# รายการอ้างอิง


ภาษาไทย

ธีระพร วีระถาวร. ตัวแบบเชิงเส้นทฤษฎีและการประยุกต์. พิมพ์ครั้งที่1. กรุงเทพฯ : บริษัทวิทยพัฒน์จำกัด, 2541.

ประชุม สุวัตถี. ทฤษฎีการอนุมานเชิงสถิติ. พิมพ์ครั้งที่2. กรุงเทพฯ : สถาบันบัณฑิตพัฒนบริหารศาสตร์, 2545.

สุชาดา กีระนันทน์. การอนุมานเชิงสถิติทฤษฎีขั้นต้น. พิมพ์ครั้งที่3. กรุงเทพฯ : สำนักพิมพ์จุฬาลงกรณ์มหาวิทยาลัย, 2545.

ภาษาอังกฤษ

Krause, A., Olson, M. The basics of S and S-Plus. New York : Springer-Verlag, 2000.

Davis, S.C.. Statistical Method for the Analysis of Repeated Measurement. New York : Springer-Verlag, 2002.

Diggle ,P.J., Liang K-Y, Zeger, S.L. Analysis of Longitudinal Data. New York : Oxford University Press, 1994.

Gill ,P.S. A robust mixed linear model analysis for longitudinal data. Statistics in Medicine 19 (2000) : 975-982.

Jones, R.H. Longitudinal data with serial correlation: a state-space approach. London : Chapman & Hall, 1993.

Laird, N.M., Ware, J.H. Random-Effect Model for Longitudinal Data. Biometric 38 (1982) : 963-984.

Mansour, H., Nordheim, E.V., and Rutledge,J.J. Maximum likelihood estimation of variance components in repeated measures design assuming autoregressive errors. Biometrics 41 (March 1985) : 287-294.

Rencher, A.C. Methods of multivariate analysis. New York : John Wiley & Son, 1985.

Seber,G.A.F., Wild, C.J. Nonlinear regression. New York : John Wiley & Son, 1989.

Searle, S.R. Linear model. New York : John Wiley & Son, 1971.

Ware, J.H. Linear models for the analysis of longitudinal studies. The American Statistician 39 (May 1985) : 95-101.

# ภาคผนวก

# ภาคผนวก ก

**ตัวอย่างการสร้างอิทธิพลของปัจจัย**

ในการประมาณค่าพารามิเตอร์ $\underset{\sim}{\beta}$ นั้นจะทำได้โดยใช้ฟังก์ชัน $\phi$ จากบทที่3 หัวข้อที่3.1 โดยกำหนดค่า $\phi$ =1.5 ในคำนวณประมาณพารามิเตอร์ระดับปัจจัยทดลองและผลกระทบร่วม ใช้ค่า $\phi$ =2.5 ในการคำนวณค่าระดับผงกระทบจากปัจจัยแบ่งบล็อกและ ผลกระทบจากปัจจัยระยะเวลา

1. กรณีที่กำหนดจำนวนระดับปัจจัยทดลอง 3 ระดับ จำนวนระยะเวลาการเก็บซ้ำ 3 ระยะเวลา และความแปรปรวนของข้อมูลมีค่า 25 จะได้ค่าดังนี้

| พารามิเตอร์ $\underset{\sim}{\beta}$ | กำหนดค่าพารามิเตอร์ |
|---|---|
| $\tau_i$ | $\tau_1 = 3.06 \quad \tau_2 = -3.06 \quad \tau_3 = 0$ |
| $\alpha_j$ | $\alpha_1 = 5.10 \quad \alpha_2 = -5.10 \quad \alpha_3 = 0$ |
| $\beta_k$ | $\beta_1 = -5.10 \quad \beta_2 = 5.10 \quad \beta_3 = 0$ |
| $\gamma_\ell$ | $\gamma_1 = 2.41 \quad \gamma_2 = -2.41 \quad \gamma_3 = 0$ |
| $(\tau\gamma)_{i\ell}$ | $(\tau\gamma)_{11} = 6.05 \ (\tau\gamma)_{12} = -6.05 \ (\tau\gamma)_{13} = 0$<br>$(\tau\gamma)_{21} = -6.05 \ (\tau\gamma)_{22} = 6.05 \ (\tau\gamma)_{23} = 0$<br>$(\tau\gamma)_{31} = 0 \ (\tau\gamma)_{32} = 0 \ (\tau\gamma)_{33} = 0$ |

