باسمه تعالی

دستورکار آزمایش دوم

عنوان: تولید و تفسیر سیگنال های مخابراتی

اهداف آزمایش:

- استفاده از مطلب برای تولید سیگنال های مهم در نظریه مخابرات
- یادگیری نحوه انجام عملیات روی سیگنال ها
- درک بیشتر سیگنال های مخابراتی و خواص آنها

تولید سیگنال ها

سیگنال ها به صورت توابعی از یک یا چند متغیر مستقل بیان می شوند. ما اغلب متغیر مستقل را زمان در نظر می گیریم. در نتیجه می توان گفت یک سیگنال تابعی از زمان است. دستورات زیر را در یک m-file بنویسید و آنرا اجرا کرده نتایج را ملاحظه فرمایید.

سیگنال زمان-پیوسته

نکته: برای رسم این نوع سیگنال ها از دستور plot استفاده نمایید.

```
% Example 2.1
% Generation of sinusoidal signals
% 2sin( 2πt-π/2)
t=[-5:0.01:5];
x=2*sin((2*pi*t)-(pi/2));
plot(t,x)
grid on;
axis([-6 6 -3 3])
ylabel ('x(t)')
xlabel ('Time(sec)')
title ('Figure 2.1')
```

خروجی را ملاحظه فرمایید. حال مقدار فاز سیگنال را تغییر داده و پس از ذخیره m-file آنرا اجرا کرده و تفاوت ها را ملاحظه فرمایید.

سیگنال زمان-گسسته

نکته: برای رسم این نوع سیگنال ها از دستور stem استفاده نمایید.

```
% Example 2.2
% Generation of discrete time signals
n = [-5:5];
x = [0 0 1 1 -1 0 2 -2 3 0 -1];
y=sign(n);
subplot(2,1,1)
stem (n,x);
axis ([-6 6 -3 3]);
xlabel ('n');
ylabel ('x[n]');
title ('Figure 2.2.1');

subplot(2,1,2)
stem(n,y)
axis ([-5 5 -1 2]);
xlabel (‘n’);
ylabel (‘y[n]’);
title (‘Figure 2.2.2’);
```

ضربه واحد گسسته

این دنباله به صورت زیر تعریف می شود:

$$\delta[n]=\begin{cases}1, & n=0\\ 0, & n\neq 0\end{cases}$$

حال تابعی با نام impseq می سازیم و در آزمایش های بعدی از این تابع استفاده می کنیم. برنامه مطلب که در زیر آمده دنباله $\delta[n-n_0]$ را برای بازه زمانی n_1 تا n_2 بدست آورده و بردار زمان را به n و مقادیر

دنباله را در این زمان ها به بردار x به عنوان خروجی می فرستد. (در مطلب زیرنویس نداریم لذا n_1 را با n1 نشان می دهیم و ....)

```
function [x,n] = impseq(n0,n1,n2)
% Generates x(n) = delta (n-n0); n1<=n,n0 <= n2
% x[x,n] = imseq(n0,n1,n2)
% n0 = impulse position, n1 = starting index, n2 = ending index
if ((n0 < n1) | (n0 > n2) | (n1 > n2))
error('arguments must satisfy n1 <= n0 <= n2')
end
n = [n1:n2];
% x = [zeros(1,(n0-n1)),1,zeros(1,(n2-n0))];
x = [(n-n0) == 0];
```

دستور error متن داخل پرانتز که بین دو تک گیومه نوشته شده است را در پنجره فرمان مطلب به رنگ قرمز نمایش می دهد. آخرین دستور این تابع بدین شکل عمل می کند که برابری دنباله n با مقدار n0 را بررسی کرده و در نقاطی که مقدار بردار با n0 برابر است مقدار ۱ قرار می دهد و در غیر این صورت مقدار ۰ قرار می دهد. در نهایت، بردار حاصل را در x ذخیره می کند. دستور غیر فعال شده عمل فوق را به صورتی دیگر انجام می دهد.

