Introduction to PySAL and Web Based Spatial Statistics

Myung-Hwa Hwang
GeoDa Center for Geospatial Analysis and Computation
School of Geographical Sciences and Urban Planning
Arizona State University

Spatial Data Analysis in Cyberspace:
Data, Methodology and Tools Workshop
58th Annual North American Meetings
of the Regional Science Association
November 09, 2011
Acknowledgements

- National Science Foundation
- National Institutes of Health
- National Institute of Justice
Outline

- GeoDa Center
- Software Implementation
- Software Library for Spatial Analysis
- PySAL
- Spatial Analytical Web Services
- GeoDa Center Services
- Next Steps
GeoDa Center
GeoDa Center

- Foundations
  - Spatial Analysis Laboratory (UIUC)
  - Regional Analysis Laboratory (SDSU)
- GeoDa center for geospatial analysis and computation
  - Method development
  - Software implementation
  - Policy-relevant research
  - Training and support
## GeoDa Center Software

### Standalone
- GeoDa
- OpenGeoDa
- STARS
- GeoDaSpace*
- GeoDaNet*

### Open Source
- STARS
- PySAL
- OpenGeoDa

### Web Services & Application
- Analytical Services*
- Web-based ESDA Tool*
- DynTM*

### Cyber Infrastructure
- Parallel PySAL

* Under development
One Implementation, Multiple Interfaces

PySAL
OpenGeoDa

Command Line Executable

Parallel Application

GIS Extension

Distributed Services

Standalone Software

Web Application

Distributed Services

Parallel Application

Command Line Executable

GIS Extension

Standalone Software

Web Application

Distributed Services

Parallel Application

Command Line Executable

GIS Extension

Standalone Software

Web Application
Software Library for Spatial Analysis
Library

- A collection of code and data that can be shared by software programs (Neumann and Hugentobler, 2008)
  - Helper software
  - Reusable
  - Modular
  - Programming
  - API

http://en.sap.info/soa-innovation-lab/16709
PySAL

- A core library and API for various spatial analysis methods (Rey and Anselin, 2007)
- Reuse existing resources (e.g. GeoDa, STARS)
- Avoid duplicated development efforts
  - Open source project (BSD license)
- Contribute to the scientific Python community
  - Spatial Econometrics, ESDA
Team

- Luc Anselin
- Daniel Arribas-Bel
- David Folch
- Myunghwa Hwang
- Nicholas Malizia
- Pedro V. Amaral
- Serge Rey
- Charlie Schmidt
- Phil Stephens
- Andrew Winslow
- Xinyue Ye
- Ran Wei
- Jing Yao
Use of PySAL

- PySAL
- GIServices
- Command-line Executable
- GIS Extension
- Standalone Software
- Web Application
Release Cycle

- Release every six months

1.0 (July 31, 2010)
1.1 (January 31, 2011)
1.2 (July 31, 2011)
1.3 (January 31, 2012)

8k+ Downloads
Core Functionality
Spatial Weights Matrices

- Spatial weights object (class) $W$
- Binary, continuous weights
- Rich set of attributes and methods
- Cacheing and properties
- Spatial lag and set-like functionality
- Current limit of about 2 million observations
Spatial Weights Types

- Contiguity weights
  - Queen, rook
- Regime weights
- Distance weights
  - Nearest neighbor, distance band
- Kernel weights
  - 7 functions, adaptive bandwidths
Performance
- Weights Creation

![Graph showing computation time to create lattice (queen) weights.
- Red line for R.
- Blue line for PySAL Spreg.

