BACALYZE - BACTERIAL GENOME ANALYZER SOFTWARE SUITE

A COMPREHENSIVE USER MANUAL

© Anantha Venkata Subramanyam G

Bioinformatics Centre • SASTRA Deemed to be University

Thanjavur, TN 613401

Phone +919499012584 Email ananthavsg@outlook.in

Table of Contents

1 Introduction

Welcome to Bacalyze - **Bac**terial Genome Ana**lyze**r Software Suite user manual. This guide is designed to get the most out of Bacalyze by providing detailed instructions and tips on how to use it effectively.

Bacalyze is a powerful tool designed to analyse bacterial genomic nucleotide sequences. The tool integrates several opensource tools for automated end-to-end processing. The tool can find Single Nucleotide Polymorphisms, k-mer frequencies, Mobile Genetic Elements (MGEs), Antibiotic Resistance Genes (ARGs), co-occurrence of MGEs, association between ARGs & MGEs, *de novo* assembly of raw reads, perform *de novo* annotation of bacterial genomes, predict antibiotic resistance phenotypes and provide insightful visualisations of the analyses[1]. It helps users to perform all the above bioinformatics analysis of bacterial genomes effectively. This tool can parallelly process and analyse multiple genomic samples at once simultaneously[2].

This tool comes with a command line interface as well as a graphical user interface. The graphical user interface simply runs the commands in the shell as an interpreter instead of the user needing to type in the commands. The tool relies on NextFlow pipeline which automates the processes. The tool can be scaled up to any level thanks to the scalability of NextFlow. More information on the tool, including its requirements and instructions on installation is provided further in detail in this manual.

[1] Not all analyses provide visualisations.

[2] Number of genomes that can be parallelly processed varies depending upon hardware.

2 System Requirements

2.1 Microsoft Windows

Windows 10: 64-bit, version 22H2 (build 19045) or higher.

Windows 11: 64-bit, version 22H2 or higher.

Windows Subsystem for Linux (WSL) 2 enabled.

Hardware virtualization enabled in BIOS.

WSL 1.1.3.0 or later.

12 GB RAM.

At least 10 GB of free space.

2.2 Apple Mac

Supported version of macOS (Intel Mac or Silicon Mac).

16 GB RAM (Intel Mac).

12 GB RAM (Silicon Mac)

At least 10 GB of free space.

2.3 Linux

2.3.1 To run natively[3]:

64-bit Linux operating system.

8 GB RAM.

At least 10 GB of free space.

2.3.2 To run in a docker virtual environment:

64-bit Linux operating system.

12 GB RAM.

At least 10 GB of free space.

[3] Note: All the dependencies must be manually installed for the user to run natively

3 Installation

3.1 Microsoft Windows

To install in Windows, please follow the instructions given below.

- Extract the archive file given in GitHub. Navigate to the extracted folder and run install.bat as administrator. Follow the on-screen instructions.
- Please refer to the YouTube installation manual bit.ly/HowToInstallBacalyze.

3.1 Apple Mac

To install in Mac, you must configure manually. Step by step instructions are given below.

- Download the source code archive file from GitHub repository.
- Install docker desktop for Mac from https://docs.docker.com/desktop/setup/install/mac-install/
- Extract the contents of the source code archive into `~/Documents/Bacalyze/`. Create the folder if necessary.
- Make sure that the directory structure is `~/Documents/Bacalyze/DockerFile/`.
- Open a terminal window inside the `DockerFile` folder.
- Run `docker build -t kani .`.

3.2 Linux

3.2.1 Native:

To run natively in Linux, you must install all dependencies of the pipeline, including NextFlow. Please follow the instructions specified in https://www.nextflow.io/docs/latest/install.html. Also please see Appendix A for the list of all required dependencies and install them on your own. After installing the dependencies, download and extract the source code archive contents to `~/Documents/Bacalyze/`.

