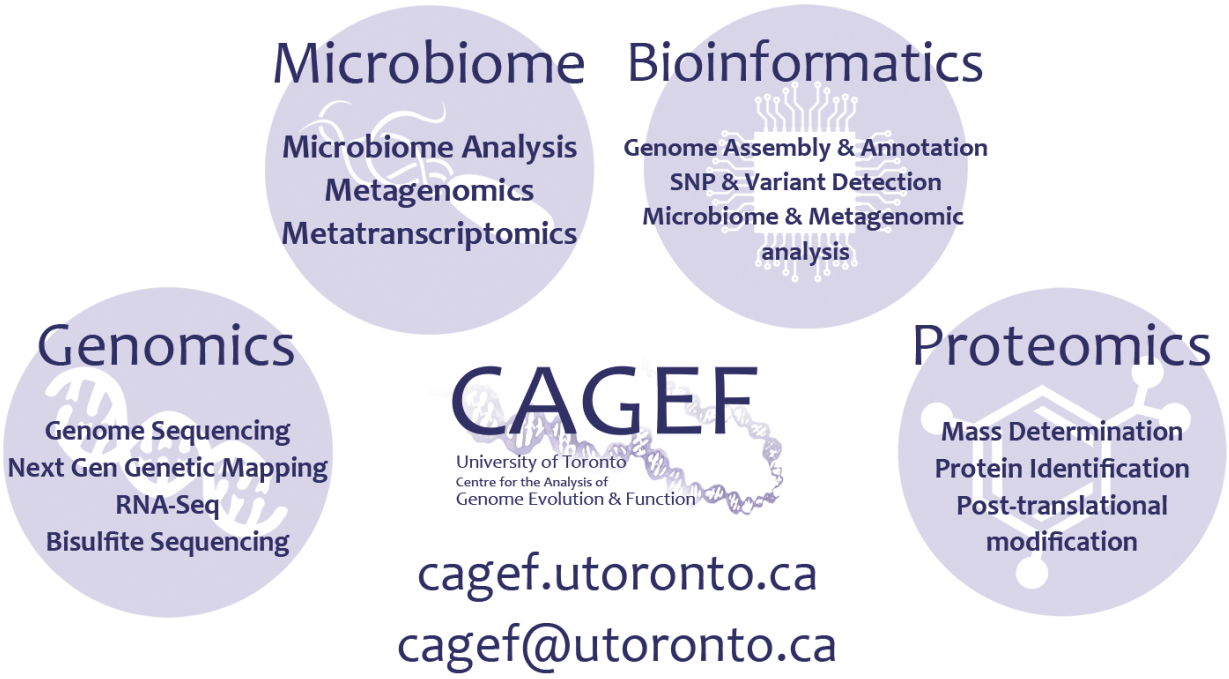


CSB1021HF LEC0131

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 CAGEF.

- 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:

- 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 usegalaxy.ca

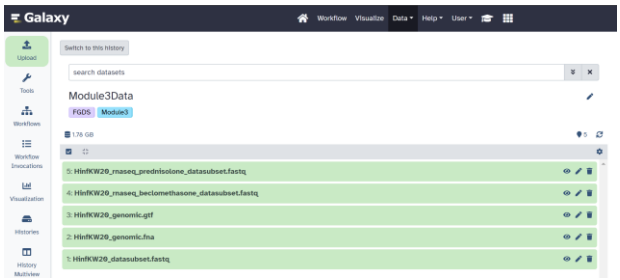
Recall from last week that we started working with this Canadian-based Galaxy server. You can use your University of Toronto login credentials here at https://usegalaxy.ca

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 *View* the shared data from the *menu > Data > Histories > Shared with me >* **Module3Data**. This shared history will contain the data from the above two data sets along with some additional data that we'll use in today's lecture.

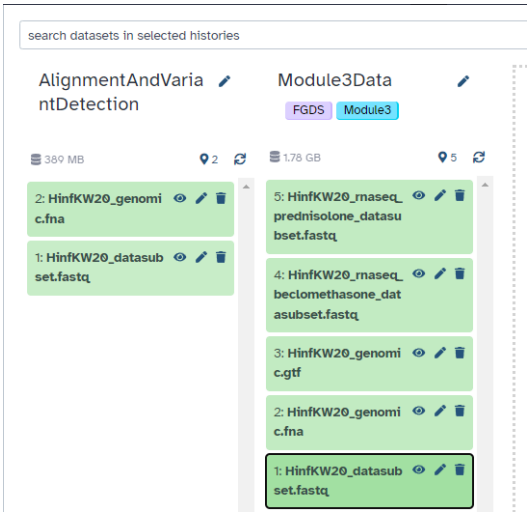


1.1.2 Import the history with the *Import this history* button on the upper-left side of the interface. Rename this imported history **Module3Data**.

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

1.1.4 Go to *History pane > Show Histories Side-bySide* and add the **Module3Data** history using the *Select histories* button.

1.1.5 One at a time, select the two datasets outlined in **1.1.0** from the **Module3Data** 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 **AssemblyandAnnotation** 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 *tools pane > Mapping > Bowtie2*.

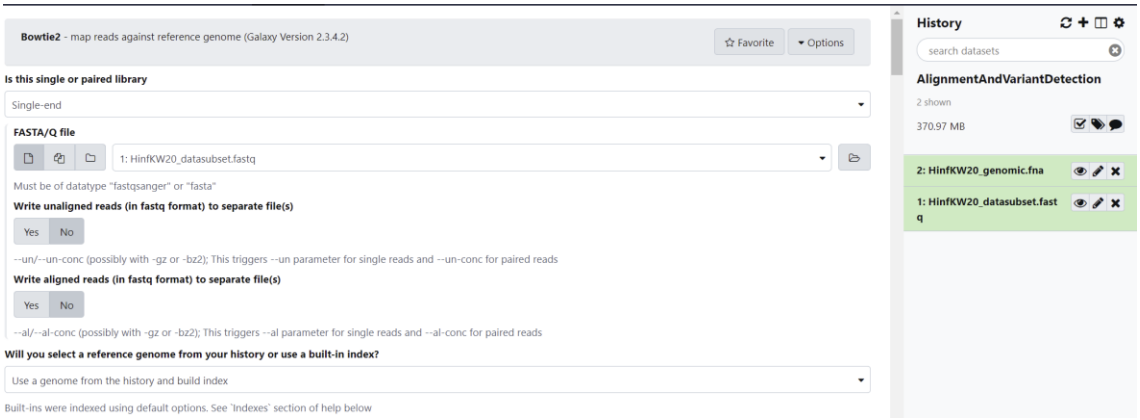
1.2.2 Read the overview for *Bowtie2*. You can revisit the remainder of the *Bowtie2* options later.

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. Proceed with "**No just use defaults**".

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

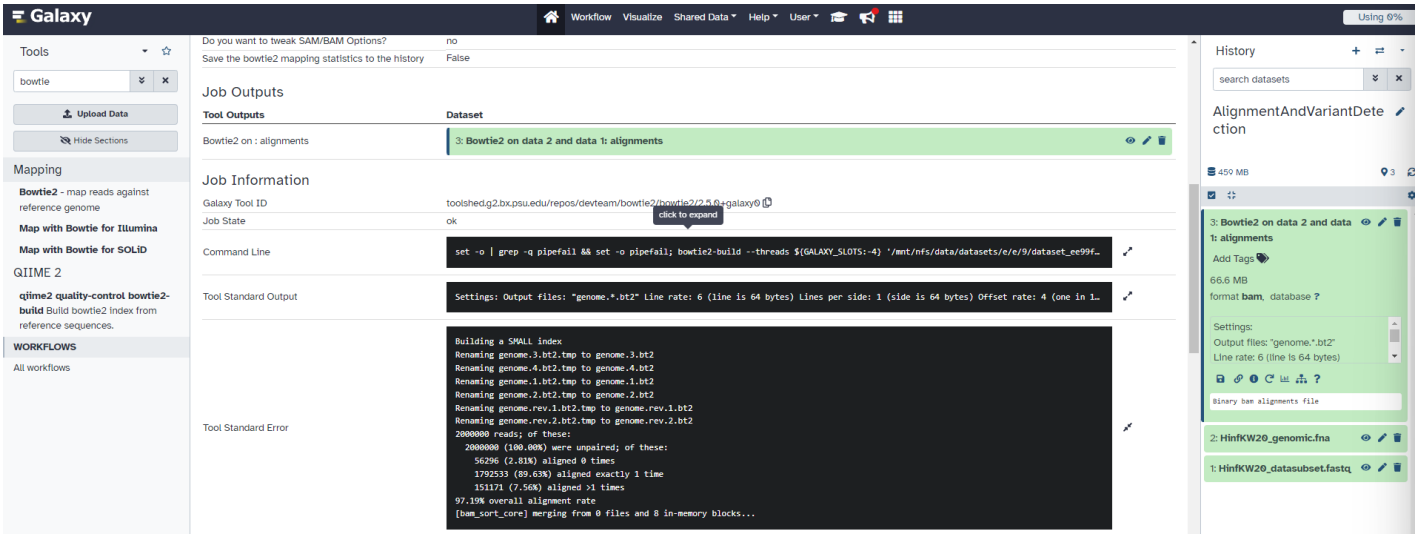
- Is this single or paired library: **Single-end**
- FASTA/Q file: **HinfKW20_datasubset.fastq**
- Will you select a reference genome from your history or use a built-in index? **Use a genome from the history and build index**
    - Select reference genome: **HinfKW20_genomic.fna**