2. กรณีที่กำหนดจำนวนระดับปัจจัยทดลอง 3 ระดับ จำนวนระยะเวลาการเก็บซ้ำ 3 ระยะเวลา และความแปรปรวนของข้อมูลมีค่า 100 จะได้ค่าดังนี้

| พารามิเตอร์ $\underset{\sim}{\beta}$ | กำหนดค่าพารามิเตอร์ |
|---|---|
| $\tau_i$ | $\tau_1 = 6.12 \quad \tau_2 = -6.12 \quad \tau_3 = 0$ |
| $\alpha_j$ | $\alpha_1 = 10.21 \quad \alpha_2 = -10.21 \quad \alpha_3 = 0$ |
| $\beta_k$ | $\beta_1 = -10.21 \quad \beta_2 = 10.21 \quad \beta_3 = 0$ |
| $\gamma_\ell$ | $\gamma_1 = 4.81 \quad \gamma_2 = -4.81 \quad \gamma_3 = 0$ |
| $(\tau\gamma)_{i\ell}$ | $(\tau\gamma)_{11} = 12.99 \ (\tau\gamma)_{12} = -12.99 \ (\tau\gamma)_{13} = 0$<br>$(\tau\gamma)_{21} = -12.99 \ (\tau\gamma)_{22} = 12.99 \ (\tau\gamma)_{23} = 0$<br>$(\tau\gamma)_{31} = 0 \ (\tau\gamma)_{32} = 0 \ (\tau\gamma)_{33} = 0$ |

3. กรณีที่กำหนดจำนวนระดับปัจจัยทดลอง 3 ระดับ จำนวนระยะเวลาการเก็บซ้ำ 4 ระยะเวลา และความแปรปรวนของข้อมูลมีค่า 100 จะได้ค่าดังนี้

| พารามิเตอร์ $\underset{\sim}{\beta}$ | กำหนดค่าพารามิเตอร์ |
|---|---|
| $\tau_i$ | $\tau_1 = 5.30 \quad \tau_2 = -5.30 \quad \tau_3 = 0$ |
| $\alpha_j$ | $\alpha_1 = 8.84 \quad \alpha_2 = -8.84 \quad \alpha_3 = 0$ |
| $\beta_k$ | $\beta_1 = -8.84 \quad \beta_2 = 8.84 \quad \beta_3 = 0$ |
| $\gamma_\ell$ | $\gamma_1 = 4.17 \quad \gamma_2 = -4.14 \quad \gamma_3 = 4.17 \quad \gamma_4 = -4.17$ |
| $(\tau\gamma)_{i\ell}$ | $(\tau\gamma)_{11} = 10.61 \; (\tau\gamma)_{12} = 10.61 \; (\tau\gamma)_{13} = -10.61 \; (\tau\gamma)_{14} = -10.61$<br>$(\tau\gamma)_{21} = -10.61 \; (\tau\gamma)_{22} = -10.61 \; (\tau\gamma)_{23} = 10.61 \; (\tau\gamma)_{24} = 10.61$<br>$(\tau\gamma)_{31} = 0 \; (\tau\gamma)_{32} = 0 \; (\tau\gamma)_{33} = 0 \; (\tau\gamma)_{34} = 0$ |

4. กรณีที่กำหนดจำนวนระดับปัจจัยทดลอง 4 ระดับ จำนวนระยะเวลาการเก็บซ้ำ 4 ระยะเวลา และความแปรปรวนของข้อมูลมีค่า 225 จะได้ค่าดังนี้

| พารามิเตอร์ $\underset{\sim}{\beta}$ | กำหนดค่าพารามิเตอร์ |
|---|---|
| $\tau_i$ | $\tau_1 = 5.63 \quad \tau_2 = -5.63 \quad \tau_3 = 5.63 \quad \tau_4 = -5.63$ |
| $\alpha_j$ | $\alpha_1 = 9.38 \quad \alpha_2 = -4.69 \quad \alpha_3 = 4.69 \quad \alpha_4 = -9.38$ |
| $\beta_k$ | $\beta_1 = -9.38 \quad \beta_2 = 4.69 \quad \beta_3 = -4.69 \quad \beta_4 = 9.38$ |
| $\gamma_\ell$ | $\gamma_1 = 4.69 \quad \gamma_2 = -4.69 \quad \gamma_3 = 4.69 \quad \gamma_4 = -4.69$ |
| $(\tau\gamma)_{i\ell}$ | $(\tau\gamma)_{11} = 11.25 \; (\tau\gamma)_{12} = 11.25 \; (\tau\gamma)_{13} = -11.25 \; (\tau\gamma)_{14} = -11.25$<br>$(\tau\gamma)_{21} = 11.25 \; (\tau\gamma)_{22} = 11.25 \; (\tau\gamma)_{23} = -11.25 \; (\tau\gamma)_{24} = -11.25$<br>$(\tau\gamma)_{31} = -11.25 \; (\tau\gamma)_{32} = -11.25 \; (\tau\gamma)_{33} = 11.25 \; (\tau\gamma)_{34} = 11.25$<br>$(\tau\gamma)_{41} = -11.25 \; (\tau\gamma)_{42} = -11.25 \; (\tau\gamma)_{43} = 11.25 \; (\tau\gamma)_{44} = 11.25$ |

