Tutorial: Assemble a large genome using Galaxy

By Anna Syme, Melbourne Bioinformatics, The University of Melbourne, Australian BioCommons. November 2021

Introduction

Overview

- The aim of this tutorial is to demonstrate how to assemble a large plant or animal genome using tools and workflows in Galaxy.
- The analysis will run with a smaller test-sized data set, but has been tested on real-sized data sets for large genomes of approximately 1 billion base pairs, or 1 Gbp.

Prerequisites

- The tutorial content is designed for people with some familiarity with biology, DNA sequencing and genomics, but no specific knowledge is assumed.
- A computer with connection to the internet is required, and one of these web browsers: Chrome, Safari, Firefox.
- This tutorial will use the Galaxy Australia server. https://usegalaxy.org.au/
- New to Galaxy? See https://www.biocommons.org.au/galaxy-australia for information on getting started, and the tutorial: https://training.galaxyproject.org/training-material/topics/introduction/tutorials/galaxy-intro-short/tutorial.html

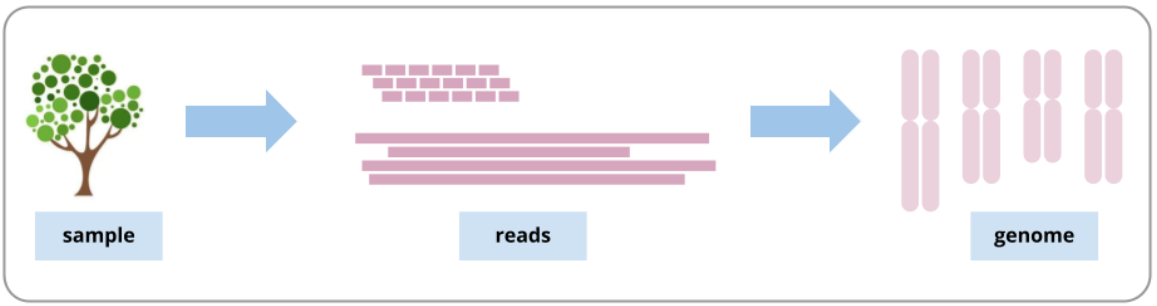
Can I use these tools and workflows on my own data?

- Yes, but the choice of tools, tool settings and workflow order will most likely need testing and changing to best fit your data and research questions.
- See the last sections in this tutorial (starting at Using your own data) for more information.

Let's get started!

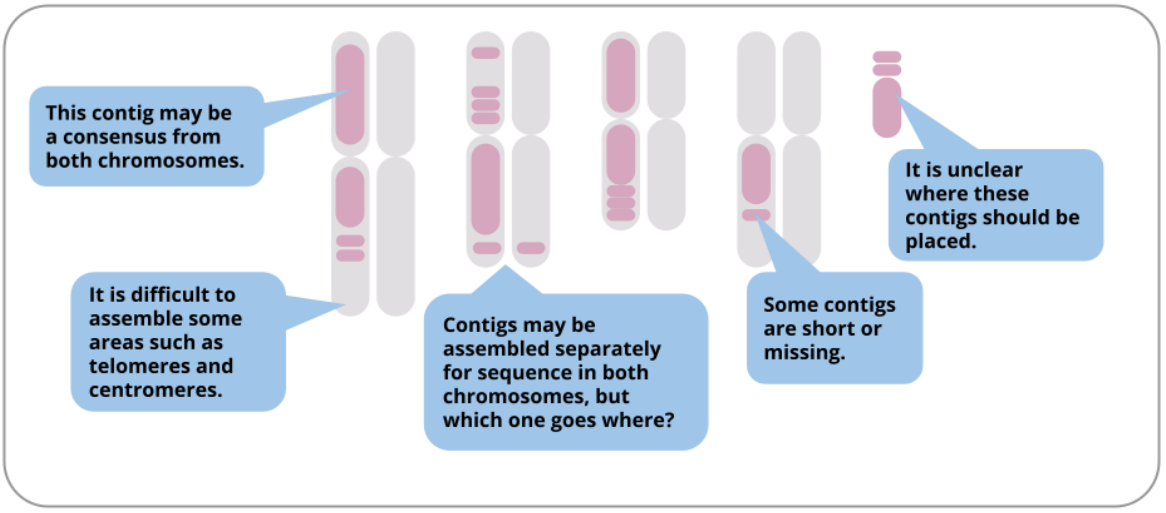
What is genome assembly?

A genome is a representation of the set of DNA in an organism, such as the set of chromosomes. When the DNA is extracted from the sample, it is broken up into fragments much smaller than the lengths of DNA in the chromosomes. These fragments are called sequencing reads. To assemble the genome, we need to join the reads back into, ideally, chromosome-sized lengths.



Assembly challenges

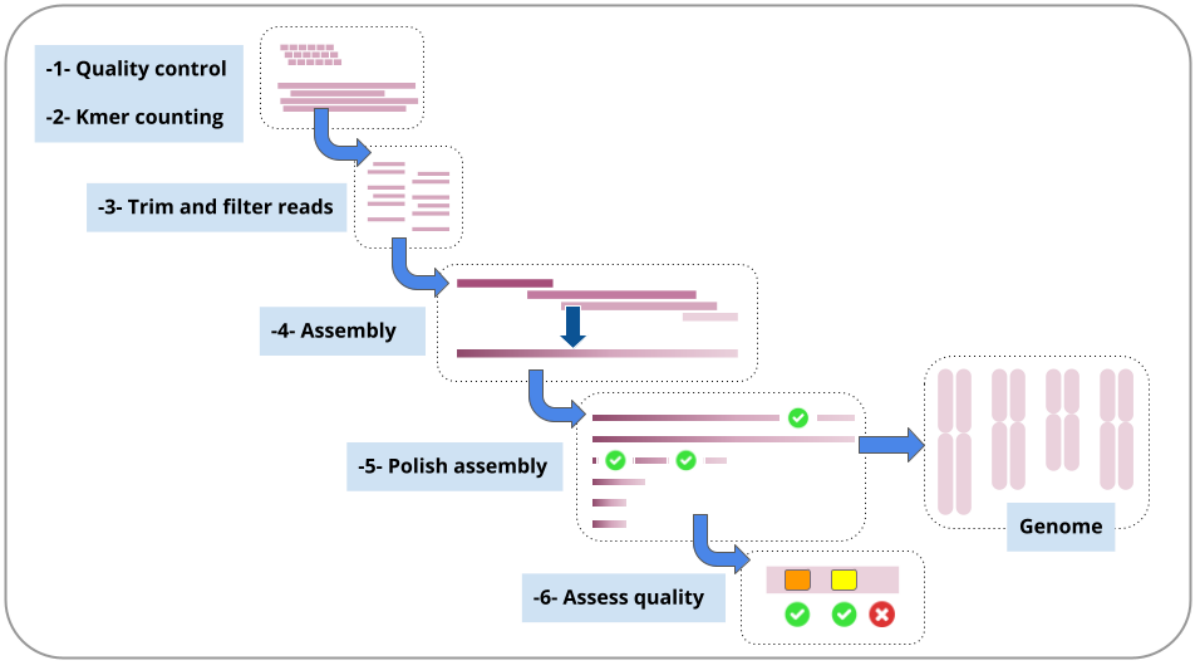
In reality, we rarely get chromosome-length assemblies, due to many challenges. Here are some examples of particular challenges in a diploid genome assembly:



Even though most assemblies are not chromosome-length, the assembly in contigs is still valuable for many research questions. Lengths of assembled contigs are increasing as sequencing technology and assembly tools improve.

Analysis workflow

In this tutorial, we will follow these steps:



- Each of these steps is described in a section in this tutorial.
- For each step, we will run a workflow.
- We will stay in the same Galaxy history throughout.

How to run a workflow in Galaxy

- Go to the top panel of Galaxy and see Shared Data -> Workflows. This shows a list of public workflows.
- Find the right workflow for the section you are in.
- Click on the drop-down arrow, and import the workflow.
- Now this will be in your own list of Workflows. (Galaxy top panel: Workflow)
- For the workflow you want to run, go to the right hand side and see the arrow button (a triangle), click
- This brings up the workflow in the centre Galaxy panel
- Click "Expand to full workflow form"
- For "Send results to a new history", leave it as "No".
- Each time you run a workflow, you need to specify the input data set (or sets). Galaxy will try to guess which file this is, but change if required using the drop-down arrow.
- At the top right, click "Run Workflow".
- The result files will appear at the top of your current history

Each workflow will be discussed in a separate section.

Log in to Galaxy Australia

- Open Galaxy Australia https://usegalaxy.org.au/ and log in.

Import tutorial data

What sequence data are we using in the tutorial?

- The data sets for genome projects can be very large and tools can take some time to run. It is a good idea to test that your planned tools and workflows will work on smaller-sized test data sets, as it is much quicker to find out about any problems.
- In this tutorial we will use a subset of real sequencing data from a plant genome, the snow gum, *Eucalyptus pauciflora*, from a genome project described in this paper: *Wang W, Das A, Kainer D, Schalamun M, Morales-Suarez A, Schwessinger B, Lanfear R; 2020, doi: 10.1093/gigascience/giz160.* Data is hosted at NCBI BioProject number: PRJNA450887.

How has this data subset been prepared?

- From NCBI, three read files were imported into Galaxy for this tutorial: nanopore reads (SRR7153076), and paired Illumina reads (SRR7153045).
- These were randomly subsampled to 10% of the original file size.
- Plant genomes may contain an excess of reads from the chloroplast genome (of which there are many copies per cell). To ensure our test data sets are not swamped from excessive chloroplast-genome reads, reads that mapped to a set of known chloroplast gene sequences were discarded.
- These steps are described in more detail, with a workflow, in the tutorial section How to prepare a test-sized set of data.

We are also using a reference genome *Arabidopsis thaliana* for a later comparison step (file `TAIR10_chr_all.fas` downloaded from https://www.arabidopsis.org/download/index-auto.jsp?dir=%2Fdownload_files%2FGenes%2FTAIR10_genome_release%2FTAIR10_chromosome_files)

Import data

- Go to https://usegalaxy.org.au/u/anna/h/eucalyptus-test-data and click at the cross at the top right to import the history with the tutorial data.

Expected results

This tutorial uses these input files and gives some examples from the results. To see histories showing the output files, see the tutorial section: Links to example histories.

Note: it is likely that your results will differ slightly (e.g. number of bases in the genome assembly). This is common, because many tools start from different random seeds. Also, tool versions are being constantly updated. Newer versions may be available since this tutorial was written and could give slightly different results.

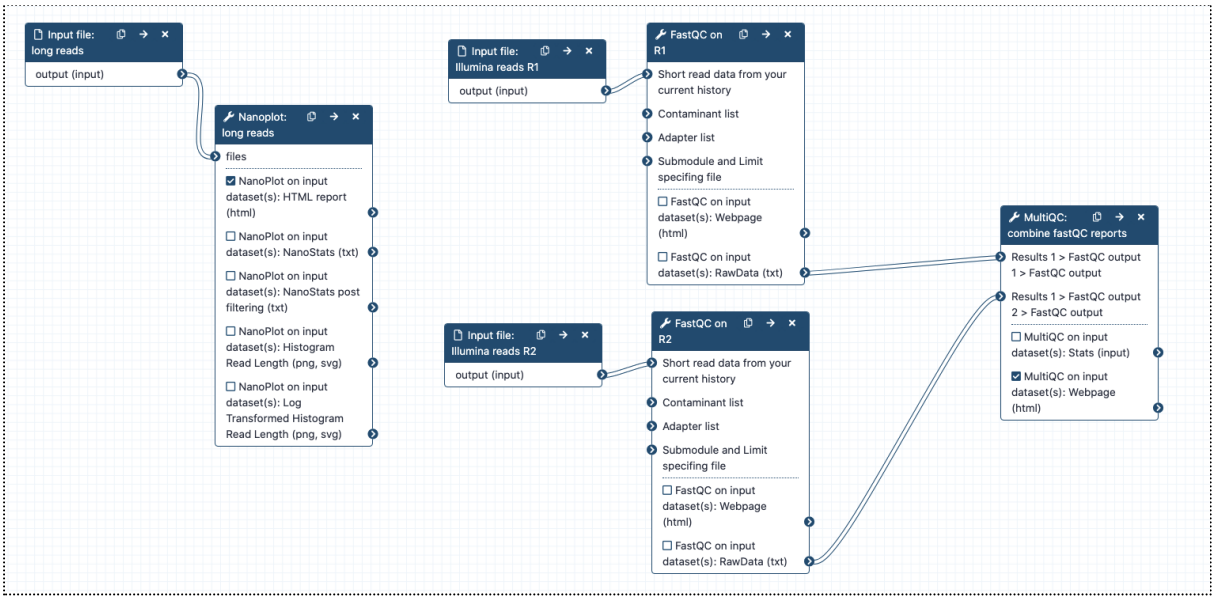
Quality control

Let's look at how many reads we have and their quality scores using the `Data QC` workflow.

