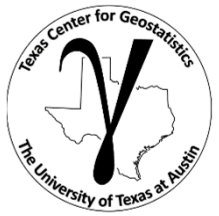
Machine Learning
Introduction



Lecture outline . . .

- **Who am I?**
- **Motivation / Goals**
- **Class Description / Objectives**
- **The Plan**
- **Resources**

Instructor: Michael Pyrcz, the University of Texas at Austin

Who Am I?

Spring 2018 Class of Introduction to Geostatistics

Oil and Gas University, Florence, Italy

Anadarko, Midland, TX

Michael Pyrcz

1. **Pyrcz**: is pronounced “perch”

2. **I’m New**: new to UT PGE, started August, 2017. Everything is new!

3. **I have practical experience**: over 17 years of experience in consulting, teaching and industrial R&D in statistical modeling, reservoir modeling and uncertainty characterization.

Who Am I?

Fall 2018 Class of Introduction to Geostatistics

Fall 2017 PGE 383

Michael Pyrcz

4. Flexible: got ideas, feedback to improve the learning opportunities. Let's work together to reach our learning objective.

5. Available: I have an open door policy. Drop by my office. Drop a line anytime.

6. An Engineer, but: My B.Sc. was Mining Engineering, my M.Sc. started as Geotechnical Engineering (then skipped to Ph.D.) and my Ph.D. was in Quantitative Geology. I spent 13 years in Earth Science R&D working with geological and geophysical reservoir modeling. I speak geo.

Who Am I?

AAPG SEPM Panel Discussion on Modeling

CPGE Webinar on Big Data

Michael Pyrcz

8. Active in Outreach, Social Media and Professional Organizations

• associate editor with Computers and Geosciences, editorial board of Mathematical Geosciences for the International Association of Mathematical Geosciences

• program chair for SPE Data Analytics Technical Section
• associate editor with Computers and Geosciences
• author of the textbook “Geostatistical Reservoir Modeling”
• board member for Mathematical Geosciences
• GeostatsGuy on Twitter, GitHub, GeostatsGuy Lectures on YouTube

I’m committed to supporting / partnering for development opportunities of working professionals

Introductions

Short Introductions:

Name

Role

Expectations from this Class

What Will You Learn?

The Goal

1. Concepts and new ideas

- Impact your work
- Confidence to start or continue

2. Tools to impact your daily work in

- machine learning for inference and prediction
- some basic workflows in Python
- what can be done

