离散序列的基本运算

MATLAB子函数

1.find

功能： 寻找非零元素的索引号。

调用格式：

find((n>＝min(n1))&(n<＝max(n1)))；在符合关系运算条件的范围内寻找非零元素的索引号。

2.fliplr

功能： 对矩阵行元素进行左右翻转。

调用格式：

x1＝fliplr(x)；将x的行元素进行左右翻转，赋给变量x1。

3.conv

功能： 进行两个序列间的卷积运算。

调用格式：

y＝conv(x，h)；用于求取两个有限长序列x和h的卷积，y的长度取x、h长度之和减1。

例如，x(n)和h(n)的长度分别为M和N，则

y＝conv(x，h)

y的长度为N＋M－1。

使用注意事项： conv默认两个信号的时间序列从n＝0开始，因此默认y对应的时间序号也从n＝0开始。

4.sum

功能：求各元素之和。

调用格式：

Z＝sum(x)；求各元素之和，常用于等宽数组求定积分。

5.hold

功能： 控制当前图形是否刷新的双向切换开关。

调用格式：

holdon；使当前轴及图形保持而不被刷新，准备接受此后将绘制的新曲线。

holdoff；使当前轴及图形不再具备不被刷新的性质。

6.pause

功能： 暂停执行文件。

调用格式：

pause；暂停执行文件，等待用户按任意键继续。

pause(n)；在继续执行之前，暂停n秒。

三、实验原理

离散序列的时域运算包括信号的相加、相乘，信号的时域变换包括信号的移位、反折、倒相及信号的尺度变换等。

在MATLAB中，离散序列的相加、相乘等运算是两个向量之间的运算，因此参加运算的两个序列向量必须具有相同的维数，否则应进行相应的处理。

下面用实例介绍各种离散序列的时域运算和时域变换的性质。

1.序列移位

将一个离散信号序列进行移位，形成新的序列：

$$x_1(n)=x(n-m)$$

当m>0时，原序列x(n)向右移m位，形成的新序列称为x(n)的延时序列；当m<0时，原序列x(n)向左移m位，形成的新序列称为x(n)的超前序列。

例3-1 $x_1(n)=u(n+6)$ $(-10<n<10)$

$x_2(n)=u(n-4)$ $(-10<n<10)$

编写一个MATLAB程序，对u(n)序列进行移位，由图3-1比较三个序列之间的关系。

```
n1＝－10；n2＝10；
k0＝0；k1＝－6；k2＝4；
n＝n1：n2；                    %生成离散信号的时间序列
x0＝［n>＝k0］；               %生成离散信号x0(n)
x1＝［(n－k1)>＝0］；%生成离散信号x1(n)
x2＝［(n－k2)>＝0］；%生成离散信号x2(n)
```

```
subplot(3，1，1)，stem(n，x0，′filled′，′k′)；
axis(［n1，n2，1.1*min(x0)，1.1*max(x0)］)；
ylabel(′u(n)′)；
subplot(3，1，2)，stem(n，x1，′filled′，′k′)；
axis(［n1，n2，1.1*min(x1)，1.1*max(x1)］)；
ylabel(′u(n＋6)′)；
subplot(3，1，3)，stem(n，x2，′filled′，′k′)；
axis(［n1，n2，1.1*min(x2)，1.1*max(x2)］)；
ylabel(u′(n－4)′)；
```

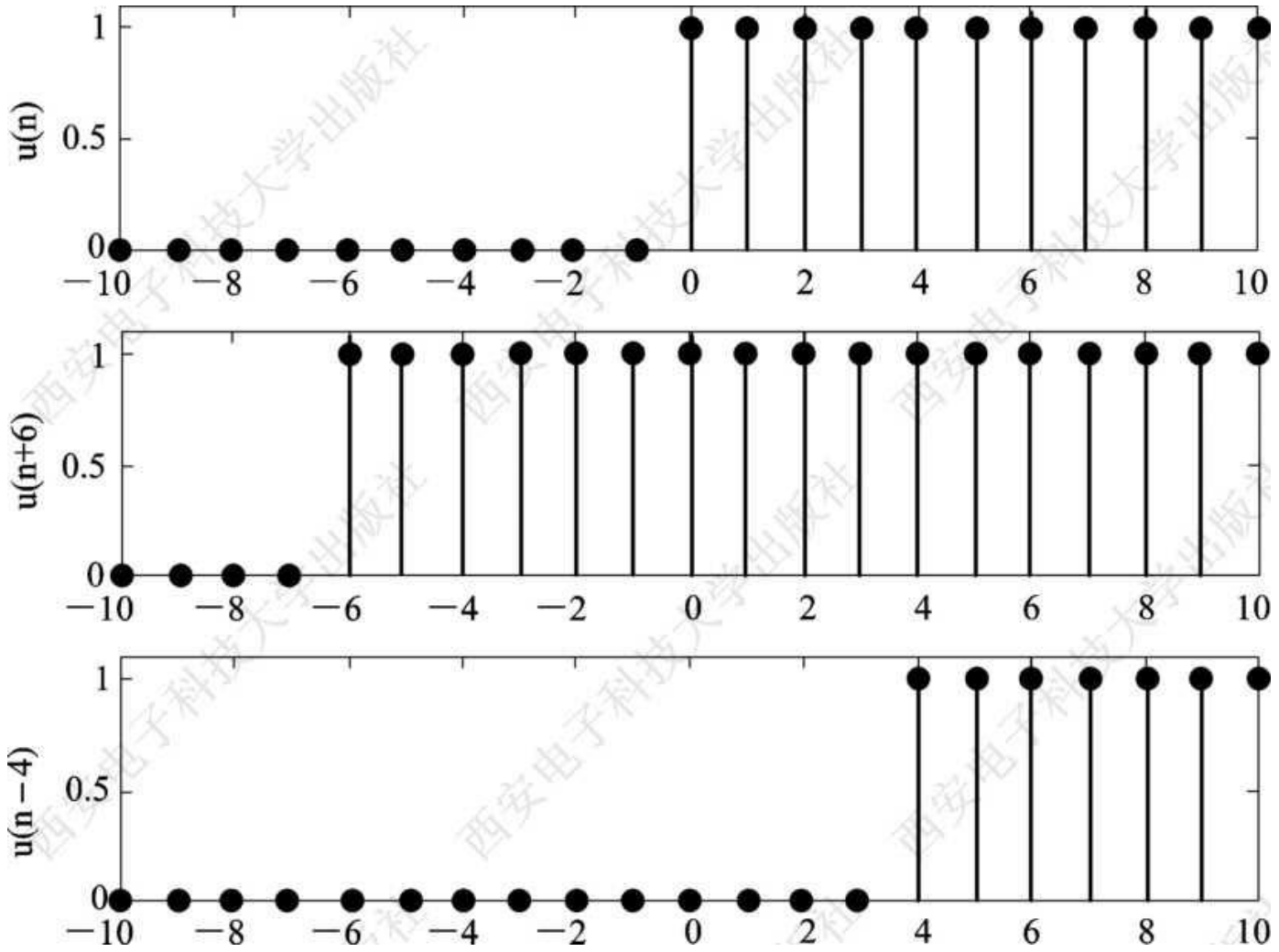

图3-1 u(n)及其位移序列u(n＋6)和u(n－4)

