RPyGeo: ArcGIS Processing using R

January 31st, 2022

Brandon Allen

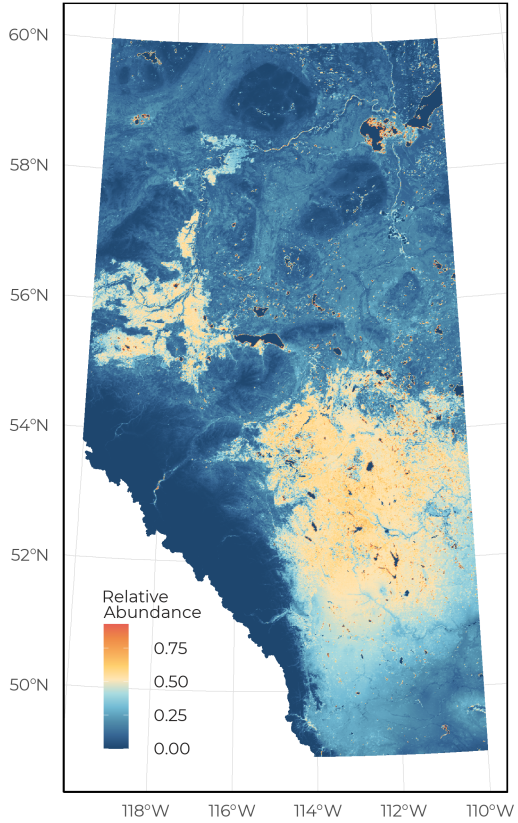
ABMI ALBERTA BIODIVERSITY MONITORING INSTITUTE

Geoprocessing in R

Where to begin…

There are no shortages of R packages available for processing geospatial data.

- Raster
- Rgdal
- Rgeos
- RPyGeo
- SF
- SP
- Terra



RPyGeo

What is this package?

RPyGeo creates a tunnel between R and Python via **Reticulate** to allow for execution of the ArcGIS Python API.

Allows users to maintain all analyses within the same scripting language, while allowing access to geoprocessing benefits of specialist programs such as ArcGIS.

RPyGeo

Marc Becker, Jannes Muenchow, Fabian Polakowski

2018-11-12

1 Introduction

RPyGeo establishes an interface to the geoprocessing tools of ArcGIS from within R. Since ArcGIS only provides a Python API for a low-level access of its C++ based geoalgorithms, **RPyGeo** establishes a tunnel to Python via the **reticulate** package. This extends R's spatial capabilities (Bivand, Pebesma, and Gómez-Rubio 2013; Hijmans 2017; Pebesma 2018) by the geoalgorithms and the geoprocessing power of ArcGIS. Combining this with the statistical and data science power of R opens the way to advanced console-base statistical geoprocessing (Muenchow, Schratz, and Brenning 2017).

To use **RPyGeo** properly, at least a basic understanding of **ArcPy** is necessary. ArcPy is a Python side-package that allows geographic data analysis through ArcGIS from the Python command line. ArcPy is organized into modules which are Python files with functions and classes. The ArcPy main module `arcpy` (or `arcgis` module in ArcGIS API for Python) provides (geographic) classes and several hundred functions. Further modules (e.g., `data access`) and extensions (e.g., `spatial analysis`) further extend the ArcPy main module. We do recommend to get familiar with ArcPy through the official help pages. There also several books available on ArcGIS and ArcPy such as Zandbergen (2013) and Pimpler (2015).

2 Tutorial

In order to use **RPyGeo** you need a working ArcMap or ArcGIS Pro installation on your computer. In addition, this tutorial requires the following packages to be installed and attached.

```
library("RPyGeo")
library("sf")
library("raster")
library("magrittr")
```

spData and **RQGIS** must also be installed since we will use the raster object `dem` and the vector object `nz` from these packages to demonstrate both raster and vector operations. To make these datasets available for the subsequent ArcMap geoprocessing, we have to save them on disk first. Therefore, we export `nz` and `dem` to a temporary directory.

```
data(dem, package = "RQGIS")
data(nz, package = "spData")
writeRaster(dem, file.path(tempdir(), "dem.tif"), format = "GTiff")
st_write(nz, file.path(tempdir(), "nz.shp"))
```