The graph plots the computation time in minutes against the number of elements (N) for two different software tools, R and PySAL Spreg.
Example
- Weights Creation

Python 2.6.6 (r266:84374, Aug 31 2010, 11:00:51)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import pysal
>>> w = pysal.lat2W(5,5,'queen')
>>> w.n
25
>>> w.pct_nonzero
0.128
>>> w.weights[0]
[1.0, 1.0]
>>> w.neighbors[0]
[5, 1]
>>> w.histogram
[(2, 4), (3, 12), (4, 9)]
>>> shp = "pysal/trunk/pysal/examples/columbus.shp"
>>> thresh = pysal.min_threshold_dist_from_shapefile(shp)
>>> thresh
0.61886415807685413
>>> wt = pysal.threshold_binaryW_from_shapefile(shp, thresh)
>>> wt.min_neighbors
1
>>> wt.histogram
[(1, 4), (2, 8), (3, 6), (4, 2), (5, 5), (6, 8), (7, 6), (8, 2), (9, 6), (10, 1), (11, 1)]
File Input and Output

- Data tables: .dbf and .csv
- Spatial data: .shp and .wkt
- Weights (1.2)
  - .gal, .gwt, .swm, .dbf, .mat, .dat, .wk1, .mtx, .txt (arcgis, stata, geobugs)
- User-friendly file IO
  - Single function to read all data files: pysal.open()
  - Convenience functions
    - ex: queen_from_shapefile()
Example
- Weights Conversion

```python
Python 2.6.6 (r266:84374, Aug 31 2010, 11:00:51)
[GCC 4.0.1 (Apple Inc. build 5493)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import pysal
>>> w = pysal.lat2W(5,5,'queen')
>>> gal = pysal.open('grid25_queen1.gal', 'w')
>>> gal.write(w)
>>> gal.close()
>>> from pysal.core.util.weight_converter import weight_convert
>>> swm_file = 'grid25_queen1.swm'
>>> weight_convert('grid25_queen1.gal', swm_file, useIdIndex=True)
>>> wnew = pysal.open(swm_file, 'r').read()
>>> w.n
25
>>> wnew.n
25
>>> 
```
Analysis Modules

- esda (1.0, 1.2)
- inequality (1.0)
- regionalization (1.0)
- spatial_dynamics (1.0, 1.2)
- spreg (1.1)
ESDA

- Measures of spatial autocorrelation
- Moran’s I, Geary’s c, Getis and Ord’s G(*)
- Map Classification
  - Natural breaks, Fisher Jenks, equal interval, and others
- Rate Smoothing
  - Empirical Bayes, age-adjusted, excess risk, and others
Example - LISA

```python
>>> import pysal
>>> import numpy as np
>>> f = pysal.open("pysal/trunk/pysal/examples/stl_hom.txt")
>>> y = np.array(f.by_col["HR8893"])
>>> w = pysal.open("pysal/trunk/pysal/examples/stl.gal").read()
>>> lm = pysal.Moran_Local(y, w)
>>> lm.Is[:10]
array([-0.07728772, 0.17282319, 0.10175766, 0.08910419, 0.16279687,
        -0.29260054, -0.0506616 , 0.14549975, -0.00092416, 0.0425654 ])  
>>> lm.permutations
999
>>> lm.p_sim[:10]
array([ 0.173, 0.065, 0.435, 0.282, 0.34 , 0.063, 0.264, 0.237,
        0.073, 0.055])
>>> from pysal.esda.getisord import G_Local
>>> dist_w = pysal.threshold_binaryW_from_shapefile('pysal/trunk/pysal/examples/stl_hom.shp', 0.6)
>>> dist_w.transform = 'b'
>>> lg = G_Local(y, dist_w)
>>> lg.Gs[:10]
array([ 0.0126883 , 0.00532696, 0.010375 , 0.00427599, 0.00598784,
        0.00360766, 0.00713529, 0.00593378, 0.00675208, 0.00719555])
>>> lg.Zs[:10]
array([-0.01764622, -0.45143752, -0.15439894, -0.51294976, -0.41629794,
        -0.54361729, -0.34098885, -0.41638556, -0.36467953, -0.33954006])
>>> lg.p_sim[:10]
array([ 0.338, 0.04 , 0.465, 0.014, 0.071, 0.009, 0.084, 0.167,
        0.068, 0.078])
```
Inequality

- Theil Index
  - Entropy based measures of spatial inequality
- Regional decompositions
  - Interregional inequality
  - Intraregional inequality
Regionalization

• max-p (Duque, Anselin, and Rey, 2012)

