序列類型

南台科大　趙春棠

1. Unit Sample sequence

```
function [x,n]=impseq(n0,n1,n2)
%Generates x(n)=delta(n-n0); n1 <= n <= n2
% [x,n]=impseq(n0,n1,n2)
n=[n1:n2];x=(n==n0);
```

例：

```
Matlab: [x,n]=impseq(1,-2,5)
x =
     0     0     0     1     0     0     0     0
n =
    -2    -1     0     1     2     3     4     5
```

2. Unit Step sequence

```
function [x,n]=stepseq(n0,n1,n2)
%Generates x(n)=u(n-n0); n1 <= n <= n2
% [x,n]=stepseq(n0,n1,n2)
n=[n1:n2];x=(n>=n0);
```

例：

```
Matlab: [x,n]=stepseq(1,-2,5)
x =
     0     0     0     1     1     1     1     1
n =
    -2    -1     0     1     2     3     4     5
```

3. Real valued exponential sequence

例： $x(n)=(0.9)^n$

Matlab 做法一：`n=[-1:5]; x=(0.9)^n;`

```
??? Error using ==> ^
Matrix must be square.
```

Matlab 做法二：`n=[-1:5]; x=(0.9).^n;`　（此時要加點）

```
n =     -1      0      1      2      3      4      5
x = 1.1111 1.0000 0.9000 0.8100 0.7290 0.6561 0.5905
```

註：`x=(0.9)*n`

```
x = -0.9000      0    0.9000    1.8000    2.7000    3.6000    4.5000
```

註：`x=exp(n)`　(`x=exp.(n)` 此時加點反而錯哦！)

```
x =    0.3679    1.0000    2.7183    7.3891   20.0855   54.5982  148.4132
```

4. Complex-valued exponential sequence

例： $x(n)=\exp^{(2+3j)n}$

```
Matlab: n=[-1:2];   x=exp((2+3j)*n);
x = -0.1340 - 0.0191i   1.0000   -7.3151 + 1.0427i   52.4235 -15.2556i
```

5. Sinusoidal sequence

例：　$x(n)= 3\cos(0.1n\,\pi + \pi/3)$

Matlab: `n=[-1:5]; x=3*cos(0.1*n*pi+pi/3);`

```
x =    2.2294    1.5000    0.6237   -0.3136   -1.2202   -2.0074   -2.5981
```

6. Random Sequence

Matlab Case1: `seq=rand(1,5);`　　Note: [0,1]區間，”均勻” 分佈，長度爲 5

```
seq =    0.9501    0.2311    0.6068    0.4860    0.8913
```

Matlab Case2: seq=randn(1,5);
Note: [0,1]區間，平均值 0，變異數 1，”高斯”分佈，長度爲 5
seq = -0.4326 -1.6656 0.1253 0.2877 -1.1465

7. Periodic Sequence
例： 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
Matlab: x=[1 2 3 4 5]; xtilde=x'*ones(1,3); xtilde=xtilde(:);xtilde=xtilde';

x = 1 2 3 4 5

a=ones(1,3); => a= 1 1 1 ;
xtilde =
1 1 1
2 2 2
3 3 3
4 4 4
5 5 5 Note: after step: xtilde=x'*ones(1,3);

xtilde =

1
2
3
4
5
1
2
3
4
5
1
2
3
4
5 Note: after step: xtilde=xtilde(:);
xtilde =

1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 Note: after step: xtilde=xtilde';

序列運算

1. Signal Addition:
```
function [y,n]=sigadd(x1,n1,x2,n2)
% implements y(n)=x1(n)+x2(n)
% x1 and x2 are signal, n1 and n2 are index
n=min(min(n1),min(n2)):max(max(n1),max(n2)); % duration of y(n)
y1=zeros(1,length(n));y2=y1;                          % initialization
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;
```
例：n1= -2:2; x1=[1 2 3 4 5]; n2= -1:4; x2=[1 2 3 4 5 6]; [y,n]=sigadd(x1,n1,x2,n2);
n = -2 -1 0 1 2 3 4
y = 1 3 5 7 9 5 6

