SUPPORTING INFORMATION

The Accounting of Noise to Solve the Problem of Negative Populations in Approximate Accelerated Stochastic Simulations

Shantanu Kadam[†] and Kumar Vanka[†*]

[†]*Physical Chemistry Division, National Chemical Laboratory, Dr. Homi Bhabha Road,*

Pashan, Pune, Maharashtra – 411 008, India

*Corresponding author. E-mail: k.vanka@ncl.res.in

(I) FIGURES:

(S1) The comparison of trajectories for the means and CVs of the probability distributions for some key species for the case of the Carletti-Burrage Model (Equation 1) discussed in the manuscript using SSA, G-P, BD-τ of Chatterjee-Vlachos-Katsoulakis for coarse-grain factor, $f = 4$ and RRA-Noise.

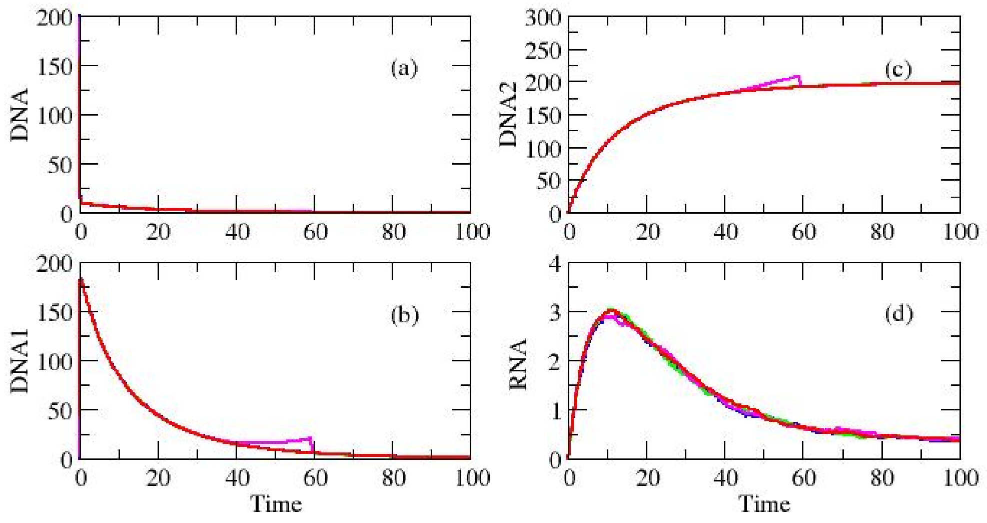

Figure S1-(a) The trajectories of the means [(a)-(d)] for the probability distributions of the species DNA, DNA1, DNA2 and RNA using SSA (blue curve), G-P (green curve), BD-τ of Chatterjee-Vlachos-Katsoulakis (magenta curve) and RRA-Noise (red curve).

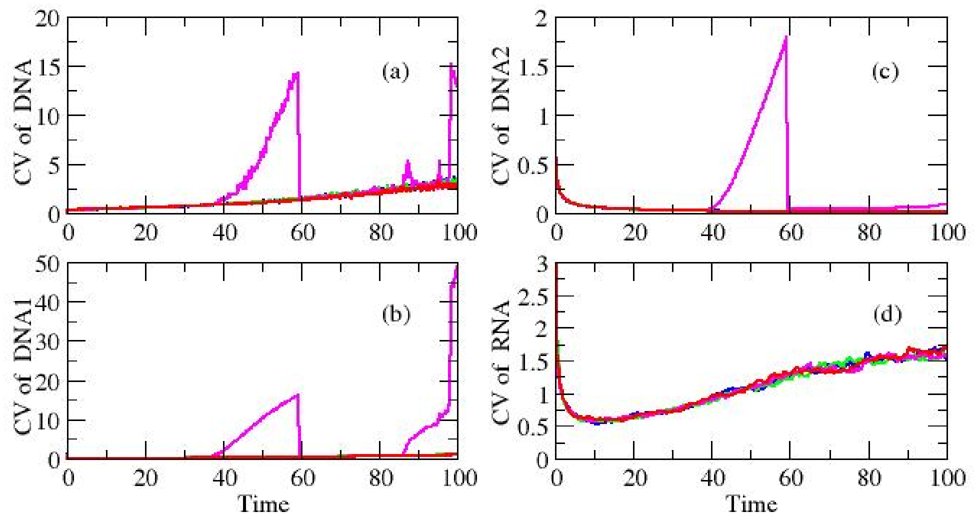

Figure S1-(b) The trajectories of the CVs [(a)-(d)] for the probability distributions of the species DNA, DNA1, DNA2 and RNA using SSA (blue curve), G-P (green curve), BD-τ of Chatterjee-Vlachos-Katsoulakis (magenta curve) and RRA-Noise (red curve).

Table S1. The average values of CPU time (in seconds) taken by different simulation methods; the coarse grain factor, *f*, for the BD-τ method is taken as 4.0 in the simulations.

Simulation Methods	SSA	G-P	BD-τ	RRA-Noise
CPU time (sec)	5.029	14.970	32.529	4.234

(S2) The comparison of trajectories for means and CVs of the probability distributions for the species X_1, X_2, X_3 for Simple Isomerization Reaction Model (Equation 2) discussed in the manuscript using SSA, G-P, BD-τ of Chatterjee-Vlachos-Katsoulakis for coarse-grain factor, $f = 50$, and RRA-Noise.

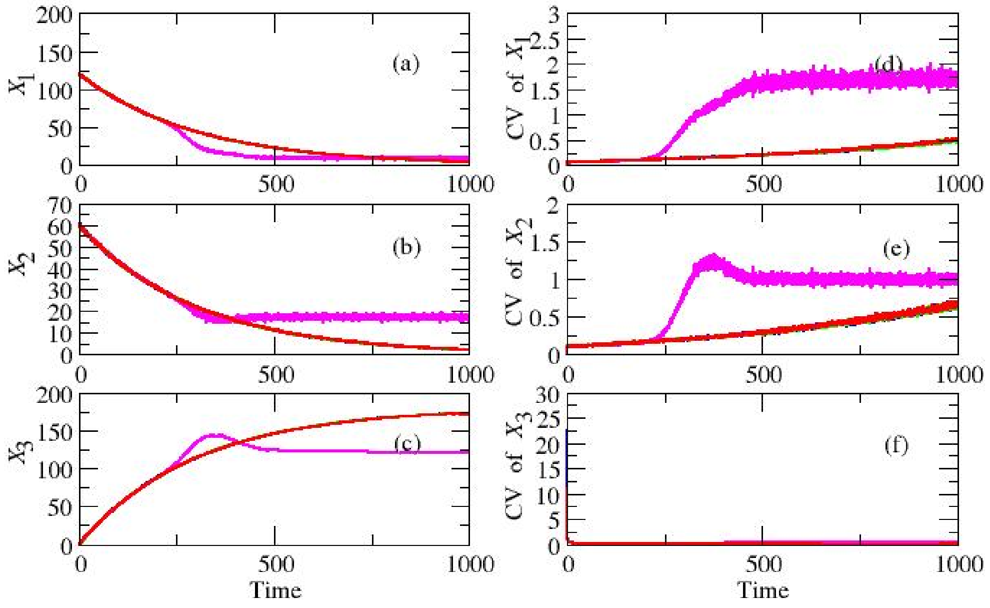

Figure S2. The trajectories of the means [(a)-(c)] and CVs [(d)-(f)] for the probability distributions of the species X_1, X_2, X_3 using SSA (blue curve), G-P (green curve), BD-τ of Chatterjee-Vlachos-Katsoulakis (magenta curve) and RRA-Noise (red curve).

Table S2. The average values of the CPU time (in seconds) taken by the different simulation methods for the case of the Simple Isomerization Reaction Model; the coarse grain factor, f, for the BD-τ method is taken as 50.0 in the simulations.

Simulation Methods	SSA	G-P	BD-τ	RRA-Noise
CPU time (sec)	83.919	33.131	25.186	36.354

(S3) The comparison of trajectories for means and CVs of the probability distributions for the species X_1, X_2, X_3 for Simple Isomerization Reaction Model (Equation 2) discussed in the manuscript using SSA, G-P, BD-τ of Chatterjee-Vlachos-Katsoulakis for coarse-grain factor, $f = 100$, and RRA-Noise.

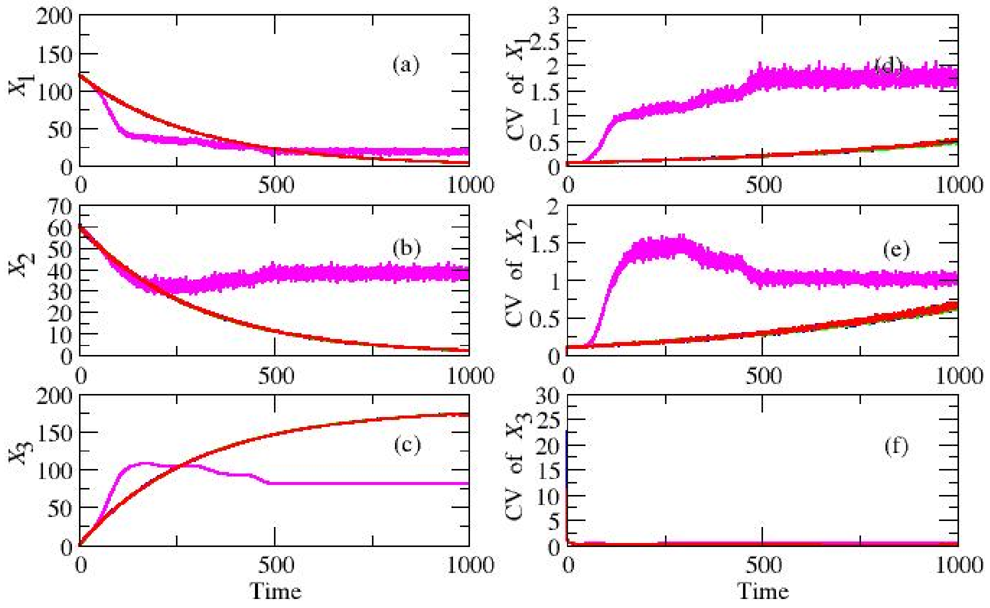

Figure S3. The trajectories of the means [(a)-(c)] and CVs [(d)-(f)] for the probability distributions of the species X_1, X_2, X_3 using SSA (blue curve), G-P (green curve), BD-τ of Chatterjee-Vlachos-Katsoulakis (magenta curve) and RRA-Noise (red curve).

