21/08/2024

Nano Vi-Phy Manual

By Kaiden R. Sewradj

Available at: github.com/Kaiden-exe/Nano_Vi-Phy

Supervised by: Ellen Carbo, Fokla Zorgdrager & Janke Schinkel
Amsterdam UMC

Nano Vi-Phy is a pipeline designed for processing Oxford Nanopore Technologies (ONT) data into whole genome sequences and using these sequences for building a phylogenetic tree with a background data set. Nano Vi-Phy was inspired by shiver (Wymant *et al*., 2018). The goal I set for this pipeline was: set up the config file, press play, get a coffee break and when you're back, you got your tree file.

Nano Vi-Phy was made for HIV whole genome sequencing, but should work on other small (viral) genomes and selected regions as well, just adjust parameters accordingly. Instructions for customisable options will be in the manual. There is no guarantee that everything won't crash when using this on large genomes. Try it at your own risk if you want.

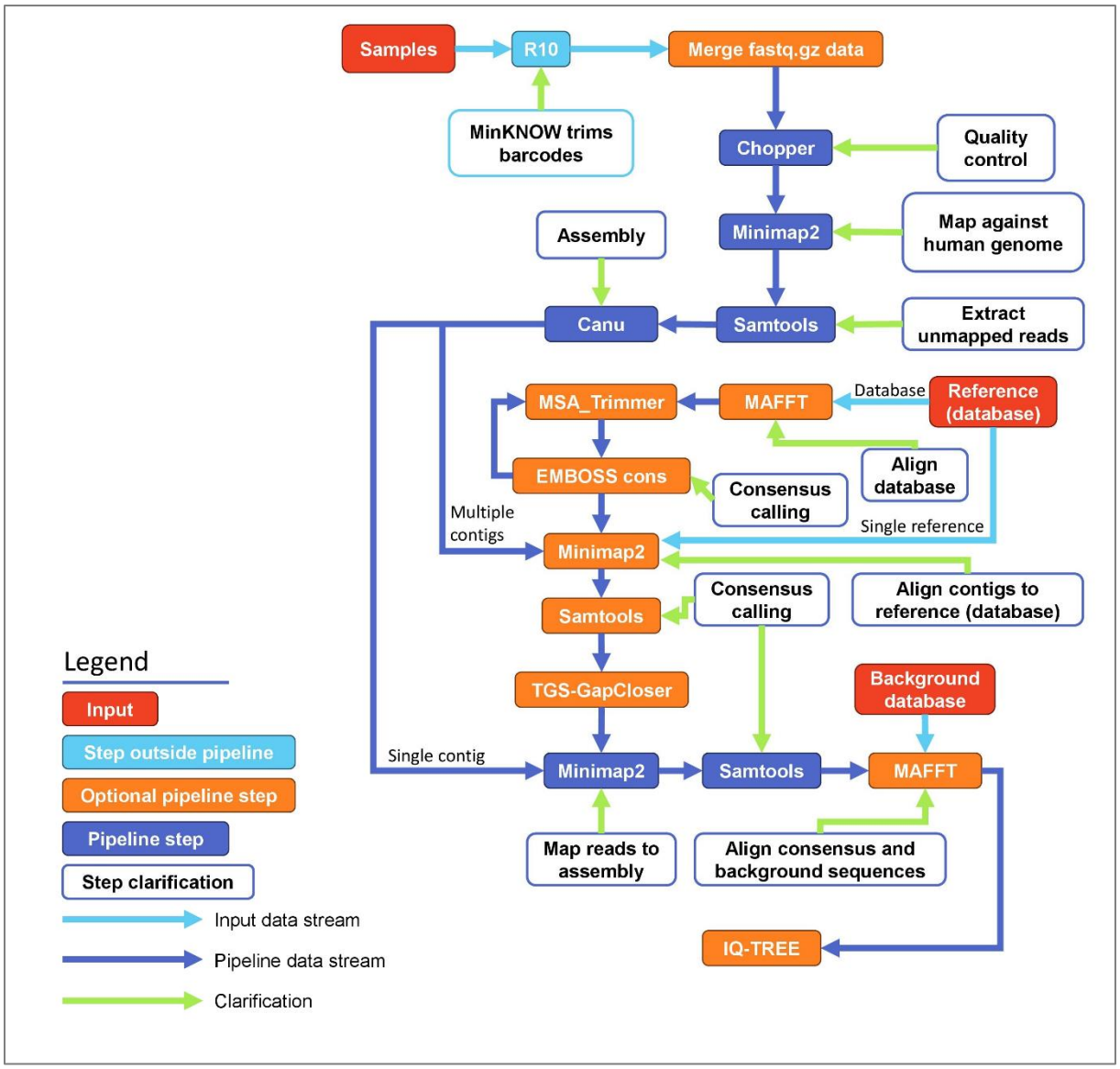

Table of Contents

0. Installation

Clone the repo

```
git clone --recurse-submodules https://github.com/Kaiden-exe/Nano_Vi-Phy.git
```

