

Day 03: Introduction to bioinformatics & computational biology

- Lay of the bioinfo-compbio land
- Reading papers | Framing the problem
- Choosing a good problem
- Resources @ MSU
- Getting help

Bioinformatics & Computational Biology

Computational biology

- The study of biology using computational techniques.
- Goal: learn new biology, knowledge about living systems. *It is about science.*



Margaret Dayhoff – One of the first bioinformaticians

Applying math & computational techniques to the sequencing of proteins and nucleic acids.

- 1965: First collection of protein seqs. Single-letter code for amino acids.
- 1966: 'Evolutionary trees'.
- 1978: First AA similarity-scoring matrix.
- 1980: Launched the Protein Information Resource, the first online database system that could be accessed by telephone line.

Bioinformatics

- The creation of tools (algorithms, databases) that solve problems.
- Goal: build useful tools that work on biological data. *It is about engineering.*

Bioinformatics & Computational Biology – Today

“Computational thinking and techniques are so central to the quest of understanding life that today **all biology is computational biology**.

- Brings order into our understanding
- Makes biological concepts rigorous and testable, and
- Provides a reference map that holds together individual insights.

The next modern synthesis in biology will be driven by mathematical, statistical, and computational methods being absorbed into mainstream biological training, turning biology into a quantitative science.”

Shifting roles of computational biologists

	Past	Current
Role in research	Supportive	Driver of research
A feeling for the biology	Computer science-centered	Biology- and computer science-centered
Environment	Isolated	Integrated
Data generation	Constrained	Resourceful
Data exploration	Largely limited to hypothesis testing	Both exploratory and hypothesis testing

Markowitz (2017) PLoS Comp. Biol.
Yanai & Chmielnicki (2017) Genome Biol.

Some broad research areas & related analytical methods

- Genome assembly and annotation
- Sequence alignment and pattern finding
- Molecular evolution and comparative genomics
- Genetic variation and quantitative genetics
- Regulatory genomics
- Functional genomics and data integration
- RNA/Protein structure prediction
- Molecular docking and dynamics simulations
- Artificial life and digital evolution
- Modeling signaling, regulatory pathways
- Metabolic reconstructions and dynamic models
- Large-scale biological networks
- de Bruijn graphs, Hidden Markov Models
- Dynamic programming
- Tree construction, Suffix trees
- Statistical inference, Multiple testing
- Expectation maximization, Gibbs sampling
- Dimensionality reduction, Machine learning
- Maximum entropy modeling
- Atomic, physical simulation
- Agent-based modeling
- Dynamical simulation, State space, Bifurcations
- Linear programming
- Graph theory, Label propagation