• Given n areas, form the maximum number (p) of regions respecting contiguity and threshold constraints

• Random regions

• Randomly construct regions given various constraints
Spatial Dynamics

- Markov transition matrices
- Classic, spatial, LISA
- Space-time interaction tests (1.2)
  - Knox, Mantel, Jacquez
- Space-time rank mobility tests
- Space-time LISA
- Directional LISA
  - Rey, Murray, and Anselin, 2011
Spatial Regression

• 1.1
  • OLS and spatial diagnostics

• 1.3
  • Spatial lag, error, and combo models
  • GMM estimation methods
  • Heteroskedasticity
  • GeoDaSpace GUI front end
PySAL 1.3+

- Support for Python 3.x
- Parallel LISA
- Contrib modules
  - New in 1.3
  - Leverage third-party libraries
    - Shapely, proj4, gdal
More About PySAL

http://pysal.org
Spatial Analytical Web Services
Web Services

- A collection of software programs that provide online access to shareable data and operations
- Remote libraries

http://en.sap.info/soa-innovation-lab/16709
Web Services vs Libraries

• Commonalities
  • Reusable, modular, API-based

• Differences

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Web Services</th>
<th>Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Where an operation runs</td>
<td>Remote server</td>
<td>Desktop</td>
</tr>
<tr>
<td>Software management</td>
<td>Service provider</td>
<td>User</td>
</tr>
<tr>
<td>Software update</td>
<td>Automatic</td>
<td>Semi-automatic</td>
</tr>
<tr>
<td>Functionality</td>
<td>Coarse-grained</td>
<td>Fine-grained</td>
</tr>
<tr>
<td>Interaction</td>
<td>Standard-based</td>
<td>No standard</td>
</tr>
</tbody>
</table>
Use of Web Services

- Software as a Service (SaaS)
- No install, on-demand, pay-as-you-go
- Publish-Find-Bind
Spatial Analytical Web Services

• Services providing access to spatial analysis methods

GIS
Map Overlay
Buffering
Spatial Join
Aggregation
Spatial Database
2D/3D Mapping
Geocoding
Digitizing

Geovisualization
Diagrams
Charts
3D Views
Maps
Digital Globe
Linking
Brushing
Animations
Fly-throughs

Spatial Weights
Nearest Neighbor
Spatial Autocorrelation
Interpolation
Kriging
Spatial Clustering
Spatial Regression
GeoDa Center Services

- Designed as an interface to PySAL
- Updated after the major releases of PySAL
- Current functionality
  - Spatial weights manipulation, ESDA, tile mapping
- Two types of access
  - APIs for developers
  - Web 2.0 applications for end users
- Python-based implementations
Service Implementation

Client Tier

- Web Application
- Command Line Shell
- Desktop GIS Extensions
- Other Clients

Server Tier

- Web Server: Apache2 modwsgi
- Web Service Wrapper: Python django
- GeoDa Center Software Libraries: PySAL
- Hardware Backend: Amazon Cloud
Service API Example
- LISA Service

• Returns local cluster types given a spatial weights file and an array of numbers

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Endpoint</td>
<td><a href="http://50.19.233.151/services/moran/json/">http://50.19.233.151/services/moran/json/</a></td>
</tr>
<tr>
<td>Request Methods</td>
<td>HTTP GET, POST (REST Approach)</td>
</tr>
<tr>
<td>Parameters</td>
<td>y, moran_type, w, w_format, permutation</td>
</tr>
<tr>
<td>Returns</td>
<td>success, q (cluster type), p_sim</td>
</tr>
<tr>
<td>Payload Encoding</td>
<td>JSON, y and w are base64-encoded</td>
</tr>
<tr>
<td>Authentication</td>
<td>-</td>
</tr>
</tbody>
</table>
Interacting with LISA Service

- **Request payload**

  ```
  http://50.19.233.151/services/moran/json/?
  y=ZAAAAAAAAAAAAAAAAAAAAAAAAAFDwPwAAAAAAhA&
  w=YzQKMCAyCjIgMQoxIDIKMCAzCjIgMgowIDMKMyAyCjEgMgo=&
  w_format=gal&moran_type=local&permutation=99
  ```

