Systems & Signals 414

Practical 3

D.S. van der Westhuizen - 17158680

In [1]:

```
%matplotlib inline
import pylab as pl
pl.style.use('bmh') #pretty plots
pl.rcParams['figure.figsize'] = (9, 2)
import numpy as np
import copy

def zpad(signal, pad_left_right):
    return np.pad(signal, pad_left_right,
                  mode='constant', constant_values=0)

def prepare_plot(title, y_label, x_label):
    pl.figure()
    pl.title(title)
    pl.ylabel(y_label)
    pl.xlabel(x_label)
```

In [2]:

```
#import IPython.display; IPython.display.display(IPython.display.SVG('radar_image.sv
g'))
```

In [3]:

```
x = [1,1,1,1,1,-1,-1,1,1,-1,1,-1,1]
w = np.random.normal(0,0.1,200)
```

3.1

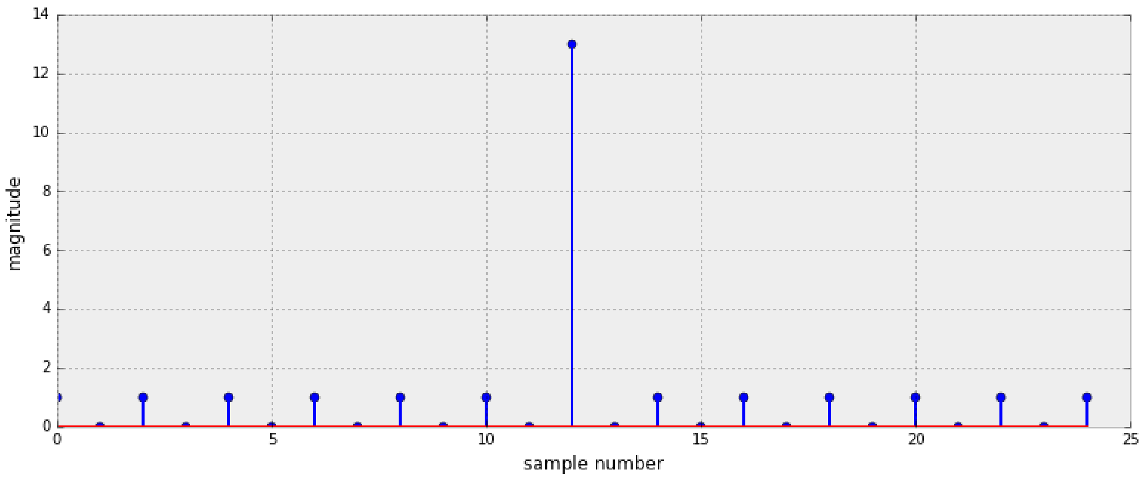
In [4]:

```
rxx = np.correlate(x,x,mode='full')
print(rxx)

t_34535 = np.linspace(0,2*(len(x)-1)+1,2*(len(x)-1)+1,False)
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_34535,rxx)
```

```
[ 1  0  1  0  1  0  1  0  1  0  1  0 13  0  1  0  1  0  1  0  1  0  1  0
 1]
```

Out[4]:

```
<Container object of 3 artists>
```



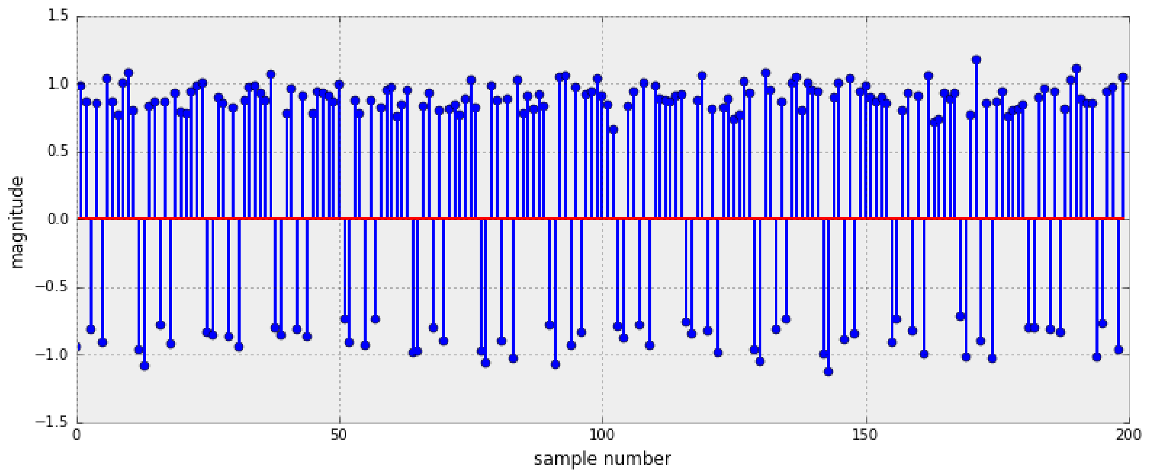
3.2

In [5]:

```
a = 0.9
D = 20

t = np.linspace(0,200,200,False)
x_cont = np.linspace(0,200,200,False)

y = np.linspace(0,200,200,False)
for s in range(0,200):
    i = copy.copy(s)
    while(i < 0):
        i += 13
    while(i > 12):
        i -= 13
    o = copy.copy(s) - copy.copy(D)
    while(o < 0):
          o += 13
    while(o > 12):
        o -= 13
    x_cont[s] = x[i]
    y[s] = a*x[o] + w[s]
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t,y)
```

