# Numerical Heat Transfer

# (数值传热学)

## Chapter 10 General Code for 2D Elliptical Fluid Flow and Heat Transfer (2)

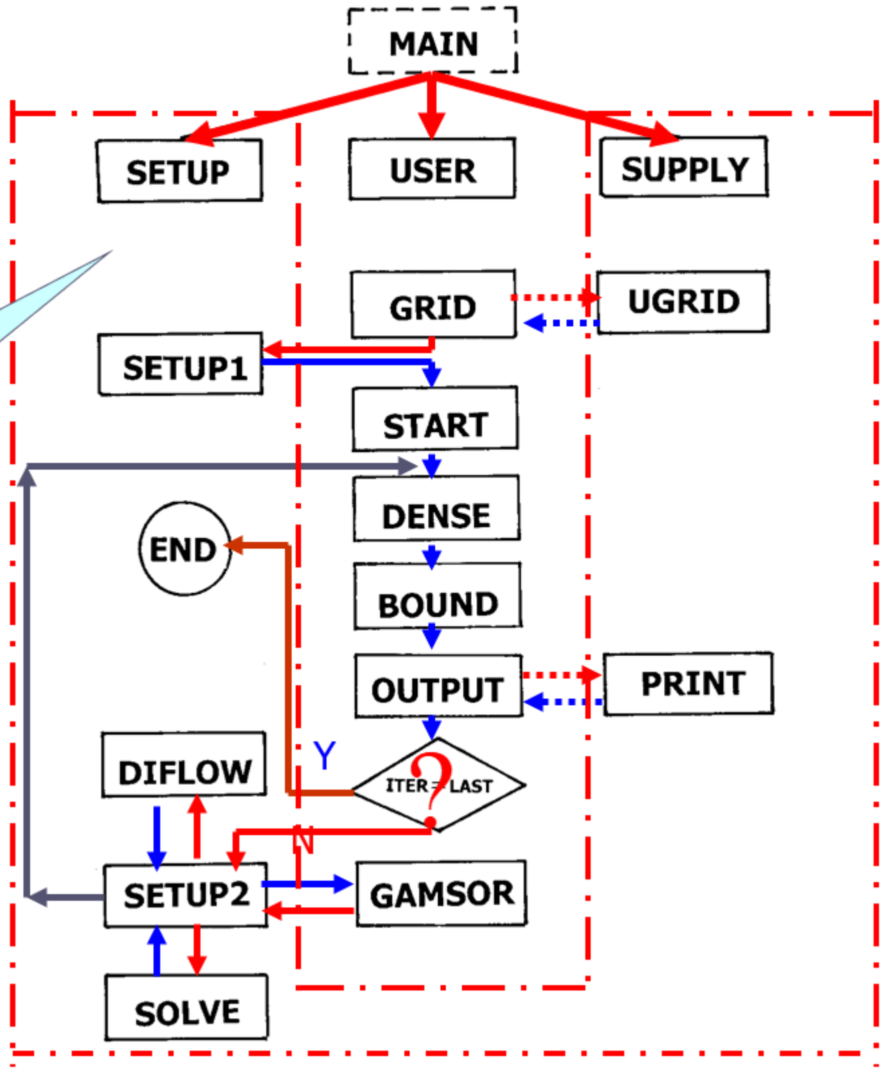
Instructor：Fang, Wen-Zhen; Tao, Wen-Quan

Email: fangwenzhen@xjtu.edu.cn

Key Laboratory of Thermo-Fluid Science & Engineering
Xi'an Jiaotong University

2023-Nov-21

# Main Program: 917 sentences




MAIN Program

Main95.f

ONE.F

```
C ------------------------MAIN PROGRAM---
C************************************
      PROGRAM MAIN
      USE START_L
      IMPLICIT NONE
C************************************
      OPEN(8,FILE='RESULT.TXT')
      CALL GRID
      CALL SETUP1
      CALL START
      DO WHILE(.NOT.LSTOP)
      CALL DENSE
      CALL BOUND
      CALL OUTPUT
      CALL SETUP2
      ENDDO
      CALL OUTPUT
      CLOSE(8)
      STOP
      END
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE DIFLOW
C************************************
      USE START_L
      IMPLICIT NONE
      REAL*8 TEMP
C************************************
```

# Chapter 10 General Code for 2D Elliptical Fluid Flow and Heat Transfer Problems (2)

## 10.6 Methods of application and explanation of Main Program

### 10.6.1 Methods of Code application

### 10.6.2 Explanation of Main Program

**10.6.2.1 MODULE START_L**

**10.6.2.2 PROGRAM MAIN**

**10.6.2.3 SUBROUTINE DIFLOW**

**10.6.2.4 SUBROUTINE SOLVE**

**10.6.2.5 SUBROUTINE SETUP**

**10.6.2.6 SUBROUTINE SUPPLY**

# 10. 6 Methods of Application and Explanation of Main Program

## 10.6.1 Methods of Code application

**1. Establishing complete mathematical formulation and comparing with the standard equation:**

$$\frac{\partial(\rho^{*}\phi)}{\partial t}+div(\rho^{*}\vec{u}\phi)=div(\Gamma_{\Phi}grad\phi)+S_{\phi}^{*}$$

**Determine** $S_{\phi}^{*}$, $\Gamma_{\phi}$, and $\rho_{\phi}^{*}$

**2. Calling (调用) a USER(will be taught in Chapter 11) similar to the problem studied**

3. Using a few nodes, 5～7 in each direction, and setting a small value of LAST, say 3－5, to go through grammatical examination; Then gradually increasing the complexity. For example, for turbulent heat transfer simulation, computing laminar flow first .

4. Making correspondent modifications for the six-ENTRY in USER, according to the problem studied, especially for following parts:

(1) LSOLVE(NF)—for variable NF to be solved setting：

.TRUE.

(2) LPRINT(NF)—for variable NF to be printed out setting:

.TRUE.

(3) TITLE(NF)—for variable NF to be printed out specifying its title (within eight letters).

(4) LBLK(NF)—for variable NF to be solved by block correction setting: .TRUE., otherwise .FALSE., Its default value is .T. .

(5) LAST—Specify iteration number, default value is 5.

(6) NTIMES(NF)—Default value equals 1; for steady nonlinear one, setting: 1 to 2; unsteady linear: 5 to 6

(7) DT—Time step, default value is $10^{30}$

For fully implicit scheme, in the $a_p$-term there is a term of $a_P^0 = \rho\Delta V/\Delta t$, if $\Delta t \to \infty, a_P^0 \to 0$ , leading to steady state results. Default value is for steady case.

(8) RELAX(NF)—Default value is 1.

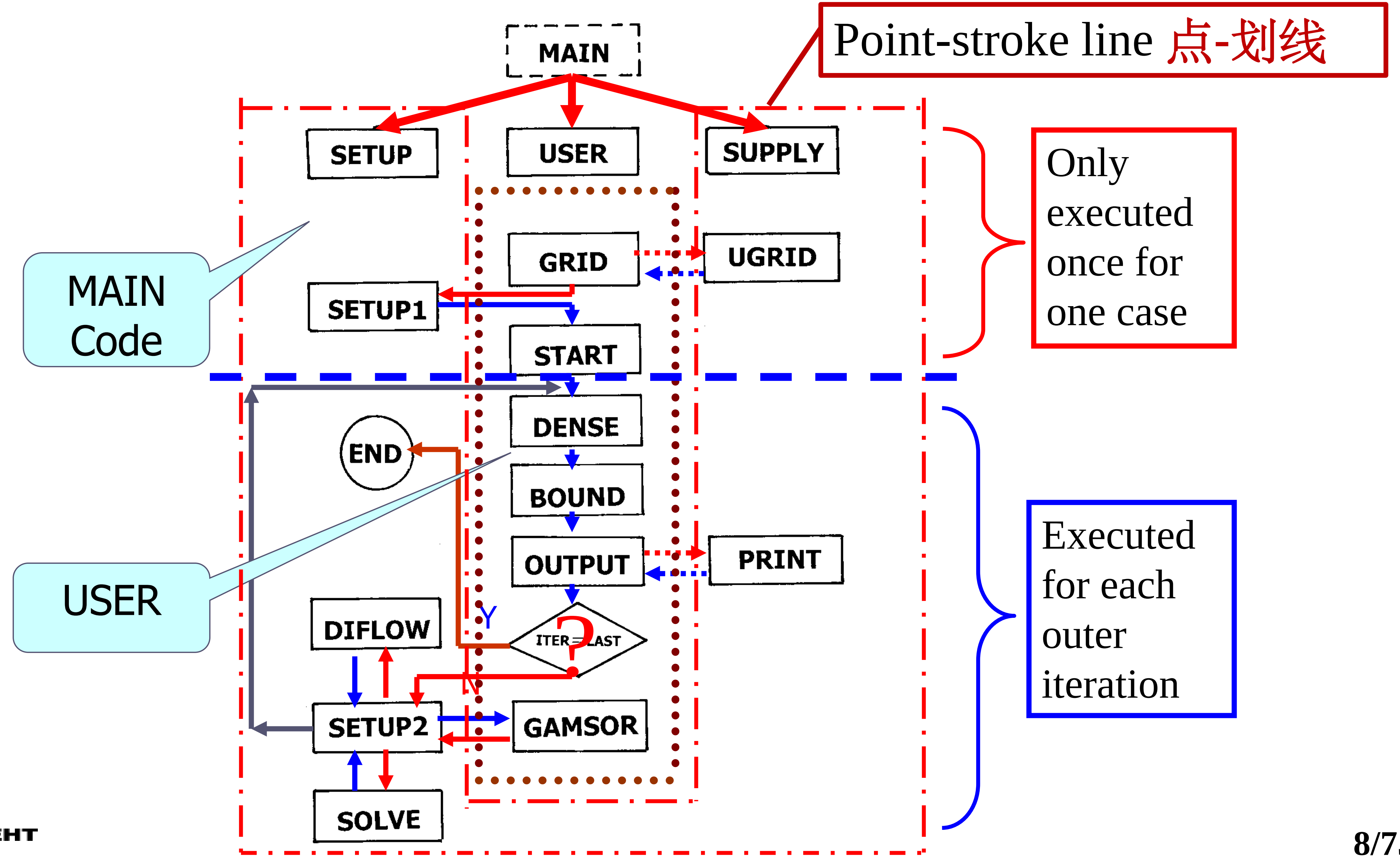
(9) IPREF, JPREF: *I*, *J* of pressure reference point , their default values are 1,1;

**5 Defining a new dependent variable, say *C*(*i*,*j*) ,as follows:**

First defining C(NI,NJ),

then using EQUIVALENCE:

**EQUIVALENCE** (F(1,1,5), C(1,1)).

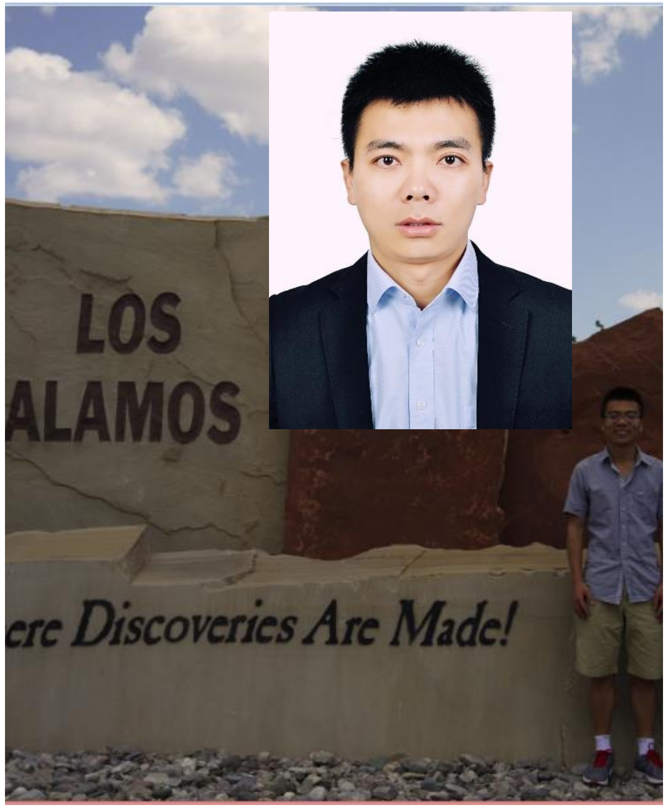

Point-stroke line 点-划线
MAIN
SETUP
USER
SUPPLY
GRID
UGRID
SETUP1
START
MAIN Code
DENSE
END
BOUND
OUTPUT
PRINT
USER
Y
ITER=LAST
?
N
DIFLOW
SETUP2
GAMSOR
SOLVE
Only executed once for one case
Executed for each outer iteration

For the Main Program, these parts will be explained in detail.

1. MODULE START_L
2.---------------------MAIN ------------------
3. SUBROUTINE DIFLOW
4. Structure of SOLVE
5. Block correction
6. Structure of SETUP
7. MODE execution
8. Determination of neighbor coefficients
9. Determination of AP coefficients
10. Structure of SIMPLER
11. Temporal storage of coefficients for SIMPLER
12. Accumulated addition
13. Storage of coefficients of p-equation
14. Nominal density for temperature
15. Iteration=Marching forward
16. Data Format for print out
17. Stream function computation
18. Data print out procedure

# 10-6-2 Explanation of Main programs

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  This computer program was copied from the graduate student course
C  program of the University of Minnesota.  Part of it was re-formulated
C  to meet the local computational environment.  Some inappropriate
C  expressions were also corrected. The program is used only for the
C  teaching purpose.  No part of it may be published. You may use it as a
C  frame to re-develop your own code for research purpose.
C --------Instructor of Numerical Heat Transfer, XJTU,2013.---------------
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C  The current version of the program was updated from Fortran 77 to
C  Fortran 95  by Dr. Yu-Tong Mu , Dr. Li Chen and  Dr.Kong Ling of NHT
C  group of XJTU during 2013.01-04
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C**********************************************************************
```




母玉同　　凌　空　　陈　黎

# 10.6.2.1 MODULE START_L

# (1)-Explained in detail

```
MODULE START_L
 PARAMETER (NI=100,NJ=200,NIJ=NI,NFMAX=10,NFX4=NFMAX+4)
C*************************************
 CHARACTER*8 TITLE(NFX4)
 LOGICAL LSOLVE(NFX4),LPRINT(NFX4),LBLK(NFX4),LSTOP
 REAL*8,DIMENSION(NI,NJ,NFX4)::F    ! One 3D function
 REAL*8,DIMENSION(NI,NJ,6)::COF,COFU,COFV,COFP   ! Four 3D functions
 REAL*8,DIMENSION(NI,NJ)::P,RHO,GAM,CP,CON,AIP,AIM,AJP,AJM,AP
 REAL*8,DIMENSION(NI):: U,V,PC,T,DU,DV,UHAT,VHAT
 REAL*8,DIMENSION(NI):: X,XU,XDIF,XCV,XCVS,XCVI,XCVIP
 REAL*8,DIMENSION(NJ)::Y,YV,YDIF,YCV,YCVS,YCVR,YCVRS,ARX,ARXJ,
&    ARXJP,R,RMN,SX,SXMN
 REAL*8,DIMENSION(NI)::FV,FVP,FX,FXM
 REAL*8,DIMENSION(NJ)::FY,FYM
 REAL*8,DIMENSION(NIJ)::PT,QT     For TDMA in Block correction
 REAL*8 RELAX(NFX3),TIME,DT,XL,YL,RHOCON
 INTEGER*4 NF,NP,NRHO,NGAM,NCP,L1,L2,L3,M1,M2,M3,
&   IST,JST,ITER,LAST,MODE,NTIMES(NFX4),IPREF,JPREF
  REAL*8 SMAX,SSUM
  REAL*8 FLOW,DIFF,ACOF
```

CON: Sc or b

AIP, AIM, AJP, AJM, AP: $a_e, a_w, a_n, a_s, a_p$

(1) Packaging data (封装数据);
(2) Initializing data (数据初始化);
(3) Declaring type of data (声明数据类型).

