Advanced SWAT Course Notes: Downloading and managing CHIRPS data

Willem Vervoort & Flora Mer

17-11-2017

Introduction

This is part of a series of teaching documents related to the "How do I use satellite and global reanalysis data for hydrological simulations in SWAT?" jointly organised by the University of Sydney, IRI (the University of Columbia) and INIA, Uruguay.

This part explains downloading and managing CHIRPS data as an alternative source of rainfall data.

CHIRPS

CHIRPS is a quasi global 30 year data set: Climate Hazards Group InfraRed Precipitation with Station data derived from blending satellite data and observational data. As a result this could be a good replacement of station input data in catchments were station data is lacking or missing. In addition it can deliver a better spatial coverage of rainfall data. Here we will demonstrate access to the daily v2.0 from the IRI data library for a period of 2000 - 2017.

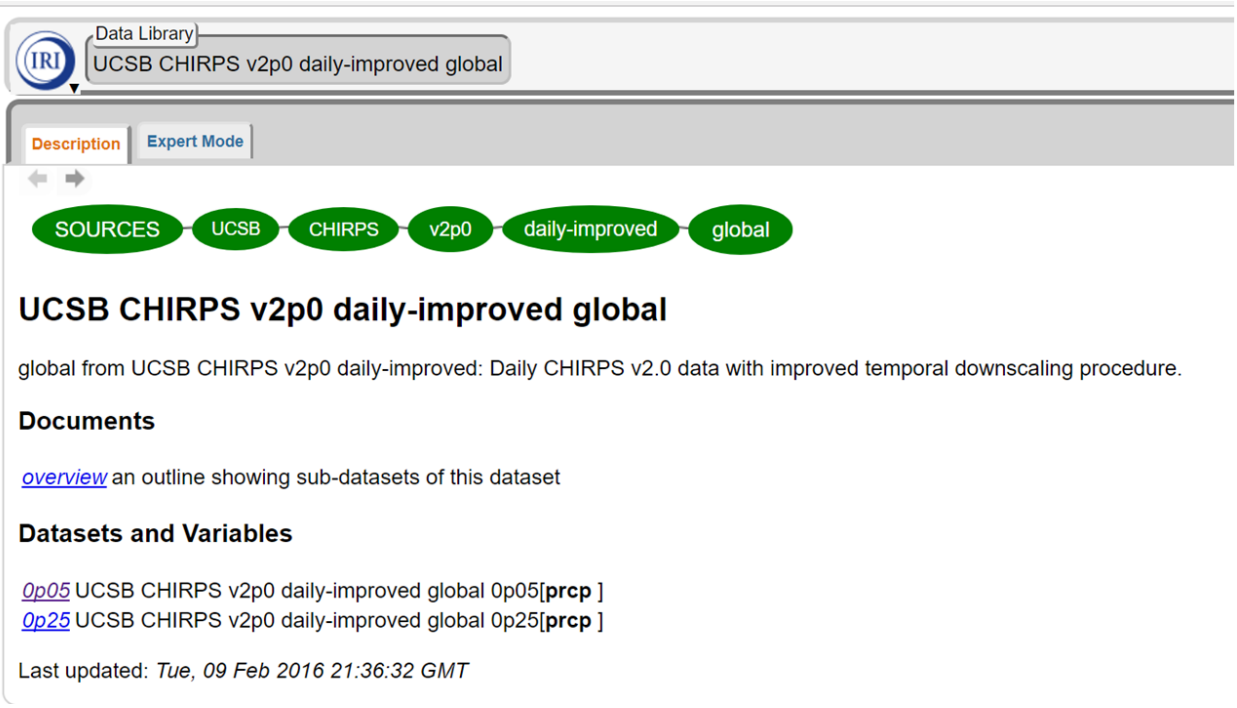
Downloading the CHIRPS data

There are essentially many different ways to access the data, we will demonstrate three of them:

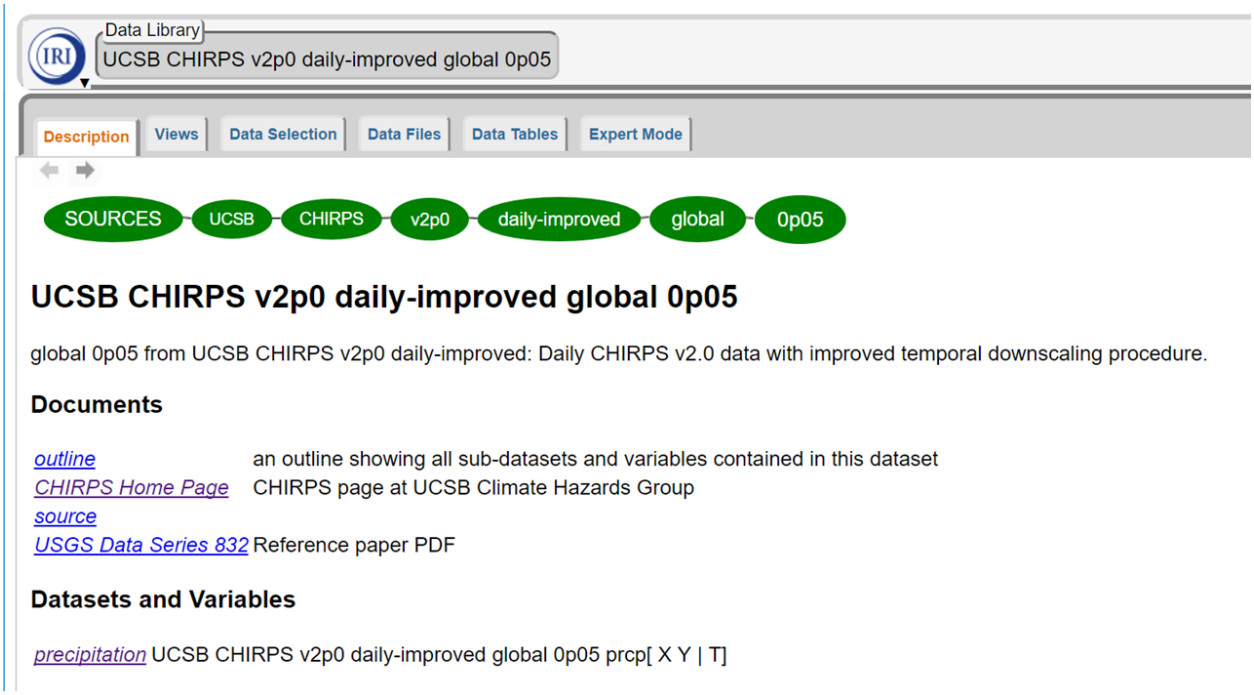
1. Using the website to define an area and time of interest and download a netcdf file
2. Using the "expert mode" on the website to define spatial and temporal coordinates and download the file
3. Using R to construct a URL for the spatial and temporal dimensions and download a netcdf file directly

Using the website

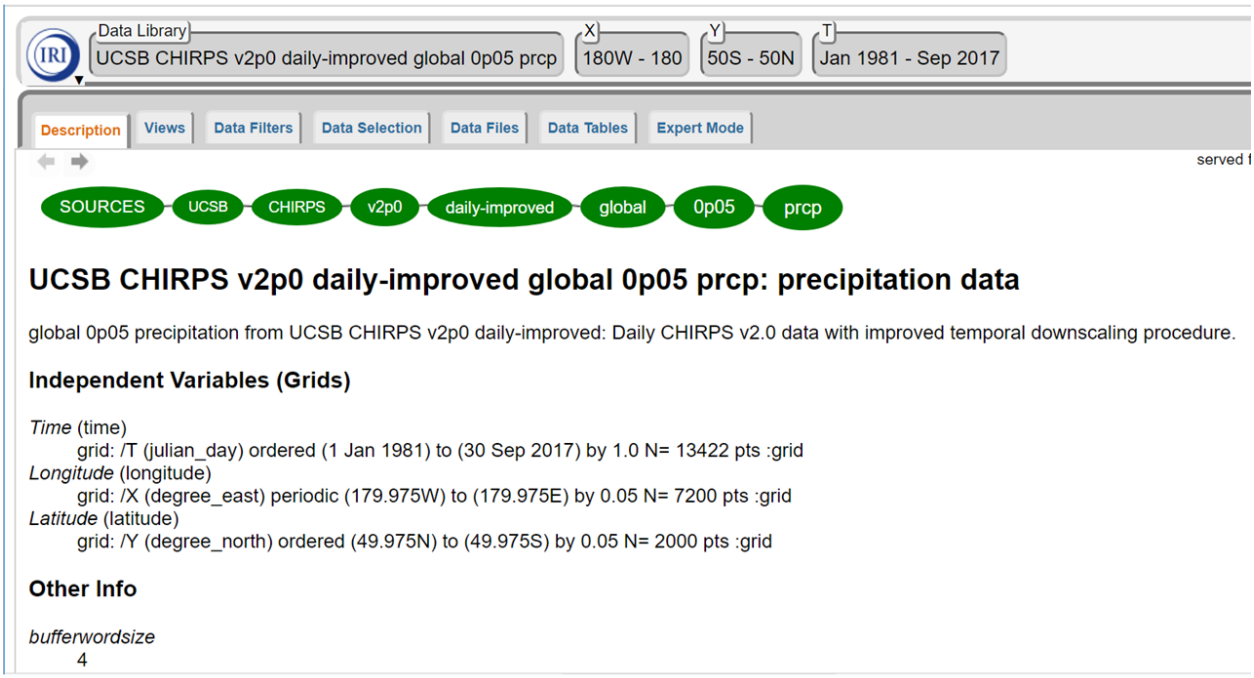
This is probably the most intuitive way of downloading the data, but it also takes the most time in terms of "point and click". The steps are fairly intuitive. Start off at the following link in the IRI data library: https://iridl.ldeo.columbia.edu/SOURCES/.UCSB/.CHIRPS/.v2p0/.daily-improved/.global/, which already bring you straight to this page:



Here you have the choice between the 2.5 by 2.5 degree product, or the much finer scaled 0.5 by 0.5 degree product, which we will select: 0p05 This brings you to the next screen, where we will select "precipitation", which is in fact the only choice of the product.

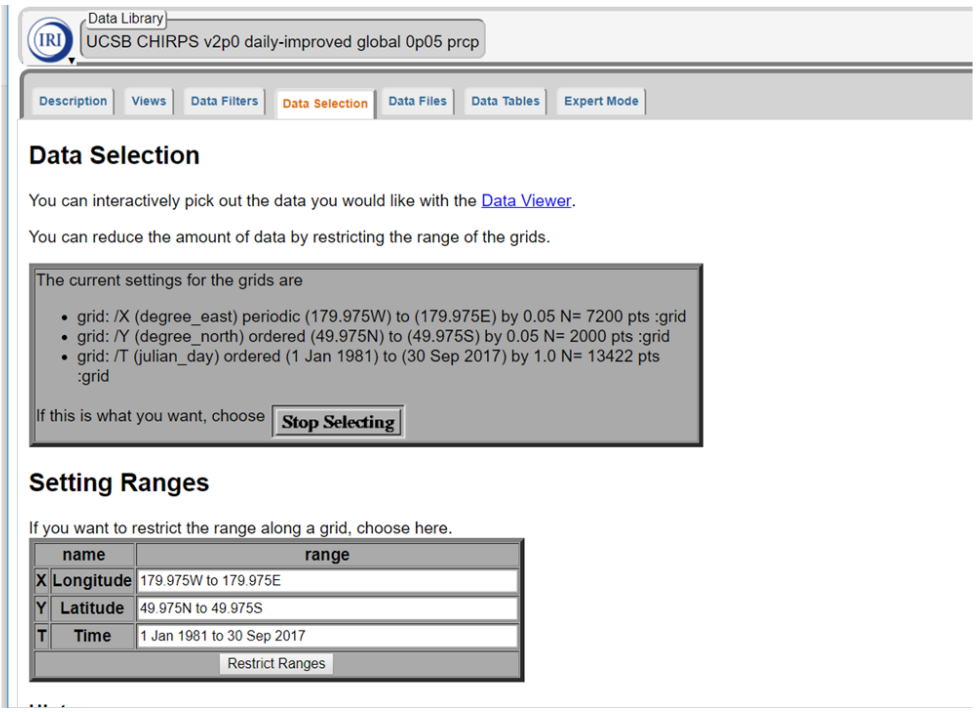


This bring us to the following screen, which outlines the product that we have so far selected. Note already the "bread crumbs" at the top of the page which show you the choices you have made so far.

