2023

Bioinformatics Lab: A Course-Based Undergraduate Research Experience Curriculum

Kristen M. DeAngelis
University of Massachusetts Amherst, deangelis@cns.umass.edu

Mallory Choudoir
University of Massachusetts Amherst

Ashley Eng
University of Massachusetts Amherst

Maureen A. Morrow
State University of New York at New Paltz

Recommended Citation
DeAngelis, Kristen M.; Choudoir, Mallory; Eng, Ashley; and Morrow, Maureen A., "Bioinformatics Lab: A Course-Based Undergraduate Research Experience Curriculum" (2023). *Microbiology Educational Materials*. 3.
https://scholarworks.umass.edu/micro_ed_materials/3

Bioinformatics Lab: A Course-Based Undergraduate Research Experience Curriculum

Authors: Kristen M. DeAngelis[1], Mallory Choudoir[1], Ashley Eng[1], Maureen Morrow[2]

Author affiliations: [1] Microbiology Department, University of Massachusetts Amherst, USA. [2] Biology Department, State University of New York New Paltz, USA

This curriculum describes a 3 credit course teaching Bioinformatics to undergraduate students. It is designed to fulfil the advanced laboratory requirement for undergraduate majors as well as the graduate level core competency requirement in Bioinformatics. This course was taught at the University of Massachusetts Amherst as MICROBIO590B in in 2021 and 2022, and is planned to be taught in 2023 as MICROBIO 567.

The overarching goal of this course is to improve students' comfort and familiarity with using computer programing and sequence analysis to ask and answer questions about organisms.

This course-based undergraduate research experience (CURE) is intended to teach students skills in unix ('command line') and markdown programming, as well as in the application of sequence data for inference into biological systems. Students work together with the instructor to assemble and annotate the same genome, then are assigned an unknown genome and work on their own to assemble, annotate, and analyze the genome. The final project is to write a genome report in the style of an American Society for Microbiology (ASM) Resource Announcements manuscript.

Included in this curriculum are the following resources:

- Course syllabus, timeline, and bibliography of assigned reading
- Bibliography used for course development
- Lab manuals for in class activities
    - LM1: Introduction to unix coding and cloud computing
    - LM2: Assembling genome sequences
    - LM3: Annotating genomes using KBase and markdown language
    - LM4: Rubric for writing the genome report

Microbiology 590b: Bioinformatics Lab

Catalog number:	23156
Units:	3
Course Instructor:	Kristen DeAngelis, deangelis@microbio.umass.edu
Office hours:	during the half hour before class, also by appointment
Meeting times:	Mondays & Wednesdays 9:05 am to 11:00 am
Meeting location:	212 Morrill III
Pre-requisites:	MICROBIO 310 or MICROBIO 311
Special note:	You will need a computer for this class with permissions to install new software. If this presents a hardship, please let the instructor know before class begins.
Textbook:	(*optional*) Nina Parker, Mark Schneegurt, Anh-Hue Thi Tu, Philip Lister, Brian M. Forster. Microbiology. Publisher/website: OpenStax. Publication date: Nov 1, 2016. https://openstax.org/details/books/microbiology

Course description

This is a computer lab course intended to give students technical and practical experience in analyzing sequencing data. We focus on bacterial genomes from recently isolated bacteria, so there is a possibility of discovery of new microbial diversity. Collaborative learning will be encouraged, but students will work individually to assemble, annotate, and analyze a bacterial genome.

Each week will begin with a review on the topic to be covered for that week. When applicable, there will be pre-recorded lectures to introduce lab topics, and students are expected to have watched these before coming to class. Our time in class will be spent as a computer lab, where students will work collaboratively and at their own pace to complete their genome sequencing and analysis. The guided analysis will transition into independent study. Students will produce a final capstone report in the style of the Microbiology Resource Announcements published by the American Society of Microbiology.

This course fulfils the Microbiology Major requirement as a Major Laboratory Elective, and the Microbiology Graduate requirement as a core graduate-level course in microbial science.

Course learning goals

The overarching goal of this course is to improve students' comfort and familiarity with using computer programing and sequence analysis to ask and answer questions about organisms.

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

- Describe the structure and function of a bacterial genome.
- Use computational tools to de novo assemble genomes, and predict physiology based on coding and non-coding genes.
- Design a genomics experiment, generate and test a hypothesis using genomics data, and tell a story about an unknown organism using only the genome sequence.
- Apply bioinformatics tools to ask and answer new questions using sequence data.

Student expectations

Students are expected to come to each class having completed the reading. Assigned reading is described in the schedule and will be provided on the course Moodle site. Students are expected to participate in class and complete their work on time. If you need help, extra time, or more information, please ask, and I will work with you.

Students are expected to attend all course meetings, except with explicit written permission from the instructor. Attendance is expected except in cases of emergency, religious holidays, or for participation in official College functions. Cheating, including plagiarism will not be tolerated.

If you tell me that you're having trouble, I will not judge you or think less of you. You do not owe me personal information about your health (mental or physical), or the health of your loved ones, but you are welcome to tell me and I will listen. If I can't help you, I usually know someone who can.

Grading

Grades will be based on the following activities. Letter grades will be based on the following numerical cutoffs, where the grade percentage will be converted into a letter grade based on the following scale.

Activity	% Total Score
Attendance & Participation	5%
4 Problem Sets	20% (5% each)
4 Quizzes	20% (5% each)
Project part 1: phylogeny & taxonomy	10%
Project part 2: hypothesis & intro	10%
Project part 3: assembly stats	10%
Project part 4: annotation stats	10%
Capstone Project report	15%
Total	100%

Grade	Score
A	≥ 93%
A-	90-92.9%
B+	87-89.9%
B	83-86.9%
B-	80-82.9%
C+	77-79.9%
C	73-76.9%
C-	70-72.9
D	60.0-69.9
F	< 60%

Attendance & Participation Policy

Attendance and participation are required. Our class meets in person twice per week, and students are expected to be in the classroom for every class. Students can miss one class without penalty to this component of their grade. An email alerting me of your absence is a courtesy and appreciated, but not required. After the first unexcused absence, each additional unexcused absence will result in a loss of 0.5% of your final grade. Contact the instructor for excused absences.

Participation is also required for this course. Students will need to work together at times to complete the assignments. There will be opportunities for discussion, asking questions, and answering questions during class. A perfect participation grade will be awarded to students who either speak up in class or use the chat; who occasionally post or answer questions on the moodle forums; and who generally engage in the course dialog with the instructor and the other students.

Class Communication

All class-related communication will be through announcements in class, verbally and posted on slides presented in class. The Moodle course site will also be an important component of class communication. This includes graded assignments, copies of slides and handouts, announcements, and reading assignments.

Accommodation Statement

The University of Massachusetts Amherst is committed to providing an equal educational opportunity for all students. If you have a documented physical, learning, or psychological disability on file with Disability Services (DS), you may be eligible for reasonable academic accommodations to help you succeed in this course. If you have a documented disability that requires an accommodation, please notify me within the first two weeks of the semester so that we may make appropriate arrangements. Please reach out to Disability Services or the UMass Dean of Students if you feel you may need accommodations in order to be successful in your course work.

Academic Honesty Statement

Since the integrity of the academic enterprise of any institution of higher education requires honesty in scholarship and research, academic honesty is required of all students at the University of Massachusetts Amherst. Academic dishonesty is prohibited in all programs of the University. Academic dishonesty includes but is not limited to: cheating, fabrication, plagiarism, and facilitating dishonesty. Appropriate sanctions may be imposed on any student who has committed an act of academic dishonesty. Instructors should take reasonable steps to address academic misconduct.

Any person who has reason to believe that a student has committed academic dishonesty should bring such information to my attention as soon as possible. Instances of academic dishonesty not related to a specific course should be brought to the attention of the appropriate department Head or Chair. Since students are expected to be familiar with this policy and the commonly accepted standards of academic integrity, ignorance of such standards is not normally sufficient evidence of lack of intent (http://www.umass.edu/dean_students/codeofconduct/acadhonesty/).

Assessments schedule

There is one survey, quiz, problem set or assignment due each week. Quizzes are available on moodle, are open book and open note, and can be taken as many times as the student wishes. All assessments (surveys, quizzes, problem sets, and projects) are due at midnight on Friday for the weeks listed in the course schedule below, to be turned in using Moodle. There is no scheduled final exam, but the capstone project is due at midnight on the last day of the finals period.

Timeline of Activities

Week	Preparation for Class	In-class Activity	Due in Moodle*
9/7	• Syllabus review • Unit 1: Introduction and overview	Introductions, syllabus	
9/12,14	• Unit 2: Genomics and molecular biology • Snyder, Introduction (p. 1-12) • Snyder, Chapter 1, Structural Features of Bacterial Genomes (p37-38), The Bacterial Genome (p 50-51), and Molecular Biology Manipulations with DNA (p. 53-64) • Beja et al. 2000 (*optional*)	Learning unix and preparations for bioinformatics	Quiz 1
9/19,21	• Unit 3: DNA sequencing • Goodwin et al. 2016 • Research Paper 1, applications (*choose one from on Moodle*) • Tedersoo et al. 2021 (*optional*)	Molecular microbiology for genomics, How to read scientific papers, MGHPCC and command line coding, Lab Manual Part 1	Problem Set 1: Unix
9/26,28	• Unit 4: Genome assembly • Goldstein et al., 2019 • Lu et al., 2016 (*optional*)	Sequencing technologies, Paper discussion, Lab Manual Part 1	Quiz 2

10/3,5	• Unit 5: Quality assessment of DNA sequence • Bowers et al. 2017 • Field et al. 2007 • Zhou & Rokas 2014 (*optional*)	Genome assembly, Lab Manual Part 2	Problem Set 2: Data QC
10/12	• Unit 6: Sequence alignment • Tyson et al. 2004 • Koonin et al. 2020 (*optional*)	Quality control of sequence data, Lab Manual Part 2	Quiz 3
10/17,19	• Unit 7: Phylogeny • Washburne et al. 2018 • Yang & Rannala 2012 (*optional*)	Paper discussion, Sequence alignment, Lab Manual Part 3	Problem Set 3
10/24,26	• Unit 8: Annotation • OpenStax Chapter 8 • Salzberg 2019	Paper discussion, Phylogeny and taxonomy, Lab Manual Part 3	Quiz 4
10/31, 11/2	• Unit 9: Introducing the genomes of study • Melillo et al. 2017 • Pold et al. 2016	Paper discussion, Annotation, Lab Manual Part 3	Problem Set 4: Assembly
11/7,9	• Unit 10: Hypothesis construction and testing • Dunning Hotopp et al. 2020 • Baltrus et al. 2019	Paper discussion, Assignment of genomes for capstone projects	Project Part 1: Table 1 due
11/14,16	• Unit 11: Pangenomes • Research Paper 2 Comparative Genomics (*choose one from on Moodle*) • Trapnell & Salzburg 2009 (*optional*)	Paper discussion, Project work	Project Part 2: Phylogeny and taxonomy (assembly)
11/21	• Thanksgiving week, no assignments	Paper discussion, Project work	
11/28,30	• Research Paper 3, Microbiology Resource Announcements (*choose one from on Moodle*) • Shakya et al. 2020 • Hedge & Wilson 2016 (*optional*)	Paper discussion, Project work	Project Part 3: Tables 2 & 3 (annotation) due
12/5,7	• Anonymous peer review • Coleman & Korem 2021 • Microbiology by the Numbers (*optional*)	Final student presentations	Project Part 4: Hypothesis & Intro due
12/12	• Last day of classes on 12/12	No scheduled final exam	
12/19	• In person or zoom project help available all week by appointment		Final report due**

* All quizzes, problem sets, and project parts are due at 11:59 PM on the Friday for the week listed in this Timeline of Activities. For example, Survey 1 will be due on Friday Sept 9th at 11:59 PM and Quiz 1 will be due on Friday Sept 15th at 11:59 PM.

** The Capstone final report is due on Monday December 19 at 11:59 PM, which is the last day of finals.

Reading list

- Baltrus, David A., Christina A. Cuomo, John J. Dennehy, Julie C. Dunning Hotopp, Julia A. Maresca, Irene LG Newton, David A. Rasko, Antonis Rokas, Simon Roux, and Jason E. Stajich. "Future-proofing your Microbiology Resource Announcements genome assembly for reproducibility and clarity." (2019): e00954-19.
- Béja, Oded, L. Aravind, Eugene V. Koonin, Marcelino T. Suzuki, Andrew Hadd, Linh P. Nguyen, Stevan B. Jovanovich et al. "Bacterial rhodopsin: evidence for a new type of phototrophy in the sea." Science 289, no. 5486 (2000): 1902-1906.
- Bowers, Robert M., Nikos C. Kyrpides, Ramunas Stepanauskas, Miranda Harmon-Smith, Devin Doud, T. B. K. Reddy, et al. "Minimum information about a single amplified genome (MISAG) and a metagenome-assembled genome (MIMAG) of bacteria and archaea." *Nature biotechnology* 35, no. 8 (2017): 725-731.
- Coleman, Izaak, and Tal Korem. "Embracing Metagenomic Complexity with a Genome-Free Approach." *Msystems* 6, no. 4 (2021): e00816-21.
- Dunning Hotopp, Julie C., David A. Baltrus, Vincent M. Bruno, John J. Dennehy, Steven R. Gill, Julia A. Maresca, Jelle Matthijnssens et al. "Best Practices for Successfully Writing and Publishing a Genome Announcement in Microbiology Resource Announcements." (2020): e00763-20.
- Field, Dawn, George Garrity, Tanya Gray, Norman Morrison, Jeremy Selengut, Peter Sterk, et al. "The minimum information about a genome sequence (MIGS) specification." *Nature biotechnology* 26, no. 5 (2008): 541-547.
- Goldstein, Sarah, Lidia Beka, Joerg Graf, and Jonathan L. Klassen. "Evaluation of strategies for the assembly of diverse bacterial genomes using MinION long-read sequencing." *BMC genomics* 20, no. 1 (2019): 23.
- Goodwin, Sara, John D. McPherson, and W. Richard McCombie. "Coming of age: ten years of next-generation sequencing technologies." *Nature Reviews Genetics* 17, no. 6 (2016): 333-351.
- Hedge, Jessica, and Daniel J. Wilson. "Practical approaches for detecting selection in microbial genomes." *PLoS computational biology* 12, no. 2 (2016): e1004739.
- Koonin, Eugene V., Mart Krupovic, Sonoko Ishino, and Yoshizumi Ishino. "The replication machinery of LUCA: common origin of DNA replication and transcription." *BMC biology* 18, no. 1 (2020): 1-8.
- Lu, Hengyun, Francesca Giordano, and Zemin Ning. "Oxford Nanopore MinION sequencing and genome assembly." *Genomics, proteomics & bioinformatics* 14, no. 5 (2016): 265-279.
- Melillo, Jerry M., Serita D. Frey, Kristen M. DeAngelis, William J. Werner, Michael J. Bernard, Francis P. Bowles, Grace Pold, Melanie A. Knorr, and A. Stuart Grandy. "Long-term pattern and magnitude of soil carbon feedback to the climate system in a warming world." *Science* 358, no. 6359 (2017): 101-105.
- Microbiology by numbers. *Nat Rev Microbiol* **9,** 628 (2011). https://doi.org/10.1038/nrmicro2644
- Pold, Grace, Andrew F. Billings, Jeff L. Blanchard, Daniel B. Burkhardt, Serita D. Frey, Jerry M. Melillo, Julia Schnabel, Linda TA van Diepen, and Kristen M. DeAngelis. "Long-term warming alters carbohydrate degradation potential in temperate forest soils." *Applied and environmental microbiology* 82, no. 22 (2016): 6518-6530.
- Salzberg, Steven L. "Next-generation genome annotation: we still struggle to get it right." (2019): 1-3.
- Shakya, Migun, Sanaa A. Ahmed, Karen W. Davenport, Mark C. Flynn, Chien-Chi Lo, and Patrick SG Chain. "Standardized phylogenetic and molecular evolutionary analysis applied to species across the microbial tree of life." *Scientific reports* 10, no. 1 (2020): 1-15.
- Snyder, Larry, et al. Molecular Genetics of Bacteria, ASM Press, 2013. ProQuest Ebook Central, http://ebookcentral.proquest.com/lib/uma/detail.action?docID=1747240. Created from uma on 2018-09-04 10:28:31
- Tedersoo L, Albertsen M, Anslan S, Callahan B. 2021. Perspectives and benefits of microbial ecology. Appl Environ Microbiol 87: e00626-21. https://doi.org/10.1128/AEM.00626-21.
- Trapnell, Cole, and Steven L. Salzberg. "How to map billions of short reads onto genomes." *Nature biotechnology* 27, no. 5 (2009): 455-457.
- Tyson, Gene W., Jarrod Chapman, Philip Hugenholtz, Eric E. Allen, Rachna J. Ram, Paul M. Richardson, Victor V. Solovyev, Edward M. Rubin, Daniel S. Rokhsar, and Jillian F. Banfield. "Community structure and metabolism through reconstruction of microbial genomes from the environment." *Nature* 428, no. 6978 (2004): 37-43.
- Washburne, Alex D., James T. Morton, Jon Sanders, Daniel McDonald, Qiyun Zhu, Angela M. Oliverio, and Rob Knight. "Methods for phylogenetic analysis of microbiome data." *Nature microbiology* 3, no. 6 (2018): 652-661.
- Yang, Ziheng, and Bruce Rannala. "Molecular phylogenetics: principles and practice." *Nature reviews genetics* 13, no. 5 (2012): 303-314.

