F77 Program 12

```
c       Fortran 77 program to do a simulation of the Weibull distribution
c       of the censored model. Realistic case Mark 2.
c       This program may be adapted for other forms of censored models as
c       well as the simplest censoring model.
c       ***************************************************************
c                By Derek Dhammaloka FDX3 - 21st Jan. 1991
c       ***************************************************************
c                Define the following variables
c
c                cdf is the cumulative density function of the probability
c                function and is between 0 and 1. The function urand will
c                generate the random numbers between 0 and 1. It has 1
c                parameter iy, the seed to initialise the generator.
c                cdfex is as cdf but for the exponential distribution
c                and has seed iy2.
c                t is the remission time in arbitary units
c                knew is the new value of kappa to be entered by the user
c                rho is the rate to be entered by the user
c                loop is used in loop counters
c                n is the no. of individuals to be entered by the user
c                d is the no. of uncensored individuals
c                cp is the censor time (recruitment) that follows an
c                exponential distribution with rate rhoex
c                rhoex is to be entered by the user
c                cl is the censor time to be obtained using the equation
c                cl=censorstart+((n-loop)/(n-1)) where censorstart is
c                the initial censor time to be entered by the user
c                c is the minimum of cp and cl
c                meanex is the mean of the exponential distribution
c                i is the indicator variable (1 if censored, 0 otherwise/
c                x is equal to t if t is less than C, C otherwise
c                swops is the no. of swops
c                sorts is the no. of sorts
c                temp is used in sorting data values
c                list assigns the values 1,n so that it can be used in
c                sorting 1 variable in ascending or descending order and
c                the other variables have the same values as the original
c                data, but is being sorted
c                psn is the no. of trials in view at a certain time
c                ple is the product limit estimator
c                prcensor is the probability of censoring
c                iy,iy2 seeds to be entered by the user
c       ***************************************************************
c                Find the product limit estimator
c       ***************************************************************
        integer i(5000),list(5000)
        real cdf(5000),cdfex(5000),t(5000),cl(5000),cp(5000),c(5000)
        real x(5000)
        real knew,censorstart,meanex
        real rho,rhoex,ple,prcensor
        integer loop,temp,swops,sorts,n,d,psn,iy,iy2
        ple=1
        swops=1
        sorts=0
```

```
c     ****************************************************************
c              Input the no. of individuals
c              Also the index (kappa) and the rate (rho)
c     ****************************************************************
      print*, How many individuals
      read*,n
c
c             Set the no. of failures to the no. of observations
c
      d=n
c
c             Set the no. of trials in view at time=0 to n+1
c
      psn=n+1
      print*, Seed for the Weibull distribution
      read*,iy
      print*, Enter the index parameter
      read*,knew
      print*, Enter the rate parameter
      read*,rho
      print*
      print*, For the censor times
      print*, Enter the initial censor time
      print*
      read*,censorstart
      print*, For the censor times exponentially distributed
      print*, Enter seed
      read*,iy2
      print*
      print*, Enter rate
      read*,rhoex
      meanex=1/rhoex
c     ****************************************************************
c              Simulate the Weibull distribution
c              using the two parameters to obtain the remission times.
c              Actual censor times are obtained by taking the minimum of
c              the recruitment times obtained by using the exponential
c              distribution and the recruitment times obtained by using
c              a straight line.
c     ****************************************************************
      do 20 loop=1,n
              cdf(loop)=urand(iy)
              cdfex(loop)=urand(iy2)
              t(loop)=(-log(1-cdf(loop)))/(rho**knew)**(1/knew)
              cp(loop)=(-log(1-cdfex(loop)))/(rhoex)
              cl(loop)=(censorstart*(n-1))+(n-loop)
              cl(loop)=(cl(loop))/(n-1)
c
c             Find the actual censor time by taking the minimum of the
c             recruitment times obtained using a Poisson process and
c             the straight line
c
              if(cp(loop).lt.cl(loop)) c(loop)=cp(loop)
              if(cp(loop).ge.cl(loop)) c(loop)=cl(loop)
c
c             Decide whether the individual is to be censored
```

```
c               using the censoring rules