Out[5]:

```
<Container object of 3 artists>
```



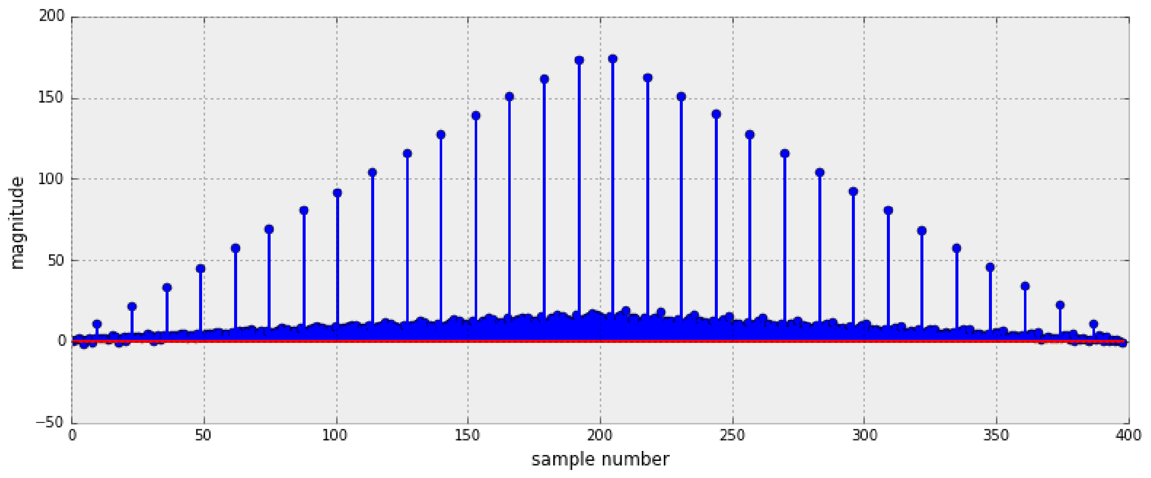
3.3

In [6]:

```
t_5464 = np.linspace(0,399,399,False)
xy_cor = np.correlate(x_cont,y,mode='full')

pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_5464,xy_cor)
```

Out[6]:

```
<Container object of 3 artists>
```

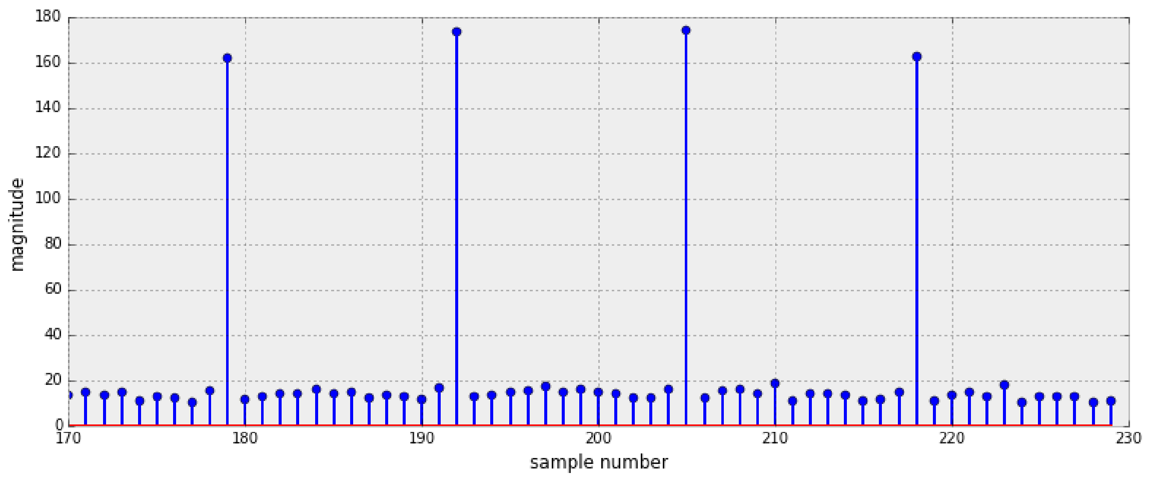


In [7]:

```
t_5464 = np.linspace(0,399,399,False)
xy_cor = np.correlate(x_cont,y,mode='full')

pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_5464[170:230],xy_cor[170:230])
```

