Open Source, Scale and Reproducibility Using GIS: Discovering the World Beyond Point-and-Click and ArcGIS

Ömer Özak

Dept. Economics, SMU

Here’s How I Do GIS

October 13, 2016
The Voyage of Homo-œconomicus into GIS

Ömer Özak

Dept. Economics, SMU

Here's How I Do GIS

October 13, 2016
"the concept in many economic theories portraying humans as consistently rational and narrowly self-interested agents who usually pursue their subjectively-defined ends optimally."
Plan for today

1. The Big Bang
2. The Dark Ages
3. The Age of Discovery
4. The Modern Era
5. The Future
6. Q&A
Why GIS?

The question that started it all
Why GIS?

The question that started it all

- What is the effect of geographical isolation on economic development?
Why GIS?

The question that started it all

- What is the effect of geographical isolation on economic development?
  - How to measure?
Why GIS?

The question that started it all

- What is the effect of geographical isolation on economic development?
  - How to measure?
  - Measure for Pre-industrial era
Why GIS?

The question that started it all

- What is the effect of geographical isolation on economic development?
  - How to measure?
  - Measure for Pre-industrial era
  - Changes due to technology
Why a New Measure?
Why a New Measure?

- Common approach: Geodesic distances
Why a New Measure?

- Common approach: Geodesic distances
  - As the crow flies
Why a New Measure?

Common approach: Geodesic distances
- As the crow flies
- Assumes flying technology or flat world
Why a New Measure?

- Common approach: Geodesic distances
  - As the crow flies
  - Assumes flying technology or flat world
  - Measurement error
Why a New Measure?

- Common approach: Geodesic distances
  - As the crow flies
  - Assumes flying technology or flat world
  - Measurement error

<table>
<thead>
<tr>
<th>Country 1</th>
<th>Country 2</th>
<th>Distance</th>
<th>Country 1</th>
<th>Country 2</th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Costa Rica</td>
<td>Panama</td>
<td>514.3561</td>
<td>Germany</td>
<td>Poland</td>
<td>515.774</td>
</tr>
<tr>
<td>Phillippines</td>
<td>Brunei</td>
<td>1262.339</td>
<td>Yemen</td>
<td>Sudan</td>
<td>1254.947</td>
</tr>
</tbody>
</table>
Proposed Solution

Construct a measure that
Proposed Solution

Construct a measure that

- Controls for

Human biological constraints
Geographical conditions
Technological conditions

Has meaning
Proposed Solution

Construct a measure that

- Controls for
  - Human biological constraints
Proposed Solution

Construct a measure that

- Controls for
  - Human biological constraints
  - Geographical conditions
Proposed Solution

Construct a measure that

- Controls for
  - Human biological constraints
  - Geographical conditions
  - Technological conditions
Proposed Solution

Construct a measure that

- Controls for
  - Human biological constraints
  - Geographical conditions
  - Technological conditions

- Has meaning
Solution: The Voyage of Homo-œconomicus

Combine data on

Infantry movement
Geographical conditions
Ship speeds in different eras

$\Rightarrow$

Human Mobility Index (HMI)
Human Mobility Index with Seafaring pre-1500CE (HMISea)
Human Mobility Index with Seafaring pre-steam engine (HMIOcean)
Solution: The Voyage of Homo-œconomicus

Combine data on

- Infantry movement
Solution: The Voyage of Homo-œconomicus

Combine data on

- Infantry movement
- Geographical conditions
Solution: The Voyage of Homo-œconomicus

Combine data on

- Infantry movement
- Geographical conditions
- Ship speeds in different eras
Solution: The Voyage of Homo-œconomicus

Combine data on

- Infantry movement
- Geographical conditions
- Ship speeds in different eras
Solution: The Voyage of Homo-œconomicus

Combine data on

- Infantry movement
- Geographical conditions
- Ship speeds in different eras

⇒

- Human Mobility Index (HMI)
Solution: The Voyage of Homo-œconomicus

Combine data on

- Infantry movement
- Geographical conditions
- Ship speeds in different eras

⇒

- Human Mobility Index (HMI)
- Human Mobility Index with Seafaring pre-1500CE (HMISea)
Solution: The Voyage of Homo-œconomicus

Combine data on

- Infantry movement
- Geographical conditions
- Ship speeds in different eras

⇒

- Human Mobility Index (HMI)
- Human Mobility Index with Seafaring pre-1500CE (HMISea)
- Human Mobility Index with Seafaring pre-steam engine (HMIOcean)
How?

Construct cost of movement using data from

Historical data on seafaring in Old World (pre-1500CE) (Casson, 1951, 1989)

Historical data on seafaring (pre-steam engine) (García-Herrera, Konnen, Wheeler, Prieto, Jones, and Koek, 2005)
How?

Construct cost of movement using data from

- U.S. Army data on human mobility (Hayes, 1994)
How?

Construct cost of movement using data from

- U.S. Army data on human mobility (Hayes, 1994)

\[ \text{Travel Time on Land} = f(slope, temp, rel. hum., terrain, sky) \]
How?

Construct cost of movement using data from

- U.S. Army data on human mobility (Hayes, 1994)

\[
\text{Travel Time on Land} = f(slope, temp, rel. \ hum., terrain, sky)
\]

- Historical data on seafaring in Old World (pre-1500CE) (Casson, 1951, 1989)
How?