Table S3. The average values of the CPU time (in seconds) taken by the different simulation methods for the case of the Simple Isomerization Reaction Model; the coarse grain factor, *f*, for the BD-τ method is taken as 100.0 in the simulations.

Simulation Methods	SSA	G-P	BD-τ	RRA-Noise
CPU time (sec)	83.919	33.131	22.994	36.354

(S4) The comparison of trajectories for means and CVs of the probability distributions for the species X_1, X_2, X_3 for the Model of First Order Reactions (Equation 3) discussed in the manuscript using SSA, BD-τ of Chatterjee-Vlachos-Katsoulakis for coarse-grain factor, $f = 5000$, and RRA-Noise.

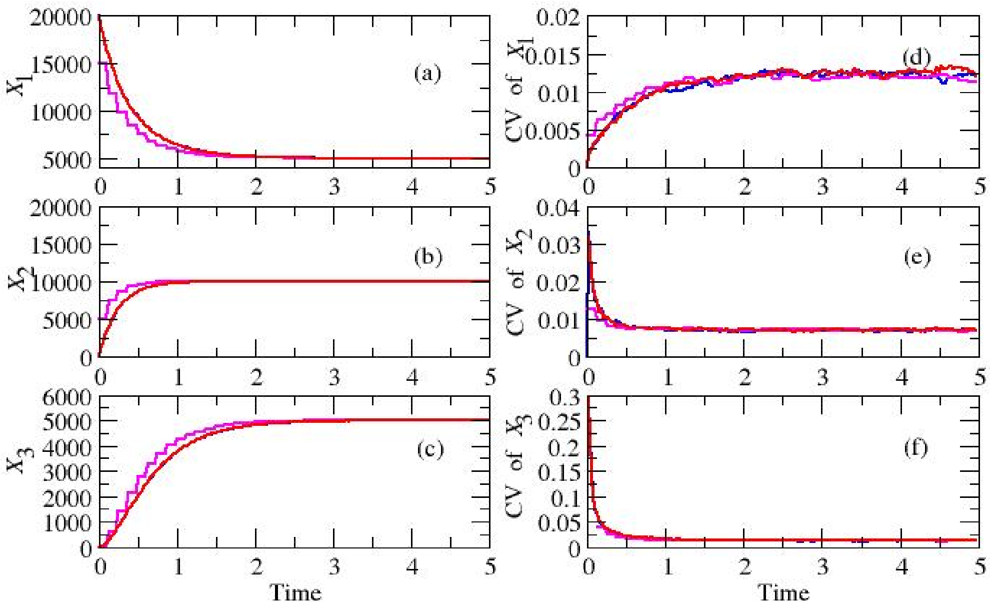

Figure S4. The trajectories of the means [(a)-(c)] and CVs [(d)-(f)] for the probability distributions of the species X_1, X_2, X_3 using SSA (blue curve), BD-τ of Chatterjee-Vlachos-Katsoulakis (magenta curve) and RRA-Noise (red curve).

Table S4. The average values of the CPU time (in seconds) taken by the different simulation methods for the case of the Model of First Order Reactions; the coarse grain factor, f, for the BD-τ method is taken as 5000.0 in the simulations.

Simulation Methods	SSA	BD-τ	RRA-Noise
CPU time (sec)	18.141	1.093	9.557

(S5) The comparison of trajectories for means and CVs of the probability distributions for the species X_1, X_2, X_3 for the Model of First Order Reactions (Equation 3) discussed in the manuscript using SSA, BD-τ of Chatterjee-Vlachos-Katsoulakis for coarse-grain factor, $f = 10000$, and RRA-Noise.

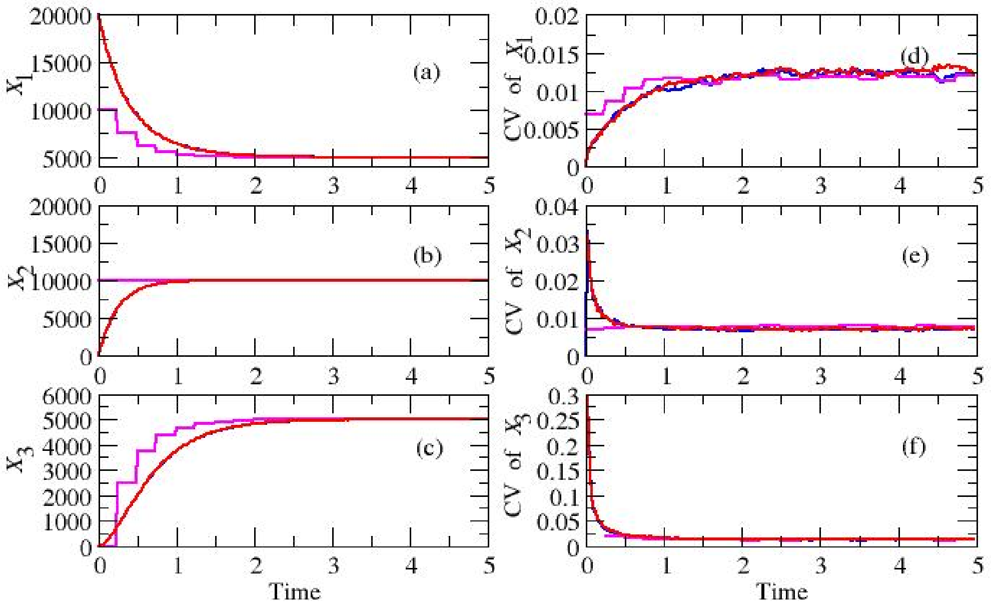

Figure S5. The trajectories of the means [(a)-(c)] and CVs [(d)-(f)] for the probability distributions of the species X_1, X_2, X_3 using SSA (blue curve), BD-τ of Chatterjee-Vlachos-Katsoulakis (magenta curve) and RRA-Noise (red curve).

Table S5. The average values of the CPU time (in seconds) taken by the different simulation methods for the case of the Model of First Order Reactions; the coarse grain factor, *f*, for the BD-τ method is taken as 10000.0 in the simulations.

Simulation Methods	SSA	BD-τ	RRA-Noise
CPU time (sec)	18.141	0.974	9.557

(S6) The comparison of trajectories for means and CVs of the probability distributions for the species X_1, X_2, X_3 for the Simple Model System (Equation 4) discussed in the manuscript using SSA, BD-τ of Chatterjee-Vlachos-Katsoulakis for coarse-grain factor, $f = 500$, and RRA-Noise.

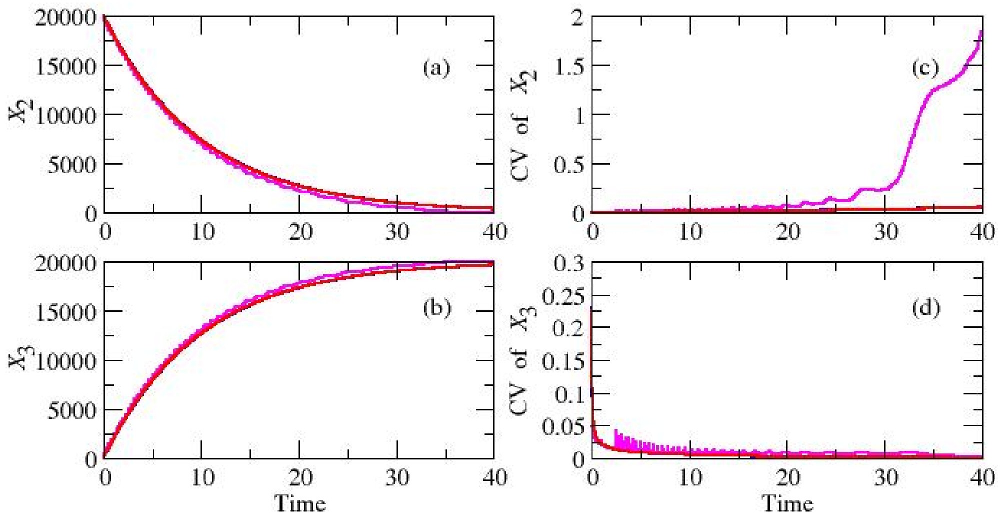

Figure S6. The trajectories of the means [(a) and (b)] and CVs [(c) and (d)] for the probability distributions of the species X_2, X_3 using SSA (blue curve), BD-τ of Chatterjee-Vlachos-Katsoulakis (magenta curve) and RRA-Noise (red curve).

Table S6. The average values of the CPU time (in seconds) taken by the different simulation methods for the case of the Simple Model System; the coarse grain factor, f, for the BD-τ method is taken as 500.0 in the simulations.

Simulation Methods	SSA	BD-τ	RRA-Noise
CPU time (sec)	8.298	6.160	7.257

(S7) The comparison of trajectories for means and CVs of the probability distributions for the species X_1, X_2, X_3 for the Simple Model System (Equation 4) discussed in the manuscript using SSA, BD-τ of Chatterjee-Vlachos-Katsoulakis for coarse-grain factor, $f = 1000$, and RRA-Noise.