```
C*****************************************
 EQUIVALENCE(F(1,1,1),U(1,1)),(F(1,1,2),V(1,1)),(F(1,1,3),PC(1,1))
&, (F(1,1,4),T(1,1))
 EQUIVALENCE(F(1,1,11),P(1,1)),(F(1,1,12),RHO(1,1)),(F(1,1,13)
&,GAM(1,1),(F(1,1,14),CP(1,1))
 EQUIVALENCE(COF(1,1,1),CON(1,1)),(COF(1,1,2),AIP(1,1)),
&(COF(1,1,3),AIM(1,1)),(COF(1,1,4),AJP(1,1)),
&(COF(1,1,5),AJM(1,1)),(COF(1,1,6),AP(1,1))
 REAL*8,DIMENSION(NI)::TH,THU,THDIF,THCV,THCVS
 REAL*8 THL
 EQUIVALENCE(X,TH),(XU,THU),(XDIF,THDIF),(XCV,THCV),
&(XCVS,THCVS),(XL,THL)
  DATA LSTOP,LSOLVE,LPRINT/.FALSE.,NFX4*.FALSE., NFX4*.FALSE./
  DATA LBLK/NFX4*.TRUE./
  DATA MODE,LAST,TIME,ITER/1,5,0.,0/
  DATA RELAX,NTIMES/NFX4*1.,NFX4*1/
  DATA DT,IPREF,JPREF,RHOCON,CPCON/1.E+30, 1,1,1.,1./
  END MODULE
```

**Default value!!** (applies to the DATA statements)

```
MODULE module_name
••••••
••••••
••••••
END MODULE
```

Module name is composed of two parts, with a hyphen(-) at bottom in between.

(1) Packaging data (封装数据);
(2) Initializing data (数据初始化);
(3) Declaring type of data (声明数据类型).

# Some explains to this most important module

**REAL*8,DIMENSION(NI,NJ,NFX4)::F**

Real variable 3-D array, array title F，F(NI,NJ,NFX4);

Variable number in three coordinates are NI,NJ and NFX4 respectively;

:: ---is the symbol for separation, separator, to make the declaration of variable type clear;

**REAL*8 SMAX,SSUM**

Real variable of SMAX and SSUM, with length of eight digits;

**INTEGER*4 NF,NP,NRHO**

Integral variable of NF,NP,NRHO, with length of four digits;

**EQUIVALENCE(F(1,1,1),U(1,1)),(F(1,1,2),V(1,1))**

Making the 1st variable of the 3D array F identical to the 2D array U; the same for (F(1,1,2)，V(1,1)))

# 10.6.2.2 PROGRAM MAIN

(2)-Explained in detail

Calling Module

Only Executed once

Executed Many times

```
C*****************************************************************
C-----------------------------------------MAIN ----------------------------------------
C*****************************************************************
      PROGRAM MAIN !Name of main program
      USE START_L        Share the variables defined in the MODULE
      IMPLICIT NONE
C*****************************************************************
      OPEN(8,FILE='RESULT.txt')  ! Result file for output
      CALL GRID      !Grid generation (setup interface positions)
      CALL SETUP1    !Set up 1-D array, not changed in iteration
      CALL START     !Set up initial field
     DO WHILE (.NOT.LSTOP)!  If LSTOP is .F., the NOT .F. is .T., following
                                               four CALLs are executed
      CALL DENSE     !Set up fluid density
      CALL BOUND     !Set up boundary condition
      CALL OUTPUT    !Print out present results
      CALL SETUP2    !Key module: set coefficients and solve ABEqs.
      ENDDO
      CALL OUTPUT !Print out some results
      CLOSE(8)   !Simulation completed  close file RESULT.TXT in Channel 8
      STOP       !Terminate computation
      END      !End of main program
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

For all other scalar variables (including *T*, etc)

$$\frac{\partial(\rho^*\phi)}{\partial t}+div(\rho^*\phi\vec{U})=div(\Gamma_\phi grad\phi)+S_\phi^*$$

$$a_P\phi_P=a_E\phi_E+a_W\phi_W+a_N\phi_N+a_S\phi_S+b$$

$$a_E=D_eA(|P_{\Delta e}|)+[\![-F_e,0]\!] \qquad a_W=D_wA(|P_{\Delta w}|)+[\![F_w,0]\!]$$

$$\frac{a_E}{D_e}$$

$$P_{\Delta e}=\frac{F_e}{D_e}$$

| Scheme | Central difference | Upwind difference |
|---|---|---|
| Definition | $1-0.5\ P_{\Delta e}$ | $1+\ [\![-P_{\Delta e},0]\!]$ |
| **Hybrid** | **Power-law** | **Exponential** |
| $[\![-P_{\Delta e},1-\frac{1}{2}P_{\Delta e},0]\!]$ | $[\![0,(1-0.1P_{\Delta e})^5]\!]+[\![0,-P_{\Delta e}]\!]$ | $\frac{P_{\Delta e}}{\exp(P_{\Delta e})-1}$ |

Determine coefficients using **Subroutine DIFLOW**

# 10.6.2.3 SUBROUTINE DIFLOW

**(3)-Explained in detail**

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

**SUBROUTINE DIFLOW** ! Determine $D \bullet A(|P_\Delta|)$ of power law scheme

Calling Module:

USE START_L ! Share the variables defined in the MODULE START_L
IMPLICIT NONE !The input variables are DIFF and FLOW
REAL*8 TEMP ! Declaration of a temporal real variable TEMP

C*************************************************************

ACOF=DIFF ! $D \bullet A(|P_\Delta|) = D$ (ACOF finally represents $D \bullet A(|P_\Delta|)$)

! 1st return for diffusion case

IF(FLOW== 0.) RETURN ! No flow, only diffusion

TEMP=DIFF-ABS(FLOW)*0.1 ! $D - 0.1|F| = D(1 - 0.1|P_\Delta|)$

ACOF=0. ! $\left\{ A(|P_{\Delta e}|) = \max[0, (1-0.1|P_{\Delta e}|)^5] \begin{cases} 0 & |P_{\Delta e}| > 10 \\ (1-0.1|P_{\Delta e}|)^5 & |P_{\Delta e}| < 10 \end{cases} \right.$

! 2nd return for $|P_{\Delta e}| > 10$

IF(TEMP.<= 0.) RETURN ! $|P_{\Delta e}| > 10$

TEMP=TEMP/DIFF ! $1 - 0.1|P_{\Delta e}|$

! 3rd return for $|P_{\Delta e}| < 10$

ACOF=DIFF*TEMP**5 ! $D \cdot (1 - 0.1|P_{\Delta e}|)^5 = D \bullet A(|P_{\Delta e}|)$

RETURN

END

!In SETUP2: $a_E = D_e A(|P_{\Delta e}|) + [0, -F_e]$

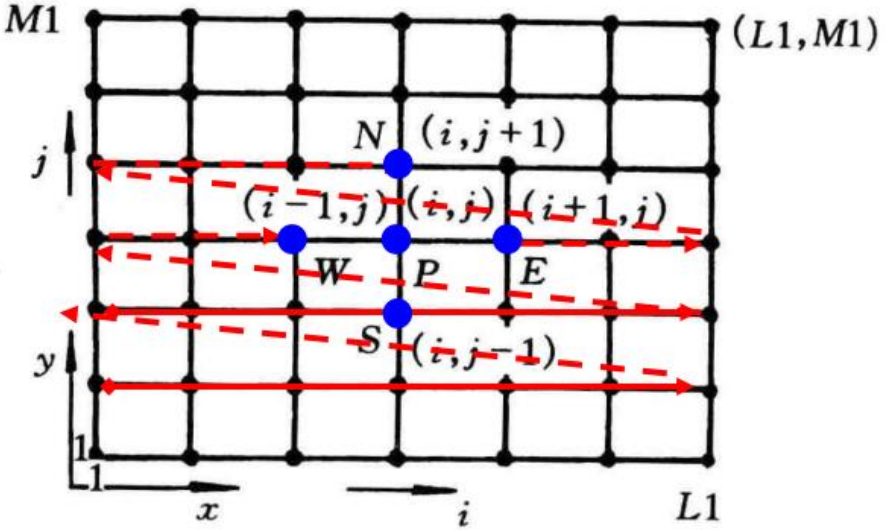
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

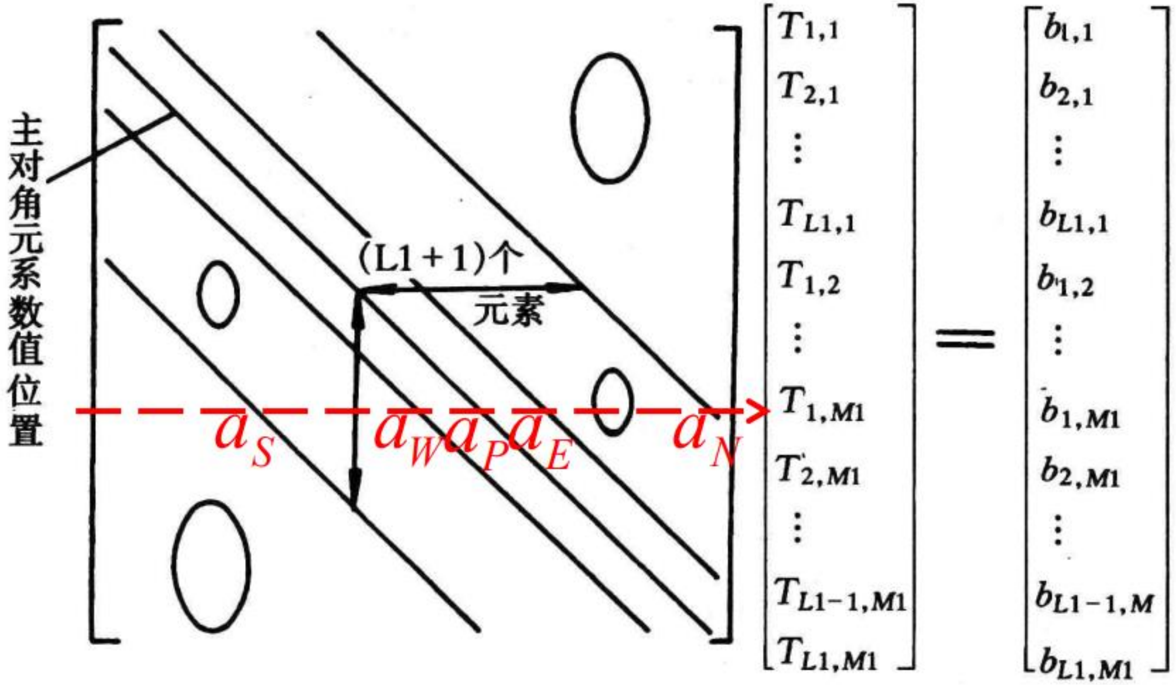
# 10.6.2.4 SUBROUTINE SOLVE

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      SUBROUTINE SOLVE   !ADI  line iteration＋Block correction
      USE START_L
      IMPLICIT NONE
      INTEGER*4 ISTF, JSTF, IT1, IT2, JT1, JT2, NT, N,I,J,II,JJ
      REAL*8 BL, BLP, BLM,  BLC, DENOM, TEMP
C*****************************************************************
```

Calling Module (USE START_L, IMPLICIT NONE)

Declaration of variable type (INTEGER*4 …, REAL*8 …)



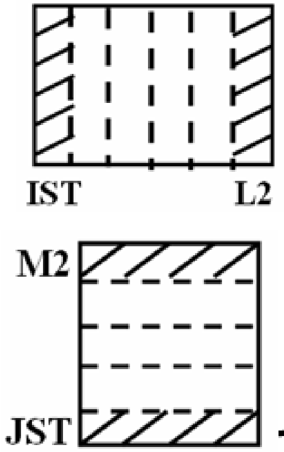



# Structure of SOLVE

## AD---alternative direction

**S O L V E**

DO 999 NT=1， NTIMES（NF）
N=NF
IF (LBLK(NF)) THEN
PT(ISTF)=0.
●●●●●●
13 ENDDO
PT(JSTF)=0.
●●●●●●
23 ENDDO
10 ENDIF




**Two times of block corrections**

**Correction**

Upward line iteration

downward line iteration

Left to right line iteration

Right to left line iteration

999 ENDDO

**Four times of line iterations**

**AD line Iteration**

**(4)-Explained in detail**

# Review on block correction

## 1.Equation for correction:

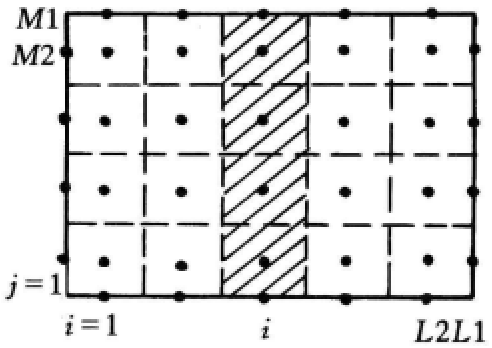
It is required that: $(\phi^*_{i,j}+\bar{\phi}'_i)$ satisfy following eq.

$$\sum_j AP(\phi^*_{i,j}+\bar{\phi}'_i)=\sum_j AIP(\phi^*_{i+1,j}+\bar{\phi}'_{i+1})+\sum_j AIM(\phi^*_{i-1,j}+\bar{\phi}'_{i-1})$$

$$+\sum_j (AJM)(\phi^*_{i,j-1}+\bar{\phi}'_i)$$

$$+\sum_j (AJP)(\phi^*_{i,j+1}+\bar{\phi}'_i)+\sum_j CON$$

$$(i=IST,....L2)$$




IST-solution starting subscript in X-direction; L2-last but one.

Here *AP*,*AIP*,*AIM*, etc. are the symbles adopted in teaching code

# Review on block correction

**(5)-Explained in detail**

$$(BL)\bar{\phi}'_i = (BLP)\bar{\phi}'_{i+1} + (BLM)\bar{\phi}'_{i-1} + BLC, i = IST, \ldots. L2$$

$$BL = \sum_{j=JST}^{M2}(AP) - \sum_{j \neq M2}(AJP) - \sum_{j \neq JST}(AJM) \qquad BLP = \sum_{j=JST}^{M2}(AIP)$$

$$BLM = \sum_{j=JST}^{M2}(AIM) \qquad BLC = \sum_{j=JST}^{M2} CON + \sum_{j=JST}^{M2}(AJP)\phi^*_{i,j+1} + \sum_{j=JST}^{M2}(AJM)\phi^*_{i,j-1}$$

$$+ \sum_{j=JST}^{M2}(AIP)\phi^*_{i+1,j} + \sum_{j=JST}^{M2}(AIM)\phi^*_{i-1,j} - \sum_{j=JST}^{M2}(AP)\phi^*_{i,j}$$

$BL = A, BLP = B,$
$BLM = C$

$$A_i\bar{\phi}'_i = B_i\bar{\phi}'_{i+1} + C_i\bar{\phi}'_{i-1} + D_i, \; i = 1,2,\ldots..M1 \rightarrow \bar{\phi}'_{i-1} = P_{i-1}\bar{\phi}'_i + Q_{i-1}$$

$$P_i = \frac{B_i}{A_i - C_i P_{i-1}}; \quad Q_i = \frac{D_i + C_i Q_{i-1}}{A_i - C_i P_{i-1}}; \qquad P_1 = \frac{B_1}{A_1}; \quad Q_1 = \frac{D_1}{A_1}$$

**DENOM=BL-PT(I-1)*BLM**

**DENOM**

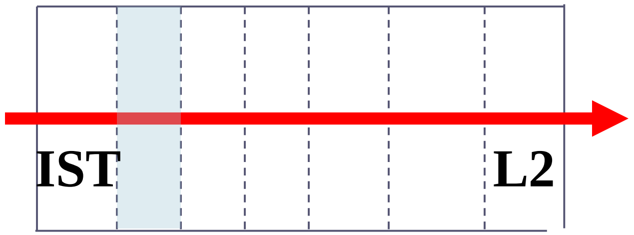
```
C*************************************************************
      ISTF=IST-1
      JSTF=JST-1
      IT1=L2+IST
      IT2=L3+IST
      JT1=M2+JST
      JT2=M3+JST
C*************************************************************
      DO 999  NT=1,NTIMES(NF) ! Solution of algebraic equation
      N=NF  !  NF: 1＝U， 2＝V， 3＝P， ……….
C-------------------------------------------------------------
     IF(LBLK(NF)) THEN !When LBLK is true, execute Block-correction
     PT(ISTF)=0.   !  Coefficient in TDMA
     QT(ISTF)=0.   !  Constant in TDMA
     DO 11 I=IST,L2
     BL=0.     !Initial value in B-correction
     BLP=0.    !Initial value in B-correction
     BLM=0. ! Initial value in B-correction
```

!Temporal integral variables for starting points of DO-loop (ISTF … JT2)

$P_{IST-1}$

$Q_{IST-1}$

I –direction B.Correction.



# (5)-Explained in detail

## In discussion of TDMA: A B C D

$$(BL)\bar{\phi}'_i = (BLP)\bar{\phi}'_{i+1} + (BLM)\bar{\phi}'_{i-1} + BLC, i = IST,..L2$$

