PBGL CNV-seq Analysis v1.0

A Laboratory Manual

Anibal E. Morales-Zambrana

Plant Breeding and Genetics Laboratory
FAO/IAEA Joint Division
Seibersdorf, Austria

Created: March, 2021
Last updated: 04 June 2021

Contents

1 Background **2**

2 Installations - Virtual Environments and Software Packages **3**
- 2.1 Miniconda3 (conda) and Mamba . . . 3
- 2.2 Git Installation and Repo Cloning . . . 4
- 2.3 Required Libraries with Mamba . . . 4
  - 2.3.1 Automatically (faster) . . . 5
  - 2.3.2 Manually (slower) . . . 5

3 Running Jupyter **6**
- 3.1 Editing the Configuration File . . . 6
- 3.2 Running a RCNV_seq-template Jupyter Notebook . . . 9
  - 3.2.1 1 - Installing Required Libraries (optional) . . . 9
  - 3.2.2 2 - Loading Required Libraries (mandatory) . . . 9
  - 3.2.3 3 - User Input (mandatory) . . . 10
  - 3.2.4 4 - CNV Calculations . . . 10
  - 3.2.5 5 - Plotting . . . 10
  - 3.2.6 6 - Plotting a Zoomed-In Region of One Chromosome . . . 11

4 References **12**

1 Background

[WORKING DRAFT]

Copy number variation (CNV) analysis using CNVseq, R, Jupyter Notebooks, Miniconda3, Mamba, and Git.

---

2 Installations - Virtual Environments and Software Packages

Before installing any necessary software, it is recommended to check if the computer is running 32-bit or 64-bit for downloading Miniconda3. Run the following to verify the system:

```
$ uname -m
```

2.1 Miniconda3 (conda) and Mamba

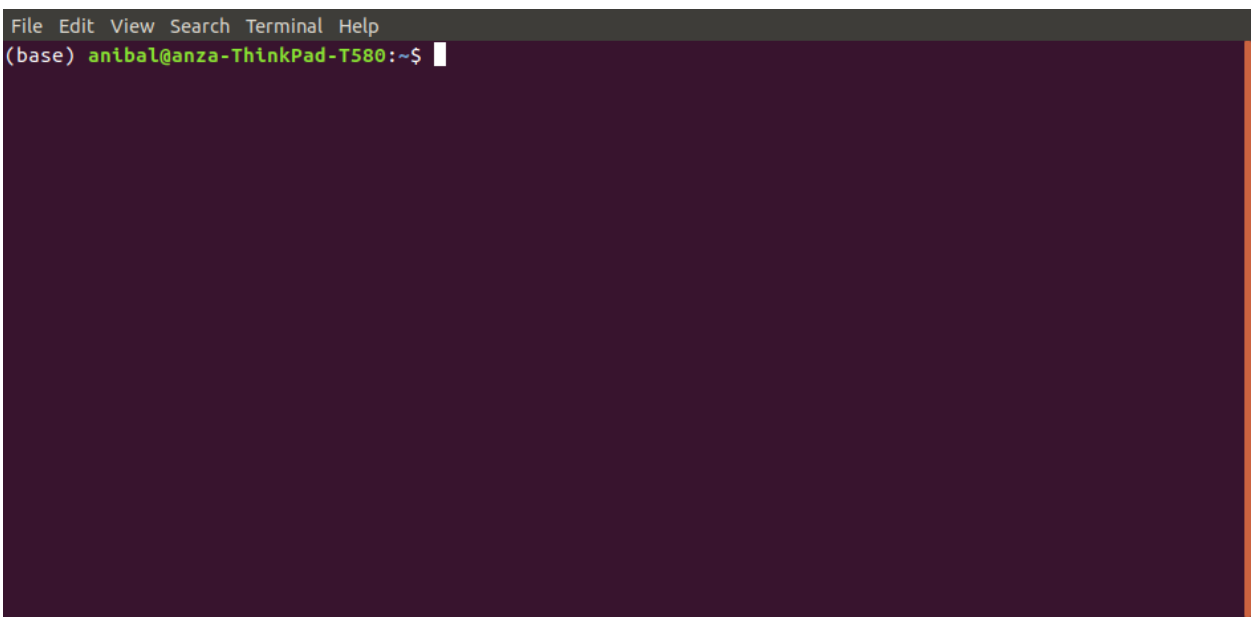
Download the Miniconda3, or simply "conda", installer:

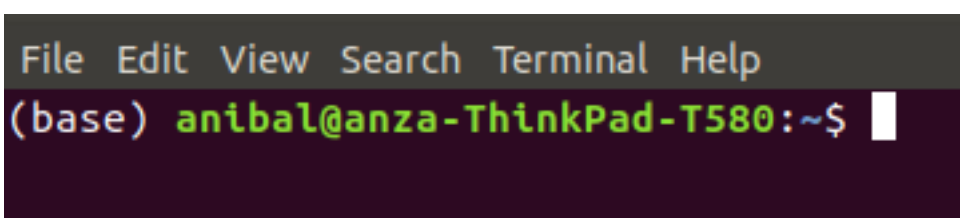
- Miniconda3 installer for Linux

Run the downloaded installer (for a 64-bit system):

```
$ bash Miniconda3-latest-Linux-x86_64.sh
```

Open a new terminal window for conda to take effect. The word *(base)* should appear in front of the computer name in the terminal window, like so:





Verify the installation and update conda in new terminal window with:

```
$ conda env list
$ conda update --all
$ conda upgrade --all
```

Install mamba library/package manager that will be used for installing software dependencies of the tool:

```
$ conda install mamba --yes
```

2.2 Git Installation and Repo Cloning

Git is required for cloning locally (downloading a copy to your local computer) the PBGL CNVseq Github repository. Git and Github are used for version control of software. It keeps track of development, releases, and issues of a software project.

Install **git** for cloning the **pbgl-cnvseq** software repository from Github, where the latetest version of the tool resides:

```
$ mamba install git --yes
```

After the instalation, clone PBGL's CNVseq repository, **pbgl-cnvseq**, to the local computer in any desired directory.

```
$ git clone https://github.com/pbgl/pbgl-cnvseq.git
```

The cloning process will depict the following:

```
Cloning into 'pbgl-cnvseq'...
remote: Enumerating objects: 628, done.
remote: Counting objects: 100% (336/336), done.
remote: Compressing objects: 100% (239/239), done.
remote: Total 628 (delta 109), reused 292 (delta 78), pack-reused 292
Receiving objects: 100% (628/628), 11.79 MiB | 7.03 MiB/s, done.
Resolving deltas: 100% (190/190), done.
```

The **pbgl-cnvseq** repository should have been clones successfully. Verify that the download is complete by listing the folders/files in the directory.

```
$ ls -l
```

The folder called **pbgl-cnvseq** should be listed in the directory.

2.3 Required Libraries with Mamba

CNVseq has multiple dependencies, listed below:

- Samtools
- R
  - configr
  - ggplot2
  - BiocManager
  - Bioconductor-GenomicAlignments
  - Bioconductor-GenomeInfoDb
- Jupyter Notebook
  - IRkernel

The necessary R packages are installed through the Jupyter Notebook. It proves as a faster and error-free way to install **configr**, **ggplot2**, **Biocmanager**, **Bioconductor-GenomicAlignments**, and **Bioconductor-GenomeInfoDb** packages.

There are two ways to install the rest of the necessary libraries to run CNV-seq: automatically or manually.

2.3.1 Automatically (faster)

One YAML file, **environment.yml**, is provided to automatically create a virtual environment and install the dependent libraries through mamba. The file creates the **cnvseq** virtual environment, along R, Jupyter Notebook, and the R-kernel in Jupyter. It also installs the dependent R libraries. Run **environment.yml**:

```
$ mamba env create --file envs/environment.yml
```

Once done, a list of the virtual environments available can be seen by running:

```
$ conda env list
```

Activate (enter) the recently-created virtual environment **cnvseq**:

```
$ conda activate cnvseq
```

Once done, the virtual environment should be activated and all the necessary packages should be installed. This can be verified with:

```
$ conda list
```

2.3.2 Manually (slower)

To manually create and activate an environment, run:

```
$ conda create --name cnvseq
$ conda activate cnvseq
```

Start running the installations of the necessary libraries:

```
$ mamba install --channel conda-forge notebook r-irkernel r-biocmanager --yes
$ mamba install --channel bioconda samtools bioconductor-genomicalignments␣
↪bioconductor-genomeinfodb --yes
$ mamba install --channel r r-ggplot2 --yes
$ mamba install --channel pcgr r-configr --yes
```

Once done, all the necessary packages should be installed. This can be verified with:

```
$ conda list
```

3 Running Jupyter

To activate Jupyter, run the following in the terminal:

```
$ jupyter notebook
```

This command will start a Jupyter session inside the directory the command is run. The user can navigate between directories, visualize files, and edit files in a web browser by clicking on directories or files, respectively.

Look for the directory **pbgl-cnvseq** and click on it. Click on **tool** directory, which contains three directories and two Jupyter Notebooks. Here is a breakdown of each:

- *config*:
  - directory containing configuration files specifying file paths, parameter definitions, and comparison lists
- *helper-functions*:
  - directory containing R scripts with functions to calculate and plot CNVs
- *output*:
  - directory that will contain both tab-files and images output after running a CNVseq analysis
- two Jupyter Notebooks:
  - RCNV-seq-sorghum-example.ipynb
    - example analysis of a comparison between a control and mutant of sorghum
  - RCNV-seq-template.ipynb
    - template for the user

Note: Jupyter lets the user duplicate, rename, move, download, view, or edit files in a web browser. This can be done by clicking the box next to a file and choosing accordingly.