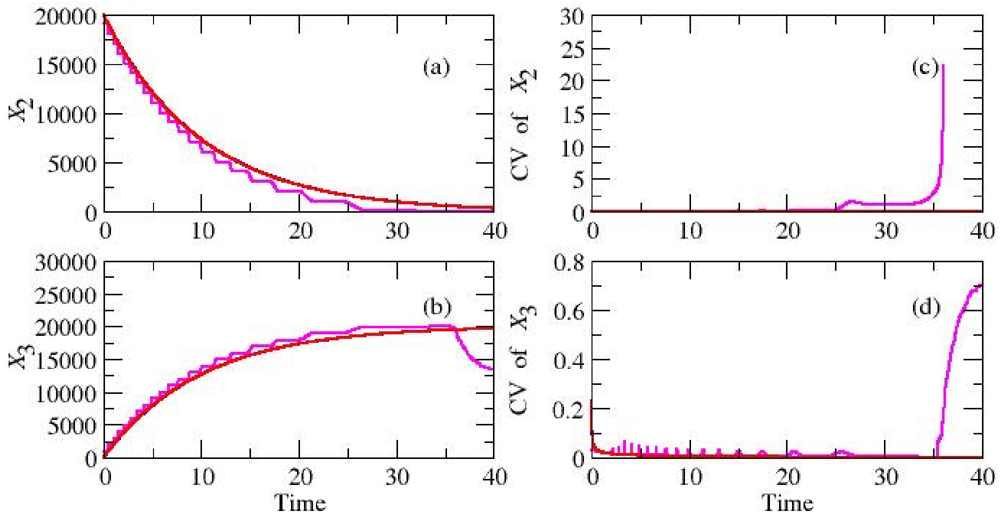

Figure S7. The trajectories of the means [(a) and (b)] and CVs [(c) and (d)] for the probability distributions of the species X_2, X_3 using SSA (blue curve), BD-τ of Chatterjee-Vlachos-Katsoulakis (magenta curve) and RRA-Noise (red curve).

Table S7. The average values of the CPU time (in seconds) taken by the different simulation methods for the case of the Simple Model System; the coarse grain factor, f, for the BD-τ method is taken as 1000.0 in the simulations.

Simulation Methods	SSA	BD-τ	RRA-Noise
CPU time (sec)	8.298	6.256	7.257

(II) TABLES:

Table S8. The rate constants for the different reactions in the Carletti-Burrage Model discussed in the manuscript.

Rate Constants used in the simulation	Numerical values of the Rate Constants
Rate constant of R_1 (c_1)	0.078
Rate constant of R_2 (c_2)	3.9E-3
Rate constant of R_3 (c_3)	7.0E-4
Rate constant of R_4 (c_4)	0.043
Rate constant of R_5 (c_5)	0.083
Rate constant of R_6 (c_6)	0.5
Rate constant of R_7 (c_7)	0.020
Rate constant of R_8 (c_8)	0.479
Rate constant of R_9 (c_9)	2.0E-4
Rate constant of R_{10} (c_{10})	8.765E-12

Table S9. The initial values of the different species in the Carletti-Burrage Model discussed in the manuscript.

Species used in the simulation	Numerical values of the Species
DNA	200
DNA1	000
DNA2	000
D	600
m	200
RNA	000

Table S10. The rate constants for the different reactions in the Simple Isomerization Reaction Model as wells as the initial values of different species.

Rate Constants used in the	Numerical values of the

simulation	Rate Constants
Rate constant of R_1 (c_1)	1.0
Rate constant of R_2 (c_2)	2.0
Rate constant of R_3 (c_3)	0.01
Initial number of X_1 species	120
Initial number of X_2 species	60
Initial number of X_3 species	0

Table S11. The rate constants for the different reactions in the Simple Reaction Model as well as the initial values of different species.

Rate Constants used in the simulation	Numerical values of the Rate Constants
Rate constant of R_1 (c_1)	10.0
Rate constant of R_2 (c_2)	0.1

Initial number of X_1 species	9
Initial number of X_2 species	20000
Initial number of X_3 species	0

Table S12. The rate constants for the different reactions in the First Order Reaction Model as well as the initial values of different species.

Rate Constants used in the simulation	Numerical values of the Rate Constants
Rate constant of R_1 (c_1)	2.0
Rate constant of R_2 (c_2)	1.0
Rate constant of R_3 (c_3)	2.0
Rate constant of R_4 (c_4)	1.0
Initial number of X_1 species	20000
Initial number of X_2 species	0
Initial number of X_3 species	0

Table S13. The rate constants for the different reactions in the Oregonator Reaction Model as well as the initial values of different species.

Rate Constants used in the simulation	Numerical values of the Rate Constants
Rate constant of R_1 (c_1)	2.0
Rate constant of R_2 (c_2)	0.1
Rate constant of R_3 (c_3)	104.0
Rate constant of R_4 (c_4)	0.016
Rate constant of R_5 (c_5)	26.0
Initial number of Y_1 species	500
Initial number of Y_2 species	1000
Initial number of Y_3 species	2000

(III) CODES:

