CHAPTER IV

CONCLUSION AND DISCUSSION

4.1 Conclusion.

In Chapter II, the techniques used to find the average density of states in heavily doped semiconductors are introduced. Using a semiclassical approximation, Kane was able to express the density of states in an analytical form, (2.2.12). Because Kane used free electron model in his calculation, the density of states obtained by him over estimated their values and predicted that the tail is of a Gaussian form. Latter Halperin and Lax used quantum mechanics to calculate this density of states. For high concentration of impurities, the distribution of impurity atoms can be treated as Gaussian distribution, Halperin and Lax found that the density of states could be expressed in the form

$$\rho_1(E) = (A(E)/\xi^2)\exp(-B(E)/2\xi) ,$$

or in terms of dimensionless functions

$$\rho_1(\nu) = (Q^3/E_Q\xi'^2)a(\nu)\exp(-b(\nu)/2\xi') .$$

By treating the impurity potentials screened Coulomb potentials, Halperin and Lax were able to calculate the values of density of states and other quantities numerically (some of them are shown in Table. 2.1). Their procedure involves solving the Hatree equation. The tail of density of states given by this method is an exponential tail which behaves roughly as $\exp(-|E|^n)$ where $0.5 < n < 2$. This result agrees with the experimental results. Since Halperin and Lax considered only the ground states contribution, the density of states calculated by them will be too low.

Edwards and Gulyaev[13] suggested that the problem of density of states in heavily doped semiconductor can be solved by using Feynman Path Integral technique. Recently, Sa-yakanit used this method to evaluate the analytical expression of density of states. When the screened Coulomb potential is used and only the ground states contribution are taken into consideration, the density of states can be expressed analytically as

$$\rho_1(\nu,Z) = (Q^3/E_Q\xi'^2)\{a(\nu,Z)/b(\nu,Z)^{3/4}\} \exp(-b(\nu,Z)/4\xi')D_{3/2}(\sqrt{b(\nu,Z)/\xi'}),$$

where Z is a variational parameter used to adjust the value of $\rho_1(\nu,Z)$. $a(\nu,Z)$ and $b(\nu,Z)$ are given by (2.4.22) and (2.4.23) respectively. For simplicity, only deep tail states, where $b(\nu,Z)/2\xi' >>1$, is considered. By using asymptotic expansion of parabolic cylinder function given by (2.4.27), the density of states at deep tail becomes

$$\rho_1(\nu,Z) = (Q^3/E_Q\xi'^2)a(\nu,Z) \exp(-b(\nu,Z)/2\xi'),$$

which is in the same form as that obtained by Halperin and Lax.

The numerical results can be evaluated by three methods, i) by minimizing $b(\nu,Z)$(denote case 1) or maximizing $\rho_1(\nu,Z)$(denote as case 2) with respect to Z as suggest by Halperin and Lax or maximizing the function

$$P(\nu,Z) = E_Q^2 \int_{\nu}^{\infty} d\nu' \int_{\nu'}^{\infty} d\nu'' \rho(\nu'',Z)$$

as proposed by Lloyd and Best (denote as case 3). The detail of these calculation are given in chapter 3. All numerical values of density of states and other relative quantities were calculated on electronic computer using standard techniques.

In the next two sections we will discuss the validity condition of $\rho_1(\nu,Z)$ at deep tail region and the numerical results given in chapter 3. Some suggestion for improvement will be given in the last section.

4.2 Validity Limit of $\rho_1(\nu,Z)$ at Deep Tail Region.

The numerical results given in chapter 3 are all calculated by using the expression of density of states at deep tail region given by the equation (2.4.26) which is the asymptotic form of (2.4.21) when $y = b(\nu,Z)/2\xi' >> 1$. To determine the validity condition of $\rho_1(\nu,Z)$ given by (2.4.26), we must consider the second term in the bracket of (2.4.27), $-p(p-1)/2x^2$. In our case $p = 3/2$ and $x^2 = 2y$, so that this correction term becomes $-3/16y$ and may be called the deep tail correction. This term will decrease the values of density of states given in chapter 3.

Halperin and Lax also evaluated a deep tail correction and set the limit of validity of $\rho(\nu)$ at $y > 3$. For this value of y, our deep tail correction is less than 1/16 or 6.25 % of the first term. We can determine the validity limit of $\rho_1(\nu,\tau)$ at any percentage value of this correction term. For example, if the required percentage is less than

1 % the deep tail condition will be that $y > 75/4$ or $b(\nu,Z) > 75\xi'/2$

4.3 Numerical Results.

4.3.1 Density of States.

From table and graphs in Chapter III, it can be seen that the density of states evaluated in the case 1 will decrease at $\nu \longrightarrow o$. But in the case 2 and 3, the values of density of states increase slowly as $\nu \longrightarrow o$. It can be said that the density of states are monotonic decreasing function. An interesting point is that all numerical results for each ξ' and at the same ν of the case 2 and 3 are almost the same values in deep tail region. When $\nu \ll 1$ these values are considerably different. However, the values which are obtained in case 3 are all less than those are obtained in case 2.

4.3.2 The Variational Condition and Variational Parameter Z.

The three equations used to calculate the variational parameter Z are that

$$\frac{D_{-4}(Z)}{D_{-3}(Z)} - \frac{2Z^{-3}}{T+\nu} = 0, \qquad 3.2.5$$

for the case 1

$$\frac{2D_{-4}(Z)}{D_{-3}(Z)} - \frac{3Z^{-3}}{2(T+\nu)} - \frac{2}{Z} - \frac{t(\nu,Z)}{2}\left(\frac{D_{-4}(Z)}{D_{-3}(Z)} - \frac{2Z^{-3}}{T+\nu}\right) = 0,$$

3.3.6

for the case 2, and

$$\left(\frac{D_{-4}(Z)}{4D_{-3}(Z)} - \frac{2}{Z}\right)\Gamma(7/4,y) - \left(\frac{3D_{-4}(Z)}{4D_{-3}(Z)} - \frac{2}{Z} - \frac{Z^{-3}}{T+\nu}\right)y^{1/2}\Gamma(5/4,y) = 0 \qquad 3.4.17$$

for the case 3.

The values of parameter Z obtained from these three equations do not vary for $\nu \gg 1$. We can show that for $\nu \gg 1$. these three equations have approximately the same forms. Let us first consider (3.3.6). If we multiply this equation by the factor $2\xi'/b(\nu,Z)$ and take the limit of very deep tail; i.e., $b(\nu,Z)/2\xi' \gg 1$, which equivalent to $\nu \gg 1$, equation (3.3.6) reduces to (3.2.5). In case 3, when an asymptotic form of $\Gamma(\alpha,y)$ at $y \gg 1$, [11]

$$\Gamma(\alpha,y) \approx y^{\alpha-1}\exp(-y) \ ; \ y \gg 1,$$

is used in (3.4.17), the equation becomes (3.2.5). Note that the meaning of $y \gg 1$ is equivalent to $b(\nu,Z)/2\xi' \gg 1$ because y is defined by (3.4.8).

When $\nu \ll 1$, the values of Z will become ∞ in the case 1 and will approach to some constants for any values of ξ' in the case 2 and 3. We can examine these results from the Table in Capter III.

4.3.3 The Quantity $n(\nu,Z)$.

As $\nu \to \infty$ the values of $n(\nu,Z)$ will go to 2, for all three cases. When $\nu \to o$ the value of $n(\nu,Z)$ will approach 0.5 in the case 1. This result agrees with Halperin and Lax result and with the experimental results. In the case 2 and 3, $n(\nu,Z) \to o$ as $\nu \to o$. This results do not agree with the experiments. However, if we consider only in very deep tail region, the value of $n(\nu,Z)$ will be between 0.5 and 2. It points out that the varidity condition of $\rho_1(\nu,Z)$ is necessary.

4.4 Suggestion.

Since all numerical values in this thesis are calculated from the approximated expression of density of states at deep tail and consider only the ground states contribution, so we can improve these results in two way, i) by using the density of states, $\rho_1(\nu,Z)$, given by (2.4.21) instead of (2.4.26) or, ii) by considering the excited states contributions. In the case i, the numerical values of density of states will be reduced by the deep tail correction term. The procedure of calculating numerical results, by maximizing $\rho_1(\nu,Z)$ in (2.4.21), will consume computer time approximatly equal to that used to calculate the density of states at deep tail by using the exact variational condition of Lloyd and Best. In the case ii, the excited state correction term will increase the numerical values of $\rho_1(\nu,Z)$, This correction term has been evaluated by Sa-yakanit and Glyde[20].

The techniques used to calculate the numerical values and the numerical values themself of density of states can be used to calculate further quantities such as Fermi energy and inverse screening length Q by the same procedure as Hwang[6,22] did. Or by using the numerical values of density of states only, we may construct the less complicated, analytic function of density of states in the same way as performed by Zee[23].

Appendix A

COMPUTER PROGRAMS

All numerical values given in Chapter III were evaluated on an IBM 370/138 computer using programs written in the FØRTRAN IV language. This appendix gives the list of main programs and subprograms for solving all the values of density of states and other relative quantities.

A.1 Main Programs.

There are three main programs each of which is used to evaluate the numerical results for each case of Chapter III. The procedurs of these main programs are all written in similar manner.

There are two groups of data cards in one set of data (each set is specified by the values of ξ'). The first group has only one data card which contains the values of ξ' and the number of data cards in the second group that will follow for this value of ξ'. The second group has any number of cards corresponding to that specified in the first data card. This group contains all the information that used to specify ν and other conditions. The representation of all input varibles in the main programs listed at the end of this section are given as follow:

PSI a fluctuation parameter ξ'.

K a number of the following data cards for each ξ'

PN a dimensionless energy ν . For an input, it is an initial value of ν .

PNL a final value of ν that we require for each data card.

DELN an increment or decrement used to vary the values of ν from initial value to final value.

BL a lower limit of the variational parameter Z.

Bu a upper limit of the variational parameter Z.

