Bioinformatics Toolbox at WPHL

Robert A. Petit III, PhD
2025 AMD Mountain Regional Binf Conference
April 22nd, 2025

Wyoming Department of Health | PUBLIC HEALTH DIVISION | WYOMING PUBLIC HEALTH LABORATORY

Overview

- Bioinformatics at WPHL

- Publicly Available Tools From WPHL
  - camlhmp (*“camel hump”*)
  - Bactopia

- In-House Bioinformatics Trainings

WPHL Binf Overview

Bioinformatics at WPHL

- Bioinformatics is conducted on the Azure cloud using the command-line

- “Bioinformatics” Adjacent Lab Members
  - *Regularly using biological data is some form (e.g. sequencing, dPCR, etc…)*
  - Bioinformaticians: 1 (*c'est moi*)
  - Molecular: 3
  - Wastewater: 5
  - LIMS: 1
  - People doing bioinformatics: 10

Bioinformatics at WPHL, continued

- We have a lot of people who are not *"bioinformaticians"* that are doing bioinformatics.

- It's important to develop user-friendly bioinformatics tools/pipelines and protocols
    - Available from conda or containers
    - Easy to install and use
    - Well documented
    - Standardized analyses into a pipeline

camlhmp (*“camel hump”*)

camlhmp - classification through YAML

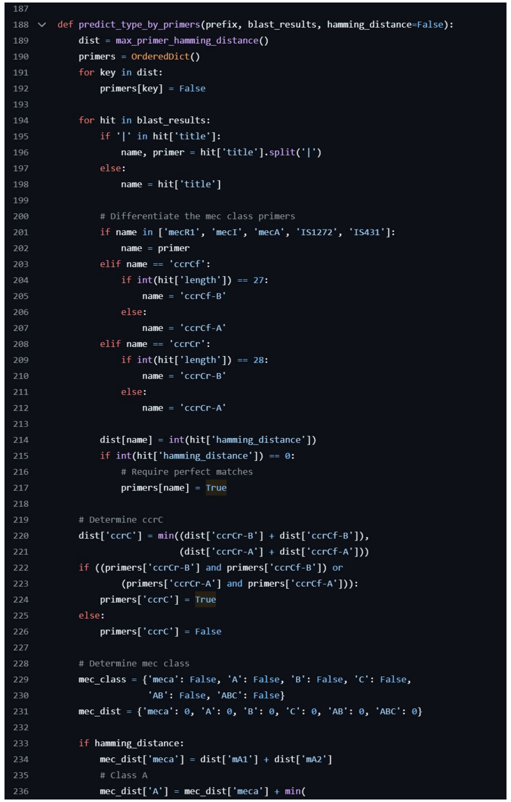
- A bioinformatics framework which uses YAML to separate "defining" sequence-based typing (SBT) logic from the "programming"

- camlhmp framework
  - Defines SBT logic in YAML, not code
  - Provides pre-built tooling to run the typing

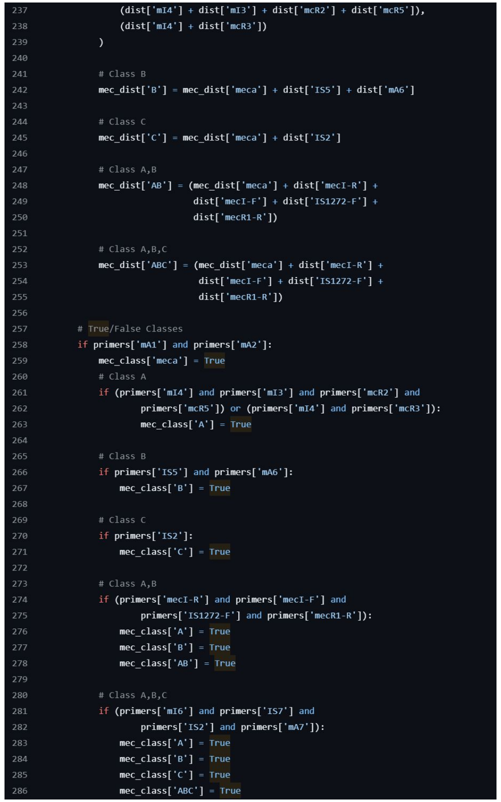
- Why YAML not *TOML*, *JSON*, *etc..?*
  - Prioritized "easy-to-read" and "simplicity"

