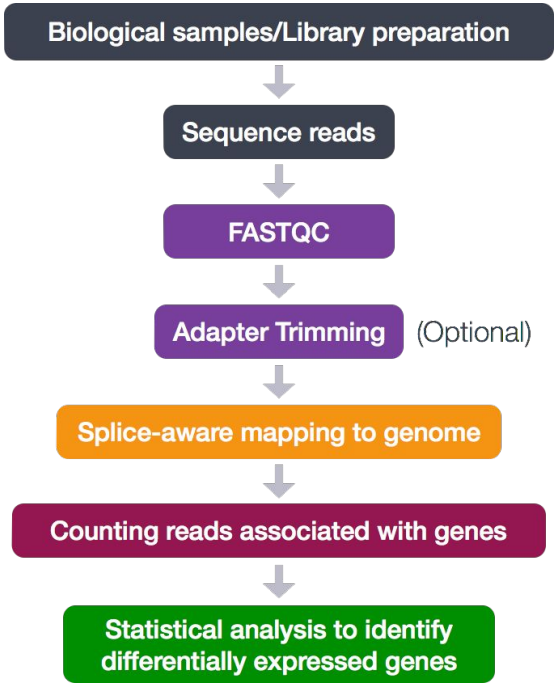
Metadata database project

20220804

Bioinformatics pipelines enable scientists to automate common processes and analyses

- Historically common to save analyses in scripts (Python, Bash, Perl)
- More complex and high-throughput analyses require ways to automate scripts into workflows
- Workflow languages developed to wrap sets of scripts/commands
  - Snakemake
  - Nextflow
  - CWL
  - WDL

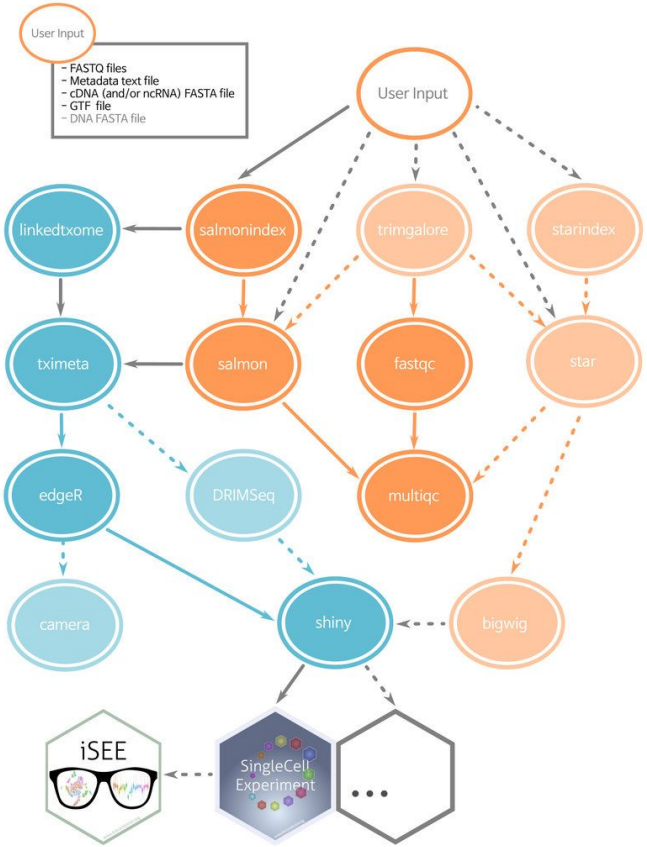
Workflow languages make it easier to reproduce complex analysis efficiently by organizing steps as graphs



Direct Acyclic Graph (DAG) of a simple RNA-seq workflow

https://hbctraining.github.io/Intro-to-rnaseq-hpc-orchestra/lessons/07_rnaseq_workflow.html

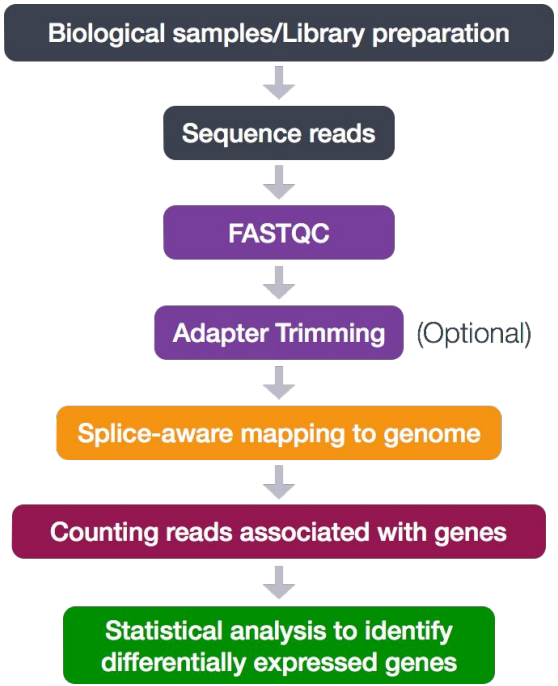
Graphs also make it easier to parallelize steps