5. กรณีที่กำหนดจำนวนระดับปัจจัยทดลอง 5 ระดับ จำนวนระยะเวลาการเก็บซ้ำ 4 ระยะเวลา และความแปรปรวนของข้อมูลมีค่า 5 จะได้ค่าดังนี้

| พารามิเตอร์ $\underset{\sim}{\beta}$ | กำหนดค่าพารามิเตอร์ |
|---|---|
| $\tau_i$ | $\tau_1 = 1.88$ $\tau_2 = -1.88$ $\tau_3 = 1.88$ $\tau_4 = -1.88$ $\tau_5 = 0$ |
| $\alpha_j$ | $\alpha_1 = 3.13$ $\alpha_2 = -3.13$ $\alpha_3 = 3.13$ $\alpha_4 = -3.13$ $\alpha_5 = 0$ |
| $\beta_k$ | $\beta_1 = -6.26$ $\beta_2 = 3.13$ $\beta_3 = -3.13$ $\beta_4 = 6.26$ $\beta_5 = 0$ |
| $\gamma_\ell$ | $\gamma_1 = 1.25$ $\gamma_2 = -1.25$ $\gamma_3 = 1.25$ $\gamma_4 = -1.25$ |
| $(\tau\gamma)_{i\ell}$ | $(\tau\gamma)_{11} = 3.75$ $(\tau\gamma)_{12} = 3.75$ $(\tau\gamma)_{13} = -3.75$ $(\tau\gamma)_{14} = -3.75$<br>$(\tau\gamma)_{21} = 3.75$ $(\tau\gamma)_{22} = 3.75$ $(\tau\gamma)_{23} = -3.75$ $(\tau\gamma)_{24} = -3.75$<br>$(\tau\gamma)_{31} = -3.75$ $(\tau\gamma)_{32} = -3.75$ $(\tau\gamma)_{33} = 3.75$ $(\tau\gamma)_{34} = 3.75$<br>$(\tau\gamma)_{41} = -3.75$ $(\tau\gamma)_{42} = -3.75$ $(\tau\gamma)_{43} = 3.75$ $(\tau\gamma)_{44} = 3.75$<br>$(\tau\gamma)_{51} = 0$ $(\tau\gamma)_{52} = 0$ $(\tau\gamma)_{53} = 0$ $(\tau\gamma)_{54} = 0$ |

ในการประมาณค่าพารามิเตอร์สำหรับกรณีอื่น ๆ ก็ทำได้โดยการคำนวณค่าพารามิเตอร์จากฟังก์ชัน $\phi$ แล้วนำค่าพารามิเตอร์ที่ได้ไปใช้ในการจำลองข้อมูลและเปรียบเทียบค่าประมาณ