تمرین ۱) برای سه مقدار متفاوت از n0 تابع فوق را فراخوانی کرده و x را بر حسب n رسم نمایید. دقت کنید برای اینکه مقدار ۱ ضربه در شکل قرار گیرد باید n0 بزرگتر یا مساوی n1 و کوچکتر یا مساوی n2 باشد.

تابع پله واحد گسسته

این دنباله به صورت زیر تعریف می شود:

$$u[n] = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases}$$

حال تابعی با نام stepseq می سازیم و در آزمایش های بعدی از این تابع استفاده می کنیم. برنامه مطلب که در زیر آمده دنباله $u[n-n_0]$ را برای بازه زمانی n_1 تا n_2 بدست آورده و بردار زمان را به n و مقادیر دنباله را در این زمان ها به بردار x به عنوان خروجی می فرستد.

```
function [x,n] = stepseq(n0,n1,n2)
% Generates x(n) = u(n-n0); n1 <= n, n0<=n2
% [x,n] = stepseq(n0,n1,n2)
if ((n0 < n1) | (n0 > n2) | (n1 > n2))
error('arguments must satisfy n1 <= n0 <= n2')
end
n = [n1:n2];
% x = [zeros(1,(n0-n1)),ones(1,(n2-n0+1))];
x = [(n-n0) >= 0];
```

آخرین دستور این تابع بدین شکل عمل می کند که در نقاطی که مقدار بردار n بزرگتر یا مساوی n0 است مقدار ۱ قرار می دهد و در غیر این صورت مقدار ۰ قرار می دهد. در نهایت، بردار حاصل را در x ذخیره می کند. دستور غیر فعال شده عمل فوق را به صورتی دیگر انجام می دهد.

تمرین ۲) برای سه مقدار متفاوت از n0 تابع فوق را فراخوانی کرده و x را بر حسب n رسم نمایید. دقت کنید برای اینکه مقدار ۱ تابع پله واحد در شکل قرار گیرد باید n0 بزرگتر یا مساوی n1 و کوچکتر یا مساوی n2 باشد.

دنباله نمایی با مقدار حقیقی (نمایی حقیقی)

این دنباله به صورت زیر تعریف می شود:

$$x[n] = a^n$$

که در آن a یک عدد حقیقی است. ناگفته پیداست که اگر n یک بردار باشد باید از عملگر آرایه ای نقطه یعنی '.^' برای به توان رساندن استفاده نمود. حال کد زیر را در پنجره فرمان مطلب نوشته و اجرا کنید.

```
n = [0:10];
x = (0.9).^n;

stem(n,x)
```

تمرین ۳) به جای 0.9 یکبار 0.9- یکبار 1.5 یکبار 1.5- و در نهایت 1 و 1- قرار داده و نتایج خروجی را ملاحظه فرمایید.

دنباله نمایی با مقدار مختلط (نمایی مختلط)

این دنباله به صورت زیر تعریف می شود:

$$x[n] = e^{(a+jb)n}$$

که در آن a و b اعدادی حقیقی هستند و a ضریب تضعیف و b فرکانس برحسب رادیان است.

تمرین ۴) حال کد زیر را در پنجره فرمان مطلب نوشته و اجرا کنید.

```
n = [0:10];
x = exp((2+3j)*n);

subplot(2,2,1)

stem(n,real(x))

title('Real Value')

subplot(2,2,3)

stem(n,imag(x))

title('Imaginary Value')

subplot(2,2,2)

stem(n,abs(x))

title('Absolute Value')

subplot(2,2,4)

stem(n,angle(x))

title('Angle of X')
```

دنباله های تصادفی

خیلی از دنباله های عملی را نمی توان به صورت توابع ریاضی مانند آنچه در بالا دیدید توصیف کرد. این دنباله ها، دنباله های تصادفی نامیده می شوند که به عواملی نظیر تابع چگالی احتمال یا گشتاورهای آماری شان وابسته هستند. در مطلب دو دستور برای تولید دنباله های تصادفی وجود دارد. دستور rand(1,N) یک دنباله تصادفی با طول N تولید می کند که مقادیر عناصر آن به طور یکنواخت بین ۰ و ۱ توزیع شده است. دستور randn(1,N) یک دنباله تصادفی با طول N تولید می کند که مقادیر عناصر آن از توزیع گوسی (نرمال) با میانگین صفر و واریانس ۱ پیروی می کنند.

