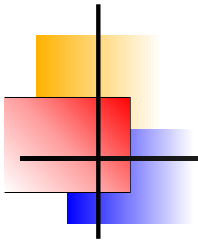
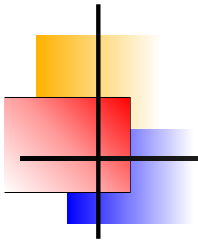


An introduction to Matlab

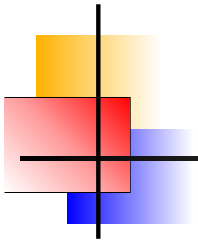
Jun Lu

January 9, 2013



Outline

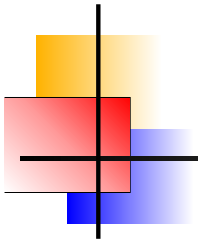
Lecture 1



What is MATLAB

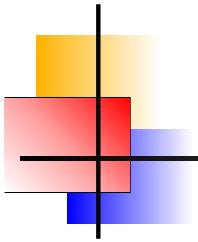
MATLAB (matrix laboratory) is a numerical computing environment. It has a very friendly interface to

- matrix manipulations
- plottings functions and data
- implementation of algorithms
- large set of building functions and libraries known as toolbox. For instance, Image Processing Toolbox.



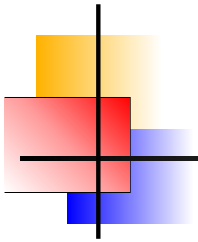
Syntax

- Variables
  - variables are defined using the assignment operator =
  - MATLAB is dynamically typed, i.e. variables can be assigned without declaring their tpe
  - MATLAB is weakly typed, i.e. types are implicitly converted
- Matrices(vectors)
  - Matrix is the basic component in MATLAB.
  - To generate a matrix, use [1,2,3;4,5,6]
  - special matrices: eye(n), magic(n), ones(m,n), zeros(m,n)



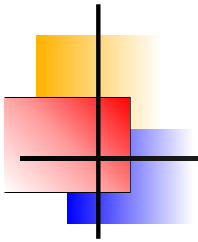
Syntax

- Vector indexing
  - a:stepsize:b 1:4
  - A=1:10
  - first element A(1), last element A(end)
  - range A(2:4)
- Matrix indexing
  - A=[1,2,3;4,5,6;7,8,9]
  - A(i,j)
  - first row A(1,:), first column A(:,1), last row A(end,:), last column A(:,end)
  - submatrix A(1:2,3:4)



Matrix operations

- A+B, A-B, A*B, A\(Ax=b)
- rank(A), eig(A)



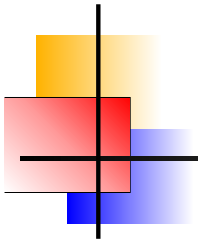
Random number generator

A random number generator generates a sequence of numbers that lacks patterns, i.e. appears random. In MATLAB, function

$$randi([min, max], m, n)$$

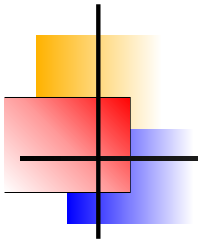
generates a matrix of size m by n whose elements are random integers in the close interval [min,max]. In other words, each integer in [min,max] will be selected with equal chance. We will make this statement mathematically rigorous later.

- flip a coin, randi([0,1],1,1) where 0 represents tail and 1 represents head
- flip a coin 100 times, randi([0,1],100,1)
- toss a die, randi([1,6],1,1)
- toss a die 100 times, randi([1,6],100,1)



Pseudorandom number vs "True" random numbers

- The random number generated by randi above is obtained by a deterministic algorithm. A sequence generated by this algorithm is completely determined by the initial value known as key. This means the sequence is inherently predictable. However, in the short run, the sequence demonstrates certain degree of randomness which is adequate for most of applications.
- However, in some applications, for example, cryptography, randomness is a critical issue. Better source could be physical pheonomenon whose unpredictability can be traced to the laws of quantum mechanics. For example, radio noise, cosmic rays.



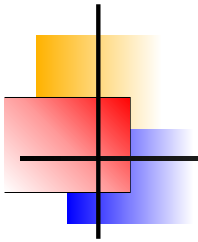
Example: tossing one die

Let write a function which returns the outcomes of flipping a fair coin n times

```
function outcome=flip_coin(n)
    outcome=randi([0,1],1,n);
end
```

Similarly, the function which returns the outcomes of tossing a fair die n times is

```
function outcome=toss_die(n)
    outcome=randi([1,6],1,n);
end
```



Example: tossing one die

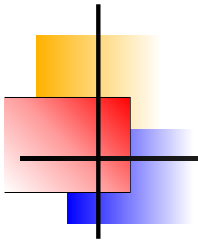
Let's do some tests on the outcomes of tossing a die we just obtained

- the number of times that 1 appears out of n

```
function p=frequency(outcomes,k)
    p=0;
    for i=1:length(outcomes)
       if outcomes(i)==k
          p=p+1;
       end
    end
end

sum(outcomes==k);
```

- the number of times that 4 appears out of n



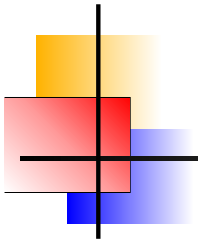
Example: tossing one die

We can count the frequencies of every element in the outcomes by a single function

$$tabulate(outcomes)$$

Let's try different n's

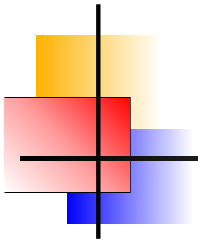
- n=10, n=100, n=1000, n=10000, n=100000
- As n increases, the frequency of outcome being 1 is n/6



Example: tossing two dice

Let's toss two fair dice at the same time and check the sum of them

```
function occurrence=toss_two_dice(n)
  outcome1=toss_die(n);
  outcome2=toss_die(n);
  outcome=outcome1+outcome2;
  table=tabulate(outcome);
  occurrence=table(:,3)/100*36;
end
```

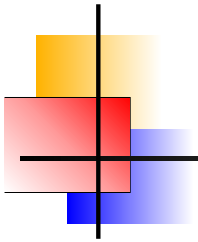


Continuous random numbers

In MATLAB, function

$$rand(m, n) \tag{1}$$

returns a matrix of size m by n whose elements are real numbers in the closed interval [0,1], i.e. each point in [0,1] is equally likely to be chosen.



Example: raindrops

Let's consider the raindrops on a unit square $[0,1] \times [0,1]$. Assume the rain is uniformly distributed and there are n raindrops in total. The X and Y coordinate of those raindrops is

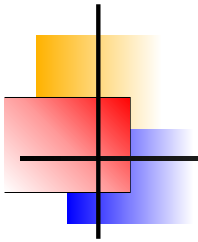
$$X = rand(1, n)$$

and

$$Y = rand(1, n)$$

We can plot it using

$$plot(X, Y, '*')$$



Example: raindrops

Let's write a function to count how many raindrops are in the disk with radius 0.5 and centered at $[0.5, 0.5]$

```
N=0;
for i=1:length(X)
  if (X(i)-0.5)^2+(Y(i)-0.5)^2<0.5^2
    N=N+1;
  end
end
```

Intuitively, the area of the disk is $pi/4$ while the area of the square is 1. So we expect the ratio of the number of raindrops (m) in the disk to the total number n to be $pi/4$. Thus, $m/n = pi/4$. We obtained a formula to compute pi here,

$$pi \approx 4 * m/n$$