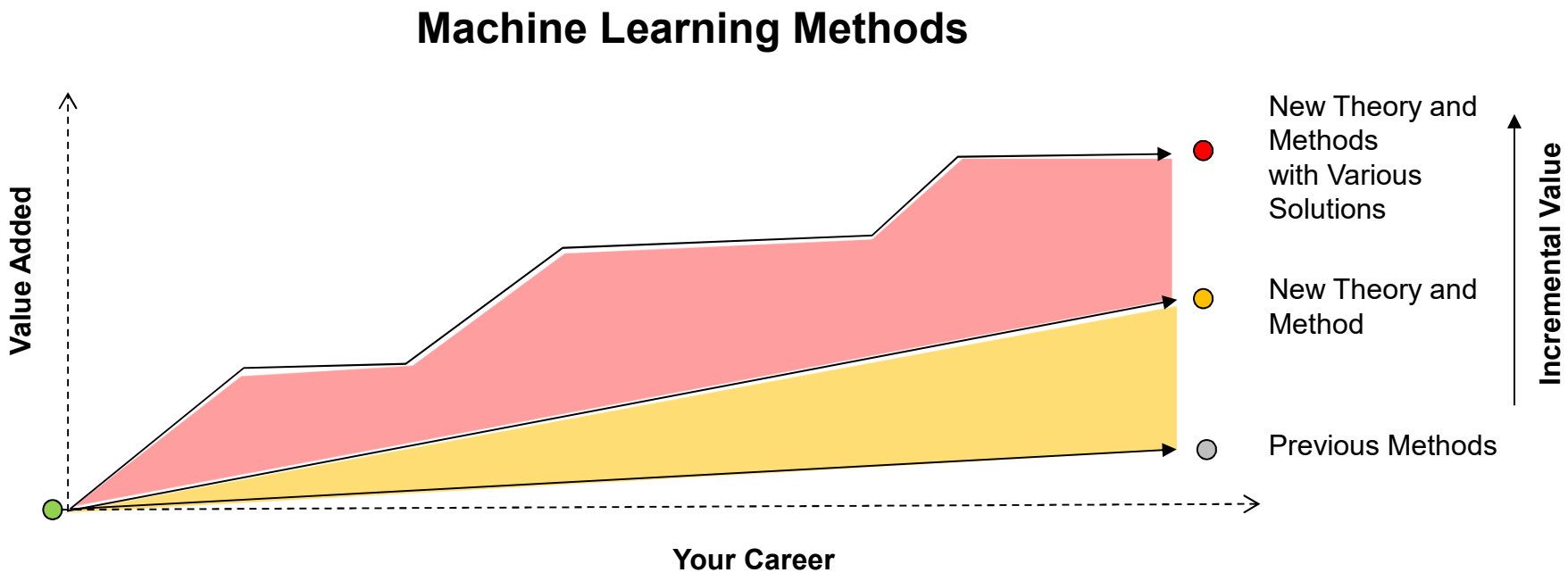
3. If you are already an expert

- Tools and ideas for teaching these concepts

What Will You Learn?

This morning is an investment in learning

- Build operational capability
- Provide incremental value



What Will You Learn?

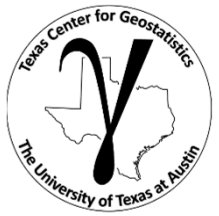
Reaching our Goal

Fast Overview of Theory and Methods:

- Cover the fundamental theory, modeling steps and common modeling workflows
- Rapid-fire coverage of a range of inference and predition methods

Of course, full workflow development would require time to investigate the problem and available data.

What Will You Learn?



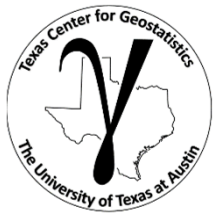
There is Much More!

- the building blocks can be reimplemented and expanded to address various other problems, opportunities.

There is much more that we can cover.

- Statistical Inference
- Representative Statistics
- Uncertainty Sources
- Trend Modeling
- Model Optimization
- Model Selection
- Feature Engineering
- Physics in Machine Learning
- Fair Spatial Model Testing
- Spatial Context
- Ordination
- Value of Information
- Recurrent Neural Networks
- Convolutional Neural Networks

How Will You Learn All of That?



Here's the Plan:

1. **Interactive** lectures / discussion to cover the theory

2. Live **demonstrations** in **Python with well-documented Jupyter notebooks**

We will **adjust for success**:

- Let me know if you are lost, stuck, something is not working or you aren't learning!
- We are learning-driven not schedule-driven
  - e.g. we could cover less and spend more time
  - e.g. we could use less Python and more Excel

Feedback welcome as we proceed to tailor and maximize your value from this time together.

Why Python?

Python:

- Is very powerful, the most resources and assistance
- Packages allow us to put together workflows with limited old-fashioned 'coding'
- Leverage the world's brilliance

'Certainly there's a phenomenon around open source. You know free software will be a vibrant area. 'There will be a lot of nest things that get done there.'