If User Input is provided, these steps can be run in parallel

https://www.researchgate.net/figure/Simplified-directed-acyclic-graph-DAG-of-the-ARMOR-workflow-Blue-ellipses-are-rules_fig1_333112051

Workflow languages development can be conceptualized as tool based and target based depending on how we traverse the DAG



Tool-based:

- Start with sequencing reads
- Use FASTQC tool to quality control initial data
- Use Adapter trimming tool to clean data
- Use Splice-aware mapping tool to map cleaned data to genome
- Use Read counter tool to assign mapped reads to features
- Use Statistical analysis tool to identify differentially expressed genes

Target-based (Make file):

- We need (to make) differentially expressed genes, requires read counts
- We need (to make) read counts, requires mapped reads
- We need (to make) mapped reads, requires clean data
- We need (to make) clean data, requires QC'd data
- We need (to make) QC'd data, need sequencing reads

Running a workflow requires a document that describes input files and workflow run parameters

- These documents typically store information such as input file location, reference data location
- Documents may also store metadata (e.g. pipeline/software version, date of runtime, who ran it, etc.)
- Most workflows that we use in lab require this document to be in a specific format, such as JSON or YAML.
  - JSON and YAML formats are interchangeable
  - JSON/YAML formats are both human and machine readable
  - Example JSON formatted workflow file

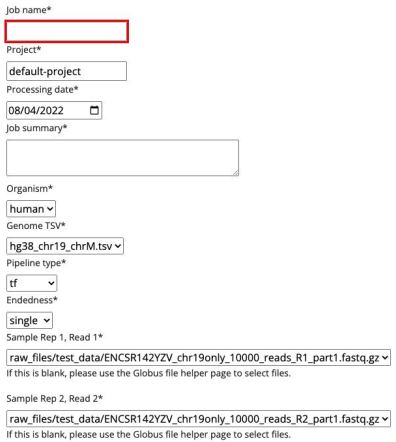
Problem 1: JSON/YAML formats are strict, users often face problems running workflows

New grad student Charlene developed a pipeline for others to run

- This new pipeline requires a JSON file
- Users create required JSON file in a text editor
- Typos or format issues in manually generated JSON file lead to errors in pipeline (e.g. entries are off by one space or bracket)
- Users complain to Charlene that her pipeline sucks
- …

Solution 1: Create a Django app that auto-generates JSON documents in correct format

- This web app will use a fool-proof form that will convert entries into the correct format



Problem 2: our lab runs many workflows/analyses without a clean way of searching what has been done already

New grad student Sam wants to generate data from XYZ cells treated with ABC

- Has anyone tried this before?
- Look up old publications from old grad student
- Try contacting old grad student
- No response
- …
- Re-generate data ($$$)

New grad student Kat wants to reproduce some results from older publication

- Has anyone tried this before?
- Old grad student supplies scripts/random code
- Code not reproducible (paths not saved, params not saved)
- …
- Spend extra time ($$$) debugging/reproducing analysis

Solution 2: Develop or find a method to organize data/analyses

- Potential Solution: Use existing software or service to store/retrieve workflow metadata
  - https://metadatacenter.org/
  - https://www.synapse.org/
  - Others?
- Potential Solution: Save/store these JSON/YAML files into a database that Yeo labbers can search through
- Open question: How to organize Jupyter notebooks?
  - Github?
  - Others?

Roadmap

- Deploy a Django site
- Develop a Django web app that creates properly formatted JSON or YAML documents for the pipelines we have
  - Start with a Django Forms tutorial
  - Work with Brian to develop final app
- Evaluate existing tools/services that organize data
  - How do these services store metadata?
  - Are these services up to date/maintained?
  - How easy is it to use/deploy?
  - How much do they cost?
- Build a metadata database
  - Build a Django web app that accesses existing tools Application Programming Interface (API) to search/upload/retrieve metadata
  - Build a Django web app that queries an in-house database
    - Deploy a (noSQL) database that can store/search JSON or YAML documents (e.g. mongodb)