TFY4235/FYS8904

Solution problemset 10 Spring 2015

NTNU

Institutt for fysikk

Problem 1.

I generate here a random walker consisting of 2^{15} steps. This I wavelet transform. I find the maximum absolute value of the wavelet coefficients, `amax`.
Filtering of data consists in setting wavelet coefficients whose absolute value is less than `filt*amax` where `filt` is between zero and one, to zero. I then transform the signal back into real space and compare the filtered and unfiltered signals. Here is the program:

Listing 1: wavcom.f

```
      program wavcom
      parameter(n=32768,filt=0.001)
      dimension a(n),b(n)
      ibm=5251
      do i=1,1000
      ibm=ibm*16807
      enddo
      rinv=0.5/(2.**31-1.)
      a(1)=0.
      b(1)=0.
      do i=2,n
      ibm=ibm*16807
      a(i)=a(i-1)+1.-2.*int(1.+ibm*rinv)
      b(i)=a(i)
      enddo
      isign=1
      call wt1(a,n,isign)
      amax=-1.e+8
      do i=1,n
      amax=max(amax,abs(a(i)))
      enddo
      adel=amax*filt
      ic=0
      do i=1,n
      if(abs(a(i)).lt.adel) then
      a(i)=0.
      ic=ic+1
      endif
      enddo
      isign=-1
      call wt1(a,n,isign)
      open(unit=1,file='wavcom.dat',status='unknown')
      do i=1,n
      write(1,*) i,b(i),a(i)
```

```
35        enddo
36        close(1)
37        write(*,*) ic
38        end
39  c-------------------------------------------
40        SUBROUTINE wt1(a,n,isign)
41        INTEGER isign,n
42        REAL a(n)
43        EXTERNAL wtstep
44  CU    USES wtstep
45        INTEGER nn
46        if (n.lt.4) return
47        if (isign.ge.0) then
48          nn=n
49  1       if (nn.ge.4) then
50            call daub4(a,nn,isign)
51            nn=nn/2
52          goto 1
53          endif
54        else
55          nn=4
56  2       if (nn.le.n) then
57            call daub4(a,nn,isign)
58            nn=nn*2
59          goto 2
60          endif
61        endif
62        return
63        END
64  C  (C) Copr. 1986-92 Numerical Recipes Software u1jw3+&9p++.
65  c-------------------------------------------
66        SUBROUTINE daub4(a,n,isign)
67        INTEGER n,isign,NMAX
68        REAL a(n),C3,C2,C1,C0
69        PARAMETER (C0=0.4829629131445341,C1=0.8365163037378079,
70       *C2=0.2241438680420134,C3=-0.1294095225512604,NMAX=32768)
71        REAL wksp(NMAX)
72        INTEGER nh,nh1,i,j
73        if(n.lt.4)return
74        if(n.gt.NMAX) pause 'wksp too small in daub4'
75        nh=n/2
76        nh1=nh+1
77        if (isign.ge.0) then
78          i=1
79          do 11 j=1,n-3,2
80            wksp(i)=C0*a(j)+C1*a(j+1)+C2*a(j+2)+C3*a(j+3)
81            wksp(i+nh)=C3*a(j)-C2*a(j+1)+C1*a(j+2)-C0*a(j+3)
82            i=i+1
83  11        continue
84          wksp(i)=C0*a(n-1)+C1*a(n)+C2*a(1)+C3*a(2)
85          wksp(i+nh)=C3*a(n-1)-C2*a(n)+C1*a(1)-C0*a(2)
86        else
87          wksp(1)=C2*a(nh)+C1*a(n)+C0*a(1)+C3*a(nh1)
```

```
88          wksp(2)=C3*a(nh)-C0*a(n)+C1*a(1)-C2*a(nh1)
89          j=3
90          do 12 i=1,nh-1
91            wksp(j)=C2*a(i)+C1*a(i+nh)+C0*a(i+1)+C3*a(i+nh1)
92            wksp(j+1)=C3*a(i)-C0*a(i+nh)+C1*a(i+1)-C2*a(i+nh1)
93            j=j+2
94  12        continue
95        endif
96        do 13 i=1,n
97          a(i)=wksp(i)
98  13    continue
99        return
100       END
101 C  (C) Copr. 1986-92 Numerical Recipes Software u1jw3+&9p++.
102 c
```

I have used the wavelet routine from *Numerical Recipes.*
The result is shown in figure 1.

