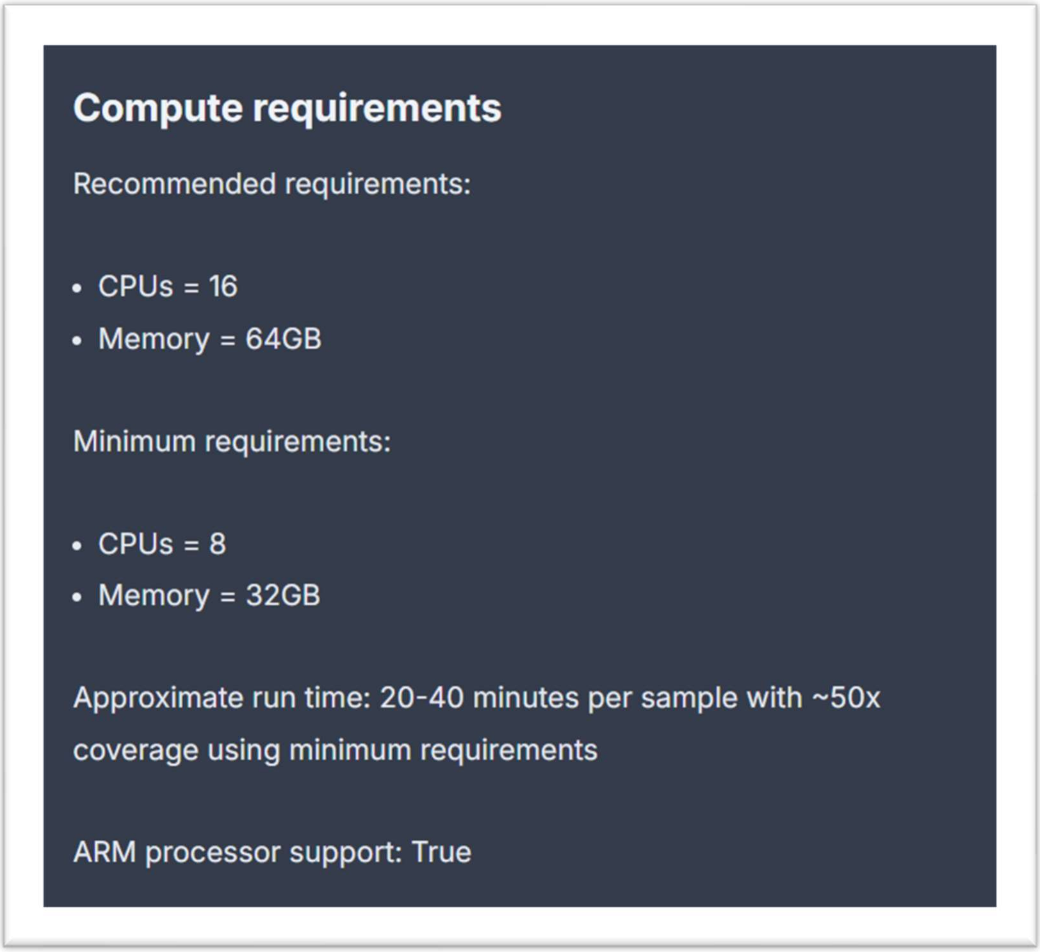
Install Nanopore EPI2ME Software on Windows computer

Followed by installation of

Bacterial assembly and annotation workflow

Version: March 2025

Minimal Requirements for Computer/Laptop to install and run Epi2ME:



Compute requirements

Recommended requirements:

- CPUs = 16
- Memory = 64GB

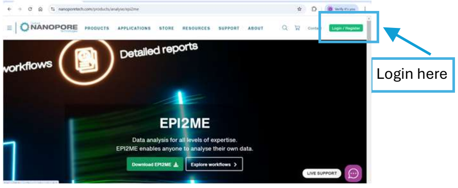
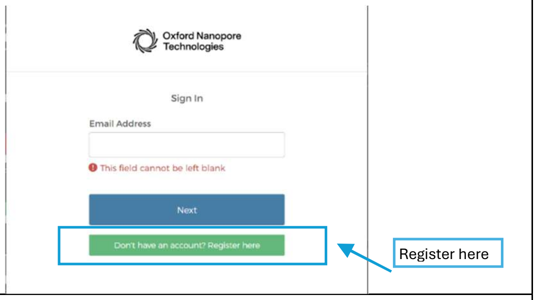
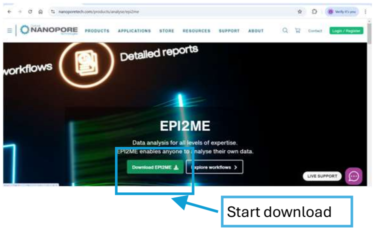
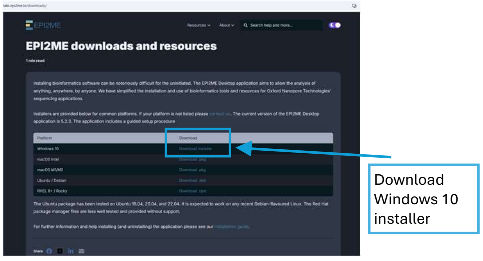
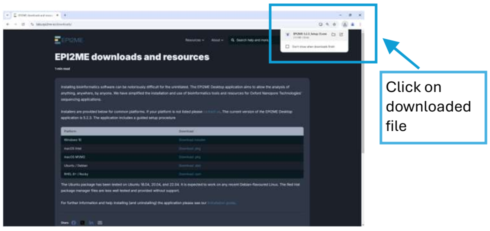
Minimum requirements:

- CPUs = 8
- Memory = 32GB

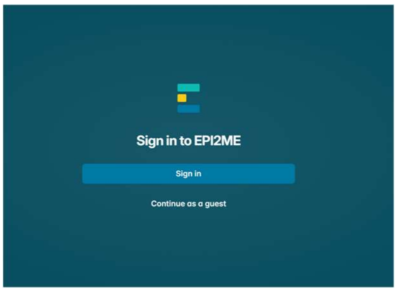
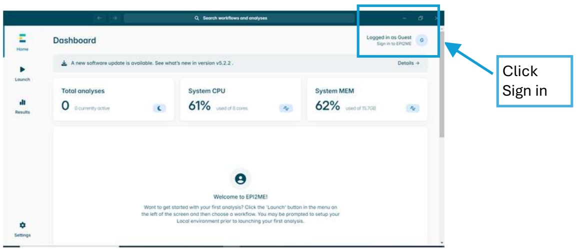
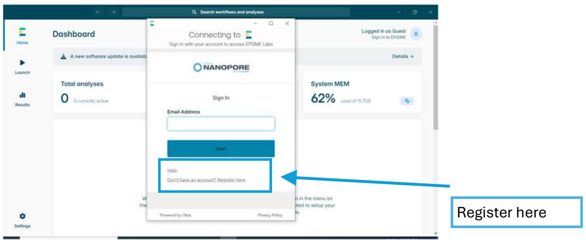
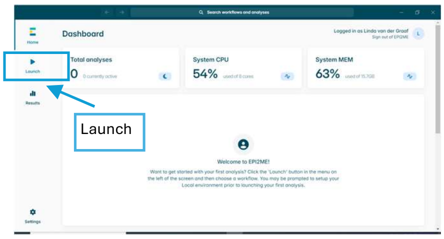
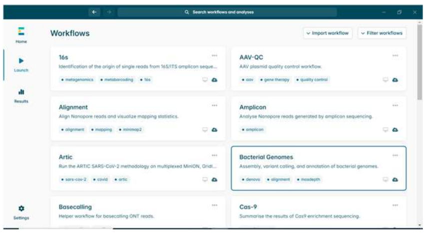
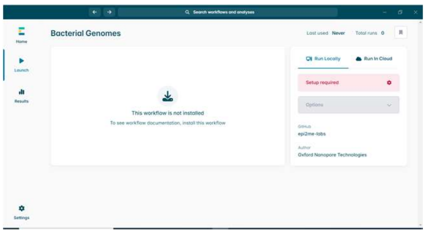
Approximate run time: 20-40 minutes per sample with ~50x coverage using minimum requirements

