

Numerical Heat Transfer (数值传热学)

Chapter 8 General Code for 2D Elliptical Fluid Flow and Heat Transfer (2)



Instructor Tao, Wen-Quan

Key Laboratory of Thermo-Fluid Science & Engineering
Int. Joint Research Laboratory of Thermal Science & Engineering
Xi'an Jiaotong University
Innovative Harbor of West China, Xian

2020-Nov.-03

Chapter 8 General Code for 2D Elliptical Fluid Flow and Heat Transfer Problems (2)

8.6 Methods of application and explanation of MAIN Program

8.6.1 Methods of Code application

8.6.2 Explanation of MAIN Program

8.6.2.1 MODULE START_L

8.6.2.2 PROGRAM MAIN

8.6.2.3 SUBROUTINE DIFLOW

8.6.2.4 SUBROUTINE SOLVE

8.6.2.5 SUBROUTINE SETUP

8.6.2.6 SUBROUTINE SUPPLY

8. 6 Methods of Application and Explanation of MAIN

8.6.1 Methods of Code application

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

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

Determine S_{ϕ}^* , Γ_{ϕ} , and ρ_{ϕ}^*

2. Calling (调用) a USER similar to the problem studied, retaining MODULE part, modifying other part and saving with a new name;

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:

.TRUE.

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

.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: **.TRUE.**, otherwise **.FALSE.**, Its default value is **.T. .**
- (5) **LAST**—Given iteration times, default values is 5.
- (6) **NTIMES(NF)**—Default values equals 1; for steady nonlinear : 1 to 2; unsteady linear: 5 to 6
- (7) **DT**—Time step, default value is 10^{30}

For fully implicit scheme, in the b-term there is a term of $a_p^0 = \rho\Delta V / \Delta t$, if $\Delta t \rightarrow \infty$, $a_p^0 \rightarrow 0$, 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)).

8-6-2 Explanation of MAIN programs

CC

**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
C as a frame to re-develop your own code for research purpose.**

C -----Instructor of Numerical Heat Transfer, XJTU,2013.-----

CC

**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 Lin of NHT
C group of XJTU during 2013.01-04**

CC

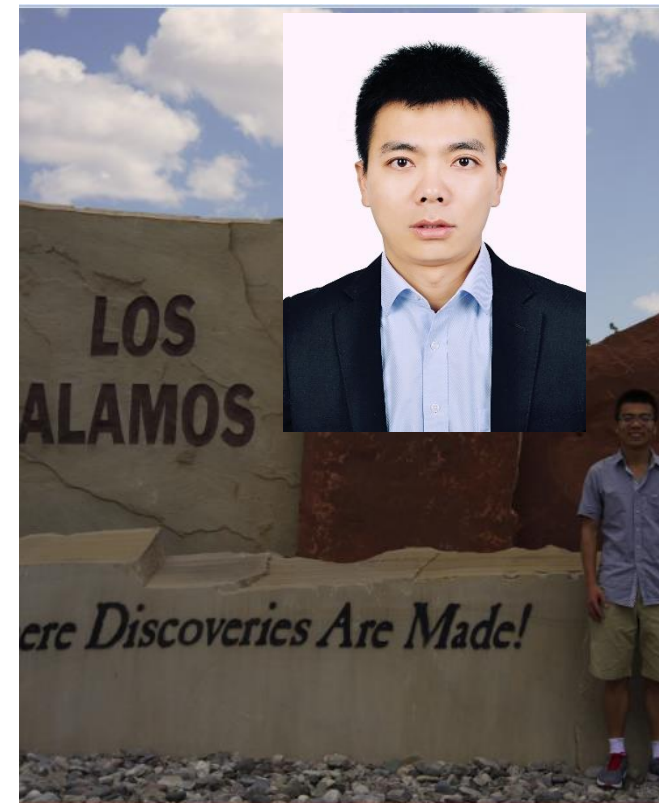
C*****



毋玉同



凌空



陈黎

8.6.2.1 MODULE START_L

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,

1 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,

1 IST,JST,ITER,LAST,MODE,NTIMES(NFX4),IPREF,JPREF

REAL*8 SMAX,SSUM

REAL*8 FLOW,DIFF,ACOF

- (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).

