USF MSDS Spring Reading List 2023

This document contains the reading list for the 2023-2024 cohort. Note that there is a strong correlation between students who struggle and those that fail to take the spring reading list seriously. We spend very little time reviewing the material during the first part of the program; failure to come prepared will limit your own ability to learn.

This document is organized into three sections: (1) Linear Algebra, (2) Probability and Statistics and (3) Computer Language and Tools. It is recommended that you spend time working through your weakest area -- not your strongest. Use your time wisely.

Linear Algebra

Below is a superset of good linear algebra textbooks for review. In the linear algebra self-paced videos and exam, the instructor will draw from a combination of these books with emphasis from (1) and (5):

1. *Matrix Analysis and Applied Linear Algebra* by Carl D. Meyer
2. *Introduction to Linear Algebra* by Gilbert Strang
3. *Applied Linear Algebra and Matrix Analysis* by Thomas S. Shores
4. *Numerical Linear Algebra* by Lloyd N. Trefethen and David Bau
5. *Elementary Linear Algebra* by Anton, 11th edition, Wiley

For the time being, you can rely on your linear algebra book from college, as well as the free linear algebra book by Jim Hefferson at:

http://joshua.smcvt.edu/linearalgebra/book.pdf

In your initial review, focus on the following topics: *vectors, matrices, and associated operations, solving linear equations, determinants, vector spaces, eigenvalues and eigenvectors, and linear transformations.* Time permitting you should engage in any problems associated with computation and implementation of all these topics.

If you are looking for an online learning venue, consider taking the OCW Scholar course in linear algebra at the Massachusetts Institute of Technology at:

http://ocw.mit.edu/courses/mathematics/18-06sc-linear-algebra-fall-2011/

For further reading, Ian GoodFellow's chapter on Linear Algebra is terse, but focused on preparation for deep learning:

https://www.deeplearningbook.org/contents/linear_algebra.html

If you want a video connecting linear algebra and python data science problems,

enjoy:

https://pyvideo.org/europython-2018/from-linear-algebra-to-machine-learning.html

Probability and Statistics

In the probability and statistics boot camp (MSDS 504), the instructor will use a combination of the following books:

1. Ghahramani, Saeed. *Fundamentals of Probability, with Stochastic Processes*, 4th edition.
2. Freund, John. *Mathematical Statistics*, 8th edition.
3. Ross, Sheldon. *Simulation*. 5th edition.
4. Moore, David. *The Basic Practice of Statistics*, 8th edition.
5. Hogg, Robert V. and Elliot A. Tanis. *Probability and Statistical Inference*, 9th edition.

Selected excerpts from the above-mentioned books will be available on the Canvas site for MSDS 504, and are also available on the Canvas site for admitted students.

As you review, focus on the following learning outcomes:

- Understanding the definitions of probability mass functions, probability density functions, cumulative distributions functions, and moments;

- Knowing the properties of the most famous examples of random variables (Bernoulli, binomial, geometric, exponential, Poisson, normal, etc.);
- Mastering the underpinnings of the most common parameter estimation technique, maximum likelihood estimation;
- Understanding the difference between a sample and a population;
- Being able to state the Central Limit Theorem, understanding its importance, and applying it in a variety of basic situations;
- Being able to implement, by hand and in Python, all elementary one- and two- sample tests of hypotheses and confidence interval constructions (e.g., means, proportions, correlation, ratios of variances, etc.);
- Understanding the fundamental axioms, rules, and laws of probability theory;
- Simulating (using Python) random numbers governed by various probability distributions using the method of inverse transformation and the acceptance- rejection technique;
- Defining, and working with examples related to, conditional probability;
- Understanding the importance of the concept of independence;
- Proving and using the Law of Total Probability;
- Using the Law of Total Probability to prove Bayes' Theorem and deploying Bayes' Theorem in a variety of practical situations;
- Working with random vectors as well as random variables;
- Working with multivariate distributions, as well as the concepts of conditional expectation and independence in a high-dimensional setting; and
- Working with the multivariate Gaussian distribution.

If you are looking for an online and non-credit bearing opportunities to review this material, there are several you might consider. We recommend the first two courses in University of Michigan's "Statistics with Python Specialization" at Coursera. We also recommend Berkeley's three-part introduction to statistics -- addressing probability, descriptive statistics, and inferential statistics (at EdX). However, previous program participants have indicated that these learning experiences are not as rigorous as our program's boot camp review of probability and statistics.

