



CSB1021HF LEC0131

FUNDAMENTALS OF GENOMIC DATA SCIENCE

0.0.0 Module 3: Reference alignment and RNA-Seq on Galaxy

0.1.0 About Fundamentals of Genomic Data Science

Fundamentals of Genomic Data Science is brought to you by the **Centre for the Analysis of Genome Evolution & Function (CAGEF)** bioinformatics training initiative. This course was developed based on feedback on the needs and interests of the Department of Cell & Systems Biology and the Department of Ecology and Evolutionary Biology.

The structure of this course is a “code-along”, hands-on style! A few hours prior to each lecture, materials will be made available for download at QUERCUS (<https://q.utoronto.ca/>). The teaching materials will consist of a weekly PDF that you can use to follow along with the instructor along with any datasets that you’ll need to complete the module. This learning approach will allow you to spend the time coding and not taking notes!

As we go along, there will be some in-class challenge questions for you to solve. Post lecture assessments will also be available for each module, building upon the concepts learned in class (see syllabus for grading scheme and percentages of the final mark).

0.1.1 Where is this course going?

We’ll take a blank slate approach here to learning genomic data science and assume you know nothing about programming or working directly with next generation sequencing data. From the beginning of this course to the end we want to guide you from potential scenarios like:

- You don’t know what to do with a set of raw sequencing files fresh from a facility like.

- You've been handed a legacy pipeline to analyse your data or maintain for the lab but you don't know what it runs or how.
- You plan on generating high-throughput data but there are no bioinformaticians around to help you out.

and get you to the point where you can:

- Recognize the basic tools in sequence analysis.
- Plan and write your own data analysis pipelines.
- Explain your data analysis methods to labmates, supervisors, and other colleagues.

0.1.2 How do we get there?

In the first half of this course, we'll focus on how to generate analysis pipelines using the Galaxy platform – a user-friendly graphical interface that provides access to common sequence analysis tools. After we are comfortable with these tools, we'll look at life through the lens of a command-line interface. It is here that we will learn the basics of file manipulation and how to program scripts that can carry out multiple tasks for us. From there we'll revisit tools from the first half and learn skills to make your data analysis life easier.

0.2.0 Goals of the module

1. Learn how to perform and analyze reference alignments with Galaxy.
2. Learn how to perform variant calling with Galaxy.
3. Learn how to analyze RNA-Seq datasets with Galaxy.
4. Visualize reference variant calls and RNA-Seq analyses with the Integrated Genome Viewer.

0.3.0 Pre-class modules with Coursera

Each week we strongly encourage you to complete the assigned Coursera modules and/or readings **before** class. These are meant to provide you with sufficient background material on each week's module so that we can focus on the act of "doing" something with that data rather than spend a lot of time on the origins of it. You'll find a section outlining the next set of Coursera modules and readings at the end of each module.

0.3.1 Go to www.coursera.org and sign up for an account with your e-mail.

0.3.2 Search the following courses and enroll to audit each course (audit):

- Genomic Data Science with Galaxy, Johns Hopkins University.
- Command Line Tools for Genomic Data Science, Johns Hopkins University.

0.4.0 Setting up your working directory

We suggest that you create a new directory (folder) for this course directly off your root directory called **"FGDS"**. Working from your root directory is not necessary, but it will make some of the aspects of the course a little easier to manage. For MacOS users, we suggest you create this as a subfolder in your **user** directory.

- 0.4.1 Within this directory, create another directory called **"Module3"**. This is where we will store the data used in this week's module.
- 0.4.2 Create a subdirectory called **"downloads"** to store the initial files as we download them before decompressing and working with them in later steps.

1.0.0 Reference alignments in Galaxy

1.0.1 Log into the class Galaxy instance

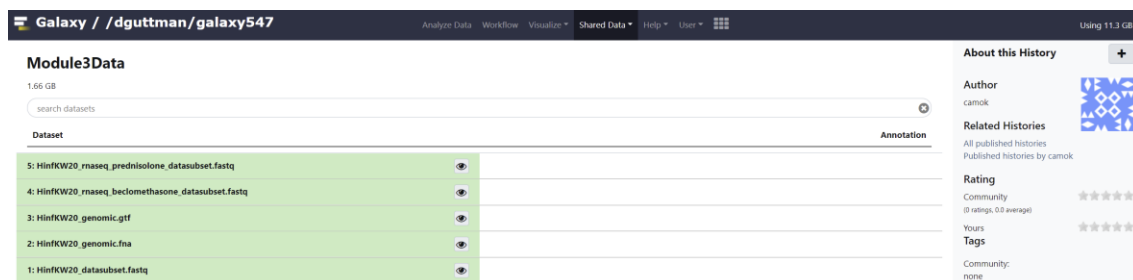
Recall from last week that we are working with a special instance of the Galaxy server. You can use your login credentials here at <https://galaxy-547-p38.p.genap.ca>

1.1.0 Add fastq files from shared history

Normally you would upload the files you're interested in working on yourself but for today's reference alignment we are interested in working with some shared files on the server:

- **HinfKW20_datasubset.fastq**: our raw fastq read data subset from last week's lecture
- **HinfKW20_genomic.fna**: a genomic reference nucleotide fasta file for *H. influenza*

1.1.1 Load shared data from the menu > Shared Data > Histories > [redacted] This shared history will contain the data from the above two data sets.

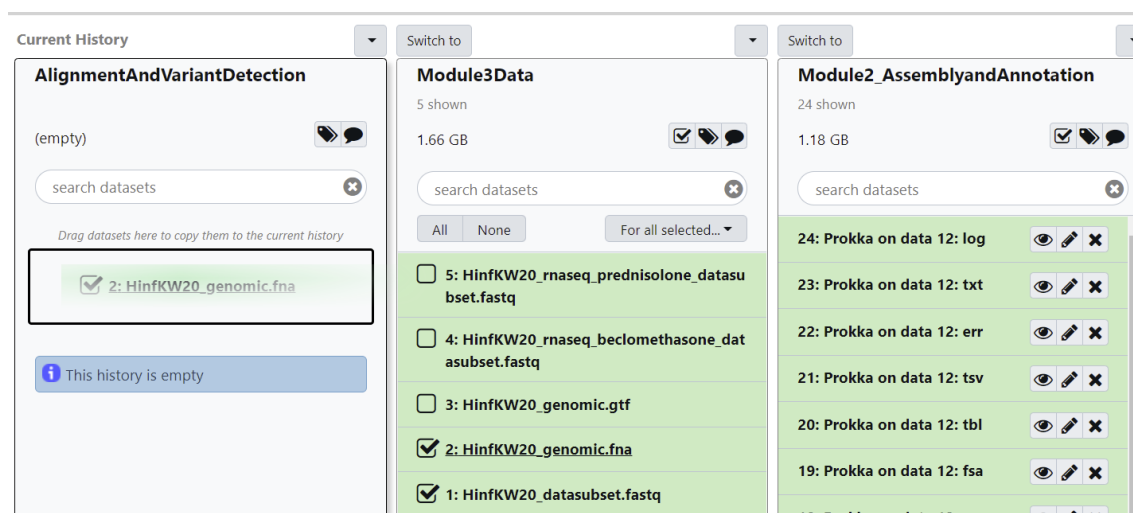


1.1.2 Import the history with the + icon in the About this History pane on the right-hand side of the interface. Name this imported history [redacted]

1.1.3 Go to History pane > Create new history and name it **AlignmentAndVariantDetection**.

1.1.4 Go to History pane > View all histories and on the [redacted] history use the Operations on multiple datasets icon (checkbox).

1.1.5 Select the two datasets outlined above from the [redacted] history and drag them to **AlignmentAndVariantDetection**.



- 1.1.6 Verify that the datatype for **HinfKW20_genomic.fna** is fasta using the [Edit attributes](#) icon (pencil) for the dataset and checking the [Datatypes](#) tab.

Upload your own data: Recall that you could also upload your own version of the **HinfKW20_genomic.fna** dataset from your Module1 data folder using the [tools pane > Get Data > Upload File from your computer](#). You could also retrieve the **HinfKW20_datasubset.fastq** file from your **Assembly and Annotation** history created in Module2.

1.2.0 Aligning reads to a reference genome

Now that we have our datasets prepared, we want to go about the practice of aligning our raw fastq reads to the reference genome. Recall from last week that the [FastQC](#) results of our raw data were decent. Although we took the time to use [Trimmomatic](#) on our dataset, to save on time, we will use the original subset directly for our alignment. For your own workflows, however, you may wish to add a quality filtering step as well.

To map reads to a reference genome, there are a few tools available to us including the Burrows-Wheeler Aligner ([BWA](#), circa 2010) and [Bowtie2](#) (Langmead and Salzberg, Nat Methods 2012) which is part of the “Tuxedo suite” of tools used in RNAseq analysis. Today, for our purposes, we’ll be working with [Bowtie2](#) which has better overall metrics than [BWA](#).

- 1.2.1 Go to the [Bowtie2](#) tool.
- 1.2.2 Read the overview for [Bowtie2](#). You can revisit the remainder of the [Bowtie2](#) options later.
- 1.2.3 Review some of the settings that you can tinker with by changing [Select Analysis Mode](#) to **Full Parameter List**.

To alter the input options, you will have to switch the dropdown [Do you want to tweak input options?](#) to **Yes**. Most options should be kept as default unless you are an advanced user and have reason to believe modifying an option would improve your alignment. Returning to the **Default setting only** option there are different pre-sets available for beginners.

