Spatio-Temporal Data Analysis

Lecture 10

- Advance Environmental data analysis
- Spatial Analysis using gridded data

Programming and Debugging

Readability: It is important to keep your code readable. You wrote the code yourself, so you should know what it does

You do now, but will you remember what you did if you have to redo that analysis six months from now on new data. Besides, you may have to share your codes with other people. To keep your code readable

Debugging: Three Kind of Bugs

- Syntax errors: if you write code that R cannot understand, you have syntax errors. Syntax errors always result in an error message and often are caused by misspelling a function or forgetting a bracket.
- Semantic errors: If you write correct code that does not do what you think it does, you have a semantic error. The code itself is correct but the outcome of the code is not.
- Logic errors: probably the hardest-to-find errors. Your code works, it does not generate any errors or warning, but it still does not return the result you expect. The mistake is not code itself, but in the logic it executes.

Going Bug Hunting

A grammatically correct program may give you incorrect results due to logic errors. In case such errors (i.e. bugs) occur, you need to find out why and where they occur so that you can fix them. The procedure to identify and fix bugs is called “debugging”. R provides a number of tools for debugging such as:

traceback()

debug()

browser()

Linear Systems of Equations

- One of the most important application of matrices is for solving linear systems of equations which appear in many different problems including statistics, and numerical methods for differential equations.
- A linear system of equations can be written:

$$a_{11}x_1 + \ldots + a_{1n}x_n = b_1$$
$$a_{21}x_1 + \ldots + a_{2n}x_n = b_2$$
$$\vdots$$
$$a_{m1}x_1 + \ldots + a_{mn}x_n = b_m$$

- This is a system of m equations and n unknowns.

Linear Equations in R

Create Matrix i.e. A

```
A <- as.matrix(data.frame(c(1,-1,0),c(1,2,1),c(-2,1,-1)))
```