Data types and repositories – some examples

Genomes & proteomes

all encompassing

Ensemble

comparative genomics

COGs | InParanoid | OrthoMCL

ref. gene/transcript sequences
& annotations

RefSeq | Entrez | GENCODE

sequences variation

1000 Genomes | dbSNP

everything protein

UniProt | InterPro | SCOP | CATH |
PDB

Functional annotations & relationships

biol. processes, mol. functions,
cellular components

Gene Ontology

pathways

Reactome, KEGG, WikiPathways

networks

BioGRID, TRANSFAC, STRING

Phenotype-, Disease-association

OMIM | GWAS Catalog | ClinVar |
COSMIC

Genome-Phenome

dbGaP | UK Biobank

Functional/regulatory genomics

data sets

NCBI GEO | EBI ArrayExpress

raw reads

NCBI SRA | EBI ENA

consortia

ENCODE | Roadmap | GTEx | TCGA

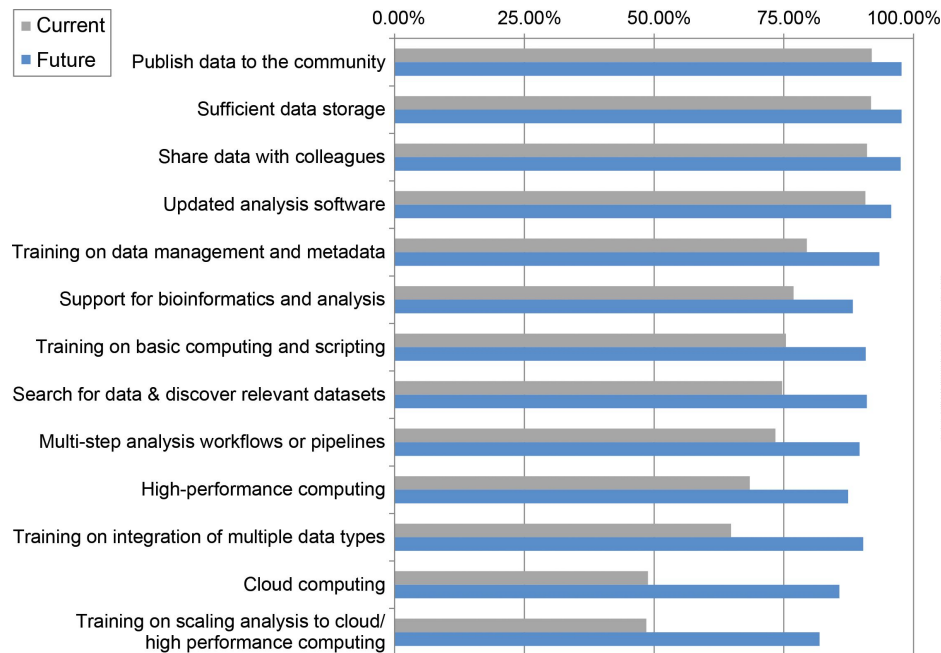
curated public data

Dryad | Repositive | Expression Atlas

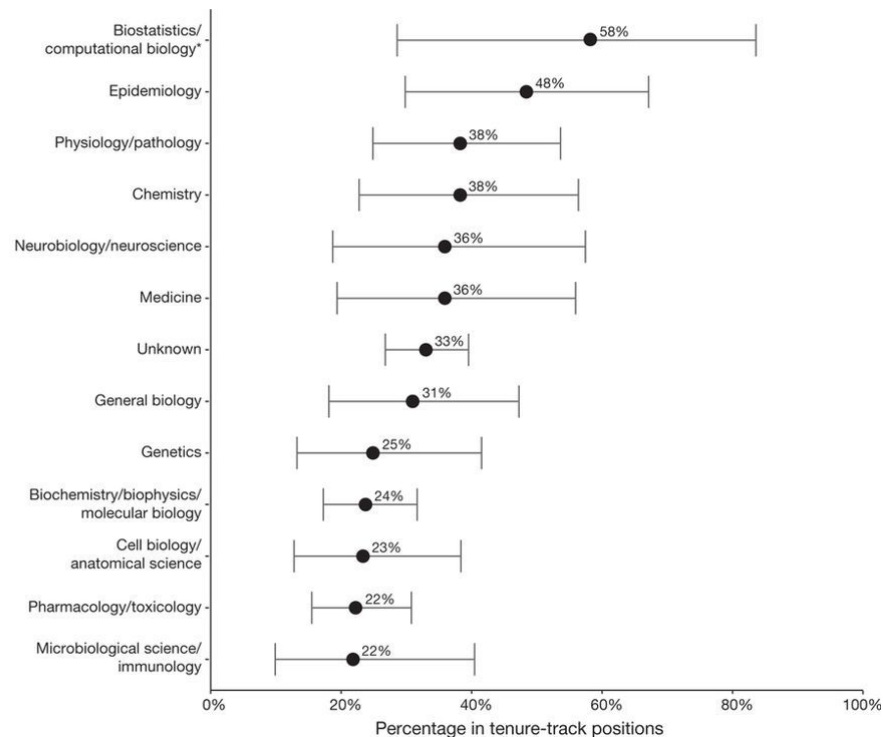