Construct cost of movement using data from

- U.S. Army data on human mobility (Hayes, 1994)

\[ \text{Travel Time on Land} = f(\text{slope, temp, rel. hum., terrain, sky}) \]

- Historical data on seafaring in Old World (pre-1500CE) (Casson, 1951, 1989)

- Historical data on seafaring (pre-steam engine) (García-Herrera, Können, Wheeler, Prieto, Jones, and Koek, 2005)
Optimal Paths

- With cost surface find minimum travel time between locations
Optimal Paths

- With cost surface find minimum travel time between locations
- Concerns
Optimal Paths

- With cost surface find minimum travel time between locations
- Concerns
  - Raster size \((12,837; 43,345)\), i.e. 556,419,765 cells!
Optimal Paths

- With cost surface find minimum travel time between locations
- Concerns
  - Raster size (12,837; 43,345), i.e. 556,419,765 cells!
  - Potentially huge graph
Optimal Paths

- With cost surface find minimum travel time between locations
- Concerns
  - Raster size (12,837; 43,345), i.e. 556,419,765 cells!
  - Potentially huge graph
  - 200+ countries
Homo-œconomicus meets GIS

Start with traditional approach...
Homo-œconomicus meets GIS

Start with traditional approach...

- Go to GIS course/workshop (1 or 2 weeks)
Homo-œconomicus meets GIS

Start with traditional approach...

- Go to GIS course/workshop (1 or 2 weeks)
  - Intro to ArcGIS (point-and-click)
Homo-œconomicus meets GIS

Start with traditional approach…

- Go to GIS course/workshop (1 or 2 weeks)
  - Intro to ArcGIS (point-and-click)
  - Get data from TIGER or ArcGIS
Homo-œconomicus meets GIS

Start with traditional approach...

- Go to GIS course/workshop (1 or 2 weeks)
  - Intro to ArcGIS (point-and-click)
  - Get data from TIGER or ArcGIS
  - Make maps using shapefiles in ArcGIS
Homo-œconomicus meets GIS

Start with traditional approach...

- Go to GIS course/workshop (1 or 2 weeks)
  - Intro to ArcGIS (point-and-click)
  - Get data from TIGER or ArcGIS
  - Make maps using shapefiles in ArcGIS
  - Some spatial stats (compute Moran I & II)
Trade as Percentage of GDP in the World in year 2000

GDP level per Capita

Trade as share of GDP
Homo-œconomicus meets GIS

Seems easy & straightforward... Thesis’s gonna be ready in 2 weeks!
What do you mean the data is not in a shapefile?

- Search for data to construct cost surface
What do you mean the data is not in a shapefile?

- Search for data to construct cost surface
  - Where do I search for data?
What do you mean the data is not in a shapefile?

- Search for data to construct cost surface
  - Where do I search for data?
  - No shapefiles...what?!
What do you mean the data is not in a shapefile?

- Search for data to construct cost surface
  - Where do I search for data?
  - No shapefiles...what?!
  - What is a raster?!
What do you mean the data is not in a shapefile?

- Search for data to construct cost surface
  - Where do I search for data?
  - No shapefiles...what?!
  - What is a raster?!

- Raster = Matrix
What do you mean the data is not in a shapefile?

- Search for data to construct cost surface
  - Where do I search for data?
  - No shapefiles...what?!
  - What is a raster?!

- Raster = Matrix

- Shapefile $\Rightarrow$ points, lines, polygons, etc.
Point-and-click and more in ArcGIS

To solve my problem I need more tools...

- Raster Calculator
Point-and-click and more in ArcGIS

Cost Distance Function

```
[Image of Cost Distance Function]
```
ArcGIS headache I

Read manual and figured out how to do it...at least in theory...
ArcGIS headache I

- Read manual and figured out how to do it...at least in theory...
- Problem...does not work on my laptop...why?!
ArcGIS headache I

- Read manual and figured out how to do it...at least in theory...
- Problem...does not work on my laptop...why?!
  - I have a Mac...need virtual machine
ArcGIS headache I

- Read manual and figured out how to do it...at least in theory...
- Problem...does not work on my laptop...why?!
  - I have a Mac...need virtual machine
  - I have student edition
ArcGIS headache I

- Read manual and figured out how to do it...at least in theory...
- Problem...does not work on my laptop...why?!
  - I have a Mac...need virtual machine
  - I have student edition
  - Need Spatial Analyst License = $2,500!!!
ArcGIS headache I

- Read manual and figured out how to do it...at least in theory...
- Problem...does not work on my laptop...why?!
  - I have a Mac...need virtual machine
  - I have student edition
- Need Spatial Analyst License = $2,500!!!
- Ok...work in university computers...GIS lab
ArcGIS headache I

- Read manual and figured out how to do it...at least in theory...
- Problem...does not work on my laptop...why?!
  - I have a Mac...need virtual machine
  - I have student edition
  - Need Spatial Analyst License = $2,500!!!
- Ok...work in university computers...GIS lab
  - Get access to 1 computer...
Produce Raster in ArcGIS

- Construct HMI data

![Image of ArcGIS interface with Raster Calculator tool selected]
HMI & HMISea

The Dark Ages

Raster Calculator

The Voyage of Homo-œconomicus

October 13, 2016 19 / 56
ArcGIS headache II

- Construct Optimal Routes and Times
ArcGIS headache II

- Construct Optimal Routes and Times

- Problem...it takes more than 1 day per source!
ArcGIS headache II

- Construct Optimal Routes and Times

- Problem...it takes more than 1 day per source!