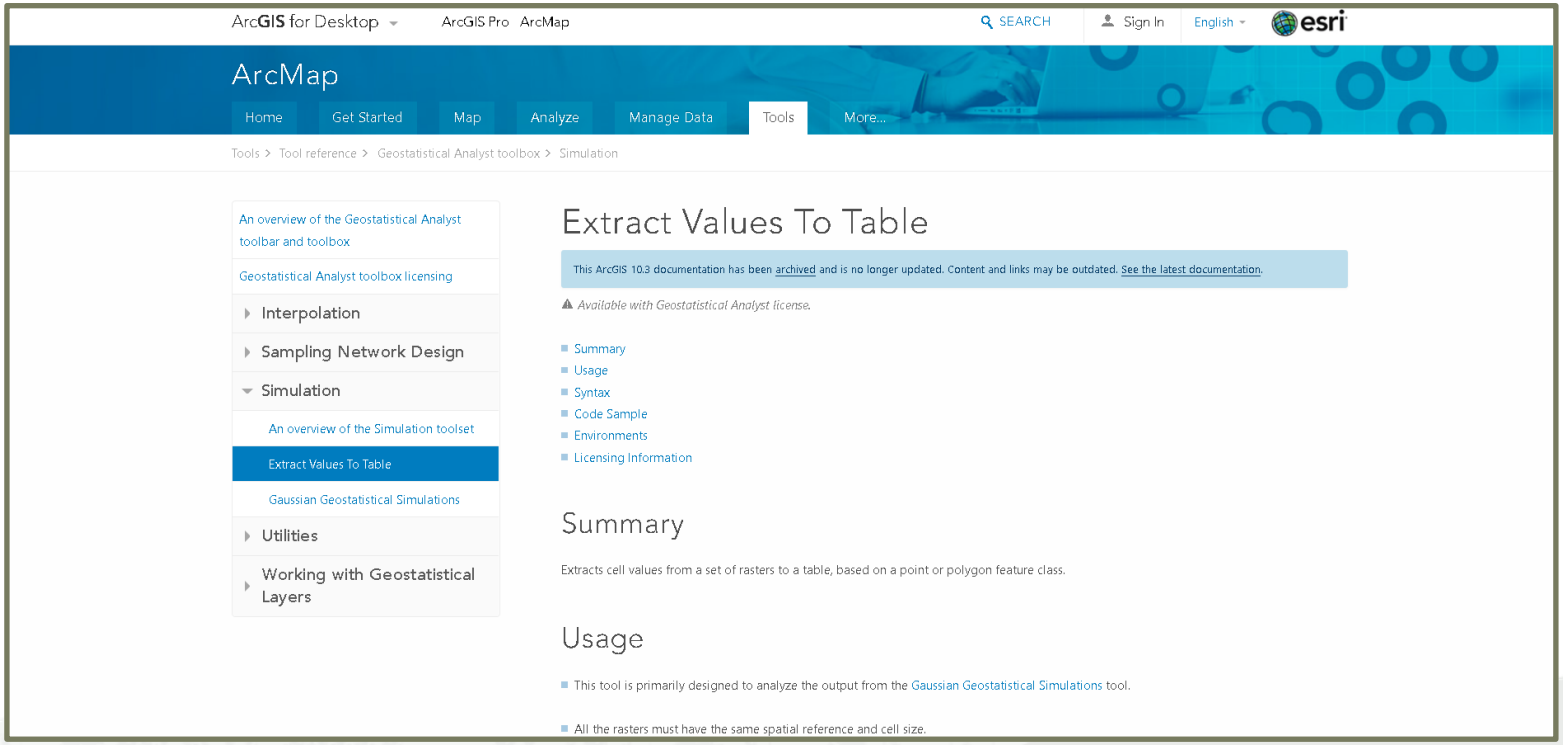
RPyGeo

```
19  # Load libraries
20  library(ggplot2)
21  library(ggpubr)
22  library(RPyGeo)
23  library(sf)
24
25  # First, we need to initiate our connection to ArCGIS
26  # Uses the reticulate package to communicate
27  # Allows you to define which ArcGIS extensions you want to load
28  arcpy <- rpygeo_build_env(path = "C:/Python27/ArcGISx6410.4/",
29                            workspace = "data/base/gis/",
30                            x64 = TRUE,
31                            overwrite = TRUE,
32                            extensions = c("Spatial", "GeoStats"))
33
34  # We can now do everything we want through our newly defined Module
35  arcpy$Select_analysis()
36  arcpy$sa$ClassifyRaster()
37  arcpy$ga$ExtractValuesToTable()
38  arcpy$Select_analysis()
39
40  ######################
41  # Select Analysis bou
42  ######################
43
```

```
in_features =
out_feature_class =
where_clause =
```

```
in_features
in_features
Press F1 for additional help
```