Element-wise Matrix Multiplication

```
A*A
```

Identity Matrix (3x3)

```
diag(3)
```

Calculate inverse of matrix

```
solve(A)
```

Matrix Determinant

```
det(A)
```

Transpose of Matrix

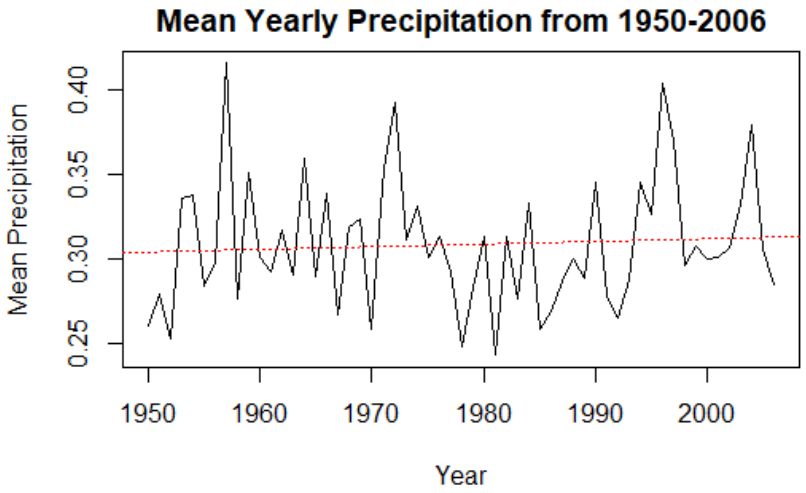
```
t(A)
```

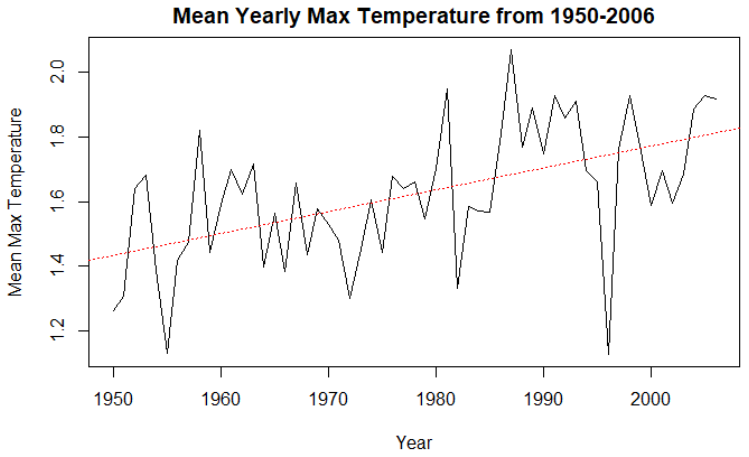
Find eigen values

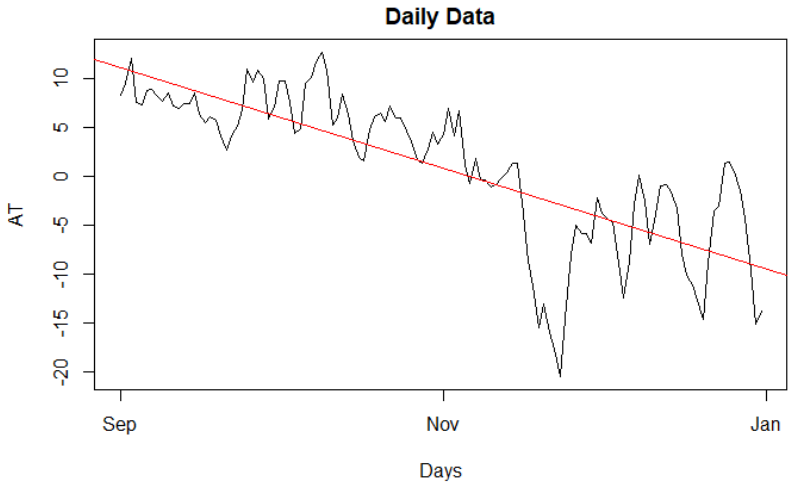
```
e <- eigen(A)
e$values # eigne values
e$vector
```