- **Response payload**

  ```
  {
    status: "success"
    q: [2, 4, 3, 1],
    p_sim: [0.41, 0.01, 0.32, 0.29]
  }
  ```
GeoDa Center Services
API Documentation

7. Spatial Weights Requests

A Spatial Weights API request takes the following form:

http://HOST/services/weights/output?parameters

The Spatial Weights service currently supports HTTP GET and POST request methods, and the POST method is recommended. When the request is successfully processed, the service returns data only in a Javascript Object Notation (JSON) object. Thus, output in the above box should be json.

7.1 Request Parameters

Certain parameters are mandatory while others are optional. As is standard in URLs, all parameters are separated using the ampersand (&) character.

The Spatial Weights API defines a request using the following URL parameters:
- geom (mandatory) - a base64-encoded zip file that contains shp, dbf, and shx files.* The value for this parameter should start with an additional byte indicating the data type. See the example in the following box.

Assume there is an ESRI shape file including 4 contiguous polygons. This file is named grid_4. Its geometry contains:

After reading grid_4.zip in binary mode, you can apply base64 encoding to the file content. In Python, you can

>>> import array, base64, urllib
>>> f = open('grid_4.zip', 'rb')
>>> cont = f.read()
>>> f.close()
>>> cont_array = array.array('c', cont)
>>> if sys.byteorder == 'big': cont_array.byteswap()
>>> cont_base64 = base64.b64encode(cont_array.typecode + cont_array.tostring())
>>> cont_base64 = urllib.urlencode({'geom':cont_base64})
Services
- Known Issues

- Performance & scalability
- Data transfer
- Algorithm
  - Permutation, optimization
- Hardware
- Long-running processes
- Authentication & authorization
Web 2.0 Application
Web 2.0 Application Implementation

**Service Tier**
- GeoDa Center Web Services

**Application Server Tier**
- Mediators
- Cache Server
- Data Manager
- Map Server

- Django
- Django cache framework
- MapServer
- DynTM

**Client Tier**
- JavaScript-based Web GUI

- OpenLayers
- Ext JS, jQuery
Example - Weights Creation

A Web-based Tool for Exploratory Spatial Data Analysis

Weights: Create | Weights: Transform | Weights: Convert | Spatial Lag | Smoothing | Local Moran’s I | Help | Service API

Start Project

To create distance-based weights, please use a Shape file that is projected. For smoothing and local Moran’s I, please use a Shape file that is not projected.

Directory Containing Only SHP, DBF, SHX Files

Input Directory: [Select a directory containing] [Browse...]

Start Upload

This tool was tested on Google Chrome 15.0 and Mozilla Firefox 7.0.1. If you have problems with this application, please visit this web page and add issues. Thank you.
Example - Weights Creation
Example - Weights Creation

A Web-based Tool for Exploratory Spatial Data Analysis

Creating Weights

Variable: FIPS

Contiguity Weights
- Contiguity type: Queen
- The order of contiguity: 1

Threshold Distance Weights

k-Nearest Neighbors Weights

Kernel Weights

Spatial Weights Matrix File

Download Close

This tool is tested on Google Chrome 15.0 and Mozilla Firefox 7.0.1. If you encounter problems with this application, please report them at this web page and add issues. Thank you.
Example
- LISA Analysis & Mapping
Example

- LISA Analysis & Mapping
Example

- LISA Analysis & Mapping
Known Issues

- Data upload
  - Up to 250 MB
- Long-running processes
- ESDA functions under development
- Missing GUI components
  - Data browser
  - Generic data views
  - Table & map
More About
Web Services & Application

http://107.20.203.40/web_esda
Next Steps

• Parallel PySAL
  • Increase performance and scalability
• Web services
  • Parallel PySAL & high-performance hardware
  • Streamline authentication & authorization
  • Support more analytical methods
• Web application
  • Add missing GUI components
  • Stable, interactive ESDA
Questions?
References