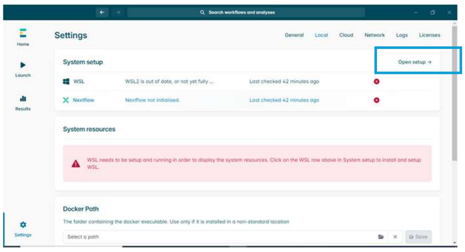
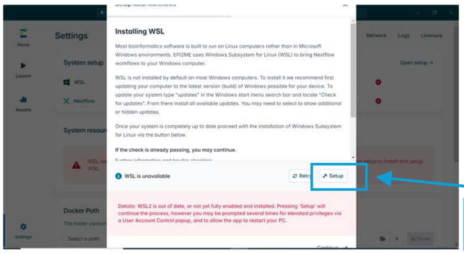
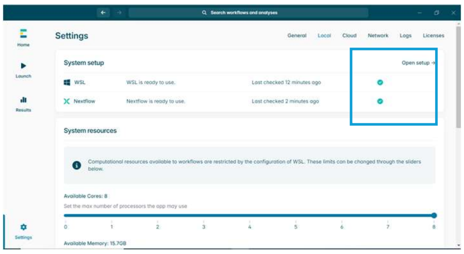
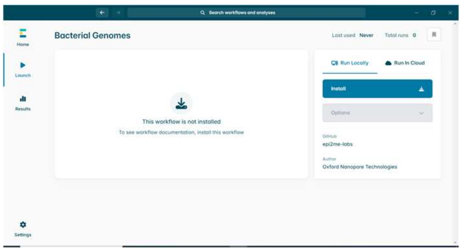
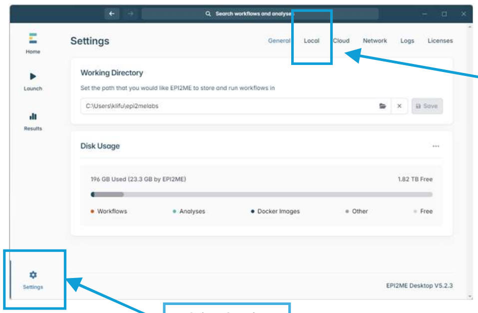
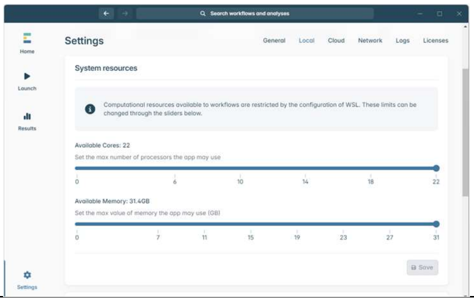
ARM processor support: True

Install Nanopore Epi2ME software on Windows computer

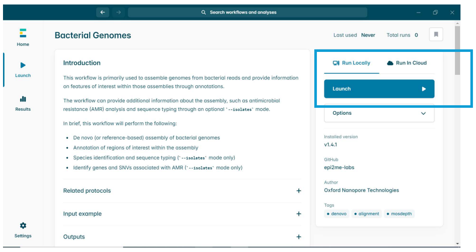
Navigate to Nanopore Epi2Me website: https://nanoporetech.com/products/analyse/epi2me Click on Login / Register	
Login with an existing account Or Create a new account by clicking “Don’t have an account? Register here” After registration, login with the new account	
Click on “Download EPI2ME”	
Click on “Windows 10” Platform – Download installer Additional: This page contains also the “installation guide” for help to (un)install the software	
Wait till download has finished and click on the downloaded file to start installation of EPI2ME	
After installation, the EPI2ME software is available and a desktop shortcut is created	
EPI2ME software can be started by clicking on Desktop Shotcut Icon	

Install Nanopore Epi2ME - Bacterial Assembly and Annotation Workflow

Open EPI2ME software and “Sign in” If you do not have an account, click “Continue as a guest”	
As guest: Click “Sign in to EPI2ME”	
As guest: Click “Don’t have an account? Register here” Register and login	
After login, click “Launch”	
Click on “Bacterial Genomes”	
Normally, setup is required before installing the workflow Click on “Setup required”	