Sc or b Ae,Aw,An,As, 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/ Default value!!  
DATA RELAX,NTIMES/NFX4*1.,NFX4*1/  
DATA DT,IPREF,JPREF,RHOCON,CPCON/1.E+30, 1,1,1.,1./  
END MODULE
```

- (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;

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 digit;

INTEGER*4 NF,NP,NRHO

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

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))

8.6.2.2 PROGRAM MAIN

C*****

C-----**MAIN**-----

C*****

Calling
Module

PROGRAM MAIN !Name of subroutine
USE START_L **Share the variables defined in the MODULE**
IMPLICIT NONE

C*****

Only
Executed
once

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

Executed
Many times

CALL DENSE !Set up fluid density
CALL BOUND !Set up boundary condition
CALL OUTPUT !Print out
CALL SETUP2 !Key module: set coefficients and solve ABEqs.
ENDDO
CALL OUTPUT !Print out some results
CLOSE(8) !Simulation completed close file RESULT.TXT
STOP !Terminate computation
END !End of subroutine

CC

8.6.2.3 SUBROUTINE DIFLOW

CC

Calling
Module

SUBROUTINE DIFLOW ! 确定 $D \cdot A(|P_{\Delta}|)$ of power law scheme
 { **USE START_L** Share the variables defined in the
IMPLICIT NONE **MODULE START_L**
REAL*8 TEMP Declaration of a temporal real variable **TEMP**

C*****

! 1st return for
diffusion case

ACOF=DIFF ! $D \cdot A(|P_{\Delta}|) = D$
IF(FLOW== 0.) RETURN ! No flow, only diffusion
TEMP=DIFF-ABS(FLOW)*0.1 ! $D - 0.1|F| = D(1 - 0.1|P_{\Delta}|)$

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

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

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

IF(TEMP.<= 0.) RETURN ! $|P_{\Delta e}| > 10$
TEMP=TEMP/DIFF ! $1 - 0.1|P_{\Delta e}|$
ACOF=DIFF*TEMP5** ! $D \cdot (1 - 0.1|P_{\Delta e}|)^5 = D \cdot A(|P_{\Delta e}|)$
RETURN

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

CC

8.6.2.4 SUBROUTINE SOLVE

Calling
Module

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC  
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*****
```

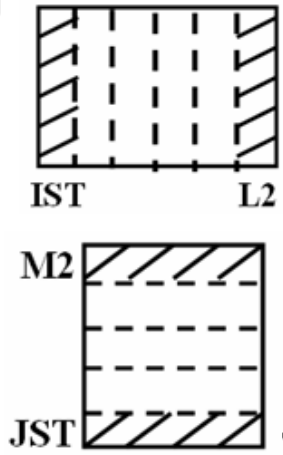

Structure of SOLVE

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
    
```

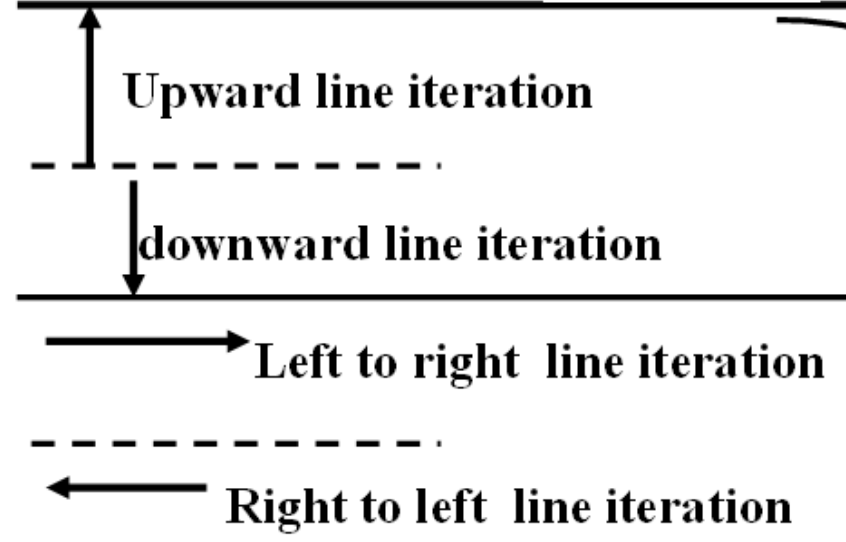


2 times of block corrections

AD Block Correction

4 times of line iterations

AD line Iteration



```
999 ENDDO
```

Review on block correction

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

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

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

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

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

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

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

$$\text{DENOM} = BL - PT(i-1) * BLM$$

DENOM

C*****

ISTF=IST-1
 JSTF=JST-1
 IT1=L2+IST
IT2=L3+IST
 JT1=M2+JST
JT2=M3+JST

!Temporal integral variables for 1st variable or starting point of DO-loop

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 P_{IST-1}

QT(ISTF)=0. ! Constant in TDMA Q_{IST-1}

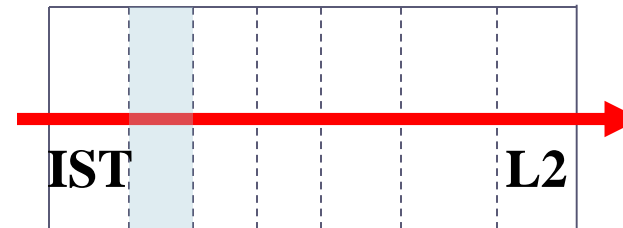
DO 11 I=IST,L2

BL=0. !Initial value in B-correction

BLP=0. !Initial value in B-correction

BLM=0. ! Initial value in B-correction

I -direction B.Correction.



In discussion of TDMA: A B C D

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

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)

1 +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

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

IF(ABS(DENOM/BL) < 1.E-10) DENOM=1.E25

PT(I)=BLP/DENOM $P_i = B_i / (A_i - C_i P_{i-1});$

QT(I)=(BLC+BLM*QT(I-1))/DENOM $Q_i = (D_i + C_i Q_{i-1}) / (A_i - C_i P_{i-1});$

11 ENDDO

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

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

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

!Ensure a meaningful correction

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

DENOM

Coefficients calculation

Elimination (消元)

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)

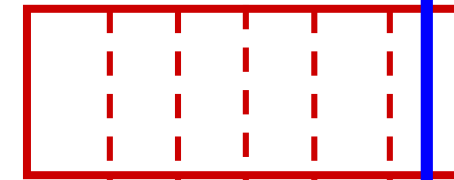
BLP=BLP+AJP(I,J)

IT1=L2+IST

I=IT1-II=L2+IST-IST=L2-Begin

I=IT1-II=L2+IST-L2=IST-End

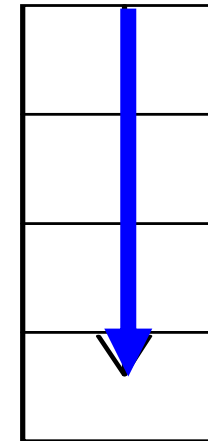
Correcting by BL for the same column



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

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

TDMA: from L2 to IST



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

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)
1 +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
```

