Monday, April 2, 2018

Bi 410/510:
Introduction to Programming for Biologists

Course Overview and Goals

- basic information
- course goals
- grading scheme
- what to expect

Course Information

Instructor	John Conery	`conery@uoregon.edu`
GTF	David Wyrick	`dwyrick@uoregon.edu`

Lectures	Mon, Wed	4:00 — 5:20, 185 Lillis
Lab	Fri	2:00 — 3:50, 129 Huestis

Textbook	*Practical Computing for Biologists*, by Haddock and Dunn `http://practicalcomputing.org`

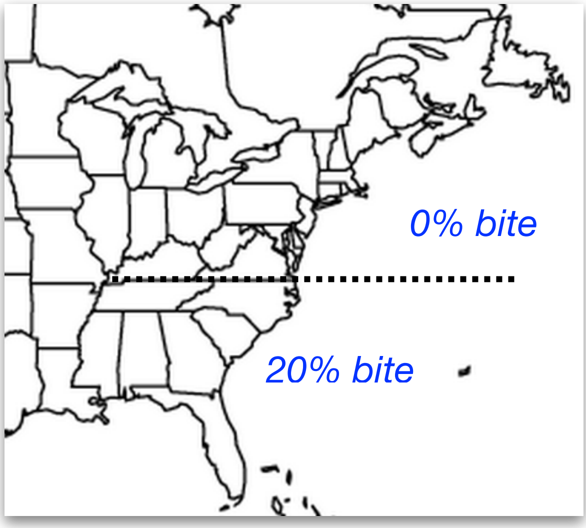
[this information, and more, is in the syllabus on Canvas]

Canvas announcements?

Why Do Biologists Need Computers?

An example: Bradshaw / Holzapfel Lab at UO

They study *Wyeomyia smithii*, a small mosquito that lives exclusively in pitcher plants on the east coast of the US and Canada



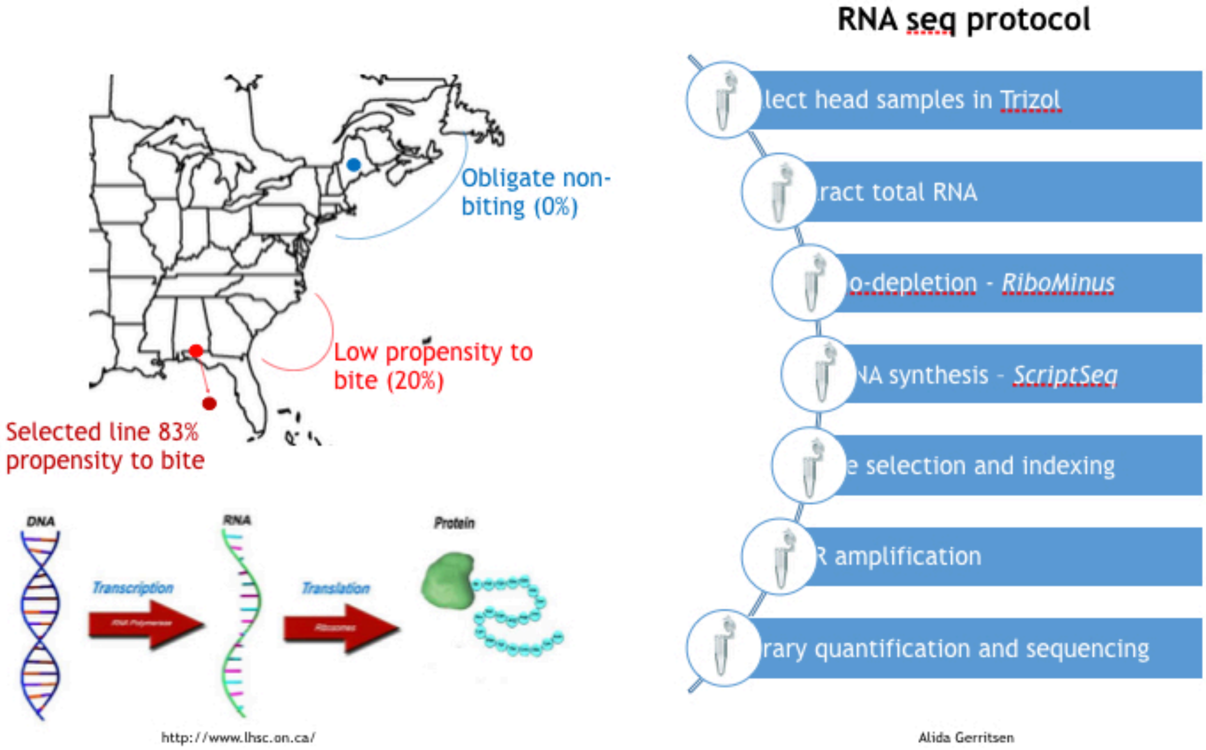
Why do mosquitoes in the south bite (like other species of mosquito) but those in the north don't?

