Procesamiento de series de tiempo en GRASS GIS

Aplicaciones en Ecología y Ambiente

Dra. Verónica Andreo
CONICET - INMeT

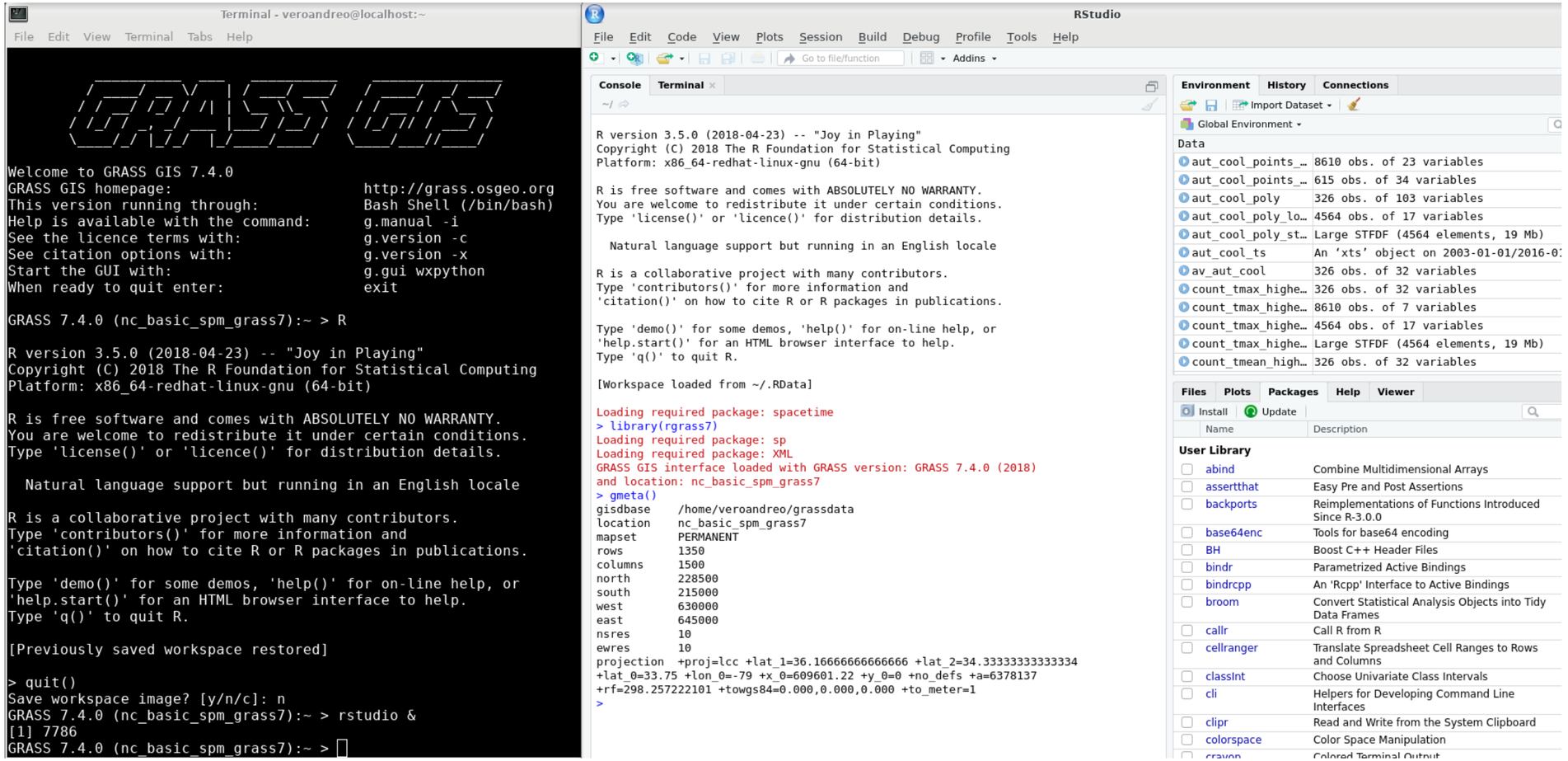
Río Cuarto, 2018

Interface GRASS - R: Bridging GIS and statistics

GRASS GIS and R can be used together in two ways:

- Using R within a GRASS GIS session,
- Using GRASS GIS within an R session,

Details and examples at the GRASS and R wiki


Terminal - veroandreo@localhost:~
File Edit View Terminal Tabs Help
Welcome to GRASS GIS 7.4.0
GRASS GIS homepage: http://grass.osgeo.org
This version running through: Bash Shell (/bin/bash)
Help is available with the command: g.manual -i
See the licence terms with: g.version -c
See citation options with: g.version -x
Start the GUI with: g.gui wxpython
When ready to quit enter: exit
GRASS 7.4.0 (nc_basic_spm_grass7):~ > R
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-redhat-linux-gnu (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[Previously saved workspace restored]
> quit()
Save workspace image? [y/n/c]: n
GRASS 7.4.0 (nc_basic_spm_grass7):~ > rstudio &
[1] 7786
GRASS 7.4.0 (nc_basic_spm_grass7):~ >
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Console Terminal
R version 3.5.0 (2018-04-23) -- "Joy in Playing"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-redhat-linux-gnu (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
[Workspace loaded from ~/.RData]
Loading required package: spacetime
> library(rgrass7)
Loading required package: sp
Loading required package: XML
GRASS GIS interface loaded with GRASS version: GRASS 7.4.0 (2018)
and location: nc_basic_spm_grass7
> gmeta()
gisdbase /home/veroandreo/grassdata
location nc_basic_spm_grass7
mapset PERMANENT
rows 1350
columns 1500
north 228500
south 215000
west 630000
east 645000
nsres 10
ewres 10
projection +proj=lcc +lat_1=36.16666666666666 +lat_2=34.33333333333334
+lat_0=33.75 +lon_0=-79 +x_0=609601.22 +y_0=0 +no_defs +a=6378137
+rf=298.257222101 +towgs84=0.000,0.000,0.000 +to_meter=1
>
Environment History Connections
Import Dataset
Global Environment
Data
aut_cool_points_... 8610 obs. of 23 variables
aut_cool_points_... 615 obs. of 34 variables
aut_cool_poly 326 obs. of 103 variables
aut_cool_poly_lo... 4564 obs. of 17 variables
aut_cool_poly_st... Large STFDF (4564 elements, 19 Mb)
aut_cool_ts An 'xts' object on 2003-01-01/2016-01
av_aut_cool 326 obs. of 32 variables
count_tmax_highe... 326 obs. of 32 variables
count_tmax_highe... 8610 obs. of 7 variables
count_tmax_highe... 4564 obs. of 17 variables
count_tmax_highe... Large STFDF (4564 elements, 19 Mb)
count_tmean_high... 326 obs. of 32 variables
Files Plots Packages Help Viewer
Install Update
Name Description
User Library
abind Combine Multidimensional Arrays
assertthat Easy Pre and Post Assertions
backports Reimplementations of Functions Introduced Since R-3.0.0
base64enc Tools for base64 encoding
BH Boost C++ Header Files
bindr Parametrized Active Bindings
bindrcpp An 'Rcpp' Interface to Active Bindings
broom Convert Statistical Analysis Objects into Tidy Data Frames
callr Call R from R
cellranger Translate Spreadsheet Cell Ranges to Rows and Columns
classInt Choose Univariate Class Intervals
cli Helpers for Developing Command Line Interfaces
clipr Read and Write from the System Clipboard
colorspace Color Space Manipulation

- Using **R within GRASS GIS session**, i.e. starting R (or RStudio) from the GRASS GIS command line.
  - we do not need to initialize GRASS with `initGRASS()`
  - we work with data already in GRASS GIS database using GRASS GIS but from R by means of `execGRASS()`
  - we use `readVECT()`, `readRAST()` to read data from GRASS DB to do analysis or plot
  - we write data back to GRASS with `writeVECT()` and `writeRAST()`

- Using **GRASS GIS within a R session**, i.e. we connect to GRASS GIS database from within R (or RStudio).
  - we need to initialize GRASS GIS with `initGRASS()`
  - we use GRASS GIS funtionalities with `execGRASS()`
  - we use `readVECT()`, `readRAST()` to read data from GRASS DB to do analysis or plot
  - we write data back to GRASS with `writeVECT()` and `writeRAST()`

The link between GRASS GIS and R is provided by the **rgrass7** package

(kudos to Roger Bivand ☺)

Download the file with code to follow this session

We will first run R within a GRASS GIS session

Open GRASS GIS in North Carolina Location and mapset user1

```
g.mapsets mapset=modis_lst,modis_ndvi operation=add
g.region -p vector=nc_state  align=MOD13C2.A2015001.006.single_CMG_0.05
t.rast.series input=LST_Day_monthly_celsius@modis_lst output=lst method
t.rast.series input=ndvi_monthly@modis_ndvi output=ndvi method=average
```

Now, launch R or RStudio from inside GRASS GIS

```
GRASS> rstudio &
GRASS> rstudio /path/to/project/folder/ &
```

Relationship between LST and elevation and NDVI

```
##############################################################
# Commands for GRASS - R interface presentation and demo
# Author: Veronica Andreo
# Date: July - August, 2018
##############################################################



#
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()
```

Relationship between LST and elevation and NDVI

```
#############################################################################
#
# Start R from GRASS
#


# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()

# set region
execGRASS("g.region", raster="lst", flags="p")

# generate random points and sample the lst raster
```

Install and load rgrass7

Relationship between LST and elevation and NDVI

```
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()

# set region
execGRASS("g.region", raster="lst", flags="p")

# generate random points and sample the datasets
execGRASS("v.random", output="samples", npoints=1000, flags = c("overwr

# this will restrict sampling to the boundaries NC
```

Read grass session metadata

Relationship between LST and elevation and NDVI

```
# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()

# set region
execGRASS("g.region", raster="lst", flags="p")

# generate random points and sample the datasets
execGRASS("v.random", output="samples", npoints=1000, flags = c("overwr

# this will restrict sampling to the boundaries NC
# we are overwriting vector samples, so we need to use overwrite flag
execGRASS("v.random", output="samples",
                      npoints=1000,
```