⇒ > 1 year to compute data!!!
Solution...Parallelize!

- Find multiple computers and **repeat exact same** process for different sources
Solution...Parallelize!

- Find multiple computers and **repeat exact same** process for different sources

- Need scripting...still slow in ArcGIS!
OMG...Now what????!!!
Main take aways

ArcGIS and point-and-click

- Advantages
Main take aways

ArcGIS and point-and-click

- Advantages
  - Easy to install and start using for basic things

- Disadvantages
  - Expensive
  - Computations are slow...
  - Support is slow
  - Difficult replication
  - Not scripting friendly
  - Only Windows compatible
Main take aways

ArcGIS and point-and-click

- Advantages
  - Easy to install and start using for basic things
  - Good for mapping

- Disadvantages
  - Expensive
  - Computations are slow...very slow
  - Support is slow
  - Difficult replication
  - Not scripting friendly
  - Only Windows compatible
Main take aways

ArcGIS and point-and-click

- Advantages
  - Easy to install and start using for basic things
  - Good for mapping
- Disadvantages
Main take aways

ArcGIS and point-and-click

- **Advantages**
  - Easy to install and start using for basic things
  - Good for mapping

- **Disadvantages**
  - Expensive
Main take aways

ArcGIS and point-and-click

- Advantages
  - Easy to install and start using for basic things
  - Good for mapping

- Disadvantages
  - Expensive
  - Computations are slow...very slow
Main take aways

ArcGIS and point-and-click

- Advantages
  - Easy to install and start using for basic things
  - Good for mapping

- Disadvantages
  - Expensive
  - Computations are slow...very slow
  - Support is slow
Main take aways

ArcGIS and point-and-click

- Advantages
  - Easy to install and start using for basic things
  - Good for mapping

- Disadvantages
  - Expensive
  - Computations are slow...very slow
  - Support is slow
  - Difficult replication
Main take aways

ArcGIS and point-and-click

- **Advantages**
  - Easy to install and start using for basic things
  - Good for mapping

- **Disadvantages**
  - Expensive
  - Computations are slow...very slow
  - Support is slow
  - Difficult replication
  - Not scripting friendly
Main take aways

ArcGIS and point-and-click

- **Advantages**
  - Easy to install and start using for basic things
  - Good for mapping

- **Disadvantages**
  - Expensive
  - Computations are slow...very slow
  - Support is slow
  - Difficult replication
  - Not scripting friendly
  - Only Windows compatible
Although many disadvantages to ArcGIS, it is costly to change
Although many disadvantages to ArcGIS, it is costly to change

Luckily for me when I moved to SMU I got a push
Although many disadvantages to ArcGIS, it is costly to change.

Luckily for me when I moved to SMU I got a push.

- Couldn’t get ArcGIS installed.
Although many disadvantages to ArcGIS, it is costly to change.

Luckily for me when I moved to SMU I got a push:

- Couldn’t get ArcGIS installed
- My computer is a Mac...and you know, once you go Mac, you don’t go back!
Although many disadvantages to ArcGIS, it is costly to change.

Luckily for me when I moved to SMU I got a push:

- Couldn’t get ArcGIS installed
- My computer is a Mac...and you know, once you go Mac, you don’t go back!
- Virtual Machine had problems with ArcGIS
Although many disadvantages to ArcGIS, it is costly to change

Luckily for me when I moved to SMU I got a push

- Couldn’t get ArcGIS installed
- My computer is a Mac...and you know, once you go Mac, you don’t go back!
- Virtual Machine had problems with ArcGIS

⇒ Time to try something different
Can I overcome disadvantages?

**Free** Point-and-click solutions

- GRASS (Geographic Resources Analysis Support System):
Can I overcome disadvantages?

**Free** Point-and-click solutions

- GRASS (Geographic Resources Analysis Support System):
  - Originally developed by the U.S. Army Construction Engineering Research Laboratories
Can I overcome disadvantages?

**Free** Point-and-click solutions

- **GRASS (Geographic Resources Analysis Support System):**
  - Originally developed by the U.S. Army Construction Engineering Research Laboratories
  - Now part of OSGEO (Open Source Geospatial Foundation)
SAGA (System for Automated Geoscientific Analyses):
SAGA (System for Automated Geoscientific Analyses):

- Originally developed by Dept. of Physical Geography, Göttingen and Hamburg
What I use...QGIS!

QGIS (Quantum GIS):

- Part of OSGEO (Open Source Geospatial Foundation)
- Can integrate GRASS & SAGA!
- Large user group, plug-ins, programmable

Omer Özak (Here’s How I Do GIS)
What I use... QGIS!

- QGIS (Quantum GIS):
  - Part of OSGEO (Open Source Geospatial Foundation)
What I use... QGIS!

