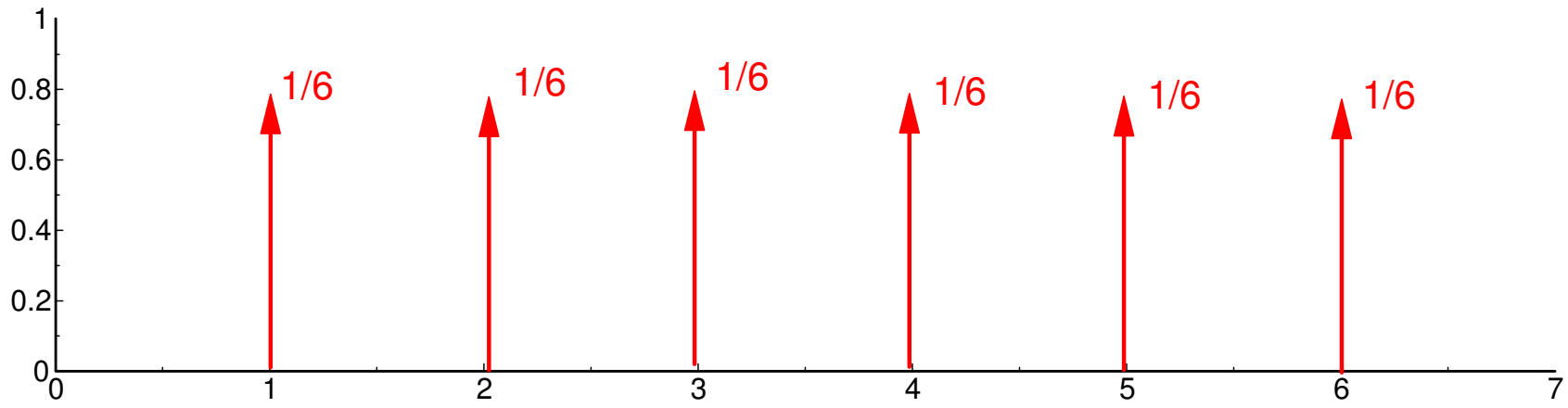
- Derivative of cdf
- Probability that x has a certain value

Example: rand is a uniform distribution over the interval (0, 1)



Probability Density Function for a uniform distribution

Discrete pdf:



Probability Density Function for a 6-sided die

This is the first concept to get across in this class:

- pdf describes the probabilty distribution
- cdf also describes the probability distribution

We'll look at some standard distributions:

- Bernoulli Trial: Flip a coin
- Binomial Distribution: Flip N coins
- Geometric Distribution:
  Flip a coin until you get a heads
- Pascal Distribution: Flip a coun until you get N heads
- Hyper Geometric: Geometric without replacement
- Poisson: Number of events in a time interval
- Normal (Gaussian) Binomial distribution where N goes to infinity

Mean, Standard Deviation, Moments

Problem: Describe a probability distribution with a number

A second concept to get across is that simply giving the mean of a random process doesn't tell you much. You need more information.

- pdf or cdf preferred
- Mean (average) and standard deviation (spread) OK
- Moment Generating Function (LaPlace transform or z-transform)
  - m0 = 1 0th moment (total probability must be 1)
  - m1 = average(X) 1st moment
  - m2 = average(X^2) 2nd moment
  - etc.

Third, suppose you have a random process and you'd like to test a hypothesis, such as

- Does Y have a mean that's greater than X?
- Is X a uniform distribution?

The first question results in comparing the means of the samples with the spread (standard deviation) taken in to account. This is a t-test.

The second question results in comparing the sample PDF or CDF's and results in a chi-squared distribution.