- Documentation: https://rpetit3.github.io/camlhmp/latest/

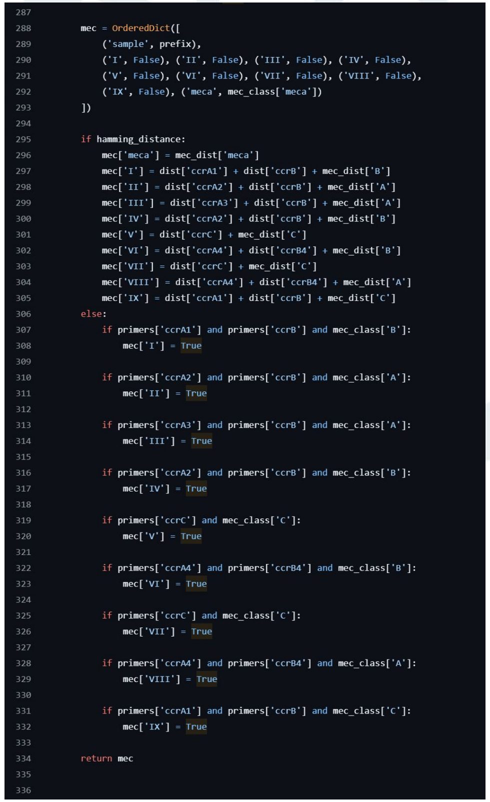
Example of an existing SBT (staphopia-sccmec)



```python
def predict_type_by_primers(prefix, blast_results, hamming_distance=False):
    dist = max_primer_hamming_distance()
    primers = OrderedDict()
    for key in dist:
        primers[key] = False

    for hit in blast_results:
        if '|' in hit['title']:
            name, primer = hit['title'].split('|')
        else:
            name = hit['title']

        # Differentiate the mec class primers
        if name in ['mecR1', 'mecI', 'mecA', 'IS1272', 'IS431']:
            name = primer
        elif name == 'ccrCf':
            if int(hit['length']) == 27:
                name = 'ccrCf-B'
            else:
                name = 'ccrCf-A'
        elif name == 'ccrCr':
            if int(hit['length']) == 28:
                name = 'ccrCr-B'
            else:
                name = 'ccrCr-A'

        dist[name] = int(hit['hamming_distance'])
        if int(hit['hamming_distance']) == 0:
            # Require perfect matches
            primers[name] = True

    # Determine ccrC
    dist['ccrC'] = min((dist['ccrCr-B'] + dist['ccrCf-B']),
                       (dist['ccrCr-A'] + dist['ccrCf-A']))
    if ((primers['ccrCr-B'] and primers['ccrCf-B']) or
            (primers['ccrCr-A'] and primers['ccrCf-A'])):
        primers['ccrC'] = True
    else:
        primers['ccrC'] = False

    # Determine mec class
    mec_class = {'meca': False, 'A': False, 'B': False, 'C': False,
                 'AB': False, 'ABC': False}
    mec_dist = {'meca': 0, 'A': 0, 'B': 0, 'C': 0, 'AB': 0, 'ABC': 0}

    if hamming_distance:
        mec_dist['meca'] = dist['mA1'] + dist['mA2']
        # Class A
        mec_dist['A'] = mec_dist['meca'] + min(
```