3.2.2 Docker:

To use docker version of the tool, you will have to install docker for Linux. Please refer their documentation to install docker in your respective distros. Step by step instruction is given below.

- Download and extract the source code archive contents to `~/Documents/Bacalyze/`.
- Open a terminal window in `~/Documents/Bacalyze/DockerFile/`.
- Run `docker build -t kani .`. Use `sudo` if needed.

4 Getting Started

4.1 Command Line Interface

The command line interface is provided through NextFlow pipeline and can be called through docker. Host file system must be mounted through docker. The command is:

```
docker run -v /host/file/system/:/mountname/ kani
nextflow run
/mountname/document_folder/Bacalyze/main/script/main.nf
```

Note that you can copy main.nf file to any location in your file system. Be sure to mount the directory for docker to access the script. The script is found in `Documents/Bacalyze/main/script/main.nf`. You can also mount multiple folders at multiple mountpoints.

For example:

```
docker run -v /script/:/pipeline/ /inputdir:/in/
/outputdir:/out/ nextflow run /pipeline/main.nf
```

.

Note that if you are running natively in Linux, you can simply run `nextflow run /path/to/main.nf`

4.1.1 Parameters and Values

You must invoke the pipeline with parameters that specify the mode of execution, the paths to input files, and the path to output directory.

a. --mode
   This parameter specifies the mode of the execution. There are multiple modes designed for multiple workflows. These modes with their functions are described below:
   i. snp
      This mode performs variant calling on the input data and predicts AMR phenotype based on SNPs.
   ii. kmer
      This mode returns kmer frequencies of the input data in a csv format.
   iii. assemble
      This mode performs *de novo* genome assembly of *single* or paired end reads using SPAdes genome assembler after filtering using fastp.
   iv. genome_annotation
      This mode performs *de novo* genome annotation on the input data. This mode will assemble the raw reads into contigs before annotation using prokka.

v. mge

This mode indentifies MGEs from the input data and returns csv file of MGEs identified per sample using MobileElement Finder.

vi. arg

This mode identifies ARGs from the input data and returns tab separated files of ARGs identified and phenotype per sample using ResFinder.

vii. snp-no-predict

This mode performs variant calling alone and does not predict AMR phenotype

viii. mge_arg or arg_mge

This mode identifies both ARGs and MGEs, identifies co-occurring MGEs, identifies ARGs associated with MGEs, and provides consolidated, insightful visualisations with csv files.

ix. super

This mode processes all the workflows.

b. --assembled

This parameter specifies whether the input genome sequences are already assembled or not.

Values:

no (Default)
yes

c. --read1

This parameter is used to specify the path of single end reads or forward of paired end reads or assembled contigs. Full path must be specified.

Single sample usage:

Single end reads

--read1 /path/to/sample.fastq (Also accepts fastqsanger.gz, fq.gz, fastq.gz, fq)

Paired end (forward):

--read1 /path/to/sample_1.fastq

Contigs

--read1 /path/to/sample.fasta (Also accepts fa, fna)

Multiple samples usage:

<u>Single end reads</u>
--read1 “$(ls /path/to/samples/*fastq)”
<u>Contigs</u>
--read1 “$(ls /path/to/samples/*fasta)”

d. --read2
This parameter is used to specify reverse reads in a paired end sample. It is used only for single sample usage. Full path must be specified.
Usage:
--read2 /path/to/sample_2.fastq

e. --reads
This parameter is only used to specify multiple samples of paired end reads. Full path must be specified.
Usage:
--reads “/path/to/samples/*{1,2}.fastq”.
Note that this example assumes that all samples’ filenames are formatted as sample_1.fastq, sample_2.fastq. But you can use anything (Eg. R, F) depending on your filename format. But make sure that all samples follow the same pattern of forward and reverse denotation.

