```
!  ----------------------------------------------------------------

!  Metropolis algorithm, generic illustration
!
!  [Note: One point particle in 3d cube. You will not get "good
!  statistics" with one particle, of course. Illustration only!]

   PROGRAM metropolis

   IMPLICIT NONE

   INTEGER, PARAMETER :: n=6, m=1000, l=50, h=100
   REAL, PARAMETER :: beta=1.0, pi2=2*3.14159265358979

   REAL, DIMENSION(h,n) :: x,xx
   REAL :: acpt,s, ekin,ekin1,ekin2,epot,epot1,epot2,C,etot1,etot2
   INTEGER :: i,ii,j,k,ith
   INTEGER, DIMENSION(n) :: seed
   REAL, DIMENSION(4) :: r

   j=17
   do i=1,n
      seed(i)=j+37*(i-1)
   enddo
   call random_seed(put=seed)
   write(*,*) seed

DO ith=1,h
!  Initial X, counters
   x(ith,1)=0.0
   x(ith,2)=0.0
   x(ith,3)=0.0
   x(ith,4)=0.5
   x(ith,5)=0.5
   x(ith,6)=0.0

end do
   i=0
   j=0
   acpt=0.0

!  Initialize observables, counters
   ekin1=0.0
   ekin2=0.0
   epot1=0.0
   epot2=0.0
   ekin=0.0
   epot=0.0
   k=0

!  For the record
   OPEN (16,file='metropolis.out',FORM='FORMATTED',STATUS='UNKNOWN')

!  Initial energies
   DO ith=1,h
   ekin=ekin+(x(ith,1)**2+x(ith,2)**2+x(ith,3)**2)/2.
   epot=epot+x(ith,6)
   enddo
   write (16,'(I6,2E14.6))') i, ekin,epot

```

```
!  Limit length of chain to m
   40 if (i.ge.m*h) go to 30
do ith=1,h
10 continue
!    Trial X'
     xx=x
!    momenta
     call random_number(r(1:4))
     s=SQRT(1.0/beta)
     xx(ith,1)=SQRT(-2.0*LOG(r(1)))*COS(pi2*r(3))*s
     xx(ith,2)=SQRT(-2.0*LOG(r(1)))*SIN(pi2*r(3))*s
     xx(ith,3)=SQRT(-2.0*LOG(r(2)))*COS(pi2*r(4))*s
!    positions
     call random_number(r(1:3))
     xx(ith,4)=r(1)
     xx(ith,5)=r(2)
     xx(ith,6)=-LOG(1.0-r(3))/beta

!    Metropolis update
     if (p(xx).ge.p(x)) then
!      Accept
       goto 20
     else
       call random_number(r(1))
       if (p(xx).gt.r(1)*p(x)) then
!        Accept
         goto 20
       else
!        Reject
         j=j+1
           goto 10
         end if
      endif

   20 continue

!     Next element of chain
      i=i+1
      DO ii=1,n
        x(ith,ii)=xx(ith,ii)
      ENDDO
      acpt=REAL(i)/REAL(i+j)
end do
!     energies
ekin=0.
epot=0.
do ith=1,h
      ekin=ekin+(x(ith,1)**2+x(ith,2)**2+x(ith,3)**2)/2.0
      epot=epot+x(ith,6)
end do
!     for the record
      write (16,'(I6,2E14.6))') i, ekin,epot

      if (mod(i,l).EQ.0) then
!       accumulate 1st and 2nd moments
        k=k+1
!       observables
        ekin1=ekin1+ekin
        ekin2=ekin2+ekin**2
        epot1=epot1+epot
```

```
          epot2=epot2+epot**2
          write (*,'(I6,F8.4,I6,E14.6)') i,acpt,k,(ekin1+epot1)/REAL(k)
        endif

        goto 40

     30 continue

  !  Final result
     etot1=ekin1+epot1
     etot2=(ekin1+epot1)**2
     C=beta**2*(etot2/real(k*h)-(etot1/real(k*h))**2)
     print*, C
     ekin1=ekin1/REAL(k)
     ekin2=ekin2/REAL(k)
     ekin2=SQRT(ekin2-ekin1**2)/SQRT(REAL(k))
     epot1=epot1/REAL(k)
     epot2=epot2/REAL(k)
     epot2=SQRT(epot2-epot1**2)/SQRT(REAL(k))
     write (*,'(1P,ES14.6,ES12.4)') ekin1,ekin2
     write (*,'(1P,ES14.6,ES12.4)') epot1,epot2

  !  ---------------------------------------------------------------------

     CONTAINS

  !  ---------------------------------------------------------------------

     FUNCTION p(x)

  ! Probability distribution function, def

     IMPLICIT NONE
     REAL :: p
     REAL, DIMENSION(h,n), INTENT(IN) :: x
     INTEGER i

     p=0.0
  ! outside box
     DO i=4,5
       IF (x(ith,i).LT.0.0 .OR. x(ith,i).GT.1.0) RETURN
     END DO
  ! below bottom
     IF (x(ith,6).LT.0.0) RETURN

  ! Boltzmann
     p=(x(ith,1)**2+x(ith,2)**2+x(ith,3)**2)/2.0
     p=p+x(ith,6)
     p=EXP(-beta*p)

     RETURN

     END FUNCTION p

  ! ---------------------------------------------------------------------

     END PROGRAM metropolis

  ! ---------------------------------------------------------------------
```