Syllabus for
CSCI 3352, Biological Networks
Fall 2024

Lectures: Tuesday and Thursday from 1:00–2:15pm in JSCBB B231 (as needed: Zoomland)

Lecturer: Aaron Clauset

Email	aaron.clauset
Zoom Office Hours	Tuesday 2:30–3:30pm and Thursday 4:00pm–5:00pm or by appointment
Course Web Page	Canvas
Q&A Forum	Piazza

Description: This course examines the computational representation and analysis of biological phenomena through the structure and dynamics of networks, from molecules to species. Attention focuses on algorithms for clustering network structures, predicting missing information, modeling flows, regulation, and spreading-process dynamics, examining the evolution of network structure, and developing intuition for how network structure and dynamics relate to biological phenomena.

Prerequisites: Data structures (CSCI 2270) or similar, Data science (CSCI 3022) or similar, Calculus I (or equivalent); programming competence is assumed

Learning Objectives: In this course, students will

- understand the representation and analysis of biological phenomena as networks
- develop intuition for how network structure shapes dynamics
- learn principles and methods for describing and clustering network data
- learn to predict missing network information
- understand and simulate models of biological network dynamics
- understand and model the evolution of biological networks
- analyze real-world biological network data

Overview:

- lectures 2 times a week, in-person (Tu/Th)
- problem sets throughout the semester (weekly, 10 total)
- problems will utilize programming, statistics, and mathematics
- a class project, presentation, and final report
- this will be a challenging and fun course; plan accordingly

Optional Texts: *Networks* (2nd ed.) by Mark Newman, and
Introduction to Systems Biology by Uri Alon

Schedule

Week 1	Fundamentals of networks	—
Week 2	Network representations and statistics	PS1 due
Week 3	Random graph models	PS2 due
Week 4	Predicting missing links and attributes	PS3 due
Week 5	Modular networks I: structure	PS4 due
Week 6	Modular networks II: inference	PS5 due
Week 7	Epidemiology I: spreading processes	project proposal due
Week 8	Epidemiology II: simulations	PS6 due
Week 9	Proteins & genes I: structure	PS7 due
Week 10	Proteins & genes II: models	PS8 due
Week 11	Metabolism I: structure	—
Week 12	Metabolism II: dynamics	PS9 due
Week 13	Ethics in computational biology	PS10 due
Week 14	Fall break (no class)	—
Week 15	Project presentations	—
Week 16	Project presentations	—
Week 17	Finals (no class)	Project report due

Assignments & Deadlines

There are 10 weekly problem sets
Problem sets are always due on Friday, by 11:55pm MT
Problem set files will be provided through the class Canvas page
All submitted work (problem sets and projects) is via Canvas

Class Project

Project proposals are due Friday, October 11, by 11:55pm MT (Week 7)
Project presentations are in class, December 3–12 (Weeks 15–16)
Final project report due Monday, December 16 by 2:00pm MT

Examinations

There will no exams in this class

Getting help

- **Attend the lectures and come to office hours.** Lectures are where all material will be introduced. Office hours are for any student wanting clarification, advice, support, feedback, or just a conversation. (If no one shows up, I do email or something for an hour, which is way more boring than talking to students about science and networks.)
- **Use the Canvas page**. All class materials, including lecture notes, supplementary videos, problem sets, and submissions will be via the class Canvas. Grades will not be tracked there; you are responsible for tracking your own progress in the course.
- **Use the Piazza forum**. Asynchronous Q&A will be done on the class Piazza, which is intended to help you get help from classmates and myself. Rather than emailing questions to me, please post your questions on our Piazza forum.
- **Email is for urgent matters**. Don't ask for help with specific parts of problem sets via email; use office hours or Piazza for those questions. Please reserve email for high-priority personal matters.

Grading

Grades will be assigned as a weighted sum of attendance and class/Piazza discussion (20%), problem set scores (40%), and class project proposal, presentation, and report (40%).[1]

Letter grades will only be calculated and assigned after the project reports are scored. Prior to that, only numerical scores will be tracked. There may be extra credit on the problem sets; doing the extra credit can only increase your final grade.