! Above is block correction, following is ADI line iteration

Solving in I-direction, scanning in J direction, SLUR

C

DO 90 J=JST,M2 $AP\phi_{i,j}^n = AIP\phi_{i+1,j}^n + AIM\phi_{i-1,j}^n + b + AJP\phi_{i,j+1}^{n-1} + AJM\phi_{i,j-1}^{n-1}; i = IST...L2$

PT(ISTF)=0. ! PT=0, QT=given boundary value, 1st kind boundary condition

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

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

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

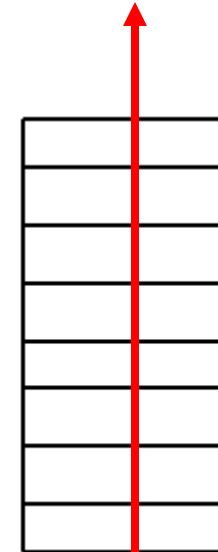
F(I,J,N)=F(I+1,J,N)*PT(I)+QT(I)

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

80 ENDDO

90 ENDDO

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



Elimination
(消元)

Back substitution
(回代)

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)

} For executing 1st kind B.C.

DO 170 I=IST,L2

DENOM=AP(I,J)-PT(I-1)*AIM(I,J)

PT(I)=AIP(I,J)/DENOM

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

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

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

170 ENDDO

DO 180 II=IST,L2

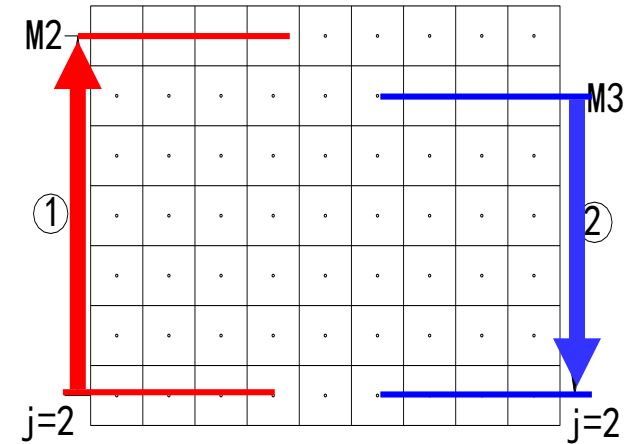
I=IT1-II !Recursive solution

F(I,J,N)=F(I+1,J,N)*PT(I)+QT(I)

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

180 ENDDO

190 ENDDO



Elimination
(消元)

Back
substitution
(回代)

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

CC

8.6.2.5 SUBROUTINE SETUP

CC

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

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
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

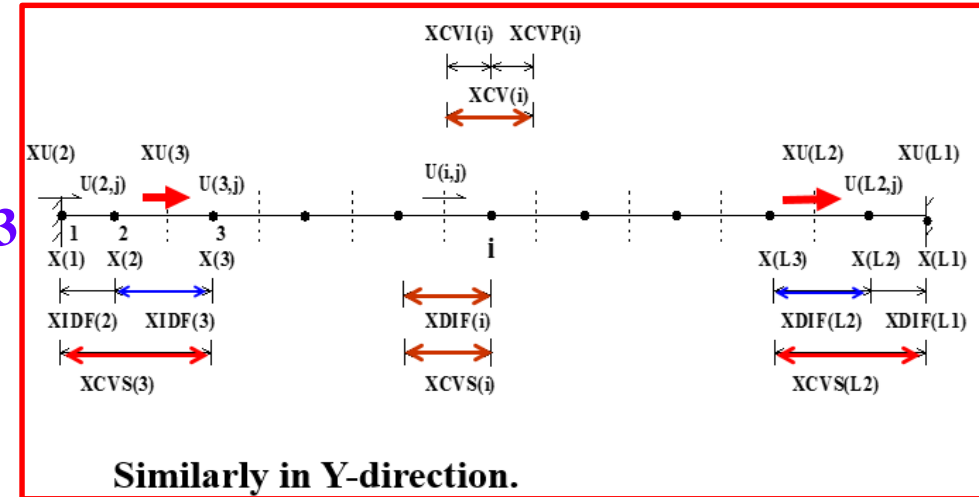
X(L1)=XU(L1)