例3-2 已知一正弦信号：

$$x(n)=2\sin\frac{2\pi n}{10}$$

求其移位信号x(n－2)和x(n＋2)在－2<n<10区间的序列波形。

解 MATLAB程序如下：

```
n＝－2：10；n0＝2；n1＝－2；
x＝2*sin(2*pi*n/10);                %建立原信号x(n)
x1＝2*sin(2*pi*(n－n0)/10);    %建立x(n－2)信号
x2＝2*sin(2*pi*(n－n1)/10);   %建立x(n＋2)信号
```

```
subplot(3，1，1)，stem(n，x，′filled′，′k′)；
ylabel(′x(n)′)；
subplot(3，1，2)，stem(n，x1，′filled′，′k′)；
ylabel(′x(n－2)′)；
subplot(3，1，3)，stem(n，x2，′filled′，′k′)；
ylabel(′x(n＋2)′)；
```

结果如图3-2所示。

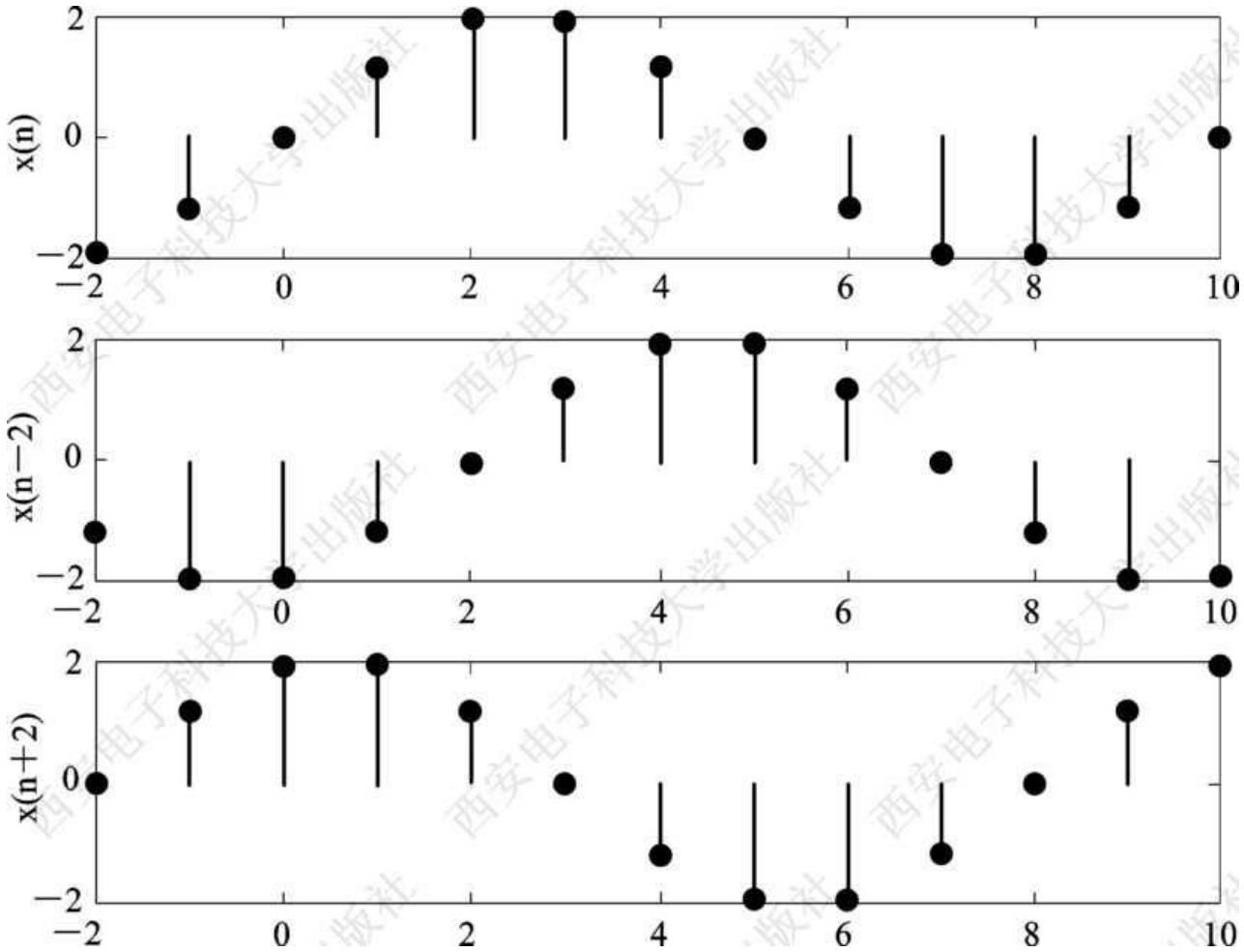

图3-2 正弦信号x(n)、x(n－2)和x(n＋2)

2.序列相加

两个离散序列相加是指两个序列中相同序号n(或同一时刻)的序列值逐项对应相加，构成一个新的序列：

$$x(n)=x_1(n)+x_2(n)$$

情况1 参加运算的两个序列具有相同的维数。

例3-3 求x(n)＝δ(n－2)＋δ(n－4)　　(0<n<10)。

解 MATLAB程序如下：

```
n1＝0；n2＝10；n01＝2；n02＝4；　　%赋初值
n＝n1：n2；
x1＝［(n－n01)＝＝0］；　　%建立δ(n－2)序列
x2＝［(n－n02)＝＝0］；　　%建立δ(n－4)序列
x3＝x1＋x2；
subplot(3，1，1)；stem(n，x1，′filled′)；
axis(［n1，n2，1.1*min(x1)，1.1*max(x1)］)；
ylabel(′δ(n－2)′)；
```

```
subplot(3，1，2)；stem(n，x2，′filled′)；
axis(［n1，n2，1.1*min(x2)，1.1*max(x2)］)；
ylabel(′δ(n－4)′)；
subplot(3，1，3)；stem(n，x3，′filled′)；
axis(［n1，n2，1.1*min(x3)，1.1*max(x3)］)；
ylabel(′δ(n－2)＋δ(n－4)′)；
```

