**PRACTICE 1 - INTRODUCTION TO DISCRETE-TIME SIGNALS USING MATLAB**

**OBJECTIVES:**

1. Using Basic instructions.
2. Generation and visualization of sequences.
3. Main operations with sequences or vectors.
4. Plotting and handling of continuous (analogic) signals.
5. Sampling of continuous (analogic) signals.
6. Definition of MATLAB functions. Main sequences.
7. Random signals. Main operations: mean, median, standard deviation, autocorrelation.

|  |  |
| --- | --- |
| **Name 1:** | Alejandro Serrano López |
| **Name 2:** | Mihnea-Iulian Popescu |

1. **Basic instructions**

When you start the MATLAB program you can see several windows in the screen. The center window, **“Command Window**”, is the working area, where you write the different commands and expressions. After pressing <Intro>, the system returns the answer.

Variable assignment is done by the statement:

>> name of the variable = value;

Keep in mind that the system is case sensitive. The instruction “clear” removes items (variables and values) from workspace, freeing up system memory.

The basic elements in MATLAB are matrices. You can define matrices entering its elements between square brackets, by rows, separated by blanks or commas and the rows separated by semicolons. A vector is a matrix with one row. The operators for matrix arithmetic are the usual (+, \*, ^). Matrices, and vectors can be multiplied element by element with (.\*). Similarly, to raise all the elements of a matrix to a power, we use (.^).

Define the vector a=[1 -2 3]and the matrix b=[1 0 5;2 1 4;1 0 -1].

Calculate, with the instruction a\*b, the product of the vector by the matrix:

|  |
| --- |
| = |

Enter the following expressions and justify the obtained answers.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Expression** | a+b | a.^2 | a^2 | a\*a’ |
| **System response** |  |  | Error using ^  Incorrect dimensions for raising a matrix to a power. Check that the matrix is square and the power is a  scalar. To operate on each element of the matrix individually, use POWER (.^) for elementwise power. | 14 |
| **Comments** | Strictly speaking, this shouldn’t be possible since both matrices should have the same dimension. However MATLAB is smart and what it did was add the vector to all of the rows, as if we had a matrix with three rows composed of the same vector. | .^ is the array operation, meaning that it squared every single value inside the matrix | ^ is the matrix operator, that is, equivalent of a\*a, but that operation requires that the first matrix has as many columns as the second has rows. | This one, with a’ being the transposed of a, guarantees that the number of rows and columns mentioned of the previous example match, so it proceeds to do  Which only leaves one row and column since the operation basically only calculates one value per first matrix row and second matrix column. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Expression** | a’\*a | b^2 | b.^2 |
| **System response** |  |  |  |
| **Comments** | From the previous one, except this time is done the other way around, so instead of having 1 column of a and 1 of b, we have 3 of a and 3 of b, so we end up with a 3 x 3 matrix | Since this one has already 3 rows and columns (square matrix), and from what was explained on the two previous examples, it will proceed to perform the matrix operation accordingly. | .^ is the array operation, meaning that it squared every single value inside the matrix |

1. **Generation and visualization of sequences**

A finite sequence of data is entered in the form:

n = n1:ni; % n is an independent discrete variable

x=[x(1), …,x(i)]; % i is the vector position

where n is a vector that takes integer values of length i, and can be represented graphically by the statements stem(n,x) or plot(n,x).

Try the following instructions and comment the results:

|  |  |  |
| --- | --- | --- |
| **Expression** | x(0)=3 | x(5)=-pi |
| **System response** | Array indices must be positive integers or logical values. | x =  0 0 0 0 -3.1416 |
| **Comments** | That means MATLAB arrays count from 1 onwards | That means if not previously declared the matrix values are initialized as 0, so in this case we commanded MATLAB to create a zero matrix with its fifth value being -π |

Plot the following sequences:

1) x(-1)=2, x(0)=0.25, x(1)=4.1, x(2)=0, x(3)=0.3.

2) x(n)=n2, for *n* from -10 to 10.

3) x(n)=2n, for *n* from -5 to 5.

