Large-Scale Computing for the Social Sciences

Spring 2025 - MACS 30123/MAPS 30123/PLSC 30123

Section 1:

- Instructor: Jon Clindaniel
  - 1155 E. 60th Street, Rm. 215
  - jclindaniel@uchicago.edu
  - Office Hours:
    - Drop-In (No appointment needed): Tuesday 2:00-4:00pm
    - Schedule an Appointment Thursday 2:00-4:00pm
- Course Information:
  - Location: 1155 E. 60th Street, Rm. 295
  - Time: Monday/Wednesday Lecture from 4:30-5:50 PM, Friday Lab Section from 4:30-5:50 PM
  - Canvas Course Site

Section 2:

- Instructor: Fabricio Vasselai
  - 1155 E. 60th Street, Rm. 213A
  - vasselai@uchicago.edu
  - Office Hours (Schedule an Appointment):
    - Mondays: remote, variable times.
    - Tuesdays and Thursdays 5:00pm-6:30pm in person at 1155 E. 60th St., Rm 213A.
- Course Information:
  - Location: 1155 E. 60th Street, Rm. 295
  - Time: Tuesday/Thursday Lecture from 12:30-1:50 PM, Friday Lab Section from 1:30-2:50 PM
  - Canvas Course Site

Teaching Assistants

Nalin Bhatt	Ethan Kozlowski	Max Zhu
nalinb@uchicago.edu	ethanjkozlowski@uchicago.edu	maxzhuyt@uchicago.edu
Office Hours: M/W, 3:30-4:30pm	**Office Hours:** W, 1:00-3:00pm	**Office Hours:** F, 10:00am-12:00pm
1155 E. 60th St., Rm 226-A	1155 E. 60th St., Rm 222	1155 E. 60th St., Rm 222

Course Description

Computational social scientists increasingly need to grapple with data that is too big and code that is too resource intensive to run on a local machine. Using Python, students in this course will learn how to effectively scale their computational methods beyond their local machines – optimizing and parallelizing their code across clusters of CPUs and GPUs, both on-premises and in the cloud. The focus of the course will be on social scientific applications, such as: accelerating social simulations by several orders of magnitude, processing large amounts of social media data in real-time, and training machine learning models on economic datasets that are too large for an average laptop to handle.

Prerequisites: MACS 30121 and MACS 30122, or equivalent (e.g. CAPP 30121 and CAPP 30122). Note that this the accelerated version of MACS 30113.

Course Structure

This course is structured into several modules, or overarching thematic learning units, focused on teaching students fundamental large-scale computing concepts, as well as giving them the opportunity to apply these concepts to Computational Social Science research problems. Students can access the readings, assignments, and resources for each of the class sessions within these modules on the Canvas course site for your section. If students have any questions about the course content, they should post these questions in the Ed Discussion forum for the course, which they can access by clicking the "Ed Discussion" tab on the left side of the screen on the Canvas course site. To see an overall schedule and syllabus for the course, as well as access additional course-related files (which we will walk through in in-class activities), students should visit (and clone/fork) the GitHub Course Repository, available here.

During regular class hours, we will meet for a mixture of lecture, group activities, and in-class coding exercises related to the topic for the day. Attendance to the class sessions is mandatory and is an important component of the final course grade. Students should prepare for these classes by reading the assigned readings ahead of every session. All readings are available online and are linked in the course schedule below (and in the corresponding module on Canvas). Starting in Week 2, there will also be an additional TA-led lab section on Fridays (in our regular classroom), focused on providing additional practice with scalable computing strategies.

In order to practice scalable computing skills and complete the course assignments, students will be given free access to on-premise cluster computing resources, Amazon Web Services (AWS) cloud computing resources, and DataCamp. More information about accessing these resources will be provided to registered students in the first several weeks of the quarter.

Grading

There will be an assignment due at the end of each unit (3 in total). Each assignment is worth 20% of the overall grade, with all assignments together worth a total of 60%. Additionally, attendance and participation will be worth 10% of the overall grade. Finally, students will complete a final project that is worth 30% of the overall grade (25% for the project itself, and 5% for an end-of-quarter video presentation).

