```
      program argon
c     file name : argon.f
c     tested by ashok kumar das and BLT (16-3-1998, 22-12-2005)
c     remodelled on 19 november 2012
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     molecular dynamics with periodic boundary conditions
c     program for simulation of the structure of liquid argon
c     energy units are in 10 * joule per mole
c     random number generator is from CDC library
c     ** this is for 64 argon atoms **
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
```

```
c     input data file
      open (11,file='ncoord.0')
c
c     output files
      open (12,file='argoout')
      open (13,file='bindat')
      open (14,file='grdat')
      open (15,file='positions')
      open (16,file='velocities')
      open (17,file='arvmd.xyz')
c
      open (21,file='testxij')
      open (22,file='testvij')
c
      pi=4.0*atan(1.0)
c
      read (11,*) irstrt,nrm1
      read (11,*) nsubav,noutf,ntot,maxbin
      read (11,*) tzero,delt,char,el,delr,q1
c
      write (12,101) irstrt,nrm1
      write (12,102) nsubav,noutf,ntot,maxbin
      write (12,103) tzero,delt,char,el,delr,q1
  101 format (2i3)
  102 format (4i5)
```

```
 103 format (6f10.5)
c
c     xhl is the half box length
      xhl=0.5*el
c
c     nq=no of quantities to be averaged
      nq=5
c
c     amasar is the mass of argon atom
      amasar=39.945
c
c     ntot3 is 3 times the no of argon atoms
      ntot3=3*ntot
c
c     boltz is boltzmann constant in (10*joule) per mole/deg K
      boltz=0.83106
c
c     sigma (A) and epsilon (K) parameters for argon
      sigma=3.41
      epsilon=120.0
c     convert epsilon into 10*joule/mole
      epsilon=120.0*boltz
c
c     e2dbk is electronic charge squared/k
      e2dbk=167022.89
```

```
      esq=e2dbk*boltz
c
c     char is the charge on the argon atoms (if chosen)
      e2solv=esq*char*char
c
c     con1 is the constant used in verlet subroutine
      con1=(delt*delt)/(2.0*amasar)
c
c     vrms1 is the rms velocity, dependent on room temperature
      vrms1=sqrt(boltz*tzero/amasar)
c
c     if irstrt=1, read initial coordinates and velocities
c     if irstrt=0, read initial coordinates only
c
      if (irstrt.eq.0) go to 7500
c
c     variable nrm1 is used to locate previous set of
c     positions and velocities in the 'bindat' file
c
      do 110 k=1,nrm1
      read (13,*) x0sp
      read (13,*) xsp
      read (13,*) vsp
  110 continue
c
```

```
c     writing the initial configurations
      write (12,111)
  111 format (/,1x,'initial positions')
      write (12,113) xsp
      write (12,112)
  112 format (/,1x,'initial velocities')
      write (12,113) vsp
  113 format (3f10.4)
      write (12,114) q1
  114 format (/,1x, 'starting random number =',f19.16)
      go to 7600
c
 7500 continue
c     irstrt=0 for cold start
      write (12,115)
  115 format (/,1x,'initial positions for a cold start')
      do 120 i=1,ntot3,3
      ip2=i+2
      read  (11,  *) (xsp(k),k=i,ip2)
      write (12,121) (xsp(k),k=i,ip2)
  120 continue
  121 format (3f10.5)
c
c     determine initial velocities from a gaussian distribution
c
```

```
      sum=0.0
      ni1=3*ntot
      call setrn (q1)
      call savern (q1)
      do 130 i=1,ni1
      call gauss (vrms1,0.d0,xx)
      sum=sum+xx*xx
      vsp(i)=xx
  130 continue
      vrmsn1=sqrt(sum/float(ni1))
c
      write (12,131) vrms1,vrmsn1
  131 format (/,1x,'vrms (Ar)          =',f10.4,
