MENG 25500 / MENG 35500
Classical Molecular and Materials Modeling
Winter 2026

Instructor: Prof. Andrew L. Ferguson
389 ERC
andrewferguson@uchicago.edu

TAs: Yuxuan Zhang
yuxuanz1@uchicago.edu
Ke-Hsin Wang
kehsinw@uchicago.edu

CRN: MENG 25500 / MENG 35500 (100 units)

Lecture: KPTC 105 ◆ 9:30–10:50 ◆ Tue, Thu

Office Hours:
ERC 389 ◆ 17:00-17:50 ◆ Wed – Instructor office hours
ERC 381 ◆ 12:00–12:50 ◆ Thu – TA office hours I
KPTC 105 ◆ 9:30–10:20 ◆ Fri – TA office hours II

Dates: 01/05/2026 – 03/14/2026

Course Summary

This course will introduce students to the methods of molecular modeling. The topics covered will include an introduction to the origin of molecular forces, a brief introduction to statistical mechanics and ensemble methods, and an introduction to molecular dynamics and Monte Carlo simulations. The course will also cover elements of advanced sampling techniques, including parallel tempering, umbrella sampling, and other common biased sampling approaches. Students will also establish expertise in scientific programming in Python 3. Course work or research experience is strongly recommended in elementary programming and physical chemistry or thermodynamics.

Prerequisites

- [**REQUIRED**] MENG 21400 – Molecular Engineering Thermodynamics
  or CHEM 26200 – Thermodynamics
  or PHYS 27900 – Statistical and Thermal Physics
- [**REQUIRED**] MATH 20100 – Mathematical Methods for Physical Sciences II
  or PHYS 22100 – Mathematical Methods in Physics
- [**RECOMMENDED**] Prior experience in elementary scientific programming

Required Text

None.

Recommended Texts

Classical Molecular Simulation:

D. Frenkel and B. Smit *Understanding Molecular Simulation: From algorithms to applications* (Academic Press, 2002)
→ https://catalog.lib.uchicago.edu/vufind/Record/11155428

M.P. Allen and D.J. Tildesley *Computer Simulation of Liquids* (Oxford University Press, 1989)
→ https://catalog.lib.uchicago.edu/vufind/Record/965300

Scientific programming in Python 3:

J. Kiusalaas *Numerical Methods in Engineering with Python* (Cambridge University Press, 2005)
→ https://catalog.lib.uchicago.edu/vufind/Record/8209396

M. Newman *Computational Physics* (2012)
→ https://www.amazon.com/Computational-Physics-Mark-Newman/dp/1480145513

Attendance

Lecture: Lectures will be split between (i) traditional instruction covering the mathematical underpinnings of the numerical tools, (ii) immersive project-based learning. **Attendance is expected.**

Discussion: The discussion section serves as TA office hours. This time may occasionally be used to provide make-up lectures. **Attendance is optional.**

Assessment

Quizzes: Short, online multiple-choice quizzes will be issued to gauge understanding and mastery of the mathematical and algorithmic principles underlying the numerical techniques. Quizzes will be available online for a specified time period, and solutions posted after the quiz closes. **Accordingly, no extensions can be granted.**

Topic Projects: Projects associated with each topic are a primary means of instruction and assessment for the course. Students will be provided with a project brief comprising a list of learning objectives, problem statement, solution approach, and set of deliverables. **Late submissions will not be accepted, but students with legitimate requests for extensions should contact Prof. Ferguson well in advance of the due date.**

Term Project: Students will design and execute a term project to code up and implement an advanced molecular simulation technique in Python. The topic projects will provide a foundational code base upon which to construct the advanced simulation implementation. Students may select from a list of potential project topics or select their own. Graduate students will complete projects individually; undergraduate students will complete projects in small teams. All project topics must be approved by the course instructor. There are two deliverables associated with the term project:

(i) A written report on theory, implementation, and demonstration of the technique (**pdf**)

- ~5 pages: single column, single spaced, excl. figures & references
- The report should contain the following sections:
  - Abstract (~100 words)
  - Introduction – motivation, background, purpose of the method
  - Methods – description of the method, mathematical and algorithmic details
  - Results – application of the method to system of interest, high quality plots, critical appraisal of results, comparison to literature/experiment
  - Conclusions – summary of key findings, significance, limitations, future work
  - References – references cited

(ii) Python code implementing and demonstrating the advanced simulation technique (**ipynb**)

- You must have *something* working in application to toy and/or molecular system(s)

Exams: None.