Course Component	**Grade Percentage**
Assignments (Total: 3)	60%
Attendance/Participation	10%
Final Project	5% (Presentation)
	25% (Project)

Grades are not curved in this class or, at least, not in the traditional sense. We use a standard set of grade boundaries: * 95-100: A * 90-95: A- * 85-90: B+ * 80-85: B * 75-80: B- * 70-75: C+ * <70: Dealt on a case-by-case basis

We curve only to the extent we might lower the boundaries for one or more letter grades, depending on the distribution of the raw scores. We will not raise the boundaries in response to the distribution.

So, for example, if you have a total score of 82 in the course, you are guaranteed to get, at least, a B (but may potentially get a higher grade if the boundary for a B+ is lowered).

If you would like to be graded on a Pass/Fail (P/F) basis, send a private message to the course staff on the Ed Discussion forum **before the Final Project is due**. A total score of 75 and above in the class will qualify for a "P" in the class.

Participation Expectations

We expect all students to participate in each class session in person (having read all of the readings listed for the day ahead of class time). Your participation grade (10% of your overall grade in the class) will be based on your engagement and completion of in-class activities.

If, for whatever reason, you cannot attend a class session, send a private message to the course staff **ahead of the class session** on the class Ed Discussion forum. We will evaluate these requests on a case-by-case basis and assign an alternative assignment to make up participation credit for the day.

Final Project

For their final project, students will write large-scale computing code that solves a social science research problem of their choosing. For instance, students might perform a computationally intensive demographic simulation, or they may choose to collect, analyze, and visualize large social media data, or do something else that employs large-scale computing strategies. Students will additionally record a short video presentation about their project. Detailed descriptions and grading rubrics for the project and presentation are available on the Canvas course site for your section.

Late Assignments/Projects

Unexcused Late Assignment/Project Submissions will be penalized 10 percentage points for every hour they are late. For example, if an assignment is due on Wednesday at 11:59pm, the following percentage points will be deducted based on the time stamp of the last commit in your private GitHub assignment repository.

Example last commit	**Percentage points deducted**
12:00am to 12:59am	-10 percentage points
1:00am to 1:59am	-20 percentage points
2:00am to 2:59am	-30 percentage points
3:00am to 3:59am	-40 percentage points
…	…
9:00am and beyond	-100 percentage points (no credit)

If, for whatever reason, you need an extension on an assignment deadline, send a private message to the course staff **ahead of the assignment deadline** on the class Ed Discussion forum and we will evaluate these requests on a case-by-case basis.

Plagiarism on Assignments/Projects

Academic honesty is an extremely important principle in academia and at the University of Chicago.

- Writing assignments must quote and cite any excerpts taken from another work.
- If the cited work is the particular paper referenced in the assignment, no works cited or references are necessary at the end of the composition.
- If the cited work is not the particular paper referenced in the assignment, you MUST include a works cited or references section at the end of the composition.
- Any copying of work other than your own will result in a zero grade and potential further academic discipline.
- If we discover that you have shared or posted questions/solutions from any class assignments or exams in a public, online space, this will also result in a zero grade and potential further academic discipline.
- You are permitted to consult with an AI Assistant, such as ChatGPT or GitHub Copilot, as you work on assignments (just note

that these tools will not always produce the most scalable or up-to-date responses). If you do use an AI Assistant, you must submit a complete log of the prompts that you used and properly cite the use of the AI tool. To get the most out of this class, we recommend that you only use an AI Assistant in the following way:

- First, write code/text on your own without AI assistance
- Turn to your AI Assistant if you are stuck and have run out of ideas on how to proceed. Some areas where AI Assistants can be helpful are in idea generation, syntax correction, and providing alternative (potentially more scalable) solutions
- Critically evaluate AI responses and assess their strength (by consulting course materials and/or writing formal tests for code)
- Rewrite code/text from scratch (in your own voice) to consolidate what you have learned

If you have any questions about citations, references, or what constitutes plagiarism, consult with your instructor.