- 1.2.4 Execute a default end-to-end alignment in Bowtie2 after adjusting the following settings:

- Is this single or paired library: **Single-end**
- FASTA/Q file: **HinfKW20_datasubset.fastq**
- Will you select a reference genome from your history or use a built-in index? **Use a genome from the history and build index**
 - Select reference genome: [HinfKW20_genomic.fna](#)

Bowtie2 - map reads against reference genome (Galaxy Version 2.3.4.2)

☆ Favorite ▼ Options

Is this single or paired library

Single-end

FASTA/Q file

1: HinfKW20_datasubset.fastq

Must be of datatype "fastqsanger" or "fasta"

Write unaligned reads (in fastq format) to separate file(s)

Yes No

--un/-un-conc (possibly with -gz or -bz2); This triggers --un parameter for single reads and --un-conc for paired reads

Write aligned reads (in fastq format) to separate file(s)

Yes No

--al/-al-conc (possibly with -gz or -bz2); This triggers --al parameter for single reads and --al-conc for paired reads

Will you select a reference genome from your history or use a built-in index?

Use a genome from the history and build index

Built-ins were indexed using default options. See "Indexes" section of help below

History

search datasets

AlignmentAndVariantDetection

2 shown

370.97 MB

2: HinfKW20_genomic.fna

1: HinfKW20_datasubset.fastq

1.3.0 Review the alignment stats

Time to take a closer look at the output results and get a sense of how well the reads were aligned to the reference genome.

1.3.1 Go to the *History pane > Bowtie 2 on data 2 and data 1: aligned reads >*

Bowtie2
Dataset Information

Number: 3
Name: Bowtie2 on data 2 and data 1: aligned reads (BAM)
Created: Fri Nov 12 19:03:08 2021 (UTC)
Filesize: 68.1 MB
Dbkey: ?
Format: bam

Job Information

Galaxy Tool ID: toolshed.g2.bx.psu.edu/repos/devteam/bowtie2/bowtie2/2.3.4.2
Galaxy Tool Version: 2.3.4.2

Tool Version: /cvmfs/soft.galaxy.v2.1/tool-dependency/_conda/envs/mulled-v1-7576289d51ff5aefa21c8a2af901e1a61eb245f4e5fd66ae894d120dd35ff8f6/bin/bowtie2-align-s version 2.3.4.3 64-bit
Built on default-f5c54195-524f-48f5-8c4c-9858b1b33b24 Sun Feb 3 20:25:11 UTC 2019 Compiler: gcc version 7.3.0 (cross-tool-NG 1.23.0.449-a04d0) Options: -O3 -m64 -msse2 -funroll-loops -g3 -fvisibility-inlines-hidden -std=c++17 -fmessage-length=0 -march=nocona -mtune=haswell -ftree-vectorize -fPIC -fstack-protector-strong -fno-plt -O2 -pipe -I/cvmfs/soft.galaxy.v2.1/tool-dependency/_conda/envs/mulled-v1-7576289d51ff5aefa21c8a2af901e1a61eb245f4e5fd66ae894d120dd35ff8f6/include -fdebug-prefix-map=/opt/conda/bld/bowtie2_1549224771007/work=/usr/local/src/conda/bowtie2-2.3.4.3 -fdebug-prefix-map=/cvmfs/soft.galaxy.v2.1/tool-dependency/_conda/envs/mulled-v1-7576289d51ff5aefa21c8a2af901e1a61eb245f4e5fd66ae894d120dd35ff8f6=/usr/local/src/conda-prefix -std=c++98 -DPOPCNT_CAPABILITY -DWITH_TBB -DNO_SPINLOCK -DWITH_QUEUELOCK=1 Sizeof (int, long, long long, void*, size_t, off_t): (4, 8, 8, 8, 8)

Tool Standard: stdout
Output: Tool Standard: stderr
Error: Tool Exit Code: 0
History Content: 4a4c5ace204a0514
API ID:

History
search datasets
AlignmentAndVariantDetection
3 shown
439.1 MB
3: Bowtie2 on data 2 and data 1: aligned reads (BAM)
68.1 MB
format: bam, database: ?
Settings:
Output files: "genome.*.bt2"
Line rate: 6 (line is 64 bytes)
Lines per side: 1 (side is 64 bytes)
Offset rate: 4 (one in 16)
Ftable chars: 10
Strings: unpacked
Max bucket size: default
Max bucket size, sqrt multiplier: default
display with IGV local
display in IGV View
Binary bam alignments file
2: HinfKW20_genomic.fna
1: HinfKW20_datanubset.fast
q

Here we can retrieve a summary of our alignment run including the overall alignment rate.

1.4.0 Review the alignment output

1.4.1 Return to the *History pane > Bowtie 2 on data 2 and data 1: aligned reads > View data* icon.

Caution! Notice that the entry says the file format is **BAM** but we are able to view it in a human readable **SAM** format when on Galaxy. If you were to download the data, however, it would not be converted!


1.4.2 In SAM format, we can see that each read is represented by one line with the columns in these files representing:

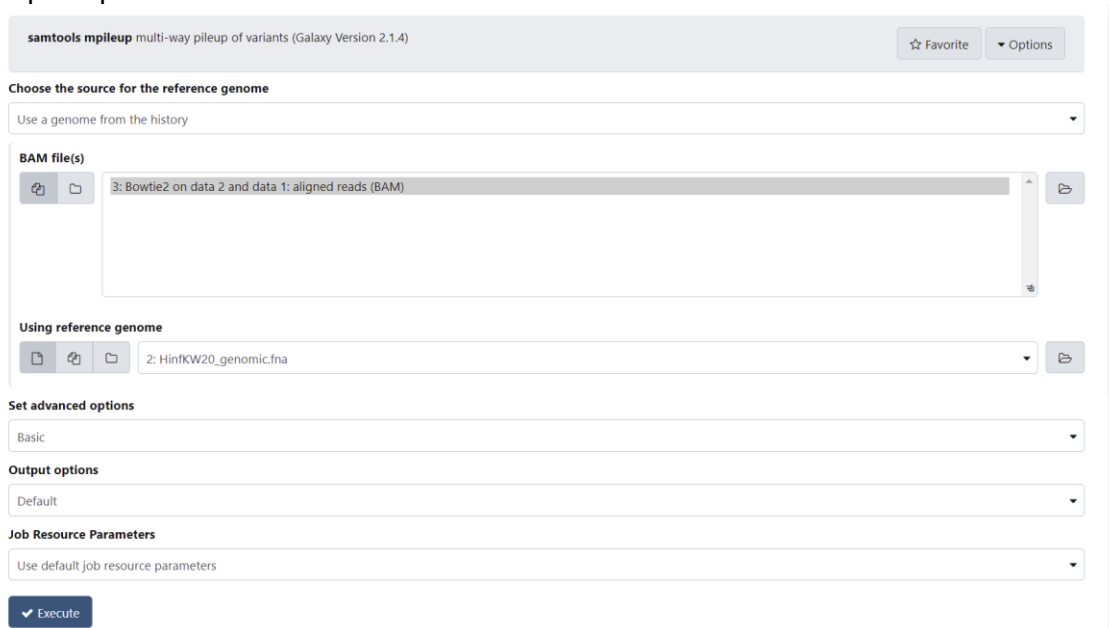
- QNAME: Read name
- FLAG: Bitwise flag that categorizes the read (eg. Unmapped or mapped)
- RNAME: Reference sequence
- POS: Leftmost mapping position
- MAPQ: Mapping quality
- CIGAR: Cigar string providing information on gaps in the alignment
- MRNM/RNEXT: Reference sequence of the mate (paired-end only)
- MPOS/PNEXT: Reference position of the mate (paired-end only)
- ISIZE/TLEN: Template length based on position of the mate (paired-end only)
- SEQ: Read sequence string
- QUAL: Read quality string (Phred scores)
- OPT: Optional field

2.0.0 Variant calling with SAMtools and BCFtools

Now that we've generated our alignment and have a binary version of a sequence alignment map, we can proceed with assessing the alignment for the presence of SNVs, indels, or even inversions. We'll begin by examining the result of an **mpileup**, which as discussed in Module 1.

2.1.0 Generate an *mpileup* of your alignment

- 2.1.1 Go to [tools pane](#) > 
- 2.1.2 Scroll down to review the purpose and input/output information for [samtools mpileup](#).
- 2.1.3 Execute [samtools mpileup](#) with the following settings:
 - Choose the source for the reference genome: **Use Genome from the history**
 - BAM file(s): **Bowtie2 on data 2 and data 1: aligned reads (BAM)**
 - Using reference genome: **HinfKW20_genomic.fna**
 - Set advanced options: Basic
 - Output Options: Default



2.2.0 Reviewing *mpileup* results

Time to review the results of the pileup we've generated. It will give us a position-by-position summary of sequence calls based on the aligned reads we generated from [Bowtie2](#).

- 2.2.1 Click on [History pane](#) > [samtools mpileup on data 2 and data 3 pileup](#) > [View data](#) icon. This will bring up a table version of the mpileup on Galaxy. The 6 columns presented represent the following:
 1. Reference chromosome/contig
 2. Reference position
 3. Reference base
 4. Reads covering the site
 5. String of all aligned bases
 - period = matched base on forward strand

- comma = matched base on reverse strand
- AGTCN = unmatched base on forward strand
- agtcn = unmatched base on reverse strand
- +1G = string denoting insertion
- -4atcg = string denoting deletion
- ^/\$ = start/end of reads
- K (or any other ASCII character following the '^') = mapping quality of the *entire read*.
Calculate ASCII value – 33 to obtain the score.

