Computational Psychology

Time & Location: TBD

Instructor: Dr. Chujun Lin
Email: chl211@ucsd.edu
Office hours and location: TBD

This syllabus is subject to changes

COURSE DESCRIPTION

The development of computational tools, such as artificial intelligence, has transformed daily life and many areas of research. For instance, the development of large language models such as ChatGPT, have changed the way people write and think. These computational tools also hold great potential for advancing psychological research. However, they are still unfamiliar to many students and psychologists.

In this class, you will learn about how to apply computational tools to facilitate practical problem solving in the real world and research at different stages, including before, during, and after data collection. Specifically, through a combination of lectures (which focus on the mathematical basics and developments of these computational methods), in-class engagements (which focus on translational application of the computational methods to solving new practical problems), and coding workshops (which focus on practical coding skills that implement the computational methods), I will introduce you to methods for handling complex stimuli, validating data quality, and modeling big data. Workshops will be administered via Colab using Python, so please be sure to bring a laptop to every class.

COURSE OBJECTIVES

Over the course of this quarter, I hope you will learn:

- The history and development of a range of computational methods
- The mathematical basics of a range of computational methods
- How to apply these computational methods to solve practical real-world problems
- How to apply these computational methods to advance a range of research questions
- How to think critically about these computational methods
- Practical skills of coding in Python and implementing these advanced models

Don’t worry if you are not familiar with Python - the aim of this course is not to test your skills but help you acquire new knowledge. Not every one of the methods I will talk about applies to your career goal or research to the same degree - some may be more relevant, and others may be less. For the ones that are more relevant to your career goal or research, you may want to look beyond the content taught in this class to keep up with the state of the art since computational tools evolve rapidly nowadays. For the ones that are seemingly less relevant to you currently, you are encouraged to an open mind since they may turn out to be relevant to you in the future.

GRADE CRITERIA

- ***Assignments - 100%***

  There will be 12 assignments spread throughout the semester. These assignments are intended to help you practice applying these sophisticated computational methods to solve practical problems through coding in Python with Google Colab. You are not required to have backgrounds in coding or Python, but you should at least reserve 8 hours to complete each assignment. If you do not have any prior experience with coding or Python, you may want to reserve at least 12 hours to complete each assignment. DO NOT WAIT TILL THE LAST MINUTE TO WORK ON YOUR ASSIGNMENT. You will have one week to complete each assignment.

  You will receive credit for completing each of the assignments accurately and on time. You can drop two assignments with the lowest scores out of the 12 assignments for your final grade (10% each for the remaining 10 assignments, 100% in total).

- ***Letter Grade***

  Finally, letter grades will be assigned according to the following percentage scale. Percentage grades will be rounded to the nearest point and the stated letter grade cutoffs will be applied without exception.

  A+ if your numeric score >= 97.00% the top score

  A if your numeric score = 93.00-96.99% of the top score

  A- if your numeric score = 90.00-92.99% of the top score

  B+ if your numeric score = 87.00 - 89.99% of the top score

  B if your numeric score = 83.00 - 86.99% of the top score

  B- if your numeric score = 80.00 - 82.99% of the top score

  C+ if your numeric score = 77.00 - 79.99% of the top score

  C if your numeric score = 73.00 - 76.99% of the top score

  C- if your numeric score = 70.00 - 72.99% of the top score

  D if your numeric score = 60.00 - 69.99 % of the top score

  F if your numeric score < 60% of the top score

COURSE POLICIES

- ***Academic Integrity***

  All students are expected to adhere to standards of academic integrity. Cheating of any kind on any assignment will not be tolerated, including copying the code from your classmates or asking AI to complete the homework for you. It is disrespectful to your peers and the university. Consequences for academic misconduct may include a failing grade in the course and official action (e.g., academic probation or expulsion) by the University. For more information on Columbia's policies on academic integrity, please see here: http://www.college.columbia.edu/academics/academicintegrity.

- ***Class Materials***

  The lecture slides and Colab Python coding guides will be made available before each class in which they are presented. These materials are intended to help you take notes during the class, they are no substitute for attendance and engagement in the class.

- ***Special Needs***

  If you require special accommodations through the Office of Disability Services (ODS), please let me know as close to the beginning of the semester as possible. More information about registering with ODS can be found here: https://www.health.columbia.edu/services/ods.