- Bill Gates

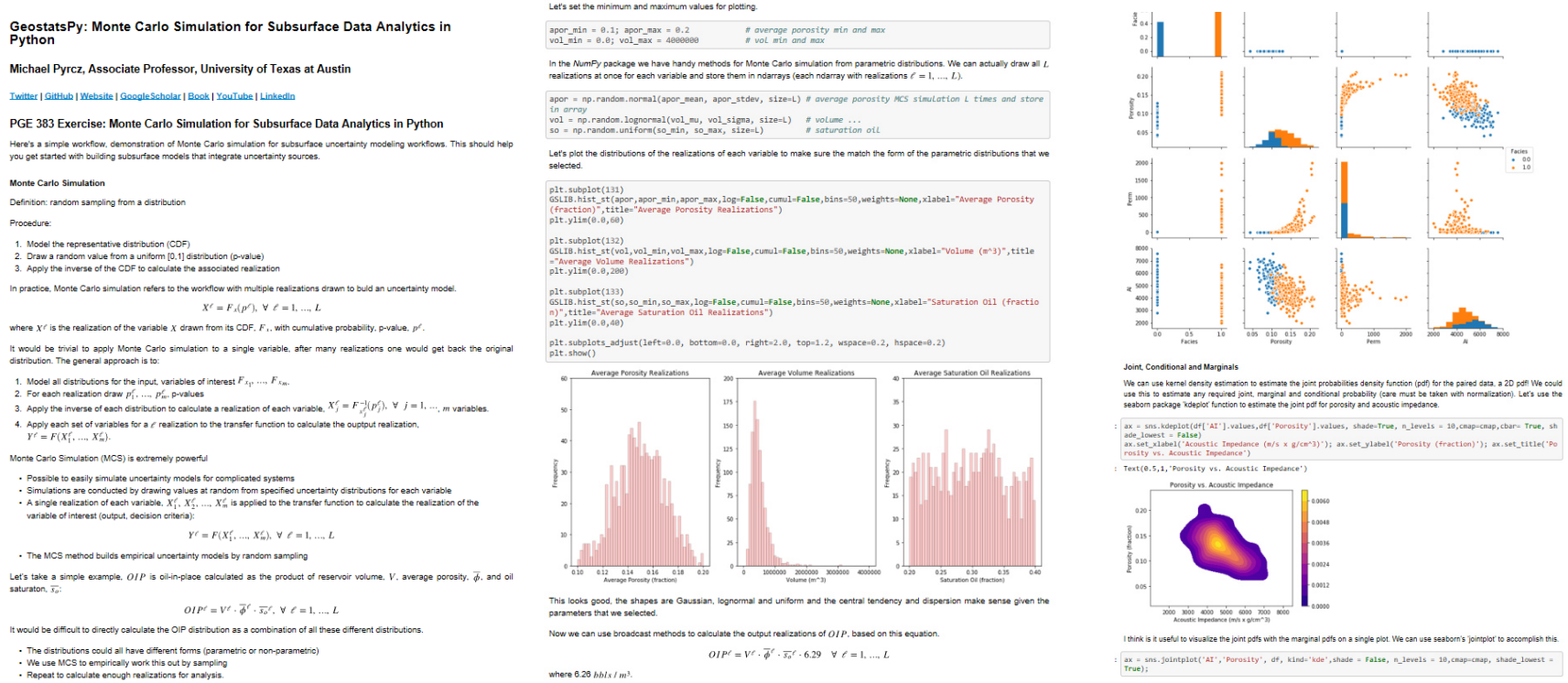
'More than 20 years with C++ and FORTRAN, but with Python I code less, but get more done.'

- Michael Pyrcz

Jupyter Notebooks?

Python with Jupyter Notebooks

- Workflows that integrate blocks of code, documentation, results



GeostatsPy: Monte Carlo Simulation for Subsurface Data Analytics in Python

Michael Pyrcz, Associate Professor, University of Texas at Austin

Twitter | GitHub | Website | GoogleScholar | Book | YouTube | LinkedIn

PGE 383 Exercise: Monte Carlo Simulation for Subsurface Data Analytics in Python

Here's a simple workflow, demonstration of Monte Carlo simulation for subsurface uncertainty modeling workflows. This should help you get started with building subsurface models that integrate uncertainty sources.

Monte Carlo Simulation

Definition: random sampling from a distribution

Procedure:

1. Model the representative distribution (CDF)
2. Draw a random value from a uniform [0,1] distribution (p-value)
3. Apply the inverse of the CDF to calculate the associated realization

In practice, Monte Carlo simulation refers to the workflow with multiple realizations drawn to build an uncertainty model.

$$X^\ell = F_X(p^\ell), \; \forall \; \ell = 1, ..., L$$

where X^ℓ is the realization of the variable X drawn from its CDF, F_x, with cumulative probability, p-value, p^ℓ.

