

《计算与编程》第六次作业报告

王超 12031012

第一题

思路 and 结果:

1. 矩阵乘法

1.1 矩阵乘法

思路: 首先把包含矩阵信息的两个文件复制到当前目录, 查看矩阵的信息, 得知 M 为 4*3 的矩阵, N 为 3*3 的矩阵, 因此编写的 Matrix_multip.f90 子程序需要实现两个矩阵相乘的算法, 基于矩阵行列已知, 且计算后的矩阵行列也知道, 因此可以定义上述行列数矩阵的算法即可, 就不定义计算任意矩阵之间相乘的算法了, 具体代码如下, 其中 MN 是 M*N 的结果矩阵:

```
subroutine Matrix_multip(M,N,MN)
implicit none
integer :: i,j
real(8),dimension(4,3),intent(in) :: M
real(8),dimension(3,3),intent(in) :: N
real(8),dimension(4,3),intent(out) :: MN
do i = 1,4
do j = 1,3
    MN(i,j) = M(i,1)*N(1,j)+M(i,2)*N(2,j)+M(i,3)*N(3,j)
end do
end do
end subroutine Matrix_multip
```

1.2 编写主程序先读取 M.dat 和 N.dat 的数据, 并将其赋值给 M 和 N 矩阵, 因为矩阵行列已知, 直接先读取再赋值即可, 代码如下所示,

```

program Main
implicit none
integer :: u,i,j
real(8), dimension (4,3) :: M
real(8), dimension (3,3) :: N
real(8), dimension (4,3) :: MN
u=50
open(u,file='M.dat',status='old')
do i=1,4
read(u,*) (M(i,j),j=1,3)
enddo
close(u)
open(u,file='N.dat',status='old')
do i=1,3
read(u,*) (N(i,j),j=1,3)
enddo
close(u)

call Matrix_multip(M,N,MN)
open(unit=u,file='MN.dat',status='replace')
write(u,"(f8.1,f8.1,f8.1)" MN(1,1),MN(1,2),MN(1,3)
write(u,"(f8.1,f8.1,f8.1)" MN(2,1),MN(2,2),MN(2,3)
write(u,"(f8.1,f8.1,f8.1)" MN(3,1),MN(3,2),MN(3,3)
write(u,"(f8.1,f8.1,f8.1)" MN(4,1),MN(4,2),MN(4,3)
close(u)
end program Main

```

1.3 这部分可以直接编译和运行上述两个代码，具体代码和结果如下所示：

```

[ese-wangc@login03 fortran]$ gfortran -c Matrix_multip.f90
[ese-wangc@login03 fortran]$ gfortran Main.f90 Matrix_multip.o
[ese-wangc@login03 fortran]$ ./a.out

```

	M.dat		N.dat		MN.dat	
1	9.4889891	15.79952	9.2889578			
2	9.2889578	12.92396	5.8621211			
3	5.8621211	11.29471	14.04269			
4	1.9356927	18.60917	18.23201			

	M.dat		N.dat		MN.dat	
1	7.7234138	14.11560	1.4449604			
2	5.5518050	14.80624	14.04269			
3	0.5965542	18.58036	2.2660391			

	M.dat		N.dat		MN.dat	
1	166.5	540.5	256.6			
2	147.0	431.4	208.2			
3	116.4	510.9	198.9			
4	129.1	641.6	305.4			

2. 计算太阳天顶角

2.1 计算特定日期的偏角

思路：这部分需要先把日期手动转化为一年之中的第 N 天后再输入，因为如果输入类似 2020-12-20 的格式，还需要编写一个转化的代码，其实手动转化也还可以，能够实现输入第 N 天，然后返回该日期下的偏角计算结果。具体代码如下所示，在 `module Declination_angle` 里面编写可以计算第 N 天的偏角的函数，方便后面调用。其中 N 代表的是第 N 天，angle 代表计算的偏角，特别要注意角度和弧度的转化。

```
module Declination_angle
implicit none
real(4) :: N
real(4) :: angle
contains

subroutine cal_angle(N,angle)
implicit none
real(4) :: N
real(4) :: angle
angle = 23.45*sin((N+284)/365*360*(3.14159/180))
end subroutine cal_angle

end module Declination_angle
```