圖解： n= -2 -1 0 1 2 3 4
x1= 1 2 3 4 5
x2= 1 2 3 4 5 6
y=x1+x2 1 3 5 7 9 5 6

註：n = -1 3 4 2 0
(n>1)&(n<4)
ans = 0 1 0 1 0
find(ans==1)
ans = 2 4

註： y=[10 11 12 13 14 15]; n=[1 2 3]; x=[1 2 3]; y(n)=x;
y = 1 2 3 13 14 15

註： y=[10 11 12 13 14 15]; n=[4 5 6]; x=[1 2 3]; y(n)=x;
y = 10 11 12 1 2 3

2. Signal Multiplication:

```
function [y,n]= sigmult(x1,n1,x2,n2)
% implements y(n) = x1(n)*x2(n)
%
n = min(min(n1),min(n2)):max(max(n1),max(n2)); % duration of y(n)
y1 = zeros(1,length(n)); 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;
```

例：n1= -2:2; x1=[1 2 3 4 5]; n2= -1:4; x2=[1 2 3 4 5 6]; [y,n]=sigmult(x1,n1,x2,n2);
n = -2 -1 0 1 2 3 4
y = 0 2 6 12 20 0 0

圖解： n= -2 -1 0 1 2 3 4
x1= 1 2 3 4 5
x2= 1 2 3 4 5 6
y=x1*x2 0 2 6 12 20 0 0

註： y1=[1 2]; y2=[11 12]; y=y1*y2;
??? Error using ==> *
Inner matrix dimensions must agree.

註： y1=[1 2]; y2=[11 12]; y=y1.*y2;
y = 11 24

3. Scaling:

例： x=[1 2 3]; y=3*x;
y = 3 6 9

4. Shifing:

```
function [y,n] = sigshift(x,m,n0)
% implements y(n) = x(n-n0)    m: index of x,    n: index of y
n = m+n0; y = x;
```

例： m=[-2 –1 0 1 2 3]; x=[0 1 2 3 4 5]; [y,n]=sigshift(x,m,3);
m= -2 -1 0 1 2 3
x= 0 1 2 3 4 5
n = 1 2 3 4 5 6
y = 0 1 2 3 4 5

5. Folding:

```
function [y,n] = sigfold(x,n)
```

```
% implements y(n) = x(-n)
y = fliplr(x); n = -fliplr(n);
```

例： m=[-2 –1 0 1 2 3]; x=[0 1 2 3 4 5]; [y,n]=sigfold(x,m);

```
m=        -2   -1   0    1    2    3
x=         0    1    2    3    4    5
n =  -3   -2   -1   0    1    2
y =   5    4    3    2    1    0
```

6. Sample Summation:

例： m=[-2 –1 0 1 2 3]; x=[0 1 2 3 4 5]; s=sum(x(1:length(x)); Note: 其實 s=sum(x)即可

```
s =   15    Note: 0+1+2+3+4+5=15
```

7. Sample Products:

例： m=[-2 -1 0 1 2 3]; x=[-1 1 2 3 4 5]; s=prod(x(1:length(x))); Note: 其實 s=prod(x)即可

```
s =   -120   Note: s= -1*1*2*3*4*5 = -120
```

8. Signal Energy:

例： x=[1 1+j 1-j]; ex1=sum(x.*conj(x)) ; ex2= sum(abs(x).^2); Note: ex1 及 ex2 分表兩種做法

```
ex1 =     5       ex2 =    5.0000     Note: ex2=1^2+(1.414^2)+(1.414^2);
```

註：conj(x)

```
ans =     1.0000                    1.0000 - 1.0000i     1.0000 + 1.0000i
```