RPyGeo

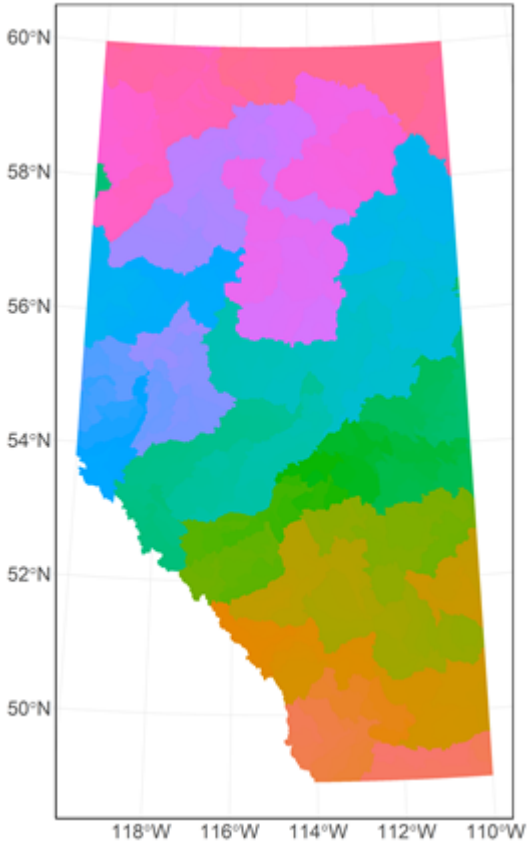

ArcGIS for Desktop
ArcGIS Pro ArcMap
SEARCH
Sign In
English
esri
ArcMap
Home
Get Started
Map
Analyze
Manage Data
Tools
More...
Tools > Tool reference > Geostatistical Analyst toolbox > Simulation
An overview of the Geostatistical Analyst toolbar and toolbox
Geostatistical Analyst toolbox licensing
Interpolation
Sampling Network Design
Simulation
An overview of the Simulation toolset
Extract Values To Table
Gaussian Geostatistical Simulations
Utilities
Working with Geostatistical Layers
Extract Values To Table
This ArcGIS 10.3 documentation has been archived and is no longer updated. Content and links may be outdated. See the latest documentation.
Available with Geostatistical Analyst license.
Summary
Usage
Syntax
Code Sample
Environments
Licensing Information
Summary
Extracts cell values from a set of rasters to a table, based on a point or polygon feature class.
Usage
This tool is primarily designed to analyze the output from the Gaussian Geostatistical Simulations tool.
All the rasters must have the same spatial reference and cell size.

SF vs RPyGeo

Select analysis

- Alberta watershed boundaries (422 polygons)

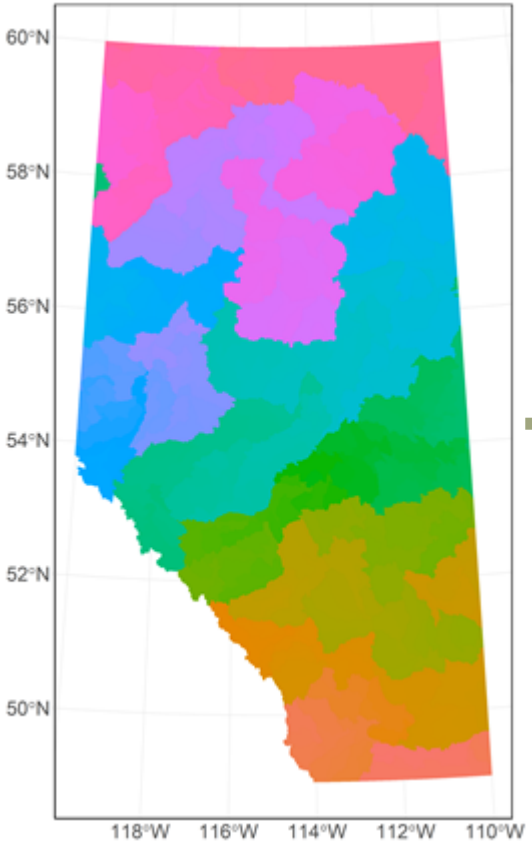
```
# SF
start.time <- Sys.time()
boundary.in <- read_sf("data/base/gis/boundaries/HUC_8_EPSG3400.shp")
boundary.in <- boundary.in[boundary.in$HUC_8 == "18030103", ]
write_sf(obj = boundary.in, dsn = "data/base/gis/examples/boundary_sf.shp", quiet = TRUE)
rm(boundary.in)
Sys.time() - start.time

# RPyGeo
start.time <- Sys.time()
arcpy$Select_analysis(in_features = "data/base/gis/boundaries/HUC_8_EPSG3400.shp",
                      out_feature_class = "examples/boundary_rpygeo.shp",
                      where_clause = paste0('"HUC_8" = ', "'", "18030103", "'"))
Sys.time() - start.time
```