     1       /,1x,'vrms (Ar) calculated =',f10.4)
      write (12,132)
  132 format (/,1x,'initial velocities for a cold start')
      write (12,133) (vsp(i),i=1,ntot3)
  133 format (3f10.5)


 7600 continue
c     redefine coordinates and velocities as double index arrays


      do 140 k=1,ntot3
      j=(k-1)/3+1
      i=k-(3*j)+3
```

```
      x(i,j)=xsp(k)
      x01(i,j)=x(i,j)
      v(i,j)=vsp(k)
  140 continue
c
c     calculate initial temperature
      write (12,141)
  141 format (/,1x,'temperature study of starting configuration')
      call atmprtr
      write (12,142) t
      write ( *,142) t
  142 format (1x,'initial temperature =',f10.4)
c
c     calculate initial forces
      write (12,143)
  143 format (/,1x,'*** list initial forces ***',/)
      write (12,144)
  144 format (9x,'n',10x,'fx',10x,'fy',10x,'fz',10x,'f',/)
      call forcal
      do 160 n=1,ntot
      sum=0.0
      do 150 i=1,3
      sum=sum+f(i,n)**2
  150 continue
      fmag=sqrt(sum)
```

```
      write (12,161) n,f(1,n),f(2,n),f(3,n),fmag
  160 continue
  161 format (i10,4f12.4)
c
c     initialise the histogram for rdf
      do 170 i=1,maxbin
      hist(i)=0.0
  170 continue
c
c     writing starting configuration for vmd
      nout=0
      call vmdwrite
c
c     start main dynamics
      write (12,171)
  171 format (/,1x,'*** start main dynamics ***',/)
c
c     entering the main loop
      nc1=0
      nsub1=0
c
      do 1000 nout=1,noutf
c
c     writing configurations for vmd in every 50 steps
      if (((nout/50)*50).eq.nout) then
```

```
      call vmdwrite
      endif
c
c     entering the subdnamic loop
c     do 250 nstep=1,nsubav
      nc1=nc1+1
      call acomcor
      call dynama
c
      write (21,*) x(1,1),x(2,1),x(3,1)
      write (22,*) v(1,1),v(2,1),v(3,1)
c
      nsub1=nsub1+1
c
c     these are for the subaverages
      qfsd(nsub1,1)=petot
      qfsd(nsub1,2)=t
      aketot=1.5*boltz*t
      etot=petot+aketot
      qfsd(nsub1,3)=aketot
      qfsd(nsub1,4)=etot
      qfsd(nsub1,5)=ptot
c
      write (15,172) x
      write (16,172) v
```

```
 172 format (3f12.5)
c
c     updating coordinates
      do 180 i=1,ntot
      x01(1,i)=x(1,i)
      x01(2,i)=x(2,i)
      x01(3,i)=x(3,i)
      x(1,i)=xnew(1,i)
      x(2,i)=xnew(2,i)
      x(3,i)=xnew(3,i)
 180 continue
c
      call bring
c
c     saving configurations in every 1000 steps
c     if ((nc1/1000*1000).eq.nc1) then
      if (((nout/1000)*1000).eq.nout) then
      do 220 i=1,ntot
      i3=3*(i-1)
      do 210 j=1,3
      x0sp(i3+j)=x01(j,i)
      xsp(i3+j)=x(j,i)
      vsp(i3+j)=v(j,i)
 210 continue
 220 continue
```

```
      write (13,*) x0sp
      write (13,*) xsp
      write (13,*) vsp
c     rewind 13
      endif
c
c     for calculating mean square displacements
c     run the diffu.f and msdp.f programs
c     difsq=0.0
c     do 230 i=1,ntot
c     rtsq=(x(1,i)*x(1,i)+x(2,i)*x(2,i)+x(3,i)*x(3,i))
c     rt=sqrt(rtsq)
c     r0sq=(x01(1,i)*x01(1,i)+x01(2,i)*x01(2,i)+x01(3,i)*x01(3,i))
c     r0=sqrt(r0sq)
c     delrtr0=rt-r0
c     difsq=difsq+delrtr0*delrtr0
c 230 continue
c     difsq=difsq/float(ntot)
c
c 250 continue
c     end of the subdynamic loop
c
c     write temperature as a check
      if (((nout/10)*10).eq.nout) then
      write (12,251) nout,t
```

```
      write ( *,251) nout,t
  251 format (1x,'nout =',i6,2x,'temperature =',e12.4)
      endif
c