註：abs(x)

```
ans =     1.0000       1.4142       1.4142
```

9. Signal Power:

實數訊號之奇偶分解

$x_e(n)=[x(n)+x(-n)]/2 \quad x_o(n)=[x(n)-x(-n)]/2$

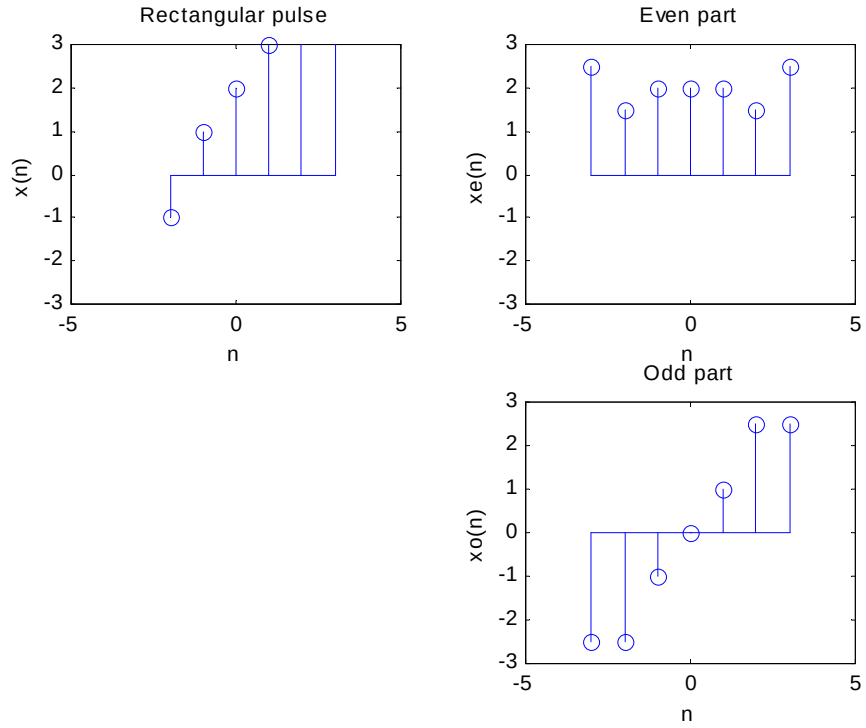
```
function [xe, xo, m] = evenodd(x,n)
% Real signal decomposition into even and odd parts
if any(imag(x) ~=0)
    error('x is not a real sequence')
end
m = -fliplr(n);
m1 = min([m,n]); m2=max([m,n]); m=m1:m2;
nm = n(1)-m(1); n1 = 1:length(n);
x1 = zeros(1,length(m));
x1(n1+nm) = x; x = x1;
xe = 0.5*(x + fliplr(x));
xo = 0.5*(x - fliplr(x));
```

例：n=[-2 -1 0 1 2 3]; x=[-1 1 2 3 4 5]; [xe,xo,m]=evenodd(x,n);

n=		-2	-1	0	1	2	3
x=		-1	1	2	3	4	5
m =	-3	-2	-1	0	1	2	3
xe =	2.5000	1.5000	2.0000	2.0000	2.0000	1.5000	2.5000
xo =	-2.5000	-2.5000	-1.0000	0	1.0000	2.5000	2.5000

圖示：

```
figure(1); clf
 subplot(2,2,1); stem(n,x); hold on; plot(n,0.*n); title('Rectangular pulse')
 xlabel('n'); ylabel('x(n)'); axis([-5,5,-3,3])
 subplot(2,2,2); stem(m,xe); hold on; plot(m,0.*m);title('Even part')
 xlabel('n'); ylabel('xe(n)'); axis([-5,5,-3,3])
 subplot(2,2,4); stem(m,xo); hold on; plot(m,0.*m);
 title('Odd part');
 xlabel('n'); ylabel('xo(n)'); axis([-5,5,-3,3])
```



