Example netdcf exercise

Dasapta Erwin Irawan, R. Willem Vervoort & Gene Melzack

22 January 2018

Introduction

This document outlines how to construct a netcdf file from spatial and temporal data using the example data Particle Size data set. As an exercise you can redo the analysis with the Moisture or the Chloride data sets in the same folder.

As outlined in the presentation, netcdf files are useful as they are *self describing* and *platform independent*. They are also good to describe spatial and temporal data. But being a specific format, there is some work to make sure it writes everything in the right format.

Example: the particle size data

Load the library and the data

This simply loads the library and the data.

```r
library(ncdf4)

PSAdata <-
  read_csv("OriginalDataFolder/Willem/Soilparticles.csv")
pander(PSAdata[1:5,1:5])
```

Long	Lat	Distance	Depth_top	Depth_bottom
149.7	-29.27	1	0	1
149.7	-29.27	1	1	2
149.7	-29.27	1	2	3
149.7	-29.27	1	3	4
149.7	-29.27	1	4	5

Defining the spatial and temporal dimensions

The next step is to clearly define the spatial, depth and temporal dimensions, this is actually a four dimensional array. However in this case the temporal dimension is a single point. This means we don't actually need this dimension, but I am putting it in as an example.

```r
# define dimensions
londim <- ncdim_def("lon","degrees_east",
                    as.double(unique(PSAdata$Long)))
latdim <- ncdim_def("lat","degrees_north",
                    as.double(unique(PSAdata$Lat)))
timedim <- ncdim_def("time","days since 2001-01-21",
                     as.double(1))
depthdim <- ncdim_def("Depth",
                    "m describing the depth of the top of the 1 m sample",
                      as.double(unique(PSAdata$Depth_top)))
```

Define the nc variables

The next step is to make a description of the variables, as this is the metadata for the individual variables. We can use the information in the readme file (the data dictionary to write these variables). If you look at the data dictionary then you can see that there are 6 columns:

- Clay percentage
- Silt percentage
- Fine sand percentage
- Coarse sand percentage
- Fine gravel percentage
- Coarse gravel percentage

Each of these variables has the same 4 dimensions mentioned above.

Note that there is a "fillvalue", which is also the missing value indicator. In this case we are using 1e32.

```
fillvalue <- 1e32
dlname <- "Percent Clay"
clay_def <- ncvar_def("clay_perc","%",
                      list(londim,latdim,timedim,depthdim),
                      fillvalue,
                      dlname, prec="single")
dlname <- "Percent Silt"
silt_def <- ncvar_def("silt_perc","%",
                      list(londim,latdim,timedim,depthdim),fillvalue,
                      dlname, prec="single")
dlname <- "Percent Fine Sand"
fsand_def <- ncvar_def("fsand_perc","%",
                       list(londim,latdim,timedim,depthdim),fillvalue,
                       dlname,prec="single")
dlname <- "Percent Coarse Sand"
csand_def <- ncvar_def("csand_perc","%",
                       list(londim,latdim,timedim,depthdim),fillvalue,
                       dlname,prec="single")
dlname <- "Percent Fine Gravel"
fgravel_def <- ncvar_def("fgravel_perc","%",
                         list(londim,latdim,timedim,depthdim),fillvalue,
                         dlname,prec="single")
dlname <- "Percent Coarse Gravel"
cgravel_def <- ncvar_def("cgravel_perc","%",
                         list(londim,latdim,timedim,depthdim),fillvalue,
                         dlname,prec="single")
```

Create arrays of the data

The next step is to reformat the data into arrays, which have the prescribed dimensions, so we are going from flat dataframes into a 4 dimensional array (actually on 3 dimensions as there is only a single time variable).

To do this we will first create a matrix for each of the variables with depth_top in the column headings and Lat and Long as the first columns. we will drop the columns "Distance" and "Depth_bottom".