Check if WSL and Nextflow are installed. If not (red cross), click “open setup”	 Settings System setup WSL WSL2 is out of date, or not yet fully ... Last checked 42 minutes ago Nextflow Nextflow not initialised. Last checked 42 minutes ago System resources WSL needs to be setup and running in order to display the system resources. Click on the WSL row above in System setup to install and setup WSL. Docker Path The folder containing the docker executable. Use only if it is installed in a non-standard location Select a path Open setup
To install WSL, click “Setup” to start installation of WSL and Nextflow	 Installing WSL Most bioinformatics software is built to run on Linux computers rather than in Microsoft Windows environments. EPI2ME uses Windows Subsystem for Linux (WSL) to bring Nextflow workflows to your Windows computer. WSL is not installed by default on most Windows computers. To install it we recommend first updating your computer to the latest version (build) of Windows possible for your device. To update your system type "updates" in the Windows start menu search bar and locate "Check for updates". From there install all available updates. You may need to select to show additional or hidden updates. Once your system is completely up to date proceed with the installation of Windows Subsystem for Linux via the button below. **If the check is already passing, you may continue.** WSL is unavailable Details: WSL2 is out of date, or not yet fully enabled and installed. Pressing 'Setup' will continue the process, however you may be prompted several times for elevated privileges via a User Account Control popup, and to allow the app to restart your PC. Click Setup
After installation of WSL and Nextflow, check if installation was successful – green checkmark will appear WSL and Nextflow are ready to use	 Settings System setup WSL WSL is ready to use. Last checked 12 minutes ago Nextflow Nextflow is ready to use. Last checked 2 minutes ago System resources Computational resources available to workflows are restricted by the configuration of WSL. These limits can be changed through the sliders below. Available Cores: 8 Set the max number of processors the app may use Available Memory: 15.7GB
Go back to “Launch” Select “Bacterial Genomes” Click “Install” (in screen Run Locally) Please note: Installation can take very long (up to hours)	 Bacterial Genomes Last used Never Total runs 0 This workflow is not installed To see workflow documentation, install this workflow GitHub epi2me-labs Author Oxford Nanopore Technologies
Set system resources to maximum 1. Click “Settings” 2. Click “Local”	 Settings Working Directory Set the path that you would like EPI2ME to store and run workflows in C:\Users\klifu\epi2melabs Disk Usage 196 GB Used (23.3 GB by EPI2ME) 1.82 TB Free EPI2ME Desktop V5.2.3 2. Click Local 1. Click Settings
Slide both “Available cores” and “Available memory” to maximum	 Settings System resources Computational resources available to workflows are restricted by the configuration of WSL. These limits can be changed through the sliders below. Available Cores: 22 Set the max number of processors the app may use Available Memory: 31.4GB Set the max value of memory the app may use (GB)

After successful installation of the Bacterial Genomes Workflow, the button “Launch” will appear,
to be used to analyse Nanopore sequencing data.



Bacterial Genomes

Introduction

This workflow is primarily used to assemble genomes from bacterial reads and provide information on features of interest within those assemblies through annotations.

The workflow can provide additional information about the assembly, such as antimicrobial resistance (AMR) analysis and sequence typing through an optional `--isolates` mode.

In brief, this workflow will perform the following:

- De novo (or reference-based) assembly of bacterial genomes
- Annotation of regions of interest within the assembly
- Species identification and sequence typing (`--isolates` mode only)
- Identify genes and SNVs associated with AMR (`--isolates` mode only)

Installed version
v1.4.1

GitHub
epi2me-labs

Author
Oxford Nanopore Technologies

Tags
denovo alignment mosdepth

ADVANCED EPI2ME installation options – change Workflow files

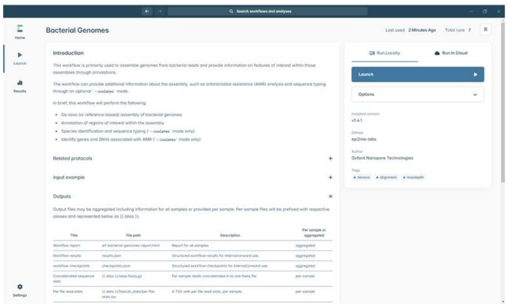
Please note: Described modifications can only be performed by advanced users who know what they are doing

1. Advanced installation; change workflow to "isolates mode". Default setting is that isolates mode of bacterial genomes workflow it turned off. This isolates mode will add analysis like species identification, MLST STs and Resfinder result.

Follow the following steps to turn isolates mode on.

The isolates mode of the Bacterial Genomes workflow will add species identification, MLST STs and AMR gene analysis.

Default setting is that this isolates mode is turned off



Introduction

This workflow is primarily used to assemble genomes from bacterial reads and provide information on features of interest within those assemblies through annotations.

The workflow can provide additional information about the assembly, such as antimicrobial resistance (AMR) analysis and sequence typing through an optional `--isolates` mode.