Fun with Monte-Carlo Simulations

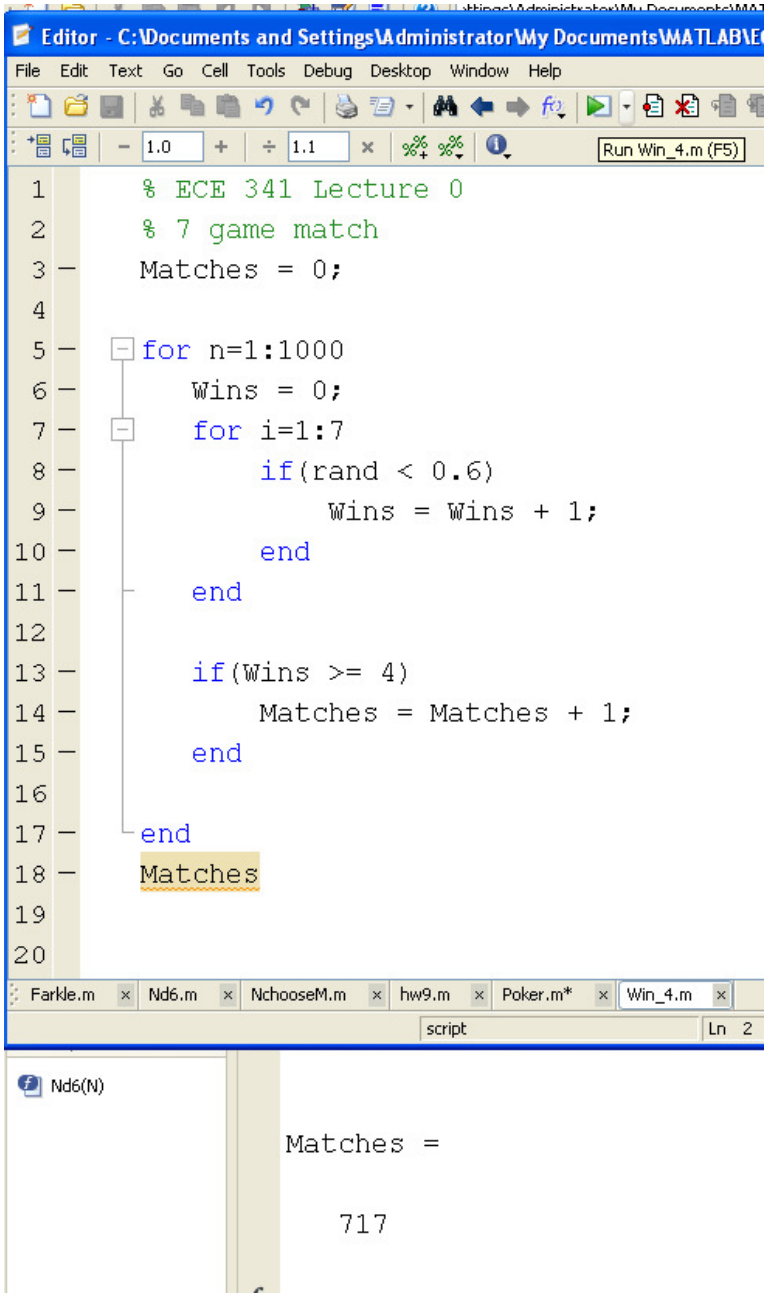
Problem: Two teams playing a match

- Team A has 60% chance of winning any given game
- The first team to win 4 games wins the match (best of 7 series)

What is the probability that team A wins the match?

Solution

- Play a single match (lines 6..14)
- Note this is a for-loop
- Then repeat 1000 times
- Team A has a 71.7% chance of winning



