Bacteriophage Annotation Documentation

Release 0.8-rc1

Center for Phage Technology

February 07, 2017

Contents:

CHAPTER 1

Introduction to Galaxy

1.1 Introduction

Galaxy is a platform for doing reproducible bioinformatics research. It provides a friendly interface to a vast number of complex command line tools, and it encourages consistent science by using identical software and interfaces across all Galaxy instances.

At the CPT we depend on Galaxy for all of our computer-based analysis; we can launch long running jobs and return to our lab work, while Galaxy keeps track of where and how far along our analysis is, during runs of our pipelines.

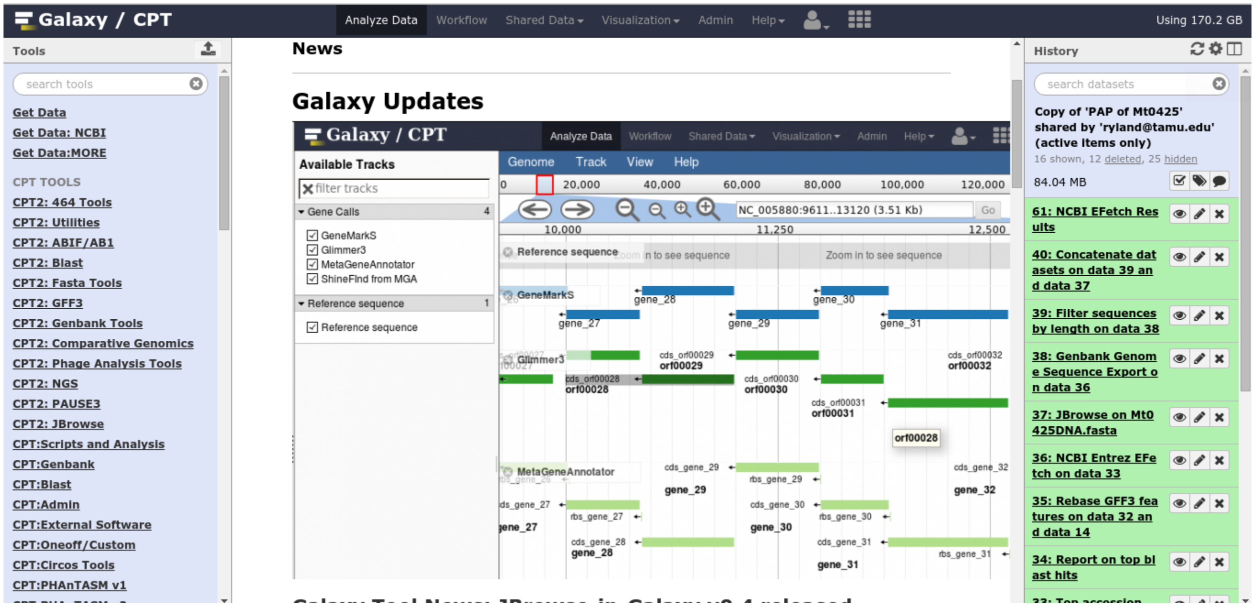
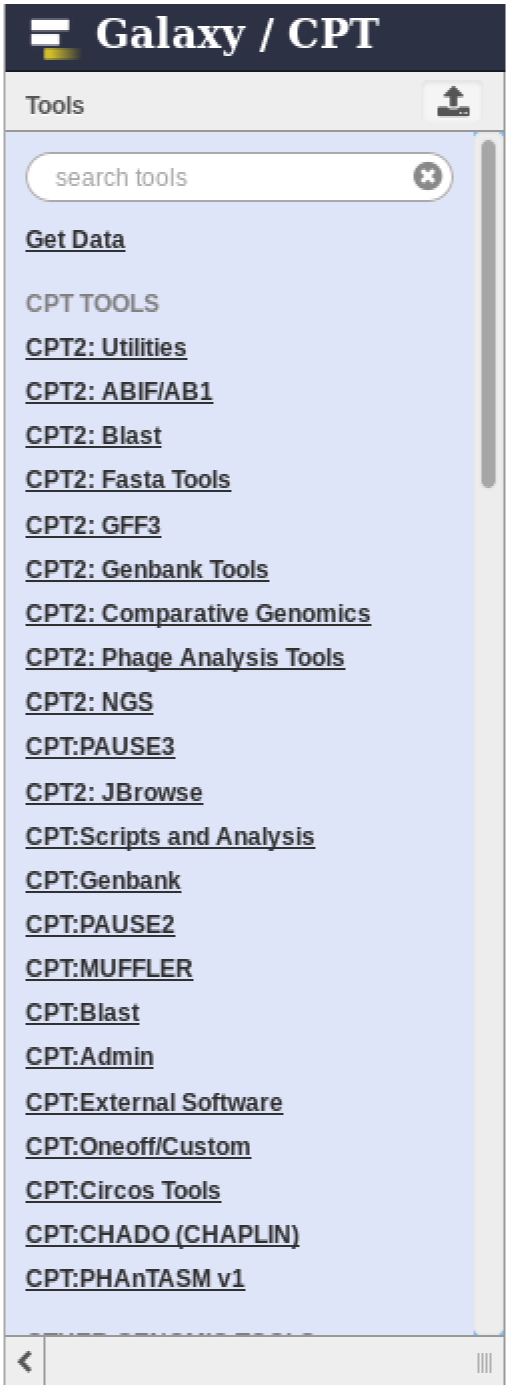


Fig. 1.1: Main Galaxy View

Galaxy consists of a three panel interface. On the left are your tools, in the center you'll do your analysis and view the results, and on the right is your history.

These are all groups of tools you can run in Galaxy. A tool is something that generates or transforms data. Some examples of tools would be a Gene Caller, which reads your genome and returns a list of gene locations, or Blast, which would search your protein sequences against a database.

In Galaxy, tools are simple interfaces to the complex software behind them. Galaxy tools enhance productivity by ensuring that your input files are the correct format at every step.



Galaxy / CPT

Tools

search tools

Get Data

CPT TOOLS

CPT2: Utilities

CPT2: ABIF/AB1

CPT2: Blast

CPT2: Fasta Tools

CPT2: GFF3

CPT2: Genbank Tools

CPT2: Comparative Genomics

CPT2: Phage Analysis Tools

CPT2: NGS

CPT:PAUSE3

CPT2: JBrowse

CPT:Scripts and Analysis

CPT:Genbank

CPT:PAUSE2

CPT:MUFFLER

CPT:Blast

CPT:Admin

CPT:External Software

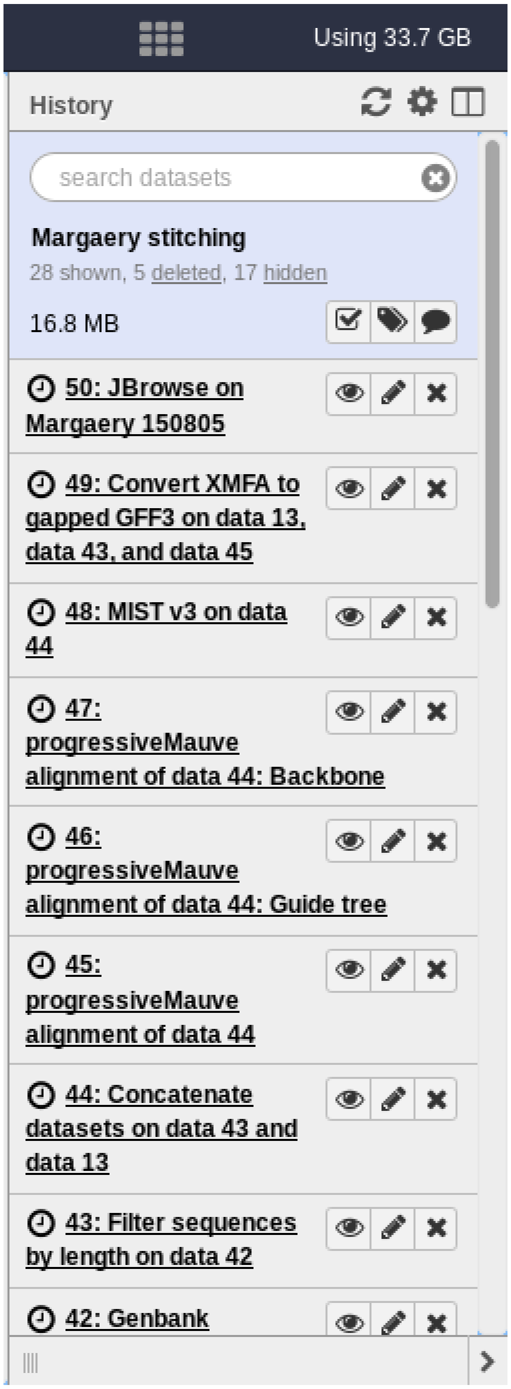
CPT:Oneoff/Custom

CPT:Circos Tools

CPT:CHADO (CHAPLIN)

CPT:PHAnTASM v1

Fig. 1.2: Tool Panel



Margaery stitching

28 shown, 5 deleted, 17 hidden

16.8 MB

50: JBrowse on Margaery 150805

49: Convert XMFA to gapped GFF3 on data 13, data 43, and data 45

48: MIST v3 on data 44

47: progressiveMauve alignment of data 44: Backbone

46: progressiveMauve alignment of data 44: Guide tree

45: progressiveMauve alignment of data 44

44: Concatenate datasets on data 43 and data 13

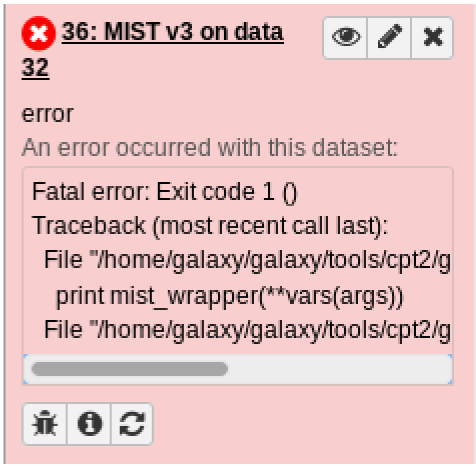
43: Filter sequences by length on data 42

42: Genbank

Fig. 1.3: History Panel

The history panel keeps track of what you've done. Each entry is called a "dataset" in Galaxy terminology. Datasets are usually just a single file (like a fasta genome), but may be complex files (like html web pages). The colors indicate in which of the three states a job could be; grey for jobs that have been submitted to Galaxy, yellow for a jobs that are currently running, and red/green for a jobs that are completed.

> **Warning:** Sometimes jobs fail and turn red! If they do, don't worry–it likely isn't your fault. Just be sure to click the bug icon so we can be made aware of what went wrong.



36: MIST v3 on data 32

error

An error occurred with this dataset:

```
Fatal error: Exit code 1 ()
Traceback (most recent call last):
  File "/home/galaxy/galaxy/tools/cpt2/g
    print mist_wrapper(**vars(args))
  File "/home/galaxy/galaxy/tools/cpt2/g
```

Fig. 1.4: The bug icon is in the bottom left. Please click it if you see it!

1.2 Tools

Tools are the central feature of Galaxy, they're what Galaxy is all about: easy-to-use access to powerful command line tools. We use the language "tool" to describe some command line program that has been "wrapped" for use in Galaxy. Many of the command line options are available to be tweaked and fiddled with in the Galaxy tool interface.

To run a tool, first read the tool's help box at the bottom, it may give you very important information regarding running the tool, and the options available to you.

Next go over the options in the tool interface, many are set to default values and those options may not be what you intended to happen.

Nearly all Galaxy tools process some input files and produce some output files. When you have an input box that lets you select a file, Galaxy will ensure that you cannot pass datasets of an incorrect format to a tool.

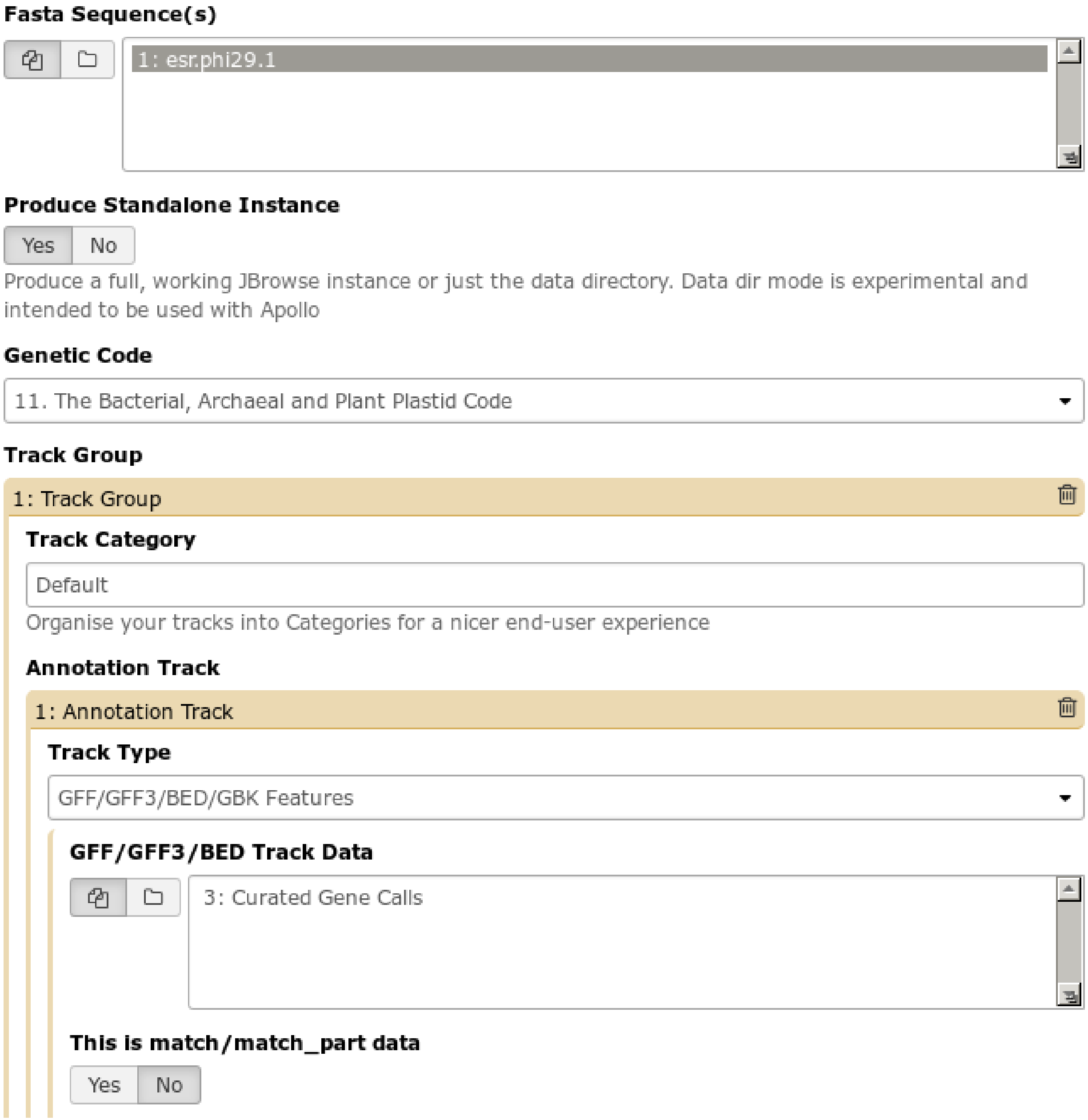
When you've finished configuring a tool, and **execute** it, it will show up as a set of output files in the history on the right, which we'll cover in the next section..