Write here the entered instructions and copy the plots:

|  |  |
| --- | --- |
| **INSTRUCTIONS** | **PLOTS** |
| 1) Observation: Try the instrucction x(1)  >> n = -1:3;  >> x = [2, 0.25, 4.1, 0, 0.3];  >> stem (n, x) |  |
| 2)  >> n = -10:10;  >> x = n.^2;  >> plot (n, x) | Gráfico, Histograma  Descripción generada automáticamente |
| 3)  >> n = -5:5;  >> x = 2.^n;  >> plot (n, x) | Gráfico  Descripción generada automáticamente |

Plot the real part, the imaginary part, the module, and the argument of the complex signal defined, for n=0:100 by x(n)=exp(2\*j\*pi\*n/100). Copy the plots.

|  |  |
| --- | --- |
| real(x)  >> plot (n, real(x))  Gráfico, Gráfico de líneas, Histograma  Descripción generada automáticamente | imag(x)  >> plot (n, imag(x))  Gráfico, Histograma  Descripción generada automáticamente |
| abs(x)  >> plot (n, abs(x))  Gráfico  Descripción generada automáticamente | angle(x)  >> plot (n, angle(x))  Gráfico, Gráfico de líneas  Descripción generada automáticamente |

1. **Operations with sequences or vectors**

Remember that:

* size(x,2) returns the number of entries in the vector x.
* sum(x(1:N)) returns the sum x(1)+…+x(N).
* cumsum(x(1:N)) returns the cumulative sum vector [x(1) x(1)+x(2) … x(1)+…+x(N)].
* prod(x(1:N)) calculates the product x(1)…x(N).
* cumprod(x(1:N)) returns the cumulative product vector [x(1) x(1)\*x(2) … x(1)\*…\*x(N)].

Enter the sequences x=[1,-1,0,1,-1,0,1,-1]; y=[1,2,3,4] (both signal start in n=-1). Complete the following table:

|  |  |  |
| --- | --- | --- |
|  | **INSTRUCTION** | **SYSTEM RESPONSE** |
| 1) The sum of the elements of x. | >> sum(x(1:end)) | ans =  0 |
| 2) The product of the elements of y. | >> prod(y(1:end)) | ans =  24 |
| 3) The cumulative sum of the elements of x. | >> cumsum(x(1:end))  OR  >> cumsum(x(:)) | ans =  1 0 0 1 0 0 1 0  ans =  1  0  0  1  0  0  1  0 |
| 4) The energy of the signal x. | >> sum(x(1:end).^2)  % or >> sum(x(:).^2) | ans =  6 |
| 5) Plot the signal x(n-2) | >> n = -1:-1+(length(x)-1);  >> >> m = n+2;  >> stem (m, x) |  |

1. **Plotting and handling of continuous (analogic) signals**

MATLAB allows working with analogic signals. The syntax is:

h = @(arglist) anonymous\_function

For example, the function f(t)=t2 would be defined as: f = @(t) t^2. To plot the function f(t), use the command fplot. For example, fplot(f,[-3 0],'b').

Plot the functions sin(t), sin(2t) and 2sin(t) in the interval [0, 2π] and explain the relationship between them (period and amplitude of each one).

|  |  |  |
| --- | --- | --- |
| **Plot of x=sin(t)** | **Plot of x=sin(2t)** | **Plot of y=2sin(t)** |
| Gráfico, Gráfico de líneas  Descripción generada automáticamente | Gráfico, Gráfico de líneas  Descripción generada automáticamente | Gráfico  Descripción generada automáticamente |
| Period= 2 \* pi  Amplitude= 1 | Period= pi  Amplitude= 1 | Period= 2\* pi  Amplitude= 2 |

*Multiplying the value inside the trigonometric function will shorten the period, while multiplying the trigonometric function by a value will proportionally affect the output.*

Plot the function on the interval [-10, 100].

*>> f = @(t) (exp(1).^t ./ (1 + exp(1).^(2\*t)));*

*>> fplot (f, [-10 100], 'b');*

|  |
| --- |
| Gráfica de |
|  |

1. **Sampling** **of continuous (analogic) signals**

It is requested to sample the analogic signal to obtain the sequence x(n).

For this exercise we need to define the following values:

* The limits of the interval in which we want to work: [t1,t2] = [-10,20].
* The sampling frequency F = 1/T = 0.5 (T is the sampling period).

First, the sampled signal of f(t) must be calculated and plotted (use stem command). Next, the analogic signal f(t) and the sampled signal will be plotted together. Finally, the sequence x(n) = f(-10 + n\*Ts) will be plotted from n = 0 to n = nf.

|  |  |  |
| --- | --- | --- |
| **Sampled signal of f(t) plot** | **Joint plots** | **x(n) plot** |
|  | Gráfico  Descripción generada automáticamente | Gráfico, Gráfico de líneas  Descripción generada automáticamente |
| **Instructions** | **Instructions** | **Instructions** |
| >> Fs = 0.5;  >> Ts = 1/Fs;  >> n = -10:Ts:20;  >> x = (exp(n)./ (1 + exp(2\* n)));  >> stem(n, x);  -----OR-----  >> Fs = 0.5;  >> Ts = 1/Fs;  >> t = -10:Ts:20;  >> F = f(t);  >> stem(t, F); | >> hold on;  >> grid on;  >> f = @(t) (exp(1).^t ./ (1 + exp(1).^(2\*t)));  >> fplot (f, [-10 20], 'g'); | >> x2 = f(-10 + n\*Ts);  >> plot (n, x2, 'r');  >> stem (n, x2, 'm'); |