- Zhou, Xiaofan, and Antonis Rokas. "Prevention, diagnosis and treatment of high-throughput sequencing data pathologies." *Molecular ecology* 23, no. 7 (2014): 1679-1700.

Research papers no. 1: Applications (*choose one*)

- Almeida, Alexandre, Stephen Nayfach, Miguel Boland, Francesco Strozzi, Martin Beracochea, Zhou Jason Shi, Katherine S. Pollard et al. "A unified catalog of 204,938 reference genomes from the human gut microbiome." *Nature biotechnology* 39, no. 1 (2021): 105-114.
- Castro-Wallace, Sarah L., Charles Y. Chiu, Kristen K. John, Sarah E. Stahl, Kathleen H. Rubins, Alexa BR McIntyre, Jason P. Dworkin et al. "Nanopore DNA sequencing and genome assembly on the International Space Station." Scientific reports 7:1 (2017): 1-12.
- Liem, Michael, Tonny Regensburg-Tuïnk, Christiaan Henkel, Hans Jansen, and Herman Spaink. "Microbial diversity characterization of seawater in a pilot study using Oxford Nanopore Technologies long-read sequencing." *BMC research notes* 14, no. 1 (2021): 1-7.
- Plesivkova, Diana, Rebecca Richards, and SallyAnn Harbison. "A review of the potential of the MinION™ single-molecule sequencing system for forensic applications." *Wiley Interdisciplinary Reviews: Forensic Science* 1, no. 1 (2019): e1323.

Research papers no. 2: Comparative Genomics (*choose one*)

- Haro-Moreno, Jose Manuel, Mario López-Pérez, and Francisco Rodriguez-valera. "Enhanced recovery of microbial genes and genomes from a marine water column using long-read metagenomics." *Frontiers in Microbiology*: 2410.
- Lieberman, Tami D., Kelly B. Flett, Idan Yelin, Thomas R. Martin, Alexander J. McAdam, Gregory P. Priebe, and Roy Kishony. "Genetic variation of a bacterial pathogen within individuals with cystic fibrosis provides a record of selective pressures." *Nature genetics* 46, no. 1 (2014): 82-87.
- Undabarrena, Agustina, Ricardo Valencia, Andrés Cumsille, Leonardo Zamora-Leiva, Eduardo Castro-Nallar, Francisco Barona-Gomez, and Beatriz Cámara. "Rhodococcus comparative genomics reveals a phylogenomic-dependent non-ribosomal peptide synthetase distribution: insights into biosynthetic gene cluster connection to an orphan metabolite." *Microbial Genomics* 7, no. 7 (2021): 000621.

Research papers no. 3: Microbiology Resource Announcements (*choose one*)

- Any paper from Microbiology Resource Announcements (https://journals.asm.org/journal/mra)
- Chaput, Gina, Jacob Ford, Lani DeDiego, Achala Narayanan, Wing Yin Tam, Meghan Whalen, et al. "Complete Genome Sequence of *Serratia quinivorans* Strain 124R, a Facultative Anaerobe Isolated on Organosolv Lignin as a Sole Carbon Source." *Microbiology resource announcements* 8, no. 18 (2019).
- DeAngelis, Kristen M., Grace Pold. "Genome Sequences of *Frankineae* sp. MT45 and *Jatrophihabitans* sp. GAS493, two novel Actinobacteria isolated from forest soil." *Microbiology Resource Ann.* (2020) 9, no. 38
- Morrow, Maureen A., Grace Pold, Kristen M. DeAngelis. "Draft Genome Sequence of a terrestrial Planctomyete, Singulisphera sp. GP187, Isolated from Forest Soil." *Microbiology Resource Announcements* (2020) 9, no. 50.
- Pold, Grace, Erin M. Conlon, Marcel Huntemann, Manoj Pillay, Natalia Mikhailova, Dimitrios Stamatis, T. B. K. Reddy et al. "Genome sequence of Verrucomicrobium sp. strain GAS474, a novel bacterium isolated from soil." *Genome announcements* 6, no. 4 (2018).
- Woo, Hannah L., Kristen M. DeAngelis, Hazuki Teshima, Karen Davenport, Hajnalka Daligault, Tracy Erkkila, Lynne Goodwin et al. "High-quality draft genome sequences of four lignocellulose-degrading bacteria isolated from puerto rican forest soil: Gordonia sp., Paenibacillus sp., Variovorax sp., and Vogesella sp." *Genome announcements* 5, no. 18 (2017).
- Woo, Hannah L., Sagar Utturkar, Dawn Klingeman, Blake A. Simmons, Kristen M. DeAngelis, Steven D. Brown, and Terry C. Hazen. "Draft genome sequence of the lignin-degrading Burkholderia sp. strain LIG30, isolated from wet tropical forest soil." *Genome announcements* 2, no. 3 (2014).

<u>Bibliography used for course development</u>:

1. Dow, Ellen, Elisha Wood-Charlson, Steven Biller, Timothy Paustian, Aaron Schimer, Cody Sheik, Jason Whitham et al. *Bioinformatic teaching resources-for educators, by educators-using KBase, a free, user-friendly, open source platform*. Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2021.
2. Foulkes, Andrea S. *Applied statistical genetics with R*. Springer, 2009.
3. Garibay, Juan C. "Creating a Positive Classroom Climate for Diversity." Published by UCLA's office of Diversity & Faculty Development. Web accessed on 17 July, 2020. https://equity.ucla.edu/wp-content/uploads/2016/06/CreatingaPositiveClassroomClimateWeb-2.pdf
4. Prost, Stefan, Sven Winter, Jordi De Raad, Raphael TF Coimbra, Magnus Wolf, Maria A. Nilsson, Malte Petersen et al. "Education in the genomics era: Generating high-quality genome assemblies in university courses." *GigaScience* 9, no. 6 (2020): giaa058.
5. Shortlidge, Erin E., and Sara E. Brownell. "How to assess your CURE: a practical guide for instructors of course-based undergraduate research experiences." *Journal of microbiology & biology education* 17, no. 3 (2016): 399-408.
6. Salazar, Alex N., Franklin L. Nobrega, Christine Anyansi, Cristian Aparicio-Maldonado, Ana Rita Costa, Anna C. Haagsma, Anwar Hiralal et al. "An educational guide for nanopore sequencing in the classroom." *PLoS computational biology* 16, no. 1 (2020): e1007314.
7. Watsa, Mrinalini, Gideon Andrew Erkenswick, Aaron Pomerantz, and Stefan Prost. "Portable sequencing as a teaching tool in conservation and biodiversity research." *PLoS Biology* 18, no. 4 (2020): e3000667.
8. Zeng, Yi, and Christopher H. Martin. "Oxford Nanopore sequencing in a research-based undergraduate course." Biorxiv (2017): 227439.

MICROBIO 590B: Bioinformatics Lab

Welcome to Bioinformatics Lab! This course is intended to give students technical, computational and practical microbial experience in sequencing and analyzing genomes. We will focus on bacterial genomes from recently isolated bacteria, so there is a possibility of discovery of new microbial diversity. Students will work in teams to sequence one bacterial genome.

We will meet twice per week, and we will begin each week with a lecture. In our second meeting of the week, we will start class with a review on the topic that week. The remainder of our time in class will be spent on our computers, where students will work collaboratively and at their own pace to complete their genome sequencing and analysis. Students will produce a final report in the style of the Microbiology Resource Announcements published by the American Society of Microbiology. This course fulfills the Microbiology Major requirement as a Major Laboratory Elective.

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

- Describe the structure and function of a bacterial genome.
- Use computational tools to *de novo* assemble genomes, and predict physiology based on coding and non-coding genes.
- Design a genomics experiment, generate and test a hypothesis using genomics data, and tell a story about an unknown organism using only the genome sequence.
- Apply bioinformatics tools to ask and answer new questions using sequence data.

The course activities are set up in four parts. In **Part 1** (*this part*), we gain access to our computing resources and begin to practice coding in the shell, working with sequence data, and submitting jobs to the MGHPCC Cluster, a cloud computer.

In **Part 2**, we begin work with the raw, basecalled sequence data from the bacterial isolate Ralstonia sp. GP101. In this part, we will work together to subset the data, assemble the genome, and perform some additional quality control assessments.

In **Part 3**, we continue working with the GP101 genome by annotating the assembly, and begin to explore the coding and non-coding regions of the genome. We will also perform phylogenetic, sequence and functional gene comparative analysis with other closely related strains.

In **Part 4**, you will be assigned your own genome that is the basis of your capstone project. These genomes are derived from bacteria we have isolated in our lab. They have been sequenced on the MinION, basecalled and QC checked to ensure that they are ready for you. Their genotype and some phenotypic information is available. You will repeat the analyses we did on GP101 for assembly (**part 2**) and annotation (**part 3**) on your genome, and write it up for your capstone project report.

Acknowledgements

The course materials consist of a lab manual distributed in four parts, and a total of ten lectures. Some slides are credited to Dr. John Gibbons with permission. Dr. Mallory Choudoir wrote much of the lab manual part 2, and Dr. Maureen Morrow wrote much of the lab manual part 3. Some images in the manual and slides are from KBase in the Classroom: Genome Exploration, Authors: Steven Biller and Ellen Dow. Otherwise, image credits are attributed in the slide with the appropriate image.

1. Online access and resources

Moodle Guest Access
URL: https://umass.moonami.com/course/view.php?id=33025
Password: BioinformaticsLab2022!
How to Login as a Guest to a Moodle Course:
https://www.umass.edu/it/support/moodle/log-a-guest-a-moodle-course

MGHPCC
Request an account here: https://www.umassrc.org/hpc/
Or, if you have an account, email me with your account name.
MGHPCC Wiki: https://www.umassrc.org/wiki/index.php/Main_Page
MGHPCC FAQ: https://www.umassrc.org/wiki/index.php/Frequently_Asked_Questions
MGHPCC File Navigator: https://www.umassrc.org:444
Our course directory on the MGHPCC: /home/kd75a/MICROBIO_590B/

KBase
URL: https://narrative.kbase.us/#signup
Sign up for a free account to get access to genomes and tools for analysis.
Watch this video for more information: https://docs.kbase.us/getting-started/quick-start
**** N.B.: Share the KBase Narratives you create in this class with KBase username "kristen"*

Other
Class Genome Database: Class-minion-seq-assembly-log
Markdown Basic Syntax Guide: https://www.markdownguide.org/basic-syntax
FastQC Tutorial & FAQ: https://rtsf.natsci.msu.edu/genomics/tech-notes/fastqc-tutorial-and-faq/
OpenStax Microbiology textbook for non-majors: https://openstax.org/details/books/microbiology
Cell Biology by the Numbers: http://book.bionumbers.org

Shell/command line tutorials
Beginners Linux Tutorial Learning the shell: http://www.linuxcommand.org/lc3_learning_the_shell.php
Command line / unix / linux tutorial - murder mystery! https://github.com/veltman/clmystery
Unix reference guide: https://sites.tufts.edu/cbi/files/2013/01/linux_cheat_sheet.pdf
**** N.B.: this is one of many available guides! Please tell me if you find a better one…*

2. Getting started

These initial activities will set you up with the accounts and resources to complete the work in this class.

1.1. Obtain your lab manual google doc (click to transfer to your google drive in Moodle), and share the lab notebook with me (kristend@umass.edu).

Lab Manual Part 1 - this document
Lab Manual Part 2
Lab Manual Part 3
Lab Manual Part 4

1.2. Go to KBase.us and sign up by creating an account.

Check to make sure you are using a supported browser when you're using KBase. If you don't have a supported browser, get one now.

Once we get our KBase accounts, we will leave them behind for a bit while we focus on cloud computing.

1.3. Request an MGHPCC account by visiting https://www.umassrc.org/hpc/. List me as your PI.

PI First Name : **Kristen**
PI Last Name: **DeAngelis**
PI E-mail: **kristend@umass.edu**
PI Phone: **413-577-4669**

1.4. Open terminal or download a terminal emulator.

For Mac users, your OS has a program called "Terminal." Open it now.

For PC users, we have had the best luck with GitBash, a terminal emulator that uses the same Unix family language as Terminal. This will help you to follow along with the rest of the class. There is also a pretty good manual for GitBash, including installation instructions. For PC users, detailed instructions are here, though you should be using GitBash and NOT PuTTY. Here are some detailed instructions for GitBash specifically.

1.5. Now open a new terminal window connected to your local machine (not GHPCC).

1.6. If you do not already have experience programming on the command line, spend some time doing one of the shell tutorials listed in the "Online access and resources" above.

1.7. We will work together on this unix tutorial: https://swcarpentry.github.io/shell-novice/

3. Cloud Computing

The course assumes the user has access to the MGHPCC cluster group project directory, and KBase. You must use a UMass email address to obtain your MGHPCC login credentials as described in the previous section.

All software described is either already available on the cluster or is available in the group project bin directory. If you are new to UMass MGHPCC, I highly recommend spending some time reviewing the Wiki, especially the pages on submitting jobs and provided software.

This protocol is written in blocks with written descriptions of steps followed by example code. Some of the steps are run in an interactive session, and others are submitted through the job scheduler. The estimated time for assembling a single genome is a few hours from start to finish (this doesn't take guppy-hac base-calling into account, which can take hours to days, depending on the raw data and computer).

1.8. To log in, type the following into your terminal, replacing "username" with your own username that was assigned when you signed up.

```
$ ssh username@ghpcc06.umassrc.org
```

Don't type the "$" symbol; that is the command prompt and signals that your computer is ready to receive a command through the terminal. The exact command prompt can vary, and may look like ">" or even have a prefix before the symbol.

I'll print "$" to denote that I'm showing you a command that you could type into your terminal prompt.

What command would you use to print your working directory?

What command would you use to list the contents of the directory you're in? Using long listing format? Sorted by modification time?

1.9. Use `pwd` to see where you are. Here's an example of what this looks like upon login. Notice that there is a prefix before the command prompt, in this case a dollar sign, that includes the login user name.

```
$ pwd
/home/kd75a
```

1.10. Navigate to our lab folder, and find some information about your instructor.

```
$ cd /home/kd75a/MICROBIO_590B/
$ less about_kristen # press q to quit out of this screen
```

1.11. When you are doing anything beyond basic copying and moving things around on GHPCC, you should be doing this in an interactive session.

When you first log in, you are on the root node. We want to request an interactive session with enough resources to do our work and get off of the login node. We use a job scheduler to do this. Basically, this is a way of handling where on the computer cluster you do your work. The job scheduler that MGHPCC uses is Modules, and here is more information on starting an interactive session.

