รายการอ้างอิง

ภาษาไทย

กัลยา วานิชย์บัญชา. หลักสถิติ. กรุงเทพมหานคร : โรงพิมพ์จุฬาลงกรณ์มหาวิทยาลัย , 2539.

กิตติศักดิ์ พลอยพานิชเจริญ. สถิติสำหรับวิศวกรรม. กรุงเทพมหานคร : สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น) , 2540.

ประชุม สุรวัตถี. ทฤษฎีการอนุมานเชิงสถิติ. กรุงเทพมหานคร : สำนักพิมพ์โอเดียนสโตร์ ,2527.

ภาวนา มาศผล. การประมาณช่วงความเชื่อมั่นของสัมประสิทธิ์การถดถอยพหุเมื่อเกิดความสัมพันธ์ระหว่างตัวแปรอิสระ. วิทยานิพนธ์ปริญญามหาบัณฑิต ภาควิชาสถิติ บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย , 2540.

ภาษาอังกฤษ

David A.Lax. Robust Estimators of Scale Finite - Sample Performance in Long - tailed Symmetric Distributions : Journal of the American Statistical Association, 1985 , pp.736 – 741.

Efron,B.and Tibshirani , R. An Intrtoduction to the Bootstrap. New York : Chapman and Hall , 1993.

Prem S. Mann. Introductory Statistics. New York:John Wiley and Sons,1995.

Seki,Takashina and Yokoyama. Bootstrap for the normal parameter : A Simulation Study. Communications in Statistics. 1993 , pp.191-203.

ภาคผนวก

ภาคผนวก ก.

การสร้างตัวเลขสุ่ม (Random Number)

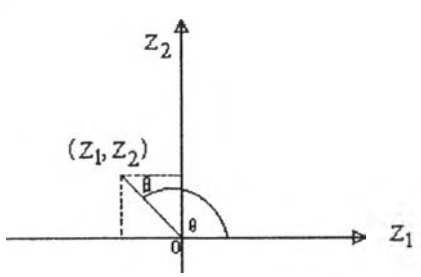
การสร้างลักษณะการแจกแจงแบบต่างๆ นั้นจะต้องอาศัยตัวเลขสุ่มเป็นพื้นฐานในการสร้าง สำหรับการวิจัยครั้งนี้จะใช้วิธีสร้างตัวเลขสุ่มตามวิธีของ White และ Sogmidt (1975) ซึ่งรายละเอียดด้วยฟังก์ชันต่อไปนี้

```
function rand (ix)
    iy = (ix * 16807)
    if (iy.lt.0)      iy = (iy + 2147483647) + 1
      rand = iy
      rand = rand / 2147483647.0
      ix = iy
    return
    end
```

ค่า ix จะเป็น seed หรือ ค่าเริ่มต้น ซึ่งจะต้องเป็นจำนวนเต็มบวกที่เป็นเลขคี่ rand เป็นค่าของตัวเลขสุ่มที่มีค่าอยู่ระหว่า 0 ถึง 1

การสร้างการแจกแจงแบบปกติ (Normal Distribution)

การผลิตเลขสุ่มที่มีการแจกแจงแบบปกติใช้วิธีของ Box และ Muller (1958) โดยผลิตเลขสุ่มที่มีการแจกแจงแบบปกติมาตรฐาน (N(0,1)) พร้อมกัน 2 ค่า และแต่ละค่าจะเป็นอิสระต่อกัน โดยใช้ตัวผลิต (Generator) Z_1 และ Z_2 พิจารณาดังรูปต่อไปนี้



พิจารณาจากรูปจะได้

$$Z_1 = B\cos(\theta) \quad (1)$$

$$Z_2 = B\sin(\theta) \quad (2)$$

เนื่องจาก $B = Z_1^2 + Z_2^2$ มีการแจกแจงแบบไคสแควร์ด้วยระดับความเป็นอิสระ 2 และเทียบเท่าการแจกแจงแบบเอ็กโปแนนเชียล มีค่าเฉลี่ยเท่ากับ 2 โดยวิธีแปลงผกผัน (Inverse Transformation) สามารถสร้างเลขสุ่มที่มีการแจกแจงแบบเอ็กโปแนนเชียลได้ดังนี้

$$B = (-2\ln R)^{1/2} \quad (3)$$

เมื่อ R เป็นเลขสุ่มที่มีการแจกแจงแบบสม่ำเสมอในช่วง (0,1)

