Free and Open Source Tools for GPS Data Management and Analysis

Alex Mandel • Geography Graduate Group • University of California, Davis

Introduction

This poster highlights methods for dealing with large amounts of spatial data generated from GPS tracking technology, specifically focusing on Free and Open Source software. The advantages to using Free and Open Source software are it is inexpensive (free), works on any operating system, uses open data formats, and has a large and friendly community for help. Examples below demonstrate how to use a combination of **Spatialite**[1], **Quantum GIS (QGIS)**[4], **R**[5] and **Python**[3] for an inter-connected system of data storage, visualization, analysis, and automation.

Data Import: Python

I started with 32 individual spreadsheets (one per animal) containing descriptive information and more than 50,000 GPS collar point locations. Using Python[3] I copied only the relevant location data into an SQLite[2] database and created a Spaitialite[1] geometry layer from the latitude and longitude. With the data now in a spatial database, analysis and visualization are possible with a few Structured Query Language (SQL) queries (Fig. 1).

Data Work-flow

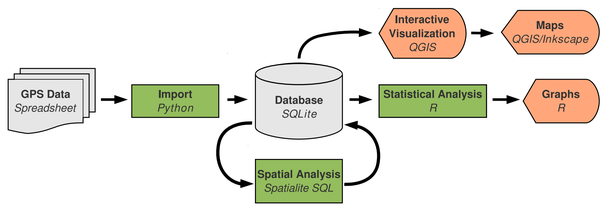

Figure 1: Flowchart of Data Management and Analysis

Data Management: Spatialite

Below is a query that merges two separate date and time columns into a single column. By putting the date and time into a single column the database is now capable of calculating differences in time records over multiple days. The new time-stamp format adheres to an international standard[6] which should make it easier to use in other applications.

```
--Create time-stamp from date and time columns.
ALTER TABLE locations
ADD COLUMN "timestamp" TEXT;

UPDATE locations SET "timestamp" = (
substr("date",7,4) || '-' ||
substr("date",1,2) || '-' ||
substr("date",4,2) || 'T' ||
"time");
```

Before		*After*
Date	Time	Time-stamp
01/16/2002	12:15:48	2002-01-16T12:15:48

The following query checks for duplicate records based on their time-stamp (a combined field of date and time).

```
--Check for duplicate data
SELECT DISTINCT("timestamp"), count(point_id) as tot, group_concat(
  point_id)
FROM locations
GROUP BY "timestamp", puma
ORDER BY tot DESC;
```

Distance vs. Time: Spatialite

A query demonstrating how to calculate distance traveled between successive GPS readings on the same individual for all pairs of points in the entire dataset:

```
--Calculate distance traveled per unit of time between GPS recordings
CREATE TABLE loc2 AS
SELECT (ROWID-1) as previous, point_id as p2, timestamp as t2,
  albers83 as end
FROM locations
WHERE point_id NOT
IN (SELECT point_id WHERE point_id LIKE '%-0001%' OR point_id LIKE '
  %-001');
```

Data Model

GPS point locations and a polygon based habitat model loaded as 2 layers in the database allowing a join by location (Fig. 2).

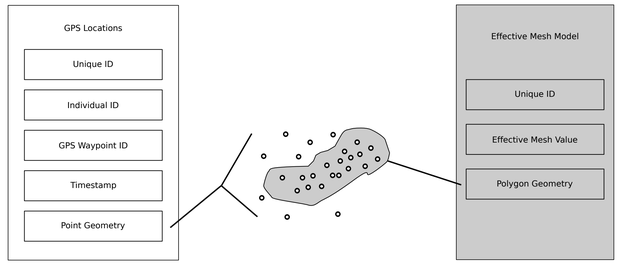

Figure 2: Basic data model of the database.

