viral-ngs Documentation

Release v2.1.8.0-5-g6c38e06f

Broad Institute Viral Genomics

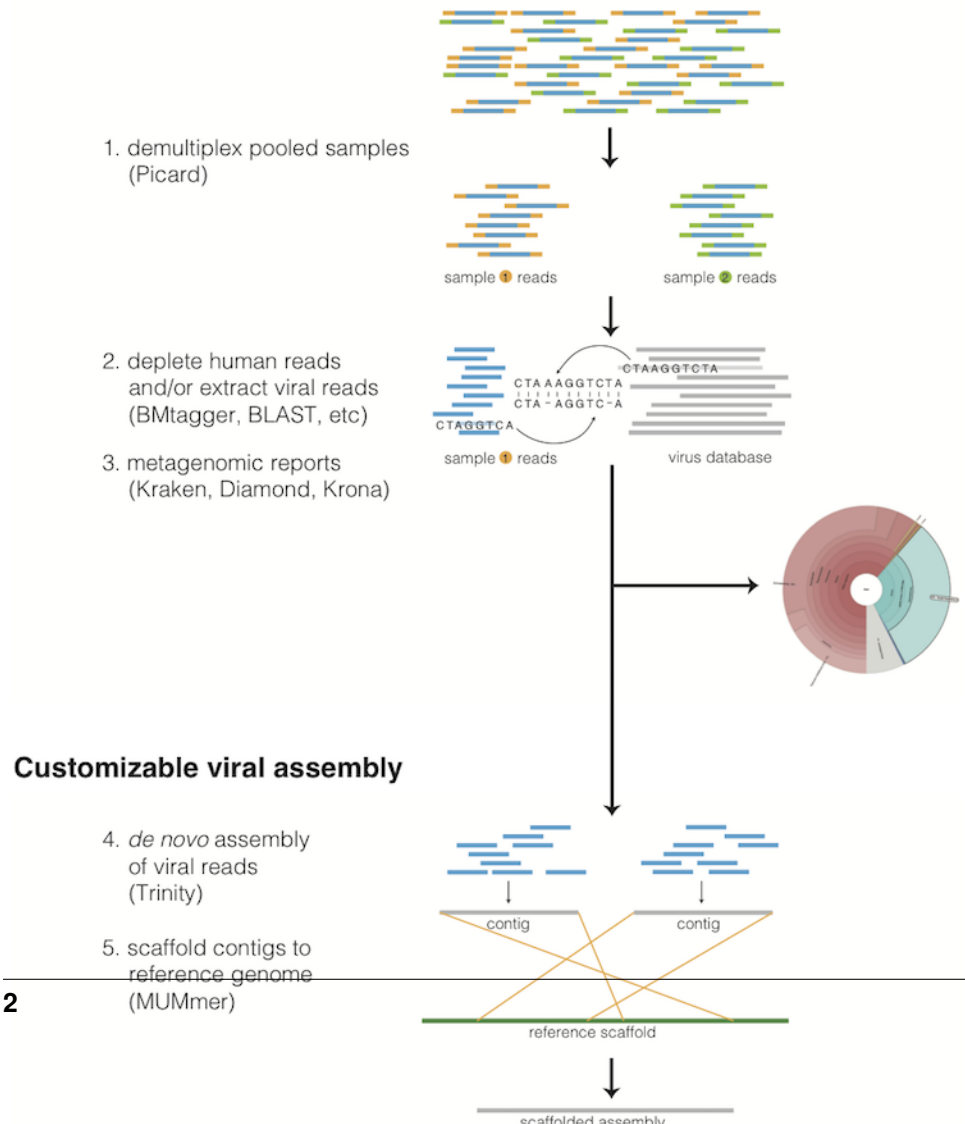
2020-07-06

Contents

CHAPTER 1

Contents

1.1 Description of the methods



1.1.1 Taxonomic read filtration

Human, contaminant, and duplicate read removal

The assembly pipeline begins by depleting paired-end reads from each sample of human and other contaminants using BMTAGGER and BLASTN, and removing PCR duplicates using M-Vicuna (a custom version of Vicuna).

