

```

!*****
PROGRAM NLSis_par
!*****
!
!*** THIS PROGRAM GENERATES STATISTICS ON PULSE PARAMETERS
!*** AFFECTED BY RANDOM NOISE ADDED PERIODICALLY TO THE
!*** NONLINEAR SCHROEDINGER EQUATION. THE
!*** NOISE IS BIASED TO PUSH THE PARAMETERS TOWARDS
!*** LOW-PROBABILITY REGIONS OF CONFIGURATION SPACE. EACH
!*** OF THE npars DIFFERENT "BIASINGS" IS RUN WITH
!*** nruns SIMULATIONS, AND AT THE END OF THE PROGRAM THE
!*** RUN RESULTS ARE ASSEMBLED AND THE STATISTICS
!*** RECONSTRUCTED.
!
  INCLUDE '/usr/rels/mpich/include/mpif.h'
!
  DECLARATIONS
!
  CALL MPI_INIT(ierr)
!
  CALL MPI_COMM_RANK(MPI_COMM_WORLD,myrank,ierr)
  CALL MPI_COMM_SIZE(MPI_COMM_WORLD,size,ierr)
!
  IF (npars.GT.size) THEN
    WRITE(*,*)'Insufficient number of processors!'
    STOP
  END IF
!
  READ IN PARAMETERS, INITIALISATIONS
!
!*** CREATE PARALLEL PROCESSOR TOPOLOGY
!*** ONE COLOUR FOR EACH BIASING TARGET PARAMETER
!
  jpar = 0
  DO jn = 0,MOD(size,npars)-1
    zerokeys(jn) = jpar
    DO k = 0,size/npars
      colvec(jpar) = jn
      keyvec(jpar) = k
      jpar = jpar + 1
    END DO
  END DO
  DO jn = MOD(size,npars),npars-1
    zerokeys(jn) = jpar
    DO k = 0,size/npars-1
      colvec(jpar) = jn
      keyvec(jpar) = k
      jpar = jpar + 1
    END DO
  END DO
END DO

```

```

!
col = colvec(myrank)
key = keyvec(myrank)
!
!*** FORM ONE COMMUNICATOR PARCOMM FOR EACH BIAS PARAMETER
!
CALL MPI_COMM_SPLIT(MPI_COMM_WORLD,col,key,parcomm,ierr)
!
!*** WITHIN EACH COMMUNICATOR, ESTABLISH WHICH KEY WILL
!*** FILL WHICH PART OF THE ARRAY OF FINAL VALUES
!
CALL MPI_COMM_SIZE(parcomm,pcsize,ierr)
DO k = 1,pcsize-1
    jstart(k) = nruns/pcsize*(k-1) + 1
    jend(k) = nruns/pcsize*k
    recvcnts(k) = jend(k)-jstart(k)+1
END DO
jstart(pcsz) = nruns/pcsize*(pcsz-1) + 1
jend(pcsz) = nruns
recvcnts(pcsz) = jend(pcsz)-jstart(pcsz)+1
!
WRITE(*,*) 'my rank:',myrank
WRITE(*,*) 'my col, key:',col,key
!
!*** Also create a communicator consisting of the
"subroots"
!*** (processors with zero key in each parcomm)
!
CALL MPI_COMM_GROUP(MPI_COMM_WORLD,worldhdl,ierr)
CALL MPI_GROUP_INCL(worldhdl,npars, &
    zerokeys,zerokeygroup,ierr)
CALL MPI_COMM_CREATE(MPI_COMM_WORLD,zerokeygroup, &
    zerokeycomm,ierr)
!
ALLOCATE MEMORY FOR ARRAYS
!
FIGURE OUT THE BIASING AT ROOT AND BROADCAST IT
TO EVERYBODY ELSE
!
!*** NOW BROADCAST FROM ROOT TO OTHER PROCESSES
!
CALL MPI_BCAST(biasset,4*Namps*npars, &
    MPI_DOUBLE_PRECISION,0,MPI_COMM_WORLD,ierr)
!
!
!
OPEN UNIQUE FILE FOR EACH PROCESS TO CHART PROGRESS OF
PROCESS THROUGH RUNS
!
LOOP THROUGH RUNS, STORING DATA FROM SIMULATIONS AND

```

```

        WRITING PROGRESS TO STATUS FILE
!
!*** NOW GATHER INFO BACK TO LEAD OF PAR COMMUNICATOR, ...
!
    CALL MPI_GATHERV(is_stat,recvcounts(key+1), &
        & MPI_DOUBLE_PRECISION,is_stat, &
        & recvcounts(1:pcsize), &
        & jstart(1:pcsize)-1,MPI_DOUBLE_PRECISION,0, &
        & parcomm,ierr)
    CALL MPI_GATHERV(loglrs,recvcounts(key+1), &
        & MPI_DOUBLE_PRECISION,loglrs, &
        & recvcounts(1:pcsize), &
        & jstart(1:pcsize)-1,MPI_DOUBLE_PRECISION,0, &
        & parcomm,ierr)
    CALL MPI_GATHERV(parlogprobs,recvcounts(key+1), &
        & MPI_DOUBLE_PRECISION,parlogprobs, &
        & recvcounts(1:pcsize), &
        & jstart(1:pcsize)-1,MPI_DOUBLE_PRECISION,0, &
        & parcomm,ierr)
!
!*** .. THEN GATHER INFO BACK TO ROOT
!
    IF (key.EQ.0) THEN
        CALL MPI_GATHER(is_stat,nruns,MPI_DOUBLE_PRECISION, &
            & is_stat,nruns,MPI_DOUBLE_PRECISION,0, &
            & zerokeycomm,ierr)
        CALL MPI_GATHER(loglrs,nruns,MPI_DOUBLE_PRECISION, &
            & loglrs,nruns,MPI_DOUBLE_PRECISION,0, &
            & zerokeycomm,ierr)
        CALL MPI_GATHER
(parlogprobs,nruns,MPI_DOUBLE_PRECISION,          &
parlogprobs,nruns,MPI_DOUBLE_PRECISION,0, &
            & zerokeycomm,ierr)
    END IF
!
    DO FINAL PROCESSING AND OUTPUT FROM ROOT PROCESS, CLOSE
        OPEN FILES, ETC.
!
    CALL MPI_FINALIZE(ierr)
!
END PROGRAM NLSis_par
!

```