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

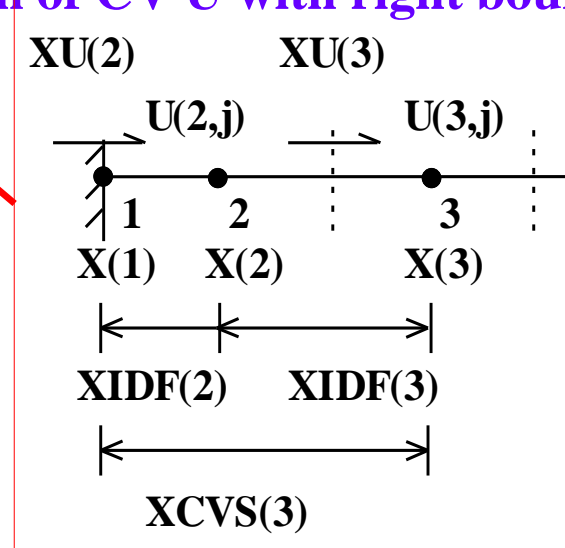


**! Practice B:
XU(I) has been
set in GRID**

```

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

```



```
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 Cartesian
coordinate**

```
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
```

Set up scaling Factor for
polar coordinate

```
57 ENDDO
```

Interface starts from J=2

```
DO 62 J=2,M2  
YCVR(J)=R(J)*YCV(J)  
ARX(J)=YCVR(J)  
IF(MODE = = 3) THEN  
ARX(J)=YCV(J)
```

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

```
62 ENDDO
```

```

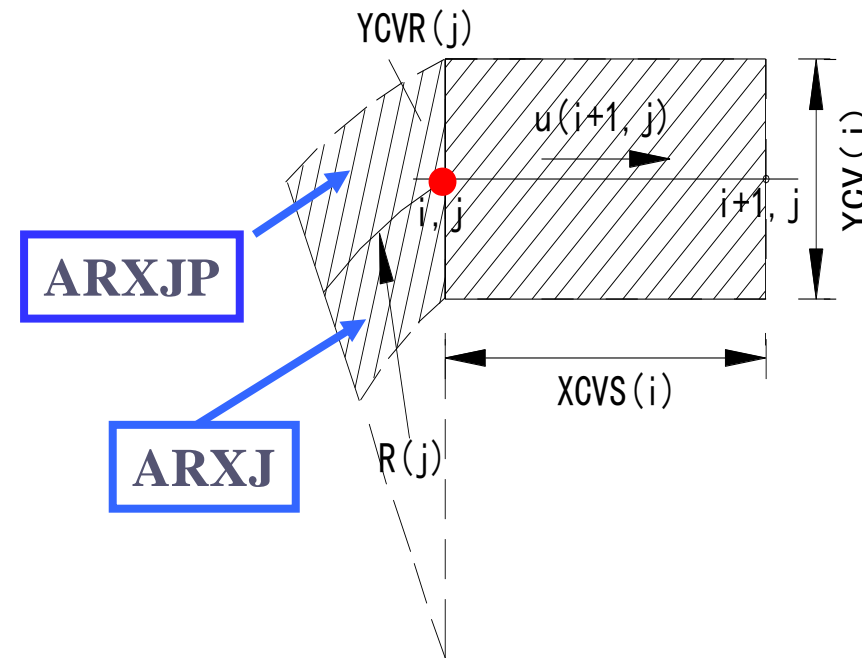
DO 64 J=4,M3
YCVRS(J)=0.5*(R(J)+R(J-1))*YDIF(J)
64 ENDDO
YCVRS(3)=0.5*(R(3)+R(1))*YCVS(3)
YCVRS(M2)=0.5*(R(M1)+R(M3))*YCVS(M2)
IF(MODE == 2) THEN
DO 65 J=3,M3
ARXJ(J)=0.25*(1.+RMN(J)/R(J))*ARX(J)
ARXJP(J)=ARX(J)-ARXJ(J)
65 ENDDO
ELSE
DO 66 J=3,M3
ARXJ(J)=0.5*ARX(J)
ARXJP(J)=ARXJ(J)
66 ENDDO
ENDIF
ARXJP(2)=ARX(2)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

```

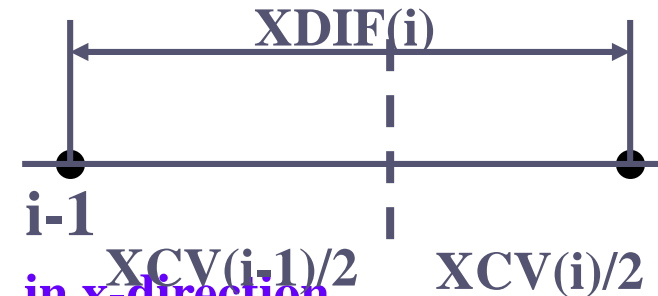
$$ARXJ(J) = \frac{1}{2} (R(j) + RMN(j)) \cdot \frac{YCV(j)}{2} =$$

$$0.25 \left[1 + \frac{RMN(j)}{R(j)} \right] \cdot \underline{R(j)} \cdot \underline{YCV(j)} =$$

$$0.25 \left[1 + \frac{RMN(j)}{R(j)} \right] \cdot \underline{ARX(j)}$$



