

CLIM0660

**Introduction to modelling in
physical geography
Partim 2 : Climatology**

Fortran 1 + NOAA Ferret

NOAA/ PMEL

FERRET

An Analysis Tool for Gridded and Non-Gridded Data

1.1 Introduction

FORTRAN means "Formula Translating System". It is faster than C but less precise.

FORTTRAN I: originally developed by IBM in the 1950s for scientific and engineering applications

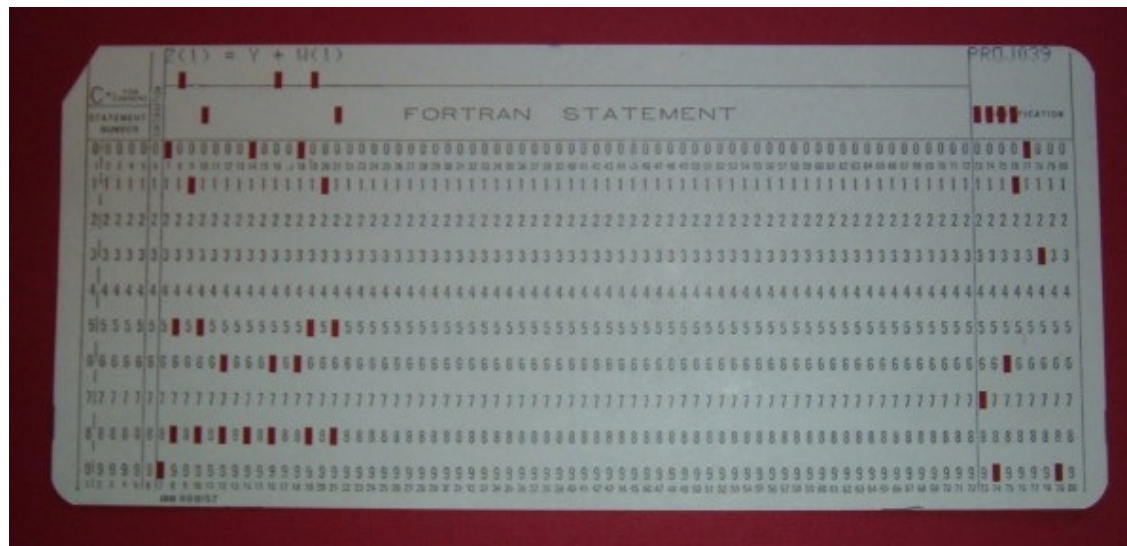
FORTTRAN 66: 1st official standard version. File extension: *.f66

Fortran 77: with the IF (...) THEN / ELSE / ENDIF. Some part of climate models are still written in f77.

Fortran 90: big revolution and a lot of restrictions (eg. number of characters by line) are withdrawn + possibility to work at the matrix level. File extension: *.f90

Fortran 95+2003: object oriented programming as C++

Fortran 2018: natively parallelised.



1.2 Introduction

To built a Fortran code:

```
gedit/nedit/... your_program.f90
```

If *.f or *.f77, instructions should be between character #6 and #72.

To compile:

```
gfortran -fcheck=all your_program.f90 (-o myprog.exe)
ifort -C your_program.f90 (-o myprog.exe)
```

To execute:

```
./myprogr.exe
```

*The execution is slower but
the array bounds are checked
for example.*



2.1 Basis – Kind of data

integer

=> integer*1 = 2^8 = - 128 à 128
=> integer*2 = 2^{16} = -32768 à 32768
=> integer*4 = 2^{32}

e.g.: 2, -12589, 20

real (float)

=> real*4 = simple precision (6 numbers after 0) till 10^{38}
=> real*8 = double precision (12 numbers after 0) till 10^{308}

e.g.: 2., -12.589e15, 3.1415d0

logical

e.g.: .true., .false.

character

e.g.: "hello"

double vs simple:
- 2 x more memory
- 2 x slower

2.2 Basis – Kind of data

Integer vs real

```
REAL          :: r, a, b
INTEGER       :: i
```

```
I = 1.15      ! => i = 1
R = 2/3        ! => r = 0.00000
R = 2./3.      ! => r = 0.6666667
r = 2d0/3d0    ! => r = 0.6666666666666667
```

```
r = 1/real(i)
r = 1/double(i)
```

```
i = 78 + int(r) 
```

```
r = max(a,b) or max(a,real(i)) or max(1.,a) or min(i,6)
```

```
But not r=max(i,6.)
```

E.g. :

```
print *,1/3, 1./3.,1d0/3d0
```

will show:

```
0    0.3333333    0.3333333333333333
```

2.3 Basis – Main structure

1st instruction

PROGRAM nom_du_programme

*can be in small
or in capital
letters*

*we need this
(as in Fortran, it is not
mandatory to declare
variables)*

IMPLICIT NONE

REAL :: r, aire
INTEGER :: i
REAL, PARAMETER :: pi=3.1415

*As in C, this allows
to write several instructions
on the same line*

variables declaration

r=10. ; i=2
aire=pi * r**2 + i

PRINT *,aire ! To print on the screen

*basis instructions : +, -, *, /, ***

END PROGRAM nom_du_programme

last instruction

comment

2.4 Basis – If and loop

combination of tests

(a>10 .and. b/=5)
(a.gt.10 .or. b.ne.5)

```
IF (r >= 5.) THEN
  r=r-5
ELSE
  r=r+5
ENDIF
```

>	.gt.
<	.lt.
>=	.ge.
==	.eq.
/=	.ne.