```
BLC=0. !Initial value
DO  12 J=JST,M2
BL=BL+AP(I,J)
IF(J /= M2)  BL=BL-AJP(I,J)
IF(J /= JST) BL=BL-AJM(I,J)
BLP=BLP+AIP(I,J)
BLM=BLM+AIM(I,J)
BLC=BLC+CON(I,J)+AIP(I,J)*F(I+1,J,N)+AIM(I,J)*F(I-1,J,N)
& +AJP(I,J)*F(I,J+1,N)+AJM(I,J)*F(I,J-1,N)-AP(I,J)*F(I,J,N)
12 ENDDO
```

Coefficients calculation

$$BL = \sum_{j=JST}^{M2}(AP) - \sum_{j\neq M2}(AJP) - \sum_{j\neq JST}(AJM)$$

$$BLP = \sum_{j=JST}^{M2}(AIP)$$

$$BLM = \sum_{j=JST}^{M2}(AIM)$$

$$BLC = \sum_{j=JST}^{M2} CON + \sum_{j=JST}^{M2}(AJP)\phi^*_{i,j+1} + \sum_{j=JST}^{M2}(AJM)\phi^*_{i,j-1} + \sum_{j=JST}^{M2}(AIP)\phi^*_{i+1,j} + \sum_{j=JST}^{M2}(AIM)\phi^*_{i-1,j} - \sum_{j=JST}^{M2}(AP)\phi^*_{i,j}$$

$$A_i\bar{\phi}'_i = B\bar{\phi}'_{i+1} + C_i\bar{\phi}'_{i-1} + D_i$$

$$\bar{\phi}'_{i-1} = P_{i-1}\bar{\phi}'_i + Q_{i-1}$$

$$P_i = \frac{B_i}{A_i - C_iP_{i-1}};\quad Q_i = \frac{D_i + C_iQ_{i-1}}{A_i - C_iP_{i-1}};$$

DENOM

Elimination (消元)

```
DENOM=BL-PT(I-1)*BLM
IF(ABS(DENOM/BL) <1.E-10) DENOM=1.E25
PT(I)=BLP/DENOM
QT(I)=(BLC+BLM*QT(I-1))/DENOM
11 ENDDO
```

!Ensure a meaningful correction

**Back substitution (回代)**

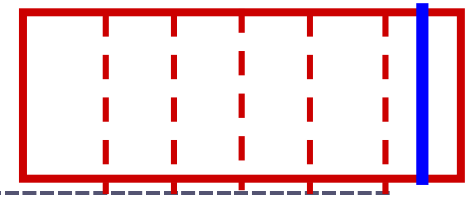
```
BL=0.   (Initial set up)
DO 13 II=IST,L2
I=IT1-II
BL=BL*PT(I)+QT(I)
DO 14 J=JST,M2
F(I,J,N)=F(I,J,N)+BL!
14 ENDDO
13 ENDDO
C--------------------------------------------------------------------------------
PT(JSTF)=0.
QT(JSTF)=0.
DO 21 J=JST,M2
BL=0.
BLP=0.
BLM=0.
BLC=0.
DO 22  I=IST,L2
BL=BL+AP(I,J)
IF(I /= L2) BL=BL-AIP(I,J)
IF(I /= IST) BL=BL-AIM(I,J)
BLP=BLP+AJP(I,J)
```

IT1=L2+IST

I=IT1-II＝L2＋IST-IST＝L2-**Begin**

I=IT1-II＝L2＋IST-L2＝IST-**End**

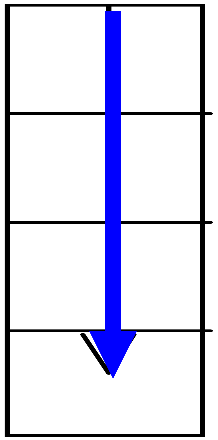
**Correcting by BL for the same column**



$$T_{i-1} = P_{i-1} T_i + Q_{i-1}$$

BL=BL*PT(I)+QT(I)

**TDMA: from L2 to IST**



$$(BL)\bar{\phi}'_j = (BLP)\bar{\phi}'_{j+1} + (BLM)\bar{\phi}'_{j-1} + BLC, J = JST, \ldots M2$$

**Y –direction B-correction**

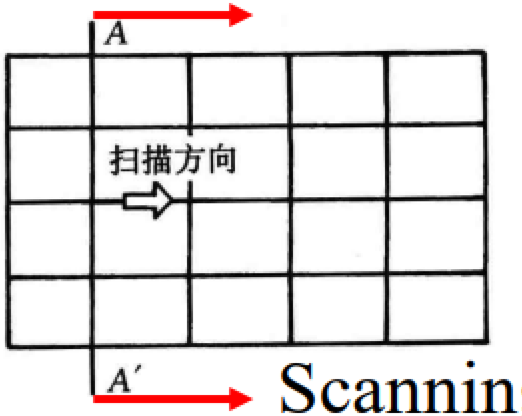
```
    BLM=BLM+AJM(I,J) !
    BLC=BLC+CON(I,J)+AIP(I,J)*F(I+1,J,N)+AIM(I,J)*F(I-1,J,N)
 & +AJP(I,J)*F(I,J+1,N)+AJM(I,J)*F(I,J-1,N)-AP(I,J)*F(I,J,N)
22 ENDDO
    DENOM=BL-PT(J-1)*BLM  !
    IF(ABS(DENOM/BL)<1.E-10) DENOM=1.E25
    PT(J)=BLP/DENOM  !
    QT(J)=(BLC+BLM*QT(J-1))/DENOM
 21 ENDDO
     BL=0.
     DO  23 JJ=JST,M2
     J=JT1-JJ
     BL=BL*PT(J)+QT(J)
     DO 24 I=IST,L2
     F(I,J,N)=F(I,J,N)+BL !Correcting by BL for the same block
 24 ENDDO
 23 ENDDO
 10 ENDIF
C———————————————————————————————————
!  Above is block correction, following is ADI line iteration
```

# line iteration:

Jakob: $a_P\phi_P^{(k+1)} = a_N\phi_N^{(k+1)} + a_S\phi_S^{(k+1)} + [a_E\phi_E^{(k)} + a_W\phi_W^{(k)} + b]$

G-S: $a_P\phi_P^{(k+1)} = a_N\phi_N^{(k+1)} + a_S\phi_S^{(k+1)} + [a_E\phi_E^{(k)} + a_W\phi_W^{(k+1)} + b]$

New $b$ term, $b'$




Scanning （扫描） in E-W direction

At the same line, TDMA is used for direct solution, from line to line iterative method is used.

$$A_i\overline{\phi}_i^{'} = B_i\overline{\phi}_{i+1}^{'} + C_i\overline{\phi}_{i-1}^{'} + D_i,\ i = 1,2,.....M1$$

# Solving in I-direction，scanning in J direction

```
C-------------------------------------------------------------------------------
      DO 90 J=JST,M2
      PT(ISTF)=0.
      QT(ISTF)=F(ISTF,J,N)
      DO 70 I=IST,L2
      DENOM=AP(I,J)-PT(I-1)*AIM(I,J)
      PT(I)=AIP(I,J)/DENOM
      TEMP=CON(I,J)+AJP(I,J)*F(I,J+1,N)+AJM(I,J)*F(I,J-1,N)
      QT(I)=(TEMP+AIM(I,J)*QT(I-1))/DENOM
   70 ENDDO
      DO 80 II=IST,L2
      I=IT1-II  !Recursive
      F(I,J,N)=F(I+1,J,N)*PT(I)+QT(I)
   80 ENDDO
   90 ENDDO
C-------------------------------------------------------------------------------
```

$$AP\phi_{i,j}^{n} = AIP\phi_{i+1,j}^{n} + AIM\phi_{i-1,j}^{n} + b + AJP\phi_{i,j+1}^{n-1} + AJM\phi_{i,j-1}^{n-1}; \quad i = IST...L2$$

! PT=0，QT=given boundary value, 1st kind boundary condition

Elimination (消元): DO 70 loop

$$P_i = \frac{B_i}{A_i - C_i P_{i-1}};$$

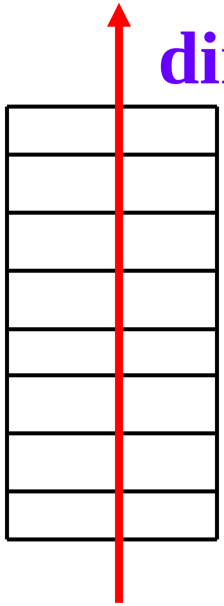
$$Q_i = \frac{D_i + C_i Q_{i-1}}{A_i - C_i P_{i-1}};$$

$$b + AJP\phi_{i,j+1}^{n-1} + AJM\phi_{i,j-1}^{n-1}$$

Back substitution (回代): DO 80 loop

$$T_{i-1} = P_{i-1}T_i + Q_{i-1}$$

Scanning direction



```
C-------------------------------------------------------------------------------
      DO 190 JJ=JST,M3 ！ Solving in I-direction， scanning from t
      J=JT2-JJ  !Starting from JT2 ,rather than from JT1
      PT(ISTF)=0.
      QT(ISTF)=F(ISTF,J,N)
      DO 170 I=IST,L2
      DENOM=AP(I,J)-PT(I-1)*AIM(I,J)
      PT(I)=AIP(I,J)/DENOM
      TEMP=CON(I,J)+AJP(I,J)*F(I,J+1,N)+AJM(I,J)*F(I,J-1,N)
      QT(I)=(TEMP+AIM(I,J)*QT(I-1))/DENOM
  170 ENDDO
      DO 180 II=IST,L2
      I=IT1-II  !Recursive solution
      F(I,J,N)=F(I+1,J,N)*PT(I)+QT(I)
  180 ENDDO
  190 ENDDO
C-------------------------------------------------------------------------------
```

For executing 1st kind B.C. (PT(ISTF)=0., QT(ISTF)=F(ISTF,J,N))

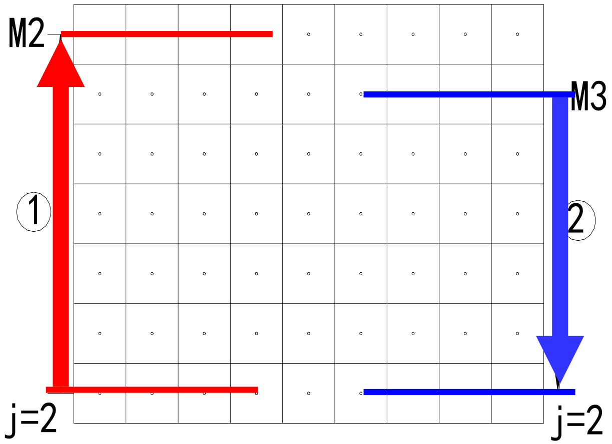
Elimination (消元): DO 170 … QT(I)

Back substitution (回代): DO 180 … F(I,J,N)

$$P_i = \frac{B_i}{A_i - C_i P_{i-1}};$$

$$Q_i = \frac{D_i + C_i Q_{i-1}}{A_i - C_i P_{i-1}};$$

$$b + AJP\phi_{i,j+1}^{n-1} + AJM\phi_{i,j-1}^{n-1}$$



```
C---------------------------------------------------------------------------------
      DO 290 I=IST,L2 !  Solving in J-direction, scanning from left to right
      DO 270 J=JST,M2
      DENOM=AP(I,J)-PT(J-1)*AJM(I,J)
      PT(J)=AJP(I,J)/DENOM
      TEMP=CON(I,J)+AIP(I,J)*F(I+1,J,N)+AIM(I,J)*F(I-1,J,N)
      QT(J)=(TEMP+AJM(I,J)*QT(J-1))/DENOM  !
  270 ENDDO
      DO 280 JJ=JST,M2
      J=JT1-JJ  !Recursive solution
      F(I,J,N)=F(I,J+1,N)*PT(J)+QT(J)  !  P100(a),
  280 ENDDO
  290 ENDDO
C---------------------------------------------------------------------------------
```

```
C-----------------------------------------------------------------------------------
    DO 390 II=IST,L3  !  Solving in J-direction, scanning from right to left
    I=IT2-II
    PT(JSTF)=0.
    QT(JSTF)=F(I,JSTF,N)
    DO 370 J=JST,M2
    DENOM=AP(I,J)-PT(J-1)*AJM(I,J)
    PT(J)=AJP(I,J)/DENOM ,
    TEMP=CON(I,J)+AIP(I,J)*F(I+1,J,N)+AIM(I,J)*F(I-1,J,N)
    QT(J)=(TEMP+AJM(I,J)*QT(J-1))/DENOM
370 ENDDO
    DO 380 JJ=JST,M2
    J=JT1-JJ  !Recursive solution
    F(I,J,N)=F(I,J+1,N)*PT(J)+QT(J)  !  P100(a),
380 ENDDO
390 ENDDO
C*************************************************************************
```