Workflow information

Workflow name	Data QC
Workflow link	https://usegalaxy.org.au/u/anna/w/data-qc
What it does	Reports statistics from sequencing reads
Inputs	• long reads (fastq.gz format) • short reads (R1 and R2) (fastq.gz format)
Outputs	For long reads: a nanoplot report (the HTML report summarizes all the information) For short reads: a MultiQC report
Tools used	• Nanoplot • FastQC • MultiQC
Input parameters	None required
Workflow steps	• Long reads are analysed by Nanoplot • Short reads (R1 and R2) are analysed by FastQC; the resulting reports are processed by MultiQC
Report shows	Workflow steps
Options	• see the tool settings options at runtime and change as required. • Alternative tool option: fastp

Workflow image



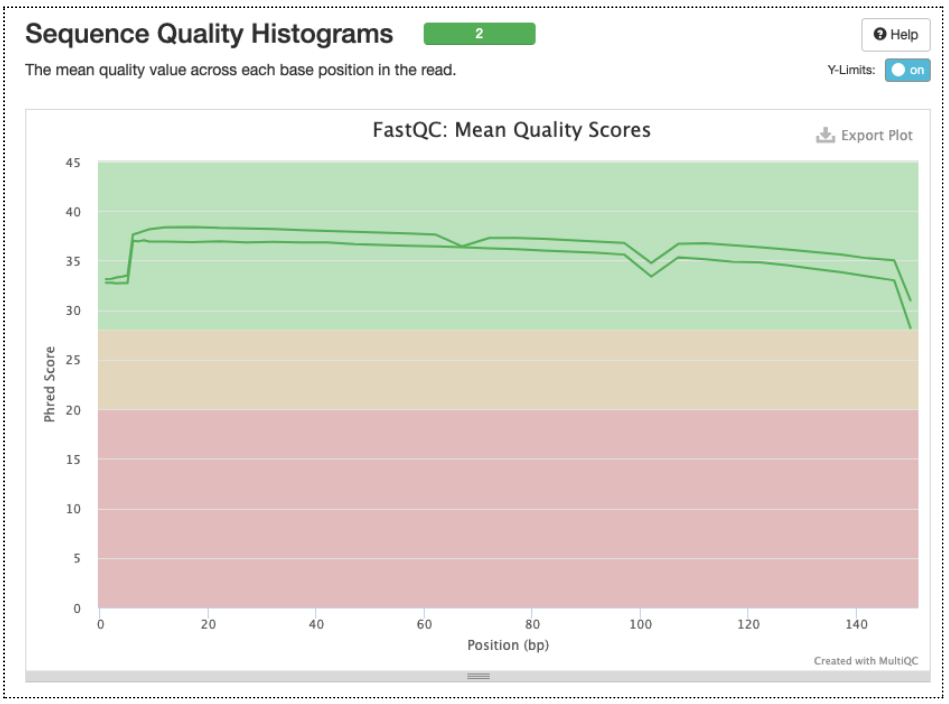
Run workflow

From your current Galaxy history (which contains the test data for this tutorial): go to the top panel in Galaxy, click **Shared Data: Workflows**, find this workflow, enter the correct input files, and run (see earlier section: How to run a workflow in Galaxy).

Data QC results

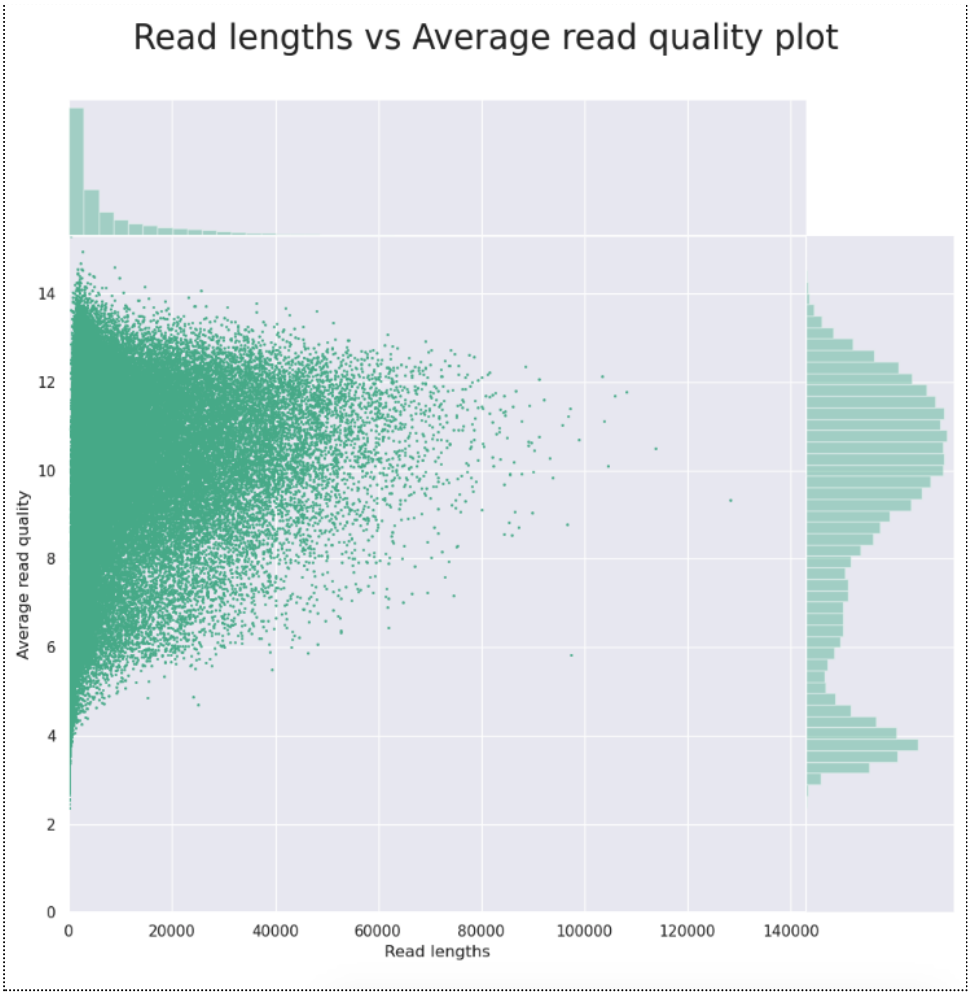
What are the results from the two output files? Are the reads long enough and of high enough quality for our downstream analyses? Will reads need any trimming or filtering? Common things to check are average read length, average quality, and whether quality varies by position in the reads.

A plot from MultiQC: read quality of Illumina reads (y axis) varies according to base position (x axis):



Here, we can see for Illumina reads that there is some drop-off in quality towards the end of the reads, which may benefit from trimming.

A plot from Nanoplot:



The nanopore reads have a mean read quality of 9.0. Depending on the size of our input read sets and the research question, we may filter out reads below a certain average quality. If we had a lot of reads, we may be able to set a higher threshold for filtering according to read quality.

More about interpreting nanoplot plots:

https://github.com/wdecoster/NanoPlot

https://gigabaseorgigabyte.wordpress.com/2017/06/01/example-gallery-of-nanoplot/

More about FastQC results:
https://www.bioinformatics.babraham.ac.uk/projects/fastqc/

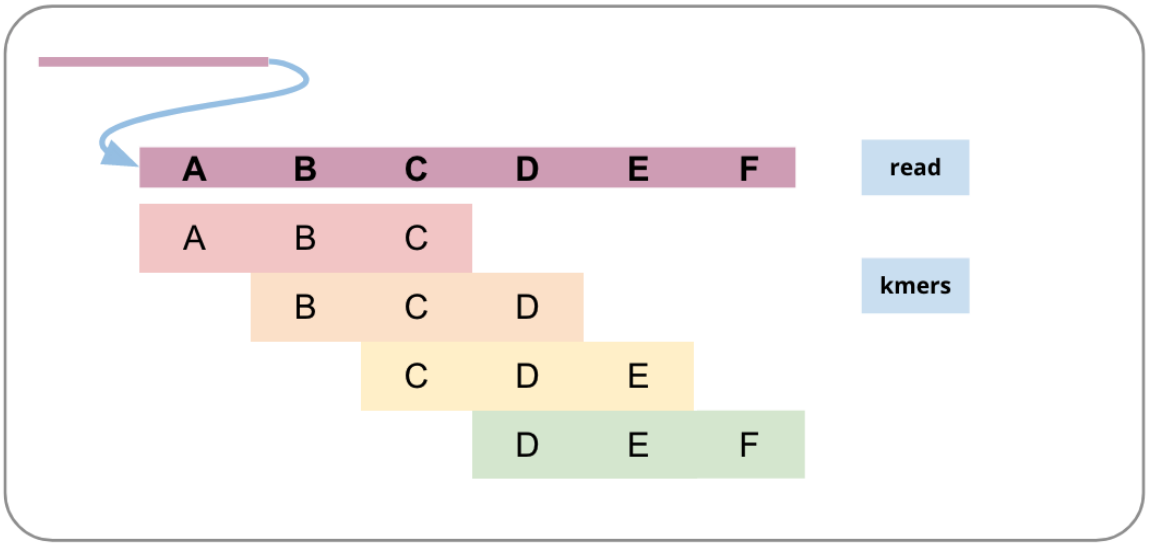
https://timkahlke.github.io/LongRead_tutorials/QC_F.html

Determine genome characteristics

To prepare for genome assembly you might want to know things about your genome such as size, ploidy level (how many sets of chromosomes) and heterozygosity (how variable the sequence is between homologous chromosomes). A relatively fast way to

estimate these things is to count small fragments of the sequencing reads (called kmers).

A read broken into kmers:



What is kmer counting?

Kmer counting is usually done with high-accuracy short reads, not long reads which may have high error rates. After counting how many times each kmer is seen in the reads, we can see what sorts of counts are common. For example, lots of kmers may have been found 24 or 25 times. A graph shows the number of different kmers (y axis) found at different counts, or depths (x axis).