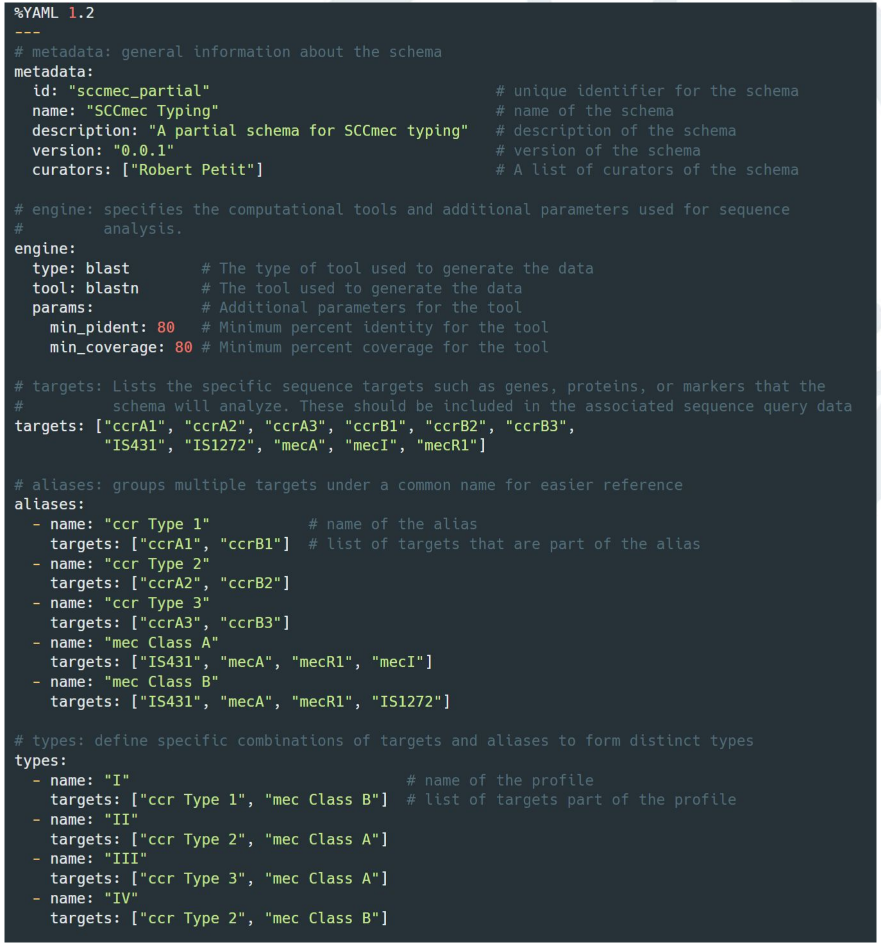
```python
            (dist['mI4'] + dist['mI3'] + dist['mcR2'] + dist['mcR5']),
            (dist['mI4'] + dist['mcR3'])
        )

        # Class B
        mec_dist['B'] = mec_dist['meca'] + dist['IS5'] + dist['mA6']

        # Class C
        mec_dist['C'] = mec_dist['meca'] + dist['IS2']

        # Class A,B
        mec_dist['AB'] = (mec_dist['meca'] + dist['mecI-R'] +
                          dist['mecI-F'] + dist['IS1272-F'] +
                          dist['mecR1-R'])

        # Class A,B,C
        mec_dist['ABC'] = (mec_dist['meca'] + dist['mecI-R'] +
                           dist['mecI-F'] + dist['IS1272-F'] +
                           dist['mecR1-R'])

    # True/False Classes
    if primers['mA1'] and primers['mA2']:
        mec_class['meca'] = True
        # Class A
        if (primers['mI4'] and primers['mI3'] and primers['mcR2'] and
                primers['mcR5']) or (primers['mI4'] and primers['mcR3']):
                mec_class['A'] = True

        # Class B
        if primers['IS5'] and primers['mA6']:
            mec_class['B'] = True

        # Class C
        if primers['IS2']:
            mec_class['C'] = True

        # Class A,B
        if (primers['mecI-R'] and primers['mecI-F'] and
                primers['IS1272-F'] and primers['mecR1-R']):
            mec_class['A'] = True
            mec_class['B'] = True
            mec_class['AB'] = True

        # Class A,B,C
        if (primers['mI6'] and primers['IS7'] and
                primers['IS2'] and primers['mA7']):
            mec_class['A'] = True
            mec_class['B'] = True
            mec_class['C'] = True
            mec_class['ABC'] = True
```



```python

    mec = OrderedDict([
        ('sample', prefix),
        ('I', False), ('II', False), ('III', False), ('IV', False),
        ('V', False), ('VI', False), ('VII', False), ('VIII', False),
        ('IX', False), ('meca', mec_class['meca'])
    ])

    if hamming_distance:
        mec['meca'] = mec_dist['meca']
        mec['I'] = dist['ccrA1'] + dist['ccrB'] + mec_dist['B']
        mec['II'] = dist['ccrA2'] + dist['ccrB'] + mec_dist['A']
        mec['III'] = dist['ccrA3'] + dist['ccrB'] + mec_dist['A']
        mec['IV'] = dist['ccrA2'] + dist['ccrB'] + mec_dist['B']
        mec['V'] = dist['ccrC'] + mec_dist['C']
        mec['VI'] = dist['ccrA4'] + dist['ccrB4'] + mec_dist['B']
        mec['VII'] = dist['ccrC'] + mec_dist['C']
        mec['VIII'] = dist['ccrA4'] + dist['ccrB4'] + mec_dist['A']
        mec['IX'] = dist['ccrA1'] + dist['ccrB'] + mec_dist['C']
    else:
        if primers['ccrA1'] and primers['ccrB'] and mec_class['B']:
            mec['I'] = True

        if primers['ccrA2'] and primers['ccrB'] and mec_class['A']:
            mec['II'] = True

        if primers['ccrA3'] and primers['ccrB'] and mec_class['A']:
            mec['III'] = True

        if primers['ccrA2'] and primers['ccrB'] and mec_class['B']:
            mec['IV'] = True

        if primers['ccrC'] and mec_class['C']:
            mec['V'] = True

        if primers['ccrA4'] and primers['ccrB4'] and mec_class['B']:
            mec['VI'] = True

        if primers['ccrC'] and mec_class['C']:
            mec['VII'] = True

        if primers['ccrA4'] and primers['ccrB4'] and mec_class['A']:
            mec['VIII'] = True

        if primers['ccrA1'] and primers['ccrB'] and mec_class['C']:
            mec['IX'] = True

    return mec