```
C********************************************************************
 999 ENDDO   !（End of solution of ABEqs )

     ENTRY RESET  !（CON， AP are accumulatively used, should be reset）
     DO 400 J=2,M2
     DO 401 I=2,L2
     CON(I,J)=0.
     AP(I,J)=0.
 401 ENDDO
 400 ENDDO
     RETURN
     END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

# 10.6.2.5 SUBROUTINE SETUP

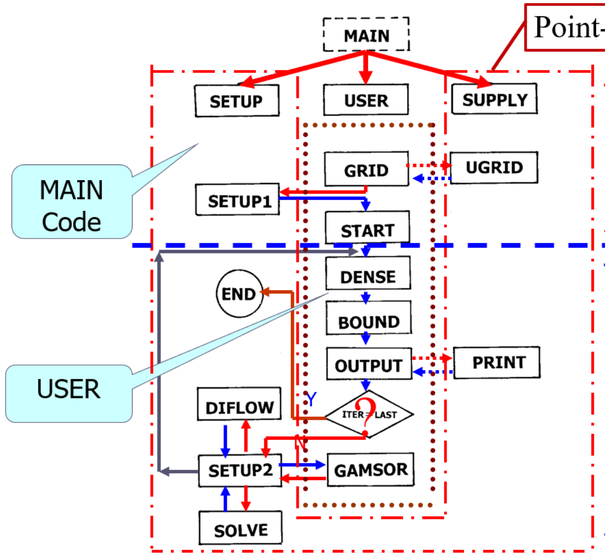
```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
  SUBROUTINE SETUP
C*********************************************************************************************
   USE START_L
   IMPLICIT NONE
   INTEGRER*4 I, J,K,N
   REAL*8  REL, FL, FLM, FLP, GM, GMM, VOL, APT, AREA, SXT,
  1 SXB, ARHO
C*************************************************************************************************
```

```
C***********************************************************************
  1 FORMAT(//15X,'COMPUTATION  IN  CARTESIAN  COORDINATES‘
     !  Print out title for Cartesian coordinate
  2 FORMAT(//15X,'COMPUTATION FOR AXISYMMETRIC SITUATION')
     !  Print out title for cylindrical coordinate
  3 FORMAT(//15X,'COMPUTATION   IN   POLAR   COORDINATES')
     !  Print out title for polar coordinate
  4 FORMAT(14X,40(1H*),//)

C-----------------------------------------------------------------------
```

# Structure of SETUP

# (6)-Explained in detail




**ENTRY SETUP1**

Setup 28 one dimensional geometric parameters;

Setup initial values

**RETURN**

**ENTRY SETUP2**

Coefficient for u equation

Coefficient for v equation

Calculate UHAT and VHAT

Coefficient for pressure equation and solve pressure

Solve u equation and v equation.

Coefficient for pressure correction equation and solve it.

Correction velocity

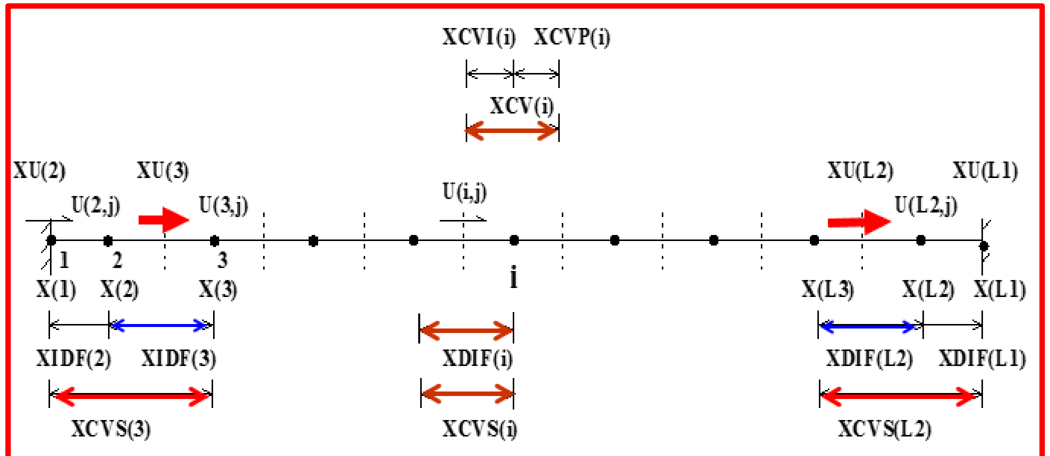
(SIMPLER)

Coefficient for other equation and solve it (from NF=5 to 10 in order)

**RETURN**

S E T U P

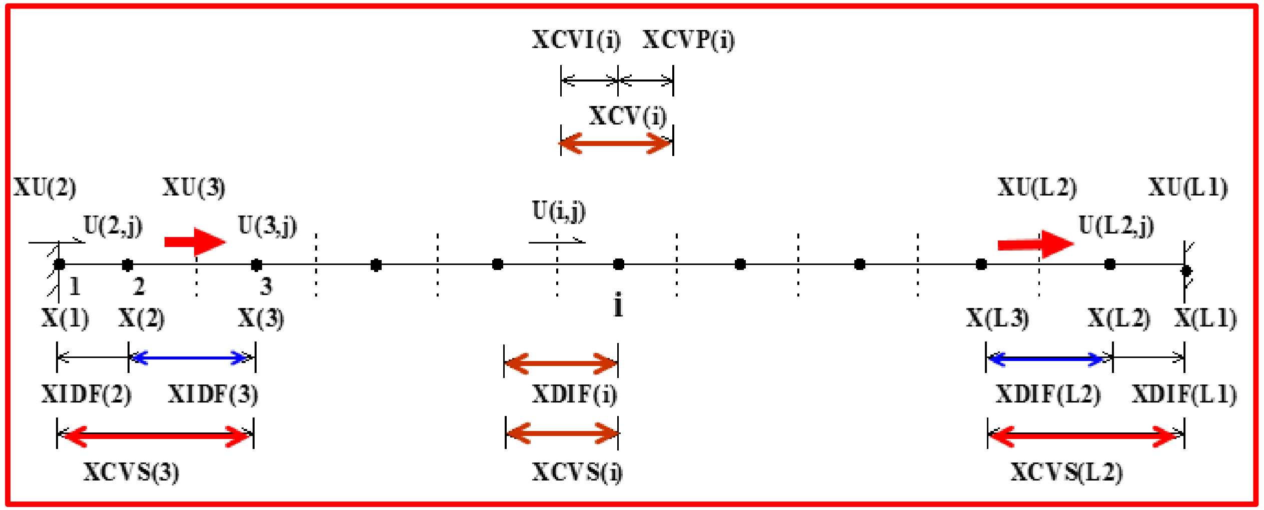
```
C------------------------------------------------------------------------
  ENTRY SETUP1 !Set up 1D arrays, not changed during iteration
  NP=NFMAX+1  ! NFMAX=10， NP＝11
  NRHO=NP +1  ! NRHO＝12
  NGAM=NRHO+1  ! NGAM＝13
  NCP=NGAM+1  ! NCP＝14
  L2=L1-1       ! Set up L2,L3,M2,M3
  L3=L2-1
  M2=M1-1
  M3=M2-1
  X(1)=XU(2)  ! X(1)＝XU(2)＝0
  DO 5 I=2,L2
  X(I)=0.5*(XU(I+1)+XU(I))
5 ENDDO
  X(L1)=XU(L1)
  Y(1)=YV(2)             !Y(1)=YV(2)＝0
  DO 10 J=2,M2
  Y(J)=0.5*(YV(J+1)+YV(J)) !Practice B
10 ENDDO
```

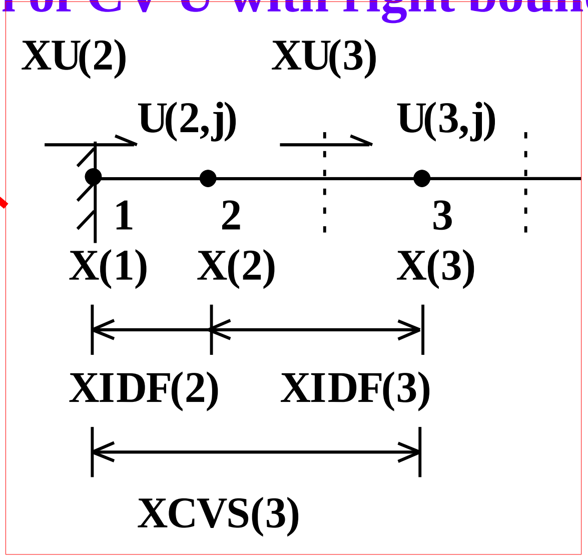



Similarly in Y-direction.

**! Practice B: XU(I) has been set in GRID**

```
      Y(M1)=YV(M1)
      DO 15 I=2,L1
      XDIF(I)=X(I)-X(I-1)
15    ENDDO
      DO 18 I=2,L2
      XCV(I)=XU(I+1)-XU(I)
18    ENDDO
      DO 20 I=3,L2
      XCVS(I)=XDIF(I)     ! Width of CV U (I,J) in x direction
20    ENDDO
      XCVS(3)=XCVS(3)+XDIF(2)!  Width of CV U connected with left boundary
      XCVS(L2)=XCVS(L2)+XDIF(L1) !  Width of CV U with right boundary
      DO 22 I=3,L3
      XCVI(I)=0.5*XCV(I)   !  $(\delta x)_{e^-}$
      XCVIP(I)=XCVI(I)      !  $(\delta x)_{e^+}$
22    ENDDO
      XCVIP(2)=XCV(2)
      XCVI(L2)=XCV(L2)
      DO 35 J=2,M1
      YDIF(J)=Y(J)-Y(J-1)
35    ENDDO
```



(7a)---
Explained in detail

```
    DO 40 J=2,M2
    YCV(J)=YV(J+1)-YV(J)  !Width of main CV in y-direction
40 ENDDO
    DO 45 J=3,M2
    YCVS(J)=YDIF(J)   ! Width of V (I,J) in y-direction
45 ENDDO
    YCVS(3)=YCVS(3)+YDIF(2)
    YCVS(M2)=YCVS(M2)+YDIF(M1)
    IF(MODE= =1) THEN
    DO 52 J=1,M1
    RMN(J)=1.0    ! Nominal radius=1
    R(J)=1.0          ! for Cartesian coordinate
52 ENDDO
    ELSE
    DO 50 J=2,M1          !Cylindrical and polar coordinates
    R(J)=R(J-1)+YDIF(J)  !R(1) has defined
50 ENDDO
    RMN(2)=R(1)
    DO 60 J=3,M2
60 RMN(J)=RMN(J-1)+YCV(J-1)   ! Radius of position of V(I,J)
60 ENDDO
    RMN(M1)=R(M1)
    ENDIF
```

R＝1 for both nodes and interfaces in Cartesian coordinate

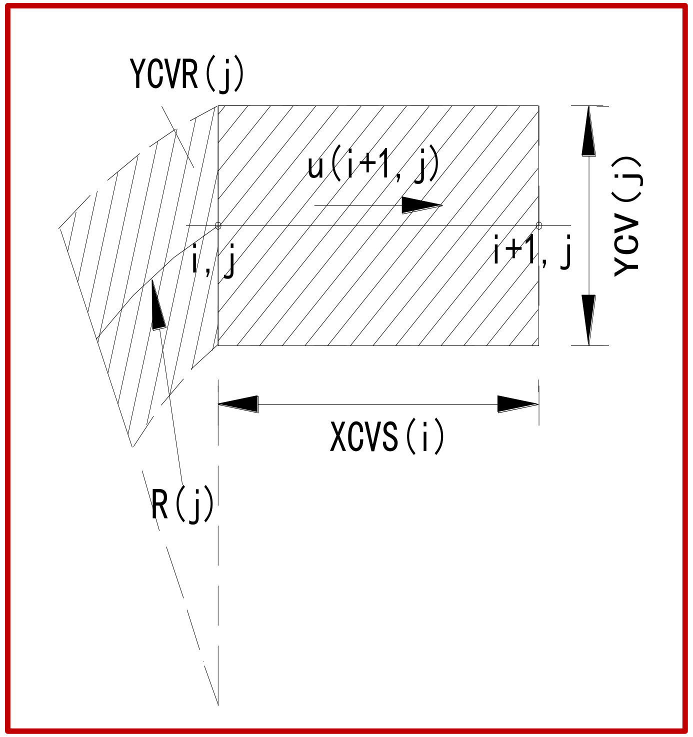
# (7b)--- Explained in detail

```
      DO 57 J=1,M1
      SX(J)=1.
      SXMN(J)=1.
      IF(MODE.== 3) THEN
      SX(J)=R(J)
      IF(J /= 1) SXMN(J)=RMN(J)
      ENDIF
57    ENDDO
      DO 62 J=2,M2
      YCVR(J)=R(J)*YCV(J)
      ARX(J)=YCVR(J)
      IF(MODE = = 3) THEN
      ARX(J)=YCV(J)
62    ENDDO
```

**Set up scaling Factor for polar coordinate**

**Interface starts from J=2**

**!E-W conduction area of CV for three cases, for Cartesian R=1**



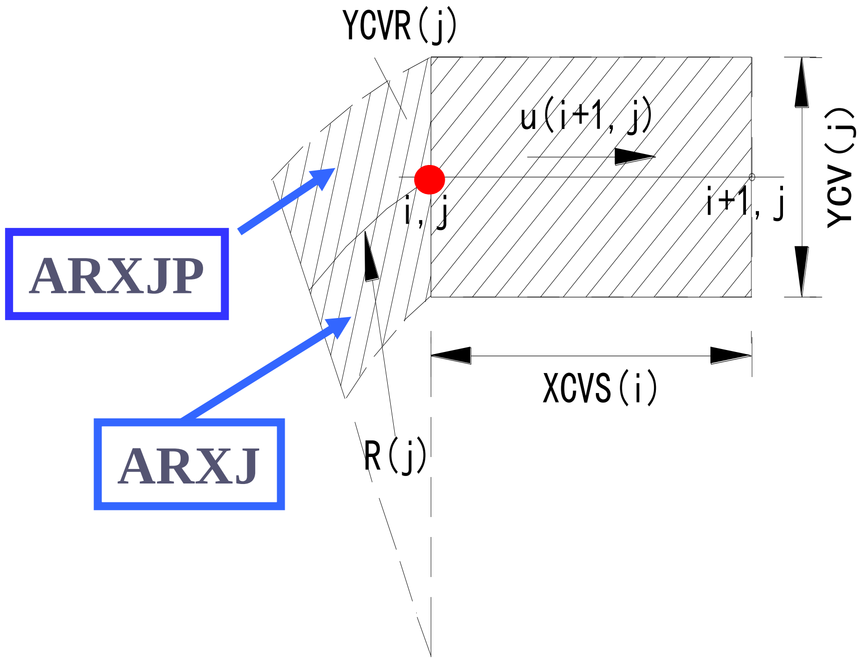
# (7c)--- Explained in detail

```
   DO 64 J=4,M3
   YCVRS(J)=0.5*(R(J)+R(J-1))*YDIF(J)
64 ENDDO
   YCVRS(3)=0.5*(R(3)+R(1))*YCVS(3)
   YCVRS(M2)=0.5*(R(M1)+R(M3))*YCVS(M2)
   IF(MODE = = 2) THEN
   DO 65 J=3,M3
   ARXJ(J)=0.25*(1.+RMN(J)/R(J))*ARX(J)
   ARXJP(J)=ARX(J)-ARXJ(J)
65 ENDDO
   ELSE
   DO 66 J=3,M3
   ARXJ(J)=0.5*ARX(J)
   ARXJP(J)=ARXJ(J)
66 ENDDO
   ENDIF
   ARXJP(2)=ARX(2)
   ARXJ(M2)=ARX(M2)
```