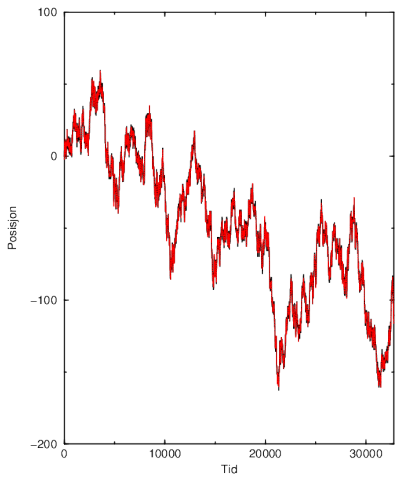

Figure 1: Compressed and non-compressed random walk.

After removing all wavelet coefficients with absolute value less than 0.001 of `amax` I get the figure above. The (black) unbroken curve is the original signal and the (red) broken curve is the signal after filtering. Only 3% of the original wavelet coefficients have survived the filtering. This gives a compression ratio of 97%!

Problem 2.

The key to using wavelets to generate random walks lays in the scaling properties of the wavelet coefficients. If a curve $x(t)$ with statistical properties $\langle x\rangle = 0$ and $\langle x^2\rangle^{1/2} \propto \sqrt{t}$, the wavelet coefficients will scale as

$$A_{a,\lambda b} = \lambda A_{a,b} \tag{1}$$

where a is position and b is scale — see Simonsen et al. Phys. Rev. E **58**, 2779–2787 (1998). This means that $A_{a,b} \propto b$. furthermore, there are no correlations between wavelet coefficients with same b but different a. Hence, one generates random wavelet coefficients drawn from a flat distribution centered around zero and width b. We then wavelet transform the signal "backwards" into real space — and the random walk has been generated. Here is the program:

Listing 2: wavspe.f

```
      program wavspe
      parameter(npk=32768,np2=npk/2)
      dimension b(npk),c(2,npk),p(np2)
      ibm=4711
      do i=1,1000
      ibm=ibm*16807
      enddo
      rinv=0.5/(2.**31-1.)
c Genererer virrevandringen
      k=1
      amp=1.
1600  continue
      if(2**k.eq.npk) goto 1700
      amp=amp*0.5
      do i=2**k+1,2**(k+1)
      ibm=ibm*16807
      ran=ibm*rinv
      b(i)=ibm*rinv*amp
      enddo
      k=k+1
      goto 1600
1700  continue
      call wt1(b,npk,-1)
      open(unit=1,file='wavsp1.dat',status='unknown')
      do i=1,npk
      write(1,*) i,b(i)
      c(1,i)=b(i)
      c(2,i)=0.
      enddo
      close(1)
c Analyserer virrevandringen
      call four1(c,npk,1)
      do i=1,np2
      p(i)=0.5*(abs(c(1,i)*c(2,i))+abs(c(1,npk+1-i)*c(2,npk+1-i)))
      enddo
      open(unit=2,file='wavsp2.dat',status='unknown')
```

```
      do i=1,np2
      write(2,*) i,p(i)
      enddo
      close(2)
      end
      SUBROUTINE wt1(a,n,isign)
      INTEGER isign,n
      REAL a(n)
      EXTERNAL wtstep
CU    USES wtstep
      INTEGER nn
      if (n.lt.4) return
      if (isign.ge.0) then
        nn=n
1       if (nn.ge.4) then
          call daub4(a,nn,isign)
          nn=nn/2
        goto 1
        endif
      else
        nn=4
2       if (nn.le.n) then
          call daub4(a,nn,isign)
          nn=nn*2
        goto 2
        endif
      endif
      return
      END
      SUBROUTINE daub4(a,n,isign)
      INTEGER n,isign,NMAX
      REAL a(n),C3,C2,C1,C0
      PARAMETER (C0=0.4829629131445341,C1=0.8365163037378079,
     *C2=0.2241438680420134,C3=-0.1294095225512604,NMAX=32768)
      REAL wksp(NMAX)
      INTEGER nh,nh1,i,j
      if(n.lt.4)return
      if(n.gt.NMAX) pause 'wksp too small in daub4'
      nh=n/2
      nh1=nh+1
      if (isign.ge.0) then
        i=1
        do 11 j=1,n-3,2
          wksp(i)=C0*a(j)+C1*a(j+1)+C2*a(j+2)+C3*a(j+3)
          wksp(i+nh)=C3*a(j)-C2*a(j+1)+C1*a(j+2)-C0*a(j+3)
          i=i+1
11      continue
        wksp(i)=C0*a(n-1)+C1*a(n)+C2*a(1)+C3*a(2)
        wksp(i+nh)=C3*a(n-1)-C2*a(n)+C1*a(1)-C0*a(2)
      else
        wksp(1)=C2*a(nh)+C1*a(n)+C0*a(1)+C3*a(nh1)
        wksp(2)=C3*a(nh)-C0*a(n)+C1*a(1)-C2*a(nh1)
        j=3
```

```
90          do 12 i=1,nh-1
91            wksp(j)=C2*a(i)+C1*a(i+nh)+C0*a(i+1)+C3*a(i+nh1)
92            wksp(j+1)=C3*a(i)-C0*a(i+nh)+C1*a(i+1)-C2*a(i+nh1)
93            j=j+2
94  12        continue
95        endif
96        do 13 i=1,n
97          a(i)=wksp(i)
98  13    continue
99        return
100       END
101       SUBROUTINE four1(data,nn,isign)
102       INTEGER isign,nn
103       REAL data(2*nn)
104       INTEGER i,istep,j,m,mmax,n
105       REAL tempi,tempr
106       DOUBLE PRECISION theta,wi,wpi,wpr,wr,wtemp
107       n=2*nn
108       j=1
109       do 11 i=1,n,2
110         if(j.gt.i)then
111           tempr=data(j)
112           tempi=data(j+1)
113           data(j)=data(i)
114           data(j+1)=data(i+1)
115           data(i)=tempr
116           data(i+1)=tempi
117         endif
118         m=n/2
119 1       if ((m.ge.2).and.(j.gt.m)) then
120           j=j-m
121           m=m/2
122         goto 1
123         endif
124         j=j+m
125 11    continue
126       mmax=2
127 2     if (n.gt.mmax) then
128         istep=2*mmax
129         theta=6.28318530717959d0/(isign*mmax)
130         wpr=-2.d0*sin(0.5d0*theta)**2
131         wpi=sin(theta)
132         wr=1.d0
133         wi=0.d0
134         do 13 m=1,mmax,2
135           do 12 i=m,n,istep
136             j=i+mmax
137             tempr=sngl(wr)*data(j)-sngl(wi)*data(j+1)
138             tempi=sngl(wr)*data(j+1)+sngl(wi)*data(j)
139             data(j)=data(i)-tempr
140             data(j+1)=data(i+1)-tempi
141             data(i)=data(i)+tempr
142             data(i+1)=data(i+1)+tempi
```

```
12          continue
            wtemp=wr
            wr=wr*wpr-wi*wpi+wr
            wi=wi*wpr+wtemp*wpi+wi
13        continue
          mmax=istep
        goto 2
        endif
        return
        end
```