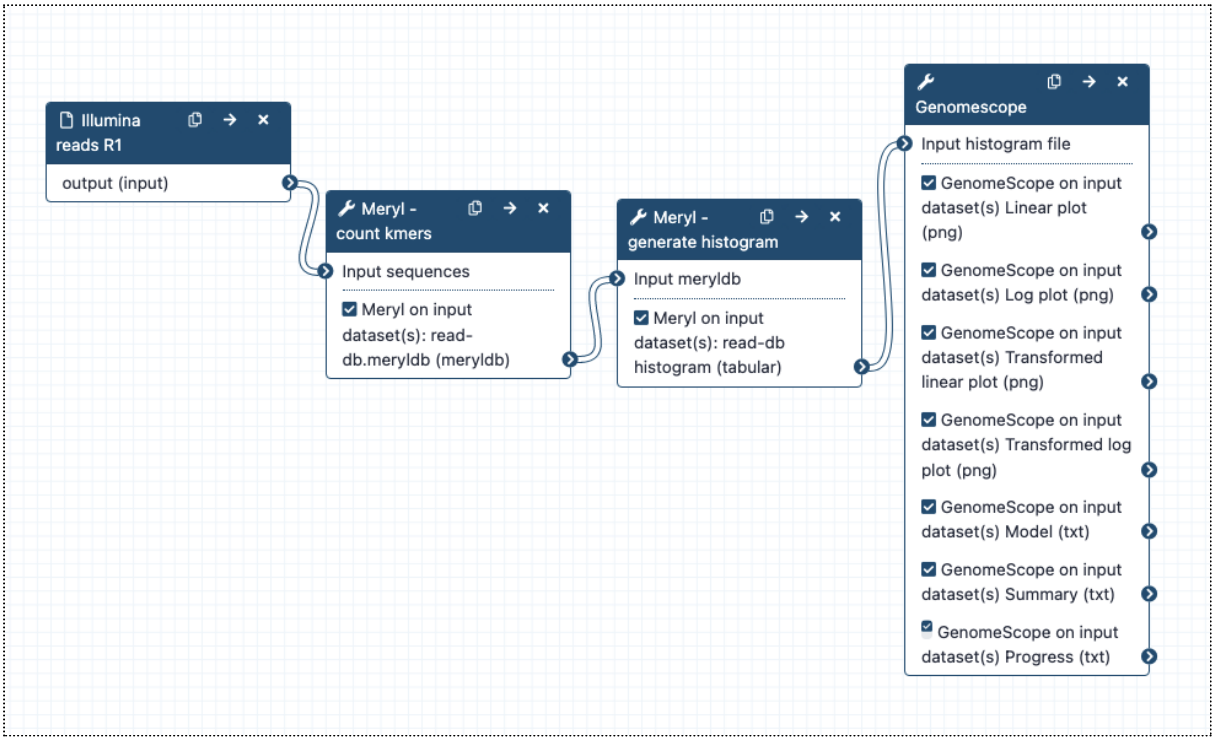
Many different kmers will be found the same number of times; e.g. X25. If kmer length approaches read length, this means the average depth of your sequencing is also ~X25, and there would be a peak in the graph at this position (smaller kmers = higher kmer depth). There may be smaller peaks of kmer counts at higher depths, e.g. X50 or X100, indicating repeats in the genome. There may be other smaller peaks of kmers found at half the average depth, indicating a diploid genome with a certain amount of difference between the homologous chromosomes - this is known as heterozygosity. Thus, the plot of how many different kmers are found at all the depths will help inform estimates of sequencing depth, ploidy level, heterozygosity, and genome size.

Workflow information

Workflow name	Kmer counting - meryl
Workflow link	https://usegalaxy.org.au/u/anna/w/kmer-counting-meryl
What it does	Estimates genome size and heterozygosity based on counts of kmers

Inputs	• One set of short reads: e.g. R1.fq.gz
Outputs	• GenomeScope graphs
Tools used	• Meryl • GenomeScope
Input parameters	None required
Workflow steps	• The tool meryl counts kmers in the input reads (k=21), then converts this into a histogram. • GenomeScope: runs a model on the histogram; reports estimates. k-mer size set to 21.
Report shows	• Workflow steps • Genomescope plot: transformed linear plot.
Options	Use a different kmer counting tool. e.g. khmer. • Advanced parameters: • k-mer size: 21 (as per this recommendation https://github.com/schatzlab/genomescope/issues/32) • n_tables: 4 • tablesize: set at 8 billion (as per this recommendation https://khmer.readthedocs.io/en/v1.0/choosing-table-sizes.html) • Will also need to run some formatting steps to convert khmer output to a two-column matrix, for the Genomscope. See this workflow: https://usegalaxy.org.au/u/anna/w/kmer-counting-khmer • note: khmer: to use both R1 and R2 read sets, khmer needs these paired reads in interleaved format.

Workflow image

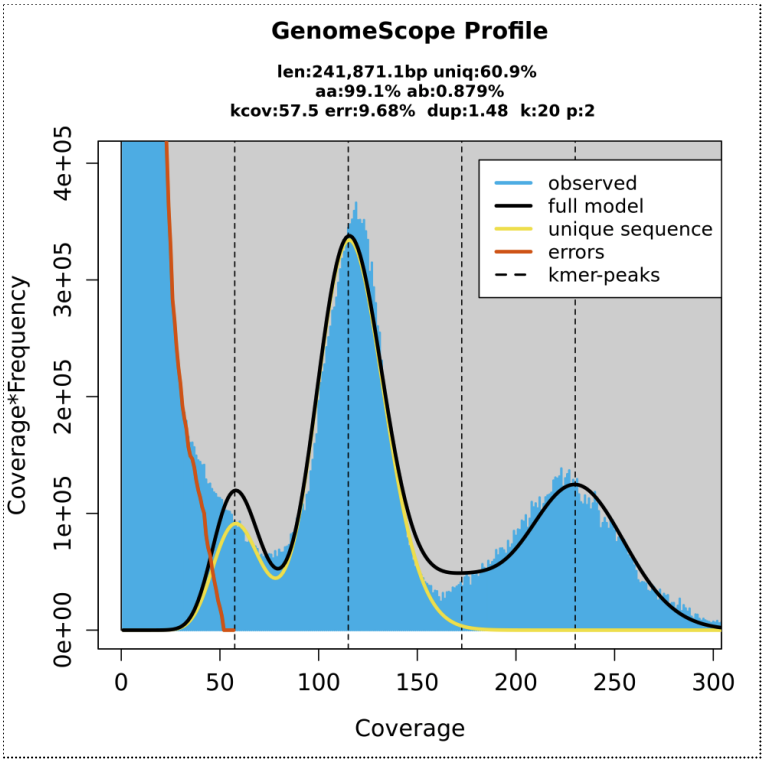


Run workflow

From your current Galaxy history, run this workflow with the required input data (see table above). For more detail see earlier section: How to run a workflow in Galaxy.

Kmer counting results

GenomeScope transformed linear plot:



Here we can see a central peak - showing that most of the different kmers were found at counts of ~ 120. These are kmers from single-copy homozygous alleles. To the left, a smaller peak at around half the coverage, showing kmers from heterozygous alleles (note that this peak gets higher than the main peak when heterozygosity is only ~ 1.2%). To the right, another smaller peak showing kmers at higher coverage, from repeat regions. Information from these three peaks provide a haploid genome length estimate of ~240,000 bp (note this is test data so smaller than whole plant genome size).

The output Summary file shows more detail:

- Genome unique length: from single copy homozygous and heterozygous alleles (under the main and left peak).
- Genome repeat length: from repeat copies (under the graph to the right of the main peak).
- **Genome haploid length**: unique length + repeat length

More about kmer counting: See https://bioinformatics.uconn.edu/genome-size-estimation-tutorial/#

Meryl: See "Rhie, A., Walenz, B.P., Koren, S. et al. Merqury: reference-free quality, completeness, and phasing assessment for genome assemblies" https://genomebiology.biomedcentral.com/articles/10.1186/s13059-020-02134-9

Genomescope: See "Vurture, G et al.GenomeScope: fast reference-free genome profiling from short reads" https://doi.org/10.1093/bioinformatics/btx153 (Note: the supplementary information is very informative).

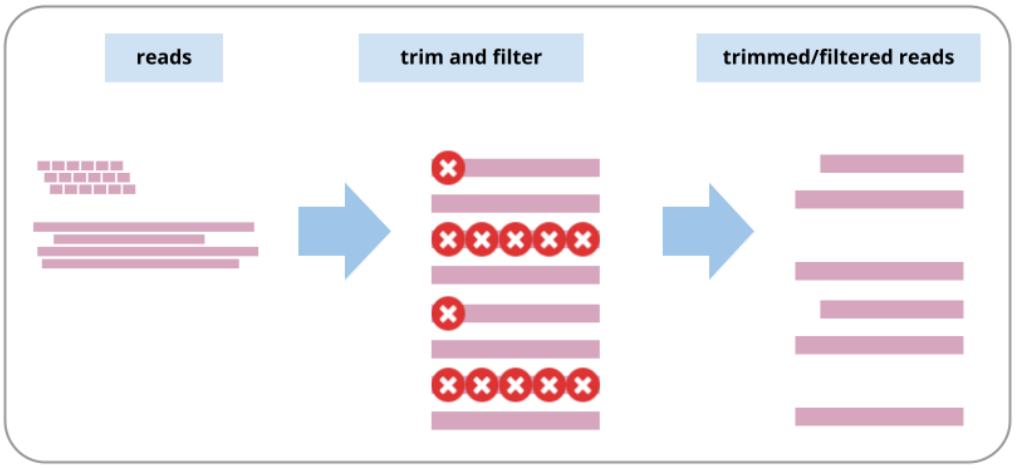
Trim and filter reads

Using information from Data QC and kmer counting, we may want to trim and/or filter reads. The settings for trimming and filtering depend on many things, including:

- your aim (accuracy; contiguity)
- your data: type, error rate, read depth, lengths, quality (average, variation by position)
- the ploidy and heterozygosity of your sample
- choice of assembly tool (e.g. it may automatically deal with adapters, low qualities, etc.)

Because of all these factors, few specific recommendations are made here, but the workflow is provided and can be customised. If you are unsure how to start, use your test data to try different settings and see the effect on the resulting size and quality of the reads, and the downstream assembly contigs. Newer assemblers are often configured to work well with long-read data and in some cases, read trimming/filtering for long reads may be unnecessary.

Trimming and filtering reads:



Workflow information

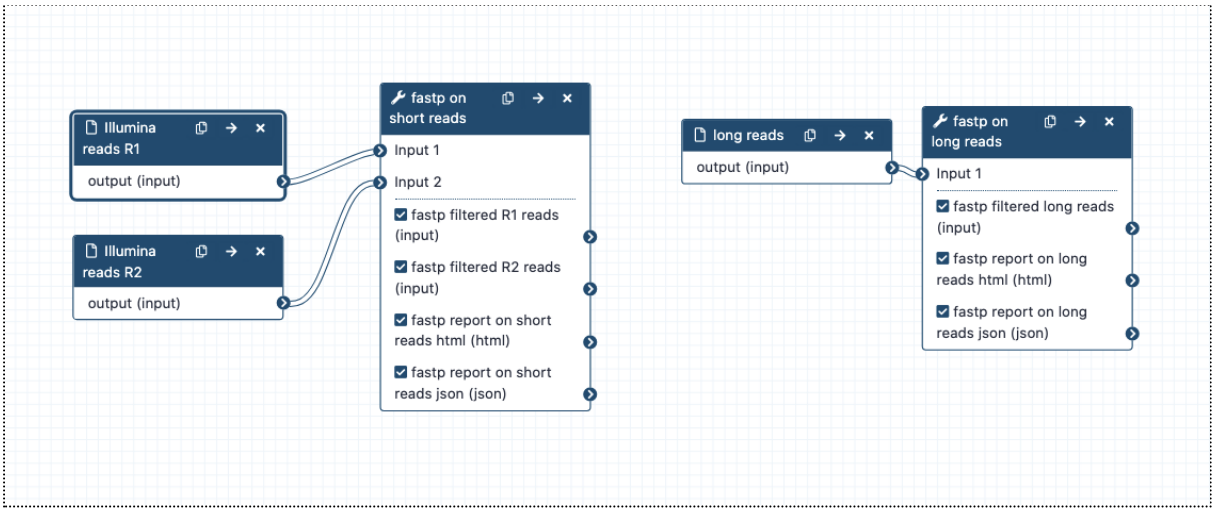
Workflow name	Trim and filter reads
Workflow link	https://usegalaxy.org.au/u/anna/w/trim-and-filter-reads-fastp
What it does	Trims and filters raw sequence reads according to specified settings.
Inputs	• Long reads (format fastq) • Short reads R1 and R2 (format fastq)
Outputs	Trimmed and filtered reads: • fastp_filtered_long_reads.fastq.gz (But note: **no** trimming or filtering is on by default) • fastp_filtered_R1.fastq.gz • fastp_filtered_R2.fastq.gz Reports: • fastp report on long reads, html • fastp report on short reads, html
Tools used	• fastp (Note. The latest version (0.20.1) of fastp has an issue displaying plot results. Using version 0.19.5 here instead until this is rectified).