Overall this is not totally correct, as the actual depth of sampling would really be the mid depth between depth top and depth column. But if we describe the samples as being 1 m in length than it will be clear

```
Clay_mat <- PSAdata[,c(1:2,4,6)] %>%
  spread(Depth_top,Clay_perc)

Silt_mat <- PSAdata[,c(1:2,4,7)] %>%
  spread(Depth_top,Silt_perc)

Fsand_mat <- PSAdata[,c(1:2,4,8)] %>%
  spread(Depth_top,Fsand_perc)

Csand_mat <- PSAdata[,c(1:2,4,9)] %>%
  spread(Depth_top,Csand_perc)

Fgravel_mat <- PSAdata[,c(1:2,4,10)] %>%
  spread(Depth_top,Fgravel_perc)

Cgravel_mat <- PSAdata[,c(1:2,4,11)] %>%
  spread(Depth_top,Cgravel_perc)

# rewrite to arrays
nlon <- length(unique(PSAdata$Long))
nlat <- length(unique(PSAdata$Lat))
nd <- ncol(Clay_mat)-2
nt <- 1
Clay_array <- array(as.matrix(Clay_mat[,3:ncol(Clay_mat)]), dim=c(nlon,nlat,nd,nt))
Silt_array <- array(as.matrix(Silt_mat[,3:ncol(Silt_mat)]), dim=c(nlon,nlat,nd,nt))
Fsand_array <- array(as.matrix(Fsand_mat[,3:ncol(Fsand_mat)]), dim=c(nlon,nlat,nd,nt))
Csand_array <- array(as.matrix(Csand_mat[,3:ncol(Csand_mat)]), dim=c(nlon,nlat,nd,nt))
Fgrav_array <- array(as.matrix(Fgravel_mat[,3:ncol(Fgravel_mat)]), dim=c(nlon,nlat,nd,nt))
Cgrav_array <- array(as.matrix(Cgravel_mat[,3:ncol(Cgravel_mat)]), dim=c(nlon,nlat,nd))
```

Create the empty netCDF file

This step simply creates the empty netCDF file using the correct dimensions, which we have define

```
ncfname <- "PCSoildata_ncdf4.nc"
ncout <- nc_create(ncfname,list(clay_def,silt_def,fsand_def,csand_def,
                                fgravel_def, cgravel_def),force_v4=T)
```

And put in the arrays

We can now put in the individual arrays into the netcdf framework

```
ncvar_put(ncout,clay_def,Clay_array)
ncvar_put(ncout,silt_def,Silt_array)
ncvar_put(ncout,fsand_def,Fsand_array)
ncvar_put(ncout,csand_def,Csand_array)
ncvar_put(ncout,fgravel_def,Fgrav_array)
ncvar_put(ncout,cgravel_def,Cgrav_array)
```

Add in additional descriptions and metadata

Because netcdf need to be *selfdescribing* we can also add the metadata to the file. We are again taking most of the information from the Readme.md file in the folder.

```
# put additional attributes into dimension and data variables
ncatt_put(ncout,"lon","axis","X") #,verbose=FALSE) #,definemode=FALSE)
ncatt_put(ncout,"lat","axis","Y")
ncatt_put(ncout,"time","axis","T")
ncatt_put(ncout,"Depth", "axis", "Z")

# add global attributes
ncatt_put(ncout,0,"title","Particle size data from 6 drill holes Palaeochannel data from Northern NSW")
ncatt_put(ncout,0,"summary","Palæochannels, or prior streams, are strings of sandier sediments that occu
ncatt_put(ncout,0,"keywords","EARTH SCIENCE > HYDROLOGY")
ncatt_put(ncout,0,"Resource type","Dataset")
ncatt_put(ncout,0,"Conventions","ACDD-1.3")
ncatt_put(ncout,0,"institution","The University of Sydney, Sydney Institute of Agriculture")
ncatt_put(ncout,0,"datasource","sample and measurement data by authors")
ncatt_put(ncout,0,"references","Vervoort, R. W., & Annen, Y. L. (2006). Palaeochannels in Northern NSW:
ncatt_put(ncout,0,"Conventions","ACDD-1.3")
ncatt_put(ncout,0, "creator", "R.W. Vervoort")
ncatt_put(ncout,0, "creator e-mail", "willem.vervoort@sydney.edu.au")
history <- paste("R.W. Vervoort", date(), sep=", ")
ncatt_put(ncout,0,"history","Downloaded an initial creation January 2018")
```

Closing the file

Finally we need to close the file, which actually writes the data to the disk.

```
# close the file, writing data to disk
nc_close(ncout)
```

Excercise

- Based on the above example, can you construct a netcdf file for the Moisture data set?
- **Extension** Can you build a netcdf file for some of your own data (if you have similar data)?