For solving the density of states by minimizing $b(\nu,Z)$, the value of ξ' in the data card must be equal to 50 only, and only one set of data is necessary to evaluate the numerical results for $\xi' = 50, 5, 0.5$ and 0.05. The other two main programs may execute many sets of data successively which each set is specified by the value of ξ'. To stop the execution a blank card is required.

Some other important variables of these main programs are described as follow:

Z a variational parameter Z.

RN equivalent to $n(\nu,Z)$

A dimensionless function $a(\nu,Z)$

B dimensionless function $b(\nu,Z)$

DENSIT a value of density of states, $\rho_1(\nu,Z)$, for ξ' specified by data card.

D2 a value of density of states, $\rho_1(\nu,Z)$, for $\xi' = 5$ (case 1 only).

D3 a value of density of states, $\rho_1(\nu,Z)$, for $\xi' = 0.5$ (case 1 only).

D4 a value of density of states, $\rho_1(\nu,Z)$, for $\xi' = 0.05$ (case 1 only).

The three main programs for case 1, 2 and 3 are listed respectively as follow :

```
      IMPLICIT REAL*8(A-H,Ø-Z)
      CØMMØN / GRO/PN
      CØMMØN / GR1/PSI,PI
      CØMMØN / GR5/A,B
      EXTERNAL ZNS
      CALL GLC96
      PI=3.1415926535897932384620O
      PS2=5.DO
      PS3=.5DO
      PS4=.5DO
      ER=1.D-12
1     READ(1,100)PSI,K
      IF(K.EQ.O)STØP
      WRITE(3,202)PSI,PS2,PS3,PS4
      N=1
      DØ5I=1,K
      READ(1,101)PN,PNL,DELN,BL,BU
      IF(DELN.EQ.O.)STØP
      NNN=DLØG10(PN)
      IF(PN.LT.1.)NNN=NNN-1
      DELP=10.DO**NNN
      Z=(BL+BU)/10.
2     M=10
      CALL RØØT (Z,M,1,ZNS, BL,BU,MES,20,ER)
      IF(MES.NE.O)GØTØ4
      DENSIT=DENS(Z)
      RN=2.DO*PN*Z*Z/(1.5DO+PN*Z*Z)
      D2=A*DEXP(-B/(2.DO*PS2))
      D3=A*DEXP(-B/(2.DO*PS3))
      D4=A*DEXP(-B/(2.DO*PS4))
      WRITE(3,200)PN,Z,RN,A,B,DENSIT,D2,D3,D4
      IF(N/5*5-N.EQ.O)WRITE(3,201)
      IF(N.NE.35)GØTØ3
      N=O
      WRITE(3,202)PSI,PS2,PS3,PS4
3     N=N+1
4     IF(DELP.GE.PN*.999DO)DELP=DELP/10.DO
      PN=PN-DELN*DELP
      IF(DABS(PN).LT.ER)PN=DELP
      IF(PN.GE.PNL)GØTØ2
5     CØNTINUE
      GØTØ1
100   FØRMAT(D10.3,I2)
101   FØRMAT(5D15.10)
200   FØRMAT(1H ,D9.2,2X,F9.4,2X,F7.4,2X,6(D12.5,2X))
201   FØRMAT(1HO)
202   FØRMAT(1H1,'NUMERICAL RESULTS ØF THE FUNCTIØN',7X,',',7X,',',7X,
     1 ' AND THE CØRRESPONDING VALUES ØF', 2X,',',2X/1H ,'BY MINIMIZING'
     2 7X,'  FØR  =',4(F6.2,',' )/1HO)
      END
      IMPLICIT REAL*8(A-H,Ø-Z)
      CØMMØN / GRO/ PN
      CØMMØN / GR1/PSI,PI
      CØMMØN / GR5 / A,B
      EXTERNAL ZNSM
      CALL GLC 96
      PI=3.141592653589793238462DO
      ER=1.D-12
1     READ(1,100)PSI,K
      IF(K.EQ.O)STØP
      WRITE(3,202)PSI
      N=1
      DØ5I=1,K
      READ(1,101)PNL,DELN,BL,BU
      IF(DELN.EQ.O.)STØP
      NNN=DLØG10(PN)
      IF(PN.LT.1.)NNN=NNN-1
      DELP=10.DO**NNN
      Z=(BL+BU)/10.
2     M=10
      CALL RØØT (A,M,-1,ZNSM,BL,BU,MES,20,ER)
      IF(MES.NE.O)GØTØ4
      DENSIT=DENS(Z)
      RN=2.DO*PN*Z*Z/(1.5DO+PN*Z*Z)
      WRITE(3,200)PN,Z,RN,A,B,DENSIT
      IF(N/5*5-N.EQ.O)WRITE(3,201)
      IF(N.NE.35)GØTØ3
      N=O
      WRITE(3,202)PSI
3     N=N+1
4     IF(DELP .GE .PN*.999DO)DELP=DELP/10.DO
      PN=PN-DELN*DELP
      IF(DABS(PN).LT.ER)PN=DELP
      IF(PN.GE.PNL)GØTØ2
5     CØNTINUE
      GØTØ1
100   FØRMAT(D10.3,I2)
101   FØRMAT(5D15.10)
200   FØRMAT(1H ,D10.2.2X,F10.6,2X,4(D14.7,2X))
201   FØRMAT(1HO)
202   FØRMAT(1H1,'NUMERICAL RESULTS ØF THE FUNCTIØN',7X,',',7X,',',7X,
     1 ' AND THE CØRRESPØNDING VALUES ØF', 2X,',',2X/1H ,'BY MA IMIZING',
     2 7X,'  FØR  =',F6.2/1HO)
      END
      IMPLICIT REAL*8(A-H,Ø-Z)
      CØMMØN / GRO/ PN
      CØMMØN / GR1/PSI,PI
      CØMMØN / GR5/A,B
      EXTERNAL VRC
      CALL GLC 96
      PI=3.14159265358979323846200
      ER=1.D-12
      CALL CONT(1.25D0,1.75D0)
1     READ(1,100)PSI,K
      IF(K.EQ.0)STØP
      WRITE(3,202)PSI
      N=1
      DØ5I=1,K
      READ(1,101)PN,PNL,DELN,BL,BU
      IF(DELN.EQ.0.)STØP
      NNN=DLØG10(PN)
      IF(PN.LT.1.)NNN=NNN-1
      DELP=10.DO**NNN
      Z=(BL+BU)/10.
2     M=10
      CALL RØØT(Z,     M,-1,VRC,BL,BU,MES,20,ER)
      IF(MES.NE.0)GØTØ 4
      DENSIT=DENS(Z)
      RN=2.DO*PN*Z*Z/(1.5DO+PN*Z*Z)
      WRITE(3,200)PN,Z,RN,A,B,DENSIT
      IF(N/5*5-N.EQ.0)WRITE(3,201)
      IF(N.NE.35)GØTØ3
      N=0
      WRITE(3,202)PSI
3     N=N+1
4     IF(DELP.GE.PN*.999DO)DELP=DELP/10.DO
      PN=PN-DELN*DELP
      IF(DABS(PN).LT.ER)PN=DELP
      IF(PN.GE.PNL)GØTØ2
5     CØNTINUE
      GØTØ1
100   FØRMAT(D10.3,I2)
101   FØRMAT(5D15.10)
200   FØRMAT(1H ,D10.2,2X,F10.6,2X,4(D14.7,2X))
201   FØRMAT(1HO)
202   FØRMAT(1H1,'NUMERICAL RESULTS ØF THE FUNCTIØN',7X,',',7X,',',7X,
      1' AND THE CORRESPONDING VALUES OF', 2X,',',2X/1H ,'BY MAXIMISING',
      27X,' FOR = ', F6.2/1HO)
      END
```

A.2 Subprograms.

There are fourteen subprograms which can be grouped into 6 groups. The programs, their description and flowchart will be listed into separated groups as follow:

A.2.1 Roots of Nonlinear Equations.

This group has only one subprogram which is used to find real roots of nonlinear equation.

RØUTINE NAME - RØØT

USAGE - CALL RØØT (X, NOIN, L, FUN, BL, BU, MES, NØL, ERRØR)

ARGUMENTS X - ØN INPUT, X SHØULD BE GUESS REAL RØØT IN THE INTERVAL [BL,BU] , BUT NOT NECESSARY. ON OUTPUT, X WILL BE THE APPRØXIMATE VALUE OF THE REAL RØØT ØF FUNCTIØN FUN WHICH LIES BETWEEN BL AND BU.

NØIN - SPECIFIED NUMBER ØF SUBINTERVALS IN EACH SIDE ØF X BETWEEN BL AND BU. (INPUT)

BL,BU - A LOWER AND UPPER BOUND OF THE INTERVAL IN WHICH THE USER WANT TO FIND THE ROOT. (INPUT)

FUN - A SINGLE - ARGUMENT REAL FUNCTION SUBPROGRAM, FUN(X), SUPPLIED BY THE USER WHICH COMPUTES FUN FOR ANY X IN THE INTERVAL (BL,BU). (INPUT)

FUN MUST BE DECLARED BY AN EXTERNAL STATEMENT IN THE CALLING PRØGRAM. FUN(X) IS THE FUNCTIØN FØR WHICH THE

THE RØØT IS TØ BE FØUND.

L - SPECIFIED SIGN ØF SLØPE ØF FUNCTIØN FUN AT THE NEIGHBØURHØØT ØF THE APPRØXIMATED RØØT. (INPUT)

IF L = -1, THE REQUIRED SLØPE IS NEGATIVE.
IF L = 0, THE SIGN ØF SLØPE IS NØT SPECIFIED.
IF L = 1, THE REQUIRED SLØPE IS POSITIVE.

MES - ERRØR MESSAGE. (ØUTPUT)

MES = 0, INDICATES X SHØULD BE THE ØNE APPRØXIMATE REAL RØØT ØF FUN IN THE INTERVAL [BL,BU] .

MES = 1, INDICATES THAT THE PRØGRAM CAN NØT FIND REAL RØØT ØF FUN IN THE INTERVAL [BL, BU].