1.3 Jobs

A job is a tool run with a set of parameters, and it sits somewhere in Galaxy's queue. We have lots of Galaxy users and can only run so many jobs at once. Jobs produce one or more output files.

Looking at an individual output dataset, there will be several features that will be of interest to you, and a couple that won't:

In this example we see dataset #29, which is a table of Start Codon Usages.



Fasta Sequence(s)

1: esr.phi29.1

Produce Standalone Instance

Yes No

Produce a full, working JBrowse instance or just the data directory. Data dir mode is experimental and intended to be used with Apollo

Genetic Code

11. The Bacterial, Archaeal and Plant Plastid Code

Track Group

1: Track Group

Track Category

Default

Organise your tracks into Categories for a nicer end-user experience

Annotation Track

1: Annotation Track

Track Type

GFF/GFF3/BED/GBK Features

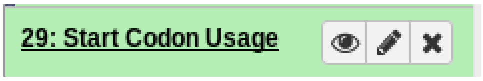
GFF/GFF3/BED Track Data

3: Curated Gene Calls

This is match/match_part data

Yes No

Fig. 1.5: This Galaxy tool has a large number of options. Many tools are highly configurable to meet the needs of a wide variety of end users.



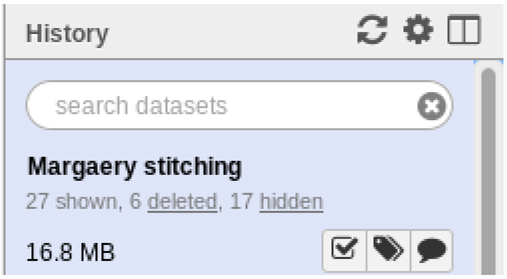
29: Start Codon Usage

Fig. 1.6: An example Dataset

This is the collapsed view of a dataset. You'll see three icons, an eye, a pencil, and an X.

- The eye icon will display the dataset in the main window
- The pencil icon is used to edit the information about the dataset (i.e. metadata, such as filename, file type, and other more obscure facts)
- And the X icon will delete a dataset, for which you no longer have any use

When you delete a dataset accidentally, don't worry! It isn't gone permanently.



History

search datasets

Margaery stitching

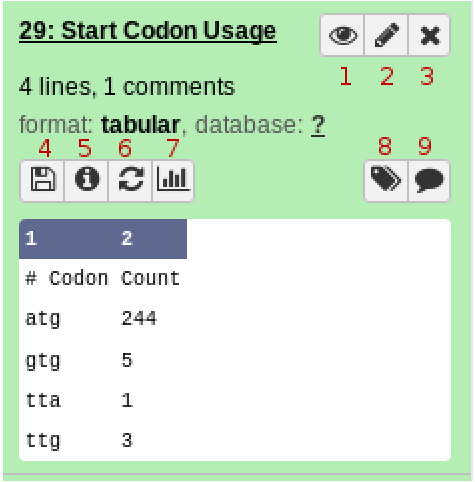
27 shown, 6 deleted, 17 hidden

16.8 MB

Fig. 1.7: History Header

See the text "6 deleted"? Clicking on the word "deleted" will show you the deleted items in your history.

Clicking anywhere on the title of the dataset, "Start Codon Usages" will expand to the full view:



29: Start Codon Usage

4 lines, 1 comments

format: **tabular**, database: ?

1	2
# Codon	Count
atg	244
gtg	5
tta	1
ttg	3

Fig. 1.8: Example Dataset in the expanded view.

At the bottom of the dataset you can see a preview of the table. Near the top you'll note the it is a "tabular" format. Galaxy keeps track of file formats to ensure you only use correctly-formatted data for tools.

The history dataset view is information dense, so we'll go over the icons, their colloquial names, and their functions.

1. **Eyeball** views the dataset
2. **Pencil** modifies metadata
3. **X** sends a dataset to the trash. Remember, you can recover deleted datasets.
4. **Save** downloads the dataset to your hard-drive. You don't *need* to do this, as Galaxy will always have a copy for you.

5. **Information** views details about the tool that was run and how it was configured.
6. **Rerun** is a very commonly used button. This lets you re-run the tool, with the same parameters configured.
    - Need to run the same tool with slightly different parameters? Don't waste time filling out the tool form; re-run it and tweak those.
    - Job failed? Try modifying the tool inputs and re-running it.
7. **Visualize** lets you visualize your dataset in a couple of different ways. We don't use this very often in the course as it isn't appropriate to our analysis. However, some of the useful visualizations include: a "charts" visualization that lets you build graphs from your tabular dataset, and an Multiple Sequence Alignment (MSA) visualization plugin in Galaxy which lets you interactively explore MSAs.
8. **Tags** let you annotate datasets with tags. We don't use this feature.
9. **Comments** let you comment on a dataset to remind yourself why you did it, or maybe to annotate some interesting results you found in the output.

1.4 Histories

Histories are used to separate out your analyses and help you keep track of what you've done. You might make a new history for a task like assembling a genome, annotating a genome, or doing some comparative analysis between phages. It's good practice to title your history something that will be useful to you when you come back later. Who knows what "Untitled History" last edited on 2012-02-01 was for? Good names are important!

At the top right of your history you'll see a couple icons. A refresh symbol, a gear, and the new history view.

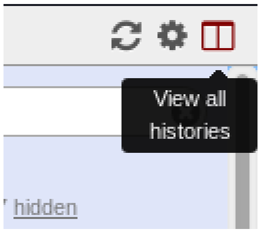
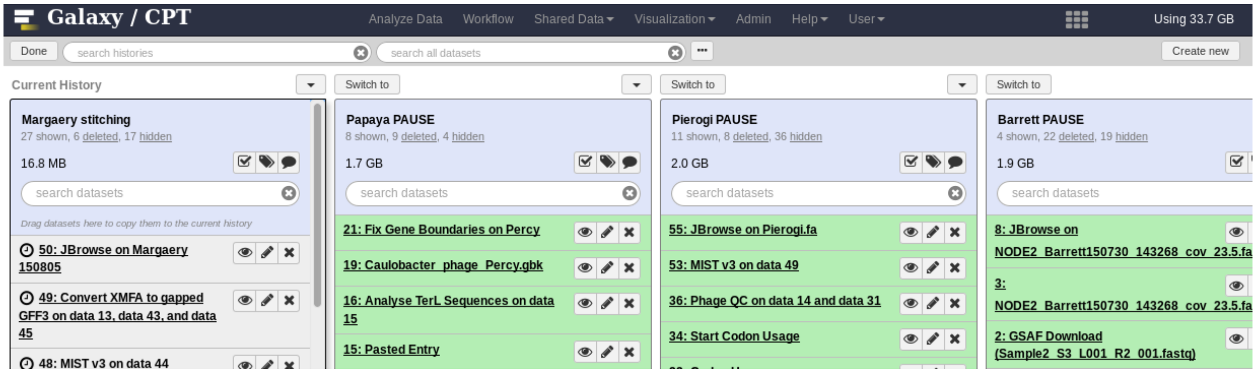


Fig. 1.9: History Menu

The refresh button can be used to refresh your history if you're impatient, like I can be. The gear icon provides you with the old interface to switch histories and modify the view. We'll be using the new "View all histories" view:



Margaery stitching
27 shown, 6 deleted, 17 hidden
16.8 MB
search datasets
Drag datasets here to copy them to the current history
50: JBrowse on Margaery 150805
49: Convert XMFA to gapped GFF3 on data 13, data 43, and data 45
48: MIST v3 on data 44

Papaya PAUSE
8 shown, 9 deleted, 4 hidden
1.7 GB
search datasets
21: Fix Gene Boundaries on Percy
19: Caulobacter_phage_Percy.gbk
16: Analyse TerL Sequences on data 15
15: Pasted Entry

Pierogi PAUSE
11 shown, 8 deleted, 36 hidden
2.0 GB
search datasets
55: JBrowse on Pierogi.fa
53: MIST v3 on data 49
36: Phage QC on data 14 and data 31
34: Start Codon Usage

Barrett PAUSE
4 shown, 22 deleted, 19 hidden
1.9 GB
search datasets
8: JBrowse on NODE2_Barrett150730_143268_cov_23.5.fa
3: NODE2_Barrett150730_143268_cov_23.5.fa
2: GSAF Download (Sample2_S3_L001_R2_001.fastq)

Fig. 1.10: The new (in 2015) Galaxy History Switcher. It is a huge improvement over the old one because you can easily move datasets between histories

At the top left you'll see a **done** button, which will let you exit this view when you're done. On the left is your current history. To the right of that are your other histories. The **switch to** button allows you to switch histories. Any new tools you run will be done in this history.

1.5 Uploading Data

Switch back to the main view of Galaxy (i.e. click **done** if you're still in the switch history menu from above). If you ever find yourself lost in Galaxy, you can always click **Analyze Data** on the big blue bar at the top, and it'll take you home.

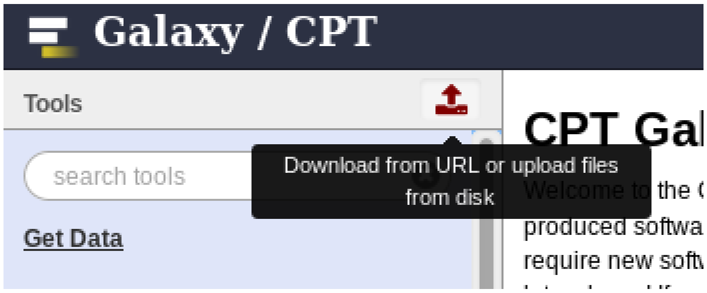
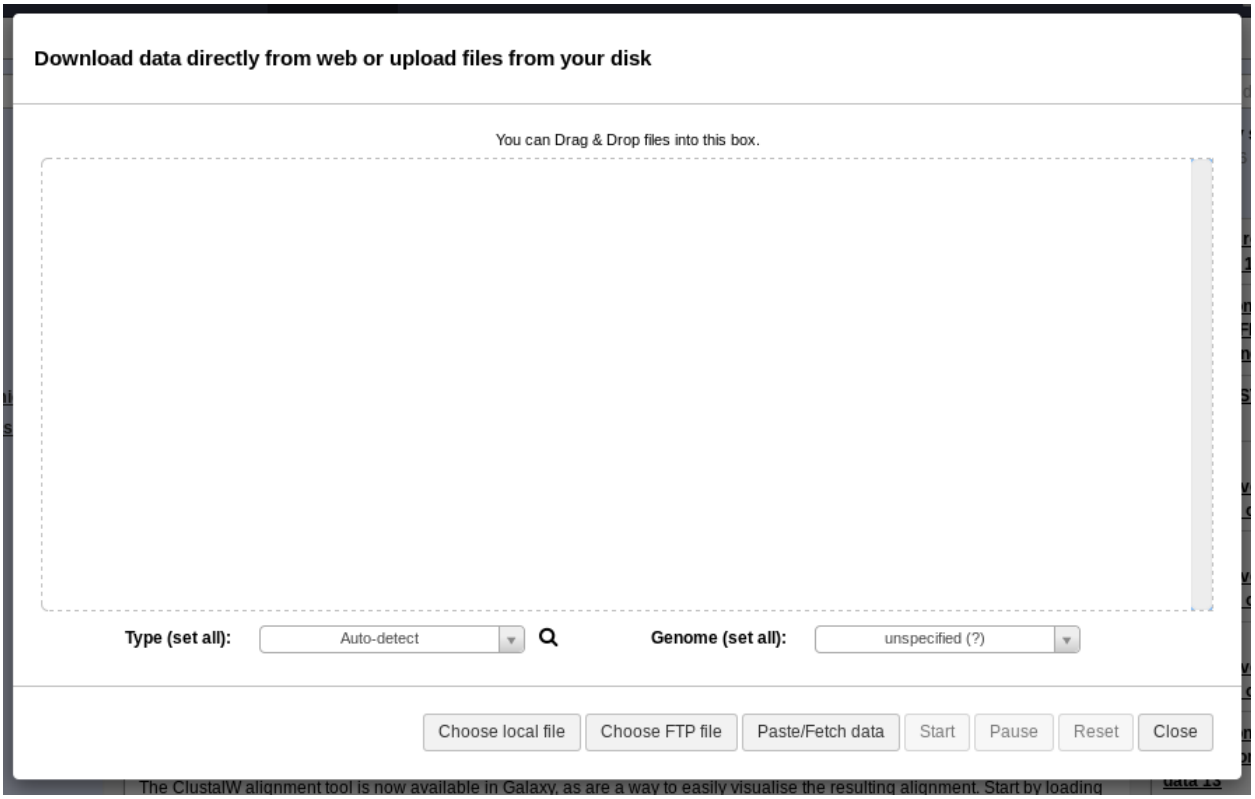


Fig. 1.11: Upload

This button will bring up your upload menu and allow you to import data into Galaxy.


Download data directly from web or upload files from your disk

You can Drag & Drop files into this box.

Type (set all): Auto-detect

Genome (set all): unspecified (?)

Choose local file
Choose FTP file
Paste/Fetch data
Start
Pause
Reset
Close
The ClustalW alignment tool is now available in Galaxy, as are a way to easily visualise the resulting alignment. Start by loading

Fig. 1.12: Upload Window

You can drag and drop files to upload them, or use the **Choose Local File** button in the bottom menu.