Class project

- In the class project, students will explore a class topic (of their choice) more deeply.
- It may be done individually or in a team of two (2). If you choose to work with another student, you are responsible for finding that person and for evenly dividing the work.
- There are three (3) deliverables associated with the class project:
  1. *Class project proposal* (due Week 7, via Canvas, PDF only)
     Must include: (1) the names of the individual(s) on the team, (2) three paragraphs describing the (i) background material, (ii) research question, and (iii) anticipated findings, and (3) a brief description of any data and algorithms you plan to use.

[1] Why is attendance required? Because: the scientific literature, e.g., here and here, shows compelling evidence that attending in-person improves your learning, and I want to maximize your learning this semester.

2. *Project presentation* (due Weeks 14–15, in class)
   A roughly 10 minute-ish presentation of the project idea and any results found.
3. *Project paper* (due Week 16, via Canvas, PDF only)
   A 6 page scientific report on the project idea, results and what you learned. The report should be formatted like a scientific paper (11pt font, 2-column, 1-inch margins), with Introduction, Methods, Results, Discussion, and Bibliography sections, with an optional Appendix section for code.[2] The Bibliography should include appropriate references for your work (primary literature, course materials, websites, source data, methods, etc.). If you would like advice about how to properly write any of these sections, I'm happy to provide it.
   Warning: Do not plagiarize any part of the report. Suspected plagiarism will receive a 0.

Advice: Try to choose a project that you can complete in 8 weeks, and which takes about 4× the time you spend on one problem set in the course. Here are four examples of projects:

1. Reproduce the results of some paper on biological networks in the scientific literature.
2. Choose an empirical biological network and then interpret the outcome of applying a variety of methods from the class.
3. Contribute a novel implementation of methods from class to `networkx` or `igraph`.
4. Make up your own idea!

Two key requirements of the project are that (1) it involves networks, and (2) it's about biology. I'm happy to provide feedback on your ideas *before the project proposal deadline*—my office hours are an excellent time to chat about it. Note: if your proposal is not well scoped, I may ask you to revise it.

Warning: you are responsible for pacing yourself on your project; the class has no "progress report" deadlines midway through the project; plan accordingly.

Problem sets

- Weekly deadline : Friday, by 11:55pm MT, via the class Canvas page
  No credit for solutions submitted any other way
  Late submissions will not be accepted
  The lowest 2 problem set grades will be dropped at the end of the semester

[2] Bibliography and Appendix sections do not count for or against your page limit. A link to a `github` repo for code is okay, but you must also include your code in the PDF, unless you request and obtain from me permission not to.

- We will use Python and Jupyter notebooks for all programming parts of assignments. Prior experience with these is not necessary, but will be useful.
- For each assignment with programming questions, all of your programming solutions must be submitted in a single Jupyter notebook. That notebook (1) must be executable without errors from top to bottom and (2) must include all required outputs. In other words, please run all code top-to-bottom before submitting, but please also ensure that code runs without errors; I may rerun snippets to check output.
- You may use standard libraries, e.g., in Python: `networkx`, `igraph`, `numpy`, `scipy`, `webweb`, as needed. If you're not sure about a library, ask.
- Solutions to non-programming questions must be included inside the Jupyter notebook you submit.
- Partial credit will be awarded; to maximize your chances of receiving some partial credit, show your work and explain your thinking.

Working in teams

- Problem sets are to be completed individually.
  The class project may be completed in a team of 1 or 2.
- Students are *encouraged* to form study groups to discuss the problems with each other.
- **How to avoid cheating?** Simple: each student must *independently* write up their *own* solution—the one they submit for credit for themselves. Copying any part (words or code) is strictly forbidden, and submitting lightly edited work produced by someone other than yourself, e.g., changing a few words or variable names, is dishonest. Instead, work together at a whiteboard or on paper, but then, even if you solved the problem together, you must write up and submit your own version (your own words!) of it.