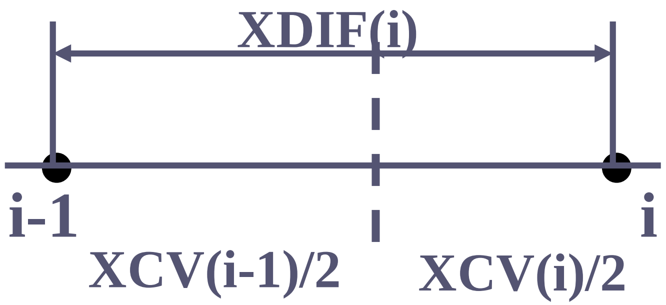
$$ARXJ(J)= \frac{1}{2}(R(j)+RMN(j)) \bullet \frac{YCV(j)}{2} =$$

$$0.25[1+\frac{RMN(j)}{R(j)}] \bullet R(j) \bullet YCV(j) =$$

$$0.25[1+\frac{RMN(j)}{R(j)}] \bullet ARX(j)$$



```
      DO 70 J=3,M3
      FV(J)=ARXJP(J)/ARX(J)
      FVP(J)=1.-FV(J) !Interpolation coefficient
   70 ENDDO
      DO 85 I=3,L2
      FX(I)=0.5*XCV(I-1)/XDIF(I) !Interpolation  in x-direction
      FXM(I)=1.-FX(I)
   85 ENDDO
      FX(2)=0.
      FXM(2)=1.
      FX(L1)=1.
      FXM(L1)=0.
      DO 90 J=3,M2
      FY(J)=0.5*YCV(J-1)/YDIF(J) ! Interpolation in y-direction
      FYM(J)=1.-FY(J)
   90 ENDDO
      FYM(2)=1.
      FY(M1)=1.
      FYM(M1)=0.
C
CON,AP,U,V,RHO,PC AND P ARRAYS ARE INITIALIZED HERE
```




$$\phi_{i-1/2} = \phi_{i-1}\frac{\text{XCV(i)/2}}{\text{XDIF(i)}} + \phi_i \frac{\text{XCV(i-1)/2}}{\text{XDIF(i)}} = \phi_{i-1} FXM(i) + \phi_i FX(i)$$

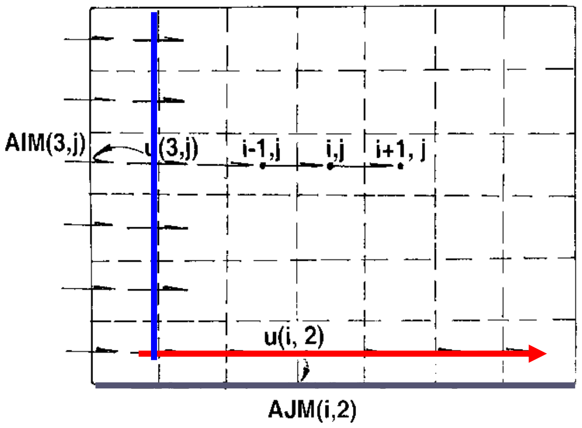
The first letter C is also used to indicate that this is an explanation line

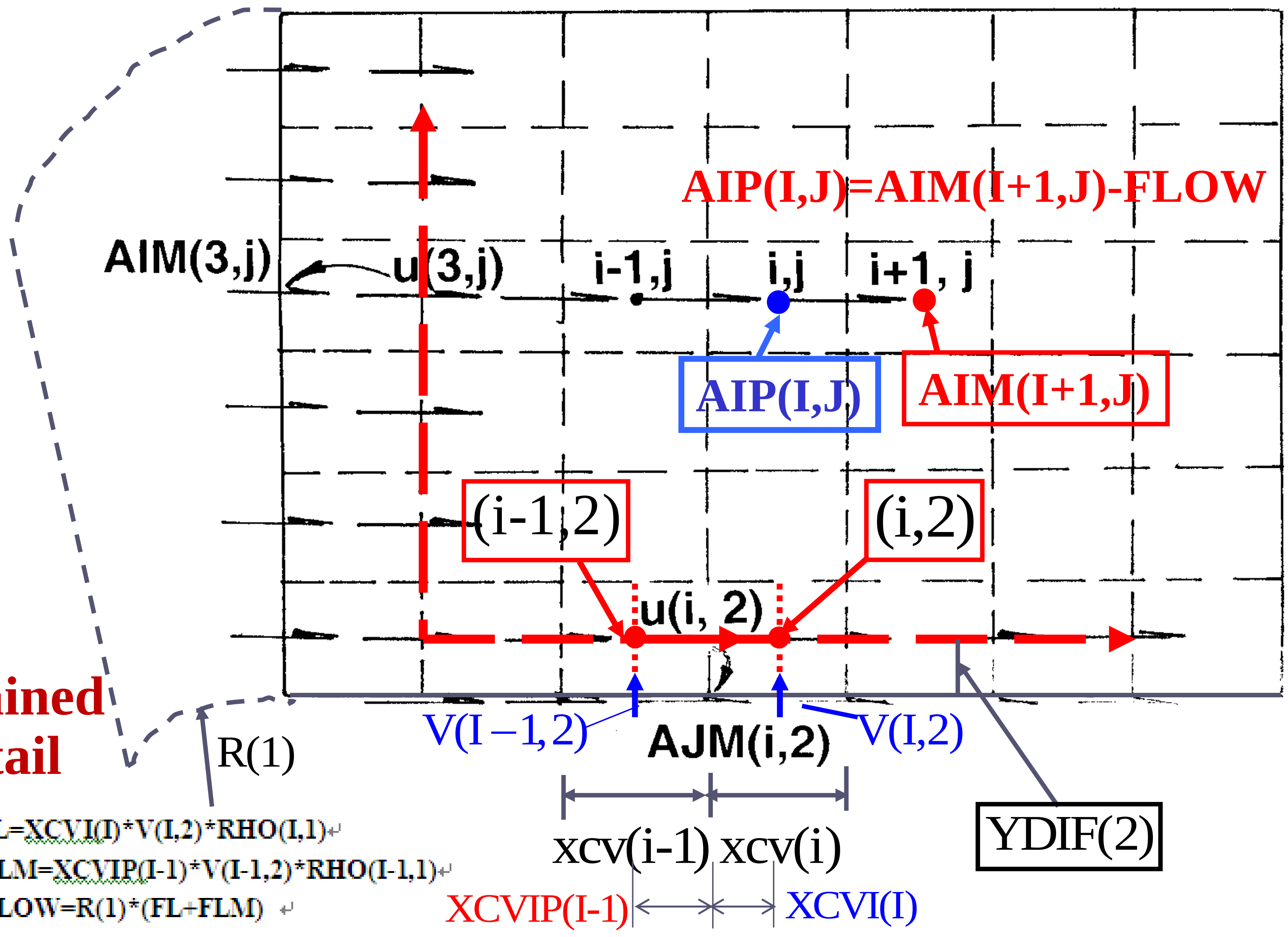
```
DO 96 J=1,M1
DO 95 I=1,L1
PC(I,J)=0.
U(I,J)=0.
V(I,J)=0.
CON(I,J)=0.
AP(I,J)=0.
RHO(I,J)=RHOCON
CP（I,J)=CPCON
P(I,J)=0.
95 ENDDO
96 ENDDO
IF(MODE= =1) PRINT 1
IF(MODE= =1) WRITE(8,1)
IF(MODE= =2) PRINT 2
IF(MODE= =2) WRITE(8,2)
IF(MODE= =3) PRINT 3
IF(MODE= =3) WRITE(8,3)
PRINT 4
WRITE(8,4)
RETURN
```

Set up initial fields for iteration

Print out coordinate title of out put data

```
C-----------------------------------------------------------------------
   ENTRY SETUP2
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
COEFFICIENTS FOR THE U EQUATION
      NF=1  ! NF=1:  U;  NF=2:  V;  NF=3:  P', P;  NF=NP:  P
      IF(LSOLVE(NF)) THEN       !
      IST=3
      JST=2
      CALL GAMSOR
      REL=1.-RELAX(NF) ! （U） underrelaxation
      DO 102 I=3,L2  !Coefficient of south boundary
      FL=XCVI(I)*V(I,2)*RHO(I,1)
      FLM=XCVIP(I-1)*V(I-1,2)*RHO(I-1,1)
      FLOW=R(1)*(FL+FLM) ! Flow rate through south interface
      DIFF=R(1)*(XCVI(I)*GAM(I,1)+XCVIP(I-1)*GAM(I-1,1))/YDIF(2)
      CALL DIFLOW !With DIFF and FLOW at hand, CALL DIFLOW to get D.A(|P|);
      AJM(I,2)=ACOF+AMAX1(0.,FLOW)    Coefficient a_S
102 ENDDO
```






**(8)---Explained in detail**

```
FL=XCVI(I)*V(I,2)*RHO(I,1)
FLM=XCVIP(I-1)*V(I-1,2)*RHO(I-1,1)
FLOW=R(1)*(FL+FLM)
DIFF=R(1)*(XCVI(I)*GAM(I,1)+XCVIP(I-1)*GAM(I-1,1))/YDIF(2)
```

# Explanation of DIFF(Diffusion conductance)

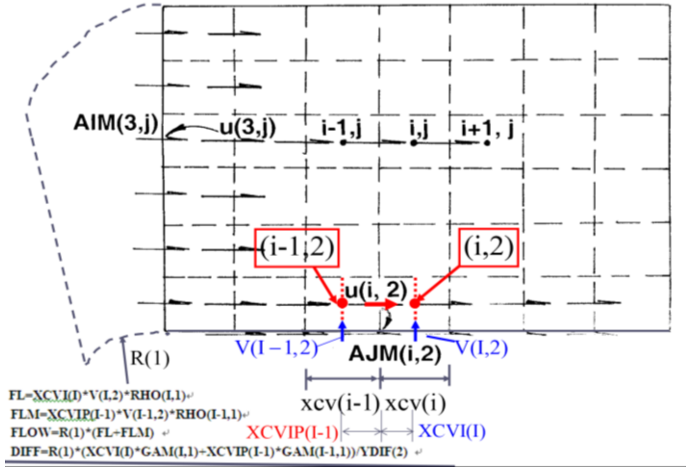
**DIFF=R(1)*(XCVI(I)*GAM(I,1)+XCVIP(I-1)*GAM(I-1,1))/YDIF(2)**

## For Cartesian coordinates:

$$D_{n-e}=\frac{(\delta x)_{e^-}}{\frac{(\delta y)_n}{\Gamma_n}}+\frac{(\delta x)_{e^+}}{\frac{(\delta y)_n}{\Gamma_{ne}}}=\frac{(\delta x)_{e^-}\Gamma_n+(\delta x)_{e^+}\Gamma_{ne}}{(\delta y)_n}$$

## For Cylindrical coordinates:

$$D_{n-e}=\mathrm{R1}\left[\frac{(\delta x)_{e^-}\Gamma_n+(\delta x)_{e^+}\Gamma_{ne}}{(\delta y)_n}\right]$$



```
DO 103 J=2,M2
FLOW=ARX(J)*U(2,J)*RHO(1,J)
DIFF=ARX(J)*GAM(1,J)/(XCV(2)*SX(J))
CALL DIFLOW        ! Get A(|P|)
AIM(3,J)=ACOF+AMAX1(0.,FLOW)  !Coefficient aW
DO 104 I=3,L2
IF(I = = L2) THEN
FLOW=ARX(J)*U(L1,J)*RHO(L1,J)
DIFF=ARX(J)*GAM(L1,J)/(XCV(L2)*SX(J)) ! DW
ELSE
FL=U(I,J)*(FX(I)*RHO(I,J)+FXM(I)*RHO(I-1,J))
FLP=U(I+1,J)*(FX(I+1)*RHO(I+1,J)+FXM(I+1)*RHO(I,J))
FLOW=ARX(J)*0.5*(FL+FLP)
DIFF=ARX(J)*GAM(I,J)/(XCV(I)*SX(J))
ENDIF
CALL DIFLOW  !  A(|P|)
AIM(I+1,J)=ACOF+AMAX1(0.,FLOW)
AIP(I,J)=AIM(I+1,J)-FLOW  ! Relationship between coefficients
```

$$D \bullet A(|P_\Delta|) + [\![ 0, F ]\!]$$

```
 IF(J = = M2) THEN
 FL=XCVI(I)*V(I,M1)*RHO(I,M1)
 FLM=XCVIP(I-1)*V(I-1,M1)*RHO(I-1,M1)
 DIFF=R(M1)*(XCVI(I)*GAM(I,M1)+XCVIP(I-1)*GAM(I-1,M1))/YDIF(M1)
 ELSE
 FL=XCVI(I)*V(I,J+1)*(FY(J+1)*RHO(I,J+1)+FYM(J+1)*RHO(I,J))
 FLM=XCVIP(I-1)*V(I-1,J+1)*(FY(J+1)*RHO(I-1,J+1)+FYM(J+1)*
& RHO(I-1,J))
  GM=GAM(I,J)*GAM(I,J+1)/(YCV(J)*GAM(I,J+1)+YCV(J+1)*GAM(I,J)+
& 1.0E-30)*XCVI(I)
  GMM=GAM(I-1,J)*GAM(I-1,J+1)/(YCV(J)*GAM(I-1,J+1)+YCV(J+1)*
& GAM(I-1,J)+1.E-30)*XCVIP(I-1)
  DIFF=RMN(J+1)*2.*(GM+GMM)
  ENDIF
  FLOW=RMN(J+1)*(FL+FLM)
  CALL DIFLOW  ! A(|P|)
 AJM(I,J+1)=ACOF+AMAX1(0.,FLOW)     ! Coefficient a_s
 AJP(I,J)=AJM(I,J+1)-FLOW   !Relationship between coefficients
```

**(9)---Explained in detail**

```
VOL=YCVR(J)*XCVS(I)        !Volume of velocity CV
APT=(RHO(I,J)*XCVI(I)+RHO(I-1,J)*XCVIP(I-1))
&/(XCVS(I)*DT)  ! Unsteady term ρ/△t;  DT---△t;
AP(I,J)=AP(I,J)-APT  ! AP (I,J)  at right side is Sp
CON(I,J)=CON(I,J)+APT*U(I,J)
AP(I,J)=(-AP(I,J)*VOL+AIP(I,J)+AIM(I,J)+AJP(I,J)+AJM(I,J))
&/RELAX(NF)  !Underrelaxation is organized during solution procedure
CON(I,J)=CON(I,J)*VOL+REL*AP(I,J)*U(I,J)  !  REL=1-α
DU(I,J)=VOL/(XDIF(I)*SX(J))  !  To get flow area
DU(I,J)=DU(I,J)/AP(I,J)  !  de in velocity correction
104 ENDDO
103 ENDDO
```

$$a_P^0 = \frac{\rho_P \Delta V}{\Delta t}$$

$$! \; d_e = A_e / a_e$$

$$b = S_c \Delta V + a_p^0 \phi_p^0 + (1-\alpha)\frac{a_P}{\alpha}\phi_P^0$$

$$a_p = \left(\sum a_{nb} + \rho_P \Delta V / \Delta t - S_p \Delta V\right) / \alpha$$