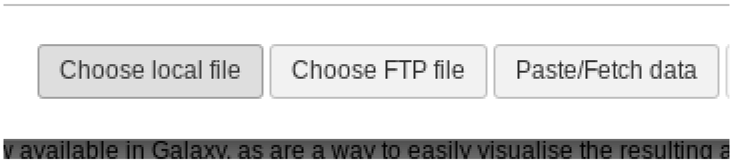
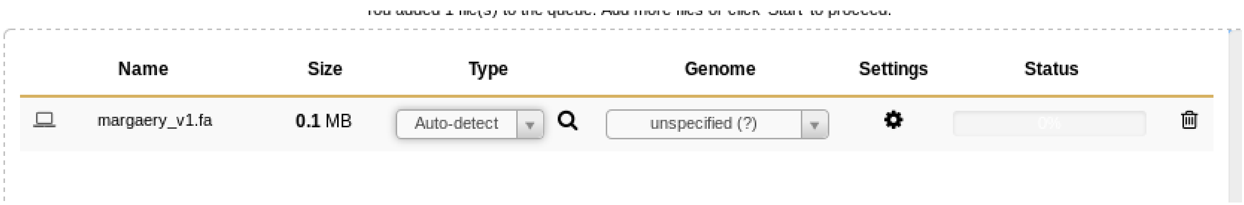

Choose local file
Choose FTP file
Paste/Fetch data
v available in Galaxy, as are a way to easily visualise the resulting a

Fig. 1.13: Upload Menu

There are a couple other options for advanced users:

- **Choose FTP file** allows you to select a file you've uploaded via FTP. For files >2GB, this is required.
- **Paste/Fetch data** allows you to paste in a bit of text or a URL. Galaxy will import that into your history.

Once the file has been detected by Galaxy, you'll see it pop up in the upload window:



You added 1 file(s) to the queue. Add more files or click 'Start' to proceed.

	Name	Size	Type	Genome	Settings	Status
	margaery_v1.fa	**0.1** MB	Auto-detect	unspecified (?)		0%

Fig. 1.14: Uploading File

You can set the file type if Galaxy doesn't detect it properly, but that is a rare case, and before overriding Galaxy you should double check that your file is formatted properly.

When you've selected all the files you wish to upload, click **Start** in the bottom right of the upload menu.

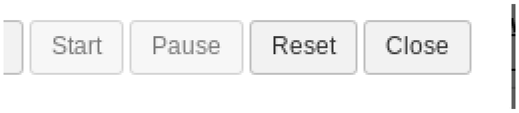


Fig. 1.15: Starting Upload

The dataset will indicate to you that it is uploaded in the upload window,

at which time you can close that window with Escape or the **Close** button in the bottom right.

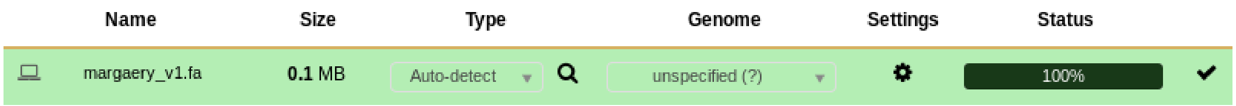
The dataset will turn yellow

And then green when it is ready.

1.6 Workflows

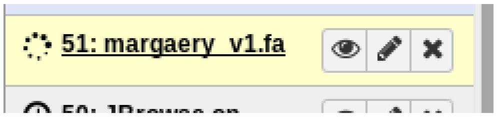
Workflows are merely collections of jobs where some jobs depend on the outputs of other jobs. Say you're faced with a task like the following:

1. Load data from apollo
2. Extract all of genes as DNA sequences



	Name	Size	Type	Genome	Settings	Status	
💻	margaery_v1.fa	0.1 MB	Auto-detect	unspecified (?)	⚙	100%	✓

Fig. 1.16: Uploading...



51: margaery_v1.fa

50: JBrowse on

Fig. 1.17: Processing an upload

3. Translate those to protein sequences
4. Run those proteins through BlastP

If you did these tasks one by one, you would have to keep track of *at least* 4 different files, one as the output of each step. You have to wait for each program to finish, before you can execute the next one. What if step 2 took 10 minutes? And Step 3 took 15? You would have to check back every few minutes to see if your job was done before you could start the next.

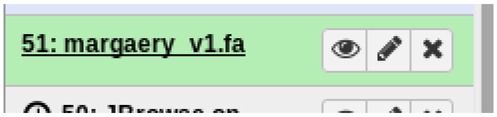
Thus, enter workflows:

Workflows solve numerous problems for us:

- Run tools immediately
  - The next step can start as soon as data is available, a human does not have to manually start it
- Discard useless data
  - In the above example we probably only cared about the output BLAST results, we don't care about storing the intermediate files forever.
- Simplified interfaces
  - In the same way that Galaxy tools hide the complexity of hundreds of command line options and working in Linux, Workflows hide the complexity of many Galaxy tools.
  - You, as a user, are probably not interested in the fact that we have to extract features from a GFF3 file, and then translate those to protein sequences.

1.6.1 Importing

You will often be asked to import workflows. You can do this by going to the **Published Workflow** page, and finding a workflow you're interested in.



51: margaery_v1.fa

50: JBrowse on

Fig. 1.18: Finished upload

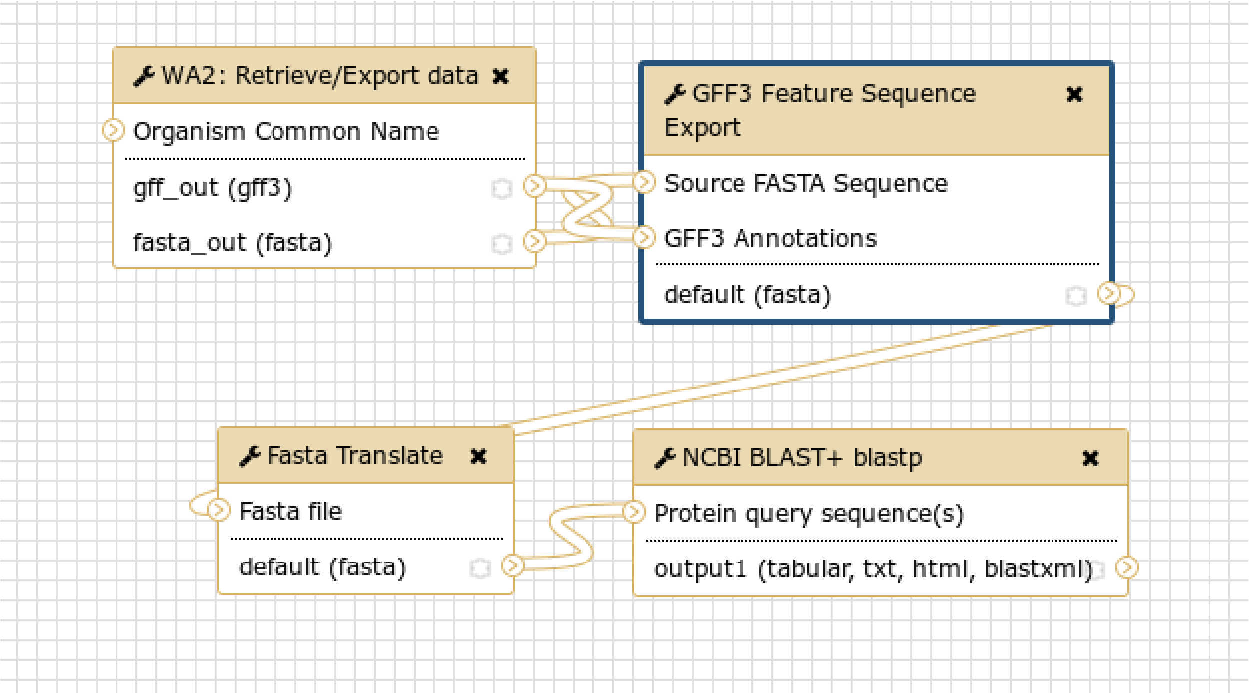


Fig. 1.19: An example workflow encapsulating the four steps from our example workflow

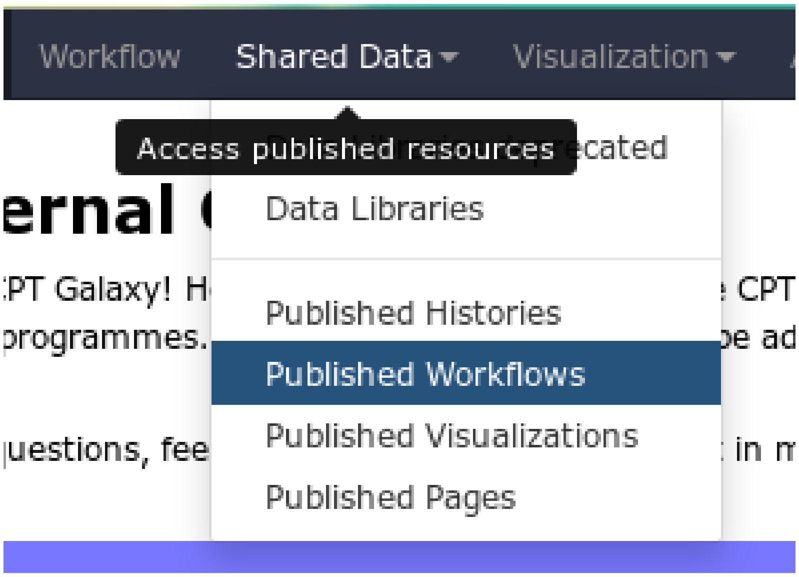


Fig. 1.20: Before you create a new workflow, check the published workflow page. Another user may have created a workflow you can use.

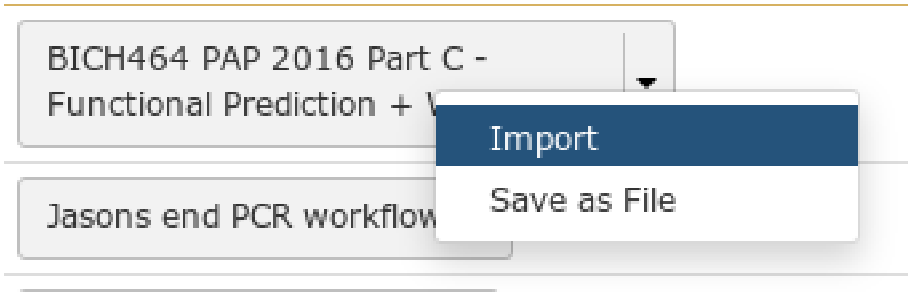


Fig. 1.21: Importing workflows is easy, just click the little down arrow, and select "Import"

1.6.2 Running

Workflows which you have created, or imported, are available under the **Workflow** menu at the top of Galaxy.

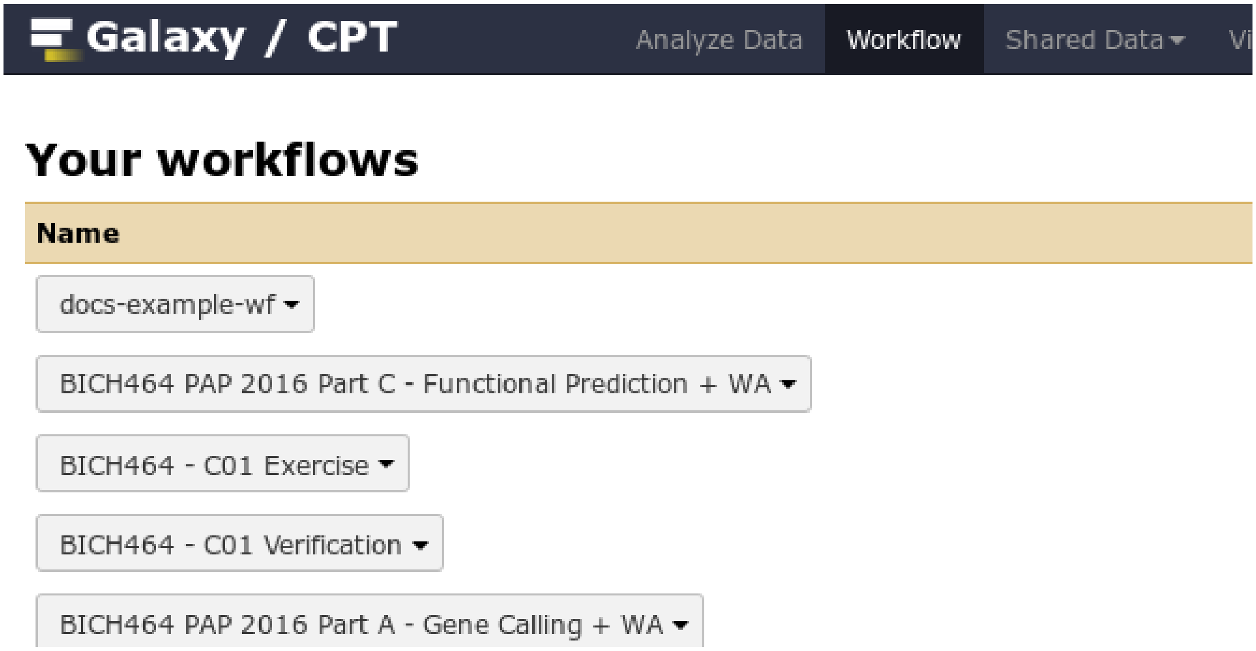

Galaxy / CPT
Analyze Data
Workflow
Shared Data
Vi
Your workflows
Name
docs-example-wf
BICH464 PAP 2016 Part C - Functional Prediction + WA
BICH464 - C01 Exercise
BICH464 - C01 Verification
BICH464 PAP 2016 Part A - Gene Calling + WA

Fig. 1.22: Some of the author's Galaxy workflows. The author has somewhere around 60 different workflows, as they are instrumental in running complex analyses on Galaxy

The run workflow interface can be somewhat overwhelming. For the large part, the tools are pre-configured for you. As the course progresses we'll cover in detail what each portion of the workflow does.

Just like with tools, there is an **Execute** button at the bottom which will launch the workflow.