Grading

Breakdown:

Quizzes		5%
Project 1 –	Intro to Python **(ungraded)**	0%
Project 2 –	Elements of Molecular Simulation	12.5%
Project 3A –	Monte Carlo I	12.5%

Project 3B – Monte Carlo II	12.5%
Project 4A – Molecular Dynamics I	12.5%
Project 4B – Molecular Dynamics II	12.5%
Project 5 – Molecular Dynamics in LAMMPS	12.5%
Term Project	20%

Letter Grades: Letter grades will be based on final aggregate student scores, with numerical cutoffs specified by the instructor. However, students with aggregate scores >95% are guaranteed *at least* an A, >85% *at least* a B, and >75% *at least* a C (i.e. cutoffs for these letter grades will not be higher than these values).

Canvas

Course announcements, materials, grades, quizzes, and projects will be posted via Canvas (https://canvas.uchicago.edu). Online quizzes and projects will be submitted via this portal. It is students' responsibility to check Canvas for announcements and updates. Canvas is best viewed using Chrome or Firefox web browsers; Safari is known to cause problems.

Slack

We will be using Slack as a virtual and public forum for questions discussion. The system allows for fast responses from the instructor, TA, and classmates, and permits students to see previous questions and answers. Rather than emailing questions to the teaching staff or using Canvas chat, please post your questions publicly on Slack. If you have not already been automatically enrolled, please sign up via https://meng255003550-awe4935.slack.com

AI-Based Language Models

AI-based large language models (e.g., ChatGPT, Bard, or GitHub Copilot) are now prevalent and useful tools in many facets of modern life. The role and implications of these technologies in academic education and research is still a matter of discussion and debate, but they do present a potential danger to ethical behavior in the classroom. **It is ethical** to use these tools to assist you in understanding materials (e.g., summarizing a technical article, walking you through challenging mathematical developments), communicating your own work and thoughts (e.g., improving technical writing, informing word choices), or to debug and improve computer code. If you do choose to use these tools in this manner, you should provide a written acknowledgement of these tools, in the same way as you would cite any other external source used to complete an assignment, along with a summary of how these tools were used. **It is not ethical** to use these tools to produce responses or code that you present as your own in the completion of your assignments for this course. This is plagiarism – you are presenting the work of another as your own. This also compromises your learning within the course, misleads the instructor as to your understanding and mastery of the course material, and runs counter to the ethos and goals of education and learning. Furthermore, these tools are far from infallible and may not provide you trustworthy or correct answers. A good discussion of these models and their role in academic coursework is provided by the University of Minnesota Library at https://libguides.umn.edu/chatgpt.

Plagiarism

Students are responsible for producing their own quiz answers, code to solve projects, and project reports. Collaborative interactions and peer learning are encouraged, but each student

must perform all calculations themselves, write their own code, and author their own reports. **The use of AI-based large language models to produce responses or code that you present as your own without written acknowledgement and citation constitutes plagiarism.** Students are responsible for awareness of the definition and penalties for plagiarism in the Student Manual (https://studentmanual.uchicago.edu/academic-policies/academic-honesty-plagiarism/). Further details on what does and does not constitute plagiarism is available here: https://internationalaffairs.uchicago.edu/page/honest-work-and-academic-integrity-plagiarism. **Plagiarism will not be tolerated and verified incidents will result in all parties receiving a zero and formal academic sanctions.**

Disability Accommodations

The University of Chicago is committed to ensuring equitable access to our academic programs and services. Students who have been approved for academic accommodations by Student Disability Services (https://disabilities.uchicago.edu) and need a reasonable accommodation to participate fully in this course should follow the procedures established by SDS. Timely notifications are required to ensure that your accommodations can be implemented. Please set up a time to meet with the instructor to discuss your access needs in this course after you have completed the SDS procedures for requesting accommodations.

Academic Integrity, Civility, and Personal Responsibility

The University of Chicago is committed to providing a safe and welcoming environment for rigorous learning, inquiry, and research within a climate of respect, civility, and inclusion. The University statements on diversity and non-discrimination are available at https://provost.uchicago.edu/handbook/clause/university-statements-diversity-and-non-discrimination. Every member of the University of Chicago intellectual community is responsible for upholding the highest standards of academic integrity, mutual respect and civility, and personal responsibility. Within the classroom, these values are critical for the success of our educational mission and your growth and development as a scholar. The expectations around integrity, civility, and responsibility are available at https://college.uchicago.edu/student-services/academic-integrity-student-conduct.
Resources for reporting instances of bias or discrimination are available here https://cares.uchicago.edu/.

Laptop Computer

A laptop computer with a working Anaconda install of Python 3 and attendant libraries is required for this course. If you do not have access to a laptop, or if your laptop fails during the course, the instructor can help make arrangements for a laptop loan.