Input parameters	None required, but recommend removing the long reads from the workflow if not using any trimming/filtering settings.
Workflow steps	Long reads: fastp settings: These settings have been changed from the defaults (so that all filtering and trimming settings are now disabled). • Adapter trimming options: Disable adapter trimming: yes • Filter options: Quality filtering options: Disable quality filtering: yes • Filter options: Length filtering options: Disable length filtering: yes • Read modification options: PolyG tail trimming: Disable • Output options: output JSON report: yes Short reads: fastp settings: • adapter trimming (default setting: adapters are auto-detected) • quality filtering (default: phred quality 15), unqualified bases limit (default = 40%), number of Ns allowed in a read (default = 5) • length filtering (default length = min 15) • polyG tail trimming (default = on for NextSeq/NovaSeq data which is auto detected) • Output options: output JSON report: yes
Report shows	Workflow steps.
Options	• Change any settings in fastp for any of the input reads. • Adapter trimming: input the actual adapter sequences. (Alternative tool for long read adapter trimming: Porechop.) • Trimming *n* bases from ends of reads if quality less than value *x* (Alternative tool for trimming long reads: NanoFilt.) • Discard post-trimmed reads if length is < x (e.g. for long reads, 1000 bp) • Example filtering/trimming that you might do on long reads: remove adapters (can also be done with Porechop),

trim bases from ends of the reads with low quality (can also be done with NanoFilt), after this can keep only reads of length x (e.g. 1000 bp)

- If not running any trimming/filtering on nanopore reads, could delete this step from the workflow entirely.

Workflow image



Run workflow

From your current Galaxy history, run this workflow with the required input data (see table above). For more detail see earlier section: How to run a workflow in Galaxy.

Trim and filter reads: results

There are two fastp reports - one for the illumina reads and one for the nanopore reads. We have only processed illumina reads in this example. Look at the fastp illumina report. (Note: the title in the report refers to only one of the input read sets but the report is for both read sets. This is a known issue under investigation.)

Filtering results from fastp on short reads:

Filtering result

reads passed filters:	3.201638 M (99.513998%)
reads with low quality:	14.960000 K (0.464990%)
reads with too many N:	390 (0.012122%)
reads too short:	286 (0.008890%)

Here we can see that less than 0.5 % of the reads were discarded based on quality. If our read set had high enough coverage for downstream analyses, we might choose to apply a stricter quality filter.

Summary of read data for genome assembly

How many reads do we have now for our genome assembly? Is the read coverage high enough?

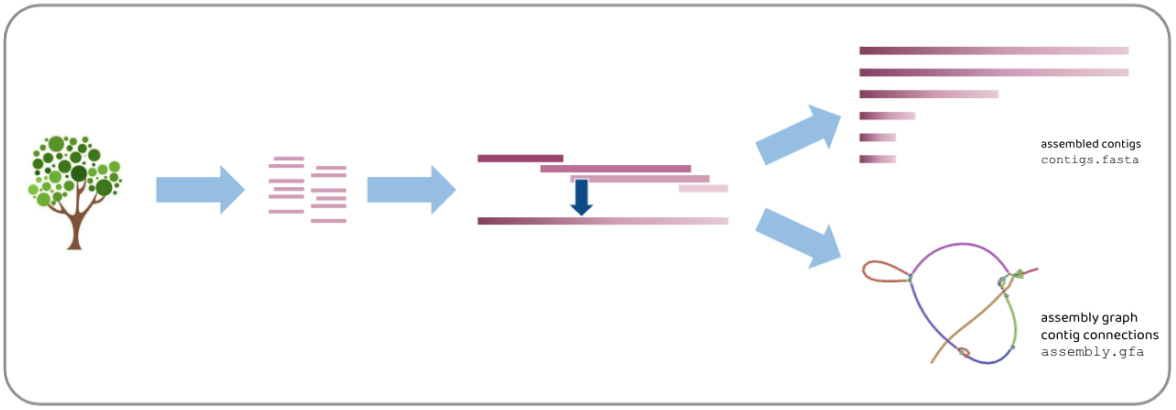
- *Genome size:* From kmer counting, the estimated genome size is ~ 240,000 bp (this is only subsampled data; full data would likely suggest a size of 0.5 - 1 Gbp for a typical plant genome)
- *Genome coverage (or depth):* total base pairs in the reads / base pairs in genome
- *Short reads:* From the fastp report after trimming and filtering short reads, there are 3.2 million reads, comprising 482 million base pairs
- *Short read coverage:* = X2008
- *Long reads:* From the fastp report of the long reads (although no filtering and trimming performed) there are 85 thousand reads, comprising 761 million base pairs. From the nanoplot tool we ran in the Data QC section , we know that the mean read length is almost 9,000 base pairs, and the longest read is > 140,000 base pairs.
- *Long read coverage:* = X3170

These coverages are very high but are ok to use with tutorial data. With a typical full data set, coverage would be more in the order of X40 to X200.

Genome assembly

Genome assembly means joining the reads up to make contiguous sections of the genome. A simplified way to imagine this is overlapping all the different sequencing reads to make a single length or contig, ideally one for each original chromosome. The output is a set of contigs and a graph showing how contigs are connected.

Extreme simplification of genome assembly:



Genome assembly algorithms use different approaches to work with the complexities of large sequencing read data sets, large genomes, different sequencing error rates, and computational resources. Many use graph-based algorithms. For more about genome assembly algorithms see https://langmead-lab.org/teaching-materials/.

Which assembly tool and approach to use?

Here, we will use the assembly tool called Flye to assemble the long reads. This is fast and deals well with the high error rate. Then, we will polish (correct) the assembly using information from the long reads (in their unassembled state), as well as the more accurate short Illumina reads.

There are many other approaches and combinations of using short and long reads, and the polishing steps. For example, the long reads can be polished before assembly (with themselves, or with short reads). This may increase accuracy of the assembly, but it may also introduce errors if similar sequences are "corrected" into an artificial consensus. Long reads are usually used in the assembly, but it is possible to assemble short reads and then scaffold these into longer contigs using information from long reads.

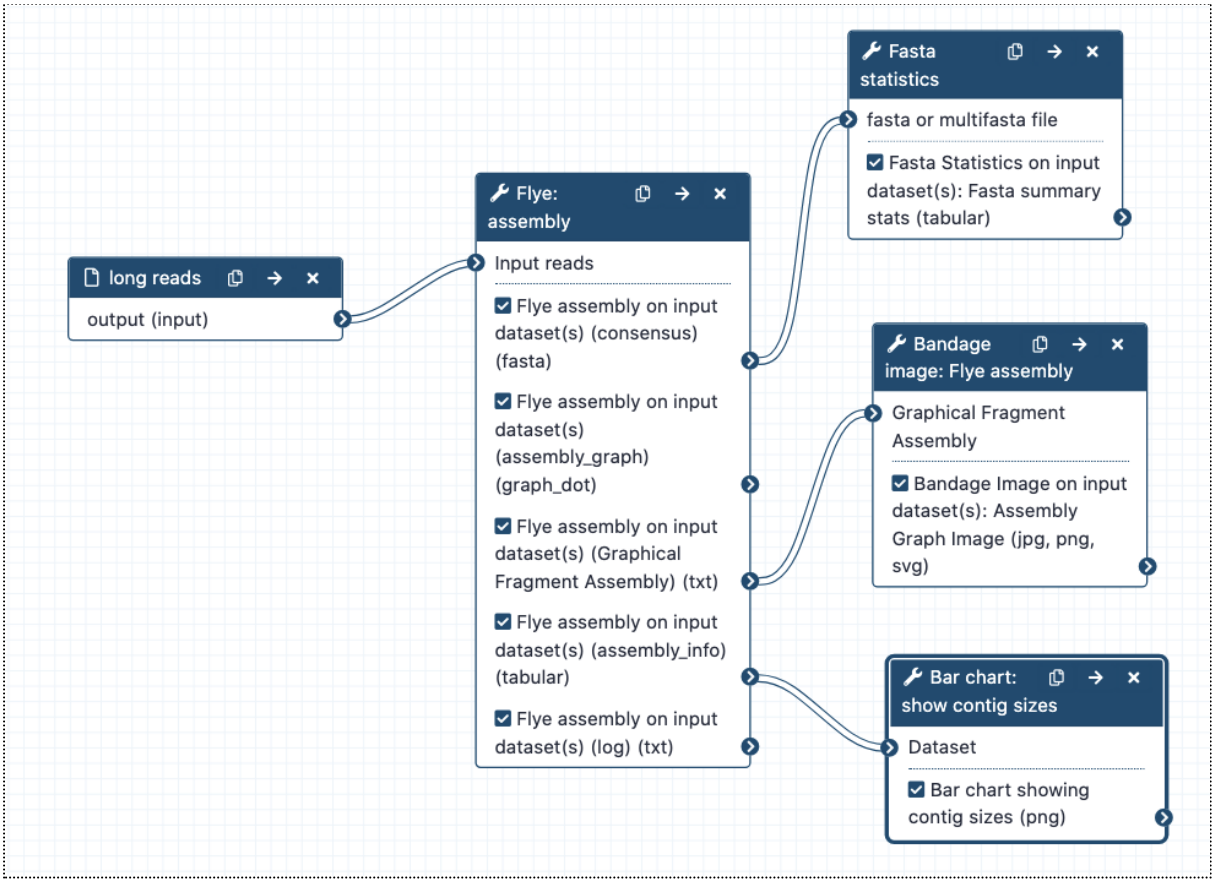
For more about the differences between current assembly and polishing tools see "Chen, Y. et al. Efficient assembly of nanopore reads via highly accurate and intact error correction." https://doi.org/10.1038/s41467-020-20236-7, and "McCartney, A. et al., An exploration of assembly strategies and quality metrics on the accuracy of the rewarewa (*Knightia excelsa*) genome" https://doi.org/10.1111/1755-0998.13406.

Workflow information

Workflow name	Assembly with Flye
Workflow link	https://usegalaxy.org.au/u/anna/w/assembly-with-flye
What it does	Assembles long reads with the tool Flye
Inputs	• long reads (may be raw, or filtered, and/or corrected); fastq.gz format
Outputs	• Flye assembly fasta. • Fasta stats on assembly.fasta • Assembly graph image from Bandage • Bar chart of contig sizes • Quast reports of genome assembly
Tools used	• Flye • Fasta statistics • Bandage • Bar chart • Quast

Input parameters	None required, but recommend setting assembly mode to match input sequence type
Workflow steps	• Long reads are assembled with Flye, using default tool settings. Note: the default setting for read type ("mode") is nanopore raw. Change this at runtime if required. • Statistics are computed from the assembly.fasta file output, using Fasta Statistics and Quast (is genome large: Yes; distinguish contigs with more that 50% unaligned bases: no) • The graphical fragment assembly file is visualized with the tool Bandage. • Assembly information sent to bar chart to visualize contig sizes
Report shows	• Workflow steps • Fasta stats • Bandage image • Bar chart of contig sizes • Quast report
Options	• See other Flye options. • Use a different assembler (in a different workflow). • Bandage image options - change size (max size is 32767), labels - add (e.g. node lengths). You can also install Bandage on your own computer and download the "graphical fragment assembly" file to view in greater detail.

Workflow image



Run workflow

From your current Galaxy history, run this workflow with the required input data (see table above). For more detail see earlier section: How to run a workflow in Galaxy.

Assembly results

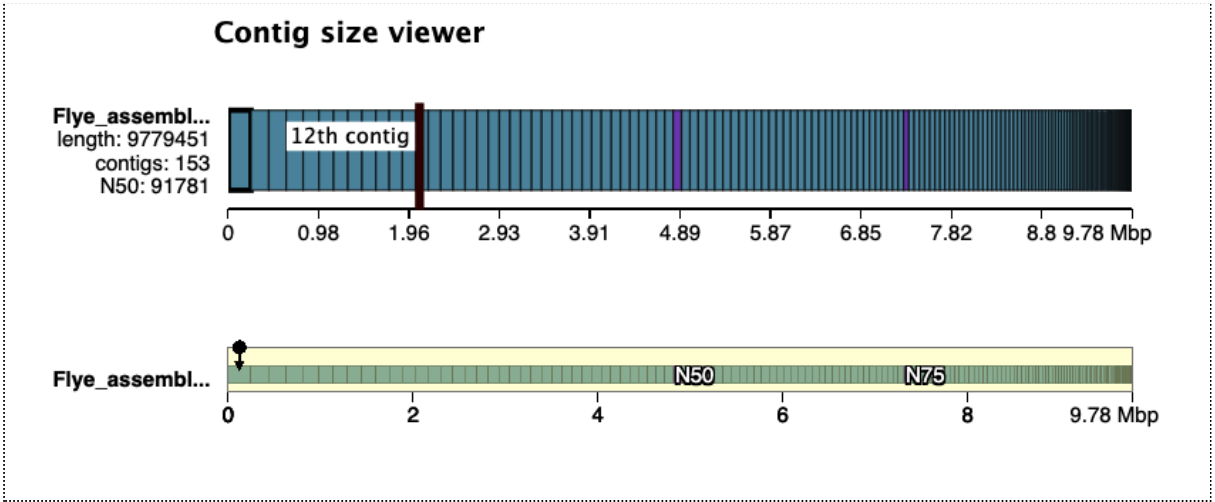
The assembled contigs are in the "Flye assembly on data X (consensus)" (X is a number that will vary depending on where it sits in your history).