```
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
```

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

$$= \phi_{i-1} FXM(i) + \phi_i FX(i)$$

```
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.
```

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

CGN,AP,U,V,RHO,PC AND P ARRAYS ARE INITIALIZED HERE


```
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)
```

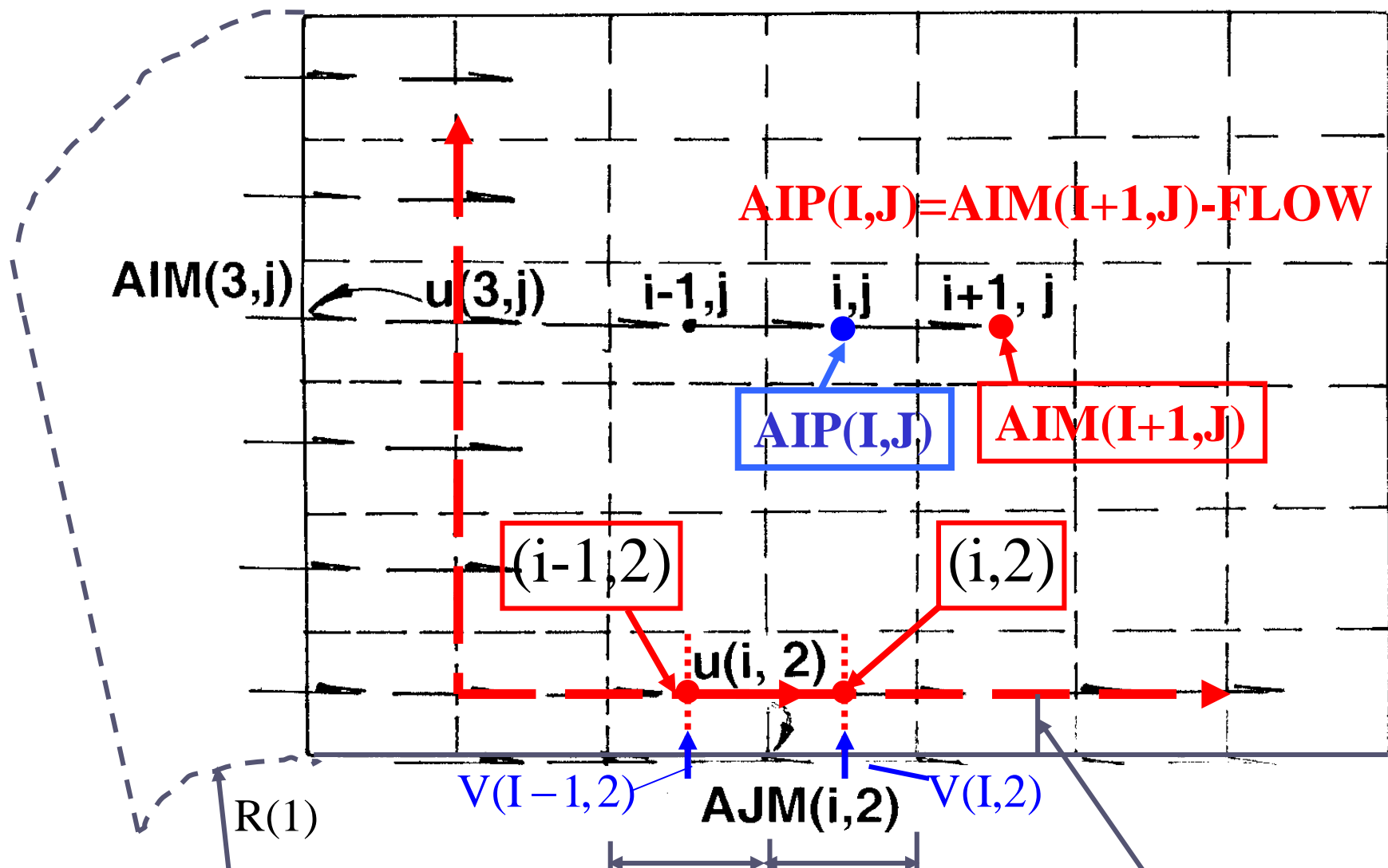
```
PRINT 4
```