Set the computational region

Relationship between LST and elevation and NDVI

```
library("rgrass7")

# print grass session info
gmeta()

# set region
execGRASS("g.region", raster="lst", flags="p")

# generate random points and sample the datasets
execGRASS("v.random", output="samples", npoints=1000, flags = c("overwr

# this will restrict sampling to the boundaries NC
# we are overwriting vector samples, so we need to use overwrite flag
execGRASS("v.random", output="samples",
                      npoints=1000,
                      restrict="nc_state",
                      flags=c("overwrite"))
```

Generate random points

Relationship between LST and elevation and NDVI

```
# set region
execGRASS("g.region", raster="lst", flags="p")

# generate random points and sample the datasets
execGRASS("v.random", output="samples", npoints=1000, flags = c("overwr

# this will restrict sampling to the boundaries NC
# we are overwriting vector samples, so we need to use overwrite flag
execGRASS("v.random", output="samples",
                      npoints=1000,
                      restrict="nc_state",
                      flags=c("overwrite"))

# create attribute table
execGRASS("v.db.addtable", map="samples",
                           columns=c("elev_state_500m double precision"
                           "ndvi double precision",
                           "lst double precision"))
```

Generate random points restricting to NC area

Relationship between LST and elevation and NDVI

```
# this will restrict sampling to the boundaries NC
# we are overwriting vector samples, so we need to use overwrite flag
execGRASS("v.random", output="samples",
                      npoints=1000,
                      restrict="nc_state",
                      flags=c("overwrite"))


# create attribute table
execGRASS("v.db.addtable", map="samples",
                           columns=c("elev_state_500m double precision"
                           "ndvi double precision",
                           "lst double precision"))

# sample individual rasters
execGRASS("v.what.rast", map="samples", raster="lst", column="lst")
execGRASS("v.what.rast", map="samples", raster="ndvi", column="ndvi")
execGRASS("v.what.rast", map="samples", raster="elev_state_500m", colum

# explore the dataset in R:
```

Add table to vector of random points

Relationship between LST and elevation and NDVI

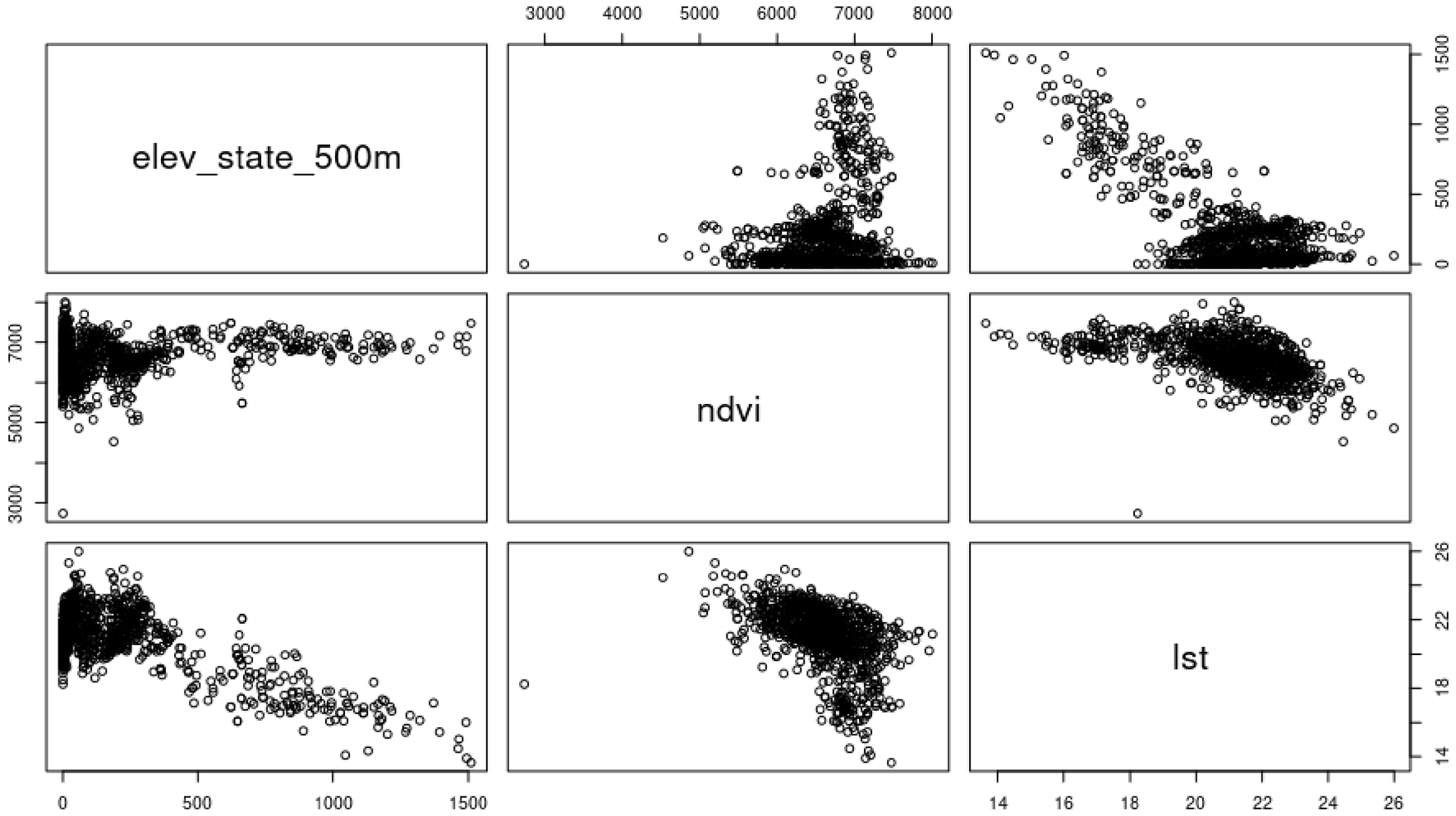
```
# create attribute table
execGRASS("v.db.addtable", map="samples",
                           columns=c("elev_state_500m double precision"
                           "ndvi double precision",
                           "lst double precision"))


# sample individual rasters
execGRASS("v.what.rast", map="samples", raster="lst", column="lst")
execGRASS("v.what.rast", map="samples", raster="ndvi", column="ndvi")
execGRASS("v.what.rast", map="samples", raster="elev_state_500m", colum

# explore the dataset in R:
samples <- readVECT("samples")
str(samples)
summary(samples)
plot(samples@data[2:4])
```

