```
1
2
 3
     1
        Metropolis algorithm, generic illustration
 4
     1
 5
     1
        [Note: One point particle in 3d cube. You will not get "good
 6
        statistics" with one particle, of course. Illustration only!]
7
8
        PROGRAM metropolis
9
10
        IMPLICIT NONE
11
12
        INTEGER, PARAMETER :: n=6, m=1000, l=50, h=100
13
        REAL, PARAMETER :: beta=1.0, pi2=2*3.14159265358979
14
15
        REAL, DIMENSION(h, n) :: x, xx
        REAL :: acpt,s, ekin,ekin1,ekin2,epot,epot1,epot2,C,etot1,etot2
16
17
        INTEGER :: i,ii,j,k,ith
18
        INTEGER, DIMENSION(n) :: seed
19
        REAL, DIMENSION(4) :: r
20
21
        j=17
22
        do i=1, n
23
           seed(i)=j+37*(i-1)
24
        enddo
25
        call random_seed(put=seed)
26
        write(*,*) seed
27
28
     DO ith=1,h
29
     ! Initial X, counters
30
        x(ith, 1) = 0.0
31
        x(ith, 2) = 0.0
32
        x(ith, 3) = 0.0
33
        x(ith, 4) = 0.5
34
        x(ith, 5) = 0.5
35
        x(ith, 6) = 0.0
36
     end do
37
        i=0
38
39
        j=0
40
        acpt=0.0
41
42
     ! Initialize observables, counters
43
        ekin1=0.0
44
        ekin2=0.0
45
        epot1=0.0
        epot2=0.0
46
47
        ekin=0.0
48
        epot=0.0
49
        k=0
50
51
     ! For the record
        OPEN (16, file='metropolis.out', FORM='FORMATTED', STATUS='UNKNOWN')
52
53
54
     ! Initial energies
55
        DO ith=1,h
56
        ekin=ekin+(x(ith,1)**2+x(ith,2)**2+x(ith,3)**2)/2.
        epot=epot+x(ith,6)
57
58
        enddo
59
        write (16, '(16, 2E14.6))') i, ekin, epot
60
```

```
61
           Limit length of chain to m
   62
           40 if (i.ge.m*h) go to 30
   63
        do ith=1,h
   64
        10 continue
             Trial X'
   65
   66
              xx=x
   67
        1
              momenta
   68
              call random_number(r(1:4))
   69
              s=SQRT(1.0/beta)
              xx(ith,1)=SQRT(-2.0*LOG(r(1)))*COS(pi2*r(3))*s
   70
   71
              xx(ith, 2) = SQRT(-2.0*LOG(r(1)))*SIN(pi2*r(3))*s
   72
              xx(ith, 3) = SQRT(-2.0*LOG(r(2)))*COS(pi2*r(4))*s
   73
        !
              positions
   74
              call random_number(r(1:3))
   75
              xx(ith, 4)=r(1)
   76
              xx(ith,5)=r(2)
   77
              xx(ith, 6) = -LOG(1.0-r(3))/beta
   78
   79
              Metropolis update
        1
              if (p(xx).ge.p(x)) then
   80
   81
        1
                Accept
   82
                goto 20
   83
              else
   84
                call random_number(r(1))
   85
                if (p(xx).gt.r(1)*p(x)) then
   86
        ļ
                  Accept
   87
                  goto 20
   88
                else
   89
        1
                  Reject
   90
                  j=j+1
   91
                    goto 10
   92
                  end if
   93
               endif
   94
           20 continue
   95
   96
   97
               Next element of chain
        1
   98
               i=i+1
   99
               DO ii=1, n
  100
                 x(ith,ii)=xx(ith,ii)
  101
  102
               acpt=REAL(i)/REAL(i+j)
  103
        end do
  104
               energies
  105
        ekin=0.
  106
        epot=0.
  107
        do ith=1,h
  108
               ekin=ekin+(x(ith,1)**2+x(ith,2)**2+x(ith,3)**2)/2.0
  109
               epot=epot+x(ith,6)
        end do
  110
  111
               for the record
              write (16, '(16, 2E14.6))') i, ekin, epot
  112
  113
  114
               if (mod(i,l).EQ.0) then
  115
                 accumulate 1st and 2nd moments
        !
  116
                 k=k+1
                 observables
  117
        1
  118
                 ekin1=ekin1+ekin
  119
                 ekin2=ekin2+ekin**2
  120
                 epot1=epot1+epot
- 2 -
```

```
121
             epot2=epot2+epot**2
122
             write (*,'(I6,F8.4,I6,E14.6)') i,acpt,k,(ekin1+epot1)/REAL(k)
123
           endif
124
125
           goto 40
126
        30 continue
127
128
129
     ! Final result
130
        etot1=ekin1+epot1
131
        etot2=(ekin1+epot1)**2
132
        C=beta**2*(etot2/real(k*h)-(etot1/real(k*h))**2)
        print*, C
133
134
        ekin1=ekin1/REAL(k)
135
        ekin2=ekin2/REAL(k)
136
        ekin2=SQRT(ekin2-ekin1**2)/SQRT(REAL(k))
137
        epot1=epot1/REAL(k)
138
        epot2=epot2/REAL(k)
        epot2=SQRT(epot2-epot1**2)/SQRT(REAL(k))
139
        write (*,'(1P,ES14.6,ES12.4)') ekin1,ekin2
140
        write (*,'(1P,ES14.6,ES12.4)') epot1,epot2
141
142
143
     144
145
       CONTAINS
146
147
148
149
       FUNCTION p(x)
150
151
     ! Probability distribution function, def
152
153
       IMPLICIT NONE
154
       REAL :: p
       REAL, DIMENSION(h,n), INTENT(IN) :: x
155
156
       INTEGER i
157
158
       p = 0.0
159
     ! outside box
160
       DO i=4,5
161
         IF (x(ith,i).LT.0.0 .OR. x(ith,i).GT.1.0) RETURN
162
       END DO
     ! below bottom
163
164
       IF (x(ith,6).LT.0.0) RETURN
165
166
     ! Boltzmann
167
       p=(x(ith,1)**2+x(ith,2)**2+x(ith,3)**2)/2.0
168
       p=p+x(ith,6)
169
       p=EXP(-beta*p)
170
       RETURN
171
172
173
       END FUNCTION p
174
175
176
177
       END PROGRAM metropolis
178
179
     ! -----
180
```

- 3 -