In brief, this workflow will perform the following:

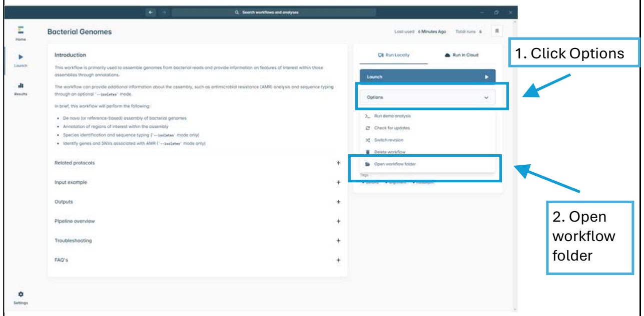
- De novo (or reference-based) assembly of bacterial genomes
- Annotation of regions of interest within the assembly
- Species identification and sequence typing (`--isolates` mode only)
- Identify genes and SNVs associated with AMR (`--isolates` mode only)

Output files may be aggregated including information for all samples or provided per sample. Per-sample files will be prefixed with respective aliases and represented below as {{ alias }}.

Title	File path	Description	Per sample or aggregated
Workflow report	wf-bacterial-genomes-report.html	Report for all samples	aggregated
Workflow results	results.json	Structured workflow results for internal/onward use.	aggregated
workflow checkpoints	checkpoints.json	Structured workflow checkpoints for internal/onward use.	aggregated
Concatenated sequence data	{{ alias }}/seqs.fastq.gz	Per sample reads concatenated in to one fastq file.	per-sample
Per file read stats	{{ alias }}/fastcat_stats/per-file-stats.tsv	A TSV with per file read stats, per sample.	per-sample

Navigate to Bacterial Genomes Workflow

1. Click "Options"
2. Click "Open workflow folder"



Introduction

This workflow is primarily used to assemble genomes from bacterial reads and provide information on features of interest within those assemblies through annotations.

The workflow can provide additional information about the assembly, such as antimicrobial resistance (AMR) analysis and sequence typing through an optional `--isolates` mode.

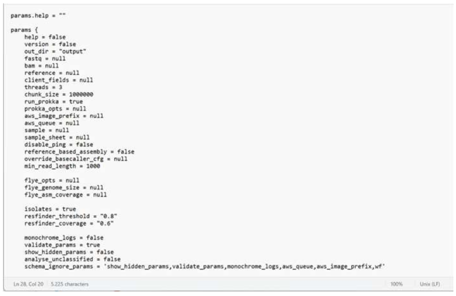
In brief, this workflow will perform the following:

- De novo (or reference-based) assembly of bacterial genomes
- Annotation of regions of interest within the assembly
- Species identification and sequence typing (`--isolates` mode only)
- Identify genes and SNVs associated with AMR (`--isolates` mode only)

Open file "nextflow" in text editor

Change:
"isolates = false" in "isolates = true"

Save file

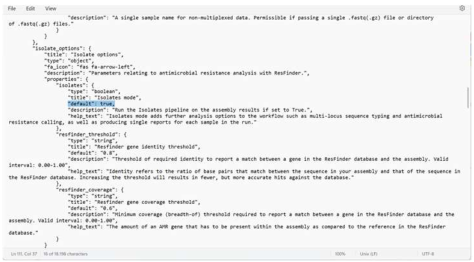


```
params.help = ""

params {
    help = false
    version = false
    out_dir = "output"
    fastq = null
    bam = null
    reference = null
    client_fields = null
    threads = 3
    chunk_size = 1000000
    run_prokka = true
    prokka_opts = null
    aws_image_prefix = null
    aws_queue = null
    sample = null
    sample_sheet = null
    disable_ping = false
    reference_based_assembly = false
    override_basecaller_cfg = null
    min_read_length = 1000

    flye_opts = null
    flye_genome_size = null
    flye_asm_coverage = null

    isolates = true
    resfinder_threshold = "0.8"
    resfinder_coverage = "0.6"

    monochrome_logs = false
    validate_params = true
    show_hidden_params = false
    analyse_unclassified = false
    schema_ignore_params = 'show_hidden_params,validate_params,monochrome_logs,aws_queue,aws_image_prefix,wf'
```

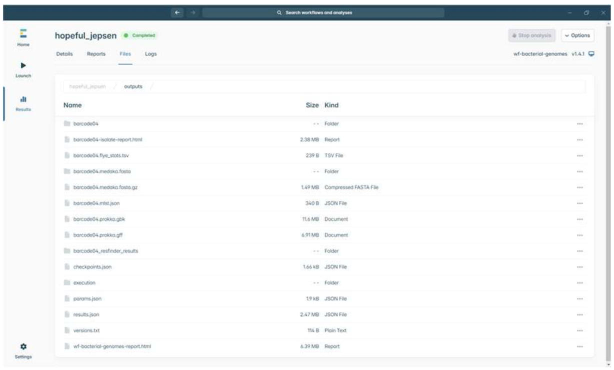
Open file "nextflow_scheme" in text editor