# ภาคผนวก ข

## โปรแกรมการจำลองข้อมูลและการประมาณค่าพารามิเตอร์

```
Seed_129
phit_1.5
phie_2.5
counter_0
p_3
output2_0
b.array_c(3, 4, 6, 9)
for (loop.b in 2 :2 )
  {
    b_b.array[loop.b]
    m_p*p*b
    obser.y_array( ,dim=c(p,p,b))
    obsery.sim_array( ,dim=c(,p,p,b))
    vec.y_array(,dim=c(m,1))
    matric.c_array( ,dim=c(b,b))
    time_array( ,dim=c(b));
    inter_array( ,dim=c(p,b))
    treat_array(,dim=c(p))
    block1_array( , dim=c(p))
    block2_array(,dim=c(p))
###############Builed Matric X ###############
    matric.t_array(0,dim=c(p-1,m));
    matric.b1_array(0,dim=c(p-1,m))
    matric.b2_array(0,dim=c(p-1,m))
    matric.time_array(0,dim=c(b-1,m))
    q _ 0;
     g _ 0;
      if (p==3)
       {
         for(j in 1:3)
           for(k in 1:3)
             {
              if(( j-k) == 0)
                i _ 1
              else
                if ( ((j-k )== -1) || ((j-k) == 2) )
                  i _ 2
                else
                  i_3
               if( i < p)
                 for(l in 1:b)
                   {
                     q _ q+1;
                     matric.t[i,q] _ 1;
                   }
                else
                  for(l in 1:b)
                    {
                     q _ q+1;
                     for(w in 1:(p-1))
                       matric.t[w,q] _ -1;
                    }
               q _g;

             if( j < p)
               for(l in 1:b)
                 {
                  q _q+1;
                  matric.b1[j,q] _ 1;
                 }
             else
               for(l in 1:b)
                 {
                  q _ q+1;
                   for(w in 1:(p-1))
                     matric.b1[w,q]_ -1;
                 }
             q_ g;
             if (k < p)
               for (l in 1:b)
                 {
                  q _q+1;
```

```
               matric.b2[k,q] _1;
              }
           else
              for(l in 1:b)
                {
                  q _q+1;
                  for(w in 1:p-1)
                    matric.b2[w,q] _ -1;
                }
           q _ g;
           for(l in 1:b)
             if (l < b)
              {
                 q _q+1;
                 matric.time[l,q] _ 1;
              }
             else
               {
                 q _ q+1;
                 for (w in 1:(b-1))
                     matric.time[w,q] _ -1;
               }
           g _q;
         }##loop for i ,j
   }##if(p==3)

 if(p== 4)
  {
   for(j in 1:4)
     for(k in 1:4)
       {
         if ((j-k)== 0)
           i _ 1;
         else
           if ( ((j-k) == -1)||( (j-k)== 3) )
             i_2;
           else
             if ( ((j-k) == -2) || ((j-k) == 2))
               i_3
             else
               i_ 4;
             if( i < p)
            for(l in 1:b)
              {
                q _ q+1;
                matric.t[i,q] _ 1;
              }
           else
             for(l in 1:b)
               {
                 q _ q+1;
                 for(w in 1:(p-1))
                   matric.t[w,q] _ -1;
               }
         q _g;
         if( j < p)
           for(l in 1:b)
             {
               q _q+1;
               matric.b1[j,q] _ 1;
             }
         else
           for(l in 1:b)
           {
              q _ q+1;
               for(w in 1:(p-1))
               matric.b1[w,q]_ -1;
           }
         q_ g;
         if (k < p)
           for (l in 1:b)
            {
               q _q+1;
               matric.b2[k,q] _1;
             }
```

```
            else
              for(l in 1:b)
                {
                  q _q+1;
                   for(w in 1:p-1)
                     matric.b2[w,q] _ -1;
                }
          q _ g;
          for(l in 1:b)
            if (l < b)
              {
                q _q+1;
                matric.time[l,q] _ 1;
              }
              else
               {
                 q _ q+1;
                  for (w in 1:(b-1))
                    matric.time[w,q] _ -1;
               }
            g _q;
        }##loop for i ,j
 }

if (p== 5)
   {
   for(j in 1:5)
     for(k in 1:5)
       {
         if( (j-k) == 0)
           i _1
         else
           if (((j-k )== -1)|| ((j-k)== 4))
             i _2
           else
             if (((j-k) == -2) || ((j-k) == 3))
               i_3