结果如图3-3所示。

情况2： 参加运算的两个序列的维数不同。

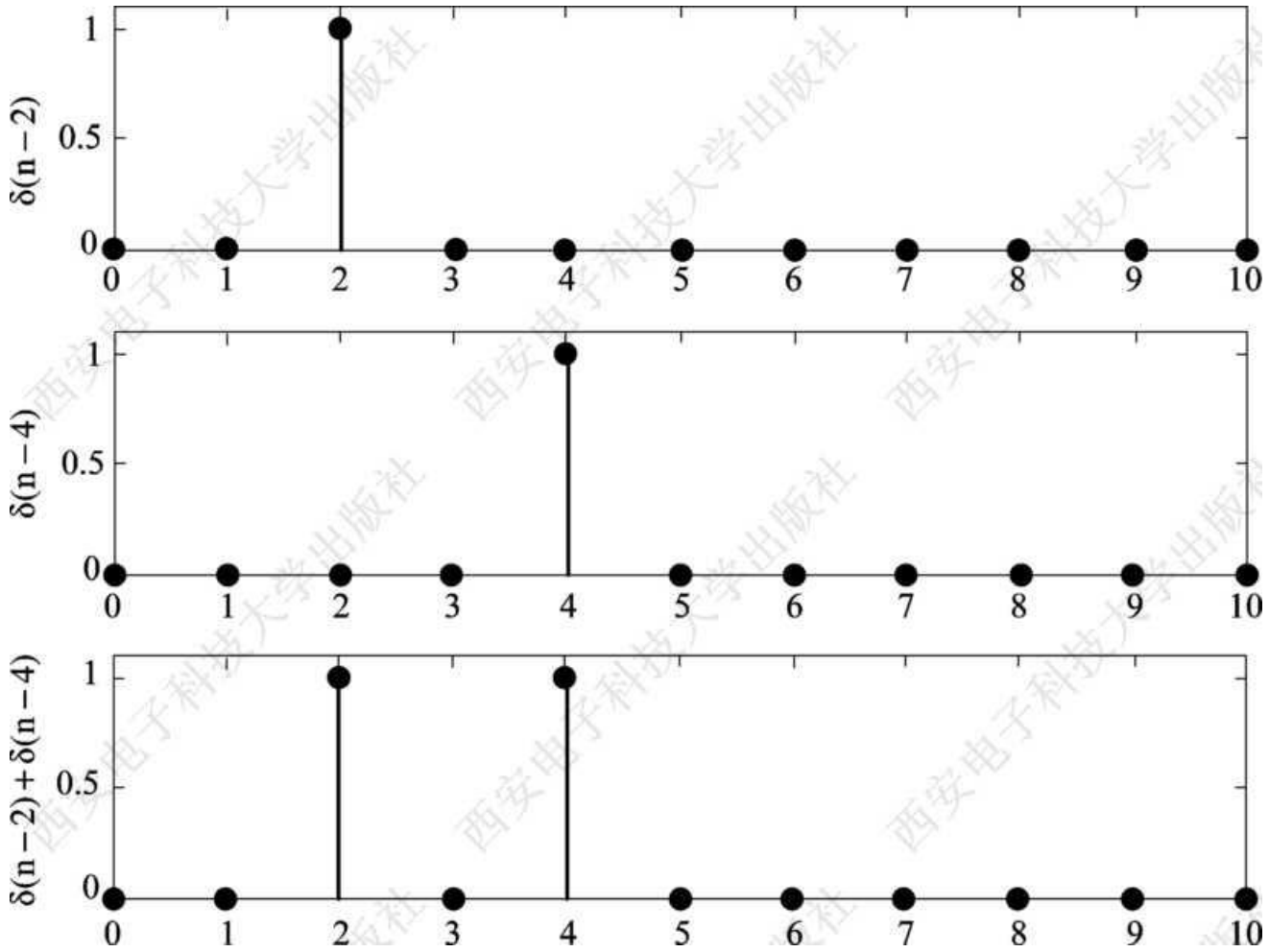

图3-3 δ(n－2)和δ(n－4)序列相加

例3-4 已知　　$x_1(n)=u(n+2)$　　$(-4<n<6)$

$x_2(n)=u(n-4)$　　$(-5<n<8)$

求

$$x(n)=x_1(n)+x_2(n)$$

解 要求上述两个序列之和，必须对长度较短的序列补零，同时保持位置一一对应。MATLAB程序如下：

```
n1＝－4：6；n01＝－2；
x1＝［(n1－n01)>＝0］；　　%建立x1信号
n2＝－5：8；n02＝4；
x2＝［(n2－n02)>＝0］；%建立x2信号
```