จากการสมมาตรของการแจกแจงแบบปกติ (Normal Distribution) จะได้ว่ามุม θ มีการแจกแจงแบบสม่ำเสมอระหว่าง 0 ถึง 2π เรเดียน และมีรัศมี B กับมุม θ เป็นอิสระต่อกันจากสมการ (1) (2) และ (3) เราสามารถสร้างเลขสุ่มที่มีการแจกแจงแบบปกติมาตรฐานจากตัวเลขสุ่ม 2 ชุด R_1 และ R_2 กล่าวคือ

$$Z_1 = (-2\ln R_1)^{1/2}\cos(2\pi R_2)$$
$$Z_2 = (-2\ln R_1)^{1/2}\sin(2\pi R_2)$$

ซึ่ง R1 และ R2 เป็นตัวเลขสุ่มที่สร้างจากฟังก์ชัน function rand(ix) เมื่อได้ตัวเลขสุ่มที่มีการแจกแจงแบบปกติมาตรฐานแล้ว จะทำการแปลงตัวเลขสุ่มดังกล่าวโดยอาศัยฟังก์ชัน

$$EX_1 = \mu + \sigma Z_1$$
$$EX_2 = \mu + \sigma Z_2$$

ซึ่งจะได้ว่า EX_1 และ EX_2 มีการแจกแจงแบบปกติด้วยค่าเฉลี่ยเท่ากับ μ และค่าความแปรปรวนเท่ากับ σ^2 ($EX_i \sim N(\mu,\sigma^2)$; i = 1, 2) โดยรายละเอียดโปรแกรมย่อยสรุป

```
      function  dnorm (dmean, sigma)
      real  rand, dmorm, ru1, ru2
      common /seed/ ix, it
         pi = 3.142857143
         if (it.eq.1) goto 10
         ru1 = rand(ix)
         ru2 = rand(ix)
         U1 = sqrt (-2 * alog (ru1)) * cos(2 * pi * ru2)
         U2 = sqrt (-2 * alog(ru1)) * sim (2 * pi * ru2)
      dnorm = dmean + sigma * U1
         it = 1
   return
10    dnorm = dmean + sigma * U2
```

```
        it = 0
    return
    end.
```

การสร้างการแจกแจงแบบลาปลาซ

การแจกแจงแบบลาปลาซ มีฟังก์ชันความหนาแน่นอยู่ในรูป

$$f(x) = \frac{1}{2\beta}\exp-\left(-\left|\frac{x-\theta}{\beta}\right|\right) \quad ; \quad -\infty < x < \infty,\ -\infty < \theta < \infty, \beta > 0$$

การสร้างตัวแปรสุ่มที่มีการแจกแจงแบบลาปลาซ ใช้วิธี Inverse Tranformation ซึ่งแสดงได้ดังนี้

$$F(x) = \int_{-\infty}^{0} f(x)\,dx = \int_{0}^{\infty} f(x)\,dx$$

$$f(x) = \frac{1}{2\beta}e^{\left(\frac{x-\theta}{\beta}\right)} \quad ; \quad x \geq 0$$

ซึ่ง

พิจารณา เมื่อ $x < 0$

$$f(x) = \frac{1}{2\beta}e^{\left(\frac{x-\theta}{\beta}\right)}$$

$$F(x) = \int_{-\infty}^{x}\frac{1}{2\beta}e^{\left(\frac{x-\theta}{\beta}\right)}dx$$

$$F(x) = \frac{1}{2}\int_{-\infty}^{x}\frac{1}{\beta}e^{\left(\frac{x-\theta}{\beta}\right)}d\left(\frac{x}{\beta}\right).\beta$$

$$= \frac{1}{2}e^{\left(\frac{x-\theta}{\beta}\right)}$$

$$2F(x) = e^{\left(\frac{x-\theta}{\beta}\right)}$$

$$x = \theta + \beta[\ln 2 + \ln(F(x))]$$

พิจารณา

$$f(x) = \frac{1}{2\beta} e^{\left(\frac{x-\theta}{\beta}\right)} + \frac{1}{2\beta} e^{-\left(\frac{x-\theta}{\beta}\right)}$$