Model organism databases

MGI | RGD | TAIR | FlyBase | WormBase
| ZFin | SGD

Opportunities & Unmet needs



Doctoral degree field



Community – Meetings / Conferences; MSU Seminars

Some National/International Conferences

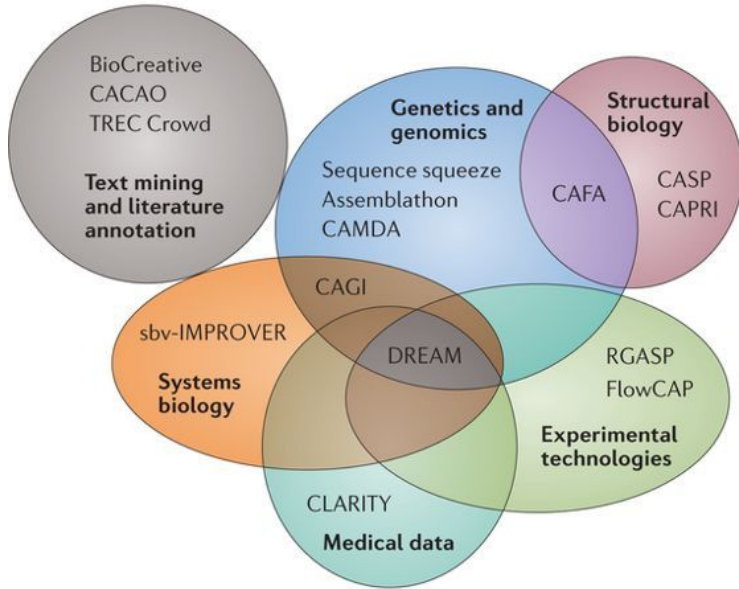
- Intelligent Systems for Molecular Biology
- Research in Computational Molecular Biology
- Pacific Symposium on Biocomputing
- ACM Conference on Bioinformatics, Computational Biology, & Health Informatics
- Rocky Mountain Bioinformatics Conference
- Great Lakes Bioinformatics Conference
- Cold Spring Harbor Laboratories Meetings (Network Biology, Genome Informatics)

Relevant MSU Seminar Series

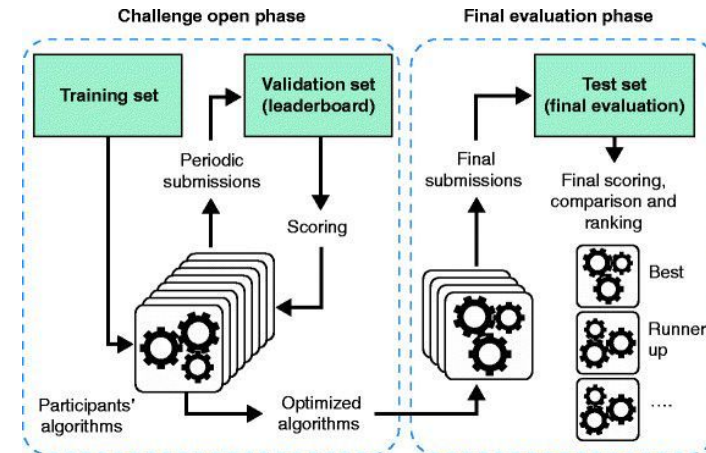
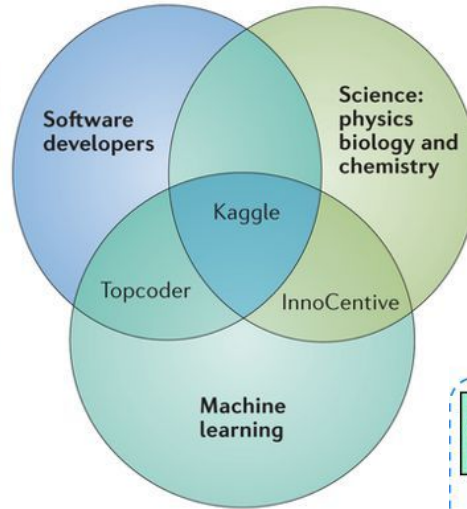
- Science at the Edge
- Machine Learning
- Computational Math, Sci, & Egr
- Institute for Quant. Health Sci. & Engg

