## F77 Program 2

```
c       Fortran 77 program to do a simulation of the Weibull distribution
c       of the simplest censoring model -
c                                    x =t if t < Constant
c                                    x =constant if t >= Constant
c       (where Constant=Mean of something, etc)
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 - 6th Mar. 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               t is the remission time in arbitary units
c               kappa is the index 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               c is the constant censor time
c               const is the minimum time for censoring to occur. It is
c               to be entered by the user.
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               surf is the survivor function
c               hazard is the hazard function
c               pdf is the probability density function
c               ihazard is the integrated hazard
c               iy is the seed to be entered by the user
c       ***************************************************************
c               Obtain the survivor, hazard and probability density
c               functions. Also the integrated hazard.
c       ***************************************************************
        integer i(5000)
        real cdf(5000),t(5000),c(5000),x(5000)
        real kappa,const,rho,surf,hazard,pdf,ihazard
        integer loop,n,d,iy
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
        print*, Seed for the Weibull distribution
        read*,iy
        print*, Enter the index parameter
        read*,kappa
```

```
      print*, 'Enter the rate parameter'
      read*,rho
      print*, 'Enter the minimum time for censoring to occur'
      read*,const
c     ****************************************************************
c           Simulate the Weibull distribution
c           using the two parameters to obtain the remission times.
c           However, the censor times are constant.
c     ****************************************************************
c           Print the headings
c     ****************************************************************
      print*
      print*, 'For the simplest censored model'
      print*
      print*, 'Simulation of the Weibull distribution with '
      print*,'Index = ',kappa,' and rate =  ',rho
      print*
      print*, 'Minimum time for censoring to occur =  ',const
      print*
      write(*,25)
25    format(t3,'time',t17,'I',t25,'sf',t32,'h',t44,'f',t55,'H')
      do 20 loop=1,n
            cdf(loop)=urand(iy)
            surf=1-cdf(loop)
            t(loop)=(-log(surf)/(rho**kappa))**(1/kappa)
            c(loop)=const
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
c
c           Recalculate the survivor function
c
            surf=exp(-(rho*x(loop))**kappa)
c
            hazard=(kappa*rho)*((rho*x(loop))**(kappa-1))
            pdf=surf*hazard
            ihazard=((rho*x(loop))**(kappa))
c     ****************************************************************
c           Output the survivor, hazard and density functions as
c           well as the integrated hazard, together with its time
c           and indicator variable
c     ****************************************************************
         write(*,40)x(loop),i(loop),surf,hazard,pdf,ihazard
40       format(f7.3,t12,i7.0,t20,f7.4,t30,f7.4,t40,f7.4,t50,f7.4,t60)
```

```
20    continue
      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
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 2

```
How many individuals
75
Seed for the Weibull distribution
2
Enter the index parameter
2.5
Enter the rate parameter
1.5
Enter the minimum time for censoring to occur
1.25

