**第二次作业-团队作业**

**第一组: 马誉高 刘倩 刘俊宏**

**一、题目**

|  |
| --- |
| **已知变量 φ 的输运过程由如下对流—扩散方程控制**      **对于如图 1 所示的计算域 ，其对应的边界条件为：**    **图 1 计算域示意图**  **若已知：L=1.0 m、=1.0 kg/m3、=0.1 kg/(ms)，要求：（1）将该计算域划分为 5 个均分的网格，分别采用中心差分格式、上风格式、混合格式（仅对 u=2.5 m/s）三种离散格式，在（i）u=0.1 m/s、（ii）u=2.5 m/s 的条件下进行求解；（2） 采用 Fortran 语言编写计算机程序代码；（3）分别以图和表的形式，将上述三种 离散格式的计算结果与精确解的计算结果进行比较。** |

**二、网格划分**

根据题目要求，划分网格



**三、差分网格**

1. 解析解的形式

该问题的解析求解的结果是



2. 对于中心差分格式

（1）对于中间的网格



即，



统一写成：



在本例程中，



（2）右边界网格

由于边界条件的特殊性，对（1）式进行差分时，有





在本例程中，，可以简化为



统一写成：



其中，



（3）左边边界网格

由于边界条件的特殊性，对（1）式进行差分时，有





在本例程中，，可以简化为



统一写成：



其中，



1. 对于上风格式
2. 对于中间的网格



即，



统一写成：



在本例程中，



（2）右边界网格

由于边界条件的特殊性，对（1）式进行差分时，有



在本例程中，，可以简化为



统一写成：



其中，



（3）左边边界网格

由于边界条件的特殊性，对（1）式进行差分时，有



在本例程中，，可以简化为



统一写成：



其中，



1. 对于混合格式

混合格式在控制体界面处可表示为



本质是根据对流与扩散的强度比*P*来选择中心差分格式和迎风格式。其格式不赘述。

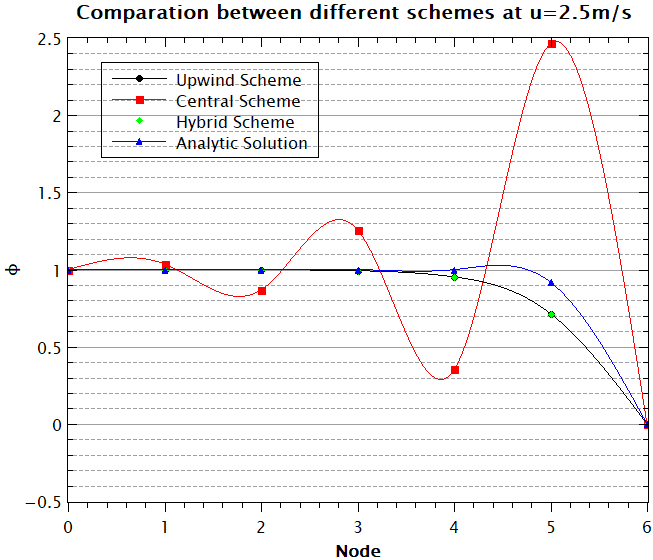
**四、计算结果**

Table 1. Comparation between different schemes at u=0.1m/s

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Node\_0 | Node\_1 | Node\_2 | Node\_3 | Node\_4 | Node\_5 | Node\_6 |
| Analytic | 1 | 0.938793 | 0.79639 | 0.622459 | 0.41002 | 0.150545 | 0 |
| Central | 1 | 0.94211 | 0.800601 | 0.627646 | 0.416256 | 0.15789 | 0 |
| Upwind | 1 | 0.933733 | 0.787947 | 0.613003 | 0.403071 | 0.151151 | 0 |

Table 2. Comparation between different schemes at u=2.5m/s

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Node\_0 | Node\_1 | Node\_2 | Node\_3 | Node\_4 | Node\_5 | Node\_6 |
| Analytic | 1 | 1 | 1 | 0.999996 | 0.999447 | 0.917915 | 0 |
| Central | 1 | 1.03563 | 0.869355 | 1.257331 | 0.352053 | 2.46437 | 0 |
| Upwind | 1 | 0.999843 | 0.99874 | 0.992126 | 0.952441 | 0.714331 | 0 |
| Hybrid | 1 | 0.999843 | 0.99874 | 0.992126 | 0.952441 | 0.714331 | 0 |