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: alignments > Dataset details > Tool Standard Error*



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 (ie your genome name)
- 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 holding meta-information about the aligned read. For more information see the bowtie2 manual

QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	MRNM	MPOS	ISIZE	SEQ	QUAL	OPT
@HD VN:1.5 SO:coordinate											
@SQ SN:L42623.1 LN:1636136											
@PG ID:bowtie2 PN:bowtie2 VN:2.5.6 CL:"/usr/local/bin/bowtie2-align-s --wrapper basic-0 -p 8 -x genome -U input_f.fastq"											
SRR965262.311561.1	0	L42623.1	1	0	4M2I36M	*	0	0	NTTATGGCAATTAAAATTGGTATCAATGGTTTTGGTCGTATC	!CCCCCCCCCCCCCCCCCCCCCCCBCC?CCCCC?CCCBC@<	AS:i:-22 XN:i:0 XM:i:3 XO:i:1 XG:i:2 NM:i:5 MD:Z:6T0A1G36 YT:Z:UU
SRR965262.466237.1	0	L42623.1	1	0	4M1I37M	*	0	0	TTATGGCAATTAAAATTGGTATCAATGGTTTTGGTCGTATCG	BBBB?@@BCBBBABBCB?6@BA@@BB>BAB5:56/5659A	AS:i:-23 XN:i:0 XM:i:3 XO:i:1 XG:i:1 NM:i:4 MD:Z:1A9T6G37 YT:Z:UU
SRR965262.767623.1	0	L42623.1	1	42	42M	*	0	0	TATGGCAATTAAAATTGGTATCACTGGTTTTGGTCGTATCGG	BBB?6A@@BB@A>>BB9>=@A=;?;91@@A1;96######	AS:i:-2 XN:i:0 XM:i:1 XO:i:0 XG:i:0 NM:i:1 MD:Z:23A18 YT:Z:UU
SRR965262.961686.1	0	L42623.1	1	42	42M	*	0	0	TATGGCAATTAAAATTGGTATCAATGGTTTTGGTCGTATCGG	B4;@?>>;?A>777<;68=/169=c89=<3:99<754245	AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:42 YT:Z:UU
SRR965262.171666.2	0	L42623.1	2	42	42M	*	0	0	ATGGCAATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGC	BACABCCCCCCC@BBB@A@C@<4<BB>CC@BABB?6?B@9B?	AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:42 YT:Z:UU
SRR965262.237919.1	0	L42623.1	2	42	42M	*	0	0	ATGGCAATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGC	BCA@ABCCCCBBBCCB<@CCBCBC=3<CBB=;=A<AACAA>A	AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:42 YT:Z:UU
SRR965262.252366.1	0	L42623.1	3	42	42M	*	0	0	TGGCAATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGCC	=9BB8<CBBb=?AAB=@=<BAB@!7BC;9)<A?(=<?6?=A	AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:42 YT:Z:UU
SRR965262.625376.2	0	L42623.1	3	42	42M	*	0	0	TGGCAATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGCC	BBCCBBCCB==9BC@6%7AC??AB@!7BCCB1BA6?>C?>BC	AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:42 YT:Z:UU
SRR965262.684741.1	0	L42623.1	4	42	42M	*	0	0	GGCAATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGCCG	BBBBBCC@?ABCCBB>BCBABBBAACBC=25A??@@B@<?>?	AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:42 YT:Z:UU
SRR965262.441416.1	0	L42623.1	5	42	42M	*	0	0	GCAATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGCCGT	?CBBCCABAABB3;?@BB@AB9>>B@=9-><>@BA99@>A>	AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:42 YT:Z:UU
SRR965262.33637.2	0	L42623.1	6	42	42M	*	0	0	CAATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGCCGGA	B@BCBBCB>CBCC:BBB?BBBC@BBBBB>@A??B@BB?=@##	AS:i:-2 XN:i:0 XM:i:1 XO:i:0 XG:i:0 NM:i:1 MD:Z:46T1 YT:Z:UU
SRR965262.658681.1	0	L42623.1	7	42	42M	*	0	0	AATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGCCGTAT	BCCCBBBBCB?<BACBBBB??BCC@)6>@?@BC>3=A?=4?	AS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:42 YT:Z:UU
SRR965262.138219.1	0	L42623.1	8	42	42M	*	0	0	ATTAAAATTGGTATCAATGGTTTTGGTCGTATCGGCCGGTTC	BCCBCCCCCBCBCCBBBCCBBCCC@??BBBBBBCBBA?####	AS:i:-4 XN:i:0 XM:i:2 XO:i:0 XG:i:0 NM:i:2 MD:Z:36T0A2 YT:Z:UU

Now that we've generated our first reference alignment, it's time to proceed to variant calling!

SAM and BAM: Sequence alignment/map (SAM) and binary alignment/map (BAM) are common file formats in bioinformatics that are produced as the result of a sequence alignment and originated in 2009 from Heng Li and Bob Handsaker et al. It has since expanded to include various features and is the standard for storing mapped data results across various pipelines and platforms.

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 was briefly described in Module 1.

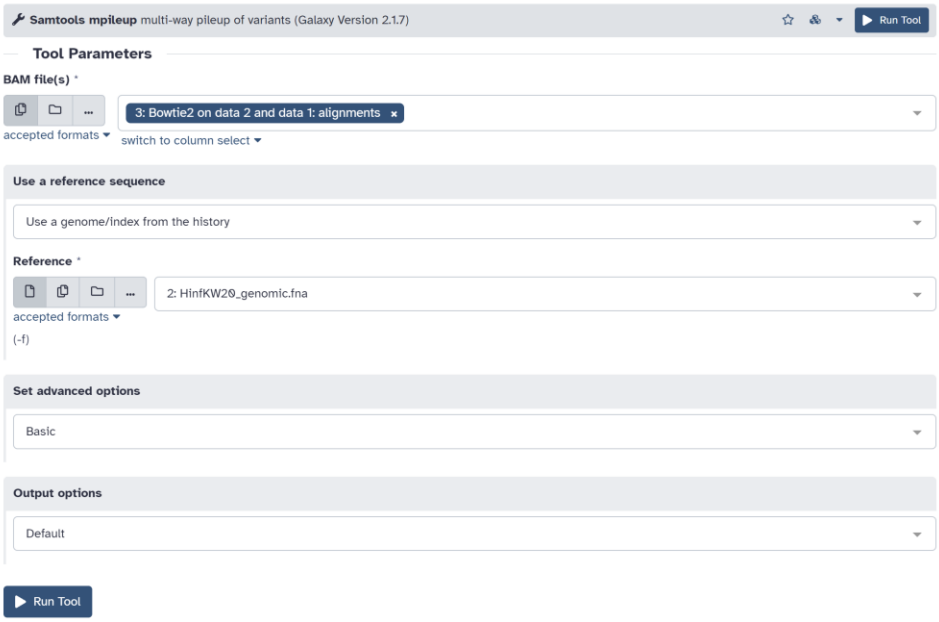
2.1.0 Generate an *mpileup* of your alignment

2.1.1 Go to *tools pane > SAM/BAM >* ***samtools mpileup*** ***multi-way pileup of variants***.

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:

- BAM file(s): **Bowtie2 on data 2 and data 1: alignments**
- Choose the source for the reference genome: **Use a genome/index from the history**
- 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 > Display* icon. This will bring up a table version of the mpileup on Galaxy. The 6-column format represents the following:

1. Reference chromosome/contig
2. Reference position
3. Reference base
4. Total reads covering the site
5. String of all aligned bases from overlapping fragments

Category	Forward (+) direction fragment	Reverse (-) direction fragment
Start of a new fragment * BAQ = Base alignment quality. Calculate ASCII value – 33 to obtain the score.	^[ASCII-BAQ] eg ^!	^[ASCII-BAQ] eg ^K
End of a fragment	$	$
Correct match with reference base	.	,
Mismatch with reference, showing the mismatch base	ACGTN	acgtn
Insertion with the inserted sequence.	+N[ACGTN]	+N[acgtn]
Deletion with the deleted sequence	-N[ACGTN]	-N[acgtn]

6. String of base qualities (ie Phred scores) for each fragment at that position

Column 1	Column 2	Column 3	Column 4	Column 5	Column 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?CCCCCCCCCB
L42023.1	10	T	15	.....^K.	CBBACCBCCCBCCBA
L42023.1	11	A	16	.....^K.	CB@>CCBB@ABBBCB?
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?BCBCCCCCBBCBCCCB
L42023.1	16	T	24	.....^K.	CCB<BCACCBBBCCCB@BBCCBBB
L42023.1	17	G	25	.....^K.	CB9;BBA@B3C?B=9BA?BCBBBCB
L42023.1	18	G	25	.....	C?>:@<B0B;C<C@9BB>C?BAABA

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 presence 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 to *SAMtools* called *BCFtools*. The purpose of this package is to handle variant calling and the manipulation of variant data.

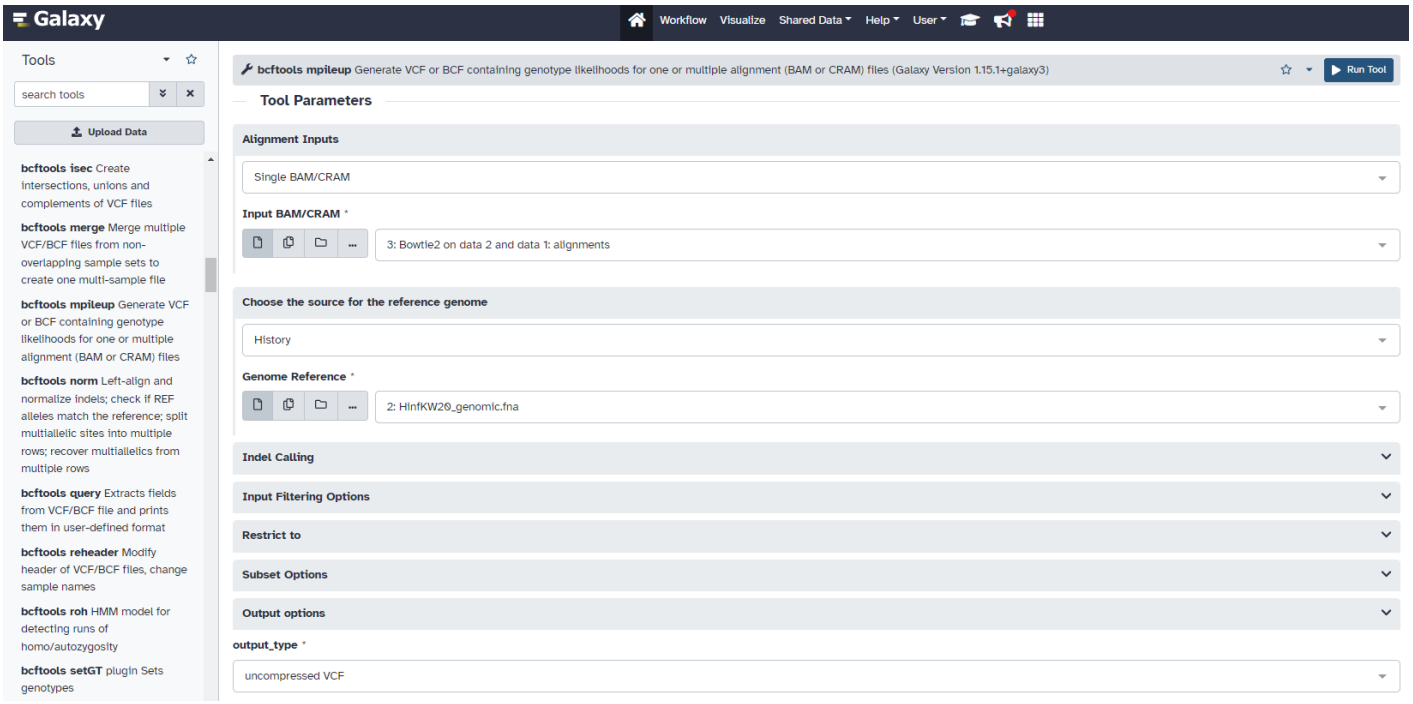
2.3.1 Click on *tools pane* > *Variant Calling* > ***bcftools mpileup***

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

Note that despite its 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: alignments**
- Choose the source for the reference genome: **History**
    - Genome Reference: **HinfKW20_genomic.fna**
- Output_type: **uncompressed VCF**
- All other settings: **default**



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 > Display* 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 a final single # 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 metrics. The codes describe the type of data and its order in the remaining columns.
10. **Sample_name**: Genotype data specific to metric(s) if specified in **FORMAT**.

* Missing values are specified with a dot (.). See VCF 4.2 specifications section 1.4.2 for more genotype information.

##INFO=<ID=IMF,Number=1,Type=Float,Description="Maximum fraction of raw 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=RPBZ,Number=1,Type=Float,Description="Mann-Whitney U-z test of Read Position Bias (closer to 0 is better)">

##INFO=<ID=MQBZ,Number=1,Type=Float,Description="Mann-Whitney U-z test of Mapping Quality Bias (closer to 0 is better)">

##INFO=<ID=BQBZ,Number=1,Type=Float,Description="Mann-Whitney U-z test of Base Quality Bias (closer to 0 is better)">

##INFO=<ID=MQSBZ,Number=1,Type=Float,Description="Mann-Whitney U-z test of Mapping Quality vs Strand Bias (closer to 0 is better)">

##INFO=<ID=SCBZ,Number=1,Type=Float,Description="Mann-Whitney U-z test of Soft-Clip Length Bias (closer to 0 is better)">

##INFO=<ID=FS,Number=1,Type=Float,Description="Phred-scaled p-value using Fisher's exact test to detect

scaled

##INFO=<ID=SGB,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;I16=3,0,0,0,82,2434,0,0,84,3528,0,0,0,0,0,0;QS=1,0;FS=0;MQ0F=0.25
L42023.1	2	.	A	<*>	0	.	DP=6;I16=4,0,0,0,118,3628,0,0,168,7056,0,0,2,2,0,0;QS=1,0;FS=0;MQ0F=0
L42023.1	3	.	T	<*>	0	.	DP=8;I16=6,0,0,0,186,5818,0,0,252,10584,0,0,6,10,0,0;QS=1,0;FS=0;MQ0F=0
L42023.1	4	.	G	<*>	0	.	DP=9;I16=7,0,0,0,217,6795,0,0,294,12348,0,0,12,28,0,0;QS=1,0;FS=0;MQ0F=0

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 such as their likelihood, and we only want to view ***non-reference variants*** in the case of output from *bcftools mpileup*.