n＝min(［n1，n2］)：max(［n1，n2］)；%为x信号建立时间序列n

N＝length(n)；%求时间序列n的点数N

y1＝zeros(1，N)；y2＝zeros(1，N)；%新建一维N列的y1、y2全0数组

y1(find((n>＝min(n1))&(n<＝max(n1))))＝x1； %为y1赋值

y2(find((n>＝min(n2))&(n<＝max(n2))))＝x2；%为y2赋值

x＝y1＋y2；

stem(n，x，′filled′)；

axis(［min(n)，max(n)，1.1*min(x)，1.1*max(x)］)；

结果如图3-4所示。

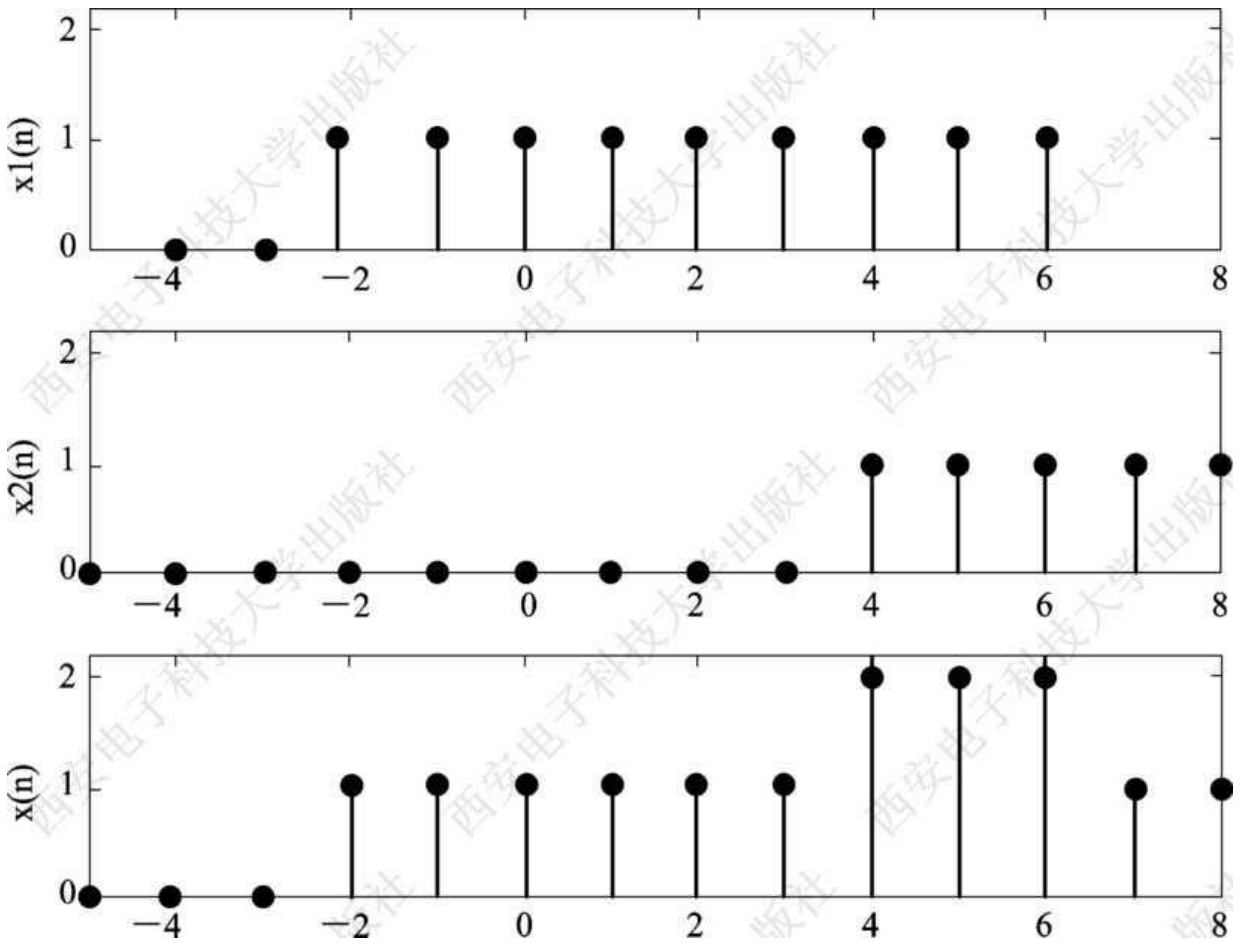

图3-4 序列维数不同的x1(n)和x2(n)相加

3.序列相乘

两个离散序列相乘是指两个序列中相同序号n(或同一时刻)的序列值逐项对应相乘，构成一个新的序列：

$$x(n)=x_1(n)\times x_2(n)$$

同样存在着序列维数相同和不同两种情况，处理方法与序列相加相同。

例3-5 已知信号：

$$x_1(n)=3e^{-0.25n}\ (-4<n<10)$$

$$x_2(n)=u(n+1)\ (-2<n<6)$$

求

$$x(n)=x_1(n)\times x_2(n)$$

解 MATLAB程序如下：

```
n1＝－4：10；
x1＝3*exp(－0.25*n);          %建立x1信号
n2＝－2：6；n02＝－1；
```

```
x2＝［(n2－n02)>＝0］；%建立x2信号
n＝min(［n1，n2］)：max(［n1，n2］)；%为x信号建立时间序列n
N＝length(n);                %求时间序列n的点数N
y1＝zeros(1，N)；%新建一维N列的y1全0数组
y2＝zeros(1，N)；%新建一维N列的y2全0数组
y1(find((n>＝min(n1))&(n<＝max(n1))))＝x1；  %为y1赋值
y2(find((n>＝min(n2))&(n<＝max(n2))))＝x2；%为y2赋值
x＝y1.*y2;
```

结果如图3-5所示。

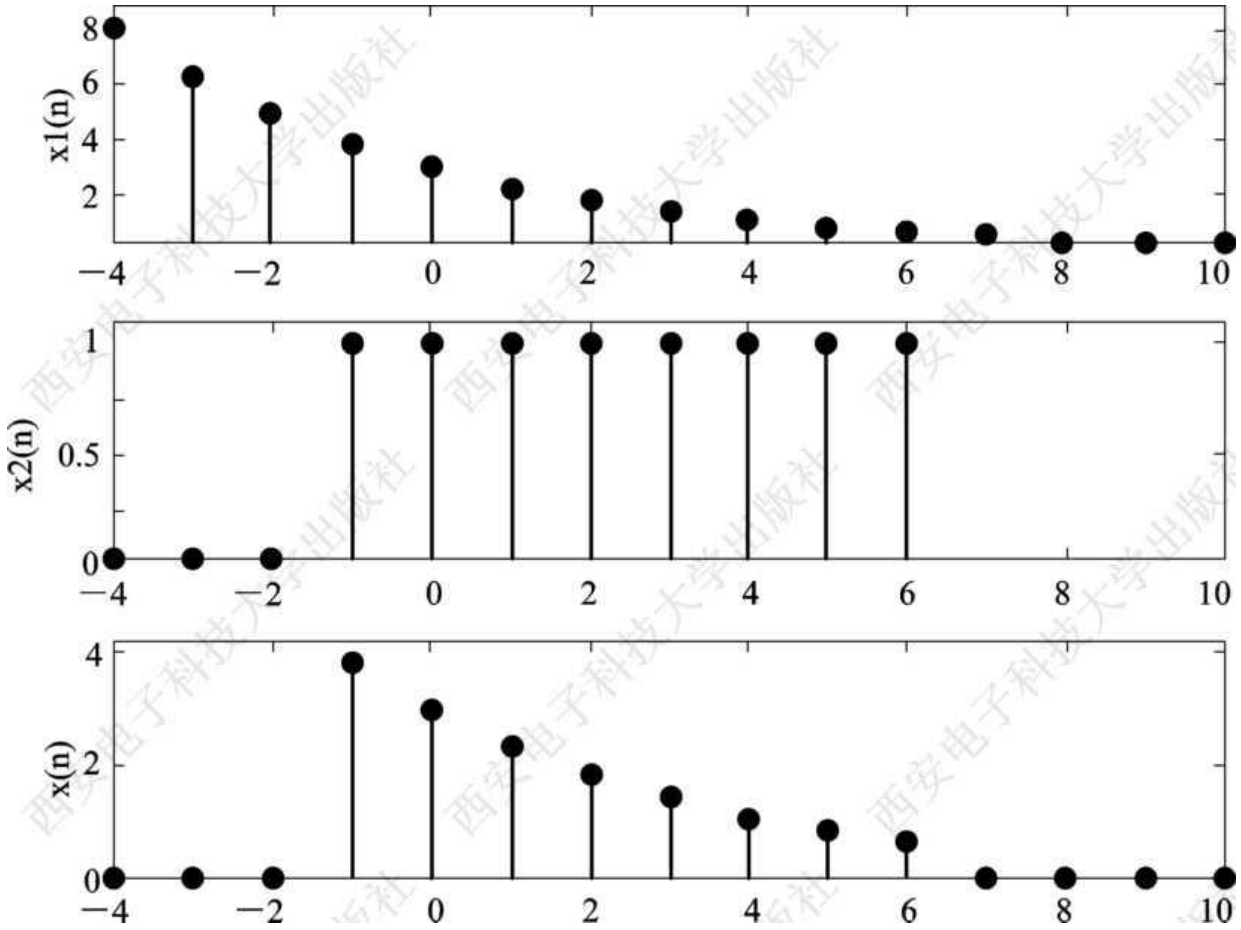

图3-5 序列x1(n)和x2(n)相乘

4.序列反折

离散序列反折是指离散序列的两个向量以零时刻的取值为基准点，以纵轴为对称轴反折。在MATLAB中提供了fliplr函数，可以实现序列的反折。

例3-6 已知一个信号：

$$x(n)=e^{-0.3*n}\ (-4<n<4)$$

求它的反折序列x(－n)。

解 MATLAB程序如下：

```
n＝－4：4;
x＝exp(－0.3*n);
```

```
x1＝fliplr(x)；
n1＝－fliplr(n)；
subplot(1，2，1)，stem(n，x，′filled′)；
title(′x(n)′)；
subplot(1，2，2)，stem(n1，x1，′filled′)；
title(′x(－n)′)；
```