Try this in Rstudio

Data Manipulation and Analysis

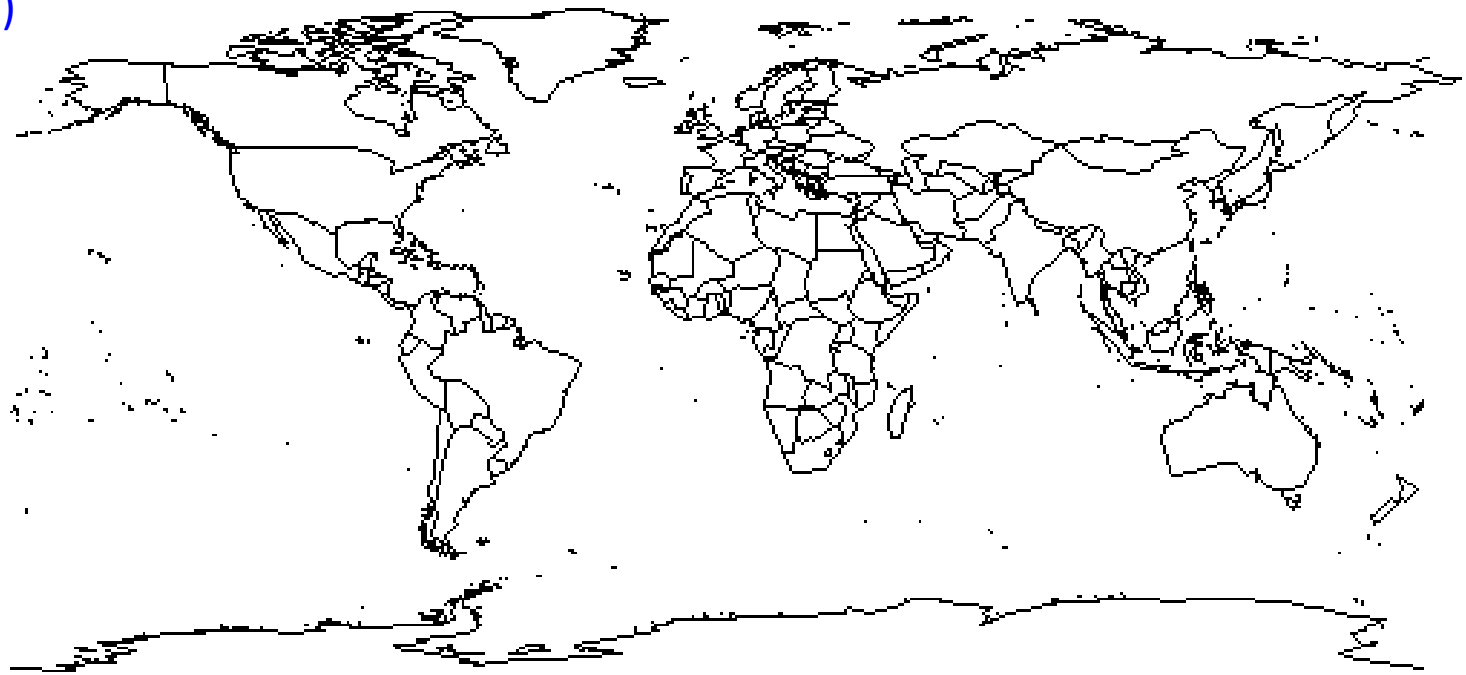






Library maps

```
library(maps)
map()
```



Library maps

```
map(xlim=c(5,15),ylim=c(53,58),fill=TRUE,col="red")
```

Library maps

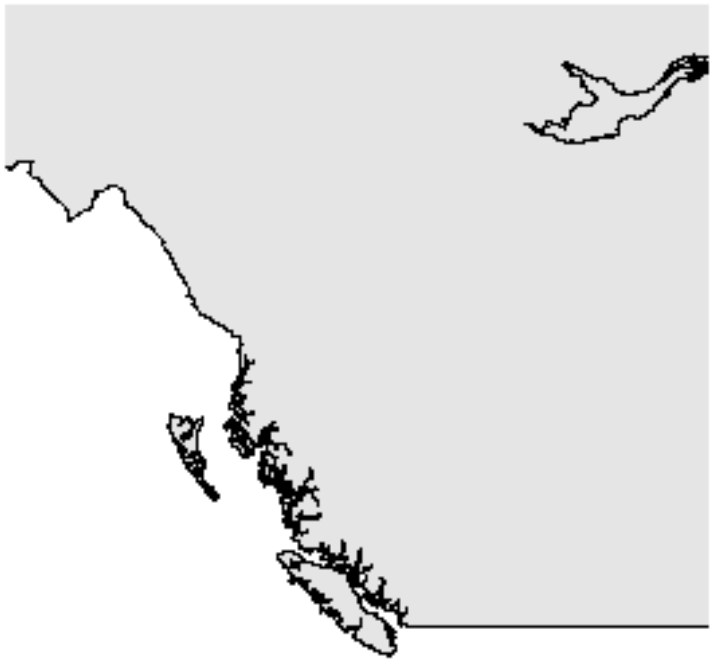
```
map(regions="Canada")
```

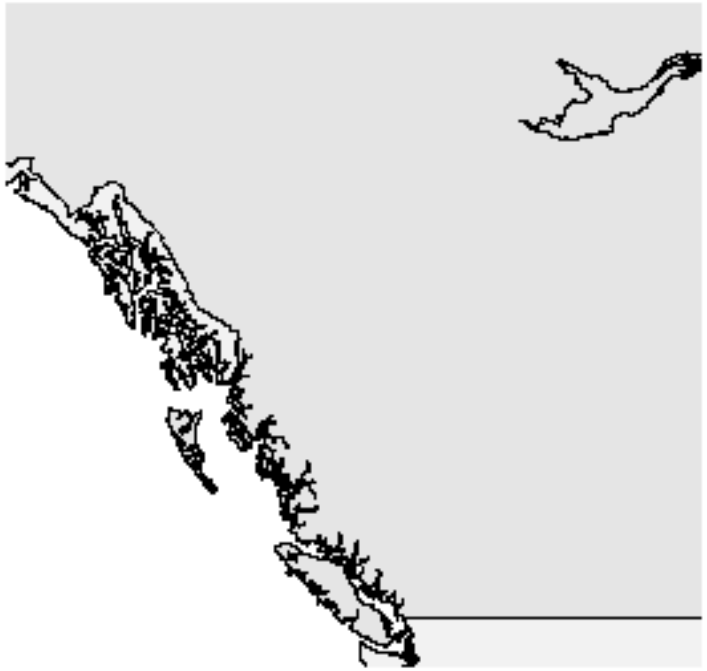
Library maps/mapproj

```
library(mapproj)
map("worldHires",xlim=c(-141,-53), ylim=c(40,85),fill=TRUE,col="red",
projection="conic",param=35)
```