```

Comparable SBT in camlhmp

- Only two files needed:
  - Schema defined in YAML
  - Reference sequences in FASTA
- Notable features
  - Pre-built CLI tools for simple SBTs
  - API for customization for complex SBTs
- Pure Python package
  - Available from PyPi and Bioconda



```yaml
%YAML 1.2
---
# metadata: general information about the schema
metadata:
  id: "sccmec_partial"                                 # unique identifier for the schema
  name: "SCCmec Typing"                                # name of the schema
  description: "A partial schema for SCCmec typing"    # description of the schema
  version: "0.0.1"                                     # version of the schema
  curators: ["Robert Petit"]                           # A list of curators of the schema

# engine: specifies the computational tools and additional parameters used for sequence
#         analysis.
engine:
  type: blast        # The type of tool used to generate the data
  tool: blastn       # The tool used to generate the data
  params:            # Additional parameters for the tool
    min_pident: 80   # Minimum percent identity for the tool
    min_coverage: 80 # Minimum percent coverage for the tool

# targets: Lists the specific sequence targets such as genes, proteins, or markers that the
#          schema will analyze. These should be included in the associated sequence query data
targets: ["ccrA1", "ccrA2", "ccrA3", "ccrB1", "ccrB2", "ccrB3",
          "IS431", "IS1272", "mecA", "mecI", "mecR1"]

# aliases: groups multiple targets under a common name for easier reference
aliases:
  - name: "ccr Type 1"             # name of the alias
    targets: ["ccrA1", "ccrB1"]  # list of targets that are part of the alias
  - name: "ccr Type 2"
    targets: ["ccrA2", "ccrB2"]
  - name: "ccr Type 3"
    targets: ["ccrA3", "ccrB3"]
  - name: "mec Class A"
    targets: ["IS431", "mecA", "mecR1", "mecI"]
  - name: "mec Class B"
    targets: ["IS431", "mecA", "mecR1", "IS1272"]

# types: define specific combinations of targets and aliases to form distinct types
types:
  - name: "I"                                     # name of the profile
    targets: ["ccr Type 1", "mec Class B"]  # list of targets part of the profile
  - name: "II"
    targets: ["ccr Type 2", "mec Class A"]
  - name: "III"
    targets: ["ccr Type 3", "mec Class A"]
  - name: "IV"
    targets: ["ccr Type 2", "mec Class B"]
```

