SIGNALS AND SYSTEMS I
Computer Assignment 1

In MATLAB, signals are represented by column vectors or as columns in matrices. Row vectors can be used; however, MATLAB typically prefers column vectors. Vector or matrices can be entered into MATLAB using several different methods including typing an explicit list of elements, using MATLAB's built-in functions, and using user defined functions.

Generating Simple Signals

Short simple signals can easily be entered into MATLAB by typing a list of the signal's elements. To enter a matrix into MATLAB by this method

1. surround the entire list of elements with square brackets, [ ],
2. separate the row elements with spaces or commas
3. separate the column elements with semicolons, ; .

For example,

```
A = [1 2 3;4 5 6;7 8 9]
```

generates the matrix,

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

and

```
x = [1 2i 2+2*j (2+2)]
```

generates the row vector,

$$x = [1 \quad 2i \quad 2+2i \quad 4].$$

After a matrix or vector has been entered, MATLAB's built-in functions can rearrange them if desired. For example, if the built-in functions

```
.' and '
```

are placed after a vector or matrix, MATLAB performs a transpose or complex conjugate transpose operation of the vector or matrix, respectively; that is,

```
y = x.' and z = x'
```

generate

$$y = \begin{matrix} 1 \\ 2i \\ 2+2i \\ 4 \end{matrix} \quad \text{and} \quad z = \begin{matrix} 1 \\ -2i \\ 2-2i \\ 4 \end{matrix}.$$

The built-in functions,

```
flipup and fliplr
```

flip a matrix's rows in the up down direction and flip a matrix's columns in the left right direction, respectively; that is,

```
B = flipup(A) and C = fliplr(A)
```

generate

$$B=\begin{bmatrix}7 & 8 & 9\\ 4 & 5 & 6\\ 1 & 2 & 3\end{bmatrix} \text{ and } C=\begin{bmatrix}3 & 2 & 1\\ 6 & 5 & 4\\ 9 & 8 & 7\end{bmatrix}.$$

Vectors and matrices can also be combined as long as their dimensions agree. For example,

```
D = [A A.']
```

generates

$$D=\begin{bmatrix}1 & 2 & 3 & 1 & 4 & 7\\ 4 & 5 & 6 & 2 & 5 & 8\\ 7 & 8 & 9 & 3 & 6 & 9\end{bmatrix}$$

Individual matrix elements can be accessed by using the typical computer row column notation. For example,

```
f = D(1,2)  and  g = D(2,3)
```

generates

$$f=2 \text{ and } g=6.$$

Multiple elements can be accessed by using vectors for the rows or columns. For example,

```
h = D([1 3],2)
```

generates

$$h=\begin{bmatrix}2\\ 8\end{bmatrix}.$$

Because MATLAB was developed as a matrix program, operators such as +, -, * and ^, perform matrix operations and not necessarily signal operations. If a matrix operation does not correspond to the corresponding signal operation, a period is placed in front of the operation to perform signal operations. For example, if

```
x = [1 2 3].'  and  y = [4 3 2].'
```

and

```
a = x+y,  b = x-y,  and  c = 2*x
```

then

$$a=\begin{bmatrix}5\\ 5\\ 5\end{bmatrix},\ b=\begin{bmatrix}-3\\ -1\\ 1\end{bmatrix} \text{ and } c=\begin{bmatrix}2\\ 4\\ 6\end{bmatrix}.$$

Because addition, subtraction and scalar multiplication for matrices and signals render identical results, matrix addition, subtraction and scalar multiplication is used to perform signal addition, subtraction and scalar multiplication, respectively. On the other hand, because matrix multiplication is different from signal multiplication, signal multiplication uses the .* operation. For example, if

```
x = [1 2 3]  and  y = [3 4 5]
```

and

```
a = x.' * y,  b = x * y.',  c = x.' .* y.',  and  d = x.' .^2
```

then

$$a=\begin{bmatrix}2 & 4 & 5\\ 6 & 8 & 10\\ 9 & 12 & 15\end{bmatrix},\ b=26,\ c=\begin{bmatrix}3\\ 12\\ 15\end{bmatrix} \text{ and } d=\begin{bmatrix}1\\ 4\\ 9\end{bmatrix}.$$

As illustrated by the example, the operators, * and ^, are matrix operators, and the operators, .* and .^, are signal operators.