$$F(x) = \int_{-\infty}^{0} \frac{1}{2\beta} e^{\left(\frac{x-\theta}{\beta}\right)} dx + \int_{0}^{x} \frac{1}{2\beta} e^{-\left(\frac{x-\theta}{\beta}\right)} dx$$

$$F(x) = \int_{-\infty}^{0} \frac{1}{2\beta} e^{\left(\frac{x-\theta}{\beta}\right)} d\left(\frac{x}{\beta}\right).\beta + \int_{0}^{x} \frac{1}{2\beta} e^{-\left(\frac{x-\theta}{\beta}\right)} d\left(\frac{x}{\beta}\right).\beta$$

$$= \frac{1}{2}\left\{e^{0} - e^{-\infty} - e^{-\left(\frac{x-\theta}{\beta}\right)} + e^{0}\right\}$$

$$= \frac{1}{2}\left\{2 - e^{-\left(\frac{x-\theta}{\beta}\right)}\right\}$$

$$2F(x) = 2 - e^{-\left(\frac{x-\theta}{\beta}\right)}$$

$$e^{-\left(\frac{x-\theta}{\beta}\right)} = 2 - 2F(x)$$

$$x = \theta - \beta\left[\ln 2 + \ln(1 - F(x)\right]$$

ภาคผนวก ข.

การสุ่มตัวอย่างแบบใส่คืน (Sampling with replacement)

เป็นการสุ่มตัวอย่างที่ยอมให้มีหน่วยตัวอย่างซ้ำกันได้ นั่นคือหน่วยตัวอย่างแต่ละหน่วยมีโอกาส (Probability) ในการถูกสุ่มเท่ากันคือ $\frac{1}{N}$ เมื่อ N เป็นขนาดของประชากร ซึ่งในการวิจัยครั้งนี้ได้ใช้คอมพิวเตอร์เป็นเครื่องมือช่วยในการสุ่มตัวอย่างแบบใส่คืน โดยใช้ตัวเลขสุ่มที่มีการแจกแจงแบบสม่ำเสมอที่มีค่าอยู่ในช่วง [0,1] เป็นตัวเปรียบเทียบกับค่าความน่าจะเป็นแบบสะสม (Cumulative Probability) เพื่อกำหนดหน่วยตัวอย่างตามจำนวนที่ต้องการ ซึ่งขั้นตอนการสุ่มตัวอย่างแบบใส่คืนสรุปได้ดังนี้

1. คำนวณหาค่าความน่าจะเป็นของแต่ละหน่วยตัวอย่างเท่ากับ $\frac{1}{N}$
2. หาค่าความน่าจะเป็นแบบสะสมแล้วจัดแบ่งเป็นช่วง
3. สร้างตัวเลขสุ่มที่มีการแจกแจงแบบสม่ำเสมอซึ่งมีค่าอยู่ในช่วง [0,1]
4. นำตัวเลขสุ่มในข้อ 3 มาเปรียบเทียบกับค่าความน่าจะเป็นแบบสะสม ถ้าตกอยู่ในช่วงใดหน่วยนั้น ๆ จะถูกเลือกมาเป็นตัวอย่าง
5. กระทำตามขั้นตอนในข้อ 3 - 4 จำนวน n ครั้ง เมื่อ n คือขนาดตัวอย่างที่ต้องการ

ตัวอย่าง การสุ่มตัวอย่างแบบใส่คืน

เมื่อ N = 10

n = 3

คำนวณหาค่าความน่าจะเป็นของแต่ละหน่วยตัวอย่างได้ $\frac{1}{10}$ = 0.1 ดังตารางต่อไปนี้

หน่วยตัวอย่าง	ความน่าจะเป็น	ความน่าจะเป็นแบบสะสม	ช่วงความน่าจะเป็นแบบสะสม
1	0.1	0.1	0.01 - 0.10
2	0.1	0.2	0.11 - 0.20
3	0.1	0.3	0.21 - 0.30
4	0.1	0.4	0.31 - 0.40
5	0.1	0.5	0.41 - 0.50

หน่วยตัวอย่าง	ความน่าจะเป็น	ความน่าจะเป็นแบบสะสม	ช่วงความน่าจะเป็นแบบสะสม
6	0.1	0.6	0.51 - 0.60
7	0.1	0.7	0.61 - 0.70
8	0.1	0.8	0.71 - 0.80
9	0.1	0.9	0.81 - 0.90
10	0.1	1.0	0.91 - 1.00