More Fun with Monte Carlo

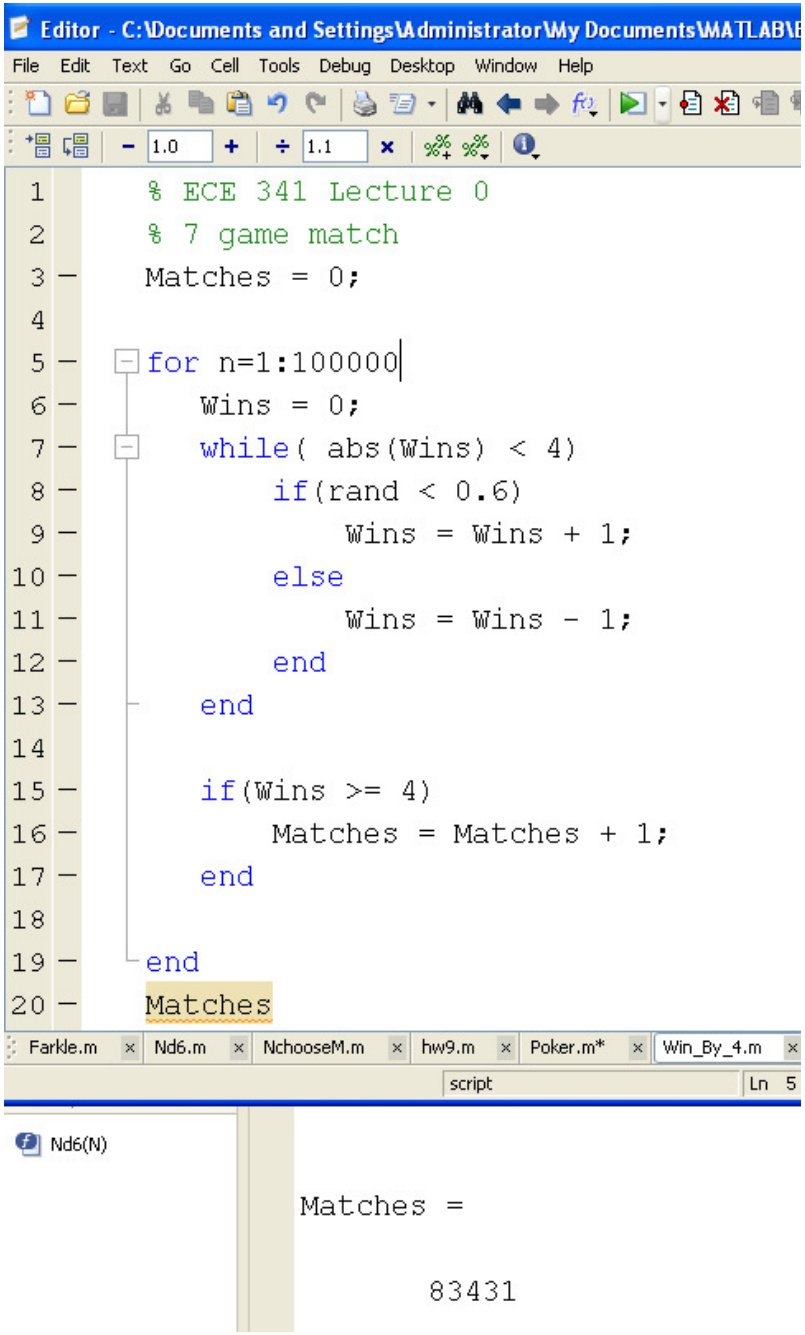
Problem: Two teams playing a match

- Team A has 60% chance of winning any given game
- The first team to win ***by*** 4 games wins the match

What is the probability that team A wins the match?

Solution

- Play a single match (lines 6..14)
- Note that this is a while-loop
- Then repeat 100,000 times
- Team A has an 83.431% of winning.



Fun with Monte Carlo Simulations: Farkle Odds

Problem: Roll 6 dice.

- What is the probability of getting 3 of a kind?
- (xxx a b c) or (xxx aa b)

Monte-Carlo Simulation

- Roll 6 dice 10,000 times
- Count how many times you get 3 of a kind

Sort of the definition of probability

- Repeatable event (roll 6 dice)

Code: Farkle.txt

- Roll 6 dice (line 8)
- Count the frequency of each number
  - lines 11-18
- Sort in descending order
  - line 19
- Check if the result was a 3 of a kind
  - xxx a b c
  - 333 1 1 1
  - xxx aa b
  - 333 22 1