(I) SSA

```
!       PROGRAM FOR CARLETTI-BURRAGE MODEL
!       USING STOCHASTIC SIMULATION ALGORITHM (SSA)
!       FORTRAN 95 COMPILER HAVE BEEN USED FOR COMPILATION
!       AUTHORS: SHANTANU KADAM AND KUMAR VANKA
!       PHYSICAL CHEMISTRY DIVISION, NATIONAL CHEMICAL LABORATORY
!       PUNE,MAHARASHTRA-411008, INDIA
        implicit none
        integer *4 i,steps,j,mu,k,tinc,cont,itr,nrun,run
        CHARACTER*30 crun
        CHARACTER*50 datfilename
        INTEGER fileunit,n_steps
        integer *4 k1,k2,k3,k4,N,exact,accel,toz
        integer *4 mspec,DNA,Dspec,RNA,DNA1,DNA2
        real *8 min,tau(10),treal,tmed,tmedp,tstep
        integer *4 x1,x2,x3,t,it,tmpo
        real *8 sr1,sr2,sr3,tp,r56,avg,tprint
        real *8 eps,z1,z2,ap,r1,r2,r3,ran3
        real *8 de1,de2,de3,de4,numer,ran
        real *8 d1,d2,d3,d4,a0,poidev,one,two,four
        real *8 c1,c2,c3,c4,tau1,tau2,tau3,tau4
        real *8 c5,c6,c7,c8,c9,c10,a5,a6,a7,a8,a9,a10
        real *8 three,r,s,a1,a2,a3,a4,taue,asum5
        integer *4 cou1,cou2,cou3,cou4,idum,ctop
        integer *4 cou5,cou6,cou7,cou8,cou9,cou10
        real *8 tin,tfi
        CHARACTER(LEN=20):: f102, num
!
        nrun = 500
!
!*******INPUT/OUTPUT FILES**********************************
!
open(897,file='cputime',form='formatted',access='sequential',status='unknown')
open(98,file='counter',form='formatted',access='sequential',status='unknown')
!
write(num,*) 3    !N_component
write(f102,*) '(e16.9,',trim(adjustl(num)),'I20)'
!
                call cpu_time(tin)
     do run=1,nrun
              n_steps = 0
              write(crun,*) run
              datfilename='./data/x1out.'//trim(adjustl(crun))
              fileunit = 1000 + run
              write(*,*) run,'fileunit=',fileunit

              open(unit=fileunit, file= trim(adjustl(datfilename)),status='replace')
!
                rewind 98
                rewind 897
          !
          !*******MAIN PROGRAM**********************
          !
                t = 0
                tstep = 0.25
                tp = 0.0
          !
                ctop = 0
                idum = 864321+run
          !
                steps = 0
                exact = 0
                treal = 0.0
                tprint = 0.0
          !
                mspec = 200
                DNA = 200
                Dspec = 600
                RNA = 00
                DNA1 = 00
                DNA2 =  00
          !
                c1 = 0.078
                c2 = 3.9E-3
                c3 = 7.0E-4
                c4 = 0.043
                c5 = 0.083
                c6 = 0.5
                c7 = 0.020
                c8 = 0.479
                c9 = 2.0E-4
                c10 = 8.765E-12
          !
                do 33 i = 1, 2000000
          !
          a1 = c1*RNA
          a2 = c2*DNA1
          a3 = c3*mspec
          a4 = c4*RNA
          a5 = c5*(mspec*(mspec-1))/2
          a6 = c6*Dspec
          a7 = c7*DNA*Dspec
          a8 = c8*DNA1
          a9 = c9*DNA1*Dspec
          a10 = c10*DNA2
!
      a0 = a1+a2+a3+a4+a5+a6+a7+a8+a9+a10
!
!     write(99,*)a1,a2,a3,a4,a5,a6,a7,a8,a9,a10
!     write(66,*) a0
      exact = exact + 1
!
!***********GENERATION OF THE RANDOM NUMBERS***************
      r1 = ran3(idum)
      taue = (1/a0)*log(1/r1)
!     write(20,*) taue
      r2 = ran3(idum)
      r3 = r2*a0
!     write(31,*) r1,r2
!
!************SELECTION OF A REACTION**********************
!
!     asum5 = a1+a2+a3+a4+a5
!
      if(a1.gt.r3)then
      mu = 1
      elseif(a1+a2.gt.r3)then
      mu = 2
      elseif(a1+a2+a3.gt.r3)then
      mu = 3
      elseif(a1+a2+a3+a4.gt.r3)then
      mu = 4
      elseif(a1+a2+a3+a4+a5.gt.r3)then
      mu = 5
      elseif(a1+a2+a3+a4+a5+a6.gt.r3)then
      mu = 6
      elseif(a1+a2+a3+a4+a5+a6+a7.gt.r3)then
      mu = 7
      elseif(a1+a2+a3+a4+a5+a6+a7+a8.gt.r3)then
      mu = 8
      elseif(a1+a2+a3+a4+a5+a6+a7+a8+a9.gt.r3)then
      mu = 9
      elseif(a0.gt.r3)then
      mu = 10
```

```
       endif
!
      treal = treal + taue
!
      if(mu.eq.1)then
      RNA = RNA - 1
      DNA1 = DNA1 + 1
      elseif(mu.eq.2)then
      RNA = RNA + 1
      DNA1 = DNA1 - 1
      elseif(mu.eq.3)then
      mspec = mspec - 1
      RNA = RNA + 1
      else if(mu.eq.4)then
      mspec = mspec + 1
      RNA = RNA - 1
      elseif(mu.eq.5)then
      mspec = mspec - 2
      Dspec = Dspec + 1
      elseif(mu.eq.6)then
      mspec = mspec + 2
      Dspec = Dspec - 1
      elseif(mu.eq.7)then
      DNA = DNA - 1
      Dspec = Dspec - 1
      DNA1 = DNA1 + 1
      elseif(mu.eq.8)then
      DNA = DNA + 1
      Dspec = Dspec + 1
      DNA1 = DNA1 - 1
      elseif(mu.eq.9)then
      Dspec = Dspec - 1
      DNA1 = DNA1 - 1
      DNA2 = DNA2 + 1
      elseif(mu.eq.10)then
      Dspec = Dspec + 1
      DNA1 = DNA1 + 1
      DNA2 = DNA2 - 1
      endif
!
!*********CHECK TO PRINT tprint*********
!
      if(treal.gt.tprint)then
      tmed = treal-tprint
      tmedp = tmed/tstep
      itr = int(tmedp)
!
      do 159 j = 0, itr
!
```

```
                if(tprint.le.100)then
                write(fileunit,trim(adjustl(f102)))tprint,DNA,DNA1,DNA2
                tprint = tprint + tstep
                n_steps = n_steps + 1
                else
                go to 303
                endif
            !
                 159    continue
            !
                else
            !
                go to 324
                endif
            !
324              steps = steps + 1
            !
                 33     continue
            303    write(88,*)'number of steps = ', steps
                write(98,*) exact, n_steps
            !
                close(fileunit)
       end do
!
CALL  compileStats(3,nrun,n_steps)
                 call cpu_time(tfi)
                 write(897,*) 'cputime',tfi-tin !CPU TIME
!
close(98)
close(897)
!
stop
end
!******MAIN PROGRAM ENDS***************************

!
!********************STATISTICS************************
!
SUBROUTINE compileStats(N_comp, N_run, n_steps)
!
IMPLICIT NONE
INTEGER N_run, n_steps, N_comp
INTEGER run, d, j, fileunit, k102, jjj, kk, jj
REAL, DIMENSION(n_steps):: tempt
DOUBLE PRECISION, DIMENSION(1:n_steps, N_comp) :: rtempc,concsum,concsumsq
REAL, DIMENSION(1:n_steps, N_comp)::stdconc, meanconc, variance
REAL, DIMENSION(1:n_steps, N_comp)::cvconc
INTEGER, DIMENSION(1:n_steps, N_comp) ::tempc
REAL tttt
```

```
!
CHARACTER(LEN=20):: f102, f151, f202, num
CHARACTER*25 momfilename, datfilename, crun
!
concsum(1:n_steps,1:N_comp) =0;
concsumsq(1:n_steps,1:N_comp) =0;
!
write(num,*) N_comp*3
write(f102,*) '(e16.9,',trim(adjustl(num)),'I20)'
!
do d=1,N_run  !****** run loop starts ******!
!
!     read data
      write(crun,*) d
!
      datfilename = './data/x1out.'//trim(adjustl(crun))
      fileunit=1000+2*d
!
      write(*,*) 'datfilename in getstats is ',datfilename
      open(fileunit, file=trim(adjustl(datfilename)))
      tempt(1:n_steps) =0.0d0
      tempc(1:n_steps,1:N_comp) =0
!
      write(*,*) 'tempc is'
!
      write(*,*) 'nsteps N_comp',n_steps,N_comp
      do j=1,n_steps
!
        read(fileunit,trim(adjustl(f102)),end=10) tempt(j),(tempc(j,k102),k102=1,N_comp)
        !write(*,*)tempt(j),(tempc(j,k102),k102=1,N_comp)
        rtempc(j,1:N_comp)=tempc(j,1:N_comp) ! converting integer conc to real to store large values
!
! for initial condition
         concsum(j,1:N_comp) = rtempc(j,1:N_comp) + concsum(j,1:N_comp)
!
          concsumsq(j, 1:N_comp)= ((rtempc(j,1:N_comp))**2) +&
   &          concsumsq(j, 1:N_comp)
!
      end do !j
 10     continue
      close(fileunit)

    end do
!
!
    meanconc(1:n_steps,1:N_comp) =0
    stdconc(1:n_steps,1:N_comp) =0
    cvconc(1:n_steps,1:N_comp) =0
!
```

```
    print *,'n_runs',N_run
    meanconc(1:n_steps,1:N_comp) = DBLE(concsum)/DBLE(N_run)
!

    do jjj=1,n_steps
      do kk = 1,N_comp
!
!
        stdconc(jjj,kk) = SQRT((DBLE(concsumsq(jjj,kk)) * DBLE(N_run)      &
   &          - DBLE(concsum(jjj,kk))**2) / DBLE(N_run *(N_run-1)))
!
!
        cvconc(jjj,kk) = stdconc(jjj,kk)/meanconc(jjj,kk)
      end do               !kk
    end do                 !jjj
!
     do jjj=1,n_steps
      do kk = 1,N_comp
          variance(jjj,kk) = ((DBLE(concsumsq(jjj,kk)) * DBLE(N_run)      &
       &          - DBLE(concsum(jjj,kk))**2) / DBLE(N_run *(N_run-1)))
      end do
     end do
!
    jj=0
!
    open(unit=71,file='finalstats', status='replace')
!
    write(71,*) ' STEP      TIME        MEAN (all components)        STDEV          VAR  CV'
!
!    write(num,*) N_comp*5
    write(num,*) N_comp*4
    write(f202,*) '(I10,e16.9,',trim(adjustl(num)),'e30.9)'
!
  !202 format(I10,e16.9,30e30.9)
    do jj=1,n_steps
    tttt=(jj-1)*tempt(2)
    write(71,trim(adjustl(f202))) jj,tttt,meanconc(jj,:),stdconc(jj,:),variance(jj,:),cvconc(jj,:)
    end do
!
    close(71)
!    stop
    write(*,*) 'all stats are in the file finalstats'
!
return
END SUBROUTINE compilestats
```