- ***Student Wellness***

  Academic life can be exceptionally challenging at times. The intention of this course is to enhance your life, not make it miserable. Please always prioritize your health. If you're ever having a tough time, and it's impacting your ability to fully participate in the course, please reach out, and we can figure something out together. The university also has many health and wellness resources available for students:

  • https://www.health.columbia.edu/content/counseling-and-psychological-services
  • https://blogs.cuit.columbia.edu/nightline/
  • https://universitylife.columbia.edu/student-resources-directory#!#health

- ***Other Requests***

  If you have other requests or questions, please feel free to talk to the instructor before or after the class, through email, attend the office hours, or schedule a private meeting.

COURSE SCHEDULE

- ***Week 1 Analyzing Naturalistic Text I***

One of the most important types of information people encounter in everyday life is text. For instance, people read news reports, books, and scientific papers to gain new knowledge, message one another on social media to maintain connection, and write comments on the internet to express themselves. In this class, you will learn about the history of text analysis and the mathematical basics of natural language processing. We will discuss how the structure and meaning of language can be captured using statistical and computational methods, and how these methods have evolved from simple word counts to more sophisticated vector representations that underlie modern natural language processing models. We will also consider the types of questions about human behavior and cognition that text analysis can help us answer.

- ***Week 2 Analyzing Naturalistic Text II***

  We will continue with analyzing naturalistic text. We will learn about the architecture of more advanced large language models that have been developed in recent years. We will then look at examples of their applications for solving practical psychological problems and advancing psychological research. Finally, we will look at the actual implementation of these models in Python via Google Colab. We will explore how these models represent meaning, generate language, and capture patterns in human communication, and we will begin hands-on work analyzing text data from psychological studies.

- ***Week 3 Analyzing Naturalistic Visual Information I***

  Another important type of information people encounter in everyday life is visual input. For instance, people try to understand one another's emotions and intentions by observing one another's facial expressions and body language. In this class, you will learn about the history of computer vision and the mathematical basics of computer vision models. We will discuss how visual information can be represented as data, how early vision systems extracted features from images, and how these ideas laid the groundwork for modern vision models. We will also examine how visual processing connects to psychological theories of perception and person understanding.

- ***Week 4 Analyzing Naturalistic Visual Information II***

  We will continue with analyzing naturalistic visual information. We will learn about the architecture of more advanced deep convolutional neural networks that have been developed in recent years. We will then look at examples of their applications for solving practical psychological problems and advancing psychological research. Finally, we will look at the actual implementation of these models in Python via Google Colab. This will include working with pre-trained models to extract features from faces and bodies and exploring how these features can be used to study emotional expression, social judgments, and behavioral prediction.

- ***Week 5 Optimization Algorithms I***

After learning about the architecture and applications of models for analyzing naturalistic text and visual information, we now turn to how these models are actually fit to data (how they are trained). In this class, you will learn about optimization algorithms, with a focus on gradient descent—the workhorse behind modern machine learning. We will walk through the mathematical intuition behind gradient descent, explain how it updates model parameters to minimize error, and demonstrate its implementation in Python. This week serves as a foundation for understanding how large-scale models learn from data.

- ***Week 6 Optimization Algorithms II***

  While gradient descent works well for smooth and differentiable loss functions, many problems in psychology and cognitive modeling involve objectives that are non-differentiable, discontinuous, or contain many local optima. In this class, you will learn about alternative optimization algorithms that are better suited for such cases. We will cover simulated annealing, genetic algorithms, and maximum variation strategies. These methods are particularly useful when the solution space is rugged or when the goal is to explore diverse configurations rather than converge on a single optimal solution. You will also learn how to implement these algorithms in Python and apply them to real psychological modeling problems.

- ***Week 7 Evaluating Reliability of Big Data***

  Once we have collected data for our question, an essential step is to assess its reliability. Reliable data provide a stable foundation for drawing meaningful conclusions and designing follow-up studies. In this class, you will learn about several methods for evaluating reliability, including leave-one-out reliability, intraclass correlation coefficient (ICC), split-half reliability, and Cronbach's alpha. We will compare these approaches in terms of what type of consistency they assess—across items, raters, or participants—and when to use each. You will also learn how to use these reliability metrics to inform the planning of future data collection. This includes estimating the number of participants, stimuli, or trials needed for stable results using tools such as jackknife resampling and the Spearman-Brown prediction formula. We will implement these techniques in Python and apply them to real psychological datasets.