สมมติเลขสุ่มตัวที่ 1 มีค่าเท่ากับ 0.65 จะได้ว่า หน่วยที่ 7 จะถูกเลือกมาเป็นตัวอย่าง

เลขสุ่มตัวที่ 2 มีค่าเท่ากับ 0.21 จะได้ว่า หน่วยที่ 3 จะถูกเลือกมาเป็นตัวอย่าง

เลขสุ่มตัวที่ 3 มีค่าเท่ากับ 0.25 จะได้ว่า หน่วยที่ 3 จะถูกเลือกมาเป็นตัวอย่าง

จะเห็นได้ว่าแต่ละหน่วยตัวอย่างมีโอกาสถูกเลือกมากกว่า 1 ครั้ง ทั้งนี้ขึ้นอยู่กับค่าของเลขสุ่มว่าจะตกอยู่ในช่วงใดของค่าความน่าจะเป็นแบบสะสม

ภาคผนวก ค.

```
**********************************
***          Main Program          ***
**********************************
      integer  n,nout,loop,bloop,check,heck,bbb
      real dnorm,nscale,lscale,llscale,scale,escale,x(100),xx(100),xb(100),q1,q3,diqr
      common /seed/ix,it
      common /count/check,heck,bbb
      ix = 489113
      it = 0
*****************************************************************
       Xsse1 = 0.0
       Xsse2 = 0.0
       Xsse3 = 0.0
       Xsse4 = 0.0
       Xsse5 = 0.0
       Xsse6 = 0.0
       Xsse7 = 0.0

       Xmse1 = 0.0
       Xmse2 = 0.0
       Xmse3 = 0.0
       Xmse4 = 0.0
       Xmse5 = 0.0
       Xmse6 = 0.0
       Xmse7 = 0.0

       SXbar  = 0.0
       SXmed = 0.0
```

```
 SXsd   = 0.0
SXunsd = 0.0
SXMbi  = 0.0
SSmbc  = 0.0
Ssmsi  = 0.0

SSXBbar  = 0.0
SSXBmed  = 0.0
SSXBsd   = 0.0
SSXBunsd = 0.0

write(*,*) 'Key Number_dist , m , n , r , c , p , dmean , sigma'
read(5,*) dist , m , n , r , c , p , dmean , sigma
          w = int(p*n)
  if(p.eq.0.05.or.p.eq.0.15) then
   if(n.eq.30.or.n.eq.50.or.n.eq.70) then
   w = w + 1
   endif
 endif

do 5 loop = 1,m
   check = 0
    heck = 0
     bbb = 0
 if (dist.eq.1) then
do 10 i = 1,n
     x(i) = 0.0
    xx(i) = 0.0
     x(i) = dnorm(dmean,sigma)
    xx(i) = x(i)
```

```
10      continue
        else if (dist.eq.2) then
      do 11 i = 1,n
        x(i) = 0.0
        xx(i) = 0.0
        x(i) = nscale(c,p,dmean,sigma,w,n)
        xx(i) = x(i)
11      continue

        else if (dist.eq.3) then
      do 12 i = 1,n
        x(i) = 0.0
        xx(i) = 0.0
        x(i) = llscale(c,p,dmean,sigma,w,n)
        xx(i) = x(i)
12      continue
        else if (dist.eq.4) then
      do 13 i = 1,n
        x(i) = 0.0
        xx(i) = 0.0
        x(i) = scale(c,p,dmean,sigma,w,n)
        xx(i) = x(i)
13      continue
        else if (dist.eq.5) then
      do 14 i = 1,n
        x(i) = 0.0
        xx(i) = 0.0
        x(i) = lscale(c,p,dmean,sigma,w,n)
        xx(i) = x(i)
14      continue
```

```
        else if (dist.eq.6) then
      do 15 i = 1,n
         x(i) = 0.0
         xx(i) = 0.0
         x(i) = escale(c,p,dmean,sigma,w,n)
         xx(i) = x(i)
15       continue
       endif
      call dmedian(n,xx,dmed)
       pmed = dmed
      call estimate(n,xx,xbar,sd,unsd)
      call biweight(n,r,pmed,x,dMbi,Smbc,Smsi)

       Xsse1 = Xsse1 + (dmean - xbar)**2
       Xsse2 = Xsse2 + (dmean - dmed)**2
       Xsse3 = Xsse3 + (sigma - sd)**2
       Xsse4 = Xsse4 + (sigma - unsd)**2
       Xsse5 = Xsse5 + (dmean - dMbi)**2
       Xsse6 = Xsse6 + (sigma - smbc)**2
       Xsse7 = Xsse7 + (sigma - smsi)**2

       SXbar = SXbar + xbar
       SXmed = SXmed + dmed
       SXmbi = SXmbi + dMbi
       SXsd  = SXsd  + sd
      SXunsd = SXunsd + unsd
      SSmbc = SSmbc + Smbc
      Ssmsi  = SSmsi +Smsi

      SXBbar = 0.0
```

```
SXBmed = 0.0
SXBsd  = 0.0
SXBunsd = 0.0

do 25 bloop = 1,k
call bootstrap(n,x,xb)
call sort(n,xb)
