Integrating R with GIS for innovative geocomputing – the examples of RQGIS and RSAGA

Jannes Muenchow, Patrick Schratz, Donovan Bangs, Alexander Brenning
R as a GIS

- More than 100 geo-related R packages (https://cran.r-project.org/web/views/Spatial.html)

But:

- R was never designed as a GIS
- Computationally demanding operations
- Missing geoalgorithms
Combining the best of two worlds

http://www.esri.com/

https://7segments.com/
Navigation

1. R as a GIS

2. R-GIS integration

3. R-GIS examples

4. RQGIS vs. RSAGA

(5. RQGIS usage)
R AS A GIS
R as a GIS

- More than 100 geo-related R packages ([https://cran.r-project.org/web/views/Spatial.html](https://cran.r-project.org/web/views/Spatial.html))
- Package `rgdal` for importing and exporting geodata

R as a GIS

- More than 100 geo-related R packages (https://cran.r-project.org/web/views/Spatial.html)
- Package rgdal for importing and exporting geodata
- Packages sp and sf for vector geodata

R as a GIS

- More than 100 geo-related R packages ([https://cran.r-project.org/web/views/Spatial.html](https://cran.r-project.org/web/views/Spatial.html))
- Package `rgdal` for importing and exporting geodata
- Packages `sp` and `sf` for vector geodata
- Package `raster` for raster geodata

R as a GIS

Defining a GIS as a system for the analysis, manipulation and visualization of geographical data (Longley, Goodchild, Maguire, and Rhind 2011), one could argue that R has become a GIS
But what about…

(digitizing)

http://www.unioneag.org

(Geodatabase-functionality and topology rules)

http://help.arcgis.com/
Computationally demanding operations

- Computationally demanding operations
Missing geoalgorithms

- Catchment area
- Catchment slope
- Saga Wetness Index
- Lidar processing
- …
R has been designed from the beginning as an interactive interface to other software packages (Chambers, 2016).
R-GIS INTEGRATION - RQGIS AND RSAGA
GIS interfaces

- **Loose coupling**: Tool and GIS communicate independently.

- **Tight coupling**: GIS and Tool functions are integrated closely, either GIS within Tool or Tool within GIS.

- **System enhancement**: Tool functions are compiled into GIS, or GIS functions are compiled into a tool.

- **Full integration**: Tool is fully integrated within GIS software.

R-GIS integration

RSAGA

RQGIS

GRASS GIS

rgrass 7

RPyGeo
QGIS – Python API

Processing Toolbox

Recently used algorithms
- Pit Remove
- Reproject layer
- SAGA Wetness Index
- v.overlay - Overlays two vector maps.
- Select by attribute
- v.split.length - Split lines to shorter segments by length.

GDAL/OGR [45 geoalgorithms]
GRASS GIS 7 commands [148 geoalgorithms]
Models [0 geoalgorithms]
Orfeo Toolbox (Image analysis) [99 geoalgorithms]
QGIS geoagorithms [98 geoalgorithms]
SAGA (2.1.2) [235 geoalgorithms]
Scripts [0 geoalgorithms]
TauDEM (hydrologic analysis) [30 geoalgorithms]
Tools for LiDAR data [86 geoalgorithms]

Advanced interface

Python Console

1 Python Console
2 Use iface to access QGIS API interface or Type help(iface) for more info
3 >>> import processing

>>> processing.alglist()
How does it work?

- Access QGIS Python API via command line (old interface)
- The new interface (still work in progress) uses the reticulate package to establish a QGIS Python-tunnel, please refer to: https://github.com/jannes-m/RQGIS/tree/rPython
Most notable features of RQGIS

- QGIS geoalgorithms
- Access to hundreds of further geoalgorithms, especially SAGA- and GRASS-geoalgorithms
- R users can stay in their environment without having to touch Python
- Convenience functions `open_help`, `get_args_man` and `run_qgis`
RSAGA interface

• The RSAGA package provides R geocomputing functions that make use of the command line interface of SAGA GIS, `saga_cmd.exe`, to execute SAGA GIS modules.
RSAGA structure

Geoprocessing environment
• List data structure with information on working directory, location of SAGA GIS binaries, etc.