Be careful to have an ending test

```
DO i=2,454,2
  r=r+1./real(i)
ENDDO
```

By default, it is ,1

```
i=2 ; r=0.
DO WHILE (r <= 3)
  r=r+1./real(i)
  i=i+2
ENDDO
```

```
i=2 ; r=0.
DO i=2,1000,2
  r=r+1./real(i)
  IF (r >=3) THEN
    GO TO 1000
  ENDIF
ENDDO
```

*old instruction from Fortran 77
but still very useful!*

```
1000 CONTINUE
```

2.5 Basis – Array

```
INTEGER,PARAMETER :: dim_x=100
INTEGER,PARAMETER :: level=60
INTEGER            :: i,k
REAL               :: array(dim_x,20:level), matrix(10)
```

```
do i = 1, dim_x
  do k = 20, level
    array(i,k)=real(i)/real(k)
  enddo
enddo
```

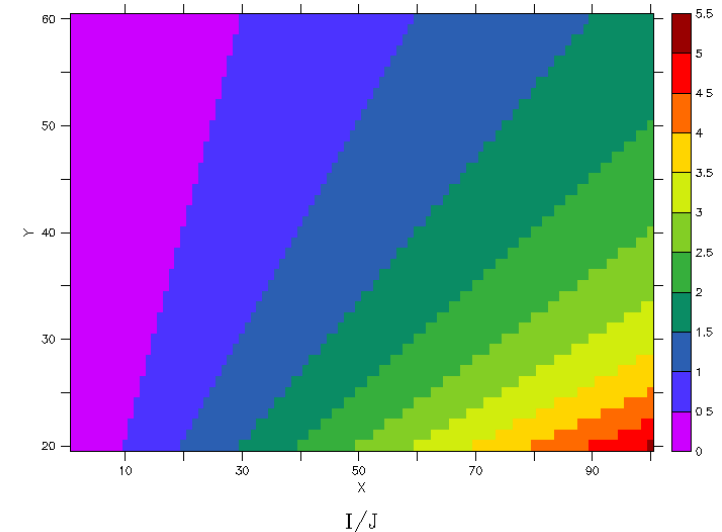
Indexes i,k of the matrix

```
array( :, :) = 1.
or array=1
```

every index is initialised to 1.

```
array(10,10) = 18
```

error if compiled with -C



By default, the 1st element is the index 1 and not the index 0 as in C.

in C

matrix(10)



in Fortran



2.6 Basis – Character

```
CHARACTER*30          :: file_name  
CHARACTER*4,PARAMETER :: ext=".txt"
```

```
file_name='mon_fichier'
```

```
print *,file_name//ext
```

to concatenate 2 strings

```
print *,TRIM(file_name)//ext
```

```
print *,INDEX(file_name,'f')
```

*to remove "white" space after
the string of characters*

```
print *,filename(2:5)
```

string of characters ~ array in fortran

print characters from #2 to #5

!!! A lot of other functions exist ==> see google

integer <=> character

```
do i=1990,2000  
  WRITE(YYYYc,'(i4)') i  
enddo
```

```
MMc="10"  
READ(MMc,'(i2)') i
```

3.1 Ascii files

ID of the file (should be >10)
unit=6 : screen

INTEGER day

OPEN(unit=**10**, file="my_file.txt", status="old", **err=1000**)

READ(**10**, *) day

CLOSE(**10**)

1000 continue

**WARNING: don't forget
to close the file**

file name (file= or name=)

error

Read or write in unit=10

file status

For reading

- **status='old'**: the file already exists

For writing:

- **status='new'**: a new file is built (it can't already exist)

- **status='replace'**: a new file is built or erased if it already exists

- **status='old', position='append'**: add to the end of a existing file

3.2 Ascii files

Most of the time a `read (10, *)` is enough as Fortran is able to read the CSV files, files with columns, ...

Fortran suggests the format

`WRITE(12,*) ...`

3 white space

10 times floats with 3+1 numbers

`WRITE(12, '(i6, 3x, f8.2, a10, 10(f5.1))') ...`

integer on 6 numbers

float on 8 numbers with 2 below zero

string of 10 characters

`WRITE(12,*) (array(i), i=1,100) ! on the same line`

`do i=1,100`

`READ(12,*) array(i)`

`enddo`

! on 100 lines

writing/reading of a table

3.3 Ascii files

to "not" read the 1st ten lines

```
do i=1,10  
  read(35,*)  
enddo
```

If the nbr of lines is not known

```
DO WHILE (.true.)  
  Read(..., end=1000)  
ENDDO  
1000 continue
```

```
do while(.true.)  
  READ(35, '(i4)', err=1009, end=1010) year  
enddo
```

end of file

1009 continue

...

1010 continue

...

```
CLOSE(35)
```

Error due to the format

WARNING: before adding `err=` and `end=`, check that your program crashes well due to these errors.