- hypothesis: a genetic mutation occurred at some point as populations moved north at the end of the last Ice Age

images from scientificamerican.com and Rudy Boroczak

Data. Lots and Lots of Data.

They set up experiments to collect mRNA (expressed genes) from biters and non-biters



http://www.lhsc.on.ca/

RNA seq protocol

lect head samples in Trizol
ract total RNA
o-depletion - *RiboMinus*
NA synthesis - *ScriptSeq*
e selection and indexing
R amplification
rary quantification and sequencing

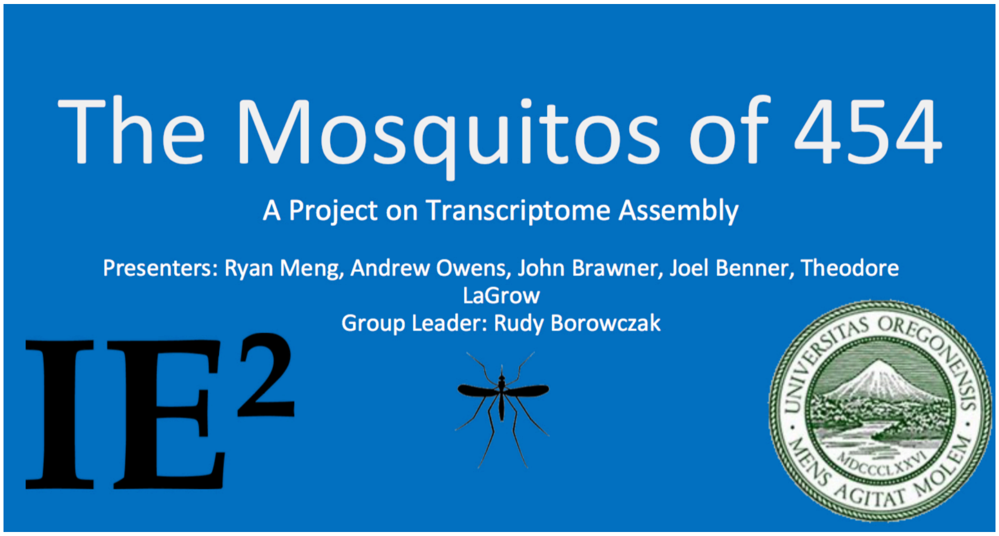
Alida Gerritsen

- these experiments generate huge data sets — hundreds of Gigabytes of raw data
- highly sophisticated algorithms analyze the data, look for differences in expressed genes

Computational Workflow

The image on the previous page showed the steps in the RNA-seq protocol — “wet lab” processes that extract and sequence mRNA

The result is a data set with millions of small fragments that are the starting point for a **computational workflow** — “dry lab” processes that assemble the data into full-length mRNAs and look for differences between populations