c     calculate the subaverages
c     call avsd (nsub1,nq)
c     do 260 i=1,nq
c     sav1(nout,i)=sav(i)
c 260 continue
c
 1000 continue
c     end of the main loop
c
c     transferring subaverages to cumulative average
c     do 320 nout=1,noutf
c     do 310 i=1,nq
c     temp=sav1(nout,i)
c     qfsd(nout,i)=temp
c 310 continue
c 320 continue
c
c     write the final averages
      call avsd (noutf,nq)
      write (12,321)
  321 format (/,1x,'*** final averages over the entire simulation ***')
```

```
      write (12,322)
  322 format (5x,'i',8x,'sav(i)',7x,'ssdv(i)')
      do 330 i=1,nq
      write (12,331) i,sav(i),ssdv(i)
  330 continue
  331 format (2x,i4,2x,e12.4,2x,e12.4)
c
c     radial distribution functions
      write (14,332)
  332 format (8x,'r',11x,'g(r)',4x,'4*pi*rsq*g(r)',3x,'n(r)')
      call paircf
      rho=float(ntot)/(el*el*el)
      sum=0.0
      do 340 i=1,maxbin
      rr=float(i-1)*delr
      fpr2gr=rho*4.0*pi*rr*rr*rdf(i)
      sum=sum+fpr2gr*delr
      write (14,341) rr,rdf(i),fpr2gr,sum
  340 continue
  341 format (4f12.4)
c
c     look at the final configuration
      write (12,342)
  342 format (/,1x,'*** look at the final configuration ***',/)
      call atmprtr
```

```
      write (12,343) t
      write ( *,343) t
  343 format (/,1x,'final temperature =',f10.4)
c
      write (12,344)
  344 format (/,1x,'*** list final forces ***',/)
      write (12,345)
  345 format (9x,'n',10x,'fx',10x,'fy',10x,'fz',10x,'f',/)
      call forcal
      do 360 n=1,ntot
      sum=0.0
      do 350 i=1,3
      sum=sum+f(i,n)*f(i,n)
  350 continue
      fmag=sqrt(sum)
      write (12,361) n,f(1,n),f(2,n),f(3,n),fmag
  360 continue
  361 format (i10,4f12.4)
c
      do 380 i=1,ntot
      i3=3*(i-1)
      do 370 j=1,3
      xsp(i3+j)=x(j,i)
      vsp(i3+j)=v(j,i)
  370 continue
```

```
  380 continue
c
      write (12,381)
  381 format (/,1x,'final positions')
      do 390 i=1,ntot
      write (12,391) i,x(1,i),x(2,i),x(3,i)
  390 continue
  391 format (i5,2x,3f10.4)
c
      write (12,392)
  392 format (/,1x,'final velocities')
      do 400 i=1,ntot
      write (12,401) i,v(1,i),v(2,i),v(3,i)
  400 continue
  401 format (i5,2x,3f10.4)
c
      write (12,402) q1
  402 format (/,1x,'random number =',f19.16)
c
      write (13,*) x0sp
      write (13,*) xsp
      write (13,*) vsp
c
      close (11,status='keep')
      close (12,status='keep')
```

```
      close (13,status='keep')
      close (14,status='keep')
      close (15,status='keep')
      close (16,status='keep')
      close (17,status='keep')
      close (21,status='keep')
      close (22,status='keep')
c
      stop
      end
c
c     *****************
      subroutine forcal
c     *****************
c     calculates the forces on particles
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
```

```
      common/com9/nout
c
      dimension xxx(3)
c
c     initialise the potential energy
      petot=0.0
c
c     initialise the forces
      do 120 n=1,ntot
      do 110 j=1,3
      f(j,n)=0.0
  110 continue
  120 continue
c
c     argon-argon pair interactions
      nn1=ntot-1
      do 160 n=1,nn1
      np1=n+1
      do 150 m=np1,ntot
      rsqnm=0.0
c
      do 130 j=1,3
      dxx=x(j,m)-x(j,n)
c     minimum image distance between n and m
      xxx(j)=dxx-el*anint(dxx/el)
```

```
      rsqnm=rsqnm+xxx(j)*xxx(j)
  130 continue
c
      rss=sqrt(rsqnm)
c
c     calculate lennard-jones interactions up to