Out[7]:

```
<Container object of 3 artists>
```



Since the mean sample is 200, D = 20, T = 13, a peak occurs at 200 + T - D - 1 = 192 Thus, using this method for determining D is only effective if we know that D<T

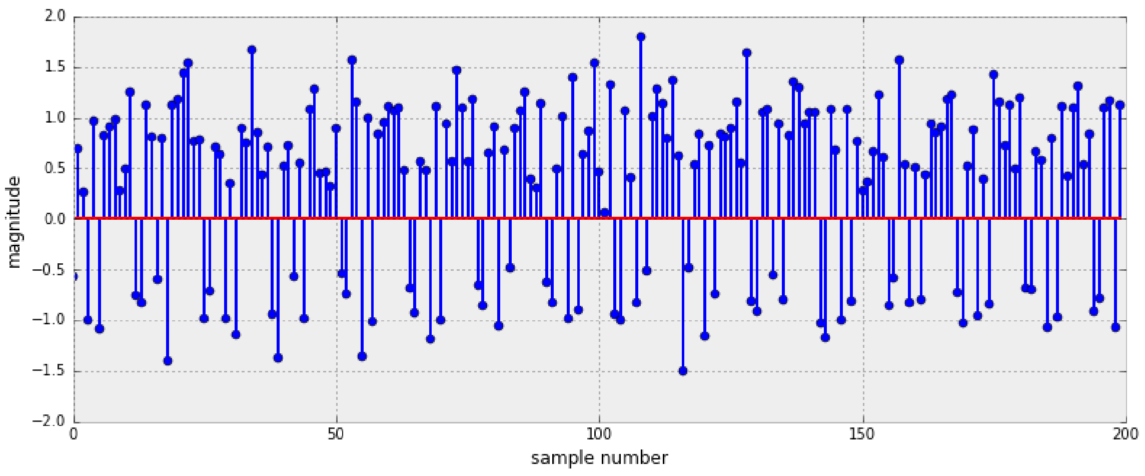
3.4

In [8]:

```
w01 = np.random.normal(0,0.33,200)

y34 = np.linspace(0,200,200,False)
for s in range(0,200):
    i = copy.copy(s)
    while(i < 0):
        i += 13
    while(i > 12):
        i -= 13
    o = copy.copy(s) - copy.copy(D)
    while(o < 0):
          o += 13
    while(o > 12):
        o -= 13
    y34[s] = a*x[o] + w01[s]
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t,y34)
```

Out[8]:

```
<Container object of 3 artists>
```

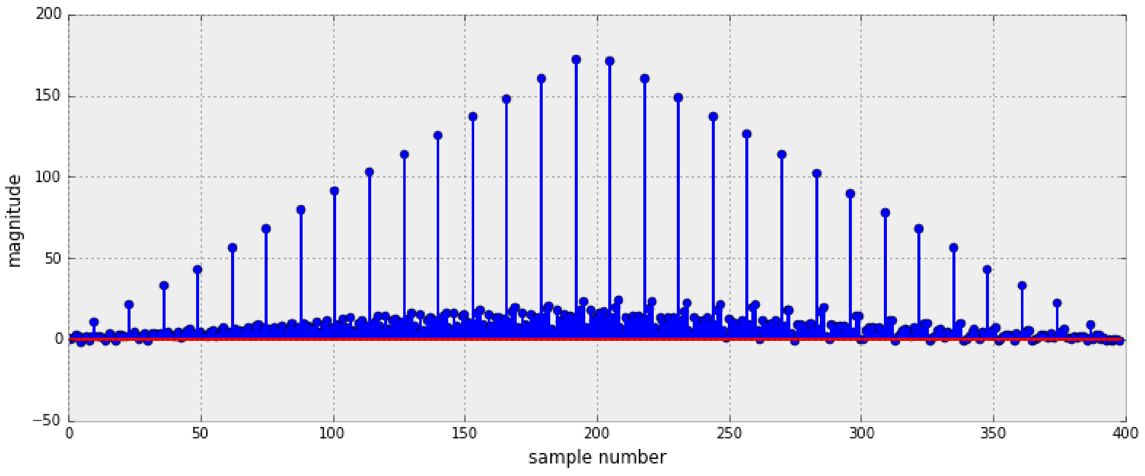


In [9]:

```
t_399 = np.linspace(0,399,399,False)
xy_cor34 = np.correlate(x_cont,y34,mode='full')

pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_399,xy_cor34)
```