结果如图3-6所示。

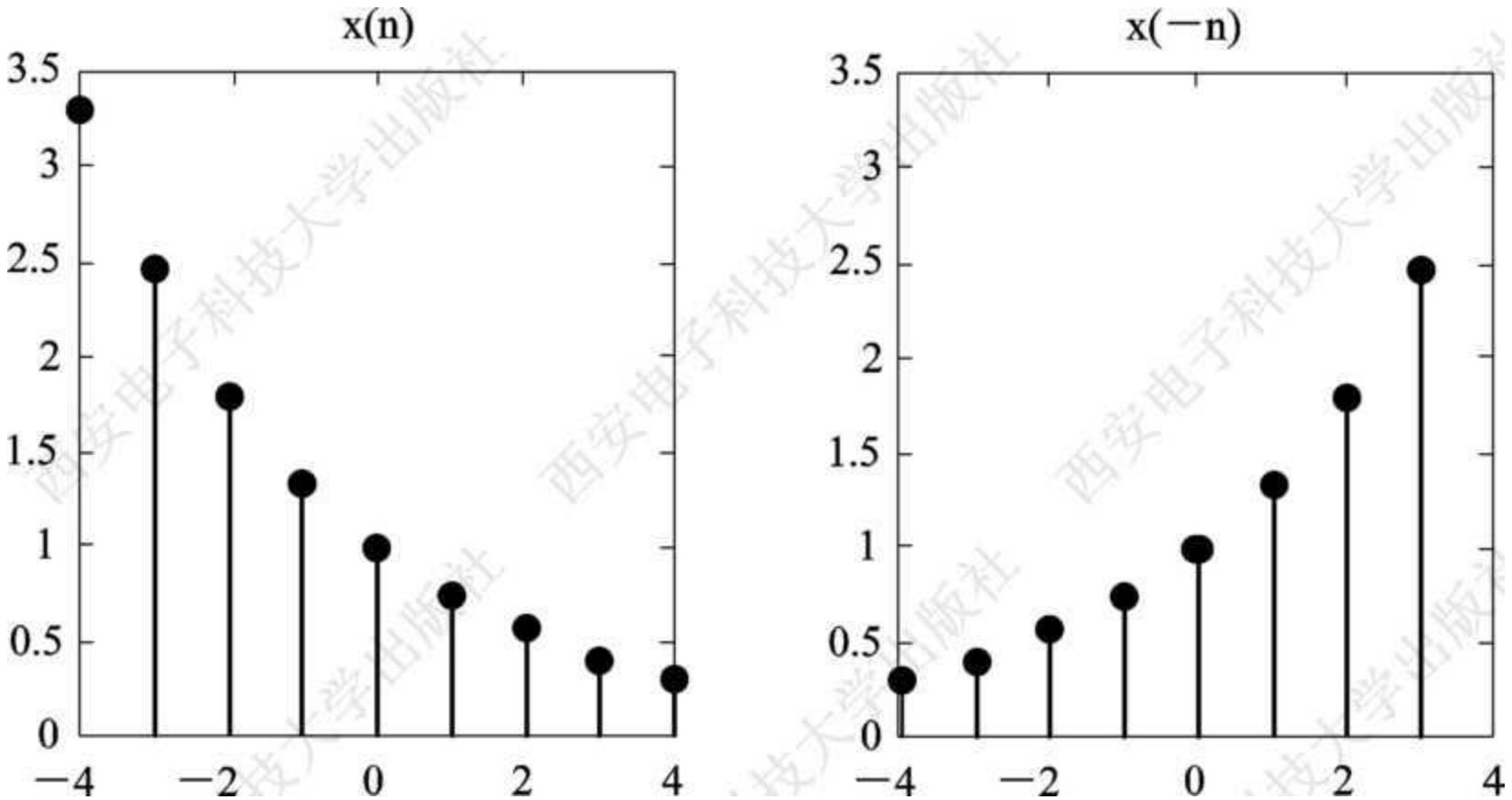

图3-6 序列x(n)和x(－n)反折序列

5.序列倒相

离散序列倒相是求一个与原序列的向量值相反，对应的时间序号向量不变的新的序列。

例3-7 将例3-6中信号：

$$x(n)=e^{-0.3*n}\ (-4<n<4)$$

倒相。

解 MATLAB程序如下：

```
n＝－4：4;
x＝exp(－0.3*n);
x1＝－x;
```

```
subplot(1，2，1)，stem(n，x，′filled′)；
title(′x(n)′)；
axis(［min(n)，max(n)，1.1*min(x1)，1.1*max(x)］)；
subplot(1，2，2)，stem(n，x1，′filled′)；
title(′－x(n)′)；
axis(［min(n)，max(n)，1.1*min(x1)，1.1*max(x)］)；
```