1. **Definition of MATLAB functions. Main sequences.**

Example: A function called “staty.m” that returns the mean and standard deviation of an input vector.

function [m,s] = staty(x)

n = length(x);

m = sum(x)/n;

s = sqrt(sum((x-m).^2/n));

end

Define the functions unit impulse, unit step and rectangular pulse with the following input variables:

* Unit impulse: function[x,n] = Delta(n0,ni,nf) %unit impulse in n0, defined in ni:nf
* Unit step: function [x,n] = Step(n0,ni,nf )%unit step defined in ni:nf and takes the value 1 from n0.
* Rectangular pulse: function [x,n] = Rectangle(n1,n2,ni,nf) %rectangular pulse defined 1n ni:nf and takes the value 1 from n1 to n2.

Define the functions:

|  |  |  |
| --- | --- | --- |
| [x,n] = Delta(n0,ni,nf) | [x,n] = Step(n0,ni,nf) | [x,n] = Rectangle(n1,n2,ni,nf) |
| function[x,n] = Delta(n0,ni,nf) %unit impulse in n0, defined in ni:nf  arguments  n0(1,1) {mustBeNumeric, mustBeFinite}  ni(1,1) {mustBeNumeric, mustBeFinite}  nf(1,1) {mustBeNumeric, mustBeFinite}  end  n = ni:nf;  x(nf-ni+1) = 0;  x(n0-ni+1) = 1;  % return (x n);  end | function[x,n] = Step(n0,ni,nf) %unit step defined in ni:nf and takes the value 1 from n0.  arguments  n0(1,1) {mustBeNumeric, mustBeFinite}  ni(1,1) {mustBeNumeric, mustBeFinite}  nf(1,1) {mustBeNumeric, mustBeFinite}  end  n = ni:nf;  x(nf-ni+1) = 0;  x(n0-ni+1:nf-ni+1) = 1;  end | function [x,n] = Rectangle(n1,n2,ni,nf) %rectangular pulse defined 1n ni:nf and takes the value 1 from n1 to n2.  arguments  n1(1,1) {mustBeNumeric, mustBeFinite}  n2(1,1) {mustBeNumeric, mustBeFinite}  ni(1,1) {mustBeNumeric, mustBeFinite}  nf(1,1) {mustBeNumeric, mustBeFinite}  end  n = ni:nf;  x(nf-ni+1) = 0;  x(n1-ni+1:n2-ni+1) = 1;  end |

Plot the sequences:

|  |  |  |
| --- | --- | --- |
| [x,n] = Delta(5,-2,10) | [x,n] = Step(5,-2,10) | [x,n] = Rectangle(0,5,-2,10) |
|  | Gráfico  Descripción generada automáticamente | Gráfico, Histograma  Descripción generada automáticamente |

1. **Random signals.**

Define a sequence x(n) of random values between 0 and 2 of length 50 starting at n=-7.

|  |  |
| --- | --- |
| **Instructions** | **Plot** |
| function [x,n] = RandomSequence(maxval,minval,ni,length) %rectangular pulse defined 1n ni:nf and takes the value 1 from n1 to n2.  arguments  maxval(1,1) {mustBeNumeric, mustBeFinite} % max range value  minval(1,1) {mustBeNumeric, mustBeFinite}  ni(1,1) {mustBeNumeric, mustBeFinite}  length(1,1) {mustBeNumeric, mustBeFinite, mustBePositive}  end  nf = ni+length-1;  n = ni:nf;  xt = rand (1, nf-ni+1)  x = xt \* (maxval-minval) + minval;  end  % The following is typed in the console  [x,n] = RandomSequence(2,0,-7,50);  stem(n,x); |  |

Obtain the mean, median and standard deviation. Paint the boxplot.

|  |  |
| --- | --- |
| **Instructions** | **Boxplot** |
| Mean= 0.9752  Median = 0.9967  Standard deviation = 0.5559  function [m,s, med] = staty(x)  n = length(x);  m = sum(x)/n;  s = sqrt(sum((x-m).^2/n));  B = sort(x);  med = B(round(n./2));  end | >> boxplot(x) |

Finally, obtain the autocorrelation of the sequence x(n). Justify the results.

|  |
| --- |
| **Autocorrelation of x(n) plot** |
| >> theCorr = xcorr(x, x);  >> stem(theCorr); |

**Comments**:

We can see the function is symmetric, with a first part having a growing tendency until reaching a maximum peak at half of the independent variable values (more or less when the function fully overlaps the other), due to the fact both functions are the same function, it’s the same correlation operation done to itself (autocorrelation).