VII. NetCDF állományok tömeges beolvasása

Olvassunk be három, azonos dimenziójú, de eltérő adatokat tartalmazó NetCDF állományt `for` ciklus alkalmazásával, majd ábrázoljuk az adataikat szintén `for` ciklus segítségével!

A proba2.nc, proba3.nc és a proba4.nc 144×73×30 dimenziójú NetCDF állományok a -80, 50 intervallumon egyenletes eloszlású számokat tartalmaznak, amelyeket R-ben állítottunk elő[1]. Tegyük fel, hogy ezek meteorológiailag releváns adatsorok, például hőmérsékleti adatok, amelyek a teljes Földre vonatkozóan 2,5°-os rácsfelbontással érhetők el, 30 időpontban.

A tömeges beolvasással kapcsolatban az alábbi kérdések merülnek fel:

1. A beolvasandó állományok címei, elérési útjai hogyan gyűjthetők automatikusan egy karakterlánc típusú vektorba, ahonnan `for` ciklussal kiolvashatók lesznek?
2. Hogyan és milyen típusú objektumba menthetők a NetCDF állományok meta adatai, amelyek alapján a dimenziók és a változó(k) kimenthetők?
3. Hogyan és milyen típusú objektumba menthetők a dimenziók és a változó(k)?

Az állományneveket tartalmazó vektort a `Sys.glob` és a `file.path` függvények kombinációjával állítjuk elő. A meta adatokat lista típusú objektumba, a dimenziókat vektor, tömb vagy lista típusú objektumba, a változót pedig lista vagy tömb típusú objektumba[2] mentjük.

Az egyszerűség kedvéért állítsuk át a munkavégzés könyvtárát arra a mappára, amelyben a beolvasandó állományok találhatók.

```
setwd("C:/…")
```

Állítsuk elő az állományneveket tartalmazó, `file` elnevezésű vektort! Mivel mindhárom fájl címe rendelkezik közös résszel, ezért reguláris kifejezést felhasználva (csillag: *) meg tudjuk adni egyszerre a három címet az alábbi módon:

```
file <- Sys.glob(file.path(pattern="proba*nc"))
```

Megjegyzés: Ha szóvégi egyezést szeretnénk, akkor a `$` jelet használhatjuk. Például az nc kiterjesztésű állományok beolvashatók az alábbi módon: `nc$`.

Ellenőrizzük, hogy a `file` objektum valóban karakterértékű vektor és hivatkozzuk az első elemét:

```
class(file)
[1] "character"

file[1]
[1] "proba2.nc"
```

Mentsük a NetCDF állományok adatait a proba.nc elnevezésű, lista típusú objektumba! Ehhez használjuk az ncdf4 csomagot!

[1] Az állományok innen tölthetők le. NetCDF állományok előállításának módja a III. NetCDF állományok készítése és beolvasása című segédletben található.

[2] Lista típusú objektumok kezeléséről a VI. Lista típusú objektumok című segédletben találhatók információk.

```
library(ncdf4)
```

```
proba.nc <- list(0)
for (i in 1:length(file)) {
  proba.nc[[i]] <- nc_open(file[i])
}
```

A `proba.nc` objektum tartalmát a képernyőre kiíratva egymás alatt kerül felsorolásra a három állomány meta adata.

A három állomány dimenziói azonosak, ezért elég csak egyszer kimenteni ezeket, például az első állományból:

```
lon <- c(0)
lat <- c(0)
time <- c(0)
lon <- ncvar_get(proba.nc[[1]], "longitude")
lat <- ncvar_get(proba.nc[[1]], "latitude")
time <- ncvar_get(proba.nc[[1]], "time")
```

Mivel a dimenziók azonosak a három állományban, a változó értékeit rögzített méretű, négydimenziós tömbbe menthetjük, ahol a dimenziók: földrajzi szélesség, földrajzi hosszúság, idő, állomány.

```
valtozo <- array(0, dim=c(dim(lon),dim(lat),dim(time),length(file)))
for (i in 1:length(file)) {
  valtozo[,,,i] <- ncvar_get(proba.nc[[i]], "var")
}
```

Zárjuk le a NetCDF állományokat!

```
for (i in 1:length(file)) {
  nc_close(proba.nc[[i]])
}
```

Készítsünk három térképet (png állományt) a változó első időpontra vonatkozó értékei alapján, a teljes Földre, az alábbi beállításokkal[3]!