Intellectual Honesty and Plagiarism

- Intellectual dishonesty or plagiarism of any form, at any level, *will not be tolerated.*
- **Discussing questions with other students is encouraged, but** you must list your collaboration in the solutions you submit. If you discussed it with 10 other people, then all 10 names should appear in your solution. If someone was particularly helpful, say so. Be generous! If you're not sure whether someone should be included in your list of collaborators, include them. For discussions in class, in section, or in office hours, where collecting names is impractical, it's okay to write something like "discussions in class." There is never a penalty for discussing problems with other students.

- **Copying from any source, in any way, is strictly forbidden**. This includes the Web, chatGPT or other generative AI systems, and other students (past or present). Asking or paying someone on the internet for answers is dishonest. If you are unsure about whether something is permitted, I'm happy to talk over what you're thinking about.

  A note about generative AI: OpenAI's `chatGPT` and other LLMs are "premium bullshit" generators that don't actually know anything and cannot reason like you can. But they are very good at stringing together and reshaping bits of text or code from their training data, into verbally pleasing or syntactically correct output. If interacting with one helps you learn the course material or figure out how to do the work correctly, that's great and permissible! But, under no circumstances should you submit LLM answers (text or code), in whole or in part, as something you yourself wrote. Because: they make mistakes, over-confidently, and it's your responsibility as a user to know the difference between bullshit and a correct answer. Moreover, for scientific topics like biological networks, its training data doesn't include examples of the kind of programming tasks or scientific interpretation I'll ask you to do in this class.

  There will be a zero-tolerance policy to violations of this policy. Violators will be removed from the class, given a grade of F, and reported to the CU Honor Council.

- **Write everything in your own words and cite all outside resources**. You are strongly encouraged to use outside resources, e.g., the Internet, other textbooks, the primary literature, etc.[3] to teach yourself how to solve any problem. But, you must write your solutions *in your own words*. I'm not interested in seeing Wikipedia's answer, or chatGPT's answer, or anyone else's, in whole or in part. The only sources you are not required to cite are the course materials. As a small bonus for having read the syllabus carefully, I will award some extra credit points if, within the first two weeks of class, you send me an email containing a picture of your favorite dinosaur species (any kind is fine).

Formatting and submitting your problem set solutions

Rule 1 **Canvas only**: All submissions must be made electronically, via the class Canvas.

Rule 2 **One file**: Submit a Jupyter notebook `.ipynb` containing your complete solutions. Do not submit files with formats other than `.ipynb`.

[3] This is the 21st century after all. It would be madness not to encourage you to use the greatest tool ever created for scholarship. But, with great power comes great responsibility. Use it wisely, you must.

Rule 3 **Include your name**: Inside any file you submit, include *your name* and *which problem set* these are solutions for.[4] Better yet, fill out and use the template provided.

Rule 4 **Figures**: Any figure, chart, table, etc., in any file you submit, must be fully labeled. *No credit for figures with unlabeled axes or data series.*

- Clearly indicate the question number each particular solution answers. No credit will be given for solutions that are not clearly labeled.
- **Solutions should be detailed and clear.** Each of your answers should include a clear, written explanation of how it answers the question. Code only, without a written explanation of what it is supposed to do and how, is almost never sufficient for full credit. The more clear the explanation, the more likely you are to receive full credit. Detailed advice for achieving this is included later on in this document.
- **Exceptional circumstances**: Only in exceptional circumstances, e.g., incapacitation due to illness or injury, may assignments be forgiven. *Unexceptional* circumstances include registering late, travel for job interviews or conferences or fun, forgetting the homework deadline, or simply not finishing on time. Final grades will be computed as if forgiven assignments did not exist; this increases other assignments' weight.

What to write

- **Answer the right question**.
- **Comment your code**. Writing code that I can clearly understand *requires* that you include comments. The more the better. Write a whole paragraph if you want. Annotate the tricky parts. Your goal is to make it easy to see that you have the correct solution.
- **Justify your answer**. Unless the problem specifically says otherwise, every solution you provide must include a written explanation of its correctness. Without one, even a perfectly correct solution is worth nothing. In particular, the sentence "It's obvious!" is not an explanation—"obvious" is often a synonym for "false"!
- **Answer the question completely**. Once you've written your answer, re-read the question fully and check that you have answered all parts of it, thoroughly. When a question asks you to "show" something, you need to provided evidence in return, and then write out in words a justification for why your evidence answers the question.