c     spherical cut off distance
      if (rss.gt.0.9*sigma.and.rss.lt.xhl) then
c
c     histograms for rdf
      nbin=int(rss/delr)+1
      if (nbin.le.maxbin) then
      hist(nbin)=hist(nbin)+2.0
      endif
c
c     calculate lennard-jones potential (u(r))
      mpn=m+n
      sgn=1.0
      if ((mpn/2)*2.ne.mpn) sgn=-1.0
c     if (rss.gt.sigma.and.rss.lt.xhl) then
      pecon1=4.0*epsilon*((sigma/rss)**12)
      pecon2=4.0*epsilon*((sigma/rss)**6)
c     else
c     pecon1=0.0
c     pecon2=0.0
```

```
c     endif
      pot=pecon1-pecon2
c     petot=petot+pot+sgn*(e2solv/rss)
      petot=petot+pot
c
c     force on the particle=-du/dr
c     if (rss.gt.sigma.and.rss.lt.xhl) then
      for1=(48.0*epsilon*(sigma**12))/(rss**14)
      for2=(24.0*epsilon*(sigma**6))/(rss**8)
c     else
c     for1=0.0
c     for2=0.0
c     endif
      temp1=(for1-for2)
c     temp2=sgn*e2solv/(rsqnm*rss)
      temp2=0.0
c
      do 140 j=1,3
      temp3=xxx(j)*(temp1+temp2)
      f(j,m)=f(j,m)+temp3
      f(j,n)=f(j,n)-temp3
  140 continue
      endif
c
  150 continue
```

```
 160 continue
c
c     compute the pressure from the virial
      vol=(el*el*el)
      virial=0.0
      do 180 n=1,ntot
      sum=0.0
      do 170 j=1,3
      sum=sum+(x(j,n)*f(j,n))
 170 continue
      virial=virial+sum
 180 continue
      pvir=virial/(3.0*vol)
c
c     ideal pressure
      pid=(boltz*tzero*float(ntot)/vol)
c
c     total pressure
      ptot=pid+pvir
c
      return
      end
c
c     *****************
      subroutine dynama
```

```
c     *****************
c     controls the dynamics of the system
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1          e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
c     calculate forces
      call forcal
c
c     calculate temperature
      call atmprtr
c
c     compute the next step position
      call averlet
c
c     recalculate temperature
      call atmprtr
```

```
c
      return
      end
c
c     *************************
      subroutine avsd (ndata,nq)
c     *************************
c     calculates averages and standard deviations
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
      do 120 i=1,nq
      accum=0.0
      do 110 j=1,ndata
      accum=accum+qfsd(j,i)
  110 continue
```

```
      sav(i)=accum/float(ndata)
  120 continue
c
      do 140 i=1,nq
      xx=0.0
      do 130 j=1,ndata
      xx=xx+(qfsd(j,i)-sav(i))**2
  130 continue
      ssdv(i)=sqrt(xx/float(ndata-1))
  140 continue
c
      return
      end
c
c     *****************
      subroutine paircf
c     *****************
c     calculates the pair correlation functions
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
```

```
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
      vol=el*el*el
      const=4.0*pi*(real(ntot)/vol)/3.0
c
      do 110 nbin=1,maxbin
      rlower=real(nbin-1)*delr
      rupper=rlower+delr
      fideal=(rupper**3)-(rlower**3)
      rdf(nbin)=hist(nbin)/real(nc1)/real(ntot)/(const*fideal)
  110 continue
c
      return
      end
c
c     ************************
      subroutine gauss (s,am,v)
c     ************************
      implicit real*8(a-h,o-z)
c
c     ix = odd integer less than 9 digits
c     s  = standard deviation of normal distribution
```

```
c     am = mean of the normal distribution
c     v  = value of the computed normal random variable
c
      a=0.0
      do 50 i=1,12