The Mosquitos of 454

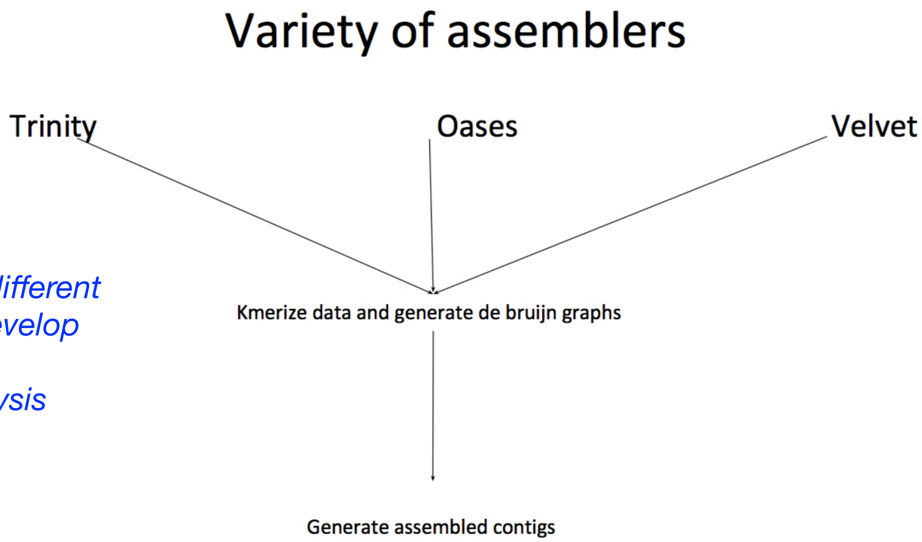
A Project on Transcriptome Assembly

Presenters: Ryan Meng, Andrew Owens, John Brawner, Joel Benner, Theodore LaGrow

Group Leader: Rudy Borowczak

IE^2

UNIVERSITAS OREGONENSIS · MENS AGITAT MOLEM · MDCCCLXXVI



Variety of assemblers

Trinity

Oases

Velvet

Kmerize data and generate de bruijn graphs

Generate assembled contigs

CIS 454 project: evaluate different “transcript assemblers”, develop computational pipeline for differential expression analysis

What Computing Skills Does a Biologist Need?

The computer programs used in the workflow are very sophisticated

- `trinity`, `oases`, `velvet`, `DEseq2`, `BLAST`, ...
- free, open source software, download from GitHub or other repositories
- created by teams of computer scientists and biologists, using advanced algorithms and years of development

A modern biologist does not need knowledge of advanced algorithms and data structures or sophisticated programming skills to implement them

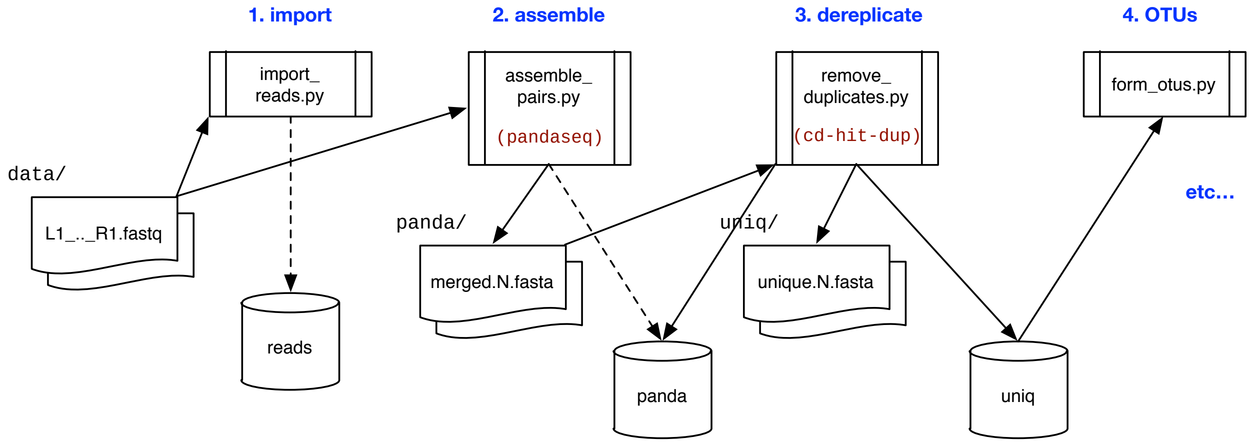
But they do need to

- download, install, run an application on their own computer (or get an account on a supercomputer where the application is already installed)
- know how to run the programs — they work on Linux, macOS, and (usually) Windows, but require scientists to know how to use a command line interface
- use a text editor or write a script to pull out data for further analysis — either for another complex application, or a statistics application like R or Matlab

META 16s rRNA Pipeline

The computational workflow we'll use for several projects this term is from the META Center here at UO

- start with sequence fragments of 16S rRNA genes from a sample (e.g. zebrafish intestine)
- goal: what bacteria are present, what are their relative abundance?



