School of Science, Technology, Engineering, and Mathematics

UNIVERSITY *of* WASHINGTON
BOTHELL

Course Title: Computational Biology Year: **2018** Quarter: **Winter**
Instructor: Dr. Jesse Zaneveld
Course Time: TTh 8:45-10:45
Course location: Disc 258
Instructor Contact:
Email: zaneveld@uw.edu
Phone: [Please use e-mail]
Office Hours: 11:00 – 12:00 Thurs & by appointment
Office Hours location: UWBB 249

COURSE DESCRIPTION

Biological research is increasing defined by the availability of very large datasets generated by high-throughput DNA sequencing, mass spectroscopy, etc. Using computational tools to work with these datasets is both practically important and a lot of fun. This course is designed to give life science students a chance to develop confidence in using programming to help with their research. Programming is a skill that doesn't require any special innate talent, but does require deliberate practice. As you learn, you can use these skills to simplify problems that you encounter in your coursework, undergraduate research, and professional work.

Because practice is so important, this class emphasizes student-directed projects that work with real data and solve real problems. The goal is to make these as close to real research as possible in a limited time. In keeping with this project-based approach, there will be short daily quizzes to make sure everyone is getting the material. However there will be no midterm or final exams (but we will still meet during finals week for Project 2 presentations).

COURSE LEARNING GOALS & OBJECTIVES

By the end of this course, students will be able to:

- Work collaboratively with a team to translate real-world problems and data into manageable computational tasks. Some specific types of biological data we'll work with will include: DNA sequence composition, microbial community ecology, and phylogenetic trees.
- Creatively apply scientific programming in the Python language to help with research tasks. For example, following this course, students should be able, with some time and effort, write a script to: extract relevant information from a large, tab-delimited text file; apply measures of microbial community diversity to a table summarizing DNA sequence data; or calculate GC content and other nucleotide composition measures from a FASTA-format file of DNA sequences.
- Employ best practices for developing software as part of an inclusive scientific community. These include writing readable and modular code; using code documentation; using a coding standard; pair programming; and conducting code reviews to get constructive feedback on your work

- Write test code to ensure that your software is reliable and robust.
- Understand enough computational biology fundamentals to explore new methods on your own.

What's *not* expected**:**

- I don't expect that you have a special talent for coding, or that you've taken a programming course. Instead expect that this is just a skill – like driving a car or speaking French – that you can pick up with some basic guidelines and a lot of deliberate practice.
- I don't expect, and you shouldn't expect, that you will be able to develop code without running into error messages. Instead expect that you'll get lots of errors, but over time, you'll learn what they mean, and how to use them to quickly correct problems with your code.
- No course can make you an expert Computational Biologist in 10 weeks – instead expect to learn some fundamental skills that you can use independently to continue to improve over time on your own. To do that, you'll have to keep finding opportunities to use coding in your class projects, work or play; keep looking up new methods and new libraries online; and share your code with others frequently.

PREREQUISITES

Introductory biology (BBIO 180) is a listed prerequisite for the course. However, if you haven't taken this course and would still like to take the class, please contact me by e-mail and we can likely make arrangements.

REQUIRED TEXT

Python for Biologists. Dr. Martin Jones. 2013

OFFICE HOURS:

One fixed office hour will be held 11:00 – 12:00 Thursday following class in my office. Because everyone's schedule is different, the second 'floating' office hour will be by appointment. For coding questions, please bring code you are working on (on a laptop, by e-mail etc). It is usually much easier to answer questions and suggest revisions when looking at a specific script than in the abstract.

TECHNOLOGY ACCESS

In this class we're going to take a hands-on approach to learning computational biology skills. This necessarily involves access to a functioning computer- but if you don't have one don't panic. There are several campus resources available that can get you the access you need to do the course. ***That said, the quarter is very (very) short, and it is your responsibility to get in touch as soon as possible (within the first week) to work to resolve the situation.*** Similarly, if you have a disability that requires special accommodations (e.g. screen reader, etc) please get in touch sooner rather than later so that we can make arrangements (see Disability services info below).

COURSE SCHEDULE

	Topic	*Biological Topic*	*Computational Topics*	*Activities & Due dates*	
Week 1 *Week of 1/1* *Th*	**Intro to Computational Biology**	Overview of Computational Biology		Types of sequence variation	
Week 2 *Week of 1/8* *Tu*	**Life in 1D:** Biological Sequences	Relating molecular biology to sequence; DNA sequence composition	Basic commandline operations; Setting up Python; Intro to data structures in Python (strings)	Commandline Treasure Hunt A first python script: Counting nucleotide content	**Reading & Exercises:** Python for Biologists, Chapter 1 (intro) & 2 (strings)
Th	**The Jungle Within**: Microbial Ecology & the Human Microbiome	The human microbiome; microbial community analysis workflows; Alpha and Beta Diversity;	Translate real problem into code; File input/output; Python (strings)	Problemset 1 assigned Patterns of Microbial Community Change Exercise	**Reading & Exercises:** Python for Biologists, **Chapter 3** (files) & **Chapter 4** (lists)
Week 3 *Week of 1/15* *Tu*		Alpha and Beta diversity (cont'd)	Writing functions program logic (for loops, if statements); functions	***Project 1 proposals**	**Reading:** Python for Biologists, **Chapter 5** (functions) & **Chapter 6** (if statements)