Taxonomic selection

Reads are then filtered to to a genus-level database using LASTAL, quality-trimmed with Trimmomatic, and further deduplicated with PRINSEQ.

1.1.2 Viral genome analysis

Viral genome assembly

The filtered and trimmed reads are subsampled to at most 100,000 pairs. *de novo* assemby is performed using Trinity. SPAdes is also offered as an alternative *de novo* assembler. Reference-assisted assembly improvements follow (contig scaffolding, orienting, etc.) with MUMMER and MUSCLE or MAFFT. Gap2Seq is used to seal gaps between scaffolded *de novo* contigs with sequencing reads.

Each sample's reads are aligned to its *de novo* assembly using Novoalign and any remaining duplicates were removed using Picard MarkDuplicates. Variant positions in each assembly were identified using GATK IndelRealigner and UnifiedGenotyper on the read alignments. The assembly was refined to represent the major allele at each variant site, and any positions supported by fewer than three reads were changed to N.

This align-call-refine cycle is iterated twice, to minimize reference bias in the assembly.

Intrahost variant identification

Intrahost variants (iSNVs) were called from each sample's read alignments using V-Phaser2 and subjected to an initial set of filters: variant calls with fewer than five forward or reverse reads or more than a 10-fold strand bias were eliminated. iSNVs were also removed if there was more than a five-fold difference between the strand bias of the variant call and the strand bias of the reference call. Variant calls that passed these filters were additionally subjected to a 0.5% frequency filter. The final list of iSNVs contains only variant calls that passed all filters in two separate library preparations. These files infer 100% allele frequencies for all samples at an iSNV position where there was no intra-host variation within the sample, but a clear consensus call during assembly. Annotations are computed with snpEff.

1.1.3 Taxonomic read identification

Metagenomic classifiers include Kraken and Diamond. In each case, results are visualized with Krona.

1.2 Using the WDL pipelines

Rather than chaining together viral-ngs pipeline steps as series of tool commands called in isolation, it is possible to execute them as a complete automated pipeline, from processing raw sequencer output to creating files suitable for GenBank submission. This utilizes the Workflow Description Language, which is documented at: https://github.com/openwdl/wdl

There are various methods for executing these workflows on your infrastructure which are more thoroughly documented in our README.

1.3 Submitting viral sequences to NCBI

1.3.1 Register your BioProject

If you want to add samples to an existing BioProject, skip to Step 2.

1. Go to: https://submit.ncbi.nlm.nih.gov and login (new users - create new login).
2. Go to the Submissions tab and select BioProject - click on New Submission.
3. Follow the onscreen instructions and then click submit - you will receive a BioProject ID (`PRJNA###`) via email almost immediately.