**! Come here we have finished the coefficients calculation for *u* velocity and should store them temporary to leave COF empty for calculation coefficients for velocity *v*.**

## ----------Review of SIMPLER algorithm-------------

1. **Assuming initial fields, determine coefficients of discretized *u*, *v* eqs.;**
2. **Calculating pseudo-velocity** $u, \tilde{v}$ ;

**(10)---Explained in detail**

$$a_e u_e = \sum a_{nb} u_{nb} + b + A_e (p_P - p_E)$$

$$u_e = \sum \frac{a_{nb} u_{nb} + b}{a_e} + \frac{A_e}{a_e}(p_P - p_E) \rightarrow u_e = \tilde{u}_e + \frac{A_e}{a_e}(p_P - p_E)$$

**and Solving pressure equation, obtaining** $p^*$ ;

$$a_P p_P = a_E p_E + a_W p_W + a_N p_N + a_S p_S + b$$

$$a_P = a_E + a_W + a_N + a_S$$

$$a_E = d_e A_e \rho_e \quad a_W = d_w A_w \rho_w \quad a_n = d_n A_n \rho_n \quad a_S = d_s A_s \rho_s$$

$$b = [(\rho u)_w - (\rho u)_s] A_e + [(\rho \tilde{v})_s - (\rho \tilde{v})_n] A_n$$

**Coefficients of *u*, *v* momentum equations are needed for determining coefficients of pressure equation.**

**3. Solving momentum equations based on** $p^*$ **, obtaining** $u^*, v^*$

**4. Solving pressure correction equation based on** $u^*, v^*$ **, obtaining** $p'$

**In pressure equation:** $b = [(\rho u)_w - (\rho u)_s]A_e + [(\rho \tilde{v})_s - (\rho \tilde{v})_n]A_n$

**In pressure correction equation:** $b = [(\rho u^*)_w - (\rho u^*)_s]A_e + [(\rho v^*)_s - (\rho v^*)_n]A_n$

5. **Correcting velocity** $u = u^* + u'; v = v^* + v'$, **where** $u'$ and $v'$ **are determined based on** $p'$

$$u_e' = \frac{A_e}{a_e}(p_P' - p_E') = d_e(p_P' - p_E')$$

6. **Taking the updated velocity , repeating steps 1-6, until convergence is reached.**

**----------End of Review of SIMPLER algorithm------------**

```
MODULE START_L
PARAMETER (NI=100,NJ=200,NIJ=NI,NFMAX=10,NFX4=NFMAX+4)
C******************************************
CHARACTER*8 TITLE(NFX4)
LOGICAL LSOLVE(NFX4),LPRINT(NFX4),LBLK(NFX4),LSTOP
REAL*8,DIMENSION(NI,NJ,NFX4)::F
REAL*8,DIMENSION(NI,NJ,6)::COF,COFU,COFV,COFP
REAL*8,DIMENSION(NI,NJ)::P,RHO,GAM,CP,CON,AIP,AIM,AJP,AJM,AP
```

```
C****************************************
 EQUIVALENCE(F(1,1,1),U(1,1)),(F(1,1,2),V(1,1)),(F(1,1,3),PC(1,1))
1, (F(1,1,4),T(1,1))
 EQUIVALENCE(F(1,1,11),P(1,1)),(F(1,1,12),RHO(1,1)),(F(1,1,13)
1,GAM(1,1)),(F(1,1,14),CP(1,1))
 EQUIVALENCE(COF(1,1,1),CON(1,1)),(COF(1,1,2),AIP(1,1)),
1(COF(1,1,3),AIM(1,1)),(COF(1,1,4),AJP(1,1)),
2(COF(1,1,5),AJM(1,1)),(COF(1,1,6),AP(1,1))
 REAL*8,DIMENSION(NI)::TH,THU,THDIF,THCV,THCVS
 REAL*8 THL
 EQUIVALENCE(X,TH),(XU,THU),(XDIF,THDIF),(XCV,THCV),
1(XCVS,THCVS),(XL,THL)
  DATA LSTOP,LSOLVE,LPRINT/.FALSE.,NFX4*.FALSE., NFX4*.FALSE./
  DATA LBLK/NFX4*.TRUE./
  DATA MODE,LAST,TIME,ITER/1,5,0.,0/
  DATA RELAX,NTIMES/NFX4*1.,NFX4*1/
  DATA DT,IPREF,JPREF,RHOCON,CPCON/1.E+30, 1,1,1.,1./
  END MODULE
```

COFU(IST:L2, JST:M2, 1:6)=COF(IST:L2,JST:M2,1:6) ! Transfer the coefficients

! Store coefficients of *U* temporary as follows:

Explain

| COF(I,J,1) | COF(I,J,2) | COF(I,J,3) | COF(I,J,4) | COF(I,J,5) | COF(I,J,6) |
|---|---|---|---|---|---|
| CON（I,J） | AIP(I,J) | AIM(I,J) | AJP(I,J) | AJM(I,J) | AP(I,J) |

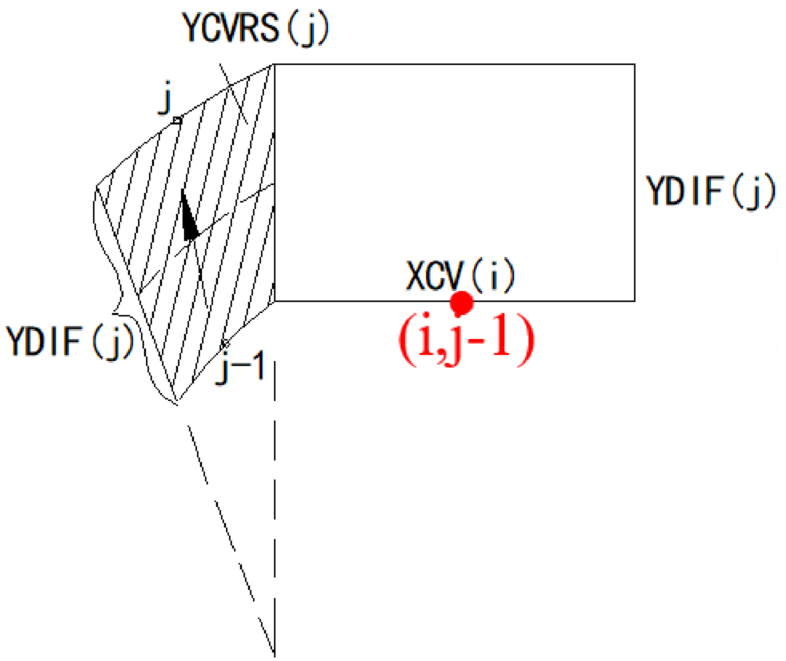
! In SIMPLER to solve pressure eq., coefficients of both *u*-eq. and
! *v*-eq. are needed. Only *u*-coefficients are not enough. Thus,
! *u*-coefficients are temporary stored, and *v*-eq. coefficients
! are computed by using array COF(I,J)

COEFFICIENTS FOR THE V EQUATION- （Determine coefficients of V

```
NF=2          !
CALL RESET  !Set zero values for AP(I,J),CON(I,J)
IST=2
JST=3
CALL GAMSOR
REL=1.-RELAX(NF)
```

(11)---Explained in detail

```
      DO 202 I=2,L2
      AREA=R(1)*XCV(I)
      FLOW=AREA*V(I,2)*RHO(I,1)
      DIFF=AREA*GAM(I,1)/YCV(2)
      CALL DIFLOW
      AJM(I,3)=ACOF+AMAX1(0.,FLOW)  ! a_S
202 ENDO
      DO 203 J=3,M2
      FL=ARXJ(J)*U(2,J)*RHO(1,J)
      FLM=ARXJP(J-1)*U(2,J-1)*RHO(1,J-1)
      FLOW=FL+FLM
      DIFF=(ARXJ(J)*GAM(1,J)+ARXJP(J-1)*GAM(1,J-1))/(XDIF(2)*SXMN(J))
      CALL DIFLOW
      AIM(2,J)=ACOF+AMAX1(0.,FLOW) ! a_W
      DO 204 I=2,L2
      IF(I= = L2) THEN
      FL=ARXJ(J)*U(L1,J)*RHO(L1,J)
      FLM=ARXJP(J-1)*U(L1,J-1)*RHO(L1,J-1)
      DIFF=(ARXJ(J)*GAM(L1,J)+ARXJP(J-1)*GAM(L1,J-
     & -1))/(XDIF(L1)*SXMN(J))
```



```
ELSE
FL=ARXJ(J)*U(I+1,J)*(FX(I+)*RHO(I+1,J)+FXM(I+1)*RHO(I,J))
FLM=ARXJP(J-1)*U(I+1,J-1)*(FX(I+1)*RHO(I+1,J-1)+FXM(I+1)*RHO(I,J-1))
GM=GAM(I,J)*GAM(I+1,J)/(XCV(I)*GAM(I+1,J)+XCV(I+1)*GAM(I,J)+
& 1.E-30)*ARXJ(J)
GMM=GAM(I,J-1)*GAM(I+1,J-1)/(XCV(I)*GAM(I+1,J-1)+XCV(I+1)*
& GAM(I,J-1)+1.0E-30)*ARXJP(J-1)
DIFF=2.*(GM+GMM)/SXMN(J)
ENDIF
FLOW=FL+FLM
CALL DIFLOW
AIM(I+1,J)=ACOF+AMAX1(0.,FLOW)        ! a_W
AIP(I,J)=AIM(I+1,J)-FLOW  !Relationship between coefficients
IF (J= =M2) THEN
AREA=R(M1)*XCV(I)
FLOW=AREA*V(I,M1)*RHO(I,M1)
```

```
DIFF=AREA*GAM(I,M1)/YCV(M2)
ELSE
AREA=R(J)*XCV(I)
FL=V(I,J)*(FY(J)*RHO(I,J)+FYM(J)*RHO(I,J-1))*RMN(J)
FLP=V(I,J+1)*(FY(J+1)*RHO(I,J+1)+FYM(J+1)*RHO(I,J))*RMN(J+1)
FLOW=(FV(J)*FL+FVP(J)*FLP)*XCV(I)
DIFF=AREA*GAM(I,J)/YCV(J)
ENDIF
CALL DIFLOW
AJM(I,J+1)=ACOF+AMAX1(0.,FLOW) ! a_S
AJP(I,J)=AJM(I,J+1)-FLOW          !Relationship
VOL=YCVRS(J)*XCV(I)        !Volume of V- CV
SXT=SX(J)
```

```
 APT=(ARXJ(J)*RHO(I,J)*0.5*(SXT+SXMN(J))+ARXJP(J-1)*RHO(I,J-1)*
&0.5*(SXB+SXMN(J)))/(YCVRS(J)*DT)
 AP(I,J)=AP(I,J)-APT
 CON(I,J)=CON(I,J)+APT*V(I,J)
 AP(I,J)=(-AP(I,J)*VOL+AIP(I,J)+AIM(I,J)+AJP(I,J)+AJM(I,J))
&/RELAX(NF)
 CON(I,J)=CON(I,J)*VOL+REL*AP(I,J)*V(I,J)
 DV(I,J)=VOL/YDIF(J)
 DV(I,J)=DV(I,J)/AP(I,J)
204 ENDDO
203 ENDDO
 COFV(IST:L2,JST:M2,1:6)=COF(IST:L2,JST:M2,1,6)
! Store coefficients of V-eq. to compute coefficients of P-equation
CALCULATE UHAT AND VHAT !
 DO 150 J=2,M2
 DO 151 I=3,L2
 UHAT(I,J)=(COFU(I,J,2)*U(I+1,J)+COFU(I,J,3)*U(I-1,J)+COFU(I,J,4)
& *U(I,J+1)+COFU(I,J,5)*U(I,J-1)+COFU(I,J,1))/COFU(I,J,6)
```

! Compute $\tilde{u}, \tilde{v}$

$$u_e = \sum \frac{a_{nb} u_{nb} + b}{a_e}$$

```
151 ENDDO
150 ENDDO
      DO 250 J=3,M2
      DO 251 I=2,L2
      VHAT(I,J)=(COFV(I,J,2)*V(I+1,J)+COFV(I,J,3)*V(I-1,J)+COFV(I,J,4)
     & *V(I,J+1)+COFV(I,J,5)*V(I,J-1)+COFV(I,J,1))/COFV(I,J,6)
251 ENDDO
250 ENDDO
COEFFICIENTS FOR THE PRESSURE EQUATION----------------------------
      NF=3
      CALL RESET
      IST=2
      JST=2
      CALL GAMSOR
      DO 410 J=2,M2,
      DO 411 I=2,L2
      VOL=YCVR(J)*XCV(I)
      CON(I,J)=CON(I,J)*VOL
411 ENDDO
410 ENDDO
```

!In the discretized pressure equation, the source term is

! $b=[(\rho u)_w-(\rho u)_e]A_e+[(\rho\tilde{v})_s-(\rho\tilde{v})_n]A_n$

!This term has to be computed for every interface of a CV!

!Volume of main CV.

!Pressure has no inherent source term, here setting this
!operation just for general purpose .Usually CON(I,J)=0

# (12) Explained in detail

DO 402 I=2,L2

ARHO=R(1)*XCV(I)*RHO(I,1)

! For boundary CV, actual velocity is used.

CON(I,2)=CON(I,2)+ARHO*V(I,2) ! Accumulative add

累加 ! $b=[(\rho u)_w-(\rho u)_e]A_e+[(\rho\tilde{v})_s-(\rho\tilde{v})_n]A_n$

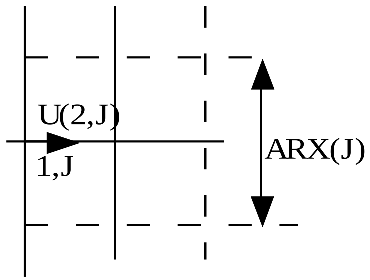
AJM(I,2)=0 !$a_S$ ＝0， Adiabatic boundary

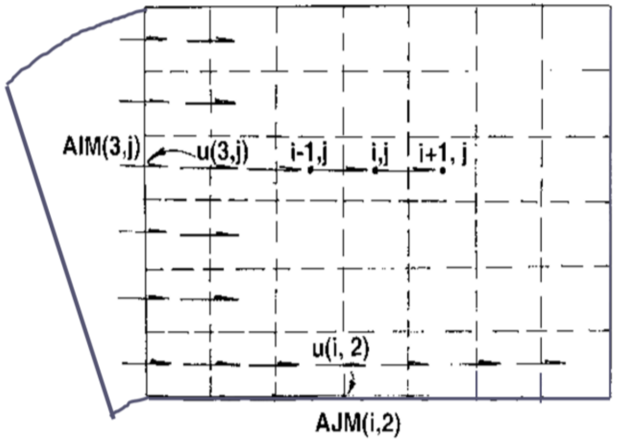
402 ENDDO

DO 403 J=2,M2

ARHO=ARX(J)*RHO(1,J)

CON(2,J)=CON(2,J)+ARHO*U(2,J) ! Accumulative additio




AIM(2,J)=0. ! $a_W$ ＝0， Adiabatic boundary

DO 404 I=2,L2

IF(I= =L2) THEN ! For boundary CV, actual velocity is used.

