APPENDIX - I (A)

C-PROGRAM FOR SECTION [3.2.2]

SINGLE EXPONENTIAL DISTRIBUTIOIN

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <time.h>
#include <conio.h>
#include <ctype.h>
void main()
{ float theta,lambda,a,r=0.5,rp,ra,raa;
  double  x,y,z,min,m1,m2,m3,an1,an2,an3,w,si,s=0.0,s1=0.0,s2=0.0,s3=0.0,
  r1=0.0,r2=0.0,r3=0.0,ar1,sr1=0.0,sdn1,sdr1,ar2,sr2=0.0,sdn2,sdr2,ar3,
  sr3=0.0,sdn3,sdr3,b1,b2,b3;
  int no,i,j=1,k,l=1,t,c=1;
  long  n1=0,sn1=0,n2=0,sn2=0,n3=0,sn3=0;
  FILE *fp;
  clrscr();
  printf("\n Enter the location and scale parameters :-");
  scanf("%f",&theta);
  scanf("%f",&lambda);
  printf("\n Enter the shape parameter loss function :-");
  scanf("%f",&a);
  fp=fopen("dsee2.dta","w+");
  randomize();
  for(no=20;no<=80;no+=5)
  {
    w=4.0/(pow(2*no/(lambda*a)-1.0,2)-1.0);
    for(j=1;j<=1000;j++)
    {
      s=0.0;
      x=theta-lambda*log(1.0-(float) random(32000)/32000);min=x;s+=x;
      for(l=1;l<8;l++)
       {  y=theta-lambda*log(1.0-(float) random(32000)/32000);
        if(y<min) min=y;s+=y;
       }
/* Purely Sequential Plan */
      s1=s;i=8;m1=min;
      do{
      i+=1;
      z=theta-lambda*log(1.0-(float) random(32000)/32000);
      if(z<m1) m1=z;
      s1+=z;
       }while(s1>(double) i*(i-1)*lambda/no+i*m1);
      n1+=i;sn1+=i*i;b1=1.0/(1.0-a*lambda/i)-a*lambda/i-1.0;r1+=b1;
      sr1+=b1*b1;
```

```
 /* Accelerated Sequential Plan */
      s2=s;i=8;m2=min;
      do{
      i+=1;
      y=theta-lambda*log(1.0 - (float) random(32000)/32000);
      if(y<m2) m2=y;
      s2+=y;
       }while(s2>(double) i*(i-1)*lambda/(no*r)+i*m2);
      si=i*(s2-i*m2)/pow(i-1,2);t= ((int) no*si/lambda)+1;
      if(i>t) k=i; else k=t;
      n2+=k;sn2+=k*k;b2=1.0/(1.0-a*lambda/k)-a*lambda/k-1.0;r2+=b2;
      sr2+=b2*b2;

 /* Alternative Accelerated Sequential Plan*/
      s3=s;i=8;m3=min;
      do{
      i+=1;
      y=theta - lambda*log(1.0 - (float) random(32000)/32000);
      if(y<m3) m3=y;
      s3+=y;
       }while(s3>(double) i*i*(i-1)*lambda/(no*r*(i+c))+i*m3);
      i=(int) i/r;
      n3+=i;sn3+=i*i;b3=1.0/(1.0-a*lambda/i)-a*lambda/i-1.0;r3+=b3;
      sr3+=b3*b3;
    }
   an1=n1/1000.0;ar1=r1/1000.0;sdn1=pow((sn1-1000*an1*an1)/999000,0.5);
   sdr1=pow((sr1-1000*ar1*ar1)/999000,0.5);
   an2=n2/1000.0;ar2=r2/1000.0;sdn2=pow((sn2-1000*an2*an2)/999000,0.5);
   sdr2=pow((sr2-1000*ar2*ar2)/999000,0.5);
   an3=n3/1000.0;ar3=r3/1000.0;sdn3=pow((sn3-1000*an3*an3)/999000,0.5);
   sdr3=pow((sr3-1000*ar3*ar3)/999000,0.5);
   rp=w+(3.5928)*pow(w,1.5)/(lambda* fabs (a));
   ra=w+(3+1/r)*pow(w,1.5)/(lambda* fabs (a));
   raa=w+(3.5928-2*c)*pow(w,1.5)/(lambda* fabs (a));

fprintf(fp,"\n %3d  %6.4f  %6.4f %6.4f %6.4f %6.4f %6.4f %6.4f %6.4f %6.4f
%6.4f",no,w,an1,ar1,rp,an2,ar2,ra,an3,ar3,raa);
fprintf(fp,"\n %6.4f %6.4f  %6.4f %6.4f %6.4f
%6.4f",sdn1,sdr1,sdn2,sdr2,sdn3,sdr3);
   j=1;l=1;s=0.0;n1=0;sn1=0;r1=0.0;sr1=0.0;sn2=0;r2=0.0;sr2=0.0;
   n2=0;n3=0;sn3=0;r3=0.0;sr3=0.0;
 }
  getch();
  fflush(stdin);
  fclose(fp);
}