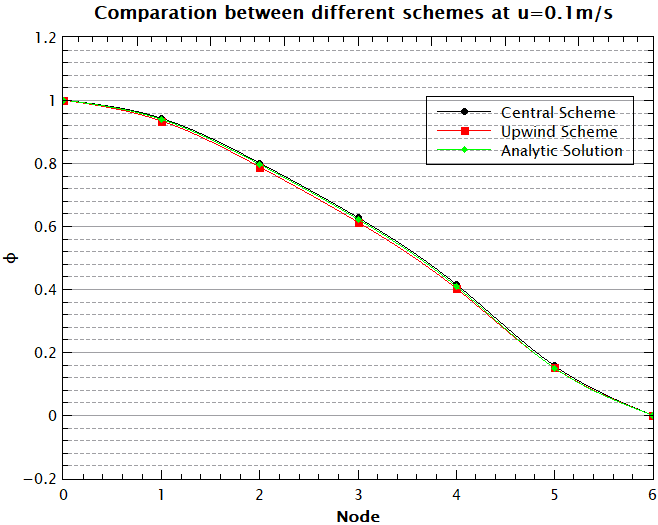


当u=2.5 m/s，



此时对流占主导，迎风格式优于中心差分。

而混合格式在该情形下，退化为迎风格式，和迎风格式精度无差别。



当u=0.1 m/s,



此时扩散占主导，中心差分优于迎风格式。

**附:**

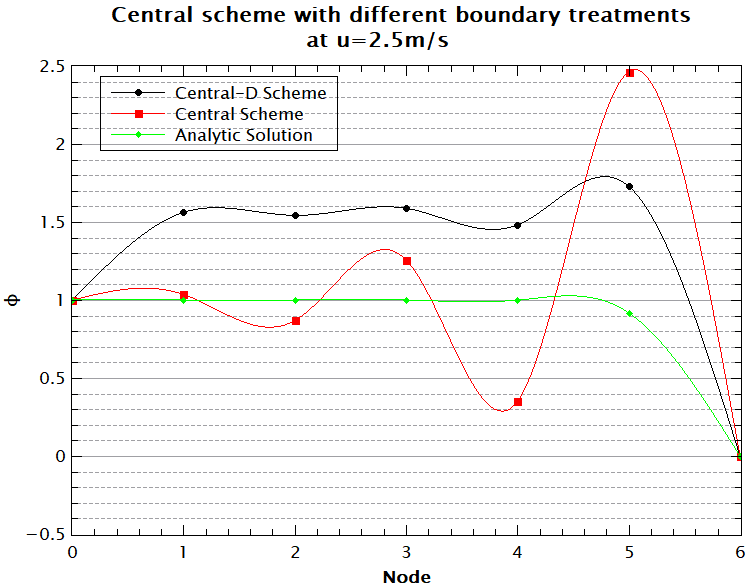
**有关边界处的控制体选择的进一步说明**

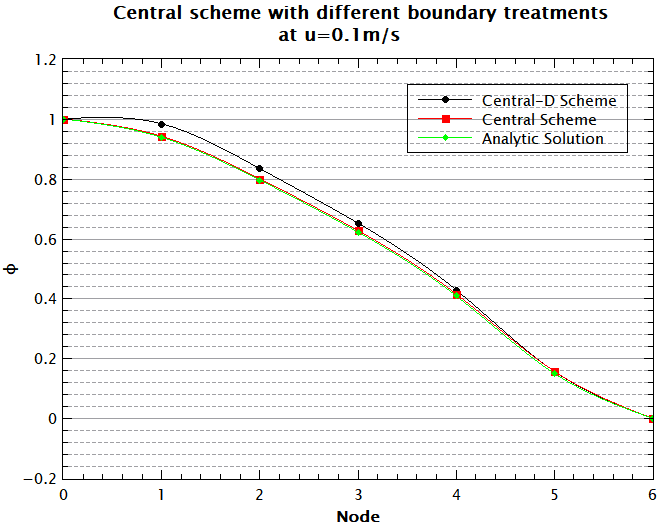
对于边界处，其实两种控制体的选择方式，如下图（以左边界为例）：



我们组对控制体的两种格式，针对迎风格式、中心格式的情形都进行了计算，结果表明，控制体格式2计算精度更高。

下面以中心差分格式为例，进行说明：





图中Central-D Scheme表征格式1，Central Scheme表征格式2，Analytic solution是解析解的结果。

根据图可以大致判断，第二种格式相对好。因此，我们在这次计算中，采用的是控制体格式2.