Statement of Diversity and Inclusion

The University of Chicago is committed to diversity and rigorous inquiry from multiple perspectives. The MAPSS, CIR, and Computation programs share this commitment and seek to foster productive learning environments based upon inclusion, open communication, and mutual respect for a diverse range of identities, experiences, and positions.

This course is open to all students who meet the academic requirements for participation. Any student who has a documented need for accommodation should contact Student Disability Services (773-702-6000 or disabilities@uchicago.edu) and the instructor as soon as possible.

Course Schedule

Note that there will also be weekly lab sections in Weeks 2-9 on Fridays in our normal classroom.

Unit	Week	Day	Topic	Readings	Assignment
Fundamentals of Large-Scale Computing	Week 1: Introduction to Large-Scale Computing for the Social Sciences (3/24-3/28)	Day 1	Introduction to the Course + Code Optimization with Cython/Numba	Faster code via static typing (Cython), A ~5 minute guide to Numba	
		Day 2	General Considerations for Large-Scale Computing	Robey and Zamora 2021 (Chapter 1)	
	Week 2: On-Premise Large-Scale CPU-computing with MPI (3/31-4/4)	Day 1	An Introduction to Computing Clusters and CPU Hardware considerations	Pacheco 2011 (Ch. 1-2; on Canvas)	
		Day 2	Cluster Computing via Message Passing Interface (MPI) for Python	Pacheco 2011 (Ch. 3; on Canvas), Dalcín et al. 2008	
	Week 3: On-Premise GPU-computing (4/7-4/11)	Day 1	An Introduction to GPUs and GPU Programming	Cheng et al. 2014 (Ch. 1; on Canvas), Scarpino 2012 (Ch. 1; on Canvas)	
		Day 2	Harnessing GPUs in Python	Klöckner et al. 2012 "PyCUDA and PyOpenCL"	
Architecting Computational Social Science Data Solutions in the Cloud	Week 4: An Introduction to Cloud Computing and Cloud HPC Architectures (4/14-4/18)	Day 1	Bursting HPC into the Cloud	Jorissen and Bouffler 2017 (Read Ch. 1, Skim Ch. 4-7; on Canvas), Armbrust et al. 2009, Introduction to HPC on AWS, HPC Architectural Best Practices (focus on the "General Design Principles" and "Scenarios" sections)	
		Day 2	An Introduction to Boto3 and Serverless HPC	Jonas et al. 2019, "What is AWS Lambda", "What is AWS Step Functions?", Boto3 Documentation (skim)	Due: Assignment 1 (Wednesday, 4/16, 11:59 PM)
	Week 5: Architecting Large-Scale Data Solutions in the Cloud (4/21-4/25)	Day 1	"Data Lake" Architectures	Data Lakes and Analytics on AWS, AWS Data Lake Whitepaper, *Introduction to AWS Boto in Python* (DataCamp Course; Practice working with S3 Data Lake in Python)	
		Day 2	Large-Scale Database Solutions	"Which Database to Use When?" (YouTube), Optional: Data	

