EEE 391 Basics of Signals and Systems
MATLAB Exercises with Solutions

1 Filling Arrays or Creating Signals in MATLAB

Create the following arrays or matrices in MATLAB without using any "for" loop:

- A 1×100 array consisting of all zero elements (Hint: Use the command **zeros**)
  Answer: **a=zeros(1,100);**
- A 10×12 matrix consisting of all ones (Hint: Use the command **ones**)
  Answer: **a=ones(10,12);**
- 5×5 identity matrix (Hint: Use the command **eye**)
  Answer: **a=eye(5);**
- A 1×100 array whose elements are $1, 2, 3, 4, \ldots, 99, 100$
  Answer: **a=[1:100];** or **a=[1:1:100];**
- A 1×4 array whose elements are $7, 17, 27, 37$
  Answer: **a=[7:10:37];**
- A 1×100 array whose elements are $3, 7, 11, 15, \ldots, 395, 399$
  Answer: **a=[3:4:399];**
- A 1×100 time array **t** such that **t(1,1)**=0, **t(1,2)**=0.01, ..., **t(1,100)**=0.99
  Answer: **t=[0:0.01:0.99];**
- A 1×100 array **x** such that **x(1,1)**=$\cos(2\pi 5 \times 0)$, **x(1,2)**=$\cos(2\pi 5 \times 0.01)$, ..., **x(1,10)**=$\cos(2\pi 5 \times 0.99)$
  Answer: **x=cos(2*pi*5*[0:0.01:0.99]);** or **x=cos(2*pi*5*t);** where **t** is as created in the previous item

2 Functions with array inputs in MATLAB

Let **t** denote the time array whose elements are $-1, -0.999, -0.998, \ldots, -0.001, 0, 0.001, \ldots, 0.998, 0.999, 1$. Recall that we can create **t** by issuing the command **t** = [−1 : 0.001 : 1];. On this time grid, compute the values of the following functions without using any "for" loop. Use as few lines of code as you can.

- $x(t) = 1$

  Answer: **x=ones(size(t));**

- $x(t) = 2t + 3$

  Answer: **x=2*t+3;**

- $x(t) = 3t^2 - 5t + 1$

  Answer: **x=3*t.^2-5*t+1;**

- $x(t) = \frac{2t^2-4t+1}{3t^3-2t^2+5t+2}$

  Answer: **x=(2*t.^2-4*t+1)./(3*t.^3-2*t.^2+5*t+2);**

- $x(t) = 2\cos(2\pi 5t + 1)$

  Answer: **x=2*cos(2*pi*5*t+1);**

- $x(t) = \sin^3(2\pi 7t)$

  Answer: **x=sin(2*pi*7*t).^3;**

- $x(t) = \cos^5(2\pi 2t^2)$

  Answer: **x=cos(2*pi*2*t.^2).^5;**

- $x(t) = 3\sin(2\pi\frac{4t+3}{2t^2+1}) - 4$

  Answer: **x=3*sin(2*pi*(4*t+3)./(2*t.^2+1))-4;**

- $x(t) = \frac{2\cos\left(\sqrt{\frac{2|t|+1}{4t^2+1}}\right)}{3\sin^3(5t-2)+4}$

  Answer: **x=(2*cos(((2*abs(t)+1)./(4*t.^2+1)).^0.5))./(3*sin(5*t-2).^3+4);**

- $x(t) = e^{j2\pi 10t}$

  Answer: **x=exp(j*2*pi*10*t);**

- $x(t) = e^{j\pi 3t^2}$

  Answer: **x=exp(j*pi*3*t.^2);**

- $x(t) = e^{-\frac{t^2}{2}}$

  Answer: **x=exp(-t.^2/2);**

- $x(t) = e^{-|t|}$

  Answer: **x=exp(-abs(t));**

3 Extracting Parts of a Matrix or an Array

Let $\mathbf{x}$= [x_1 x_2 x_3 x_4 ... x_{98} x_{99} x_{100}]. Prepare the following arrays using single-line commands: To test your codes, you may take $\mathbf{x}$= [1 2 3 ... 98 99 100].

- $\mathbf{y}$ = [x_{22} x_{23} x_{24} ... x_{55} x_{56}]
  Answer: **y=x(22:1:56);**
- $\mathbf{y}$ = [x_{61} x_{60} x_{59} ... x_{42} x_{41}]
  Answer: **y=x(61:-1:41);**
- $\mathbf{y}$ = [x_2 x_4 x_6 ... x_{98} x_{100}]
  Answer: **y=x(2:2:100);**
- $\mathbf{y}$ = [x_1 x_3 x_5 ... x_{97} x_{99}]
  Answer: **y=x(1:2:99);**
- $\mathbf{y}$ = [x_{12} x_{19} x_{26} ... x_{75} x_{82}]
  Answer: **y=x(12:7:82);**
- $\mathbf{y}$ = [x_{97} x_{92} x_{87} ... x_{37} x_{32}]
  Answer: **y=x(97:-5:32);**
- $\mathbf{y}$ = [x_1 0 0 0 x_2 0 0 0 x_3 0 0 0 ... x_{99} 0 0 0 x_{100} 0 0 0]
  Answer: **y=zeros(1,400); y(1:4:400)=x;**
- $\mathbf{y}$ = [0 0 x_1 0 0 0 x_2 0 0 0 x_3 0 ... 0 0 x_{99} 0 0 0 x_{100} 0]
  Answer: **y=zeros(1,400); y(3:4:400)=x;**
- $\mathbf{y}$ = [0 x_{100} 0 0 x_{99} 0 0 x_{98} 0 ... 0 x_2 0 0 x_1 0]
  Answer: **y=zeros(1,300); y(2:3:300)=x(100:-1:1);**
- $\mathbf{y}$ = [0 0 x_{42} 0 0 0 0 x_{46} 0 0 0 0 x_{50} 0 0 ... 0 0 x_{78} 0 0 0 0 x_{82} 0 0]
  Answer: **y=zeros(1,55); y(3:5:53)=x(42:4:82);**
- $\mathbf{y}$ = [0 0 x_{95} 0 0 x_{91} 0 0 x_{87} ... 0 0 x_{39} 0 0 x_{35}]
  Answer: **y=zeros(1,48); y(3:3:48)=x(95:-4:35);**