1.3.2 Register your BioSamples

1. Go to: https://submit.ncbi.nlm.nih.gov and login.
2. Go to the Submissions tab and select BioSample - click on New Submission.
3. Follow instructions, selecting "batch submission type" where applicable.
4. The metadata template to use is likely: "Pathogen affecting public health" (Pathogen.cl.1.0.xlsx).
5. Follow template instructions to fill in the sheet. Pay particular attention to the Excel comments that are attached to each column header: they describe the intended content for these columns, the valid formatting, and controlled vocabulary.
    a. For example, "organism" should always match the long name that is given by the NCBI Taxonomy database for that species.
    b. Date fields seem to have multiple acceptable formats, but we prefer ISO8601 (YYYY-MM-DD) just to reduce ambiguity. Note that this format will trigger a warning when uploading, if you don't have HH:MM time values as well (it will suggest an edit for you).
    c. You will likely need to duplicate your sample_name to the host_subject_id column (or something like it)–if you do not, then any samples that happen to have the same attribute values will trigger an error when trying to register new BioSamples because they look like duplicates. Assuming that your sample_names are one-to-one corresponding to a human patient, host_subject_id is probably the most appropriate place to duplicate the value in order to make all entries unique.
    d. Populate the isolate column using the naming convention you want to apply to this organism (most viral species have a specific, structured naming convention you should follow). Our workflow will re-use this value for the Genbank record name.
6. Export to text and submit as .txt file. You will receive BioSamples IDs (`SAMN####`) via email (often 1-2 days later).
7. After NCBI accepts your submission and registers your samples, retrieve the text-formatted "attributes table" associated with this submission from the portal at https://submit.ncbi.nlm.nih.gov/subs/ and clicking on "Download attributes file with BioSample accessions". You will need this file later. Do not use the file that was attached to the NCBI response email–it does not contain the full record and is formatted differently.
8. If you wish to amend/correct any metadata in your submissions, you can always do so at a future time – however, you will need BioSample IDs before any of the following steps, so it's best to register as soon as you have collection_date and sample_name for everything. This can be a super-set of anything you submit to NCBI in the

future (Genbank or SRA), so we typically register BioSamples for every viral sample we *attempt* to sequence, regardless of whether we successfully sequenced it or not.

1.3.3 Set up an NCBI author template

If different author lists are used for different sets of samples, create a new .sbt file for each list

1. Go to: https://submit.ncbi.nlm.nih.gov/genbank/template/submission/
2. Fill out the form including all authors and submitter information (if unpublished, the reference title can be just a general description of the project).
3. At the end of the form, include the BioProject number from Step 1 but NOT the BioSample number'
4. Click "create template" which will download an .sbt file to your computer'
5. Save file as "authors.sbt" or similar. If you have multiple author files, give each file a different name and prep your submissions as separate batches, one for each authors.sbt file.

1.3.4 Prepare requisite input files for your submission batches

1. Stage the above files you've prepared and other requisite inputs into the environment you plan to execute the *genbank* WDL workflow. If that is Terra, push these files into the appropriate GCS bucket, if DNAnexus, drop your files there. If you plan to execute locally (e.g. with `miniwdl run`), move the files to an appropriate directory on your machine. The files you will need are the following:
    a. The files you prepared above: the submission template (authors.sbt) and the biosample attributes table (attributes.tsv).
    b. All of the assemblies you want to submit. These should be in fasta files, one per genome. Multi-segment/multi-chromosome genomes (such as Lassa virus, Influenza A, etc) should contain all segments within one fasta file.
    c. Your reference genome, as a set of fasta files, one per segment/chromosome. The fasta sequence headers should be Genbank accession numbers. This can come directly from Genbank.
    d. Your reference gene annotations, as a set of TBL files, one per segment/chromosome. These must correspond to the accessions in your reference genome. These must be presented in the same order as the reference genome fasta files, which must also be in the same order as all the sequences in all of your assembled fasta files.
    e. A genome coverage table as a two-column tabular text file (optional, but helpful).
    f. The organism name (which should match what NCBI taxonomy calls the species you are submitting for). This is a string input to the workflow, not a file.
    g. The sequencing technology used. This is a string input, not a file.
    h. The NCBI Taxonomy taxid. This is a numeric input, not a file.
2. The reference genome you provide should be annotated in the way you want your genomes annotated on NCBI. If one doesn't exist, see the addendum below about creating your own feature list.
3. Note that you will have to run the pipeline separately for each virus you are submitting AND separately for each author list.