/*PROGRAM END */
```

## APPENDIX -I (B)

## C-PROGRAM FOR SECTION [3.2.2]

### MIXTURE OF TWO EXPONENTIAL POPULATIONS

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <time.h>
#include <conio.h>
#include <ctype.h>
void main()
{ float  t1,l1,t2,l2,w,alpha,a;
  double
  w,y,min,s=0.0,r=0.5,sr1=0.0,r1=0.0,r2=0.0,r3=0.0,an1,ar1,sdn1,sdr1,an2,ar2,
  sr2=0.0,sdn2,sdr2,an3,ar3,sr3=0.0,sdn3,sdr3,b1,b2,b3,m1,m2,m3,s1,s2,s3,si,
  rp,ra,raa;
  int no,i,j,k,t,c=1,l;
  long n1=0,sn1=0,n2=0,sn2=0,n3=0,sn3=0;
  FILE *fp;
  clrscr();
  printf("\n Enter the location and scale parameters of two exponential :-");
  scanf("%f",&t1);
  scanf("%f",&l1);
  scanf("%f",&t2);
  scanf("%f",&l2);
  printf("\n Enter the shape parameter of loss function :-");
  scanf("%f",&a);
  printf("\n Enter the mixture term :-");
  scanf("%f",&alpha);
  fp=fopen("dsse1.dta","w+");
  randomize();
  for(no=20;no<=80;no+=5)
  { w=4.0/(pow(2*no/(l1*a)-1.0,2)-1.0);
    for(j=1;j<=1000;j++)
    {s=0.0;
     x=(1.0-alpha)*(t1-l1*log(1.0-(float) random(32000)/32000))
      +alpha*(t2-l2*log(1.0-(float) random(32000)/32000));
     min=x;s+=x;
     for(l=2;l<=8;l++)
      { x=(1.0-alpha)*(t1-l1*log(1.0-(float) random(32000)/32000))
      +alpha*(t2-l2*log(1.0-(float) random(32000)/32000));
      if(x<min) min=x;s+=x;
      }
```

```
/* Purely Sequential Plan */
    s1=s;i=8;m1=min;
    do{
     i+=1;
     y=(1.0-alpha)*(t1-l1*log(1.0-(float) random(32000)/32000))
           +alpha*(t2-l2*log(1.0-(float) random(32000)/32000));

     if(y<m1) m1=y;
     s1+=y;
      }while(s1>(double) i*(i-1)*l1/no+i*m1);
    n1+=i;sn1+=i*i;b1=1.0/(1.0-a*l1/i)-a*l1/i-1.0;r1+=b1;
    sr1+=b1*b1;

/* Accelerated Sequential Plan */
    s2=s;i=8;m2=min;
    do{
     i+=1;
     y=(1.0-alpha)*(t1-l1*log(1.0-(float) random(32000)/32000))
     +alpha*(t2-l2*log(1.0-(float) random(32000)/32000));
     if(y<m2) m2=y;
     s2+=y;
      }while(s2>(double)i*(i-1)*l1/(no*r)+i*m2);
    si=i*(s2-i*m2)/pow(i-1,2);t= (int) (no*si/l1)+1;
    if(i>t) k=i; else k=t;
    n2+=k;sn2+=k*k;b2=1.0/(1.0-a*l1/k)-a*l1/k-1.0;r2+=b2;
    sr2+=b2*b2;

/* Alternative Accelerated Sequential Plan */
    s3=s;i=8;m3=min;
    do{
     i+=1;
     y=(1.0-alpha)*(t1-l1*log(1.0-(float) random(32000)/32000))
         +alpha*(t2-l2*log(1.0-(float) random(32000)/32000));
     if(y<m3) m3=y;
     s3+=y;
      }while(s3>(double)i*i*(i-1)*l1/(no*r*(i+c))+i*m3);
    i=(int) i/r;
    n3+=i;sn3+=i*i;b3=1.0/(1.0-a*l1/i)-a*l1/i-1.0;r3+=b3;
    sr3+=b3*b3;
  }
```

```
  an3=n3/1000.0;ar3=r3/1000.0;sdn3=pow((sn3-1000*an3*an3)/999000,0.5);
  sdr3=pow((sr3-1000*ar3*ar3)/999000,0.5);
  an2=n2/1000.0;ar2=r2/1000.0;sdn2=pow((sn2-1000*an2*an2)/999000,0.5);
  sdr2=pow((sr2-1000*ar2*ar2)/999000,0.5);
  an1=n1/1000.0;ar1=r1/1000.0;sdn1=pow((sn1-1000*an1*an1)/999000,0.5);
  sdr1=pow((sr1-1000*ar1*ar1)/999000,0.5);
  rp=w+(3.5928)*pow(w,1.5)/(11* fabs (a));
  ra=w+(3+1/r)*pow(w,1.5)/(11* fabs (a));
  raa=w+(3.5928-2*c)*pow(w,1.5)/(11* fabs (a));

  fprintf(fp,"\n %3d  %6.4f  %6.4f %6.4f %6.4f %6.4f %6.4f %6.4f
%6.4f %6.4f",no,w,an1,ar1,rp,an2,ar2,ra,an3,ar3,raa);

  fprintf(fp,"\n %6.4f %6.4f  %6.4f %6.4f %6.4f
%6.4f",sdn1,sdr1,sdn2,sdr2,sdn3,sdr3);

  l=1;j=1;n1=0;sn1=0;r1=0.0;sr1=0.0;sn2=0;r2=0.0;sr2=0.0;
  n2=0;s=0.0;n3=0;sn3=0;r3=0.0;sr3=0.0;
  getch();
 }

fflush(stdin);
fclose(fp);
}

/* PROGRAM END */
```