结果如图3-7所示。

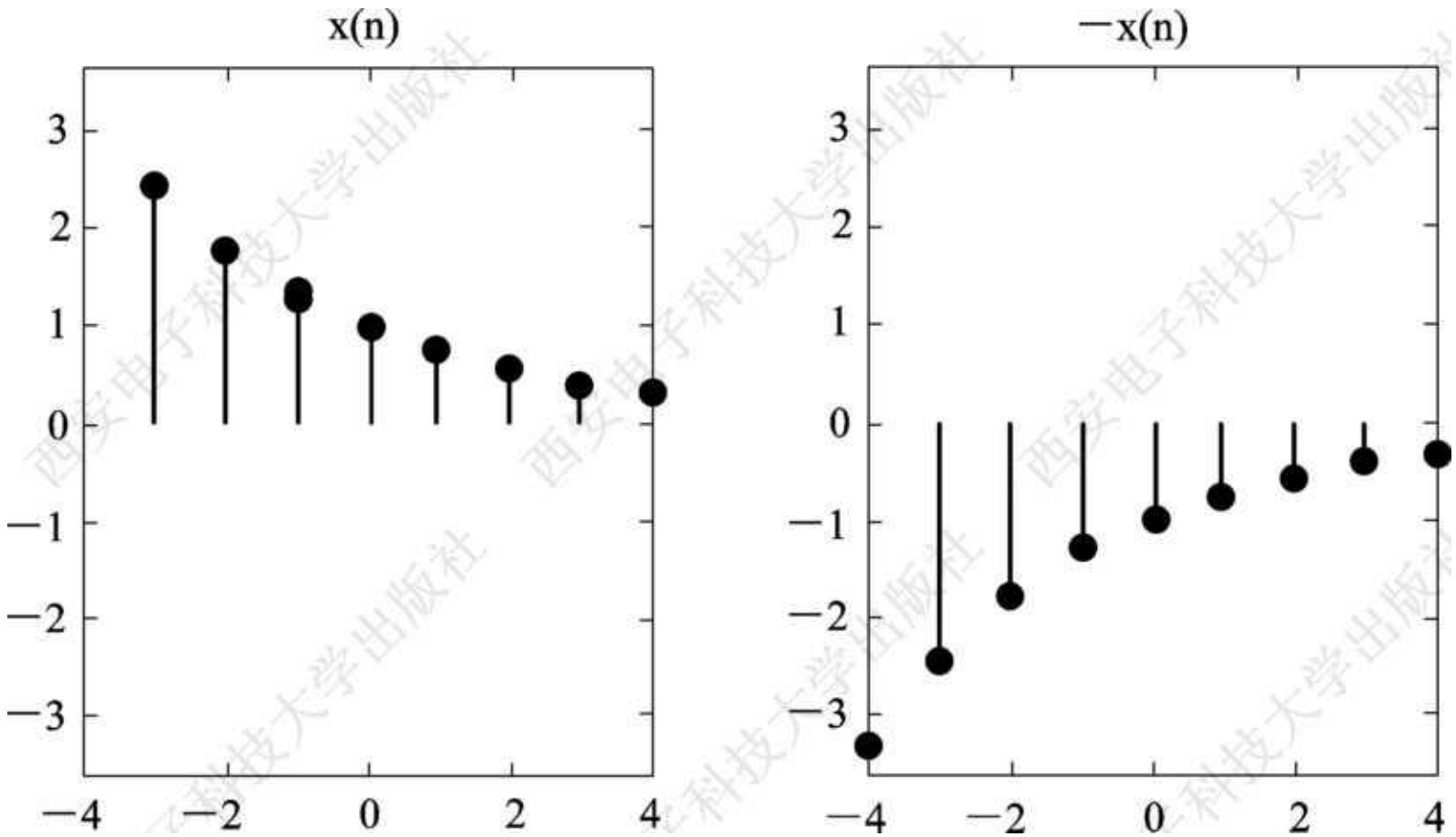

图3-7 序列x(n)和倒相序列－x(n)

6.序列的尺度变换

对于给定的离散序列x(n)，序列x(mn)是x(n)每隔m点取一点形成，相当于时间轴n压缩了m倍；反之，序列x(n/m)是x(n)作m倍的插值而形成的，相当于时间轴n扩展了m倍。

例3-8 已知信号x(n)＝sin(2πn)，求x(2n)和x(n/2)的信号波形。为研究问题的方便，取0<n<20，并将n缩小20倍进行波形显示。

解 MATLAB程序如下：

```
n＝(0：20)/20;
```

```
x＝sin(2*pi*n)；      %建立原信号x(n)
x1＝sin(2*pi*n*2)；%建立x(2n)信号
x2＝sin(2*pi*n/2)；%建立x(n/2)信号
subplot(3，1，1)，stem(n，x，′filled′)；
ylabel(′x(n)′)；
subplot(3，1，2)，stem(n，x1，′filled′)；
ylabel(′x(2n)′)；
subplot(3，1，3)，stem(n，x2，′filled′)；
ylabel(′x(n/2)′)；
```

结果如图3-8所示。

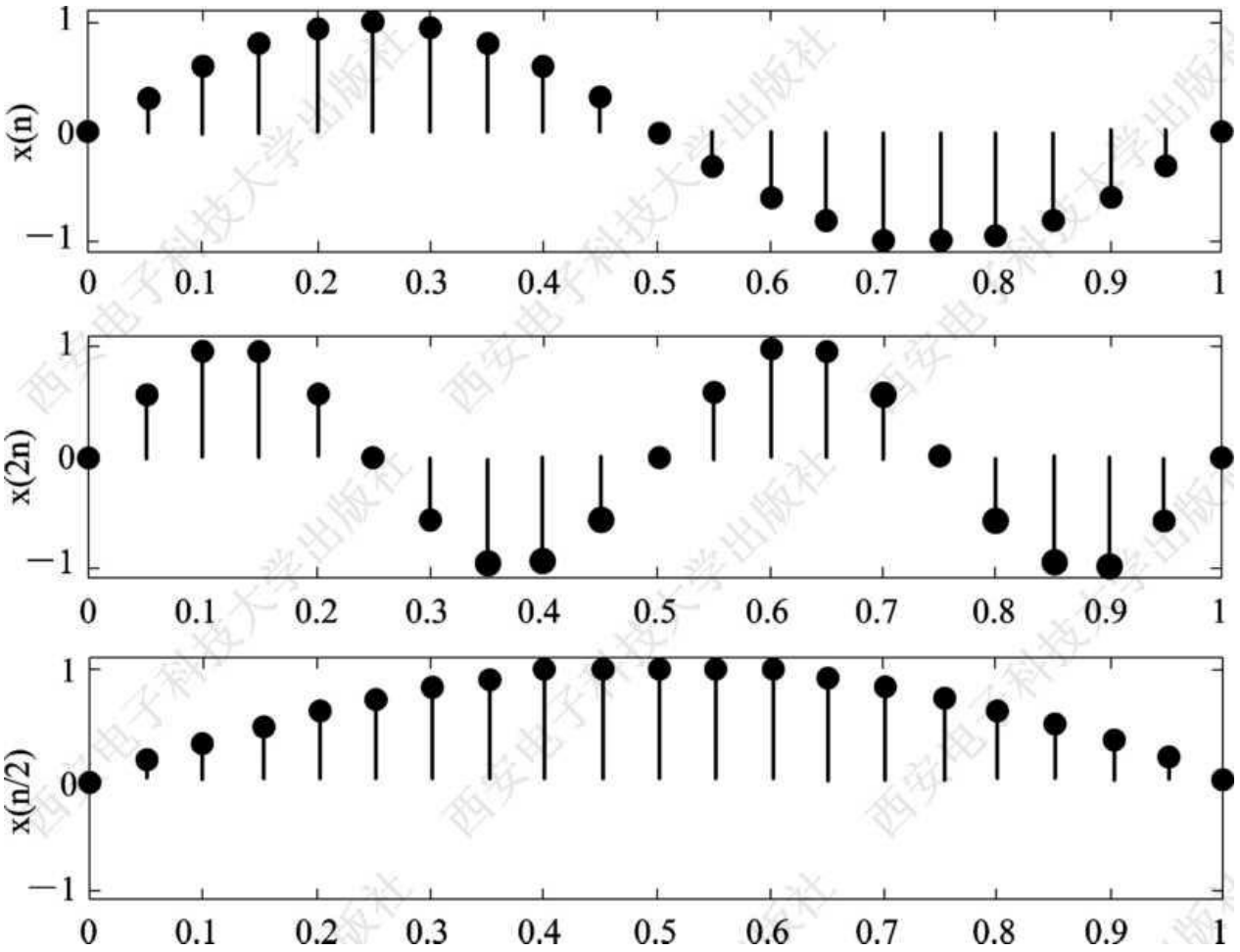

图3-8 序列x(n)、x(2n)和x(n/2)