Library maps/mapproj

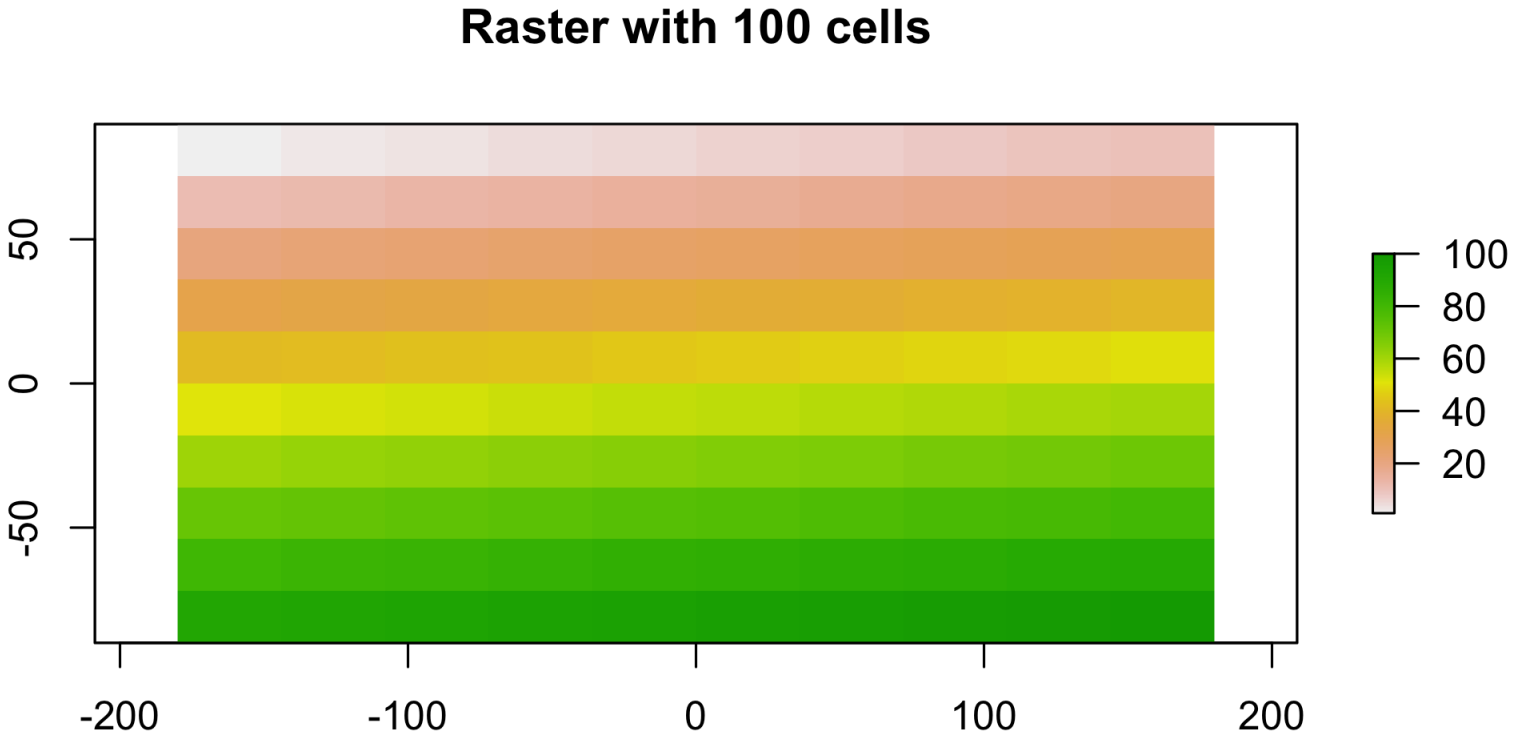
map("worldHires","Canada",xlim=c(-140,-110),ylim=c(48,64),col="gray90",fill=TRUE)





map("worldHires","Canada",xlim=c(-140,-110),ylim=c(48,64),col="gray90",fill=TRUE)
map("worldHires","usa",xlim=c(-140,-110),ylim=c(48,64),col="gray95",fill=TRUE,add=TRUE)