Python programs run bioinformatics applications, extract data to hand off to the next step in the pipeline:

`assemble_pairs.py` ⇨ `pandaseq`

`remove_duplicates.py` ⇨ `cd-hit-dup`

etc

META 16s rRNA Pipeline (cont’d)

Several of our projects will be based on this pipeline

- shell commands to move files around, remove temporary files, etc
- Python programs that read and process data
- Python scripts that run bioinformatics applications

Use Your Own Data

Attention grad students (or undergrads who work in a lab):

- you can use your own computational pipeline instead of the META 16S pipeline
- make an appointment, we’ll talk about ways you can use your data and applications

Our Goals for Bi 410/510

Learn to use a **command line interface**

- alternative to GUI such as Finder (Mac) or File Browser (Windows)
- widely used to run scientific applications
- required for remote connections to supercomputers

Collect commands into a **script**

- automate sequences of operations
- a simple form of programming

Introduction to **Python**

- popular programming language widely used by scientists
- manage data files, data analysis, visualization, more
- an excellent scripting language for more complex workflows

Advanced topic(s) TBD: **databases, statistics, visualization**

- there are Python libraries for each of these topics

Grading Scheme

Course grades will be based on points earned during the term

- programming projects
- milestone exams
- in-class exercises

Programming Projects (8 projects, total of 80 points)

- group projects allowed (small groups of 2-3 people)
- self-paced, do them at any time
- 10 points per project
- revise and resubmit until you earn full credit

Milestone Exams (8 exams, total of 150 points)

- short quizzes taken during Friday lab sessions
- take an exam any time after completing the associated project
- not group projects
- open notes / open book
- retake exam to improve your score (max score used to compute grades)

In-Class Exercises (20 points)

- random throughout the term
- more information at the start of each exercise

Grades (Bi 410)

Total		Comment
90	D	complete all group projects, no milestones or participation
110	C−	
125	C	all projects, minimum score on exams, some participation
140	C+	
160	B−	
175	B	all projects, pass all exams, full participation
190	B+	
210	A−	
220	A	all projects, good style on exams, full participation*

Notes

- milestone exams will have scores from 1 to 20
- a "passing" score on programming projects: 10 points if the program works, even if it has style issues
- we'll have lots of examples in lectures of what is required to earn the full 20 points

✳ it is possible to earn an A without doing any in-class exercises (but there is little margin for error)

Bi 510: *one additional project or written assignment, different point scale (I'll be sending e-mail with more information).*

What to Expect

You will be frustrated often

- very, very frustrated

“Learn to fail like a computer scientist”

- recognize where the problems are
- try experiments to isolate, fix the problem

Learn by doing

- you will not learn to program by reading about programs
- you **must do the projects**
- when a program is finally working, learn from your mistakes: know why it failed, why the changes you make fixed the problems
- start early, don't put things off

Get Organized!

- start a “lab notebook”
- create a “snippets” files with code examples
- keep track of what works, what doesn’t
- have your notebook handy during projects and exams

In-Class Exercise for Apr 4

Our first in-class exercise:

- read a paper by Ben Marwick from the University of Washington:

  “How computers broke science — and what we can do to fix it”

- the paper was published on a website called The Conversation, find it at

  `http://theconversation.com/how-computers-broke-science-and-what-we-can-do-to-fix-it-49938`

At the beginning of class on Wednesday I’ll ask a question based the article

- break into small (2-4 people) groups to discuss the question
- turn in your answer, either on paper or by e-mail (include all group member names)