MES = 2, NUMBER ØF ITERATIØNS HAS REACHED THE MAXIMUM ALLØWABLE NUMBER.

MES = 3, THE VALUE ØF APPRØXIMATED RØØT EVALUATED BY MULTIPLE ITERATIØN METHØD IS ØUT ØF THE INTERVAL [BL, BU] .

NØL - THE MAXIMUM NUMBER ØF ITERATIØNS TØ BE TAKEN TØ FIND THE RØØT BY MULTIPLE ITERATIØN METHØD .

ERRØR - THE REQUIRED ACCURACY USED TØ TERMINATE PRØGRAM.

PRECISIØN - DØUBLE PRECISIØN

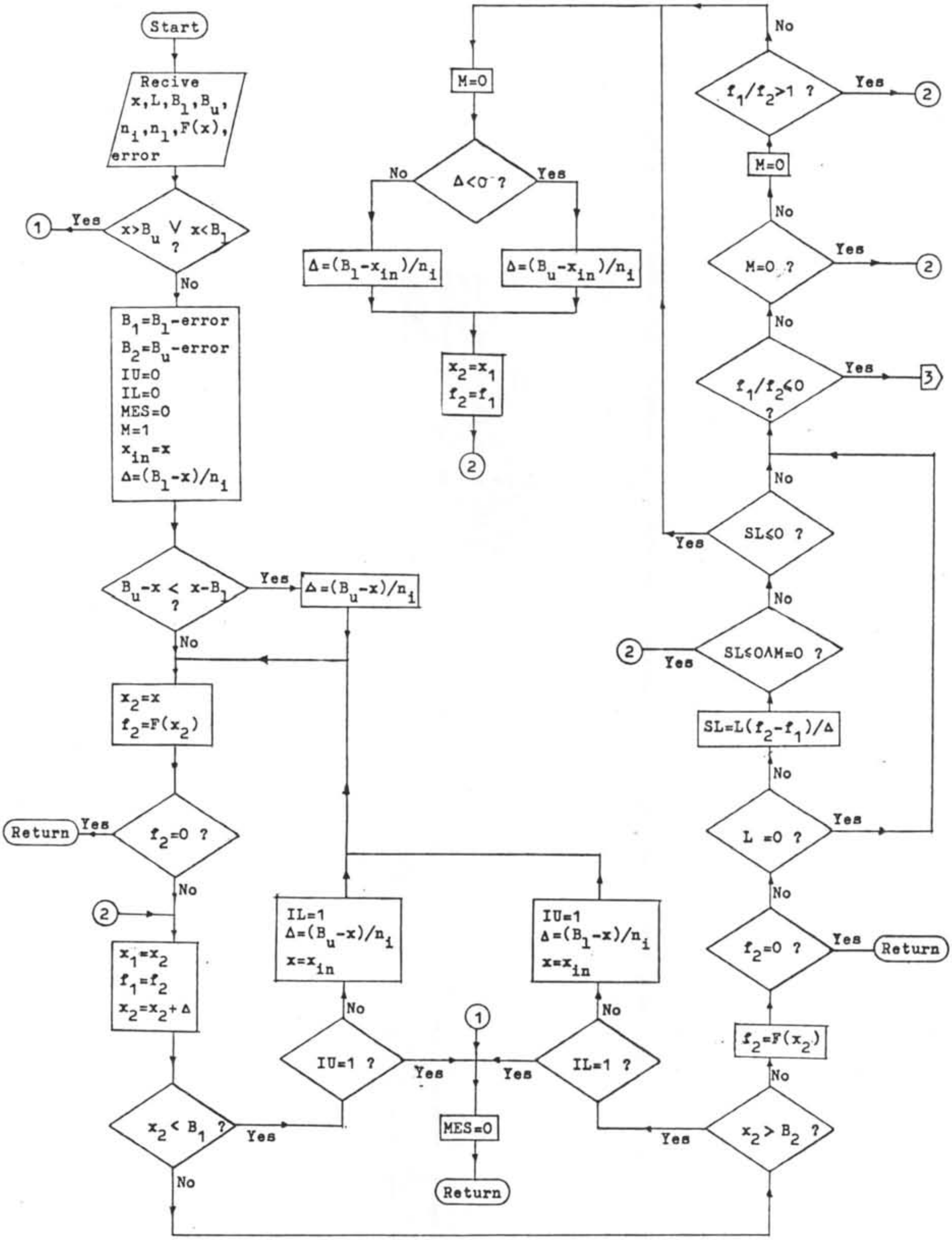
Purpose and Algorithm.

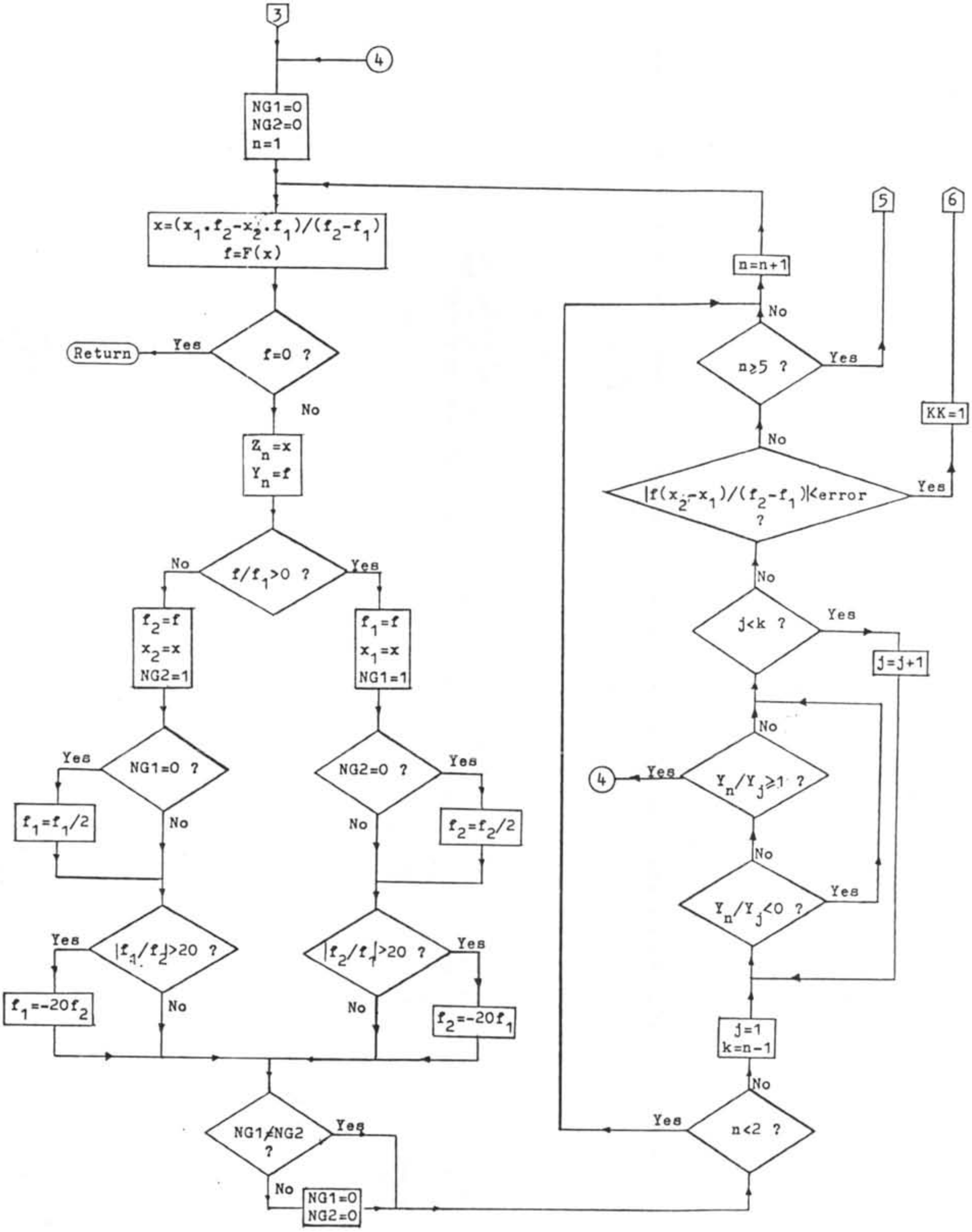
This subroutine subprogram RØØT is used to find an approximated real root of an equation f(x) = 0 in an interval [BL,BU] . It can be separated into three parts. The first part is used to find subinterval in which the required root lies. The second part finds a crude approximation of required real root by using a modified regula falsi method (section B.2.2). And the third part finds the required approximated real root in the range of specified accuracy by using multiple iteration technique (section B.2.3). These three parts are executed successivcly. The program and its flow-chart are given as follow :