1.3.5 Run the genbank submission pipeline

1. Run the *genbank* WDL workflow. Most of the metadata files described above (BioSample map, source modifier table, genome coverage table) are allowed to be a super-set of the samples you are submitting–the extra metadata will be ignored by the workflow. The samples that are included in this batch are the ones you provide to the `assemblies_fasta` input field. Any missing samples in the metadata inputs should not cause failures, but will produce less descriptive submission files. Viral genomes should set molType to `cRNA`.
2. The *genbank* workflow performs the following steps: it aligns your assemblies against a Genbank reference sequence, transfers gene annotation from that Genbank reference into your assemblies' coordinate spaces, and then takes your genomes, the transferred annotations, and all of the sample metadata prepared above, and produces a zipped bundle that you send to NCBI. There are two zip bundles: `sequins_only.zip` is the file to email to NCBI. `all_files.zip` contains a full set of files for your inspection prior to submission.
3. In the `all_files.zip` output, for each sample, you will see a `.sqn`, `.gbf`, `.val`, and `.tbl` file. You should also see an `errorsummary.val` file that you can use to check for annotation errors (or you can check the `.val` file for each sample individually). Ideally, your samples should be error-free before you submit them to NCBI unless you're confident enough in the genomic evidence for unusual coding effects and frameshifts. For an explanation of the cryptic error messages, see: https://www.ncbi.nlm.nih.gov/genbank/genome_validation/.
4. We currently use a bioconda wrapper of NCBI's *tbl2asn* tool called *tbl2asn-forever*. This works around some deficiencies in NCBI's tool but has the side effect of setting the submission date to Jan 1, 2019 for all submission, regardless of today's date. Unfortunately, until NCBI releases a fixed tool, you will need to search-replace the date in the SQN files in a text editor prior to submission.
5. Check your `.gbf` files for a preview of what your genbank entries will look like. Once you are happy with your files email the `sequins_only.zip` file to gb-sub@ncbi.nlm.nih.gov.
6. It often takes 2-8 weeks to receive a response and accession numbers for your samples. Do follow up if you haven't heard anything for a few weeks!

1.4 WDL Workflows

Documentation for each workflow is provided here. Although there are many workflows that serve different functions, some of the primary workflows we use most often include:

- *demux_plus* (on every sequencing run)
- *classify_multi* (metagenomics & host depletion)
- *assemble_denovo* (for most viruses)
- *assemble_refbased* (for less diverse viruses, such as those from single point source human outbreaks)
- *augur_from_assemblies* (for nextstrain-based visualization of phylogeny)
- *genbank* (for NCBI Genbank submission)

1.4.1 align_and_count_report

Align reads to reference with minimap2 and count the number of hits. Results are returned in the format of 'samtools idxstats'.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.2 align_and_count_multiple_report

Count the number of times reads map to provided reference sequences. Useful for counting spike-ins, etc.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.3 align_and_plot

Align reads to reference and produce coverage plots and statistics.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.4 assemble_denovo

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.5 assemble_refbased

Reference-based microbial consensus calling. Aligns NGS reads to a singular reference genome, calls a new consensus sequence, and emits: new assembly, reads aligned to provided reference, reads aligned to new assembly, various figures of merit, plots, and QC metrics. The user may provide unaligned reads spread across multiple input files and this workflow will parallelize alignment per input file before merging results prior to consensus calling.

Inputs

Required inputs

Other common inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.6 augur_from_assemblies

Align assemblies, build trees, and convert to json representation suitable for Nextstrain visualization. See https://nextstrain.org/docs/getting-started/ and https://nextstrain-augur.readthedocs.io/en/stable/

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.7 augur_from_beast_mcc

Visualize BEAST output with Nextstrain. This workflow converts a BEAST MCC tree (.tree file) into an Auspice v2 json file. See https://nextstrain-augur.readthedocs.io/en/stable/faq/import-beast.html for details.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.8 augur_from_msa

Build trees, and convert to json representation suitable for Nextstrain visualization. See https://nextstrain.org/docs/getting-started/ and https://nextstrain-augur.readthedocs.io/en/stable/

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.9 augur_from_newick

Convert a newick formatted phylogenetic tree into a json suitable for auspice visualization. See https://nextstrain-augur.readthedocs.io/en/stable/usage/cli/export.html

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.10 bams_multiqc

Run FastQC on a set of BAM files, and then MultiQC to summarize all outputs.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.11 beast_gpu

Runs BEAST (v1) on a GPU instance. Use with care–this can be expensive if run incorrectly.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.12 classify_kaiju

Taxonomic classification of reads with kaiju.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.13 classify_kraken2

Taxonomic classification of sequences via kraken2 (or kraken2x, depending on the database provided).