Here's an example of an interactive job request. This command requests 1G of memory and a single core for two hours in an interactive session.

```
$ bsub -Is -q interactive -n1 -R "span[hosts=1]" -R rusage[mem=1024] -W 2:00
/bin/bash
```

1.12. Routinely make sure you are still in an interactive session. pwd

You should not be working on the head node `username@ghpcc06`. Make it a habit to notice if you are working in an interactive session or have been kicked off onto the root node.

If your interactive session is interrupted or finished, just start a new one. You can usually use the up-arrow to get the last command you ran on the root node.

1.13. In general, you will write batch scripts to run your commands. These scripts are written in a text editor (like nano). In the script, you will specify all the parameters of your command in this job submission script. Then, when you are ready to run the command, you enter `bsub < <batch file>` onto the command line. The command automatically enters a queue on MGHPCC and you are free to do other things.

Structure of a job submission script

The following describes the anatomy of a batch script. Please note that the top of the script must start with `#!/bin/bash` (or whatever interpreter you need, if you don't know, use bash), and then immediately follow with #SBATCH <param> parameters. An example of common SBATCH parameters and a simple script is below; this script will allocate 4 CPUs and one GPU in the GPU partition.

```
#!/bin/bash
#BSUB -n 1                  # Nodes, where X is in the set {1..X}
#BSUB -J job_name           # Job Name
#BSUB -o job-std-out.out    # Output log file
#BSUB -e job-std-err.err    # Error log file
#BSUB -q short              # Queue {short, long, parallel, GPU,
interactive}
#BSUB -W 00:15              # How much time does your job need (HH:MM)
#BSUB -R rusage[mem=500]    # Memory needed (1024 =~ 1 GB ram)

executable command
```

Where:

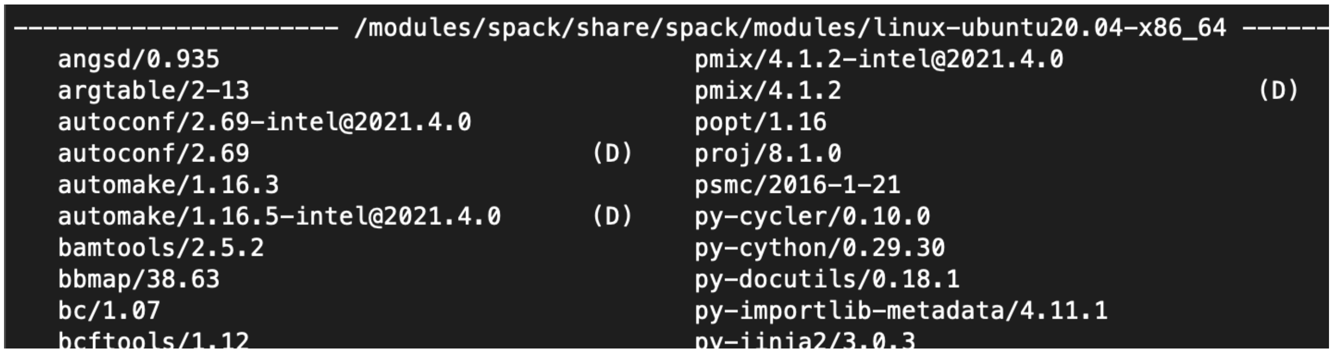
```
  -n = Number of nodes. You will usually use 1 node. The exception would be if
        a specific program could run in parallel (split the big job into many
        small jobs and merge output).
  -J = Job name. This is a unique name that you can use to track the status of
        your job (with the "bjobs" command)
  -o = Standard output (stdout). This is the name of a file that you specify
        to contain the standardized streams of data that are produced by
        command line programs. The stdout file is helpful for debugging, and
        can give helpful information about your command line run. These can
        also contain little or no information. Each program is different.
  -e = Standard error (stderr). This is the name of a file that you specify to
        contain error messages. Depending on the program, error messages are
        sometimes written to the stdout file, so it is important to check both
        files when troubleshooting a failed job.
  -q = Queue. This option specifies which queue to submit your job to. Short =
        jobs < 4 hours, long = jobs > 4 hours.
  -W = Wall time. In this option, you specify how long it will take to run
        your job. It is important to overestimate a you are your job will stop
        prematurely if you exceed the wall time you have specified. You should
        see a message about exceeding wall time limits in your stdout or
        stderr files.
  -R = Memory. This option specifies how much RAM your job needs. This value
        is in megabyte, so to specify 1 GB or ram you would use mem=1000, to
        specify 35 GB of ram, you would use mem=35000.

  executable command = the command line you are executing
```

The executable command may contain a command from a module that is loaded onto our local environment (usually in our lab folder, **/home/kd75a/MICROBIO_590B/**), but it can also contain a command from a module already loaded on MGHPCC.

Many software packages are installed on MGHPCC as "**modules**". By loading software as a module, dependency software is also loaded. For example, if **program_X** uses **java/1.6.0**, **fasttree/2.1.10**, and **python/2.7.14**, loading the **program_X** module will also load the appropriate versions of the aforementioned software packages.

1.14. To view all modules, execute the command below (**module** is the command and **av** is the option, which is an abbreviation of available). You should see a huge list of software packages. Part of this list is shown in a screenshot below.

```
$ module avail
```



```
---------------------- /modules/spack/share/spack/modules/linux-ubuntu20.04-x86_64 ------
   angsd/0.935                                  pmix/4.1.2-intel@2021.4.0
   argtable/2-13                                pmix/4.1.2                            (D)
   autoconf/2.69-intel@2021.4.0                 popt/1.16
   autoconf/2.69                        (D)     proj/8.1.0
   automake/1.16.3                              psmc/2016-1-21
   automake/1.16.5-intel@2021.4.0       (D)     py-cycler/0.10.0
   bamtools/2.5.2                               py-cython/0.29.30
   bbmap/38.63                                  py-docutils/0.18.1
   bc/1.07                                      py-importlib-metadata/4.11.1
   bcftools/1.12                                py-jinja2/3.0.3
```

Why do you think that there are so many versions of the same program available?

If the avail list is too long, consider trying:

```
$ module avail s          # all modules that begin with the letter s
$ module avail sra        # all modules that include the keyword 'sra', for
example;
                         any keyword can be used
```

1.15. To load a specific module the command is "**`module load module_path`**" where **`module_path`** is the full path for a software module as listed with the "**`module avail`**" command.

The NCBI Sequence Read Archive (SRA) has created a software package called the **sratoolkit** to access data, convert data formats, and more. To load the **sratoolkit** module, execute the following command.

```
$ module load sratoolkit/2.10.8
```

When the module is loaded correctly, you should see the following text:

```
sratoolkit 2.10.8 is located under /share/pkg/sratoolkit/2.10.8
```

1.16. In this next lab activity, we will practice submitting a job to the MGHPCC server, retrieving data from the public database NCBI, writing shell scripts using nano in the terminal, and checking on the status of running jobs.

The NCBI SRA contains a massive amount of sequence data from Next Generation Sequencers. Obtaining this data is fairly straightforward and can be accomplished through the cluster. For space reasons, the SRA data is compressed beyond normal "zipping", so you'll have to use specific software to convert from **`SRA`** format to **`FASTQ`** format.

- <u>**NCBI SRA**</u>: https://www.ncbi.nlm.nih.gov/sra

The SRA allows you to download data with accession numbers from publications, or simply by searching for different data attributes (for example, you can search for "human gut" or "*Salmonella enterica*", and then further restrict data to DNA or RNA sequencing etc.).

1.17. Once sratoolkit is loaded, you will now be able to run the "**`prefetch`**" command.

The downloaded files will be in a new folder called **`SRR12931341/`** located in the local folder you were in when you ran the prefetch function.

What command would you use to change your directory location to your home directory?
Hint: there is more than one answer!

To download the "**.sra"** files, use the following command:

```
$ prefetch SRR12931341
```

The .sra files are specially compressed to save space. Unfortunately, we can't use **.sra** files directly. We need to convert **.sra** files to **.fastq** format. Within the **`sratoolkit`** is a program called **`fastq-dump.`**

1.18. Convert your data using the syntax below. This conversion will result in a file compressed with "**`gzip`**". Most software can take **`gzipped`** files as input. Using compressed files saves space.

```
fasterq-dump -O /your/output/directory/ file.sra
```

Remember to include the **FULL PATH** to directories (output directory), and files (.sra file).

Components of the command line:

Calling the script:	fastq-dump
Option 1:	-O (path to output directory; don't use this if you're in the output dir)
Target file:	SRR12931341.sra (give full path)

Record here your full command line (remember to include the full path to your output directory and your .sra file)? *Don't execute the command line yet!*

Creating a fastq.gz file from an SRA file can take a LONG TIME if the file is large (minutes to hours). If we all execute this via command line, we might crash the login node. Therefore, we should submit a command like this as a JOB.

4. Submitting Jobs to the Cloud Server

By submitting this command as a job, the scheduler system can allocate resources across the cluster so that memory is not consumed on a single node. **Any commands estimated to take more than 1 minute, should be submitted as jobs.**

The most straightforward way to submit a job is through a job submission script. This script will give information to the job scheduling system about the resources needed (number of nodes, amount of memory, amount of time *etc.*) so that your job can be run most efficiently.

1.19. (1.19) You can find a example job submission script here:

`/home/kd75a/MICROBIO_590B/sra.bsub`

Copy the example script to your class directory. What command did you use?

The following is an example of a batch script. This script will run the SRAtoolKit to retrieve the genome of *Paludibaculum fermentans* strain:P105, found in the NCBI Sequence Read Archive under accession number SRR12931341.

Example job submission script

```
#!/bin/bash
#BSUB -n 4                    # Number of Cores per Task
#BSUB ---mem=100              # Requested Memory
#BSUB -J sratoolkit-test                 # main job output file name
#BSUB -o SRR12931341-fastq.out           # log file mame
#BSUB -e SRR12931341-fastq.err           # error log file name
#BSUB -R rusage[mem=1024]        # Memory allocated for the job
#BSUB -W 0:30                    # time for job to complete in HH:MM
#BSUB -q short          # queue length: short (up to 4h), long (up to 30d)

module load sratoolkit/2.10.8
fasterq-dump -O /home/kd75a/MICROBIO_590B/SRR12931341 \
/home/kd75a/MICROBIO_590B/SRR12931341/SRR12931341.sra
```

***Using the "\" at the end of a command line will tell the unix system that the next line is part of the same command. It is used here to better visualize the command line because the command line is wider than the width of the page. This is optional and purely for convenience.
In this example we are actually executing 2 commands.

```
1. module load sratoolkit/2.10.8
```

This loads the sratoolkit so we can use the fastq-dump program

```
2. fasterq-dump -O /home/kd75a/MICROBIO_590B/SRR12931341 \
/home/kd75a/MICROBIO_590B/SRR12931341/SRR12931341.sra
```

The executable command is “fasterq-dump” (convert SRA file format to fastq file format)

- The options are:
  - **`“-O”` (the full path to the output directory)**
  - **with more described here: https://hpc.nih.gov/apps/sratoolkit.html**
- The target is: **`SRR12931341.sra` (full path to file name)**

1.20. Use a text editor (like nano) to edit the job submission script to run the **`fasterq-dump`**.

To use nano, type “nano” on the command line, and start typing! This program is controlled all by keyboard commands, and available commands are listed at the bottom of the screen. Here is a brief tutorial for more information on nano.

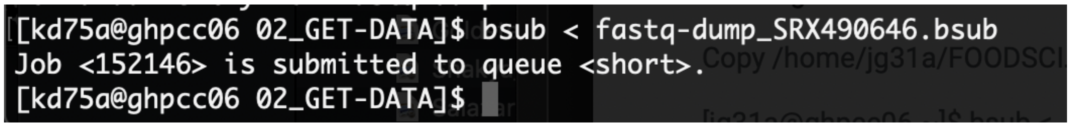
You can also type nano and then follow this with the name of the file you want to create. This command brings up an empty file where you can type in your commands with this file name:

```
$ nano sra.bsub
```

Finally, you can also type nano and the name of an existing file, like the example template.bsub file, edit this, and then save this as a different name.

1.21. To submit the job to the scheduler, execute the **`bsub`** command (make sure this file name matches the one above):

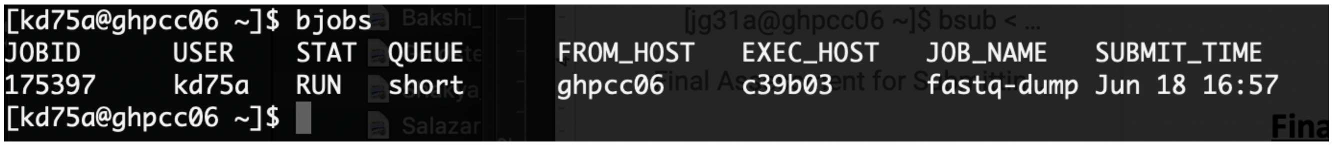
```
bsub < sra.bsub
```



```
[kd75a@ghpcc06 02_GET-DATA]$ bsub < fastq-dump_SRX490646.bsub
Job <152146> is submitted to queue <short>.
[kd75a@ghpcc06 02_GET-DATA]$
```

1.22. To check the status of your job, use the **`bjobs`** command.

```
$ bjobs
```



```
[kd75a@ghpcc06 ~]$ bjobs
JOBID     USER    STAT  QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME    SUBMIT_TIME
175397    kd75a   RUN   short      ghpcc06     c39b03      fastq-dump  Jun 18 16:57
[kd75a@ghpcc06 ~]$
```

- **JOB ID = your individual job ID (generated from job scheduler)**

```
o USER           = your user ID
o STAT           = status of the job (PD = pending, RUN = running)
o QUEUE          = which queue (short, long, interactive)
o SUBMIT_TIME    = time job began (HH:MM)
o JOB_NAME       = the number of nodes in use
```

1.23. If the job executes successfully, you should have a "`.out`" file and a "`.err`" file. The "`.out`" file has information about your job.

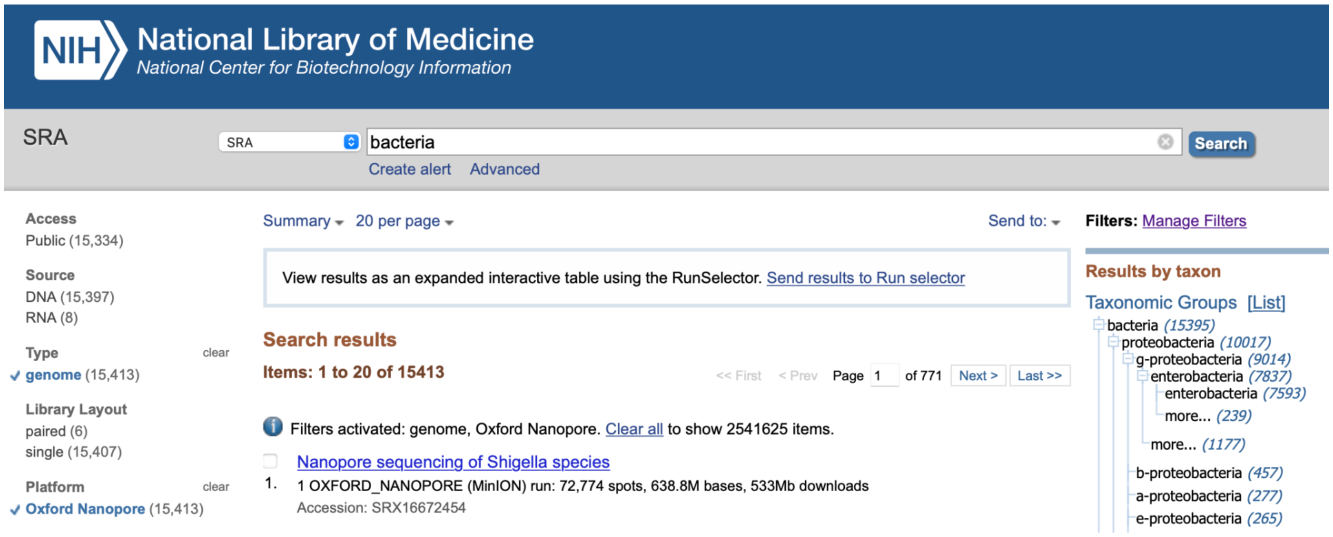
What command can you use to look at the contents of the "`.out`" file?