For the simplest censored model

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

Minimum time for censoring to occur =      .1250000E+01
```

| time | I | sf | h | f | H |
|---|---|---|---|---|---|
| 1.250 | 1 | .0081 | 9.6279 | .0781 | 4.8140 |
| .412 | | .7401 | 1.8244 | 1.3503 | .3010 |
| .461 | | .6714 | 2.1587 | 1.4494 | .3983 |
| .670 | | .3639 | 3.7744 | 1.3735 | 1.0109 |
| .509 | | .6011 | 2.5007 | 1.5032 | .5090 |
| .365 | | .8006 | 1.5214 | 1.2181 | .2224 |
| .329 | | .8425 | 1.3014 | 1.0964 | .1714 |
| .706 | | .3156 | 4.0849 | 1.2893 | 1.1532 |
| .628 | | .4229 | 3.4273 | 1.4492 | .8607 |
| .415 | | .7372 | 1.8389 | 1.3556 | .3049 |
| .495 | | .6220 | 2.3987 | 1.4919 | .4749 |
| .571 | | .5073 | 2.9719 | 1.5076 | .6787 |
| .233 | | .9305 | .7738 | .7200 | .0721 |
| .547 | | .5437 | 2.7858 | 1.5147 | .6093 |
| .636 | | .4114 | 3.4924 | 1.4368 | .8882 |
| 1.207 | | .0121 | 9.1381 | .1108 | 4.4128 |
| .612 | | .4461 | 3.2978 | 1.4711 | .8073 |
| 1.043 | | .0468 | 7.3388 | .3436 | 3.0612 |
| .665 | | .3701 | 3.7367 | 1.3828 | .9941 |
| .679 | | .3510 | 3.8550 | 1.3529 | 1.0471 |
| .244 | | .9219 | .8318 | .7669 | .0813 |
| .524 | | .5776 | 2.6165 | 1.5113 | .5489 |
| .674 | | .3578 | 3.8122 | 1.3640 | 1.0278 |
| .531 | | .5675 | 2.6667 | 1.5133 | .5665 |
| .852 | | .1582 | 5.4136 | .8564 | 1.8440 |
| 1.147 | | .0205 | 8.4663 | .1739 | 3.8854 |
| .946 | | .0911 | 6.3344 | .5770 | 2.3959 |
| .882 | | .1337 | 5.7046 | .7627 | 2.0121 |
| .494 | | .6235 | 2.3913 | 1.4909 | .4724 |
| .742 | | .2708 | 4.4019 | 1.1922 | 1.3062 |
| .249 | | .9180 | .8581 | .7877 | .0856 |
| .172 | | .9666 | .4926 | .4761 | .0339 |
| .552 | | .5358 | 2.8257 | 1.5141 | .6240 |
| .489 | | .6309 | 2.3551 | 1.4859 | .4606 |
| .903 | | .1184 | 5.9093 | .6995 | 2.1339 |

| | | | | |
|---|---|---|---|---|
| .689 | .3376 | 3.9401 | 1.3301 | 1.0859 |
| .673 | .3592 | 3.8037 | 1.3661 | 1.0240 |
| .830 | .1777 | 5.2060 | .9251 | 1.7277 |
| .878 | .1364 | 5.6707 | .7734 | 1.9922 |
| .194 | .9554 | .5886 | .5623 | .0457 |
| .663 | .3725 | 3.7219 | 1.3864 | .9875 |
| .488 | .6320 | 2.3501 | 1.4851 | .4589 |
| .476 | .6500 | 2.2624 | 1.4706 | .4307 |
| 1.250 | 1 .0081 | 9.6279 | .0781 | 4.8140 |
| .598 | .4673 | 3.1829 | 1.4872 | .7609 |
| .397 | .7607 | 1.7227 | 1.3105 | .2735 |
| .287 | .8854 | 1.0600 | .9385 | .1217 |
| .630 | .4192 | 3.4479 | 1.4454 | .8694 |
| .341 | .8296 | 1.3705 | 1.1370 | .1868 |
| .328 | .8442 | 1.2922 | 1.0909 | .1694 |
| .964 | .0811 | 6.5172 | .5285 | 2.5121 |
| .831 | .1769 | 5.2145 | .9223 | 1.7324 |
| .467 | .6636 | 2.1964 | 1.4576 | .4100 |
| .770 | .2384 | 4.6547 | 1.1099 | 1.4336 |
| .569 | .5095 | 2.9605 | 1.5083 | .6744 |
| .285 | .8875 | 1.0476 | .9297 | .1194 |
| .502 | .6121 | 2.4471 | 1.4977 | .4909 |
| .533 | .5649 | 2.6798 | 1.5137 | .5712 |
| .184 | .9605 | .5457 | .5241 | .0403 |
| .599 | .4655 | 3.1925 | 1.4860 | .7647 |
| 1.101 | .0300 | 7.9589 | .2391 | 3.5051 |
| .537 | .5580 | 2.7138 | 1.5144 | .5833 |
| .939 | .0951 | 6.2661 | .5958 | 2.3529 |
| .847 | .1619 | 5.3724 | .8699 | 1.8207 |
| .390 | .7692 | 1.6804 | 1.2926 | .2624 |
| .726 | .1030 | 6.1378 | .6321 | 2.2732 |
| .527 | .5733 | 2.6377 | 1.5122 | .5563 |
| .413 | .7394 | 1.8278 | 1.3515 | .3019 |
| .491 | .6273 | 2.3728 | 1.4884 | .4663 |
| .580 | .4930 | 3.0463 | 1.5018 | .7072 |
| .669 | .3649 | 3.7681 | 1.3751 | 1.0080 |
| .984 | .0709 | 6.7245 | .4766 | 2.6468 |
| .900 | .1204 | 5.8804 | .7083 | 2.1166 |
| .616 | .4396 | 3.3335 | 1.4655 | .8218 |
| .865 | .1473 | 5.5381 | .8158 | 1.9152 |