Exercises

1. Generate the following signals for $-5 \leq n \leq 5$ by typing an explicit list of elements

   a) $\delta[n]$, the unit impulse sequence.
   b) $u[n]$, the unit step sequence.
   c) $r(n) = n$
   d) $x = \{0,1,2,3,2,1,0,-1,-2,-3,-2\}$
   e) $y = \{-1,0,1,2,3,2,1,0,-1,-2\}$

   Plot your results using the **figure** and **stem** function. Use the syntax **stem(r,** signal) where **r** is the signal you created in part c). **figure(#)** opens a window numbered, #.

2. Using the signals that you created in exercise 1,
   a) append the signals x and y so that the resulting signal is a periodic signal. Plot your results using the **stem** function.
   b) Extract the 3rd sample of $r(n)$, $x(n)$, and $y(n)$; that is, print $r(3)$, $x(3)$, and $y(3)$.

3. Using the elementary signals that you created in exercise 1, generate the following new signals:

   a) $c[n] = u(-n)$
   b) $d[n] = x(n) + y(n)$
   c) $e[n] = x(n) - y(n)$
   d) $f[n] = 2x(n)\delta(n)$
   e) $g[n] = 2r(n)u(-n)$

   Plot your results using the **stem** function. Again, use the syntax **stem(time, signal)**.

Generating Signals using the Colon Operator

MATLAB's colon operator, :, can be used to easily define simple vectors (signals). Using these simple vectors, more complicated signals can easily be created. The colon operator's syntax is

```
start : increment (default = 1) : end
```

For example,

```
m = 10 : -2 : -10
```

generates

$$m = [10 \quad 8 \quad 6 \quad 4 \quad 2 \quad 0 \quad -2 \quad -4 \quad -6 \quad -8 \quad -11]$$

and

```
n = 0 : 10
```

generates

$$n = [0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \quad 10].$$

More complicated signals can then be generated using these vectors. For example,

```
x = (0.5).^n  and  y = sin(pi.*n/3)
```

generates the signals

$$x(n) = \left(\frac{1}{2}\right)^n \quad \text{and} \quad y(n) = \sin\left(\frac{\pi}{3}n\right)$$

for $0 \le n \le 10$.

The vectors generated using MATLAB's colon operator, :, can be also be used to access elements of a matrix. For example, if

```
n = 0 : 10
```

then

```
m = ([1:2:10])  or  m = ([1:2:end])
```

generates

$$m = [0 \quad 2 \quad 4 \quad 6 \quad 7 \quad 10]$$

Used by itself, the colon operator will include an entire row or column. For example, if

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

then

```
a = A(1,:)
```

generates

$$a = [1 \quad 2 \quad 3].$$

MATLAB has built-in functions that can easily create commonly used vectors and matrices. For example,

```
W = ones(2,3)
```

generates a 2x3 matrix of ones, that is,

$$W = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Similarly, the MATLAB built-in function

```
zeros(N,M)
```

generates a NxM matrix of zeros.

Exercises

4. Generate the following signals for $-25 \le n \le 25$.

   a) $\delta[n]$, the unit impulse sequence.
   b) $\upsilon[n]$, the unit step sequence.
   c) $x[n] = \left(\frac{5}{6}\right)^n$
   d) $y[n] = \left(-\frac{5}{6}\right)^n$
   e) $z[n] = \cos(\pi n / 5)$
   f) $f[n] = \cos(11\pi n / 5)$

   Plot your results using the **stem** function. Use the syntax **stem(n,** signal) where **n** is an appropriate time vector).

5. Using the signals that you generated in Exercise 3, generate the following new signals:

   a) $c(n) = z(n) - f(n)$
   b) $d[n] = x(2n)$
   c) $e(n) = u(n-5)$
   d) $g(n) = u(-n)$
   e) $h(n) = x(-2n+1)$

   Plot your results using the **stem** function. Use the syntax **stem(n,** signal) where **n** is an appropriate time vector).

6. Generate and plot $r(n) = e^{j\pi n/5}$