1.24. The command fastq-dump creates a zipped file, indicated by the file extension "**.gz**". To unzip, use the command gunzip. This may not be necessary!

```
$ gunzip SRR12931341.fastq.gz
```

Once this is completed, the fastq files are ready for analysis.

1.25. Next, find an SRA file to download by searching the SRA for a bacterial genome generated using Oxford Nanopore technology. Try starting here: https://www.ncbi.nlm.nih.gov/sra/?term=bacteria



Access
Public (15,334)

Source
DNA (15,397)
RNA (8)

Type clear
✓ genome (15,413)

Library Layout
paired (6)
single (15,407)

Platform clear
✓ Oxford Nanopore (15,413)

View results as an expanded interactive table using the RunSelector. Send results to Run selector

Search results

Items: 1 to 20 of 15413

<< First < Prev Page 1 of 771 Next > Last >>

Filters activated: genome, Oxford Nanopore. Clear all to show 2541625 items.

1. Nanopore sequencing of Shigella species
1 OXFORD_NANOPORE (MinION) run: 72,774 spots, 638.8M bases, 533Mb downloads
Accession: SRX16672454

Results by taxon
Taxonomic Groups [List]
bacteria (15395)
proteobacteria (10017)
g-proteobacteria (9014)
enterobacteria (7837)
enterobacteria (7593)
more... (239)
more... (1177)
b-proteobacteria (457)
a-proteobacteria (277)
e-proteobacteria (265)

The course activities are set up in four parts. In **Part 1**, we gained access to our computing resources, practiced coding in the shell, worked with sequence data, and submitted jobs to the GHPCC cloud computer.

In **Part 2** (*this part*), we begin work with the raw, basecalled sequence data from the bacterial isolate Ralstonia sp. GP101. In this part, we will work together to subset the data, assemble the genome, and perform some additional quality control assessments.

In **Part 3**, we continue working with the GP101 genome by annotating the assembly, and begin to explore the coding and non-coding regions of the genome. We will also perform phylogenetic, sequence and functional gene comparative analysis with other closely related strains.

In **Part 4**, you will be assigned your own genome that is the basis of your capstone project. These genomes are derived from bacteria we have isolated in our lab. They have been sequenced on the MinION, basecalled and QC checked to ensure that they are ready for you. Their genotype and some phenotypic information is available. You will repeat the analyses we did on GP101 for assembly (**part 2**) and annotation (**part 3**) on your genome, and write it up for your capstone project report.

1. Overview of Protocol for minION Genome Assembly

Mallory Choudoir v.2021.01.14
Kristen DeAngelis

Overview: This protocol describes the pipeline for *de novo* assembly and quality assessment of genomes sequenced on the minION in the DeAngelis Lab. The initial input for this pipeline includes the raw sequence files (fast5) generated from rapid base calling with MinKNOW. The final output of this pipeline is a draft assembly that will be uploaded to IMG for annotation and archiving.

Major pipeline steps (and the associated software):

1. High-accuracy base calling (HAC) with Guppy (done ahead of time)
2. Subsample based on read quality with Filtlong
3. *De novo* assembly with Flye
4. Generate a consensus assembly with Racon
5. Final polishing with Medaka
6. Quality assessment with Quast and CheckM

In section 3, we will begin to perform our example assembly for strain GP101, following the pipeline above. But before we begin that, we need to organize our directories, check permissions, and plan for future files.

2. Directory structures

2.1. At this point you should be on the GHPCC cluster. If you haven't done any work in terminal on your local machine since your last login, you can likely just hit the up-arrow button. This scrolls backwards through your command history.

2.2. Start an interactive session.

Review the details of starting a batch or interactive session here.

```
 $ bsub -Is -q interactive -n1 -R "span[hosts=1]" -R rusage[mem=1024] -W 3:00
/bin/bash
```

When you are doing anything beyond basic copying and moving things around on GHPCC, you should be doing this in an interactive session. This command requests 4GB of memory and two cores in an interactive session.

Routinely make sure you are still in an interactive session. You should not be working on the head node `username@login1`. If your interactive session is interrupted or finished, just start a new one.

Which prompts below suggest that the user is in an interactive session?

```
a. [kd75a@ghpcc06 ~]$
b. [kd75a@c39b16 ~]$
c. [jg31a@c05b10 ~]$
d. [jg31a@ghpcc06 ~]$
```

2.3. Before moving on, take some time to organize your home directory by making subdirectories for your problem sets, practice genome assembly (GP101), and your capstone assembly.

The class directory is configured so that you cannot write there, so you will have to copy files from the class directory to your work directory to do the work.

What command shows you the permissions attached to the class directory?

What part of the output of that command indicates the permissions that you have as part of our group (those whose PI is indicated as pi_kristen_deangelis)? What if you are not in that group?

2.4. Navigate to your home directory, if you aren't there already.

```
$ cd ~
```

2.5. Make a directory called MICROBIO_590B in your work directory.

```
$ pwd
/home/kd75a/
$ mkdir MICROBIO_590B
$ ls -lt
total 1
drwxrwxr-x 2 kristend_umass_edu kristend_umass_edu 2 Jul 26 14:59 MICROBIO_590B
```

Next, make a directory called `GP101` in your `MICROBIO_590B` directory. This is the folder where we will do the analysis for assembling the GP101 genome.

2.6. We want to set up some directories in our work space that will accommodate the work we do in class. Being organized now will help a lot later, as we generate more and more files.

Make a directory called `GP101` in your `MICROBIO_590B` directory.

2.7. Navigate to the folder `/home/kd75a/MICROBIO_590B/GP101`

Notice that the MinION generates many fastq files from each run, and that the permissions are set up so that we cannot write into this directory.

We could copy all these files, but they are huge and we don't need them. What we will do instead is to write one new file, into our own directory, with all the fasta files concatenated together.

3. Subsample reads based on quality

For *de novo* assembly we want to subsample the reads to about 30-50X coverage, and we want to select these based on quality using the software Filtlong. Both input and output files are fastq.

2.8. Navigate back to the folder where you want your fastq files to end up.

If you need to be reminded of the address of your work directory, go to /work and print the directory information. Note that on the command line, everything after the hashtag ("#") is ignored by the computer; this is a way of annotating scripts.

```
$ cd ~    # command to go home
$ cd /work/<username>
$ pwd     # prints the working directory
$ ls -lt # prints the local directory contents in long form (-l) sorted by
            time last modified (-l)
$ ls -lt MICROBIO_590B/  # prints the specified directory contents
```

2.9. Combine all of the fastq files into a single file using cat, but specify the full directory information of the output file.

My directory permissions is set to write-only, so you'll have to use the redirect operator ">" to write into your own directory. Navigate to your GP101 directory (or check that you're already there) to run this command.

```
$ cat /project/uma_kristen_deangelis/GP101/fastq* > GP101_all_guppy.fastq
```

There are 186 fastq files, so this command might take a few seconds to execute.

2.10. Explore this new concatenated file a bit using the commands head and tail, which preview your file in a screen print output.

```
$ head -8 GP101_all_guppy.fastq
$ tail -8 GP101_all_guppy.fastq
```

2.11. Another way to check that this worked correctly, and to make sure your fastq file is correctly formatted, is to count the number of lines in your concatenated file. It should be divisible by 4. To do this, use the command `wc`.

```
$ wc -l GP101_all_guppy.fastq
```

2935548 / 4 = 733,887

2935548 /project/uma_kristen_deangelis/GP101/GP101_all_guppy.fastq
2726653 GP101_all_guppy.fastq

Count lines. For example, 2935548 is divisible by 4, which is good! If this is not the case, try the cat command again.

The following steps are only necessary if the lines are ***not*** divisible by 4. You can also use the FastQValidator program. This program gives you error messages related to fastq formatting criteria. The executable is located in uma_kristen_deangelis/bin and requires a fastq file as input.

```
$ /project/uma_kristen_deangelis/bin/fastQValidator/bin/fastQValidator
--file GP101_all_guppy.fastq
```

You might get errors like the following, which you can ignore:

```
ERROR on Line 3: Sequence Identifier on '+' line does not equal the one on the '@'
line.
') in quality string.lid character ('
```

2.12. We need to now calculate the coverage of all of the reads. Coverage is a measure of the number of total base pairs in the combined fastq file divided by our estimate of genome size.

The following commands are all ways to count the number of base pairs in your fastq.

```
$ grep "^[ACGTN]" GP101_all_guppy.fastq | tr -d "\n" | wc -m
$ cat test.fastq | paste - - - - | cut -f 2 | tr -d '\n' | wc -c
  # print the file
  # paste - - - -: print four consecutive lines in one row (tab delimited)
  # cut -f 2: print only the second column (after paste this is the second line of
the fastq-format, meaning the sequence)
  # trim newline characters
  # wc -c: count the characters
```

Which part of the above command contains the regular expression?

What does the carat (^) mean?

What do the square brackets ([ ]) mean?

Here are some other variations for other file types.

```
$ zgrep "^[ACGTN]" GP101_all_guppy.fastq | tr -d "\n" | wc -m  # use this if your
fastq is compressed, like from gpu-guppy
$ grep -v ">" file.fasta | wc | awk '{print $3-$1}' # only works for fasta
```

2.13. Estimate the genome size of your isolate by assuming it's approximately the same size as another genome in our collection, or find an estimate on the internet. Use the 16S rRNA gene sequence to find a closely related genome using NCBI BLAST.

For example, the genome size of a Burkholderia strain closely related to GP101 was 5.4 Mb, so 40X would be 216,000,000 bp. Our estimated coverage is 5646076454/5400000 = 1045.6 so unfiltered reads are 1000X! (This is a lot.) Record this in the class assembly tracking sheet (like our lab assembly tracking sheet).

What is the estimated coverage of your genome?

2.14. Now we need calculate how much sequence we want FiltLong to put in our subsampled file (*i.e.* coverage). The MinION generates way more sequence than we can practically assemble, so we subsample based on sequence length and sequence quality.

Calculate this by multiplying the estimated genome size by desired coverage. For example, for a 5.4 Mb genome 40X coverage would be 5,191,830 x 40 = 207,673,200 bp.

The FiltLong binary is located in the lab group folder. Use this command to subsample based on read quality with Filtlong.

```
$ /project/uma_kristen_deangelis/bin/Filtlong/bin/filtlong -t 216000000 --
min_length 1000 --min_mean_q 85 GP101_all_guppy.fastq > GP101_all_guppy_40X.fastq
```

2.15. Before proceeding to the assembly step, let's once again make sure our subsample fastq format is valid using the program FastQValidator.

```
$ /project/uma_kristen_deangelis/bin/fastQValidator/bin/fastQValidator --file
GP101_all_guppy_40X.fastq
```

4. *De novo* assembly with Flye

2.16. The Flye assembler is available pre-installed on the cluster as a conda environment. Conda is a package and environment management system that can make installing software much easier.

2.17. We will submit this job to the short queue using a shell script (example below named **flye.bsub**) and the bsub command.

To execute this next command, you will need the estimated genome size that you obtained from your search conducted in step **2.12**. This only needs to be an approximation rounded to the nearest 0.5 Mbp.

The first part of the script is about job resources (memory, time, etc.) and the bottom part is the code the job will run. Parameters you will need to modify are (1) --nano-raw (your subsampled fastq), (2) --out-dir (output directory), (3) -g (genome size in Mb)

```
#!/bin/bash
#BSUB -n 6
#BSUB -J flye_GP101
#BSUB -o flye_GP101.out
#BSUB -e flye_GP101.err
#BSUB -R rusage[mem=2048]
#BSUB -W 1:00
#BSUB -q short
#BSUB -R span[hosts=1]

module load flye/2.8.1
source /share/pkg/condas/2018-05-11/bin/activate && conda activate flye_2.8.1_py38
flye --nano-raw GP101_all_guppy_40X.fastq --out-dir flye --threads 4 -g 5.4m
conda deactivate && conda deactivate
```

Once this script is written and saved, submit to the MGHPCC using the following command.

```
$ bsub < flye.bsub
$ flye.bsub > bsub
```

You may need to increase the memory or time request for your assembly job to complete (e.g. -n 6 cores and -W 2:00 hours).

Default terminal does not work well with copy and paste, so if this (or any command you copy and paste) isn't working, you may need to type this out or use a template found HERE. Remember to use tab-complete to make the typing go faster and with fewer mistakes.

2.18. Check the status of a job using "bjobs"

2.19. If you are having problems submitting your job, you may have to change the permissions of your script! To check permissions use 'ls -l yourfile' and to change permission use 'chmod XXX yourfile'.

2.20. After the job is complete, the finished assembly will be in the file named assembly.fasta, but we probably want to rename this something more useful.

```
$ mv flye/assembly.fasta flye/GP101_flye_assembly.fasta ## rename assembly
```

5. Generate a consensus assembly with Racon and Minimap2

This step requires two programs, Minimap2 which is pre-installed on the cluster and Racon, which is installed in the project bin directory. Minimap2 is a versatile mapper and pairwise aligner for nucleotide sequences. It works with short reads, assembly contigs and long noisy genomic DNA reads. The purpose of Racon is to create a genomic consensus with improved quality compared to the *de novo* assembly.

Racon requires three input files, assembly contigs in fasta format, subsampled reads in fastq, and an alignment file between the reads and the contigs in sam format. The output is also a fasta.

Both of these steps can be done in an interactive session (see section 2). If you don't have one running, start an interactive session now.

2.21. First, build the alignment/mapping sam file with minimap2.

```
$ module load minimap2/2.17
$ minimap2 -a flye/GP101_flye_assembly.fasta GP101_all_guppy_40X.fastq >
GP101_flye.sam
```

2.22. Next run Racon. Both of these should only take a few minutes max each.

```
$ /project/uma_kristen_deangelis/bin/racon/build/bin/racon
GP101_all_guppy_40X.fastq GP101_flye.sam flye/GP101_flye_assembly.fasta >
GP101_racon_assembly.fasta
```

6. Final polishing step with medaka

The last step is a final polishing with medaka, which includes configuring your conda environment. Medaka is installed with conda located in the project directory /project/uma_kristen_deangelis/.conda
The '.' before the file name means that it is invisible, if you want to see it, you can use the command

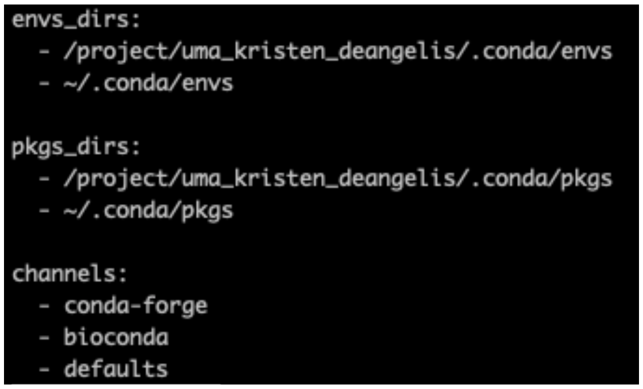
```
$ cd ~
$ ls -a
```

If you haven't created a .condarc file in your work directory, do that next. Otherwise, you can skip to the next step.
.