4.1 Function and subroutine

!! A variable in the main program is not known in the sub-programs.

```
real    tt(5), ave
integer i
tt(1)=6;tt(2)=8;tt(3)=-99.9;tt(4)=13;tt(5)=10
CALL check_time_series(tt)
print *,ave(tt,5)
end program
```



The argument must be what the sub-program is waiting.

```
SUBROUTINE check_time_series(ts)
implicit none
real ts(5)
integer i,j
do i=2,4
  if (ts(i)==-99.9) then
    ts(i)=(ts(i-1)+ts(i+1))/2.
  endif
enddo
END SUBROUTINE check_time_series
```

A subroutine manipulates the arguments

```
FUNCTION ave(to,n)
implicit none
integer n,i
real to(n),ave
ave=0
do i=1,n
  ave=ave+to(i)
enddo
ave=ave/real(n)
END FUNCTION ave
```

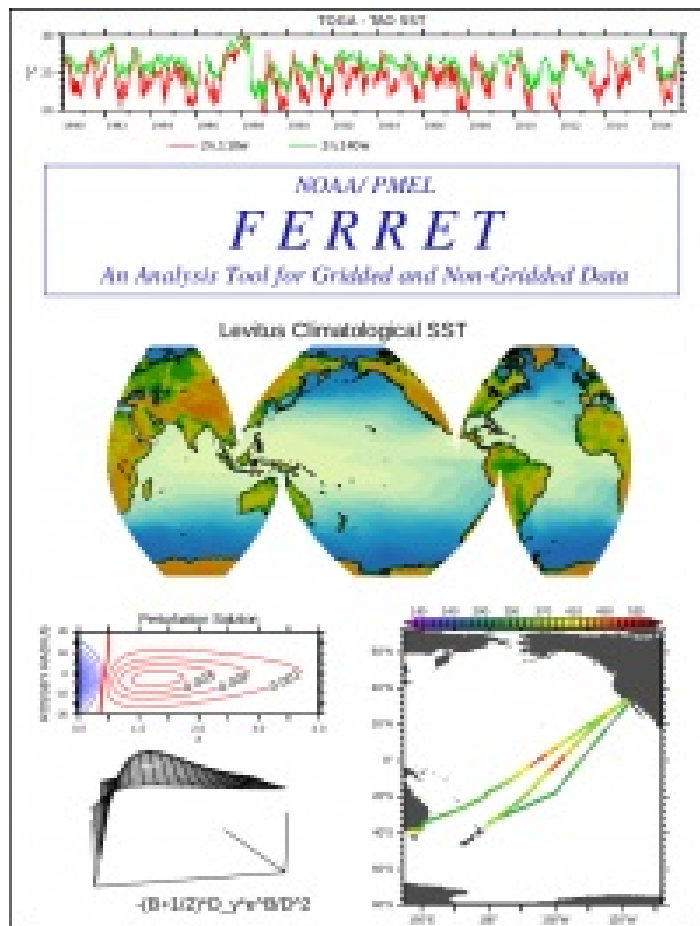
A function is an kind of data

5.1 Exercise 1

1. Show the 1000 first prime numbers knowing that $\text{mod}(i, j) = 0$ if i is divisible by j
2. Generate the calendar (including the day of the week) of the next 10 years.
3. With the help of `./generate1.bash` generating x files with \$DD \$MM, print the missing dates for a no leap year.
4. With the help of `./generate2.bash` generating 9 files with \$id \$evaluation (or \$id a), compute the average evaluation of the 9 students (\$id) when they are not absent.
5. On the basis of 6min weather measurements (`unzip data.zip`) at Sart-Tilman in January 2020, compute the average temperature (V_TD_1) and the standard deviation.

```
chmod +x generate.bash ; ./generate.bash
```

6.1 NOAA Ferret



Web site:

<https://ferret.pmel.noaa.gov/Ferret/>

User guide

Tutorial

To run ferret:

```
fettweis@xavier 10:29:54 ~ 1 > ferret
NOAA/PMEL TMAP
FERRET v7.43 (optimized)
Linux 3.10.0-862.11.6.el7.x86_64 64-bit
22-Apr-22 10:29
```

```
yes? plot/i=1:10 sin(i)
```