call quartile(n,xb,q1,q3,diqr)
call dmedian(n,xb,dmed)
call estimate(n,xb,xbar,sd,unsd)

SXBbar = SXBbar + xbar
SXBmed = SXBmed + dmed
SXBsd  = SXBsd + sd
SXBunsd = SXBunsd + unsd
25  continue

XBbar = SXBbar/k
XBmed = SXBmed/k
XBsd  = SXBsd/k
XBunsd = SXBunsd/k

SSXBbar = SSXBbar + XBbar
SSXBmed = SSXBmed + XBmed
SSXBsd  = SSXBsd + XBsd
SSXBunsd = SSXBunsd + XBunsd

XBsse1 = XBsse1 + (dmean - XBbar)**2
XBsse2 = XBsse2 + (dmean - XBmed)**2
XBsse3 = XBsse3 + (sigma - XBsd)**2
```

```
      XBsse4 = XBsse4 + (sigma - XBunsd)**2
5     continue

      Xbar = SXbar/m
      Xmed = SXmed/m
      Xsd  = SXsd/m
      Xunsd = SXunsd/m
      Xmbi = SXmbi/m
      XSmbc = SSmbc/m
      Xsmsi  = SSmsi/m

      Bbar = SSXBbar/m
      Bmed = SSXBmed/m
      Bsd  = SSXBsd/m
      Bunsd = SSXBunsd/m

      write(*,*)'XBAR  =',Xbar
      write(*,*)'XMED  =',Xmed
      write(*,*)'XBBAR =',Bbar
      write(*,*)'XBMED =',Bmed
      write(*,*)'XMBI  =',Xmbi
      write(*,*)'XSSMBC  =',XSmbc
      write(*,*)'XSSMSI  =',XSmsi
      write(*,*)'
      write(*,*)'XSD   =',Xsd
      write(*,*)'XUNSD =',Xunsd
      write(*,*)'XBSD  =',Bsd
      write(*,*)'XBUNSD =',Bunsd

      Xmse1 = xsse1/m
```

```
Xmse2 = xsse2/m
Xmse3 = xsse3/m
Xmse4 = xsse4/m
Xmse5 = xsse5/m
Xmse6 = xsse6/m
Xmse7 = xsse7/m

XBmse1 = XBsse1/m
XBmse2 = XBsse2/m
XBmse3 = XBsse3/m
XBmse4 = XBsse4/m

CVxbar  = (sqrt(Xmse1)/dmean)
CVxmed  = (sqrt(Xmse2)/dmean)
CVxsd   = (sqrt(Xmse3)/sigma)
CVxunsd = (sqrt(Xmse4)/sigma)
CVmbi   = (sqrt(Xmse5)/dmean)

CVxbbar  = (sqrt(XBmse1)/dmean)
CVxbmed  = (sqrt(XBmse2)/dmean)
CVxbsd   = (sqrt(XBmse3)/sigma)
CVxbunsd = (sqrt(XBmse4)/sigma)

write(*,*)'SCALE FACTOR =',C,'  % CONTAMINATE = ',P
write(*,*)'MSE FOR',m,'LOOP AND BOOTSTRAP',k,'  ROUND  N = ',N
write(*,*)'****************************************'
write(*,*)'  MSE OF LOCATION PARAMETER '
write(*,*)'    MSE  OF MEAN  =',Xmse1
```

```
      write(*,*)'    MSE OF MEDIAN =',Xmse2
      write(*,*)'    MSE OF BMEAN =',XBmse1
      write(*,*)'   MSE OF BMEDIAN =',XBmse2
      write(*,*)'      MSE OF MBI =',Xmse5
      write(*,*)'****************************************'
     write(*,*)'SCALE FACTOR =',C,'  % CONTAMINATE = ',P
     write(*,*)'MSE FOR',m,'LOOP AND BOOTSTRAP',k,'  ROUND  N = ',N
      write(*,*)'****************************************'
      write(*,*)'  MSE OF SCALE PARAMETER      '
      write(*,*)'     MSE  OF SD  =',Xmse3
      write(*,*)'     MSE OF UNSD =',Xmse4
      write(*,*)'      MSE OF BSD =',XBmse3
      write(*,*)'    MSE OF BUNSD =',XBmse4
      write(*,*)'    MSE OF SMBC =',Xmse6
      write(*,*)'    MSE OF SMSI =',Xmse7
      write(*,*)'****************************************'
     stop
     end
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
*******************************
***          Random Number         ***
*******************************
     function rand(ix)
        iy = (ix*16807)
      if (iy.lt.0)  iy = (iy + 2147483647) + 1
       rand = iy
       rand = rand/2147483647.0
         ix = iy
     return
     end
```

```
*******************************
***      Normal Distribution      ***
*******************************
      function dnorm(dmean,sigma)
      real rand,dnorm,ru1,ru2
      common /seed/ix,it
      pi = 3.142857143
       if (it.eq.1) goto 10
         ru1 = rand(ix)
         ru2 = rand(ix)