2.23. To configure conda so you can install your own environments, you need to create and modify a file in your work directory named .condarc that tells conda where to look for environments that are installed locally (and, in the project directory). To create this file, run

```
$ cp /share/pkg/condas/2018-05-11/condarc_example ~/.condarc
```

Now you need to modify this file to tell conda to always look first in the .conda file in the project directory. Navigate to your work file and open .condarc using nano. Conda will look to the first location listed, so you want this to be /project/uma_kristen_deangelis/.conda/env for the envs_dirs and /project/uma_kristen_deangelis/.conda/pkgs for the pkgs_dirs. Your .condarc file should look something like this:



```
envs_dirs:
  - /project/uma_kristen_deangelis/.conda/envs
  - ~/.conda/envs

pkgs_dirs:
  - /project/uma_kristen_deangelis/.conda/pkgs
  - ~/.conda/pkgs

channels:
  - conda-forge
  - bioconda
  - defaults
```

Now, when you load the condas module and list the available environments, at the top of the list you should see medaka located in /uma_kristen_deangelis/.conda/env, and below that all of the conda environments that are available pre-installed on the cluster.

```
$ module load condas/2018-05-11
$ conda env list
```

Alternatively, you can install medaka in your work directory. This is relatively straight forward with these installation instructions here. If you choose this option, you do not need to modify your ~/.condarc file.

Once you're done with this step, navigate back to your GP101 directory to configure a .bsub file to run medaka.

2.24. After we've configured our .condarc file, we can run medaka. Similar to Flye, we will submit this job with a shell script named medaka.bsub

Medaka needs two input files, the subsampled fastq reads and the fasta consensus assembly output from Racon. The output is also a fasta.

Use this template to create a file called medaka.bsub using nano. Parameters you will need to modify are (i) --i subsampled_reads.fastq, (ii) --racon_assembly.fasta, (iii) dates if you like, and (iv) and output if you like --o output_dir

```
#!/bin/bash
#BSUB -n 4
#BSUB -J medaka_20210625
#BSUB -o medaka_20210625.out
#BSUB -e medaka_20210625.err
#BSUB -R rusage[mem=2048]
#BSUB -W 1:00
#BSUB -q short
#BSUB -R 'span[hosts=1]'

module load condas/2018-05-11
source activate medaka36
medaka_consensus -i GP101_all_guppy_40X.fastq -d GP101_racon_assembly.fasta \
 -o medaka -m r941_min_high_g351
## change the model to r941_min_high_g360 for Guppy-GPU v4.5.4 or Guppy v5.0.16
conda deactivate && conda deactivate
```

The model (--m) option is selected based on 1) pore type 2) sequencing platform (min == minION) and 3) base calling method (high_gXXX == high accuracy guppy). We used either Guppy-CPU v3.5.2 or Guppy-GPU v4.5.4. Use r941_min_high_g351 for v3.5.2 and r941_min_high_g360 for v4.5.4. See recommendations here.

Our capstone genomes (2022) were sequencing by SeqCenter (seqcenter.com). Nanopore samples were prepared for sequencing using Oxford Nanopore's "Genomic DNA by Ligation" kit and protocol. All samples were run on Nanopore R9 flow cells (R9.4.1) on a MinION. Post-sequencing basecalling was performed by Guppy, version 5.0.16 in high-accuracy basecalling mode. Based on these methods, the r941_min_high_g360 model is recommended.

Once your .bsub file is configured, submit the job to the queue.

```
$ bsub < medaka.bsub
```

If this doesn't work, you should exit the interactive session and start a new one.

2.25. If you need to rerun medaka, it will not overwrite old files. So you will need to manually remove these files before generating new ones (or make sure to rename them).

2.26. After this job finishes you should have a pretty decent genome assembly! Let's again rename the output to something more useful (i.e. GP101_final_assembly.fasta)

```
$ mv medaka/consensus.fasta GP101_final_assembly_40X.fasta ## rename final
assembly
```

7. Use QUAST to check your assembly

2.27. There are a few ways to check to determine how good your assembly is. One strategy is to measure assembly architecture (e.g. how many fragments is your assembly?), and another is to use the sequence information to determine completeness (e.g. how many conserved single copy genes are present?) For the first part, we will use a program written in python called QUAST which is already installed on the cluster as a module.

2.28. Make sure you are still in an interactive session! Then load the quast/5.0.2 module. You may get a number of errors about conflicting software versions, don't worry about these. The basic usage requires your assembly fasta file and the name of your output directory (you can name this quast) and is called using the command quast.py.

```
## Are you still in an interactive bash session? Good :)
$ module load quast/5.0.2
$ quast.py -o quast GP101_final_assembly.fasta
```

2.29. Use the less command to explore your output. In the output directory is a file named report.txt with summary stats. There are also a bunch of other output files generated that may be of interest.

```
$ less quast/report.txt
```

2.30. At minimum, take a look at report.txt and note the number of contigs, total length, GC, and N50. Record this information on your tracking sheet. If we are still happy, you can now move this file to your laptop using scp (initiated from your personal computer)!

8. Construct a new assembling using more raw data

This step is optional for the test run we will do together in class, but is required for your project genome.

In step 2 of this part, we subsampled reads based on quality. Using filtlong, we included enough of the highest quality reads to achieve an estimated 40X final genome coverage. Will we get an improved assembly if we include enough reads to achieve an estimated 50X final genome coverage?

2.31. Calculate how many basepairs you would need to achieve 50X final coverage. For example, for a 5.4 Mb genome 50X coverage would be 5.4e6 x 50 = 270,000,000 bp.

2.32. Return to step 2.6, and run filtlong using these new parameters. Don't forget to rename the output file to reflect this expanded amount of data.

```
$ /project/uma_kristen_deangelis/bin/Filtlong/bin/filtlong -t 270000000 --
min_length 1000 --min_mean_q 85 GP101_all_guppy.fastq > GP101_all_guppy_50X.fastq
```

2.33. Before proceeding to the assembly step, let's once again make sure our subsample fastq format is valid using the program FastQValidator.

```
$ /project/uma_kristen_deangelis/bin/fastQValidator/bin/fastQValidator
--file GP101_all_guppy_50X.fastq
```

2.34. Continue through all the steps in parts 4 through 7 (de novo assembly with Flye, consensus assembly with Racon and Minimap2, polishing with Medaka, quality control assessment using QUAST).

Be sure to update your output file names to reflect this new 50X assembly, so you don't accidentally overwrite your old files!

2.35. Record this new information on your tracking sheet.

Which amount of data resulted in a better genome assembly? What is the evidence that indicates which assembly is better?

2.36. You can now choose one assembly and move this file to your laptop using scp (initiated from your personal computer).

```
$ scp
an90a@ghpcc06.umassrc.org:/project/uma_kristen_deangelis/minIONgenomes/GP101/medak
a/*fasta ~/

$ scp
kd75a@ghpcc06.umassrc.org:/home/kd75a/MICROBIO590b/GP101/GP101_final_assembly.fast
a \ ~/Desktop
\ ./Desktop
```

2.37. Upload your final assembly to KBase.

9. Quality assessment

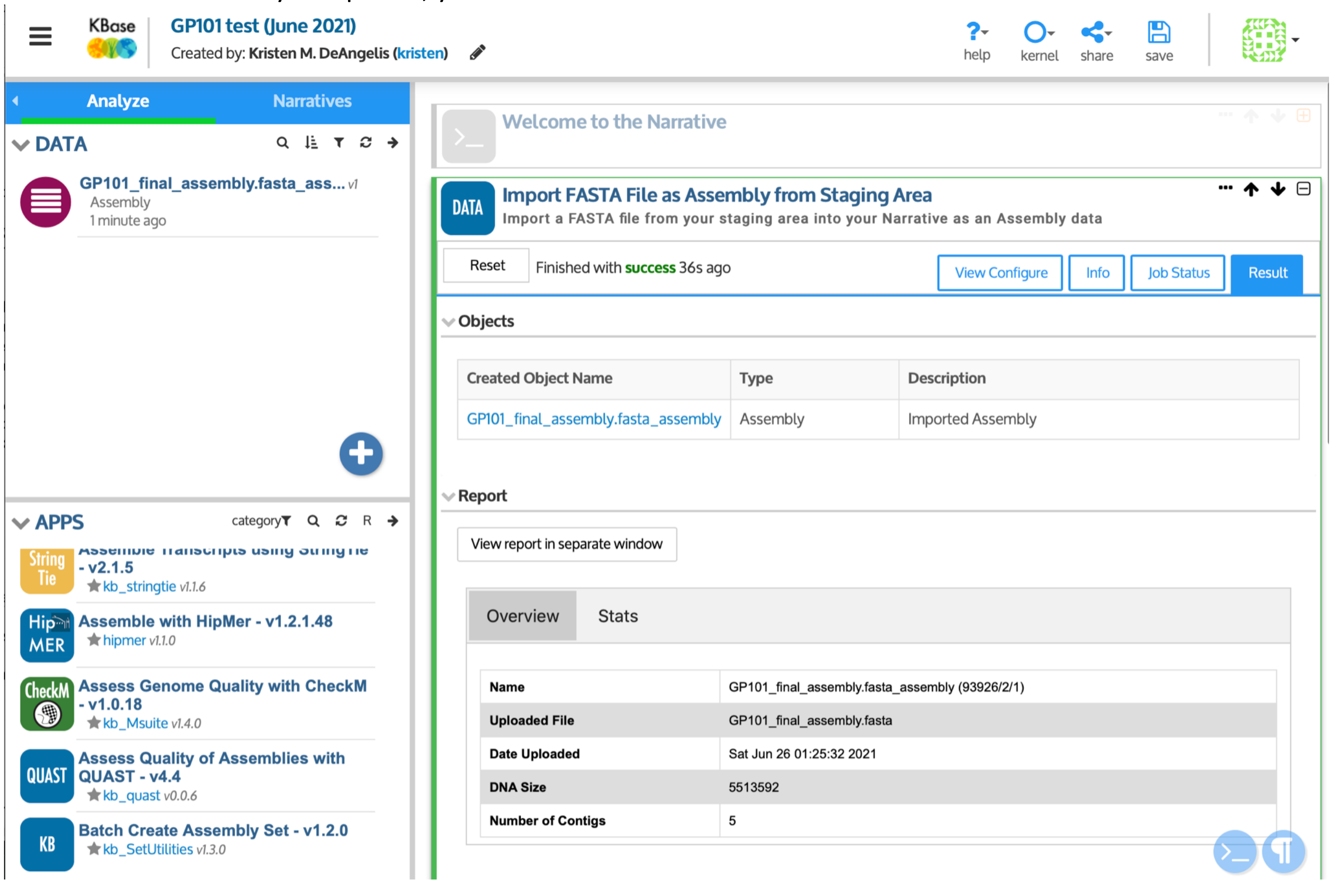
2.38. Upload your final assembly to KBase.

When you login to KBase, click the large green button in the upper righthand corner "**+ New Narrative**". KBase will open a new window to an Untitled Narrative workbook. Click on the title "**Untitled Narrative**" and give the new narrative a descriptive name, using the format "**GP101_username_2021**".

Before you proceed, share the Narrative by clicking on the "**share**" button in the upper right-hand corner. Under users, type "**kristen**" then use the dropdown menu to select "**Edit, save and share**", and finally click "**Apply**".

Click the "**Add Data**" box in the upper lefthand corner. A new set of menus will appear. Here you can choose "**Import**," and drag and drop or select your finished assembly fasta file. Use the dropdown menu "**Import As…**" to select "**Assembly**." Then click the "**Upload**" button (it looks like a box with an up-arrow). This creates your first App in the Narrative. Keeping the defaults, select "**Run**".

Once this is successfully completed, your screen should look like this.



KBase GP101 test (June 2021)
Created by: Kristen M. DeAngelis (kristen)

GP101_final_assembly.fasta_ass... v1
Assembly
1 minute ago

Assemble Transcripts using StringTie - v2.1.5
kb_stringtie v1.1.6
Assemble with HipMer - v1.2.1.48
hipmer v1.1.0
Assess Genome Quality with CheckM - v1.0.18
kb_Msuite v1.4.0
Assess Quality of Assemblies with QUAST - v4.4
kb_quast v0.0.6
Batch Create Assembly Set - v1.2.0
kb_SetUtilities v1.3.0

Welcome to the Narrative

Import FASTA File as Assembly from Staging Area
Import a FASTA file from your staging area into your Narrative as an Assembly data

Finished with success 36s ago

Objects

Created Object Name	Type	Description
GP101_final_assembly.fasta_assembly	Assembly	Imported Assembly

Report

Name	GP101_final_assembly.fasta_assembly (93926/2/1)
Uploaded File	GP101_final_assembly.fasta
Date Uploaded	Sat Jun 26 01:25:32 2021
DNA Size	5513592
Number of Contigs	5

This shows that the assembly was completed in 5 contigs with a total of 5.5 Mbp.

2.39. Under the Genome Assembly apps, assess the quality of your assembly with QUAST and CheckM. QUAST will give you information about assembly architecture, including number of contigs, assembly length, and N50. CheckM will give you information on completeness, contamination, and taxonomy.

To do this, choose these programs from the Apps folder in the lower left-hand corner of the Narrative. When you click on the app, it appears in the Narrative pane on the right. Ensure that the narrative is in order by using the arrows in the upper right-hand corner of each app box. For this part, the correct order is FASTA, QUAST, CheckM.

CheckM and QUAST require the output of the Data app to work, so you have to wait for the import to be completed before you can run these QC programs.

Once you've completed your QC, update your progress on your genome and track the assembly stats on this minion-seq-assembly-log-spreadsheet.

If the genome is in >6 contigs, <96% complete, or >3% contamination we probably want to optimize the assembly.

If you need to repeat and optimize the assembly, organize data from each assembly independently (e.g. create a directory named assembly_v1 in the genome folder). Name your assemblies by version (i.e. assembly_v1, assembly_v2) for the tracking sheets.

This lab module includes a series of activities that will guide you through the analysis of a genome of a bacterium using KBase. KBase is a “software and data science platform designed to meet the grand challenge of systems biology: predicting and designing biological function”.

The course activities are set up in four parts. In **Part 1**, we gained access to our computing resources, practiced coding in the shell, worked with sequence data, and submitted jobs to the MGHPCC cloud computer.

In **Part 2**, we worked with the raw, basecalled sequence data from the bacterial isolate Ralstonia sp. GP101. We will work together to subset the data, assemble the genome, and perform some additional quality control assessments.

In **Part 3** (*this part*), we continue working with the GP101 genome by annotating the assembly, and begin to explore the coding and non-coding regions of the genome. We will also perform phylogenetic, sequence and functional gene comparative analysis with other closely related strains. This manual includes an introduction to KBase, genome annotation, classification (taxonomy and phylogeny), and an exploration of genome characteristics, exploration of carbohydrate-active enzymes, calculation of Average Nucleotide Identity (ANI) using nearest relatives, and pangenome analysis.

In **Part 4**, you will be assigned your own genome that is the basis of your capstone project. These genomes are derived from bacteria we have isolated in our lab. They have been sequenced on the MinION, basecalled and QC checked to ensure that they are ready for you. Their genotype and some phenotypic information is available. You will repeat the analyses we did on GP101 for assembly (**part 2**) and annotation (**part 3**) on your genome, and write it up for your capstone project report.

In this section, you will apply various bioinformatics analyses to a whole genome sequence. You will apply information you have learned in this class, and possibly others, to the analysis of the genome. Overall, you will be telling a story about the bacterium.

If you find that pages are slow to load, a private browser window might help (especially for Firefox).

All KBase questions must be posted to the Moodle discussion dedicated to KBase. Please help when you can by providing answers to other students’ questions. I will check the discussion board regularly, but don’t expect answers during the hours leading up to when assignments are due. Avoid answering each other’s questions in private communications as this may perpetuate a wrong answer.

Acknowledgements
This part is adapted from Dr. Maureen Morrow, SUNY New Paltz, and KBase in the Classroom: Genome Exploration, Authors: Steven Biller and Ellen Dow.

