2-Day Course – Spatial Modeling with Geostatistics

Prof. Michael J. Pyrcz, Ph.D., P.Eng.
Associate Professor

Hildebrand Department of Petroleum & Geosystems Engineering
University of Texas at Austin

Bureau of Economic Geology, Jackson School of Geosciences
University of Texas at Austin

“In two days, what a geoscientists needs to know about geostatistics, and workflows to get you started with applying geostatistics to impact your work.”

Spatial Modeling with Geostatistics

Prerequisites

Lecture outline . . .

- Who am I?
- Class Objectives
- Class Strategy
- Essential Pre-work
  – Installation, set up

Prerequisites

Introduction

Probability Theory

Representative Sampling

Spatial Data Analysis

Spatial Estimation

Stochastic Simulation

Uncertainty Management

Machine Learning

Class Description. . .

Who am I?

- started at UT Austin, Petroleum and Geosystems Engineering Fall, 2018
- over 17 years of experience in consulting, teaching and industrial R&D in statistical modeling, reservoir modeling and uncertainty characterization.
- associate editor with Computers and Geosciences, nominee to International Assoc. of Mathematical Geosciences committee.
- member of scientific committee for Geostatistical Congress 2016.
- author of the textbook “Geostatistical Reservoir Modeling” and > 40 peer reviewed publications, patents etc.
- I think Geostatistical knowledge empowers Geoscientists

Class Objectives

Teach theory and practical methods for geostatistics.

Communicate:

- the benefits and uses of geostatistics,
- the common spatial and uncertainty modeling workflows,
- how to better integrate their domain knowledge into the geostatistical model.

Provide knowledge and resources to start geoscientists building their own workflows.

Initial experience with workflow construction with open source

Class Strategy

A combination of lecture, demonstration and hands-on

Lectures

- Provide the fundamental theory with a focus on practice

Demonstrations

- Illustrate the use of geostatistics with open source to solve practical problems
- I will use Python / GSLIB, Excel, and R workflows that are available to students

Hands-on

- Experiential learning with R / R Studio and “gstat” package

Hands-On

We will conduct hands-on in:

1. Paper-based
2. Excel
3. R / Rstudio
4. Python / GSLIB

Hands-On in R / RStudio

We will conduct hands-on in R, because it is easiest for getting started, and has robust packages for geostatistics and data analytics

It will require students to complete the following before the class.

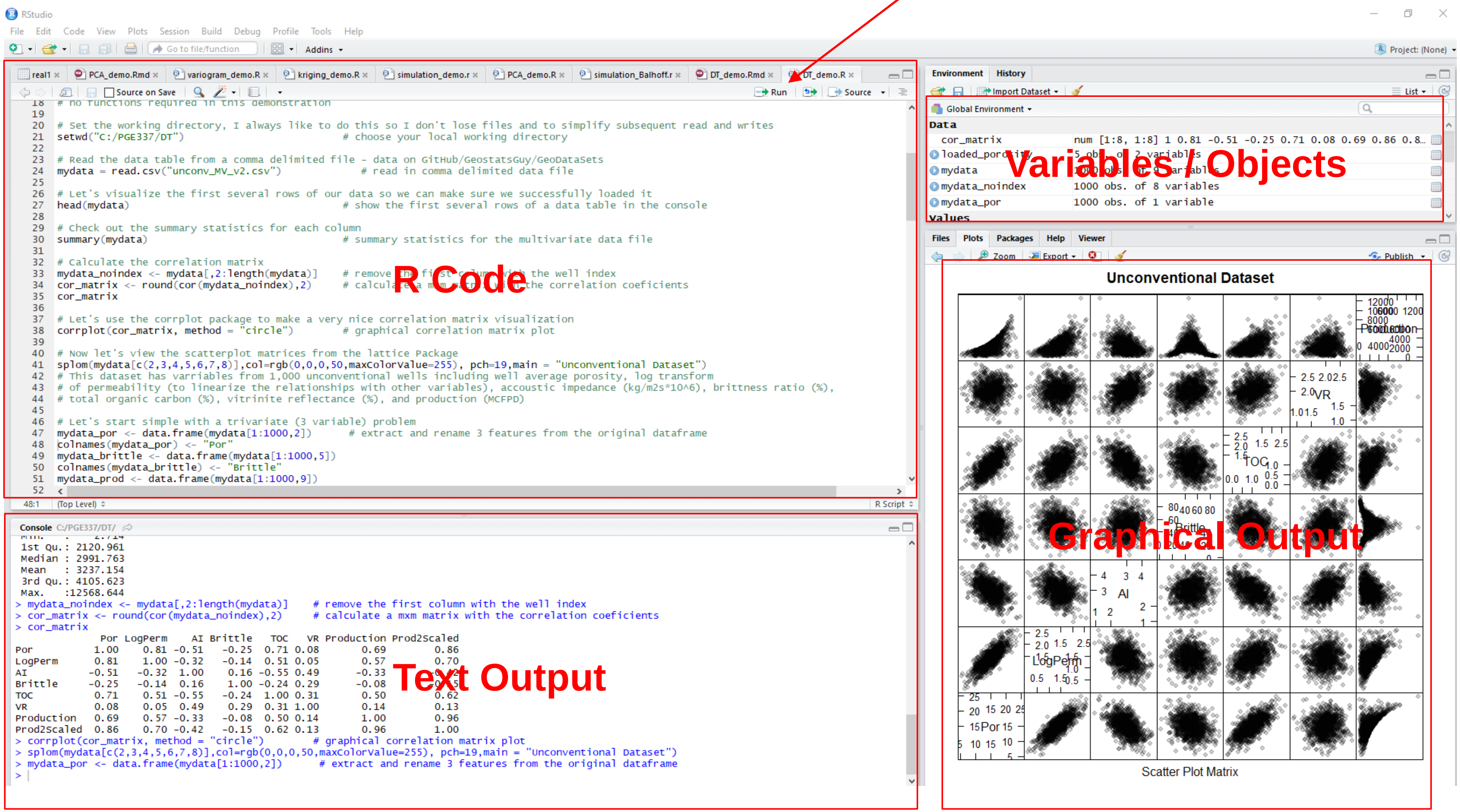
1. Install R from one of the mirror sites (e.g. http://cran.wustl.edu/)
2. Install R Studio from https://www.rstudio.com/products/rstudio/download/. The free version is fine.