SBTs using camlhmp

- pasty - in silico serogrouping of *Pseudomonas aeruginosa* isolates
- pbptyper - In silico Penicillin Binding Protein (PBP) typer for *Streptococcus pneumoniae* assemblies
- sccmec - A tool for typing SCCmec cassettes in *Staphylococcus aureus* assemblies
- tulatyper - Subtyping of *Francisella tularensis* subtypes

Bactopia

Bactopia for accessible bioinformatics analysis

- ***End-to-end pipeline for bacterial genome analysis***
    - Written in Nextflow, following nf-core standards
    - Available from Conda, Docker, or Singularity
    - Supports numerous compute infrastructures

- ***Wraps 150+ bioinformatic tools into stand-alone modules***

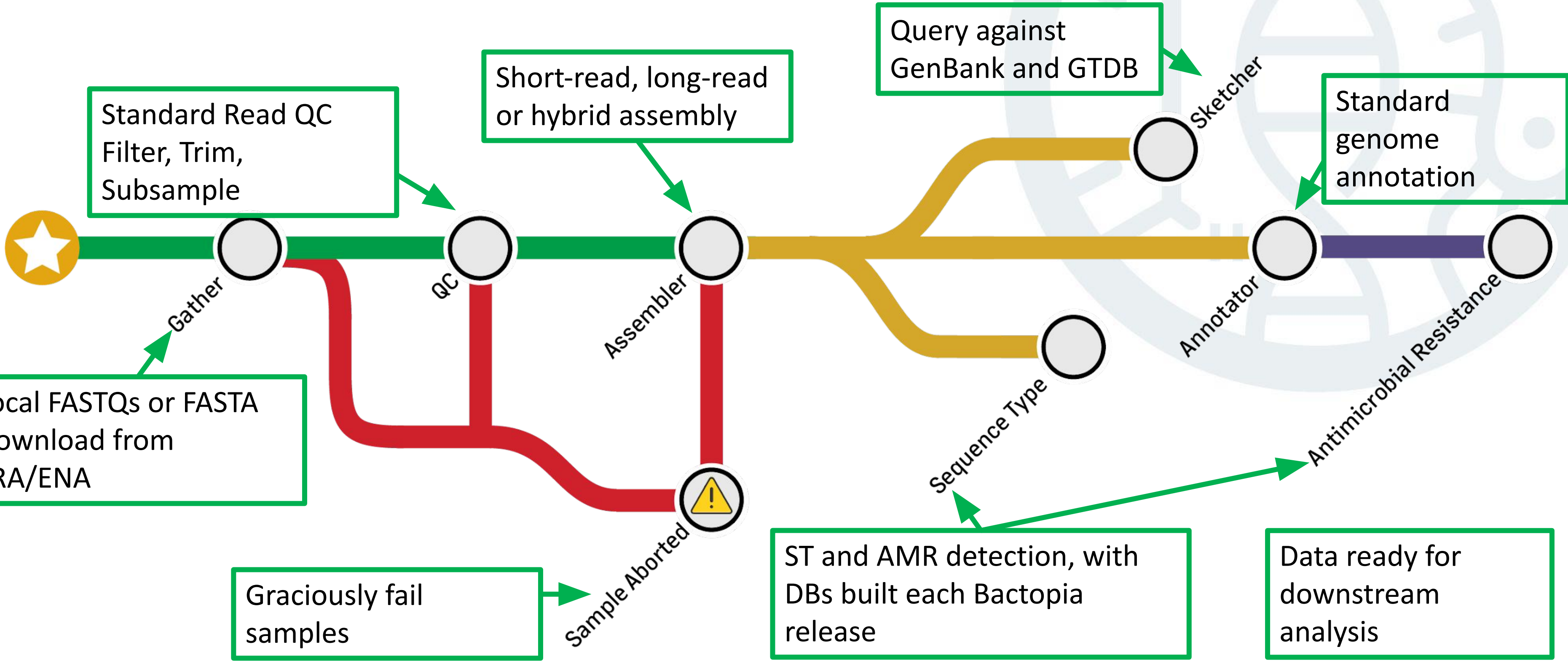
- ***Active user-base providing regular feedback***
    - Includes folks at WPHL, as well as global users from academia, government, and commercial