Sample rasters with random points

Relationship between LST and elevation and NDVI

```
                                "lst double precision"))

# sample individual rasters
execGRASS("v.what.rast", map="samples", raster="lst", column="lst")
execGRASS("v.what.rast", map="samples", raster="ndvi", column="ndvi")
execGRASS("v.what.rast", map="samples", raster="elev_state_500m", colum

# explore the dataset in R:
samples <- readVECT("samples")
str(samples)
summary(samples)
plot(samples@data[2:4])

# compute multivariate linear model:
linmodel <- lm(lst ~ elev_state_500m + ndvi, samples@data)
summary(linmodel)

# predict LST using this model:
```

Explore the dataset


elev_state_500m
ndvi
lst
3000
4000
5000
6000
7000
8000
0
500
1000
1500
3000
5000
7000
14
16
18
20
22
24
26

Relationship between LST and elevation and NDVI

```
############################################################
# Commands for GRASS - R interface presentation and demo
# Author: Veronica Andreo
# Date: July - August, 2018
############################################################



#
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()
```

Relationship between LST and elevation and NDVI

```
execGRASS("v.what.rast", map="samples", raster="elev_state_500m", colum

# explore the dataset in R:
samples <- readVECT("samples")
str(samples)
summary(samples)
plot(samples@data[2:4])

# compute multivariate linear model:
linmodel <- lm(lst ~ elev_state_500m + ndvi, samples@data)
summary(linmodel)

# predict LST using this model:
execGRASS("r.mapcalc",
          expression="lst_pred = 29.8 - 0.0042 * elev_state_500m - 0.00
          flags = c("overwrite"))

# set color ramp, read raster and plot
```

Compute linear model

Relationship between LST and elevation and NDVI

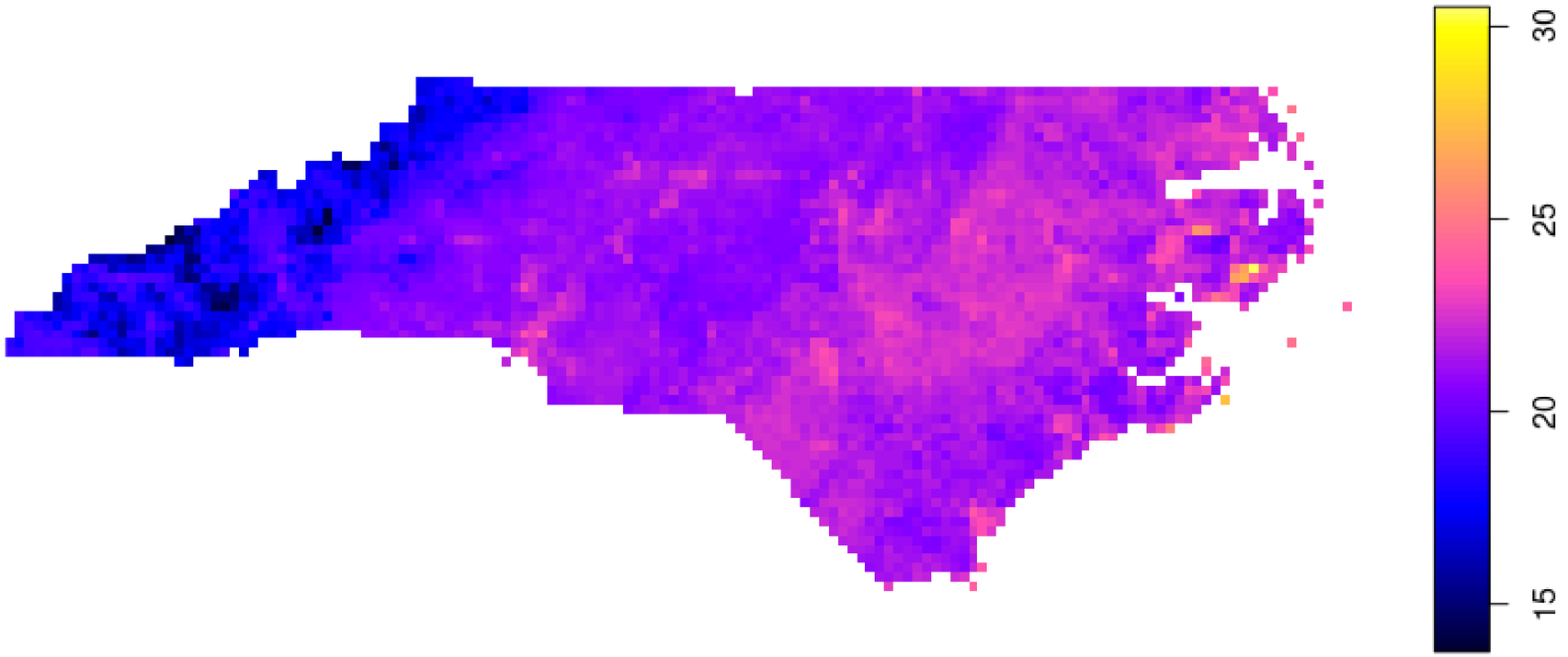
```
summary(samples)
plot(samples@data[2:4])

# compute multivariate linear model:
linmodel <- lm(lst ~ elev_state_500m + ndvi, samples@data)
summary(linmodel)

# predict LST using this model:
execGRASS("r.mapcalc",
          expression="lst_pred = 29.8 - 0.0042 * elev_state_500m - 0.00
          flags = c("overwrite"))

# set color ramp, read raster and plot
execGRASS("r.colors", map="lst_pred", color="celsius")
lst_pred <- readRAST("lst_pred")
plot(lst_pred)

# compare simple linear model to real data:
```