**附2：**

**全部代码**

|  |
| --- |
| ! -------------------------------------------------------  ! Campare Discrete Format:Exponential format  ! Group1  ! 2018.4.9  ! -------------------------------------------------------  PROGRAM Teamwork1  IMPLICIT NONE  REAL::Velo  INTEGER::N  WRITE(\*,\*) 'INPUT THE NUMBER OF CELLS'  READ(\*,\*)N  WRITE(\*,\*)'INPUT THE INLET VELOCITY'  READ(\*,\*)Velo  CALL Center\_Format(N,Velo)  CALL Center\_D(N,Velo)  CALL Upwind\_D(N,Velo)  CALL Mixed\_D(N,Velo)  !IF (Velo==2.5) THEN  ! CALL Mixed\_D(N,Velo)  !END IF  CALL Analysic\_Sol(N,Velo)  CALL Analytic\_solutions(N,Velo)  Pause  END PROGRAM Teamwork1    !Center Format  SUBROUTINE Center\_D(N,Velo)  REAL::Velo  INTEGER::N  REAL::Len=1,Dens=1,Gama=0.1,D,F,Delta\_x  REAL,ALLOCATABLE::A\_e(:),A\_p(:),A\_w(:),X(:),Y(:),A(:),B(:),C(:),L(:),U(:),K(:)  ALLOCATE(A\_e(N))  ALLOCATE(A\_p(N))  ALLOCATE(A\_w(N))  ALLOCATE(X(N))  ALLOCATE(Y(N))  ALLOCATE(A(N))  ALLOCATE(B(N))  ALLOCATE(C(N))  ALLOCATE(L(N))  ALLOCATE(U(N))  ALLOCATE(K(N))  Delta\_x=Len/N  D=Gama/Delta\_x  F=Dens\*Velo  !the coefficient for the left node  !sencond boundary  A\_e(1)=D-F/2  A\_w(1)=0  A\_p(1)=3\*D  !the coefficient for the middle nodes  DO i=2,N-1,1  A\_e(i)=D-F/2  A\_w(i)=D+F/2  A\_p(i)=A\_e(i)+A\_w(i)  END DO  !the coefficient for the right node  !second boundary  A\_e(N)=2\*D-F/2  A\_w(N)=D+F/2  A\_p(N)=3\*D  DO i=1,N,1  A(i)=-A\_w(i)  B(i)=A\_p(i)  C(i)=-A\_e(i)  L(i)=B(i)  U(i)=C(i)  END DO  !left boundary  Y(1)=(2\*D+F)\*1  !right boundary  Y(N)=(2\*D-F)\*0  DO i=2,N-1,1  Y(i)=0  END DO  DO i=1,N-1,1  K(i)=A(i+1)/L(i)  L(i+1)=L(i+1)-U(i)\*K(i)  Y(i+1)=Y(i+1)-Y(i)\*K(i)  END DO  X(N)=Y(N)/L(N)  DO i=N-1,1,-1  X(i)=(Y(i)-U(i)\*X(i+1))/L(i)  END DO  OPEN(UNIT=1,FILE='Center\_D.txt')  DO i=1,N,1  WRITE(1,\*) X(i)  END DO  RETURN  END SUBROUTINE  !Upwind Format  SUBROUTINE Upwind\_D(N,Velo)  REAL::Velo  INTEGER::N  REAL::Len=1,Dens=1,Gama=0.1,D,F,Delta\_x,Fai\_0=1,Fai\_L=0,Zero=0;  REAL,ALLOCATABLE::A\_e(:),A\_p(:),A\_w(:),X(:),A\_c(:),d\_(:),c\_(:)  ALLOCATE(A\_e(N))  ALLOCATE(A\_p(N))  ALLOCATE(A\_w(N))  ALLOCATE(A\_c(N))  ALLOCATE(X(N))  ALLOCATE(d\_(N))  ALLOCATE(c\_(N))  Delta\_x=Len/N  D=Gama/Delta\_x  F=Dens\*Velo  !