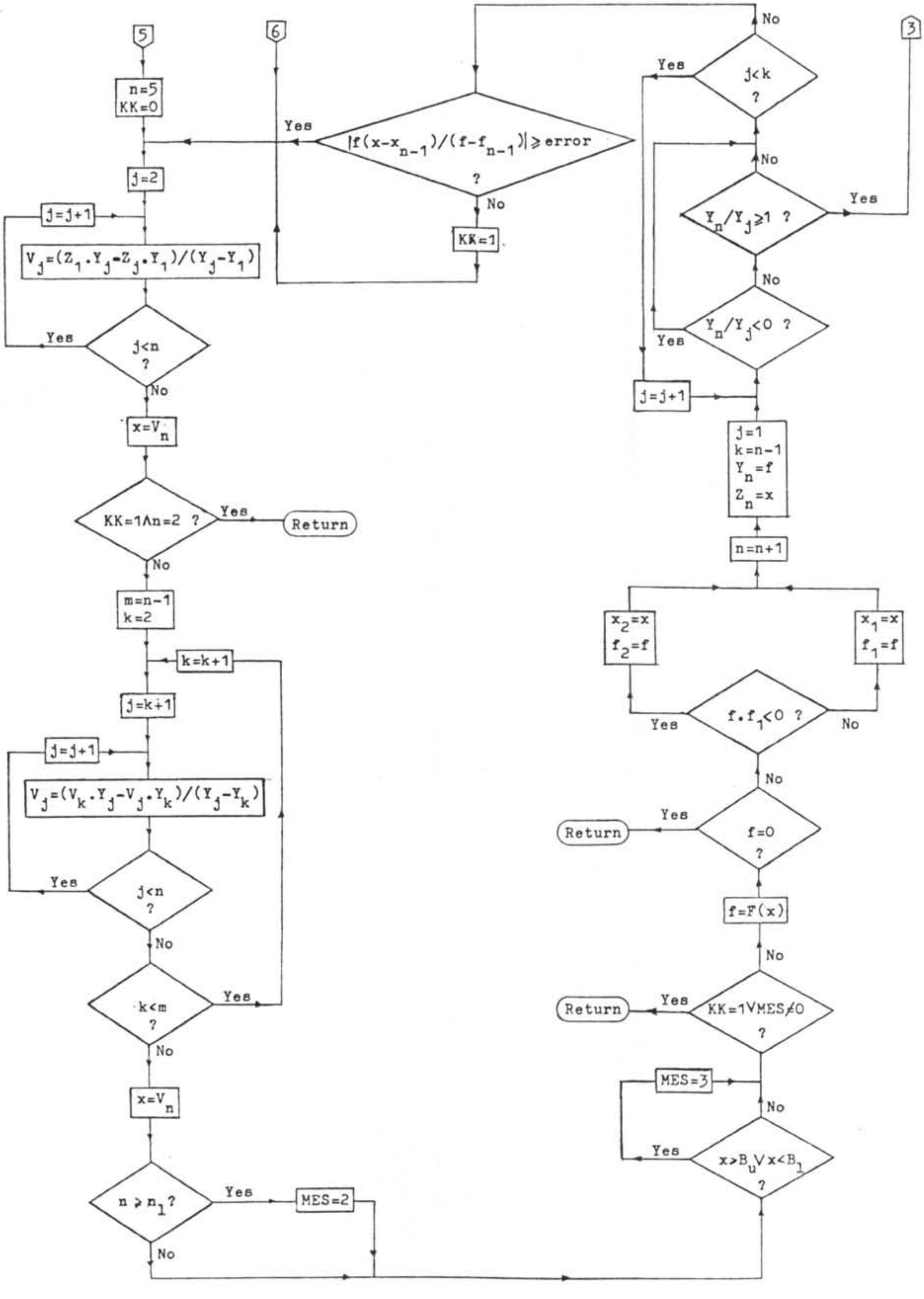
Program

```
      SUBRØUTINE RØØT( X,NØIN,L,FUN,BL,BU,MES,NØL,ERRØR)
      IMPLICIT REAL*8(A-H,Ø-Z)
      DIMENSIØN V(NØL),XX(NØL),FF(NØL)
      IF(X.GT.BU.ØR.X.LT.BL)GØTØ25
      B1=BL-ERRØR
      B2=BU+ERRØR
      IU=0
      IL=0
      MES=0
      M=1
      XIN=X
      DEL=(BL-X)/NØIN
      IF(BU-X.LT.X-BL)DEL=(BU-X)/NØIN
 1    X2=X
      F2=FUN(X2)
      IF(F2.EQ.0.DO)RETURN
 2    X1=X2
      F1=F2
      X2=X2+DEL
      IF(X2.LT.B1)GØTØ23
      IF(X2.GT.B2)GØTØ24
      F2=FUN(X2)
      IF(F2.EQ.0.DO)RETURN
      IF(L.EQ.0)GØTØ3
      SL=(F2-F1)/DEL*L
      IF(SL.LE.0.DO.AND.M.EQ.0)GØTØ2
      IF(SL.LE.0.DO)GØTØ4
 3    IF(F1/F2.LE.0.DO)GØTØ8
      IF(M.EQ.0)GØTØ2
      M=0
      IF(F1/F2.GT.1.DO)GØTØ2
 4    M=0
      IF(DEL.LT.0.)GØTØ6
      DEL=(BL-XIN)/NØIN
      GØTØ7
 6    DEL=(BU-XIN)/NØIN
 7    X2=X1
      F2=F1
      GØTØ2
 8    NG1=0
      NG2=0
 9    DO14N=1,5
      X=(X1*F2-X2*F1)/(F2-F1)
      F=FUN(X)
      IF(F.EQ.0.DO)RETURN
      XX(N)=X
      FF(N)=F
      IF(F/F1.LT.0.DO)GØTØ10
      F1=F
      X1=X
      NG1=1
      IF(NG2.EQ.0)F2=F2/2.
      IF(DABS(F2/F1).GT.20.)F2=-F1*20.
      GØTØ11
 10   F2=F
      X2=X
      NG2=1
      IF(NG1.EQ.0)F1=F1/2.
      IF(DABS(F1/F2).GT.20.)F1=-F2*20.
 11   IF(NG1.NE.NG2)GØTØ12
      NG1=0
      NG2=0
 12   IF(N.LT.2)GØTØ14
      IJJ=N-1
      DO13J=1,IJJ
      IF(FF(N)/FF(J).LT.0.D0)GØTØ13
      IF(FF(N)/FF(J).GE.1.D0)GØTØ8
 13   CØNTINUE
      IF(DABS(F*(X2-X1)/(F2-F1)).LT.ERRØR)GØTØ22
 14   CØNTINUE
      N=5
      KK=0
 5    DØ16J=2,N
 16   V(J)=(XX(1)*FF(J)-XX(J)*FF(1))/(FF(J)-FF(1))
      X=V(N)
      IF(KK.EQ.1.AND.N.EQ.2)RETURN
      N1=N-1
      DØ17K=2,N,
      IJ=K+1
      DØ17J=IJ,N
 17   V(J)=(V(K)*FF(J)-V(J)*FF(K))/(FF(J)-FF(K))
      X=V(N)
      IF(N.GE.NOL)MES=2
      IF(X.GT.BU.ØR.X.LT.BL)MES=3
      IF(KK.EQ.1.OR.MES.NE.0)RETURN
      F=FUN(X)
      IF(F.EQ.0.D0)RETURN
      IF(F*F1.LT.0.)GØTØ18
      F1=F
      X1=X
      GØTØ19
 18   F2=F
      X2=X
 19   N=N+1
      XX(N)=X
      FF(N)=F
      IJJ=N-1
      DØ20 J=1,IJJ
      IF(FF(N)/FF(J).LT.0.D0)GØTØ20
      IF(FF(N)/FF(J).GE.1.D0)GØTØ8
 20   CØNTINUE
      IF( DABS(F*(X-XX(N-1))/(F-FF(N-1))).LT.ERRØR)KK=1
      GØTØ5
22    KK=1
      GØTØ5
23    IF(IU.EQ.1)GØTØ25
      X=XIN
      DEL=(BU-X)/NØIN
      IL=1
      GØTØ1
24    IF(IL.EQ.1)GØTØ25
      X=XIN
      DEL=(BL-X/NØIN
      IU=1
      GØTØ1
25    MES=1
      RETURN
      END
```

Flowchart


Start
Recive $x, L, B_1, B_u, n_i, n_1, F(x),$ error
$x>B_u \vee x<B_1$?
Yes
1
No
$B_1=B_1-$error
$B_2=B_u-$error
IU=0
IL=0
MES=0
M=1
$x_{in}=x$
$\Delta=(B_1-x)/n_i$
$B_u-x < x-B_1$?
Yes
$\Delta=(B_u-x)/n_i$
No
$x_2=x$
$f_2=F(x_2)$
$f_2=0$?
Yes
Return
No
2
$x_1=x_2$
$f_1=f_2$
$x_2=x_2+\Delta$
$x_2<B_1$?
Yes
No
IU=1 ?
Yes
No
IL=1
$\Delta=(B_u-x)/n_i$
$x=x_{in}$
1
MES=0
Return
IL=1 ?
Yes
No
IU=1
$\Delta=(B_1-x)/n_i$
$x=x_{in}$
$x_2>B_2$?
Yes
No
$f_2=F(x_2)$
$f_2=0$?
Yes
Return
No
L =0 ?
Yes
No
$SL=L(f_2-f_1)/\Delta$
$SL\leq 0 \wedge M=0$?
Yes
2
No
$SL\leq 0$?
Yes
No
$f_1/f_2\leq 0$?
Yes
3
No
M=0 ?
Yes
2
No
M=0
$f_1/f_2>1$?
Yes
2
No
M=0
$\Delta<0$?
No
Yes
$\Delta=(B_1-x_{in})/n_i$
$\Delta=(B_u-x_{in})/n_i$
$x_2=x_1$
$f_2=f_1$
2


3
4
NG1=0
NG2=0
n=1
x=(x1.f2-x2.f1)/(f2-f1)
f=F(x)
f=0 ?
Yes
Return
No
Zn=x
Yn=f
f/f1>0 ?
No
Yes
f2=f
x2=x
NG2=1
f1=f
x1=x
NG1=1
NG1=0 ?
Yes
No
f1=f1/2
NG2=0 ?
Yes
No
f2=f2/2
|f1/f2|>20 ?
Yes
No
f1=-20f2
|f2/f1|>20 ?
Yes
No
f2=-20f1
NG1≠NG2
?
Yes
No
NG1=0
NG2=0
n<2 ?
Yes
No
j=1
k=n-1
Yn/Yj<0 ?
Yes
No
Yn/Yj≥1 ?
Yes
4
No
j<k ?
Yes
j=j+1
No
|f(x2-x1)/(f2-f1)|<error
?
Yes
KK=1
6
No
n≥5 ?
Yes
5
No
n=n+1


5
n=5
KK=0
j=2
j=j+1
$V_j=(Z_1 \cdot Y_j - Z_j \cdot Y_1)/(Y_j - Y_1)$
Yes
j<n
?
No
$x=V_n$
KK=1∧n=2 ?
Yes
Return
No
m=n-1
k=2
k=k+1
j=k+1
j=j+1
$V_j=(V_k \cdot Y_j - V_j \cdot Y_k)/(Y_j - Y_k)$
Yes
j<n
?
No
k<m
?
Yes
No
$x=V_n$
$n \geq n_1$?
Yes
MES=2
No
6
Yes
$|f(x-x_{n-1})/(f-f_{n-1})| \geq$ error
?
No
KK=1
No
3
Yes
j<k
?
No
$Y_n/Y_j \geq 1$?
Yes
No
$Y_n/Y_j < 0$?
Yes
j=j+1
j=1
k=n-1
$Y_n=f$
$Z_n=x$
n=n+1
$x_2=x$
$f_2=f$
$x_1=x$
$f_1=f$
$f \cdot f_1 < 0$?
Yes
No
No
Yes
Return
f=0
?
f=F(x)
No
Yes
Return
KK=1∨MES≠0
?
MES=3
No
Yes
$x>B_u \lor x<B_1$
?

