ESE-2014 Lab 1

Name: Shreya Mamadapur Student ID: C0774035
Instructor: Takis Zourntos

Introduction

Matlab has an easy to use environment where usually we compute/program using mathematical notations. We can get visual representation of data we enter by plotting graphs in 2D or 3D. This is helpful to analyse various mathematical computations visually.

This document describes some script commands for matrix manipulation of datas in MATLABWe will go through some commands details in matlab by solving 15 problems.

Discussion

To begin with, it's always a good practise to clear the screen, close any existing windows of figures and clear things in the cached memory. To do this, we write the following commands:

```
clear;
close all;
clc;
```

Let's look at some matrix manipulation first.

To create a matrix we use similar command to this: `M = [1 2 3; 4 5 6; 0 8 9];`

The semicolons inside the square brackets are to go to the next row within the matrix.

```
M =
    1   2   3
    4   5   6
    0   8   9
```

To transpose a matrix(flipping the matrix along the diagonal), we use either of these commands:

M_t = M.'; or M_t = transpose(M);

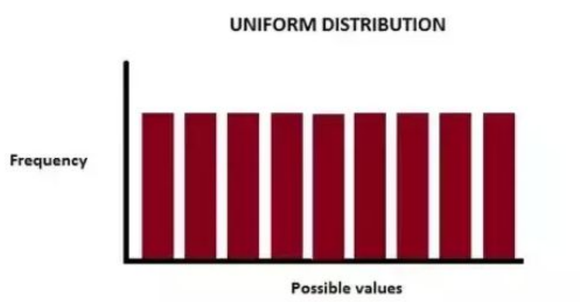
```
M_t =
    1   4   0
    2   5   8
    3   6   9
```

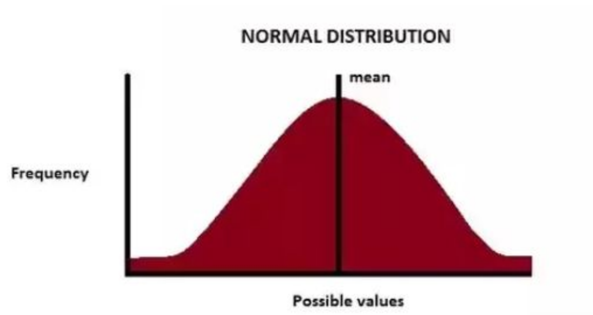
ESE-2014 Lab 1

To create a desirable vector/matrix of zeros we simply write for eg: zeros(2,3)

```
ans =
  0   0   0
  0   0   0
```

To create a vector of random numbers, we can use rand or randn commands.
rand command gives a uniformly distributed random vector, whereas randn gives normally distributed random vector (concentrated near the mean value).






```
r_15 = randn(1,5);
r_15 =
   0.95115  -0.18331   0.97212  -0.87039  -1.08458

R_15 = rand(1,5);
R_15 =
   0.0052849   0.5703582   0.6452792   0.3490813   0.4408916
```

There are many applications where finding a cross product and dot product of vectors is useful.
Dot product is related to the angle between two vectors, so the sign of your result can help you to identify if the vectors are perpendicular, or if they have an obtuse angle between them (if the result is negative).

Cross product: mostly used to calculate vectors perpendicular to the plane, or also to describe the particular condition of parallelism. Cross product gives exact value of the area of parallelogram formed between any two vectors.

```
dot_p = dot([1 2 3],[4 5 6]);
dot_p =  32 %dot product always gives a scalar value
```

ESE-2014 Lab 1

```
cross_p = cross([1 2 3],[4 5 6]);
cross_p =
  -3   6  -3
```

To create a vector of many elements with equal intervals this command is useful while programming:

```
vec_0to500 = linspace(0, 500, 100);
```

