BSAvis
BSA VISUALISATION TOOL

User Manual

BSAvis Version 1.0

Authors: *Elisabetta Galatola, Rebecca Guy, Weiyi Huang, Sweta Jajoriya, Claudia Rey-Carrascosa*

Group 1 Applied Bioinformatics MSc 2020/2021
Cranfield University - Cranfield, Bedford, UK

Table of Contents

Introduction

BSAvis is a flexible tool with implemented Bulk Segregant Analysis (BSA) methods to analyse bulks variants, capable of generating publication-quality plots.

This user manual follows use-case scenarios and provides information in two parts:

- **Part 1:** refers to the section to run the <u>BSAvis package as a script</u> within RStudio. This section describes both combined and stepwise BSA and plot functionalities, per implemented BSA method.
- **Part 2:** refers to the section to run the BSAvis package as an <u>interactive R-Shiny dashboard</u>. This section describes the BSA plotting options and investigative tool functionality for the implemented BSA methods.

Prerequisites

This tool was tested and can be run both on Windows and MacOS.
R and RStudio integrated development environment (IDE) (versions >=3.6.1) are required. If not yet installed on your computer, download the newest versions from the following links:

- **RStudio (free version):** https://www.rstudio.com/products/rstudio/download/
- **R:** https://cran.r-project.org

Additionally, BSAvis package requires merged Variant Calling Format (VCF) files as input files, generated with `GATK4` variant calling functions. Considering that the VCF file needs to include an `AD` (Allelic Depth) column, previous versions to `GATK4` <u>are not</u> recommended. Joint genotyping is also required to obtain a single VCF file that includes variants from both bulks. For best results from the BSAvis package and to obtain the required VCF file, the following (alignment and variant calling) sample scripts are recommended

- https://github.com/FadyMohareb/BSAvis_GP_2020/blob/main/QC_Alignment_VC/alignment_variantCalling/steps/alignment_steps.txt
- https://github.com/FadyMohareb/BSAvis_GP_2020/blob/main/QC_Alignment_VC/alignment_variantCalling/steps/variantCalling_steps.txt

Availability
Code and Documentation can be accessed at: https://github.com/FadyMohareb/BSAvis_GP_2020

Important Notes
Please refer to the **technical documentation** that accompanies the package for a better understanding of the implemented functions.

<u>Be sure to follow steps **A. B. C.** and **D.** on the next page before proceeding with the analyses</u>.

A. Test Files

For testing purposes (and to make this user manual easier to follow), the following files have been provided:

- `test.RData`
- `test.vcf`

Note that these files refer to a subset of a VCF file which contains information for only two chromosomes (for both bulks).

Follow the steps below to download the test files and set your working directory on RStudio:

1. **Create a new folder** on your computer, **naming it BSAvis**

2. **Download the BSAvis repository** from GitHub by clicking on the green "Code" 1 button on the upper-right corner and selecting the "Download ZIP" 2 option. **(Figure 1)**

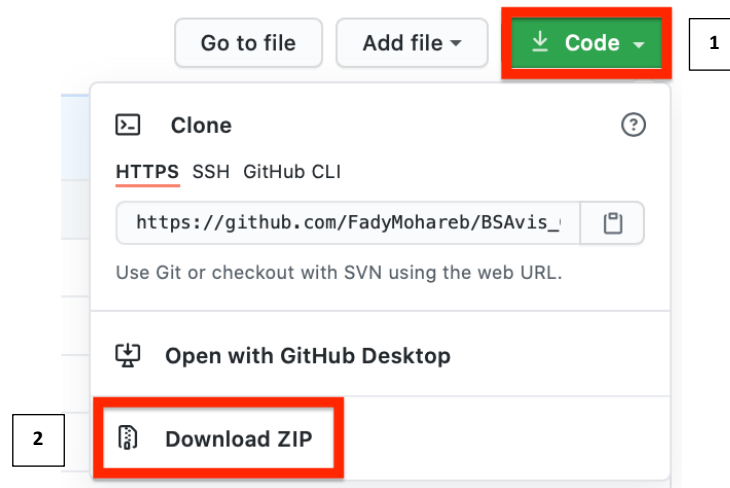   *Repository link: https://github.com/FadyMohareb/BSAvis_GP_2020*

Figure 1. GitHub screenshot, showing the required buttons to click.