      q1=rannum(0)
      call savern (q1)
      a=a+q1
   50 continue
      v=(a-6.0)*s+am
      return
      end
c
c     ********************
      function rannum (i)
c     ********************
      implicit double precision (q)
      double precision rannum
      common/cran/qbase,qa1,qa2,qb1,qb2
c
      qd2=qa2*qb2
      qe2=dint(qd2/qbase)
      qc2=qd2-qbase*qe2
      qb1=dmod(qe2+dmod(qa1*qb2,qbase)
     1          +dmod(qa2*qb1,qbase),qbase)
```

```
      qb2=qc2
      j=i+1
      rannum=qb1/qbase
      return
      end
c
c     **********
      subroutine savern (q)
c     **********
      implicit double precision (q)
      common/cran/qbase,qa1,qa2,qb1,qb2
      q=(qb1+qb2/qbase)/qbase
      return
      end
c
c     **********
      subroutine setrn (q)
c     **********
      implicit real*8(q)
      common/cran/qbase,qa1,qa2,qb1,qb2
      qa1=2057713.0
      qa2=16676923.0
      qbase=2**24
      if (q.ge.1.0.or.q.le.0.0) call ranrn (q)
      qc=dint(qbase*(qbase*q))
```

```
      qb1=dint(qc/qbase)
      qb2=qc-qb1*qbase
      qb1=dmod(qb1,qbase)
      qb2=dint(qb2/2.0)*2.0+1.0
      return
      end
c
c     ********************
      subroutine ranrn (q)
c     ********************
      implicit real*8(q)
      double precision rannum
      if (abs(q-0.5).lt.0.5) go to 10
c
      i1=5
      i2=10
      i3=1985
      t1=24.0
      q1=(i1/13.0)+(i2/32.0)+(i3/100.0)+(t1/86400.0)
      q1=mod(q1,1.d0)
      call setrn (q1)
c
c     randomize twice
      q2=rannum(0)
      q2=rannum(0)
```

```
c
c     now find seed the and print it
    5 call savern (q)
      write (12,6) q
    6 format (/,1x,'random number seed in ranrn =',f19.16)
      return
   10 call setrn (q)
      go to 5
      end
c
c     ****************
      subroutine bring
c     ****************
c     this subroutine is to bring those particles whose coordinates
c     go beyond the box-length into the box
c
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
```

```
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
      do 120 j=1,ntot
      do 110 i=1,3
      x(i,j)=x(i,j)-el*anint(x(i,j)/el)
  110 continue
  120 continue
c
c     do 20 j=1,ntot
c     do 10 i=1,3
c     m=int(x(i,j)/xhl)
c     if (m.ne.0) then
c     xx=el*float(m)
c     x(i,j)=x(i,j)-xx
c     x01(i,j)=x01(i,j)-xx
c     endif
c  10 continue
c  20 continue
c
      return
      end
c
c     ********************
      subroutine vmdwrite
```

```
c     ******************
c     writes the input file for
c     the argon atoms for vmd
c
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
      write (17,101) ntot,ntot,nout
  101 format (i6,/,1x,'total no of atoms',i6,2x,'step no',i6)
c
c     writing the coordinates for vmd
c     atom no 15 is tagged
      do 110 i=1,ntot
      if (i.lt.15) then
      write (17,111) x(1,i),x(2,i),x(3,i)
      endif
```

```
      if (i.eq.15) then
      write (17,112) x(1,i),x(2,i),x(3,i)
      endif
      if (i.gt.15) then
      write (17,113) x(1,i),x(2,i),x(3,i)
      endif
  110 continue
  111 format (2x,'Ar',2x,3f12.6)
  112 format (2x,'N ',2x,3f12.6)
  113 format (2x,'Ar',2x,3f12.6)
c
      return
      end
c
c     ******************
      subroutine acomcor
c     ******************
c     centre of mass correction for the box
c
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
```

```
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