Predict LST using the model

Relationship between LST and elevation and NDVI

```
summary(linmodel)

# predict LST using this model:
execGRASS("r.mapcalc",
          expression="lst_pred = 29.8 - 0.0042 * elev_state_500m - 0.00
          flags = c("overwrite"))

# set color ramp, read raster and plot
execGRASS("r.colors", map="lst_pred", color="celsius")
lst_pred <- readRAST("lst_pred")
plot(lst_pred)

# compare simple linear model to real data:
execGRASS("r.mapcalc", expression="diff = lst - lst_pred", flags = c("o
execGRASS("r.colors", map="diff", color="differences")

# read raster and plot
diff <- readRAST("diff")
```

Set color palette, read raster and plot


30
25
20
15

Relationship between LST and elevation and NDVI

```
##############################################################
# Commands for GRASS - R interface presentation and demo
# Author: Veronica Andreo
# Date: July - August, 2018
##############################################################



#
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()
```

Relationship between LST and elevation and NDVI

```
          expression="lst_pred = 29.8 - 0.0042 * elev_state_500m - 0.00
          flags = c("overwrite"))

# set color ramp, read raster and plot
execGRASS("r.colors", map="lst_pred", color="celsius")
lst_pred <- readRAST("lst_pred")
plot(lst_pred)

# compare simple linear model to real data:
execGRASS("r.mapcalc", expression="diff = lst - lst_pred", flags = c("o
execGRASS("r.colors", map="diff", color="differences")

# read raster and plot
diff <- readRAST("diff")
plot(diff)


#
```

Compare model to real data

Relationship between LST and elevation and NDVI

```
execGRASS("r.colors", map="lst_pred", color="celsius")
lst_pred <- readRAST("lst_pred")
plot(lst_pred)

# compare simple linear model to real data:
execGRASS("r.mapcalc", expression="diff = lst - lst_pred", flags = c("o
execGRASS("r.colors", map="diff", color="differences")

# read raster and plot
diff <- readRAST("diff")
plot(diff)


#
# Start GRASS from R
#
```

Read raster and plot


0
-5
-10

We'll now learn how to start GRASS from within R or Rstudio

*Attention Windows users! Start the the OSGeo4W Shell, change to a directory with writing permission and **start R or RStudio**. To start RStudio run: "C:/Program Files/RStudio/bin/rstudio.exe"*

```
#############################################################################
# Commands for GRASS - R interface presentation and demo
# Author: Veronica Andreo
# Date: July - August, 2018
#############################################################################



#
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()
```

```
plot(dem)


#
# Start GRASS from R
#



# find out the path to the GRASS GIS library
# OSGeo4W users: nothing to do
# Linux, Mac OSX users:
grass74 --config path

# Call GRASS GIS 7 functionality from R
# define the GRASS settings:

## MS-Windows users:
library(rgrass7)
# initialisation and the use of North Carolina sample dataset
```

Find out the path to the GRASS GIS library

```
# Linux, Mac OSX users:
grass74 --config path


# Call GRASS GIS 7 functionality from R
# define the GRASS settings:


## MS-Windows users:
library(rgrass7)
# initialisation and the use of North Carolina sample dataset
initGRASS(gisBase = "C:/OSGeo4W/apps/grass/grass74",
          gisDbase = "C:/Users/username/grassdata/",
          location = "nc_spm_08_grass7",
          mapset = "user1",
          SG = "elevation")

## Linux, Mac OSX users:
library(rgrass7)
# initialisation and the use of North Carolina sample dataset
```

Define the GRASS settings: Windows

```
initGRASS(gisBase = "C:/OSGeo4W/apps/grass/grass74",
          gisDbase = "C:/Users/username/grassdata/",
          location = "nc_spm_08_grass7",
          mapset = "user1",
          SG = "elevation")


## Linux, Mac OSX users:
library(rgrass7)
# initialisation and the use of North Carolina sample dataset
initGRASS(gisBase = "/usr/local/grass74",
          home = tempdir(),
          gisDbase = "/home/veroandreo/grassdata/",
          location = "nc_spm_08_grass7",
          mapset = "user1",
          SG = "elevation")

# Note: the optional SG parameter is a 'SpatialGrid' object to define
# the 'DEFAULT_WIND' of the temporary location.
```