```
WRITE(8,4) (20151216)
```

```
RETURN
```

Set up initial fields for iteration

Print out coordinate title of out put data



$$FL = XCVI(I) * V(I,2) * RHO(I,1) \leftarrow$$

$$FLM = XCVIP(I-1) * V(I-1,2) * RHO(I-1,1) \leftarrow$$

$$FLOW = R(1) * (FL + FLM) \leftarrow$$

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

$$xcv(i-1) \quad xcv(i)$$

$$XCVIP(I-1) \quad XCVI(I)$$

C-----

ENTRY SETUP2

CC

COEFFICIENTS FOR THE U EQUATION

NF=1 !NF=1: U; NF=2: V; NF=3: 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)

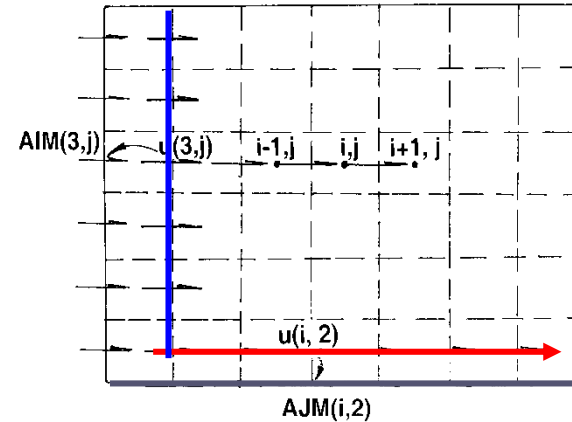
FLOW=R(1)*(FL+FLM) ! Flow rate through south interface

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

CALL DIFLOW !Get D.A(|P|);

AJM(I,2)=ACOF+AMAX1(0.,FLOW) Coefficient a_s

102 ENDDO



Explanation of DIFF(Diffusion conductance)

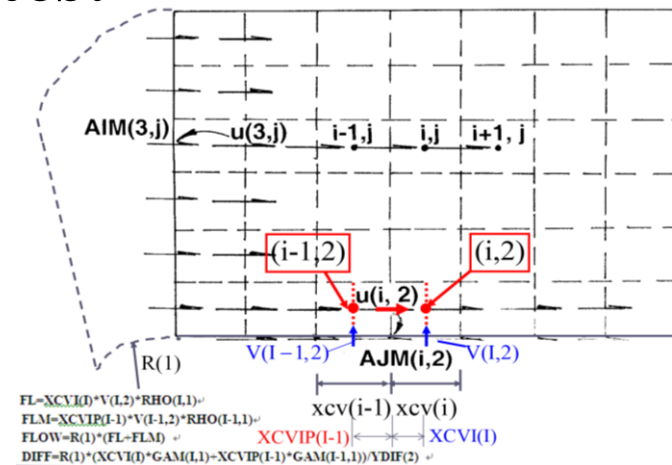
$$DIFF=R(1)*(XCVI(I)*GAM(I,1)+XCVIP(I-1)*GAM(I-1,1))/YDIF(2)$$

For Cartesian coordinates:

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

For Cylindrical coordinates:

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



```

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  $a_w$ 
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)  $D \bullet A(|P_\Delta|) + \|0, F\|$ 
AIP(I,J)=AIM(I+1,J)-FLOW ! Relationship between coefficients
    
```

```
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)*
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 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)*
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)
AJP(I,J)=AJM(I,J+1)-FLOW !Relationship between coefficients
```

VOL=YCVR(J)*XCVS(I) !Volume of velocity CV

APT=(RHO(I,J)*XCVI(I)+RHO(I-1,J)*XCVIP(I-1))

1/(XCVS(I)*DT) ! Unsteady term $\rho/\Delta t$; DT--- Δt ;

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))

1/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

$$a_p^0 = \frac{\rho_p \Delta V}{\Delta t}$$

104 ENDDO

103 ENDDO

$$\frac{A_e}{a_e}$$

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

$$a_p = (\sum a_{nb} + a_p^0 - S_p \Delta V) / \alpha$$

Review of SIMPLER algorithm(1)

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

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

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

$$u_e = \underline{u} + \left(\frac{A_e}{a_e}\right)(p_P - p_E) = u + d_e(p_P - p_E); \quad v_n = \tilde{v}_n + d_n(p_P - p_N)$$

and Solving pressure equation, obtaining p^* ;

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

$$b = \frac{(\rho_P^0 - \rho_P) \Delta x \Delta y}{\Delta t} + [(\rho u)_w - (\rho u)_s] A_e + [(\rho \tilde{v})_s - (\rho \tilde{v})_n] A_n$$

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

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

$$a_E = \rho \left(\frac{A_e}{a_e} \right) \Delta y$$

$$a_N = \rho \left(\frac{A_n}{a_n} \right) \Delta x$$

Review of SIMPLER algorithm (2)

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

Boundary velocities take the specified values.

5. Correcting velocity $u = u^* + u'$; $v = v^* + v'$, where u' and v' are determined based on p'
6. Taking the updated velocity, repeating steps 1-6, until convergence is reached.

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

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,

1 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

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

$COFU(IST:L2, JST:M2, 1:6)=COF(IST:L2,JST:M2,1:6)$

! Store coefficients of U temporary as follows:

| | | | | | |
|-------------------|-------------------|------------------------|-------------------|-------------------|-------------------|
| 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

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)