          u1 = sqrt(-2*alog(ru1))*cos(2*pi*ru2)
          u2 = sqrt(-2*alog(ru1))*sin(2*pi*ru2)
        dnorm = dmean + sigma*u1
          it = 1
      return
10      dnorm = dmean+sigma*u2
          it = 0
      return
      end
*******************************
***    Exponential Distribution    ***
*******************************
      function expo(c)
      real expo,delta,rd
      common/seed/ix,it
       delta = 1/(2*c)
          rd = rand(ix)
        expo = -(1/delta)*alog(rd)
      return
      end
```

```
*******************************
***     Lasplace Distribution      ***
*******************************
     function dlas(dmean,sigma2)
      real rand,dlas,q1
      common /seed/ix,it
          q1 = rand(ix)
        beta = sigma2/sqrt(2)
          rq = 2*q1
          wq = 1-q1
      if(q1.gt.0.and.q1.lt.0.5) then
        dlas = dmean + (beta*(alog(2) + alog(rq)))
      else if(q1.le.0.5.and.q1.lt.1) then
        dlas = dmean - (beta*(alog(2) + alog(wq)))
      endif
     return
     end
*********************************************
***    Contaminate Between Normal&Normal     ***
*********************************************
     function scale (c,p,dmean,sigma,w,n)
     real dmean,sigma2,q,scale
     integer check,heck,bbb
     common /seed/ix,it
     common /count/check,heck,bbb
      sigma2 = c*sigma
          q = rand(ix)
      if((q-p).le.0) then
        check = check + 1
```

```
      if(check.gt.w) then
       heck =  heck + 1
      scale = dnorm(dmean,sigma)
      else
          bbb = bbb + 1
        scale = dnorm(dmean,sigma2)
      endif
     else
       heck = heck + 1
       if(heck.gt.(n-w)) then
        bbb = bbb + 1
       scale = dnorm(dmean,sigma2)
       else
       scale = dnorm(dmean,sigma)
       endif
     endif
     return
     end
function nscale (c,p,dmean,sigma,w,n)
     real dmean,sigma2,q,scale
     integer check,heck,bbb
     common /seed/ix,it
     common /count/check,heck,bbb
      dmean2 = c*dmean
          q = rand(ix)
      if((q-p).le.0) then
       check = check + 1
      if(check.gt.w) then
       heck =  heck + 1
      nscale = dnorm(dmean,sigma)
```

```
      else
        bbb = bbb + 1
        nscale = dnorm(dmean2,sigma)
      endif
    else
      heck = heck + 1
      if(heck.gt.(n-w)) then
        bbb = bbb + 1
      nscale = dnorm(dmean2,sigma)
      else
      nscale = dnorm(dmean,sigma)
      endif
    endif
return
    end

*******************************************
***  Contaminate Between Normal & Lasplace  ***
*******************************************
    function lscale(c,p,dmean,sigma,w,n)
    real dmean,dlas,sigma2,q,lscale
    integer check,heck,bbb
    common /seed/ix,it
    common /count/check,heck,bbb
      sigma2 = c*sigma
         q = rand(ix)
    if((q-p).le.0) then
      check = check + 1
     if(check.gt.w) then
       heck = heck + 1
```

```
        lscale = dnorm(dmean,sigma)
       else
          bbb = bbb + 1
        lscale = dlas(dmean,sigma2)
       endif
     else
        heck = heck + 1
      if(heck.gt.(n-w)) then
         bbb = bbb + 1
       lscale = dlas(dmean,sigma2)
      else
       lscale = dnorm(dmean,sigma)
      endif
     endif
return
     end

function llscale(c,p,dmean,sigma,w,n)
     real dmean,dlas,dmean2,q,lscale
     integer check,heck,bbb
     common /seed/ix,it