The random walk that this program generates is shown in figure 2.

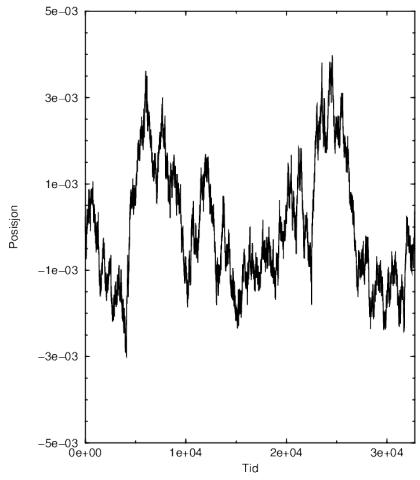

Figure 2: Random walk.

We calculate the power spectrum (in the program above). The power spectrum scales as

$$P(f) \propto \frac{1}{f^2} \,. \tag{2}$$

This is true for all free random walkers (why?). I have plotted the power spectrum from the random walk on log-log scale in figure 3. I have in this figure added a line following Eq. (2).

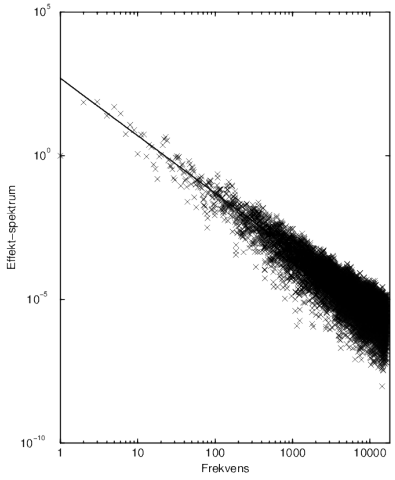

Figure 3: Power spectrum