f. --qual
This parameter is used to specify the minimum quality to trim using fastp. By default, it is 15.
Usage:
--qual 25

g. --annotation and --gff
This parameter is used only in snp and snp-no-predict modes. It specifies whether the variants should be annotated or not. Default value is yes. This additionally requires a bed file for annotation. This uses bcftools for annotation.
Usage:
--annotation yes --bed /path/to/bed_file.bed

h. --ref
This parameter is used to specify the path to reference genome. It is used only with snp and snp-no-predict modes.
Usage:
--ref /path/to/reference.fasta (Also accepts fa, fna)

i. --pca
This parameter is used to specify the path to pickle file of pca generaliser.
Usage:
--pca /path/to/pca.pkl

j. --model
This parameter is used to specify the path to pickle file of ML/DL model. More details on model are given in Appendix B.
Usage:
--model /path/to/model.pkl

k. --label
This parameter is used to specify the path to a text file which contains names of predicting antibiotics, one antibiotic per line.
Usage:
--label /path/to/labels.txt

l. --k
This parameter is used to specify the value of ‘k’ while running in kmer mode.
Usage:
--kmer 5

m. --species
This parameter is used to specify the name of the species with respect to resfinder. This is used along with ARG, or ARG_MGE workflow.

Example command:

Assume that you are going to find AMR traits using SNPs without annotating, in command line with paired ended fastq reads, by mounting custom directories to docker. Your command will look like:

```
docker run -v /script_path/:/script/ /seq_path/:/input/ /dependent_files/:/files/
/out_path/:/out/ kani nextflow run /script/main.nf –mode snp --reads “/input/*{R,F}.fastq” --
ref /files/ref.fasta --model /files/model.pkl --pca /files/pca.pkl --labels /files/labels.txt -
-outdir /out/
```

4.1.2 Parameter Chart

A table of parameters associated with modes is given below along with their default values (if any). All the parameters are mandatory except those have default values.