Define the GRASS settings: Linux

```
            home = tempdir(),
            gisDbase = "/home/veroandreo/grassdata/",
            location = "nc_spm_08_grass7",
            mapset = "user1",
            SG = "elevation")

# Note: the optional SG parameter is a 'SpatialGrid' object to define
# the 'DEFAULT_WIND' of the temporary location.

# set computational region to default
execGRASS("g.region", raster="elevation", flags=c("d","p"))
# alternatively:
# system("g.region -dp")

# verify metadata
gmeta()

# list available vector maps:
execGRASS("g.list", parameters = list(type = "vector"))
```

Set computational region

```
                SG = "elevation")

# Note: the optional SG parameter is a 'SpatialGrid' object to define
# the 'DEFAULT_WIND' of the temporary location.

# set computational region to default
execGRASS("g.region", raster="elevation", flags=c("d","p"))
# alternatively:
# system("g.region -dp")


# verify metadata
gmeta()

# list available vector maps:
execGRASS("g.list", parameters = list(type = "vector"))

# list selected vector maps (wildcard):
execGRASS("g.list", parameters = list(type = "vector",
                  pattern = "elev*"))
```

Verify metadata

```
execGRASS("g.region", raster="elevation", flags=c("d","p"))
# alternatively:
# system("g.region -dp")


# verify metadata
gmeta()

# list available vector maps:
execGRASS("g.list", parameters = list(type = "vector"))

# list selected vector maps (wildcard):
execGRASS("g.list", parameters = list(type = "vector",
                  pattern = "elev*"))

# save selected vector maps into R vector:
my_vmaps <- execGRASS("g.list", parameters = list(type = "vector",
                                                  pattern = "elev*"))
attributes(my_vmaps)
```

List vector maps

```
# list available vector maps:
execGRASS("g.list", parameters = list(type = "vector"))

# list selected vector maps (wildcard):
execGRASS("g.list", parameters = list(type = "vector",
                  pattern = "elev*"))

# save selected vector maps into R vector:
my_vmaps <- execGRASS("g.list", parameters = list(type = "vector",
                                                  pattern = "elev*"))
attributes(my_vmaps)
attributes(my_vmaps)$resOut

# list available raster maps:
execGRASS("g.list", parameters = list(type = "raster"))

# list selected raster maps (wildcard):
```

Save list of vector maps

```
                    pattern = "elev*"))

# save selected vector maps into R vector:
my_vmaps <- execGRASS("g.list", parameters = list(type = "vector",
                                                  pattern = "elev*"))
attributes(my_vmaps)
attributes(my_vmaps)$resOut

# list available raster maps:
execGRASS("g.list", parameters = list(type = "raster"))

# list selected raster maps (wildcard):
execGRASS("g.list", parameters = list(type = "raster",
                                      pattern = "lsat7_2002*"))

# get two raster maps into R space
ncdata <- readRAST(c("geology_30m", "elevation"), cat=c(TRUE, FALSE))

# calculate data and object summaries
```

List raster maps

```
attributes(my_vmaps)
attributes(my_vmaps)$resOut

# list available raster maps:
execGRASS("g.list", parameters = list(type = "raster"))

# list selected raster maps (wildcard):
execGRASS("g.list", parameters = list(type = "raster",
                                      pattern = "lsat7_2002*"))

# get two raster maps into R space
ncdata <- readRAST(c("geology_30m", "elevation"), cat=c(TRUE, FALSE))

# calculate data and object summaries
summary(ncdata)
summary(ncdata$elevation)
summary(ncdata$geology_30m)
```

Get raster maps into R

```
execGRASS("g.list", parameters = list(type = "raster"))

# list selected raster maps (wildcard):
execGRASS("g.list", parameters = list(type = "raster",
                                      pattern = "lsat7_2002*"))

# get two raster maps into R space
ncdata <- readRAST(c("geology_30m", "elevation"), cat=c(TRUE, FALSE))

# calculate data and object summaries
summary(ncdata)
summary(ncdata$elevation)
summary(ncdata$geology_30m)

# verify the new R object:
str(ncdata)
str(ncdata@data)
```

Summaries

```
# get two raster maps into R space
ncdata <- readRAST(c("geology_30m", "elevation"), cat=c(TRUE, FALSE))

# calculate data and object summaries
summary(ncdata)
summary(ncdata$elevation)
summary(ncdata$geology_30m)

# verify the new R object:
str(ncdata)
str(ncdata@data)

# plot
image(ncdata, "elevation", col = terrain.colors(20))

# boxplot and histogram
boxplot(ncdata@data$elevation ~ ncdata@data$geology_30m, medlwd = 1)
```