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.14 classify_krakenuniq

Taxonomic classification of reads using krakenuniq v1.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.15 classify_multi

Runs raw reads through taxonomic classification (Kraken2), human read depletion (based on Kraken2), de novo assembly (SPAdes), and FASTQC/multiQC of reads.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.16 classify_single

Runs raw reads through taxonomic classification (Kraken2), human read depletion (based on Kraken2), de novo assembly (SPAdes), and FASTQC/multiQC of reads.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.17 contigs

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.18 coverage_table

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.19 demux_metag

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.20 demux_only

Picard-based demultiplexing and basecalling from a tarball of a raw BCL directory.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.21 demux_plus

Picard-based demultiplexing and basecalling from a tarball of a raw BCL directory, followed by basic metagenomics and QC metrics. Intended for automatic triggering post upload on DNAnexus.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.22 deplete_only

Taxonomic depletion of reads matching unwanted taxa (such as human).

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.23 diff_genome_sets

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.24 downsample

Random subsampling of reads.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.25 fastq_to_ubam

Convert reads from fastq format (single or paired) to unaligned BAM format.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.26 fetch_annotations

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.27 fetch_sra_to_bam

Retrieve reads from the NCBI Short Read Archive in unaligned BAM format with relevant metadata encoded.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.28 filter_classified_bam_to_taxa

Taxonomic filtration of reads utilizing output from a classifier such as kraken1/2/uniq. Can filter out or filter to a specified taxonomic grouping.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.29 filter_sequences

Filter and subsample a sequence set. See https://nextstrain-augur.readthedocs.io/en/stable/usage/cli/filter.html

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.30 genbank

Prepare assemblies for Genbank submission. This includes annotation by simple coordinate transfer from Genbank annotations and a multiple alignment. See https://viral-pipelines.readthedocs.io/en/latest/ncbi_submission.html for details.

Inputs

Required inputs

Other common inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.31 isnvs_merge_to_vcf

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.32 isnvs_one_sample

Intrahost variant calling with V-Phaser2. Requires an assembled genome and a BAM of aligned reads against that same genome.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.33 kraken2_build

Build a Kraken2 (or 2X) database.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.34 mafft

MAFFT multiple-alignment for a set of possibly multi-segment genomes.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.35 mafft_and_snp

Align assemblies with mafft and find SNPs with snp-sites.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.36 mafft_and_trim

MAFFT based multiple alignment followed by trimal-based edge trimming.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.37 merge_bams

Merge, reheader, or merge-and-reheader BAM files.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.38 merge_metagenomics

Combine metagenomic reports from single samples into an aggregate report.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.39 merge_tar_chunks

Combine multiple tar files (possibly compressed by gzip, bz2, lz4, zstd, etc) into a single tar file. Originally meant for combining streaming upload chunks from a sequencing run.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.40 merge_vcfs

Merge VCFs from multiple samples using GATK3.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.41 merge_vcfs_and_annotate

Merge VCFs emitted by GATK UnifiedGenotyper and annotate with snpEff.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.42 multiqc_only

Combine multiple FastQC reports into a single MultiQC summary.

Inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.43 scaffold_and_refine

Scaffold de novo contigs against a set of possible references and subsequently polish with reads.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.44 subsample_by_metadata

Filter and subsample a sequence set. See https://nextstrain-augur.readthedocs.io/en/stable/usage/cli/filter.html

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.45 trimal

Trim a multiple sequence alignment with Trimal.

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)