For git <2.13 use `--recursive` instead of `--recurse-submodules`.

```
cd Nano_Vi-Phy
```

[Optional] Create a new conda environment

```
conda create -n nanoviphy
```

```
conda activate nanoviphy
```

Install dependencies

Dependencies can be installed manually, but if you have (mini)conda installed, the main file can be used to automatically install all dependencies that you do not already have: `bash nanoviphy.sh --install`. No matter what method you choose, please run `bash nanoviphy.sh --test` before you run anything else. This way you can check if all dependencies are working.

WARNING: `bash nanoviphy.sh --install` installs canu in the tools directory, but it doesn't add the executable to PATH. You should do that manually:

```
export PATH=path/to/tools/canu-2.2/bin:$PATH
```

Nano Vi-Phy uses:

- Biopython v1.84 (Cock, *et al.*, 2009)
- Canu v2.2 (Koren *et al.*, 2017)
- Chopper v0.8.0 (De Coster & Rademakers, 2023)
- EMBOSS v6.6.0 (Rice *et al.*, 2000)
- IQ-TREE v2.3.6 (Minh *et al.*, 2020)
- MAFFT v7.526 (Katoh & Standley, 2013)
- minimap2 v2.28 (Li, 2018)
- MSA_Trimmer (Kremer, 2019)
- Python v3.12.2
- samtools v1.20 (Danecek *et al.*, 2021)
- TGS-GapCloser v1.2.1 (Xu *et al.*, 2020)

1. What is in the repo?

nanoviphy.sh

The main file. For options run `bash nanoviphy.sh -h`

config.sh

Most of the tweaking and personalisation happens in here. Instructions are in the comments inside this file, but will be more elaborate in the next section.

code

Scripts are in here. Some also function as a (semi-)standalone script. Such as:

- check_MSA_func.sh : determines whether a FASTA file is aligned or not by checking for '-' outside of headers. Returns 'True' or 'False' to stdout.
- fasta_header_replace.py : Replaces the header of a FASTA file
- human_filter_func.sh : Aligns reads to a reference and filters out reads that map to the reference. Settings were tweaked for human genome, but should work with other references as well.
- merge_func.sh : Merges fastq.gz files in a folder into a single FASTQ file.
- noambig.py : Replaces all IUPAC bases that are not A, T, G or C. Takes base frequency into account when replacing bases.
- reads_cons.sh : Aligns reads to a reference and calls consensus.
- ref_cons.sh : Requires MSA_Trimmer and noambig.py. Creates a consensus sequence from a multi-FASTA file.
- scaffold_func.sh : Scaffolds contigs through alignment to a reference and consensus calling.

All these scripts have some documentation inside them if you want to use them separately.

tools

MSA_Trimmer is in here and canu if you let the main script install it.

2. Configuration

The ‘input’ and ‘parameters’ sections are important to change to taste. The ‘logging’ files just contain file names. The last section with ‘new directories’ just defines the names of the output directories. None of these directories nor their parent directories need to be made before running.

Input

data=datadir

See right for the expected layout of datadir. The subdirectories can be named anything, but need unique names. The names of the subdirectories will be used as an identifier. I advise against special characters and spaces as well. There is no limit to the amount of subdirectories or fastq.gz files.

```
datadir
|--- barcode1
|    |--- file1.fastq.gz
|    |--- file2.fastq.gz
|    |--- file3.fastq.gz
|
|--- barcode2
|    |--- file1.fastq.gz
|    |--- file2.fastq.gz
|    |--- file3.fastq.gz
…
```

humanRef=reference.fna

Used to filter out human reads. The reference can be from another organism to filter out those reads. I recommend this reference.

refDB=reference.fasta

Used for scaffolding. This can be a single-sequence FASTA or multi-sequence FASTA. Using a single sequence is quicker. With multiple sequences, a consensus sequence with a maximum length of genomeSize will be created. If your data is not whole genome, do not provide a whole genome reference. Your consensus sequence will have trailing N’s covering the rest of the genome.

backgroundDB=background.fasta

A background dataset to include in the phylogenetic tree. This can be an MSA or unaligned sequences.