It would be trivial to apply Monte Carlo simulation to a single variable, after many realizations one would get back the original distribution. The general approach is to:

1. Model all distributions for the input, variables of interest $F_{X_1}, ..., F_{X_m}$.
2. For each realization draw $p_1^\ell, ..., p_m^\ell$, p-values
3. Apply the inverse of each distribution to calculate a realization of each variable, $X_j^\ell = F_{X_j}^{-1}(p_j^\ell), \; \forall \; j = 1, \cdots, m$ variables.
4. Apply each set of variables for a ℓ realization to the transfer function to calculate the output realization, $Y^\ell = F(X_1^\ell, ..., X_m^\ell)$.

Monte Carlo Simulation (MCS) is extremely powerful

- Possible to easily simulate uncertainty models for complicated systems
- Simulations are conducted by drawing values at random from specified uncertainty distributions for each variable
- A single realization of each variable, $X_1^\ell, X_2^\ell, ..., X_m^\ell$ is applied to the transfer function to calculate the realization of the variable of interest (output, decision criteria):

$$Y^\ell = F(X_1^\ell, ..., X_m^\ell), \; \forall \; \ell = 1, ..., L$$

- The MCS method builds empirical uncertainty models by random sampling

Let's take a simple example, OIP is oil-in-place calculated as the product of reservoir volume, V, average porosity, $\overline{\phi}$, and oil saturaton, $\overline{s_o}$.

$$OIP^\ell = V^\ell \cdot \overline{\phi}^\ell \cdot \overline{s_o}^\ell, \; \forall \; \ell = 1, ..., L$$

It would be difficult to directly calculate the OIP distribution as a combination of all these different distributions.

- The distributions could all have different forms (parametric or non-parametric)
- We use MCS to empirically work this out by sampling
- Repeat to calculate enough realizations for analysis.

Let's set the minimum and maximum values for plotting.

```
apor_min = 0.1; apor_max = 0.2          # average porosity min and max
vol_min = 0.0; vol_max = 4000000        # vol min and max
```

In the *NumPy* package we have handy methods for Monte Carlo simulation from parametric distributions. We can actually draw all L realizations at once for each variable and store them in ndarrays (each ndarray with realizations $\ell = 1, ..., L$).

```
apor = np.random.normal(apor_mean, apor_stdev, size=L) # average porosity MCS simulation L times and store in array
vol = np.random.lognormal(vol_mu, vol_sigma, size=L)   # volume ...
so = np.random.uniform(so_min, so_max, size=L)         # saturation oil
```

Let's plot the distributions of the realizations of each variable to make sure the match the form of the parametric distributions that we selected.

```
plt.subplot(131)
GSLIB.hist_st(apor,apor_min,apor_max,log=False,cumul=False,bins=50,weights=None,xlabel="Average Porosity (fraction)",title="Average Porosity Realizations")
plt.ylim(0.0,60)

plt.subplot(132)
GSLIB.hist_st(vol,vol_min,vol_max,log=False,cumul=False,bins=50,weights=None,xlabel="Volume (m^3)",title="Average Volume Realizations")
plt.ylim(0.0,200)

plt.subplot(133)
GSLIB.hist_st(so,so_min,so_max,log=False,cumul=False,bins=50,weights=None,xlabel="Saturation Oil (fraction)",title="Average Saturation Oil Realizations")
plt.ylim(0.0,40)

plt.subplots_adjust(left=0.0, bottom=0.0, right=2.0, top=1.2, wspace=0.2, hspace=0.2)
plt.show()
```


This looks good, the shapes are Gaussian, lognormal and uniform and the central tendency and dispersion make sense given the parameters that we selected.

Now we can use broadcast methods to calculate the output realizations of OIP, based on this equation.

$$OIP^\ell = V^\ell \cdot \overline{\phi}^\ell \cdot \overline{s_o}^\ell \cdot 6.29 \quad \forall \; \ell = 1, ..., L$$

where 6.26 $bbls \, / \, m^3$.