Raster Plots

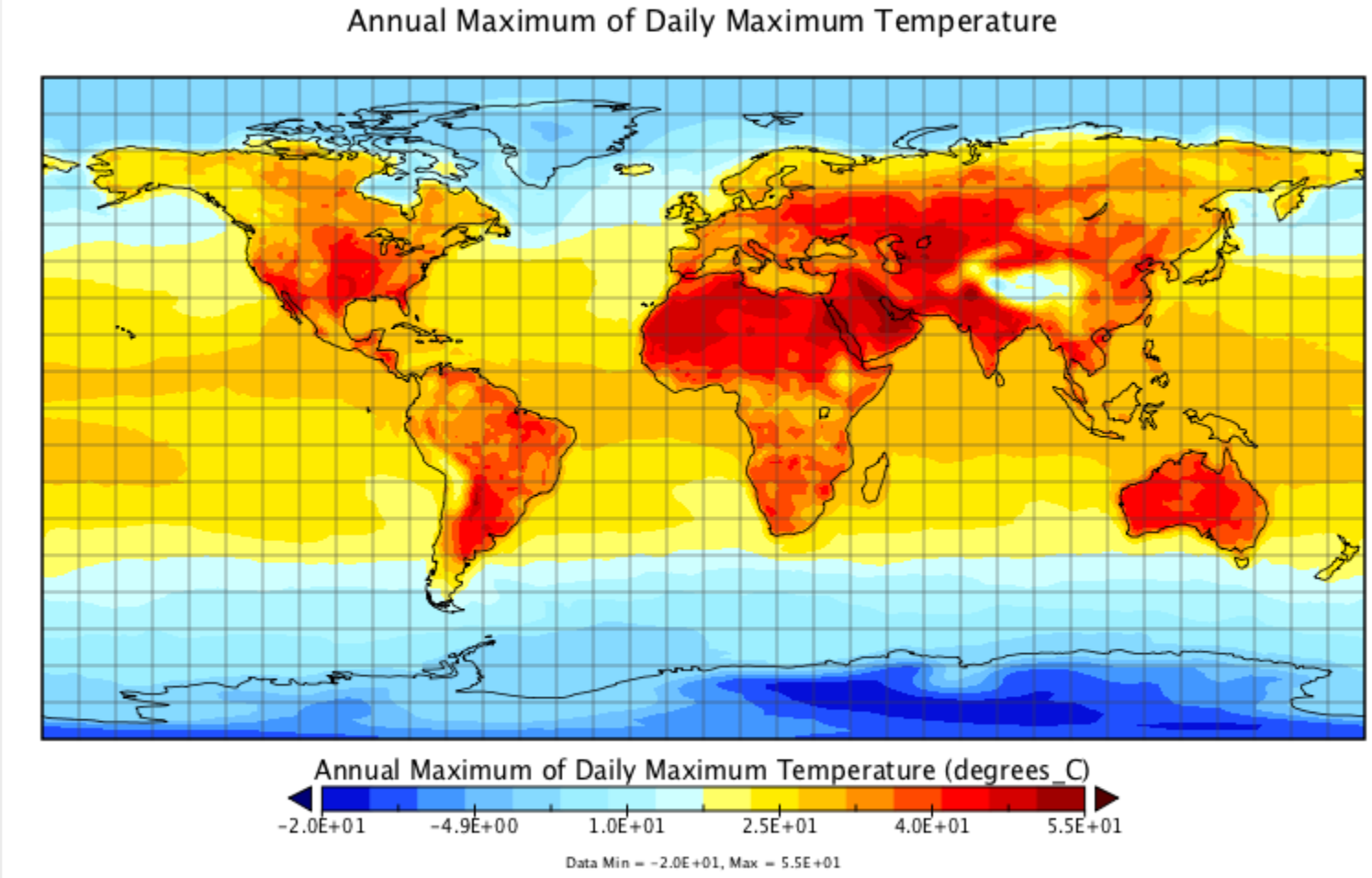



```
r <- raster(ncol=10, nrow=10)
ncell(r)
values(r) <- 1:ncell(r)
hasValues(r)
```

library(raster)