Out[9]:

```
<Container object of 3 artists>
```

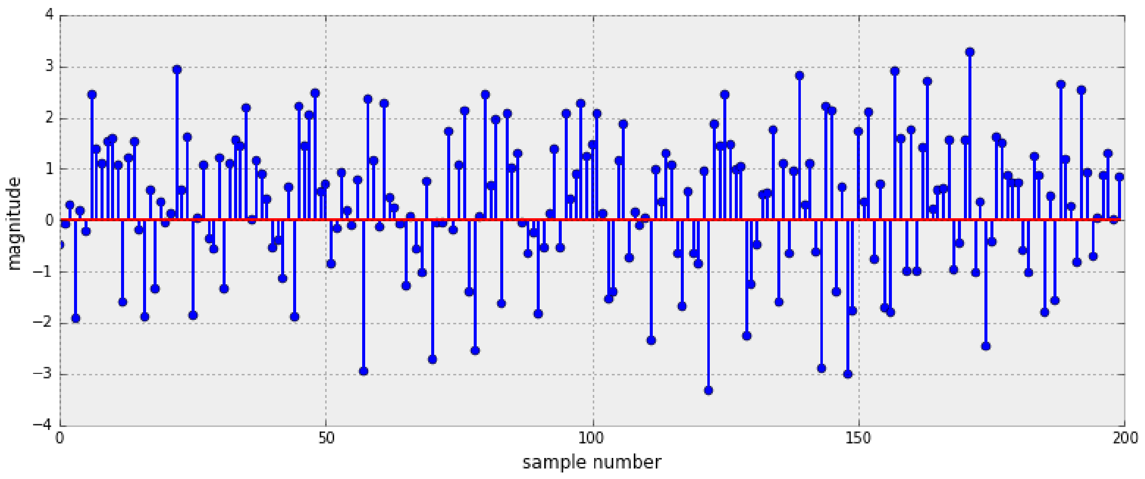


In [10]:

```
w1 = np.random.normal(0,1,200)

y34_2 = np.linspace(0,200,200,False)
for s in range(0,200):
    i = copy.copy(s)
    while(i < 0):
        i += 13
    while(i > 12):
        i -= 13
    o = copy.copy(s) - copy.copy(D)
    while(o < 0):
          o += 13
    while(o > 12):
        o -= 13
    y34_2[s] = a*x[o] + w1[s]
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t,y34_2)
```

Out[10]:

```
<Container object of 3 artists>
```

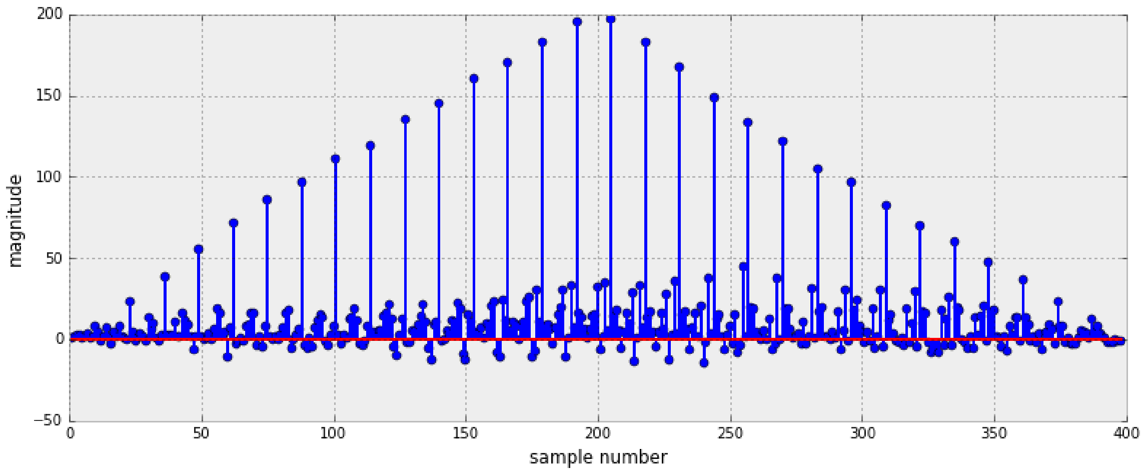


In [11]:

```
t_399 = np.linspace(0,399,399,False)
xy_cor34_2 = np.correlate(x_cont,y34_2,mode='full')

pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_399,xy_cor34_2)
```