A.2.2 Integration.

This set of subprograms is used to find the value of definite integral of any real, continuous function. There are two subprograms in this set as describe as follow :

RØUTINE NAME - GLI96

USAGE - RESULT = GLI96(A,B,N,FUN)

ARGUMENTS A,B - THE TWØ ENDPØINTS ØF THE INTERVAL ØF INTEGRATIØN. (INPUT)

N - NUMBER ØF SUBINTERVAL BETWEEN A AND B.

FUN - A SINGLE - ARGUMENT REAL FUNCTIØN SUBPRØGRAM SUPPLIED BY THE USER. (INPUT)
FUN MUST BE DECLARED EXTERNAL IN THE CALLING PRØGRAM.

GLI96 - ESTIMATE ØF THE INTEGRAL ØF FUN (X) FROM A TØ B. (ØUTPUT)

PRECISIØN - DØUBLE PRECISIØN

REQD. RØUTINES - GLC96

REMARKS GLC96 MUST BE CALLED BY MAIN PRØGRAM ØR ANY SUBPRØGRAMS AT LEAST ØNCE BEFØRE GLI96 IS USED.

Purpose and Algorithm

GLI96 attempts to solve the following problem : Given the name FUN of a real function subprogram , two real number A and B, and positive integer N, find a value of integral such that

$$\int_A^B \mathrm{FUN}(x)dx \approx \sum_{i}^{m} \alpha_{i,m} \sum_{j=1}^{N} \mathrm{FUN}(A+(B-A)(x_i+2j-1)/2N).$$

This routine uses the algorithm of composite Legendre - Gauss quadrature formula which all necessary constants are given by subroutine GLC96. The detail of this quadrature formula is given in section B.1.2. The program and its flowchart are given at the end of this subsection.

RØUTINE NAME	- GLC96
USAGE	- CALL GLC96
ARGUMENTS	- NØ ARGUMENT
PRECISIØN	- DØUBLE PRECISIØN

Purpose and Algorithm

GLC96 sets all necessary constants which are used in Legendre - Gauss quadrature formula of order 96(used 96 nodes of integration).

Programs

```
      REAL FUNCTIØN GLI96*8(A1,A2,N,FIN)
      IMPLICIT REAL*8(A-H,Ø-Z)
      CØMMØN / SET96/W(48),Z(48),M
      Ø=N
      D=A2-A1
      E=D/2./Ø
      GLI96=0.D0
      DØ 2J=1,M
      GL=0.D0
      DØ1 I=1,N
      G=DFLØAT(2*I-1)
      X=A1+E*(G-Z(J))
      Y=A1+E*(G+Z(J))
   1  GL=GL+ FIN(X)+FIN(Y)
   2  GLI96 =GLI96 +GL*W(J)
      GLI96 =GLI96*E
      RETURN
      END


      SUBRØUTINE GLC96
      IMPLICIT REAL*8(A-H,Ø-Z)
      CØMMØN /SET96/W(48),        Z(48),M
      M=48
      Z(1)=1.6276744849602970D-2
      Z(2)=4.8812985136049731D-2
      Z(3)=8.1297495464425559D-2
      Z(4)=.1136958501106659 2D0
      Z(5)=.1459737146548969 4D0
      Z(6)=.1780968823676186 0D0
      Z(7)=.2100313104605672 0D0
      Z(8)=.2417431561638400 1D0
      Z(9)=.2731988125910491 4D0
      Z(10)=.3043649443544963 5D0
      Z(11)=.3352085228926254 2D0
      Z(12)=.3656968614723136 4D0
      Z(13)=.3957976498289086 0D0
      Z(14)=.4254789884073005 4D0
      Z(15)=.4547094221677430 1D0
      Z(16)=.4834579739205963 6D0
      Z(17)=.5116941771546676 7D0
      Z(18)=.5393881083243574 4D0
      Z(19)=.5665104185613971 7D0
      Z(20)=.5930323647775720 8D0
      Z(21)=.6189258401254685 7D0
      Z(22)=.6441634037849671 1D0
      Z(23)=.6687183100439161 5D0
      Z(24)=.6925645366421715 6D0
      Z(25)=.7156768123489676 3D0
      Z(26)=.7380306437444001 3D0
      Z(27)=.7596023411766475 0D0
      Z(28)=.7803690438674332 2D0
```

```
Z(29)=.80030874413914082D0
Z(30)=.81940031073793168D0
Z(31)=.83762351122818712D0
Z(32)=.85495903343460146D0
Z(33)=.87138850590929650D0
Z(34)=.88689451740242042D0
Z(35)=.90146063531585234D0
Z(36)=.91507142312089807D0
Z(37)=.92771245672230869D0
Z(38)=.93937033975275522D0
Z(39)=.95003271778443764D0
Z(40)=.95968829144874254D0
Z(41)=.96832682846326421D0
Z(42)=.97593917458513647D0
Z(43)=.98251726356301468D0
Z(44)=.98805412632962380D0
Z(45)=.99254390032376262D0
Z(46)=.99598184298720929D0
Z(47)=.99836437586318168D0
Z(48)=.99968950388323077D0
W(1)=.32550614492363166D-1
W(2)=.32516118713868836D-1
W(3)=.32447163714064269D-1
W(4)=.32343822568575928D-1
W(5)=.32206204794030251D..1
W(6)=.32034456231992663D-1
W(7)=.31828758894411006D-1
W(8)=.31589330770727168D-1
W(9)=.31316425596861356D-1
W(10)=.3 010332586313837D-1
W(11)=.30671376123669149D-1
W(12)=.30299915420827594D-1
W(13)=.29896344136328386D-1
W(14)=.29461089958167906D-1
W(15)=.28994614150555236D-1
W(16)=.28497411065085386D-1
W(17)=.27970007616848334D-1
W(18)=.27412962726029243D-1
W(19)=.26826866725591762D-1
W(20)=.26212340735672414D-1
W(21)=.25570036005349361D-1
W(22)=.24900633222483610D-1
W(23)=.24204841792364691D-1
W(24)=.23483399085926220D-1
W(25)=.22737069658329374D-1
W(26)=.21966644438744349D-1
W(27)=.21172939892191300D-1
W(28)=.20356797154333324D-1
W(29)=.19519081140145022D-1
W(30)=.18660679627411467D-1
```

```
W(31)=.1778250231604526 1D-1
W(32)=.16885479864245172D-1
W(33)=.15970562902562291D-1
W(34)=.15038721026994938D-1
W(35)=.14090941772314861D-1
W(36)=.13128229566961573D-1
W(37)=.12151604671088320D-1
W(38)=.11162102099838498D-1
W(39)=.10160770535008416D-1
W(40)=.91486712307833866D-2
W(41)=.81268769256987592D-2
W(42)=.70964707911538653̄-2
W(43)=.60585455042359617D-2
W(44)=.50142027429275177D-2
W(45)=.39645543384446867D-2
W(46)=.29107318179349464D-2
W(47)=.18539607889469217D-2
W(48)=.79679206555201243D-3
RETURN
END
```

A.2,3 Parabolic Cylinder Function.

This group of subprograms is used to evaluate the values of two functions related to parabolic cylinder functions, and the ratio of two parabolic cylinder functions of different parameter but at the same points. There are three subprograms in this group.

```
RØUTINE  NAME        - DM
USAGE                - CALL DM(A,B,X,P1,P2,P3)
ARGUMENTS A,B        - TWØ  PARAMETERS ØF TWØ PARABØLIC CYLINDER
                       FUNCTIØNS. (INPUT)
          X          - INPUT ARGUMENT ØF PARABØLIC CYLINDER FUNCTIØN.
          P1         -- VALUE OF PARABØLIC CYLINDER FUNCTIØN WITH
                       PARAMETER A MULTIPLIED BY THE VALUE ØF
                       DEXP(X**2/4.DO), (ØUTPUT)
          P2         - VALUE ØF PARABØLIC CYLINDER FUNCTIØN WITH
                       PARAMETER B MULTIPLIED BY  THE VALUE ØF
                       DEXP(X**2/4.DO),(ØUTPUT)
          P3         - VALUE ØF P1/P2.
PRECISIØN            - DØUBLE PRECISIØN
REQD. RØUTINES       - D
REMARSK  IF A=B,  THE PRØGRAM CALCULATES ØNLY P1 AND SETS P2 = P3 = 0.
```

Purpose and Algorithm

DM computes the values of two functions,

$$P1 = \exp(x^2/4)D_A(x) ,$$

$$P2 = \exp(x^2/4)D_B(x),$$

and the ratio of these two functions

$$P3 = P1/P2 .$$

$D_p(x)$ is parabolic cylinder function which has p being its parameter, and $D_p(x)$ is defined by

$$D_p(x) = \{\exp(-x^2/4)/\Gamma(-p)\} \int_0^\infty t^{-p-1} \exp(-xt-t^2/2)dt.$$

ROUTINE NAME - D

USAGE - RESULT = D(A,X)

ARGUMENTS D - THE VALUES ØF PARABØLIC CYLINDER FUNCTIØN WITH PARAMETER A,EVALUATED AT THE PØINT X, MULTIPLIED BY DEXP(X**2/4.DO)*DGAMMA(-A),(ØUTPUT)

A - THE PARAMETER OF PARABOLIC CYLINDER FUNCTION. (INPUT)

X - THE SPECIFIED PØINT USED TØ EVALUATE THE VALUE ØF D. (INPUT)

PRECISIØN - DØUBLE PRECISIØN

REQD. RØUTINES - DDK, GLI96

Purpose and Algorithm.