(II) GP

```
!     PROGRAM FOR CARLETTI-BURRAGE MODEL
!      USING GILLESPIE-PETZOLD ALGORITHM (GP)
!      FORTRAN 95 COMPILER HAVE BEEN USED FOR COMPILATION
!      AUTHORS: SHANTANU KADAM AND KUMAR VANKA
!      PHYSICAL CHEMISTRY DIVISION, NATIONAL CHEMICAL LABORATORY
!      PUNE,MAHARASHTRA-411008, INDIA
       implicit none
       integer *4 i,steps,j,mu,k,tinc,cont,itr,nrun,run
       CHARACTER*30 crun
       CHARACTER*50 datfilename
       INTEGER fileunit,n_steps
       integer *4 k1,k2,k3,k4,N,exact,accel,toz
       real *8 tau(10),treal,tmed,tmedp,tstep
       integer *4 x1,x2,x3,t,it,tmpo
       real *8 sr1,sr2,sr3,tp,r56,avg,tprint
       real *8 least,mu91,mu92,sgma9sq2,sgma9sq1
       real *8 exp1,exp2,exp3,exp4,exp5,exp6,exp7
       real *8 exp8,exp9,exp10,sgma9sq3
       integer *4 mspec,DNA,Dspec,RNA,DNA1,DNA2
       real *8 a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,asum5,asum10
       real *8 a11,a12,a13,a14,a15,a16,a17,a18,a0,a01,a02
       real *8 mu1,mu2,mu3,mu4,mu5,mu6,mu7,mu8,mu9,mu10
       real *8 sgma1sq,sgma2sq,sgma3sq,sgma4sq,sgma5sq,sgma6sq
       real *8 sgma7sq,sgma8sq,sgma9sq,sgma10sq
       real *8 arg1,arg2,arg3,arg4,arg5,arg6,arg7,arg8,arg9,arg10,arg11
       real *8 arf1,arf2,arf3,arf4,arf5,arf6,arf7,arf8,arf9,arf10,arf11
       real *8 c1,c2,c3,c4,c5,c6,c7,c8,c9,c10
       integer *4 n1,n2,n3,n4,n5,n6,n7,n8,n9,n10
       real *8 eps,z1,z2,ap,r1,r2,r3,sr4,sr5,sr6
       real *8 de1,de2,de3,de4,numer,ran3
       real *8 d1,d2,d3,d4,poidev,one,two,four
       real *8 tau1,tau2,tau3,tau4
       real *8 three,r,s,taue
       integer *4 idum,ctop
       integer *4 kit,kat,xf,NHP,irec,kit1,kat1
       real *8 tin,tfi,xx1(1000),xx2(1000),xx3(1000)
       CHARACTER(LEN=20):: f102, num
!
       nrun = 500
!
!*******INPUT/OUTPUT FILES********************************
!
open(897,file='cputime',form='formatted',access='sequential',status='unknown')
open(13,file='extra',form='formatted',access='sequential',status='unknown')
open(98,file='counter',form='formatted',access='sequential',status='unknown')
!
write(num,*) 3    !N_component
write(f102,*) '(e16.9,',trim(adjustl(num)),'I20)'
!
                    call cpu_time(tin)
```

```
    do run=1,nrun
            n_steps = 0
            write(crun,*) run
            datfilename='./data/x1out.'//trim(adjustl(crun))
            fileunit = 1000 + run
            write(*,*) run,'fileunit=',fileunit
!
            open(unit=fileunit, file= trim(adjustl(datfilename)),status='replace')
!
              rewind 13
              rewind 98
              rewind 897
          !
          !*******MAIN PROGRAM**********************
          !
              t = 0
              tstep = 0.25
              tp = 0.0
          !
              ctop = 0
              idum = 84321+run
          !
              steps = 0
              exact = 0
              accel = 0
              treal = 0.0
              tprint = 0.0
          !
              mspec = 200
              DNA = 200
              Dspec = 600
              RNA = 0
              DNA1 = 0
              DNA2 = 0
          !
              c1 = 0.078
              c2 = 3.9E-3
              c3 = 7.0E-4
              c4 = 0.043
              c5 = 0.083
              c6 = 0.5
              c7 = 0.020
              c8 = 0.479
              c9 = 2.0E-4
              c10 = 8.765E-12
          !
              do 33 i = 1,50000
          !
                  a1 = c1*RNA
```

```
               a2 = c2*DNA1
               a3 = c3*mspec
               a4 = c4*RNA
               a5 = c5*(mspec*(mspec-1))/2
               a6 = c6*Dspec
               a7 = c7*DNA*Dspec
               a8 = c8*DNA1
               a9 = c9*DNA1*Dspec
               a10 = c10*DNA2
        !
             a0 = a1 + a2 + a3 + a4 + a5 + a6 + a7 + a8 + a9 + a10
        !
            mu1 = c1*(-a1 + a2 + a3 - a4)
            mu2 =  c2*(a1 - a2 + a7 - a8 - a9 + a10)
            mu3 = c3*(-a3 + a4 - 2*a5 + 2*a6)
            mu4 = c4*(-a1 + a2 + a3 - a4)
            mu5 = (a4 - a3)*(c5*((2*mspec-1))/2) + (a6 - a5)*(c5*(2*mspec-1))
            mu6 = c6*(a5 - a6 - a7 + a8 - a9 + a10)
            mu7 = c7*DNA*(a5 - a6 - a7 + a8 - a9 + a10) + c7*Dspec*(-a7 + a8)
            mu8 = c8*(a1 - a2 + a7 - a8 - a9 + a10)
            mu91 = (c9*Dspec)*(a1 - a2) + (c9*DNA1)*(a5 - a6) + (c9*Dspec - c9*DNA1)*a7 +
(c9*DNA1 - c9*Dspec)*a8                              mu92 = (-c9*DNA1 -c9*Dspec)*a9 + (c9*DNA1 +
c9*Dspec)*a10
            mu9 = mu91 + mu92
            mu10 = c10*(a9 - a10)
        !
            eps = 0.03  ! ERROR CONTROL PARAMETER EPSILON
        !
           sgma1sq = (c1*c1)*(a1 + a2 + a3 + a4)
           sgma2sq = (c2*c2)*(a1 + a2 + a7 + a8 + a9 + a10)
           sgma3sq = (c3*c3)*(a3 + a4 + 4*a5 + 4*a6)
           sgma4sq = (c4*c4)*(a1 + a2 + a3 + a4)
           sgma5sq = (a3 + a4)*(((c5*(2*mspec-1))/2)**2) + (a5 + a6)*(((c5*(2*mspec-1)))**2)
           sgma6sq = (a5 + a6 + a7 + a8 + a9 + a10)*((c6)**2)
           sgma7sq = (a5 + a6 + a9 + a10)*((c7*DNA)**2) + (a7 + a8)*(((c7*DNA)**2) +
((c7*Dspec)**2) + (2*Dspec*DNA*c7*c7))
           sgma8sq = (a1 + a2 + a7 + a8 + a9 + a10)*((c8)**2)
           sgma9sq1 = ((c9*Dspec)**2)*(a1 + a2) + ((c9*DNA1)**2)*(a5 + a6)
           sgma9sq2 = (((c9*DNA1)**2) + ((c9*Dspec)**2) - (2*Dspec*DNA1*c9*c9))*(a7+a8)
           sgma9sq3 = (((c9*DNA1)**2) + ((c9*Dspec)**2) + (2*Dspec*DNA1*c9*c9))*(a9+a10)
           sgma9sq  = sgma9sq1 + sgma9sq2 + sgma9sq3
           sgma10sq = (a9 + a10)*(c10*c10)
        !
             arg1 = (eps*a0)/abs(mu1)
             arg2 = (eps*a0)/abs(mu2)
             arg3 = (eps*a0)/abs(mu3)
             arg4 = (eps*a0)/abs(mu4)
             arg5 = (eps*a0)/abs(mu5)
             arg6 = (eps*a0)/abs(mu6)
```

```
      arg7 = (eps*a0)/abs(mu7)
      arg8 = (eps*a0)/abs(mu8)
      arg9 = (eps*a0)/abs(mu9)
      arg10 = (eps*a0)/abs(mu10)
!
      numer = (eps**2.0)*(a0**2.0)
!
      arf1 = numer/sgma1sq
      arf2 = numer/sgma2sq
      arf3 = numer/sgma3sq
      arf4 = numer/sgma4sq
      arf5 = numer/sgma5sq
      arf6 = numer/sgma6sq
      arf7 = numer/sgma7sq
      arf8 = numer/sgma8sq
      arf9 = numer/sgma9sq
      arf10 = numer/sgma10sq
!
      tau(1) = min(arg1,arf1)
      tau(2) = min(arg2,arf2)
      tau(3) = min(arg3,arf3)
      tau(4) = min(arg4,arf4)
      tau(5) = min(arg5,arf5)
      tau(6) = min(arg6,arf6)
      tau(7) = min(arg7,arf7)
      tau(8) = min(arg8,arf8)
      tau(9) = min(arg9,arf9)
      tau(10) = min(arg10,arf10)
!
      least = tau(1)
      do 30 j = 2,10
      if(tau(j).lt.least)then
      least = tau(j)
      end if
30      continue
!
      ap = 2/a0
!
      if(least.lt.ap)then
      exact = exact + 1
!
!      write(56,*) 'USE SSA'
!***********GENERATION OF THE RANDOM NUMBERS************
      r1 = ran3(idum)
      taue = (1/a0)*log(1/r1)
!      write(93,*) taue
      r2 = ran3(idum)
      r3 = r2*a0
!
```

```
!************SELECTION OF A REACTION*******************
!
        asum5 = a1+a2+a3+a4+a5