     common /count/check,heck,bbb
       dmean2 = c*dmean
           q = rand(ix)
     if((q-p).le.0) then
        check = check + 1
      if(check.gt.w) then
         heck = heck + 1
       lscale = dnorm(dmean,sigma)
      else
```

```
        bbb = bbb + 1
       lscale = dlas(dmean2,sigma)
      endif
     else
        heck = heck + 1
      if(heck.gt.(n-w)) then
        bbb = bbb + 1
       lscale = dlas(dmean2,sigma)
      else
       lscale = dnorm(dmean,sigma)
      endif
     endif
     return
     end

*****************************************
***   Contaminate Between Normal&Exponential  ***
*****************************************
     function escale(c,p,dmean,sigma,w,n)
     real escale,expo,dnorm,q
     integer check,heck,bbb
     common /seed/ix,it
     common /count/check,heck,bbb
           q = rand(ix)
     if((q-p).le.0) then
        check = check + 1
      if(check.gt.w) then
        heck = heck + 1
       escale = dnorm(dmean,sigma)
      else
```

```
        bbb = bbb + 1
       escale = expo(c)
      endif
     else
        heck = heck + 1
      if(heck.gt.(n-w)) then
        bbb = bbb + 1
       escale = expo(c)
      else
       escale = dnorm(dmean,sigma)
      endif
     endif
     return
     end

****************************************
***                Rank Data          ***
****************************************
     subroutine sort(n,x)
     real bave,x(100)
        bave = 0.0
          k = n - 1
      do 20 i = 1,k
      do 21 j = i+1,n
       if(x(i).gt.x(j)) then
        bave = x(j)
        x(j) = x(i)
        x(i) = bave
       endif
21     continue
```

```
20     continue
return
    end
*********************************
***      FIND Q1,Q3 & IQR        ***
*********************************
    subroutine qoatile(n,x,q1,q3,diqr)
    real q1,q3,diqr,x(100)
     if(n.eq.20) then
       q1 = x(5)
       q3 = x(15)
      else if(n.eq.30) then
       q1 = (x(7)+x(8))/2
       q3 = (x(22)+x(23))/2
      else if(n.eq.50) then
       q1 = (x(12)+x(13))/2
       q3 = (x(37)+x(38))/2
      else if(n.eq.70) then
       q1 = (x(17)+x(18))/2
       q3 = (x(52)+x(53))/2
      endif
      diqr = q3-q1
    return
    end

*****************************************
***  Estimation  Location & Scale Parameter  ***
*****************************************
    subroutine estimate (n,x,xbar,sd,unsd)
    real x(100),sx,sxx,sd,xbar,unsd
```

```
      common /seed/ix,it
        sx = 0.0
        sxx = 0.0
      do 5 i = 1,n
        sx = sx + x(i)
        sxx = sxx + (x(i)**2)
5     continue
      xbar = sx/n
        var = (sxx - (n*(xbar**2)))/(n-1)
        sd = sqrt(var)
      if(n.eq.20) then
         c = 0.328804197
      else if(n.eq.30) then
         c = 0.264939407
      else if(n.eq.50) then
         c = 0.203104701
      else if(n.eq.70) then
         c = 0.170903639
      endif
       unsd = c*sqrt((sxx-(n*(xbar**2)))/2)
      return
      end

*****************************
***       Estimate Median       ***
*****************************
      subroutine dmedian(n,x,dmed)
      real save ,x(100)
      common /seed/ix,it
      save = 0.0
```

```
      a = 0.0
      b = 0.0
      c = 0.0
      k = n-1
     do 15 i = 1,k
      do 20 j = i+1,n
       if(x(i).gt.x(j)) then
        save = x(j)
        x(j) = x(i)
        x(i) = save