The function subprogram D(A,X) is used to compute the value of

$$D = \Gamma(-A) \exp(X^2/4) D_A(X),$$

where $D_A(X)$ is a parabolic cylinder function with parameter A,

$$D_A(X) = \{\exp(-X^2/4)/\Gamma(-A)\} \int_0^\infty t^{-A-1} \exp(-Xt -t^2/2)dt.$$

Then the function D becomes

$$D = \int_0^\infty t^{-A-1} \exp(-Xt - t^2/2)dt.$$

The integral will be evaluated by using subprogram GLI96, and for more accuracy the value of A must be very negative. We can tranform this integral to very negative value of A by using the property of parabolic cylinder function[24], the recursion relation,

$$D_{p+n}(x) = \alpha_n D_p(x) - \beta_n D_{p-1}(x) ,$$

where

$$\alpha_n = x\alpha_{n-1} - \beta_{n-1} ,$$

$$\beta_n = p\alpha_{n-1} ,$$

$$\alpha_0 = 1$$

$$\beta_0 = 0 .$$

This program transforms any value of A to -6 and combines the integrand of two parabolic cylinder functions together. Then the function D becomes

$$D = \int_0^\infty t^{-A} \exp(-Xt - t^2/2)(\alpha_n/t + \beta_n/A)dt.$$

After this, the approximated maximum point, t_{max}, of the integrand is evaluated and obtained that

$$t_{max} \simeq \{x^2 - 4(A-1)\}^{1/2}/2 .$$

This t_{max} is used to reduce the maximum value of the integrand to have a value approximately 1, like this

$$D = \int_0^\infty dt(t/t_{max})^{-A}\exp(-(t-t_{max})\{(t+t_{max})/2+X\})(\alpha_n/t+\beta_n/A)/(\alpha_n/t_{max}+\beta_n/A)$$

RØUTINE NAME		- DDK
USAGE		- RESULT = DDK(X)
ARGUMENTS	X	- INPUT ARGUMENT.
	DDK	- ØUTPUT VALUE ØF THE FUNCTIØN DDK.
PRECISIØN		- DØUBLE PRECISIØN

Purpose and Algorithm

The DDK computes the value of function

$$DDK\ (X) = (X/Y)^{-r}\exp(W-(X-Y)\{(X+Y)/2+Z\})(\alpha/X+\beta/r)/(\alpha/Y+\beta/r)$$

where r, Y, Z, W are parameters given by main program

Programs

```
    SUBRØUTINE  DM(A,B,Z,P1,P2,P3)
    IMPLICIT REAL*8(A-H,Ø-Z)
    CØMMØN / PCF / S,T,A1,B1,W,Y
    W=75.DO
    RA=D(A,Z)
    R=DGAMMA(-S)
    P1=RA/R
    P2=O.
    P3=O.
    IF(A.EQ.B)RETURN
    RB=D(B,Z)
    R=DGAMMA(-S)
    P2=RB/R
    P3=P1/P2
    RETURN
    END


    REAL FUNCTIØN D*8(A,Z)
    IMPLICIT REAL*8(A-H,Ø-Z)
    CØMMØN/PCF / S,T,A1,B1,W,X
    EXTERNAL DDK
    S=A
    T=Z
    A1=1.DO
    B1=O.
    IF(S.LT.-6.DO)GØTØ2
  1 S=S-1.DO
    B2=B1
    B1=A1*S
    A1=Z*A1-B2
    IF(S.GE.-6.DO)GØTØ1
  2 X=(DSQRT(Z*Z-4.*(S-1.))-Z)/2.
    C2=2.DO*X
    D=GLI96(O.DO,C2,1,DDK)
    C1=C2
    C2=C2+32.DO
    DO 3N=1,20
    IF(DDK(C2).LT.1.D OO)GØTØ4
  3 C2=C2+32.DO
  4 D=D+GLI96(C1,C2,N,DDK)
    D=D*DEXP(-Z*X-X*X/2.DO-W)*X**(-S)*(A1/X+B1/S)
    RETURN
    END


    REAL FUNCTIØN DDK*8(X)
    IMPLICIT REAL*8(A-H,Ø-Z)
    CØMMØN /PCF/A,Z,A1,B1,W,Y
    G=W-(X-Y)*((X+Y)/2.DO+Z)
    DDK=DEXP( G)*(X/Y)**(-A)*(A1/X+B1/A)/(A1/Y+B1/A)
    RETURN
    END
```

A.2.4 Incomplete Gamma Function.

This set of subprograms is used to evaluate the numerical values that relate to two incomplete gamma functions with different parameter. Four subprograms are necessary. These subprograms are written for special purpose in this thesis only. For general case they must be modified.

ROUTINE NAME - SICGAM

USAGE - CALL SICGAM(A,B,X,F1,F2)

ARGUMENTS A,B - T PARAMETERS ØF TWØ RELATED INCØMPLETE GAMMA FUNCTIØNS. (INPUT)

X - INPUT ARGUMENT ØF THE TWØ RELATED INCØMPLETE GAMMA FUNCTIØNS.

F1 - THE VALUE ØF THE PRØDUCT ØF INCØMPLETE GAMMA FUNCTIØN ØF PARAMETER A WITH DEXP(X). (ØUTPUT)

F2 - THE VALUE ØF THE PRØDUCT ØF INCØMPLETE GAMMA FUNCTIØN ØF PARAMETER B WITH DEXP(X). (ØUTPUT)

PRECISIØN DØUBLE PRECISIØN

REQD. RØUTINES - CØNT, GLI96, FU, FM

REMARKS CØNT MUST BE CALLED BY MAIN PRØGRAM AT LEAST ØNCE BEFØRE SICGAM IS USED.

Purpose and Algorithm

This subprogram SICGAM evaluates the two values which are given as

$$F1 \;=\; \exp(X)\int_{X}^{\infty}\exp(-t)t^{A-1}\,dt = \exp(X)\Gamma(A,X),$$

$$F2 = \exp(X) \int_X^\infty \exp(-t)t^{B-1}\,dt = \exp(X)\Gamma(B,X),$$

or

$$F1 = \int_X^\infty \exp(X-t)t^{A-1}\,dt$$

$$F2 = \int_X^\infty \exp(X-t)t^{B-1}\,dt$$

where A and B are two parameters. By changing variable X-t=-Z, the above relation will become

$$F1 = \int_0^\infty \exp(-Z)(X+Z)^{A-1}\,dZ$$

$$F2 = \int_0^\infty \exp(-Z)(X+Z)^{A-1}\,dZ.$$

To evaluate the value of F1 and F2 this program considers the two cases where $X\leq 1$. and $X>1$. If $X\leq 1$ this program uses the summation form of $(A,X)^{11}$

$$\Gamma(A,X) = \Gamma(A) - \sum_{n=0}^{\infty} (-1)^n X^{A+n}/(A+n)n!$$

to evaluate F1 and F2. By computing the summation ($X\leq 1$), it takes much less time than performing numerical integration which gives the same accuracy.

For $X>1$ the program selects to use the integration form of F1 an F2 to evaluate their values.

RØUTINE NAME	- CØNT
USAGE	- CALL GØNT(A,B)

ARGUMENT A,B - TWØ INPUT PARAMETER.

PRECISIØN - DØUBLE PRECISIØN

REQD. RØUTINE - GLI96, FU, FM

Purpose and Algorithm

This subprogram is used to set the constants used in SICGAM.

RØUTINE NAME - FU

USAGE - RESULT = FU(X)

ARGUMENTS X - INPUT ARGUMENT.

FU - ØUTPUT VALUE ØF THE FUNCTIØN.

REQD. RØUTINE - NØ REQUIREMENT.

Purpose and Algorithm

This subprogram computes the value of

$$FU = \exp(-X)X^{\nu}$$

where ν is any parameter specified by the calling program.

RØUTINE NAME - FM

USAGE - RESULT = FM(X)

ARGUMENTS X - INPUT ARGUMENT

FM - ØUTPUT VALUE ØF THE FUNCTIØN FM.

PRECISIØN - DØUBLE PRECISIØN

REQD. RØUTINE - NØ REQUIREMENT

Purpose and Algorithm

This subprogram computes the value of

$$FM = \exp(-X)(Y+X)^{\nu}$$

where ν , and Y are parameters specified by the calling program.