3.1 Editing the Configuration File

In order to run the CNVseq Jupyter Notebook, the user needs to feed it with a configuration file (**config-CNVseq.yml**) that specifies the paths to the bam files, comparisons to be done, chromosomes to analyze, and parameter definitions for calculating and plotting CNVs.

The configuration file **config-CNVseq.yml** can be found in the **pbgl-cnvseq/tool/config** directory. The configuration file contains the following:

```
# parameters for CNV/window size calculations defaults from HLiang:
# https://github.com/hliang/cnv-seq
parameters:
  annotate: TRUE
  bed_file_present: FALSE
  bigger: 1.5
  log2: 0.6
  pvalue: 0.001
  window_size: 10000

# list of chromosomes to be analyzed
chromosomes:
```

(continues on next page)

(continued from previous page)

```
  - Chromosome1
  - Chromosome2
  - Chromosome3
  - AnotherChromosome

# folder to store outputs; should be inside "output" directory, example:
# output_path: output/run-name or output/organism
output_path: output/organism-being-analyzed

# path to bed file for varying window sizes (optional)
bed_path: /path/to/bed/file.bed

# paths to bam files
paths:
  control-1: &control-1 /path/to/bam/file/control-1.bam
  mutant-1: &mutant-1 /path/to/bam/file/mutant-1.bam
  mutant-2: &mutant-2 /path/to/bam/file/mutant-2.bam

# comparisons to be analyzed
comparisons:
  control-1-vs-mutant-1:
    control: *control-1
    mutant: *mutant-1
  control-1-vs-mutant-2:
    control: *control-1
    mutant: *mutant-2
```

Note: The user needs to edit **config-CNVseq.yml** to point towards bam/bed files; specify comparisons and chromosomes to analyze; and define the parameters to calculate/plot CNVs.

One example configuration files is provided (**config-CNVseq-sorghum-example.yml**). The configuration file **config-CNVseq.yml** contains multiple fields to be defined by the user.

- *parameters*:
  - parameters used to create window sizes, thresholds, plots, etc
  - the parameter defaults are provided accroding to HLiang's original values
- *chromosomes*:
  - list of chromosome names to analyze
  - chromosome names can be found in a bam file's header using the following samtools command:

```
$ samtools view -h my_bam_file.bam | less -S
```

- *output_path*:
  - directory that will contain both tab-files and images output after running a CNVseq analysis
  - the defined path will be inside the **pbgl-cnvseq/tool/output** directory
  - *images*:
    - directory that will store the CNV image outputs per comparison of:
    - all chromosomes in one plot
    - each chromosome individually in one plot

        - any zoomed-in region plot of one chromosome
    - *tab-files*:
        - directory that will contain two types of output tab-delimited files:
            - hits used to calculate CNVs, containing chromosome, start, end, width, control coverage, mutant coverage
            - CNVs per chromosome per comparison, containing CNV number, chromosome, start, end, size, log2, and p-value
- *bed_path*:
    - path pointing to bed file containing targets if using varying window sizes
    - it is recommended to run the analysis without a **.bed** file; plots will only reflect the targets in a **.bed** file
    - if a **.bed** file is provided, the *bed_file_present* parameter under the *parameters* section has to be changed to *TRUE*
- *paths*:
    - sample names and their respective paths to **.bam** files
    - samples can be named as desired but the sample name must be repeated after the colon and prefixed with a & sign
    - the & prefix sign is used to reference the sample's path in different places of the same configuration file
    - example use:

```
paths:
  mysample: &mysample /home/username/bam_files/mysample.bam
  XYZ-123: &XYZ-123 /home/username/bam_files/XYZ-123.bam
  potato95: &potato95 /home/username/bam_files/potato95.bam
```

- *comparisons*:
    - comparison names with respective control and mutant samples per comparison
    - each comparison can be named as desired
    - the sample names to be used as *control* and *mutant* need to be prefixed by a * sign
    - the * prefixed sign is used to extract the sample's path defined in the *paths* section
    - example:

```
comparisons:
  comparison-1:
    control: *mysample
    mutant: *potato95
  a-different-comparison-278asd:
    control: *mysample
    mutant: *XYZ-123
```

3.2 Running a RCNV_seq-template Jupyter Notebook

Note: It is recommended to duplicate the **RCNV-seq-template** notebook and then renaming the copy before doing any edits to the notebook.