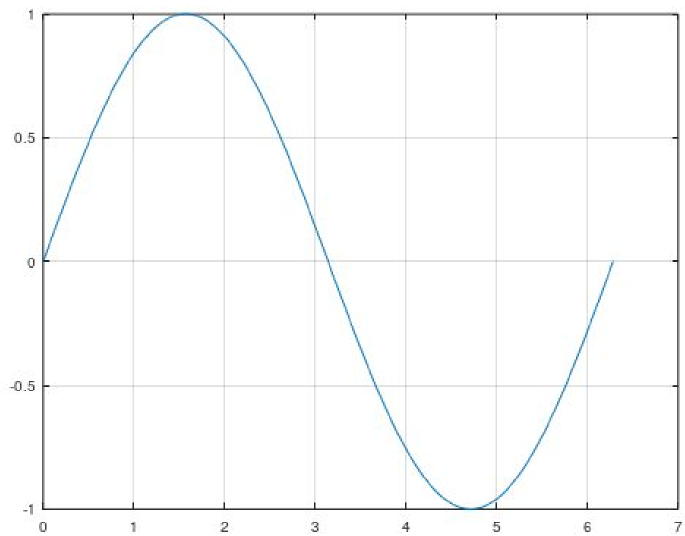
A 100 element vector is generated with values ranging from 0 to 500.

Now, let's dig a little deeper to know how graphs work in matlab. Let's look at how to create a sine wave between 0 and 2pi time period.

In order for a set of points to traverse in a sine wave, we need a set of elements to compute sine of each of those values. We get these elements by,

```
time_r = 0:res:(T);   %where, T = 2pi in this case and res(resolution is our desired   sampling interval)= T/number of points to plot
```

Then we simply use `sin(2*pi*f0*time_r)` to get our sine wave values.



To get a 3D graph let's look at how an exponential surface plot is obtained.

The following commands set our desired coordinates. div is the number of elements in the particular axis.

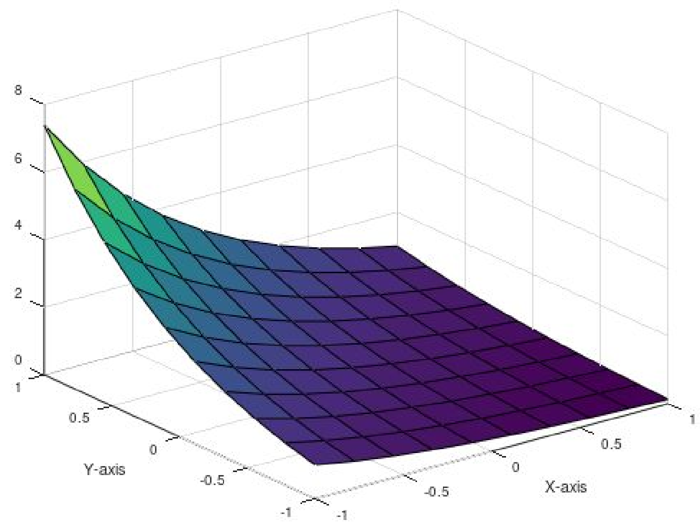
```
xr = linspace(-1, 1, div); %x-axis ranges from -1 to 1 with div=10 values in between
yr = linspace(-1, 1, div); %y-axis values range from -1 to 1
```

"exp()" performs the exponential function. So we perform exp function on both xr and yr values and to get z coordinate values we can simply multiply the transpose of exp(xr) with exp(yr). We are transposing here so that both vectors are not 1x10.

```
z_r = e_xr' * e_yr;
```

Resulting z_r is a matrix of many elements.

Then we can use “surf(x,y,z)” to get our beautiful 3D surface plot.



In question 14, we are calculating the mean of 20 random numbers and then comparing each random number to the mean we obtained to display whether it is higher/lower or equal to the mean.

For the details of the program, refer to the appendix.

Result of this solution:

```
Value is -0.48479 and it is lesser than mean 0.0621244
Value is 1.0372 and it is greater than mean 0.0621244
Value is -0.730922 and it is lesser than mean 0.0621244
Value is -0.598408 and it is lesser than mean 0.0621244
Value is 1.24302 and it is greater than mean 0.0621244
Value is 1.69325 and it is greater than mean 0.0621244
Value is 0.937887 and it is greater than mean 0.0621244
Value is 0.436544 and it is greater than mean 0.0621244
Value is -0.986383 and it is lesser than mean 0.0621244
Value is -0.568477 and it is lesser than mean 0.0621244
```

ESE-2014 Lab 1

```
Value is -1.23842 and it is lesser than mean 0.0621244
Value is 1.94901 and it is greater than mean 0.0621244
Value is -0.228163 and it is lesser than mean 0.0621244
Value is 0.274894 and it is greater than mean 0.0621244
Value is -0.379099 and it is lesser than mean 0.0621244
Value is -0.375551 and it is lesser than mean 0.0621244
Value is -1.11083 and it is lesser than mean 0.0621244
Value is -0.858101 and it is lesser than mean 0.0621244
Value is 0.0578097 and it is lesser than mean 0.0621244
Value is 1.17201 and it is greater than mean 0.0621244
```