- A térképet mentsük 10 hüvelyk széles, 6 hüvelyk magas, 100 dpi felbontású png fájlba!
- A térkép legyen négyzetes hengervetületű!
- Az alapértelmezett betűméret legyen 14 pt!
- Használjunk kék-piros, nullára szimmetrikus színpalettát az RColorBrewer csomagból!
- 40° fokonként kerüljön szaggatott függőleges/vízszintes vonal a földrajzi hosszúságokhoz/ szélességekhez!
- Illesszünk kontinenshatárokat az ábrára!
- Az ábrának legyen fekete kerete! (A keret színe a `col` paraméterrel állítható be.)

Egységes színskála a három adatsor „abszolút” minimális és maximális értékének meghatározásával lehetséges.

```
# Töltsük be az ábrázoláshoz szükséges kiegészítő csomagokat!
library(fields) # az image.plot függvény használatához
library(maps) # kontinenshatárok illesztéséhez
library(RColorBrewer) # színpaletta illesztéséhez
```

[3] Mezők térképes ábrázolásáról információk a IV. NetCDF állományok adatainak kezelése című dokumentumban találhatók.

Egységes színskála szerkesztéséhez határozzuk meg a minimumokat es a maximumokat!

```
min(valtozo[,,,1])
[1] -79.92234

max(valtozo[,,,1])
[1] 49.79795
```

Nullára szimmetrikus, kék-piros színpaletta készítése:

```
brks <- seq(-80,80,10)
paletta <- rev(colorRampPalette(brewer.pal(11,"RdBu"))(length(brks)-1))
```

Távolítsuk el a felesleges színeket, vagyis azokat az intervallumokat, amelyekbe nem esik érték! (Utolsó három, piros színárnyalat.)

```
brks2 <- seq(-80,50,10)
paletta2 <- paletta[1:(length(paletta)-3)]
```

Diagram- és állománycímhez segédobjektum:

```
cim <- "abra"
```

A png állományok címét előállíthatjuk rögzített kifejezések és ciklusváltozó kombinációjával, például a `paste` vagy a `paste0` függvénnyel. Az állományok címe legyen a következő: abra1.png, abra2.png, abra3.png.

Próbáljuk ki az állománynevek képernyőre való kiíratásával, hogyan fog működni a `for` ciklusban való címgenerálás:

```
for (i in 1:length(file)) {
  print(paste0(cim,i,".png"))
}
```

A tengelyfeliratok legyenek az alábbiak:

```
x <- c("-160°","-120°","-80°","-40°","0°","40°","80°","120°","160°")
y <- c("-80° ","-40° ","0°","40° ","80° ")
```

Az ábrák az alábbi utasítással generálhatók a munkavégzés könyvtárába, `for` ciklussal:

```
for (i in 1:length(file)){
  png(paste0(cim,i,".png"), units="in", width=10, height=6, res=100, pointsize=14)
  image.plot(lon, lat, valtozo[,,1,i], col=paletta2,
             breaks=brks2, lab.breaks=brks2, ann=FALSE, xaxt="n", yaxt="n")
  title(main=paste0(cim,i))
  abline(h=seq(-80,80,20), v=seq(-180,160,20), lty=2)
  map("world", interior=FALSE, add=TRUE)
  axis(1, at=seq(-160,160,40), labels=x)
  axis(2, at=seq(-80,80,40), labels=y, las=2)
  box(lty=1)
  graphics.off()
}
```

Megjegyzés: Természetesen más függvény is választható a `for` ciklusban az adatok vizualizációjára, például a `plot`, a `hist` és a `barplot` függvény.

Felhasznált irodalom és csomagok:

R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.
A https://www.rdocumentation.org/ oldalon elérhető leírások a függvények paramétereiről.

David Pierce (2017). ncdf4: Interface to Unidata netCDF (Version 4 or Earlier) Format Data Files. R package version 1.16. https://CRAN.R-project.org/package=ncdf4

Original S code by Richard A. Becker, Allan R. Wilks. R version by Ray Brownrigg. Enhancements by Thomas P Minka and Alex Deckmyn. (2017). maps: Draw Geographical Maps. R package version 3.2.0. https://CRAN.R-project.org/package=maps

Douglas Nychka, Reinhard Furrer, John Paige and Stephan Sain (2017). "fields: Tools for spatial data." doi: 10.5065/D6W957CT (URL: http://doi.org/10.5065/D6W957CT), R package version 9.6, <URL: www.image.ucar.edu/~nychka/Fields>.

Erich Neuwirth (2014). RColorBrewer: ColorBrewer Palettes. R package version 1.1-2. https://CRAN.R-project.org/package=RColorBrewer