1.7 Recap

At this point you should be fairly comfortable:

- Uploading data
- Job Outputs

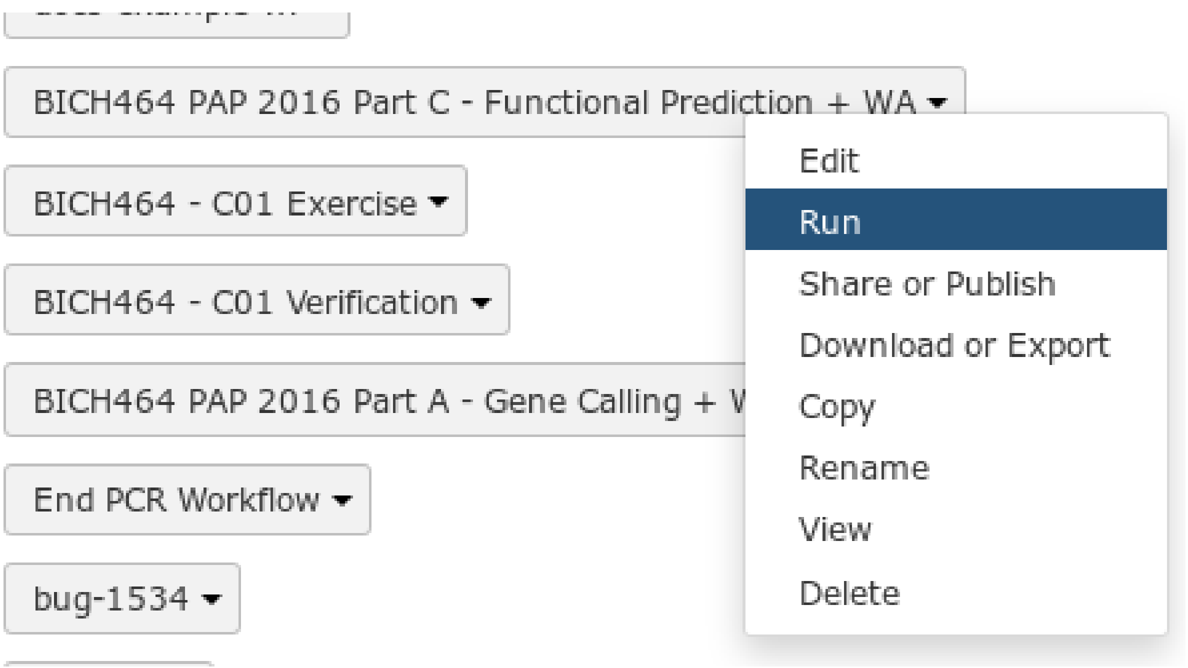
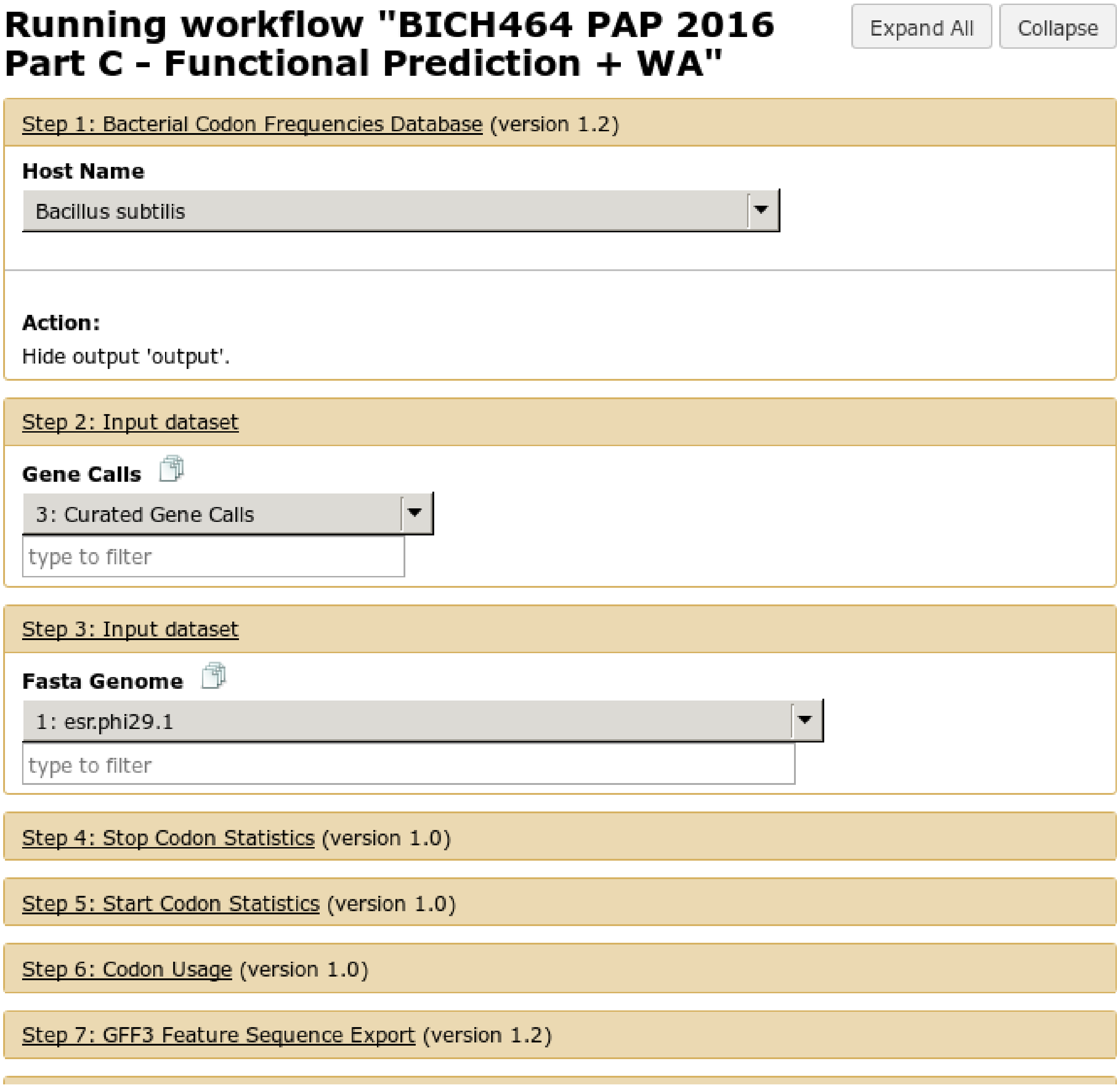

BICH464 PAP 2016 Part C - Functional Prediction + WA
BICH464 - C01 Exercise
BICH464 - C01 Verification
BICH464 PAP 2016 Part A - Gene Calling + W
End PCR Workflow
bug-1534
Edit
Run
Share or Publish
Download or Export
Copy
Rename
View
Delete

Fig. 1.23: To run a workflow, click the down arrow, and select the **Run** option

- Running Tools
- Switching between histories
- Importing workflows
- Running workflows



Running workflow "BICH464 PAP 2016 Part C - Functional Prediction + WA"

Step 1: Bacterial Codon Frequencies Database (version 1.2)

Host Name
Bacillus subtilis

Action:
Hide output 'output'.

Step 2: Input dataset

Gene Calls
3: Curated Gene Calls

Step 3: Input dataset

Fasta Genome
1: esr.phi29.1

Step 4: Stop Codon Statistics (version 1.0)

Step 5: Start Codon Statistics (version 1.0)

Step 6: Codon Usage (version 1.0)

Step 7: GFF3 Feature Sequence Export (version 1.2)

Fig. 1.24: Running a workflow. Some boxes which are automatically expanded may require your attention, the ones which are closed may not require attention

CHAPTER 2

Apollo

2.1 Background

Apollo is a Genome Browser. It lets you visualize genes on a genome, create and edit genes, and create and edit annotations on those genes.

Genome Browsers are a vital tool for rapidly annotating genomes. They let you visualize multiple data sources and help you synthesize those into a rational set of annotations.

2.1.1 Definitions

Static Unmodifiable. This is often used to refer to a set of "static" files made available over the web, or some other computer resource which cannot be modified. (Note that cannot be modified simply means that that exact file cannot be modified, but it is often possible to replace it with an updated version which is modified)

Instance A specific copy of (usually) a web service made available over the internet. Given that the same web service could be duplicated and both could be accessible to users, we use the term "instance" to refer to a specific copy of a service.

Credentials Username and password. Also may be used to mean an "API Key" which you can consider as a combined username and password

Site This is sometimes used to mean your lab or organisation. Generally the people who have both deployed an Apollo instance for you, and you work with to annotate a genome.

2.1.2 GMOD, GBrowse, JBrowse, Apollo, what's it all mean?

This section will be a bit of history about Genome Browsers, and while not important to the annotation process, it can be helpful to know what the terms means and how the parts all fit together.

We use a lot of software under the umbrella term of GMOD, the Generic Model Organism Database

The GMOD project collects open source software under a single umbrella, all related to the idea of publicly accessible, open source Model Organism Database (MOD). These are important, as historically every lab did their own thing. By having, at the very least, a common set of software for MODs, everyone could benefit from interoperability. Software that talked to one MOD could be re-used when talking to another MOD.

GBrowse

GBrowse was one of the earlier genome browsers, and continues to have wide popularity as it handles massive datasets quite well.

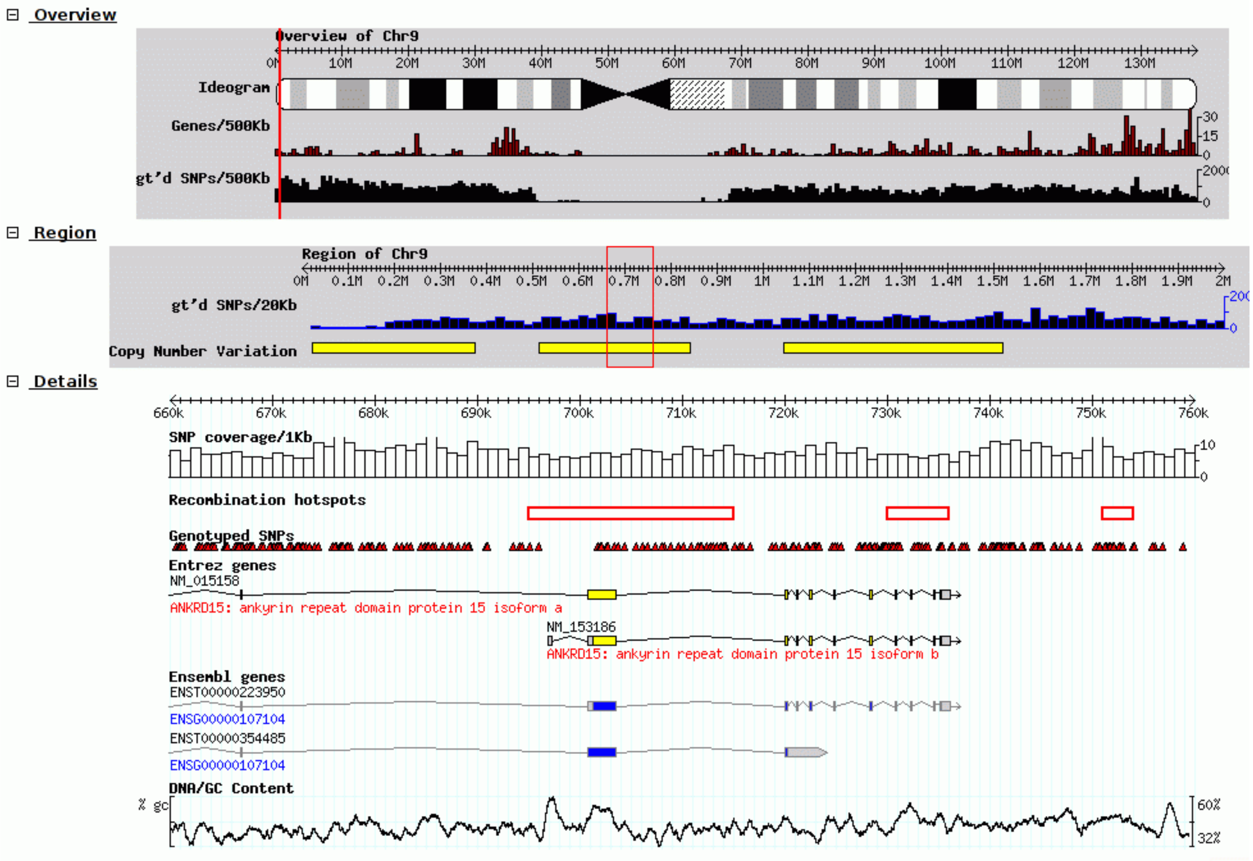


Fig. 2.1: Screenshot of GBrowse from GMOD Wiki

Tracks are rendered on the server, and the client view the genome and datasets through a set of static images shown to the user.

JBrowse

JBrowse and GBrowse are related and attempt to solve the same problem, JBrowse is a more modern, javascript version that does all of the processing on the client. This can make JBrowse much more responsive to user interactions, like how Google Maps was an improvement over previous web maps which required you to click and the page to refresh to load a new section of the map.

Many labs have deployed JBrowse instances to help showcase their annotation efforts to the community, and to make their data accessible. Here you can see a demonstration of *Drosophila melanogaster* genes from FlyBase in JBrowse and play around with it.

Note that JBrowse is a *static visualization tool*, you cannot make any changes to the data, you cannot provide annotation and save them. It is a “Read Only” view of genomes and annotations.

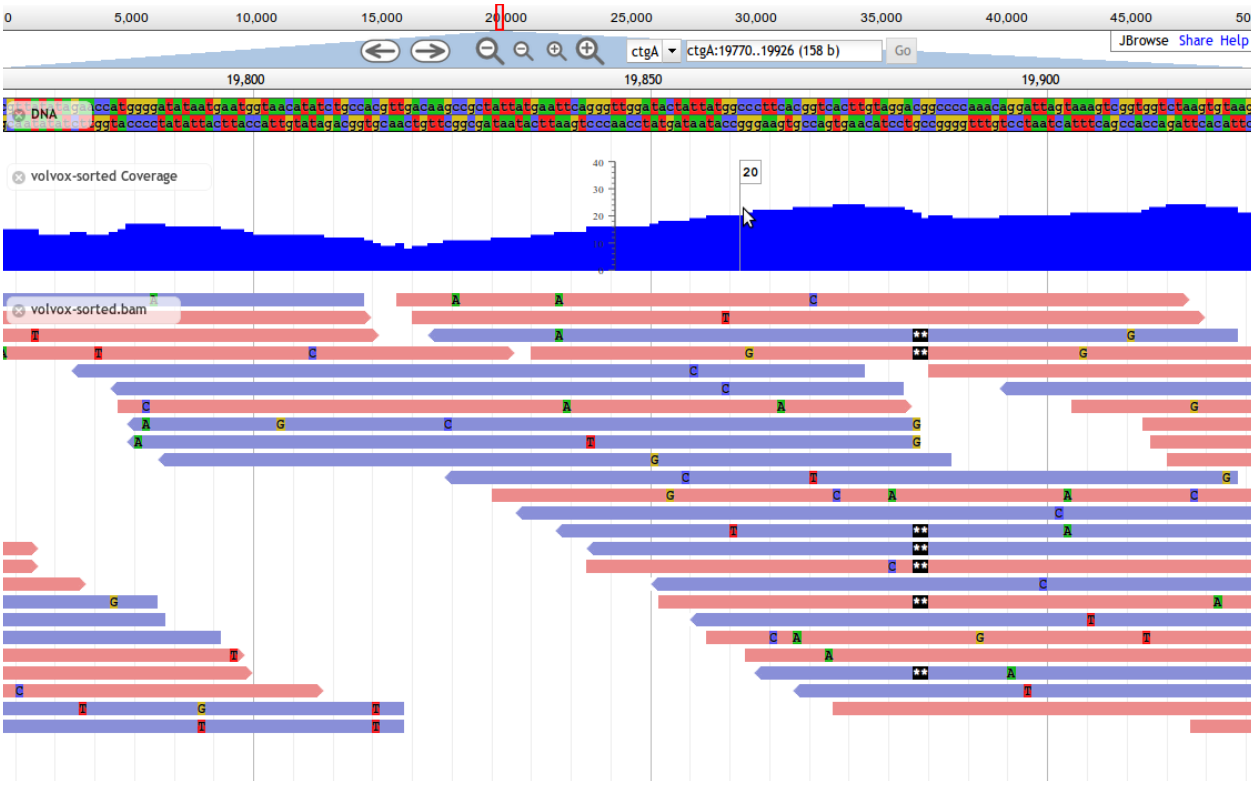


Fig. 2.2: Screenshot of a JBrowse instance. In this figure, JBrowse is shown displaying some alignments of sequencing reads to a genome. JBrowse has the ability to do things like dynamically colouring reads based on multiple properties, include read quality and direction.