Out[11]:

```
<Container object of 3 artists>
```



4.1

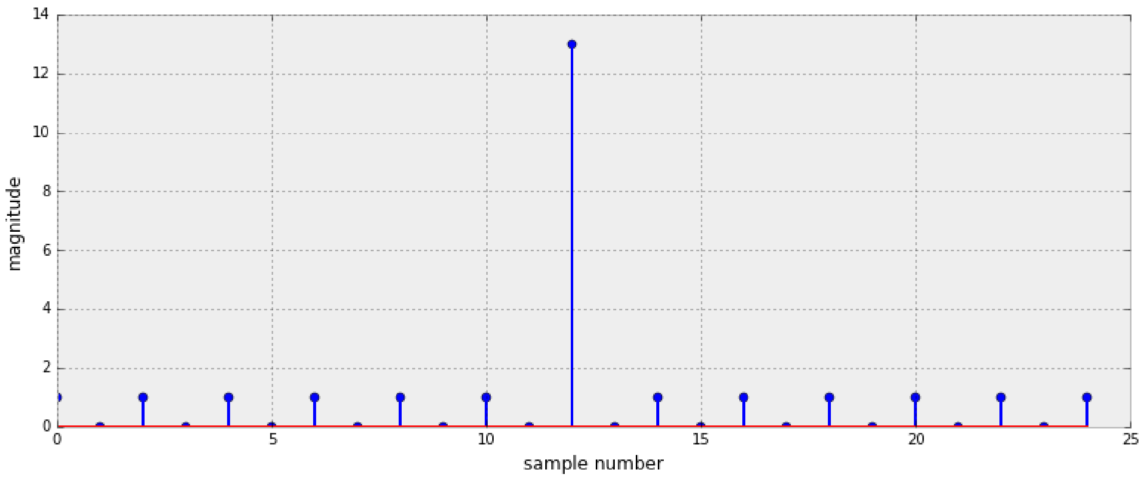
In [12]:

```
x4 = [1,1,1,1,0,0,0,0,0,1,1,1,1]
rxx4 = np.correlate(x4,x4,mode='full')
print(rxx4)

t_41 = np.linspace(0,2*(len(x4)-1)+1,2*(len(x4)-1)+1,False)
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_41,rxx)
```

```
[1 2 3 4 3 2 1 0 0 2 4 6 8 6 4 2 0 0 1 2 3 4 3 2 1]
```

Out[12]:

```
<Container object of 3 artists>
```



This is not suitable because none of the values rise well above the others.

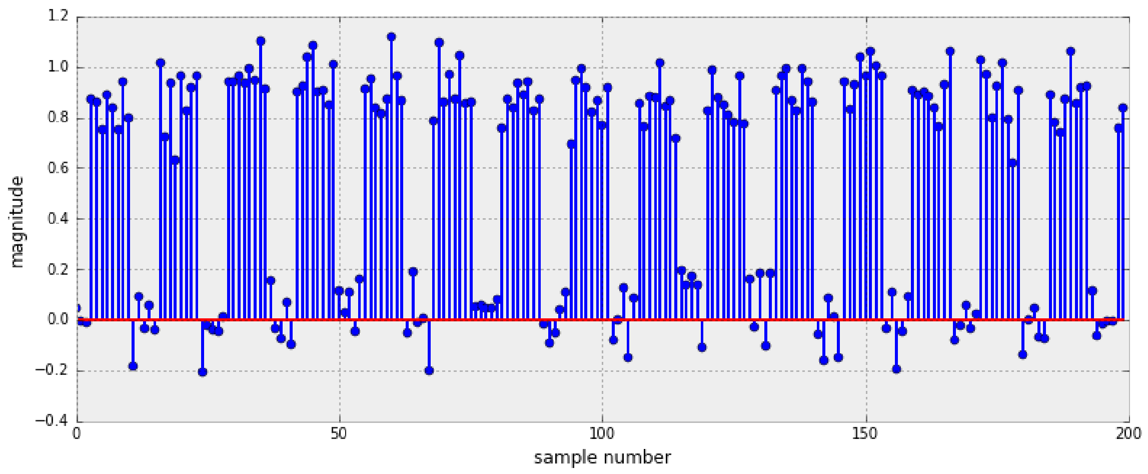
4.2

In [13]:

```
w01 = np.random.normal(0,0.1,200)

a = 0.9
D = 20

t421 = np.linspace(0,200,200,False)
x4_cont = np.linspace(0,200,200,False)

y42 = np.linspace(0,200,200,False)
for s in range(0,200):
    i = copy.copy(s)
    while(i < 0):
        i += 13
    while(i > 12):
        i -= 13
    o = copy.copy(s) - copy.copy(D)
    while(o < 0):
          o += 13
    while(o > 12):
        o -= 13
    x4_cont[s] = x4[i]
    y42[s] = a*x4[o] + w01[s]
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t421,y42)
```