Change for isolates mode:
""default": false," in "default": true,"

Save file



```
                    "description": "A single sample name for non-multiplexed data. Permissible if passing a single .fastq(.gz) file or directory
of .fastq(.gz) files."
                }
            }
        },
        "isolate_options": {
            "title": "Isolate options",
            "type": "object",
            "fa_icon": "fas fa-arrow-left",
            "description": "Parameters relating to antimicrobial resistance analysis with ResFinder.",
            "properties": {
                "isolates": {
                    "type": "boolean",
                    "title": "Isolates mode",
                    "default": true,
                    "description": "Run the Isolates pipeline on the assembly results if set to True.",
                    "help_text": "Isolates mode adds further analysis options to the workflow such as multi-locus sequence typing and antimicrobial
resistance calling, as well as producing single reports for each sample in the run."
                },
                "resfinder_threshold": {
                    "type": "string",
                    "title": "Resfinder gene identity threshold",
                    "default": "0.8",
                    "description": "Threshold of required identity to report a match between a gene in the ResFinder database and the assembly. Valid
interval: 0.00-1.00",
                    "help_text": "Identity refers to the ratio of base pairs that match between the sequence in your assembly and that of the sequence in
the ResFinder database. Increasing the threshold will results in fewer, but more accurate hits against the database."
                },
                "resfinder_coverage": {
                    "type": "string",
                    "title": "Resfinder gene coverage threshold",
                    "default": "0.6",
                    "description": "Minimum coverage (breadth-of) threshold required to report a match between a gene in the ResFinder database and the
assembly. Valid interval: 0.00-1.00",
                    "help_text": "The amount of an AMR gene that has to be present within the assembly as compared to the reference in the ResFinder
database."
                }
            }
```

<table>
<tr><td></td><td></td></tr>
<tr><td>Launch Bacterial Genomes workflow again and check if mlst and resfinder files are generated</td><td>


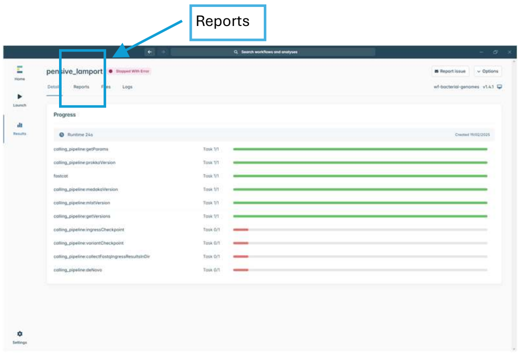
hopeful_jepsen ● Completed

wf-bacterial-genomes v1.4.1

hopeful_jepsen / outputs /

<table>
<tr><th>Name</th><th>Size</th><th>Kind</th></tr>
<tr><td>barcode04</td><td>- -</td><td>Folder</td></tr>
<tr><td>barcode04-isolate-report.html</td><td>2.38 MB</td><td>Report</td></tr>
<tr><td>barcode04.flye_stats.tsv</td><td>239 B</td><td>TSV File</td></tr>
<tr><td>barcode04.medaka.fasta</td><td>- -</td><td>Folder</td></tr>
<tr><td>barcode04.medaka.fasta.gz</td><td>1.49 MB</td><td>Compressed FASTA File</td></tr>
<tr><td>barcode04.mlst.json</td><td>340 B</td><td>JSON File</td></tr>
<tr><td>barcode04.prokka.gbk</td><td>11.6 MB</td><td>Document</td></tr>
<tr><td>barcode04.prokka.gff</td><td>6.91 MB</td><td>Document</td></tr>
<tr><td>barcode04_resfinder_results</td><td>- -</td><td>Folder</td></tr>
<tr><td>checkpoints.json</td><td>1.66 kB</td><td>JSON File</td></tr>
<tr><td>execution</td><td>- -</td><td>Folder</td></tr>
<tr><td>params.json</td><td>1.9 kB</td><td>JSON File</td></tr>
<tr><td>results.json</td><td>2.47 MB</td><td>JSON File</td></tr>
<tr><td>versions.txt</td><td>114 B</td><td>Plain Text</td></tr>
<tr><td>wf-bacterial-genomes-report.html</td><td>6.39 MB</td><td>Report</td></tr>
</table>
</td></tr>
</table>

2. *Troubleshooting; Workflow stops with notification that not enough memory is available for workflow, while computer/laptop has the prescribed minimal 32gb RAM*