- ***Consistently maintained for 5+ years***

BACTOPIA
COMPLETE ANALYSIS OF BACTERIAL GENOMES

bactopia.github.io

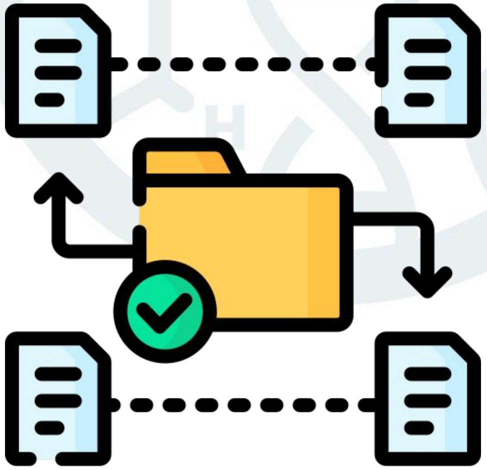
Bactopia – Shaping Data for Next-Step Analyses


Standard Read QC Filter, Trim, Subsample
Short-read, long-read or hybrid assembly
Query against GenBank and GTDB
Sketcher
Standard genome annotation
Gather
QC
Assembler
Annotator
Sequence Type
Antimicrobial Resistance
Local FASTQs or FASTA Download from SRA/ENA
Sample Aborted
Graciously fail samples
ST and AMR detection, with DBs built each Bactopia release
Data ready for downstream analysis

Bactopia organizes your project for you

The ”sample” is the organizational unit

- Tool specific directories
  - Outputs, logs and version info
- Timestamped run specific directories
  - Resource usage and automatic merging of delimited outputs
- Programmatic access to all outputs for automatic import into Bactopia Tools



Bactopia Tools – Simplifying comparative genomics

- 65+ workflows for more science
  - Uses standardized output structure
  - Single parameter change (*--wf*)

- Categories
  - Organism-specific
  - Mobile elements
  - Antimicrobial resistance & virulence
  - Pan-genome
  - SNP/InDel
  - Taxon Classification
  - Phylogeny

Example Bactopia Tool Usage

```
# Process Staphylococcus aureus samples
bactopia --samples saureus.txt

# Run Bactopia Tools

# Staph-specific tools (agr, sccmec, spa)
bactopia --wf staphtyper

# Call SNPs, build core-snp tree
bactopia --wf snippy --accession GCF_000009645
```

Think of Bactopia as a framework of interchangeable modules

- Every step in Bactopia is an *independent* plug and play module
  - Defined input, outputs, and parameters
  - Compatible modules are easily linked

- Allows for easy reshaping of Bactopia to meet user needs
  - *I just want to call SNPs, not all the other stuff*
  - *I just want to see what's in my metagenomic sample*
  - *I just want the species and genome size automatically determined*

Weekly Binf Trainings

Weekly Bioinformatics Trainings

- Training focused on gaining ***practical*** bioinformatics experience
  - Command-line focused
  - Conda and containers (Singularity) usage
  - Slides with high-level information, then practical examples to work through