             else
               if (((j-k) == -3) || ((j-k)== 2))
                 i_ 4
               else
                 i _ 5 ;
              if( i < p)
             for(l in 1:b)
               {
                 q _ q+1;
                 matric.t[i,q] _ 1;
               }
           else
             for(l in 1:b)
               {
                 q _ q+1;
                  for(w in 1:(p-1))
                    matric.t[w,q] _ -1;
               }
       q _g;
       if( j < p)
         for(l in 1:b)
           {
             q _q+1;
             matric.b1[j,q] _ 1;
           }
       else
         for(l in 1:b)
           {
             q _ q+1;
             for(w in 1:(p-1))
               matric.b1[w,q]_ -1;
           }
       q_ g;
       if (k < p)
         for (l in 1:b)
           {
             q _q+1;
             matric.b2[k,q] _1;
```

```
                }
              else
                for(l in 1:b)
                  {
                    q _q+1;
                    for(w in 1:p-1)
                      matric.b2[w,q] _ -1;
                  }
              q _ g;
              for(l in 1:b)
                if (l < b)
                  {
                    q _q+1;
                    matric.time[l,q] _ 1;
                  }
                else
                  {
                    q _ q+1;
                    for (w in 1:(b-1))
                      matric.time[w,q] _ -1;
                  }
              g _q;
            }##loop for i ,j
        }

      matric.inter_array(0,dim=c(1,m))
      for(j in 1:(b-1))
          for(i in 1:(p-1))
          {
            matric.inter_rbind(matric.inter,matric.t[i,1:m]*matric.time[j,1:m])
          }
      matric.inter_matric.inter[-1,1:m]
      matric.xt_rbind(1,matric.t,matric.b1,matric.b2,matric.time,matric.inter)
      matric.x_t(matric.xt)
      stand.array_c(5, 10, 15)
      for (loop.stand in 2:2)
       {
          stand.sim_stand.array[loop.stand]

##############Assing Parameter###############
          if( p != 4)
            {
              phistart.t _ (phit*stand.sim)/(b*(p-1))^0.5
              for( i in 1:(p-1))
               {
                 if ((i  %% 2)==0)
                   treat[i]_-phistart.t
                else
                   treat[i]_phistart.t
               }
              treat[p]_ 0
            }
          else
            {
              phistart.t _ (phit*stand.sim)/(b*p)^0.5
             for( i in 1 : p)
             {
               if((i %%2)==0)
                 treat[i]_phistart.t
               else
                 treat[i]_ -phistart.t
             }
            }

######block
        if( p ==3)
          {
            phistart.e_(phie*stand.sim)/(b*(p-1))^0.5
            block1[1]_-phistart.e
            block2[1] _phistart.e
            block1[2]_ phistart.e
            block2[2]_-phistart.e
            block1[3]_0
            block2[3] _0
          }
```

```
else
  if( p == 4)
    {
     phistart.e_(phie*stand.sim)/(b*(8))^0.5
     block1[1]_-phistart.e
     block2[1] _phistart.e
     block1[2]_-2*phistart.e
     block2[2]_2*phistart.e
     block1[3]_2*phistart.e
     block2[3] _-2*phistart.e
     block1[4]_phistart.e
     block2[4] _-phistart.e
    }
  else
    {
     phistart.e_(phie*stand.sim)/(b*(8))^0.5
     block1[1]_-phistart.e
     block2[1] _phistart.e
     block1[2]_-2*phistart.e
     block2[2]_2*phistart.e
     block1[3]_2*phistart.e
     block2[3] _-2*phistart.e
     block1[4]_phistart.e
     block2[4] _-phistart.e
     block1[5]_0
     block2[5] _0
    }

#######time#####
  if (b !=3 && b!=9 )
    {
     phistart.e_(phie*stand.sim)/(p^2*b)^0.5
     for(i in 1 : b )
      {
       if( (i %%2)==0)
         time[i]_phistart.e
       else
         time[i]_ - phistart.e
      }
    }
   else
    {
     phistart.e_(phie*stand.sim)/(p^2*b)^0.5
     for(i in 1 : (b-1))
      {
       if ((i %%2)==0)
         time[i]_phistart.e
       else
         time[i]_-phistart.e
      }
      time[b]_0
    }
############
  if( (p ==4) && (b != 3 ||b!=9 ))
    {
     phistrar.t_(phit*stand.sim)/(p)^0.5
     for (i in 1: p)
      for(j in 1 : b)
       {
        k_p %/% 2
        l _b %/%2
        if (( (i <= k )&&(j <= l)  )||( (i > k )&&(j > l) ))
          inter[i,j] _phistrar.t
        else
          inter[i,j] _ -phistrar.t
       }
    }
###########