c
                if(t(loop).ge.c(loop)) then
                                       i(loop)=1
                                       d=d-1
                                       x(loop)=c(loop)
                endif
                if(t(loop).lt.c(loop)) then
                                       i(loop)=0
                                       x(loop)=t(loop)
                endif
                list(loop)=loop
20      continue
c       ***************************************************************
c               Print the headings
c       ***************************************************************
        print*
        print*,'Realistic case model II'
        print*
        print*,'Simulation of the Weibull distribution with '
        print*,'Index = ',knew,' and rate = ',rho
        print*
        print*,'Initial censor time = ',censorstart
        print*
        print*,'Recruitment times exponentially distributed with'
        print*,'Rate = ',rhoex,' ie Mean = ',meanex
        print*
        write(*,25)
25      format(t3,'T',t12,'CL',t20,'CP',t30,'C',t40,'I',t55,'x')
c       ***************************************************************
c               Output the remission and censor times
c               Also the indicator variable
c       ***************************************************************
        do 30 loop=1,n
                write(*,40)t(loop),cl(loop),cp(loop),c(loop),i(loop),x(loop)
40              format(f7.3,t8,f7.3,t16,f7.3,t26,t38,i7.0,t50,f7.3,t68)
30      continue
        print*
c
c               Sort the x values in ascending order
c
50      if(swops.ne.0.and.sorts.lt.n-1) then
          swops=0
          sorts=sorts+1
          do 60 loop=1,n-sorts
                if(x(list(loop)).gt.x(list(loop+1))) then
                  temp=list(loop)
                  list(loop)=list(loop+1)
                  list(loop+1)=temp
                  swops=swops+1
                  endif
60        continue
          goto 50
        endif
c
c               Print headings
```

```
c
      print*
      print*,'Calculation of the product limit estimator'
      print*,'(No. of failures assumed to be 1 for uncensored obs.)'
      print*
      write(*,63)
63    format(t3,'atom',t14,'r',t27,'(1-(1/r))',t42,'ple')
c
c           Calculate the product limit estimator using a do loop
c           Assume that the no. of failures for each x value is 1
c           providing that the individual is not censored since
c           the times are on a continuous scale and it is rare
c           for ties to occur.
c
      do 65 loop=1,n
            psn=psn-1
c
c           Only print the atom, position, censor probability and
c           the product limit if there is no censoring in the
c           individual.
c
            if(i(list(loop)).eq.0) then
                  prcensor=psn-1
                  prcensor=prcensor/psn
                  ple=ple*(prcensor)
                  write(*,75)x(i(list(loop))),psn,prcensor,ple
75                format(f7.3,t8,i7.0,t25,f7.4,t40,f7.4,t55)
                  endif
65    continue
c
c           Print the final product limit estimator
c
            print*
            print*,'Product limit estimator = ',ple
      stop
      end

      real function urand(iy)
      integer iy
c     ****************************************************************
c           Urand is a uniform random number generator based on
c           theory and suggestions given by KNUTH (1969). The
c           integer iy should be initialised to an arbitary integer
c           prior to the first call to urand. The calling program
c           should not alter the value of iy between subsequent
c           calls to urand. Values of urand will be returned in the
c           interval (0,1).
c     ****************************************************************
c           Reference - Problem solving with Fortran 77
c                       Brian D.Hahn 1987
c     ****************************************************************
      integer ia,ic,itwo,m2,m,mic
      double precision halfm
      real s
      data m2/0/,itwo/2/
```

```
c       If first entry, compute machine integer word length
        if(m2.eq.0) then
                    m=1
10      if(m.gt.m2) then
                    m2=m
                    m=itwo*m2
                    goto 10
        endif
        halfm=m2

c       Compute multiplier and increment for linear congruential method
        ia=8*int(halfm*atan(1.d0)/8.d0)+5
        ic=2*int(halfm*(0.5d0-sqrt(3.d0)/6.d0))+1
        mic=(m2-ic)+m2

c       s is the scale factor for converting to floating point
        s=0.5/halfm
        endif

c       Compute next random number
        iy=iy*ia

c       The following statement is for computers which do not allow
c       integer overflow on addition
        if(iy.gt.mic) iy=(iy-m2)-m2
        iy=iy+ic

c       The following statement is for computers where the word length
c       is greater than for multiplication
        if(iy/2.gt.m2) iy=(iy-m2)-m2

c       The following statement is for computers where integer overflow
c       affects sign bit
        if(iy.lt.0) iy=(iy+m2)+m2
        urand=float(iy)*s
        return
        end
```