2.2 计算特定日期和时间的 AST

这部分为了简化在 Fortran 里面的计算步骤，时间输入需要转化为数值形式，例如时间为 8:30 时，需要先把 8:30 转化为 8.5 再输入进去，方便代码识别和计算。具体的代码如下所示，在 `module cal_AST` 里面定义 `calcu_ast` 的子程序，输入为 LST, N, Lon, 返回输出为 AST，同样注意弧度和角度的相互转化，计算过程根据学习资料的公式计算即可。

```
module cal_AST
implicit none
real(4) :: N0,LSTM0
real(4) :: angle0,lon0,et0,ast0,lst0
contains

subroutine calcu_ast(lst0,N0,lon0,ast0)
implicit none
real(4) :: N0,LSTM0
real(4) :: angle0,lon0,et0,ast0,lst0
angle0 = 360*(N0-81)/365
et0 = 9.87*sin(2*angle0*(3.14159/180))-7.53*cos(angle0*(3.14159/180))-1.5*sin(angle0*(3.14159/180))
LSTM0 = 15*floor(lon0/15)
ast0 = lst0+(4*(LSTM0-lon0)+et0)/60
!write(*,*) 'cal_AST:',lst0,angle0,et0,LSTM0,ast0,lon0
end subroutine calcu_ast

end module cal_AST
```

2.3 计算并且输出 SZA

这部分为主函数，需要调用之前的 module 以及子程序，同时这部分需要用 read 函数来读取输入，并且用 write 函数来显示输入的提示，主要是格式的说明，需要输入的变量有：第 N 天，经纬度以及当地时间。读取后再调用 `cal_angle` 和 `calcu_ast` 子程序，根据资料所给公式计算所需的 SZA 值即可，用 write 函数打印出 SZA。具体代码如下所示，计算过程需要注意角度和弧度的转化过程。

```
program Cal_SZA
use Declination_angle
use cal_AST

implicit none

real(4) :: N1,LSTM1,H1,SZA,cosSZA
real(4) :: lon1,lat1,lst1,ast1,angle1,et1

write(*,*) 'please input N, such as 202, instead of 2012-7-21'
read(*,*) N1
write(*,*) 'please input lon, such as 112.00, instead of 112.00°'
read(*,*) lon1
write(*,*) 'please input lat, such as 33.43, instead of 33.43°'
read(*,*) lat1
write(*,*) 'please input LST, such as 8.50, instead of 8:30'
read(*,*) lst1

call cal_angle(N1,angle1)
call calcu_ast(lst1,N1,lon1,ast1)

H1 = (ast1*60-720)/4
cosSZA = cos(lat1*3.14159/180)*cos(angle1*3.14159/180)*cos(H1*3.14159/180)+sin(lat1*3.14159/180)*sin(angle1*3.14159/180)
SZA = ACOS(cosSZA)/3.14159*180
!write(*,*) 'angle=',angle1
!write(*,*) 'AST=',ast1
write(*,*) 'So,the calculated SZA = ',SZA

end program Cal_SZA
```

2.4 创建以及编译

这部分主要是将上文两个模块和主程序连接到统一的库，并且进行编译，具体代码如下所示，最后的 `./Cal_SZA.x` 是运行代码。

```
[ese-wangc@login03 fortran]$ gfortran -c Matrix_multip.f90
[ese-wangc@login03 fortran]$ gfortran -c Declination_angle.f90
[ese-wangc@login03 fortran]$ gfortran -c cal_AST.f90
[ese-wangc@login03 fortran]$ ar rcvf libsolar.a Declination_angle.o cal_AST.o
r - Declination_angle.o
r - cal_AST.o
[ese-wangc@login03 fortran]$ gfortran Cal_SZA.f90 -o Cal_SZA.x -L. -lsolar
[ese-wangc@login03 fortran]$ ./Cal_SZA.x
```

2.5 计算最终的 SZA

在 2.4 中运行代码后，按照提示输入数据即可，最运行代码的 `./Cal_SZA.x` 是，最终会打印出所求的 SZA 值，具体代码如下所示，所求结果的单位是度（°）。

```
[ese-wangc@login03 fortran]$ ./Cal_SZA.x  
please input N, such as 202, instead of 2012-7-21  
355  
please input lon, such as 112.00, instead of 112.00°  
114.062996  
please input lat, such as 33.43, instead of 33.43°  
22.542883  
please input LST, such as 8.50, instead of 8:30  
14.5833  
So,the calculated SZA = 54.4795074
```