6. String of all base qualities

1	2	3	4	5	6
L42023.1	1	T	3	^!^K.^K.	1BB
L42023.1	2	A	4	..^K.^K.	B4BB
L42023.1	3	T	6^K.^K.	B;AC=B
L42023.1	4	G	7^K.	?@CA9BB
L42023.1	5	G	10^K.	C@8?A@BCB?
L42023.1	6	C	11^K.	C@A>BABCBCB
L42023.1	7	A	12^K.	CB@>CBBBBB@B
L42023.1	8	A	13^K.	CB@:CC<BBBBCB
L42023.1	9	T	14^K.	CCB?CCCCCCCCB
L42023.1	10	T	15^K.	CBBACCBCCCBCBA
L42023.1	11	A	16^K.	CB@>CCBB@ABBBB?
L42023.1	12	A	17^K.	CBA:CBB=?BCBCBBA;
L42023.1	13	A	19^K.^K.	CA>7CB>=AABBCCCB;==
L42023.1	14	A	22^K.^K.^K.	CB>7@B=9BA>BCCCB=>>BBB
L42023.1	15	T	23^K.	CBB?BC?BCBCCCCBBCBCCCB
L42023.1	16	T	24^K.	CCB<BCACCBBCCB@BCCB
L42023.1	17	G	25^K.	CB9;BBA@B3C?B=9BA?BCBBBCB

2.3.0 Convert BAM files to a variant call format

Although the *mpileup* file is a useful way to quickly glance around the genome, it does not necessarily summarize all the variants in a friendly way. Which reads support the presences of variants or indels? We can, however, convert our Bowtie2 BAM file to the variant call format (VCF) which further summarizes the potential variants across the genome while accounting for specific parameters like genome coverage. These can also be imported by other programs like IGV as we'll see in later steps. To accomplish this task, we'll use a related package SAMtools called BCftools. The purpose of this package is to handle variant calling and the manipulation of variant data.

2.3.1 Click on *tools pane* >

2.3.2 Scroll down to review the purpose and input/output information for *bcftools mpileup*.

Note that despite it's name, this is not creating an mpileup for us but rather a VCF file. The naming is a legacy to when SAMtools used to provide a VCF tool.

2.3.3 Execute *bcftools mpileup* with the following settings:

- Alignment inputs: **Single BAM/CRAM**
- Input BAM/CRAM: **Bowtie2 on data 2 and data 1: aligned reads (BAM)**
- Choose the source for the reference genome: **History**
 - Genome Reference: **HinfKW20_genomic.fna**
- Output_type:
- All other settings: **default**

bcftools mpileup Generate VCF or BCF containing genotype likelihoods for one or multiple alignment (BAM or CRAM) files (Galaxy Version 1.9+galaxy2) ☆ Favorite ▼ Options

Alignment inputs

Single BAM/CRAM

Input BAM/CRAM

3: Bowtie2 on data 2 and data 1: aligned reads (BAM)

Choose the source for the reference genome

History

Genome Reference

2: HinfKW20_genomic.fna

Indel Calling

Input Filtering Options

Output format

2.4.0 Review your VCF output

With the variant calling completed, a new VCF file is available in the [History pane](#).

2.4.1 Click on [History pane > bcftools mpileup on data 2 and data 3 pileup > View data](#) icon. This will bring up a view to the VCF file on Galaxy. The start of the file contains metadata represented as a series of lines with # at the beginning. Metadata can contain information like how the VCF file was generated (for reproducibility!) or how to interpret column codes. The metadata is followed by the final # line which includes the column names. The 10 columns of the header represent the following:

1. **Chrom:** Reference chromosome/contig
 2. **Pos:** Reference position. Note that there may be multiple entries with the same position.
 3. **ID:** Identifier of the variant. A semicolon-separated list of unique identifiers where available.
 4. **Ref:** Reference base(s)
 5. **Alt:** Alternative base(s) or <*> (homozygous reference site)
 6. **Qual:** A Phred-scaled quality score for the assertion made in "Alt".
 7. **Filter:** Filter status of the call. Usually PASS if the position has passed all filters.
 8. **Info:** A semicolon-separated list of additional information with codes listed in the metadata.
 9. **Format:** An [optional genotype information format descriptor](#) consisting of many colon-separated fields. The codes describe the type of data and its order in the remaining columns.
 10. **Sample_name:** Genotype data field(s) if **Format** is specified.
- * Missing values are specified with a dot (.). See VCF 4.2 specifications section 1.4.2 for more genotype information.

```
##reference=file:///ref.fa
##contig=<ID=L42023.1,length=1830138>
##ALT=<ID=*,Description="Represents allele(s) other than observed.">
##INFO=<ID=INDEL,Number=0,Type=Flag,Description="Indicates that the variant is an INDEL.">
##INFO=<ID=IDV,Number=1,Type=Integer,Description="Maximum number of reads supporting an indel">
##INFO=<ID=IMF,Number=1,Type=Float,Description="Maximum fraction of reads supporting an indel">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Raw read depth">
##INFO=<ID=VDB,Number=1,Type=Float,Description="Variant Distance Bias for filtering splice-site artefacts in RNA-seq data (bigger is better)",Version="3">
##INFO=<ID=RPB,Number=1,Type=Float,Description="Mann-Whitney U test of Read Position Bias (bigger is better)">
##INFO=<ID=MQB,Number=1,Type=Float,Description="Mann-Whitney U test of Mapping Quality Bias (bigger is better)">
##INFO=<ID=BQB,Number=1,Type=Float,Description="Mann-Whitney U test of Base Quality Bias (bigger is better)">
##INFO=<ID=MQSB,Number=1,Type=Float,Description="Mann-Whitney U test of Mapping Quality vs Strand Bias (bigger is better)">
##INFO=<ID=SQB,Number=1,Type=Float,Description="Segregation based metric">
##INFO=<ID=MQ0F,Number=1,Type=Float,Description="Fraction of MQ0 reads (smaller is better)">
##INFO=<ID=I16,Number=16,Type=Float,Description="Auxiliary tag used for calling, see description of bcf_callret1_t in bam2bcf.h">
##INFO=<ID=QS,Number=R,Type=Float,Description="Auxiliary tag used for calling">
##FORMAT=<ID=PL,Number=G,Type=Integer,Description="List of Phred-scaled genotype likelihoods">
#CHROM      POS      ID      REF      ALT      QUAL      FILTER      INFO
L42023.1    1        .        T        <*>      0        .        DP=4:116=3,0,0,0,82,2434,0,0,84,3528,0,0,0,0,0,0;QS=1
L42023.1    2        .        A        <*>      0        .        DP=6:116=4,0,0,0,118,3628,0,0,168,7056,0,0,2,2,0,0;QS
L42023.1    3        .        T        <*>      0        .        DP=8:116=6,0,0,0,186,5818,0,0,252,10584,0,0,6,10,0,0;C
L42023.1    4        .        G        <*>      0        .        DP=9:116=7,0,0,0,217,6795,0,0,294,12348,0,0,12,28,0,0
L42023.1    5        .        G        <*>      0        .        DP=10:116=10,0,0,0,311,9765,0,0,336,14112,0,0,30,120
```

2.5.0 Filter VCF for variants only

You may notice from your output that you are getting base calls at every position. While good for confirmation, again there is an excess of information that we do not need. Instead, we want to filter the variants based some criteria like their likelihood, and we only want to view non-reference variants in the case of output from *bcftools mpileup*.

2.5.1

2.5.2 Scroll down to review the purpose and input/output information for *bcftools call*.

2.5.3 Execute *bcftools call* with the following settings:

- VCF/BCF Data: **bcftools mpileup on data 2 and data 3**
- Input/output Options
 -
- output_type: **uncompressed VCF**
- All other settings: **default**

2.5.4 Review your output with *History pane > bcftools call on data 5 > View data*.

Notice now that our data has been filtered to include only variant information.