ARHO=ARX(J)*RHO(L1,J)

CON(I,J)=CON(I,J)-ARHO*U(L1,J) ! Accumulative addition

AIP(I,J)=0. !$a_E$=0

! $b=[(\rho u)_w-(\rho u)_e]A_e+[(\rho\tilde{v})_s-(\rho\tilde{v})_n]A_n$

ELSE

ARHO=ARX(J)*(FX(I+1)*RHO(I+1,J)+FXM(I+1)*RHO(I,J))

```
FLOW=ARHO*UHAT(I+1,J) !    ! For inner CV, UHAT is used.
CON(I,J)=CON(I,J)-FLOW
CON(I+1,J)=CON(I+1,J)+FLOW !
AIP(I,J)=ARHO*DU(I+1,J)    ! a_E
AIM(I+1,J)=AIP(I,J) !Relationship between (a_W) and (a_E)_i+1
ENDIF
IF(J= =M2) THEN
ARHO=RMN(M1)*XCV(I)*RHO(I,M1)
CON(I,J)=CON(I,J)-ARHO*V(I,M1) ! Accumulative addition
AJP(I,J)=0. ! North coefficient of M2
ELSE
ARHO=RMN(J+1)*XCV(I)*(FY(J+1)*RHO(I,J+1)+FYM(J+1)*RHO(I,J))
FLOW=ARHO*VHAT(I,J+1)    ! For inner CV, VHAT is used.
CON(I,J)=CON(I,J)-FLOW
CON(I,J+1)=CON(I,J+1)+FLOW
AJP(I,J)=ARHO*DV(I,J+1)
AJM(I,J+1)=AJP(I,J) !Relationship between coefficients
ENDIF
```

```
      AP(I,J)=AIP(I,J)+AIM(I,J)+AJP(I,J)+AJM(I,J)
404 ENDDO
403 ENDDO
      DO 421 J=2,M2
      DO 422 I=2,L2
      AP(I,J)=AP(I,J)/RELAX(NP)  ! Pressure underrelaxation
      CON(I,J)=CON(I,J)+(1.0-RELAX(NP))*AP(I,J)*P(I,J)
422 ENDDO
421 ENDDO
      COFP(IST:L2,JST:M2,2:5)=COF(IST:L2,JST:M2,2:5)
```

**(13) Explained in detail**

!Store $a_E$, $a_W$, $a_N$, $a_S$ for *p*-correction equation

! while CON（b） and AP ($a_P$) are not stored； Because AP has been

! underrelaxed, and the velocity in b term of *p*-correction eq. is different.

```
      NF=NP !NFMAX+1; P(I,J) is one member of F(I,J,NF)
      CALL SOLVE  !  Solving P-equation
```

**COMPUTE U AND V！ Pressure has been solved**

$$a_e u_e = \sum a_{nb} u_{nb} + b + (p_P - p_E) A_e$$

```
      NF=1
      IST=3
      JST=2
      COF(IST:L2,JST:M2,1:6)=COFU(IST:L2,JST:M2,1:6)！ Coefficients of U
      DO 551 J=JST,M2
      DO 552 I=IST,L2
      CON(I,J)=CON(I,J)+DU(I,J)*AP(I,J)*(P(I-1,J)-P(I,J))
522 ENDDO
521 ENDDO
      CALL SOLVE  !Solving U equation
C------------------------------------------------------------------------------------------
      NF=2
      IST=2
      JST=3
      COF(IST:L2,JST:M2,1:6)=COFV(IST:L2,JST:M2,1:6) !Coefficients of V
      DO 553 J=JST,M2
      DO 554 I=IST,L2
      CON(I,J)=CON(I,J)+DV(I,J)*AP(I,J)*(P(I,J-1)-P(I,J))
```

```
      CON(I,J)=CON(I,J)+DV(I,J)*AP(I,J)*(P(I,J-1)-P(I,J))
554 ENDDO
553 ENDDO
      CALL SOLVE ! Solving V-equation. Such U V are temporary, need to be
   ! improved
COEFFICIENTS FOR THE PRESSURE CORRECTION EQUATION
      NF=3      ! P-correction equation
      CALL RESET     ! Zero of CON(I,j)  and AP(i,j)
      IST=2
      JST=2
      COF(IST:L2,JST:M2,2:5)=COFP(IST:L2,JST:M2,2:5)
   ! Transfer coefficients of P-eq. to P-correction equation.
      CALL GAMSOR
      SMAX=0.
      SSUM=0.
```

! $b = [(\rho u^*)_w - (\rho u^*)_e]A_e + [(\rho v^*)_s - (\rho v^*)_n]A_n$

! The velocities just solved are u* and v*

```
        DO 510 J=2,M2
        DO 511 I=2,L2
        VOL=YCVR(J)*XCV(I)        ! Volume of PCV
        CON(I,J)=CON(I,J)*VOL
511 ENDDO
510 ENDDO
        DO 502 I=2,L2
        ARHO=R(1)*XCV(I)*RHO(I,1)
        CON(I,2)=CON(I,2)+ARHO*V(I,2)  ! Source term b
502 ENDDO
        DO 503 J=2,M2
        ARHO=ARX(J)*RHO(1,J)
        CON(2,J)=CON(2,J)+ARHO*U(2,J)
        DO 504 I=2,L2
        IF(I= =L2) THEN
        ARHO=ARX(J)*RHO(L1,J)
        CON(I,J)=CON(I,J)-ARHO*U(L1,J)  ! Calculate b-term
        ELSE
        ARHO=ARX(J)*(FX(I+1)*RHO(I+1,J)+FXM(I+1)*RHO(I,J))
        FLOW=ARHO*U(I+1,J)  ! Adopt U*,V* to solve P'
        CON(I,J)=CON(I,J)-FLOW
        CON(I+1,J)=CON(I+1,J)+FLOW
```

! $$b=[(\rho u^*)_w-(\rho u^*)_e]A_e+[(\rho v^*)_s-(\rho v^*)_n]A_n$$

Do loop 502—504 for mass source of each CV

```
 ENDIF
    IF(J= =M2) THEN
    ARHO=RMN(M1)*XCV(I)*RHO(I,M1)
    CON(I,J)=CON(I,J)-ARHO*V(I,M1
    ELSE
    ARHO=RMN(J+1)*XCV(I)*(FY(J+1)*RHO(I,J+1)+FYM(J+1)*RHO(I,J))
    FLOW=ARHO*V(I,J+1)
    CON(I,J)=CON(I,J)-FLOW
    CON(I,J+1)=CON(I,J+1)+FLOW
    ENDIF
    AP(I,J)=AIP(I,J)+AIM(I,J)+AJP(I,J)+AJM(I,J)      <- For AP
    PC(I,J)=0.  ! Initial field
    SMAX=AMAX1(SMAX,ABS(CON(I,J)))  ! Take the maximum
    SSUM=SSUM+CON(I,J)  ! Summation of b
504 ENDDO
503 ENDDO
    CALL SOLVE ! Solving p-correction equation
```

COME HERE TO CORRECT THE VELOCITIES

```
      DO 521 J=2,M2
      DO 522 I=2,L2
      IF(I/=2) U(I,J)=U(I,J)+DU(I,J)*(PC(I-1,J)-PC(I,J)) !  Correcting velocity u
      IF(J/=2) V(I,J)=V(I,J)+DV(I,J)*(PC(I,J-1)-PC(I,J)) !  Correcting velocity v
522 ENDDO
521 ENDDO
500 ENDIF
COEFFICIENTS FOR OTHER EQUATIONS----------------------------------------
      IST=2
      JST=2
      DO 600 N=4,NFMAX  !NF>=4
      NF=N
      IF(LSOLVE(NF)) THEN
      CALL GAMSOR
      IF(LSOLE(4)) THEN
      DO I=1,L1
      DO J=1,M1
      RHO(I,J)=RHO(I,J)*CP(I,J)    !  Nominal density for temperature
      ENDDO
      ENDDO
      REL=1.-RELAX(NF)
```

**(14) Explain in detail**

```
    DO 602 I=2,L2
    AREA=R(1)*XCV(I)
    FLOW=AREA*V(I,2)*RHO(I,1)
    DIFF=AREA*GAM(I,1)/YDIF(2)
    CALL DIFLOW
    AJM(I,2)=ACOF+AMAX1(0.,FLOW)
602 ENDDO
    DO 603 J=2,M2
    FLOW=ARX(J)*U(2,J)*RHO(1,J)
    DIFF=ARX(J)*GAM(1,J)/(XDIF(2)*SX(J))
    CALL DIFLOW
    AIM(2,J)=ACOF+AMAX1(0.,FLOW)
    DO 604 I=2,L2
    IF(I==L2) THEN
    FLOW=ARX(J)*U(L1,J)*RHO(L1,J)
    DIFF=ARX(J)*GAM(L1,J)/(XDIF(L1)*SX(J))
    ELSE
    FLOW=ARX(J)*U(I+1,J)*(FX(I+1)*RHO(I+1,J)+FXM(I+1)*RHO(I,J))
    DIFF=ARX(J)*2.*GAM(I,J)*GAM(I+1,J)/((XCV(I)*GAM(I+1,J)+
```

```
& XCV(I+1)*GAM(I,J)+1.0E-30)*SX(J))
  ENDIF
  CALL DIFLOW
  AIM(I+1,J)=ACOF+AMAX1(0.,FLOW)
  AIP(I,J)=AIM(I+1,J)-FLOW
  AREA=RMN(J+1)*XCV(I)
   IF(J= = M2) THEN
   FLOW=AREA*V(I,M1)*RHO(I,M1)
   DIFF=AREA*GAM(I,M1)/YDIF(M1)
   ELSE
   FLOW=AREA*V(I,J+1)*(FY(J+1)*RHO(I,J+1)+FYM(J+1)*RHO(I,J))
   DIFF=AREA*2.*GAM(I,J)*GAM(I,J+1)/(YCV(J)*GAM(I,J+1)+
& YCV(J+1)*GAM(I,J)+1.0E-30)
   ENDIF
   CALL DIFLOW
```

```
      AJM(I,J+1)=ACOF+AMAX1(0.,FLOW)
      AJP(I,J)=AJM(I,J+1)-FLOW
      VOL=YCVR(J)*XCV(I)
      APT=RHO(I,J)/DT  ! Transient term ρ/△t without volume
      AP(I,J)=AP(I,J)-APT
      CON(I,J)=CON(I,J)+APT*F(I,J,NF)
      AP(I,J)=(-AP(I,J)*VOL+AIP(I,J)+AIM(I,J)+AJP(I,J)+AJM(I,J))
     &/RELAX(NF)
      CON(I,J)=CON(I,J)*VOL+REL*AP(I,J)*F(I,J,NF)
604 ENDO
603 ENDO
      CALL SOLVE !
      IF（LSLVE(4)) THEN ! If the temp. eq. is solved ,then Reset density back to rho
      DO I=I,L1
      DO J=1,M1
      RHO(I,J)=RHO(I,J)/CP(I,J)  ! Reset density back to rho
      ENDDO
      ENDDO
      ENDIF
      ENDIF
600 ENDDO（End of the solving process）
      TIME=TIME+DT  !  Forward time
      ITER=ITER+1   !Increase the indicator
      IF(ITER>= LAST) LSTOP=.TRUE. RETURN
      END
```

$$a_p = (\sum a_{nb} + \rho_P \Delta V/\Delta t - S_p \Delta V)/\alpha$$

$$b = S_c \Delta V + a_p^0 \phi_p^0 + (1-\alpha)\frac{a_P}{\alpha}\phi_P^0$$

Transient, Linear---- Steady, nonlinear

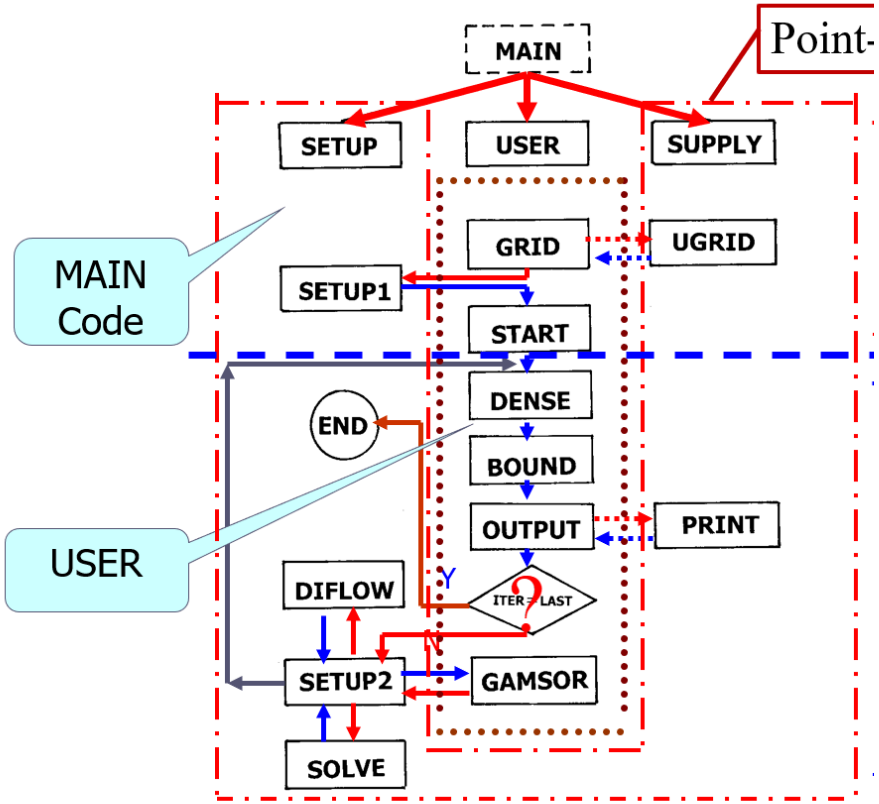
**(15)---Explained in detail**



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

# 10.6.2.6 SUBROUTINE SUPPLY

```
      SUBROUTINE SUPPLY
C*********************************************************************
      USE START_L
      IMPLICIT NONE
      REAL*8 DX,DY,RHOM,PREF
      INTEGER*4 I,J,N,JJ,IEND,JEND,IBEG,JBEG,IFST,JFST,JFL
C*********************************************************************
```



```
C******************************************************************
 10 FORMAT(1X,26(1H*),3X,A10,3X,26(1H*))
 20 FORMAT(1X,4H I =,I6,6I9)
 30 FORMAT(1X,' J')
 40 FORMAT(1X,I3,2X,1P7E9.2)
 50 FORMAT(1X,1H )
 51 FORMAT(2X,'I =',2X,7(I4,5X))
 52 FORMAT(2X,'X =',1P7E9.2)
 53 FORMAT(1X,' TH =',1P7E9.2)
 54 FORMAT(2X,'J =',2X,7(I4,5X))
 55 FORMAT(2X,'Y =',1P7E9.2)
C******************************************************************
   ENTRY UGRID
   XU(2)=0.
   DX=XL/FLOAT(L1-2)
   DO 1 I=3,L1
   XU(I)=XU(I-1)+DX
 1 ENDDO
   YV(2)=0.
   DY=YL/FLOAT(M1-2)

   DO 2 J=3,M1
   YV(J)=YV(J-1)+DY
 2 ENDDO
   RETURN
```