Joint, Conditional and Marginals

We can use kernel density estimation to estimate the joint probabilities density function (pdf) for the paired data, a 2D pdf! We could use this to estimate any required joint, marginal and conditional probability (care must be taken with normalization). Let's use the seaborn package 'kdeplot' function to estimate the joint pdf for porosity and acoustic impedance.

```
ax = sns.kdeplot(df['AI'].values,df['Porosity'].values, shade=True, n_levels = 10,cmap=cmap,cbar= True, shade_lowest = False)
ax.set_xlabel('Acoustic Impedance (m/s x g/cm^3)'); ax.set_ylabel('Porosity (fraction)'); ax.set_title('Porosity vs. Acoustic Impedance')
```

```
Text(0.5,1,'Porosity vs. Acoustic Impedance')
```


I think is it useful to visualize the joint pdfs with the marginal pdfs on a single plot. We can use seaborn's 'jointplot' to accomplish this.

```
ax = sns.jointplot('AI','Porosity', df, kind='kde',shade = False, n_levels = 10,cmap=cmap, shade_lowest = True);
```

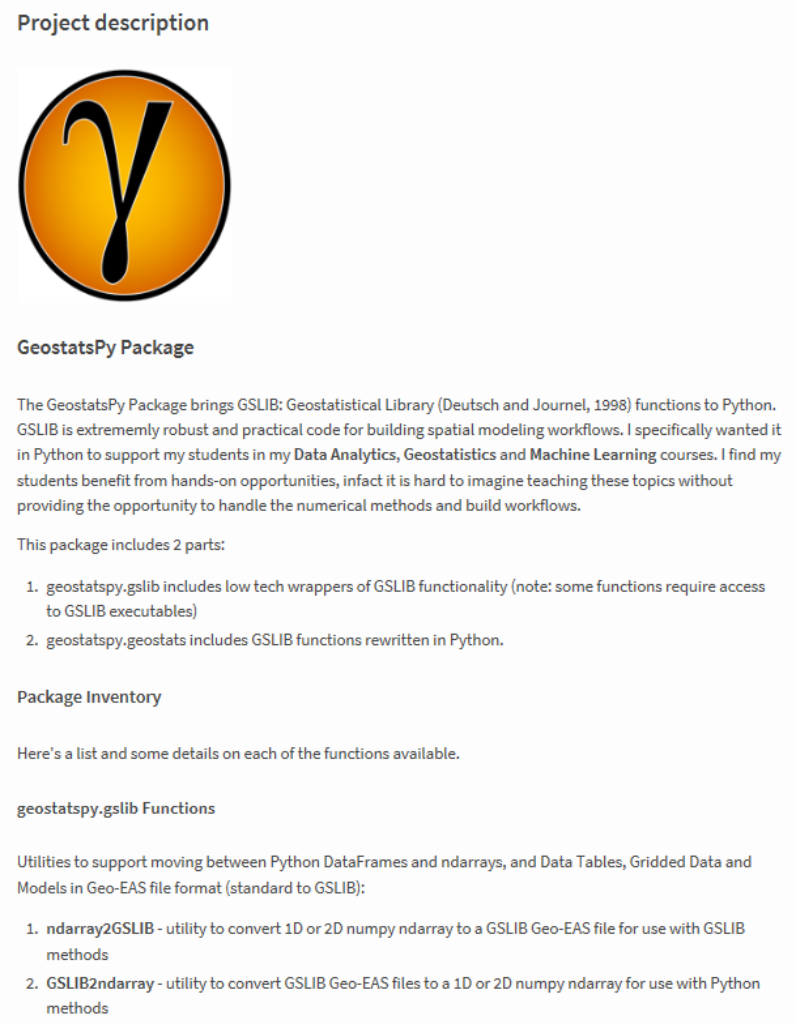
- Work with a variety of kernels (Python, R, C, javascript, etc.)
- Make professional workflows with Markdown docs
- Use containers and run online (e.g. Docker)

GeostatsPy?