Open the Quast tabular report to see the assembly statistics:

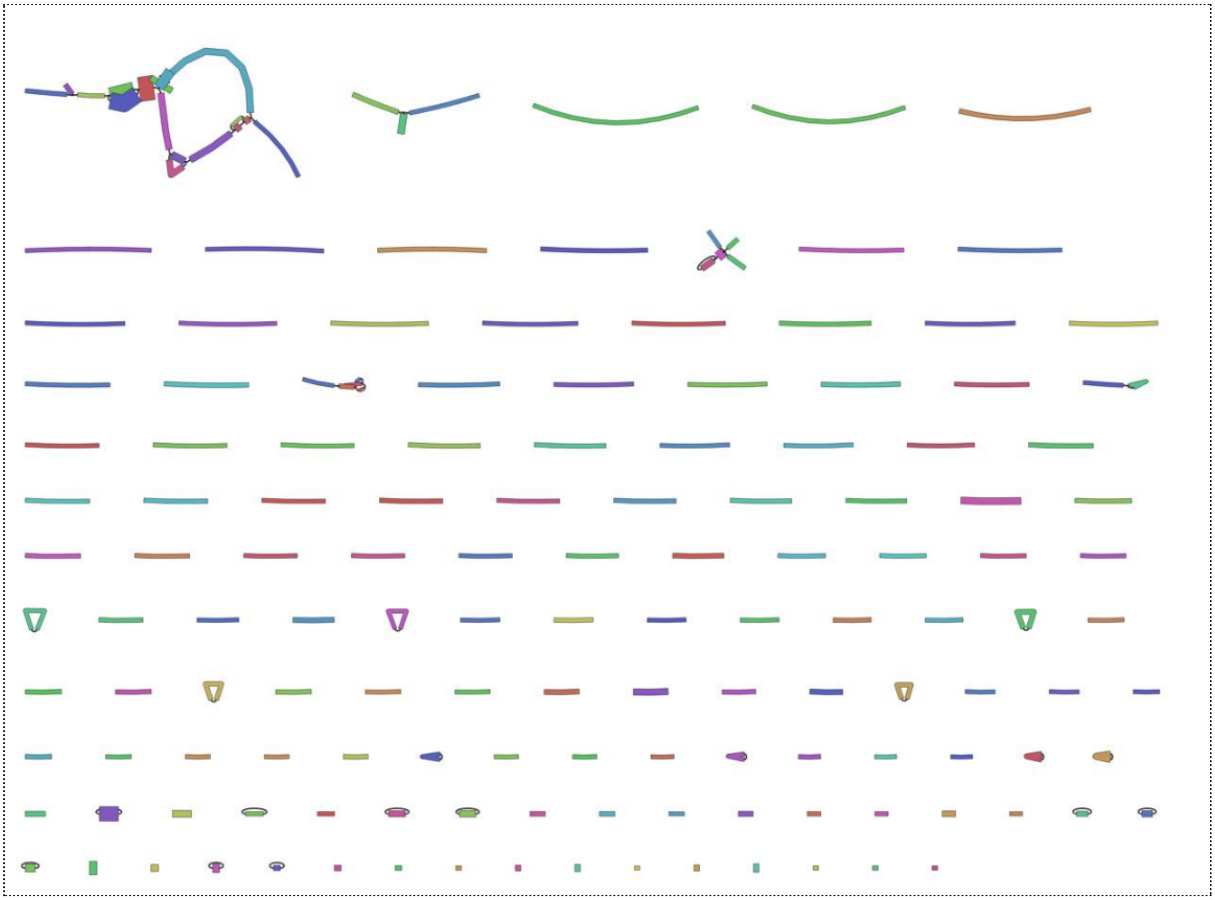
Assembly	Flye_assembly_on_data_1__consensus_
# contigs (>= 0 bp)	153
# contigs (>= 1000 bp)	153
Total length (>= 0 bp)	9779451
Total length (>= 1000 bp)	9779451
# contigs	153
Largest contig	245571
Total length	9779451
GC (%)	40.28
N50	91781
N75	57778
L50	38
L75	71
# N's per 100 kbp	0.00

There are 153 contigs, largest is ~246,000 bp, and total length almost 10 million bp. This is a fair bit longer than the estimated genome size from kmer counting (which was ~240,000 bp), but the difference is likely mainly due to idiosyncrasies of using a subsampled data set. The read coverage was likely <1, causing many kmers to have frequency of <1 and be classed as errors, rather than contributing to the genome size estimate.

Open the Quast HTML report, then click on "View in Icarus contig browser". This is a way to visualize the contigs and their sizes:



View the Bandage image of the assembly graph:



As this is a subsampled data set, it is not surprising that most of the contigs are unjoined. The joined contigs at the top left are likely to be part of the mitochondrial genome as these reads were probably over-represented in our subsampled data set.

What about centromeres and telomeres?

Some genomic areas such as centromeres, telomeres, and ribosomal DNA arrays, are much harder to assemble. These are long stretches of very similar repeats. With improved sequencing accuracy, length, and technologies (particularly long-range scaffolding), these may soon be much easier to assemble. The latest human genome assembly has a good demonstration of the techniques used for this. See "The complete sequence of a human genome" https://www.biorxiv.org/content/10.1101/2021.05.26.445798v1.full.pdf, and in particular, Figure 2: Bandage graphs of the human genome chromosomes, with the grey shading showing centromeric regions.

What about haplotigs?

Although our sample may be diploid, with pairs of chromosomes, the resulting assembly is often a haploid (or "collapsed") assembly. This is not the sequence of one of the chromosomes, but a mix of the two.

Some assemblers will produce extra contigs called haplotigs. These are parts of the assembly from heterozygous regions (that is, the sequence is relatively different between the chromosome pair). There are tools to remove haplotigs from the assembly if that is preferred.

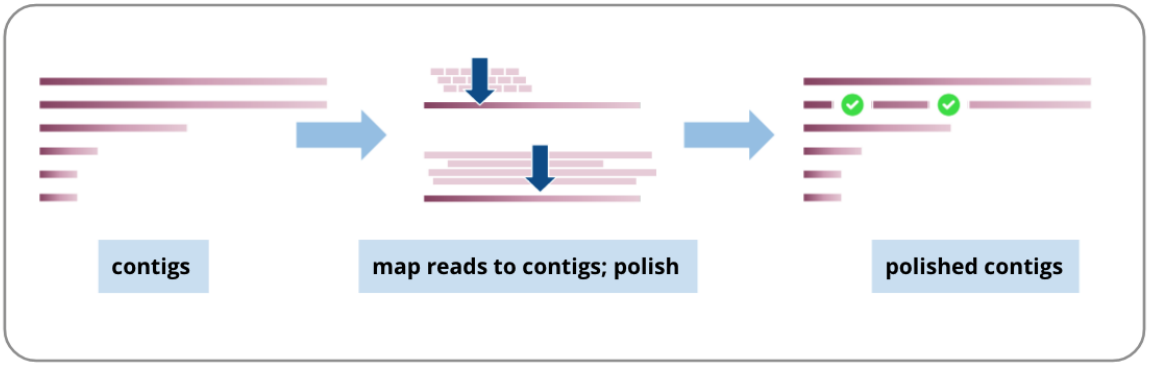
For more on differences between collapsed, primary/alternate and partially-phased assemblies, with a great visual representation: see http://lh3.github.io/2021/04/17/concepts-in-phased-assemblies

For more on the phased assemblies, particularly for diploids or polyploids, see "Garg, S. Computational methods for chromosome-scale haplotype reconstruction" https://genomebiology.biomedcentral.com/articles/10.1186/s13059-021-02328-9

Assembly polishing

We will polish the assembly using both the long reads and short reads. This process aligns the reads to the assembly contigs, and makes corrections to the contigs where warranted. For more, see "Aury, J; Istace, B. Hapo-G, haplotype-aware polishing of genome assemblies with accurate reads" https://academic.oup.com/nargab/article/3/2/lqab034/6262629, particularly for a discussion about polishing diploid genomes.

Assembly polishing:



Workflow information

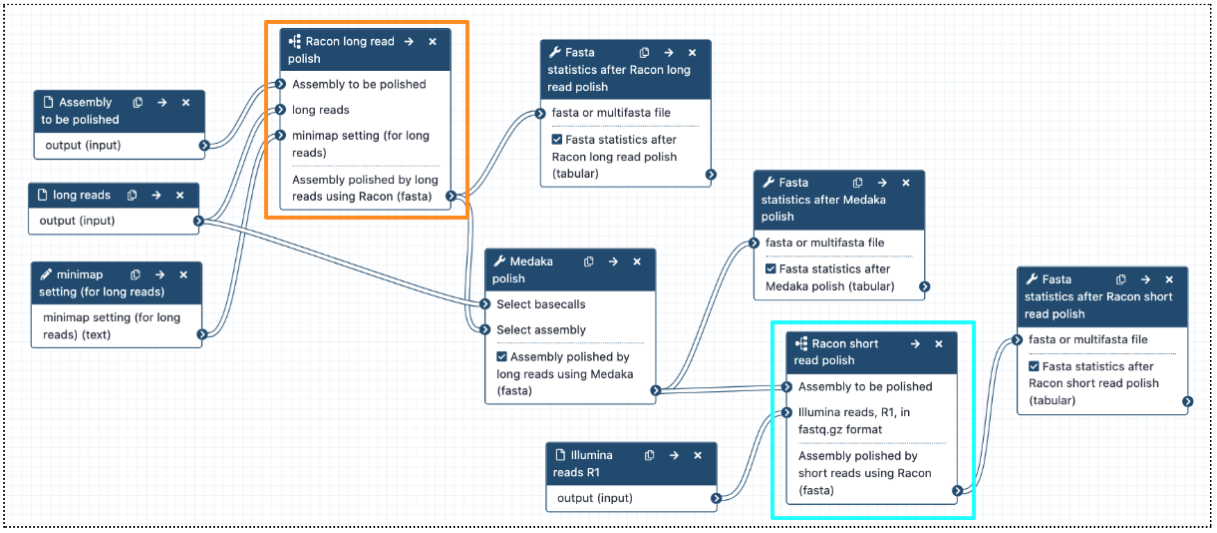
Workflow name and links	Assembly polishing https://usegalaxy.org.au/u/anna/w/assembly-polishing This includes two subworkflows: Racon polish with long reads, x 4 https://usegalaxy.org.au/u/anna/w/racon-polish-with-long-reads-x4 Racon polish with illumina reads, x2 https://usegalaxy.org.au/u/anna/w/racon-polish-with-illumina-reads-x2
What it does	Polishes (corrects) an assembly, using long reads (with the tools Racon and Medaka) and short reads (with the tool Racon) (Note: medaka is only for nanopore reads, not PacBio reads).
Inputs	• assembly to be polished: assembly.fasta • long reads - the same set used in the assembly (e.g. may be raw or filtered) fastq.gz format • short reads, R1 only, in fastq.gz format
Outputs	• Racon+Medaka+Racon polished_assembly. fasta • Fasta statistics after each polishing tool
Tools used	• Minimap2 • Racon • Fasta statistics

