viral-ngs Documentation

Release v2.0.21.4

Broad Institute Viral Genomics

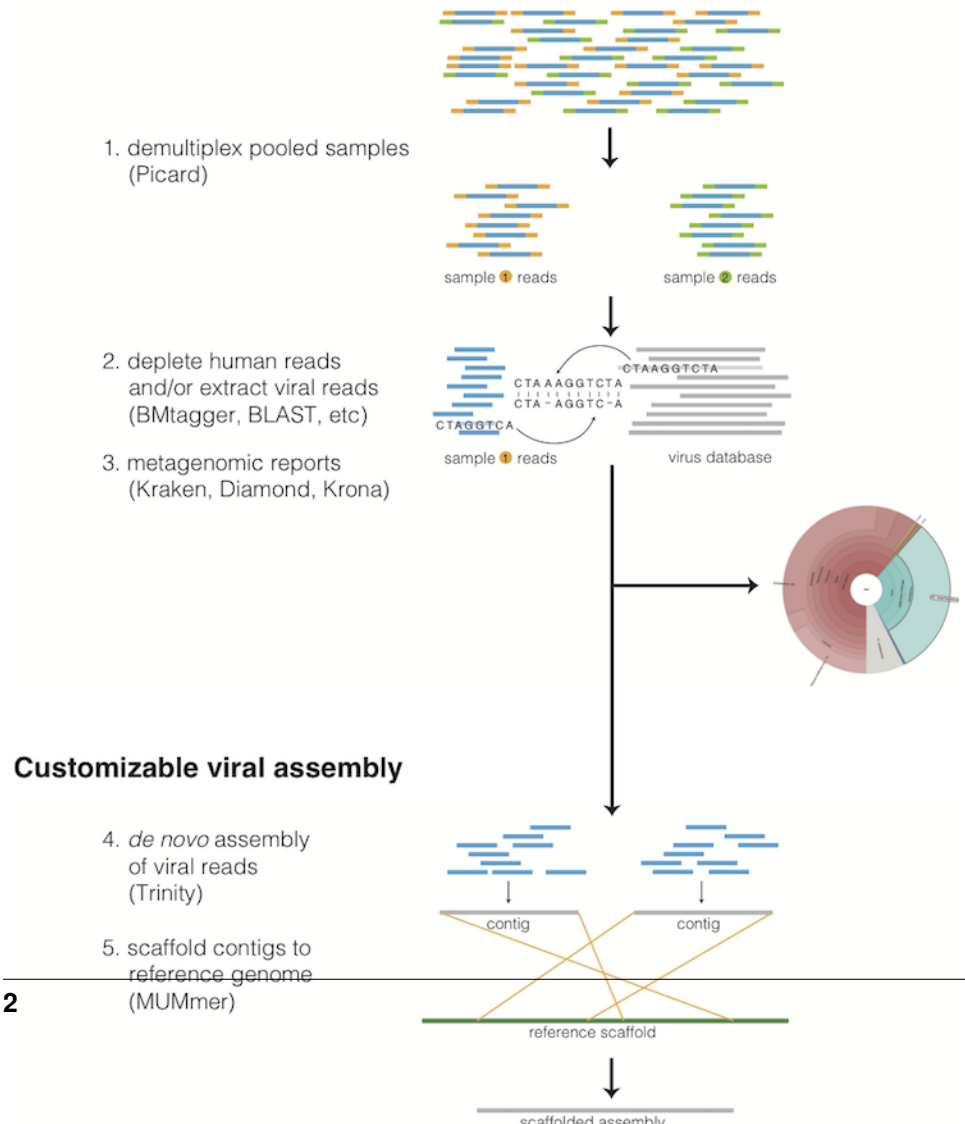
2020-05-18

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.
    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.
6. Export to text and submit as .txt file. You will receive BioSamples IDs (`SAMN####`) via email (often 1-2 days later).
7. 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 Set up the BioSample map file

1. Set up an Excel spreadsheet in exactly the format below:

sample	BioSample
sample1-1	SAMNxxxxxxxxx
sample2-1	SAMNxxxxxxxxx

2. The BioSample is the BioSample number (i.e., `SAMNxxxxxxxxx`) given to you by NCBI.
3. The sample name should match the FASTA header (not necessarily the file name).

a. Make sure your FASTA headers include segment numbers (i.e., IRF001-1) – viral-ngs will fail otherwise!

b. If submitting a segmented virus (i.e., Lassa virus), each line should be a different segment, see example below (assumes sample2 is a 2-segmented virus)

c. For samples with multiple segments, the BioSample number should be the same for all segments

sample	BioSample
sample1-1	SAMN04488486
sample2-1	SAMN04488657
sample2-2	SAMN04488657
sample3-1	SAMN04489002

4. Save the file as as a tab delimited text file (e.g. "biosample-map.txt"). This file can describe *more* samples than you plan to run in a submission batch (the extras will be ignored).
5. If preparing the file on a Mac computer in Microsoft Excel (which saves tab files in a 20th-century era OS9 format), ensure that tabs and newlines are entered correctly by opening the file (via the command line) in an editor such as Nano and unchecking the [Mac-format] option (in Nano: edit the file, save the file, then click OPTION-M). You can also opt to create this file directly in a text editor, ensuring there is exactly one tab character between columns (i.e., sample<tab>BioSample in the first row). Command line converters such as `mac2unix` also work.