تمرین ۵) حال دستورات زیر ار در مطلب اجرا کنید.

```
% example 2.3
%Generation of random sequence
n = [0:10];
x = rand (1, length (n));
y = randn (1, length (n));
plot (n,x) ;
grid on;
hold on;
plot(n,y,'r');
ylabel ('x & y')
xlabel ('n')
title ('Figure 2.3')
```

دنباله های متناوب

دنباله های متناوب دنباله هایی هستند که یک الگوی مشخص را در بازه های زمانی مساوی تکرار می کنند. کوچکترین بازه تکرار دوره تناوب اصلی سیگنال نامیده می شود.

تمرین ۶) حال دستور زیر را در مطلب نوشته و اجرا کنید. (می توانید برای جلوگیری از تکرار این دستورات را در یک m-file وارد کرده و با نامی مشخص آنرا ذخیره نمایید.)

```
% Example 2.4
%Generation of periodic sequences Lab Manual of Analog & Digital
Communication n = [0:4];
x = [1 1 2 -1 0];
subplot (2,1,1);
```

```
stem (n,x);
grid on;
axis ([0 14 -1 2]);
xlabel ('n');
ylabel ('x(n)');
title ('Figure 2.4(a)');
xtilde = [x,x,x];
length_xtilde = length (xtilde);
n_new = [0:length_xtilde-1];
subplot (2,1,2);
stem (n_new,xtilde,'r');
grid on;
xlabel ('n');
ylabel ('perodic x(n)');
title ('Figure 2.4(b)');
```

تمرین ۷) با استفاده از اطلاعاتی که در درس سیگنال آموختید دوره تناوب سیگنال های گسسته زیر بیابید و با رسم آنها پاسخ خود را بررسی نمایید.

الف) $x[n] = \sin(n/10)$ ب) $x[n] = -5\cos(n\pi/10)$

ج) $x[n] = 1$ د) $x[n] = \sqrt[3]{5}\cos(\sqrt{2}n\pi/7)$

انجام عملیات روی سیگنال ها

جمع کردن دو سیگنال

در این حالت نمونه های سیگنال ها نظیر به نظیر با یکدیگر جمع می شوند. به عبارتی در هر زمان مقدار دو سیگنال در آن زمان با یکدیگر جمع می شوند. لازم به ذکر است که طول سیگنال ها و زمان های آن ها باید با یکدیگر برابر باشد. بدین منظور تابع مطلب زیر ابتدا زمان و طول دو سیگنال x1 و x2 را یکسان کرده و سپس آنها را با هم جمع می کند.

تمرین ۸) برنامه مطلب زیر را در یک m-file نوشته و ذخیره کنید.

توجه: n1 زمان های مربوط به دنباله x1 و n2 زمان های مربوط به دنباله x2 می باشد. تابع min کمینه مقادیر بردار داخل آنرا می دهد و اگر به آن ماتریس داده شود مقدار کمینه هر ستون آنرا محاسبه کرده و به عنوان یک بردار سطری با طولی برابر تعداد ستون های ماتریس داده شده بر می گرداند.

```
function [y,n] = sigadd(x1,n1,x2,n2)
% implement y(n) = x1(n) + x2 (n)
% [y,n] = sigadd (x1,n1,x2,n2)
% y = sum sequence over n, which include n1 and n2
% x1 = first sequence over n1
% x2 = second sequence over n2 (n2 can be different from n1)
n = min(min(n1),min(n2)): max(max(n1),max(n2)); %duration of y(n)
y1 = zeros(1,length(n)); % initialization
y2 = y1;
y1(find((n>=min(n1))&(n<=max(n1))==1))=x1; % x1 with duration of y
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2; % x2 with duration of y
y = y1 + y2;
```

توجه: دستور find اندیس عناصری را که عبارت داخل پرانتز برای آنها معتبر (درست) است بر می گرداند.