Verify the object

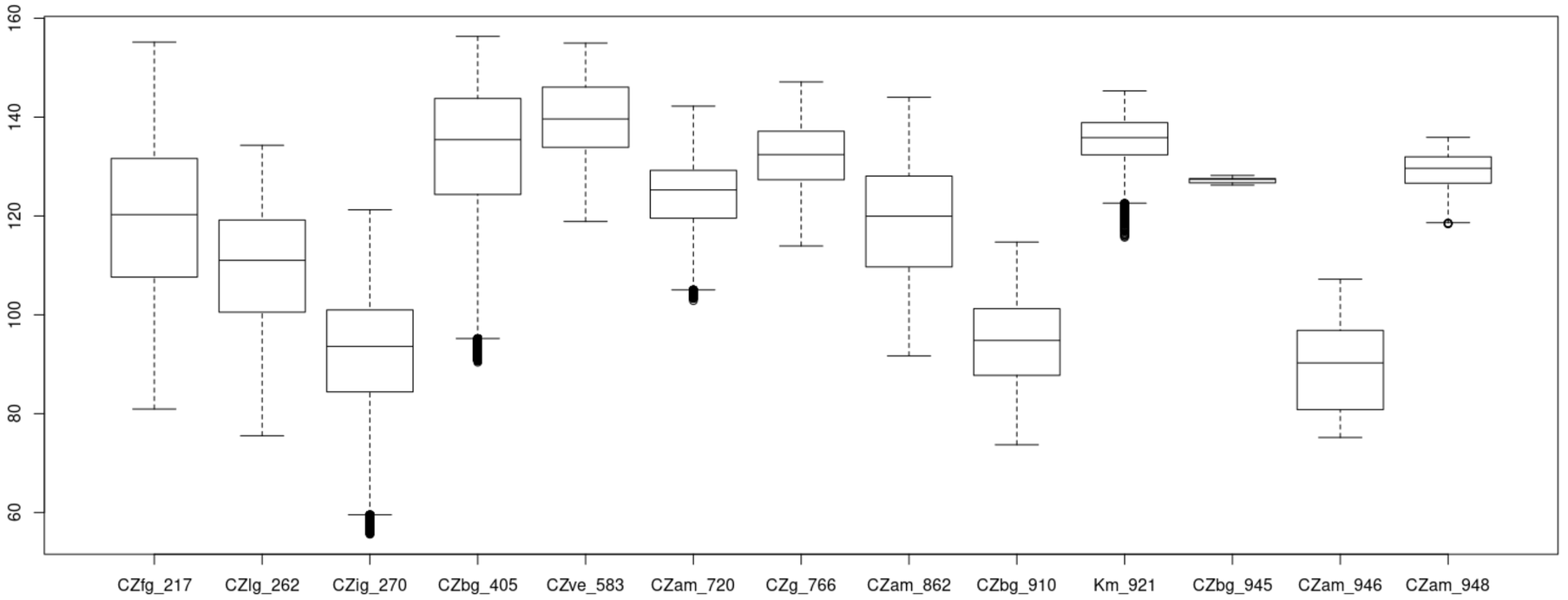
```
# calculate data and object summaries
summary(ncdata)
summary(ncdata$elevation)
summary(ncdata$geology_30m)

# verify the new R object:
str(ncdata)
str(ncdata@data)

# plot
image(ncdata, "elevation", col = terrain.colors(20))

# boxplot and histogram
boxplot(ncdata@data$elevation ~ ncdata@data$geology_30m, medlwd = 1)
hist(ncdata@data$elevation)

# query raster map and transfer result into R
```

Plot

```
##############################################################################
# Commands for GRASS - R interface presentation and demo
# Author: Veronica Andreo
# Date: July - August, 2018
##############################################################################



#
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()
```

```
summary(ncdata$geology_30m)

# verify the new R object:
str(ncdata)
str(ncdata@data)

# plot
image(ncdata, "elevation", col = terrain.colors(20))

# boxplot and histogram
boxplot(ncdata@data$elevation ~ ncdata@data$geology_30m, medlwd = 1)
hist(ncdata@data$elevation)

# query raster map and transfer result into R
goutput <- execGRASS("r.what", map="elev_state_500m",
                               points="geodetic_pts",
                               separator=",", intern=TRUE)
str(goutput)
```

Boxplot and histogram


160
140
120
100
80
60
CZfg_217
CZlg_262
CZig_270
CZbg_405
CZve_583
CZam_720
CZg_766
CZam_862
CZbg_910
Km_921
CZbg_945
CZam_946
CZam_948

```
#############################################################################
# Commands for GRASS - R interface presentation and demo
# Author: Veronica Andreo
# Date: July - August, 2018
#############################################################################



#
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()
```

```
# plot
image(ncdata, "elevation", col = terrain.colors(20))

# boxplot and histogram
boxplot(ncdata@data$elevation ~ ncdata@data$geology_30m, medlwd = 1)
hist(ncdata@data$elevation)

# query raster map and transfer result into R
goutput <- execGRASS("r.what", map="elev_state_500m",
                               points="geodetic_pts",
                               separator=",", intern=TRUE)
str(goutput)

# parse it
con <- textConnection(goutput)
go1 <- read.csv(con, header=FALSE)
str(go1)
```

Query a raster map

```
boxplot(ncdata@data$elevation ~ ncdata@data$geology_30m, medlwd = 1)
hist(ncdata@data$elevation)

# query raster map and transfer result into R
goutput <- execGRASS("r.what", map="elev_state_500m",
                               points="geodetic_pts",
                               separator=",", intern=TRUE)
str(goutput)

# parse it
con <- textConnection(goutput)
go1 <- read.csv(con, header=FALSE)
str(go1)

# square root of elevation
ncdata$sqdem <- sqrt(ncdata$elevation)

# export data from R back into a GRASS raster map:
```

