A comparison of command-line raster map algebra frameworks

Lorenzo Booth

ES 207

April 21, 2016

Keywords: map algebra, PostGIS, raster, Rlang, reproducible research

Abstract

Many applications in the natural sciences leverage remotely sensed measurements of the environment. Satellite data products in particular expose these measurements as gridded (raster) parameters. Geographic information systems (GIS) have emerged as the de-facto framework for managing, analyzing, and visualizing spatial data. Entire new applications of computational geometry have entered common use in what are now routine techniques for exploring spatially referenced data. Map algebra is one such technique that applies mathematic operations across individual elements of a raster data set. Combinations of simple algebraic, logical, combinatorial and relational operations are routinely used to construct complex new derivative data products that describe geographic systems in manners beyond what would have been possible to obtain from an individual measurement.

The current wealth of remotely-sensed data has moved the frontier of environmental monitoring from improving data coverage to the production of novel analytics methods; large volumes of information present new challenges in large-scale data management and performance computing. This paper explores the current state of two open-source methods for performing map algebra on gridded data sets: the raster R library and PostGIS. The speed and memory use efficiency of different subset and map algebra operations are compared within the open-source statistical computing environment, R. We found that for operations that fit in memory are far more efficient in R than on a PostGIS database. However, PostGIS is more appropriate for certain operations on large data sets that do not fit in memory depending on parameters that can be tuned to the target data set.

Introduction

Many sectors of environmental science (e.g. hydrology, climatology, ecology, agriculture) leverage gridded datasets of environmental variables from interpolated surfaces and/or satellite images. The outputs of environmental models (e.g. general circulation models) can also emulate physical through “cells” which simplify the spatial heterogeneity of the Earth’s features within a particular unit simulation segment (Intergovernmental Panel on Climate Change, 2014, p. 201). Similarly, these applications produce sets of gridded parameters.

Advances in earth observation platforms and environmental models have created a situation where environmental scientists must face the challenges associated with managing and visualizing high-speed and high-resolution data products. These challenges align with the “3 V’s” of data management identified by Doug Laney in the early 2000s, terms that have entered modern parlance as the descriptors of “big data” systems and database platforms (Laney, 2001). Warehousing geospatial data in relational database management systems (RDBMS) has a long history, tracing back to Canadian Hydrographic Service, that added spatial data capabilities to Oracle 4, which later became incorporated into Oracle Spatial (Varma et al., 1990).

In the past decade, open-source geographic information systems (GIS) and RDBMS have matured, matching many of the features found in established GIS and database systems. The R statistical computing environment has emerged as a platform for literate programming and reproducible research through the publication of data sets and their respective analysis code (Kuhn, 2015). Functions for manipulating raster datasets are provided by the open-source Geospatial Data Abstraction Library (GDAL) and are exposed within the raster R package (Hijmans, 2015). The raster package implements classes for raster data within R and implements functions for raster algebra and overlay functions, vector to raster conversion, plotting, reprojection, and other manipulation operations. The raster package can also perform operations on data that reside on-disk, contrasting with the base functions of R, which operate on data that resides in-memory.

PostGIS is an extension to the open-source PostgreSQL RDBMS that provides functions and provisions for the storage and manipulation of spatial data (PostGIS Project Steering Committee, 2013). Provisions for operating on raster data were first introduced with the WKT Raster extension for PostGIS in 2008, before becoming incorporated into the upstream project with the release of PostGIS 2.0 in Spring 2012 (Loskot et al., 2008). PostGIS uses GiST (Generalized Search Tree) or R-Tree indexes to provide spatial indexing. Base raster operations (e.g. resampling, clipping, reclassification, and map algebra) are implemented both on a Postgres database, or on individual raster files through a GDAL driver.

Methods

Consistent with other benchmarking efforts, we use two typical queries for the benchmark: spatial overlay (containment query) and arithmetic operation (selection query) (Aji et al., 2013). Many GIS applications in ecosystem science use these primitive operations. For example, the construction of an area Normalized Difference Vegetation Index (NVDI) can be broken into three arithmetic operations: one subtraction of images corresponding to the visible and near infrared bands, one addition of the the aforementioned bands, and a division of the intermediate difference and sum.