        asum10 = a6+a7+a8+a9+a10
!
        if(a1.gt.r3)then
        mu = 1
        elseif(a1+a2.gt.r3)then
        mu = 2
        elseif(a1+a2+a3.gt.r3)then
        mu = 3
        elseif(a1+a2+a3+a4.gt.r3)then
        mu = 4
        elseif(asum5.gt.r3)then
        mu = 5
        elseif(asum5+a6.gt.r3)then
        mu = 6
        elseif(asum5+a6+a7.gt.r3)then
        mu = 7
        elseif(asum5+a6+a7+a8.gt.r3)then
        mu = 8
        elseif(asum5+a6+a7+a8+a9.gt.r3)then
        mu = 9
        elseif(asum5+asum10.gt.r3)then
        mu = 10
        endif
!
       treal = treal + taue
!
       if(mu.eq.1)then
       RNA = RNA - 1
       DNA1 = DNA1 + 1
       elseif(mu.eq.2)then
       RNA = RNA + 1
       DNA1 = DNA1 - 1
       elseif(mu.eq.3)then
       mspec = mspec - 1
       RNA = RNA + 1
       elseif(mu.eq.4)then
       mspec = mspec + 1
       RNA = RNA - 1
       elseif(mu.eq.5)then
       mspec = mspec - 2
       Dspec = Dspec + 1
       elseif(mu.eq.6)then
       mspec = mspec + 2
       Dspec = Dspec - 1
       elseif(mu.eq.7)then
       DNA = DNA - 1
```

```
      Dspec = Dspec - 1
      DNA1 = DNA1 + 1
      elseif(mu.eq.8)then
      DNA = DNA + 1
      Dspec = Dspec + 1
      DNA1 = DNA1 - 1
      elseif(mu.eq.9)then
      Dspec = Dspec - 1
      DNA1 = DNA1 - 1
      DNA2 = DNA2 + 1
      elseif(mu.eq.10)then
      Dspec = Dspec + 1
      DNA1 = DNA1 + 1
      endif
!
!*********CHECK TO PRINT tprint*********
!
      if(treal.gt.tprint)then
      tmed = treal-tprint
      tmedp = tmed/tstep
      itr = int(tmedp)
!     tprint = tprint + 1
!
      do 159 j = 0, itr
!     tprint = tprint + 1
      if(tprint.le.100)then
      write(fileunit,trim(adjustl(f102)))tprint,RNA,DNA1,DNA2
      tprint = tprint + tstep
      n_steps = n_steps + 1
      else
      go to 303
      endif
!
  159 continue
!     endif
!     write(234,*) tprint
      else
      go to 324
      endif
!
      else
!
      accel = accel + 1
!     write(156,*) 'USE GASA'
!     tprint = tprint + 1
      treal = treal + least
!
          exp1 = a1*least
          exp2 = a2*least
```

```
            exp3 = a3*least
            exp4 = a4*least
            exp5 = a5*least
            exp6 = a6*least
            exp7 = a7*least
            exp8 = a8*least
            exp9 = a9*least
            exp10 = a10*least
!
         n1 = poidev(exp1,idum)
         n2 = poidev(exp2,idum)
         n3 = poidev(exp3,idum)
         n4 = poidev(exp4,idum)
         n5 = poidev(exp5,idum)
         n6 = poidev(exp6,idum)
         n7 = poidev(exp7,idum)
         n8 = poidev(exp8,idum)
         n9 = poidev(exp9,idum)
         n10 = poidev(exp10,idum)
!        write(143,*) n1,n2,n3,n4,n5,n6,n7,n8,n9,n10
!
         RNA = RNA - n1 + n2 + n3 - n4
         DNA1 = DNA1 + n1 - n2 + n7 - n8 - n9 + n10
         mspec = mspec - n3 + n4 - 2*n5 + 2*n6
         Dspec =  Dspec + n5 - n6 - n7 + n8 - n9 + n10
         DNA = DNA - n7 + n8
         DNA2 = DNA2 + n9 - n10
!
!*********CHECK TO PRINT tprint*********
!
        if(treal.gt.tprint)then
        tmed = treal-tprint
        tmedp = tmed/tstep
        itr = int(tmedp)
!
        do 59 j = 0, itr
!       tprint = tprint + 1
        if(tprint.le.100)then
!       ctop = ctop  +  1
!       write(2134,*) ctop
        write(fileunit,trim(adjustl(f102)))tprint,RNA,DNA1,DNA2
!       tprint = tprint + 1
        tprint = tprint + tstep
!       write(291,*) tprint
        n_steps = n_steps + 1
        else
        go to 303
        endif
         59     continue
```

```
            !
                 else
            !     write(237,*) 'hi'
                 go to 324
                 endif
            !
                 endif
            !
324                steps = steps + 1
            !
                  33    continue
            !
            303    write(88,*)'number of steps = ', steps
                 write(98,*) exact, accel, n_steps
            !
                 close(fileunit)
      end do

CALL compileStats(3,nrun,n_steps)
                  call cpu_time(tfi)
                  write(897,*) 'cputime',tfi-tin !CPU TIME
!
close(98)
close(897)
!
stop
end
!******MAIN PROGRAM ENDS***************************
!
!********************STATISTICS************************
!
SUBROUTINE compileStats(N_comp, N_run, n_steps)
!
IMPLICIT NONE
INTEGER N_run, n_steps, N_comp
INTEGER run, d, j, fileunit, k102, jjj, kk, jj
REAL, DIMENSION(n_steps):: tempt
DOUBLE PRECISION, DIMENSION(1:n_steps, N_comp) :: rtempc,concsum,concsumsq
REAL, DIMENSION(1:n_steps, N_comp)::stdconc, meanconc, variance
REAL, DIMENSION(1:n_steps, N_comp)::cvconc
INTEGER, DIMENSION(1:n_steps, N_comp) ::tempc
REAL tttt
!
CHARACTER(LEN=20):: f102, f151, f202, num
CHARACTER*25 momfilename, datfilename, crun
!
concsum(1:n_steps,1:N_comp) =0;
concsumsq(1:n_steps,1:N_comp) =0;
!
```

```
write(num,*) N_comp*3
write(f102,*) '(e16.9,',trim(adjustl(num)),'I20)'
!
do d=1,N_run  !****** run loop starts ******!
!
!    read data
      write(crun,*) d
!
      datfilename = './data/x1out.'//trim(adjustl(crun))
      fileunit=1000+2*d
!
      write(*,*) 'datfilename in getstats is ',datfilename
      open(fileunit, file=trim(adjustl(datfilename)))
      tempt(1:n_steps) =0.0d0
      tempc(1:n_steps,1:N_comp) =0
!
      write(*,*) 'tempc is'
!
      write(*,*) 'nsteps N_comp',n_steps,N_comp
!read time data
!read conc data
      do j=1,n_steps
!
        read(fileunit,trim(adjustl(f102)),end=10) tempt(j),(tempc(j,k102),k102=1,N_comp)
        !write(*,*)tempt(j),(tempc(j,k102),k102=1,N_comp)
        rtempc(j,1:N_comp)=tempc(j,1:N_comp) ! converting integer conc to real to store large values
!
! for initial condition
       concsum(j,1:N_comp) = rtempc(j,1:N_comp) + concsum(j,1:N_comp)
!
        concsumsq(j, 1:N_comp)= ((rtempc(j,1:N_comp))**2) +&
   &        concsumsq(j, 1:N_comp)
!
      end do !j
 10    continue
      close(fileunit)

    end do
!
    meanconc(1:n_steps,1:N_comp) =0
    stdconc(1:n_steps,1:N_comp) =0
    cvconc(1:n_steps,1:N_comp) =0
!
    print *,'n_runs',N_run
    meanconc(1:n_steps,1:N_comp) = DBLE(concsum)/DBLE(N_run)
!
    do jjj=1,n_steps
      do kk = 1,N_comp
        stdconc(jjj,kk) = SQRT((DBLE(concsumsq(jjj,kk)) * DBLE(N_run)     &
    &          - DBLE(concsum(jjj,kk))**2) / DBLE(N_run *(N_run-1)))

!
          cvconc(jjj,kk) = stdconc(jjj,kk)/meanconc(jjj,kk)
        end do                !kk
     end do                   !jjj
!
     jj=0
!
     open(unit=71,file='finalstats', status='replace')
!
     write(71,*) ' STEP      TIME         MEAN (all components)         STDEV         CV'
!
!     write(num,*) N_comp*5
     write(num,*) N_comp*4
     write(f202,*) '(I10,e16.9,',trim(adjustl(num)),'e30.9)'
!
   !202 format(I10,e16.9,30e30.9)
     do jj=1,n_steps
     tttt=(jj-1)*tempt(2)
     write(71,trim(adjustl(f202))) jj,tttt,meanconc(jj,:),stdconc(jj,:),variance(jj,:),cvconc(jj,:)
     end do
!
     close(71)
!     stop
     write(*,*) 'all stats are in the file finalstats'
!
return
END SUBROUTINE compilestats
```