Or

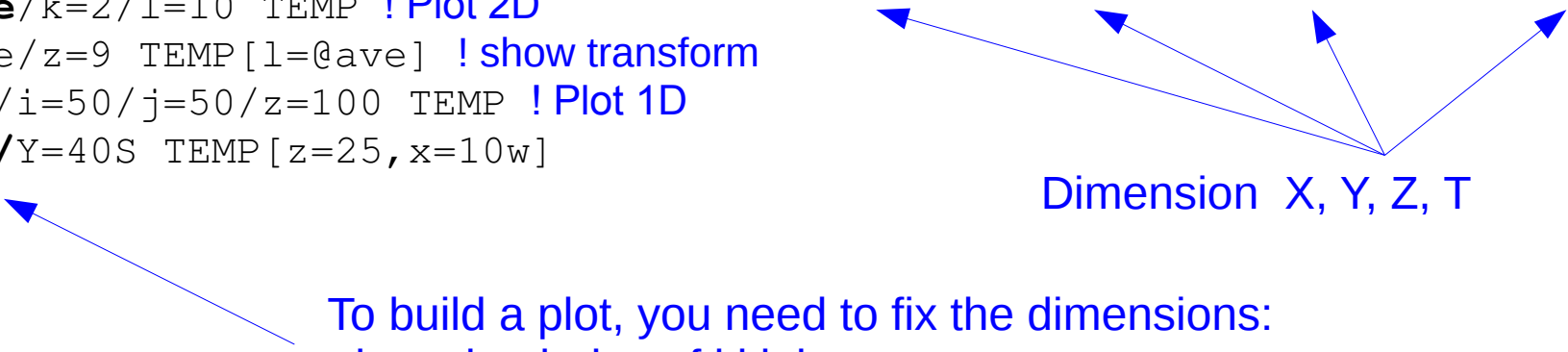
```
yes? go my_script.jnl
```

With `plot/i=1:10 sin(i)` into `my_script.jnl`

6.2 NOAA Ferret – plot 1D/2D

```
yes? use ocean_atlas_subset.nc      ! to open a netcdf file
yes? show data                      ! to show the content of the file
      currently SET data sets:
1> /opt/ferret/fer_dsets/data/ocean_atlas_subset.nc (default)
name      title
TEMP      Temperature
I          1:180
J          1:90
K          1:19
L          1:12
```

yes ? **shade**/k=2/l=10 TEMP ! Plot 2D
yes ? **shade**/z=9 TEMP[l=@ave] ! show transform
yes ? **plot**/i=50/j=50/z=100 TEMP ! Plot 1D
yes ? **plot**/Y=40S TEMP[z=25,x=10w]



Dimension X, Y, Z, T

To build a plot, you need to fix the dimensions:
- by using index of i,j,k,l
- by using values of x,y,z,t in the units of dimensions

```
yes ? list/X=10W/Y=40S/z=25 TEMP ! to list values
yes ? list/nohead/format=(f12.6)/file=toto.dat/clobber TEMP ! to list values in a file
```

6.3 NOAA Ferret – plot 1D/2D

```
use "/opt/ferret/fer_dsets/data/coads_climatology.cdf"
```

```
shade/l=10 SST
```

```
fill/title="my title" SST[l=@ave] ; go land
```

← title

```
fill/levels=(-2,32,2) SST[l=1] ; go land
```

← To select the level shown in the palette

```
fill/X=60W:20E/Y=70S:80N SST[l=2] ; go land  
vector/over uwnd[l=@ave],vwnd[l=@ave]
```

← To draw above

```
set region/l=7/X=25W:5E/Y=20N:70N ! To fix area  
set windows/aspect=1.2/size=1.0 ! to change the windows aspect
```

```
fill/title="wind"/levels"(-inf)(1,7,0.5)(inf)" (uwnd^2+vwnd^2)^0.5  
vector/color=2/lenght=5/xskip=1/yskip=1/l=7/over uwnd,vwnd  
contour/over/levels=20/color=4 SLP  
go land_detail
```

← By default, there are 7 colors

6.4 NOAA Ferret - variables

Graphic outputs:

```
shade SST[l=1]
frame/file=image.gif ! in gif (low resolution)
Or
set mode metafile
shade SST[l=1]
go (seb)output image 1 ! in png (high resolution)
```

```
cancel data/all
cancel var/all
cancel region/all
```

To remove everything

```
use "/opt/ferret/fer_dsets/data/etopo20.cdf"
use "/opt/ferret/fer_dsets/data/coads_climatology.cdf"
```

```
shade rose[d=1] ; go land
```

To use data set 1

```
set reg/X=30W:30E/Y=30N:70N
```

```
let topo = if ( rose[d=1] gt 0 ) then rose[d=1] else 0
```

To declare a variable with if then else

```
shade ignore0(topo) ; go land
vector/over/l=1 UWND[d=2],VWND[d=2]
```

6.5 NOAA Ferret – loop – read of files

Loops:

```
repeat/name=month/range=1:12 (\ ! to write on a new line
shade/l=`month`/lev=(-2,32,2) SST ;\
pause)
```

Read of files with axis:

To put Linux command into ferret

```
sp ← rm -f "yearly_wind.dat"
repeat/name=YYYY/range=1982:1993 (\
list/file="yearly_wind.dat"/format=(f12.2)/append \
UWND[t=01-JAN-`YYYY`:31-DEC-`YYYY`@ave])

define axis/t=1982:1993:1 axis_t ! time axis
define grid/x=UWND/y=VWND/t=axis_t grid_1 ! X and Y axis + grid
file/var=UWND_mean/col=1/grid=grid_1 "yearly_wind.dat"
save/file=yearly_wind.nc UWND_mean
```

Sometimes, we need to
define axis with units

```
define axis/t="01-JAN-2020 03:00":"31-DEC-2022 00:00":3/units=hours time
Define axis/y=40:90:1/units=degree latitude
```

6.6 NOAA Ferret – regridding

```
use monthly_navy_winds.cdf
```

```
! time axis
```

```
define axis/t=15-aug-1982:15-aug-1992:1/units=years time2
```

```
plot/i=1/j=1/d=1 UWND[gt=time2@ave]
```

```
plot/i=1/j=1/d=1 UWND[gt=time2]
```

Be careful to the leap years...

```
! x/y axis if data are in latitude/longitude
```

```
define axis/x=180w:180e:1/units=degrees xx
```

```
define axis/y=90s:90n:1/units=degrees yy
```

```
shade UWND[gx=xx, gy=yy]
```

```
save/file=toto.nc UWND[gx=xx, gy=yy] ! To save in a new file
```