1. Resources

Shell/command line tutorials
Beginners Linux Tutorial Learning the shell: http://www.linuxcommand.org/lc3_learning_the_shell.php
Command line / unix / linux tutorial - murder mystery! https://github.com/veltman/clmystery
Unix reference guide: https://sites.tufts.edu/cbi/files/2013/01/linux_cheat_sheet.pdf
**** N.B.: this is one of many available guides! Please tell me if you find a better one…*

MGHPCC
Request an account here: https://www.umassrc.org/hpc/
Or, if you have an account, email me with your account name.
MGHPCC Wiki: https://www.umassrc.org/wiki/index.php/Main_Page
MGHPCC FAQ: https://www.umassrc.org/wiki/index.php/Frequently_Asked_Questions
Our course directory on the MGHPCC: /home/kd75a/MICROBIO_590B/
GHPCC File Transfer: https://www.umassrc.org:444

KBase
URL: https://narrative.kbase.us/#signup
Sign up for a free account to get access to genomes and tools for analysis.
Watch this video for more information: https://docs.kbase.us/getting-started/quick-start
**** N.B.: Share the KBase Narratives you create in this class with KBase username "kristen"*

Other
Class Genome Database: Class-minion-seq-assembly-log
Markdown Basic Syntax Guide: https://www.markdownguide.org/basic-syntax
FastQC Tutorial & FAQ: https://rtsf.natsci.msu.edu/genomics/tech-notes/fastqc-tutorial-and-faq/
OpenStax Microbiology textbook for non-majors: https://openstax.org/details/books/microbiology
Cell Biology by the Numbers: http://book.bionumbers.org

2. Introduction

Initially, we will continue working with the data for the recently sequenced bacteria GP101. Once you are assigned your genome for your capstone project, you will repeat these analyses using this new data.

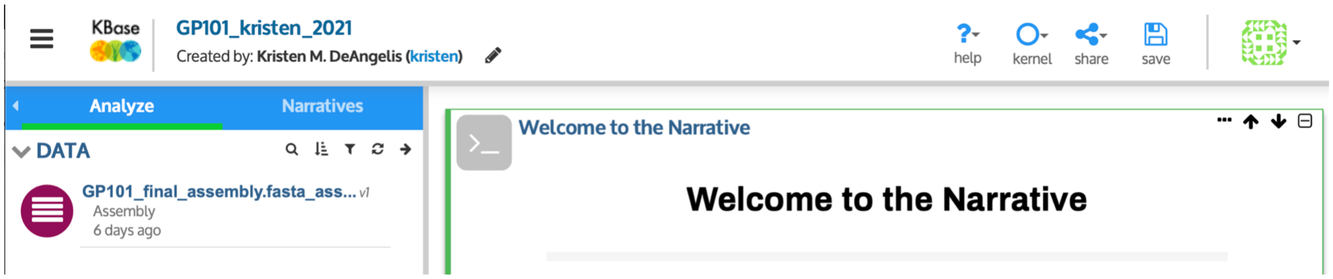
These data were created using NextGen sequencing with the goal of determining the entire DNA sequence of the bacterium, also known as whole genome sequencing (WGS). After you gain experience with a variety of programs and KBase apps, you will work with a genome that has not been previously studied. This genome will be the basis of your independent Genome Analysis Project.

3.1. Set up your KBase account.

This should have already been done, but take the time to ensure that the account is set up and that you are able to log in using your preferred browser.

3.2. Open the narrative we created at the end of Part 2, or create a new narrative. Name (or rename) it: **GenomeName_username_2022**. To rename it, first click on 'Untitled' in upper left of page, then enter the new name: **GenomeName_username_2022**

For example, my narrative for the example genome would be kristen:GP101. I'm using 'kristen' because that's my KBase user ID. Please use your KBase user ID as your name here!



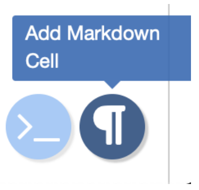
Take the Narrative tour (located in the '**help**' menu in the upper right hand corner) on your own time to get to know how KBase is set up.

3.3. Share the narrative with your professor.

Before you proceed, share the Narrative by clicking on the "**share**" button in the upper right-hand corner. Under users, type "**kristen**" then use the dropdown menu to select "**Edit, save and share**", and finally click "**Apply**".

You may have done this already, but check that this is shared.

3.4. Create your lab notebook in Kbase.



To create your notebook, you will need a new markdown cell. It looks like the welcome cell that is automatically included; it's icon is a grey box with the paragraph symbols (like a backwards capital P). The button to add a markdown cell is at the bottom right hand corner of the page.

You need to use markdown language to record information in the cell. The changes are not automatically saved. Click the save icon often!!!!!!!!!

You may wish to view (or review) the [KBase Documentation](), which is an extensive user guide to the KBase Narrative and its application towards genomics.

Please use the following KBase notebook guidelines each time you use KBase. These notes will be recorded in the Narrative dach time you use KBase. For each new day or time you work in KBase, indicate:

- Date (bold)
- goals for that day
- apps you used
- data that was downloaded
- other relevant information or observations, as appropriate

3.5. Examine the markdown language

To see the markdown for the welcome cell, double click on the cell window. Note that the title is proceed by ##. This is part of the markdown language and it indicates the heading font. The number of # indicates the level of the heading.

Some other markdown shortcuts:

- Note that ** before and after text converts the text to bold.
- <br> creates a line break
- The language for subscript would look like: N<sub>50</sub> for N_{50}
- To add a link, copy the address of page into your narrative. Place parentheses around the address. Immediately in front of the parentheses, give the link a name such as "FASTQC Staph. aureus reads" and surround that name with brackets. Example: [FASTQ Staph. aureus raw reads](https://kbase.us/dynserv/66e9a37e7d3a56b64cc372c89d1eaafaa605c397.HTMLFileSetServ/api/v1/80311/5/1/$/0/index.html)

Play around with this language to build your skills. To learn more markdown language, see the Markdown Guide Basic Syntax.

To write in the markdown cell you just created,

a) double click the text (not the title). A grey bar should appear.
b) type into the new cell. Click save.
c) Click outside of the cell to see the changes.
d) Double click on the cell to see the markdown language again.
e) use the ## heading to label the cell as your notebook
f) after a paragraph break, indicate the date in bold
g) after a paragraph break, state the goals
h) after a paragraph break, indicate:

- app you will use (Import SRA File as Reads From Web)
- data that was downloaded (*S. aureus* MRSA177 sequence)
- other relevant information, as appropriate (see below, explain HMP)

In the future, you will need to enter the date (bolded) followed by information about your work (goals for that day, apps your used data that was downloaded, etc). No cells should be added to the narrative without stating the goals).

Click the KBase **save** icon often *!!!*

2. Genome Annotation

3.6. Check the QC on your genome sequence data (optional).

When you return to a KBase Narrative after it's closed, the results should still be displayed. You can collapse these windows by clicking on the **Collapse Cell** button located in the upper right hand corner of the cell.

If you want, you can rerun the QC analyses by clicking Reset and then Run. You will have an opportunity to choose different parameters. This will delete your old data unless you copy this cell to a new one!

3.7. Annotate your genome using RAST

Annotation is essentially an exercise in pattern recognition - luckily, a skill that computers are pretty good at! Given known examples of certain types of sequences, plus known rules about how protein-coding genes are structured, we can pick out many different components of a genome.

By comparing our sequences here to reference databases of sequences that have experimentally determined functions, we can also make guesses as to what the function of genes in our genome might be. But remember - these annotations are just hypotheses!

Use the "**Annotate Genome/Assembly with RASTtk - v1.073**" app. This uses the RAST pipeline to annotate the chosen assembled genome.

***** read the names of the apps carefully, there are multiple RAST apps***********

- Choose the correct assembly as the input object
- Provide an informative output name. This name will be in your data panel and should be given a name that will make sense at a later time, such as StaphRASTgenome.
- Click "**Run**" to run the app.

Once it is running, you will see the window change and a message saying "**Launching...**", shortly replaced by a message saying "**Running...**". Apps can take a few minutes (or more) to complete their task.

What kinds of data are you expecting to get as a result of this annotation? List as many as you can.

When completed, this will result in the output of a "genome" object. In KBase, a genome is defined as an object describing the genes and other genetic elements encoded within an organism, not just the raw sequence of the genome which came from our assembly.

Look at the summary of the RASTtk annotation, and ensure that the number of contigs matches what you expected. Next, record the number of features in your genome annotation table (Table 2). You will turn this in as part of your Project Part 3 for your capstone, but this is just practice for GP101.

3.8. Annotate your genome using Prokka

Use the **Annotate Assembly/Re-annotate Genomes with Prokka** app to do a second annotation. Prokka will ask you for a taxonomic assignment for your genome, but you can leave this blank.

Once this is complete, record the genes predicted, Number of protein coding genes, Number of genes with non-hypothetical function, and Number of genes with EC-number in your genome annotation table (Table 2).

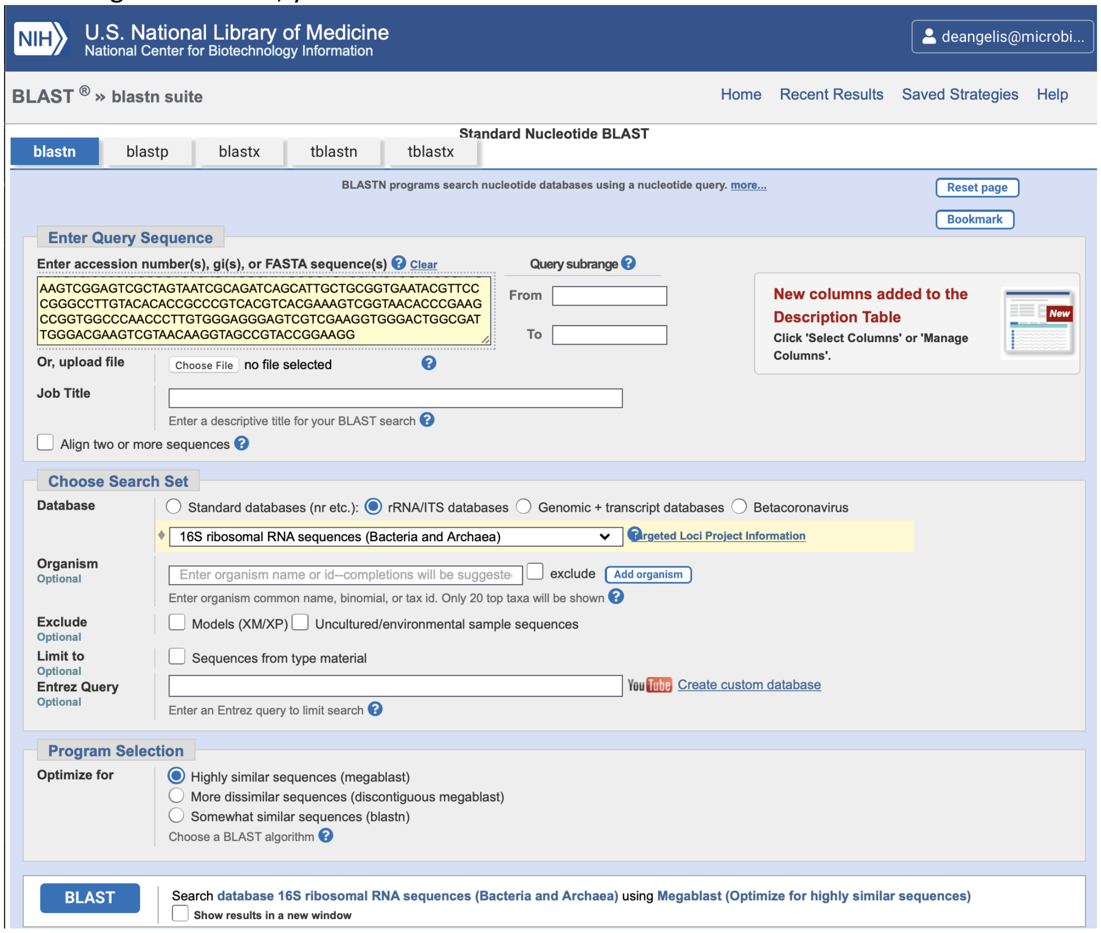
Click the KBase **save** icon often *!!!*

3. Classification: taxonomy and phylogeny

3.9. Initial classification may be done using the 16S ribosomal RNA gene sequence. Navigate to the NCBI blastn suite, to perform a Standard Nucleotide BLAST.

3.10. In the box labeled "Enter Query Sequence", enter the 16S ribosomal RNA gene sequence, which is included in the class genome database.

3.11. In the box labeled "Choose Search Set", choose the 16S ribosomal RNA sequences (Bacteria and Archaea) database. The rest of the optional items can be left blank. When you're ready to begin the search, your window should look like this:



BLASTN programs search nucleotide databases using a nucleotide query. more...

Enter Query Sequence

Enter accession number(s), gi(s), or FASTA sequence(s)

```
AAGTCGGAGTCGCTAGTAATCGCAGATCAGCATTGCTGCGGTGAATACGTTCC
CGGGCCTTGTACACACCGCCCGTCACGTCACGAAAGTCGGTAACACCCGAAG
CCGGTGGCCCAACCCTTGTGGGAGGGAGTCGTCGAAGGTGGGACTGGCGAT
TGGGACGAAGTCGTAACAAGGTAGCCGTACCGGAAGG
```

Query subrange: From / To

New columns added to the Description Table. Click 'Select Columns' or 'Manage Columns'.

Or, upload file: no file selected

Job Title: Enter a descriptive title for your BLAST search

Align two or more sequences

Choose Search Set

Database: Standard databases (nr etc.): [✓ rRNA/ITS databases] Genomic + transcript databases Betacoronavirus

16S ribosomal RNA sequences (Bacteria and Archaea) — Targeted Loci Project Information

Organism (Optional): Enter organism name or id--completions will be suggested; exclude. Enter organism common name, binomial, or tax id. Only 20 top taxa will be shown

Exclude (Optional): Models (XM/XP) Uncultured/environmental sample sequences

Limit to (Optional): Sequences from type material

Entrez Query (Optional): Enter an Entrez query to limit search

Program Selection

Optimize for: [✓ Highly similar sequences (megablast)] More dissimilar sequences (discontiguous megablast) Somewhat similar sequences (blastn). Choose a BLAST algorithm

Search database 16S ribosomal RNA sequences (Bacteria and Archaea) using Megablast (Optimize for highly similar sequences)

Show results in a new window

3.12. When you're ready, click the big blue "BLAST" button at the bottom of the page. Your request will be put into a queue, and may take a few minutes to load.

3.13. Once your query is completed, you will be redirected to a page showing sequences producing significant alignments, listed in descending order of percent sequence identity.

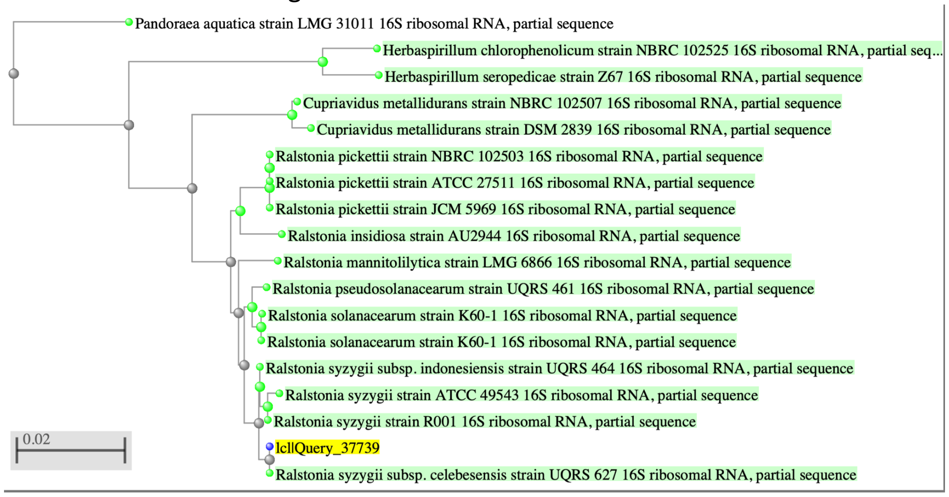
If your query turns up a match with 100% identity for the 16S ribosomal RNA genes, is it likely that these are the same organisms? Name two reasons why two genomes with identical 16S rRNA genes might be different.