#bcftools_callCommand=call -m --prior 0.0011 --variants-only --output-type v --threads 4 input.vcf.gz; Date=Mon Nov 15 20:51:16 2021									
#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO		
L42023.1	1056	.	CG	CGG	47.401	.	INDELIDV=5;IMF=0.5;DP=10;VDB=0.520708;SGB=-0.590765;MQSB=0.916482;MQOF=0;ICB=1;HOB=0.5		
L42023.1	3808	.	T	C	225	.	DP=34;VDB=0.0173125;SGB=-0.69312;MQSB=0.942118;MQOF=0;AC=2;AN=2;DP4=0.0,13,19,MQ=40		
L42023.1	20891	.	AG	AGG	166	.	INDELIDV=39;IMF=0.886364;DP=44;VDB=0.623729;SGB=-0.693145;MQSB=0.919965;MQOF=0;AC=2;AN		
L42023.1	29688	.	A	T	225	.	DP=38;VDB=0.251571;SGB=-0.693132;MQSB=0.994173;MQOF=0;AC=2;AN=2;DP4=0.0,12,22,MQ=39		
L42023.1	32329	.	GAAAAA	GAAAAAA	47.106	.	INDELIDV=17;IMF=0.708333;DP=24;VDB=0.947761;SGB=-0.69168;MQSB=0.486752;MQOF=0;ICB=1;HOB		
L42023.1	36715	.	G	A	218	.	DP=43;VDB=0.650351;SGB=-0.693143;MQSB=0.578387;MQOF=0;AC=2;AN=2;DP4=0.0,34,4,MQ=40		
L42023.1	45496	.	G	A	225	.	DP=29;VDB=0.433253;SGB=-0.692976;MQSB=0.81328;MQOF=0;AC=2;AN=2;DP4=0.0,12,14,MQ=39		
L42023.1	46301	.	A	T	225	.	DP=32;VDB=0.0188474;SGB=-0.693079;MQSB=0.378815;MQOF=0;AC=2;AN=2;DP4=0.0,18,11,MQ=39		
L42023.1	47284	.	A	T	225	.	DP=96;VDB=0.0608299;SGB=-0.693147;MQSB=0.993662;MQOF=0;AC=2;AN=2;DP4=0.0,70,13,MQ=40		
L42023.1	52066	.	CAAAA	CAAAAA	135	.	INDELIDV=56;IMF=0.823529;DP=68;VDB=0.665014;SGB=-0.693147;MQSB=0.967528;MQOF=0;AC=2;AN		
L42023.1	52337	.	G	A	225	.	DP=77;VDB=0.0323241;SGB=-0.693147;MQSB=0.337194;MQOF=0;AC=2;AN=2;DP4=0.0,26,47,MQ=40		
L42023.1	52396	.	G	A	225	.	DP=48;VDB=0.407416;SGB=-0.693147;MQSB=0.413808;MQOF=0;AC=2;AN=2;DP4=0.0,17,30,MQ=39		
L42023.1	54392	.	C	A	225	.	DP=77;VDB=0.318516;SGB=-0.693147;MQSB=0.933517;MQOF=0;AC=2;AN=2;DP4=0.0,37,28,MQ=40		
L42023.1	55727	.	G	A	225	.	DP=71;VDB=0.177135;SGB=-0.693147;MQSB=0.0572852;MQOF=0;AC=2;AN=2;DP4=0.0,17,41,MQ=40		
L42023.1	61735	.	A	C	225	.	DP=34;VDB=0.00587795;SGB=-0.69311;MQSB=0.906252;MQOF=0;AC=2;AN=2;DP4=0.0,10,21,MQ=40		
L42023.1	68938	.	C	T	81	.	DP=6;VDB=0.211317;SGB=-0.556411;MQSB=1;MQOF=0;AC=2;AN=2;DP4=0.0,2,2,MQ=40		
L42023.1	77720	.	G	A	225	.	DP=29;VDB=0.817083;SGB=-0.693079;MQSB=0.311089;MQOF=0;AC=2;AN=2;DP4=0.0,11,18,MQ=39		
L42023.1	77924	.	ATTTT	ATTTTT	151	.	INDELIDV=18;IMF=0.947368;DP=19;VDB=0.0279288;SGB=-0.69168;MQSB=0.980199;MQOF=0;AC=2;AN		
L42023.1	84550	.	C	T	224	.	DP=40;VDB=0.00445402;SGB=-0.69312;MQSB=0.995309;MQOF=0;AC=2;AN=2;DP4=0.0,29,3,MQ=40		

2.6.0 Download your VCF data

As with our data from Module 2 we can download our VCF file here. Looking in the History pane, you can find details about the file size.

2.6.1 Return to your output list on the *History pane* and select the *Operations on multiple datasets* icon.

2.6.2 Select the following datasets.

- *Bowtie2* on data 2 and data 1: aligned reads (BAM)
- *samtools mpileup* on data 2 and data 3 pileup
- *bcftools mpileup* on data 2 and data 3
- *bcftools call* on data 5

2.6.3 Select *History pane > For all selected > Build Dataset List*.

2.6.4 Name the dataset list **RefAlignDownload**.

2.6.5 Use the *Create list* button.

2.6.6 Select History pane > RefAlignDownload > Download Collection.

2.6.7 Save the .tgz file to **~/FGDS/Module3/downloads** (~172 Mb).

2.6.8 Unarchive the contents of your .tgz file and decompress the contents to **~/FGDS/Module3**.

2.6.9 Rename your files using the following table


Download name	New name
Bowtie2 on data 2 and data 1_ aligned reads (BAM).bam	HinfKW20_refalign.bam
Bowtie2 on data 2 and data 1_ aligned reads (BAM).bam.bai	HinfKW20_refalign.bam.bai
bcftools mpileup on data 2 and data 3.vcf	HinfKW20.vcf
bcftools call on data 5.vcf	HinfKW20_filtered.vcf
samtools mpileup on data 2 and data 3 pileup.pileup	HinfKW20.mpileup

2.7.0 Save your history as a workflow

Now that you've completed your first reference alignments and variant calling analysis, you can save your history as a new workflow for future use. You can come back and edit the finer details of the workflow at a later time.

2.7.1 Go to History pane > History options > Extract Workflow

2.7.2 Name your workflow **AlignmentAndVariantDetection** and review the tools/input files that are going to be used in the workflow.

2.7.3 Deselect the  you've created so that it won't become part of the Workflow.

2.7.4 Once you are ready, select Create Workflow.

The following list contains each tool that was run to create the datasets in your current history. Please select those that you wish to include in the workflow.

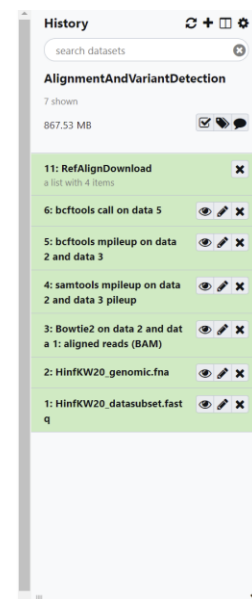
Tools which cannot be run interactively and thus cannot be incorporated into a workflow will be shown in gray.

Workflow name

AlignmentAndVariantDetection

Create Workflow Check all Uncheck all

Tool	History Items created
<div>Upload File</div> <div>This tool cannot be used in workflows</div>	<div>1 HinfKW20_datasubset.fastq</div> <div><input checked="" type="checkbox"/> Treat as input dataset HinfKW20_datasubset.fast</div>
<div>Upload File</div> <div>This tool cannot be used in workflows</div>	<div>2 HinfKW20_genomic.fna</div> <div><input checked="" type="checkbox"/> Treat as input dataset HinfKW20_genomic.fna</div>
<div>Bowtie2</div> <div><input checked="" type="checkbox"/> Include "Bowtie2" in workflow</div>	<div>3 Bowtie2 on data 2 and data 1: aligned reads (BAM)</div>
<div>samtools mpileup</div> <div><input checked="" type="checkbox"/> Include "samtools mpileup" in workflow</div>	<div>4 samtools mpileup on data 2 and data 3 pileup</div>
<div>bcftools mpileup</div> <div><input checked="" type="checkbox"/> Include "bcftools mpileup" in workflow</div>	<div>5 bcftools mpileup on data 2 and data 3</div>
<div>bcftools call</div> <div><input checked="" type="checkbox"/> Include "bcftools call" in workflow</div>	<div>6 bcftools call on data 5</div>
<div>Dataset Collection Creation</div> <div>Dataset collection created in a way not compatible with workflows</div>	<div>11 RefAlignDownload</div> <div><input type="checkbox"/> Treat as input dataset RefAlignDownload</div>



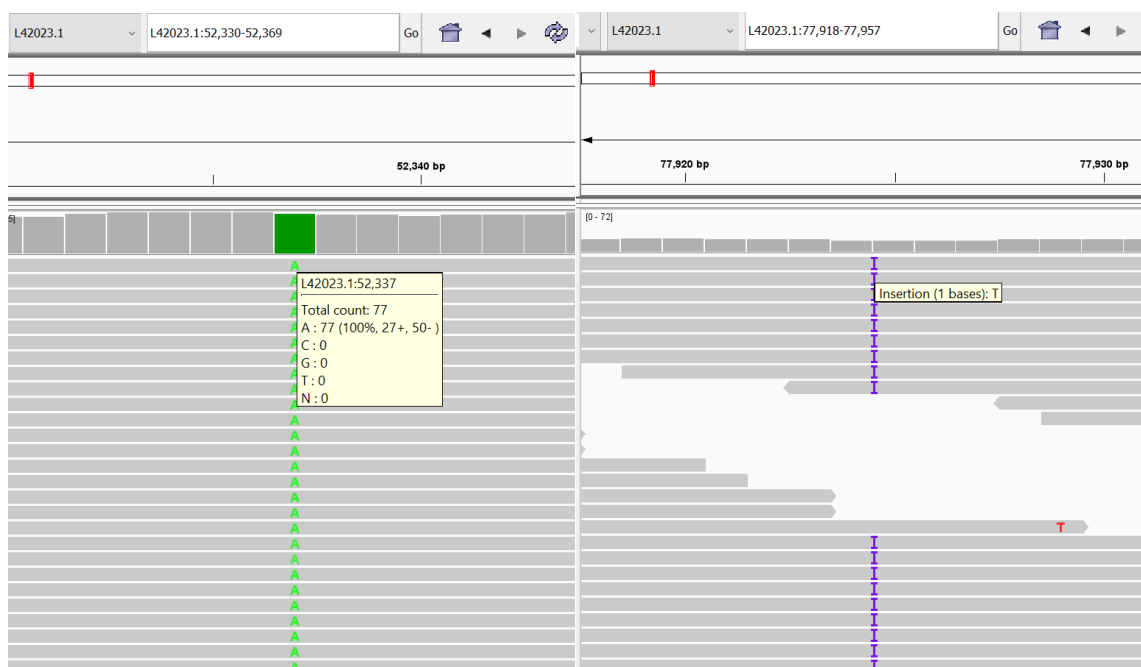
3.0.0 Viewing your reference alignment in IGV

Now that we've downloaded our reference alignment and VCF data from Galaxy, we can use a graphical interface like IGV to view the read alignments themselves using the VCF information.