	• Medaka
Input parameters	None required, but recommended to set the Medaka model correctly (default = r941_min_high_g360). See drop down list for options.
Workflow steps	**-1- Polish with long reads: using Racon** Long reads and assembly contigs => Racon polishing (subworkflow): • minimap2 : long reads are mapped to assembly => overlaps.paf. • overaps, long reads, assembly => Racon => polished assembly 1 • using polished assembly 1 as input; repeat minimap2 + racon => polished assembly 2 • using polished assembly 2 as input, repeat minimap2 + racon => polished assembly 3 • using polished assembly 3 as input, repeat minimap2 + racon => polished assembly 4 Racon long-read polished assembly => Fasta statistics Note: The Racon tool panel can be a bit confusing and is under review for improvement. Presently it requires sequences (= long reads), overlaps (= the paf file created by minimap2), and target sequences (= the contigs to be polished) as per "usage" described here https://github.com/isovic/racon/blob/master/README.md Note: Racon: the default setting for "output unpolished target sequences?" is No. This has been changed to Yes for all Racon steps in these polishing workflows. This means that even if no polishes are made in some contigs, they will be part of the output fasta file. Note: the contigs output by Racon have new tags in their headers. For more on this see https://github.com/isovic/racon/issues/85. **-2- Polish with long reads: using Medaka** Racon polished assembly + long reads => medaka polishing X1 => medaka polished assembly

	Medaka polished assembly => Fasta statistics **-3- Polish with short reads: using Racon** Short reads and Medaka polished assembly =>Racon polish (subworkflow): • minimap2: short reads (R1 only) are mapped to the assembly => overlaps.paf. Minimap2 setting is for short reads. • overlaps + short reads + assembly => Racon => polished assembly 1 • using polished assembly 1 as input; repeat minimap2 + racon => polished assembly 2 Racon short-read polished assembly => Fasta statistics
Report shows	• Workflow steps • Fasta statistics for the polished assembly after each tool
Options	• Change settings for Racon long read polishing if using PacBio reads: The default profile setting for Racon long read polishing: minimap2 read mapping is "Oxford Nanopore read to reference mapping", which is specified as an input parameter to the whole Assembly polishing workflow, as text: *map-ont*. If you are not using nanopore reads and/or need a different setting, change this input. To see the other available settings, open the minimap2 tool, find "Select a profile of preset options", and click on the drop down menu. For each described option, there is a short text in brackets at the end (e.g. *map-pb)*. This is the text to enter into the assembly polishing workflow at runtime instead of the default (*map-ont*). • Other options: change the number of polishes (in Racon and/or Medaka). There are ways to assess how much improvement in assembly quality has occurred per polishing round (for example, the number of corrections made; the change in Busco score - see section "Genome quality assessment" for more on Busco). • Option: change polishing settings for any of these tools. Note: for Racon - these will have to be changed within those subworkflows first. Then, in the main workflow, update the subworkflows, and re-save.

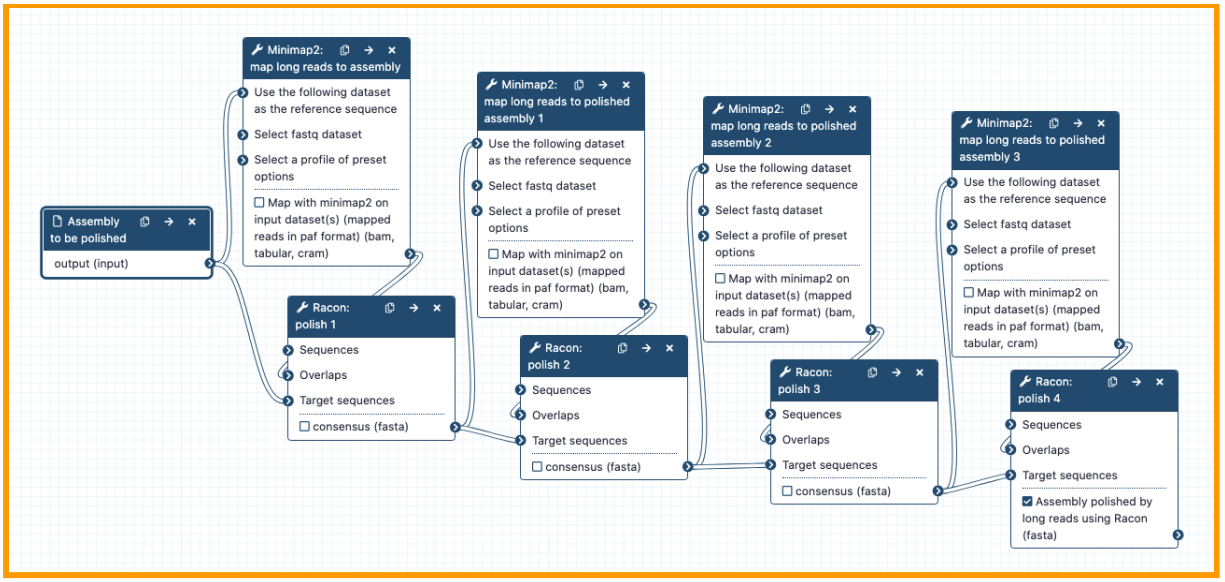
Workflow images

Assembly polishing: combined workflow

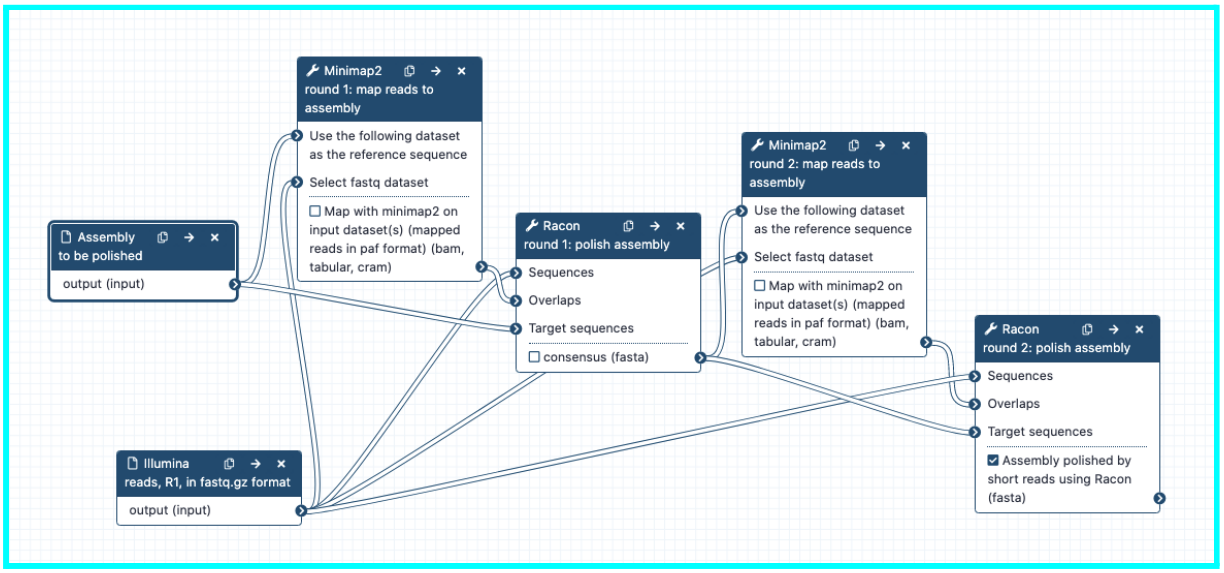


Subworkflow: Racon long read polish - from orange box above.

Note: For clarity, two inputs (long reads, and minimap setting) and their connections have been removed from the image. The long read data set connects as input to each of the minimap2 steps at "Select fastq dataset", and to each of the Racon steps at "Sequences". The minimap setting connects as input to each of the minimap2 steps at "Select a profile of preset options".



Subworkflow: Racon short read polish - from blue box above.



Run workflow

From your current Galaxy history, run this workflow with the required input data (see table above). For more detail see earlier section: How to run a workflow in Galaxy.

Polishing results

The polished assembly is the final Racon file.

Look at the Fasta Statistics output files for the status of the assemblies after each polishing tool. From these, we can see that some polishes decrease or increase the size of the assembly. The final size is approximately 300,000 bp shorter than the original Flye assembly. (These numbers may vary slightly even if the same input data is used).

Genome quality assessment

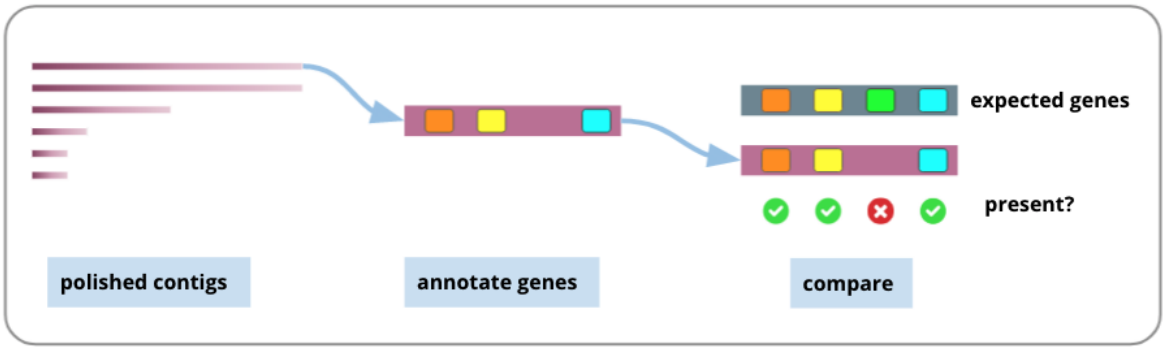
The polished genome is in 145 contigs with a total length of ~ 9.6 million base pairs. We have some idea of how these contigs may be joined (or not) from the Bandage assembly graph.

How good is the assembly? One measure is the N50, a number that indicates how large the contigs are (although, have the contigs been joined correctly)? Another measure is to see if expected gene sequences are found in the assembly, using a tool called BUSCO. For a discussion on these and other methods, see "Wang, W. et al. The draft nuclear genome assembly of *Eucalyptus pauciflora:* a pipeline for comparing de novo assemblies" https://academic.oup.com/gigascience/article/9/1/giz160/5694103

Here, we will use the BUSCO tool to annotate the genome and then assess whether expected genes are found. Note: this is a brief annotation only, not the full genome annotation that would typically be done following genome assembly.

More about Busco: See "Simão, F. et al. BUSCO: assessing genome assembly and annotation completeness with single-copy orthologs" https://academic.oup.com/bioinformatics/article/31/19/3210/211866

Genome assessment:



We will also map the assembled contigs to a known reference genome using the tool Quast, to see how they align. More about quast: http://quast.sourceforge.net/docs/manual.html More about the Icarus browser "Mikheenko, A. et al. Icarus: visualizer for de novo assembly evaluation" https://academic.oup.com/bioinformatics/article/32/21/3321/2415080