Community – Open Challenges

Researcher-driven domain-specific Challenges



Intermediary commercial Challenge platforms



Open science

Code: The field has dramatically shifted in thinking on how to publish code.

- Code used in research should be made available for research use free of charge.
- This is not just code for downloading & using. Original code must be made publicly available for others to use, review, and edit.
- Most common way to share code: GitHub.

Scientific publishing: Preprints

- Rapid publication of new science + free access (e.g. bioRxiv).
- Major source of cutting-edge research.
- Can have multiple (progressively better) versions of each manuscript.
- Preprints have NOT been peer-reviewed for quality and soundness of science.
So, read/use with caution.

Reading primary research papers – Learning to frame the problem

Great way to learn how to frame a problem, choose the methods/tools, set up an analysis workflow, establish groundwork, & generate a series of supportive results towards answering the central question.

Types of computational research studies

- New analytical/computational method
- Improvement of an existing method
- Evaluation of existing methods
- Development of (re-)usable software, web-service, or database
- New insights w/ new/existing methods

Journals to follow

Bioinformatics	Cell
bioRxiv Bioinformatics	eLife
bioRxiv Genomics	Nature
BMC Bioinformatics	Nature Biotechnology
Briefings in Bioinformatics	PNAS
Cell Systems	Science
Genome Biology	Science Translational Med.
Genome Research	
Molecular Systems Biology	PubMed Alerts
Nature Genetics	Google Scholar Alerts
Nature Methods	
Nucleic Acids Research	
PLoS Computational Biology	

Reading primary research papers

Title & Abstract

1. Use **Title & Abstract** for only selecting paper. Read them again last!

Introduction

2. Read the **Introduction**:

- a. Identify *the* question. What is the big challenge the authors are trying to solve?
- b. What are the then current approaches for solving that problem? What are their limitations that, according to the authors, need to be addressed?
- c. What are the *specific* questions this paper is going to answered?

Data & Methods

Results

3. Read **Data & Methods**: [Be critical!]

- a. For each specific Q, note data (type & source) & method (algorithms/techniques, software, & approach). Pay attention to the **Supplemental materials**. These days much of the good stuff is in here!
- b. Make detailed notes on: 1) what's unclear, 2) what you might do differently.

Discussion

References

Reading primary research papers

Title & Abstract

4. Read the **Results**: [Be critical!]

- a. Go figure-by-figure, panel-by-panel. Based on your reading of Data & Methods, is there enough information to know/reproduce that analysis?
- b. Try to interpret each figure/panel, then read the figure legend and the part of the results that explains it. [**Supplemental figures/tables** abound!]
 - i. Do your interpretations match that of the authors'?
 - ii. Are the results answering the specific Qs?
- c. Make detailed notes on: 1) what's unclear, 2) what you might do differently.

Introduction

Data & Methods

Results

Discussion

5. Read the **Discussion/Conclusions**, Title, & Abstract:

- a. Step back to think about contributions, limitations, open Qs, & next steps.