```
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.E.= 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 1))/(XDIF(L1)*SXMN(J))
```

```
ELSE
FL=ARXJ(J)*U(I+1,J)*(FX(I+1)*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 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)*
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) ! aw
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) ! as
AJP(I,J)=AJM(I,J+1)-FLOW !Relationship
VOL=YCVRS(J)*XCV(I) !Volume of V- CV
SXT=SX(J)
```

$$APT=(ARXJ(J)*RHO(I,J)*0.5*(SXT+SXMN(J))+ARXJP(J-1)*RHO(I,J-1)*10.5*(SXB+SXMN(J)))/(YCVRS(J)*DT)$$

$$AP(I,J)=AP(I,J)-APT$$

$$CON(I,J)=CON(I,J)+APT*V(I,J)$$

$$AP(I,J)=(-AP(I,J)*VOL+AIP(I,J)+AIM(I,J)+AJP(I,J)+AJM(I,J))$$

$$1/RELAX(NF)$$

$$CON(I,J)=CON(I,J)*VOL+REL*AP(I,J)*V(I,J)$$

$$DV(I,J)=VOL/YDIF(J)$$

$$DV(I,J)=DV(I,J)/AP(I,J)$$

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

$$UHAT(I,J)=(COFU(I,J,2)*U(I+1,J)+COFU(I,J,3)*U(I-1,J)+COFU(I,J,4)$$

$$1 *U(I,J+1)+COFU(I,J,5)*U(I,J-1)+COFU(I,J,1))/COFU(I,J,6)$$

! Compute u, \tilde{v}

$$\bar{u}_e = \sum \frac{a_{nb} u_{nb} + b}{a_e}$$

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)
 1 *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** !In pressure equation no source term for the generality

DO 410 J=2,M2 ! source term is still computed.

DO 411 I=2,L2

VOL=YCVR(J)*XCV(I) !Volume of main CV.

CON(I,J)=CON(I,J)*VOL

411 ENDDO

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

! For boundary actual velocity is used.

DO 402 I=2,L2

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

累加

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

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 addition

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

DO 404 I=2,L2

IF(I= =L2) THEN

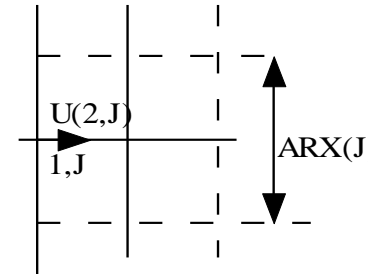
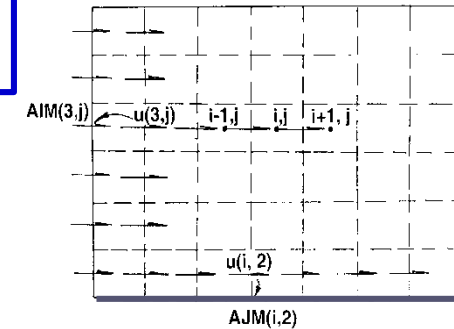
ARHO=ARX(J)*RHO(L1,J)

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

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

ELSE

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



$$\begin{cases} \Delta X \cdot 1 & r = 1 \text{ Cartesian} \\ r \cdot \theta(\theta = 1) \cdot \Delta x & \text{Cylindrical} \\ r \Delta \theta \cdot 1 & \text{Polar} \end{cases}$$

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)
!Store  $a_E, a_W, a_N, a_S$  for p-correction equation
! while CON (b) and AP (aP) are not stored; Because AP has been
! underrelaxed, and the velocity in 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

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.
```

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

! 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
    
```

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

$RHO(I,J)=RHO(I,J)*CP(I,J)$! This is the temperature

ENDDO

ENDDO

REL=1.-RELAX(NF)

```
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)+
```

```
1 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)+
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  $\rho/\Delta t$  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))
1/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
DO I=1,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
    
```

$$b = S_C \Delta V + a_P^0 \phi_P^0$$

$$a_P = a_E + a_W + a_N + a_S + a_P^0 - S_P \Delta V$$

Transient,
Linear----
Steady,
nonlinear

8.6.2.6 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)

!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)

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

C*****

ENTRY **PRINT** ! For print out, NF=3
IF(LPRINT(3)) THEN represents stream function

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)

1*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

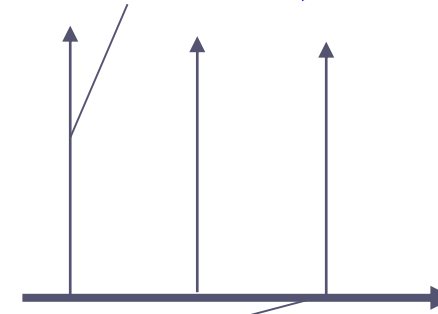
81 ENDDO

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

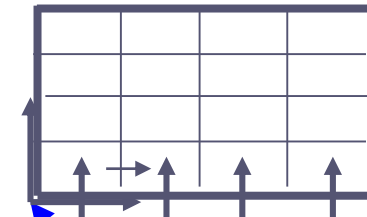
For bottom, from left to right $\psi_{i,2} = \psi_{i-1,2} - \sum_{i=3} \rho_{i-1,1} v_{i-1,2} r(1) \Delta x_i$

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

DO 82 J=3,M1



DO 82 I=2,L1



F(2,2,3)=0