3.1.0 Loading your files in IGV

- 3.1.1 Open up IGV and, if necessary, load your genome again from Module 1 using Genomes > Load Genome from File > HinfKW20_genomic.fna.
- 3.1.2 Load your BAM file generated in this module using File > Load from File > HinfKW20_refalign.bam
- 3.1.3 Briefly return to Galaxy to view your filtered VCF file. You can pick a site with some interesting variants. In our case we will pick a single nucleotide polymorphism (SNP) at position 53,227.
- 3.1.4 In the IGV search box, input L42023.1:52337. IGV will automatically convert this into a 41bp range centred on your point of interest.
- 3.1.5 Search for an INDEL site at [redacted] Note the number of reads that cover the site and the reads that support the INDEL call.

How many reads support the presence of an INDEL? How does this compare to our VCF data?



3.2.0 Importing your VCF data

Now that you have your aligned reads imported onto the *H. influenzae* genome you can also import your VCF file for comparison. As we will see, however, there are some portions of the BCFtools VCF that IGV is unable to interpret due to the presence of ambiguous nucleotide symbols.

- 3.2.1 Load your VCF file using File > Load from File > HinfKW20_filtered.vcf. You will produce an error in IGV during import.
- 3.2.2 Load your HinfKW20_filtered.vcf in a text editor and search for line 68.

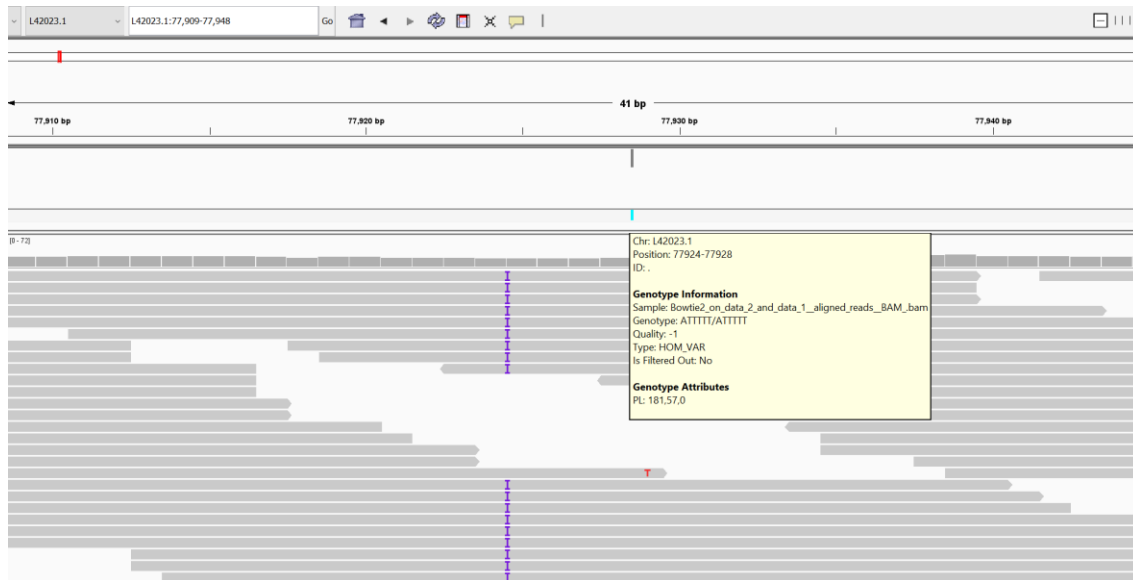
3.2.3

3.2.4

3.2.5 Save the file as **HinfKW20_filtered_edit.vcf**.

3.2.6 Attempt to load the edited VCF file using File > Load from File > HinfKW20_filtered_edit.vcf.

3.2.7 Look around the genome with IGV and return to L42023.1:77,924. You can now see the VCF entry for this deletion, providing a quick summary about the evidence supporting the call.



3.3.0 Save your session

You can return to this session later to look closer at other variants and examine the genome. What do you think you'll find at position 1,289,372?

3.3.1 Save the session using File > Save Session > HinfKW20_alignment.xml.

4.0.0 RNA-Seq analysis with *HISAT2*, *featureCounts*, and *DESeq2*

In a similar vein to genomic reference alignments are the alignment of RNA-Seq reads to a reference alignment. While similar in idea, we now have two options for alignment: **genome mapping** to a genomic reference or **transcriptome mapping** to a specific transcriptome. In either case we require a reference sequence much like our genomic alignment. Alternatively, you could attempt a *de novo* analysis by assembling transcript reads into contigs and then using this as your reference for counting transcripts afterwards.

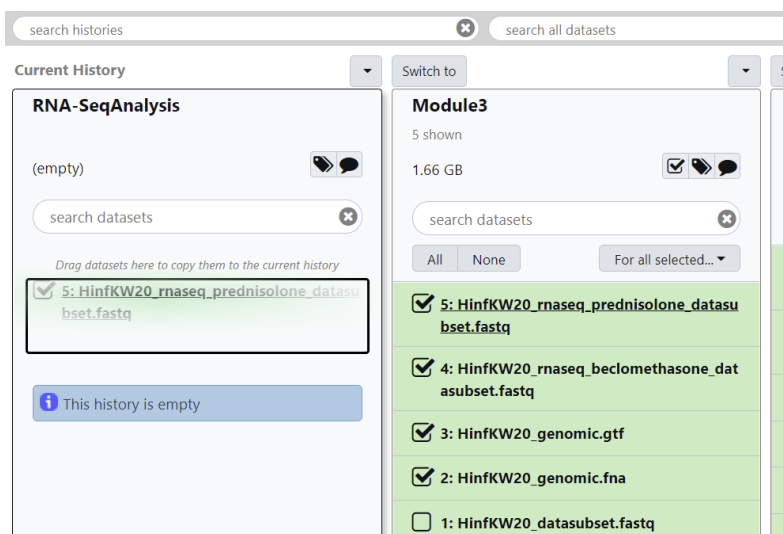
For your first RNA-Seq analysis we are going to use a new RNA-Seq dataset that was collected for *H. influenzae* KW20 under two different antibiotic treatments (**beclomethasone** and **prednisolone**). The goal is to identify whether any genes are differentially regulated during these antibiotic treatments. We will be quantifying the expression of all genes in the transcriptome under both treatments and measuring how gene expression differs between these two treatments. However, it is important to note that in the interest of time, we are only using one replicate from each treatment in this portion of the module. In practice, you would have at least three biological replicates under each condition and include a control set of replicates.

4.1.0 Load data for the analysis

Before we begin we will need to pull our data from the Shared History page and put it into a new history called **RNA-SeqAnalysis**.

- 4.1.1 Go to History pane > Create new history and name the history **RNA-SeqAnalysis**.
- 4.1.2 Go to History pane > View all histories.
- 4.1.3 From the **Module3Data** history, select the Operation on multiple datasets icon and selection the following history entries:

-
-
-
-



- 4.1.4 Return to the Galaxy homepage.

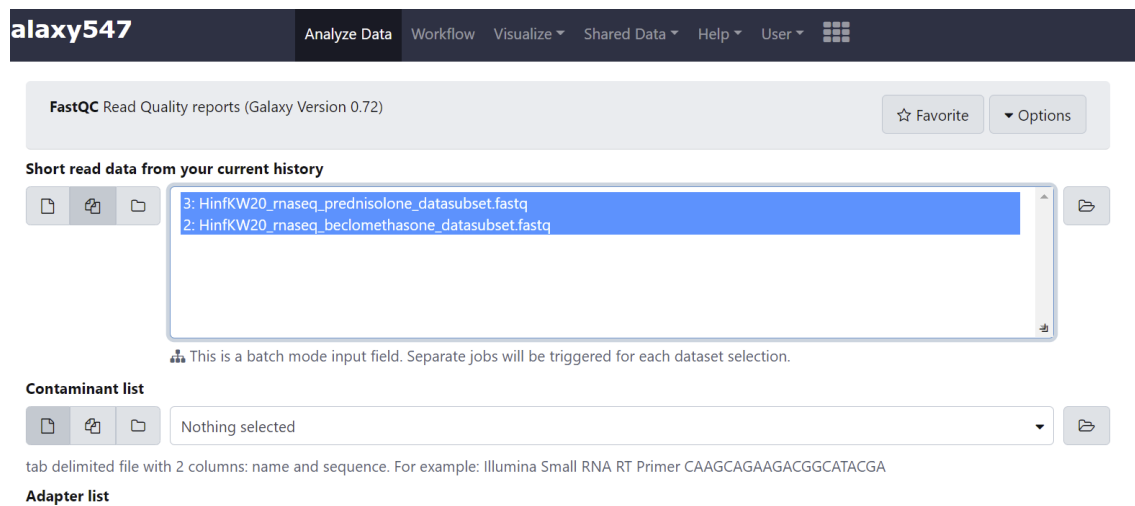
4.2.0 Assess the quality of your RNA-Seq data with *FastQC*

Before we go through the effort of analysing these reads, we should assess their quality with FastQC and see how they differ from our random subset of raw genomic data.

