GCB537 Advanced Computational Biology

Term: Spring 2023
Instructor: Yoseph Barash & Noam Auslander
Objectives:

1. Review important concepts for computer science and statistics as they apply to computational biology
2. Discuss current topics and related papers in genomics and computational biology
3. Learn to evaluate, criticize, and summarize research papers in genomics and computational biology
4. Experiment, evaluate, and try to improve tools/algorithms from topics covered in the course.

Requirement: Background in statistics, biology, genetics and genomics, and computer science.
NOTE:

1. **This is NOT a bioinformatics lab.**
2. **None-GCB students need to be approved by the instructors.**

Time and Location: Tuesdays and Thursdays 1:45-3:15pm, 252 BRB

Course format:

The course consists of lectures, paper discussions, and final project presentation. Some lectures review current topics in computational biology, while others review material in computer science and statistical modeling relevant for these topics and compbio in general. In paper discussion classes, papers are selected to match topics discussed in the review lectures, with emphasis on algorithm design and best practice for data interpretation and presentation. In parallel, students will be given coding and analysis assignments that correspond to topics and/tools discussed in class. Students will be divided into small teams that will pick a project matching a compbio topic, perform analysis for it, and present their results as the final project in the course. Teams will present their project and submit a written summary of it at the end of the semester.

Paper discussion.

1. The course is divided into units covering current topics in comp bio research
   Each unit starts with a review lecture, followed by one or more paper discussion classes led by the instructors or students.
2. Students will be divided into teams for each of the units.
3. For student-led paper discussion courses, one team will be designated to present the paper. Emphasis will be given to understanding the computational methods, model assumptions, evaluation process, overall significance and open issues/directions. To ensure the quality of the presentation, send PowerPoint files to the instructor or discuss with the instructor *at least two days before the scheduled presentation*.
4. The leading team will also submit questions on the paper to the instructors two days

before the discussion. The question list will be circulated to the entire class before the class. Students will submit their anonymous responses and the presenters will grade those responses, together with the instructors.

5. After the class, all other students at the presentation will send a grade (between 1=unprepared and 5=excellent) or any constructive comments to the instructor by email *by end of the day*. Comments will be forwarded to the presenters anonymously.

Coding/analysis assignments:

As the course progresses students will be given analysis tasks that involve coding. The analysis tasks will relate to topics covered in class. Some of those tasks may be over predefined data/tools and some may be more open ended ("bring your data" and/or "come up with a scientific question and matching analysis for a given/chosen dataset"). Learning proper scientific coding will be implicitly included in these tasks while a major focus will be on the analysis of the data. Some of the tasks will be interconnected and build upon each other.

Final project:

The students will team up (2-3 students per team) and:

1. Work with the instructors to select a specific topic to evaluate tools on a particular bioinformatic/computational topic *by end of January.* Preference would be given to extending in a topic already presented in the course or related ones. Other topics may be approved as well on a special case basis if there is a strong drive from a specific group.
2. Submit a 1-2 page proposal for final presentation, including topics, list of papers, or plans for experimental comparison by end of February. Discuss with the instructors.
3. Make a 25-minute presentation towards the end of the semester.
4. To ensure quality of the presentation, send the instructor your PowerPoint slides or meet with the instructor *at least two days before the two scheduled presentations*.
5. After the class, all other students at the presentation will send a grade (between 1=unprepared and 5=excellent) or any constructive comments to the instructor by email by end of the day; comments will be forwarded to the presenters anonymously.
6. Submit a summary report that includes: Background, the main issues they addressed/evaluated, their results, and conclusions. Summary is no longer than 4 pages 12pt font, 1' margin.

Course evaluation:

Evaluation will be a combination of class participation, paper presentation, answers to papers' questions, coding/analysis assignments and the final project.

Tentative list of topics:

Efficient coding in genomics (TA reviews): Scripting, unix text editors, reproducible coding,

numerical stability, efficient coding using matrix manipulations.
Optimization methods - (Stochastic) Gradient Descent, generalized EM.
Probabilistic view of regression
Clustering: supervised/unsupervised, probabilistic models (Naive Bayes, GMM), spectral
Dimensionality reduction methods
Ensemble learning: Boosting vs Bagging, Decision trees, Random forest vs Tree boosting.
Current computational topics in human genetics: GWAS, QTL, Fine Mapping, Colocalization.
DL models for genomics: CNN, (V)AE, Transformers.
DL optimization and interpretation for genomics.
RNA Sequencing (Expression and Splicing)
Peak Calling (ChiP/ATAC/CLIP etc.)
Sequence Motif Finding - From traditional ML to DL
Measures of performance, class imbalance and overfitting with biological data
Feature selection: filter, wrapper, and embedded methods. Utility and caveats
Multi omics integration
Molecular evolution: phylogenetics and selection
Assembly: naïve, string graph assembly, de Bruijn graph-based (briefly)
Cancer genomics and ML/DL in cancer research
Missing data and data censoring. Data imputation, semi supervised and PU learning