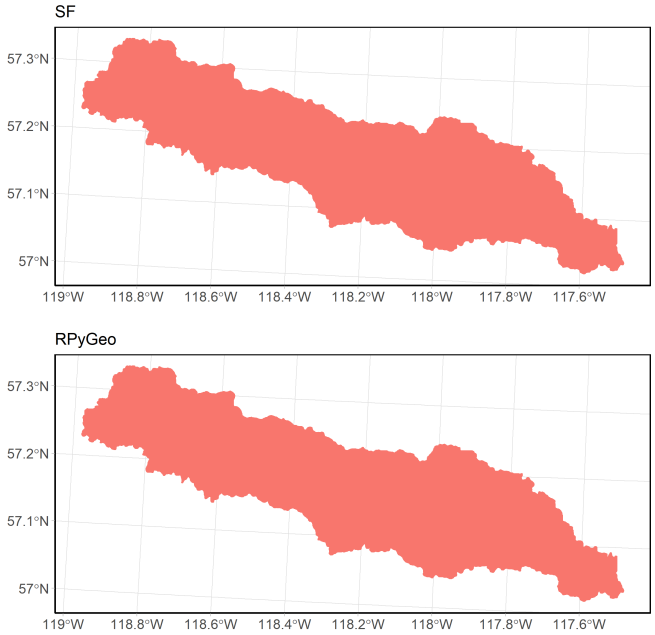


SF vs RPyGeo

Select analysis

- SF completed in 0.25s
- RPyGeo completed in 0.61s



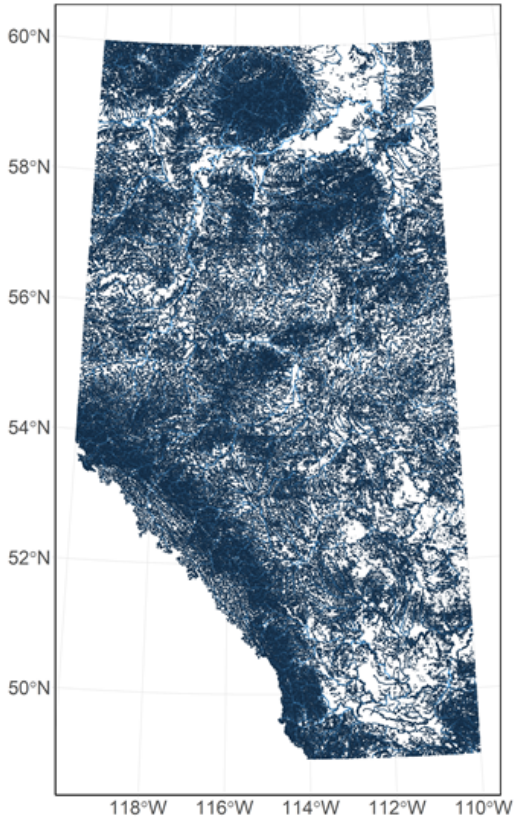


SF vs RPyGeo

Clip analysis

- Alberta stream network (629,003 lines)

```r
# SF
start.time <- Sys.time()
watershed.in <- read_sf("data/base/gis/strahler_stream_order/cleaned-network/stream_network_merged.shp")
boundary.in <- read_sf("data/base/gis/examples/boundary_sf.shp")
watershed.clip <- st_intersection(watershed.in, boundary.in)
write_sf(obj = watershed.clip, dsn = "data/base/gis/examples/watershed_sf.shp", quiet = TRUE)
rm(watershed.in)
rm(boundary.in)
Sys.time() - start.time

# RPyGeo
start.time <- Sys.time()
arcpy$Clip_analysis(in_features = "data/base/gis/strahler_stream_order/cleaned-network/stream_network_merged.shp",
                    clip_features = "data/base/gis/examples/boundary_rpygeo.shp",
                    out_feature_class = "data/base/gis/examples/watershed_rpygeo.shp")
Sys.time() - start.time