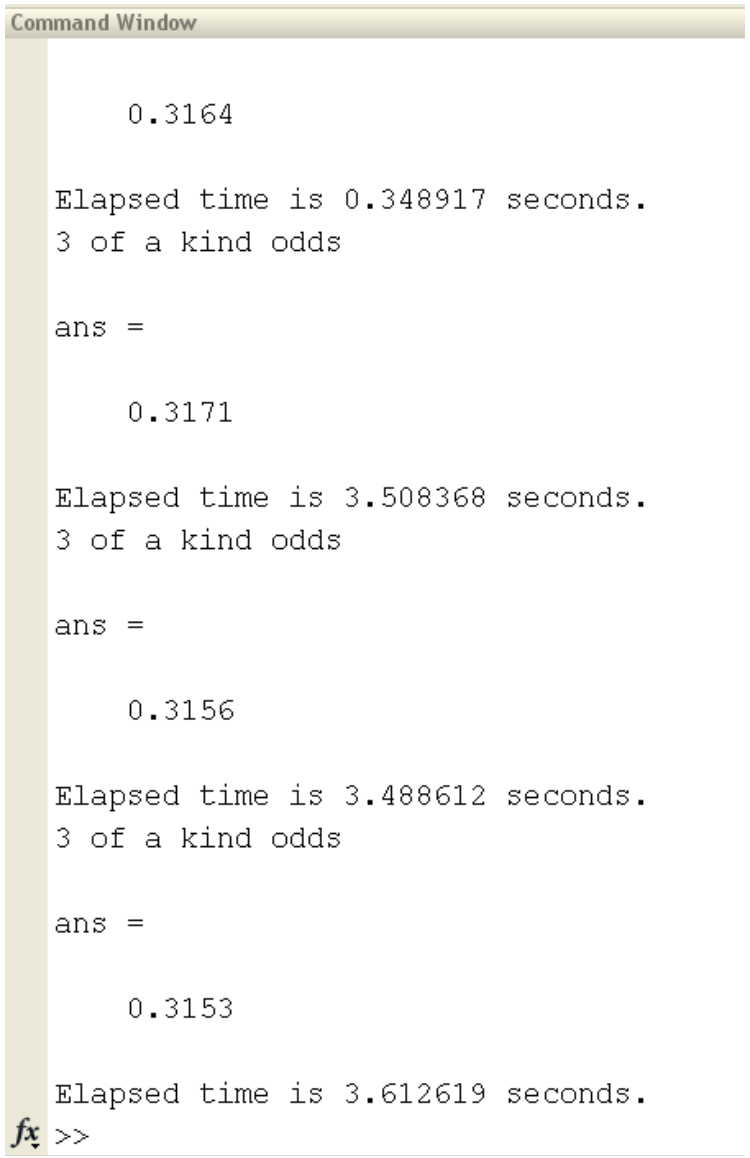
```
1    % Lecture #1: Monte Carlo
2    % Roll six dice
3    % Count how many times you get 3 of a kind
4    % xxx a b c
5    % xxx aa b
6    Pair3 = 0;
7    for games = 1:1e4
8       Dice = ceil( 6*rand(1,6) );
9       Dice = sort(Dice);
10      % check for pairs
11      N = zeros(1,6);
12      for i=1:6
13         for j=1:6
14            if(Dice(j) == i)
15               N(i) = N(i) + 1;
16            end
17         end
18      end
19      [N,b] = sort(N, 'descend');
20   if ( (N(1) == 3) & (N(4) < 3))
21       Pair3 = Pair3 + 1;
22       end
23   end
24   % probability:
25   disp('3 of a kind odds')
26   Pair3 / 1e4
27
```

Result: 100,000 rolls

- Each time you run the code you get a different result
- It's random
- Probability is about 31.6% chance

You can determine the 90% confidence interval for the probability

- t-test
- coming soon



Monte Carlo Simulations: Poker Odds

Determine the odds of drawing 3 of a kind in poker

- 52 card deck
- Draw 5 cards
- Results is 3 of a kind
  - xxx y z
  - x, y, z are different values (Ace to King)

Wikipedia: $p = 0.02113$

Verify using Monte-Carlo techniques

Matlab Code

Shuffle the deck

- Line 8 - 9

Draw 5 cards

- Line 10:
- Draw the top 5 cards

Determine suit & value

- Line 11-12:

Count frequency of A..K

- Line 14-17

Success = 3 of a kind

- Line 21:
- N = [3, 1, 1, 0, 0, ...]