- QGIS (Quantum GIS):
  - Part of OSGEO (Open Source Geospatial Foundation)
  - Can integrate GRASS & SAGA!
What I use...QGIS!

- QGIS (Quantum GIS):
  - Part of OSGEO (Open Source Geospatial Foundation)
  - Can integrate GRASS & SAGA!
  - Large user group, plug-ins, programmable
What I use...QGIS!

- QGIS (Quantum GIS):
  - Part of OSGEO (Open Source Geospatial Foundation)
  - Can integrate GRASS & SAGA!
  - Large user group, plug-ins, programmable
  - Ömer’s Basic QGIS Tutorial
What this solves

• Free as in

- beer (no cost)
- freedom (users have the freedom to run, copy, distribute, study, change and improve the software)

Open source =

⇒

Large community/support

Change code

Propose plug-ins, features, etc.
What this solves

- Free as in
  - beer (no cost)
What this solves

- Free as in
  - beer (no cost)
  - freedom (users have the freedom to run, copy, distribute, study, change and improve the software)
What this solves

- Free as in
  - beer (no cost)
  - freedom (users have the freedom to run, copy, distribute, study, change and improve the software)
- Works many OS: Android, Linux, OSX, Windows
What this solves

- Free as in
  - beer (no cost)
  - freedom (users have the freedom to run, copy, distribute, study, change and improve the software)

- Works many OS: Android, Linux, OSX, Windows

- Open source ➞
What this solves

- Free as in
  - beer (no cost)
  - freedom (users have the freedom to run, copy, distribute, study, change and improve the software)
- Works many OS: Android, Linux, OSX, Windows
- Open source ⇒
  - Large community/support
What this solves

- Free as in
  - beer (no cost)
  - freedom (users have the freedom to run, copy, distribute, study, change and improve the software)
- Works many OS: Android, Linux, OSX, Windows
- Open source $\Rightarrow$
  - Large community/support
  - Change code
What this solves

- Free as in
  - beer (no cost)
  - freedom (users have the freedom to run, copy, distribute, study, change and improve the software)

- Works many OS: Android, Linux, OSX, Windows

- Open source ➞
  - Large community/support
  - Change code
  - Propose plug-ins, features, etc.
How I Learned to Stop Clicking and Love the Code

Finally started using

- Python
- IPython
- Jupyter
General Purpose Programming Language
General Purpose Programming Language
- Open source
General Purpose Programming Language

- Open source
- Easy to learn and code
General Purpose Programming Language

- Open source
- Easy to learn and code
- Clean code
General Purpose Programming Language

- Open source
- Easy to learn and code
- Clean code
- Powerful
General Purpose Programming Language

- Open source
- Easy to learn and code
- Clean code
- Powerful
- Versatile

Lots of packages to get things done
Large community (Stackoverflow, Github, Bitbucket)
Used in ArcGIS, QGIS, Google, Yahoo!, LANL, Netflix, National Weather Service, NASA, etc.

"Omer Ozak (Here's How I Do GIS)
General Purpose Programming Language

- Open source
- Easy to learn and code
- Clean code
- Powerful
- Versatile
- Lots of packages to get things done
General Purpose Programming Language

- Open source
- Easy to learn and code
- Clean code
- Powerful
- Versatile
- Lots of packages to get things done
- Large community (Stackoverflow, Github, Bitbucket)
General Purpose Programming Language

- Open source
- Easy to learn and code
- Clean code
- Powerful
- Versatile
- Lots of packages to get things done
- Large community (Stackoverflow, Github, Bitbucket)
- Used in ArcGIS, QGIS, Google, Yahoo!, LANL, Netflix, National Weather Service, NASA, etc.
IPython: Interactive Computing

- Interactive Python
Interactive Python

- GUI/Kernel for Python/Jupyter
IP[y]: IPython
Interactive Computing

- Interactive Python
  - GUI/Kernel for Python/Jupyter
  - Easy to use
IPython: Interactive Python

- Interactive Python
  - GUI/Kernel for Python/Jupyter
  - Easy to use
  - Visually appealing
IPython: Interactive Computing

- Interactive Python
  - GUI/Kernel for Python/Jupyter
  - Easy to use
  - Visually appealing
  - Interactive data visualization
Interactive Python

- GUI/Kernel for Python/Jupyter
- Easy to use
- Visually appealing
- Interactive data visualization
- Debugging
**IP[y]: IPython**

**Interactive Computing**

- Interactive Python
  - GUI/Kernel for Python/Jupyter
  - Easy to use
  - Visually appealing
  - Interactive data visualization
  - Debugging
  - Tab completion
**Interactive Python**

- GUI/Kernel for Python/Jupyter
- Easy to use
- Visually appealing
- Interactive data visualization
- Debugging
- Tab completion
- High performance tools for parallel computing
Interactive Python

