Space Syntax Analysis in R

Petros Koutsolampros
petros.koutsolampros.11@ucl.ac.uk
Kimon Krenz
k.krenz@ucl.ac.uk

The Bartlett School of Architecture
University College London

This workshop will present a workflow for working with spatial data common to the space syntax field in the R programming language. The workshop will introduce you to the *rdepthmap* package by Petros Koutsolampros, Fani Kostourou and Kimon Krenz. Participants will become familiar with 1) importing spatial data for urban and building scale, 2) running space syntax analysis with the depthmapX cli and 3) managing and plotting these and other related datasets.

RStudio: https://www.rstudio.com/products/rstudio/download/#download
rdepthmap repository: https://github.com/orange-vertex/rdepthmap
depthmapX: https://github.com/SpaceGroupUCL/depthmapX/releases/tag/v0.7.0

07 2019 - BEIJING, CHINA

1 RStudio basics

Introduction to RStudio / How to work with R?

We will introduce you to the general Interface of RStudio and its environment. Specifically focus is on the:

- Console (window in which you can type R commands)
- Environment (the place where you store variables)
- R Help (Information on R commands: *help()* and *?*)
- R Notebooks (Markup document with executable chunks)

Tasks:

1. Create a Notebook
2. Load *cars* dataset
3. View dataframe column names and data
4. Plot and explore variables of the *cars* dataset

Useful commands:
load()
names()
str()
plot()
hist()

2 Spatial Data in R

Introduction to classes and methods for spatial data in R

We will introduce you to the main spatial data types (points, lines, polygons and pixels) necessary for performing space syntax analysis in R. Each spatial data type has different attributes and properties, which you will learn through a series of tasks.

Tasks:

1. Load MIF file (barnsbury_small_axial)
2. Visualise data

Useful commands:
readOGR()
plot()

Useful readings:
https://cran.rstudio.com/web/packages/rgeos/rgeos.pdf
https://cran.r-project.org/web/packages/rgdal/rgdal.pdf

https://cran.r-project.org/web/packages/maptools/maptools.pdf

2.1 Spatial Points

Working with spatial point data / What are SpatialPointsDataFrame?

Spatial points data frames consist of a series of *points* (which may have 2 or 3 dimensions). Points are the most basic spatial data objects. They are generated out of either a single coordinate or a set of coordinates, like a two-column matrix or a dataframe with a column for latitude and one for longitude.

Task:

1. Load spatial points data (barnsbury_ped_gatecounts)
2. View spatial points data frame column names
3. Visualise spatial points
4. Generate Convex Hull around research area

Useful commands:
gConvexHull()

2.2 Spatial Lines

Working with spatial line data / What are SpatialLineDataFrames?

Spatial line data frames are a collection of lines. Lines are generated out of Line objects. A Line object is a consecutive collection of 2D coordinates (or points for this matter) and is generated out of a two-column matrix or a dataframe with a column for latitude and one for longitude.

Task:

1. Load spatial line data
2. View spatial line data frame column names
3. Visualise spatial lines
4. Calculate total line length
5. Generate line centroids
6. Plot using different colours depending on a column

Useful commands:
gLength()
gCentroid()
Plot using colours:
plot(map, col = cols[cut(map$column, 100)])

2.3 Spatial Polygons

Working with spatial polygon data / What are SpatialPolygonsDataFrame?

Spatial polygons data frames are a collection of polygons. A Polygon object is a very much like lines a consecutive collection of 2D coordinates, but with equal first and last coordinates and is generated out of a two-column matrix or a dataframe with a column for latitude and one for longitude.

Task:

1. Load spatial polygon data (barnsbury_plots)
2. View spatial polygon data frame names
3. Visualise spatial polygon
4. Calculate and visualise boundary of area

Useful commands:
gCentroid()
gBoundary()
gConvexHull()
gArea()

2.4 Spatial Pixel

Working with spatial pixel data

Spatial pixel data frames are based on a spatial grid including information on the offset, cell size and dimensions.

Task:

1. Load spatial point data (gallery_vga)
2. Convert to pixel data
3. View spatial pixel data frame names
4. Visualise spatial pixels

Useful commands:
gLength()
gCentroid()

3.0 rdepthmap

Introduction to the rdepthmap package / What are the basic functions?

We will introduce you to the structure and main functions of the *rdepthmap* package. This includes importing, exporting, converting and analysing different types of spatial data at the scale of urban areas and buildings.

To load the library:

```
library(devtools)
install_github("orange-vertex/rdepthmap")
library(rdepthmap)
```

Useful readings:
Al Sayed, K. (2018) *Space Syntax Methodology*. The Bartlett School of Architecture, UCL: London.

3.1 Axial Analysis

Performing an Axial Analysis in R

The axial line is defined as the longest straight line representing the maximum extension of a point of space. We will show you how to perform an axial analysis in R, how to visualise the results and how to further subset and explore the data.

Task:

1. Import line data into graph files
2. Convert line data to axial map
3. Run axial analysis
4. Plot results of 3 different metrics (e.g. Integration, Choice, etc.)
5. Plot the Integration Core
6. Plot with depthmap colours

Useful commands:
rdepthmap::importLines()
rdepthmap::convertMap()
rdepthmap::axialAnalysis()
rdepthmap::getShapeGraph()
rdepthmap::getTopFeatures()

3.1 Segment Analysis

Performing an Angular Segment Analysis in R

Segment analysis is any analysis of a segment map, including topological, angular and metric analyses. A segment is the section of an axial line or street or path lying between two intersections. We will show you how to perform an angular segment analysis and work with the results.

Task:

1. Convert Axial map to Segment map
2. Run segment analysis
3. Plot results of 3 different metrics
4. Subset dataset

Useful commands:
rdepthmap::convertMap()
rdepthmap::segmentAnalysis()

3.2 Visibility Graph Analysis

Performing a Visibility Graph Analysis in R

Visibility graph analysis (VGA) investigates the properties of a visibility graph derived from a spatial environment. We will show you how to run a VGA through *rdepthmap* in R. Moreover, you will learn how to visualise and work with the analysis results.

Task:

1. Import building plan
2. Run VGA
3. Plot results of 6 different metrics
4. Import additional polygons
5. Aggregate values on polygon boundary

Useful commands:
rdepthmap::vga()

3.3 Agent Based Analysis

Performing an Agent Based Model in R

An agent-based model is a simulation of individual movement behaviour in which 'agents' choose their direction of movement based on a defined visual field derived from visibility graph analysis. We will show you how to perform such a simulation through *rdepthmap* in R.

Task:

1. Run Agent Based Model
2. Plot results of 6 different metrics

Useful commands:
rdepthmap::agentAnalysis()

4.0 Spatial Correlations

Linear correlation modelling / How can we correlate different metrics and different spatial data types with each other?

We will introduce you to linear models. You will learn how to fit linear models (LM) in R. LM's can be used to carry out regression, single stratum analysis of variance and analysis of covariance. You will learn how to

Task:

1. Spatially join gate count data to segments
2. Fit linear model
3. Plot results

Useful commands:
as.formula()
lm()
summary()