Optimized Spatial Query: Spatialite

An optimized spatial query identifying which polygon each point location occurred within:

```
--Create the ram cached spatial index of locations
SELECT CreateMbrCache('locations','albers83');
--Optimized spatial query
CREATE TABLE locationsXfg3geom as
SELECT locations.puma as puma,locations.point_id as point_id,
fg3geom.RBUASPW as UID
FROM locations, fg3geom
WHERE locations.rowid IN
(SELECT rowid
FROM cache_locations_albers83
WHERE mbr = FilterMbrWithin
(MbrMinX(fg3geom.Geometry), MbrMinY(fg3geom.Geometry),
MbrMaxX(fg3geom.Geometry), MbrMaxY(fg3geom.Geometry) ))
AND Within(locations.albers83,fg3geom.Geometry);
```

Data Analysis: R

Data from Spatialite[1] can be used directly in R[5] by opening a database connection and retrieving the results of query for statistical analysis.

```
#Connect to database
library(RSQLite)
m <- dbDriver("SQLite")
con <- dbConnect(m, dbname = "puma080307test.db",loadable.extensions =
   TRUE)
#Query only animals with n > 100, Join to results of intersection of
  points with polygons with attributes from original point data
d3a <- dbGetQuery(con, 'SELECT resultsJfg3.puma as pumaid, point_id,
  UID,EffMeshCBC FROM resultsJfg3 Join (SELECT DISTINCT puma, count(
  puma) as total FROM locations GROUP BY puma) as number ON pumaid =
  number.puma WHERE number.total > 100')
```

Results

- Data can easily be normalized, joined, and summarized with SQL.
- Quick way to locate errors and anomalies in a dataset.
- Spatialite[1] and spatial SQL are capable of typical vector analysis.
- Spatialite[1] offers some unique speed optimization methods.

Visualization: Quantum GIS

Viewing GPS points overlaying habitat fragmentation model polygons (Fig. 3).

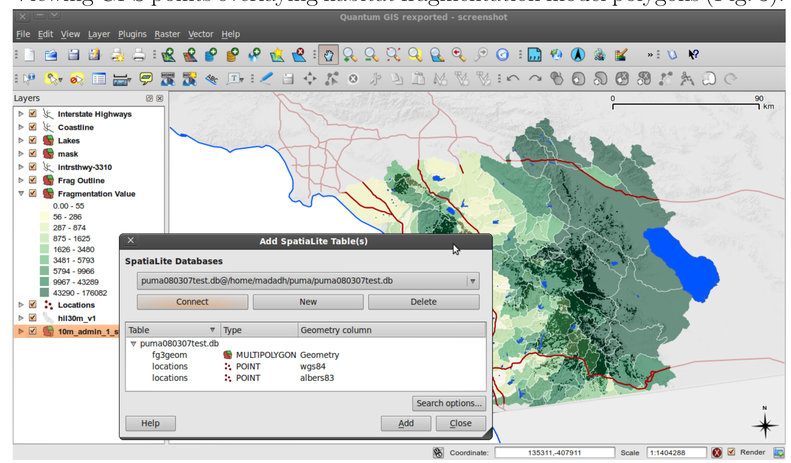

Figure 3: QGIS[4] Visualization with Spatialite[1] data loading dialog

Discussion

- Using a database makes management of large datasets less laborious.
- Spreadsheets are useful for some tasks, but prone to errors.
- Spatial SQL is a viable and efficient tool for modern GIS.
- Learning to mix Spatialite[1], R[5], QGIS[4], and Python[3] takes some time but is not difficult.

References

[1] Alessandro Furieri. *SpatiaLite Cookbook*. 2011. URL: http://www.gaia-gis.it/spatialite-2.4.0-4/spatialite-cookbook.

[2] Dr. Richard Hipp. *SQLite*. 2011. URL: http://www.sqlite.org.

[3] Python Software Foundation. *Python Programming Language*. 2011. URL: http://www.python.org.

[4] Quantum GIS Development Team. *Quantum GIS Geographic Information System*. Open Source Geospatial Foundation. 2011. URL: http://qgis.osgeo.org.

[5] R Development Core Team. *R: A Language and Environment for Statistical Computing*. ISBN 3-900051-07-0. R Foundation for Statistical Computing. Vienna, Austria 2010. URL: http://www.R-project.org.

[6] ISO International Organization for Standardization. *ISO - International Organization for Standardization*. http://www.iso.org/iso/date_and_time_format. Text. 2011. URL: http://www.iso.org/iso/date_and_time_format.

Acknowledgements

Walter Boyce, Wildlife Health Center, UC Davis; Jim Thorne, Jim Quinn and Evan Grivetz, Information Center for the Environment, UC Davis; Alessandro Furieri; the Quantum GIS community; the R community; the Open Source Geospatial Foundation's Python community

email: aimandel@ucdavis.edu