6.7 NOAA Ferret – exercise 1

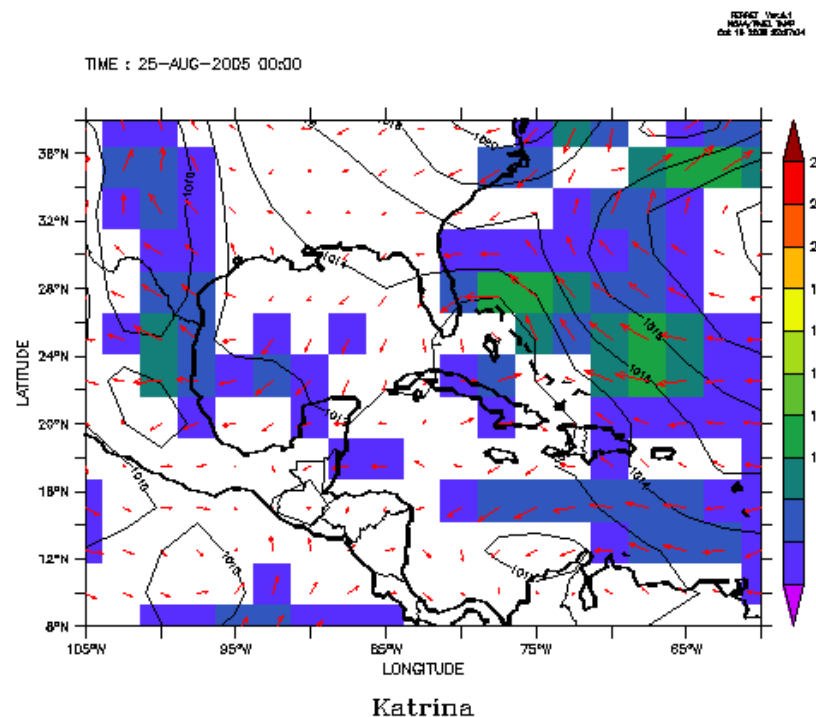
1. With the help of the NCEP reanalysis: <http://www.cdc.noaa.gov/cdc/data.ncep.reanalysis.html>, build an animation of a hurricane using

a) 6-hourly outputs.

b) the loop in ferret: `repeat/l=100:120`

c) `frame/file=fig-`l+100`.gif` and after

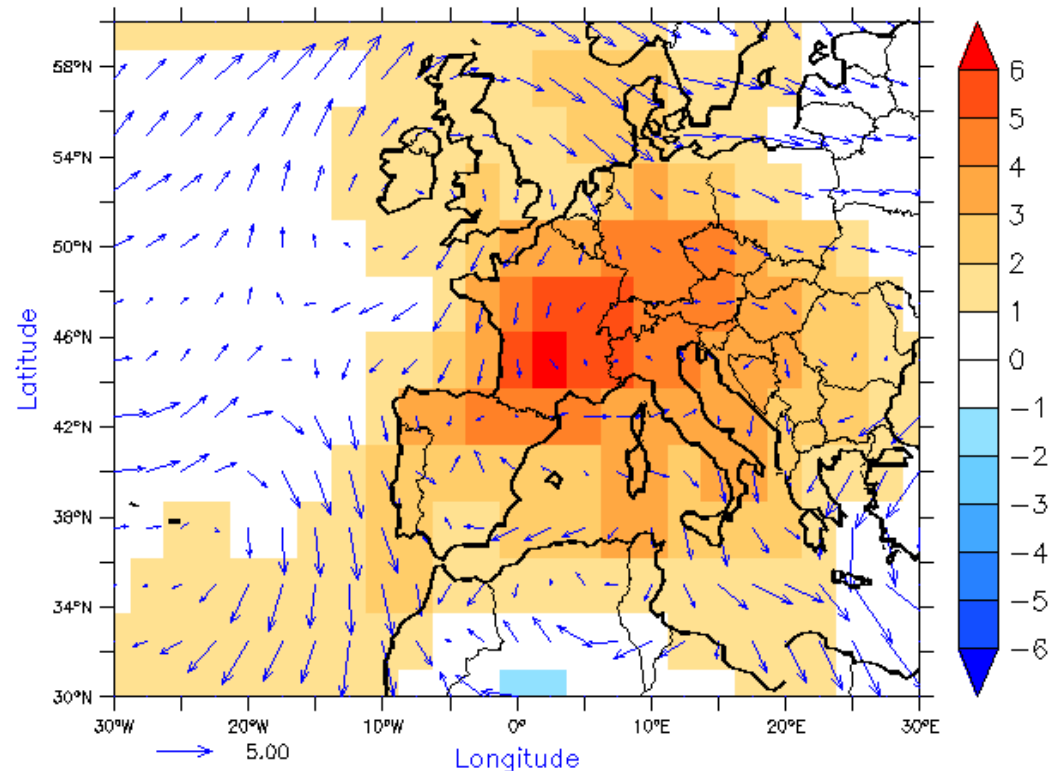
`gifsicle fig-*.gif > anim.gif ; gifview anim.gif`



6.8 NOAA Ferret – exercise 2

2. With the help of the NCEP reanalysis (monthly data), compute the temperature anomaly of the month August 2003. With `/palette=centered`, you can change the used palette (see in `/opt/ferret/ppl/`).