References

6. Read what other researchers (**papers that cite this paper**) say about this paper.

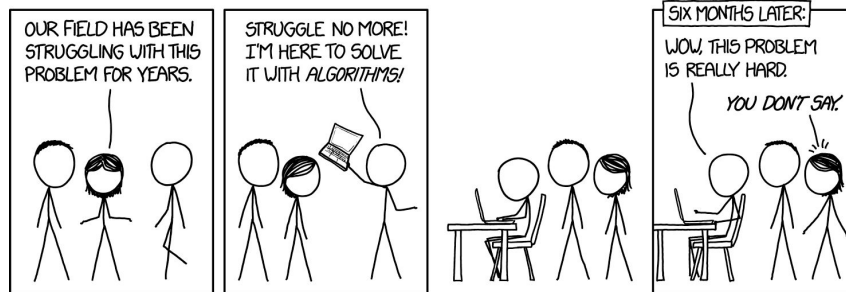
Reading primary research papers

Reading, Retention, and Reuse

- Make reading papers a habit.
- Critically analyze what you read/hear. Don't be swayed by high-profile papers, media hype, or current dogma. Don't be swayed by the narrative.
- Use a reference manager (e.g. Zotero), put *everything* you read into it. Add add notes about specific take-homes. Use tags to group papers by subfield/method/data.
- Create and maintain a single source of all the technical terms and vocabulary for your project.
- Create and maintain a single source (R/Jupyter Notebook) with notes/text-excerpts/figures from all papers & reading materials.
- Contextualize what you read in relation to everything else you know / have read. Specifically consider limitations. Analyze information in terms of you and your project.

Choosing a good computational biology problem

xkcd.com/1831



New

Area / system

Problem / question

Algorithm / technique

I Want

Insights / improvement / clarity / efficiency / usability

Interesting example



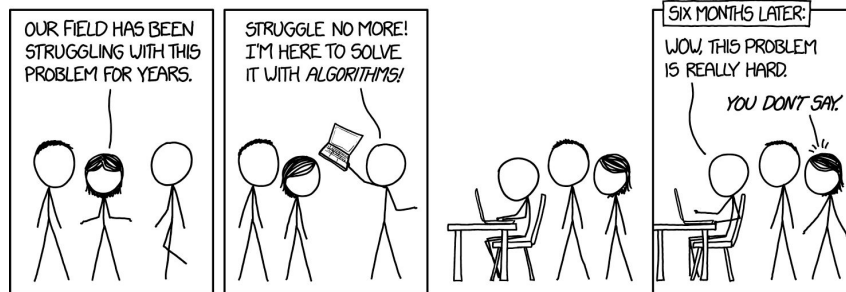
Generalized problem



Large-scale solution/insight

Choosing a good computational biology problem

xkcd.com/1831



Explore and prototype early to fail fast and learn

- Exploration + prototyping: critical for determining if the problem is well-defined & tractable.
- Perform preliminary analysis with simple baselines, sample datasets, and toy examples.
- Don't speculate or make assumptions. Instead, implement something and check them.
- The value lies not in the code/plots you produce, but in the lessons you learn.

Resources @ MSU

Institute for Cyber-Enabled Research

- High-Performance Computing Cluster: wiki.hpcc.msu.edu
- Training resources: www.icer.msu.edu/education-events/training-resources
- Seminars and workshops: www.icer.msu.edu/upcoming-workshops →
- Regular open office hours:
 - Every Monday & Thursday 1-2 p.m. at BPS Room 1440.

Working/student groups

- R-Ladies: <https://rladies-eastlansing.github.io/>
- MSU Data Science: <http://msudatascience.com/>

JAN
09

CMSE Bioinformatics Spring 2019 Modular Courses

These (1 month, 1 credit) graduate level modules are designed for busy graduate students who need to learn computational skills while balancing their work in the research lab. Track 1 provides a practical introduction to basic programming, statistical and data handling concepts. Track 2 focuses on analyzing bioinformatics data including genomic and RNA-seq data.

JAN
11

Introduction to Python

This is an introductory python workshop intended for participants who are beginning programmers or are new to the python language.

JAN
17

Introduction to Linux

Learn how to navigate the UNIX file system and write a basic shell script as a prerequisite for submitting computational jobs on the HPCC.