حال بگویید در هر خط از تابع فوق چه عملی انجام می شود. به عبارتی تابع فوق را خط به خط توضیح دهید.

تمرین ۹) دستورات زیر را در پنجره فرمان مطلب نوشته و اجرا کنید.

```
 % Example 2.5
% signal addition using sigadd function
clear;
clc;
n1 = [0:10];
x1 = sin (n1);
n2 = [-5:7];
x2 = 4*sin(n2);
[y,n] = sigadd(x1,n1,x2,n2);
subplot (3,1,1);
stem (n1,x1);
grid on;
axis ([-5 10 -5 5]);
xlabel ('n1'); ylabel ('x1(n)');
title ('1st signal');
subplot (3,1,2);
stem (n2,x2);
grid on; hold on;
axis ([-5 10 -5 5]);
```

```
xlabel ('n2'); ylabel ('x2(n)');
title ('2nd signal');
subplot (3,1,3); stem (n,y,'r');
grid on;

axis ([-5 10 -5 5]);
xlabel ('n'); ylabel ('y(n)');
title ('Added Signals');
```

ضرب کردن دو سیگنال

در این حالت نمونه های سیگنال ها نظیر به نظیر در یکدیگر ضرب می شوند. به عبارتی در هر زمان مقدار دو سیگنال در آن زمان در یکدیگر ضرب می شوند. لازم به ذکر است که طول سیگنال ها و زمان های آن ها باید با یکدیگر برابر باشد. بدین منظور تابع مطلب زیر ابتدا زمان و طول دو سیگنال x1 و x2 را یکسان کرده و سپس آنها را در یکدیگر ضرب می کند. ناگفته پیداست که در این حالت باید از ضرب عنصر محور یا همان ضرب آرایه ای '.*' استفاده نمود.

تمرین ۱۰) برنامه مطلب زیر را در یک m-file نوشته و ذخیره کنید.

```
function [y,n] = sigmult (x1,n1,x2,n2)
% implement y(n) = x1(n) * x2 (n)
% [y,n] = sigmult (x1,n1,x2,n2)
% y = product sequence over n, which include n1 and n2
% x1 = first sequence over n1
% x2 = second sequence over n2 (n2 can be different from n1)
n = min(min(n1),min(n2)): max(max(n1),max(n2)); %duration of y(n)

y1 = zeros(1,length(n)); % initialization
y2 = y1;
y1(find((n>=min(n1))&(n<=max(n1))==1))=x1; % x1 with duration of y
y2(find((n>=min(n2))&(n<=max(n2))==1))=x2; % x2 with duration of y
y = y1 .* y2;
```

تمرین ۱۱) دستورات زیر را در پنجره فرمان مطلب وارد کرده و اجرا نمایید.

```
% Example 2.6
```

```
% signal multiplication using sigmult function
clear;
clc;
n1 = [0:100];
x1 = sin (n1/10);
n2 = [-50:70];
x2 = 4*sin (n2/10);
[y,n] = sigmult(x1,n1,x2,n2);
subplot (3,1,1);
stem (n1,x1);
grid on;
axis ([-50 100 -5 5]);
xlabel ('n1');
ylabel ('x1(n)');
title ('1st signal');
subplot (3,1,2);
stem (n2,x2);
grid on;
hold on;
axis ([-50 100 -5 5]);
xlabel ('n2');
ylabel ('x2(n)');
title ('2nd signal');
subplot (3,1,3);
stem (n,y,'r');
grid on;
axis ([-50 100 -5 5]);
xlabel ('n');
ylabel ('y(n)');
title ('Multiplied Signals');
```

تمرین ۱۲) با استفاده از توابع impseq و stepseq سیگنال های زیر را برای مقادیر دلخواهی از n محاسبه کرده و رسم نمایید.

a. $x[n] = 2\sin(3n)\,\delta[n-1] + 2\cos(3n) + \delta[n-2]$

b. $x[n] = u[n] + 4\cos(3n) + 2\,\delta[n+1]$

c. $x[n] = n[u(n) - u(n-10)] + 10e^{-0.3(n-10)}[u(n-10) - u(n-20)]$

همواره شاد و پیروز باشید.