7.直接使用conv进行卷积运算

求解两个序列的卷积，很重要的问题在于卷积结果的时宽区间如何确定。在MATLAB中，卷积子函数conv默认两个信号的时间序列从n＝0开始，y对应的时间序号也从n＝0开始。

例3-9 已知两个信号序列：

$$f_1 = 0.8n \ (0<n<20)$$

$$f_2 = u(n) \ (0<n<10)$$

求两个序列的卷积和。

编写MATLAB程序如下：

```
nf1＝0：20；                %建立f1的时间向量
f1＝0.8.^nf1；%建立f1信号
subplot(2，2，1)；stem(nf1，f1，′filled′)；
title(′f1(n)′)；
nf2＝0：10；%建立f2的时间向量
lf2＝length(nf2)；%取f2时间向量的长度
f2＝ones(1，lf2)；%建立f2信号
subplot(2，2，2)；stem(nf2，f2，′filled′)；
title(′f2(n)′)；
y＝conv(f1，f2)；%卷积运算
```

subplot(2，1，2)；stem(y，′filled′)；

title(′y(n)′)；

结果如图3-9所示。

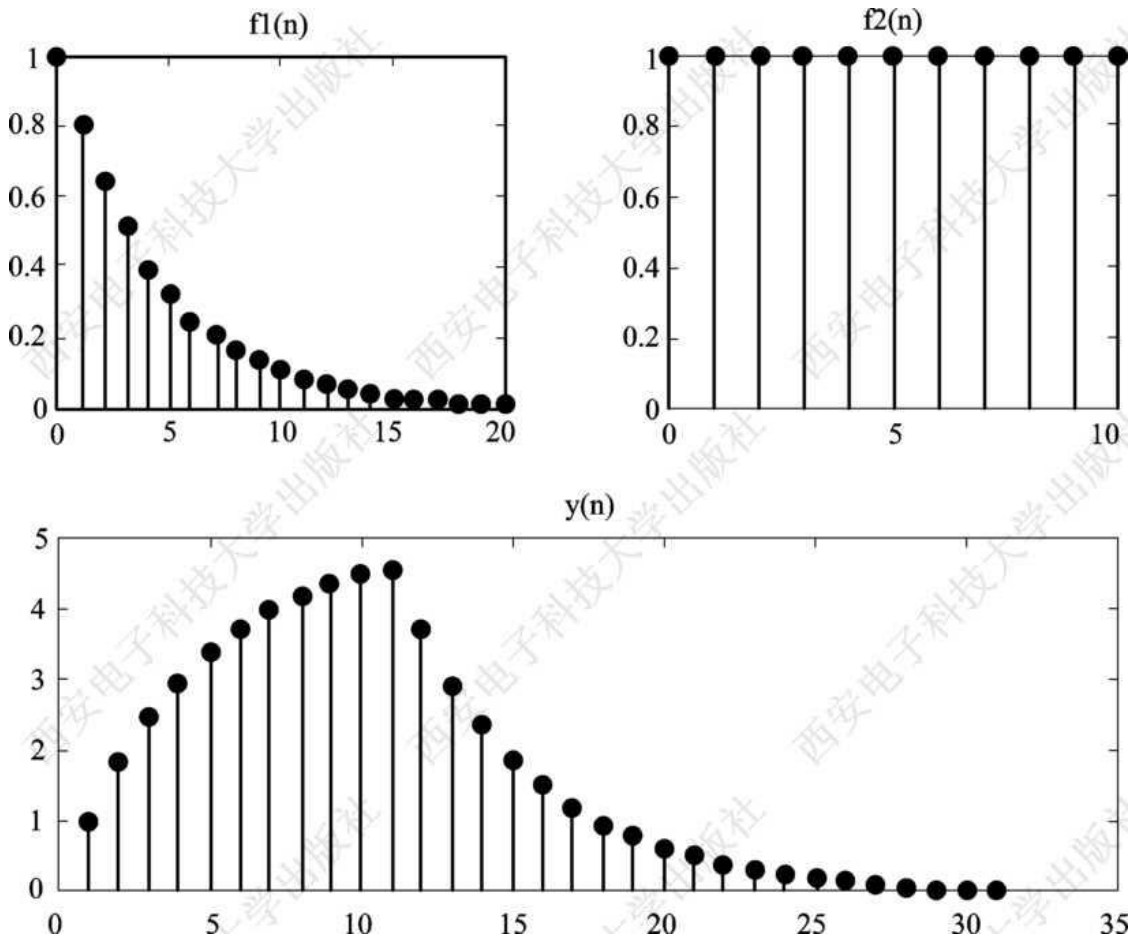

图3-9 例3-9f1(n)、f2(n)、y(n)的波形

8.复杂序列的卷积运算

由于MATLAB中卷积子函数conv默认两个信号的时间序列从n＝0开始，因此，如果信号不是从0开始，则编程时必须用两个数组确定一个信号，其中，一个数组是信号波形的幅度样值，另一个数组是其对应的时间向量。此时，程序的编写较为复杂，我们可以将其处理过程编写成一个可调用的通用子函数。

下面是在conv基础上进一步编写的新的卷积子函数convnew，是一个适用于信号从任意时间开始的通用程序。

function［y，ny］＝convnew(x，nx，h，nh) %建立convnew子函数

%x为一信号幅度样值向量，nx为x对应的时间向量

%h为另一信号或系统冲激函数的非零样值向量，nh为h对应的时间向量

%y为卷积积分的非零样值向量，ny为其对应的时间向量

n1＝nx(1)＋nh(1)；%计算y的非零样值的起点位置

n2＝nx(length(x))＋nh(length(h))；%计算y的非零样值的宽度

ny＝［n1：n2］；%确定y的非零样值时间向量

y＝conv(x，h)；

用上述程序可以计算两个离散时间序列的卷积和，求解信号通过一个离散系统的响应。

例3-10 两个信号序列：f1为0.5n(0<n<10)的斜变信号序列；f2为一个u(n＋2)(－2<n<10)的阶跃序列，求两个序列的卷积和。

解 从信号序列n的范围可见，f2的时间轴起点不是n＝0，因此，该程序需使用卷积子函数convnew进行计算。

编写MATLAB程序如下：

```
nf1＝0：10；          %f1的时间向量
f1＝0.5*nf1；
nf2＝－2：10；%f2的时间向量
nt＝length(nf2)；%取f2时间向量的长度
f2＝ones(1，nt)；
```