Parameters

phyloAnalysis=’Y’

Turn phylogenetic analysis on (Y) or off (N). If you’re just interested in the assembly and mapping reads to assembly part, you can skip the phylogenetic analysis.

premerged=’N’

If you already merged the fastq.gz files into FASTQ files, you can skip the merging step by changing the value to ‘Y’.

genomeSize=9500

The expected genome size. I recommend going for a little larger than you expect it to be if your sample organism is known for having a high indel rate. This value is used for both canu and for creating a consensus of reference sequences for scaffolding.

minReadLength=200; maxReadLength=4000
Minimum/maximum read length to include. These values are both used for chopper, and canu only uses the minimum length. The values depend on lab methods.

phred=20
Minimum Phred score used for filtering low quality reads with chopper. I recommend 20 if you have high coverage.

trimDB='N'
Say you sequenced the whole genome, but now you're only interested in a certain region. Just make the background database only of that region and change this value to 'Y' to trim the rest of the whole genome off after alignment with the background dataset.

threads=8
Maximum amount of threads to use.

Logging files

timer=timer.txt
Creates a csv with each process and the amount of seconds it took.

logfile=log.log
This file contains the stderr outputs. It is not very readable, but it might come in handy during troubleshooting.

sumFile=log.summary
This file contains the commands used on which files with which parameters. Also some summary stats, for example: how many reads pass the quality control, how many contigs a sample has after assembly and which samples needed to be scaffolded.

New directories

mergedData=output/merged-data
Output for merged FASTQ files. If you already have merged the fastq.gz files into FASTQ files, change the value to the location of the FASTQ files.

```
merged-data
|--- barcode1.fastq
|--- barcode2.fastq
|--- barcode3.fastq
```

tmp=tmp
Temporary file directory. Will be gone by the end.

chopperOutput=output/trimmed-reads
Chopper output is here.

hivReads=output/hiv-reads
Filtered reads are here.

contigs=output/contigs
Canu output is here.

scaffoldDir=output/scaffolds
Scaffolds are here.

gapFillDir=output/gapfilled-scaffolds
TGS-GapCloser output is here.

assemblyAln=output/assembly-aligned
Reads aligned to assembly and the consensus called from that are here.

treeOutput=output/tree
IQ-TREE output is here.

3. Usage

1) After downloading the repository and installing all dependencies, test all dependencies first: `bash nanoviphy.sh -t`
2) After testing, make sure you adjusted the configuration file (config.sh) to your needs. (See section 2)
3) To run Nano Vi-Phy: `bash nanoviphy.sh -c config.sh`
4) Break time! ☕

Expected performance

Nano Vi-Phy was tested on HIV whole genome sequencing data of 6 samples and ran with 8 threads.

Process	time (sec.)	Data to process
Merging data	23.90	1 361 442 reads
Trimming with chopper	4.92	1 361 442 reads
Extracting non-human reads	303.99	234 849 reads
Assembling contigs	125.46	216 374 reads
Building consensus from reference database	10.87	64 sequences
Scaffolding	0.12	15 contigs
Filling gaps	6.24	3 samples
Mapping reads and calling consensus	110.52	216 374 reads
Fixing consensus headers	0.06	6 samples
Aligning consensus to background	24.45	64 background sequences + 6 consensus sequences
Building tree with IQ-TREE	374.99	70 sequences

Total time for this run was 16 minutes and 25.5 seconds

4. Output

mergedData=output/merged-data

Each sample's fastq.gz files in one FASTQ file.

```
merged-data
|--- barcode1.fastq
|--- barcode2.fastq
|--- barcode3.fastq
|--- barcode4.fastq
```

chopperOutput=output/trimmed-reads

Quality controlled reads in one FASTQ file. If none of the reads passed the quality check, there will be no fastq file here for that sample.

```
trimmed-reads
|--- barcode1.fastq
|--- barcode2.fastq
|--- barcode3.fastq
```

hivReads=output/hiv-reads

Filtered reads. BAM files with the alignment of the reads against the reference and FASTQ files with the unmapped reads. If the sample needed to be scaffolded and gap-filled, there is also a FASTA file with all the reads.

```
hiv-reads
|--- barcode1.bam
|--- barcode1.fastq
|--- barcode2.bam
|--- barcode2.fasta
|--- barcode2.fastq
|--- barcode3.bam
|--- barcode3.fasta
|--- barcode3.fastq
```

contigs=output/contigs

Nano Vi-Phy only uses the contig files, but the rest of the canu output is still here in case you're interested.

```
contigs
|--- barcode1
|    …
|    |--- barcode1.contigs.fasta
|    …
|--- barcode2
|    …
|    |--- barcode2.contigs.fasta
|    …
|--- barcode3
|    …
|    |--- barcode3.contigs.fasta
|    …
```