  if ((p == 4) && (b== 3||b==9))
    {
     phistart.t_(phit*stand.sim)*(1/2)^0.5
       for(i in 1:p)
         for (j in 1:b)
           {
            k _p %/% 2
```

```
                l_b %/% 2
                if ( ( (i <= k )&&(j <= l) ) || ( (i > k )&&(j > l) ))
                  inter[i,j]_ phistart.t
                else
                  inter[i,j] _ -phistart.t
              }
          for(i in 1:p)
              inter[i,b] _0
        }
##########
        if ((p != 4) &&(b != 3 || b!=9))
          {
           phistart.t_(phit*stand.sim)* ( (p*b)/ ( p*(p-1)*b ) ) ^0.5
           for(i in 1:p)
             for (j in 1:b)
               {
                 k_p %/% 2;
                 l_b %/%2;
                 if (( (i <= k )&&(j <= l) ) || ( (i > k )&&(j > l) ))
                   inter[i,j]_phistart.t
                 else
                   inter[i,j]_-phistart.t
               }
              for(i in 1:b)
              inter[p,i] _ 0
          }
#######
          if ((p !=4)&&(b == 3||b==9))
           {
             phistart.t _(phit*stand.sim)*( (p*b) / (p*(p-1)*(b-1)) )^0.5
             for (i in 1:p)
               for(j in 1:b)
                {
                  k _ p %/% 2
                  l  _ b%/% 2
                  if (((i <= k )&&(j <= l) )||((i > k )&&(j > l)))
                    inter[i,j] _ phistart.t
                  else
                    inter[i,j] _ -phistart.t
                }
              for (i in 1:p)
                inter[i,b]_0
             for(i in 1:b)
                inter[p,i] _0
          }

##################Set Value Table First Obser##########
        if (p==3)
        {
         for(l in 1:b )
           for(j in 1:3)
             for(k in 1:3)
               {
                 if(( j-k) == 0)
                   i _ 1
                 else
                   if ( ((j-k )== -1) || ((j-k) == 2) )
                     i _ 2
                   else
                     i_3
                   obser.y[j,k,l] _ 50+treat[i] + block1[j]+ block2[k]+time[l]+inter[i,l];
               }##loop for i ,j,k
          }
        if(p== 4)
         {
          for(l in 1:b )
            for(j in 1:4)
              for(k in 1:4)
               {
                 if ((j-k)== 0)
                   i _ 1;
                 else
                   if ( ((j-k) == -1)||( (j-k)== 3) )
                     i_2;
                   else
```

```
                    if ( ((j-k) == -2) || ((j-k) == 2))
                      i_3
                    else
                      i_ 4;
                       obser.y[j,k,l] _ 50+treat[i] + block1[j]+ block2[k]+time[l]+inter[i,l]
                  }##loop for i ,j
            }
      if (p== 5)
       {
        for(l in 1:b )
          for(j in 1:5)
            for(k in 1:5)
             {
               if( (j-k) == 0)
                i _1
               else
                 if (((j-k )== -1)|| ((j-k)== 4))
                   i _2
                 else
                   if (((j-k) == -2) || ((j-k) == 3))
                     i_3
                    else
                     if (((j-k) == -3) || ((j-k)== 2))
                       i_ 4
                     else
                       i _ 5 ;
                   obser.y[j,k,l] _ 50+treat[i] + block1[j]+ block2[k]+time[l]+inter[i,l];
              }##loop for i ,j
        }##if

      Rnd <- function()
        {
          a_7^5
          m_(2^31)-1
          x1_(a*Seed)  %% m
          z_x1/m
          assign("Seed",x1,where=1)
          return(z)
        }

      Normal <- function()
       {
         repeat
          {
            u1_Rnd()
            u2_Rnd()
            v1_2*u1-1
            v2_2*u2-1
            s_(v1^2)+(v2^2)
            if (s <= 1) break
          }
         x_sqrt(-2*log(s)/s)*v1
         y_sqrt(-2*log(s)/s)*v2
         return(x)
       }
      corr.array_c(0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9)
      for (loop.cor in 1 :1)
        {
          corr.sim_corr.array[loop.cor]
          print(corr.sim)

################## Builed Matric Decomposition ############
          matric.c_array( ,dim=c(b,b));
          for(i in 1:b)
            for(j in 1:b)
             {
               z_i-j
               if (z >= 0)
                 matric.c[i,j]_corr.sim^(i-j)
               else
                 matric.c[i,j] _0
             }