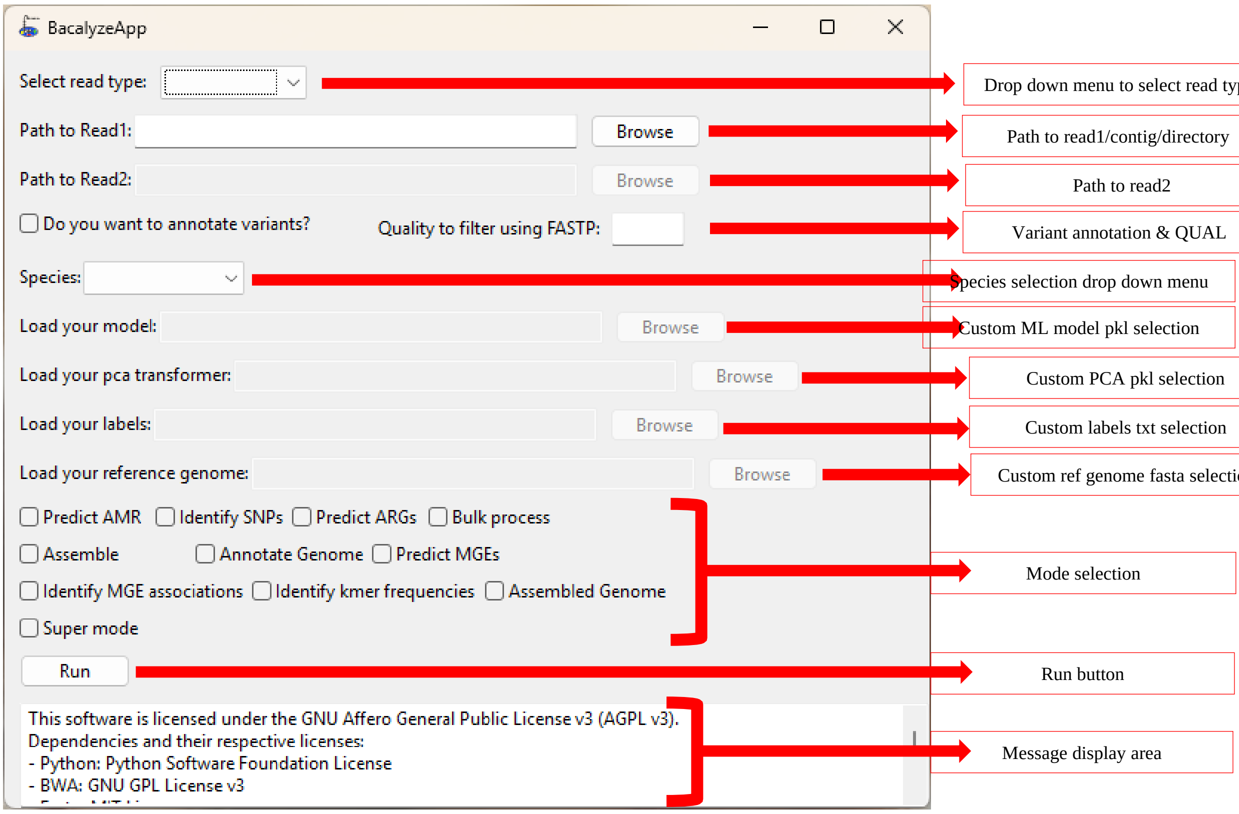
Sl. No	Mode	Parameter	Value type	Default value
1	snp	ref	Path to DNA fasta file	-
		reads	Path to multiple paired end DNA fastq files	-
		read1	Path to forward read.	-
			Path to assembled contigs	-
		read2	Path to Reverse read	-
		assembled	String (yes / no)	no
		model	Path to ML/DL model pkl file	-
		pca	Path to PCA pkl file	-
		labels	Path to labels text file	-
2	snp-no-predict	**All parameters of snp mode except model, pca and labels**		
3	kmer	reads	Path to multiple paired end DNA fastq files	-
		read1	Path to forward read.	-
			Path to assembled contigs	-
		read2	Path to Reverse read	-
		assembled	String (yes / no)	no
		k	Integer value of ‘k’	5
4	assemble	reads	Path to multiple paired end DNA fastq files	-
		read1	Path to forward read.	-
		read2	Path to Reverse read	-
5	genome_annotation	reads	Path to multiple paired end DNA fastq files	-
		read1	Path to forward read.	-
			Path to assembled contigs	-
		read2	Path to Reverse read	-
6	mge	reads	Path to multiple paired end DNA fastq files	-
		read1	Path to forward read.	-
			Path to assembled contigs	-
		read2	Path to Reverse read	-
7	arg	reads	Path to multiple paired end DNA fastq files	-
		read1	Path to forward read.	-
			Path to assembled contigs	-
		read2	Path to Reverse read	-
		species	A string value specifying supported species name. **Please check with resfinder CLI’s github page for the list of supported species**	“other”
8	mge_arg or arg_mge	reads	Path to multiple paired end DNA fastq files	-
		read1	Path to forward read.	-
			Path to assembled contigs	-
		read2	Path to Reverse read	-
9	super	**All the above-mentioned parameters**		

Table 1: All modes, their required parameters and their default & supported values

4.2 Graphical User Interface

Bacalyze comes with a graphical user interface (GUI). The GUI is made using wxPython. The script is then compiled into Ubuntu binaries and Windows executables. The whole tool is packaged as self-installing archive which can be installed in both OSes. For Mac, the manual installation option said earlier in section 3.2 applies.