3.14. Select about 12 closely related sequences, as well as about 5 additional less-closely related sequences that are in the top 100 sequences displayed. Then click the **<u>Distance tree of results</u>** button.

This should display a tree produced using BLAST pairwise alignments, with the default tree calculated using the Fast Minimum Evolution method.

Use the tools to create a view of the tree that clearly shows the nearest neighbors to the query sequence. Once you are satisfied, you can download (or screen capture) the 16S rRNA gene tree. Be sure to include the legend with the final version!

Your finished tree should look something like this:


Pandoraea aquatica strain LMG 31011 16S ribosomal RNA, partial sequence
Herbaspirillum chlorophenolicum strain NBRC 102525 16S ribosomal RNA, partial seq...
Herbaspirillum seropedicae strain Z67 16S ribosomal RNA, partial sequence
Cupriavidus metallidurans strain NBRC 102507 16S ribosomal RNA, partial sequence
Cupriavidus metallidurans strain DSM 2839 16S ribosomal RNA, partial sequence
Ralstonia pickettii strain NBRC 102503 16S ribosomal RNA, partial sequence
Ralstonia pickettii strain ATCC 27511 16S ribosomal RNA, partial sequence
Ralstonia pickettii strain JCM 5969 16S ribosomal RNA, partial sequence
Ralstonia insidiosa strain AU2944 16S ribosomal RNA, partial sequence
Ralstonia mannitolilytica strain LMG 6866 16S ribosomal RNA, partial sequence
Ralstonia pseudosolanacearum strain UQRS 461 16S ribosomal RNA, partial sequence
Ralstonia solanacearum strain K60-1 16S ribosomal RNA, partial sequence
Ralstonia solanacearum strain K60-1 16S ribosomal RNA, partial sequence
Ralstonia syzygii subsp. indonesiensis strain UQRS 464 16S ribosomal RNA, partial sequence
Ralstonia syzygii strain ATCC 49543 16S ribosomal RNA, partial sequence
Ralstonia syzygii strain R001 16S ribosomal RNA, partial sequence
lclQuery_37739
Ralstonia syzygii subsp. celebesensis strain UQRS 627 16S ribosomal RNA, partial sequence
0.02

3.15. For a different perspective on taxonomy and classification, we turn back to [KBase](). Navigate back to this page and find your genome.

We will now build species trees using two apps that rely on phylogenetic marker genes other than the 16S rRNA gene: GTDB-Tk and the Insert Genome into Species Tree App.

3.16. In KBase, locate the **GTDB-Tk classify app** and insert it into your KBase narrative.

For the input file, select the Prokka annotation, and then select **Run**.

When GTDB-Tk is run on the genome with default parameters, the genome is assigned a taxonomic classification using a set of domain-specific phylogenetic marker genes.

3.17. The GTDB-Tk app gives the domain, phylum, class, order, family, genus and (if possible) species level taxonomic classification. This taxonomic classification will be included in the final capstone report.

3.18. In KBase, locate the **Insert Genome into Species Tree app** and add it to your narrative.

To generate a phylogeny based on single copy housekeeping gene markers instead of the 16S ribosomal RNA gene markers, we will use the **Insert Genome into Species Tree app** in KBase.

When the Insert Genome into Species Tree app is called using default parameters, it generates a species tree called using a set of 49 core, universal genes defined by COG (Clusters of Orthologous Groups) gene families.

3.19. For the input file, select the Prokka annotation. This tool requires you to name the output tree and output genome set. Be sure to name these something memorable, e.g., GP101_tree and GP101_genomeset. Once these are defined, then select **Run**.

Have patience; this process can sometimes take a few minutes to run.

3.20. Save this second phylogenetic tree as an image file, and include it in your final report.

Why might these two different trees show different relationships for the same organism?

Click the KBase **save** icon often *!!!*

4. Explore genome characteristics

In this section, we will examine the annotated genome for characteristics of ribosomal RNA (rRNA) operons, central carbon metabolism, and an operon of your choice. Overall, you will assess the extent to which this genome contains elements found in many genomes.

3.21. Gene analysis using the Prokka annotation

**Note*: the sequencing data is not perfect (you have a draft, not complete, genome). Take this fact into consideration as you answer these questions.

After annotation you should have two annotated Genomes in your data. If this data is not already in a window, drag it into your narrative.

3.22. Browse the features in the Prokka annotation.

In the Prokka cell, click the **Browse Features** tab and note the Search Features window into which you can type a specific gene name. Click the arrows in the upper left hand corner of the cell to navigate the (probably thousands of) pages of annotated genes.

Note that this is a lot of information! You'll see that many genes are listed as 'hypothetical'. These portions of the genome have an ORF, but it does not have a significant enough of a match to a known gene to be assigned. WGS analysis reveals many hypothetical genes.

How many genes are annotated as 'hypothetical'? What percent of genes is this? Hint: use the "Search Features" box to search.

3.23. When you come across a gene that is interesting to you, click on the hypertext link in the Feature ID column on the left. Examine the Feature Context and Location windows for the orientation (+ or – strand).

Hover over nearby genes to see what they are. Click on a gene at the upstream or downstream end of the DNA in the window to move to the next section of that contig.

Name the gene you chose to explore, and list its orientation and location.

3.24. Next, browse the contigs in the Prokka annotation.

In the Prokka cell, click the Browse Contigs tab, and analyze the genome for characteristics of the ori.

Record the contig (node #) and location for the gene for the DnaA protein located, as well as the size of this contig. Look for other ori related genes nearby (upstream or downstream). Name them and their locations.

Do you notice any bias in the orientation of the ori related genes (examine both upstream and downstream of the DnaA gene)? Discuss the likelihood of the ori being located on this contig region. Support with specific data.

3.25. Analyze the genome for the rRNA operon(s).

Find the 16S ribosomal RNA gene sequence and, make note of the following.

- **On which contig (node #), and at what position is the 16S rRNA gene(s) located?**
- **What is the size of this contig? (see Browse Contigs tab.)**
- **Are the other rRNA genes (5S, 23S) linked on the same contig? Find them and their locations. Is there an rRNA operon with linked rRNA genes?**

Note: the sequencing data is not perfect (you have a draft, not complete, genome). Take this fact into consideration as you answer these questions.

3.26. Identify and describe a likely operon on any of the contigs not used for parts 5 and 6.

Use your knowledge of operons to envision how they would appear in the Feature Context window.

Name the genes in the operon. State the contig, location and orientation of the operon. Using internet resources, describe the role of the proteins produced from this operon. Include information about how each protein contributes to the overall role of the operon.

Click the KBase **save** icon often *!!!*

5. Define metabolic pathways encoded in the genome

Browsing through the genome annotation will give you a list of proteins contained in the genome, but it can be pretty laborious to sift through genes individually. Over the years, many people have worked to develop databases that put individual genes into the context of *pathways*. This helps to simplify the interpretation of a genome by abstracting that information to a higher order that is easier for mere mortals to comprehend.

3.27. Build a metabolic model.

Choose the **Build Metabolic Model** app from the Apps panel, and add it to your narrative. Use the Prokka annotation data as your input, and name the output something useful. Leave all other optional inputs either blank or as their defaults.

Have patience. This app might take 4-15 minutes to run.

3.28. Explore the predicted metabolic pathways, beginning with Glycolysis.

When you open up the resulting model object, you'll be able to examine all of the predicted biochemical reactions this organism is capable of, the chemical compounds produced, the pathways present, and more.

The output contains multiple tabs that allows you to navigate the reactions, compounds, genes, pathways and other aspects of the metabolic system of predicted by the genome.

If you click on a pathway name under the **Pathway** tab, it will bring up a map showing the reactions involved. In blue are highlighted enzymes predicted to be present in this genome which can carry out that reaction. More detailed information about these pathways can be found in the KEGG Pathways database.

Select the **Glycolysis / Gluconeogenesis** pathway, and refer to your outside resources (lecture notes, OpenStax textbook, internet resources) to answer the following questions. You may need to confer with other related pathways, like the **Pentose phosphate** pathway, to answer these questions.

Does this organism process glucose using Embden-Meyerhoff-Parnas glycolysis, Entner-Doudoroff glycoloysis, or both? What is the evidence for this?

3.29. Click on the link for the citrate cycle (TCA).

The enzymes that are highlighted with blue are those that the app found in the genome. Compare the KBase generated pathway with the one provide in your textbook or slides.

Starting with citrate, trace the pathway (note that some of the intermediates are described with alternate names, and thus will differ from the figure(ex) 2-oxo-glutarate = α-Ketoglutarate). Use internet resources if you need more information.

Are there any enzymes missing from the TCA cycle? If so indicate the KEGG number of the enzyme(s).

Other questions that you can ask of your genome with the metabolic model include:
- (Overview tab) How many total enzymatic reactions are predicted?
- (Reactions tab) search with the term ATPase then identify the F1 subunit reaction row. How many genes (CDS) are associated with this function?
- Copy the name of <u>one</u> of these genes (right click, copy) and search (browse features) for it in your RAST annotated genome.
- Click on the feature ID link and examine the genes that co-locate with the one you searched. What gene organization do you observe?

3.30. Scroll through and find the **Oxidative phosphorylation** pathway.

This can be found in the **Pathways** tab as part of the Output from Build Metabolic Model. Once you open this tab, you will notice that the pathway is denoted in a graphical representation. Here, the enzymes are NOT color coded for their presence or absence.

To determine whether your organism has an electron transport chain, you must examine the number of reactions and compounds in the model.

You can also search for the relevant genes in the Prokka annotation, which is listed in one of the earlier outputs.

Does this organism have an electron transport chain, or any capability for respiration? What is your evidence for this?

Click the KBase **save** icon often *!!!*

6. Compare our novel genome to known relatives

In this next section, you will find nearest neighbor genomes, then compare your genome to these close relatives using pangenome analysis and by calculating average nucleotide identity (ANI).

3.31. Review your genomes that were included in your SpeciesTree, and use this to decide how many genomes should be included in an analysis of close relatives.

For GP101, there are 9 genomes (including GP101) that are in the Ralstonia/Cupriavidus clade. For your capstone project genomes, this number may vary, but aim for a number between about 6 and 12.

3.32. In KBase, locate the **Insert Genome into Species Tree app** and add it to your narrative.

This will be the second time this app is used in your narrative. It's ok to duplicate the apps!

This time, instead of accepting the default parameters for **Neighbor Public Genome Count**, enter the number you came up with in the prior step. In the case for GP101, we will use 9.

3.33. For the input file, select the Prokka annotation. This tool requires you to name the output tree and output genome set. Be sure to name these something memorable, e.g., GP101_tree_9 and GP101_genomeset_9. Once these are defined, then hit **Run**.

Have patience; this process can sometimes take a few minutes to run.

3.34. This app builds a tree and a genome set that you will use for the pangenome and the ANI analyses that follow. Check in your **DATA** panel to see these objects; you may need to scroll down to see them.

3.35. To compare our GP101 genome to its nearest neighbors, we'll use KBase's pangenome analysis tools, part of the Comparative Genomics collection.

These tools will match the genes that are found in common between a set of genes, and allow us to see what genes are common vs. those found in only one of the two genomes. Note that this approach can scale to many more genomes - but for now we'll just compare these two.

A **pangenome** represents all genes found within a collection of related organisms, grouped by how similar sequences are to one another, also referred to as sequence homology. One primary purpose of creating a pangenome is to distinguish which genes are **orthologs** - vertically inherited genes - and which genes arose from duplication events (**paralogs**). Paralogs tend to evolve, and thus change after the duplication. This enables the cell to potentially gain a new gene (and function).

To learn more about how organisms are related or even how specific genes came about, we can use pangenomes to examine similarities and differences across a collection of genomes. Part of the theory behind pangenomes is the existence of a core genome, which is consistent across all strains or species, and then the flexible or non-core genome where variation exists. Genes can be categorized into various general functions (e.g., virulence), and families of genes are further categorized under these functions.

3.36. Use the **Build Pangenome with OrthoMCL** app to construct the Pangenome object.

This clusters all genes into groups of related gene sequences. Select the genome set you created in the previous section, set the output object name to something descriptive, and hit run.

3.37. Use the **Compare Genomes from Pangenome** app to generate lists of differentially present genes in the two genomes.

The pangenome object should be the pangenome object you created previously, and name the output object something that is informative and makes sense to you. The output will appear in a new cell.

3.38. Go to the output genome comparison object, click on it, and look around!

This app generates a huge data set. It can be helpful to set the cell to show 100 entries at a time. The output from this App is a table with four tabs: (i) Overview, (ii) Genomes, (iii) Functions, and (ix) Families.

- The Overview tab indicates the genome comparison object name, the genome comparison workspace name, the number of total identified core functions, the number of identified core protein families, the KBase-specific protein comparison object ID, the object owner/author, and a timestamp for the creation of the object.
- The Genome Comparison tab indicates a count of the number of protein families and number of protein function groups found within the queried genomes.
- The Families tab indicates the distribution of each gene family across each of the query genomes.
- The Functions tab indicates the functions identified across the query genomes; specifically, the function name and its associated subsystem, primary and secondary class. In addition, the Functions tab indicates the distribution for each function across each of the query genomes, including the total families and genes identified.

3.39. Look under the '**Functions**' tab. Sort the **Primary class** column so the beginning of the alphabet is at the top of the table.

Examine the class associated with carbohydrate degradation. Click on a gene that is present in only 1 Genome; this will open up another view that allows you to confirm whether that gene is present only in your target strain.

Record in your notebook:

1. Name of the gene
2. Name of the bacterium(a) in which it is present
3. Does this information support the fact that *your target strain* is enriched in carbohydrate degradation enzymes compared to nearest relatives?

3.40. Now go back into the pangenome and find another gene associated with carbohydrate degradation.

Record in your notebook:

1. Name of the gene
2. Name of the bacterium(a) in which it is present
3. Does this information support the fact that *your target strain* is enriched in carbohydrate degradation enzymes compared to nearest relatives?

Remember that this KBase analysis is only a prediction of the bacterium's abilities. The output depends on how the program is coded.

3.41. A common and fast metric for comparing genomes is known as Average Nucleotide Identity (ANI).

Remember the old adage about not comparing apples to oranges? If we want to look for fine genetic differences between two genomes, it will only be useful if the two genomes are pretty closely related to each other. Otherwise, the differences between genomes could be related to many other biological features that have nothing to do with Methicillin resistance.

Essentially, this is a calculation that aligns two genomes to one another and simply asks for each region that is shared between the genomes, on average, what % of the genomes are identical at the nucleotide level. This metric is now often used to assess whether two genomes belong to the same "species", with a rough cutoff of >95% ANI indicating species level similarity.

The ANI analysis also assesses the synteny between the bacteria. Synteny is the extent of conservation at the level of the physical location of the genes on the chromosome (https://www.nature.com/scitable/topicpage/synteny-inferring-ancestral-genomes-44022/).

3.42. First, find the nearest neighbor genome based on your phylogenetic analysis.

You'll want to choose the genome that is the most closely related to your unknown genome. This will give you an indication of how novel your genome is.

Based on your analysis so far, what is the most closely related organism?

3.43. Next, import the genome data for this nearest neighbor.

In the **DATA** box (upper lefthand corner of the screen), there is a blue circle with a white plus sign in it. Press this button, and a window pops out with a menu along the top. Select the "Public" tab, then enter the name of the nearest neighbor organism.

If there are many options, it may be best to enter the GCF number. The "GCF" denotes that this is an NCBI genome with a RefSeq assembly (as opposed to a GenBank assembly). From the RefSeq Frequently Asked Questions (FAQ): **What is the difference between RefSeq and GenBank?**

> "GenBank archival sequence database includes publicly available DNA sequences submitted from individual laboratories and large-scale sequencing projects. GenBank is part of the International Nucleotide Sequence Database Collaboration (INSDC) along with the European Nucleotide Archive and the DNA Data Bank of Japan (DDBJ).
>
> "RefSeq records are created from submissions to the International Nucleotide Sequence Database Collaboration (INSDC) to provide non-redundant curated data representing our current knowledge of known genes. RefSeq records are owned by NCBI and therefore can be updated as needed to maintain current annotation or to incorporate additional information."