Geoprocessor (using SAGA GIS)
• Workhorse that calls SAGA GIS and provides low-level access to all SAGA GIS modules

User-level interface functions (using SAGA GIS):
• e.g., rsaga.local.morphometry, rsaga.hillshade

Local and focal functions (written in R):
• e.g., multi.focal.function, grid.predict

Utility functions (written in R):
• e.g., pick.from.ascii.grid
R-GIS EXAMPLES
Spatial prediction of alpha diversity

Muenchow et al. (2013): Predictive mapping of species richness and plant species' distributions.
Landslide susceptibility

Brenning et al. (2015): Landslide susceptibility near highways.
Rock glaciers/permafrost

- Computation of direct and diffuse incoming solar radiation using RSAGA

Azócar et al. (2017): Permafrost distribution modeling.
Geomarketing

- Unioning postal code with municipality layers
Further applications

- Soil classes and mapping (e.g., Brungard et al. 2015)
- Stream networks (e.g., Hengl et al. 2010)
- Climatology (rainfall prediction; e.g., Hengl et al. 2010)
- Archeology (e.g., Borck 2016)
- Socio-demography (population index prediction; e.g., Bajat et al. 2012)
- ...
COMPARING RQGIS AND RSAGA
RQGIS vs. RSAGA/rgrass7

- Unified interface to SAGA and GRASS
- Interface to further 3rd-party providers
- Easier-to-use
  - `get_args_man`, `open_help`
  - On-the-fly import/export of spatial objects (`run_qgis`)
  - Automatic data conversions (e.g., asc, tif, etc.)
But:

- QGIS does not support the most recent SAGA versions
- QGIS does not provide access to all SAGA and GRASS functionalities
- RSAGA has special geocomputing functions (written in R)
Wrap-up

• We can use R as a GIS
• Geoprocessing is better done in a GIS
• R-GIS integration combines the best of two worlds
• RQGIS, RSAGA, rgrass7 are all great


RQGIS USAGE
RQGIS on github

Find out how to use RQGIS on github:

http://jannes-m.github.io/RQGIS/index.html
Access to the online help

Let's run:

```r
library("RQGIS")
qgis_env <- set_env("C:/OSGeo4W64/")
open_help(alg = "grass7:r.slope.aspect",
        qgis_env = qgis_env)
```
Automatic argument retrieval

The QGIS Python API does not provide a way to collect function arguments of a geoalgorithm. You can only do that:

```
processing.alghelp("grass7:r.slope.aspect")
```
ALGORITHM: r.slope.aspect - Generates raster layers of slope, aspect, curvatures and partial derivatives from a elevation raster layer.

elevation <ParameterRaster>
format <ParameterSelection>
precision <ParameterSelection>
-a <ParameterBoolean>
-zscale <ParameterNumber>
min_slope <ParameterNumber>
GRASS_REGION_PARAMETER <ParameterExtent>
GRASS_REGION_CELLSIZE_PARAMETER <ParameterNumber>
slope <OutputRaster>
aspect <OutputRaster>
pcurvature <OutputRaster>
tcurvature <OutputRaster>
dx <OutputRaster>
dy <OutputRaster>
dxx <OutputRaster>
dyy <OutputRaster>
dxy <OutputRaster>

format(Format for reporting the slope)
  0 - degrees
  1 - percent

precision(Type of output aspect and slope layer)
  0 - FCELL
  1 - CELL
  2 - DCELL
Convenience function get_args_man

However, get_args_man lets you automatically collect all function arguments and default values of a specific geoalgorithm.

```r
R> params <- get_args_man(alg = "grass7:r.slope.aspect",
                          options = TRUE, qgis_env = qgis_env)
R+ params

$elevation
[1] "None"

/format
[1] "0"
```

...
Let's run_qgis

data("dem")
params$elevation <- dem
params$slope <- file.path(tempdir(), "slope.tif")

slope <- run_qgis("grass7:r.slope.aspect",
params = params,
qgis_env = qgis_env,
load_output = TRUE)

Spatial object residing in R

Loads automatically the QGIS output back into R