2.5.1 Click on *tools pane > Variant Calling >* ***bcftools call*** **SNP/indel variant calling from VCF/BCF**

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:

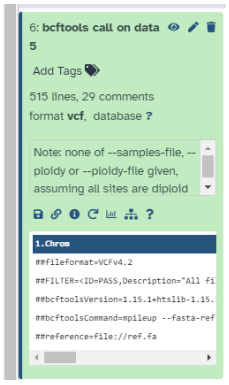
- VCF/BCF Data: **bcftools mpileup on data 2 and data 3**
- Input/output Options
  - Output variant sites only: **Yes**
- output_type: **uncompressed VCF**
- All other settings: **default**

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

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 1 Input.vcf.gz; Date=Mon Jun 3 16:16:46 2024

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO
L42023.1	1056	.	CG	CGG	80.2092	.	INDEL;IDV=5;IMF=0.5;DP=10;VDB=0.604698;SGB=-0.616816;RPBZ=-1.5667;MQBZ=-2.886
L42023.1	3808	.	T	C	225.417	.	DP=34;VDB=0.0471191;SGB=-0.693132;MQSBZ=-0.476756;FS=0;MQ0F=0;AC=2;AN=2;DP4
L42023.1	20891	.	AG	AGG	162.172	.	INDEL;IDV=39;IMF=0.886364;DP=44;VDB=0.568577;SGB=-0.693144;RPBZ=-0.722435;M
L42023.1	29688	.	A	T	225.417	.	DP=38;VDB=0.411578;SGB=-0.693143;MQSBZ=0.750424;FS=0;MQ0F=0;AC=2;AN=2;DP4=
L42023.1	32329	.	GAAAAAA	GAAAAAAA	135.341	.	INDEL;IDV=17;IMF=0.708333;DP=24;VDB=0.237631;SGB=-0.690438;RPBZ=-1.11435;MQB
L42023.1	36715	.	G	A	228.413	.	DP=43;VDB=0.832997;SGB=-0.693146;RPBZ=-0.887444;MQBZ=2.11693;MQSBZ=-2.12656
L42023.1	45496	.	G	A	225.417	.	DP=29;VDB=0.737192;SGB=-0.693079;MQSBZ=-0.670288;FS=0;MQ0F=0;AC=2;AN=2;DP4
L42023.1	46361	.	A	T	225.417	.	DP=32;VDB=0.0499261;SGB=-0.69312;MQSBZ=2.29797;FS=0;MQ0F=0;AC=2;AN=2;DP4=0
L42023.1	47284	.	A	T	225.417	.	DP=96;VDB=0.377221;SGB=-0.693147;MQSBZ=0.925737;FS=0;MQ0F=0;AC=2;AN=2;DP4=
L42023.1	52066	.	CAAAA	CAAAAA	210.292	.	INDEL;IDV=56;IMF=0.823529;DP=68;VDB=0.0587182;SGB=-0.693147;RPBZ=-2.7777;MQB
L42023.1	52337	.	G	A	225.417	.	DP=77;VDB=0.152445;SGB=-0.693147;MQSBZ=-2.9262;FS=0;MQ0F=0;AC=2;AN=2;DP4=0,
L42023.1	52396	.	G	A	225.421	.	DP=48;VDB=0.444947;SGB=-0.693147;MQSBZ=-2.6082;FS=0;MQ0F=0;AC=2;AN=2;DP4=(



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 *Select Items* (checkmark) icon.

2.6.2 Select the following datasets.

- *Bowtie2* on data 2 and data 1: alignments
- *bcftools call* on data 5

2.6.3 Select *History pane > 2 of 6 selected > Build Dataset List*.

2.6.4 Name the dataset list **RefAlignDownload** and uncheck the *Hide original elements?* box

2.6.5 Use the *Create collection* button.

2.6.6 Select *History pane > RefAlignDownload > Download*.

2.6.7 Save the .zip file to **~/FGDS/Module3/downloads** (~68 Mb).

2.6.8 Decompress the contents of your .zip file to **~/FGDS/Module3**.

2.6.9 Rename your files using the following table

Download name	New name
Bowtie2 on data 2 and data 1_ alignments.bam	HinfKW20_refalign.bam
Bowtie2 on data 2 and data 1_ alignments.bam.bai	HinfKW20_refalign.bam.bai
bcftools call on data 5	HinfKW20_refalign.VCF

2.7.0 Save your history as a workflow

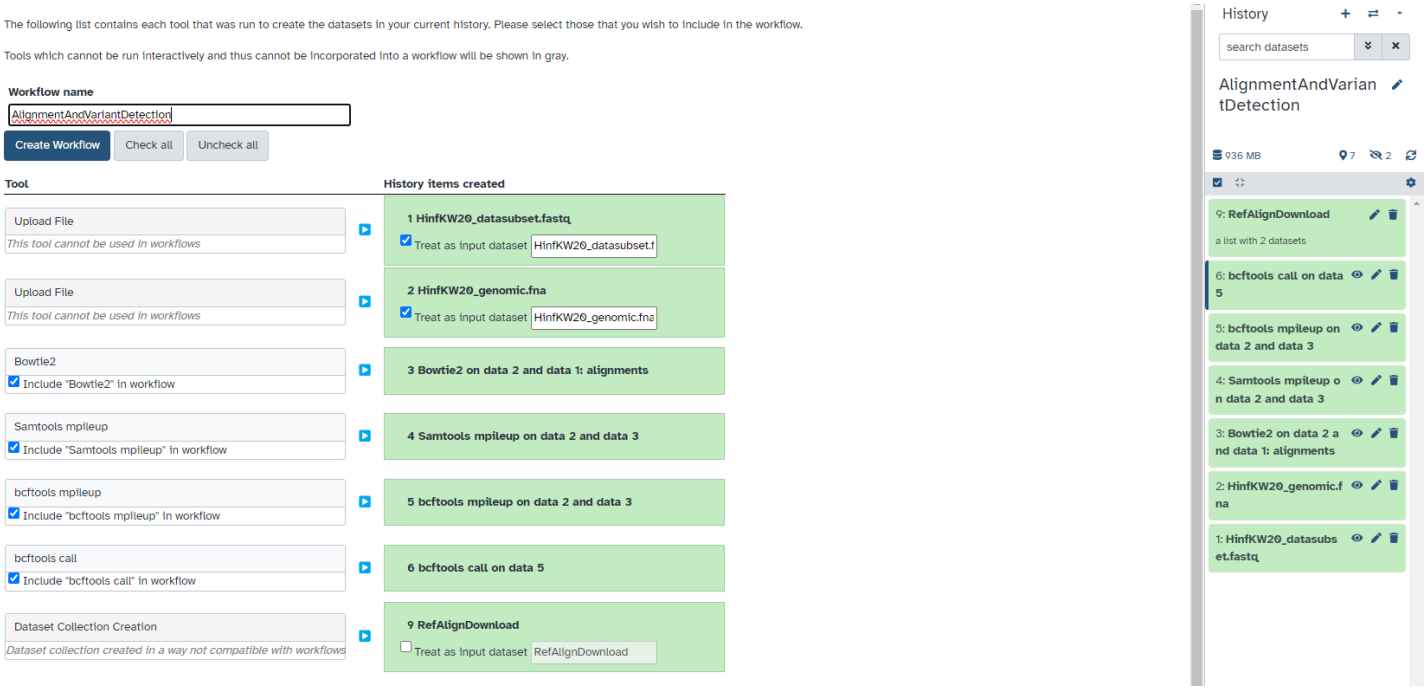
Now that you've completed your first reference alignments and variant calling analysis, you can save your history as a new workflow for future use. You can come back and edit the finer details of the workflow at a later time.

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 **Data List** (**RefAlignDownload**) you've created so that it won't become part of the Workflow.

2.7.4 Once you are ready, select *Create Workflow*.



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**.

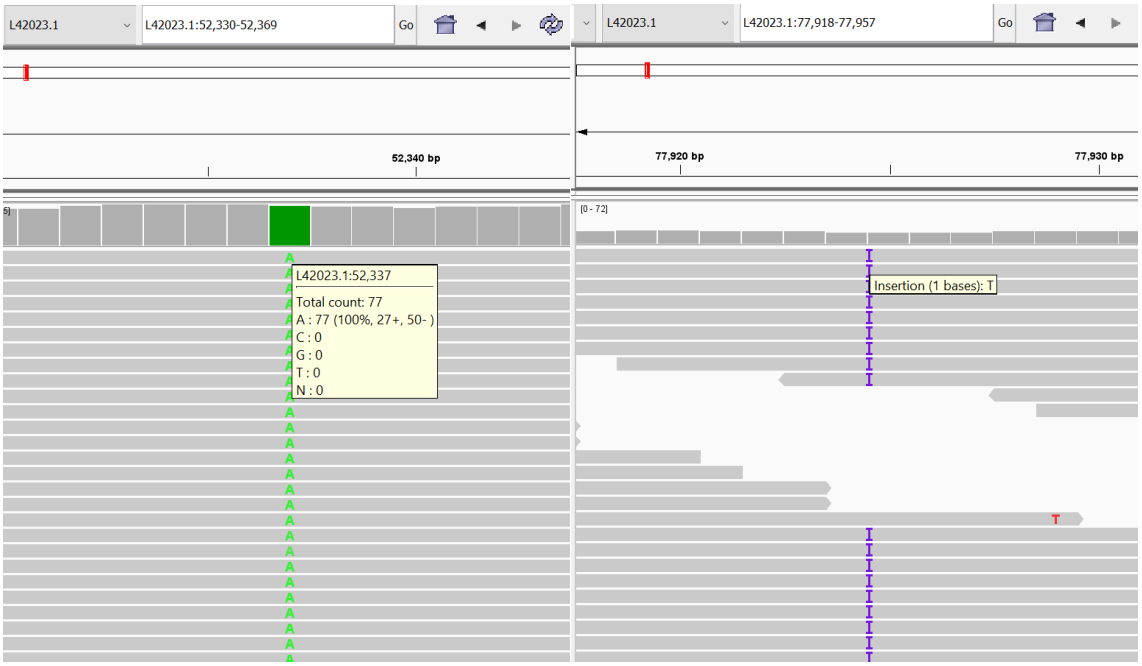
3.1.2 Load your BAM file generated in this module using *File > Load from File >* **HinfKW20_refalign.bam**

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 52,337.