[4] Imagine that I will print out your solutions; if the printout doesn't have a name on it...

Advice for writing up your solutions

The most important thing you can do is to make it easy for me to figure out what you mean within the short time I have to grade your solution. Solutions that are difficult to read or understand will cause you to lose points.

- **Write carefully**. You will only be graded on what you write (words or code), not what you mean. I cannot read your mind. If your solution is ambiguous, I may choose the interpretation that makes the solution wrong (as in worst-case analysis).

- **Write clearly**. If I cannot decipher your answer quickly, I cannot give you credit for it, even if it is correct. Write your answers clearly. Separate them from other problems. Box or otherwise clearly indicate your results. Solve problems in the order they are listed.

- **Explain your solution**. A complete solution nearly always requires a written explanation of how your answer solves the question. Code can provide precision where English cannot not, but English provides context and interpretability that code does not. Both can be necessary for a high-quality solution.

- **Make it understandable**. Start by describing the key ideas, insights, or assumptions behind your approach to solving the problem before going into its details. If you do this well, I am more likely to understand your approach without looking at details you include later. A clearly written solution that includes all the main ideas but omits some low-level details is worth more than a complete, correct, detailed, but opaque solution.

- **Be succinct**. Your solution should be long enough to convey exactly why your answer is correct, yet short enough to be easily understood.

- **Don't bullshit**. If you don't know the answer, just say so. Answering "I don't know" (and nothing else) is worth 20% (rounded down) partial credit. The "I don't know" rule applies to every problem and sub-problem on every problem set, but not to extra credit problems or any part of the class project. A blank page is worth zero points. Readable, correct, but suboptimal solutions are always worth more than 20%.

- **Numerical experiments**: Many of the programming problems will require you to conduct a numerical experiment. In addition to showing your results, you must describe your experimental design, i.e., how you set up the experiment, what it should show, and what you observe about the outcome.

  For example, suppose I ask you to show that an algorithm takes $T(n) = O(n \log n)$

time. First, choose a wide range of values of n, and then measure the number of atomic operation $T(n)$ for each value of n. Plot this set of (x, y) coordinates. To show that this (x, y) scatter follows the predicted pattern, overlay a continuous line of the form $T(n) = c\,n \log n$ for some choice of c that makes the line go "through" the data.

Pro tip 1: Many of the problem set experiments will involve randomness. In these cases, you'll need to make multiple measurements of $T(n)$ for a given n. Then, plot the *average* measured value $\langle T(n) \rangle$, to get a smoother, clearer figure.
Pro tip 2: How wide a range of n? Use a dozen or so exponentially spaced values of n, e.g., $n = \{2^4, 2^5, \ldots, 2^{16}, \ldots\}$.

Computing
A central learning goal of this course is to master network analysis and modeling concepts and methods, both their theory and their practice.

- Our main tool supporting this effort will be Python 3 with Jupyter Notebooks. It is estimated that around 50% of practicing data scientists do most of their analysis using Python (while the other roughly 50% use R; the use of R in this course is not forbidden, but if you plan on using R Markdown, you must obtain permission from me first; I do not have R installed on my computer). The Jupyter Notebook is a browser-based programming environment that allows you to seamlessly mix Python code (as well as many other languages), graphics, and exposition (including LaTeX, in Markdown). The Jupyter Notebook has become ubiquitous in the data science community for rapidly prototyping ideas and sharing them with colleagues and the rest of the world.

- It is strongly recommended that you install Python 3 and Jupyter on your local machine. By far, the easiest way to do this is by installing the Anaconda distribution of Python 3.6. This distribution comes with many Python packages useful for data science and scientific computation in general. It also comes with Jupyter by default.

- Lacking a local installation, you may be able to use the Jupyter Hub platform in the CS department, which is accessible at `coding.csel.io`.

Suggestions: Suggestions for improvement are welcome at any time. Any concern about the course should be brought first to my attention. Further recourse is available through the office of the Department Chair or one of the Academic Advisors, all accessible on the 7th floor of the Engineering Center Office Tower.