Convolution

x[m]=[1 1 1 0 0 0]; (System I/P)

h[n]=[0.7^0 0.7^1 0.7^2 0.7^3 0.7^4]; （Impulse Response）

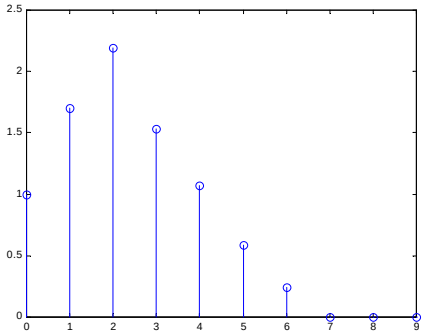
y[k]=? (System O/P)

Case 1: m=0:5; n=0:4; (k=(0+0):(5+4)=0:9)

Matlab: x=[1 1 1 0 0 0]; n=0:4; h=(0.7).^n; y=conv(x,h);

y =1.0000 1.7000 2.1900 1.5330 1.0731 0.5831 0.2401 0 0 0

Matlab: (O/P y[k] Sketch) k=0:9; stem(k,y);



Case 2: m=0:5; n=2:6; (l=(0+2):(5+6)=2:11)

如果仍用以上 conv 指令，則需自行決定 k，以下介紹直接使用 conv_m 指令

function [y,ny]=conv_m(x,nx,h,nh)

% Modified convolution routine for signal processing

%

nyb=nx(1)+nh(1); nye=nx(length(x))+nh(length(h));

ny= [nyb:nye];

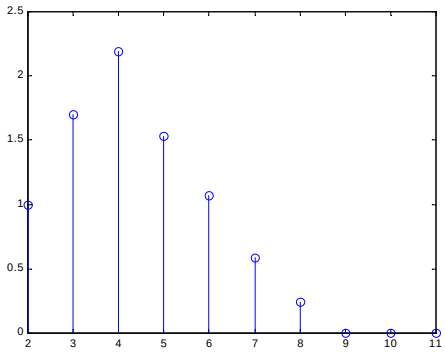
y= conv(x,h);

Matlab: x=[1 1 1 0 0 0]; nx=0:5; n=0:4; h=(0.7).^n; nh=2:6; [y,ny]=conv_m(x,nx,h,nh);

ny = 2 3 4 5 6 7 8 9 10 11 (與預測同)

y= 與 Case 1 中同

Matlab: (O/P y[ny] Sketch) stem(ny,y);



Case 3: 以上系統可表爲 y(n)-0.7y(n-1)=x(n)

先考慮以上 **Case1** 情形：

Matlab: b=[1]; a=[1 –0.7]; x=[1 1 1 0 0 0]; y=filter(b,a,x);

y = 1.0000 1.7000 2.1900 1.5330 1.0731 0.7512

咦！與前述結果不同，若將 x 改爲 x=[1 1 1 0 0 0 0 0 0]; 則得

y = 1.0000 1.7000 2.1900 1.5330 1.0731 0.7512 0.5258
0.3681 0.2577

註：y=filter(b,a,x) 指令有一個特點，那就是**輸出 y 的長度與輸入 x 相同**，發現了嗎？

釋疑：”以上系統可表爲 y(n)-0.7y(n-1)=x(n)”這句話是有問題的，因爲表成如此的差分方程式，系統脈衝響應 h(n)爲無限長，如本 Case 的模擬結果。

Case 4: 以上系統可表爲 $y(n)= 0.7^0 x(n)+0.7^1 x(n-1)+0.7^2 x(n-2)+0.7^3 x(n-3)+0.7^4 x(n-4)$

如此爲 FIR 形式

Matlab: n=0:4; b=(0.7).^n; a=[1]; x=[1 1 1 0 0 0 0 0 0]; y=filter(b,a,x);

y = 1.0000 1.7000 2.1900 1.5330 1.0731 0.5831 0.2401 0 0

看到了嗎？這樣的結果就與 Case1 相同了。

Correlation

訊號 1： x(nx)