Unit	Week	Day	Topic	Readings	Assignment
				Warehousing on AWS Whitepaper, Big Data Analytics Options on AWS	
	Week 6: Large-Scale Data Ingestion and Processing (4/28-5/2)	Day 1	Event-Driven Ingestion and Processing	"Scalable serverless event-driven architectures with SNS, SQS & Lambda" (YouTube) Optional: "Using Lambda with Amazon SQS", "Fanout to Amazon SQS Queues", "Using AWS Lambda with Amazon S3"	
		Day 2	Batch Processing with Apache Hadoop and MapReduce	White 2015 (read Ch. 1-2, Skim 3-4; on Canvas), Dean and Ghemawat 2004, "What is Amazon EMR?", Running MapReduce Jobs with Python's "mrjob" package on EMR (Fundamentals and Elastic MapReduce Quickstart)	
High-Level Paradigms for Large-Scale Data Analysis, Prediction, and Presentation	Week 7: Spark (5/5-5/9)	Day 1	Large-Scale Data Processing and Analysis with PySpark	*Introduction to PySpark* (DataCamp Course), Optional: Videos about accelerating Spark with GPUs (via Horovod for deep learning, and the RAPIDS libraries for both ETL and ML acceleration in Spark 3.0)	
		Day 2	A Deeper Dive into the PySpark Ecosystem	*Machine Learning with PySpark* (DataCamp Course), Hunter 2017 (Spark Summit Talk), GraphFrames Documentation for Python, Spark NLP Documentation, Optional: *Feature Engineering with PySpark* (DataCamp Course)	Due: Assignment 2 (Wednesday, 5/7, 11:59 PM)
	Week 8: Dask (5/12-5-16)	Day 1	Introduction to Dask	"Why Dask," *Parallel Programming with Dask* (DataCamp Course)	
		Day 2	Accelerating Dask		
	Week 9: Presenting Data and Insights from Large-Scale Data Pipelines (5/19-5/23)	Day 1	Building and Deploying (Scalable) Public APIs and Web Applications with Flask and AWS Elastic Beanstalk	Documentation for Elastic Beanstalk and skim through the Flask "Forward" as well as the current documentation	
		Day 2	Visualizing Large Data	Documentation for DataShader and Bokeh, and integrating the two libraries using HoloViews	Due: Assignment 3 (Wednesday, 5/21, 11:59 PM)
Student Projects	Week 10: Final Projects (5/26-5/30)				Due: Final Project + Presentation Video (Friday, 5/30, 11:59 PM)

Works Cited

"A ~5 minute guide to Numba." https://numba.readthedocs.io/en/stable/user/5minguide.html. Accessed 3/2021.

Armbrust, Michael, Fox, Armando, Griffith, Rean, Joseph, Anthony D., Katz, Randy H., Konwinski, Andrew, Lee, Gunho, Patterson, David A., Rabkin, Ariel, Stoica, Ion, and Matei Zaharia. 2009. "Above the Clouds: A Berkeley View of Cloud Computing." Technical report, EECS Department, University of California, Berkeley.

"Big Data Analytics Options on AWS." July 2021. AWS Whitepaper.

"AWS Elastic Beanstalk Developer Guide." https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/Welcome.html. Accessed 3/2021.

"Building Big Data Storage Solutions (Data Lakes) for Maximum Flexibility." July 2017.

Cheng, John, Grossman, Max, and Ty McKercher. 2014. *Professional CUDA C Programming*. Indianapolis, Indiana: John Wiley & Sons.

Dalcín, Lisandro, Paz, Rodrigo, Storti, Mario, and Jorge D'Elía. 2008. "MPI for Python: Performance improvements and MPI-2 extensions." *J. Parallel Distrib. Comput.* 68: 655-662.

"Data Lakes and Analytics on AWS." https://aws.amazon.com/big-data/datalakes-and-analytics/. Accessed 3/2023.

"Data Warehousing on AWS." January 2021. AWS Whitepaper.

"DataShader Documentation." https://datashader.org/index.html. Accessed 3/2021.

Dean, Jeffrey, and Sanjay Ghemawat. 2004. "MapReduce: Simplified data processing on large clusters." In *Proceedings of Operating Systems Design and Implementation (OSDI)*. San Francisco, CA. 137-150.

Evans, Robert and Jason Lowe. "Deep Dive into GPU Support in Apache Spark 3.x." https://www.youtube.com/watch?v=4MI_LYah900. Accessed 3/2021.

"Fanout to Amazon SQS queues." https://docs.aws.amazon.com/sns/latest/dg/sns-sqs-as-subscriber.html. Accessed 3/2022.

"Faster code via static typing." http://docs.cython.org/en/latest/src/quickstart/cythonize.html. Accessed 3/2021

Feature Engineering with PySpark. https://learn.datacamp.com/courses/feature-engineering-with-pyspark. Accessed 3/2020.

"Flask Documentation." https://flask.palletsprojects.com/. Accessed 2/2023.

"Flask Forward." https://web.archive.org/web/20211106135422/https://flask-doc.readthedocs.io/en/latest/foreword.html. Accessed 2/2023.

"GraphFrames user guide - Python." https://docs.databricks.com/spark/latest/graph-analysis/graphframes/user-guide-python.html. Accessed 3/2020.