#########################################
         counter_counter+1
         numloop_1
```

```
        output_array(0,dim=c(1,6))
        for(loop in 1: numloop)
         {
           print(c(counter,loop))

##################Simulation Multivariate##################
           sumsim_0;
           for (i in 1:p)
             for (j in 1:p)
              {
                 for (n in 1 :b)

                   z[n]_Normal()

                 for(l in 1:b)
                 {
                   sumsim_0;
                   for (q in 1:l)
                    {
                      sumsim _ sumsim+ z[q]*stand.sim*matric.c[l,q];
                    }
                      obsery.sim[i,j,l] _obser.y[i,j,l]+ sumsim;
                   }
              }

##################set vecter observation###################
            q_0;
            for(i in 1:p)
              for (j in  1:p)
                for(k in 1:b)
                  {
                   q _q + 1;
                    vec.y[q,1] _obsery.sim[i,j,k];
                  }

##############estimate OLS####################

            vec.err1_array(,dim=c(m-p^2,1));
            vec.err2_array(,dim=c(m-p^2,1));
            betahat.ols_(ginverse(matric.xt%*%matric.x))%*%(matric.xt%*%vec.y)
            errorhat.ols_vec.y-(matric.x%*%betahat.ols)
             k _0;
             l_0;
             for (i in 1:m)
              {
                if ((i %% b) != 1 )
                  {
                    k _ k+1;
                    vec.err1[k,1] _errorhat.ols[i];
                  }
                 if ((i %%b) != 0)
                  {
                     l _ l+1;
                     vec.err2[l,1] _errorhat.ols[i];
                   }
              }
                    sqr.err12_sum(vec.err1*vec.err2)
                    sqr.err1_sum(vec.err1^2)
                    sqr.err2_sum(vec.err2^2)
                    rhohat.ols_sqr.err12/sqr.err2;     ##rhohat OLS####

#############estimate two-stage####################
###builed matric v####
          matric.v_array( ,dim=c(b,b))
          calcorr_rhohat.ols
          for(i in 1:b)
            for(j in 1:b)
            {
              sumcorr _ 0
              for (k in 0:(min(i,j)-1))
                {
                   sumcorr _sumcorr + calcorr^(2*k)
                }
              matric.v[i,j] _ (calcorr^abs(i-j))*sumcorr;
            }
```

```
####tmatric##
            matric.inv_ginverse(matric.v)      ###inverse matric v##
            tmatric.inv_array( 0,dim=c(m,m) )
             for( i in 1: (p*p) )
               for( j in 1:b)
                 for (k in 1:b)
                   {
                     tmatric.inv[(b*(i-1))+j,(b*(i-1))+k]_matric.inv[j,k]
                   }

######
            betahat.two_ginverse(matric.xt%*%tmatric.inv%*%matric.x)%*%(matric.xt%*%
                         tmatric.inv%*%vec.y)##estimate beta.two
             r_2*p+p*b-2

            mse.two_t(vec.y-(matric.x%*%betahat.two))%*%tmatric.inv%*%(vec.y-
                              (matric.x%*%betahat.two))/(m-r)

             sigmahat.ols_mse.two          ##Sigmahat TWO

################Estimate MLE##############
             rhohat.mle_rhohat.ols;
             betahat.mle_betahat.two
             sigmahat.mle_mse.two
             reloop_0;
             repeat
              {
                reloop_reloop+1
                matric.v_array( ,dim=c(b,b))
                calcorr_rhohat.mle
                for(i in 1:b)
                 for(j in 1:b)
                   {
                     sumcorr _ 0
                     for (k in 0:(min(i,j)-1))
                       {
                         sumcorr _sumcorr + calcorr^(2*k)
                       }
                     matric.v[i,j] _ (calcorr^abs(i-j))*sumcorr;
                   }
                 dff.matricv_array(,dim=c(b,b))
                 for(i in 1:b)
                   for (j in 1:b)
                     {
                      head_calcorr^abs(i-j)
                      dff.head_(calcorr^(abs(i-j)-1))*abs(i-j)
                       sumcorr_0
                     for (k  in 0:(min(i,j)-1))
                        sumcorr _ sumcorr+ calcorr^(2*k)
                     back_sumcorr
                     sumcorr _ 0
                     for (k  in 0:(min(i,j)-1))
                        sumcorr _sumcorr + (2*k)*(calcorr^(2*k-1))