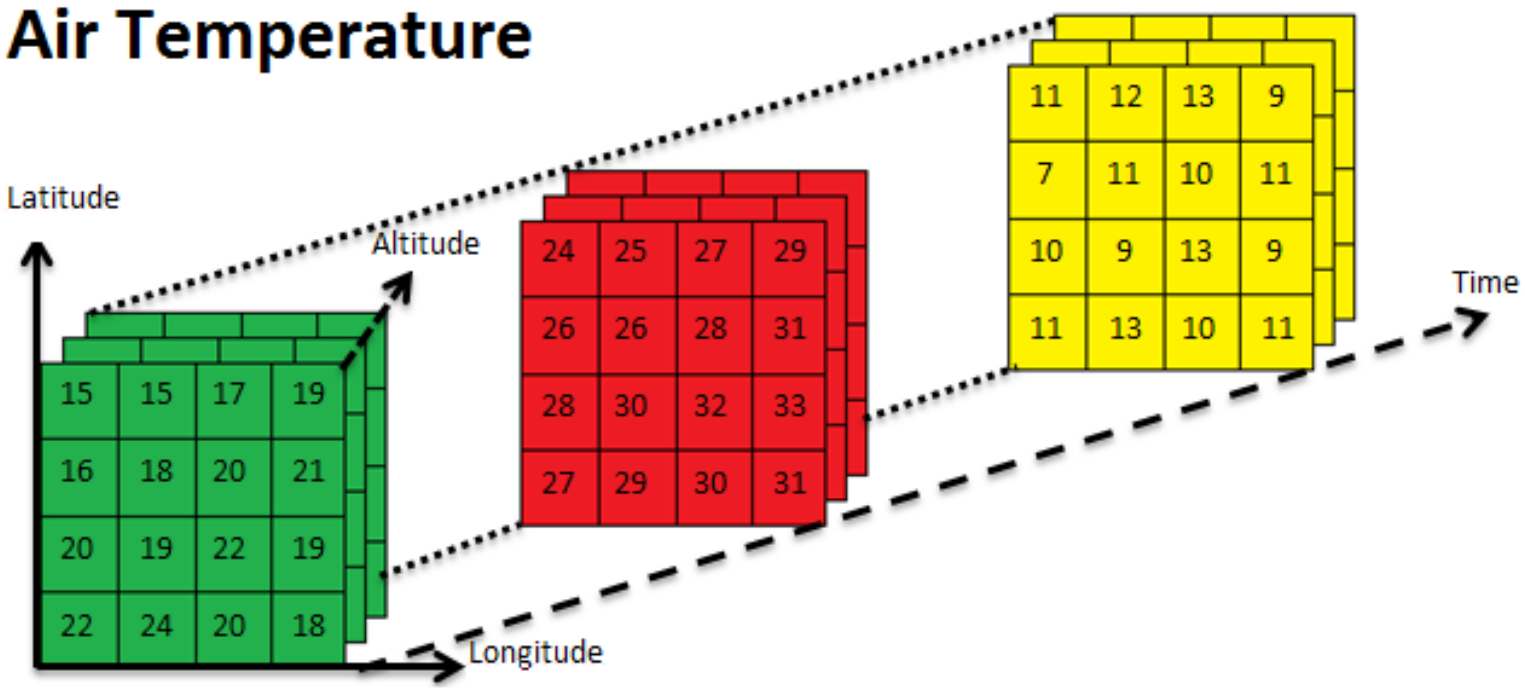
Spatial/Gridded Data

txxETCCDI_yr_MIROC5_historical_r2i1p1_1850-2012.nc



Source: https://live.osgeo.org/en/overview/netcdf_dataset_overview.html

Spatial/Gridded Data



NetCDF (Network Common Data Form) has been defined by the Unidata program and it is widely used to store data and metadata for Climate models and Remote Sensing. A NetCDF file may be considered as a multidimensional array, made of different variables and potentially different dimensions for each variables.

