

# ESE5023 Assignment 6

12132198 Gong Guoqing

## 1. Matrix multiplication

1.1 [5 points] Write a program Main.f90 to read fortran\_demo1/M.dat as the matrix M, and fortran\_demo1/N.dat as the matrix N.

Main.f90:

```
program Main
implicit none
integer
real(8),dimension(:,,:),allocatable :: a, b, mcol, mrow, ncol, nrow, i, j
:: M, N

a=50
b=51
mcol=3
mrow=4
ncol=4
nrow=3

open(unit=a,file='M.dat',status='old')
open(unit=b,file='N.dat',status='old')
allocate(M(mrow,mcol))
allocate(N(nrow,ncol))
do i=1,mrow
  read(a,*) M(i,:)
enddo

do i=1,nrow
  read(b,*) N(i,:)
enddo

do i=1,mrow
  write(*,*) "Line",i,":",M(i,:)
enddo
do i=1,nrow
  write(*,*) "Line ",i,":",N(i,:)
enddo

deallocate(M)
deallocate(N)
end Program Main
```

Output:

```
[ese-gongqq@login02 fortran_demo1]$ ./Main.x
Line 1 : 19.480000000000000 15.789999999999999 19.280000000000001
Line 2 : 19.280000000000001 12.920000000000000 15.859999999999999
Line 3 : 15.859999999999999 11.289999999999999 14.039999999999999
Line 4 : 11.930000000000000 18.600000000000001 18.230000000000000
Line 1 : 7.719999999999999 4.110000000000000 1.439999999999999 4.799999999999999
Line 2 : 5.549999999999999 4.799999999999999 4.040000000000000 0.589999999999999
Line 3 : 0.589999999999999 8.580000000000001 2.259999999999999 7.719999999999999
[ese-gongqq@login02 fortran_demo1]$
```

1.2 [5 points] Write a subroutine `Matrix_multip.f90` to do matrix multiplication.

**Matrix\_multip.f90:**

```
gong — ssh ese-gongggq@172.18.6.175 — 142x29
subroutine Matrix_multip(M,N,MN)

implicit none

real(8),dimension(4,3),intent(in) :: M
real(8),dimension(3,4),intent(in) :: N
real(8),dimension(4,4),intent(out) :: MN
integer :: i,j,k
real(8) :: t

do i=1,4
  do j=1,4
    t=0
    do k=1,3
      t=t+M(i,k)*N(k,j)
    enddo
    MN(i,j)=t
  enddo
enddo

end subroutine Matrix_multip
~
~
~
"Matrix_multip.f90" 24L, 420C 1,1 全部
```

1.3 [5 points] Call the subroutine `Matrix_multip()` from `Main.f90` to compute  $M*N$ ; write the output to a new file `MN.dat`, values are in formats of f9.2.

**Main\_read.f90:**

```
gong — ssh ese-gongggq@172.18.6.175 — 142x51
Program Main_read
implicit none
integer :: a, b, mcol, mrow, ncol, nrow, i, j
real(8), dimension(:, :), allocatable :: M, N
real(8), dimension(4,4) :: MN
a=50
b=51
mcol=3
mrow=4
ncol=4
nrow=3
open(unit=a,file='M.dat',status='old')
open(unit=b,file='N.dat',status='old')

allocate(M(mrow,mcol))
allocate(N(nrow,ncol))

do i=1,mrow
  read(a,*) M(i,:)
enddo

do i=1,nrow
  read(b,*) N(i,:)
enddo

close(a)
close(b)

do i=1,mrow
  write(*,*) "Line ",i,":",M(i,:)
enddo

do i=1,nrow
  write(*,*) "Line ",i,":",N(i,:)
enddo

call Matrix_multip(M,N,MN)
do i=1,4
  write(*,*) "Line ",i,":",MN(i,:)
enddo
open(unit=a,file='new1.dat',status='replace')
do i=1,4
  write(a, '(f9.2)') MN(i,:)
enddo

close(a)
deallocate(M)
deallocate(N)
End Program Main_read
~
~
~
"Main_read.f90" 49L, 765C 41,1 全部
```

## Output:

```
[ese-gongggq@login02 fortran_demo1]$ vi Main_read.f90
[ese-gongggq@login02 fortran_demo1]$ gfortran Main_read.f90 Matrix_multip.f90 -o Main_read.x
[ese-gongggq@login02 fortran_demo1]$ ./Main_read.x
Line      1 : 19.4800000000000000      15.789999999999999      19.280000000000001
Line      2 : 19.280000000000001      12.920000000000000      15.859999999999999
Line      3 : 15.859999999999999      11.289999999999999      14.039999999999999
Line      4 : 11.930000000000000      18.600000000000001      18.230000000000000
Line      1 : 7.719999999999999      4.110000000000003      1.439999999999999      4.799999999999999
Line      2 : 5.549999999999999      4.799999999999998      4.040000000000000      0.589999999999997
Line      3 : 0.589999999999997      8.580000000000001      2.259999999999998      7.719999999999998
Line      1 : 249.39530000000002      321.2771999999999      135.4155999999998      251.6617000000000
Line      2 : 229.9049999999997      277.3356000000000      115.8036000000000      222.6059999999999
Line      3 : 193.3822999999999      239.8398000000000      100.1803999999999      191.1778999999999
Line      4 : 206.0852999999999      294.7256999999996      133.5230000000000      208.9736000000000
```

## vi new1.dat: output

```
gong — ssh ese-gongggq@172.18.6.175 — 152x51
321.28
135.42
251.66
229.90
277.34
115.80
222.61
193.38
239.84
100.18
191.18
206.09
294.73
133.52
208.97
~
~
~
```

## 2. Calculate the Solar Elevation Angle

2.1 [5 points] Write a module Declination\_angle that calculates the *declination angle* on a given date.