Out[13]:

```
<Container object of 3 artists>
```

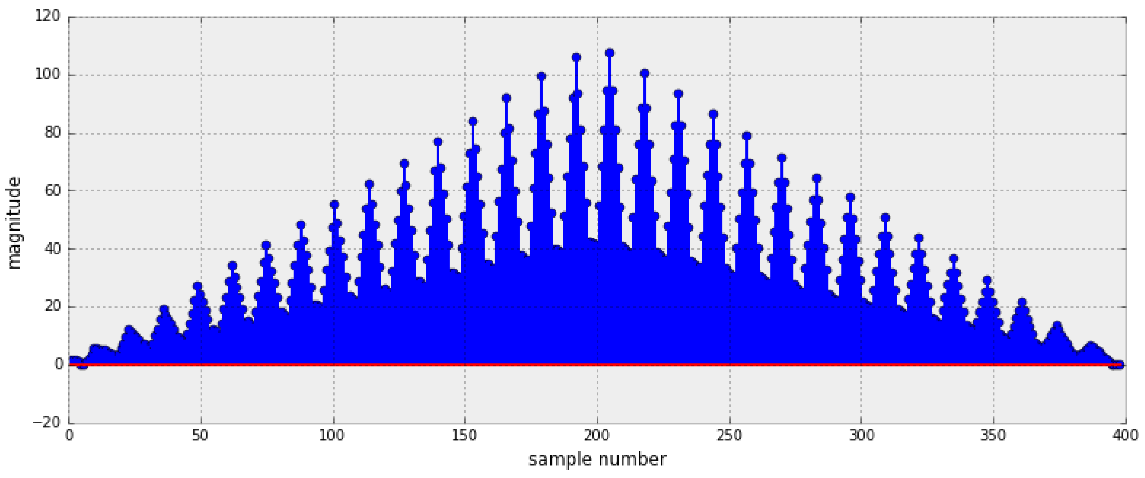


In [14]:

```
t_422 = np.linspace(0,399,399,False)
xy4_cor = np.correlate(x4_cont,y42,mode='full')

pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_422,xy4_cor)
```

Out[14]:

```
<Container object of 3 artists>
```



Yes, D is still identifiable by the peak at n=192

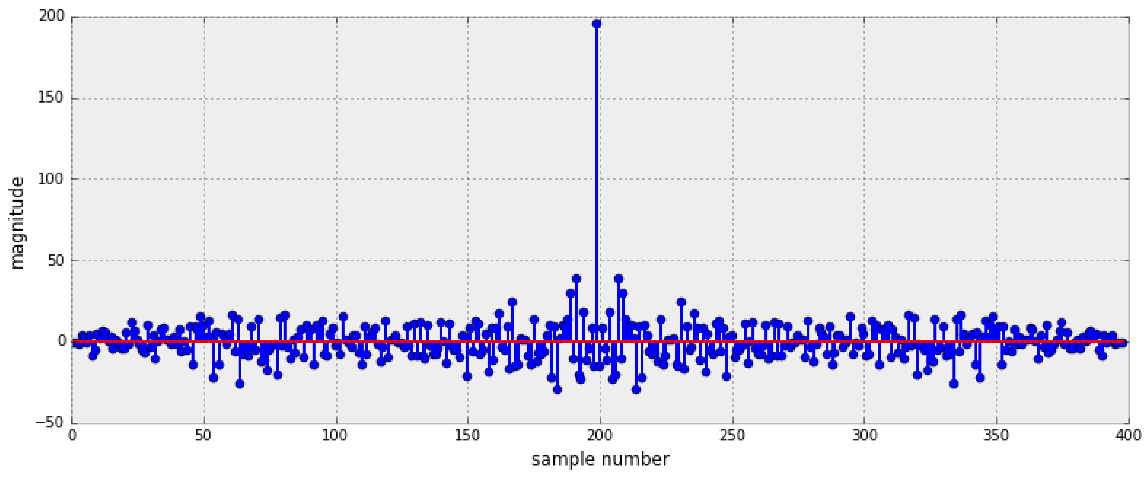
5.1

In [15]:

```
x5 = np.random.normal(0,1,200)
rxx5 = np.correlate(x5,x5,mode='full')

t_5 = np.linspace(0,len(rxx5),len(rxx5),False)
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_5,rxx5)
```

Out[15]:

```
<Container object of 3 artists>
```



The autocovariance of an x consisting of white noise has a discernable peak and thus will work as long as x is not subjected to an equal or greater white noise

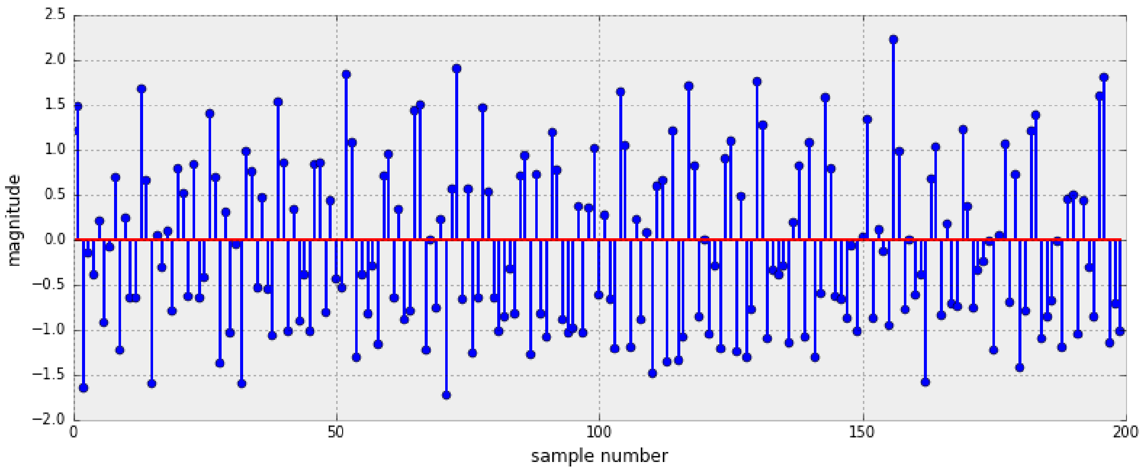
5.2

In [16]:

```
w01 = np.random.normal(0,0.33,200)

a = 0.9
D = 20

t521 = np.linspace(0,200,200,False)
x5_cont = np.linspace(0,200,200,False)

y52 = np.linspace(0,200,200,False)
for s in range(0,200):
    i = copy.copy(s)
    while(i < 0):
        i += 13
    while(i > 12):
        i -= 13
    o = copy.copy(s) - copy.copy(D)
    while(o < 0):
          o += 13
    while(o > 12):
        o -= 13
    x5_cont[s] = x5[i]
    y52[s] = a*x5[o] + w01[s]
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t521,y52)
```