3.1.4 In the IGV search box, input *L42023.1:52,337*. IGV will automatically convert this into a 41bp range centred on your point of interest.

3.1.5 Search for an INDEL site at *L42023.1:77,924*. 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. The error provided leads us to a possible problem with line 69.

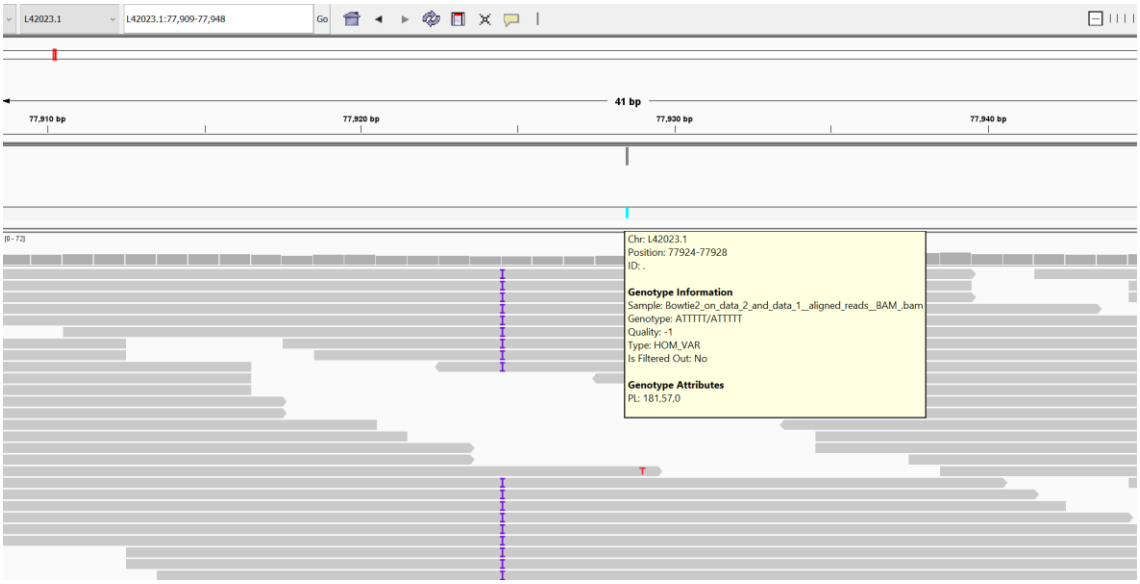
3.2.2 Load your HinfKW20_filtered.vcf in a text editor and search for line 69. Delete this entry.

3.2.3 Subsequently, delete line 388 at position 1289372.

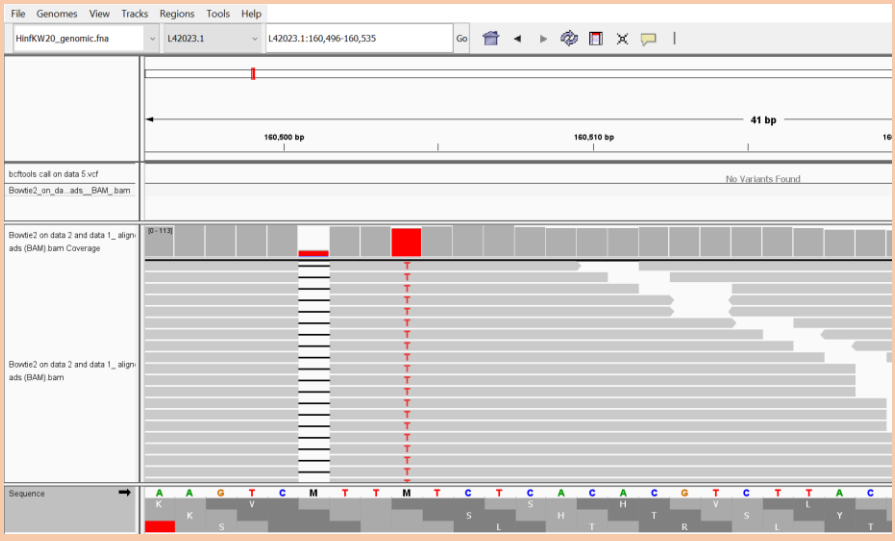
3.2.4 Save the file as **HinfKW20_filtered_edit.vcf**.

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

3.2.6 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.



Not all genomes assemblies are perfect! Recall that our genome assemble for *H. influenzae* originates from the NCBI. Regardless of where we downloaded it, you may find that this file is not necessarily compatible with other programs as we see in the results of our *bowtie2* alignment. Other IUPAC base symbols like K or M, may result in slightly offset or incorrect alignments of reads. The resulting BAM and downstream VCF file created a few entries that IGV could not interpret. What do you think is happening at 160,501 in our reference genome?



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 related vein as aligning whole genome sequencing reads to a reference genome is the aligning of RNA-Seq reads to a reference sequence. 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 a 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**.

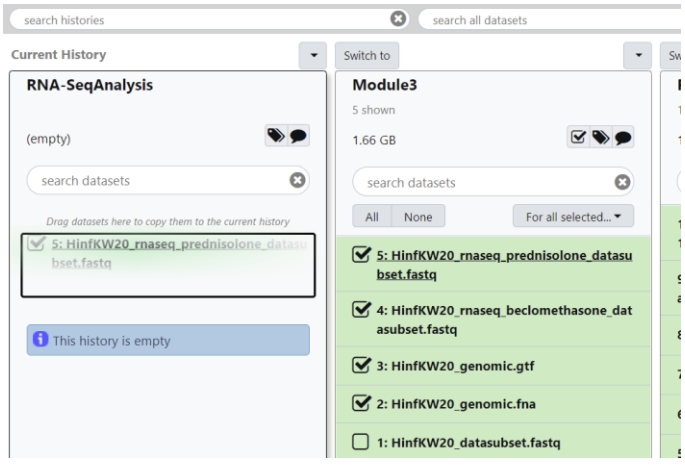
4.1.1 Go to *History pane > Create new history* and name the history **RNA-SeqAnalysis**.

4.1.2 Go to *History pane > Show Histories Side-by-Side*.

4.1.3 Use the **Select histories** button to choose your **RNA-SeqAnalysis** and **Module3Data** histories for viewing.

4.1.4 From the **Module3Data** history, select the *Operation on multiple datasets* icon and select the following history entries:

- 2: HinfKW20_genomic.fna
- 3: HinfKW20_genomic.gtf
- 4: HinfKW20_rnaseq_beclomethasone_datasubset.fastq
- 5: HinfKW20_rnaseq_prednisolone_datasubset.fastq