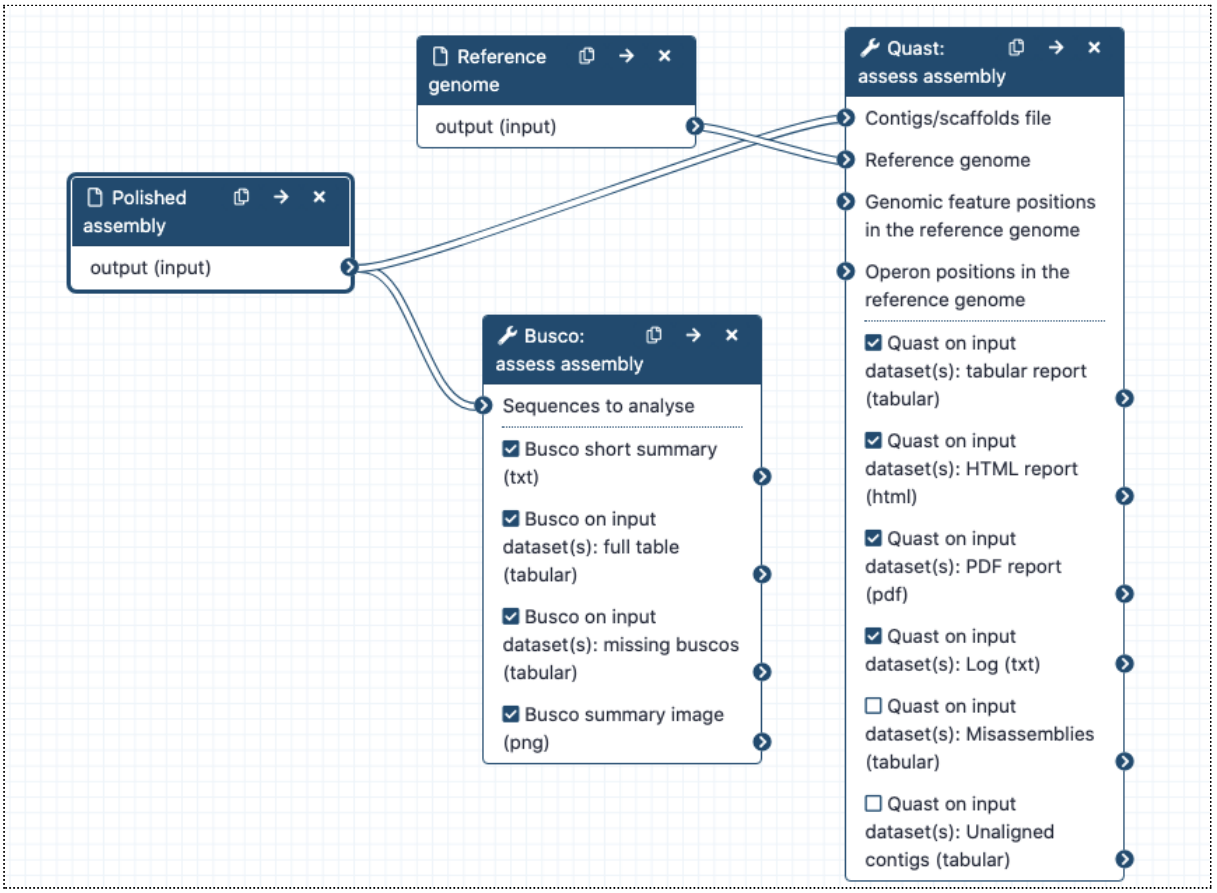
Workflow information

Workflow name	Assess genome quality
Workflow link	https://usegalaxy.org.au/u/anna/w/assess-genome
What it does	Assesses the quality of the genome assembly: generate some statistics and determine if expected genes are present; align contigs to a reference genome.
Inputs	• polished assembly • reference_genome.fasta (e.g. of a closely-related species, if available).
Outputs	• Busco table of genes found • Quast HTML report, and link to Icarus contigs browser, showing contigs aligned to a reference genome
Tools used	• Busco • Quast
Input parameters	• None required

Workflow steps	Polished assembly => Busco • First: predict genes in the assembly: using Metaeuk • Second: compare the set of predicted genes to the set of expected genes in a particular lineage. Default setting for lineage: Eukaryota Polished assembly and a reference genome => Quast • For the tutorial we will use the *Arabidopsis* genome. This is not closely related to our *Eucalyptus* species but will give an idea of how to use Quast. • Contigs/scaffolds file: polished assembly • Type of assembly: Genome • Use a reference genome: Yes • Reference genome: Arabidopsis genome • Is the genome large (> 100Mbp)? Yes. (Our test data set won't be, but this will still give us some results). • All other settings as defaults, except second last setting: Distinguish contigs with more than 50% unaligned bases as a separate group of contigs?: change to No
Report shows	• Workflow steps • Busco summary table and image • Quast tabular report
Options	**Gene prediction:** Change tool used by Busco to predict genes in the assembly: instead of Metaeuk, use Augustus (Metaeuk is meant to be faster; unsure which is better). • select: Use Augustus; Use another predefined species model; then choose from the drop down list. • select from a database of trained species models. list here: https://github.com/Gaius-Augustus/Augustus/tree/master/config/species • Note: if using Augustus: it may fail if the input assembly is too small (e.g. a test-size data assembly). It can't do the training part properly. **Compare genes found to other lineage:** Busco has databases of lineages and their expected genes.

	Option to change lineage. Not all lineages are available - there is a mix of broader and narrower lineages. • list of lineages here: https://busco.ezlab.org/list_of_lineages.html. • To see the groups in taxonomic hierarchies: Eukaryotes: https://busco.ezlab.org/frames/euka.htm • For example, if you have a plant species from Fabales, you could set that as the lineage. • The narrower the taxonomic group, the more total genes are expected.

Workflow image



Run workflow

From your current Galaxy history, run this workflow with the required input data (see table above). For more detail see earlier section: How to run a workflow in Galaxy.

Assessment results

The output is a set of Quast and Busco reports.

Busco: As this is a test dataset, the assembly is small (~ 10 million base pairs, rather than ~ 1 billion base pairs), and likely missing most of the real genome (and genes).

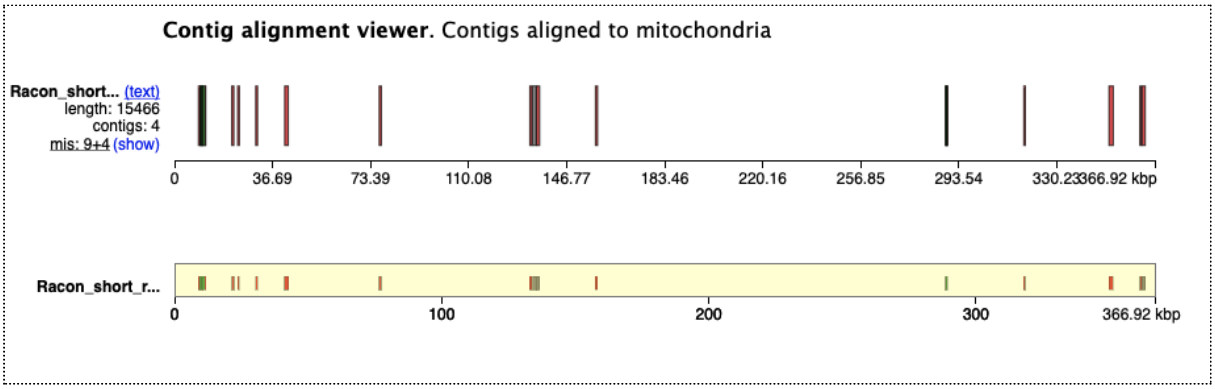
Thus, for this test case, we would expect most of the genes not to be found, but we can view the Busco results to see an example of how it works. Open the Busco short summary file: only 1 complete BUSCO has been found, out of 255 expected. You can re-run the workflow and change the lineage to "Embryophyta" to see that more BUSCOs are found (8 out of 1614).

Open the Quast HTML report, and at the top of this, click on "View in Icarus contig browser". This shows how our assembly contigs have mapped to the reference genome. For this reference genome there are 5 chromosomes and two organelles. As this reference genome species is not closely related, not many contigs have mapped well. But we can see that some of them match the organelles (which is expected, as these reads are likely to be overrepresented in the test data). For the nuclear genome, there are some matches to parts of chromosome 2 and 3.

Select a viewer of contigs aligned to one of the following reference genomes:

Genome	# fragments	Length, bp	# assemblies	Mean genome fraction, %	# misassembled blocks
1	1	30 427 671	0	0.000	0
2	1	19 698 289	1	0.068	16
3	1	23 459 830	1	0.018	10
4	1	18 585 056	0	0.000	0
5	1	26 975 502	0	0.000	0
chloroplast	1	154 478	1	13.441	13
mitochondria	1	366 924	1	4.204	18

Click on the "mitochondria" to see how the assembly contigs align to the reference mitochondrial genome. Click the -5x button to zoom out to the full length.



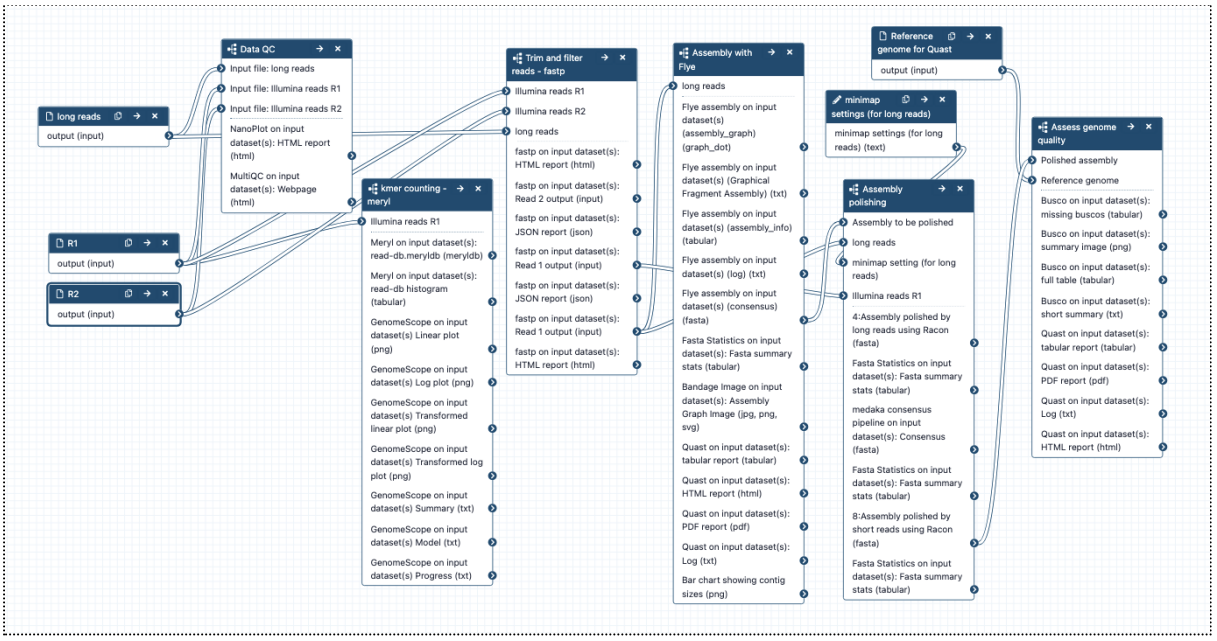
Combining workflows

We can combine these galaxy workflows into a single workflow. (See https://training.galaxyproject.org/training-material/topics/galaxy-interface/tutorials/workflow-editor/tutorial.html for more information.)

Workflow information

Workflow name	Combined workflows for large genome assembly
Workflow link	https://usegalaxy.org.au/u/anna/w/combined-large-genome-workflow
What it does	A workflow for genome assembly, containing subworkflows: • Data QC • Kmer counting • Trim and filter reads • Assembly with Flye • Assembly polishing • Assess genome quality
Inputs	• long reads and short reads in fastq format • reference genome for Quast
Outputs	• Data information - QC, kmers • Filtered, trimmed reads • Genome assembly, assembly graph, stats • Polished assembly, stats • Quality metrics - Busco, Quast
Tools used	Sum of tools in each of the subworkflows
Input parameters	None required
Workflow steps	For detail see each subworkflow
Report shows	Workflow steps
Options	• Omit some steps - e.g. Data QC and kmer counting • Replace a module with one using a different tool - e.g. change assembly tool

Combined workflow image



Run workflow

From your current Galaxy history, run this workflow with the required input data (see table above). For more detail see earlier section: How to run a workflow in Galaxy.

Next steps

Re-run with different test data

Import the data from this history and re-run:

https://usegalaxy.org.au/u/anna/h/banana-test-data

Run with your own data

See the next section.

Using your own data

This tutorial has been tested on real-sized data sets and should work with your own data. There will most likely be some modifications required to tools and settings.

In the next sections we cover how to set up Galaxy for large analyses.

Configure your Galaxy account for large analyses

If you are using real-sized large data sets, please contact the Galaxy Australia team first so we can ensure your analyses are being sent to the right high-capacity computers. email: help@genome.edu.au

How to prepare a test-sized set of data

Data sets are large and some tools can take a long time to run. It is much easier to troubleshoot and test parameters on a smaller sized data set.

Here, we will subsample our full data set to make a smaller test data set, to run on all the planned workflow steps. If this runs successfully, we can then run the analysis with the full data set.