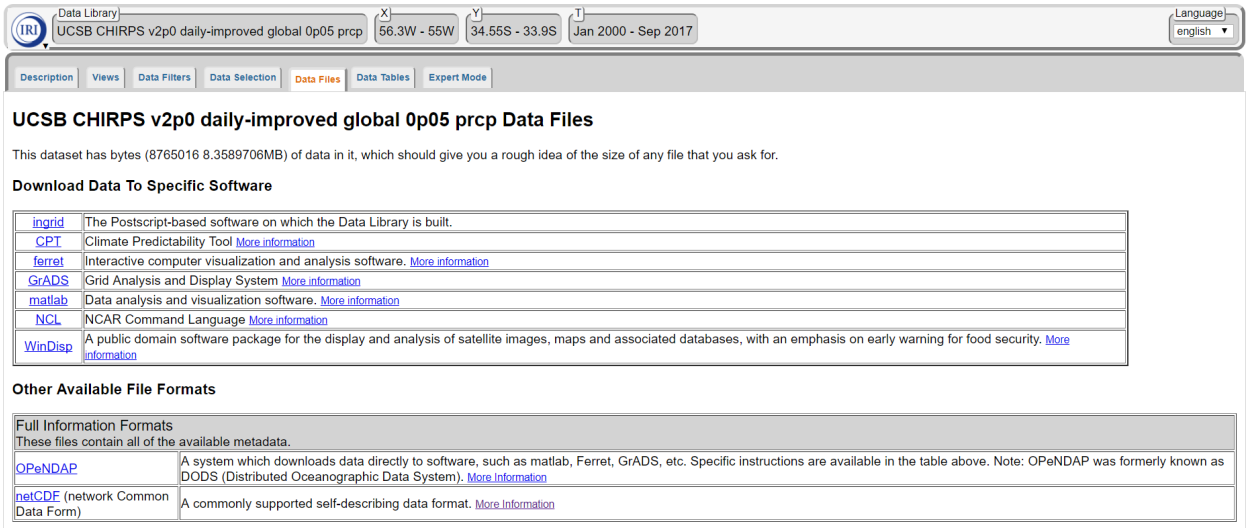


We now want to click on the "Data Selection" link at the top of the page to actually define the space time domain of interest. The folllowing screen has several components. We want to first go to the bottom box, which allows entering the Latitudes and Longitudes as well as the time domain of interest. Note that the data only runs from 1981 and is only land based data. In this case we want to type in the latitudes and longitudes for the Santa Lucia sub-catchment and use the timeperiod 2000 - 2017.

Variable	Min	Max
Longitude	56.3W	55W
Latitude	34.55S	33.9S
Time	2000	2017



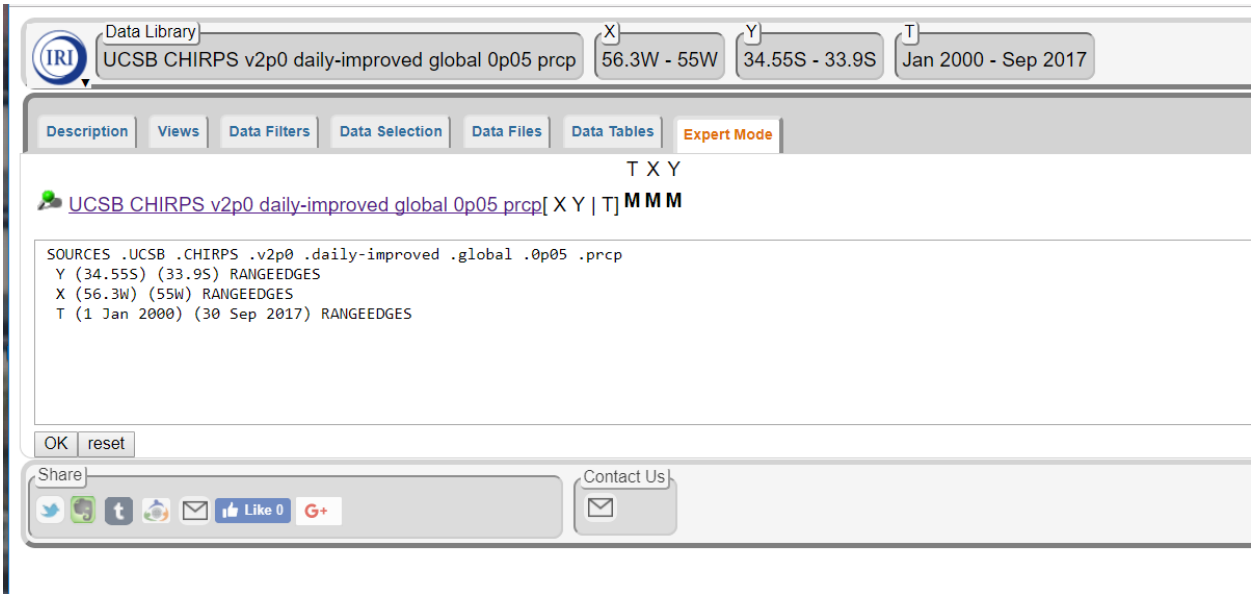
Subsequently click on "Restrict Ranges" and if you are happy with the selection on "Stop selecting". You can then go to the tab with "Data Files" and this shows the different options to download the data. You can download is as a straight text file, but this is a very inefficient data file, and therefore becomes difficult for large areas or slow connections. In this case we are going to use the "netcdf" option.