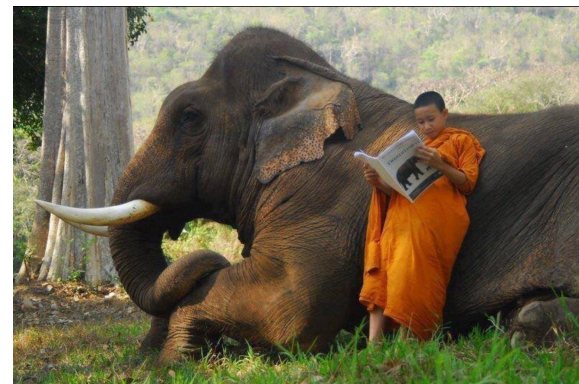
JAN
22

Monthly Workshop: Introduction to HPCC


This is a hands-on introductory workshop on using MSU's High Performance Computing Center (HPCC).

Getting help

- **Linux** | rik.smith-unna.com/command_line_bootcamp, commandline.guide, & swcarpentry.github.io/shell-novice
- **Python** | Introduction: learnpythonthehardway.org/book & developers.google.com/edu/python | Data analysis: jakevdp.github.io/WhirlwindTourOfPython | Visualization: www.r-graph-gallery.com
- **R** | Introduction: swcarpentry.github.io/r-novice-inflammation & swirlstats.com ('R Programming' & 'Data Analysis') | Data analysis: r4ds.had.co.nz | Visualization: python-graph-gallery.com
- **Git & GitHub** | swcarpentry.github.io/git-novice/, speakerdeck.com/alicebartlett/git-for-humans, & rogerdudler.github.io/git-guide/
- **Probability and Statistics** | Nature Collection (Statistics for Biologists | Practical Guides | Points of Significance): www.nature.com/collections/qghhqm
- **Genetics and Molecular Biology** | learn.genetics.utah.edu/ & www.genomicseducation.hee.nhs.uk



Getting help

 ... so many excellent blog posts!

 **stackoverflow**

 **Biostars**
— BIOINFORMATICS EXPLAINED —

Several video lessons/courses on YouTube

No shame!

StackOverflow Importer

Do you ever feel like all you're doing is copy/pasting from Stack Overflow?

Let's take it one step further.

from stackoverflow import quick_sort will go through the search results of [python] quick sort looking for the largest code block that doesn't syntax error in the highest voted answer from the highest voted question and return it as a module. If that answer doesn't have any valid python code, it checks the next highest voted answer for code blocks.

```
>>> from stackoverflow import quick_sort, split_into_chunks

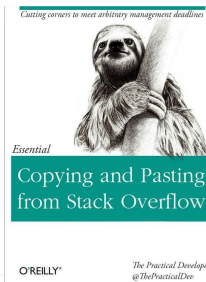
>>> print(quick_sort.sort([1, 3, 2, 5, 4]))
[1, 2, 3, 4, 5]

>>> print(list(split_into_chunks.chunk("very good chunk func")))
['very ', 'good ', 'chunk', ' func']

>>> print("I wonder who made split_into_chunks", split_into_chunks.__author__)
I wonder who made split_into_chunks https://stackoverflow.com/a/35107113

>>> print("but what's the license? Can I really use this?", quick_sort.__license__)
but what's the license? Can I really use this? CC BY-SA 3.0

>>> assert("nice, attribution!")
```



Teaching philosophy

My intertwined core values:

- Democratizing science-training-and-education,
- Leveling playing fields, and
- Improving access to (& entry into) our work.

For me, the classroom is an environment for inclusive collaborative learning:

- Everyone belongs and brings a unique strength & a valuable perspective.
- Peer-to-peer interaction, teaching, feedback, and collaboration is key.

The most underrated part of teaching is learning. It's such a joy!

- I learn a ton every time I teach a course and I design courses that help me learn. And you students point out my blind spots.