scaffoldDir=output/scaffolds

Contains the scaffold files and the BAM files of how the contigs were aligned to the reference to create the scaffold. If the reference was a consensus built from a multi-FASTA file, the consensus sequence will also be in this folder, along with a version that does not contain ambiguous bases [W, S, M, K, R, Y, B, D, H, V, N]. If the reference database was not already aligned, the alignment and the trimmed alignment will also be here. Alignment gets trimmed to avoid the consensus sequence from being too long.

```
scaffolds
|--- barcode2_contigsaln.bam
|--- barcode2_scaffold.fasta
|--- barcode3_contigsaln.bam
|--- barcode3_scaffold.fasta
|--- database_aligned.fasta
|--- database_aligned_trimmed.fasta
|--- database_consensus.fasta
|--- database_consensus_noambig.fasta
```

gapFillDir=output/gapfilled-scaffolds

Some scaffolds already have no internal gaps. Those will show up in scaffoldDir, but not here. Nano Vi-Phy only uses the filled scaffold, but the rest of the TGS-GapCloser output will remain here if you're interested.

```
gapfilled-scaffolds
|--- barcode3
|    …
|    |--- barcode3.scaff_seqs
|    …
```

assemblyAln=output/assembly-aligned

BAM files with the reads aligned to the assembly and FASTA files with the consensus called from the BAM file.

```
assembly-aligned
|--- barcode1
|    |--- barcode1.bam
|    |--- barcode1_consensus.fasta
|--- barcode2
|    |--- barcode2.bam
|    |--- barcode2_consensus.fasta
|--- barcode3
|    |--- barcode3.bam
|    |--- barcode3_consensus.fasta
```

treeOutput=output/tree

IQ-TREE output is here and a FASTA file of the consensus sequences aligned with backgroundDB. The consensus treefile has the extension .contree and the best maximum-likelihood treefile has .treefile as the extension.

```
tree
|--- consensus_background_Aln.fasta
…
|--- consensus_background_Aln.fasta.contree
…
|--- consensus_background_Aln.fasta.treefile
```

5. References

Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., … others. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11), 1422–1423.

Danecek, P., Bonfield, J. K., Liddle, J., Marshall, J., Ohan, V., Pollard, M. O., Whitwham, A., Keane, T., McCarthy, S. A., Davies, R. M., & Li, H. (2021). Twelve years of SAMtools and BCFtools. *Gigascience*, 10(2). DOI: 10.1093/gigascience/giab008

Katoh, K., & Standley, D. M. (2013). MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Molecular biology and evolution*, 30(4), 772–780. DOI: 10.1093/molbev/mst010

Koren, S., Walenz, B. P., Berlin, K., Miller, J. R., Bergman, N. H., & Phillippy, A. M. (2017). Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome Research*, 27(5), 722–736. DOI: 10.1101/gr.215087.116

Kremer, L. P. M. (2019), MSA_trimmer, GitHub repository, Retrieved on July 11, 2024, from https://github.com/LKremer/MSA_trimmer

Li, H. (2018). Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18), 3094–3100. DOI: 10.1093/bioinformatics/bty191

Minh, B. Q., Schmidt, H. A., Chernomor, O., Schrempf, D., Woodhams, M. D., Von Haeseler, A., & Lanfear, R. (2020). IQ-TREE 2: New Models and Efficient Methods for Phylogenetic Inference in the Genomic Era. *Molecular Biology And Evolution*, 37(5), 1530–1534. DOI: 10.1093/molbev/msaa015

Rice, P., Longden, I., & Bleasby, A. (2000). EMBOSS: The European Molecular Biology Open Software Suite. *Trends in Genetics*, 16(6), 276–277. DOI: 10.1016/s0168-9525(00)02024-2

Wymant, C., Blanquart, F., Golubchik, T., Gall, A., Bakker, M., Bezemer, D., Croucher, N. J., Hall, M., Hillebregt, M., Ong, S. H., Ratmann, O., Albert, J., Bannert, N., Fellay, J., Fransen, K., Gourlay, A., Grabowski, M. K., Gunsenheimer-Bartmeyer, B., Günthard, H. F., . . . Fraser, C. (2018). Easy and accurate reconstruction of whole HIV genomes from short-read sequence data with shiver. *Virus Evolution*, 4(1). DOI: 10.1093/ve/vey007

Xu, M., Guo, L., Gu, S., Wang, O., Zhang, R., Peters, B. A., Fan, G., Liu, X., Xu, X., Deng, L., & Zhang, Y. (2020). TGS-GapCloser: A fast and accurate gap closer for large genomes with low coverage of error-prone long reads. *GigaScience*, 9(9). https://doi.org/10.1093/gigascience/giaa094