Workflow information

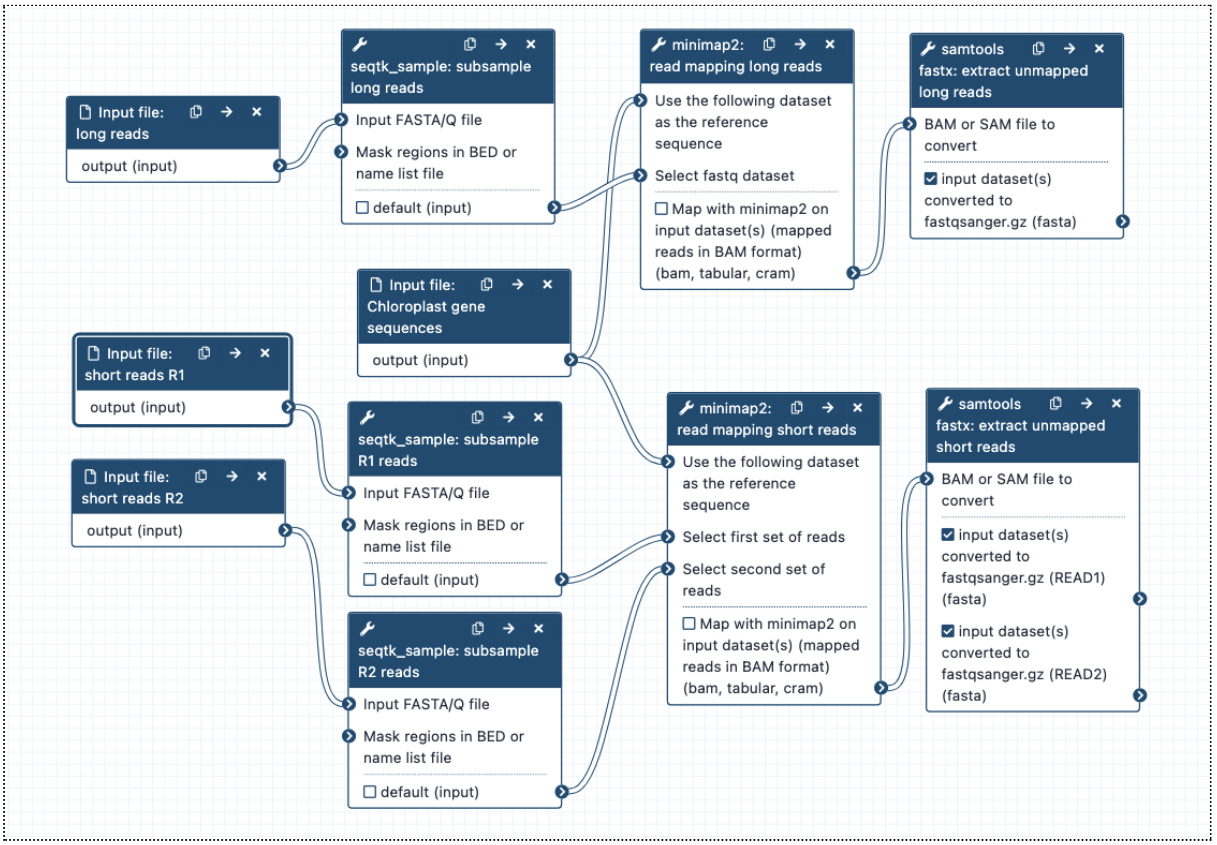
Workflow name	Prepare test data - remove chloroplast reads and subsample to 10%
Workflow link	https://usegalaxy.org.au/u/anna/w/prepare-test-data-remove-chloro-subsample-10-pc
What it does	Reduce input read file sizes for testing purposes
Inputs	Sequencing reads • long reads: longreads.fastq.qz • short reads: R1.fastq.gz, R2.fastq.gz Chloroplast gene sequences, seqs.fasta • e.g. well-conserved genes from the chloroplast are matK and rbcL. • If reads match these sequences, they are probably (but not always) from the chloroplast genome rather than the nuclear genome. • We will exclude these sequences from our read sets so they don't swamp the test data sets. • We will use sequences from a closely-related species: *Eucalyptus gunnii*, from NCBI. (rbcL sequence: accession KM360776.1; matK sequence: accession KT632904.1) • Import this file from: https://usegalaxy.org.au/u/anna/h/eucalyptus-chloroplast-gene-sequences
Outputs	subsampled read sets, with chloroplast reads filtered out
Tools used	• seqtk_seq • minimap2 • samtools fastx

Input parameters	• None required, but recommend setting the subsampling percentage to required level - default is 10% but you may prefer a smaller subsample for initial testing, particularly if your input files are very large.
Workflow steps	• Randomly subsample the reads to 10% using seqtk_seq. (Note: for paired reads, the "random seed" is set to the same number for each read set, to preserve pairing information.) • Map all reads to some closely-related chloroplast gene sequences using minimap2. For long read mapping, the default preset is "PacBio/Oxford Nanopore read to reference mapping". For short read mapping, the default preset is "short reads without splicing". This makes a bam file. • From this file, use samtools fastx to extract only the **unmapped** reads (i.e. likely non-chloroplast reads) and convert to a fastq file. For long reads, set the flag "read is unmapped". For short reads, set the flags "read is paired, read is unmapped, mate is unmapped". This takes out reads that may be chloroplast reads so they don't swamp the smaller subsampled data set. (Note: For the full data assembly step later on, it is ok to use all the reads as it is possible that some of these chloroplast sequences are integrated into the nuclear genome.) • The output files are re-named: subsampled_long_reads.fastq.gz, subsampled_R1.fastq.gz, subsampled_R2.fastq.gz
Report shows	• Workflow steps
Options	• Workflow: omit the step mapping to the chloroplast sequences. • Workflow: filter out other sets of reads (e.g. mitochondrial, potential contaminant, etc.) using a different input sequences.fasta • Prior to subsampling: filter reads by some criteria (e.g. length and/or quality). e.g. see the tools fastp or filtlong. • Subsampling: add additional options such as "drop sequences with length shorter than INT" - add a number for a minimum length required. • Subsampling: change fraction required. The default

setting is 0.1 (= 10%). e.g. may need 20% subsample of long reads for the assembly steps to be tested properly.

- Read mapping: change the "Select a profile of preset options" to best match your input data sets. The default is "PacBio/Oxford Nanopore read to reference mapping" which should be sufficient for most data sets in this workflow, but more specific settings are available.
- Other options: see the tool settings options at runtime and change as required.

Workflow image: Prepare test data



Example history
https://usegalaxy.org.au/u/anna/h/prepare-test-data---e-pauciflora

Links to tutorial data

Snow gum data: Eucalyptus pauciflora

From NCBI BioProject number: PRJNA450887; Paper: *Wang W, Das A, Kainer D, Schalamun M, Morales-Suarez A, Schwessinger B, Lanfear R; 2020, doi: 10.1093/gigascience/giz160.*

From NCBI, three read files were imported into Galaxy for this tutorial: nanopore reads (SRR7153076), and paired Illumina reads (SRR7153045). For the test data set: these were randomly subsampled to 10% of the original file size, and reads mapping to related chloroplast gene sequences (rbcL sequence: accession KM360776.1; matK sequence: accession KT632904.1) were excluded.

Test-sized data: Eucalyptus pauciflora

https://usegalaxy.org.au/u/anna/h/eucalyptus-test-data

Full-sized data: Eucalyptus pauciflora

Note that this is **not** the full data set for this entire NCBI project - only some has been downloaded. However, this set is called the "full-sized data" in comparison to the test-sized set. It is useful to use only some of these project files for this tutorial, but note that some results may be affected (e.g. kmer counting: genome size estimate is smaller than expected).

https://usegalaxy.org.au/u/anna/h/eucalytpus-full-data

Banana data: Musa acuminata

From BioProject PRJEB35002; Paper: https://www.nature.com/articles/s42003-021-02559-3?proof=t%29. Three sets of reads were downloaded from NCBI: Banana_nanopore_ERR5455028.fq.g; Banana_illumina_ERR3606950_R1.fq.gz; Banana_illumina_ERR3606950_R2.fq.gz. For the test-sized dataset, these reads were subsampled to 1%, and sequences matching chloroplast genes (FJ871594.2 Musa acuminata matK; EU017045.1 Musa acuminata rbcL) were excluded.

Test-sized data: Banana

https://usegalaxy.org.au/u/anna/h/banana-test-data

Full-sized data: Banana

https://usegalaxy.org.au/u/anna/h/banana-full-data

Links to tutorial workflows

In the top Galaxy tabs, go to Shared Data -> Workflows. These workflows are published and are in this list, they are all tagged with "LG-WF". Click on the drop-down arrow next to the workflow name and import.

Links to example histories

For the **combined assembly workflow**, run on each of these data sets.

Note: we do not recommend using full-sized data in workflows until you have set up any necessary access to additional Galaxy storage quota and high-memory nodes. Please see section: Configure your Galaxy account for large analyses.

History: Assembly of test-sized data: Eucalyptus pauciflora

https://usegalaxy.org.au/u/anna/h/eucalyptus-test-combined-wf

History: Assembly of full-sized data: Eucalyptus pauciflora

https://usegalaxy.org.au/u/anna/h/eucalyptus-full-combined-wf

History: Assembly of test-sized data: Banana

https://usegalaxy.org.au/u/anna/h/banana-test-combined-wf

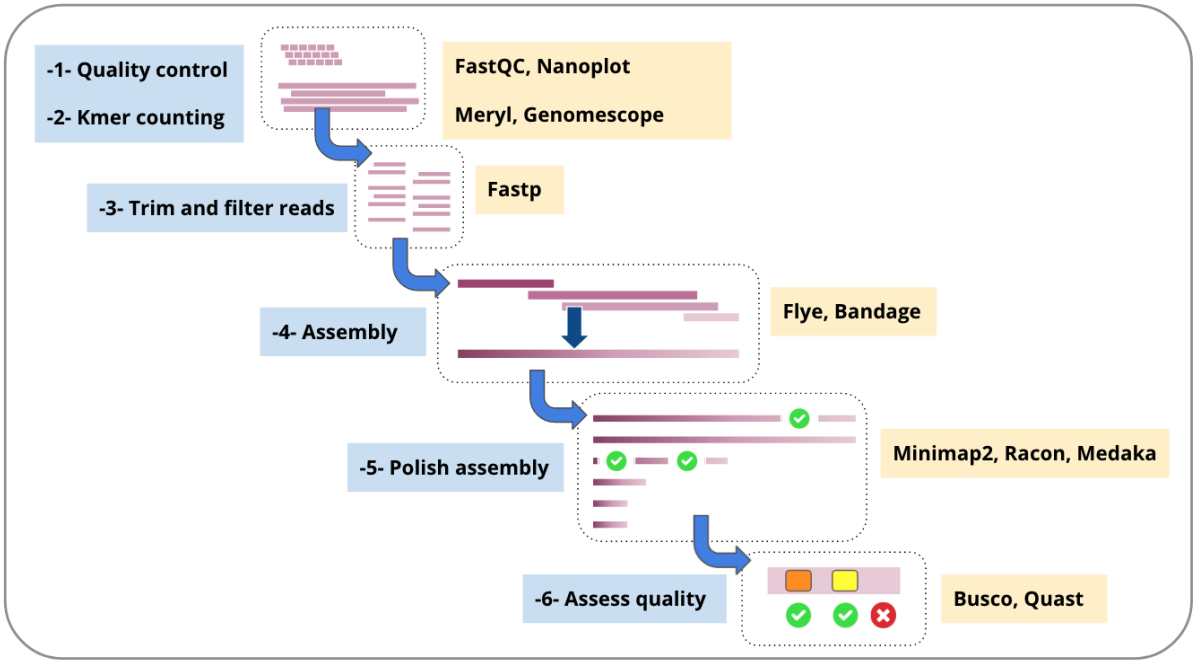
History: Assembly of full-sized data: Banana

https://usegalaxy.org.au/u/anna/h/banana-full-combined-wf

Summary

In this tutorial we have assembled sequencing reads into contigs, using tools and workflows in Galaxy Australia. Although we have used test data, these tools and workflows should work on real-sized eukaryotic data sets.

A summary of the workflow steps and the main tools used:



We plan to extend this tutorial to include addtional modules covering scaffolding, phasing haplotypes, and genome comparison. For other Galaxy Training material please see https://training.galaxyproject.org/training-material/

We hope this has been useful for both learning about genome assembly concepts and as a customisable example for your own assembly data.

With thanks to: the Galaxy Australia team for all their work and support in configuring tools and infrastructure; the global Galaxy team and the Galaxy Training Network; Rob Lanfear for supporting use of the E pauciflora data in tutorials, and the Australian BioCommons for support and feedback.