(III) BDTAU

```
!       PROGRAM FOR CARLETTI-BURRAGE MODEL
!       FORTRAN 95 COMPILER HAVE BEEN USED FOR COMPILATION
!       AUTHORS: SHANTANU KADAM AND KUMAR VANKA
!       PHYSICAL CHEMISTRY DIVISION, NATIONAL CHEMICAL LABORATORY
!       PUNE,MAHARASHTRA-411008, INDIA
        implicit none
        integer *4 i,steps,j,mu,tinc,cont,itr,nrun,run
        CHARACTER*30 crun
        CHARACTER*50 datfilename
        INTEGER fileunit,n_steps
        integer *4 N,exact,accel,toz
        real *8 tau,tmed,tmedp,tstep
        integer *4 it,tmpo,ip
        real *8 sr1,sr2,sr3,tp,r56,avg,tprint
        real *8 eps,z1,z2,ap,r1,r2,r3,ran3
        real *8 de1,de2,de3,de4,numer,ran
        real *8 b1,b2,a1,a2,a0,one,two,four
```

```
      real *8 c1,c2,c3,c4
      integer, dimension(20)::k1max,K
      integer, dimension(20,20)::nu
      integer, dimension(10)::X(10),X1twl(10)
      real *8 a(20),prob(20),bnldev
      real *8 c5,c6,c7,c8,c9,c10
      real *8 three,r,s,asum5
      integer *4 cou1,cou2,cou3,cou4,idum,ctop
      real *8 tin,tfi,ftr,treal
      CHARACTER(LEN=20):: f102, num
!
      nrun = 500
!
!*******INPUT/OUTPUT FILES*********************************
!
open(68,file='nuvalues',form='formatted',access='sequential',status='unknown')
open(76,file='nuwrite',form='formatted',access='sequential',status='unknown')
open(897,file='cputime',form='formatted',access='sequential',status='unknown')
open(98,file='counter',form='formatted',access='sequential',status='unknown')
open(88,file='steps',form='formatted',access='sequential',status='unknown')
open(11,file='species',form='formatted',access='sequential',status='unknown')
!
write(num,*) 3    !N_component
write(f102,*) '(e16.9,',trim(adjustl(num)),'I20)'
!
                  call cpu_time(tin)
      do run=1,nrun
              n_steps = 0
              write(crun,*) run
              datfilename='./data/x1out.'//trim(adjustl(crun))
              fileunit = 1000 + run
              write(*,*) run,'fileunit=',fileunit
!
              open(unit=fileunit, file= trim(adjustl(datfilename)),status='replace')
!
                rewind 98
                rewind 897
                rewind 68
                rewind 76
            !
            !*******MAIN PROGRAM**********************
            !
                ftr = 2.00
                tstep = 0.25
                tp = 0.0
            !
                ctop = 0
                idum = 6321 + run
            !
                steps = 0
                treal = 0.0
                tprint = 0.0
            !
                X(1) = 200
                X(2) = 200
                X(3) = 600
                X(4) = 000
                X(5) = 000
                X(6) = 000
            !     do i = 1, 6
            !     read(11,*) X(i)
            !     write(59,*) X(i)
            !     enddo
            !     close(11)
            !     close(59)
            !
                c1 = 0.078
                c2 = 3.9E-3
                c3 = 7.0E-4
                c4 = 0.043
                c5 = 0.083
                c6 = 0.5
                c7 = 0.020
                c8 = 0.479
                c9 = 2.0E-4
                c10 = 8.765E-12
            !
                do j = 1,10
                do i = 1,6
                read(68,*) nu(i,j)
                write(76,*) nu(i,j)
                enddo
                enddo
            !
                do 33 i = 1, 20000
            !
                     a(1) = c1*X(4)
                     a(2) = c2*X(5)
                     a(3) = c3*X(1)
                     a(4) = c4*X(4)
                     a(5) = c5*(X(1)*((X(1))-1))/2
                     a(6) = c6*X(3)
                     a(7) = c7*X(2)*X(3)
                     a(8) = c8*X(5)
                     a(9) = c9*X(5)*X(3)
                     a(10) = c10*X(6)
            !
            a0 = a(1)+a(2)+a(3)+a(4)+a(5)+a(6)+a(7)+a(8)+a(9)+a(10)
```

```
         !
         X1twl(1)  =  X(1)
         X1twl(2)  =  X(2)
         X1twl(3)  =  X(3)
         X1twl(4)  =  X(4)
         X1twl(5)  =  X(5)
         X1twl(6)  =  X(6)
         !
         if(a0.eq.0.0)then
         go to 324
         endif
         !
         tau = ftr/a0
         treal = treal + tau
         !
         do 111 j = 1,10
!
         k1max(1) = int(X1twl(4)/abs(nu(4,1)))
         k1max(2) = int(X1twl(5)/abs(nu(5,2)))
         k1max(3) = int(X1twl(1)/abs(nu(1,3)))
         k1max(4) = int(X1twl(4)/abs(nu(4,4)))
         k1max(5) = int(X1twl(1)/abs(nu(1,5)))
         k1max(6) = int(X1twl(3)/abs(nu(3,6)))
!
         a1 = int(X1twl(2)/abs(nu(2,7)))
         b1 = int(X1twl(3)/abs(nu(3,7)))
!
         k1max(7) = min(a1,b1)
         k1max(8) = int(X1twl(5)/abs(nu(5,8)))
!
         a2 = int(X1twl(3)/abs(nu(3,9)))
         b2 = int(X1twl(5)/abs(nu(5,9)))
!
         k1max(9) = min(a2,b2)
         k1max(10) = int(X1twl(6)/abs(nu(6,10)))
!
         do 112 ip = 1,6
!
         if(nu(ip,j).lt.0)then
!
         if((a(j).eq.0.0).and.(k1max(j).eq.0))then
         prob(j) = 0.0
         else
         prob(j) = (a(j)*tau)/k1max(j)
         endif
!
         if((prob(j).lt.1.0).and.(k1max(j).ge.0))then
!
         K(j) = bnldev(prob(j),k1max(j),idum)
```

```
          X1twl(ip) = X1twl(ip) + nu(ip,j)*K(j)
!
          elseif((prob(j).ge.1.0).and.(k1max(j).gt.0))then
          prob(j) = 1.0
!
          K(j) = bnldev(prob(j),k1max(j),idum)
          X1twl(ip) = X1twl(ip) + nu(ip,j)*K(j)
!
          endif
!
!         else
!         write(159,*) 'vuij is +ve'
!
          endif
112         continue
111         continue
!
          X(4) = X(4) - K(1) + K(2) + K(3) - K(4)
          X(5) = X(5) + K(1) - K(2) + K(7) - K(8) - K(9) + K(10)
          X(1) = X(1) - K(3) + K(4) - 2*K(5) + 2*K(6)
          X(3) = X(3) + K(5) - K(6) - K(7) + K(8) - K(9) + K(10)
          X(2) = X(2) - K(7) + K(8)
          X(6) = X(6) + K(9) - K(10)
!
          !
          !*********CHECK TO PRINT tprint*********
          !
               if(treal.gt.tprint)then
               tmed = treal-tprint
               tmedp = tmed/tstep
               itr = int(tmedp)
          !
               do 159 j = 0, itr
          !
               if(tprint.le.100)then
               write(fileunit,trim(adjustl(f102)))tprint,X(2),X(5),X(6)
               tprint = tprint + tstep
               n_steps = n_steps + 1
               else
               go to 303
               endif
          !
                159    continue
          !
               else
          !
              go to 324
              endif
          !
```

```
324                 steps = steps + 1
             !
                  33     continue
             303    write(88,*)'number of steps = ', steps
                  write(98,*) n_steps
             !
                  close(fileunit)
      end do
!            close(11)
!            close(59)
             close(68)
             close(76)
!
CALL compileStats(3,nrun,n_steps)
                  call cpu_time(tfi)
                  write(897,*) 'cputime',tfi-tin !CPU TIME
!
             close(11)
             close(59)
!            close(68)
!            close(76)
             close(98)
             close(897)
!
stop
end
!******MAIN PROGRAM ENDS***************************
!
!********************STATISTICS************************
!
SUBROUTINE compileStats(N_comp, N_run, n_steps)
!
IMPLICIT NONE
INTEGER N_run, n_steps, N_comp
INTEGER run, d, j, fileunit, k102, jjj, kk, jj
REAL, DIMENSION(n_steps):: tempt
DOUBLE PRECISION, DIMENSION(1:n_steps, N_comp) :: rtempc,concsum,concsumsq
REAL, DIMENSION(1:n_steps, N_comp)::stdconc, meanconc, variance
REAL, DIMENSION(1:n_steps, N_comp)::cvconc
INTEGER, DIMENSION(1:n_steps, N_comp) ::tempc
REAL tttt
!
CHARACTER(LEN=20):: f102, f151, f202, num
CHARACTER*25 momfilename, datfilename, crun
!
concsum(1:n_steps,1:N_comp) =0;
concsumsq(1:n_steps,1:N_comp) =0;
!
write(num,*) N_comp*3
write(f102,*) '(e16.9,',trim(adjustl(num)),'I20)'
!
do d=1,N_run  !****** run loop starts ******!
!
!     read data
        write(crun,*) d
!
        datfilename = './data/x1out.'//trim(adjustl(crun))
        fileunit=1000+2*d
!
        write(*,*) 'datfilename in getstats is ',datfilename
        open(fileunit, file=trim(adjustl(datfilename)))
        tempt(1:n_steps) =0.0d0
        tempc(1:n_steps,1:N_comp) =0
!
        write(*,*) 'tempc is'
!
        write(*,*) 'nsteps N_comp',n_steps,N_comp
!
        do j=1,n_steps
!
          read(fileunit,trim(adjustl(f102)),end=10) tempt(j),(tempc(j,k102),k102=1,N_comp)
          !write(*,*)tempt(j),(tempc(j,k102),k102=1,N_comp)
          rtempc(j,1:N_comp)=tempc(j,1:N_comp) ! converting integer conc to real to store large values
!
! for initial condition
         concsum(j,1:N_comp) = rtempc(j,1:N_comp) + concsum(j,1:N_comp)
!
          concsumsq(j, 1:N_comp)= ((rtempc(j,1:N_comp))**2) +&
     &          concsumsq(j, 1:N_comp)
!
        end do !j
 10      continue
        close(fileunit)

     end do
!
     meanconc(1:n_steps,1:N_comp) =0
     stdconc(1:n_steps,1:N_comp) =0
     cvconc(1:n_steps,1:N_comp) =0
!      write(*,*) meanconc(1,1),'here before stdev'
     print *,'n_runs',N_run
     meanconc(1:n_steps,1:N_comp) = DBLE(concsum)/DBLE(N_run)
!
     do jjj=1,n_steps
        do kk = 1,N_comp
!
!
          stdconc(jjj,kk) = SQRT((DBLE(concsumsq(jjj,kk)) * DBLE(N_run)      &
    &         - DBLE(concsum(jjj,kk))**2) / DBLE(N_run *(N_run-1)))
!
!
          cvconc(jjj,kk) = stdconc(jjj,kk)/meanconc(jjj,kk)
        end do              !kk
      end do                !jjj
!
       do jjj=1,n_steps
         do kk = 1,N_comp
              variance(jjj,kk) = ((DBLE(concsumsq(jjj,kk)) * DBLE(N_run)     &
          &         - DBLE(concsum(jjj,kk))**2) / DBLE(N_run *(N_run-1)))
         end do
       end do
!
      jj=0

      open(unit=71,file='finalstats', status='replace')
!
      write(71,*) ' STEP     TIME       MEAN (all components)       STDEV         VAR  CV'
!
!     write(num,*) N_comp*5
      write(num,*) N_comp*4
      write(f202,*) '(I10,e16.9,',trim(adjustl(num)),'e30.9)'
!
   !202 format(I10,e16.9,30e30.9)
      do jj=1,n_steps
      tttt=(jj-1)*tempt(2)
      write(71,trim(adjustl(f202))) jj,tttt,meanconc(jj,:),stdconc(jj,:),variance(jj,:),cvconc(jj,:)
      end do
!
      close(71)
!     stop
      write(*,*) 'all stats are in the file finalstats'
!
return
END SUBROUTINE compilestats
```