The 2003 August



Temperature anomaly on 1948–2007

7.1 Read/write NetCDF files

NetCDF (network Common Data Form)

In command line:

NetCDF type : ncdump -k

Conversion : nccopy -k 3 file.nc4 file.nc3

nccopy -k 4 -d 2 -s f.nc3 f.nc4

```
ncdump -h uwnd.sig995.2005.nc
```

```
netcdf uwnd.sig995.2005 {
```

```
dimensions:
```

```
lon = 144 ;
```

```
lat = 73 ;
```

```
time = UNLIMITED ; // (1460 currently)
```

```
variables:
```

```
float lat(lat) ;
```

```
lat:units = "degrees_north" ;
```

```
lat:actual_range = 90.f, -90.f ;
```

```
lat:long_name = "Latitude" ;
```

```
...
```

```
ncdump uwnd.sig995.2005.nc | less
```

```
netcdf uwnd.sig995.2005 {
```

```
...
```

```
data:
```

```
lat = 90, 87.5, 85, 82.5, 80, 77.5, 75, 72.5, 70, 67.5, 65, 62.5, 60, 57.5,  
55, 52.5, 50, 47.5, 45, 42.5, 40, 37.5, 35, 32.5, 30, 27.5, 25, 22.5, 20,  
17.5, 15, 12.5, 10, 7.5, 5, 2.5, 0, -2.5, -5, -7.5, -10, -12.5, -15,  
-17.5, -20, -22.5, -25, -27.5, -30, -32.5, -35, -37.5, -40, -42.5, -45,  
-47.5, -50, -52.5, -55, -57.5, -60, -62.5, -65, -67.5, -70, -72.5, -75,  
-77.5, -80, -82.5, -85, -87.5, -90 ;
```

dimension

variable name (as read by Fortran)

values

7.2 Read/write NetCDF files

Compilation: ifort libUN.f ex6_1.f90 -lnetcdf_ifort
 gfortran libUN.f ex6_1.f90 -lnetcdf -lnetcdf

```
integer, parameter :: dx=144
integer, parameter :: dy=73
integer, parameter :: dz=17
integer, parameter :: dt=365
integer i,j,k,t

real hgt(dx,dy,dz,dt), slp(dx,dy,dt), var2D(dx,dy), var3D(dx,dy,dz)

do t=1,dt ! To read/write a NetCDF, we need a loop on time
  call CF_READ2D("slp.2005.nc", 'slp', t, dx, dy, 1, var2D)

  call CF_READ3D("hgt.2005.nc", 'hgt', t, dx, dy, dz, var3D)
  do i=1,dx; do j=1,dy

    slp(i,j,t) = var2D(i,dy-j+1) ! Because the y axis is inverted
    do k=1,dz
      hgt(i,j,k,t)= var3D(i,dy-j+1,dz-k+1)
    enddo
  enddo ; enddo
enddo
```

7.3 Read/write NetCDF files

Axis reading of the NetCDF files (useful to build a new file with the same grid)

```
integer, parameter :: dy=73  
integer, parameter :: dz=17  
integer i,j,k,l  
real lat(dy),level(dz)
```

```
call CF_READ2D('slp.2005.nc','lat',1,dy,1,1,lat)
```

```
call CF_READ2D('hgt.2005.nc','level',1,dz,1,1,level)
```

float lat(lat) ;

lat:units = "degrees_north" ;
lat:actual_range = 90.f, -90.f ;
lat:long_name = "Latitude" ;

float level(level) ;

level:units = "millibar" ;
level:actual_range = 1000.f, 10.f ;
level:long_name = "Level" ;
level:positive = "down" ;

"x-axis"



WARNING:
axis is 1D and only
the 1st dimension is read

7.4 Read/write NetCDF files

Writing a NetCDF file (1/2)

```
call CF_INI_FILE('toto.nc','my NetCDF file') ! Name and title of the file
```

“time” axis is needed even for a 2D file

```
call CF_CREATE_DIM("time","-",1,1) ! Axis: name, units, dimension, value
call CF_CREATE_DIM("lon","degree",dx,lon)
call CF_CREATE_DIM("lat","degree",dy,lat)
```

no z axis => “-”

```
call CF_CREATE_VAR("mean","Mean SLP","hPa","-", "lon","lat","-" )
! Variable: name, description, units, axis name (T,X,Y,Z)
```

```
call CF_CREATE_FILE('toto.nc') ! building of the file
```

```
CALL CF_write ('toto.nc','mean',1,dx,dy,1,ave)
! writing of the file with the value of the table ave(dx,dy)
```

time step (1 if no time dimension)

7.5 Read/write NetCDF files

Writing a NetCDF file (2/2)

with a time dimension (here: monthly to yearly)

```
integer, parameter :: dt=720

real time(dt/12), ave(dx,dy)

do t=1,dt/12
  time(t)= real ((351      +(t+1947 -1902) *365      &
                  +(t+1947 -1901) / 4 + 366./2.) * 24 )
enddo

...

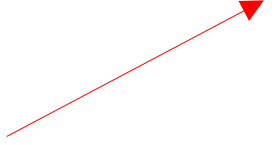
call CF_CREATE_DIM("time", "HOURS since 1901-01-15 00:00:00", dt/12, time)

...
do t=1,dt
  ave = xxx(t)
  CALL CF_write ('toto.nc', 'mean', t, dx, dy, 1, ave)
enddo
```

to be recognized by ferret



time step (a loop on time is needed !!)