Clicking on the link will actually not directly result in a download of the data. The system will first have to generate the netcdf file based on your choices and this takes up to 20 minutes, so be a bit patient! Save the netcdf file into your favourite directory.

Using the expert mode on the website

As the last option takes a bit of clicking, and once you are familiar with all the steps, you might want to try the "expert mode". This tab simply allows you to directly enter code to extract data. If you have several data sets to download, you might be quicker doing this. After the last explanation, the step is fairly simple. Click on the "Expert mode" tab, and this screen becomes visible:



As you can see the code in the box reflects the data we have entered before (i.e. the table on page 3). In other words, once you understand the time space domain variables that you need and the product remains the same you can very quickly enter this information in the expert mode box, click OK, and go again to "Data Files" to download the netcdf file.

Using R to directly download the weburl

A final way to access the CHIRPS data is to directly use the URL of the netcdf file to access the file. However, for this you need to know something about the structure of the url and how this can be used to define the spatial and temporal domain considered. You will need to have the package `httr` installed on your computer

```
if (!require(httr) == TRUE) install.packages("httr")
```

```
## Loading required package: httr
```

The basic url for any netcdf to download from the CHIRPS dataset consists of (using the example link for the whole dataset): The base url that defines the "product", the 0.5 by 0.5 degree precipitation product from CHIRPS:
https://iridl.ldeo.columbia.edu/SOURCES/.UCSB/.CHIRPS/.v2p0/.daily-improved/.global/.0p05/.prcp
This followed by the description of the Y range (the North-South domain):
/Y/%2849.975N%29%2849.975S%29RANGEEDGES
This is slightly tricky due to the use of the %28 (start of the latitude and %29 (end of the latitude) separators.
The description of the X range is (the whole East-West domain):
/X/%28179.975W%29%28179.975E%29RANGEEDGES
Which has the same separators. And finally the component for the time range:
/T/%281%20Jan%201981%29%2830%20Sep%202017%29RANGEEDGES
Which is has %28 (start time/date) and %20 (within separators) and %29 (end of the time/date). And the description that we want the netcdf data files /data.nc

So, if we want to extract the data directly, based on the table on page 3, we could use the following script, which makes use of the package `httr`:

```
# ldeo
base1 <- "https://iridl.ldeo.columbia.edu/SOURCES/.UCSB/.CHIRPS/.v2p0"
base_url <- paste(base1, "/.daily-improved/.global/.0p05/.prcp", sep = "")

Lats <- c(-34.55, -33.9)
```

```
Y_section <- paste("/Y/%28", abs(Lats[1]), ifelse(Lats[1] < 0, "S", "N"), "%29%28",
    abs(Lats[2]), ifelse(Lats[2] < 0, "S", "N"), "%29RANGEEDGES", sep = "")

Longs <- c(-56.3, -55)
X_section <- paste("/X/%28", abs(Longs[1]), ifelse(Longs[1] < 0, "W", "E"), "%29%28",
    abs(Longs[2]), ifelse(Longs[2] < 0, "W", "E"), "%29RANGEEDGES", sep = "")

Time <- c("2000-Jan-01", "2017-Sep-30")
T_section <- paste("/T/%28", substr(Time[1], 10, 11), "%20", substr(Time[1], 6, 8),
    "%20", substr(Time[1], 1, 4), "%29%28", substr(Time[2], 10, 11), "%20", substr(Time[2],
        6, 8), "%20", substr(Time[2], 1, 4), "%29RANGEEDGES", sep = "")

final_URL <- paste(base_url, Y_section, X_section, T_section, "/data.nc", sep = "")

# now extract
require(httr)
dl <- GET(final_URL, write_disk("satellitecalibration/Data/CHIRPS/netcdf_download.nc",
    overwrite = TRUE))
```

Extracting the data from the netcdf file

The netcdf file is a specific format that has been designed for space time applications. To extract the data from this file, we need to use the package `ncfd4`.

```
if (!require(ncdf4) == TRUE) install.packages("ncdf4")
```

```
## Loading required package: ncdf4
```