- GUI/Kernel for Python/Jupyter
- Easy to use
- Visually appealing
- Interactive data visualization
- Debugging
- Tab completion
- High performance tools for parallel computing
- Open source
Terminal/Command Line

```python
Python 2.7.3 (default, Jul 10 2012, 18:48:25)
Type "copyright", "credits" or "license" for more information.

IPython 0.13.1 -- An enhanced Interactive Python.
?                      -> Introduction and overview of IPython's features.
%quickref              -> Quick reference.
help                    -> Python's own help system.
object?                 -> Details about 'object', use 'object??' for extra details.
In [1]: import numpy as np
In [2]: N = 3000
In [3]: a = np.random.rand(N,N)
In [4]: b = np.random.rand(N,N)
In [5]: np.dot
   np.dot
   np.double
In [5]: np.dot(a, b)
Out[5]:
array([[  65.45679169,   64.96918252,  -120.2955101 , ...,   46.52919413,
   1.62384273, -117.27453877],
   [  103.8332094 ,  -63.19741333,   25.63850851, ...,   10.43730591,
   -98.22728902,   -9.16795735],
   [ -36.45895885,   44.32183553,  -17.58969917, ...,  -125.12907291,
   -70.58209094,  -32.85797429],
   ...,
   [ -42.46168724,   36.45522834,   28.8765628 , ...,   39.40943867,
   -16.43199427,   -63.81943649],
   [ -64.46717927,   28.06738004,  -32.09826395, ...,   -42.127647 ,
   -116.20291834,  -32.02266909],
   [  56.79843374,   23.68037948,   52.24793136, ...,  -35.53881726,
   -21.19194341,  -151.71414646]])
In [6]: %timeit np.dot(a,b)
1 loops, best of 3: 2.17 s per loop
In [7]: np.show()
Display all 551 possibilities? (y or n) np.show np.convolve np.iscomplex np.ravel np.ROUND np.BUFFER_SIZE np.copy np.iscomplexobj
```
IPython: Interactive Computing

- QtConsole

```python
In [1]: imshow(imread('baboon.png'))
Out[1]: <matplotlib.image.AxesImage at 0x401658>
```
Notebook (Web Application)

```python
ax.set_title("Walking back to my\nfront door at night:"

ax.set_xlim(0, 1)
ax.set_ylim(0, 1.5)

# modify all the axes elements in-place
XKCDify(ax, expand_axes=True)
```

Out[7]: `<matplotlib.axes.AxesSubplot at 0x2fef210>`
Open source, interactive data science and scientific computing across over 40 programming languages!
Open source, interactive data science and scientific computing across over 40 programming languages!

- Spin off from IPython
Open source, interactive data science and scientific computing across over 40 programming languages!

- Spin off from IPython
- Based on Notebook
Open source, interactive data science and scientific computing across over 40 programming languages!

- Spin off from IPython
- Based on Notebook
- Kernels for IPython, R, Julia, Scala, etc.
Open source, interactive data science and scientific computing across over 40 programming languages!

- Spin off from IPython
- Based on Notebook
- Kernels for IPython, R, Julia, Scala, etc.
- Big data ready...Spark
**Open source, interactive data science and scientific computing across over 40 programming languages!**

- Spin off from IPython
- Based on Notebook
- Kernels for IPython, R, Julia, Scala, etc.
- Big data ready...Spark

- Share notebooks (Web, \LaTeX)
Open source, interactive data science and scientific computing across over 40 programming languages!

- Spin off from IPython
- Based on Notebook
- Kernels for IPython, R, Julia, Scala, etc.
- Big data ready...Spark

- Share notebooks (Web, \LaTeX)
- Use multiple language simultaneously (e.g. Python & R)
If they’re so good, why aren’t we all using it?

- Lack of knowledge
If they’re so good, why aren’t we all using it?

- Lack of knowledge
- Change is costly
If they’re so good, why aren’t we all using it?

- Lack of knowledge
- Change is costly
- Disadvantages of General & Open Source Software
If they’re so good, why aren’t we all using it?

- Lack of knowledge
- Change is costly
- Disadvantages of General & Open Source Software
  - Installation may not be straightforward
If they’re so good, why aren’t we all using it?

- Lack of knowledge
- Change is costly
- Disadvantages of General & Open Source Software
  - Installation may not be straightforward
  - Learning curve
If they’re so good, why aren’t we all using it?

- Lack of knowledge
- Change is costly
- Disadvantages of General & Open Source Software
  - Installation may not be straightforward
  - Learning curve
  - Not always GIS ready
If they’re so good, why aren’t we all using it?

- Lack of knowledge
- Change is costly
- Disadvantages of General & Open Source Software
  - Installation may not be straightforward
  - Learning curve
  - Not always GIS ready

⇒ Scary!
But...

- Becoming easier & more generalized
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
  - Continuum distributes Anaconda & Conda
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
  - Continuum distributes Anaconda & Conda
  - Enthought distributes Canopy
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
  - Continuum distributes Anaconda & Conda
  - Enthought distributes Canopy
  - Homebrew on OSX
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
  - Continuum distributes Anaconda & Conda
  - Enthought distributes Canopy
  - Homebrew on OSX
  - Scientific Linux
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
  - Continuum distributes Anaconda & Conda
  - Enthought distributes Canopy
  - Homebrew on OSX
  - Scientific Linux
- Lots of community support
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
  - Continuum distributes Anaconda & Conda
  - Enthought distributes Canopy
  - Homebrew on OSX
  - Scientific Linux

- Lots of community support
  - Fast
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
  - Continuum distributes Anaconda & Conda
  - Enthought distributes Canopy
  - Homebrew on OSX
  - Scientific Linux

- Lots of community support
  - Fast
  - Solutions
But...