```
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
```



```
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 too 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 53,(X(I),I=IBEG,IEND) ! Print out x-coordinates
    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, in**
NF=N **Page 29 of the PPT**
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)

```
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
```

Data format of TECPLOT

```
DO I=1, L1
```

```
WRITE(9, ' (/ , E11. 3, E11. 3, $)') X(I), Y(J)
```

```
DO NF=1, NCP
```

Data format of TECPLOT

```
IF (LPRINT(NF)) THEN
```

```
FSHOW=F(I, J, NF)
```

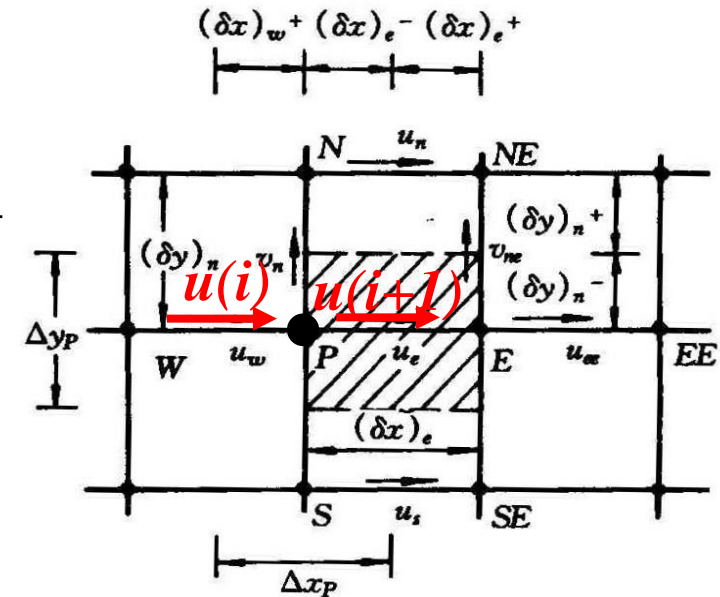
```
IF (NF==1) THEN
```

```
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
```



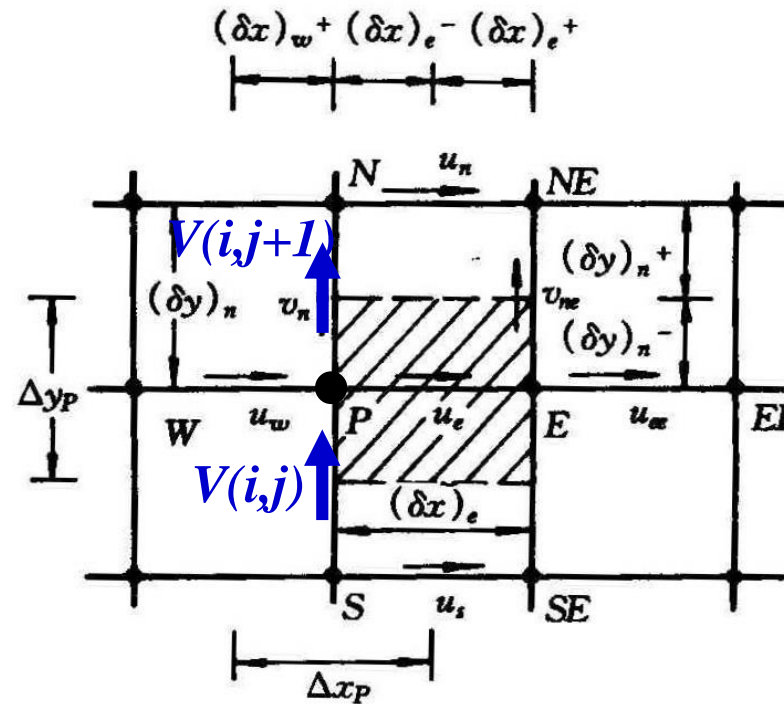
} **For u**

```

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



! WRITE(8,10) TITLE(NF) ! 10 FORMAT(1X,26(1H*),3X,A10,3X,26(1H*))

```
***** .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
```

! 40 FORMAT(1X,I3,2X,1P7E9.2)

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)

本组网页地址: <http://nht.xjtu.edu.cn> 欢迎访问!
Teaching PPT will be loaded on ou website



同舟共济
渡彼岸!

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