Geospatial analysis in Scala

Roxana Tesileanu

roxana.te@web.de
INCDS, Romania

October 2017

Contents

1 Introduction

The aim of the present paper is to investigate the use of some of the existing libraries for geospatial analysis available in Scala, the Geospatial Data Abstraction Library (GDAL) and Geotrellis, for performing the main geospatial analysis tasks: manipulating vector and raster data (geoprocessing) and geostatistics. The former task will be approched using GDAL and the later using Geotrellis.

2 Geoprocessing using GDAL

Using a programming language for geospatial analysis allows you to customize your analyses instead of being limited to what the software user interface allows. This is one of the most important advantages of open source software [1].

The GDAL library is one of the open source libraries used in this work. It was written in C and C++ and has bindings for several languages (Java, Perl and Python).

In order to use GDAL, you need to install it on your machine, and for its import in Scala you need to install its Java bindings along with it. For installation details you can look at the GDAL homepage http://www.gdal.org/, download GDAL and follow the instructions for building from source, which might not be an easy task, depending on your operating system. Thanks to the efforts of the UbuntuGIS team (https://wiki.ubuntu.com/UbuntuGIS), on Ubuntu, the installation procedure of GDAL and its bindings is done rapidly. Firstly, you need to add the ubuntugis PPA, which offers the official stable UbuntuGIS packages, to your system (https://launchpad.net/ ubuntugis/+archive/ubuntu/ppa). This is done with the commands:

sudo add-apt-repository ppa:ubuntugis/ppa
sudo apt-get update.

Next, you install GDAL on your machine with the commands (http://www.saras afavi.com/installing-gdalogr-on-ubuntu.html, https://packages.ubuntu.com/sou rce/trusty/gdal):

sudo apt-get install libproj-dev, gdal-bin, libgdal-dev, libgdal-doc
sudo apt-get update.

Finally, you add the Java bindings to your GDAL package (https://launchpad.ne t/ubuntu/+source/proj):

sudo apt-get install libgdal-java, libproj-java.

In order to import GDAL in Scala, you have to add its jar to the project's classpath. An easy way of managing dependencies of a Scala project is to use SBT (for further details see [2]). In this way you can take advantage of the most convenient way to place the gdal jar to the project's classpath, namely, to place a copy of it into the lib directory of the Scala project, now that the actual installation has already taken place.

The following subsections will offer a background in geoprocessing, starting with manipulating vector data (reading and writing files of different vector data formats and performing overlay and proximity analyses), and continuing with manipulating raster data (reading and writing files of different raster data formats, resizing pixels, performing moving window analyses and map algebra).

2.1   Types of spatial data

Spatial data are divided in two categories: vector data and raster data. Vector data provide information about distinct features in space, i.e. different distinct

items of interest, and are made up of points, lines and polygons [1]. The features of interest could be for example:

- roads, rivers, road networks, hidrological networks, country boundaries, city boundaries as examples of features represented by lines,

- mountain peaks, volcano peaks, weather stations, restaurants, as examples of features represented by points, and

- lakes, oceans, ownership status as examples of features represented by polygons.

Features have attributes attached to them such as the name of the individual observations (for example the wheather stations's name) and other recorded variables (like for example different concentrations of air pollutants, temperature or wind regime for each individual weather station). As it can be noticed, the multiple attributes which can be attached to features, can be of different types, and they actually represent different types of recorded variables (they might be dicrete or continuous numerical variables or categorical variables).

On the other hand, raster data provide information about characteristics of interest which take the form of a continuum like gradients, with no distinct boundaries. They are represented as two- or three-dimensional arrays of data values which form grids of values [1]. Because they can cope well with gradients, they capture local variation more easily than vector geometries, and are used in digital elevation models (DEMs). Also because the data source is pixel-based (e.g. aerial photos, satellite imagery) they can be used in vegetation mapping.

2.2 Reading vector data

The main objective of vector data analysis is to investigate relationships between features, by overlapping them on another or measuring distances between them [1]. A typical example for vector analyses is the investigation of GPS-collared wildlife to see the direction of travel, distances covered and how they interact with man-made features like roads [1].
In order to perform such vector-based analyses, we need to be able to read, edit and write vector data. This kind of functionality is offered by the OGR Simple Features Library for geoprocessing vector data, which is included in GDAL.

3   Geostatistics using Geotrellis

References

[1] Chris Garrard. *Geoprocessing with Python*. Manning Publications Co., Shelter Island, 2016.

[2] Roxana Tesileanu. Using Linux as a development platform for Scala projects, 2017.