- Becoming easier & more generalized
  - OSGeo distributes installers
  - Continuum distributes Anaconda & Conda
  - Enthought distributes Canopy
  - Homebrew on OSX
  - Scientific Linux

- Lots of community support
  - Fast
  - Solutions
  - Suggestions
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
- GeoStats (My project for Statistics - Private)
- HMI (My project for HMI - Private)
- RasterStats
- Fiona
- Shapely
- rTree
- OGR/GDAL
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
- GeoStats (My project for Statistics - Private)
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
- GeoStats (My project for Statistics - Private)
- HMI (My project for HMI - Private)
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
- GeoStats (My project for Statistics - Private)
- HMI (My project for HMI - Private)
- RasterStats
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
- GeoStats (My project for Statistics - Private)
- HMI (My project for HMI - Private)
- RasterStats
- Fiona
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
- GeoStats (My project for Statistics - Private)
- HMI (My project for HMI - Private)
- RasterStats
- Fiona
- Shapely
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
- GeoStats (My project for Statistics - Private)
- HMI (My project for HMI - Private)
- RasterStats
- Fiona
- Shapely
- rTree
My Main Workhorses

Main GIS Packages I use:

- GeoPandas (Major Project for Geometries)
- GeoRasters (My project for Rasters - Public)
- GeoStats (My project for Statistics - Private)
- HMI (My project for HMI - Private)
- RasterStats
- Fiona
- Shapely
- rTree
- OGR/GDAL
My Workflow

- Download data
My Workflow

- Download data
- Check it in QGIS
My Workflow

- Download data
- Check it in QGIS
- Write & Test Code using IPython QtConsole or Notebook
My Workflow

- Download data
- Check it in QGIS
- Write & Test Code using IPython QtConsole or Notebook
- Deploy to Server if needed
My Workflow

- Download data
- Check it in QGIS
- Write & Test Code using IPython QtConsole or Notebook
- Deploy to Server if needed
- Get results and analyze in Stata (soon to be replaced by R o StatsModels)
My Workflow

- Download data
- Check it in QGIS
- Write & Test Code using IPython QtConsole or Notebook
- Deploy to Server if needed
- Get results and analyze in Stata (soon to be replaced by R or StatsModels)
- Write paper in \LaTeX
Examples

- Ömer’s intro to GIS with IPython
Examples

- Ömer’s intro to GIS with IPython
- CSI
Examples

- Ömer’s intro to GIS with IPython
- CSI
- Google Location History
Example: Compute Zonal Stats I

#!/usr/bin/env python
# coding: utf-8

================================================================================
Author: Ömer Özak, 2014 (ozak at smu.edu)
Website: http://omerozak.com
GitHub: https://github.com/ozak/
================================================================================
Python Script Template

from __future__ import division
import sys, os, time
# Math, data
import numpy as np
import pandas as pd
pd.set_option(‘display.width’, 140)