Consequently, we are also placing onto the previously-mentioned Canvas site for admitted students Viviana Peña-Márquez's class notes from a previous instance of MSDS 504. We recommend that you go through those notes quickly, marking any concepts that seem less familiar to you. Then go back through those notes a second

time, reading those areas that you marked more carefully. Finally, using the Summer 2022 course outline for MSDS 504, cross-reference the topics for which you need do the most review. Then read the associated excerpts from the reference materials at the Canvas site for admitted students.

Computer Programming Languages and Tools

Programming

The computation bootcamp is a “review” course and instructors assume that you already learned the concepts while you were taking pre-requisite courses. Those topics include the following subjects.

- Basic Computer Architecture
- Terminal and Shell Commands
- Version Control & Git
- Unit Test (pytest)
- Code Standards
- Python Programming
    - Debugging and Error Handling
    - Conditional Statements
    - Loop
    - File I/O
    - Data Aliasing
    - Function
    - Packages, Libraries and Modules
    - Object Oriented Programming

When students are struggling on the programming assignments, our first question to them is: “What did you do in the months prior to the bootcamp?” The answer is typically not studying programming enough. Every year, a few students do not pass the computational boot camp and must exit the program. We have created the following guide so you can properly prepare yourself.

You’ll learn most concepts in this program through coding. The easier it is for you to code and use programming tools, the easier you will find the entire curriculum.

How to learn

Many of you have already taken programming courses, either in-person or online, but you might not have gotten that much out of it. If you want to learn how to write code, there is no substitute for actually typing code to solve problems. Do not just listen and watch the instructor write a program. You need to write the code. Just like you don't learn to play an instrument by listening to music. When learning to play a musical instrument, first you just cover popular tunes (exactly copying what the instructor is typing). Then you start writing your own riffs and songs (solving problems and doing projects). Just like playing songs and creating is the best part of playing music, solving problems is the best part of coding.

Do not copy and paste code while learning. Manually typing code is part of the learning process. You might make small typos which will help hone your debugging skills and overall coding writing abilities.

Also, write code in an interactive environment, see Jupyter Notebook section below, so you get immediate feedback about what works and what does not work. Later we'll write scripts in .py files and run the scripts at the command line. For now, writing scripts will slow down your learning curve.

One of the best ways of studying is to sit in front of a blank screen with a coding prompt for a problem you have already solved. Solve the problem from memory without looking at your previous solution or the internet. Only use those resources when you are completely stuck. Drilling skills from memory will reinforce what you have already learned.

One of our favorite tools is **Python Tutor,** http://pythontutor.com/. It visualizes what happens when you run Python code. It is useful to understand existing code or debug broken code. Being able to visualize code execution is a critical skill for all programmers.

We suggest everyone complete Python for Everybody (PY4E) https://www.py4e.com/lessons. As stated above, you have to learn through doing so create a login and complete the exercises which are auto-graded.

Here are additional online courses:

- https://www.coursera.org/learn/python

- https://www.coursera.org/learn/interactive-python-1
- https://www.edx.org/course/cs-all-introduction-computer-science-harveymuddx- cs005x-0

Tools

Throughout the MSDS program, you will use the same tools as professional Data Scientists. That means by the time you start Practicum or a job, you'll be ready to contribute to the team right away.

Please check out:

1. A quick introduction to the hardware and software elements of your machine
2. A broad overview of python and tools used in our MSDS program

Before arriving at orientation, you should have Anaconda 3 installed on your laptop and have some familiarity with the command line (Terminal.app or iTerm2 etc…). Using the command line is a critical skill in this program (all the rest of the examples assume use of the command line). The command line is also called "the shell." Data Scientists use the command line every day to run scripts, manage files, or use computers in the cloud. Go through a course, such as https://guide.bash.academy/, to make sure you are familiar with the command line.

We mostly use Jupyter Notebooks, for more information see http://jupyter.org/. Notebooks combine programs and text which is nice to interweave your code with your thoughts. Data Science programming is particularly challenging because of the variety of tools we use. Notebooks are extremely helpful because data, data frames, code, graphs, and output are all in the same document. Here is a video tutorial to check out: https://www.youtube.com/watch?v=HW29067qVWk.