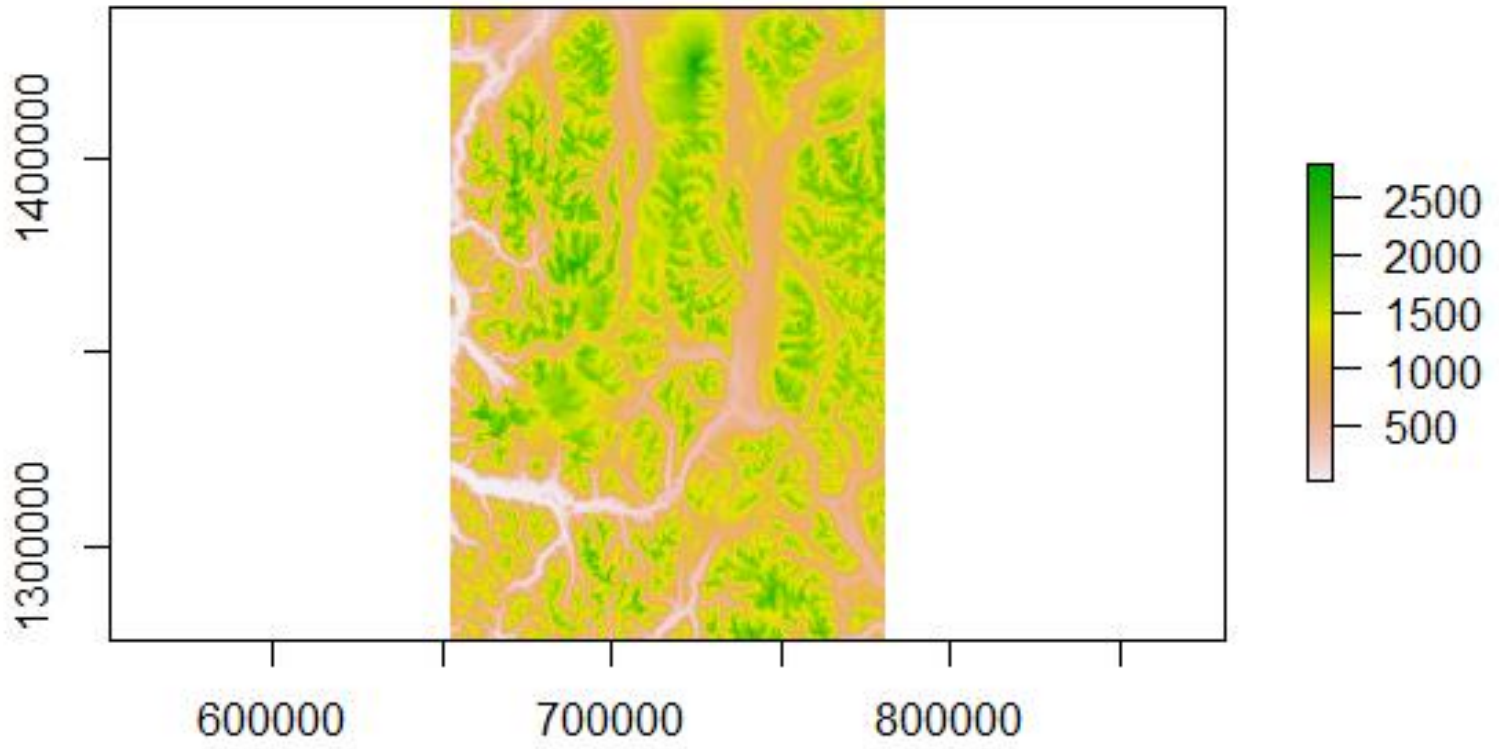
Source: https://geoserver.geo-solutions.it/edu/en/multidim/netcdf/netcdf_basics.html