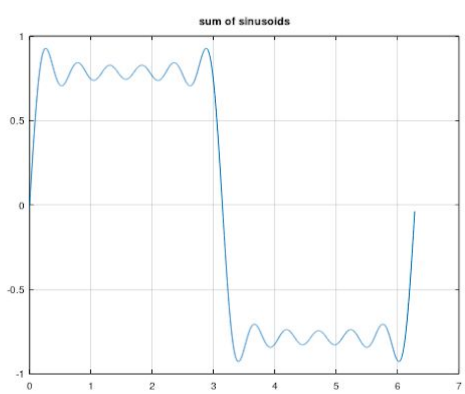
In problem 15, let's visually see that adding n number of odd harmonics of a sine wave can get us a square wave. This basically is to show how sine waves can be used to create other waveforms.

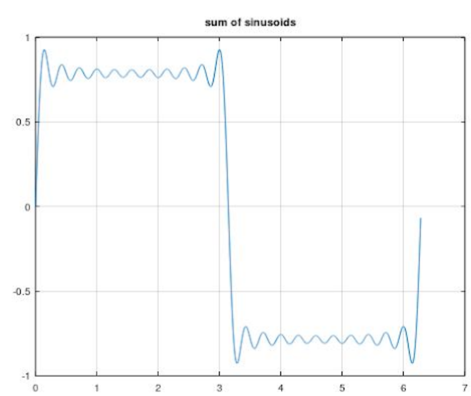
S = sin(t) + sin(3t)/3 + sin(5t)/5 +.....

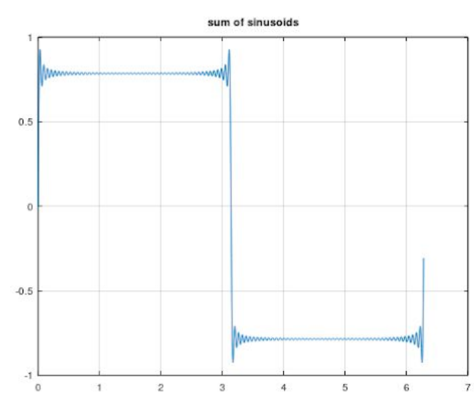
This is the fourier transform of a square wave. This shows the concept of fourier transform, which basically says that any periodic waveform can be decomposed into sine/cosine waves.

In program 15, we generate the fundamental sine wave and add its odd harmonics to it by using the addsines() function, which basically sums a matrix whose rows contains the value of each sine wave.

Three graphs are shown below with number of sine waves added = 5, 10 and 50 respectively. The change in the impact of the number of sine waves is very noticeable.







ESE-2014 Lab 1

Appendix

All commands in executable form:

```
%Instructor: Takis Zourntos
%Student: Shreya Mamadapur
clear;
close all;
clc;


%1. Create a vector.
x_row = [0 1 2 3 4]; %1x5 row vector
y_col = [1;2;4;7;5]; %5x1 column vector

%2. Create a matrix.
M = [1 2 3; 4 5 6; 0 8 9]; %3x3 matrix


%3. Create a 5x1 vector of zeros. Create a 1x5 vector of random numbers.
Z_51 = zeros(5,1);
R_15 = rand(1,5); %uniformly distributed 1x5 random vector
r_15 = randn(1,5); %normally distributed 1x5 random vector


%4. Transpose a matrix.
M_t = transpose(M);
%or
M_t = M.';


%5. Compute the inner product of two matrices. Compute the cross product of
two matrices. %Compute the inverse of a matrix.


x=[1 2 3];
y=[4 5 6];
cross_p =cross(x,y);
dot_p = dot(x,y);
I = inv(M);


%6. Compute the element wise multiplication of a matrix and a scalar.
e1= M*pi; %pi is a scalar quantity
```