       endif
20     continue
15     continue
          a = n/2
          b = int(n/2)
          c = (n+1)/2
    if((a-b).eq.0.5) then
       dmed = x(c)
    else
       dmed = (x(a)+x(a+1))/2
    endif
    return
    end

**************************************
***              Bootstrap           ***
**************************************
    subroutine bootstrap(n,x,xb)
    real rc,x(100),xb(100),cf(100)
    common /seed/ix,it
```

```
      do  5 i = 1,n
       cf(i) = float(i)/float(n)
5     continue
      do 15   j = 1,n
           rc = rand(ix)
     do 20 ip = 1,n
           ip1 = ip-1
       if(ip.eq.0) then
           a1 = 0.0
       else
           a1 = cf(ip1)
       endif
           a2 = cf(ip)
       if(rc.gt.a1.and.rc.le.a2) then
         xb(j) = x(ip)
       endif
20     continue
15     continue
     return
     end

**********************************************
***                Check Otuliers Data              ***
**********************************************
     subroutine check(n,x,q1,q3,diqr,nout)
     integer n,nout
     real x(100)
       nout = 0
       con1 = 1.5*diqr
       con2 = 3*diqr
```

```
        b1 = q1 - con2
        b2 = q1 - con1
        b3 = q3 + con1
        b4 = q3 + con2
      do 30 i = 1,n
         pu = x(i)
      if(pu.lt.b1) then
        nout = nout + 1
        write(*,*)'Data have Extreme Outliers'
       else if(pu.gt.b4) then
        nout = nout + 1
        write(*,*)'Data have Extreme Outliers'
       else if(pu.ge.b1.and.pu.le.b2) then
        nout = nout + 1
        write(*,*)'Data have Mild Outliers'
       else if(pu.ge.b3.and.pu.le.b4) then
        nout = nout + 1
        write(*,*)'Data have Mild Outliers'
      endif
30     continue
     return
     end

*****************************************
***      Biweight Estimator of Location          ***
*****************************************
     subroutine biweight(n,r,pmed,x,dMbi,Smbc,Smsi)
     real x(100),d(100),u(100),wb(100),wu(100),cu(100),dMAD,Smbc
     common /seed/ix,it
      swb  = 0.0
```

```
      sxwb = 0.0
      sxtwb = 0.0
      ssin = 0.0
      scos = 0.0

     do 60 i = 1,n
      d(i) = abs(x(i) - pmed)
60    continue
      call dmedian(n,d,dmed)
      dMAD = dmed
     do 61 i = 1,n
       u(i) = (x(i) - pmed)/(r*dMAD)
      if (u(i).gt.-1.and.u(i).lt.1) then
        wb(i) = (1 - (u(i)**2))**2
      else
        wb(i) = 0.0
      endif
      if (u(i).gt.-22/7.and.u(i).lt.22/7) then
        wu(i) = sin(u(i))
        cu(i) = cos(u(i))
      else
        wu(i) = 0.0
        cu(i) = 0.0
      endif
       swb = swb + wb(i)
      sxwb = sxwb + (x(i)*wb(i))
     sxtwb = sxtwb + sqrt((((x(i) - pmed)**2)*wb(i))**2)
      ssin = ssin + wu(i)**2
      scos = scos + cu(i)
61    continue
```

```
    dMbi = sxwb/swb
      ps = n/(sqrt(n-1))
    Smbc = (ps*sqrt(sxtwb))/abs(swb)
      pa = (n*2.1*dMad)/(sqrt(n-1))
    Smsi = pa*atan((sqrt(ssin))/abs(scos))
return
end
```

ประวัติผู้เขียน

นางสาวคารณี ตั้งโชฎิกะ เกิดวันที่ 6 กันยายน พ.ศ. 2515 สำเร็จการศึกษาปริญญาวิทยาศาสตรบัณฑิต (วท.บ.) สาขาสถิติประยุกต์ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ประยุกต์ สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ ในปีการศึกษา 2536 และเข้าศึกษาต่อในหลักสูตรสถิติศาสตรมหาบัณฑิต ภาควิชาสถิติ คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัย ในปีการศึกษา 2538