JBrowse automatically has generated both a sequencing depth track (blue) and a set of individual reads. Neither of these tracks were pre-computed.

Apollo

Apollo takes JBrowse one step further and adds infrastructure for community annotation; it provides a “Read+Write” view of genomes. You can create new gene features, new annotations on those features, and these are shared with everyone who has access to the Apollo server.

Apollo embeds JBrowse, so if you are familiar with JBrowse, many of the same skills apply.

2.1.3 Annotation File Formats

There are two formats you need to be aware of during genome annotation.

1. Fasta, the format used to store DNA and protein sequences.
2. GFF3, a widely used format for storing genome annotations.
3. GenBank, an older format used by NCBI

Fasta

Many of you may have seen a fasta formatted sequence before, but briefly it looks like:

```
>phiX  Complete genome sequence of phage X
ACTGACTGATCGACTGCGTACGATCGACTGACT
CTGCGTACGATCGACTGACTACTGACTGATCGA
...
```

Each sequence starts with a >, and has a “fasta ID” after it. Some sequences have a “description” after the sequence, like the in the above “Complete genome...”

The sequences contained within a fasta file may be DNA, RNA, or protein sequences.

GFF3

Eukaryotic Gene Model

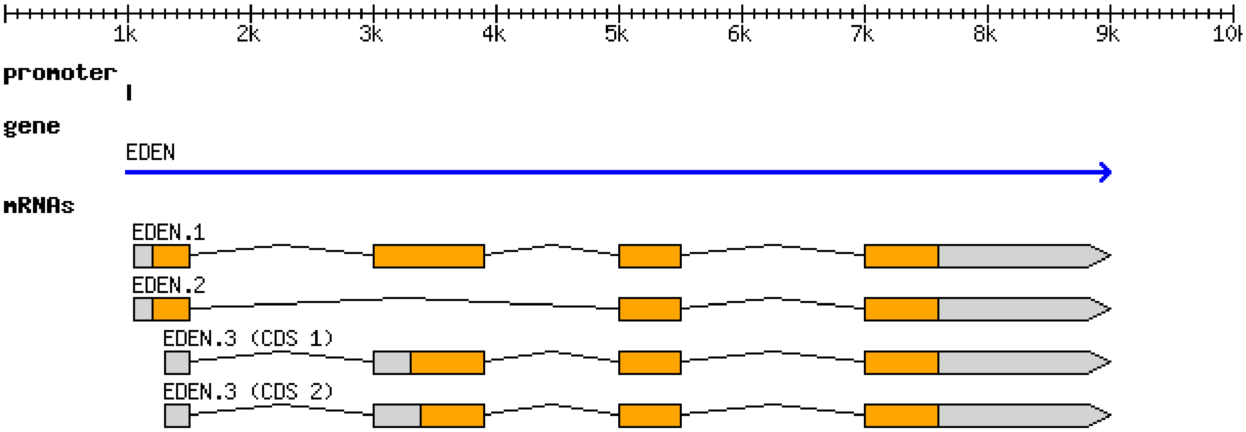
Many of you are probably familiar with the eukaryotic gene model. This model captures a lot of information about the biological process behind producing proteins from DNA, such as mRNAs, transcription, and alternative splicing. GFF3 files thus have to encode these complex, hierarchical, parent-child relationships.

Let’s look at what a GFF3 file looks like, briefly:

```
##gff-version 3.2.1
##sequence-region   ctg123 1 1497228
ctg123 . gene            1000  9000   .  +  .  ID=gene00001;Name=EDEN

ctg123 . mRNA            1050  9000   .  +  .  ID=mRNA00001;Parent=gene00001;Name=EDEN.1

ctg123 . exon            1201  1500   .  +  .  ID=exon00002;Parent=mRNA00001
ctg123 . exon            3000  3902   .  +  .  ID=exon00003;Parent=mRNA00001
ctg123 . exon            5000  5500   .  +  .  ID=exon00004;Parent=mRNA00001
ctg123 . exon            7000  9000   .  +  .  ID=exon00005;Parent=mRNA00001

ctg123 . CDS             1201  1500   .  +  0  ID=cds00001;Parent=mRNA00001;Name=edenprotein.1
ctg123 . CDS             3000  3902   .  +  0  ID=cds00001;Parent=mRNA00001;Name=edenprotein.1
```

```
ctg123 . CDS            5000  5500   .  +  0  ID=cds00001;Parent=mRNA00001;Name=edenprotein.1
ctg123 . CDS            7000  7600   .  +  0  ID=cds00001;Parent=mRNA00001;Name=edenprotein.1
```

And the visual representation of the text



At the top level we see a "gene" (3rd column), which spans from 1000 to 9000, on the forward strand (7th column), with an ID of `gene00001` and a Name of `EDEN`.

Below the gene, is an mRNA feature. We can infer that it is "below" in the hierarchy based on the last column which has a *Parent* of `gene00001`. Similarly all four exons and all four CDSs have a Parent of `mRNA00001`. `ID`, `Name`, and `Parent` are all known as *feature attributes*. Metadata about a feature. However, more information than just the names, IDs, and relationships goes into feature attributes. Often you will see Notes, sometimes Products, and many other tags besides. Only a couple of these attributes have standards defining what information they contain, the rest are free to be used as your organisation specifies, or as you like.

All of this is a little bit excessive for phages where real introns are rare, and mRNAs not involved, but nevertheless, we want to make sure our data is accessible to other researchers so they can do experiments building on our work.

(It is more important that you know the format exists, and that it encodes parent-child biological relationships, than that you know the precise specifics of what each column means.)

GenBank

In stark contrast to the elegance of the GFF3 format (tab separated, key-value pairs, easy to work with), we have the older GenBank format. This is a fixed-width format which has a "flat" gene model, and lacks any way to represent the hierarchical relationships that are biologically relevant.

```
LOCUS       NC_001133             230218 bp    DNA     linear   PLN 14-JUL-2011
DEFINITION  Saccharomyces cerevisiae S288c chromosome I, complete sequence.
ACCESSION   NC_001133
VERSION     NC_001133.9  GI:330443391
DBLINK      Project: 128
KEYWORDS    .
SOURCE      Saccharomyces cerevisiae S288c
  ORGANISM  Saccharomyces cerevisiae S288c
            Eukaryota; Fungi; Dikarya; Ascomycota; Saccharomycotina;
            Saccharomycetes; Saccharomycetales; Saccharomycetaceae;
            Saccharomyces.
REFERENCE   1  (bases 1 to 230218)
  AUTHORS   Goffeau,A., Barrell,B.G., Bussey,H., Davis,R.W., Dujon,B.,
            Feldmann,H., Galibert,F., Hoheisel,J.D., Jacq,C., Johnston,M.,
            Louis,E.J., Mewes,H.W., Murakami,Y., Philippsen,P., Tettelin,H. and
            Oliver,S.G.
  TITLE     Life with 6000 genes
  JOURNAL   Science 274 (5287), 546 (1996)
   PUBMED   8849441
FEATURES             Location/Qualifiers
     source          1..230218
                     /organism="Saccharomyces cerevisiae S288c"
                     /mol_type="genomic DNA"
                     /strain="S288c"
                     /db_xref="taxon:559292"
                     /chromosome="I"
     gene            complement(1807..2169)
                     /gene="PAU8"
                     /locus_tag="YAL068C"
                     /db_xref="GeneID:851229"
     mRNA            complement(<1807..>2169)
                     /gene="PAU8"
                     /locus_tag="YAL068C"
                     /transcript_id="NM_001180043.1"
                     /db_xref="GI:296142466"
                     /db_xref="GeneID:851229"
     CDS             complement(1807..2169)
                     /gene="PAU8"
                     /locus_tag="YAL068C"
                     /note="hypothetical protein, member of the seripauperin
                     multigene family encoded mainly in subtelomeric regions"
                     /codon_start=1
                     /protein_id="NP_009332.1"
                     /db_xref="GI:6319249"
                     /db_xref="SGD:S000002142"
                     /db_xref="GeneID:851229"
...
ORIGIN
        1 ccacaccaca cccacacacc cacacaccac accacacacc acaccacacc cacacacaca
       61 catcctaaca ctaccctaac acagccctaa tctaaccctg gccaacctgt ctctcaactt
```

There are a few major regions of a GenBank file:

1. The header (Starting with LOCUS...)
2. The feature table (Starting with FEATURES)
3. The sequence

The *header* will tell you information like:

- Sequence ID, NC_001133 in the above example,
- Genome or chromosome length
- Annotation set version (9, from `VERSION NC_001133.9`)
- References

The *feature table* usually starts with a "source" type feature which contains metadata about the chromosome or genome. Features consist of a feature type key on the left, and key value pairs on the right formatted as `/key="Value..."`.

Lastly, there is the sequence data. In contrast to GFF3 which stores sequence data in standardised fasta format, GenBank uses sequence separated into six columns of ten characters, with the sequence index annotated on the left.

2.2 Annotation

On to actually using Apollo! We'll go through an example annotation.You're welcome to follow along with this at home and familiarize yourself with Apollo before class. The example presented here will be open for everyone in the class to use, so images may not reflect the current annotations made.

There are two primary components to an annotation pipeline:

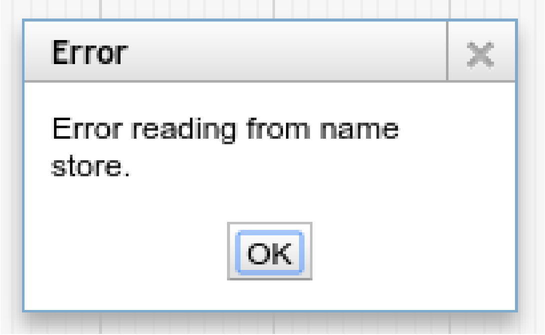
1. Structural annotation
2. Functional annotation

In structural annotation you will likely take the output of gene callers, and perhaps other evidence tracks, and use these results to annotate putative genes in Apollo. Structural annotations consist of locations of genomic features, like genes, terminators, and tRNAs.

Functional annotation will entail identifying possible gene functions based on different evidence sources. We will go into more detail in the first lecture on what it means to do structural and functional annotations.

2.2.1 Apollo in Galaxy

This section will cover the generalised use of Apollo in Galaxy, not specific to any annotation workflow implementation.



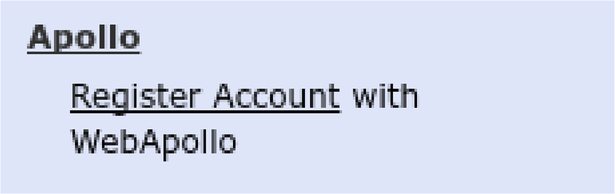
Error

Error reading from name store.

Fig. 2.4: This error might appear, from time to time. It is safe to ignore.

Registration

In order to log in to Apollo, you'll need to register for an account using the Galaxy tool, if your site has not already set up one for you.

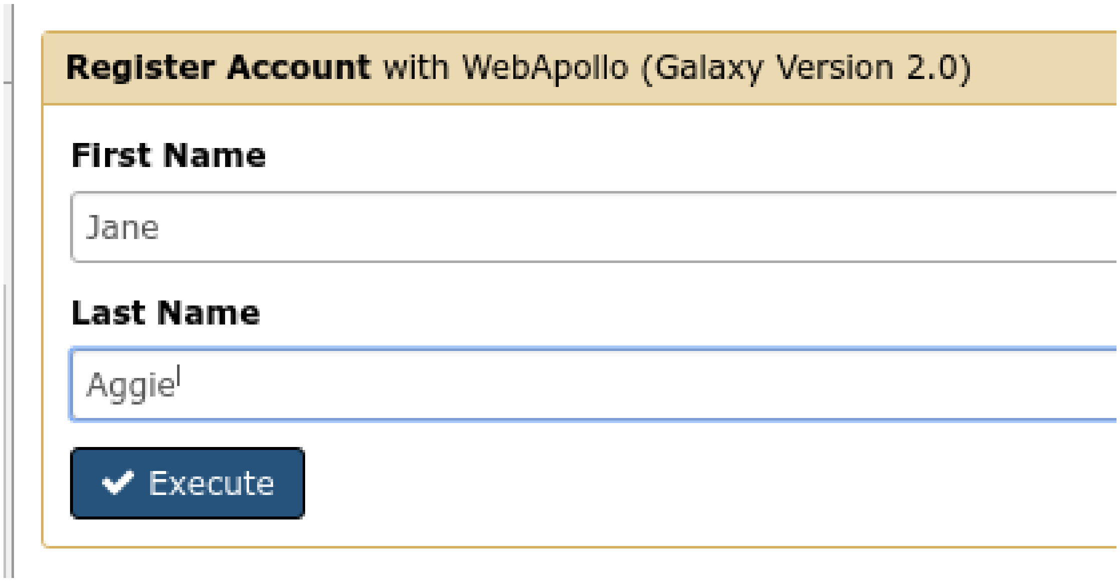


Apollo

Register Account with WebApollo

In the integrated Galaxy-Apollo workflow, you can register for an account by running a Galaxy tool, which will generate your credentials for you. If you ever forget your credentials and cannot find the item in your history, you can re-run this, and it will generate a new password for you.

Simply fill out the form:



Register Account with WebApollo (Galaxy Version 2.0)

First Name

Jane

Last Name

Aggie

And hit the **Execute** button. Once the tool is done running, the dataset will turn green. You will then click the "View Dataset" eyeball button to see your password. (You don't need to memorize this password or write it down anywhere. You can always come back to Galaxy to view it.)

JBrowse In Galaxy

If you're familiar with JBrowse, a view of Apollo should look familiar to you:

The CPT developed a tool called JBrowse-in-Galaxy (JiG) which allows you to build JBrowse instances within Galaxy. JBrowse instances are traditionally configured through a complex and manual process at the command line. JiG represents the first ever visual JBrowse configuration and construction tool.

Apollo takes, as its input, complete JBrowse instances. To view any data in Apollo, a JBrowse instance needs to be configured first.

Once you've created a JBrowse instance, you'll find it in your history

If you chose to produce a "standalone instance," you'll be able to click the eyeball icon and view the dataset.