Th	**Comparative Genomics**	Gene finding; gene function annotation.	Looking up gene function annotations	***Problem Set 1 Due**	**Chapter 8:** Dictionaries **Optional:** Chapter 7: Regular Expressions
Week 4 *Week of 1/22* *Tu*	**The Visual Display of Quantitative Information**	Displaying quantitative information; manuscript figures	Graphing in Python with numpy and matplotlib;	***Project 1 update**. **Exercise:** Fix this graph!	**Reading:** Effective graphs (Tufte); Approaches to graphing (Canvas)
Th	**The Hotel Monte Carlo**	Probability and statistics review	Monte Carlo methods (aka how to cheat at stats)		**Reading:** Probability Paradoxes (Norvig);
Week 5 *Week of 1/29* *Tu*	**BLAST'd things: Search & local alignment**	Sequence evolution; homology	Alignment algorithms; Revising code: modularity, DRY, coding to a standard		
Th		Sequence evolution (cont'd)	Alignment methods (cont'd)	***Project 1 due**	
Week 6 *Week of 2/5* *Tu*	**Phylogenetic Trees: *The Origin of Species'* Only Figure**	Reading phylogenies; Tree inference; Ancestral state reconstruction with parsimony	Using classes in python; Object-oriented programming;	Problemset 2 assigned	Monster phylogeny
Th			Intro to tree inference methods (ML, Bayesian); model selection	***Project 2 Proposals**	
Week 7 *Week of*	**Horizontal Gene Transfer**	Horizontal gene transfer in evolution; Gene	Compositional methods for detecting gene transfer	***Problemset 2 Due**	

2/12 *Tu*		transfer mechanisms;		**Python Library Presentation (Group 1)**	
Th		Core and variable genomes; Phylogenetic gene transfer detection	Python (Sets)	**Python Library Presentation (Group 2)**	
Week 8 *Week of 2/19* *Tu*	**Biological Networks**	Species interaction networks	Visualizing interaction networks; network properties	***Project 2 Update** **Python Library Presentation (Group 3)**	How to always win 6 degrees to Kevin Bacon (IMDB)
Th		Gene regulatory networks; Metabolic networks	Types of metabolic models; exploring seed compound metabolic models	**Python Library Presentation (Group 4)**	
Week 9 *Week of 2/26* *Tu*	**A multivariate toolbox**	Extracting meaning from large datasets	Scientific python with scikit-learn. Clustering methods (e.g. k-means clustering)		
Th		Importance of standardized metadata	Ordination methods (PCoA)	***Submit code for peer code review**	
Week 10 *Week of 3/5* *Tu*	Polishing & Publishing Computational Biology Projects	Reproducibility; Validation; "Gold standard" datasets; Test code revisited	Strategies for collaboration; Jupyter/iPython notebooks; GitHub;	***Peer code review** (bring Project 2 code draft)	
Week *Th*	Presenting computational biology work	Revising scientific writing;	Collaborative writing & editing; Documentation; User support;	***Peer editing** (bring Project 2 draft)	

Finals Week *Tu*	Project 2 Celebration 3/13	Project 2 Due; Project 2 presentations

Grading Criteria:
A major goal of this course is to give you a chance to develop applied computational biology skills that you can use in future work. Therefore, there <u>will not be midterms or final exams.</u> Instead, we will have two problemsets that cover basic python coding and commandline operations, with most points based on two projects (plus associated updates & presentations). Because the emphasis of the class is on developing a polished project, I reserve the right to adjust your grade *upward* (but never down) if your final project score is much higher than other work in the course.

Work	**#**	**points**	**total points**	**Percent**
Problemsets	2	50	100	0.25
Project 1 Proposal	1	15	15	0.04
Project 1 Update	1	15	15	0.04
Python Library Presentation & Responses	1	15	15	0.04
Project 1 Final	1	50	50	0.12
Project 2 Proposal	1	15	15	0.04
Peer Code Review	1	15	15	0.04
Project 2 Final	1	150	150	0.37
Project 2 Presentation	1	30	30	0.07
		Total	405	1

Attendance Policy: Late arrivals interrupt our in-progress activities and discussions. If you must miss a class session, let the instructor know as soon as possible so that you can make up the work that you miss.

Late policy: Assignments should be returned *before class* on the due date via Canvas. In case of technical difficulty with uploads (only) you may turn in assignments by e-mail. Grades for late work will be reduced by 5% per day.

Technology in the Classroom: You are welcome and encourages to use laptops/phones to look up coding methods etc (but see notes on academic integrity below). Please turn volume off, though, so everyone can focus.

Incompletes: STEM strongly discourages incompletes, and incompletes will not generally be given. University rules state that "an incomplete is given only when the student has been in attendance and has done satisfactory work until within two weeks at the end of the quarter and has furnished proof satisfactory to the instructor that the work cannot be completed because of illness or other circumstances beyond the student's control."