(IV) NOISE

```
!      FORTRAN 95 COMPILER HAVE BEEN USED FOR COMPILATION
!      AUTHORS: SHANTANU KADAM AND KUMAR VANKA
!      PHYSICAL CHEMISTRY DIVISION, NATIONAL CHEMICAL LABORATORY
!      PUNE,MAHARASHTRA-411008, INDIA
       implicit none
       integer *4 i,steps,j,mu,k,tinc,cont,itr,nrun,run
       CHARACTER*30 crun
       CHARACTER*50 datfilename
       integer *4 fileunit,n_steps
       integer *4 DNA,DNA1,DNA2,RNA,Dspec,mspec
```

```
      integer *4 DNA_b,DNA1_b,DNA2_b,RNA_b,Dspec_b,mspec_b
      real *8 exp1,exp2,exp3,exp4,tau_R1,Npm
      real *8 exp5,exp6,exp7,exp8,exp9,exp10
      real *8 exp1p,exp2p,exp3p,exp4p
      real *8 exp5p,exp6p,exp7p,exp8p,exp9p,exp10p
      real *8 sig1,sig2,sig3,sig4,sig5
      real *8 sig6,sig7,sig8,sig9,sig10
      integer *4 n1,n2,n3,n4,n5,n6,n7,n8,n9,n10
      integer *4 n1p,n2p,n3p,n4p,n5p,n6p,n7p,n8p,n9p,n10p
      real *8 treal,tmed,tmedp,tstep,theta
      integer *4 x1,x2,x3,t,it,tmpo
      real *8 tp,r56,avg,tprint,ftt,a0,a9,a10
      real *8 eps,z1,z2,ap,r1,r2,r3,k01,k02,delta_a0
      real *8 numer,ran3,tau_R,poidev,a5,a6,a7,a8
      real *8 c1,c2,c3,c4,N0_R
      real *8 three,r,s,a1,a2,a3,a4,taue,k0,x0
      integer *4 idum,ctop,noise,poisson
      real *8 tin,tfi,c5,c6,c7,c8,c9,c10
      CHARACTER(LEN=20):: f102, num
!
      nrun = 500
!*******INPUT/OUTPUT FILES*******************************
!
open(897,file='cputime',form='formatted',access='sequential',status='unknown')
open(88,file='tot_steps',form='formatted',access='sequential',status='unknown')
open(98,file='counter',form='formatted',access='sequential',status='unknown')
!
write(num,*) 3    !N_component
write(f102,*) '(e16.9,',trim(adjustl(num)),'I20)'
!
                  call cpu_time(tin)
      do run=1,nrun
              n_steps = 0
              write(crun,*) run
              datfilename='./data/x1out.'//trim(adjustl(crun))
              fileunit = 1000 + run
              write(*,*) run,'fileunit=',fileunit
!
              open(unit=fileunit, file= trim(adjustl(datfilename)),status='replace')
!
                rewind 88
                rewind 98
                rewind 897
           !
           !*******MAIN PROGRAM*********************
           !
                tstep = 0.25
                eps = 0.06
                tp = 0.0
```

```
!
      ctop = 0
      idum = 46321 + run
!
      steps = 0
      poisson = 0
      noise = 0
      treal = 0.0
      tprint = 0.0
!
      mspec = 200
      DNA = 200
      Dspec = 600
      RNA = 0
      DNA1 = 0
      DNA2 = 0
!
      c1 = 0.078
      c2 = 3.9E-3
      c3 = 7.0E-4
      c4 = 0.043
      c5 = 0.083
      c6 = 0.5
      c7 = 0.020
      c8 = 0.479
      c9 = 2.0E-4
      c10 = 8.765E-12
!
      do 33 i = 1,20000
!
      RNA_b = RNA
      DNA1_b = DNA1
      mspec_b = mspec
      Dspec_b = Dspec
      DNA_b = DNA
      DNA2_b = DNA2
!
      a1 = c1*RNA
      a2 = c2*DNA1
      a3 = c3*mspec
      a4 = c4*RNA
      a5 = c5*(mspec*(mspec-1))/2
      a6 = c6*Dspec
      a7 = c7*DNA*Dspec
      a8 = c8*DNA1
      a9 = c9*DNA1*Dspec
      a10 = c10*DNA2
!
      a0 = a1+a2+a3+a4+a5+a6+a7+a8+a9+a10
```

```
!
k01 = (a1/a0)*c1+(a2/a0)*c2+(a3/a0)*c3+(a4/a0)*c4+(a5/a0)*c5
k02 = (a6/a0)*c6+(a7/a0)*c7+(a8/a0)*c8+(a9/a0)*c9+(a10/a0)*c10
k0 = k01 + k02
!
x0 = (k0 + sqrt(k0*k0+8.0*a0*k0))/(2.0*k0)
!
N0_R = (16*eps*a0)/((2*x0-1)*k0)
!
tau_R = N0_R/a0
!
exp1 = a1*tau_R
exp2 = a2*tau_R
exp3 = a3*tau_R
exp4 = a4*tau_R
exp5 = a5*tau_R
exp6 = a6*tau_R
exp7 = a7*tau_R
exp8 = a8*tau_R
exp9 = a9*tau_R
exp10 = a10*tau_R
!
n1 = poidev(exp1,idum)
n2 = poidev(exp2,idum)
n3 = poidev(exp3,idum)
n4 = poidev(exp4,idum)
n5 = poidev(exp5,idum)
n6 = poidev(exp6,idum)
n7 = poidev(exp7,idum)
n8 = poidev(exp8,idum)
n9 = poidev(exp9,idum)
n10 = poidev(exp10,idum)
!
treal = treal + tau_R
!
RNA = RNA - n1 + n2 + n3 - n4
DNA1 = DNA1 + n1 - n2 + n7 - n8 - n9 + n10
mspec = mspec - n3 + n4 - 2*n5 + 2*n6
Dspec =  Dspec + n5 - n6 - n7 + n8 - n9 + n10
DNA = DNA - n7 + n8
DNA2 = DNA2 + n9 - n10
!
!*************************************************
!
if((DNA.lt.0).or.(DNA2.lt.0).or.(DNA1.lt.0).or.(Dspec.lt.0).or.(mspec.lt.0) &
.or.(RNA.lt.0))then
!
noise = noise + 1
!
```

```
RNA = RNA_b
DNA1 = DNA1_b
mspec = mspec_b
Dspec = Dspec_b
DNA = DNA_b
DNA2 = DNA2_b
!
sig1 = sqrt(exp1)
sig2 = sqrt(exp2)
sig3 = sqrt(exp3)
sig4 = sqrt(exp4)
sig5 = sqrt(exp5)
sig6 = sqrt(exp6)
sig7 = sqrt(exp7)
sig8 = sqrt(exp8)
sig9 = sqrt(exp9)
sig10 = sqrt(exp10)
!
exp1p = exp1 - sig1
exp2p = exp2 - sig2
exp3p = exp3 - sig3
exp4p = exp4 - sig4
exp5p = exp5 - sig5
exp6p = exp6 - sig6
exp7p = exp7 - sig7
exp8p = exp8 - sig8
exp9p = exp9 - sig9
exp10p = exp10 - sig10
!
n1 = poidev(exp1p,idum)
n2 = poidev(exp2p,idum)
n3 = poidev(exp3p,idum)
n4 = poidev(exp4p,idum)
n5 = poidev(exp5p,idum)
n6 = poidev(exp6p,idum)
n7 = poidev(exp7p,idum)
n8 = poidev(exp8p,idum)
n9 = poidev(exp9p,idum)
n10 = poidev(exp10p,idum)
!
!****NOISE****
!
RNA = RNA - n1 + n2 + n3 - n4
DNA1 = DNA1 + n1 - n2 + n7 - n8 - n9 + n10
mspec = mspec - n3 + n4 - 2*n5 + 2*n6
Dspec =  Dspec + n5 - n6 - n7 + n8 - n9 + n10
DNA = DNA - n7 + n8
DNA2 = DNA2 + n9 - n10
!
```

```
!*********CHECK TO PRINT tprint*********
!
      if(treal.gt.tprint)then
      tmed = treal-tprint
      tmedp = tmed/tstep
      itr = int(tmedp)
!
      do 1159 j = 0, itr
!
      if(tprint.le.100)then
      write(fileunit,trim(adjustl(f102)))tprint,Dspec,mspec,RNA
      tprint = tprint + tstep
      n_steps = n_steps + 1
      else
      go to 303
      endif
!
1159   continue
!
      else
      go to 324
      endif
!
else
!
poisson = poisson + 1
!
!*********CHECK TO PRINT tprint*********
!
      if(treal.gt.tprint)then
      tmed = treal-tprint
      tmedp = tmed/tstep
      itr = int(tmedp)
!
      do 159 j = 0, itr
!
      if(tprint.le.100)then
      write(fileunit,trim(adjustl(f102)))tprint,Dspec,mspec,RNA
      tprint = tprint + tstep
      n_steps = n_steps + 1
      else
      go to 303
      endif
!
159   continue
!
      else
      go to 324
      endif
```

```
                endif
            !
324                 steps = steps + 1
       33      continue
       303     write(88,*)'number of steps = ', steps
            write(98,*) poisson,noise,n_steps
            !
            close(fileunit)
            !
       end do
!
       CALL compileStats(3,nrun,n_steps)
                call cpu_time(tfi)
                write(897,*) 'cputime',tfi-tin !CPU TIME
!
close(88)
close(98)
close(897)
!
stop
end
!******MAIN PROGRAM ENDS***************************
!*****************************************************
SUBROUTINE compileStats(N_comp, N_run, n_steps)
!
IMPLICIT NONE
INTEGER N_run, n_steps, N_comp
INTEGER run, d, j, fileunit, k102, jjj, kk, jj
REAL, DIMENSION(n_steps):: tempt
DOUBLE PRECISION, DIMENSION(1:n_steps, N_comp) :: rtempc,concsum,concsumsq
REAL, DIMENSION(1:n_steps, N_comp)::stdconc, meanconc, variance
REAL, DIMENSION(1:n_steps, N_comp)::cvconc
INTEGER, DIMENSION(1:n_steps, N_comp) ::tempc
REAL tttt
!
CHARACTER(LEN=20):: f102, f151, f202, num
CHARACTER*25 momfilename, datfilename, crun
!
concsum(1:n_steps,1:N_comp) =0;
concsumsq(1:n_steps,1:N_comp) =0;
!
write(num,*) N_comp*3
write(f102,*) '(e16.9,',trim(adjustl(num)),'I20)'
!
!*********LOOP OVER THE TOTAL NUMBER OF RUNS***********
!
       do d=1,N_run
!
        write(crun,*) d
```

```
!
        datfilename = './data/x1out.'//trim(adjustl(crun))
        fileunit=1000+2*d
!
        write(*,*) 'datfilename in getstats is ',datfilename
        open(fileunit, file=trim(adjustl(datfilename)))
        tempt(1:n_steps) =0.0d0
        tempc(1:n_steps,1:N_comp) =0
!
        write(*,*) 'tempc is'
!
        write(*,*) 'nsteps N_comp',n_steps,N_comp
!
        do j=1,n_steps
!
          read(fileunit,trim(adjustl(f102)),end=10) tempt(j),(tempc(j,k102),k102=1,N_comp)
!
          rtempc(j,1:N_comp)=tempc(j,1:N_comp)
!
          concsum(j,1:N_comp) = rtempc(j,1:N_comp) + concsum(j,1:N_comp)
!
          concsumsq(j, 1:N_comp)= ((rtempc(j,1:N_comp))**2) +&
   &           concsumsq(j, 1:N_comp)
!
        end do
 10     continue
        close(fileunit)

      end do
!
!****************INITIALIZATION**************
!
      meanconc(1:n_steps,1:N_comp) = 0
      stdconc(1:n_steps,1:N_comp) = 0
      cvconc(1:n_steps,1:N_comp) = 0
!
      print *,'n_runs',N_run
      meanconc(1:n_steps,1:N_comp) = DBLE(concsum)/DBLE(N_run)
!
      do jjj=1,n_steps
        do kk = 1,N_comp
!
          stdconc(jjj,kk) = SQRT((DBLE(concsumsq(jjj,kk)) * DBLE(N_run)      &
   &          - DBLE(concsum(jjj,kk))**2) / DBLE(N_run *(N_run-1)))
!
!
          cvconc(jjj,kk) = stdconc(jjj,kk)/meanconc(jjj,kk)
        end do
      end do
```

```
!
       do jjj=1,n_steps
         do kk = 1,N_comp
                variance(jjj,kk) = ((DBLE(concsumsq(jjj,kk)) * DBLE(N_run)      &
           &          - DBLE(concsum(jjj,kk))**2) / DBLE(N_run *(N_run-1)))
         end do
       end do
!
      jj=0
!
     open(unit=71,file='finalstats', status='replace')
!
      write(71,*) ' STEP      TIME         MEAN (all components)         STDEV           VAR    CV'
!
      write(num,*) N_comp*4
      write(f202,*) '(I10,e16.9,',trim(adjustl(num)),'e30.9)'
!
      do jj=1,n_steps
      tttt=(jj-1)*tempt(2)
      write(71,trim(adjustl(f202))) jj,tttt,meanconc(jj,:),stdconc(jj,:),variance(jj,:),cvconc(jj,:)
      end do
!
      close(71)
!
return
END SUBROUTINE compilestat
```