Moving Data from Galaxy to Apollo

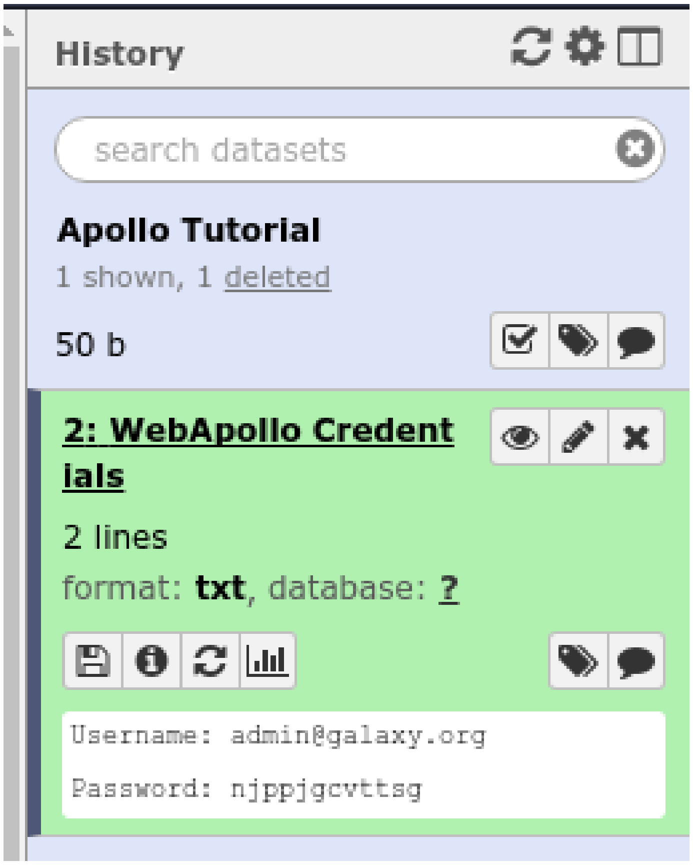
Now that you have:

1. A complete JBrowse instance
2. Apollo credentials

You're ready to start talking to the Apollo service.

The first tool we'll use is a tool named **Create or Update** which lets us create, or update, an organism in Apollo with new data from Galaxy in the form of a JBrowse instance.

This step will transfer data to Apollo, and produce a JSON file. The output JSON file contains some metadata about the organism. You will never need any information from this file.



```
Username: admin@galaxy.org
Password: njppjgcvttsg
```

Fig. 2.5: If you ever lose your Galaxy history with the password, just come back to Galaxy and re-run the tool. The password will be saved there for you.

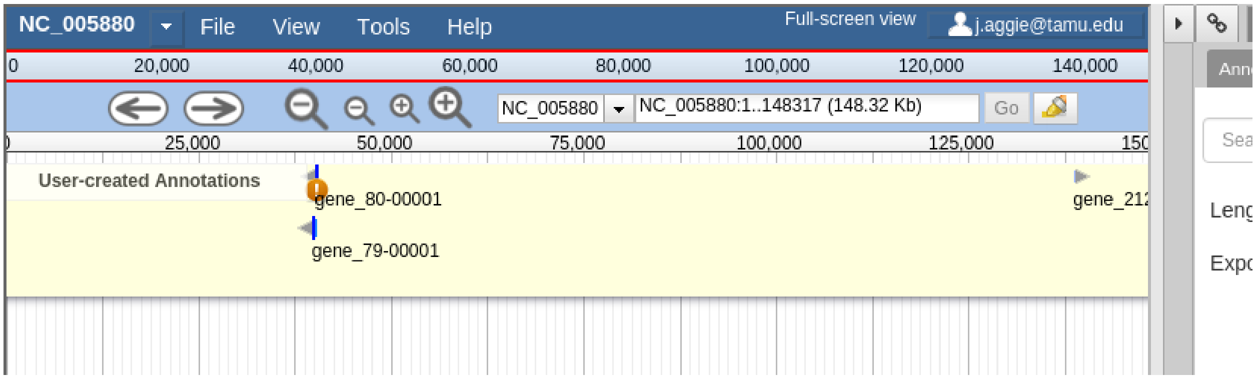


Fig. 2.6: Notice the JBrowse window embedded within the Apollo interface. Apollo integrates with the JBrowse software to provide the ability to make annotations and save them.

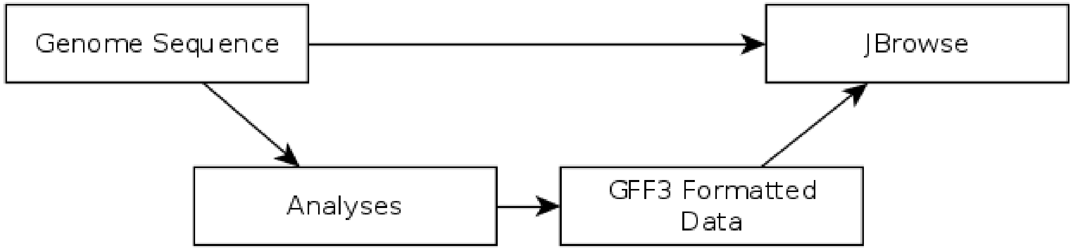
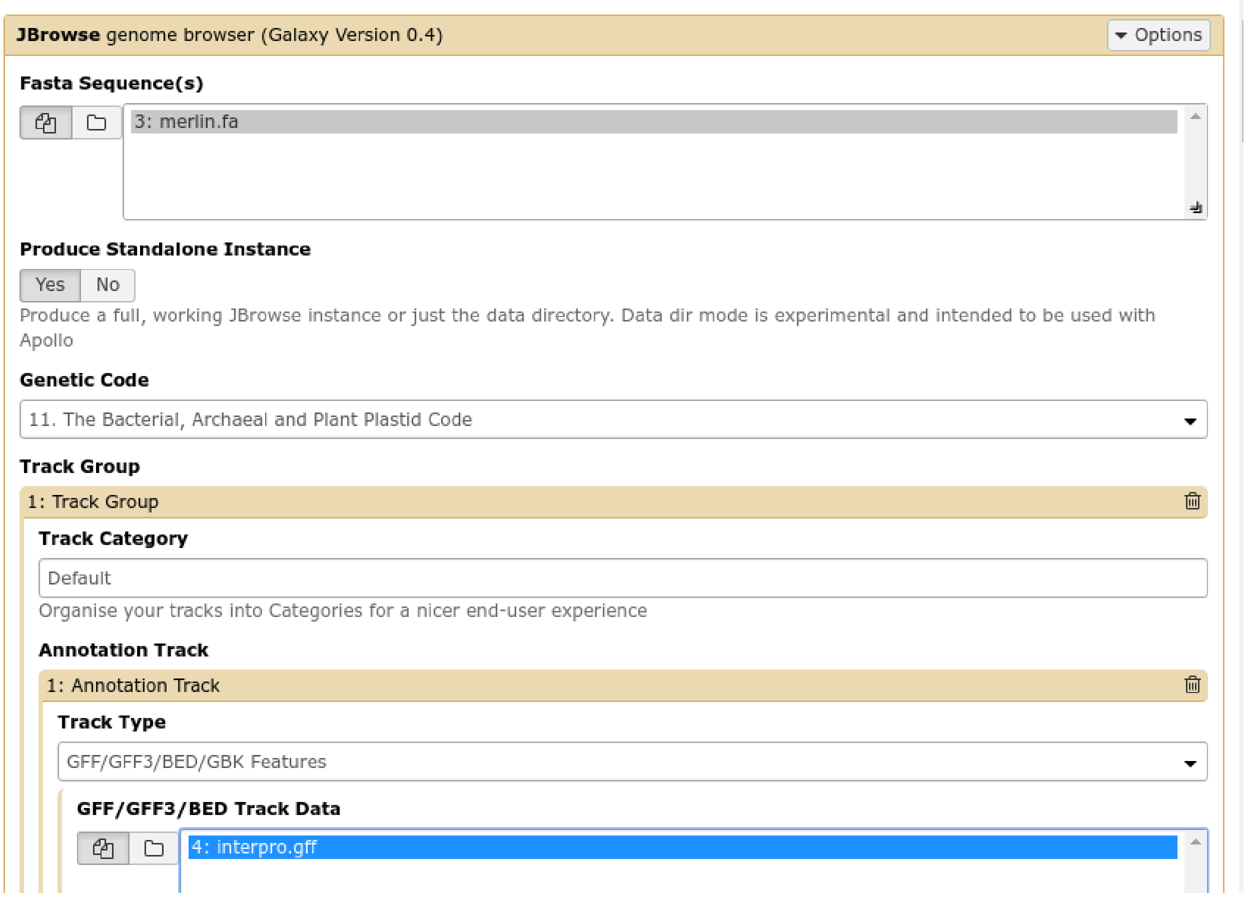


Fig. 2.7: The generalized JBrowse workflow. JBrowse is simply a tool for displaying the results of a bioinformatic analysis in a standardised way.



JBrowse genome browser (Galaxy Version 0.4)

Fasta Sequence(s)

3: merlin.fa

Produce Standalone Instance

Yes No

Produce a full, working JBrowse instance or just the data directory. Data dir mode is experimental and intended to be used with Apollo

Genetic Code

11. The Bacterial, Archaeal and Plant Plastid Code

Track Group

1: Track Group

Track Category

Default

Organise your tracks into Categories for a nicer end-user experience

Annotation Track

1: Annotation Track

Track Type

GFF/GFF3/BED/GBK Features

GFF/GFF3/BED Track Data

4: interpro.gff

Fig. 2.8: The JBrowse-in-Galaxy tool is an extremely complex tool, with a very detailed manual (at the bottom of the page in Galaxy). If you need to do anything beyond showing simple GFF3 files, you'll need to read this manual. If you just wish to display the genome and associated datasets in Apollo, you do not need to produce a "standalone instance." That is only required if you wish to view the (static) JBrowse instance in Galaxy.

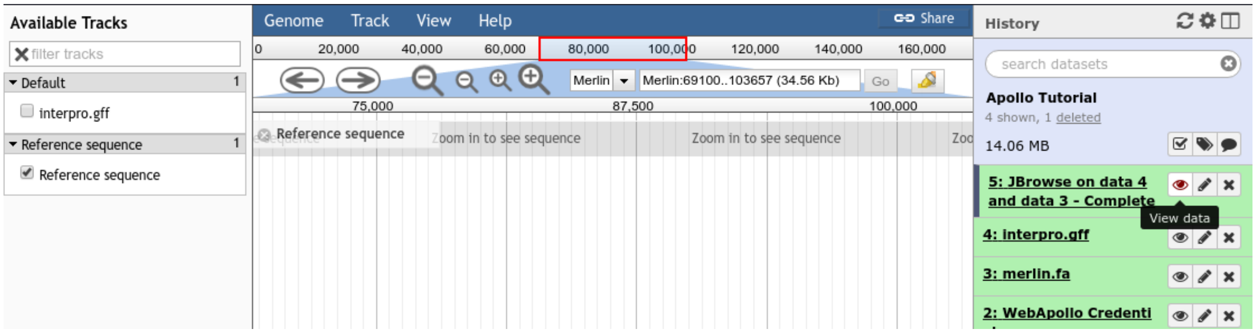


Fig. 2.9: Viewing a JBrowse instance produced within Galaxy.

Now that your data is available in Apollo, you can access it at Apollo, or via the **Annotate** convenience method which is provided. The Annotate tool takes the JSON file from a *Create or Update* step, and loads Apollo, directly in Galaxy.

2.2.2 Finding Our Way Around

You'll be presented with a two-pane display. On the left is an embedded JBrowse instance:

JBrowse, embedded in Apollo, is slightly different than a normal JBrowse. The movement controls are all the same:

- you can use the magnifying glasses to zoom in and out of the genome and its data
- the arrow icons will move you up and downstream along the genome
- Selecting or clicking on locations along the genome ruler (they grey box at the top of the genome, 0 bp; 20,000bp; 40,000bp; etc.) will allow you to zoom in and move to specific regions

The menu bar has some useful options, some that aren't available in "standard" JBrowse:

- **File** allows adding some special track types. We will not be using these options, but it's recommended that you explore them.
- **View** will let you set some useful options:
  - "Color by CDS frame" is a popular option during annotation. It will colour each coding sequence by which frame the reading frame is in.
  - "Show Track Label" is an incredibly useful feature to hide the track's labelling, allowing you to annotate small features near the end of the genome, which would otherwise be hidden by the track label (E.g. "User created annotations")

The **pale yellow** track that is visible is the **User Created Annotation** track. During the annotation of a genome, gene features will be added to this track and edited, thus this track will always be visible to you.

Back to the overview ,on the right is the **Genome Selector**, which lists all of the organisms accessible to you.

The **Ref Sequence** tab lists all of the sequences (associated with a given organism) that are accessible to you.

For those familiar with JBrowse, you will notice that the track selection menu is missing. You will find it under the **Tracks** tab on the right hand side.