4 Some Common Programming Mistakes

4.1

Suppose $\mathbf{x}$ of size 1×1000 represents a signal $x(t)$, and $\mathbf{y}$ of size 1×1000 represents a signal $y(t)$. Let $\mathbf{g}$ represent the signal $g(t)$ defined as $g(t) = x(t)\, y(t)$. The following code tries to compute $\mathbf{g}$ but it contains a mistake so that MATLAB gives an error message. Find the mistake. What is the message that MATLAB gives?

g=x*y

Answer: MATLAB gives the error message:
Error using mtimes, Inner matrix dimensions must agree.
The reason is, the command **g=x*y** orders MATLAB to perform the **matrix multiplication** of **x** and **y**. However, under our definitions, the sizes of **x** and **y** are not suitable for matrix multiplication, so MATLAB gives an error message. Actually, even if their size were suitable, our intention is not to compute the matrix multiplication of **x** and **y**, but rather compute a new 1×1000 vector (that we name **g**) such that **g(1)=x(1)y(1)**, **g(2)=x(2)y(2)** and so on. The correct command to accomplish this is:
g=x.*y
Note that when we introduce the dot before the multiplication symbol, MATLAB understands that we want to perform **elementwise multiplication** of **x** and **y**, and gives us the desired **g**.

4.2

Suppose we have an image $x[m, n]$ that is stored in a matrix **x** of size 512×512. Let $y[m, n] = x^2[m, n]$. Now, we want to compute the matrix **y** which is again 512×512 and which contains $y[m, n]$. The following code tries to do it but it contains a mistake. What is the mistake?
y=x ^ 2

Answer: MATLAB recognizes the above command as the **matrix multiplication** of **x** with itself. That is, **y** computed in this manner represents a new matrix which is obtained as **y=x*x**. Note that since **x** is 512 by 512, the matrix multiplication **x*x** is defined and computed by MATLAB. However, this is not what we want. We actually want the relation between **x** and **y** to be **y(1,1)=x(1,1)*x(1,1)**, **y(1,2)=x(1,2)*x(1,2)** and so on. The correct command should be:
y=x .^ 2

4.3

The following code tries to sum 100 complex sinusoids over a time array given by **t**. The frequencies are contained within an array named **omega** and the amplitudes are contained within **A**. However, it contains a bug. Find it.

```
MySum=zeros(size(t));
for j=1:100
MySum=MySum+A(j)*exp(j*omega(j)*t);
end
```

Answer: Note that the letter **j** is defined on the second line as the counter parameter of the for loop. On the third line, it is also used to represent the unit imaginary number, i.e., $\sqrt{-1}$. However, since **j** is defined on line 2, MATLAB does not recognize it as $\sqrt{-1}$ any more, so **MySum** turns out to be quite different than intended.

One practice of avoiding such bugs is to reserve the letter **j** for $\sqrt{-1}$ and use different letters or names for the counters of loops. A better way to write the above code is:

```
MySum=zeros(size(t));
```

```
for i=1:100
MySum=MySum+A(i)*exp(j*omega(i)*t);
end
```

4.4

Suppose we have a 1×1000 array named **x** which consists only of zeroes and ones. Our purpose is to search the array from the beginning and set an alarm when the number of ones reach 10. The following code tries to do this but it contains a mistake. In particular, one line of code appears in a place where it should not be. Find that line and put it in the correct position so that the code works.

```
alarm=0;
index=0;
while(alarm==0)
count=0;
index=index+1;
if x(index)==1    count=count+1;    end
if count==10    alarm=1;    end
end
```

Answer: The given code sets count = 0 in each iteration of the for loop, so it cannot calculate the number of ones. To correct the mistake, the line "count=0;" must be moved anywhere before the for loop so that it increases by one when the value 1 is encountered in the array **x**.

4.5

The following code tries to form a periodic signal $x(t)$ by adding the Fourier series components for $-10 \leq k \leq 10$. Suppose the coefficients are given within a 1×21 array whose name is **X**. However, the code contains a small programming mistake so that MATLAB gives an error message. Find that mistake. What is the error message that MATLAB gives?

```
x=zeros(size(t));
for k=-10:1:10
x=x+X(k)*exp(j*2*pi*k*t/T);
end
```

Answer: Suppose the program recently enters the for loop, so that **k=-10** as indicated on line 2. On line 3, MATLAB tries to fetch the -10th value of **X**. The problem is, index values must be positive integers in MATLAB. In other words, **X(1)**, **X(2)**,..., **X(21)** are all defined but **X(-10)** is not defined. Therefore, MATLAB returns the error:

Index exceeds matrix dimensions

The correct way to write the code is as follows:

```
x=zeros(size(t));
for k=-10:1:10
x=x+X(k+11)*exp(j*2*pi*k*t/T);
end
```