Texas Center for Geostatistics
The University of Texas at Austin

Hands-On in R / RStudio

3. Open R Studio

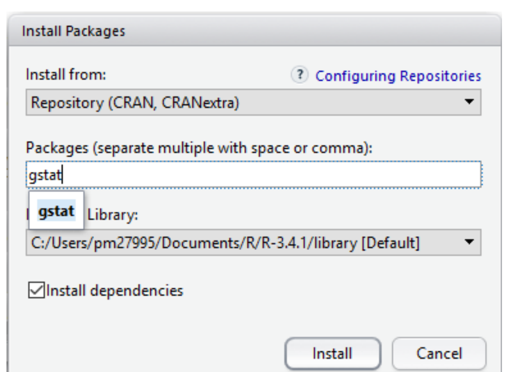
Step Through R Code



Hands-On in R / RStudio

4. Install the following packages
   - gstat geostatistics package by Edzer Pebesma
   - sp adds spatial to DataFrames
   - plyr manipulating data
   - ggplot2 plotting
   - fields plotting regular grid models
   - lattice matrix scatter plots
   - corrplot correlation plots
   - tree decision trees

To install packages go to Tools/ Install Packages…



Enter name of package and select install.

Ignore R Version warnings.

Hands-Onin R / RStudio

5. Open provided R code, kriging_demo.R
https://github.com/GeostatsGuy/geostatsr/blob/master/kriging_demo.html

6. Change the working directory

```
# Set the working directory, I always like to do this so I don't lose files and to simplify subsequent read and writes
setwd("C:/PGE337")
```

- Change from C:/PGE337 to a folder of your choice on your computer

7. Download these datasets from GitHub and put them in your working folder.
   - **2D_MV_200Wells.csv**
     - **https://github.com/GeostatsGuy/GeoDataSets/blob/master/2D_MV_200wells.csv**
   - **unconv_MV_v2.csv**
     - **https://github.com/GeostatsGuy/GeoDataSets/blob/master/unconv_MV_v2.csv**
   - **unconv_MV_v3.csv**
     - **https://github.com/GeostatsGuy/GeoDataSets/blob/master/unconv_MV_v3.csv**

Hands-On in R / RStudio

8. Back in R Studio window, place the cursor at the top of the code and step through the code with the “run” button indicated on slide 6.
9. Watch the text and graphical output. Check text output for errors. Warnings are fine.
   - A couple of the numerical methods may take 15 to 30 seconds to complete so be patient.
   - Resist the temptation to “machine gun” the run button as this may cause a crash of R studio.
10. If you get to the end of the code file then you should be set up and good-to-go for the hands-on sections.