边界条件  A\_e(1)=D+MAX(-F,Zero)  A\_w(1)=0  A\_p(1)=A\_e(1)+2\*D+MAX(F,Zero)  A\_c(1)=(2\*D+MAX(F,Zero))\*Fai\_0  A\_e(N)=0  A\_w(N)=D+MAX(F,Zero)  A\_p(N)=A\_w(N)+2\*D+MAX(-F,Zero)  A\_c(N)=(2\*D+MAX(-F,Zero))\*Fai\_L  DO i=2,N-1,1  A\_e(i)=D+MAX(-F,Zero)  A\_w(i)=D+MAX(F,Zero)  A\_p(i)=A\_e(i)+A\_w(i)  A\_c(i)=0  END DO  call TDMA(A\_p,A\_e,A\_w,A\_c,N,X)  OPEN(UNIT=1,FILE='Upwind\_D.txt')  DO i=1,N,1  WRITE(1,\*) X(i)  END DO  RETURN  END SUBROUTINE  !Mixed Format  SUBROUTINE Mixed\_D(N,Velo)  REAL::Velo  INTEGER::N  REAL::Len=1,Dens=1,Gama=0.1,D,F,Delta\_x,Fai\_0=1,Fai\_L=0,Zero=0,P=0  REAL,ALLOCATABLE::A\_e(:),A\_p(:),A\_w(:),X(:),A\_c(:),d\_(:),c\_(:)  ALLOCATE(A\_e(N))  ALLOCATE(A\_p(N))  ALLOCATE(A\_w(N))  ALLOCATE(A\_c(N))  ALLOCATE(X(N))  ALLOCATE(d\_(N))  ALLOCATE(c\_(N))  Delta\_x=Len/N  D=Gama/Delta\_x  F=Dens\*Velo  P=F/D  !边界条件  !比较糟糕的一种做法，理论上应该一个一个判断  IF (ABS(P)<2 ) THEN  A\_e(1)=D-F/2  A\_w(1)=0  A\_p(1)=3\*D+F/2  A\_c(1)=(F+2\*D)  A\_e(N)=0  A\_w(N)=D+F/2  A\_p(N)=3\*D-F/2  A\_c(N)=0  ELSE  A\_e(1)=D+MAX(-F,Zero)  A\_w(1)=0  A\_p(1)=A\_e(1)+2\*D+MAX(F,Zero)  A\_c(1)=(2\*D+MAX(F,Zero))\*Fai\_0  A\_e(N)=0  A\_w(N)=D+MAX(F,Zero)  A\_p(N)=A\_w(N)+2\*D+MAX(-F,Zero)  A\_c(N)=(2\*D+MAX(-F,Zero))\*Fai\_L  END IF  DO i=2,N-1,1  IF (ABS(P)<2 ) THEN  A\_e(i)=(D-F/2)  A\_w(i)=(D+F/2)  A\_p(i)=2\*D  A\_c(i)=0  ELSE  A\_e(i)=D+MAX(-F,Zero)  A\_w(i)=D+MAX(F,Zero)  A\_p(i)=A\_e(i)+A\_w(i)  A\_c(i)=0  END IF  END DO  call TDMA(A\_p,A\_e,A\_w,A\_c,N,X)  OPEN(UNIT=1,FILE='Mixed\_D.txt')  DO i=1,N,1  WRITE(1,\*) X(i)  END DO  RETURN  END SUBROUTINE  !Analysic Solution  SUBROUTINE Analysic\_Sol(N,Velo)  REAL::Velo  INTEGER::N  REAL::Len=1,Dens=1,Gama=0.1,Delta\_x  REAL,ALLOCATABLE::X(:)  Delta\_x=Len/N  ALLOCATE(X(N+2))  X(1)=1  X(N+2)=0;  DO i=2,N+1,1  X(i)=-1\*(EXP(Dens\*Velo\*(i-1.5)\*Delta\_x/Gama)-1)/(EXP(Dens\*Velo\*Len/Gama)-1)+1  END DO  OPEN(UNIT=1,FILE='Analysic\_Sol.txt')  DO i=1,N+2,1  WRITE(1,\*) X(i)  END DO  RETURN  END SUBROUTINE  !Analytic Solution  !edit by mayugao  !