<table>
<tr><td>If workflow "Stopped with error", go to "Reports"</td><td>


pensive_lamport ● Stopped With Error

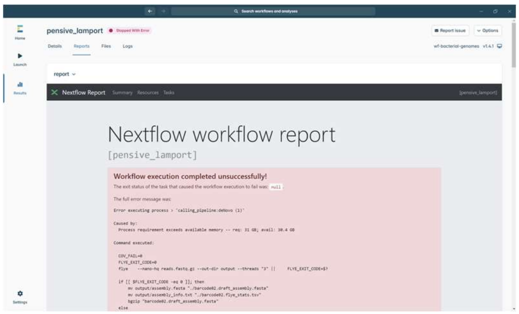
wf-bacterial-genomes v1.4.1

Progress

Runtime 24s Created 11/02/2025

<table>
<tr><th></th><th></th></tr>
<tr><td>calling_pipeline:getParams</td><td>Task 1/1</td></tr>
<tr><td>calling_pipeline:prokkaVersion</td><td>Task 1/1</td></tr>
<tr><td>fastcat</td><td>Task 1/1</td></tr>
<tr><td>calling_pipeline:medakaVersion</td><td>Task 1/1</td></tr>
<tr><td>calling_pipeline:mlstVersion</td><td>Task 1/1</td></tr>
<tr><td>calling_pipeline:getVersions</td><td>Task 1/1</td></tr>
<tr><td>calling_pipeline:ingressCheckpoint</td><td>Task 0/1</td></tr>
<tr><td>calling_pipeline:variantCheckpoint</td><td>Task 0/1</td></tr>
<tr><td>calling_pipeline:collectFastqIngressResultsInDir</td><td>Task 0/1</td></tr>
<tr><td>calling_pipeline:deNovo</td><td>Task 0/1</td></tr>
</table>
</td></tr>
<tr><td>Check error.
If de novo assembly has stopped and "Caused by: Process requirement exceeds available memory"…

…continue with following steps

If other error is reported, do not continue with following steps</td><td>


pensive_lamport ● Stopped With Error

wf-bacterial-genomes v1.4.1

Nextflow workflow report

[pensive_lamport]

Workflow execution completed unsuccessfully!

The exit status of the task that caused the workflow execution to fail was: `null`.

The full error message was:

<pre>
Error executing process > 'calling_pipeline:deNovo (1)'

Caused by:
  Process requirement exceeds available memory -- req: 31 GB; avail: 30.4 GB

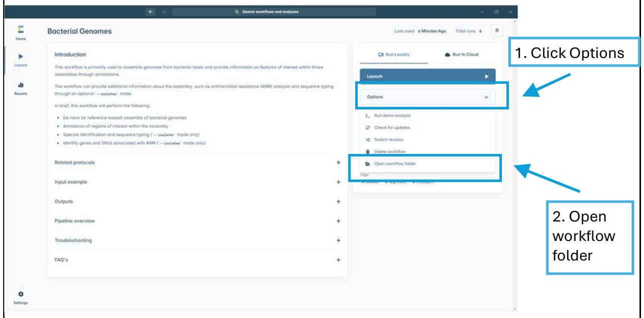
Command executed:

  COV_FAIL=0
  FLYE_EXIT_CODE=0
  flye   --nano-hq reads.fastq.gz --out-dir output --threads "3" || FLYE_EXIT_CODE=$?

  if [[$FLYE_EXIT_CODE -eq 0]]; then
      mv output/assembly.fasta "./barcode02.draft_assembly.fasta"
      mv output/assembly_info.txt "./barcode02.flye_stats.tsv"
      bgzip "barcode02.draft_assembly.fasta"
  else
</pre>
</td></tr>
</table>

Navigate back to Bacterial Genomes Workflow

3. Click “Options”
4. Click “Open workflow folder”



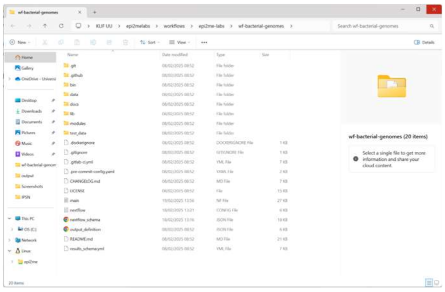
This workflow is primarily used to assemble genomes from bacterial reads and provide information on features of interest within those assemblies through annotations.

The workflow can provide additional information about the assembly, such as antimicrobial resistance (AMR) analysis and sequence typing through an optional `--isolates` mode.