c     amasar=mass of a single argon molecule
c     tmasar=mass of all the molecules in the box
c
      ramasar=1.0/amasar
      tmasar=float(ntot)*amasar
      rboxmas=1.0/tmasar
c
      sumx=0.0
      sumy=0.0
      sumz=0.0
c
c     total momentum in each direction
      do 120 i=1,ntot
      sumx=sumx+amasar*v(1,i)
      sumy=sumy+amasar*v(2,i)
      sumz=sumz+amasar*v(3,i)
  120 continue
c
      facx=sumx*rboxmas
```

```
      facy=sumy*rboxmas
      facz=sumz*rboxmas
c
c     correcting velocities
      do 140 i=1,ntot
      v(1,i)=v(1,i)-facx
      v(2,i)=v(2,i)-facy
      v(3,i)=v(3,i)-facz
  140 continue
c
      return
      end
c
c     ******************
      subroutine averlet
c     ******************
c     regular verlet algorithm for next position
c
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
```

```
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
c     rmasar is reciprocal mass of an argon molecule
      rmasar=1.0/amasar
c
c     deltsq is delt*delt
      deltsq=(delt*delt)
c
c     regular verlet algorithm
      do 120 i=1,ntot
c
      if (nout.eq.1) then
      xnew(1,i)=x(1,i)+delt*v(1,i)
     1          +0.5*deltsq*f(1,i)*rmasar
      xnew(2,i)=x(2,i)+delt*v(2,i)
     1          +0.5*deltsq*f(2,i)*rmasar
      xnew(3,i)=x(3,i)+delt*v(3,i)
     1          +0.5*deltsq*f(3,i)*rmasar
      endif
c
      if (nout.gt.1) then
      xnew(1,i)=2.0*x(1,i)-x01(1,i)+deltsq*f(1,i)*rmasar
```

```
      xnew(2,i)=2.0*x(2,i)-x01(2,i)+deltsq*f(2,i)*rmasar
      xnew(3,i)=2.0*x(3,i)-x01(3,i)+deltsq*f(3,i)*rmasar
      endif
c
  120 continue
c
c     calculate velocities
      coeffvel=1.0/(2.0*delt)
      do 130 i=1,ntot
      v(1,i)=coeffvel*(xnew(1,i)-x01(1,i))
      v(2,i)=coeffvel*(xnew(2,i)-x01(2,i))
      v(3,i)=coeffvel*(xnew(3,i)-x01(3,i))
  130 continue
c
      return
      end
c
c     ******************
      subroutine atmprtr
c     ******************
c     calculates temperature from velocities
c
      implicit real*8(a-h,o-z)
      common/com1/x01(3,64),x(3,64),xnew(3,64),v(3,64),f(3,64)
      common/com2/el,xhl,delr
```

```
      common/com3/ntot,nc1,maxbin
      common/com4/boltz,amasar,sigma,epsilon,delt,
     1           e2solv,petot,aketot,etot
      common/com5/pi,con1,tzero,t,tcor,pvir,pid,ptot
      common/com6/hist(100),rdf(100)
      common/com7/sav(5),ssdv(5),qfsd(10000,5),sav1(100,5)
      common/com8/x0sp(192),xsp(192),vsp(192)
      common/com9/nout
c
c     calculate kinetic energy
      sum=0.0
      do 110 i=1,ntot
      vsq=v(1,i)*v(1,i)+v(2,i)*v(2,i)+v(3,i)*v(3,i)
      sum=sum+amasar*vsq
  110 continue
      aketot=0.5*sum
c
c     temperature from kinetic energy
c     allen and tildesley, page 47
c
c     for no shake
      free=3.0*boltz*float(ntot)
      t=(2.0*aketot)/free
c
c     factor for velocity rescaling
```

```
      if (t.gt.0.0) then
      tcor=sqrt(tzero/t)
      else
      tcor=1.0
      endif
c
c     rescale velocities, if required
      if (abs(tzero-t).gt.2.0) then
c
      do 120 i=1,ntot
      v(1,i)=tcor*v(1,i)
      v(2,i)=tcor*v(2,i)
      v(3,i)=tcor*v(3,i)
  120 continue
c
c     recalculate kinetic energy and temperature
      sum2=0.0
      do 130 i=1,ntot
      vsq=v(1,i)*v(1,i)+v(2,i)*v(2,i)+v(3,i)*v(3,i)
      sum2=sum2+amasar*vsq
  130 continue
      aketot=0.5*sum2
      t=(2.0*aketot)/free
c
      endif
```

```
c
      return
      end
c
```