The first step is to actually open the file using `nc_open`, and this allows looking at the different components of the file using `str()`.

```
# first open the netcdf file
chirps_nc <- nc_open("satellitecalibration/data/chirps/netcdf_download.nc")
str(chirps_nc)
```

```
## List of 14
##  $ filename   : chr "satellitecalibration/data/chirps/netcdf_download.nc"
##  $ writable   : logi FALSE
##  $ id         : int 65536
##  $ safemode   : logi FALSE
##  $ format     : chr "NC_FORMAT_CLASSIC"
##  $ is_GMT     : logi FALSE
##  $ groups     :List of 1
##   ..$ :List of 7
##   .. ..$ id   : int 65536
##   .. ..$ name : chr ""
##   .. ..$ ndims: int 3
##   .. ..$ nvars: int 4
##   .. ..$ natts: int 0
##   .. ..$ dimid: int [1:3(1d)] 0 1 2
##   .. ..$ fqgn : chr ""
##   .. ..- attr(*, "class")= chr "ncgroup4"
##  $ fqgn2Rindex:List of 1
##   ..$ : int 1
```

```
##  $ ndims      : num 3
##  $ natts      : num 0
##  $ dim        :List of 3
##   ..$ Y:List of 10
##   .. ..$ name         : chr "Y"
##   .. ..$ len          : int 13
##   .. ..$ unlim        : logi FALSE
##   .. ..$ group_index  : int 1
##   .. ..$ group_id     : int 65536
##   .. ..$ id           : int 0
##   .. ..$ dimvarid     :List of 5
##   .. .. ..$ id         : int 0
##   .. .. ..$ group_index: int 1
##   .. .. ..$ group_id   : int 65536
##   .. .. ..$ list_index : num -1
##   .. .. ..$ isdimvar   : logi TRUE
##   .. .. ..- attr(*, "class")= chr "ncid4"
##   .. ..$ units        : chr "degree_north"
##   .. ..$ vals         : num [1:13(1d)] -34.5 -34.5 -34.4 -34.4 -34.3 ...
##   .. ..$ create_dimvar: logi TRUE
##   .. ..- attr(*, "class")= chr "ncdim4"
##   ..$ X:List of 10
##   .. ..$ name         : chr "X"
##   .. ..$ len          : int 26
##   .. ..$ unlim        : logi FALSE
##   .. ..$ group_index  : int 1
##   .. ..$ group_id     : int 65536
##   .. ..$ id           : int 1
##   .. ..$ dimvarid     :List of 5
##   .. .. ..$ id         : int 1
##   .. .. ..$ group_index: int 1
##   .. .. ..$ group_id   : int 65536
##   .. .. ..$ list_index : num -1
##   .. .. ..$ isdimvar   : logi TRUE
##   .. .. ..- attr(*, "class")= chr "ncid4"
##   .. ..$ units        : chr "degree_east"
##   .. ..$ vals         : num [1:26(1d)] -56.3 -56.2 -56.2 -56.1 -56.1 ...
##   .. ..$ create_dimvar: logi TRUE
##   .. ..- attr(*, "class")= chr "ncdim4"
##   ..$ T:List of 11
##   .. ..$ name         : chr "T"
##   .. ..$ len          : int 6483
##   .. ..$ unlim        : logi FALSE
##   .. ..$ group_index  : int 1
##   .. ..$ group_id     : int 65536
##   .. ..$ id           : int 2
##   .. ..$ dimvarid     :List of 5
##   .. .. ..$ id         : int 2
##   .. .. ..$ group_index: int 1
##   .. .. ..$ group_id   : int 65536
##   .. .. ..$ list_index : num -1
##   .. .. ..$ isdimvar   : logi TRUE
##   .. .. ..- attr(*, "class")= chr "ncid4"
##   .. ..$ units        : chr "julian_day"
```

```
##   .. ..$ calendar     : chr "standard"
##   .. ..$ vals         : num [1:6483(1d)] 2451545 2451546 2451547 2451548 2451549 ...
##   .. ..$ create_dimvar: logi TRUE
##   .. ..- attr(*, "class")= chr "ncdim4"
##  $ unlimdimid : num -1
##  $ nvars      : num 1
##  $ var        :List of 1
##   ..$ prcp:List of 22
##   .. ..$ id                :List of 5
##   .. .. ..$ id         : num 3
##   .. .. ..$ group_index: num -1
##   .. .. ..$ group_id   : int 65536
##   .. .. ..$ list_index : num 1
##   .. .. ..$ isdimvar   : logi FALSE
##   .. .. ..- attr(*, "class")= chr "ncid4"
##   .. ..