                    dff.back_sumcorr
                    dff.matricv[i,j]_(head*dff.back)+(back*dff.head)

             }
#### tmatric #######
              matric.inv_ginverse(matric.v)      ###inverse matric v##
              tmatric.inv_array( 0,dim=c(m,m) )
              tmatric.dff_array(0,dim=c(m,m))
              for( i in 1: (p*p) )
                for( j in 1:b)
                  for (k in 1:b)
                    {
                     tmatric.inv[(b*(i-1))+j,(b*(i-1))+k]_matric.inv[j,k]
                     tmatric.dff[(b*(i-1))+j,(b*(i-1))+k]_dff.matricv[j,k]
                    }

######

         ff.rho_t(vec.y(matric.x%*%betahat.mle))%*%tmatric.inv%*%tmatric.dff%*%tmatric.inv%*%
                 (vec.y-(matric.x%*%betahat.mle))/(2*sigmahat.mle)
         hesian_matric.inv%*%dff.matricv%*%matric.inv%*%dff.matricv
```

```
                hesian2_0
                for(i in 1:b)
                 {
                   hesian2_hesian2+hesian[i,i]
                 }
                hesian2_hesian2*p^2/2
                rhohat.mle1_rhohat.mle
                rhohat.mle_rhohat.mle+dff.rho/hesian2

#####Estimate###
              matric.v_array( ,dim=c(b,b))
              calcorr_rhohat.mle
              for(i in 1:b)
                for(j in 1:b)
                 {
                   sumcorr _ 0
                   for (k in 0:(min(i,j)-1))
                    {
                      sumcorr _sumcorr + calcorr^(2*k)
                    }
                   matric.v[i,j] _ (calcorr^abs(i-j))*sumcorr;
                 }
                  dff.matricv_array(,dim=c(b,b))
              for(i in 1:b)
               for (j in 1:b)
                {
                  head_calcorr^abs(i-j)
                  dff.head_(calcorr^(abs(i-j)-1))*abs(i-j)
                  sumcorr_0
                  for (k  in 0:(min(i,j)-1))
                    sumcorr _ sumcorr+ calcorr^(2*k)
                  back_sumcorr
                      sumcorr _ 0
                  for (k  in 0:(min(i,j)-1))
                    sumcorr _sumcorr + (2*k)*(calcorr^(2*k-1))
                 dff.back_sumcorr
                 dff.matricv[i,j]_(head*dff.back)+(back*dff.head)
                }

#### tmatric #######
             matric.inv_ginverse(matric.v)      ###inverse matric v##
             tmatric.inv_array( 0,dim=c(m,m) )
             tmatric.dff_array(0,dim=c(m,m))
             for( i in 1: (p*p) )
               for( j in 1:b)
                 for (k in 1:b)
                  {
                    tmatric.inv[(b*(i-1))+j,(b*(i-1))+k]_matric.inv[j,k]
                           tmatric.dff[(b*(i-1))+j,(b*(i-1))+k]_dff.matricv[j,k]
                  }
            betahat.mle_ginverse(matric.xt%*%tmatric.inv%*%matric.x)%*%(matric.xt%*%
                       tmatric.inv%*%vec.y)##estimate beta.two
            mse.mle_t(vec.y-(matric.x%*%betahat.mle))%*%tmatric.inv%*%(vec.y-
                    (matric.x%*%betahat.mle))

######################
             mse.mle_mse.mle/(m-r)
             sigmahat.mle_mse.mle
             if( ((rhohat.mle-rhohat.mle1 )^2<0.000001  )||(reloop==20)) break;
          }
########################################
       sqr.two_mse.two#(sum((vec.par-betahat.two)^2))^0.5
       sqr.mle_mse.mle#(sum((vec.par-betahat.mle)^2))^0.5
       sqrsigmahat.two_(stand.sim-sigmahat.ols^0.5)^2
       sqrsigmahat.mle_(stand.sim-sigmahat.mle^0.5)^2
       sqrrhohat.two_(corr.sim-rhohat.ols)^2
       sqrrhohat.mle_(corr.sim-rhohat.mle)^2
       output_output+c(sqr.two,sqrsigmahat.two,sqrrhohat.two,sqr.mle,
               sqrsigmahat.mle,sqrrhohat.mle)

     }          ######end for(loop)#########

     output_output/numloop;
     output_c(p,b,corr.sim,stand.sim,output)
     output2_rbind(output2,output)
```

```
###############end program###########
    }  ######end loop.cor#####
  }  ######end loop.stad#####
 }  ######end loop.b#####

print(output2)

########## END PROGRAM############
```

# ประวัติผู้เขียนวิทยานิพนธ์

นายประหยัด แสงงาม เกิดเมื่อวันที่ 8 กันยายน พ.ศ. 2522 ที่จังหวัดร้อยเอ็ด สำเร็จการศึกษาปริญญาวิทยาศาสตร์บัณฑิต (สถิติ) จากภาควิชาสถิติ คณะวิทยาศาสตร์ มหาวิทยาลัยมหาสารคาม เมื่อปีการศึกษา 2544 เข้าศึกษาต่อในหลักสูตรสถิติศาสตรมหาบัณฑิต ภาควิชาสถิติ คณะพาณิชยศาสตร์และการบัญชี จุฬาลงกรณ์มหาวิทยาลัย เมื่อปีการศึกษา 2545 และได้รับทุนพัฒนาอาจารย์ มหาวิทยาลัยศิลปากร เมื่อปีการศึกษา 2546