Output from F77 Program 12

```
How many individuals
75
Seed for the Weibull distribution
2
Enter the index parameter
2.5
Enter the rate parameter
1.5

For the censor times
Enter the initial censor time

0.5
For the recruitment times exponentially distributed
Enter seed
-2

Enter rate
0.8

Realistic case model II

Simulation of the Weibull distribution with
Index =      .2500000E+01 and rate =      .1500000E+01

Initial censor time =      .5000000E+00

Recruitment times exponentially distributed with
Rate =     .8000000E+00 ie Mean =       .1250000E+01
```

T	CL	CF	C	I	x
1.339	1.500	.694	.694	1	.694
.412	1.486	.363	.363	1	.363
.461	1.473	1.393	1.393		.461
.670	1.459	.031	.031	1	.031
.509	1.446	3.358	1.446		.509
.365	1.432	.074	.074	1	.074
.329	1.419	.976	.976		.329
.706	1.405	1.452	1.405		.706
.628	1.392	.247	.247	1	.247
.415	1.378	2.588	1.378		.415
.495	1.365	1.702	1.365		.495
.571	1.351	2.230	1.351		.571
.233	1.338	.604	.604		.233
.547	1.324	3.532	1.324		.547
.636	1.311	.866	.866		.636
1.207	1.297	.808	.808	1	.808
.612	1.284	2.989	1.284		.612
1.043	1.270	.817	.817	1	.817
.665	1.257	.184	.184	1	.184
.679	1.243	1.334	1.243		.679
.244	1.230	.963	.963		.244
.524	1.216	.733	.733		.524
.674	1.203	1.933	1.203		.674

.531	1.189	.268	.268	1	.268
.852	1.176	2.666	1.176		.852
1.147	1.162	1.161	1.161		1.147
.946	1.149	.317	.317	1	.317
.882	1.135	.130	.130	1	.130
.494	1.122	.424	.424	1	.424
.742	1.108	.765	.765		.742
.249	1.095	3.173	1.095		.249
.172	1.081	2.293	1.081		.172
.552	1.068	.097	.097	1	.097
.489	1.054	2.830	1.054		.489
.903	1.041	3.302	1.041		.903
.689	1.027	.240	.240	1	.240
.673	1.014	.740	.740		.673
.830	1.000	.191	.191	1	.191
.878	.986	1.364	.986		.878
.194	.973	1.921	.973		.194
.663	.959	3.394	.959		.663
.488	.946	2.736	.946		.488
.476	.932	.289	.289	1	.289
1.312	.919	.699	.699	1	.699
.598	.905	1.470	.905		.598
.397	.892	.177	.177	1	.177
.287	.878	.266	.266	1	.266
.630	.865	2.098	.865		.630
.341	.851	.899	.851		.341
.328	.838	1.336	.838		.328
.964	.824	2.879	.824	1	.824
.831	.811	3.392	.811	1	.811
.467	.797	2.774	.797		.467
.770	.784	1.271	.784		.770
.569	.770	1.254	.770		.569
.285	.757	.086	.086	1	.086
.502	.743	5.082	.743		.502
.533	.730	.531	.531	1	.531
.184	.716	4.342	.716		.184
.599	.703	.224	.224	1	.224
1.101	.689	2.042	.689	1	.689
.537	.676	.644	.644		.537
.939	.662	6.154	.662	1	.662
.847	.649	2.363	.649	1	.649
.390	.635	.921	.635		.390
.926	.622	1.058	.622	1	.622
.527	.608	.127	.127	1	.127
.413	.595	1.675	.595		.413
.491	.581	.211	.211	1	.211
.580	.568	.981	.568	1	.568
.669	.554	.725	.554	1	.554
.984	.541	.465	.465	1	.465
.900	.527	1.060	.527	1	.527
.616	.514	1.482	.514	1	.514
.865	.500	1.382	.500	1	.500

Calculation of the product limit estimator
(No. of failures assumed to be 1 for uncensored obs.)

atom	r	(1-(1/r))	ple
.172	69	.9855	.9855
.184	66	.9848	.9706
.194	64	.9844	.9554
.233	61	.9836	.9397
.244	59	.9831	.9238
.249	57	.9825	.9076
.328	52	.9808	.8902
.329	51	.9804	.8727
.341	50	.9800	.8553
.390	48	.9792	.8374
.413	47	.9787	.8196
.415	46	.9783	.8018
.461	44	.9773	.7836
.467	42	.9762	.7649
.488	41	.9756	.7463
.489	40	.9750	.7276
.495	39	.9744	.7089
.502	37	.9730	.6898
.509	36	.9722	.6706
.524	34	.9706	.6509
.537	31	.9677	.6299
.547	30	.9667	.6089
.569	27	.9630	.5864
.571	26	.9615	.5638
.598	25	.9600	.5413
.612	24	.9583	.5187
.630	22	.9545	.4951
.636	21	.9524	.4715
.663	18	.9444	.4453
.673	17	.9412	.4192
.674	16	.9375	.3930
.679	15	.9333	.3668
.706	11	.9091	.3334
.742	10	.9000	.3001
.770	9	.8889	.2667
.852	4	.7500	.2000
.878	3	.6667	.1334
.903	2	.5000	.0667
1.147	1	.0000	.0000

Product limit estimator = .0000000E+00