# GIS packages
#from osgeo import ogr, osr, gdal, gdalnumeric
#from gdalconst import *
Example: Compute Zonal Stats II

```python
# from PIL import Image, ImageDraw
# from pyGDsandbox.dataIO import df2dbf, dbf2df
# from rasterstats import zonal_stats
# import shutil, glob
# import pysal as ps
# from pysal.contrib import shapely_ext
# import shapely
from shapely.geometry import Polygon, Point
# from shapely.wkt import loads, dumps
# from shapely.ops import cascaded_union
import pyproj
# import georasters as gr
import hmi
# import fiona
import geopandas as gp
# import geopandas.tools as gpt
# from geopandas.tools import sjoin
from geopy.distance import great_circle
import geostats

# Various other packages
```
Example: Compute Zonal Stats III

```python
import isounidecode  # Decode and encode text

# Directory
try:
    %cd Islam
except:
    path = os.path.abspath(__file__)
    dir_path = os.path.dirname(path)
    os.chdir(dir_path)
%bookmark Islam

# Buffer size in meters
buf = 50000

# Set paths
path='../../data/'
if not os.path.exists(path):
    os.mkdir(path)
pathout='../../data/GIS/Cities/'
if not os.path.exists(pathout):
    os.mkdir(pathout)
```
Example: Compute Zonal Stats IV

```python
# Geographical characteristics of each Ethnicity in a Buffer of buffer kms
cities = pd.read_stata(path+'AllCities.dta')
cities['geometry'] = cities.apply(lambda x: Point(x.lon, x.lat), axis=1)
cities.drop('aaanameofcity', axis=1, inplace=True)
cities = gp.GeoDataFrame(cities, crs=geostats.wgs84)
cities['city'] = cities.city.apply(lambda x: isounidecode.unidecode(x[:x.find(' (')]))
cities.to_file(pathout+'AllCities.shp')
cities = cities.to_crs(geostats.cea)
cities.to_file(pathout+'AllCitiesCyl.shp')
cities.geometry = cities.geometry.buffer(buf)

# Create geostats object and compute statistics
Stats = geostats.geostats(cities)
Stats.geostats()

# Export data
Stats.df.to_csv(path+'/AllCities'+str(int(buf/1000))+'.csv', index=False)
```
Example: HMI Distances with MP I

```python
# coding: utf-8
# #Import packages

Author: Ömer Özak, 2014 (ozak at smu.edu)
Website: http://omerozak.com
GitHub: https://github.com/ozak/

Program to create HMI data for Islam Project using MultiProcessing to accelerate computations
Michalopoulos and Özak (2016)

from __future__ import division
# Parallel
from IPython.parallel import Client
## Setup the clients, direct views, and balanced views
```
Example: HMI Distances with MP II

```python
import sys, os, time
# Math, data
import numpy as np
import pandas as pd
pd.set_option('display.width', 140)
from scipy.interpolate import griddata
import scipy.interpolate as interpolate
from scipy.spatial import cKDTree as KDTree
# GIS packages
from osgeo import ogr, osr, gdal, gdalnumeric
from gdalconst import *
from PIL import Image, ImageDraw
# from pyGDsandbox.dataIO import df2dbf, dbf2df
from rasterstats import zonal_stats
import shutil, glob
import pysal as ps
import shapely
from shapely.geometry import Polygon, Point
from shapely.wkt import loads, dumps
```
Example: HMI Distances with MP III

```python
from pysal.contrib import shapely_ext
from shapely.ops import cascaded_union
import pyproj
import geopandas as gp
import georasters as gr
from geopy.distance import great_circle
import fiona
import hmi
import isounidecode  # Decode and encode text
import datetime
import matplotlib.pyplot as plt

# Set directories
try:
    %cd Islam
    %matplotlib inline
except:
    path = os.path.abspath(__file__)
    dir_path = os.path.dirname(path)
    os.chdir(dir_path)

# Set paths
```
Example: HMI Distances with MP IV

```python
path='../../data/'
if not os.path.exists(path):
    os.mkdir(path)

pathout='../../data/HMI/'
if not os.path.exists(pathout):
    os.mkdir(pathout)

# Import Cities shapefile using GeoPandas
cities = pd.read_stata(path+'/AllCities.dta')
cities['LAT']=cities['lat']
cities['LON']=cities['lon']
cities['city']=cities.city.apply(lambda x: isounidecode.unidecode(x[(x.find(' (')==-1)*len(x)+(x.find(' ('))>0]))
cities = cities[['LAT','LON','code','city','ID']]  

# Define a function that calls HMI, HMISea, HMIOcean with start point only one row in the cities dataframe and returns all distances to that country's centroid
#start_points=pd.DataFrame([cities.loc[0,:]], columns=cities.columns.values)

def computeHMI(row):
    """Compute HMI for starting at row and ending in all of cities""
    A = hmi.HMI(pd.DataFrame([row[1]], columns=cities.columns.values), cities, lat='LAT', lon='LON', projected=False)
    A.HMIdistance(export_shape=True, path=pathout+str(row[1]['code'])+str(row[0]))
    return A.hmidist
```
Example: HMI Distances with MP V

```python
def computeHMISea(row):
    """Compute HMI for starting at row and ending in all of cities""
    A = hmi.HMISea(pd.DataFrame([row[1]], columns=cities.columns.values), cities, lat='LAT', lon='LON', projected=False)
    A.HMIdistance(export_shape=True, path=pathout+str(row[1][’code’])+str(row[0]))
    return A.bmidist

def computeHMIOcean(row):
    """Compute HMI for starting at row and ending in all of cities""
    A = hmi.HMIOcean(pd.DataFrame([row[1]], columns=cities.columns.values), cities, lat='LAT', lon='LON', projected=False)
    A.HMIdistance(export_shape=True, path=pathout+str(row[1][’code’])+str(row[0]))
    return A.bmidist

# Now compute all distances in parallel, merge GeoPandas Frames
dfhmi = view.map_async(computeHMI, cities.iterrows())
dfhmisea = view.map_async(computeHMISea, cities.iterrows())
dfhmioccean = view.map_async(computeHMIOcean, cities.iterrows())
# Get results
dfhmi = dfhmi.get()
dfhmisea = dfhmisea.get()
dfhmioccean = dfhmioccean.get()
```
Example: HMI Distances with MP VI

# Concatenate the results
dfhmi = pd.concat(dfhmi)
dfhmisea = pd.concat(dfhmisea)
dfhmiocean = pd.concat(dfhmiocean)
# Convert to GeoPandas again
dfhmi = gp.GeoDataFrame(dfhmi, crs=hmi.cea)
dfhmisea = gp.GeoDataFrame(dfhmisea, crs=hmi.cea)
dfhmiocean = gp.GeoDataFrame(dfhmiocean, crs=hmi.cea)
# Export Shape files
dfhmi.to_file(path+'HMI10.shp')
dfhmisea.to_file(path+'HMISea10.shp')
dfhmiocean.to_file(path+'HMIOcean10.shp')

# Merge Distances and export them
cols=dfhmi.columns
cols=cols.drop('geometry')
dfout = dfhmi[cols].copy()
dfout = dfout.merge(dfhmisea[['city_1','city_2','HMISea10dist','HMISea10Iso']], how='right')
dfout = dfout.merge(dfhmiocean[['city_1','city_2','HMIOcean10dist','HMIOcean10Iso']], how='right')
dfout.sort(['city_1','city_2'], inplace=True)
dfout.to_stata('../../data/HMI10dists.dta', write_index=False)
Example: Extension of Original Project