4.2.1 Select *tools pane > FastQC Quality Control > FastQC Read Quality reports*

4.2.2 Execute *FastQC* with the following settings:

- Short read data from your current history: **multiple datasets**
 - Select both RNA-seq datasets using multi-select (**ctrl+select**).
- Other options: **default**



4.3.0 Review the RNAseq FastQC output

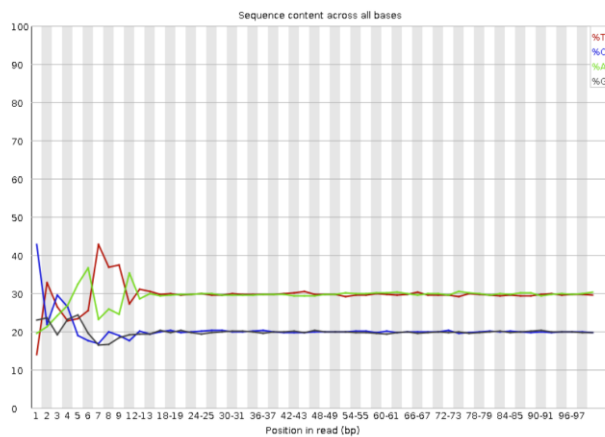
4.3.1 Review the HTML output from both data files by selecting the *View data* icon on the history entry.

- **5: FastQC on data 2: Webpage**
- **7: FastQC on data 3: Webpage**

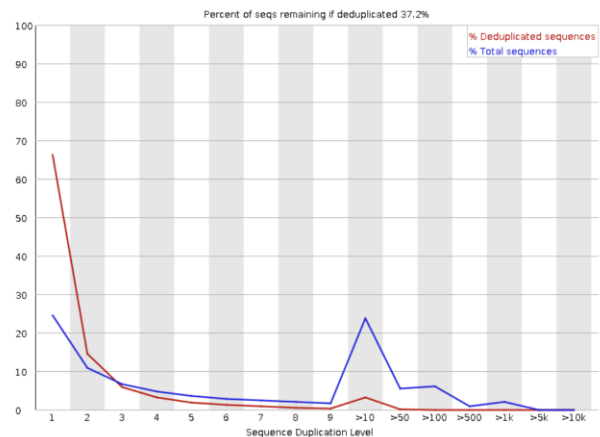
You should notice that the overall quality of the sequencing data is good, but that there are a few tests that produce warnings or fail (**Per base sequence content**, **Sequence Duplication Levels**, and **Overrepresented sequences**). When looking at the plots, you will notice that the per base sequence content warning is caused by the same issue that we saw in Module 2 – bias in the base content at the start of reads. As we noted in Module 2, this tends to be indicative of slight enzyme bias in library preparation, and we want to leave this alone.

As for the **Sequence Duplication Levels** and **Overrepresented sequences**, this is a common observation in RNA-Seq data because the expression levels of different genes can vary by orders of magnitude, so you would expect to encounter more duplicated reads from those genes. There is considerable debate on the extent to which you should deduplicate RNA-Seq data, with some lines of research suggesting that you shouldn't deduplicate at all. For simplicity, today we will proceed without deduplicating our dataset, but we will look for cases where there is unequal coverage within genes when we visualize our alignments.

! Per base sequence content



Sequence Duplication Levels



! Overrepresented sequences

Sequence	Count	Percentage	Possible Source
CTTCGATCCTCAAACGGTGGCTTCCACACGATCTCGTTTTGTTTGACTC	4501	0.22505	No Hit
CTGGATTCGACGGGATTAGCGAAGCCCAAGGTGCACGTCGAGGTGCGGTA	4172	0.2086	No Hit
CTCGTCTTTAATTTCACTTAAGCATGCGGACAGACACGCTAAACTTAAGC	4089	0.20445	No Hit
GTCCGAAATTACTCTACCTTCAGTACTACACGTTTAGTCTCGTCTTTAAT	3474	0.1737	No Hit
CGGGATTAGCGAAGCCCAAGGTGCACGTCGAGGTGCGGTAGGCCTCGTAA	3151	0.15755	No Hit
CAATCGCTGGTTTATTGAAGCCCTTAACCGTATTATACGACCTAGTGGG	2520	0.126	No Hit
CGCGGAGTCAAACCAAAACGAGATCGTGTGGAAGCCACCGTTTGAGGATC	2471	0.1235500000000001	No Hit
CTCGTTTTGTTTGACTCCGCGTTATCCCCTTACGAGCGGAAGCTGG	2016	0.1008	No Hit

4.4.0 Align your RNA-Seq reads with *HISAT2*

Now that we've looked at the quality of our reads we can align them to a genome to create BAM file. This is very much like our previous reference alignment. We'll even use the same reference sequence but now we will only achieve alignment to the coding regions of the genome (assuming no genomic contamination!).

4.4.1 Select *tools pane > RNA-seq >*

4.4.2 Scroll down to review the purpose and input/output information for *HISAT2*.

4.4.3 Execute *HISAT2* with the following settings:

- Source for the reference genome: Use a genome from history
 - HinfKW20_genomic.fna
- Is this a single or paired library: **Single-end**
- Other options: **default**

4.4.4 Note that *HISAT2* will create a BAM file for each RNA-Seq dataset which you will rename.

History name	New name
HISAT2 on data 3 and data 1: aligned reads (BAM)	HISAT2 on data 1 and beclo_R2: aligned reads (BAM)
HISAT2 on data 4 and data 1: aligned reads (BAM)	HISAT2 on data 1 and pred_R3: aligned reads (BAM)

4.5.0 Convert your alignments into count data with *featureCounts*

Now that we've completed a reference alignment of our RNA-Seq reads, we can begin an analysis of those alignments to produce counts. Depending on the algorithm these will analyse the depth of the sequencing reads per segment of genome. Two helpful tools are *featureCounts* and *htseq-count*, both of which are quite efficient for mapping gene counts to a known transcriptome where reads will map uniquely to a single transcript. Other tools such as *StringTie* can produce novel transcripts and annotation for genes that may not exist in a supplied reference file.

Since we are working with a prokaryotic genome, we aren't expecting any alternative splicing or isoforms to deal with. Therefore, we'll be working with the *featureCounts* tool which will produce output compatible for further downstream analysis as well. The read counts produced by this tool are raw, unnormalized read counts.

What are the normalization units for RNA-Seq counts? When genome reads are analysed, for depth, one should consider the length of genes and the probability of sequencing their entire coding sequence from a library. These values can be influenced by gene length, and the overall read depth of your library. For more information on these units of measure check out this helpful primer:

https://www.reneshbedre.com/blog/expression_units.html

4.5.1

4.5.2 Scroll down to review the purpose and input/output information for *featureCounts*.

4.5.3 Execute *featureCounts* with the following settings:

- Alignment file: **Multiple datasets**
 - Select both BAM datasets using multi-select (ctrl+select).
- Gene annotation file: **in your history**
 - Gene annotation file: **HinfKW20_genomic.gtf**
- Output format: **Gene-ID "t" read-count (MultiQC/DESeq2/edgeR/limma-voom compatible)**
- Advanced options:
 - [Redacted]
 - [Redacted]
- Other options: **default**

4.5.4 Note that *featureCounts* will create two files for each *HISAT2* BAM dataset.

4.6.0 Review your *featureCounts* output

4.6.1 Select *History pane > featureCounts on data 2 and data 9: summary > View data*.

Here we can see how many reads were assigned to features in a unique manner. You'll find additional information on unassigned reads and why they may be unassigned.

4.6.2 Select *History pane > featureCounts on data 2 and data 9 > View data*.

Here we can see a simple tabular format with the GeneID and read counts listed. If we had chosen the default output this would have included additional column information based on the GTF file: Chr, start, end, strand, and length. See below for an example contrasting the two outputs.

Geneid	HISAT2 on data 3 and data 1: aligned reads (BAM)	Geneid	Chr	Start	End	Strand	Length	HISAT2 on data 3 and data 1: aligned reads (BAM)
Geneid	HISAT2 on data 3 and data 1: aligned reads (BAM)	Geneid	Chr	Start	End	Strand	Length	HISAT2 on data 3 and data 1: aligned reads (BAM)
HI_0001	5475	HI_0001	L42023.1	2	1021	+	1020	5475
HI_0002	823	HI_0002	L42023.1	1190	3013	+	1824	823
HI_0003	194	HI_0003	L42023.1	3050	3838	-	789	194
HI_0004	151	HI_0004	L42023.1	3854	4318	-	465	151
HI_0005	137	HI_0005	L42023.1	4579	5391	-	813	137
HI_0006	1318	HI_0006	L42023.1	5656	8748	+	3093	1318
HI_0007	360	HI_0007	L42023.1	8750	9688	+	939	360
HI_0008	267	HI_0008	L42023.1	9681	10397	+	717	267
HI_0009	74	HI_0009	L42023.1	10467	11375	+	909	74
HI_0010	38	HI_0010	L42023.1	11414	11854	-	441	38
HI_0011	127	HI_0011	L42023.1	11857	12261	-	405	127
HI_0012	80	HI_0012	L42023.1	12367	13359	+	993	80
HI_0013	228	HI_0013	L42023.1	13423	14331	-	909	228
HI_0014	159	HI_0014	L42023.1	14328	15011	-	684	159
HI_0015	544	HI_0015	L42023.1	15013	16062	-	1050	544
HI_0016	860	HI_0016	L42023.1	16071	17867	-	1797	860
HI_0017	2700	HI_0017	L42023.1	18035	18418	-	384	2700
HI_0018	106	HI_0018	L42023.1	18676	19335	+	660	106
HI_0019	1047	HI_0019	L42023.1	19405	20829	-	1425	1047
HI_0020	1305	HI_0020	L42023.1	21248	22687	-	1440	1305

4.6.3 Rename the history entries for *featureCounts* to the following:

History name	New name
featureCounts on data 2 and data 9	featureCounts on data 2 and beclo_R2
featureCounts on data 2 and data 9: summary	featureCounts on data 2 and beclo_R2: summary
featureCounts on data 2 and data 10	featureCounts on data 2 and pred_R3
featureCounts on data 2 and data 10: summary	featureCounts on data 2 and pred_R3: summary

4.7.0 Add additional *featureCounts* data to your History

Before we can run our next tool, we'll want to include additional replicates for our dataset. Normally you would have included these in our initial analyses, but to save on time I've included them in their final *featureCounts* form. We'll include these in the next step for visualization and analysis purposes.