Software

It is a primary learning objective of the course to provide training in Python 3 as a popular and ubiquitous language of numerical computing. **Programming assignments must be completed and submitted as Python 3 Jupyter notebooks.** It is **highly recommended** that regardless of any current Python installations students install the Anaconda Python 3 distribution and setup the class virtual environment following the instructions provided.

What is Python? Python is a powerful, versatile, and user-friendly programming language that has become a popular and ubiquitous standard in scientific computing and data science. It is easy to use and is equipped with numerous libraries to perform common numerical tasks.

What is the difference between Python 2 and Python 3? Python 3 was a major revision of the language, with the main changes to do with print syntax, integer division, and library support. Unfortunately, Python 3 is not fully backwards compatible with Python 2. A large body of existing code is written in Python 2, but Python 3 is the future. Accordingly, this course will work exclusively in Python 3.

What is Anaconda? Anaconda is a popular and free Python distribution that comes prepackaged with all of the commonly needed Python libraries and sets up an environment in which to manage and control the Python implementation on your system.

How do I install Anaconda Python 3? Navigate to https://www.anaconda.com/download/ and download and install the latest Python 3 release appropriate for your operating system. Windows, Linux, and MacOS X are all supported. Specific installation instructions are here: https://docs.anaconda.com/anaconda/install/

What is a Jupyter Notebook? A Jupyter Notebook is a library within Python (and other programming languages) that provides an open-source web application within which one can develop code, provide narrative comments, import and export data, and generate visualizations. In short, it provides a place to develop code within an environment that is useable, readable, reproducible, and sharable.

Course Coverage

I. Introduction to Python 3

Anaconda installation, Python programming, modules, Jupyter notebooks, operators, containers, basic flow control, I/O, visualization, scoping, virtual environments, libraries (numpy, scipy, matplotlib, pandas, seaborn, scikit-learn, pytorch)

II. Elements of Molecular Simulation

Variables, ensembles, dimensions and units, boundary conditions, initialization, statistical uncertainty, ergodicity, mechanical and thermal observables, thermodynamic averages and fluctuations, dynamical equilibrium and detailed balance

III. Monte-Carlo Simulation

Multidimensional integrals, stochastic and deterministic algorithms, variance and scaling, random numbers, boundary conditions, detailed balance, acceptance probabilities, importance sampling, trial moves, convergence, interaction potentials, Metropolis-Hastings algorithm

IV. Molecular Dynamics Simulation

Lagrangian and Hamiltonian formulations, Integrators, reversibility, boundary conditions, force fields, thermostats, barostats, electrostatics, dynamical observables, transport properties and time-correlation functions

V. Molecular Dynamics Simulation in LAMMPS

Tutorial and practice using the LAMMPS molecular dynamics simulation suite

VI. Advanced Simulation Techniques

C*hain molecules, configurational bias, parallel tempering, pruned-enriched Rosenbluth, expanded ensembles, Wang-Landau density of states, Gibbs ensemble, histogram reweighting, umbrella sampling, metadynamics, committor probabilities*

Tentative Schedule

Class	Date	Day	Module	Topic	Due
1	Jan 6*	T	I	Introduction to Python	
2	Jan 8*	R	I	Introduction to Python	
	Jan 9	F			–
3	Jan 13	T	II	Elements of Mol Sim	
4	Jan 15	R	II	Elements of Mol Sim	
	Jan 16	F			Quiz 1 Project 1 (ungraded)
5	Jan 20	T	III	Monte-Carlo	
6	Jan 22	R	III	Monte-Carlo	
	Jan 23	F			Quiz 2 Project 2
7	Jan 27	T	III	Monte-Carlo	
8	Jan 29	R	III	Monte-Carlo	
	Jan 30	F			Quiz 3 Project 3A
9	Feb 3	T	IV	Molecular Dynamics	
10	Feb 5	R	IV	Molecular Dynamics	
	Feb 6	F			Project 3B
11	Feb 10	T	IV	Molecular Dynamics	
12	Feb 12	R	IV	Molecular Dynamics	
	Feb 13	F			Quiz 4 Project 4A Term Project Topic (+ Team)
13	Feb 17	T	V	LAMMPS	
14	Feb 19	R	V	LAMMPS	
	Feb 20	F			Project 4B

15	Feb 24	T	V	LAMMPS	
16	Feb 26	R	VI	Advanced Techniques	
	Feb 27	F			Project 5
17	Mar 3	T	VI	Advanced Techniques	
18	Mar 5	R	VI	Advanced Techniques	
	Mar 13	F			Term Project Code + Report

* Prof. Ferguson on travel these dates, appropriate arrangements TBA.