Declination\_angle.f90:

```
module Declination_angle
implicit none
real, parameter      :: pi=3.1415926536
contains
  subroutine cal_angle(m,d,da)
    implicit none
    integer,intent(in)      :: m, d
    real(8),intent(out)     :: da
    integer                 :: doy
    doy=(m-1)*30+d
    da=asin(sin(-23.44/180*pi)*cos(((360/365.24)*(doy+10)+360/pi*0.0167*sin(360/365.24*(doy-2)))/180*pi))
    da=da/pi*180
  end subroutine cal_angle
end module Declination_angle
```

1,1 全部

Date.f90:

```
Program Date
use Declination_angle
implicit none
real(8)          :: angle
integer           :: m, d
m=12
d=18
call cal_angle(m,d,angle)
write(*,*) angle
end program Date
```

"Date.f90" 17L, 189C 17,1 全部

Output:

```
[ese-gongggq@login02 newdir]$ vi Declination_angle.f90
[ese-gongggq@login02 newdir]$ vi Date.f90
[ese-gongggq@login02 newdir]$ gfortran Date.f90 Declination_angle.f90 -o Date.x
[ese-gongggq@login02 newdir]$ ./Date.x
-23.335786319814492
```

**2.2 [10 points]** Write a module `Solar_hour_angle` that calculates the *solar hour angle* in a given location for a given date and time.

**Solar\_hour\_angle.f90:**

```
module Solarhourangle
implicit none
real, parameter :: pi=3.1415926536
contains
  subroutine cal_sha(lon,m,d,t,sha)
    implicit none
    integer,intent(in) :: m, d
    real(8),intent(in) :: lon, t
    real(8),intent(out) :: sha
    integer :: doy
    real(8) :: offset, eot, gam
    doy=(m-1)*30+d
    gam=2*pi/365*(doy-1+(t-12)/24)
    eot=229.18*(0.000075+0.001868*cos(gam)-0.032077*sin(gam)-0.014615*cos(2*gam)-0.040849*sin(2*gam))
    offset=eot*MOD(lon,15.0)
    sha=15*(t-12)+offset/60
  end subroutine cal_sha
end module Solarhourangle
```

"Solar\_hour\_angle.f90" 28L, 621C 18,3 全部

**Location.f90:**

```
program location
use Solarhourangle
implicit none
real(8) :: t,lon,h
integer :: m,d
t=18
lon=118.24
m=12
d=18
call cal_sha(lon,m,d,t,h)
write(*,*) h
end program location
```

"location.f90" 20L, 194C 13,1 全部

**Output:**

```
[ese-gongggq@login02 newdir]$ gfortran location.f90 Solar_hour_angle.f90 -o location.x
[ese-gongggq@login02 newdir]$ ./location.x
90.308419527005640
```

**Solar\_elevation\_angle.f90:**

**Output:**

**2.4 [5 points]** Create a library (libsea.a) that contains Declination\_angle.o and Solar\_hour\_angle.o. Compile Solar\_elevation\_angle.f90 using libsea.a. Print the SEA for Shenzhen (22.542883N, 114.062996E) at 10:32 (Beijing time; UTC+8) on 2021-12-31.

shenzhen.f90:

```

program shenzhen

use Declination_angle
use Solarhourangle

implicit none

real, parameter      :: pai=3.1415926536
real(8)              :: lat,lon,t,sha,da
integer              :: m,d
real(8)              :: SEA

lat=22.542883
lon=114.062996
t=10.0+32/60
m=12
d=31

call cal_angle(m,d,da)
call cal_sha(lon,m,d,t,sha)

SEA=asin(sin(lat/180*pai)*sin(da/180*pai)+cos(lat/180*pai)*cos(da/180*pai)*cos(sha/180*pai))
SEA=SEA/pai*180.0

write(*,*) SEA

end program shenzhen
~
~
~
~
~
~
"shenzhen.f90" 28L, 481C
10,1 全部

```

## Output:

```
[ese-gongqq@login02 newdir]$ vi shenzhen.f90
[ese-gongqq@login02 newdir]$ gfortran -c Declination_angle.f90
[ese-gongqq@login02 newdir]$ gfortran -c Solar_hour_angle.f90
[ese-gongqq@login02 newdir]$ ar rcvf libsea.a Declination_angle.o Solar_hour_angle.o
a - Declination_angle.o
a - Solar_hour_angle.o
[ese-gongqq@login02 newdir]$ gfortran shenzhen.f90 -o shenzhen.x -L. -lsea
[ese-gongqq@login02 newdir]$ ./shenzhen.x
    35.790305803200272
[ese-gongqq@login02 newdir]$
```

#####

**In 1.1, Peng Han explained to me the use of "allocate".**

**In 1.2, Peng Han explained to me how to create a "subroutine".**

**In 2.1, Peng Han explained to me the calculation of "declination angle" and the call of module.**

**In 2.4, Peng Han explained libsea.a to me.**

**Thanks a lot!**