Out[16]:

```
<Container object of 3 artists>
```

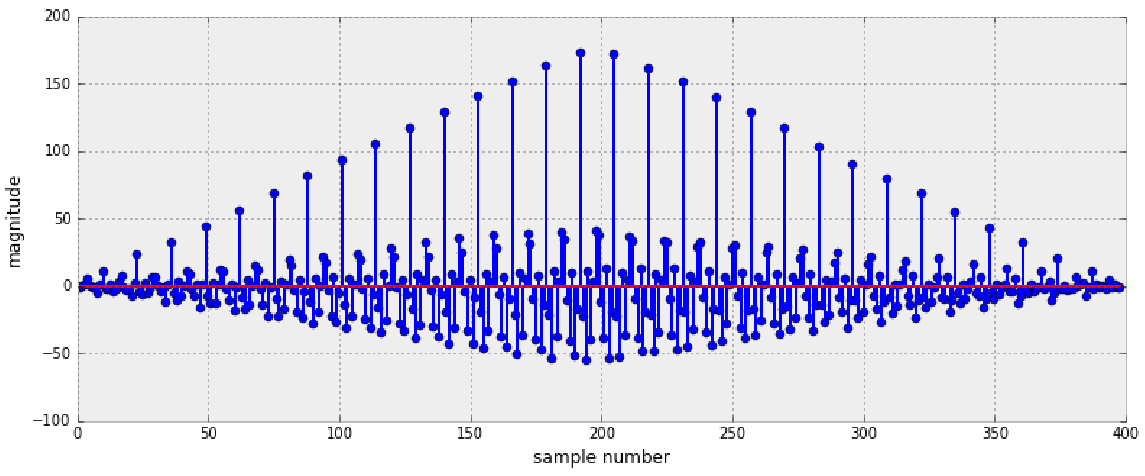


In [17]:

```
t_522 = np.linspace(0,399,399,False)
xy5_cor = np.correlate(x5_cont,y52,mode='full')

pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_522,xy5_cor)
```

Out[17]:

```
<Container object of 3 artists>
```



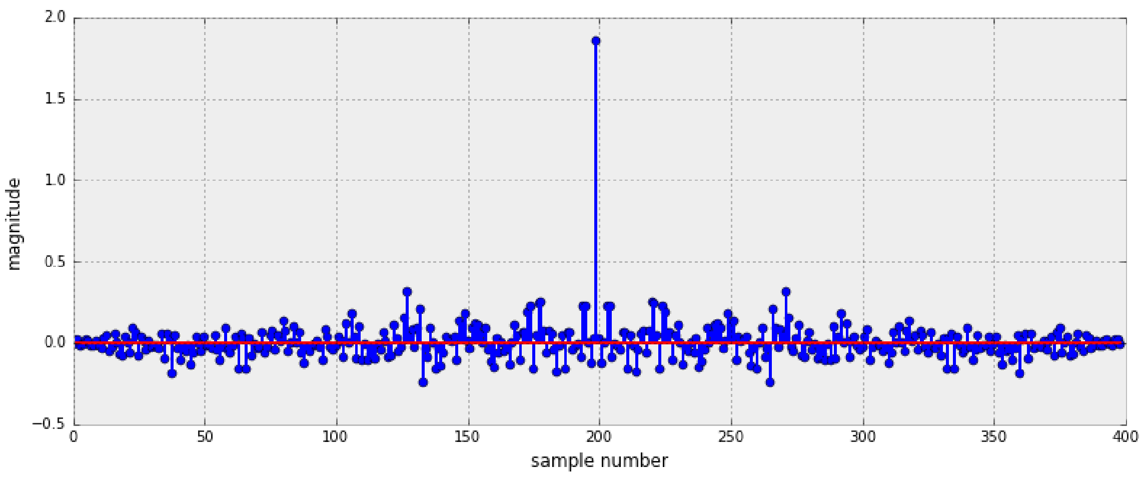
6

In [18]:

```
x6 = np.random.normal(0,0.1,200)
rxx6 = np.correlate(x6,x6,mode='full')

t_61 = np.linspace(0,len(rxx6),len(rxx6),False)
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_61,rxx6)
```