4.7.1 Select *Shared Data > Histories > Module3_replicates*.

4.7.2 You will see 4 additional sets of *featureCounts* data and summaries that you can import using the *Import history (+)* icon.

4.7.3 Select *History pane > View all histories*.

4.7.4 Switch to RNA-SeqAnalysis.

4.7.5 Multiselect and add all 8 history entries to your **RNA-SeqAnalysis** history.

4.7.6 Return to the Galaxy homepage.


4.8.0 Use MultiQC to quickly summarize multiple datasets

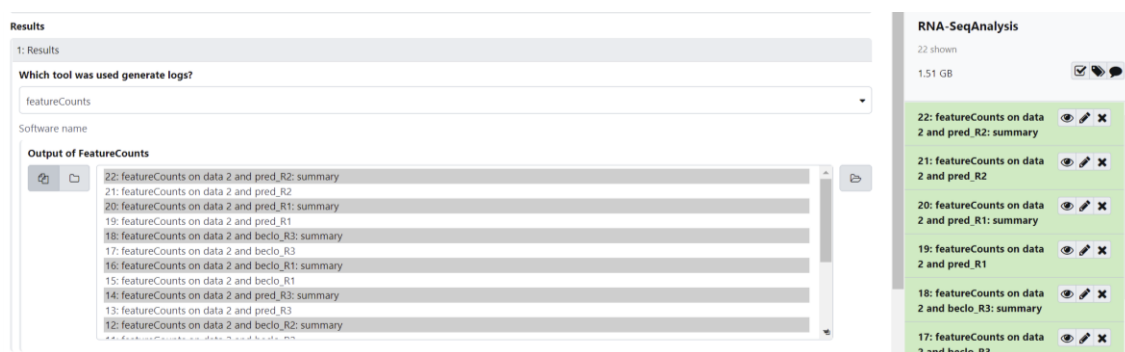
When you are working with many conditions and their replicates, as would be the case with RNA-Seq analysis, it is helpful to have a visual summary of all the samples in a single place. MultiQC can accomplish that for you by summarizing data/results across your many datasets. In this case, we will summarize our featureCounts results to get a sense of how well the reads aligned to the reference genome features we provided.

4.8.1

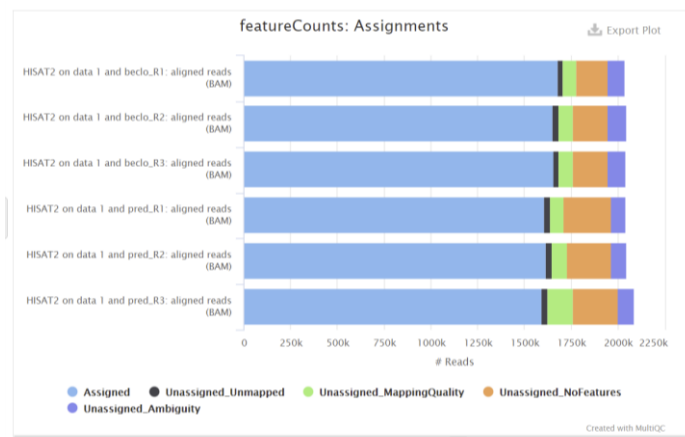
4.8.2 Scroll down to review the purpose and input/output information for MultiQC.

4.8.3 Execute MultiQC with the following settings:

- Results > Which tool was used generate logs?: **featureCounts**
- Software name > Output of FeatureCounts:
 - 
- Report title: **featureCounts summary analysis**
- Other options: **default**



4.8.4 Review the output of MultiQC using History pane > MultiQC on data 22 and data 20, and others: Webpage > View data. Here you'll be able to see a nice visualization of the assigned and unassigned reads based on category.



4.9.0 Identifying differentially expressed genes with DESeq2

Now that we have multiple replicates we can proceed with a proper analysis of our RNA-Seq data. Normally, you would be able to run DESeq2 without replicates – while not recommended this can be useful for exploratory analysis. The Galaxy interface, however, has removed the option to perform such a run without replicates for computation and technical reasons. We'll see in later modules how to accomplish this from the command line. For now, we'll run with all our replicate data.

Didn't you say we need normalized data? DESeq2 runs the normalization of your read counts internally using the data that has been provided. Therefore, you can provide the raw read counts generated by featureCounts. For more information on this check out the [DESeq2 manual here](#).

4.9.1 Select *tools pane > RNA-seq > **DESeq2** Determines differentially expressed features from count tables*

4.9.2 Scroll down to review the purpose and input/output information for DESeq2.

4.9.3 Execute DESeq2 with the following settings:

-
-
-



- Files have header? **Yes**
- Choice of input data: **Count data (eg from HTSeq-count, featureCounts or StringTie)**
- Visualizing the analysis results: **Yes**
- Output normalized counts table: **Yes**
- Other options: **default**

4.9.4 The DESeq2 tool will generate 3 output files for us to review: **DESeq2 result file**, **DESeq2 plots**, and a **Normalized counts file**.

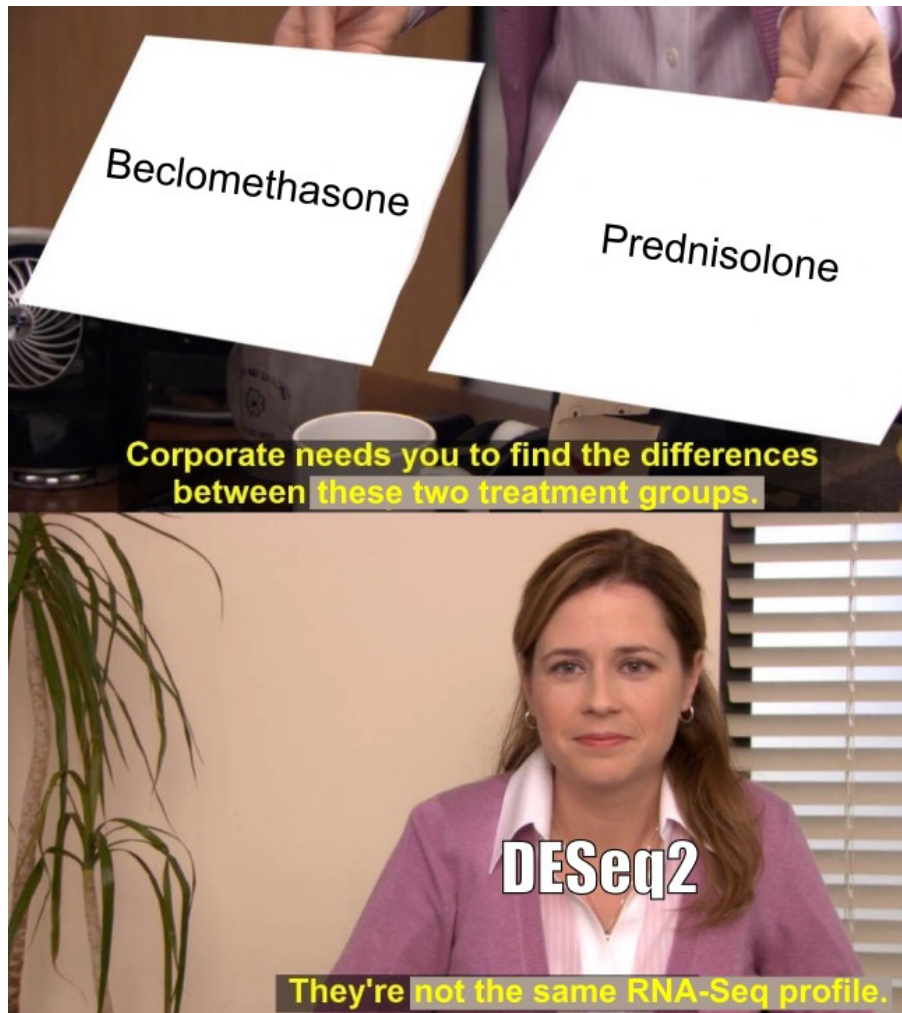
4.10.0 Review your DESeq2 output

That takes us to the end of this initial analysis. Let's briefly look at our output and see how the treatments compare.

4.10.1 View the plots generated by DESeq2 by selecting *History pane > View data* on the **DESeq2 plots** entry.

4.10.2 View a table of the log2 fold-change data along with p-values by selecting *History pane > View data* on the **DESeq2 result** entry. You can import these into additional tools for further analysis or use the data to create your own plots!