Academic integrity:

General UWB information: See http://www.uwb.edu/academic/policies/faculty-guide for crucial information regarding academic integrity. The library also has an extremely useful website with resources at http://libguides.uwb.edu/ai. Students are responsible for knowing what constitutes a violation of the University of Washington Student Code, and they will be held responsible for any such violations whether it was intentional or not. Plagiarism is one of the most common violations of academic integrity, so please pay attention to both the web information and listen when your instructor explains all of this in class. Some resources on plagiarism are available here: http://www.uwb.edu/learningtech/plagiarism

Course Specific information: In this course, you will likely use many online tutorials to learn material. When practicing a new method or exploring a new library (but not in the projects), this will likely involve a bit of typing out working code from e.g. an online tutorial and tinkering with it to understand how it works. This is just fine for personal practice. However, please never copy and paste external code into your project – this is a serious problem and I have personally seen workers expelled from academic labs over injection of plagiarized blocks of code into academic projects.

For the projects, I would like you to first understand new coding methods by working with them externally (e.g. in an interactive python terminal or a small practice script), and only then write a fresh version of your own for use in the project based on your understanding. If something feels like magic, or you feel afraid to rearrange it, you likely need to spend more time playing with it in a simple test/practice script to understand it better before using it in a project. Additionally, if you used an external resource to understand how to do something, please note it in the comments. A secondary benefit of doing things this way, in addition to academic integrity, is that you will understand your own code better, and everything will be written with a more consistent style.

Similar considerations apply to written work like the project description. Be sure to separate out your writing into an exploration phase where you read and distill external materials, and a second writing phase where you write based on your own understanding (but without direct reference to external materials).

Respect for Diversity:

Diverse backgrounds and experiences are essential to the exchange of viewpoints that is at the heart of university education. At UW Bothell, students are expected to respect individual differences which may include, but are not limited to: age, cultural background, disability, ethnicity, family status, gender presentation, immigration status, national origin, race, religious and political beliefs, sex, sexual orientation, socioeconomic status, and veteran status.

Students seeking support around these issues can find more information and resources at http://www.uwb.edu/diversity.

Disability Resources for Students (DRS)

Access and Accommodations: Your experience in this class is important to us, and it is the policy and practice of the University of Washington to create inclusive and accessible learning environments consistent with federal and state law. If you experience barriers based on disability, please seek a meeting with DRS to discuss and address them. If you have already established accommodations with DRS, please communicate your approved accommodations to your instructor at your earliest convenience so we can discuss your needs in this course.

Disability Resources for Students (DRS) offers resources and coordinates reasonable accommodations for students with disabilities. Reasonable accommodations are established through an interactive process between you, your instructor(s) and DRS. If you have not yet established services through DRS, but have a temporary or permanent disability that requires accommodations (this can include but not limited to; mental health, attention-related, learning, vision, hearing, physical or health impacts), you are welcome to contact DRS at 425.352.5307 rlundborg@uwb.edu.

Wondering how to address faculty?
At UWB we appreciate our ability to interact closely with our students, and we value faculty-student collaboration very highly. It is also important to maintain an appropriately professional relationship, both for professional practice, and as an equity issue. When addressing any faculty member, it is most appropriate to refer to them as Dr. X unless told to do otherwise, where X is the last name of the faculty member in question. Please do not actually refer to all faculty as "Dr. X", as folks might be confused. Personally, I like to be called Dr. Zaneveld.

H1N1 and Other Communicable Diseases Action Steps:
As part of the campus community's shared responsibility for minimizing the possible spread of H1N1 virus and other diseases this year, it is critical that all students are familiar with the symptoms of H1N1 Flu described on the UW Bothell website at

http://www.uwb.edu/flu. Any student or instructor with flu-like symptoms is encouraged to stay at home until at least 24 hours after they no longer have a fever without the use of fever-reducing medications. If you are sick and have an extended absence, please speak with me regarding alternative ways to maintain your progress in your courses. If I am sick and need to cancel class, I will post an announcement on Canvas.

Inclement Weather:
Please check if the campus may be closed due to weather. Information about suspension of operations will be made public and available through the media. Students can learn of campus operations status from the website or by calling the Campus Information Hotline 425.352.3333. We recommend that you sign up with the alert system that will contact you via email or text message if classes are canceled. For more information on the alert process, please see http://www.uwb.edu/alert. Class activities will be rescheduled as needed.

Student Support Services:

Quantitative Skills Center: http://www.uwb.edu/qsc 425-352-3170
Writing and Communication Center: www.uwb.edu/WritingCenter/ 425-352-5253
CSS Tutoring: http://www.uwb.edu/css/advising
IT Helpdesk: IT@uwb.edu , 425-352-3456
Library: http://library.uwb.edu 425-352-5340
Student Success Services: http://www.uwb.edu/studentsuccess 425-352-3776
Career Services: http://www.uwb.edu/careers 425-352-3706
Student Counseling Services: http://www.uwb.edu/studentservices/counseling 425-352-3183