4.2.1 Understanding the interface


BacalyzeApp
Select read type:
Path to Read1:
Browse
Path to Read2:
Browse
□ Do you want to annotate variants?
Quality to filter using FASTP:
Species:
Load your model:
Browse
Load your pca transformer:
Browse
Load your labels:
Browse
Load your reference genome:
Browse
□ Predict AMR
□ Identify SNPs
□ Predict ARGs
□ Bulk process
□ Assemble
□ Annotate Genome
□ Predict MGEs
□ Identify MGE associations
□ Identify kmer frequencies
□ Assembled Genome
□ Super mode
Run
Drop down menu to select read ty
Path to read1/contig/directory
Path to read2
Variant annotation & QUAL
Species selection drop down menu
Custom ML model pkl selection
Custom PCA pkl selection
Custom labels txt selection
Custom ref genome fasta selecti
Mode selection
Run button
Message display area

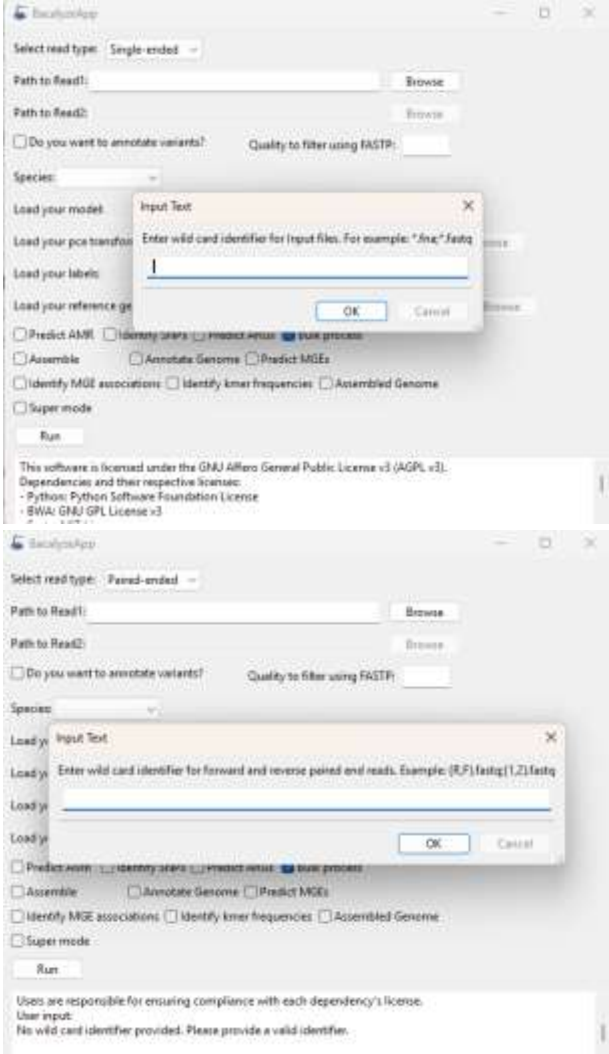
```
This software is licensed under the GNU Affero General Public License v3 (AGPL v3).
Dependencies and their respective licenses:
- Python: Python Software Foundation License
- BWA: GNU GPL License v3
```

Figure 1: Screenshot of the user interface

As you can see in the GUI screenshot, read type, with input fields for input files are provided. Single end reads, forward reads of paired end reads and assembled contigs must be given as input in “Read 1” field. Reverse reads of paired end reads must be given as input in "Read 2” field. Note that the above-mentioned case only applies when you want to analyse one sample. You must check “Assembled Genome” to use assembled contigs as inputs.

Warning: Currently there is no mechanism programmed to automatically detect the input format. So please choose correct options to ensure proper running, else there will be errors in running the pipeline or there will be broken outputs.

For multiple inputs, please select read type if the inputs are raw reads and then check “Bulk process”. A pop-up box will appear for wildcard input format. Please type in the wildcard input format and click “Ok”. If there are multiple contig files as inputs, please check “Assembled Genome” option before checking “Bulk process”. Below screenshots show the pop-up box asking for wildcard input format for paired end and single end / assembled contig inputs.