ESE-2014 Lab 1

```
% 7. concatenate two matrices:
C1 = [M e1]; %side-by-side or horizontal concatenation
C2 = [M e1]; %vertical concatenation

% 8. create a vector of complex numbers:
V_C = [ 2+3j 3-j ; 7-2j 4+9j];

%9. Multiply a row of a matrix with an element of that same matrix.
%This multiplies the second row,second column element with the first row.
Generates a 1x3 vector
M_S = M(2,2)*M(1,:);

%10.Generate a vector of values ranging from 0 to 500 with 100 elements.
vec_0to500 = linspace(0, 500, 100);
%or
Vect_0to500 = 0:(500/99):500;

%11.Create a 2D plot of the sine function between 0 and 2π.
T = 2*pi;f = 1/T;
n_pts = 100; % number of points to plot
res = T/n_pts; % sampling at discrete intervals
time_r = 0:res:(T); %getting 100(app may increment 1 to fit correctly)values
%compute sine function of 2xpixfxT_r for each T_r value
sine = sin(2*pi*f0*time_r);
figure;plot(time_r,sine); grid;

%12.Create a 3D plot of a surface by creating a grid along the X and Y axes
and plotting the %Z-coordinate according to the exponential function.
div = 10; % number of elements in X and Y axis
xr = linspace(-1, 1, div); % x-axis ranges from -1 to 1 with 10 divisions
yr = linspace(-1, 1, div); %y-axis values range from -1 to 1 with 10 values
in between
e_xr = exp(xr); %performs exponential function of xr
e_yr = exp(-yr);
z_r = e_xr' * e_yr; %to get a matrix of z values multiplying e_xr transpose
to e_yr
figure;
surf(xr, yr, z_r); %using surf to get 3D plot
xlabel('X-axis');ylabel('Y-axis');
```

ESE-2014 Lab 1

```
%13.Write a script to plot a vector of random data. Draw a horizontal line
at the mean. Save the %script and run it from the command line.
res_pts = 512; % number of data points
xdata = 0:(res_pts-1); % independent variable
ydata = randn(1,res_pts); %random 512 fractional data(normally distributed
i.e, mean is 0)
mean_val = mean(ydata);
figure;
plot(xdata,ydata,'p.',xdata,mean_val*ones(size(xdata)),'r-'); %data are
displayed as purple points. Mean value is displayed as a red line
throughout 512 points.
grid;
legend('Data','Mean value'); %gives symbol representation of data and mean
value in top-right corner


%14.Write another script that calculates the mean of five samples of data
from a vector of %random data. Calculate the overall mean. Use a for loop
to perform the calculations. For each %iteration of the loop print out the
intermediate results. Use an if..else control block to display %the results
depending on whether the mean of the samples is less than, greater than or
equal %to the overall mean.


data= randn(1, 20); %a vector of 20 random elements

      mean_samples = mean(data); % find mean of the samples
      for i=1:20 %Loop
      if data(i) > mean_samples %Check if greater
            printf("Value is %d and it is greater than mean %d\n" ,
data(i), mean_samples);
      elseif data(i) < mean_samples % Check if smaller
            printf("Value is %d and it is lesser than mean %d\n" , data(i),
mean_samples);
      else %Check if equal
            printf("Value is %d and it equal to mean %d\n" , data(i),
mean_samples);
      end;
      end;


%15.Create a function that calculates the sum of an arbitrary number of
```

```
sinusoidal terms. Call this function from the MATLAB command line or in a
MATLAB script (.m).

%Program to add n number of odd harmonics of a sine wave to get a square
wave. This shows how sine waves can be used to create other waveforms
clear;
clc;
function Ysum = addsines(Y)
      Ysum = sum(Y, 1);
endfunction

T0 = 2*pi; % desired period of sine
f0 = 1/T0; % frequency in Hz is just the reciprocal of the period
Npts = 1024; % number of points to plot
delta = T0/Npts; % increment or "sampling period"
trange = 0:delta:(T0-delta);

Nterms = 10; %number of sine ways to add
Y = zeros(Nterms, Npts);
for i = 1:2:Nterms*2+1; % increment by two to get the odd harmonics
      oddHarmonic = i*f0;
      sineValue = (sin(2*pi*oddHarmonic*trange))/i;%Get the value of the
harmonic
Sines(i,:) = sineValue;
end;

Ytotal = addsines(Sines);
figure; plot(trange, Ytotal); grid; title('sum of sinusoids');
```

Conclusion

This Lab activity helps us familiarise with octave/Matlab. It helps understand many matrix operation command scripts. We also learn to graphically represent data in both 2D and 3D and understand the significance of our computations visually.