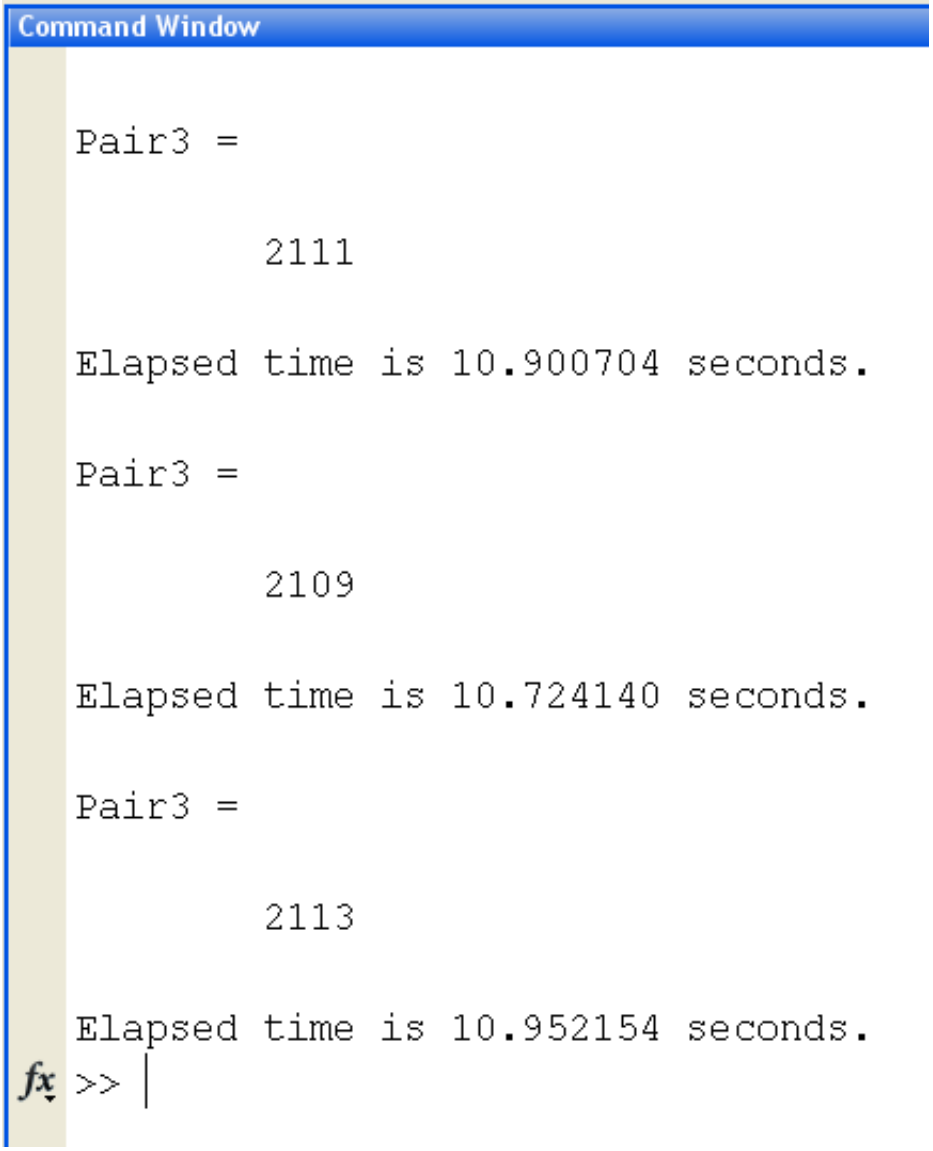
```
1     % lecture 1: Monte Carlo 5-Card Stud
2     % Probability of 3 of a kind
3     tic
4     Pair3 = 0;
5
6     for i0 = 1:1e5
7
8     X = rand(1,52);
9     [a,Deck] = sort(X);
10    Hand = Deck(1:5);
11    Value = mod(Hand,13) + 1;
12    Suit = floor(Hand/13) + 1;
13
14    N = zeros(1,13);
15    for n=1:13
16        N(n) = sum(Value == n);
17    end
18
19    [N,a] = sort(N, 'descend');
20
21    if ((N(1) == 3)*(N(2) < 2)) Pair3 = Pair3 + 1; end
22
23    end
24
25    [Pair3]
26    toc
```

Poker Results: 100,000 hands

- Results vary each time you run the code
  - It's a random process
- About 2111 successes each 100,000 hands
- p(x) = 0.02111 ish

From Wikipedia:

- p(x) = 0.02113



Summary

A random process is

- Repeatable
- The results are different each time

Probability is

- The ratio (success / trials)
- As the number of trials goes to infinity

Monte-Carlo simulations can approximate this

- Results will vary each time you run the simulation
- Gets more accurate as the number of trials goes to infinity