Out[18]:

```
<Container object of 3 artists>
```



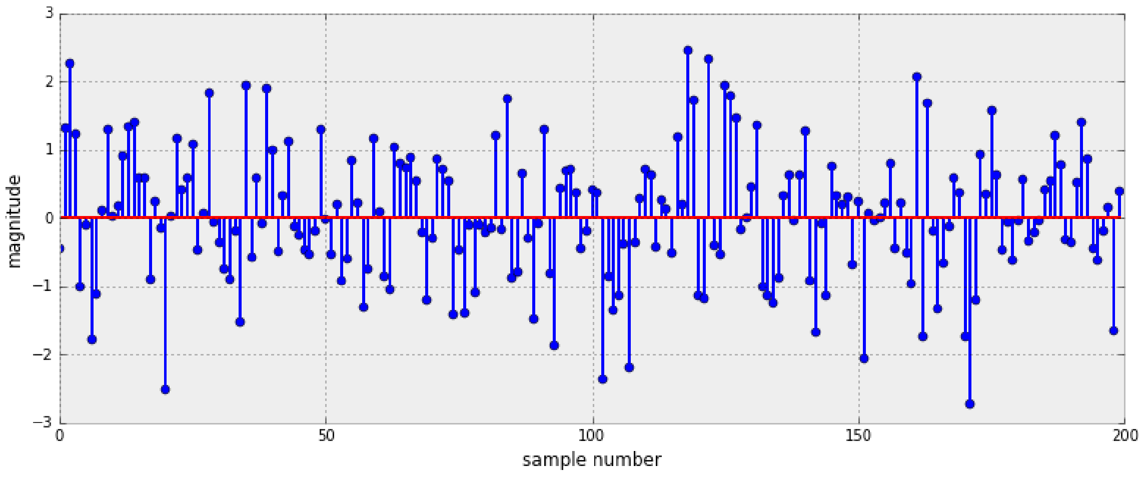
The autocovariance of an x consisting of white noise has a discernable peak and thus will work unless x is subjected to an equal or greater white noise, as it is here.

In [19]:

```
w1 = np.random.normal(0,1,200)

a = 0.9
D = 20

t621 = np.linspace(0,200,200,False)
x6_cont = np.linspace(0,200,200,False)

y62 = np.linspace(0,200,200,False)
for s in range(0,200):
    i = copy.copy(s)
    while(i < 0):
        i += 13
    while(i > 12):
        i -= 13
    o = copy.copy(s) - copy.copy(D)
    while(o < 0):
          o += 13
    while(o > 12):
        o -= 13
    x6_cont[s] = x6[i]
    y62[s] = a*x6[o] + w1[s]
pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t621,y62)
```

Out[19]:

```
<Container object of 3 artists>
```

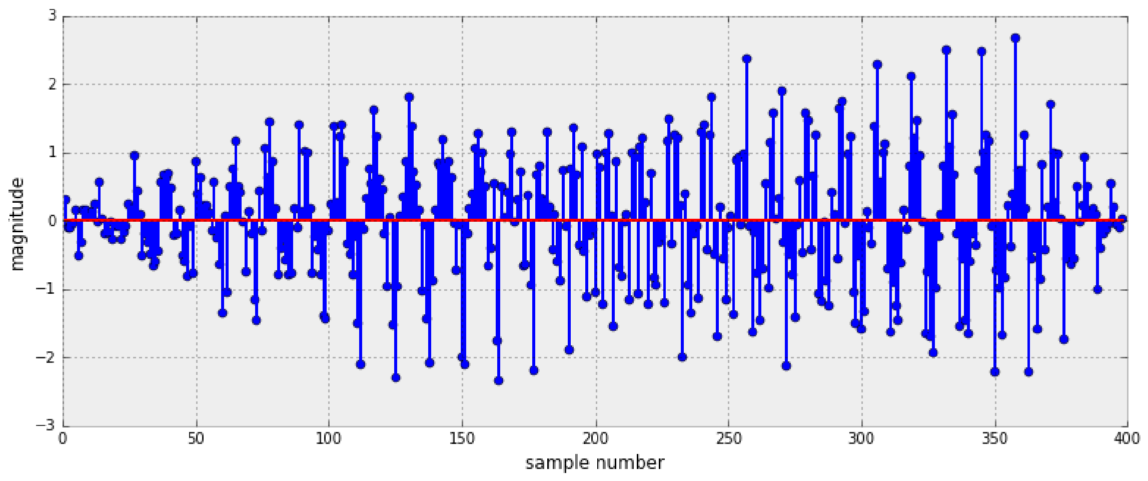


In [20]:

```
t_622 = np.linspace(0,399,399,False)
xy6_cor = np.correlate(x6_cont,y62,mode='full')

pl.figure(figsize=(13,5))
pl.ylabel("magnitude")
pl.xlabel("sample number")
pl.stem(t_622,xy6_cor)
```

Out[20]:

```
<Container object of 3 artists>
```



D is not discernable anymore.

The Barker sequence work better.