GeostatsPy

- Set of Functions in Python
  - GeostatsPy is a set of Python functions for most of the required workflow steps
  - Much is reimplemented in Python.
  - Package written by myself, we will tailor, augment to support training.
  - I welcome feedback.
  - Open Source anyone can use it
  - Free for any use
  - Download it from PyPi with:

'pip install geostatspy'



Project description

GeostatsPy Package

The GeostatsPy Package brings GSLIB: Geostatistical Library (Deutsch and Journel, 1998) functions to Python. GSLIB is extrememly robust and practical code for building spatial modeling workflows. I specifically wanted it in Python to support my students in my **Data Analytics**, **Geostatistics** and **Machine Learning** courses. I find my students benefit from hands-on opportunities, infact it is hard to imagine teaching these topics without providing the opportunity to handle the numerical methods and build workflows.

This package includes 2 parts:

1. geostatspy.gslib includes low tech wrappers of GSLIB functionality (note: some functions require access to GSLIB executables)
2. geostatspy.geostats includes GSLIB functions rewritten in Python.

Package Inventory

Here's a list and some details on each of the functions available.

geostatspy.gslib Functions

Utilities to support moving between Python DataFrames and ndarrays, and Data Tables, Gridded Data and Models in Geo-EAS file format (standard to GSLIB):

1. **ndarray2GSLIB** - utility to convert 1D or 2D numpy ndarray to a GSLIB Geo-EAS file for use with GSLIB methods
2. **GSLIB2ndarray** - utility to convert GSLIB Geo-EAS files to a 1D or 2D numpy ndarray for use with Python methods

More on Coding

More on Software / Coding:

- This is not a coding / software workshop.
- I can't teach Python in 1 day.
- We will demonstrate well-documented workflows in Python.
- We will focus on the steps, inputs and outputs.
- Don't be concerned if you don't completely understand the code.

The Plan - *We will use well-documented workflows in Python to demonstrate concepts!*

- *I have partnered to teach data science to industry professionals.*
- *Learn more at: https://daytum.org/*

Reasons to Learn Coding

Transparency – *no compiler accepts hand waiving!* Coding forces your logic to be uncovered for any other scientist or engineer to review.

Reproducibility – *run it, get an answer, hand it over, run it, get the same answer.* This is a main principle of the scientific method.

Quantification – *programs need numbers.* Feed the program and discover new ways to look at the world.

Open-source – *leverage a world of brilliance.* Check out packages, snippets and be amazed with what great minds have freely shared.

Break Down Barriers – *don't throw it over the fence*. Sit at the table with the developers and share more of your subject matter expertise for a better product.

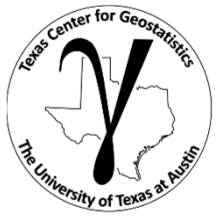
Deployment – *share it with others and multiply the impact*. Performance metrics or altruism, your good work benefits many others.

Efficiency – *minimize the boring parts of the job*. Build a suite of scripts for automation of common tasks and spend more time doing science and engineering!

Always Time to Do it Again! – *how many times did you only do it once?* It probably takes 2-4 times as long to script and automate a workflow. Usually worth it.

Be Like Us – *it will change you*. Users feel limited, programmers truly harness the power of their applications and hardware.

Reasons to Learn Coding



Caveats for the previous reasons for coding:

1. Any type of coding, scripting, workflow automation matched to your working environment is great. We don't all need to be C++ experts.
2. I respect the experience component of geoscience and engineering expertise. This is beyond coding and is essential to workflow logic development, best use of data etc.
3. Some expert judgement will remain subjective and not completely reproducible. I'm not advocating for the geoscientist or engineer being replaced by a computer.

Deliverables

1. **New Ideas, Learnings** to impact your work – increased digitization of your work.

2. **Example workflows** in Python to build out new workflows.

3. **Course notes and references** support continued learning.

4. **Recorded Lectures** on YouTube to support review.

Machine Learning
Introduction

Lecture outline . . .

- **Who am I?**
- **Motivation / Goals**
- **Class Description / Objectives**
- **The Plan**
- **Resources**

Instructor: Michael Pyrcz, the University of Texas at Austin