Like other benchmarking efforts, we use real environmental datasets to emulate real-world applications of spatial operations. The primary gridded dataset used was the 2010 California Multisource Augmented Landcover Map (CAML) (Hollander, 2010). The data set fuses multiple surveys including: land cover maps from the California Department of Water Resources, pesticide use reports from the California Department of Pesticide Regulation, farmland maps from the California Department of Conservation, and previous multi-source land cover maps from the California Department of Forestry and Fire Protection. Each pixel represents 50 square meters of land surface area. The primary vector dataset used was the 2014 Cartographic Boundary File for the state of California. Specifically, county political boundaries were used from the US Census Bureau’s geographic database (MAF/TIGER, 2014). Table 1 shows general properties of the data sets.

To test the standalone performance of each method, operations were run on a single node as a single thread application. Tests were run with the latest development versions of R (v 3.2.4, 2016-03-16 r70336), the raster package (v. 2.5-2) PostgreSQL (v. 9.5.2) and PostGIS (v. 2.2.2, r14797) within a single thread Debian Stretch virtual machine (Linux kernel 4.1.13) with access to a maximum of 6GB system memory.

The raster package has two classes for multi-layer data objects, the RasterStack and the Raster Brick. In essence, a RasterStack is a list of potentially separate RasterLayer objects. RasterBricks are multi-layered objects that can only refer to the multiple bands of a single file. RasterBricks are also created only in memory, sacrificing some flexibility for speed.

Queries on objects created with the raster package and PostGIS were evaluated through R, either as direct calls to raster functions or as system calls to PostGIS queries. The speed of different operations was evaluated through the system.time function of the base R library, a function that returns the CPU time required for the expression to be evaluated.

Results

For raster layers that are small enough to fit in-memory, the raster package is several orders of magnitude faster for subset operations (Figure 2). This was evaluated by coercing the land cover data into a RasterBrick object (which forces the file into memory). Initial evaluations of PostGIS raster functions were performed by registering the raster data file outside of the Postgres database. This way, only the raster metadata and file path are stored in the database system, and PostGIS functions are evaluated on the source data file directly. The results of these operations are summarized in Table 2.

Conclusion

For polygon overlays over datasets that reside on disk, PostGIS is faster by orders of magnitude. However, for small data sets, the raster package excels on operations that can reside in memory. PostGIS tunable parameters include the tile size over which gridded data are split to perform iterative operations. An optimum tile size may depend on the structure of the gridded data set and/or the size of the file. Further benchmarks could compare the in-memory performance of the raster package to PostGIS operations over data that is stored in a ramdisk.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data Type | File Name | Size | Resolution | Description |
| Raster | caml2010.3310 | 390.7 MB | 10546x9140 | California landcover map |
| Polygon | ca24k09\_poly\_noislands | 5.2 MB | 1:5,000,000 (5m) | California county political boundaries |

Table 1: Description of data sets and associated metadata

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Library | Structure | Time (seconds) |
| Point overlay | Raster | RasterBrick | 0.008 |
| Polygon overlay | Raster | RasterLayer | 307.47 |
| Polygon overlay | Raster | RasterBrick | 50.91 |
| Polygon overlay | PostGIS | st\_union, st\_clip | 10.21 |

Table 2: Time elapsed for completion of benchmarked overlay functions.

Review:

The main point of this work is to introduce and compare two existing open source software packages, the raster R library and PostGIS. These packages are used for interpreting and analyzing gridded/rasterized data-sets. The speed and memory use efficiency of each operation is the metric by which to compare each of the two packages. I brief history of gridded data and GIS packages is given. The methods for the experiment are given, where raster and polygon gridded data-sets are interpreted via the libraries previously stated. The results of the experiment are disseminated through a table that outlines the computation time for each method. A conclusion is drawn that there are benefits and drawbacks from both methods, depending on the scale of the data-set. However, according to the findings, there is clear superiority when analyzing polygonal data through the use of the PostGIS.

This work is publishable; however, it needs work along the analysis of performance. The abstract and introduction and clear and well presented. The methods for the experiment could be slightly clearer and maybe performed on a few more datasets for validity. Overall, a very interesting and relevant paper.