7.4 NetCDF format – exercise 1

With the help of this Netcdf files

<https://downloads.psl.noaa.gov/Datasets/ncep.reanalysis.derived/surface/hgt.sfc.nc>

containing the topography used in NCEP-NCARv1, compute the spatial resolution of each pixel following both X and Y axis and plot the 2 resolution in 2D with the help of ferret.

This Fortran function can be used to compute the distance between 2 pixels (where lat/lon are given in radian and not in degree).

```
function distance(lon2,lat2,lon1,lat1)

  implicit none

  real,parameter :: R      = 6378.
  real :: lon1,lat1,lon2,lat2,distance
  real :: dlat,dlon,a,c

  dlat = (lat2-lat1)
  dlon = (lon2-lon1)
  a = sin(dLat/2.) * sin(dLat/2.) + cos(lat1) * cos(lat2) * sin(dLon/2.) * sin(dLon/2.)
  c = 2. * atan2(sqrt(a), sqrt(1.-a))
  distance = max(0.01,R * c)

end function
```

7.3 NetCDF format – exercise 2

On the basis of the CMIP6 daily outputs (see: <https://esgf-node.llnl.gov/search/cmip6/>), evaluate the future evolution of heat waves over a defined area (in lat/lon) using the TAS variables knowing that an heat wave is 5 days with $TAS > AVE(TAS_JJA) + 1 \times STD(TAS_JJA)$ with at least 3 days with $TAS > AVE(TAS_JJA) + 2 \times STD(TAS_JJA)$.

1. Evaluate the ability of 10 ESMs (~ 30 are available) over current climate (**Historical** scenario) with respect to NCEP-NCAR v1, to simulate the mean TAS_JJA over 1980-2010 and $STD(TAS_JJA)$ **using monthly outputs**. The ferret gx/gy transform can be used to put every model outputs on the same grid (limited to the studied area).

2. Select 3 of the 10 models, comparing the best with NCEP and representing the range/spread around the ensemble mean of the annual TAS evolution. Your selection needs to be justified.

3. Compute the evolution of nbr of heat waves (an heat wave of 6 days is considered as 2 heat waves) **using daily outputs** of the **SSP5-8.5** scenario.

4. Plot time series of mean nbr of heat waves by year over the studied area as well as compare 2071-2100 vs 1981-2010 in 2 dimensions: TAS_JJA and nbr of heat waves.

Source ID

☒ MPI-ESM1-2-LR (19)

Institution ID

Source Type

Nominal Resolution

Experiment ID

☒ historical (9)

☒ ssp585 (10)

Sub-Experiment

Variant Label

☒ r1i1p1f1 (19)

Grid Label

Table ID

Frequency

☒ day (10)

☒ mon (9)

Realm

Variable

☒ tas (19)

CF Standard Name

To concat Netcdf files: `ncrcat file1.nc file2.nc -o file.nc`

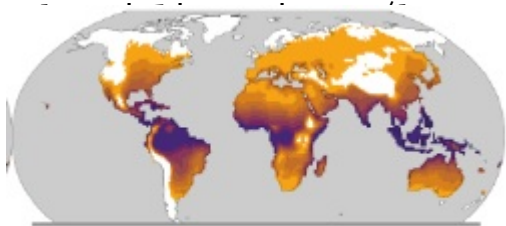
7.3 NetCDF format – exercise 3

On the basis of the CMIP6 daily outputs (see: <https://esgf-node.llnl.gov/search/cmip6/>), evaluate the future evolution of extreme days (TAS/HUMIDEX > 40°C) over a defined area (in lat/lon) using the daily TAS, HUS, PS, PR from IPSL-CM6A-LR/ssp585/r1i1p1f1 till 2300.

1. Evaluate the ability of IPSL to simulate current climate (JJA TAS/PR in 2D as well as extreme days averaged over your area of interest) with respect to ERA5 over 1981-2010
2. Plot in 2D the evolution of summer TAS/PR in 2100, 2200 and 2300 with respect to current climate (1981-2010) after having corrected TAS/PR with respect to ERA5
3. Plot in 1D the evolution of (corrected) extreme days using TAS and HUMIDEX for the main city of your area of interest.

```
let  eg_hPa  = sp *QQ / (622+.378*QQ)
let  HUMIDEX = TT + 5 * (eg_hPa - 10) / 9
```

where SP is the surface pressure en hPa, QQ the specific humidity and TT is the temperature in °C.