4.1.5 Drag and drop the items into your RNA-SeqAnalysis history.

4.1.6 Return to the Galaxy homepage.

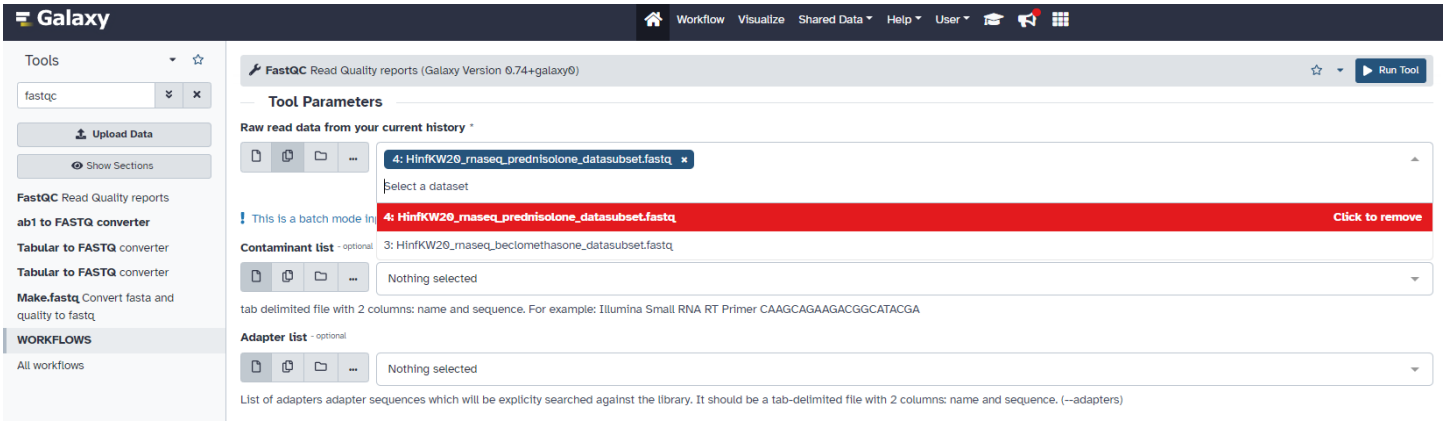
4.2.0 Assess the quality of your RNA-Seq data with *FastQC*

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 (dual folder icon)**
  - Select both RNA-seq datasets by clicking on them.
- Other options: **default**



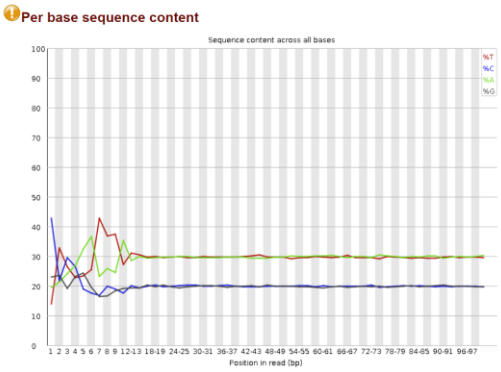
4.3.0 Review the RNAseq FastQC output

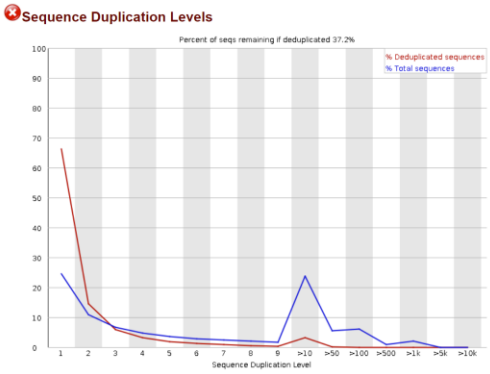
4.3.1 Review the HTML output from both data files by selecting the *View data* icon on the history entry.

- **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 and/or hexamer 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. A well-made library should not have much in the way of overduplication of fragments. 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.





Overrepresented sequences

Sequence	Count	Percentage	Possible Source
CTTCGATCCTCAAACGGTGGCTTCCACACGATCTCGTTTTGGTTTGACTC	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
CAATCGCTGGTTTATTGAAGCCCTTAACCGTATTTATACGACCTAGTGGG	2520	0.126	No Hit
CGCGGAGTCAAACCAAAACGAGATCGTGTGGAAGCCACCGTTTGAGGATC	2471	0.12355000000000001	No Hit
CTCGTTTTGGTTTGACTCCGCGTTATCCCCGTCTTACGAGCGGAAGCTGG	2016	0.1008	No Hit

4.4.0 Align your RNA-Seq reads with *HISAT2*

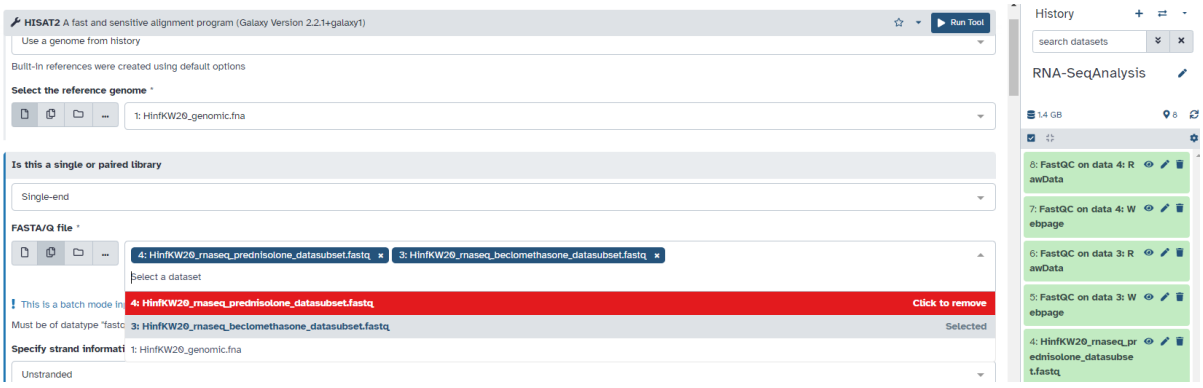
Now that we've looked at the quality of our reads, we can align them to a genome to create a BAM file. This is very much like our previous reference alignment. We'll even use the same reference sequence ***but*** now we will only achieve alignment to the coding regions of the genome (assuming no genomic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

4.4.1 Select *tools pane > Mapping >* ***HISAT2 A fast and sensitive alignment program***.

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**
- FASTA/Q file: **Multiple datasets (dual folder icon)**
  - Select both RNA-Seq fastq datasets by clicking on them.
- Other options: **default**



4.4.4 Note that *HISAT2* will create a BAM file for each RNA-Seq dataset which you will rename using the *History pane > Edit attributes* button (pencil icon).

- After renaming a dataset attribute, be sure to use the *Save* icon!

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 Select *tools pane > RNA-seq >* ***featureCounts*** ***Measure gene expression in RNA-Seq experiments from SAM or BAM files***.

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 (dual folder icon)**
  - Select both BAM datasets using multi-select (checkbox selection).
- Gene annotation file: **A GFF/GTF file in your history**
  - Gene annotation file: **HinfKW20_genomic.gtf**
- GFF feature type filter: **gene**
- GFF gene identifier: **gene_id**
- Output format: **Gene-ID "\t" read-count (MultiQC/DESeq2/edgeR/limma-voom compatible)**
- 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 > Display*.

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: Counts > Display*.

Here we can see a simple tabular format with the GeneID and read counts listed. If we had chosen the *featureCounts* 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 1 and pred_R3: aligned reads (BAM)
Geneid	HISAT2 on data 1 and pred_R3: aligned reads (BAM)
HI_0001	5475
HI_0002	823
HI_0003	194
HI_0004	151
HI_0005	137
HI_0006	1318
HI_0007	360
HI_0008	267
HI_0009	74
HI_0010	38
HI_0011	127
HI_0012	80
HI_0013	228
HI_0014	159
HI_0015	544
HI_0016	860
HI_0017	2700
HI_0018	106
HI_0019	1047
HI_0020	1305

Geneid	Chr	Start	End	Strand	Length	HISAT2 on data 1 and pred_R3 aligned reads (BAM)
Geneid	Chr	Start	End	Strand	Length	HISAT2 on data 1 and pred_R3 aligned reads (BAM)
HI_0001	L42023.1	2	1021	+	1020	5475
HI_0002	L42023.1	1190	3013	+	1824	823
HI_0003	L42023.1	3050	3838	-	789	194
HI_0004	L42023.1	3854	4318	-	465	151
HI_0005	L42023.1	4579	5391	-	813	137
HI_0006	L42023.1	5656	8748	+	3093	1318
HI_0007	L42023.1	8750	9688	+	939	360
HI_0008	L42023.1	9681	10397	+	717	267
HI_0009	L42023.1	10467	11375	+	909	74
HI_0010	L42023.1	11414	11854	-	441	38
HI_0011	L42023.1	11857	12261	-	405	127
HI_0012	L42023.1	12367	13359	+	993	80
HI_0013	L42023.1	13423	14331	-	909	228
HI_0014	L42023.1	14328	15011	-	684	159
HI_0015	L42023.1	15013	16062	-	1050	544
HI_0016	L42023.1	16071	17867	-	1797	860
HI_0017	L42023.1	18035	18418	-	384	2700
HI_0018	L42023.1	18676	19335	+	660	106
HI_0019	L42023.1	19405	20829	-	1425	1047
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: Counts	featureCounts on data 2 and beclo_R2: Counts
featureCounts on data 2 and data 9: Summary	featureCounts on data 2 and beclo_R2: Summary
featureCounts on data 2 and data 10: Counts	featureCounts on data 2 and pred_R3: Counts
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 need 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 *menu > Data > Histories > Shared with me*