Hands-On in Python / GSLIB

It is possible to just run GSLIB executables:

- GSLIB includes a set of executables for
  - Data manipulation / transforms
  - Geostatistical calculations and modeling
  - Visualization in Post Script

I will demonstrate that.

One could set up .bat files with entire workflows.

Others have used scripting approaches to loop the executables and summarize the results.

Hands-On in Python / GSLIB

GeostatsPy

- I have reimplement most of the plotting in MatPlotLib and wrapped the numerical methods
  - The wrapping simply writes out a parameter file, calls the executable, reads the results into Python DataFrames and ndarrays
  - Note if you are unfamiliar with Python DataFrames and ndarrays I have put together a basis tutorial with subsurface datasets.

https://github.com/GeostatsGuy/PythonNumericalDemos/blob/master/PythonDataBasics_DataFrame.ipynb

https://github.com/GeostatsGuy/PythonNumericalDemos/blob/master/PythonDataBasics_ndarrays.ipynb

Hands-On in Python / GSLIB

Jupyter / Markdown

- The workflows are in Jupyter Notebooks with Markdown documentation.

Spatial Declustering in Python for Engineers and Geoscientists

Michael Pyrcz, Associate Professor, University of Texas at Austin

Contacts: Twitter/@GeostatsGuy | GitHub/GeostatsGuy | www.michaelpyrcz.com | GoogleScholar | Book

This is a tutorial for / demonstration of **spatial declustering in Python with simple wrappers and reimplementations of GSLIB: Geostatistical Library methods** (Deutsch and Journel, 1997). Almost every spatial dataset is based on biased sampling. This includes clustering (increased density of samples) over specific ranges of values. For example, more samples in an area of high feature values. Spatial declustering is a process of assigning data weights based on local data density. The cell-based declustering approach (Deutsch and Journel, 1997; Pyrcz and Deutsch, 2014; Pyrcz and Deutsch, 2003, paper is available here: http://gaa.org.au/pdf/DeclusterDebias-CCG.pdf) is based on the use of a mesh over the area of interest. Each datum's weight is inverse to the number of data in each cell. Cell offsets of applied to smooth out influence of mesh origin. Multiple cell sizes are applied and typically the cell size that minimizes the declustered distribution mean is applied for preferential sampling in the high-valued locations (the maximizing cell size is applied if the data is preferential sampled in the low-valued locations). If there is a nominal data spacing with local clusters, then this spacing is the best cell size.

This exercise demonstrates the cell-based declustering approach in Python with wrappers and reimplimentation of GSLIB methods. The steps include:

1. generate a 2D sequential Guassian simulation using a wrapper of GSLIB's sgsim method
2. apply regular sampling to the 2D realization
3. preferentially removing samples in the low-valued locations
4. calculate cell-based declustering weights
5. visualize the location map of the declustering weights and the original exhaustive, sample and the new declustered distributions.

To accomplish this I have provide wrappers or reimplementation in Python for the following GSLIB methods:

1. sgsim - sequantial Gaussian simulation limited to 2D and unconditional
2. hist - histograms plots reimplemented with GSLIB parameters using python methods
3. locmap - location maps reimplemented with GSLIB parameters using python methods
4. pixelplt - pixel plots reimplemented with GSLIB parameters using python methods
5. locpix - my modification of GSLIB to superimpose a location map on a pixel plot reimplemented with GSLIB parameters using Python methods
6. affine - affine correction adjust the mean and standard deviation of a feature reimplemented with GSLIB parameters using Python methods

These methods are all in the functions declared upfront. To run this demo all one has to do is download and place in your working directory the following executables from the GSLIB/bin directory:

1. sgsim.exe
2. declus.exe
3. nscore.exe (not currently used in demo, but wrapper is included)

The GSLIB source and executables are available at http://www.statios.com/Quick/gslib.html. For the reference on using GSLIB check out the User Guide, GSLIB: Geostatistical Software Library and User's Guide by Clayton V. Deutsch and Andre G. Journel.

I did this to allow people to use these GSLIB functions that are extremely robust in Python. Also this should be a bridge to allow so many familar with GSLIB to work in Python as a kept the parameterization and displays consistent with GSLIB. The wrappers are simple functions declared below that write the parameter files, run the GSLIB executable in the working directory and load and visualize the output in Python. This will be included on GitHub for anyone to try it out https://github.com/GeostatsGuy/.