4.2 – 5.4°C

To concat Netcdf files: `ncrcat file1.nc file2.nc -o file.nc`

Source ID	-
☒ IPSL-CM6A-LR (12)	
Institution ID	+
Source Type	+
Nominal Resolution	+
Experiment ID	-
☒ ssp585 (12)	
Sub-Experiment	+
Variant Label	-
☒ r1i1p1f1 (12)	
Grid Label	+
Table ID	+
Frequency	-
☒ day (6)	
☒ mon (6)	
Realm	+
Variable	-
☒ hus (5)	
☒ pr (2)	
☒ ps (3)	
☒ tas (2)	
CF Standard Name	+

7.3 NetCDF format – exercise 4

On the basis of daily MAR outputs over Belgium:

1. Evaluate the ability (mean bias in 2D) of the 6 SSP370 MAR future projections to simulate P01/P99 of the Tmax or T min or Vmax over 1981-2010 with respect to MAR-ERA5.

```
CMCC-CM2-SR5_ssp370_r1i1p1f1  
EC-Earth3-Veg_ssp370_r1i1p1f1  
IPSL-CM6A-LR_ssp370_r1i1p1f1  
MIROC6_ssp370_r1i1p1f1  
MPI-ESM1-2-HR_ssp370_r1i1p1f1  
NorESM2-MM_ssp370_r1i1p1f1
```

2. On the basis of the CMIP6 outputs (<https://aims2.llnl.gov/search/cmip6/>), compute the 30 yrs periods corresponding to a +3°C world with respect to 1850-1900.

3. Shade the P99/P01 in +3°C world as well as change in nbr of days/yr using present day P99/P01 in a +3°C world using the 2 best CMIP6 models (i.e. with lowest bias over 1981-2010 with respect to MAR-ERA5)

4. Plot a time series of nbr days above the present day P99/P01 over 1980-2100 for Liège, Bastogne and Ostende.

To make a figure on the MAR grid: `shade sh,lon,lat ; go land_detail 7 overlay 1 4 4 4`

The MAR X/Y axis are: X19_1034_64 and Y17_8632_69 and not lat/lon! Units are km.

Be careful to the level scale and palette used in Ferret!

7.3 NetCDF format – exercise 5

On the basis of daily SST (+ anomaly) available on:

<https://www.ncei.noaa.gov/data/sea-surface-temperature-optimum-interpolation/v2.1/access/avhrr/>

and the ERA5 reanalysis (T2M, T850 and surface pressure), study a marine heat wave event (duration: ~ 1 month) over 2023, 2024 or 2025 over the El Nino, North Atlantic and Mediterranean areas.

1. Download the daily SST files (with `wget -r`) and build a yearly netcdf over your area of interest.
2. Download daily ERA5 and project the outputs on the same grid.
3. Identify an area in LAT/LON where you will extract a mean SST and check that the correlation with our SST vs the SST of each pixel is larger than 0.9.
4. Compute the correlation of each atmospheric pixels at 2m , 850hPa + surface pressure vs your mean SST. Apply a delay of 1-15 days between atmosphere and ocean to check if it improves the correlation i.e. if the marine heat wave is driven by atmosphere or ocean induces an atmospheric heat wave.
5. On the basis of monthly ERA5, compute the 2m, 850hPa, SST and surface pressure anomalies during your event vs 1981-2010.

7.3 NetCDF format – exercise 4

On the basis of daily MAR outputs over Belgium:

1. Evaluate the ability (mean bias in 2D) of the 6 SSP370 MAR future projections to simulate P01/P99 of the Tmax or T min or Vmax over 1981-2010 with respect to MAR-ERA5.

```
CMCC-CM2-SR5_ssp370_r1i1p1f1  
EC-Earth3-Veg_ssp370_r1i1p1f1  
IPSL-CM6A-LR_ssp370_r1i1p1f1  
MIROC6_ssp370_r1i1p1f1  
MPI-ESM1-2-HR_ssp370_r1i1p1f1  
NorESM2-MM_ssp370_r1i1p1f1
```

2. On the basis of the CMIP6 outputs (<https://aims2.llnl.gov/search/cmip6/>), compute the 30 yrs periods corresponding to a +3°C world with respect to 1850-1900.

3. Shade the P99/P01 in +3°C world as well as change in nbr of days/yr using present day P99/P01 in a +3°C world using the 2 best CMIP6 models (i.e. with lowest bias over 1981-2010 with respect to MAR-ERA5)

4. Plot a time series of nbr days above the present day P99/P01 over 1980-2100 for Liège, Bastogne and Ostende.

To make a figure on the MAR grid: `shade sh,lon,lat ; go land_detail 7 overlay 1 4 4 4`

The MAR X/Y axis are: X19_1034_64 and Y17_8632_69 and not lat/lon! Units are km.

Be careful to the level scale and palette used in Ferret!

8.1 Exam May 2022

On the basis of daily MAR outputs:

- write a Fortran program to find the closest pixel of a given city **(5/10)**.
- write a Ferret program to compute the climatology of the 19th May over 1981-2010 (i.e. the average and standard deviation of 30 values) for a given city **(3/10)**.
- write a Fortran program to compute the anomaly (absolute value and normalized value) of the 19th May 2022 with respect to 1981-2010 using the MAR based weather forecast product, available at a temporal resolution of 15min (1 day => 96 time steps) but on the same grid, for a given city **(2/10)**.

City: Liège, Ostende, Arlon, Mont-Rigi, ...

MAR files: /climato_tmp1/fettweis/MARdata/
repeat/name=YYYY ../ (use /climato_tmp1/fettweis/MARdata/ICE.`YYYY`.nc ; list ...
TT[k=1] ; cancel data/all)

To concat Netcdf files: `ncrcat file1.nc file2.nc -o file.nc`