4.7.2 Select *Module3_replicates* > *View*.

4.7.3 You will see 4 additional sets of *featureCounts* data and summaries that you can import using the *Import this history* button on the upper-left. Import this as **Module3_replicates** and then *Copy History*

4.7.4 Select *History pane > Show Histories Side-by-Side*.

4.7.5 Use the *Select histories* link and add the **Module3_replicates** history and then the *Change selected* button.

4.7.6 *Switch to* **RNA-SeqAnalysis.**

4.7.7 From the **Module3_replicates** history drag and drop all 8 history entries to your **RNA-SeqAnalysis** history. (*Select Items > Select All > Drag and drop*)

4.7.8 Return to the Galaxy homepage.

4.8.0 Use MultiQC to quickly summarize multiple datasets

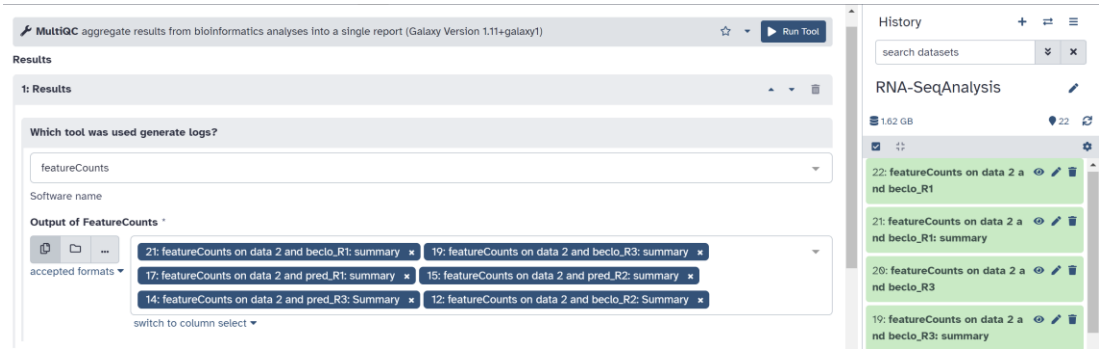
When you are working with many conditions and their replicates, as would be the case with RNA-Seq analysis, it is helpful to have a visual summary of all the samples in a single place. *MultiQC* can accomplish that for you by summarizing data/results across your many datasets. In this case, we will summarize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

4.8.1 Select *tools pane > Quality Control > **MultiQC** **aggregate results from bioinformatics analyses into a single report***

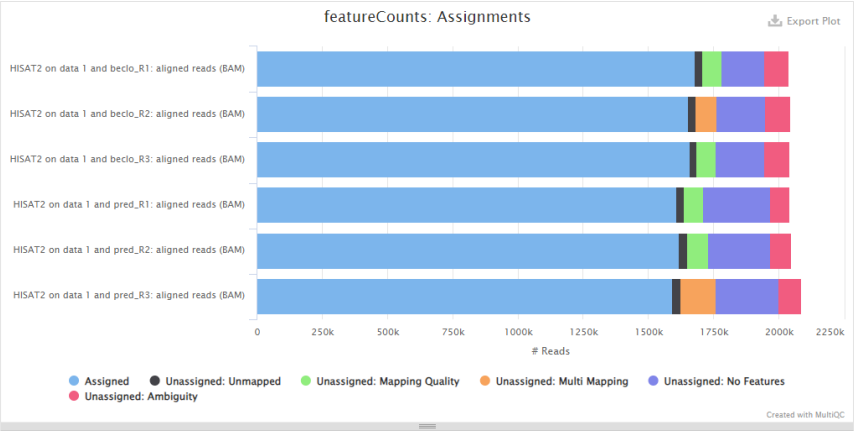
4.8.2 Scroll down to review the purpose and input/output information for *MultiQC*.

4.8.3 Execute *MultiQC* with the following settings:

- Results > Which tool was used [to] generate logs?: **featureCounts**
- Software name > Output of FeatureCounts:
    - Select all 6 *featureCounts* ***summary*** datasets using multi-select (checkboxes).
- Report title: **featureCounts summary analysis**
- Other options: **default**



4.8.4 Review the output of *MultiQC* using *History pane > MultiQC on data 21, data 19, and others: Webpage > Display*. 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:

- How: **Select datasets per level**
- Factor
  - Specify a factor name: **treatment**
- 1: Factor level
  - Specify a factor level: **beclomethasone**
  - Counts files(s): **featureCounts on data 2 and beclo_R***
- 2: Factor level
  - Specify a factor level: **prednisolone**
  - Counts files(s): **featureCounts on data 2 and pred_R***
- Files have header? **Yes**
- Choice of input data: **Count data (eg from HTSeq-count, featureCounts or StringTie)**
- Visualizing the analysis results: **Yes**
- Output options: (to select)
  - Generate plots for visualizing the analysis results
  - Output normalized counts

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**.

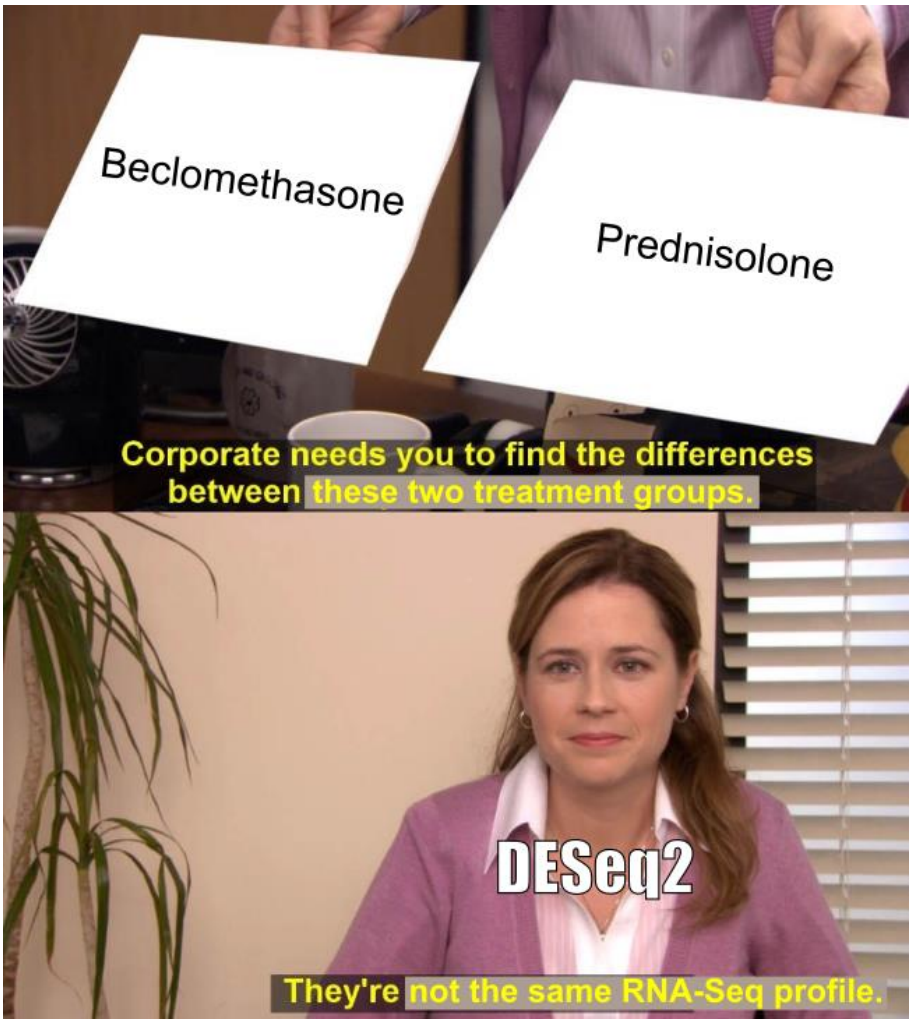
4.10.0 Review your *DESeq2* output

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 > Display* on the **DESeq2 plots** entry.