In brief, this workflow will perform the following:

- De novo (or reference-based) assembly of bacterial genomes
- Annotation of regions of interest within the assembly
- Species identification and sequence typing (`--isolates` mode only)
- Identify genes and SNVs associated with AMR (`--isolates` mode only)

Open file “main” in text editor



Scroll down to “process deNovo” and change memory “31 GB” to “24 GB”

Save file

```
}

process deNovo {
    label "wfbacterialgenomes"
    cpus params.threads
    memory "31 GB"
    input:
        tuple val(meta), path("reads.fastq.gz")
    output:
        tuple val(meta),
            path("${meta.alias}.draft_assembly.fasta.gz"),
            path("${meta.alias}.flye_stats.tsv"),
            optional: true, emit: asm
        tuple val(meta), env(COV_FAIL), emit: failed
    script:
    // flye may fail due to low coverage; in this case we don't want to cause the whole
    // workflow to crash --> exit with `0` and don't emit output files
    def flye_opts = params.flye_opts ?: ""
    def genome_size = params.flye_genome_size ? "--genome-size " + params.flye_genome_size : ""
    def asm_coverage = params.flye_asm_coverage ? "--asm-coverage " + params.flye_asm_coverage : ""
    """
    COV_FAIL=0
    FLYE_EXIT_CODE=0
    flye $flye_opts $genome_size $asm_coverage --nano-hq reads.fastq.gz --out-dir output --threads "${task.cpus}" || \
    FLYE_EXIT_CODE=\$?

    if [[ \$FLYE_EXIT_CODE -eq 0 ]]; then
        mv output/assembly.fasta "./${meta.alias}.draft_assembly.fasta"
        mv output/assembly_info.txt "./${meta.alias}.flye_stats.tsv"
        bgzip "${meta.alias}.draft_assembly.fasta"
    else
        # flye failed --> check the log to check why
        edge_cov=\$(
            grep -oP 'Mean edge coverage: \\K\\d+' output/flye.log \
            || echo $FLYE_MIN_COVERAGE_THRESHOLD
        )
        ovlp_cov=\$(
            grep -oP 'Overlap-based coverage: \\K\\d+' output/flye.log \
            || echo $FLYE_MIN_COVERAGE_THRESHOLD
        )
        if [[
            \$edge_cov -lt $FLYE_MIN_COVERAGE_THRESHOLD ||
            \$ovlp_cov -lt $FLYE_MIN_COVERAGE_THRESHOLD
        ]]; then
            echo -n "Caught Flye failure due to low coverage (either mean edge cov. or "
            echo "overlap-based cov. were below $FLYE_MIN_COVERAGE_THRESHOLD)".
            COV_FAIL=1
        elif grep -q "No disjointigs were assembled" output/flye.log; then
            echo -n "Caught Flye failure due to disjointig assembly."
            COV_FAIL=2
        else
            # exit a subshell with error so that the process fails
            ( exit \$FLYE_EXIT_CODE )
        fi
```

Open file “nextflow” in text editor

Scroll down to awsbatch process and change
“memory = 31 G” to “memory = 24 G”

Save file

```
    conda {
        conda.enabled = true
    }

    awsbatch {
        process {
            executor = 'awsbatch'
            queue = "${params.aws_queue}"
            memory = '31G'
            withLabel:wf_common {
                container = "${params.aws_image_prefix}-wf-common:${params.wf.common_sha}"
            }
            withLabel:wfbacterialgenomes {
                container = "${params.aws_image_prefix}-wf-bacterial-genomes:${params.wf.container_sha}"
            }
            withLabel:prokka {
                container = "${params.aws_image_prefix}-prokka:${params.wf.container_sha_prokka}-root"
            }
            withLabel:medaka {
                container = "${params.aws_image_prefix}-medaka:${params.wf.container_sha_medaka}"
            }
            withLabel:amr {
                container = "${params.aws_image_prefix}-resfinder:${params.wf.container_sha_resfinder}"
            }
            withLabel:mlst {
                container = "${params.aws_image_prefix}-mlst:${params.wf.container_sha_mlst}"
            }
            withLabel:seqsero2 {
                container = "${params.aws_image_prefix}-seqsero2:${params.wf.container_sha_seqsero}"
            }
            shell = ['/bin/bash', '-euo', 'pipefail']
        }
    }
}
```

Launch Bacterial Genomes workflow again and check if it runs without errors till “Completed”