"HPC Architectural Best Practices." https://docs.aws.amazon.com/wellarchitected/latest/high-performance-computing-lens/general-design-principles.html. Accessed 2/2023.

Hunter, Tim. October 26, 2017. "GraphFrames: Scaling Web-Scale Graph Analytics with Apache Spark." https://www.youtube.com/watch?v=NmbKst7ny5Q.

Introduction to AWS Boto in Python. https://campus.datacamp.com/courses/introduction-to-aws-boto-in-python. Accessed 3/2020.

"Introduction to HPC on AWS." n.d. AWS Whitepaper.

Introduction to PySpark. https://learn.datacamp.com/courses/introduction-to-pyspark. Accessed 3/2020.

Jonas, Eric, Schleier-Smith, Johann, Sreekanti, Vikram, and Chia-Che Tsai. 2019. "Cloud Programming Simplified: A Berkeley View on Serverless Computing." Technical report, EECS Department, University of California, Berkeley.

Jorissen, Kevin, and Brendan Bouffler. 2017. *AWS Research Cloud Program: Researcher's Handbook*. Amazon Web Services.

Klöckner, Andreas, Pinto, Nicolas, Lee, Yunsup, Catanzaro, Bryan, Ivanov, Paul, and Ahmed Fasih. 2012. "PyCUDA and PyOpenCL: A Scripting-Based Approach to GPU Run-Time Code Generation." *Parallel Computing* 38(3): 157-174.

Machine Learning with PySpark. https://campus.datacamp.com/courses/machine-learning-with-pyspark. Accessed 3/2020.

"mrjob v0.7.1 documentation." https://mrjob.readthedocs.io/en/latest/index.html. Accessed 3/2020.

Pacheco, Peter. 2011. *An Introduction to Parallel Programming*. Burlington, MA: Morgan Kaufmann.

Parallel Programming with Dask. https://learn.datacamp.com/courses/parallel-programming-with-dask-in-python. Accessed 3/2020.

Petrossian, Tony, and Ian Meyers. November 30, 2017. "Which Database to Use When?" https://youtu.be/KWOSGVtHWqA. AWS re:Invent 2017.

Pirtle, Justin. December 8, 2020. "Scalable serverless event-driven architectures with SNS, SQS, and Lambda." https://www.youtube.com/watch?v=8zysQqxqj0I. AWS re:Invent 2020.

Robey, Robert and Yuliana Zamora. 2021. *Parallel and High Performance Computing*. Shelter Island, NY: Manning.

Scarpino, Matthew. 2012. OpenCL in Action. Shelter Island, NY: Manning.

Sergeev, Alex. March 28, 2019. "Distributed Deep Learning with Horovod." https://www.youtube.com/watch?v=D1By2hy4Ecw.

"Spark NLP Documentation." https://nlp.johnsnowlabs.com/. Accessed 3/2021.

Storage Best Practices for Data and Analytics Applications." November 2021. AWS Whitepaper.

"The Bokeh Visualization Library Documentation." https://bokeh.org/. Accessed 3/2021.

"Use an Amazon EMR Studio." https://docs.aws.amazon.com/emr/latest/ManagementGuide/use-an-emr-studio.html. Accessed 2/2023.

"Using AWS Lambda with S3." https://docs.aws.amazon.com/lambda/latest/dg/with-s3.html. Accessed 3/2022.

"Using Lambda with Amazon SQS." https://docs.aws.amazon.com/lambda/latest/dg/with-sqs.html. Accessed 3/2022.

"What is Amazon EMR." https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-what-is-emr.html. Accessed 3/2020.

"What is AWS Lambda?" https://docs.aws.amazon.com/lambda/latest/dg/welcome.html. Accessed 3/2022.

White, Tom. 2015. *Hadoop: The Definitive Guide*. Sebastopol, CA: O'Reilly.

"Why Dask." https://docs.dask.org/en/latest/why.html. Accessed 3/2020.

"Working with large data using datashader." http://holoviews.org/user_guide/Large_Data.html. Accessed 3/2021.