［y，ny］＝convnew(f1，nf1，f2，nf2)；　%调用convnew卷积子函数

subplot(2，2，1)，stem(nf1，f1，′filled′)；%显示f1信号

subplot(2，2，2)，stem(nf2，f2，′filled′)；%显示f2信号

subplot(2，1，2)，stem(ny，y，′filled′)；%卷积积分结果

程序执行的结果如图3-10所示。

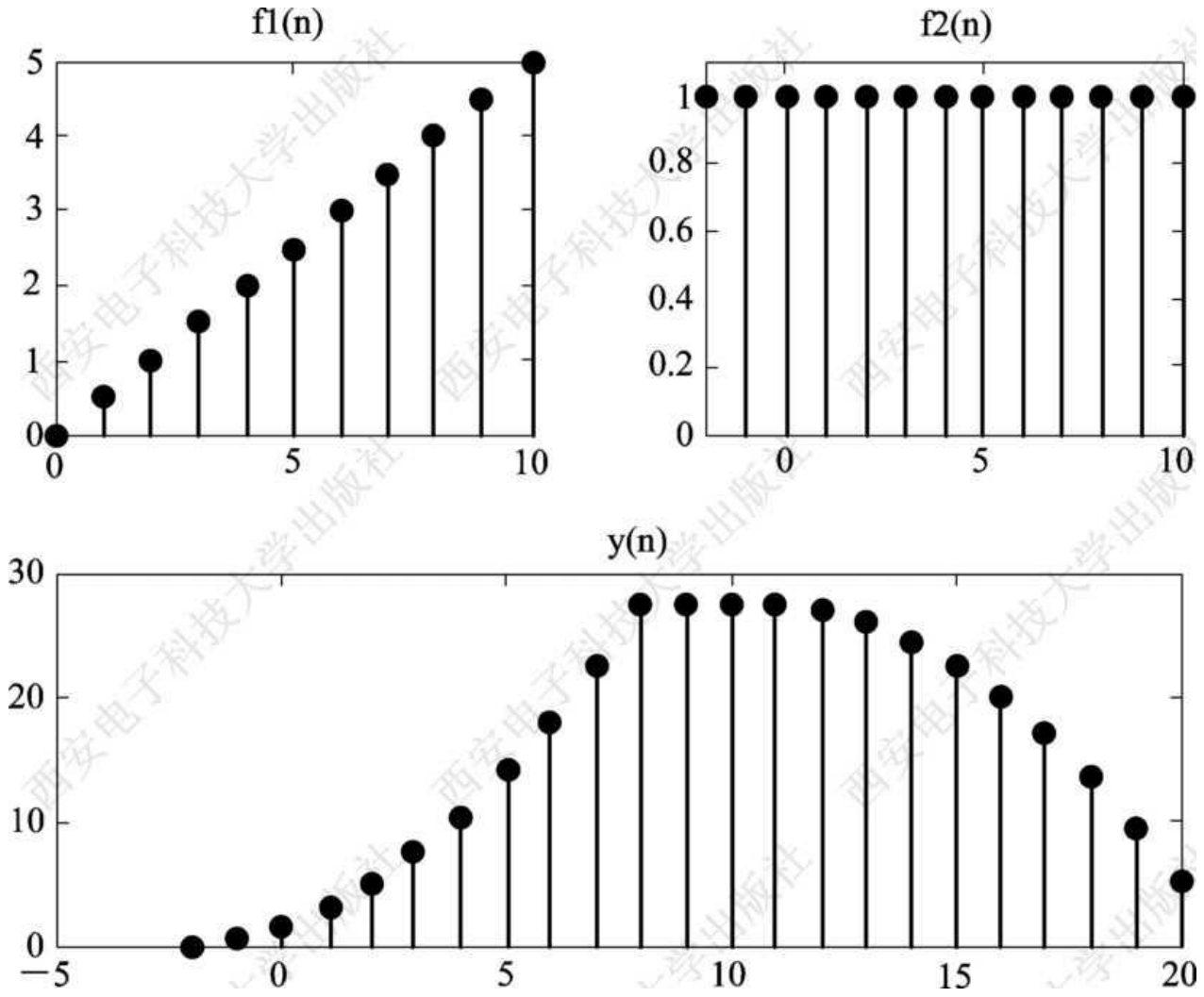

图3-10 例3-10f1(n)、f2(n)、y(n)的波形

9.卷积积分的动态过程演示

为了更深入地理解两个序列卷积的原理，下面提供一段演示卷积积分的动态过程的MATLAB程序。

例3-11 动态地演示例3-9求解信号序列

$$f_1 = 0.8n \ (0<n<20)$$

$$f_2 = u(n) \ (0<n<10)$$

卷积和的过程。

编写MATLAB程序如下：

```
clf;                    %图形窗清屏
nf1＝0：20；        %建立f1的时间向量
f1＝0.8.^nf1；      %建立f1序列
lf1＝length(f1)；%取f1时间向量的长度
nf2＝0：10；        %f2的时间向量
lf2＝length(nf2)；%取f2时间向量的长度
f2＝ones(1，lf2)；%建立f2序列
lmax＝max(lf2，lf1)；%求最长的序列
```

```
iflf2>lf1nf2＝0；nf1＝lf2－lf1；%若f2比f1长，对f1补
                                        nf1个0
elseiflf2<lf1nf1＝0；nf2＝lf1－lf2；%若f1比f2长，对
                                            f2补nf2个0
elsenf2＝0；lf1＝0；      %若f1与f2同长，不补0
end
lt＝lmax;                 %取长者为补0长度基础
%先将f2补得与f1同长，再将两边补最大长度的0
u＝［zeros(1，lt)，f2，zeros(1，nf2)，zeros(1，lt)］；
t1＝(－lt＋1：2*lt);
```

```
%先将f1补得与f2同长，再将左边补2倍最大长度的0
f1＝［zeros(1，2*lt)，f1，zeros(1，nf1)］；
hf1＝fliplr(f1);              %将f1作左右反折
N＝length(hf1);
y＝zeros(1，3*lt);            %将y存储单元初始化
fork＝0：2*lt                 %动态演示绘图
p＝［zeros(1，k)，hf1(1：N－k)］；%使hf1向右循环
                                  移位
y1＝u.*p；[KG－4]             %使输入和翻转移位的脉冲过
                                  渡函数逐项相乘
yk＝sum(y1);                  %相加
```

```
y(k＋lt＋1)＝yk；%将结果放入数组y
subplot(4，1，1)；stem(t1，u)；
subplot(4，1，2)；stem(t1，p)；
subplot(4，1，3)；stem(t1，y1)；
subplot(4，1，4)；stem(k，yk)；%作图表示每一次卷
                                积的结果
axis(［－20，50，0，5］)；holdon[KG－1] %在图形
                    窗上保留每一次运行的图形结果
pause(1)；          %停顿1秒钟
end
```