1.3.5 Set up the metadata file (aka Source Modifier Table)

1. Set up an Excel spreadsheet in exactly the format below

   a. This example shows sample2 as a 2-segmented virus.

   b. All data should be on the same line (there are 9 columns). Here they are shown as separate tables simply for space reasons.

   c. The "Sequence_ID" should match the "sample" field in the BioSample map (see Step 4). Note that this should match the FASTA header.

d. Shown are the some of the fields we typically use in NCBI submissions, but fields can be added or removed to suit your sample needs. Other fields we often include are: "isolation_source" (i.e., serum), "collected_by" (i.e., Redeemer's University), and "genotype". Here are more details and examples provided by NCBI: https://www.ncbi.nlm.nih.gov/WebSub/html/help/genbank-source-table.html. A longer list of accepted column headers is provided here: https://www.ncbi.nlm.nih.gov/Sequin/modifiers.html.

e. The database cross-reference (db_xref) field number can be obtained by navigating to https://www.ncbi.nlm.nih.gov/taxonomy, searching for the organism of interest, and copying the "Taxonomy ID" number from the webpage.

Sequence_ID	collection_date	country	isolate	organism	lab_host	host	db_xref	note
sample-1	10-Mar-2014	Nigeria	Ebola virus/H.sapiens-tc/GIN/2014/Makona-C05	Zaire ebolavirus	Vero cells	Homo sapiens	taxon:186538	Harvest date: 01-Jan-2016; passaged 2x in cell culture (parent stock: SAMN01110234)
sample-1	22-Mar-2014	Nigeria	Lassa virus Macenta	Lassa mammarenavirus	Vero cells	Homo sapiens	taxon:11620	
sample-2	22-Mar-2014	Nigeria	Lassa virus Macenta	Lassa mammarenavirus	Vero cells	Homo sapiens	taxon:11620	
sample-1	16-Mar-2014	Nigeria	Ebola virus/H.sapiens-tc/GIN/2014/Makona-1121	Zaire ebolavirus	Vero cells	Homo sapiens	taxon:186538	This sample was collected by Dr. Blood from a very sick patient.

2. The data in this table is what actually shows up on NCBI with the genome. In many cases, it is a subset of the metadata you submitted when you registered the BioSamples.

3. Save this table as sample_meta.txt. If you make the file in Excel, double check the date formatting is preserved when you save – it should be dd-mmm-yyyy format. This file can describe *more* samples than you plan to run in a submission batch (the extras will be ignored).

4. If preparing the file on a Mac computer in Microsoft Excel (which saves tab files in a 20th-century era OS9 format), ensure that tabs and newlines are entered correctly by opening the file (via the command line) in an editor such as Nano and unchecking the [Mac-format] option (in Nano: edit the file, save the file, then click OPTION-M). You can also opt to create this file directly in a text editor, ensuring there is exactly one tab character between columns (i.e., sample<tab>BioSample in the first row). Command line converters such as `mac2unix` also work.

1.3.6 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), the biosample map (biosample-map.txt), and the source modifier table (sample_meta.txt)

   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 fasta file. Multi-segment/multi-chromosome genomes should contain all segments within one fasta file. The fasta sequence headers should be Genbank accession numbers.

   d. Your reference gene annotations, as a series of TBL files, one per segment/chromosome. These must correspond to the accessions in you reference genome.

   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.

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.

1.3.7 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.

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. 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.

5. 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_krakenuniq* (included in demux_plus)

- *assemble_denovo* (for most viruses)
- *assemble_refbased* (for less diverse viruses, such as those from single point source human outbreaks)
- *build_augur_tree* (for nextstrain-based visualization of phylogeny)
- *genbank* (for NCBI Genbank submission)

1.4.1 align_and_count_report

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.2 align_and_plot

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.3 assemble_denovo

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.4 assemble_refbased

Reference-based microbial consensus calling. Aligns short 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)

1.4.5 bams_multiqc

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.6 beast_gpu

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.7 beast_to_auspice

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 build_augur_tree

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)

1.4.9 classify_kaiju

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.10 classify_kraken2

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.11 classify_krakenuniq

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.12 classify_multi

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

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.13 contigs

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.14 coverage_table

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.15 demux_metag

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.16 demux_only

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.17 demux_plus

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.18 deplete_only

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.19 downsample

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.20 fastq_to_ubam

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.21 fetch_annotations

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.22 multi_Fetch_SRA_to_BAM

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.23 filter_classified_bam_to_taxa

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.24 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)

1.4.25 isnvs_merge_to_vcf

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.26 isnvs_one_sample

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.27 kraken2_build

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.28 mafft

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.29 mafft_and_trim

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.30 merge_bams

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.31 merge_metagenomics

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.32 merge_tar_chunks

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.33 multiqc_only

Inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.34 newick_to_auspice

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)

1.4.35 scaffold_and_refine

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)

1.4.36 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)

1.4.37 trimal

Inputs

Required inputs

Other inputs

Generated using WDL AID (0.1.1)