Try searching "Ralstonia pickettii", "12J", or "GCF_000020205.1".

3.44. Once you find the correct genome, click the "**< Add**" button, and the genome will show up in your **DATA** box.

This button becomes available to you when your mouse hovers over the area to the left of this item. I know, this is a terrible feature. But here we are.

You can press the plus button to collapse this window again.

3.45. Use the **Compute ANI with FastANI** app to compare the two genomes (see procedure above for importing the second genome). Make sure both are set as Input Objects (click on + to add each genome), then press **Run**.

Questions to answer in your notebook:

1. How similar are the two genomes (state percentage) at the ANI level? Would you consider them to be the same species?

2. Observe one of the pdf synteny outputs and create a link to it in your notebook. To what extent do the 2 genomes line up along the chromosome? Is this what you expected, assuming the 2 bacteria are evolutionarily related?
3. Suggest a mechanism that might contribute to the gene arrangements (synteny) along the 2 chromosomes.
4. How might the results differ if you compared to bacteria at the level amino acid identity instead of nucleotide identity?

Click the KBase **save** icon often *!!!*

7. Explore carbohydrate degradation

Carbohydrates are an important source of carbon and energy for organoheterotrophs. Oligotrophs survive in environments with relatively low nutrient concentration, often by digesting recalcitrant carbon sources.

Carbohydrate Active Enzymes (CAZy) have been categorized into various families, some of which degrade carbohydrates and some which degrade lignin (http://www.cazy.org/). The collection of CAZys within a genome provides some insight into the carbohydrate utilizing ability of that bacterium.

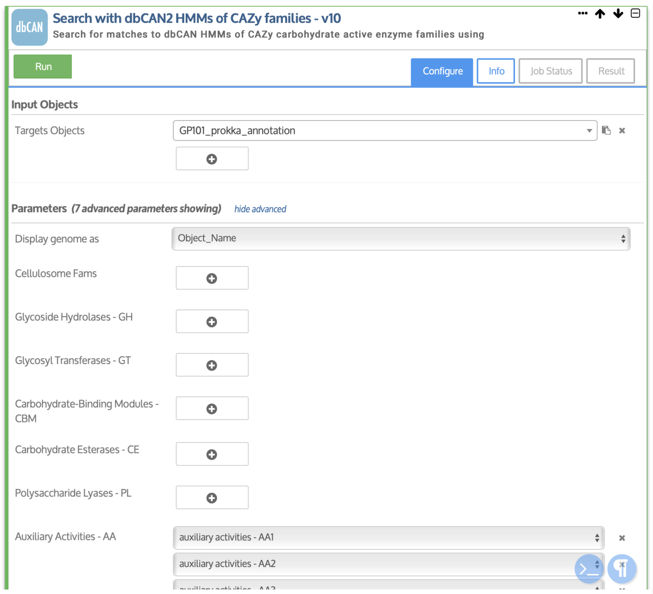
The AA class has 9 families of ligninolytic enzymes and 7 families of lytic polysaccharide mono-oxygenases. Lignin breakdown enzymes include laccases, oxidases and peroxidases. These enzymes do not act on carbohydrates, but because lignin is invariably and intimately associated with carbohydrates in the plant cell wall, they do cooperate with polysaccharide depolymerases.

Bacteria in the human gut produce a great variety of CAZys that humans do not produce, thus increasing the availability of nutrients to the human host(*doi: 10.1093/nar/gky418*). Soil bacteria, likewise, can supply nutrients to plants while plants employ GTs (glycosyltransferases) to create complex, often recalcitrant, carbohydrates.

The goal of this section is to determine the CAZy profile of our organism of interest.

3.46. Search the genome for genes annotated in the CAZy database.

Run the Search with **dbCAN HMMs of CAZy families - v10** app with the Prokka annotation as the Target Object. Select “ALL” for **Auxiliary Activities (AA) family classification**, and otherwise keep parameters as the default settings. Be sure to enter a name for the output feature set. Below is a screen shot of what this will look like when you’re ready to hit **Run**.



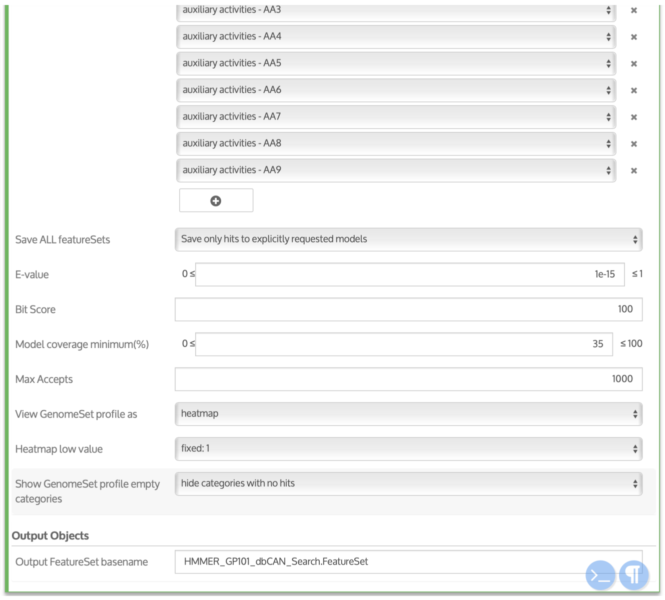
Search with dbCAN2 HMMs of CAZy families - v10
Search for matches to dbCAN HMMs of CAZy carbohydrate active enzyme families using

Input Objects

Targets Objects	GP101_prokka_annotation

Parameters *(7 advanced parameters showing)* *hide advanced*

Display genome as	Object_Name
Cellulosome Fams	
Glycoside Hydrolases - GH	
Glycosyl Transferases - GT	
Carbohydrate-Binding Modules - CBM	
Carbohydrate Esterases - CE	
Polysaccharide Lyases - PL	
Auxiliary Activities - AA	auxiliary activities - AA1
	auxiliary activities - AA2



	auxiliary activities - AA3
	auxiliary activities - AA4
	auxiliary activities - AA5
	auxiliary activities - AA6
	auxiliary activities - AA7
	auxiliary activities - AA8
	auxiliary activities - AA9
Save ALL featureSets	Save only hits to explicitly requested models
E-value	0 ≤ 1e-15 ≤ 1
Bit Score	100
Model coverage minimum(%)	0 ≤ 35 ≤ 100
Max Accepts	1000
View GenomeSet profile as	heatmap
Heatmap low value	fixed: 1
Show GenomeSet profile empty categories	hide categories with no hits

Output Objects

Output FeatureSet basename	HMMER_GP101_dbCAN_Search.FeatureSet

Once this app is done running, it will generate a report and a feature set for the AAs detected.

3.47. Click on each AA FeatureSet in the DATA window. Enter the number of genes found for each AA family in the table below.

3.48. We would like to determine whether this genome is enriched for carbohydrate degradation genes. To do this, we will need to select genomes that are close matches, and compare them.

We found near neighbor genomes in the last section. Run dbCAN for the nearest match genome you selected for ANI, and enter those values into the table below.

genome	AA1	AA2	AA3	AA4	AA5	AA6	AA7	AA8	AA9
GP101									

3.49. For your capstone project genomes only, record the number of AA enzymes for each of the categories in the Class genome sequencing workbook. If there are categories with no values, enter a zero.

Now compare the results of the number of CAZymes in each family for your unknown genome from the nearest relative. Which one has more? Or do they seem to be about the same?

Click the KBase **save** icon often *!!!*

The course activities are set up in four parts. In **Part 1**, we gained access to our computing resources, practiced coding in the shell, worked with sequence data, and submitted jobs to the MGHPCC cloud computer.

In **Part 2**, we worked with the raw, basecalled sequence data from the bacterial isolate Ralstonia sp. GP101. We will worked together to subset the data, assemble the genome, and perform some additional quality control assessments.

In **Part 3**, we continued working with the GP101 genome by annotating the assembly, exploring the coding and non-coding regions of the genome. We also performed phylogenetic, sequence and functional gene comparative analysis with other closely related strains.

In **Part 4** (*this part*), you will be assigned your own genome that is the basis of your capstone project. These genomes are derived from bacteria we have isolated in our lab. They have been sequenced on the MinION, basecalled and QC checked to ensure that they are ready for you. Their genotype and some phenotypic information is available. You will repeat the analyses we did on GP101 for assembly (**part 2**) and annotation (**part 3**) on your genome, and write it up for your capstone project report.

The capstone project for this course is a report written in the style of a Microbiology Resource Announcement. The specific guidelines for writing, organizing and grading this report are described below.

Grading

The grade breakdown for the assignments related to the capstone report is as follows. The phylogeny and taxonomy (part 1), hypothesis and introduction (part 2), and Tables 1 and 2 (parts 3 & 4) are meant to be included as part of the final research report.

- Project Part 1: Table 1 (assembly) (10%)
- Project Part 2: Phylogeny and taxonomy (10%)
- Project Part 3: Table 2 (annotation) (10%)
- Project Part 4: Hypothesis & Intro (10%)
- Final research report (15%)

Due dates. Due dates are described in the syllabus. These parts are due ahead of time to ensure that the final report is done in parts throughout the semester, with ample time for feedback from instructor and classmates. Students are expected to incorporate any feedback from the project part assignments into their final research reports, or risk being penalized for the same deficiencies more than once.

Regrades. All four project parts can be revised and resubmitted one time for a regrade. Revisions will be accepted within one week of receiving comments and a grade. Assignments will be graded anew, according to the existing rubric. Students should prioritize incorporation of comments on the corrected first version, though instructor comments should be considered a minimum for revising. An assignment may only be resubmitted once.

Draft comments. Because the final report is due during the finals period at the end of the semester, a regrade of this report will not be possible. Students are encouraged to share drafts of their capstone reports with the instructor ahead of the final due date for early feedback. Comments on work in progress will be provided within 2 days of request.

Plagiarism. Plagiarism will not be tolerated, and students are expected to write their lab reports in their own words. TurnItIn is set to "Generate reports immediately, reports can be overwritten until the due date," and will check against the literature, other student reports stored, and the internet. Reports that are plagiarized or include substantial direct quotes will be returned for resubmission without grading.

Instructions

Final research reports should be prepared in the style of the American Society for Microbiology (ASM) journal Microbiology Resource Announcements (MRA). This handout describes the guidelines for your paper, and where necessary will be supplemented by the published online instructions for this style of manuscript.

Organization. Capstone reports will be written as a single report without subheaders. Inclusion of a brief abstract is optional. Though there are no subheaders, the essay will include introduction, methods, results and discussion sections.

Word count. MRA papers have a word count limit of 500 words, but your student papers will have a word count limit of 1000 words. There will be a penalty of -2 points (out of a total possible 20 points) for every 100 words over the limit. Word count will include the main text only, and will not include the tables, citations, or figure legends.

Displays. A display is defined as a table or figure. Final research reports should include at least two displays, Table 1 (assembly) and Table 2 (annotation). These have separate due dates ahead of the due date for the final paper. A blank template will be provided for these tables. It is permitted to add rows, but rows may not be deleted.

Citations. Capstone reports should include a section for citations, also called references. These can be included using any style, though it is suggested that students use a reference manager for their manuscripts. There are many options for free reference manager software, including Zotero, Mendeley, Papers, ReadCube Papers, Paperpile, and many more.

Style. The manuscript should be written in the style of an academic research paper. Introduction and discussion sections are generally written in the present tense, while methods and results are generally written in the past tense. Avoid using the passive voice.

There are also two assigned papers that describe how to write an effective genome announcement paper. These will be further discussed in class.

- Baltrus, David A., Christina A. Cuomo, John J. Dennehy, Julie C. Dunning Hotopp, Julia A. Maresca, Irene LG Newton, David A. Rasko, Antonis Rokas, Simon Roux, and Jason E. Stajich. "Future-proofing your Microbiology Resource Announcements genome assembly for reproducibility and clarity." (2019): e00954-19.

- Dunning Hotopp, Julie C., David A. Baltrus, Vincent M. Bruno, John J. Dennehy, Steven R. Gill, Julia A. Maresca, Jelle Matthijnssens et al. "Best Practices for Successfully Writing and Publishing a Genome Announcement in Microbiology Resource Announcements." (2020): e00763-20.

Rubric

The following rubric will be used to grade the parts of the project (where appropriate) as well as the final research report. Additional line items may be added to the rubric if necessary, but then these additional guidelines will be applied to all papers.

1. Scientific content (score 1-6)
   - Full marks for a paper that
     - has a title that summarizes clearly the main point of the entire story,
     - has a well-defined and well-justified thesis or hypothesis, and
     - contains all of the necessary **technical elements** (*see checklist below*).
   - Points cannot be awarded for a poorly defined, vague or inappropriate topic.

2. Incorporation of scientific, peer-reviewed sources (score 1-3)
   - Full marks for a paper that
     - includes citations for software used as well as for scientific context in the introduction and conclusion sections of the paper,
     - cites research papers from our class and from the student's independent research,
     - with each citation being a peer-reviewed original research paper pertinent to the topic.
   - Points cannot be awarded for a paper with no cited sources, or sources cited with direct quotes instead of communicated in the author's own words.

3. Narrative structure (score 1-3)
   - Full marks for a paper that
     - is well-structured with a clear thesis in the first paragraph, supporting ideas in the middle paragraphs, and concluding paragraph that returns to the main idea with synthesizing statements,
     - Includes all scientific sections (Introduction, methods, results and discussion),
     - describes science based on the structure of our work (assembly, annotation, hypothesis testing),
   - Points cannot be awarded for a poorly structured paper with weak or no topic sentences, non-linear structure.

4. Accessible language and Grammar (score 1-3)
   - Full marks for a paper that
     - has language that is clear accessible to peer reviewers, with correctly used and defined technical terms as part of the narrative,
     - is grammatically sound (avoid passive voice, be consistent with verb tense, use complete sentences, among others),
     - has strong topic sentences in each paragraph, featuring concise sentences, no stray or extraneous ideas, and few misspelled words or other typos, and
     - is 1000 words and not much more or less.

- Points cannot be awarded for a paper that does not use technical terms, or does not define technical terms, or is written in a way that is confusing or unclear, or for a grammatically poor paper with run-on sentences, incomplete sentences, misspelled words and many typos.

Technical elements: for full credit, manuscripts must also include each of the following:

- A rationale or significance for the sequencing, using Project Part 4 as a template.
- The following information should be found in the class MinION sequence assembly log, found in our shared class google doc. Questions about this content can be directed to the instructor.
  - The provenance for the organisms sequenced.
  - A description of how the organism was isolated and growth conditions for cultivation.
  - Detailed methods for DNA isolation, library preparation, and sequencing (including the technology and chemistry used).
- A description of how the reads were quality controlled, and a clear statement describing whether the sequence is high quality and why.
- Details on how the genome was assembled, including a table with pertinent statistics describing the quality of the sequence and assembly using Project Part 1, Table 1 as a template.
- Taxonomic identification from domain down to genus for bacterial isolates. This element and the next are done using Project Part 2 as a template.
- Two phylogenetic trees (based on 2 methods) which show nearest relatives, cultured and uncultured, including an outgroup and legend, and a description of how they are similar and why they are different.
- Describe in detail how the genome was annotated, using Project Part 3, Table 2 template.
- Describe the organism's metabolism, including how it generates energy and biomass.
- Test of the instructor's hypothesis, that your organism has more genes annotated as carbohydrate-associated enzyme counts per genome compared to the nearest neighbor according to the CAZy database. Use Project Part 3, Table 3 as a template. Describe how you found the nearest neighbor, and report its average nucleotide identity (ANI).
- A new statement of hypothesis regarding the organism, which is tested using the genome sequence. This should be written using Project Part 4 as a template.
- A statement on whether the stated hypothesis is supported, and a related statement based on this conclusion describing the next steps or research direction.
- A citation and version number for every piece of software used. In most places, the lab manual can be cited if you wish.