NetCdf File Structure

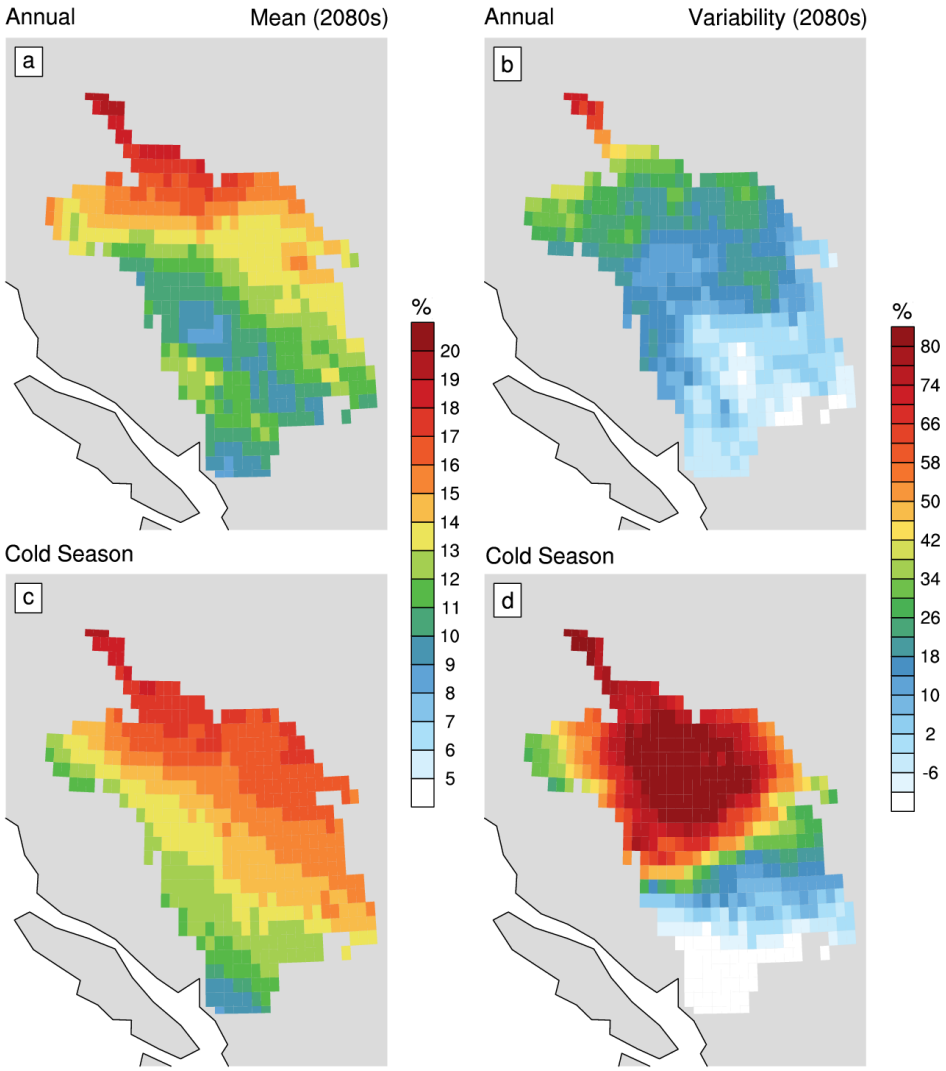
```
netcdf file:XXX.nc {
 dimensions:
      time = UNLIMITED;
      z = 14;
      lat = 96;
      lon = 80;
 variables:
      float O3(time=24, z=14, lat=96, lon=80);
        :long_name = "Ozone concentration";
        :standard_name = "mass_concentration_of_ozone_in_air";
        :unit = "microgram/m3";
        :missing_value = NaN;
global attributes:
       :Conventions = "CF-1.4";
       :title = "File Title";
       :summary = "Summary of the file";
       :keywords = "KEYWORDS, TO, USE";
       :history = "Long History";
```

Raster Plots

```
library(raster)
fl1<-raster("surf_topo_test.nc")
plot(fl1)
```



Spatial/Gridded Data



Important Libraries

- library(sp) #classes and methods for spatial data
- library(maptools) #tools for reading and handling spatial objects
- library(maps) #for creating geographical maps
- library(mapdata) #contains basic data to go along with 'maps'
- library(mapproj) #for creating projected maps
- library(raster) #tools to deal with raster maps
- library(rgdal) #bindings for the geospatial data
- abstraction library