Programs

```
   SUBRØUTINE SICGAM(A,B,Y,F1,F2)
   IMPLICIT REAL*8(A-H,Ø-Z)
   CØMMØN/GAMAB/G1,G2,X(7),R1(7),R2(7)
   CØMMØN /CØN / C,YYY
   EXTERNAL FU,FM
   YYY=Y
   F1=G1
   F2=G2
   IF(Y.EQ.0.)RETURN
   IF(Y.GT.1.)GØTØ2
   T1=Y**A
   T2=Y**B
   S1=T1/A
   S2=T2/B
   TN=1.DO
 1 T1=-T1*Y/TN
   T2=-T2*Y/TN
   S1=S1+T1/(TN+A)
   S2=S2+T2/(TN+B)
   TN=TN+1.DO
   IF(DABS(T1+T2).GT.1.D-18)GØTØ1
   F1=(F1-S1)*DEXP(Y)
   F2=(F2-S2)*DEXP(Y)
   RETURN
 2 Q=A-1.DO
   D=B-1.DO
   F1=R1(7)
   F2=R2(7)
   DØ3I=1,7
   IF(Y.LE.X(I))GØTØ4
 3 CØNTINUE
   C=Q
   F1=GLI96(0.DO,160.DO,5,FM)
   C=D
   F2=GLI96(0.DO,160.DO,5,FM)
   RETURN
 4 IF(I.EQ.7)GØTØ6
   DØ5J=I,6
   F1=F1+R1(J)
 5 F2=F2+R2(J)
 6 W=DEXP(Y)
   F1=F1*W
   F2=F2*W
   IF(Y.EQ.X(I))RETURN
   X2=X(I)-Y
   C=Q
   F1=F1+GLI96(0.DO,X2,1,FM)
   C=D
   F2=F2+GLI96(0.DO,X2,1,FM)
   RETURN
   END
```

```
  SUBRØUTINE CØNT(A,B)
  IMPLICIT REAL*8(A-H,Ø-Z)
  CØMMØN/GAMAB/G1,G2,X(7),R1(7),R2(7)
  CØMMØN /CØN/C,Y
  EXTERNAL FU
  G1=DGAMMA(A)
  G2=DGAMMA(B)
  X(1)=1.D0
  X(2)=2.D0
  X(3)=4.D0
  X(4)=8.D0
  X(5)=16.D0
  X(6)=32.D0
  X(7)=64.D0
  Q=A-1.D0
  D=B-1.D0
  DØ1I=1,6
  X1=X(I)
  X2=X(I+1)
  C=Q
  R1(I)=GLI96(X1,X2,1,FU)
  C=D
1 R2(I)=GLI96(X1,X2,1,FU)
  R2(7)=GLI96(64.D0,224.D0,5,FU)
  R1(7)=GLI96(64.D0,224 .D0,5,FU)
  C=Q
  RETURN
  END


  REAL FUNCTIØN FU*8(Z)
  IMPLICIT REAL*8(A-H,Ø-Z)
  CØMMØN / CØN/C,Y
  FU=DEXP(-Z)*(Z**C)
  RETURN
  END


  REAL FUNCTIØN FM*8(Z)
  IMPLICIT REAL*8(A-H,Ø-Z)
  CØMMØN / CØN/C,Y
  FM=DEXP(-Z)*((Y+Z)**C)
  RETURN
  END
```

A.2.5 Variational Condition (Equation)

There are three subprograms in this group. Each of them is used to evaluate the value of function at the left side of variational condition in each case of Chapter III (equations (3.2.5), (3.3.6) and (3.4.17)).

RØUTINE NAME		- ZNS
USAGE		- RESULT = ZNS(Z)
ARGUMENTS	Z	- INPUT AGRUMENT.
	ZNS	- ØUTPUT VALUE ØF THE FUNCTIØN ZNS.
REQD. RØUTINE		- DM
PRECISIØN		- DØUBLE PRECISIØN

Purpose and Algorithm

This subprogram computes the value of the function

$$\mathrm{ZNS} = (T+\nu)Z^3\, D_{-4}(Z)/D_{-3}(Z)-2$$

T,ν, Z, $D_{-3}(Z)$ and $D_{-4}(Z)$ are all defined in section 3.2. This function is modified from the left side of (3.2.5) by multiplying with the factor $(T+\nu)Z^3$. This subprogram is used when the variational condition of the case 1 (minimize $b(\nu,Z)$)is considered.

RØUTINE NAME		- ZNSM
USAGE		- RESULT = ZNSM(Z)
ARGUMENTS	Z	- INPUT ARGUMENT
	ZNSM	- ØUTPUT VALUE ØF THE FUNCTIØN ZNSM(X).
PRECISIØN		- DØUBLE PRECISIØN
REQD. RØUTINE		- ZNS

Purpose and Algorithm

This subprogram computes the value of the function

$$ZNSM \;=\; 4(T+\nu)Z^2(ZA-1)-3-b(\nu,Z)\{(T+\nu)Z^3A-2\}/\xi'$$

where $A = D_{-4}(Z)/D_{-3}(Z)$; T, ν, Z, $b(\nu,Z)$ and ξ' are all defined in section 3.3. This function is modified from the left side of (3.3.6) by multiplying with the factor $2(T+\nu)Z^3$. This subprogram is used when the variational condition of the case 2 (maximize $\rho_1(\nu,Z)$)is considered.

RØUTING		- VRC
USAGE		- RESULT = VRC(Z)
ARGUMENTS	Z	- INPUT ARGUMENT
	VRC	- ØUTPUT VALUE ØF THE FUNCTIØN VAC(X)
PRECISIØN		- DØUBLE PRECISIØN
REQD. RØUTINE		- SICGAM, DM

Purpose and Argument

This subprogram computes the value of the function

$$VRC = (ZA-8)\ \exp(y)\Gamma(7/4,y)-\{3ZA-8-4Z^2/(T+\nu)\}\ \exp(y)\Gamma(5/4,y)$$

where $A = D_{-4}(Z)/D_{-3}(Z)$; T,ν , Z, y are all defined in section 3.4. This function is modified from the left side of (3.4.17) by multiplying with the factor $4Z\ \exp(y)$. This subprogram is used when the variational condition of the case 3 in Chapter III (maximize $P(\nu,Z)$) is condisered.

Programs

```
REAL FUNCTIØN ZNS*8(Z)
IMPLICIT REAL*8(A-H,Ø-Z)
CØMMØN/GRO/PN/GR1/PSI,PI
CØMMØN / GR4/A,P,Q,B
Q=1.5DO+PN*Z*Z
CALL DM(-4.DO,-3.DO,Z,P1,P,A)
ZNS=A*Z*Q-2.DO
RETURN
END

REAL FUNCTIØN ZNSM*8(Z)
IMPLICIT REAL*8(A-H,Ø-Z)
CØMMØN/GRO/PN/GRI/PSI,PI
CØMMØN/GR4/A,P,Q,B
R=ZNS(Z)
Q1=1.5DO/Z/Z+PN
B=DSQRT(PI/2.DO)*Q1/P*Q1/2.DO
ZNSM=4.DO*(R-Q)-B*R/PSI+5.DO
RETURN
END

REAL FUNCTIØN VRC*8(Z)
IMPLICIT REAL *8(A-H,Ø-Z)
CØMMØN/GRO/PN/GRI/PSI,PI
Q=1.5DO+PN*Z*Z
Q1=1.5DO/Z/Z+PN
CALL DM(-4.DO,-3.DO,Z,P1,P,A)
B=DSQRT(PI/2.DO)*Q1/P*Q1/2.DO
Y=B/PSI/2.DO
CALL SICGAM(1.25DO,1.75DO,Y,F1,F2)
VRC=(Z*A-8.DO)*F2-(3.DO*A*Z-8.DO-4.DO/Q)*F1*DSQRT(Y)
RETURN
END
```

A.2.6 Density of States

There is only one subprogram that evaluates the value of density of states $\rho_1(\nu,Z)$ at the specified ν and Z.

RØUTINE NAME		-	DENS
USAGE		-	RESULT = DENS(Z)
ARGUMENT	Z	-	INPUT ARGUMENT
	DENS	-	ØUTPUT VALUE ØF FUNCTION DENS(Z)
PRECISIØN		-	DØUBLE PRECISIØN
REQD. RØUTINE		-	DM

Purpose and Algoritm

This subprogram computes the value of the function

$$\mathrm{DENS}(Z) = a(\nu,Z)\ \exp(-b(\nu,Z)/2\xi')$$

where $a(\nu,Z)$, $b(\nu,Z)$ and ξ' are defined in section 2.4.

Program

```
REAL FUNCTIØN DENS*8(Z)
IMPLICIT REAL*8(A-H,Ø-Z)
CØMMØN/GRO/PN
CØMMØN/GR5/A,B
Q=1.5DO+PN*Z*Z
CALL DM(-4.DO,-3.DO,Z,P1,P;A)
B=DSQRT(PI/2.DO)*Q/Z/Z*Q/Z/Z/P/2.DO
A=DSQRT((Q**3)/2.DO)/(Z**3)/P/(Z**3)/P/PI/8.DO/(Z**3)
DENS=A*DEXP(-B/PSI/2.DO)
RETURN
END
```

Appendix B

NUMERICAL METHODS

This appendix will be concerned with the numerical method used for calculating the numerical results given in Chapter III. The two procedures are considered, numerical integration and a procedure for finding the roots of nonlinear equation.

B.1 Numerical Integration.[25-28]

Some types of function can not be integrated analytically. We must therefore seek appropriate numerical procedures to approximate the value of the definite integral,

$$I\{f\} \equiv \int_a^b f(x)dx.$$

The approximations to the above equation are all essentially of the form

$$I_n\{f\} = \sum_{i=1}^{n} \alpha_{i,n} f(x_i).$$

Sums of this form are called a numerical quadrature formula. The n distinct points, x_i, are called the quadrature points or nodes and the quantities $\alpha_{i,n}$ are called the quadrature coefficients. To insure the maximum degree of precision, the nodes x_i must be specially picked. The maximum degree of precision will be achieved if the nodes x_i are all the roots of an orthogonal polynomial (see reference 25,26).

B.1.1 Legendre-Gauss Quadrature.[25]

In the case of an interval of integration $[a,b] = [-1,1]$, the sequence of polynomials which is required must be orthogonal over $[-1,1]$. Such a sequence of orthogonal polynomials is the Legendre polynomials. The integration formula based on these polynomials is called the Legendre-Gauss quadrature formula,

$$\int_{-1}^{1} f(x)dx = \sum_{i=1}^{n} \alpha_{i,n} f(x_i) + E_n\{f\}$$

The values of nodes and coefficients of Legendre-Gauss quadrature formula for some n are listed in Table B.1.