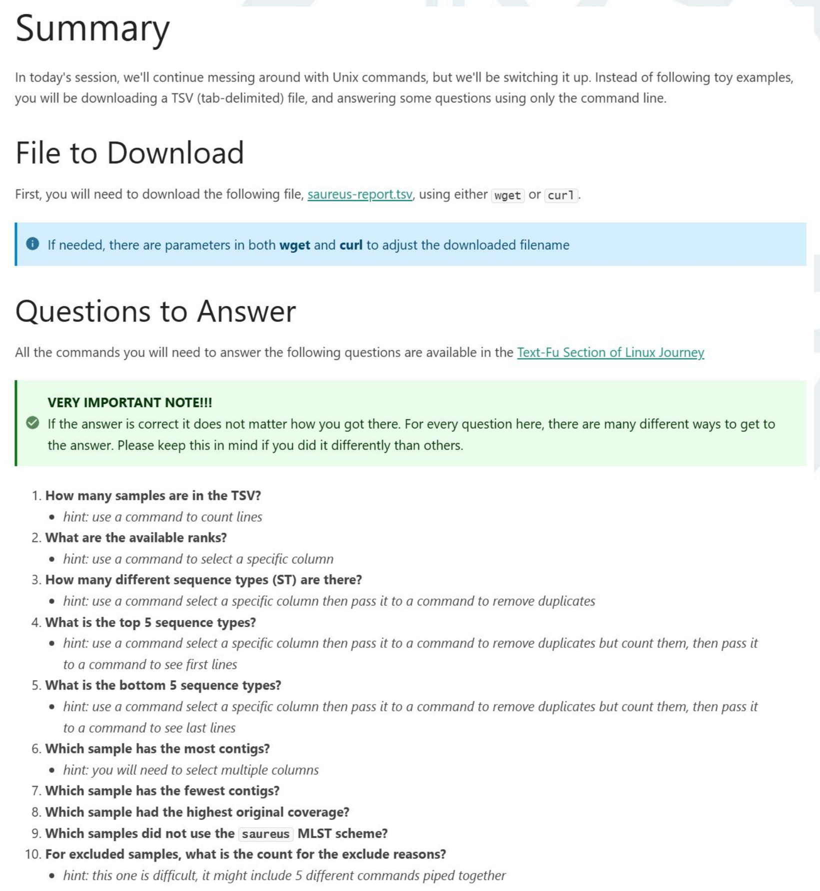
- Who can attend?
  - Anyone that is willing to listen to me for 1-2 hours per-week
  - Mostly the folks from Molecular and Wastewater

- Resources Available
  - We have a COW (*moo!*) with laptops that have WSL2 installed

Example Weekly Training

- Participants were to download a TSV file, then answer questions using commands such as:
  - cut
  - grep
  - head
  - sort
  - tail
  - uniq
  - wc
  - etc...

- *Reminder, these participants are not bioinformaticians*



Summary

In today's session, we'll continue messing around with Unix commands, but we'll be switching it up. Instead of following toy examples, you will be downloading a TSV (tab-delimited) file, and answering some questions using only the command line.

File to Download

First, you will need to download the following file, saureus-report.tsv, using either `wget` or `curl`.

If needed, there are parameters in both **wget** and **curl** to adjust the downloaded filename

Questions to Answer

All the commands you will need to answer the following questions are available in the Text-Fu Section of Linux Journey

VERY IMPORTANT NOTE!!!
If the answer is correct it does not matter how you got there. For every question here, there are many different ways to get to the answer. Please keep this in mind if you did it differently than others.

1. **How many samples are in the TSV?**
   - *hint: use a command to count lines*
2. **What are the available ranks?**
   - *hint: use a command to select a specific column*
3. **How many different sequence types (ST) are there?**
   - *hint: use a command select a specific column then pass it to a command to remove duplicates*
4. **What is the top 5 sequence types?**
   - *hint: use a command select a specific column then pass it to a command to remove duplicates but count them, then pass it to a command to see first lines*
5. **What is the bottom 5 sequence types?**
   - *hint: use a command select a specific column then pass it to a command to remove duplicates but count them, then pass it to a command to see last lines*
6. **Which sample has the most contigs?**
   - *hint: you will need to select multiple columns*
7. **Which sample has the fewest contigs?**
8. **Which sample had the highest original coverage?**
9. **Which samples did not use the `saureus` MLST scheme?**
10. **For excluded samples, what is the count for the exclude reasons?**
    - *hint: this one is difficult, it might include 5 different commands piped together*

Let's wrap this up!

What’s down the road

- Camlhmp
    - More SBTs and methods for typing (e.g. phylotyping, kmers)

- Bactopia v4 next few weeks
    - Full rewrite to support incoming Nextflow changes (>v25)
    - More named workflows (e.g. Corral for parallelizing downloads)

- Weekly training transitioning away from a “binf” focus to more general technology focus
    - Examples: Proper usage of AI, basic scripting for table analysis
    - Extract high-level information for “Advocate Training”

Acknowledgements

The many developers of open source software and the users of Bactopia that are regularly providing feedback.

<u>WPHL</u>
Taylor Fearing
Chayse Rowley
Jim Mildenberger
Rob Christensen
Joseph Reed
Wastewater and Molecular Groups

Thank you and Questions?

Contact Info
robert.petit@wyo.gov

Wyoming Department of Health | PUBLIC HEALTH DIVISION | WYOMING PUBLIC HEALTH LABORATORY