- ***Week 8 Analyzing Big Data – Dimensionality Expansion and Reduction***

  Psychological datasets often contain hundreds or thousands of features, ranging from questionnaire responses and neural activity patterns to high-dimensional embeddings of text and visual input. Making sense of this complexity requires tools for reducing dimensionality while preserving meaningful structure. In this class, you will learn about several dimensionality reduction techniques, including exploratory factor analysis, principal component analysis (PCA), Uniform Manifold Approximation and Projection (UMAP), and autoencoders. We will compare their underlying assumptions, strengths, and ideal use cases. In addition, we will explore the idea of dimensionality expansion—cases where transforming the data into a higher-dimensional space (e.g., via network modeling) can reveal latent patterns or relationships that are not easily captured by linear

methods. You will gain hands-on experience using these methods in Python and learn how they can be used to uncover psychological structure from complex data.

- ***Week 9 Analyzing Big Data – Handling Repeated Measures***

  Many psychological datasets involve repeated measures—multiple observations from the same participant, or responses to multiple stimuli nested within higher-level groupings. In this class, we will begin by reviewing the fundamentals of linear regression, including model specification, interpretation of coefficients, and key assumptions. We will then focus on linear mixed-effects models, which allow us to account for both fixed effects (predictors of interest) and random effects (such as participant or stimulus variability). These models are essential when data are hierarchically structured and observations are not independent. You will learn how to specify, fit, and interpret linear mixed models in Python via Colab, and how they can provide more accurate and generalizable inferences in psychological research.

- ***Week 10 Analyzing Big Data – Handling High-dimensional Independent Variables***

  If you want to predict a dependent variable using more independent variables than the number of observations you have, you encounter the problem of overfitting. For instance, in this case, there will be no unique solution to a standard linear regression. In this class, you will learn about ridge regression, a form of regularized regression that addresses overfitting by penalizing large coefficients. We will discuss how ridge regression improves model stability and generalization by trading off bias and variance. You will also learn how to optimize the regularization strength—a key hyperparameter—using cross-validation. By the end of the class, you will understand how to implement ridge regression in Python, apply it to high-dimensional psychological datasets, and interpret its results in the context of psychological theory.

- ***Week 11 Analyzing Big Data – Handling High-dimensional Dependent and Independent Variables I***

  Sometimes you may assess an outcome variable using multiple measures. For instance, to understand an individual's personality, you might collect self-reports on a list of personality traits. A common approach is to average these traits and use the composite score as the dependent variable in a standard regression. However, this approach can obscure the rich variability across individual traits and the nuanced structure within the data. In this class, you will learn about representational similarity analysis (RSA), a powerful method that allows you to retain high-dimensional outcome variables by comparing patterns of similarity rather than reducing them to a single score. We will explore how to construct and interpret representational dissimilarity matrices (RDMs), apply RSA to both predictors and outcomes, and test the significance of these associations using nonparametric permutation tests. You will also gain hands-on experience implementing RSA in Python, and see how this framework can be applied to behavioral, neural, and perceptual data to uncover deep structure in psychological representations.

- ***Week 12 Analyzing Big Data – Handling High-dimensional Dependent and Independent Variables II***

  Building on last week's focus on high-dimensional outcomes, this class introduces artificial neural networks as a flexible modeling approach for simultaneously handling high-dimensional dependent and independent variables. Unlike traditional regression methods, neural networks such as multi-layer perceptrons (MLPs) can model complex, nonlinear relationships across vast numbers of input and output features. In this class, you will learn about the architecture of feedforward neural networks, how information flows through layers of nodes, and how weights are learned through backpropagation. We will walk through how to design, train, and evaluate neural networks using real psychological datasets where the goal is to capture rich multivariate mappings—for example, predicting full behavioral profiles from image or text embeddings. You will also learn how to prevent overfitting in neural networks using techniques such as dropout and early stopping, and how these models can provide new insights into complex psychological processes.

- ***Week 13 Analyzing Big Data - Analyzing Big Data – Handling Time-Series Data***

  So far, you have learned about several methods that can handle high-dimensional psychological data. However, these methods generally assume that each observation is independent of the others. In many real-world scenarios, this assumption does not hold. Psychological and behavioral data often unfold over time—for example, physiological responses during a conversation, eye-tracking patterns across a screen, or sequences of words in a sentence. In such cases, the value of a given observation depends on those that came before it. In this class, you will learn about how to model such temporal dependencies using artificial neural networks that incorporate memory, focusing on the Long Short-Term Memory (LSTM) architecture. LSTMs are a type of recurrent neural network (RNN) designed to retain and update information over long sequences, making them well-suited for time-series and sequential data. You will explore how LSTMs work, how they differ from standard feedforward models, and how to implement and train them in Python. We will also discuss practical applications in psychology, such as modeling emotion dynamics, behavioral trajectories, and language processing.