GeneID	Base mean	log2(FC)	StdErr	Wald-Stats	P-value	P-adj
HI_0815	7725.51229546391	-1.94976561340641	0.173674191475063	-11.2265708384562	3.01971930803936e-29	5.23015384152418e-26
HI_0809	17335.6876653991	-1.96257340565055	0.179322400872408	-10.9443850634532	7.06958161221749e-28	6.12225767618035e-25
HI_0092	530.631994865901	-1.93743565706793	0.190318093417858	-10.1799866858383	2.43594702251414e-24	1.40635341433149e-21
HI_0017	2610.2943176883	-1.75701888248677	0.175270156235434	-10.0246323745306	1.18800881204146e-23	5.14407815613951e-21
HI_0053	362.446113307527	-1.42278732793269	0.143476074987594	-9.91654760597341	3.52746731043013e-23	1.221914676333e-20
HI_0685	683.654780685379	-2.11355601355447	0.214752914499201	-9.84180363038723	7.43655121632135e-23	2.14668445111143e-20
HI_0181	700.446500349877	-1.24913246188805	0.139341674774787	-8.96452883824588	3.11610209009626e-19	7.71012688578104e-17
HI_0524	3046.23321704348	-0.939771404226934	0.107384981434715	-8.75142307305117	2.10678938472514e-18	4.56119901792993e-16
HI_0564	646.531292971648	-1.46742472868358	0.172308679701472	-8.51625542732915	1.64795145763108e-17	2.85425192461704e-15
HI_0813	929.341394434185	-0.975276674246421	0.114478742954091	-8.51928182542615	1.60545738860729e-17	2.85425192461704e-15
HI_0131	6129.98510334137	-1.01938078271225	0.123859891373252	-8.23011203554456	1.87038424846448e-16	2.94500501667316e-14
HI_0812	2018.78698350492	-0.794993285866024	0.0984431473092692	-8.0756589726705	6.71132074620412e-16	9.44248766183564e-14
HI_1331	294.603339705785	1.3345744842189	0.165395167124244	8.06900532478303	7.08731752909141e-16	9.44248766183564e-14
HI_0629	5260.67933812452	-0.983986856467144	0.124630035620245	-7.89526257912184	2.89704661487911e-15	3.58406052640758e-13
HI_0822	12494.7567226504	-1.23162950794794	0.15633202972631	-7.87828409328796	3.31907399715845e-15	3.83242410871896e-13
HI_1048	660.373501996682	1.16475617659731	0.149365160006278	7.79804458113492	6.28737995913352e-15	6.80608880576204e-13
HI_0091	301.661858192185	-1.52835271810429	0.196964224412142	-7.75954477350291	8.52347715731509e-15	8.68391908027632e-13
HI_0098	526.296290134172	1.37654992556427	0.180357275041752	7.63235042914466	2.3051182438584e-14	2.21803599909042e-12
HI_0751	2382.39934186511	-0.787893025847878	0.104224325447353	-7.55958863217461	4.04346091239391e-14	3.6859338422454e-12



5.0.0 Review your RNA-Seq analysis in IGV

Now that we've completed our analysis, we can download all of the generated data to our local directories and view them in IGV.

5.1.0 Download your dataset from Galaxy

5.1.1 Return to your output list on the [History pane](#) and select the [Operations on multiple datasets](#) icon.

5.1.2 Select the following datasets.

- HinfKW20_genomic.gtf
- HISAT2 on data data 1 and [beclo_R2/pred_R3]: aligned reads (BAM) [2 entries]
- featureCounts on data 2 and [beclo_R2/pred_R3] [2 entries]
- DESeq2 result file [1 entry]
- DESeq2 plots [1 entry]

5.1.3 Select [History pane > For all selected > Build Dataset List](#).

5.1.4 Name the dataset list **RNA-SeqDownload**.

5.1.5 Use the [Create list](#) button.

5.1.6 Select [History pane > RNA-SeqDownload > Download Collection](#).

5.1.7 Save the .tgz file to **~/FGDS/Module3/downloads** (~215 Mb).

5.1.8 Unarchive the contents of your .tgz file **and** decompress the contents to **~/FGDS/Module3**.

5.1.9 Rename your files using the following table

Download name	New name
HISAT2 on data 1 and beclo_R2_ aligned reads (BAM).bam	HinfKW20_beclo_R2_HISAT2.bam
HISAT2 on data 1 and beclo_R2_ aligned reads (BAM).bam.bai	HinfKW20_beclo_R2_HISAT2.bam.bai
HISAT2 on data 1 and pred_R3_ aligned reads (BAM).bam	HinfKW20_pred_R3_HISAT2.bam
HISAT2 on data 1 and pred_R3_ aligned reads (BAM).bam.bai	HinfKW20_pred_R3_HISAT2.bam.bai
featureCounts on data 12 and beclo_R2.tabular	HinfKW20_beclo_R2_fCounts.tsv
featureCounts on data 12 and pred_R3.tabular	HinfKW20_pred_R3_fCounts.tsv
DESeq2 result file on data 48, data 46, and others.tabular	DESeq2_results.tsv
DESeq2 plots on data 48, data 46, and others.pdf	DESeq2_plots.pdf
HinfKW20_genomic.gtf.gtf	HinfKW20_genomic.gtf

5.2.0 Loading your *HISAT2* alignments in IGV

5.2.1 Open up IGV and select [File > New Session](#).

5.2.1 If necessary, load your genome again from Module 1 using [Genomes > Load Genome from File](#) ≥ **HinfKW20_genomic.fna**.

5.2.2 Load your HISAT2 BAM files using File > Load from File. Multi-select your **HinfKW20_*_HISAT2.bam** files and the **HinfKW20_genomic.gtf** file as well.

5.2.3 Expand the **HinfKW20_genomic.gtf** track.

5.3.0 Explore your HISAT2 alignments

5.3.1 Use the Search bar to locate the feature [REDACTED]

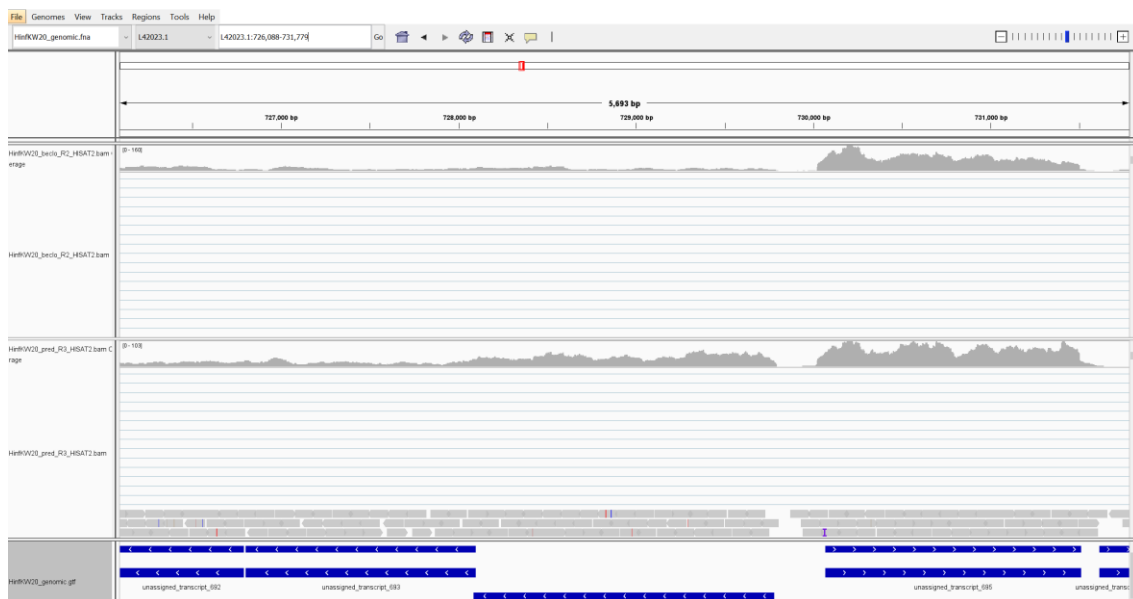
Zoom out to about 50% and notice that this gene appears to have considerably more reads aligned to it in both treatments than surrounding genes, suggesting high expression in both treatments.

5.3.2 Use the Search bar to locate the feature [REDACTED]

Zoom out to about 50% and notice that two adjacent transcripts appears to be differentially expressed between our two treatments. The prednisolone treatment appears to produce an upregulation in *H.influenzae* KW20 or beclomethasone may downregulate this gene. A proper untreated control and additional replicates would be used to help examine this effect further.

5.3.3 Use the Search bar to locate the feature [REDACTED]

Zoom out to about 50% and see another great example of differential expression. What do you think the adjusted p-value on this difference is according to DESeq2?



5.3.4 Save your session with File > Save Session > RNASeq.xml

6.0.0 Class summary

That concludes our third lecture and introduction to reference alignment and RNA-Seq analysis on the Galaxy platform. Next module we will begin our journey into exploring the command-line, it's basic commands and useful tools. Altogether we've explored the following in this module:

- Reference sequence alignment using Bowtie2.
- Converting BAM files into ***mpileup*** and ***VCF*** formats for analysis.
- Loading BAM and VCF files in IGV for visual analysis
- Generating RNA-Seq alignments to a reference genome with HISAT2.
- Producing raw read counts with featureCounts.
- Summarizing multiple datasets with MultiQC.
- Applying differential expression analysis with the DESeq2 tool.
- Reviewing and investigating HISAT2 RNA-Seq alignments and DESeq2 analyses with IGV.

6.1.0 Post-lecture assessment (10% of final grade)

Soon after this lecture, a homework assignment will be made available on Quercus in the assignment section. It will build on the ideas and/or data generated within this lecture. Each homework assignment will be worth 10% of your final mark. If you have assignment-related questions, please try the following steps in the order presented:

- Check the internet for a solution – read forums and learn to navigate for answers.
- Generate a discussion on Quercus outlining what you've tried so far and see if other students can contribute to a solution.
- Contact course teaching assistants or the instructor.

6.2.0 Suggested class preparation for Module 4

Next week we will begin exploring the Galaxy interface and learning how to perform quality control and genome assembly on data from sources like the SRA. To prepare for this, we suggest the following Coursera Modules:

- Command Line Tools for Genomic Data Science, **Lilana Florea, PhD**:
 - Module 1: Basic Unix Commands (1hr, 40mins)