4.10.2 View a table of the log_2 fold-change data along with p-values by selecting *History pane > Display* 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	7725.19388239346	-2.07496858433065	0.184818219972345	-11.2222684745811	3.17247318066464e-29	5.494723548911l5e-26
HI_0809	17335.4808907276	-2.09712088839906	0.191662696307272	-10.9417269442822	7.27998161729792e-28	6.30446356097999e-25
HI_0092	536.611411032847	-2.09028358427851	0.205657689112388	-10.1638970723638	2.87355743120654e-24	1.65960049028324e-21
HI_0017	2610.17417253772	-1.87142885154697	0.18677039129907	-10.019943945774	1.24573577674043e-23	5.394035913286l4e-21
HI_0053	362.424034449593	-1.4824096952944	0.149486966884383	-9.91704647215034	3.50986633041137e-23	1.21582462485l5e-20
HI_0685	683.629761148889	-2.33648815765456	0.237763618648482	-9.8269620352899	8.61792761092216e-23	2.48770826381953e-20
HI_0181	700.490731088757	-1.29826586927887	0.144783870802957	-8.96692333254256	3.04912207325059e-19	7.54439918695718e-17
HI_0524	3046.15290592059	-0.969996011967862	0.109740596797175	-8.75697818323325	2.00554456809121e-18	4.34200398991748e-16
HI_0813	929.271389708606	-1.09066023326205	0.117330593792476	-8.52855338797598	1.48189105649891e-17	2.85181701095124e-15
HI_0564	646.494429120175	-1.55946293545802	0.183251274510035	-8.50997047429882	1.7397765109165e-17	3.01329291690738e-15
HI_0131	6129.53020763956	-1.0506396049579	0.127584537589607	-8.2348505924552	1.79781865037277e-16	2.83074716404l5e-14
HI_0812	2618.62313590619	-0.810217670857363	0.100185099385704	-8.08721460251696	6.10446543737514e-16	8.81077844794479e-14
HI_1331	294.555414773307	1.41040250924426	0.175052280878355	8.05703583619318	7.81667398649228e-16	1.04142148804651e-13
HI_0629	5269.62280105302	-1.01450491027716	0.128456118473765	-7.89767682793835	2.84149729665261e-15	3.515338084143l7e-13
HI_0822	12494.3016655775	-1.29385678238879	0.164203144010445	-7.87961028508985	3.28403727761074e-15	3.79196837654787e-13
HI_1048	669.264937651005	1.21734481649975	0.156186089321349	7.79419487221487	6.48204456355488e-15	7.01681324004816e-13
HI_0091	301.646477004857	-1.65907335783803	0.214106482920913	-7.74882355360938	9.27477872851898e-15	9.44936279870286e-13
HI_0098	526.204246279717	1.47171312727952	0.193058590269484	7.62314241330882	2.47573478917033e-14	2.38220703046834e-12
HI_0751	2382.38522921062	-0.80509987097279	0.106346033069918	-7.56972166929375	3.74024722655599e-14	3.46953062968157e-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 S*elect Items* 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]: Counts [2 entries]
- DESeq2 result file [1 entry]
- DESeq2 plots [1 entry]

5.1.3 Select *History pane > 7 of 27 selected > Build Dataset List*.

5.1.4 Name the dataset list **RNA-SeqDownload** and uncheck **Hide original elements?** box.

5.1.5 Use the *Create list* button.

5.1.6 Unselect the *Select Items* icon.

5.1.7 Select *History pane > RNA-SeqDownload > Download*.

5.1.8 Save the .zip file to **~/FGDS/Module3/downloads** (~215 Mb).

5.1.9 Decompress the contents to **~/FGDS/Module3**.

5.1.10 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: Counts.tabular	HinfKW20_beclo_R2_fCounts.tsv
featureCounts on data 12 and pred_R3: Counts.tabular	HinfKW20_pred_R3_fCounts.tsv
DESeq2 result file on data 22, data 20, and others.tabular	DESeq2_results.tsv
DESeq2 plots on data 22, data 20, 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]

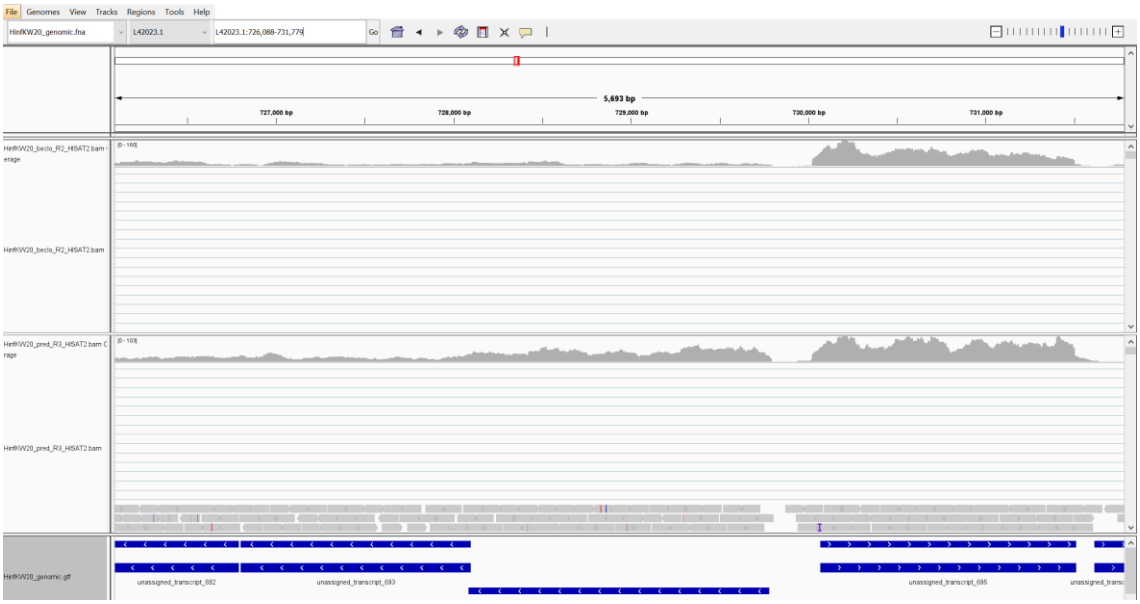
Zoom out to about 50% and notice that this gene appears to have considerably more reads aligned to it in both treatments than surrounding genes, suggesting high expression in both treatments.

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

Zoom out to about 50% and notice that two adjacent transcripts (HI_1457, and HI_1458) appear to be differentially expressed between our two treatments. In HI_1457, 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, its 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 *IVG* 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* package.
- Reviewing and investigating *HISAT2* RNA-Seq alignments with *IGV*.

6.1.0 Post-lecture assessment (9% 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 9% 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 command-line interface and learning navigate our way around it. 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)

6.3.0 Acknowledgements

This course was originally conceived and produced by Dr. David Guttman and Dr. Marcus Dillon.

- **Revision 1.0.0**: edited and prepared by David Guttman, Ph.D. and Marcus Dillon, Ph.D.
- **Revision 2.0.0**: edited and prepared for *CSB1021HF LEC0131, 10-2021* by Calvin Mok, Ph.D., Education and Outreach, CAGEF.
- **Revision 2.0.1**: edited and prepared for *CSB1021HF LEC0131, 10-2023* by Calvin Mok, Ph.D., Education and Outreach, CAGEF.
- **Revision 2.1.0**: edited and prepared for *CSB1021HF LEC0131, 10-2024* by Calvin Mok, Ph.D., Education and Outreach, CAGEF.

6.4.0 References

- Danacek P., et al. (GigaScience 2021). *Twelve years of SAMtools and BCFtools.* DOI: 10.1093/gigascience/giab008

- Ewels P., et al. (Bioinformatics 2016*). MultiQC: Summarize analysis results for multiple tools and samples in a single report*. DOI: 10.1093/bioinformatics/btw354

- Kim D., et al. (Nat Biotechnology 2019*). Graph-based genome alignment and genotyping with HISAT2 and HISAT-genotype.* DOI: 10.1038/s41587-019-0201-4

- Langmead B., Salzberg S.L. (Nat Methods 2012). *Fast gapped-read alignment with Bowtie 2.* DOI: 10.1038/nmeth.1923

- Liao Y., Smyth G.K., Shi W. (Bioinformatics 2014). *featureCounts: an efficient general purpose program for assigning sequence reads to genomic features.* DOI: 10.1093/bioinformatics/btt656

- Love M.I., Huber W., Anders S. (Genome Biology 2014). *Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2*. DOI: 10.1186/s13059-014-0550-8