```

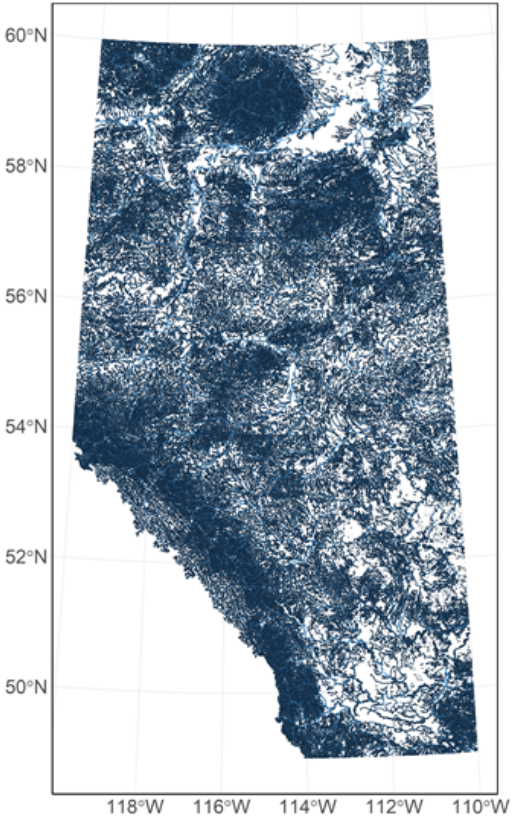


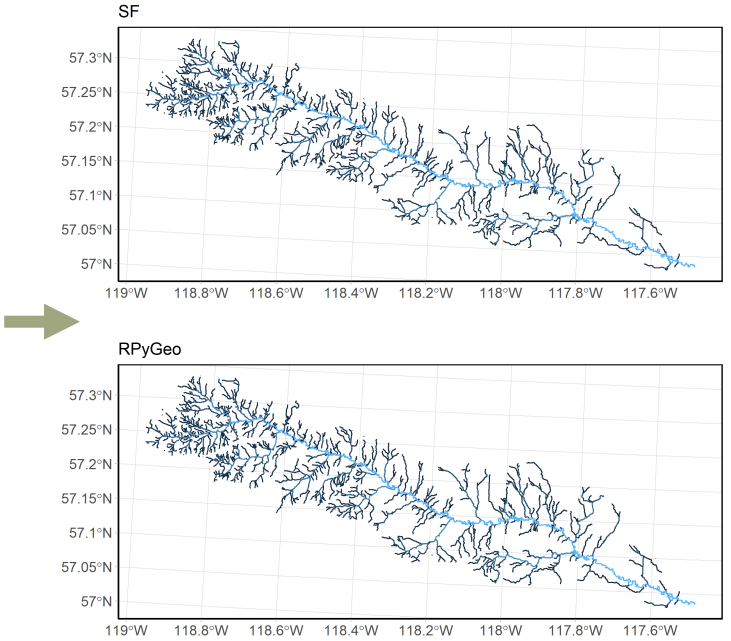
SF vs RPyGeo

Clip analysis

- SF completed in 39.3s
- RPyGeo completed in 2.0s

- SF = 4.6 hours for 422 watersheds
- RPyGeo = 14.1 minutes for 422 watersheds



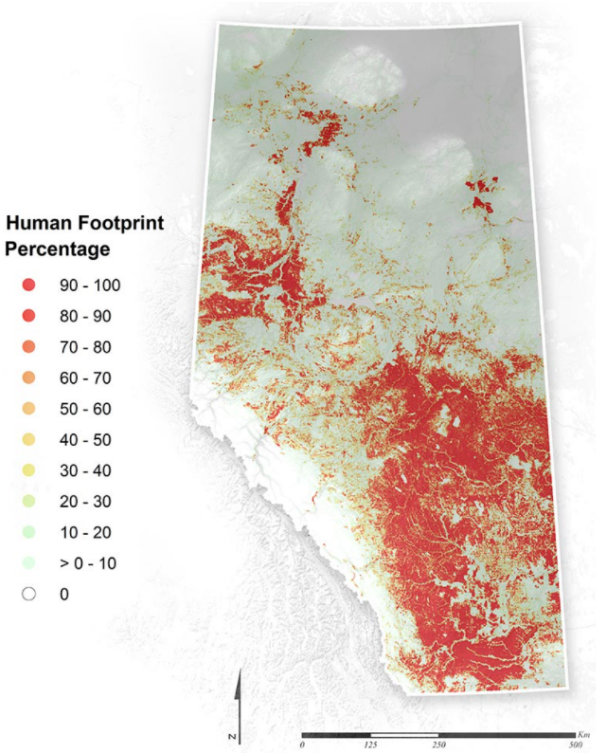


SF vs RPyGeo

Clip analysis

- Alberta Human Footprint (5,384,315 polygons)