If you select all three of the tracks (*GeneMarkS*, *MetaGeneAnnotator*, and *Glimmer3*), they will show up in JBrowse. You may find that this produces an absolutely overwhelming amount of information:

In order to combat that, you should zoom in

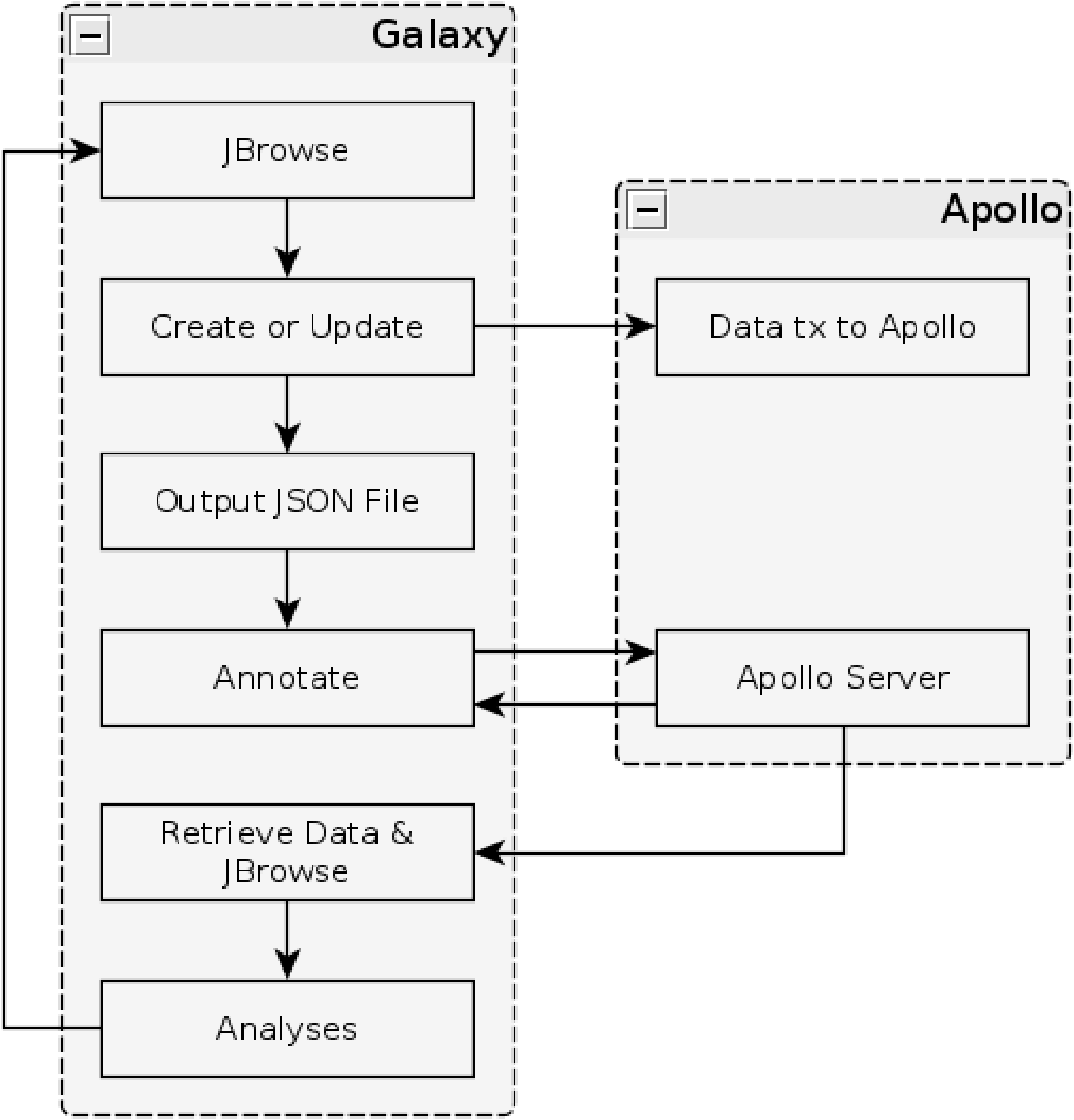
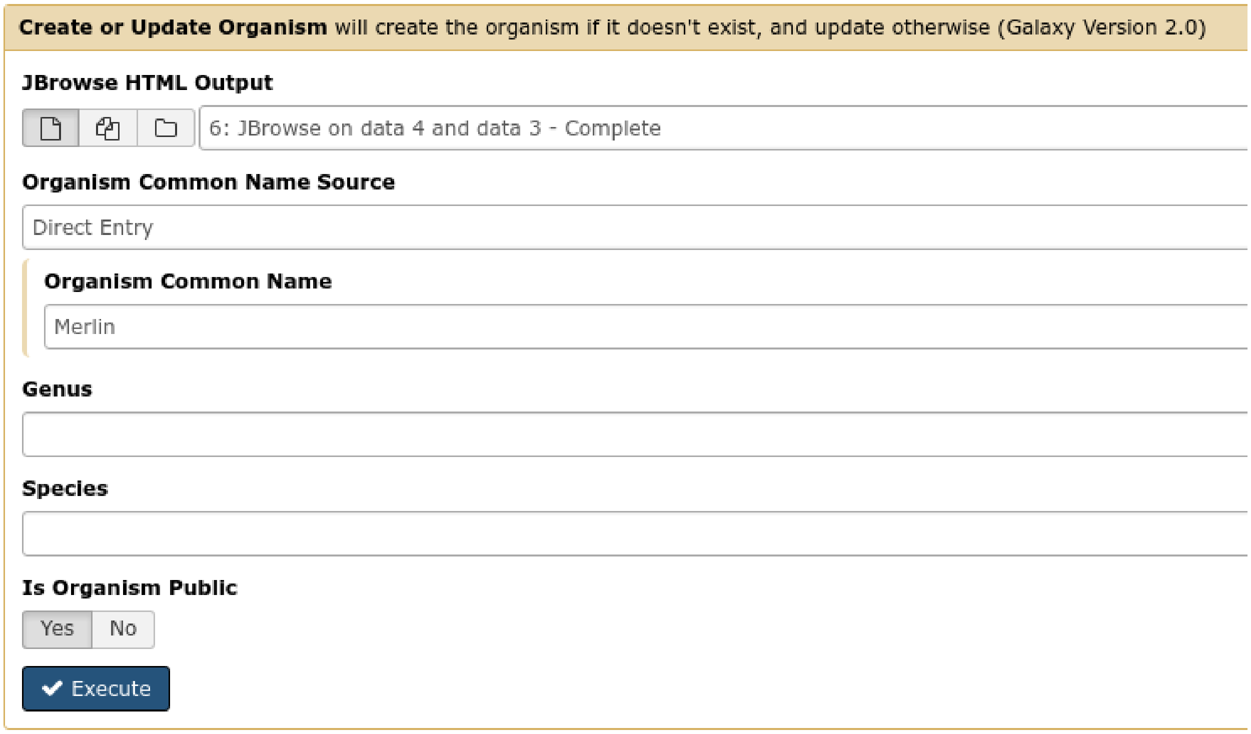


Fig. 2.10: The general Apollo/JiG/Galaxy workflow. Data is built up in Galaxy in the form of a JBrowse instance, which is pushed to the Apollo service in the Create or Update step, and transfers data to Apollo. The Annotate step is simple a convenience method for accessing Apollo. Apollo is also available at https://your-organisations-galaxy-instance/apollo. These methods both point at the same instance of Apollo.



Create or Update Organism will create the organism if it doesn't exist, and update otherwise (Galaxy Version 2.0)

JBrowse HTML Output

6: JBrowse on data 4 and data 3 - Complete

Organism Common Name Source

Direct Entry

Organism Common Name

Merlin

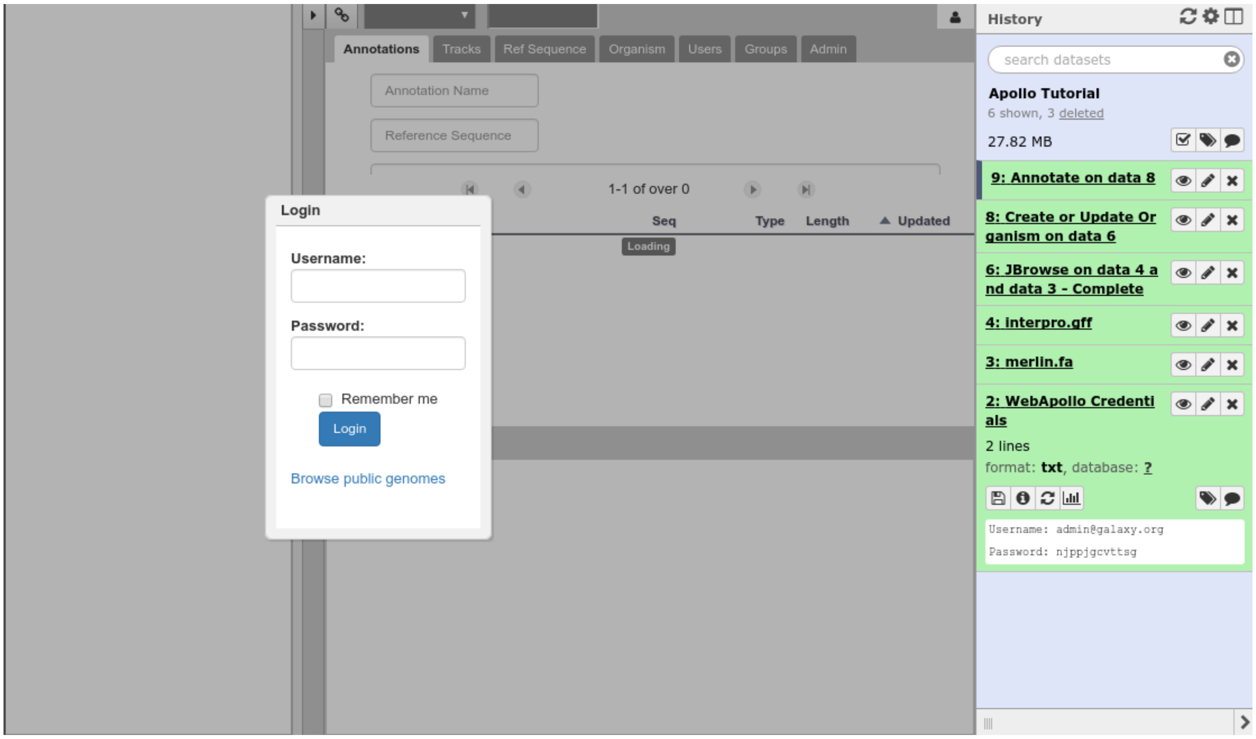
Genus

Species

Is Organism Public

Yes No

Fig. 2.11: It is not required (but highly recommended) to fill out the species field appropriately. Additionally it is not required to make anything public (available to the public at large) but it is encouraged.



Apollo Tutorial
6 shown, 3 deleted
27.82 MB

9: Annotate on data 8
8: Create or Update Organism on data 6
6: JBrowse on data 4 and data 3 - Complete
4: interpro.gff
3: merlin.fa
2: WebApollo Credentials
2 lines
format: **txt**, database: ?

```
Username: admin@galaxy.org
Password: njppjgcvttsg
```

Fig. 2.12: Apollo accessed from within Galaxy

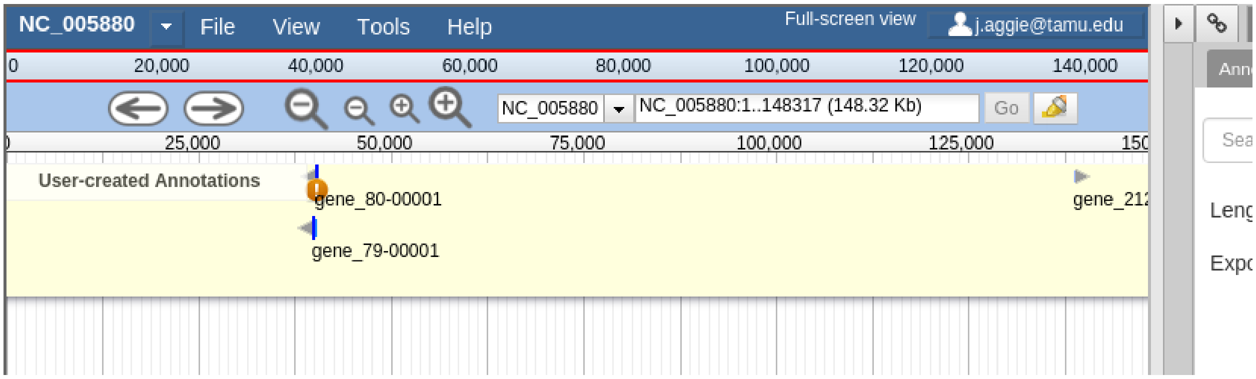


Fig. 2.13: JBrowse is a key component of Apollo. Apollo adds some additional options to JBrowse's top menu, and the pale yellow track labeled "User-created Annotations"

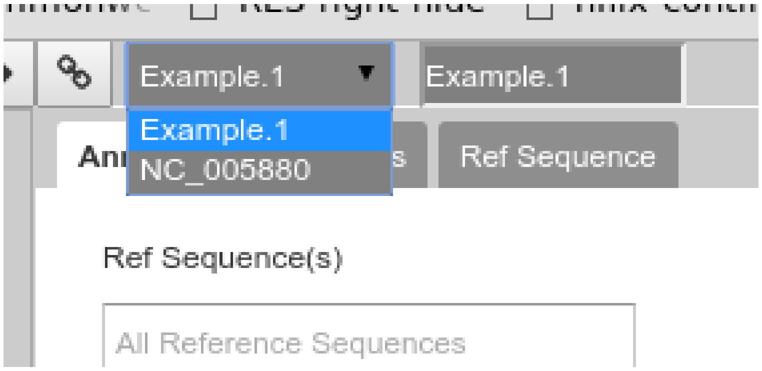
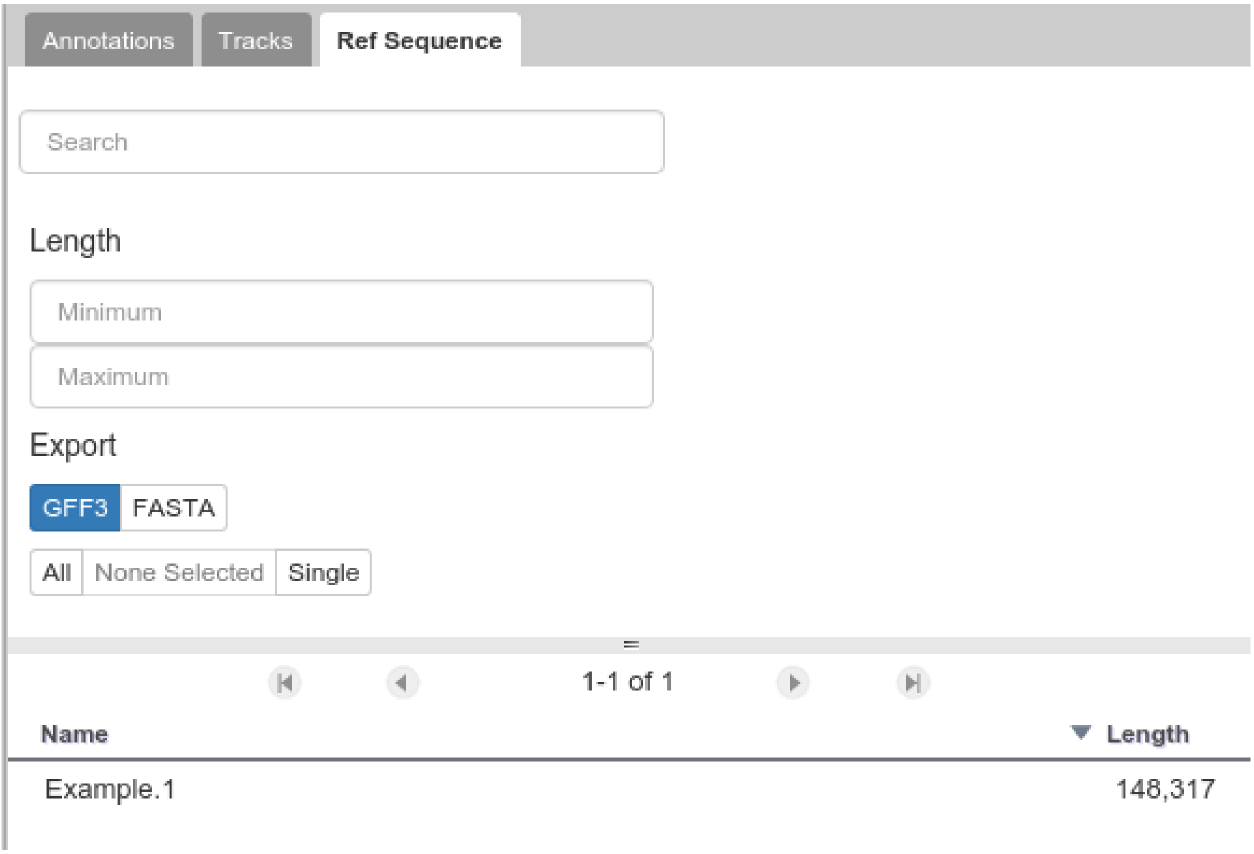