和刘倩讨论后合并  SUBROUTINE Analytic\_solutions(N,Velo)  REAL::Velo  INTEGER::N  REAL::Len=1,Dens=1,Gama=0.1,Delta\_x,P,Fi\_L=0,Fi\_0=1  REAL,ALLOCATABLE::X(:)  ALLOCATE(X(N+2))  Delta\_x=Len/N  P=Dens\*Velo\*Len/Gama;  DO i=1,N,1  X(i)=(Fi\_L-Fi\_0)\*(EXP(P\*(real(i)-0.5)/N)-1)/(EXP(P)-1)+Fi\_0;  END DO  OPEN(UNIT=1,FILE='Analytic\_Solution.txt')  DO i=1,N,1  WRITE(1,\*) X(i)  END DO  RETURN  END SUBROUTINE  !Center Format  !mayugao edit  SUBROUTINE Center\_Format(N,Velo)  REAL::Velo  INTEGER::N  REAL::Len=1,Dens=1,Gama=0.1,D,F,Delta\_x  REAL,ALLOCATABLE::A\_e(:),A\_p(:),A\_w(:),X(:),A\_c(:),d\_(:),c\_(:)  ALLOCATE(A\_e(N))  ALLOCATE(A\_p(N))  ALLOCATE(A\_w(N))  ALLOCATE(A\_c(N))  ALLOCATE(X(N))  ALLOCATE(d\_(N))  ALLOCATE(c\_(N))  Delta\_x=Len/N  D=Gama/Delta\_x  F=Dens\*Velo  A\_e(1)=D-F/2  A\_w(1)=0  A\_p(1)=3\*D+F/2  A\_c(1)=(F+2\*D)  A\_e(N)=0  A\_w(N)=D+F/2  A\_p(N)=3\*D-F/2  A\_c(N)=0  DO i=2,N-1,1  !中心差分的系数  A\_e(i)=(D-F/2)  A\_w(i)=(D+F/2)  A\_p(i)=2\*D  A\_c(i)=0  END DO  call TDMA(A\_p,A\_e,A\_w,A\_c,N,X)  OPEN(UNIT=1,FILE='Center\_Format.txt')  DO i=1,N,1  WRITE(1,\*) X(i)  END DO  RETURN  END SUBROUTINE  ! TDMA算法  ! TRI-DIAGONAL SOLUTION ALGORITHM  ! NOTE: ARRAYS B AND D ARE DESTROYED IN THE PROCESS.  ! FOR THE SOLUTION OF: -A\_w\*T(I-1) + A\_p\*T(I) - A\_e\*T(I+1) = A\_c  ! 改编自柯道友老师的讲解版本  ! 可运行  SUBROUTINE TDMA(A\_p,A\_e,A\_w,A\_c,N,X)  INTEGER::N  REAL :: A\_e(N),A\_p(N),A\_w(N),X(N),A\_c(N)  A\_e(1)=(A\_e(1))/(A\_p(1))  A\_c(1)=A\_c(1)/(A\_p(1))  DO I = 2 , N  A\_e(I)=A\_e(I)/((A\_p(I))-A\_w(I)\*A\_e(I-1))  A\_c(I)=(A\_c(I)+A\_w(I)\*A\_c(I-1))/((A\_p(I))-A\_w(I)\*A\_e(I-1))  ENDDO  X(N)=A\_c(N)  DO I = N-1 , 1 , -1 !step size = -1  X(I)=A\_e(I)\*X(I+1)+A\_c(I)  ENDDO  END SUBROUTINE  ! A\_p\*T(I) = A\_w\*T(I-1)+A\_e\*T(I+1)+A\_c  ! TDMA求解方程  ! 可运行，接口和运行结果和上面的TMDA（）无差别  SUBROUTINE TDMA2(A\_p,A\_e,A\_w,A\_c,N,X)  INTEGER::N  REAL :: A\_e(N),A\_p(N),A\_w(N),X(N),A\_c(N),c\_(N),d\_(N)  c\_(1)=-A\_e(1)/A\_p(1)  d\_(1)=A\_c(1)/A\_p(1)  DO I=2,(N-1)  c\_(I)=(-A\_e(I))/(A\_p(I)-c\_(I-1)\*(-A\_w(I)))  d\_(I)=(A\_c(I)-d\_(I-1)\*(-A\_w(I)))/(A\_p(I)-c\_(I-1)\*(-A\_w(I)))  ENDDO  d\_(N)=(A\_c(N)-d\_(N-1)\*(-A\_w(N)))/(A\_p(N)-c\_(N-1)\*(-A\_w(N-1)))  X(N)=d\_(N)  DO I=1,(N-1)  X(N-I)=d\_(N-I)-c\_(N-I)\*X(N-I+1)  ENDDO  END SUBROUTINE |