To evaluate the integral $\int_a^b f(x)dx$ using Legendre-Gauss quadrature formula, we must first transform the interval of integration $[a,b]$ to $[-1,1]$ by changing the variable x as

$$y = \tau x + \sigma .$$

The requirements of limit of integration are

$$y = 1 \quad \text{when} \quad x = b$$

and $$y = -1 \quad \text{when} \quad x = a .$$

Thus we obtain

$$y = (2x - a - b)/(b - a) \ ,$$

$$x = \{(b - a)y + (a + b)\}/2 \ ,$$

and the integral $\int_a^b f(x)dx$ will become to

$$\int_a^b f(x)dx = \{(b - a)/2\}\sum_{i=1}^{n} \alpha_{i,n}\, g(y_i) + E_n\{g\} ,$$

where $$g(y) = f(\{(a - b)y + (a + b)\}/2)$$

and y_i and $\alpha_{i,n}$ are nodes and coefficients of Legendre-Gauss quadrature formula of order n. The error term is given as

$$E_n\{g\} = (b - a)^{2n+1}(n!)^4\, f^{(2n)}(\eta)/(2n + 1)(2n!)^3 \ ; \ a < \eta < b .$$

B.1.2 Composite Legendre-Gauss Quadrature Formula.[28]

The magnitude of error term can be reduced (without increasing the order n of formula) by breaking up the interval [a,b] into a number, say m, of subintervals, then on each subinterval applying a quadrature formula and then sum these results. By dividing an interval [a,b] into m subinterval and using n-points quadrature formula in every intervals, an extra factor, $1/m^{2n}$, will be introduced into the error term,

$$E_n\{f\}_m = (b - a)^{2n+1}(n!)^4 f^{(2n)}(\eta)/(2n + 1)(2n!)^3 m^{2n}; \ a < \eta < b.$$

and the integral will be approximate by the following equation

$$\int_a^b f(x)dx = \sum_{i=1}^{n} \alpha_{i,n} \sum_{j=1}^{m} f(a +(b -a)(x_i + 2j - 1)/2) + E_n\{f\}_m \ .$$

A quadrature formula of this type is called a composite quadrature formula.

B.2 Solving Roots of Nonlinear Equation.[28-30]

One of the frequently encounter problems in scientific work is to find the roots of equations of the form

$$f(x) = 0. \qquad B.2.1$$

Some types of $f(x)$ can not be solved for its roots analytically. In general we can hope to obtain only approximate solutions, relying on some numerical computation techniques. " Approximate solution " mean either a point η , for which (B.2.1) is approximately satisfied (for which $f(\eta) \approx 0$), or a point η which is close to a solution, ξ, of (B.2.1).

B.2.1 First Procedure for Iterative Method.

To find the real roots of any equations by iterative method, it is necessary first to find an interval that the root is contained.

The theorem state that :

If $f(x)$ is continuous on $[a, b]$ and if $f(a)$ and $f(b)$ have opposite signs,

$$f(a)\,f(b) < 0 \ , \qquad B.2.2$$

then there is at least one real root between a and b.

So that to find an interval contains real root we only find the point a and b which satisfy the condition (B.2.2)

B.2.2 Regula Falsi and Modified Regula Falsi Methods.[29,30]

When an interval $[x_i,x_j]$ where $f(x_i)\,f(x_j) < 0$ has been found, the next approximate real root can be obtain by using the formula

$$x_{i+1} = \{x_j f(x_i) - x_i \ f(x_j)\}/\{f(x_i)-f(x_j)\} \ ,$$

where $i > 2,3,\ldots$ and j may be any positive integer not greater than $i - 1$ that makes $f(x_i)\ f(x_j) < 0$. Then, use the new point x_{i+1} and one of the point x_i or x_j which gives the value at that point has opposite sign to $f(x_{i+1})$ and calculate the next approximate real root. The procedure will be used repeatedly until we satisfy. This method converges for any continuous function but it fails completely to give a small interval in which a root is known to lie and this cause this method to converge slowly. The problem can be overcome by using modified regula falsi method given regerence 29.

B.2.3 Functional Iteration or Multiple Points Iteration.[28]

Let $f(x)$ be a continuous real-value function with as many derivatives as required. Furthermore, we assume that, in some neighbourhood of the desired root ξ of $f(x) = 0$, the function $f(x)$ has an inverse ; i.e., that $f'(x) \neq 0$. Let $F(y)$ be an inverse function of $f(x)$. If the points y_i; $i=1,2,\ldots, n$, were given, we may approximate $F(y)$ by using Lagrange interpolation formula and then set $y = 0$ we will obtain the root

$$\xi = (-1)^{n-1} \sum_{i=1}^{n} \prod_{\substack{j=1 \\ j\neq i}}^{n} \{y_i/(y_i-y_j)\} \ x_i + (F^{(n)}(\eta)/n!) \prod_{i=1}^{n} y_i \ .$$

and approximated root will be

$$x_{n+1} = (-1)^{n-1} \sum_{i=1}^{n} \prod_{\substack{j=1 \\ j\neq i}}^{n} \{y_i/(y_i-y_j)\} x_i \qquad \text{B.2.1}$$

In practical the Aitken's method of interpolation[25,31] is used instead of Lagrang interpolation polynomial. These two methods of interpolation give the same result but Aitken's method is convenient to calculate by computer.

The procedure of using (B.2.1) to find the root of any equations is that : Using the first n points x_i and n values $f(x_i)$; $i=1,2,\ldots, n$, we can calculate x_{n+1} and $f(x_{n+1})$. Then using the n+1 points x_i and n+1 values of $f(x_i)$; $i=1, 2, \ldots, n+1$, we can calculate the next x_{n+2}, and so on.

It can be sure that, if $F(y)$ exist in an interval that real root ξ lies in, this method will converge rapidly.

Table B.1. ABSCISSAS AND WEIGHT FACTORS FOR GAUSSIAN INTEGRATION[12]

$$\int_{-1}^{+1} f(x)dx = \sum_{i=1}^{n} \alpha_i f(x_i)$$

Abscissas $= \pm x_i$ (Zeros of Legendre Polynomials) Weight Factors $= \alpha_i$

$\pm x_i$		α_i
	$n=2$	
0.57735 02691 89626		1.00000 00000 00000
	$n=3$	
0.00000 00000 00000		0.88888 88888 88889
0.77459 66692 41483		0.55555 55555 55556
	$n=4$	
0.33998 10435 84856		0.65214 51548 62546
0.86113 63115 94053		0.34785 48451 37454
	$n=5$	
0.00000 00000 00000		0.56888 88888 88889
0.53846 93101 05683		0.47862 86704 99366
0.90617 98459 38664		0.23692 68850 56189
	$n=6$	
0.23861 91860 83197		0.46791 39345 72691
0.66120 93864 66265		0.36076 15730 48139
0.93246 95142 03152		0.17132 44923 79170
	$n=7$	
0.00000 00000 00000		0.41795 91836 73469
0.40584 51513 77397		0.38183 00505 05119
0.74153 11855 99394		0.27970 53914 89277
0.94910 79123 42759		0.12948 49661 68870

$\pm x_i$		α_i
	$n=8$	
0.18343 46424 95650		0.36268 37833 78362
0.52553 24099 16329		0.31370 66458 77887
0.79666 64774 13627		0.22238 10344 53374
0.96028 98564 97536		0.10122 85362 90376
	$n=9$	
0.00000 00000 00000		0.33023 93550 01260
0.32425 34234 03809		0.31234 70770 40003
0.61337 14327 00590		0.26061 06964 02935
0.83603 11073 26636		0.18064 81606 94857
0.96816 02395 07626		0.08127 43883 61574
	$n=10$	
0.14887 43389 81631		0.29552 42247 14753
0.43339 53941 29247		0.26926 67193 09996
0.67940 95682 99024		0.21908 63625 15982
0.86506 33666 88985		0.14945 13491 50581
0.97390 65285 17172		0.06667 13443 08688
	$n=12$	
0.12523 34085 11469		0.24914 70458 13403
0.36783 14989 98180		0.23349 25365 38355
0.58731 79542 86617		0.20316 74267 23066
0.76990 26741 94305		0.16007 83285 43346
0.90411 72563 70475		0.10693 93259 95318
0.98156 06342 46719		0.04717 53363 86512

$\pm x_i$		α_i
	$n=16$	
0.09501 25098 37637 440185		0.18945 06104 55068 496285
0.28160 35507 79258 913230		0.18260 34150 44923 588867
0.45801 67776 57227 386342		0.16915 65193 95002 538189
0.61787 62444 02643 748447		0.14959 59888 16576 732081
0.75540 44083 55003 033895		0.12462 89712 55533 872052
0.86563 12023 87831 743880		0.09515 85116 82492 784810
0.94457 50230 73232 576078		0.06225 35239 38647 892863
0.98940 09349 91649 932596		0.02715 24594 11754 094852
	$n=20$	
0.07652 65211 33497 333755		0.15275 33871 30725 850698
0.22778 58511 41645 078080		0.14917 29864 72603 746788
0.37370 60887 15419 560673		0.14209 61093 18382 051329
0.51086 70019 50827 098004		0.13168 86384 49176 626898
0.63605 36807 26515 025453		0.11819 45319 61518 417312
0.74633 19064 60150 792614		0.10193 01198 17240 435037
0.83911 69718 22218 823395		0.08327 67415 76704 748725
0.91223 44282 51325 905868		0.06267 20483 34109 063570
0.96397 19272 77913 791268		0.04060 14298 00386 941331
0.99312 85991 85094 924786		0.01761 40071 39152 118312
	$n=24$	
0.06405 68928 62605 626085		0.12793 81953 46752 156974
0.19111 88674 73616 309159		0.12583 74563 46828 296121
0.31504 26796 96163 374387		0.12167 04729 27803 391204
0.43379 35076 26045 138487		0.11550 56680 53725 601353
0.54542 14713 88839 535658		0.10744 42701 15965 634783
0.64809 36519 36975 569252		0.09761 86521 04113 888270
0.74012 41915 78554 364244		0.08619 01615 31953 275917
0.82000 19859 73902 921954		0.07334 64814 11080 305734
0.88641 55270 04401 034213		0.05929 85849 15436 780746
0.93827 45520 02732 758524		0.04427 74388 17419 806169
0.97472 85559 71309 498198		0.02853 13886 28933 663181
0.99518 72199 97021 360180		0.01234 12297 99987 199547