Parse the output

```
#############################################################################
# Commands for GRASS - R interface presentation and demo
# Author: Veronica Andreo
# Date: July - August, 2018
#############################################################################



#
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()
```

```
                                  points="geodetic_pts",
                                  separator=",", intern=TRUE)
str(goutput)

# parse it
con <- textConnection(goutput)
go1 <- read.csv(con, header=FALSE)
str(go1)

# square root of elevation
ncdata$sqdem <- sqrt(ncdata$elevation)

# export data from R back into a GRASS raster map:
writeRAST(ncdata, "sqdemNC", zcol="sqdem", ignore.stderr=TRUE, flags =

# check that it was imported properly:
execGRASS("r.info", parameters=list(map="sqdemNC"))
```

Do something with a raster map

```
str(goutput)

# parse it
con <- textConnection(goutput)
go1 <- read.csv(con, header=FALSE)
str(go1)

# square root of elevation
ncdata$sqdem <- sqrt(ncdata$elevation)

# export data from R back into a GRASS raster map:
writeRAST(ncdata, "sqdemNC", zcol="sqdem", ignore.stderr=TRUE, flags =

# check that it was imported properly:
execGRASS("r.info", parameters=list(map="sqdemNC"))


#
```

Write it into GRASS

```
con <- textConnection(goutput)
go1 <- read.csv(con, header=FALSE)
str(go1)

# square root of elevation
ncdata$sqdem <- sqrt(ncdata$elevation)

# export data from R back into a GRASS raster map:
writeRAST(ncdata, "sqdemNC", zcol="sqdem", ignore.stderr=TRUE, flags =

# check that it was imported properly:
execGRASS("r.info", parameters=list(map="sqdemNC"))


#
# GRASS within R in batch mode
#
```

Check metadata of exported map

GRASS within R in batch mode

Run the script from the terminal with:

```
R CMD BATCH batch.R
```

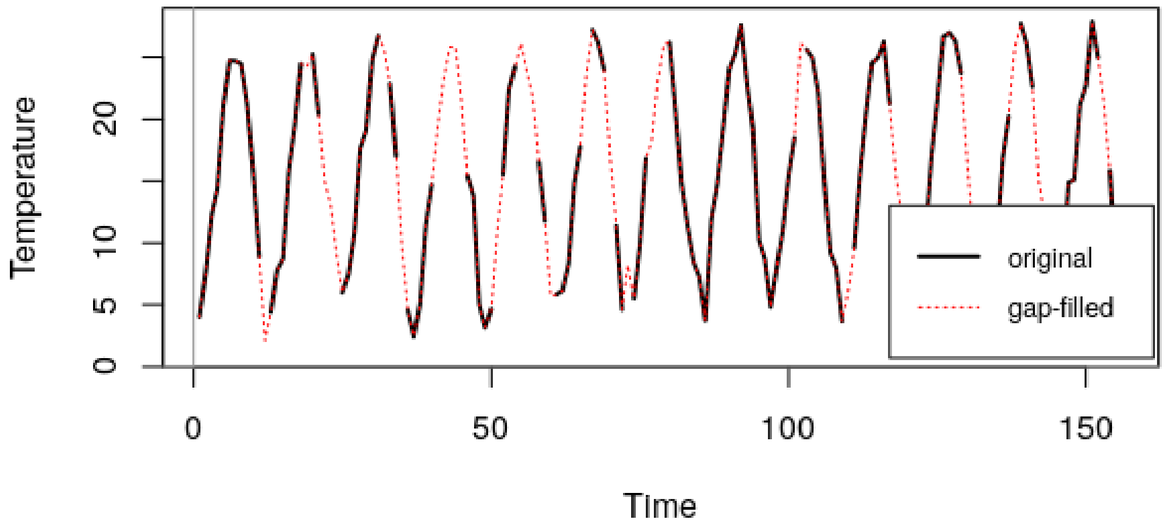
```
#############################################################################
# Commands for GRASS - R interface presentation and demo
# Author: Veronica Andreo
# Date: July - August, 2018
#############################################################################



#
# Start R from GRASS
#



# Install packages
install.packages("rgrass7")
library("rgrass7")

# print grass session info
gmeta()
```

```
# load library
library(rgrass7)
# initialisation and the use of north carolina dataset
initGRASS(gisBase = "/usr/local/grass74",
          home = tempdir(),
          gisDbase = "/home/veroandreo/grassdata/",
          location = "nc_spm_08_grass7",
          mapset = "user1",
          SG="elevation",
          override = TRUE)
# set region to default
system("g.region -dp")
# verify
gmeta()
# read data into R
ncdata <- readRAST(c("geology_30m", "elevation"), cat=c(TRUE, FALSE))
# summary of geology map
summary(ncdata$geology_30m)
```

The script might look like this

Learn more:

Example of GRASS - R for raster time series



There is another R package that provides link to GRASS and other GIS:

link2GI

See the vignette on how to set GRASS database with link2GI for further details

QUESTIONS?

Thanks for your attention!!

Move on to:

GRASS and R: Predicting species distribution

Presentation powered by

GitPitch