Select read type: Single-ended

Input Text

Enter wild card identifier for input files. For example: *.fna;*.fastq

This software is licensed under the GNU Affero General Public License v3 (AGPL v3).
Dependencies and their respective licenses:
- Python: Python Software Foundation License
- BWA: GNU GPL License v3

Select read type: Paired-ended

Input Text

Enter wild card identifier for forward and reverse paired end reads. Example: {R,F}.fastq;{1,2}.fastq

Users are responsible for ensuring compliance with each dependency's license.
User input:
No wild card identifier provided. Please provide a valid identifier.

Figure 2: Pop-up boxes asking for wildcard input formats

Warning: Currently there is no mechanism programmed to automatically detect the input format and sort the input files accordingly. Please do not mix the input file types. You can only use multiple paired ended reads, multiple single ended reads or multiple assembled contigs. You cannot mix the inputs. We are actively working on a feature to automatically sort out the input files. Please refer to the GitHub page for updates.

4.2.2 Selecting mode

From the screenshot provided in figure 1, you can use check boxes to select which mode to run. Subject to resource availability, some modes that involve genome assembly may fail to run.

Note: While you can select multiple checkboxes, only one mode will be run. The modes are being invoked using if-else conditions. We are actively working on a feature to streamline multiple mode execution and prevent redundant processes. As of now, “Super” can be used to execute all modes without redundancy. Please refer to GitHub page for updates.

4.2.3 Output directory

Bacalyze will create a directory in DDMMYYYYHHMMSS format of time in User’s Document Folder/Bacalyze/main/output/.

The output files of analysis can be accessed from that directory.

APPENDIX A

Sl No.	Dependencies	Command to install (Ubuntu)
1	Python version 3.11 and above	Must manually install from source for Ubuntu LTS version < 20.04. Available by default for Ubuntu LTS version 20.04 and above
2	BWA	apt install bwa
3	Fastp	apt install fastp
4	SAMtools	apt install samtools
5	VCFtools	apt install vcftools
6	Curl	apt install curl
7	OpenJDK 17 +	apt install openjdk-17-jdk-headless
8	Git	apt install git
9	NextFlow	Refer NextFlow documentation here.
10	Pandas	pip3 install pandas
11	Scikit-learn	pip3 install scikit-learn
12	Mlxtend	pip3 install mlxtend
13	Biopython	pip3 install biopython
14	Resfinder	pip3 install resfinder
15	MobileElementFinder	pip3 install MobileElementFinder
16	Ncbi-blast++	apt install ncbi-blast++
17	Seaborn	pip3 install seaborn
18	SPAdes	apt install spades
19	BCFtools	apt install bcftools
20	wxPython	apt install python-wxtools && pip3 install wxPython[4]

Note: Use pip or pip3 depending upon the binary name in your system. Use –break-system-packages option to install in system environment, or else, create a new virtual environment of our choice, and install all dependencies within that virtual environment. Please refer appropriate documentation for managing virtual environments.

[4] Note: Installing wxPython in linux system is tedious, please refer to wxPython's documentation for more details.

APPENDIX B

The dataset should contain SNP positions as columns, with 0,1,2,3 and 4 as label encoding for A, T, C, G and N. First column could be genome ID or sample ID. Last column should be antibiotic with label encoded classification “1” for resistance, “0” for susceptible. There can also be multilabel dataset columns, with multiple columns representing multiple antibiotics with label encoded classification each. But make sure the model algorithm you are planning to train with supports multilabel classification (Example: Random Forest Classification).

Once the dataset is ready with the above format, use PCA to reduce dimensions and save the PCA object in pkl format. Now train your ML model using the dataset with reduced dimensions and save the ML model object as pkl file. Also save a separate text file using a text editor of your choice with antibiotics in the same order of the labels in dataset, one antibiotic per line (even if there is only one antibiotic). Make sure you have the reference genome through which you obtained the SNPs.

You should now specify paths to these inputs for using in prediction in the tool.