$ name              : chr "prcp"
##   .. ..$ ndims             : int 3
##   .. ..$ natts             : int 14
##   .. ..$ size              : int [1:3] 26 13 6483
##   .. ..$ dimids            : int [1:3] 1 0 2
##   .. ..$ prec              : chr "float"
##   .. ..$ units             : chr "mm/day"
##   .. ..$ longname          : chr "precipitation"
##   .. ..$ group_index       : int 1
##   .. ..$ chunksizes        : logi NA
##   .. ..$ storage           : num 1
##   .. ..$ shuffle           : logi FALSE
##   .. ..$ compression       : logi NA
##   .. ..$ dims              : list()
##   .. ..$ dim               :List of 3
##   .. .. ..$ :List of 10
##   .. .. .. ..$ name         : chr "X"
##   .. .. .. ..$ len          : int 26
##   .. .. .. ..$ unlim        : logi FALSE
##   .. .. .. ..$ group_index  : int 1
##   .. .. .. ..$ group_id     : int 65536
##   .. .. .. ..$ id           : int 1
##   .. .. .. ..$ dimvarid     :List of 5
##   .. .. .. .. ..$ id         : int 1
##   .. .. .. .. ..$ group_index: int 1
##   .. .. .. .. ..$ group_id   : int 65536
##   .. .. .. .. ..$ list_index : num -1
##   .. .. .. .. ..$ isdimvar   : logi TRUE
##   .. .. .. .. ..- attr(*, "class")= chr "ncid4"
##   .. .. .. ..$ units        : chr "degree_east"
##   .. .. .. ..$ vals         : num [1:26(1d)] -56.3 -56.2 -56.2 -56.1 -56.1 ...
##   .. .. .. ..$ create_dimvar: logi TRUE
##   .. .. .. ..- attr(*, "class")= chr "ncdim4"
##   .. .. ..$ :List of 10
##   .. .. .. ..$ name         : chr "Y"
##   .. .. .. ..$ len          : int 13
##   .. .. .. ..$ unlim        : logi FALSE
##   .. .. .. ..$ group_index  : int 1
##   .. .. .. ..$ group_id     : int 65536
```

```
##   .. .. .. ..$ id           : int 0
##   .. .. .. ..$ dimvarid     :List of 5
##   .. .. .. .. ..$ id         : int 0
##   .. .. .. .. ..$ group_index: int 1
##   .. .. .. .. ..$ group_id   : int 65536
##   .. .. .. .. ..$ list_index : num -1
##   .. .. .. .. ..$ isdimvar   : logi TRUE
##   .. .. .. .. ..- attr(*, "class")= chr "ncid4"
##   .. .. .. ..$ units        : chr "degree_north"
##   .. .. .. ..$ vals         : num [1:13(1d)] -34.5 -34.5 -34.4 -34.4 -34.3 ...
##   .. .. .. ..$ create_dimvar: logi TRUE
##   .. .. .. ..- attr(*, "class")= chr "ncdim4"
##   .. .. ..$ :List of 11
##   .. .. .. ..$ name         : chr "T"
##   .. .. .. ..$ len          : int 6483
##   .. .. .. ..$ unlim        : logi FALSE
##   .. .. .. ..$ group_index  : int 1
##   .. .. .. ..$ group_id     : int 65536
##   .. .. .. ..$ id           : int 2
##   .. .. .. ..$ dimvarid     :List of 5
##   .. .. .. .. ..$ id         : int 2
##   .. .. .. .. ..$ group_index: int 1
##   .. .. .. .. ..$ group_id   : int 65536
##   .. .. .. .. ..$ list_index : num -1
##   .. .. .. .. ..$ isdimvar   : logi TRUE
##   .. .. .. .. ..- attr(*, "class")= chr "ncid4"
##   .. .. .. ..$ units        : chr "julian_day"
##   .. .. .. ..$ calendar     : chr "standard"
##   .. .. .. ..$ vals         : num [1:6483(1d)] 2451545 2451546 2451547 2451548 2451549 ...
##   .. .. .. ..$ create_dimvar: logi TRUE
##   .. .. .. ..- attr(*, "class")= chr "ncdim4"
##   .. ..$ varsize           : int [1:3] 26 13 6483
##   .. ..$ unlim             : logi FALSE
##   .. ..$ make_missing_value: logi TRUE
##   .. ..$ missval           : num -9999
##   .. ..$ hasAddOffset      : logi FALSE
##   .. ..$ hasScaleFact      : logi FALSE
##   .. ..- attr(*, "class")= chr "ncvar4"
##  - attr(*, "class")= chr "ncdf4"
```