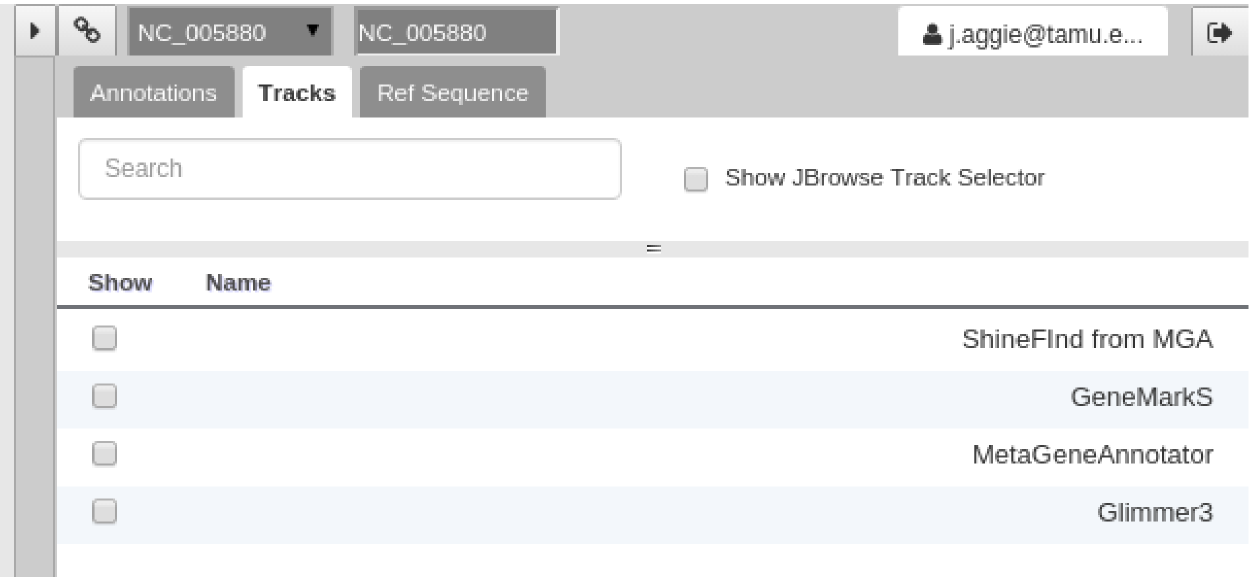
Fig. 2.14: Apollo uses the concept of "Organisms" with "reference sequences" below it. Each organism can have one or more reference sequences. In higher order organisms those often correspond to multiple chromosomes. For phage uses they are most often used to correspond to different assemblies of the genome.



Name	▼ Length
Example.1	148,317

Fig. 2.15: This panel allows you to switch between reference sequences and filter them (in the event that there are many reference sequences).

Double clicking on the name will cause that sequence to load in the JBrowse window on the left.



Show	Name
□	ShineFInd from MGA
□	GeneMarkS
□	MetaGeneAnnotator
□	Glimmer3

Fig. 2.16: Track Selection

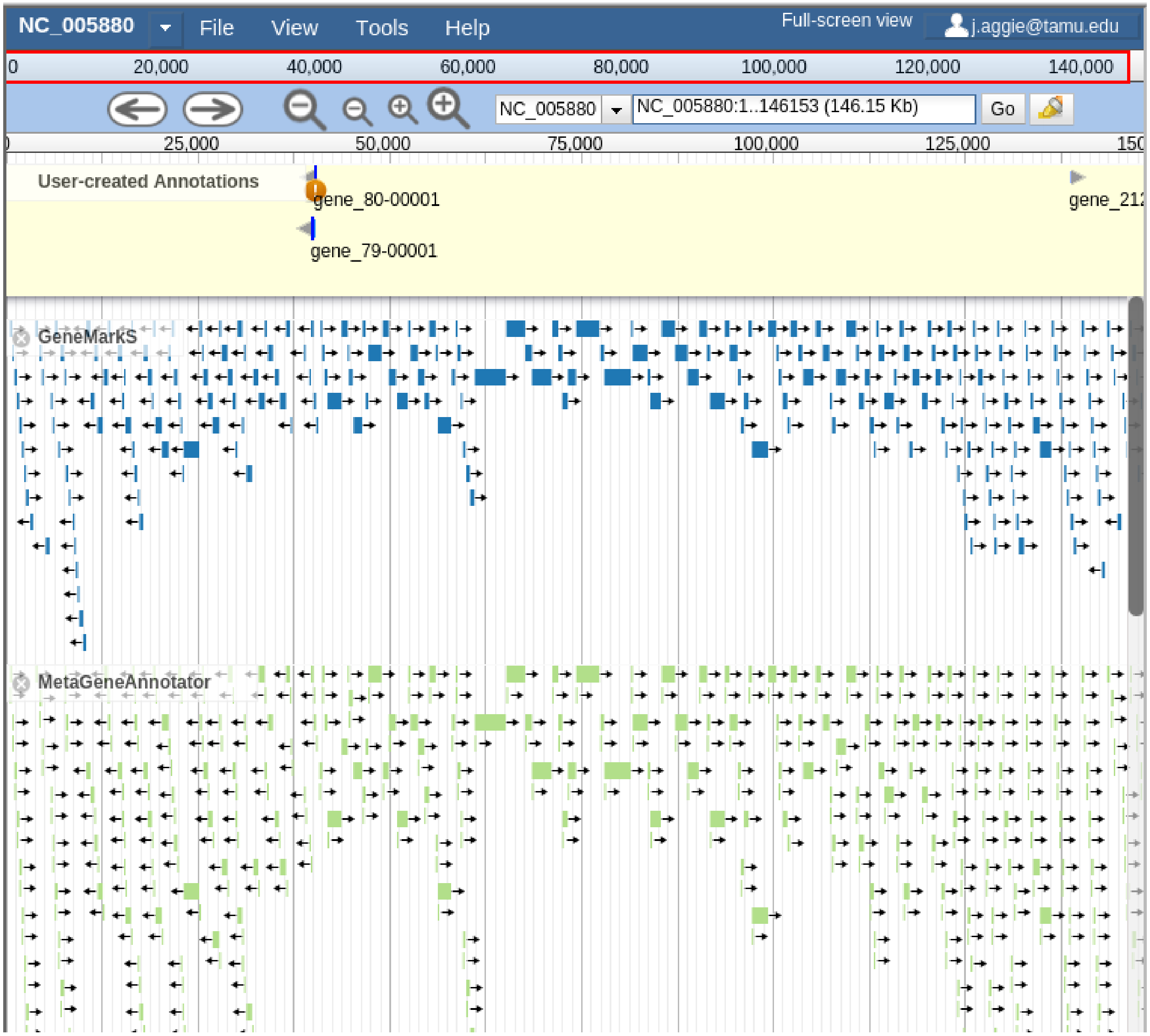


Fig. 2.17: Overwhelming

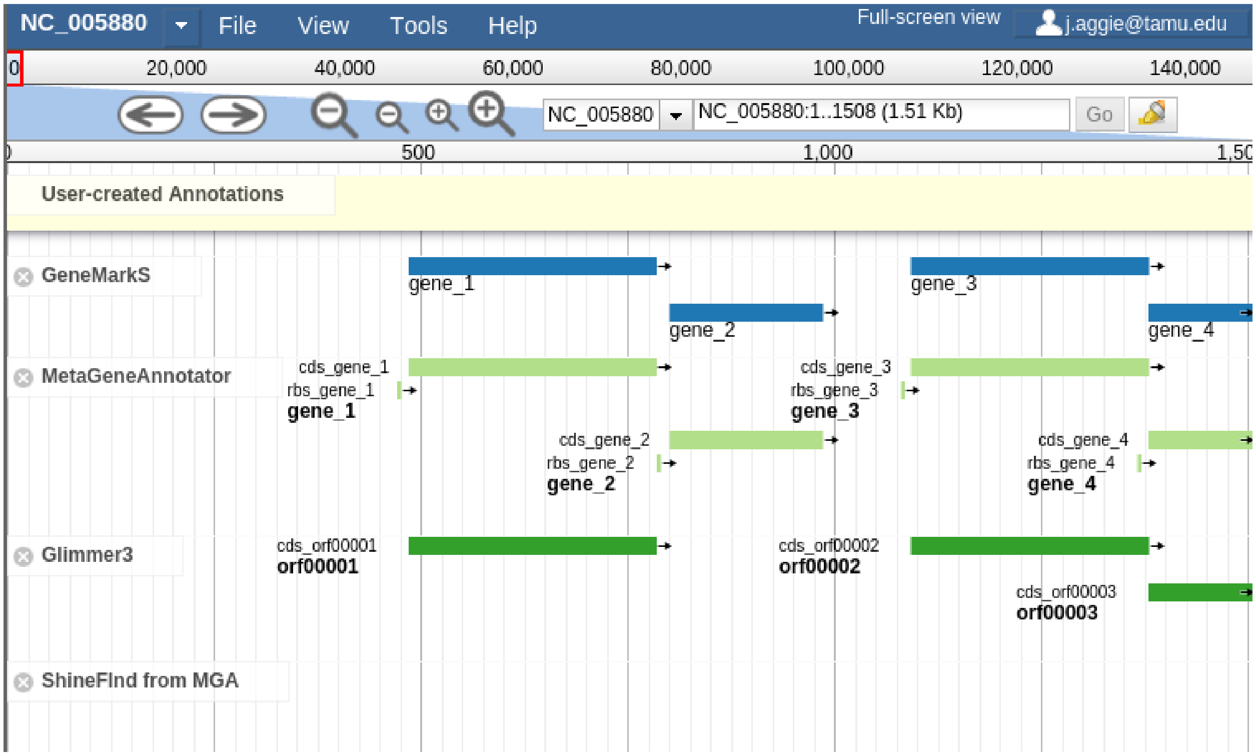


Fig. 2.18: Zooming

You may find that you wish to focus solely on the annotation process, without any distractions from the Apollo portion of the interface. You can hide that easily.

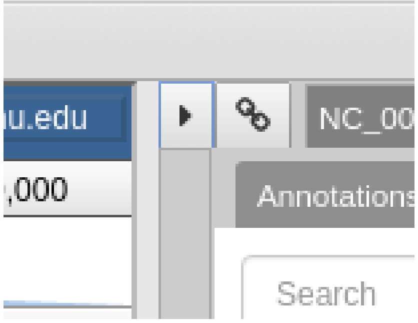


Fig. 2.19: Hiding Apollo

Let's zoom down to the level of a single gene:

Great! Here we see the very first gene called by the three *gene callers* that we use.

Note: Your work is saved automatically, instantaneously. You do not need to worry about losing changes.

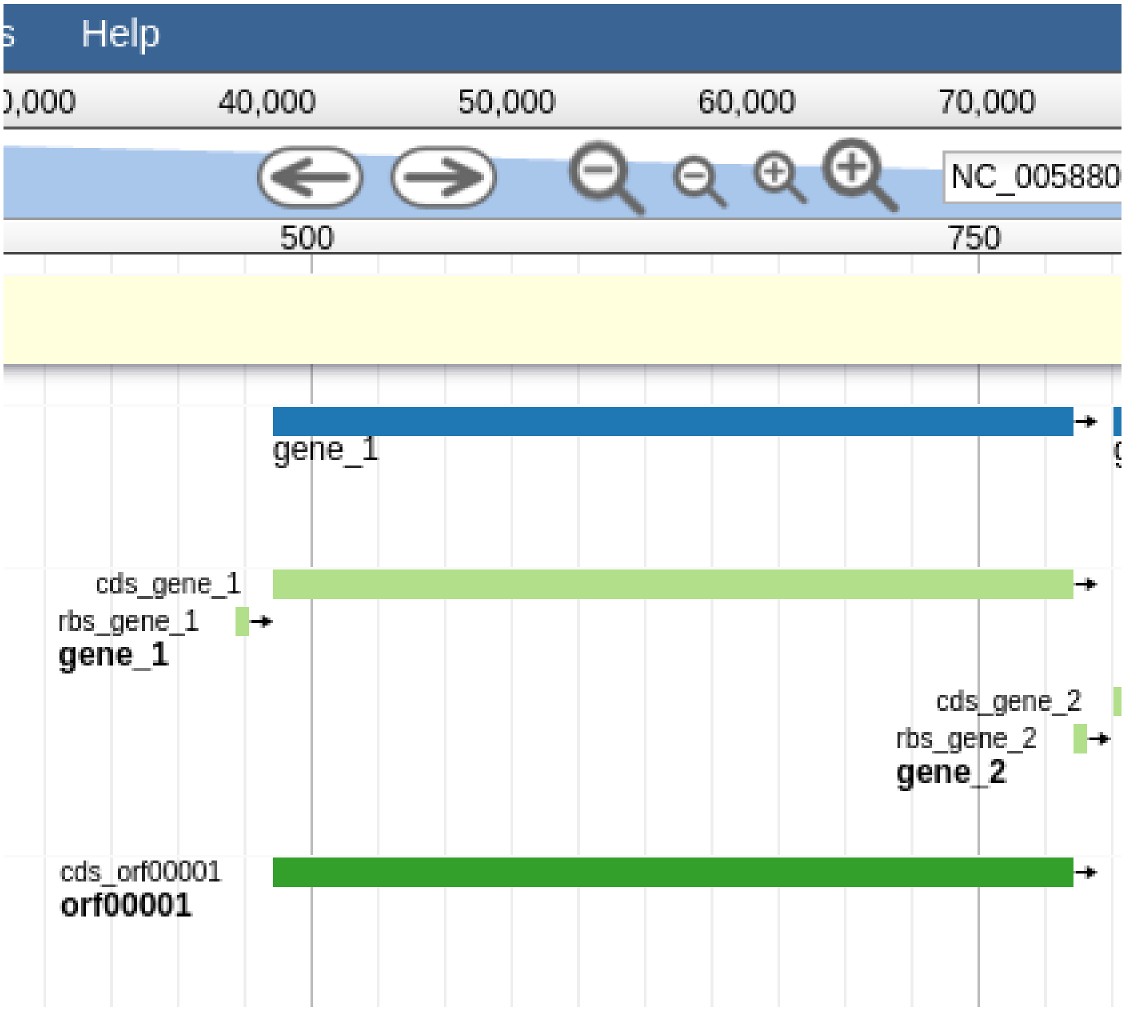


Fig. 2.20: Here we can begin to compare the gene models of these three genes. One of the three has a Shine Dalgarno sequence anotated. The CPT filters all SD sequences to ensure that only high quality ones are visible.