Matlab: nx=[-2:3]; x=[3,11,7,0,1,4]; stem(nx,x);

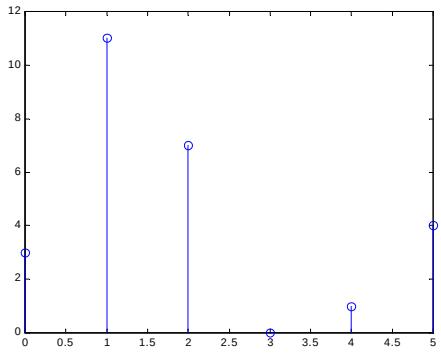
訊號 2： y(n)=x(n-2)+w(n) ;w(n):平均 0、變異數 1 之高斯分佈

Matlab: [y,ny]=sigshift(x,nx,2); w=randn(1,length(y)); nw=ny; stem(ny,y);

ny =nw= 0 1 2 3 4 5

y = 3 11 7 0 1 4

w = 2.1832 -0.1364 0.1139 1.0668 0.0593 -0.0956



Matlab: [y,ny] = sigadd(y,ny,w,nw);

ny = 同上

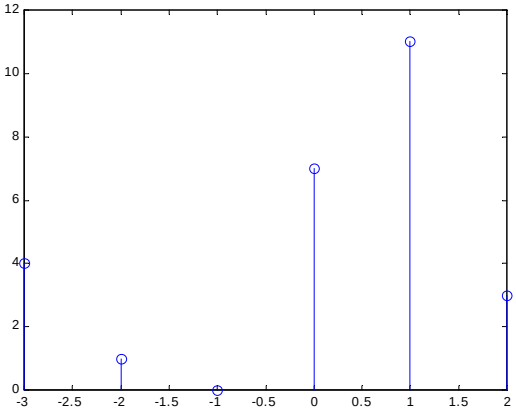
y = 5.1832 10.8636 7.1139 1.0668 1.0593 3.9044

取訊號 1 之 folding:

Matlab: (取 x(-n)) [x,nx] = sigfold(x,nx);

nx = -3 -2 -1 0 1 2

x = 4 1 0 7 11 3



取 Convulution: rxy=y(n)*x(-n)

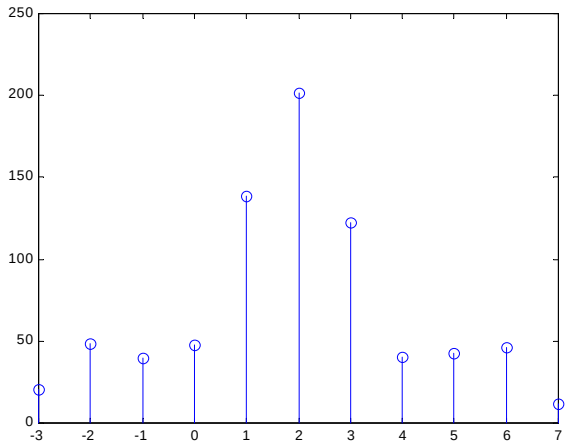
做法一：(建議使用)

Matlab: [rxy,nrxy] = conv_m(y,ny,x,nx); stem(nrxy,rxy); % crosscorrelation

nrxy = -3 -2 -1 0 1 2 3 4 5 6 7

rxy = 20.7327 48.6376 39.3193 47.6633 138.3642 201.5234 122.2158

40.4912 42.1829 46.1257 11.7131



做法二：

Matlab:　rxy=xcorr(x,y);

x =　　3　　11　　7　　0　　1　　4　(未經 folding 的 x(n)哦！)

y =　　5.1832　　10.8636　　7.1139　　1.0668　　1.0593　　3.9044

rxy =　　20.7327　　48.6376　　39.3193　　47.6633　　138.3642　　201.5234　　122.2158

　　40.4912　　42.1829　　46.1257　　11.7131

優點：x(n)不需經過 folding，**缺點**：無提供時間訊息