Honor Code: As members of the CU academic community, we are all bound by the CU Honor Code. I take the Honor Code very seriously, and I expect that you will, too. Any significant violation will result in a failing grade for the course and will be reported. Here is the University's statement about the matter:

All students enrolled in a University of Colorado Boulder course are responsible for knowing and adhering to the Honor Code. Violations of the Honor Code may include but are not limited to: plagiarism (including use of paper writing services or technology [such as essay bots]), cheating, fabrication, lying, bribery, threat, unauthorized access to academic materials, clicker fraud, submitting the same or similar work in more than one course without permission from all course instructors involved, and aiding academic dishonesty. Understanding the course's syllabus is a vital part in adhering to the Honor Code.

All incidents of academic misconduct will be reported to Student Conduct & Conflict Resolution: StudentConduct@colorado.edu. Students found responsible for violating the Honor Code will be assigned resolution outcomes from the Student Conduct & Conflict Resolution as well as be subject to academic sanctions from the faculty member. Additional information regarding the Honor Code academic integrity policy can be found on the Honor Code website.

Accommodations for Disabilities: If you qualify for accommodations because of a disability, please submit your accommodation letter from Disability Services to your faculty member in a timely manner so that your needs can be addressed. Disability Services determines accommodations based on documented disabilities in the academic environment. Information on requesting accommodations is located on the Disability Services website. Contact Disability Services at 303-492-8671 or dsinfo@colorado.edu for further assistance. If you have a temporary medical condition, see Temporary Medical Conditions on the Disability Services website.

If you have a temporary illness, injury or required medical isolation for which you require adjustment, please send me email directly.

Preferred Student Names and Pronouns: CU Boulder recognizes that students' legal information doesn't always align with how they identify. Students may update their preferred names and pronouns via the student portal; those preferred names and pronouns are listed on instructors' class rosters. In the absence of such updates, the name that appears on the class roster is the student's legal name.

Classroom Behavior: Students and faculty are responsible for maintaining an appropriate learning environment in all instructional settings, whether in person, remote, or online. Failure to adhere to such behavioral standards may be subject to discipline. Professional courtesy and sensitivity are especially important with respect to individuals and topics dealing with race, color, national origin, sex, pregnancy, age, disability, creed, religion, sexual orientation, gender identity, gender expression, veteran status, marital status, political affiliation, or political philosophy.

For more information, see the classroom behavior policy, the Student Code of Conduct, and The Office of Institutional Equity and Compliance.

Sexual Misconduct, Discrimination and Harassment and/or Related Retaliation: CU Boulder is committed to fostering an inclusive and welcoming learning, working, and living environment. University policy prohibits protected-class discrimination and harassment, sexual misconduct (harassment, exploitation, and assault), intimate partner abuse (dating or domestic violence), stalking, and related retaliation by or against members of our community on- and off-campus. The Office of Institutional Equity and Compliance (OIEC) addresses these concerns, and individuals who have been subjected to misconduct can contact OIEC at 303-492-2127 or email cureport@colorado.edu. Information about university policies, reporting options, and support resources can be found on the OIEC website.

Please know that faculty and graduate instructors must inform OIEC when they are made aware of incidents related to these policies regardless of when or where something occurred. This is to ensure that individuals impacted receive outreach from OIEC about resolution options and support resources. To learn more about reporting and support for a variety of concerns, visit Dont Ignore It.

Religious Accommodations: Campus policy requires faculty to provide reasonable accommodations for students who, because of religious obligations, have conflicts with scheduled exams, assignments or required attendance. Please communicate the need for a religious accommodation in a timely manner. In this class, I will make reasonable efforts to accommodate such needs if you notify me of their specific nature by the end of the 3rd week of class. See the campus policy regarding religious observances for full details.

Mental Health and Wellness: The University of Colorado Boulder is committed to the well-being of all students. If you are struggling with personal stressors, mental health or substance use concerns that are impacting academic or daily life, please contact Counseling and Psychiatric Services (CAPS) located in C4C or call (303) 492-2277, 24/7.

Free and unlimited telehealth is also available through Academic Live Care. The Academic Live Care site also provides information about additional wellness services on campus that are available to students.