**!1P7E9.2**

**!1P---1 integral digit of each data;**

**!7E---7 data in scientific expression**

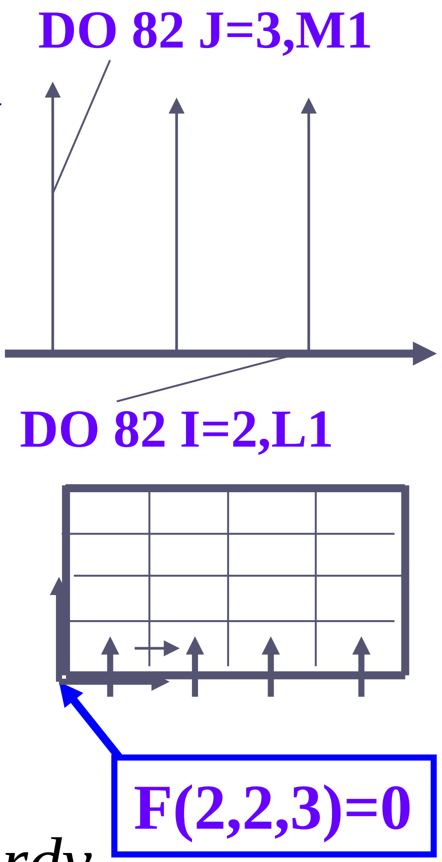
**! 9.2---Each data contains 9 places, and there are two decimal places (小数2位)**

| | | | | | | |
|---|---|---|---|---|---|---|
| 2.00E+00 | 2.30E+00 | 2.90E+00 | 3.50E+00 | 4.10E+00 | 4.70E+00 | 5.00E+00 |
| 1.80E+00 | 2.08E+00 | 2.64E+00 | 3.20E+00 | 3.76E+00 | 4.32E+00 | 4.60E+00 |
| 1.40E+00 | 1.64E+00 | 2.12E+00 | 2.60E+00 | 3.08E+00 | 3.56E+00 | 3.80E+00 |
| 1.00E+00 | 1.20E+00 | 1.60E+00 | 2.00E+00 | 2.40E+00 | 2.80E+00 | 3.00E+00 |
| 6.00E-01 | 7.60E-01 | 1.08E+00 | 1.40E+00 | 1.72E+00 | 2.04E+00 | 2.20E+00 |
| 2.00E-01 | 3.20E-01 | 5.60E-01 | 8.00E-01 | 1.04E+00 | 1.28E+00 | 1.40E+00 |
| 0.00E+00 | 1.00E-01 | 3.00E-01 | 5.00E-01 | 7.00E-01 | 9.00E-01 | 1.00E+00 |

**(16)---Explained in detail**

```
C*****************************************************
      ENTRY PRINT
      IF(LPRINT(3)) THEN
CALCULATE THE STREAM FUNCTION
      F(2,2,3)=0.
      DO 81 I=2,L1
      IF(I.NE.2) F(I,2,3)=F(I-1,2,3)-RHO(I-1,1)*V(I-1,2)
     &*R(1)*XCV(I-1)  ! I=2， F(2,2,3)=0;
      DO 82 J=3,M1
      RHOM=FX(I)*RHO(I,J-1)+FXM(I)*RHO(I-1,J-1)
      F(I,J,3)=F(I,J-1,3)+RHOM*U(I,J-1)*ARX(J-1) !
   82 ENDDO
   81 ENDDO
```

! For print out， NF=3 represents stream function




$$\rho u r = \frac{\partial \psi}{\partial y}; \rho v r = -\frac{\partial \psi}{\partial x} \quad \psi = -\int \rho v r dx \quad \psi = \int \rho u r dy$$

**(17)---Explained in detail**

**For bottom, from left to right** $\psi_{i,2} = \psi_{i-1,2} - \sum_{i=3} \rho_{i-1,1} v_{i-1,2} r(1) \Delta x_i$

**For vertical, from bottom to top** $\psi_{i,j} = \psi_{i,j-1} + \rho_m u_{i,j-1} r(j) \Delta y_j$

```
IF(LPRINT(NP)) THEN
CONSTRUCT BOUNDARY PRESSURES BY EXTRAPOLATION
   DO 91 J=2,M2
   P(1,J)=(P(2,J)*XCVS(3)-P(3,J)*XDIF(2))/XDIF(3)
   P(L1,J)=(P(L2,J)*XCVS(L2)-P(L3,J)*XDIF(L1))/XDIF(L2)
91 ENDDO
   DO 92 I=2,L2
   P(I,1)=(P(I,2)*YCVS(3)-P(I,3)*YDIF(2))/YDIF(3)
   P(I,M1)=(P(I,M2)*YCVS(M2)-P(I,M3)*YDIF(M1))/YDIF(M2)
92 ENDDO
   P(1,1)=P(2,1)+P(1,2)-P(2,2)
   P(L1,1)=P(L2,1)+P(L1,2)-P(L2,2)
   P(1,M1)=P(2,M1)+P(1,M2)-P(2,M2)
   P(L1,M1)=P(L2,M1)+P(L1,M2)-P(L2,M2)
   PREF=P(IPREF,JPREF) ! Reference point of pressure
   DO 93 J=1,M1
   DO 93 I=1,L1
   P(I,J)=P(I,J)-PREF  ! Relative pressure
 94ENDDO
 93ENDDO
   ENDIF
```

(17a)---Explained in detail

```
PRINT 50 ! Print out to screen
WRITE(8,50) ! Output into file
IEND=0
DO WHILE (IEND/=L1)
IBEG=IEND+1
IEND=IEND+7  ! !7 data in each line
IEND=MIN0(IEND,L1) ! Take minimum
PRINT 50
WRITE(8,50)
PRINT 51,(I,I=IBEG,IEND) !From IBEG to IEND for printing
WRITE(8,51) (I,I=IBEG,IEND)
IF(MODE/=3) THEN
PRINT 52,(X(I),I=IBEG,IEND)
WRITE(8,52) (X(I),I=IBEG,IEND)
ELSE                          ! Print out x-coordinates
PRINT 53,(X(I),I=IBEG,IEND)
WRITE(8,53) (X(I),I=IBEG,IEND)
ENDIF
ENDDO
IF(IEND= =L1) THEN
```

```
JEND=0
PRINT 50
WRITE(8,50)
DO WHILE(JEND/=M1) THEN
JBEG=JEND+1
JEND=JEND+7
JEND=MIN0(JEND,M1)
PRINT 50
WRITE(8,50)
PRINT 54,(J,J=JBEG,JEND)
WRITE(8,54) (J,J=JBEG,JEND)
PRINT 55,(Y(J),J=JBEG,JEND)  ! Print out y-coordinates
WRITE(8,55) (Y(J),J=JBEG,JEND) GO TO 311
ENDDO
ENDIF
```

```
DO 999 N=1,NCP            ! NCP has been defined as 14 in SETUP1
NF=N
IF(LPRINT(NF)) THEN       ! Print out F(I,J,NF) field
PRINT 50
WRITE(8,50)
PRINT 10,TITLE(NF)
WRITE(8,10) TITLE(NF)     ! Print out title of variable F(I,J,NF)
IFST=1
JFST=1
IF(NF==1.OR.NF==3) IFST=2
IF(NF==2.OR.NF==3) JFST=2
IBEG=IFST-7
DO WHILE（IEND<L1.OR.IBEG== -5.OR.IBRG== -6)
IBEG=IBEG+7  ! Starting point for each line (7data)
IEND=IBEG+6  ! Ending point of the line
IEND=MIN0(IEND,L1)
PRINT 50 WRITE(8,50)
```

************************** .TEMP. **************************

| I = | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| J | | | | | | | |
| 7 | 2.00E+00 | 2.30E+00 | 2.90E+00 | 3.50E+00 | 4.10E+00 | 4.70E+00 | 5.00E+00 |
| 6 | 1.80E+00 | 2.08E+00 | 2.64E+00 | 3.20E+00 | 3.76E+00 | 4.32E+00 | 4.60E+00 |
| 5 | 1.40E+00 | 1.64E+00 | 2.12E+00 | 2.60E+00 | 3.08E+00 | 3.56E+00 | 3.80E+00 |
| 4 | 1.00E+00 | 1.20E+00 | 1.60E+00 | 2.00E+00 | 2.40E+00 | 2.80E+00 | 3.00E+00 |
| 3 | 6.00E-01 | 7.60E-01 | 1.08E+00 | 1.40E+00 | 1.72E+00 | 2.04E+00 | 2.20E+00 |
| 2 | 2.00E-01 | 3.20E-01 | 5.60E-01 | 8.00E-01 | 1.04E+00 | 1.28E+00 | 1.40E+00 |
| 1 | 0.00E+00 | 1.00E-01 | 3.00E-01 | 5.00E-01 | 7.00E-01 | 9.00E-01 | 1.00E+00 |

**(17b)—Explained in detail**

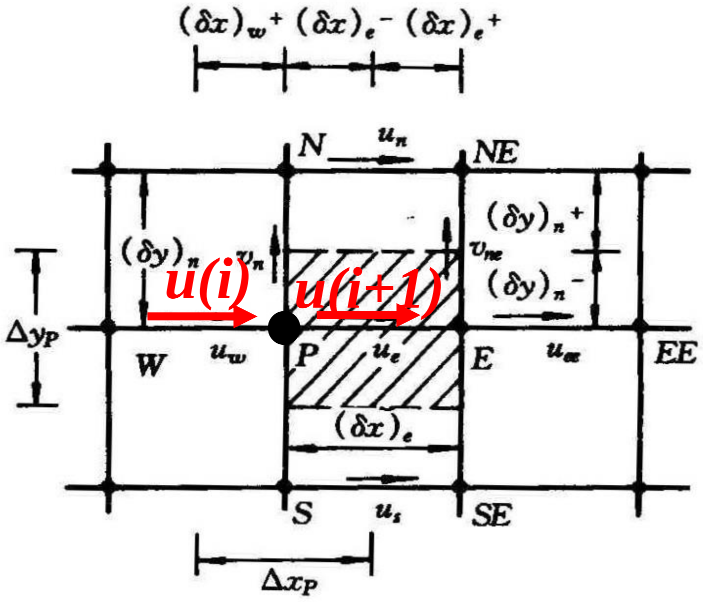
```
      PRINT 20,(I,I=IBEG,IEND)
      WRITE(8,20) (I,I=IBEG,IEND)
      PRINT 30
      WRITE(8,30)
      JFL=JFST+M1  .
      DO 115 JJ=JFST,M1
      J=JFL-JJ
      PRINT 40, J, (F(I,J,NF),I=IBEG,IEND)
      WRITE(8,40) J,(F(I,J,NF),I=IBEG,IEND)
  115 ENDDO
      ENDDO
      ENDIF
  999 END  （End of print do-loop）
```

# Transformation of data format for Tecplot

```
OPEN(9,FILE="RESULT.DAT")
    WRITE(9,'("VARIABLES=X,Y",$)')
          DO NF=1,NCP
          IF(LPRINT(NF)) WRITE(9,'(",",A7,$)') TITLE(NF)
          ENDDO
    WRITE(9,'(/,"ZONE I=",I4,",J=",I4,",T=T",$)') L1,M1
    DO J=1,M1
    DO I=1,L1
          WRITE(9,'(/,E11.3,E11.3,$)') X(I),Y(J)
    DO NF=1,NCP
          IF(LPRINT(NF)) THEN
    FSHOW=F(I,J,NF)
    IF(NF==1) THEN
      IF(I==1) FSHOW=U(2,J)
      IF(I>=2.AND.I<=L2)  FSHOW=(U(I,J)+U(I+1,J))/2
      IF(I==L1) FSHOW=U(L1,J)
    ENDIF
```

Data format of TECPLOT

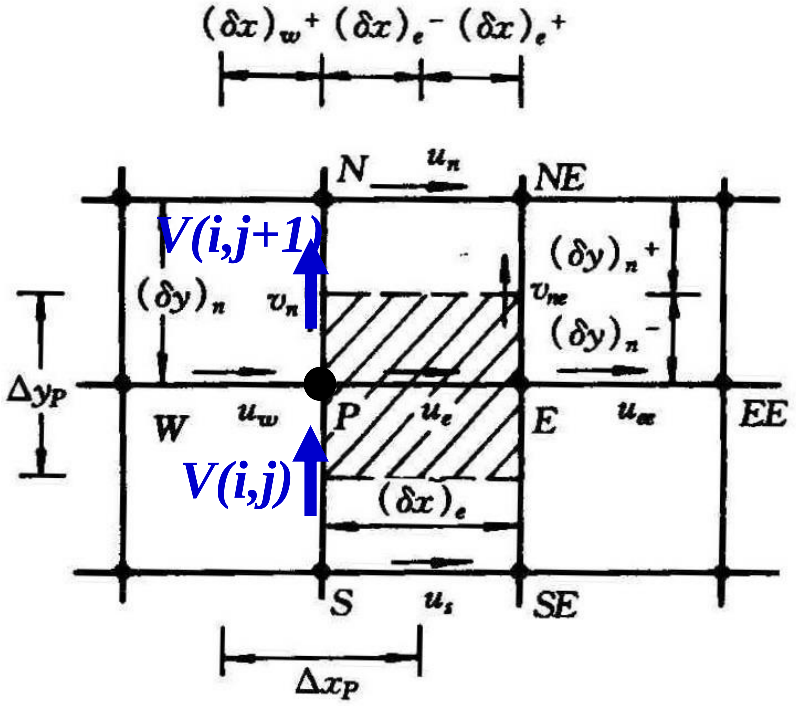
Data format of TECPLOT




For u

```
IF(NF==2) THEN
    IF(J==1) FSHOW=V(I,2)
    IF(J>=2.AND.J<=M2) FSHOW=(V(I,J)+V(I,J+1))/2
    IF(J==M1) FSHOW=V(I,M1)
  ENDIF
  WRITE(9,'(E11.3,$)') FSHOW
  ENDIF
  ENDDO
      ENDDO
  ENDDO
  CLOSE(9)
  RETURN
  END
```

For v



# Comments and Recommendations for Teaching Code Study

1.It is the students' responsibility to study the code line by line to completely understanding the function of each line and the numerical techniques included.

You should understand every detail included in each line, for example:

```
IF(MODE.== 3) THEN
SX(J)=R(J)
IF(J /= 1) SXMN(J)=RMN(J)
ENDIF
```

why here J=1 should not be included?

2.You can understand a numerical algorithm, say SIMPLER, completely only when you know how to implementing the algorithm by code.

3.If you meet some difficulty in understanding the teaching code you may contact me by email (fangwenzhen@mail.xjtu.edu.cn) at any time. I will be happy to communicate with you.

4. Our teaching assistants will give you instruction on how to run the code.

本组网页地址：http://nht.xjtu.edu.cn 欢迎访问！

*Teaching PPT will be loaded on our website*


同舟共济
渡彼岸！

**People in the same boat help each other to cross to the other bank, where....**