This shows that the dimensions and he content of the data file, and indicates how we can extract the different components of the information. The dimensions of the data file are hidden in a set of lists.

```
dims <- chirps_nc$var$prcp$size
lats <- chirps_nc$dim$Y$vals
longs <- chirps_nc$dim$X$vals
```

To show an example, here is the extraction of the data for one of the CHIRPS points

```
# read in data for 1 point from the file
test <- ncvar_get(chirps_nc, count = c(1, 1, dims[3]))
plot(seq.Date(as.Date("2000-01-01"), as.Date("2017-09-30"), by = 1), test, type = "h",
    col = "blue")
```

We now want to repeat the above analysis to construct a data file that includes the output for all the different points, but also includes the latitude and longitude information.

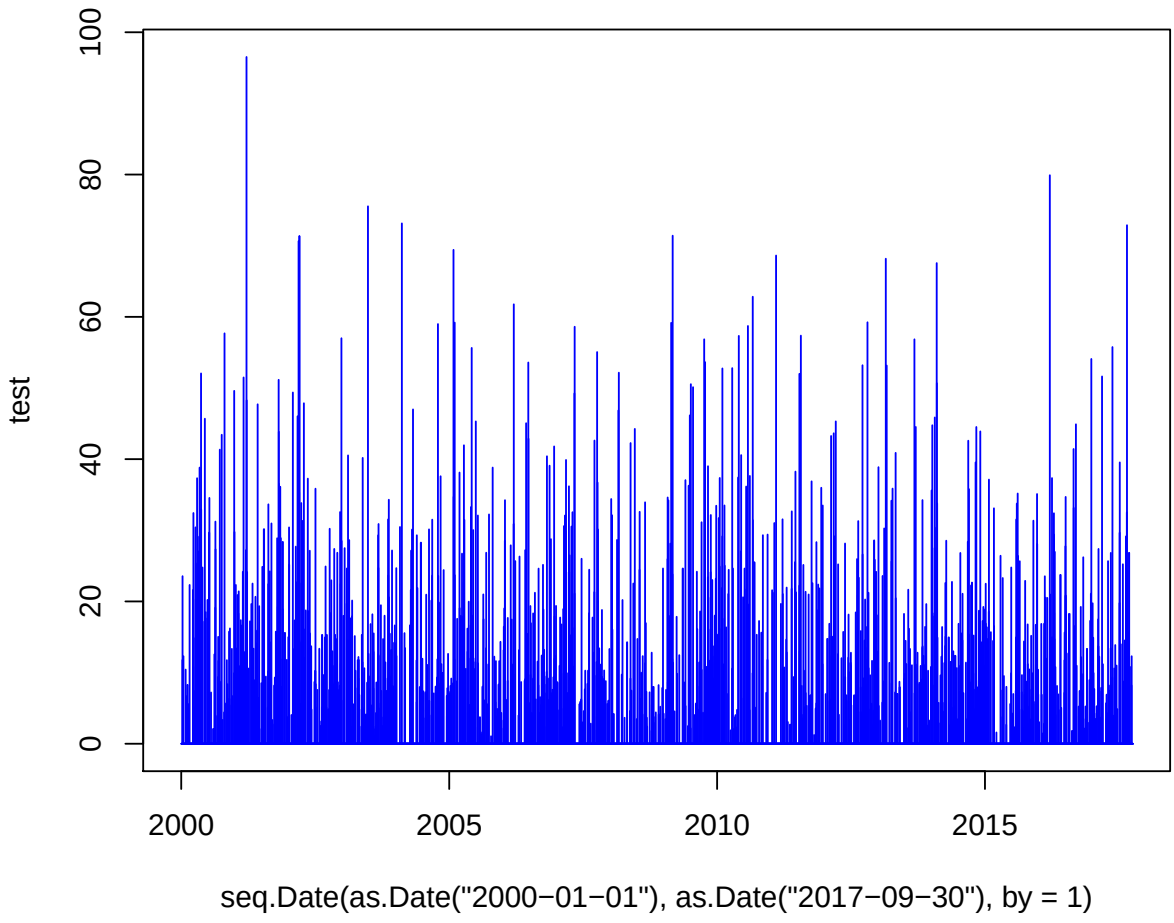


Figure 1: Rainfall extracted from CHIRPS netcdf file for one point

The approach taken here is to use a list, in which each list element is a specific station, and this is once again a list with the Latitude, Longitude and rainfall (precipitation) data.
We then apply a double for loop to insert the data from the netcdf archive into the list using `nvar_get()`

```
output <- list()

for (i in 1:dims[1]) {
    for (j in 1:dims[2]) {
        output[[(i - 1) * dims[2] + j]] <- list(x = longs[i], y = lats[j], prcp = ncvar_get(chirps_nc,
            start = c(i, j, 1), count = c(1, 1, dims[3])))
    }
}
nc_close(chirps_nc)
```

Finally the data can be saved as and RDS archive (which is an efficient binary file) that can be read back into R using `readRDS()`.

```
saveRDS(output, "satellitecalibration/data/CHIRPS/output.RDS")
```