In the **pbgl-cvnseq/tool** directory, click on **RCNV-seq-template** and a new tab in your web-browser will open the notebook.

The notebook contains cells that are populated by text or code. Instructions are provided in the notebook to guide the user. To run a cell, click on the corresponding cell and click on the *Run* button on the top of the notebook. Another way to run a cell can be done by clicking on the corresponding cell and pressing **Ctrl + Enter** or **Shift + Enter**.

The notebook consists of 6 sections:

1. Installing Required Libraries (optional)
2. Loading Required Libraries (mandatory)
3. User Input (mandatory)
4. CNV Calculations
5. Plotting
6. Plotting a Zoomed-In Region of One Chromosome

3.2.1 1 - Installing Required Libraries (optional)

- libraries being installed:

```
In [ ]: # install necessary libraries using R functions
        if (!requireNamespace("BiocManager", quietly = TRUE))
            install.packages("BiocManager")

        BiocManager::install(c("GenomicAlignments", "GenomeInfoDb"))
        install.packages("configr")
        install.packages("ggplot2")
```

- to be run if the mamba installations were not successful or the loading of the required libraries fails under the **Loading Required Libraries** section

3.2.2 2 - Loading Required Libraries (mandatory)

- libraries being loaded:

```
In [ ]: # load necessary libraries
        library(GenomicAlignments)
        library(ggplot2)
        library(configr)

        # specify source R script with helper functions
        source("helper-functions/RCNV_seq-helper.R")
        source("helper-functions/cnvHLiang.R")
```

- this cell will load necessary libraries and scripts containing the necessary functions to be used
- if running this cell fails, some libraries may be missing from the installation

- to fix this issue, run the installations under the **Installing Required Libraries**

3.2.3 3 - User Input (mandatory)

- the user needs to write the appropriate name of the configuration file being used in the following cell:

```
In [ ]: configPath <- "config/config-CNVseq.yml"
```

- the bottom cell will extract the fields defined in the configuration file:

```
In [ ]: config <- read.config(configPath)
```

3.2.4 4 - CNV Calculations

- function to calculate all the hits and CNVs:

```
In [ ]: cnvCalculate(config)
```

- output tabulated files are stored inside **pbgl-cnvseq/tool/output/name_defined_in_output_path_of_config/tab-files** directory

3.2.5 5 - Plotting

- function to plot two types of images:
  1. CNVs of all chromosomes in the same plot
  2. CNVs of one chromosome per plot

```
In [ ]: cnvPlot(config, imgType="", yMin= , yMax= )
```

- function parameters are:
  - *config* - configuration file defined under **User Input** section
  - *imgType* - image extention to use; available options are: *png*, *jpeg*, *svg*, and *pdf*; default to *png*
  - *yMin* - y-axis bottom limit; default to *-5*
  - *yMax* - y-axis upper limit; default to *5*
- output images are stored inside the **pbgl-cnvseq/tool/output/name_defined_in_output_path_of_config/images** directory
- it is recommended to inspect the CNV-plots with default y-limits and then modify
- *cnvPlotting* can be used in two ways: with default values or user-defined values
  - with default values, the parameters can be omitted and set to *imgType="png"*, *yMin=-5* and *yMax=5*
  - the following two commands have the same parameter values and will output the same plots:

```
cnvPlot(config)
cnvPlot(config, imgType="png", yMin=-5, yMax=5)
```

3.2.6 6 - Plotting a Zoomed-In Region of One Chromosome

- function to plot a specific zoomed-in region of one chromosome

```
In [ ]: cnvPlotZoom(config, tabFile, chromosome="", start= , end= , yMin= , yMax= ,␣
↪imgType="")
```

- function parameters are:
  - *config* - configuration file defined under **User Input** section
  - *tabFile* - tabulated file containing all hits; requires user-input to define path to all-hits tabulated file
  - *chromosome* - chromosome name to focus on; default to *NA*
  - *start* - start of window in bp; both scientific notation is accepted: *100000* or *10e4*; defaults to *NA*
  - *end* - end of window in bp; the same applies as in the *start* parameter; defaults to NA
  - *yMin* - y-axis bottom limit; default to *-5*
  - *yMax* - y-axis upper limit; default to *5*
  - *imgType* - image extention to use; available options are: *png*, *jpeg*, *svg*, and *pdf*; defaults to *png*
- requires path definition of *tabFile* parameter in the cell:

```
In [ ]: tabFile <- "output/run-name/tab-files/tab-file.tab"
```

4 References

BMC Bioinformatics Publication:

- CNV-seq, a new method to detect copy number variation using high-throughput sequencing

GitHub repositories:

- hliang/cnv-seq
- Bioconductor/copy-number-analysis
- pbgl/pbgl-cnvseq
- amora197/pbgl-cnvseq