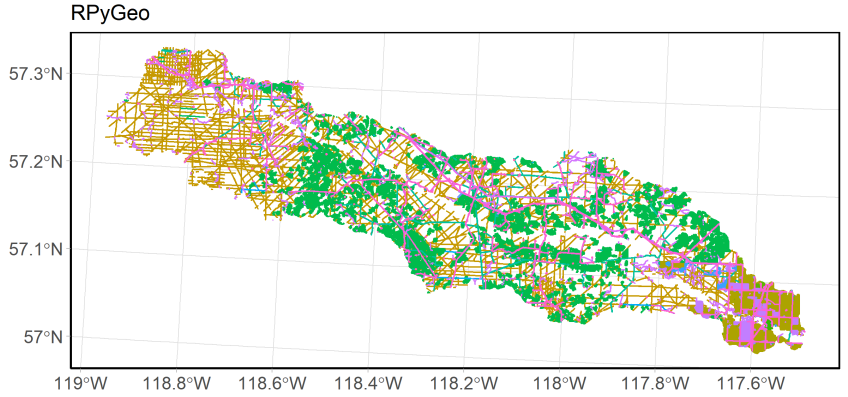
```
# RPyGeo
start.time <- Sys.time()
arcpy$Clip_analysis(in_features = "//gisserver.abmi.ca/GIS/Anthropogenic/HumanFootprint/HFI/HFI_2018/HFI_2018_v1.gdb/HFI_2
                    clip_features = "data/base/gis/examples/boundary_rpygeo.shp",
                    out_feature_class = "data/base/gis/examples/landcover_rpygeo.shp")
Sys.time() - start.time
```



SF vs RPyGeo

Clip analysis

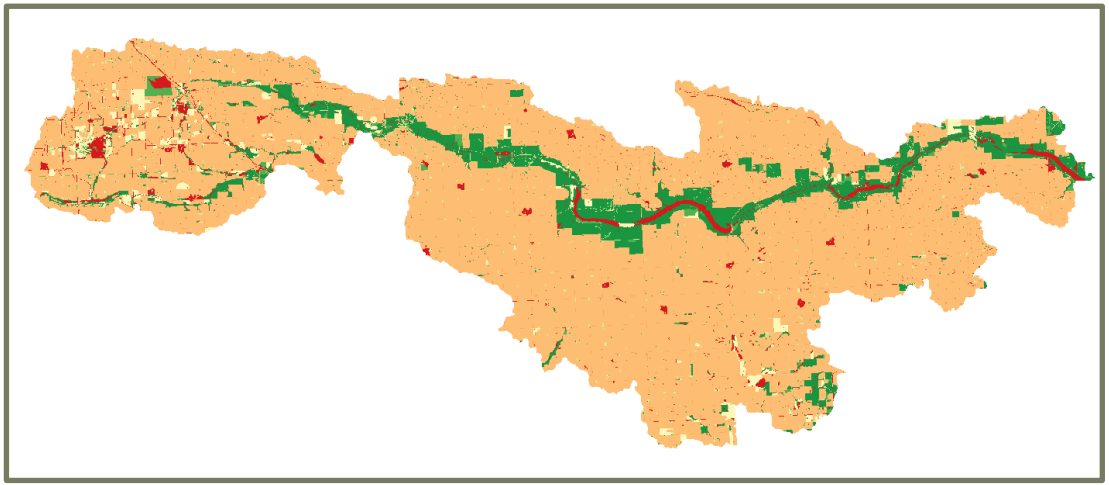
- SF failed to load into memory
- RPyGeo completed in 45.0s



RPyGeo

Wide potential

- Cost raster
- Euclidean distances
- Proximity
- Contour layers
- Table queries



Which approach to use?

SF

- Datasets are small.
- Require the output for visualization.
- No access to ArcGIS.

RPyGeo

- Datasets are medium to large.
- Do not require outputs to be visualized.
- Access to ArcGIS.
- Require unique functions such as Cost Distance.
- Use Raster and Vector datasets within single analysis.
- Unfortunately, package is no longer being updated.

R-ArcGIS Bridge

- R-ArcGIS Bridge is a new package for linking R and ArcGIS.
- Supports integration with Rstudio and ArcGIS Pro.

Thank You!