New Project...similar to original one but using city data

- 4669 cities
Example: Extension of Original Project

New Project...similar to original one but using city data

- 4669 cities
- 9 versions of HMI*10
Example: Extension of Original Project

New Project...similar to original one but using city data

- 4669 cities
- 9 versions of HMI*10
- Using 149 cores on server
Example: Extension of Original Project

New Project...similar to original one but using city data

- 4669 cities
- 9 versions of HMI*10
- Using 149 cores on server
- Less than 1 day for full results (data, networks, MST, etc.)
Advantages

- High speed
Advantages

- High speed
- Reproducible research
Advantages

- High speed
- Reproducible research
- Portable across computers
Advantages

- High speed
- Reproducible research
- Portable across computers
- Shareable across users
Advantages

- High speed
- Reproducible research
- Portable across computers
- Shareable across users
- Easy parallelization
Advantages

- High speed
- Reproducible research
- Portable across computers
- Shareable across users
- Easy parallelization
- Adaptability to users needs
Advantages

- High speed
- Reproducible research
- Portable across computers
- Shareable across users
- Easy parallelization
- Adaptability to users needs
- Access to large set of tools (GIS and non-GIS)
Advantages

- High speed
- Reproducible research
- Portable across computers
- Shareable across users
- Easy parallelization
- Adaptability to users needs
- Access to large set of tools (GIS and non-GIS)
- Support
When should you use?

- Point-and-click
When should you use?

- Point-and-click
  - Fast or simple tasks/analyses
When should you use?

- Point-and-click
  - Fast or simple tasks/analyses
  - Non-repeating tasks/analyses
When should you use?

- **Point-and-click**
  - Fast or simple tasks/analyses
  - Non-repeating tasks/analyses
- **Code**
When should you use?

- Point-and-click
  - Fast or simple tasks/analyses
  - Non-repeating tasks/analyses
- Code
  - Repetitive tasks/analyses
When should you use?

- **Point-and-click**
  - Fast or simple tasks/analyses
  - Non-repeating tasks/analyses

- **Code**
  - Repetitive tasks/analyses
  - Too specific or complex tasks/analyses
Standing on the Shoulders of Giants

- No need to start from scratch
Standing on the Shoulders of Giants

- No need to start from scratch
- Use other people's code
Standing on the Shoulders of Giants

- No need to start from scratch
- Use other people’s code
- Reuse your own code
Standing on the Shoulders of Giants

- No need to start from scratch
- Use other people’s code
- Reuse your own code
- Write your own libraries/packages
Standing on the Shoulders of Giants

- No need to start from scratch
- Use other people’s code
- Reuse your own code
- Write your own libraries/packages
- Share…share…share!
Standing on the Shoulders of Giants

- No need to start from scratch
- Use other people’s code
- Reuse your own code
- Write your own libraries/packages
- Share…share…share!
- Ömer’s Github 🤖
Where are we going?

- Simplification

- Easier to install
- Easier to work with
- Power
  - More packages
  - More speed
  - More parallelization
- Interactivity/Interaction
  - Cooler graphs, widgets, dashboards
  - With other users
  - With other languages
Where are we going?

- Simplification
  - Easier to install

-Simplification
  - Easier to install

-Power
  - More packages
  - More speed
  - More parallelization

-Interactivity/Interaction
  - Cooler graphs, widgets, dashboards
  - With other users
  - With other languages
Where are we going?

- Simplification
  - Easier to install
  - Easier to work with
Where are we going?

- Simplification
  - Easier to install
  - Easier to work with
- Power
Where are we going?

- **Simplification**
  - Easier to install
  - Easier to work with

- **Power**
  - More packages
Where are we going?

- Simplification
  - Easier to install
  - Easier to work with

- Power
  - More packages
  - More speed
Where are we going?

- **Simplification**
  - Easier to install
  - Easier to work with
- **Power**
  - More packages
  - More speed
  - More parallelization
Where are we going?

- **Simplification**
  - Easier to install
  - Easier to work with

- **Power**
  - More packages
  - More speed
  - More parallelization

- **Interactivity/Interaction**
Where are we going?

- **Simplification**
  - Easier to install
  - Easier to work with

- **Power**
  - More packages
  - More speed
  - More parallelization

- **Interactivity/Interaction**
  - Cooler graphs, widgets, dashboards
Where are we going?

- **Simplification**
  - Easier to install
  - Easier to work with

- **Power**
  - More packages
  - More speed
  - More parallelization

- **Interactivity/Interaction**
  - Cooler graphs, widgets, dashboards
  - With other users
Where are we going?

- **Simplification**
  - Easier to install
  - Easier to work with

- **Power**
  - More packages
  - More speed
  - More parallelization

- **Interactivity/Interaction**
  - Cooler graphs, widgets, dashboards
  - With other users
  - with other languages
The Voyage of Homo-œconomicus into GIS

Ömer Özak

Dept. Economics, SMU

Here’s How I Do GIS

October 13, 2016