I used this tutorial in my Introduction to Geostatistics undergraduate class (PGE337 at UT Austin) as part of a first introduction to geostatistics and Python for the engineering undergraduate students. It is assumed that students have no previous Python, geostatistics nor machine learning experience; therefore, all steps of the code and workflow are explored and described. This tutorial is augmented with course notes in my class. The Python code and markdown was developed and tested in Jupyter.

Markdown

Make a 2D spatial model

The following are the basic parameters for the demonstration. This includes the number of cells in the 2D regular grid, the cell size (step) and the x and y min and max along with the color scheme.

Then we make a single realization of a Gausian distributed feature over the specified 2D grid and then apply affine correction to ensure we have a reasonable mean and spread for our feature's distribution, assumed to be Porosity (e.g. no negative values) while retaining the Gaussian distribution. Any transform could be applied at this point. We are keeping this workflow simple. *This is our truth model that we will sample.*

The parameters of *GSLIB_sgsim_2d_uncond* are (nreal,nx,ny,hsiz,seed,hrange1,hrange2,azi,output_file). nreal is the number of realizations, nx and ny are the number of cells in x and y, hsiz is the cell siz, seed is the random number seed, hrange and hrange2 are the variogram ranges in major and minor directions respectively, azi is the azimuth of the primary direction of continuity (0 is aligned with Y axis) and output_file is a GEO_DAS file with the simulated realization. The ouput is the 2D numpy array of the simulation along with the name of the property.

Python Code

In [4]:

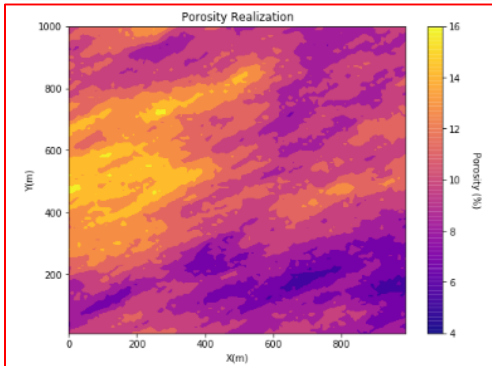
```python
nx = 100; ny = 100; cell_size = 10                          # grid number of cells and cell size
xmin = 0.0; ymin = 0.0;                                      # grid origin
xmax = xmin + nx * cell_size; ymax = ymin + ny * cell_size   # calculate the extent of model
seed = 74073                                                 # random number seed  for stochastic simulation
range_max = 1800; range_min = 500; azimuth = 65              # Porosity variogram ranges and azimuth
mean = 10.0; stdev = 2.0                                     # Porosity mean and standard deviation
#cmap = plt.cm.RdYlBu
vmin = 4; vmax = 16; cmap = plt.cm.plasma                    # color min and max and using the plasma color map

# calculate a stochastic realization with standard normal distribution
sim,value = GSLIB_sgsim_2d_uncond(1,nx,ny,cell_size,seed,range_max,range_min,azimuth,"simulation")
sim = affine(sim,mean,stdev)                                 # correct the distribution to a target mean and standard dev
```

Let's look at our 2D model with a pixel plot. The parameters below for the pixelplt function are (array,xmin,xmax,ymin,ymax,step,vmin,vmax,title,xlabel,ylabel,vlabel,cmap). Array is a 2D numpy array with the realization (the output from the *GSLIB_sgsim_2d_uncond*), the xmin, xmax, ymin, ymax are the extents of the model and step is the cell size, vmin, vmax are the min and max of the feature, title, xlabel, ylabel and vlabel are the plot labels and cmap is the color map.

Output

In [5]:

```python
pixelplt(sim,xmin,xmax,ymin,ymax,cell_size,vmin,vmax,"Porosity Realization","X(m)","Y(m)","Porosity (%)",cmap)
```



Hands-On in Python / GSLIB

Jupyter / Markdown

More Comments:

- Very powerful for prototyping, developing workflows.
- May load a variety of Kernels. I also do my R in Jupyter sometimes.
- May use interactive widgets to make exercises.
- May publish online so anyone can just run the sheet (without any need to set up their environment).

Hands-On

- If you are not able to complete this set up, then you can pair up with another student to work together on the hands on sections.

- If we run into significant issues we won’t be hung up as we have flexibility to switch up.

What did you just learn?

Lecture outline . . .

- Who am I?
- Class Objectives
- Class Strategy
- Prerequisites

Prerequisites

Introduction

Probability Theory

Representative Sampling

Spatial Data Analysis

Spatial Estimation

Stochastic Simulation

Uncertainty Management

Machine Learning