3. **Unzip** the downloaded repository and search for the test files, moving them inside the previously generated BSAvis folder

4. Open RStudio and set you working directory, either by

- Running one of the following commands on the console of RStudio (depending on your Operating System and after editing the portion highlighted in blue):

    o **For MacOS users**: `setwd("`*/Users/...yourPath...*`/`**BSAvis**`")`

    o **For Windows users**: `setwd("`*C:\Users\...yourPath...*`\`**BSAvis**`")`

To run commands on the console of RStudio, paste them in and press enter on your keyboard. Also note that if the BSAvis folder was stored on your Desktop, your file path would be:

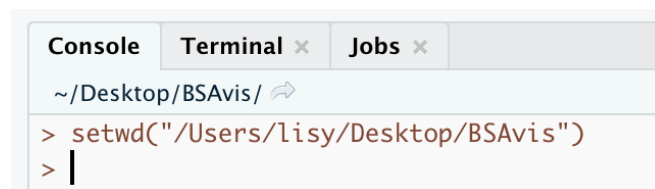`/Users/YourUserName/Desktop/BSAvis` **(Figure 2)**

Figure 2. Screenshot of RStudio console (bottom-left windows on RStudio). Example for setting the working directory on MacOS.

- or manually, by searching the folder inside the bottom-right window in RStudio (in the "Files" panel), clicking on the gear button and selecting "Set as working directory" **(Figure 3)**
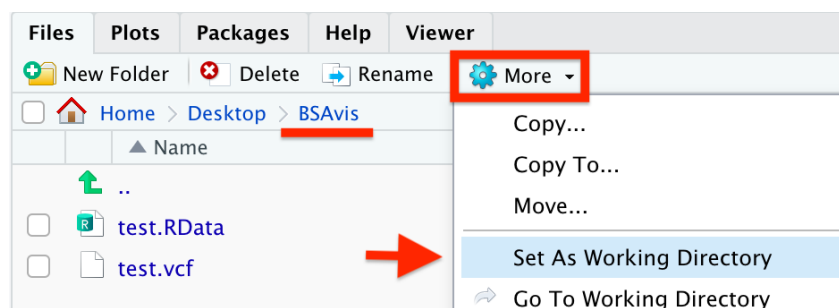

Figure 3. Screenshot of RStudio, showing how to manually set the working directory.

Please note that steps **2.** and **3.** can be skipped if you wish to analyze your own merged VCF files.

B. Installing BSAvis Dependencies

Install and load all the required dependencies, using the following commands on RStudio:

1. **Install required packages**:

   - ```install.packages(C("vcfR","ggplot2","dplyr","tidyr","devtools"))```

2. **Load packages**:

   - ```library(vcfR)```
   - ```library(ggplot2)```
   - ```library(dplyr)```
   - ```library(tidyr)```

C. Installing BSAvis Package

To start your analyses, you will need to install and load the BSAvis Package from GitHub.

This can be done following two simple steps, on RStudio:

1. **Install the BSAvis package** using the following command:

   - ```devtools::install_github("FadyMohareb/BSAvis_GP_2020/BSAvis")```

2. **Load the BSAvis package** with the following command:

   - ```library(BSAvis)```

D. Reading the VCF file

After successfully loading the BSAvis package, the merged VCF file requires to be read and loaded inside the working environment.

Be aware that **this step is common for every implemented method and is needed to run the rest of the package functions.**

Using the provided testing file, proceed running the following command:

- `vcf_list <- readBSA_vcf("`*`test.vcf`*`")`

This step will take up to 20-30 seconds to complete. Alternatively, to familiarize yourself with the package, you can directly load the `vcf_list` object using the provided .RData file (`test.RData`) to skip the `readBSA_function()` and save time.

To do so, simply run the following command (be sure to have set your working directory in the BSAvis folder, as described on page 3, step 4.):

- `load("`*`test.RData`*`")`

Once the VCF data is loaded in the environment **(Figure 4),** follow the steps in **Part 1** to run BSAvis via scripts, or move to **Part 2** to run BSAvis via an interactive R-Shiny application.
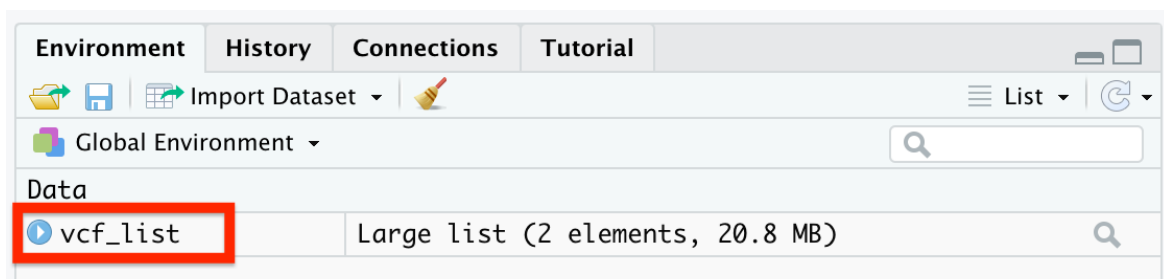

Figure 4. Screenshot of the loaded `vcf_list` object inside the RStudio working environment.

To use BSAvis with your own merged VCF file, run the `readBSA_vcf()` function as mentioned at the beginning of page 5:

- `vcf_list <- ` ***`readBSA_vcf("yourFile.vcf`***`")`

Important – when using a complete VCF file, this step might take time to complete (approximately 15-20 minutes). It is strongly recommended to leave RStudio open and running, to properly process the data. Proceed only after the red button in the top right corner of the console disappears. **(Figure 5)**
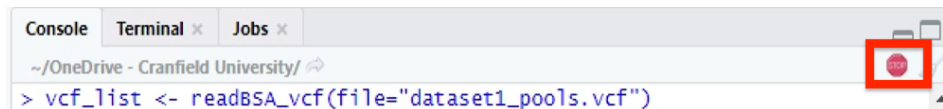

Figure 5. Screenshot of the console in RStudio. Highlighted in red can be found the "STOP" button, which will disappear once the `vcf_list` object gets loaded inside the working environment.

1. Combined BSA and Plotting

Functions included in this section will apply the chosen BSA method and return plots.
A step-by-step approach to BSA and plotting is also included (see Page 10).

Important note – be sure to have followed <u>steps A-D</u> and run the `readBSA_vcf()` function before proceeding (see section **D.** on page 5). All of the examples refer to the testing dataset.

1.1. SNP-index Method

To apply the SNP-index method and plot the results, the user needs to run the `SNPindex_plot()` function.

Example:
```
SNPindex _plot(vcf.list=vcf_list,
          wtBulk="pool_minus",
          mBulk="pool_plus",
          variants="SNP",
          min.SNPindex=0.3,
          max.SNPindex=0.9,
          min.DP=50,
          max.DP=200,
          min.GQ=99,
          chrID="SL4.0ch03",
          chr=3,
          windowSize=1000000,
          windowStep=10000,
          filename="plot_SNPindex_ch",
          path="/currentWorkingDirectory",
          dpi=1200, # if set, the plot gets saved
          width=7.5,
          height=5,
          units="in")
```

Important:
- the minimum required parameters are: `vcf.list`, `wtBulk`, `mBulk`, `chrID` and `chr`
- `filename`, `path`, `width`, `height` and `units` are all part of the plot-saving functionality and are directly linked with the `dpi` parameter. Without setting the `dpi`, all the parameters highlighted in blue will be ignored
- parameters highlighted in bold are already set to default but can be customized.
- to properly implement the function, please refer to the help page (by typing `?SNPindex_plot` on RStudio) or the technical documentation

1.2. Δ(SNP-index) Method

To apply the Δ(SNP-index) method (which extends the SNP-index method) and plot the results, the user needs to run the `deltaSNPindex_plot()` function.

Example:

```
deltaSNPindex_plot(vcf.list=vcf_list,
              wtBulk="pool_S3781_minus",
              mBulk="pool_S3781_plus",
              variants="SNP",
              min.SNPindex=0.3,
              max.SNPindex=0.9,
              min.DP=50,
              max.DP=200,
              min.GQ=99,
              chrID="SL4.0ch03",
              chr=3,
              windowSize=1000000,
              windowStep=10000,
              filename="plot_deltaSNPindex_ch",
              path="/currentWorkingDirectory",
              dpi=1200, # if set, the plot gets saved
              width=7.5,
              height=5,
              units="in")
```

Important:

- the minimum required parameters are: `vcf.list`, `wtBulk`, `mBulk`, `chrID` and `chr`
- `filename`, `path`, `width`, `height` and `units` are all part of the plot-saving functionality and are directly linked with the `dpi` parameter. Without setting the `dpi`, all the parameters highlighted in blue will be ignored
- parameters highlighted in bold are already set to default but can be customized
- to properly implement the function, please refer to the help page (by typing `?deltaSNPindex_plot` on RStudio) or the technical documentation

1.3. SNP-ratio Method

To apply the SNP-ratio method and plot the results, the user needs to run the `SNPratio_plot()` function.

Example:
```
SNPratio_plot(vcf.list=vcf_list,
        wtBulk="pool_S3781_minus",
        mBulk="pool_S3781_plus",
        variants="SNP",
        min.SNPratio=0.1,
        min.DP=50,
        max.DP=200,
        chrID="SL4.0ch03",
        chr=3,
        degree=2,
        span=0.03,
        filename="plot_SNPratio_ch",
        path="/currentWorkingDirectory",
        dpi=1200, # if set, the plot gets saved
        width=7.5,
        height=5,
        units="in")
```

Important:
- the minimum required parameters are: `vcf.list`, `wtBulk`, `mBulk`, `chrID` and `chr`
- `filename`, `path`, `width`, `height` and `units` are all part of the plot-saving functionality and are directly linked with the `dpi` parameter. Without setting the `dpi`, all the parameters highlighted in blue will be ignored
- parameters highlighted in bold are already set to default but can be customized.
- to properly implement the function, please refer to the help page (by typing `?SNPratio_plot` on RStudio) or the technical documentation

2. Stepwise BSA and Plotting

Functions included in this section will apply the chosen BSA method and return the plots to the user in a multistep process, conversely to the previous section which describes the process for a simple combined functionality approach to BSA and plotting.

Important note – be sure to have followed <u>steps A-D</u> and run the `readBSA_vcf()` function before proceeding (see section **D.** on page 5). All of the examples refer to the testing dataset.

2.1. SNP-index/Δ(SNP-index) Method

To apply the SNP-index method (and subsequent Δ(SNP-index) method), you will need to follow the steps listed below.

Note – parameters highlighted in bold are already set to default but can be customized. In case you are happy with these, you do not need to specify them inside the function.

Please refer to the package function help page (by calling `?nameOfTheFunction`) or technical documentation for a better understanding of the functions.

1. **Calculate SNP-indices for both bulks** using the `calc_SNPindex()` function

Example:

```
vcf_df_SNPindex <- calc_SNPindex(vcf.df=vcf_list$df,
                                 wtBulk="pool_minus",
                                 mBulk="pool_plus",
                                 variants="SNP")
```

2. **Filter variants** using the `filter_SNPindex()` function.

Example:
```
vcf_df_SNPindex_filt <- filter_SNPindex(
                                vcf.df.SNPindex=vcf_df_SNPindex,
                                min.SNPindex=0.3,
                                max.SNPindex=0.9,
                                min.DP=50,
                                max.DP=200,
                                min.GQ=99)
```

3. **Extract chromosome IDs** using the `extract_chrIDs()` function.

Example:
```
chromList <- extract_chrIDs(meta=vcf_list$meta)
```

4. **Calculate the sliding windows** based on the chromosome length of the specified
   chromosome using the `slidingWindow()` function**.**

Example:
```
SNPindex_windows <- slidingWindow(
                    meta=vcf_list$meta,
                    chrList=chromList,
                    chrID="SL4.0ch03",
                    windowSize=1000000,
                    windowStep=10000,
                    vcf.df.SNPindex.filt=vcf_df_SNPindex_filt)
```

5. **Plot SNP-index** across the positions of a given chromosome using the
   `plot_SNPindex()` function.

Example:
```
plot_SNPindex(SNPindex.windows=SNPindex_windows,
              chr=3,
              filename="plot_SNPindex_ch",
              path="currentWorkingDirectory",
              dpi=1200,
              width=7.5,
              height=5,
              units="in")
```

6. **Calculate Δ(SNP-index)** using the `calc_deltaSNPindex()` function.

Example:
```
deltaSNPindex_windows <- calc_deltaSNPindex(
                            SNPindex.windows=SNPindex_windows)
```

7. **Plot Δ(SNP-index) across the positions of a given chromosome** using the `plot_deltaSNPindex()` function.

Example:
```
plot_deltaSNPindex(
                deltaSNPindex.windows=deltaSNPindex_windows,
                chr=3,
                filename="plot_deltaSNPindex_ch",
                path="currentWorkingDirectory",
                dpi=1200,
                width=7.5,
                height=5,
                units="in")
```

2.2. SNP-ratio Method

To apply the SNP-ratio method, the following steps are required.

Note – parameters highlighted in bold are already set to default but can be customized. In case you are happy with these, you do not need to specify them inside the function.

Please refer to the package function help page (by calling `?nameOfTheFunction`) or technical documentation for a better understanding of the functions.

1. **Calculate SNP-ratios** for both bulks using the calc_SNPratio() function

   Example:
   ```
   vcf_df_SNPratio <- calc_SNPratio(vcf.df=vcf_list$df,
                                    wtBulk="pool_S3781_minus",
                                    mBulk="pool_S3781_plus",
                                    variants="SNP")
   ```

2. **Filter variants** using the `filter_SNPratio()` function

   Example:
   ```
   vcf_df_SNPratio_filt <- filter_SNPratio(
                           vcf.df.SNPratio=vcf_df_SNPratio,
                           min.SNPratio=0.1,
                           min.DP=50,
                           max.DP=200)
   ```

3. **Extract chromosome IDs** using the `extract_chrIDs()` function

   Example:
   ```
   chromList <- extract_chrIDs(vcf_list$meta)
   ```

4. **Plot SNP-ratio** across the positions of a given chromosome using the `plot_SNPratio()` function

Example:
```
plot_SNPratio(vcf.df.SNPratio.filt=vcf_df_SNPratio_filt,
              chrList=chromList,
              chrID="SL4.0ch03",
              chr=3,
              min.SNPratio=0.1,
              degree=2,
              span=0.3,
              filename="plot_SNPratio_ch",
              path="currentWorkingDirectory",
              dpi=1200,
              width=7.5,
              height=5,
              units="in")
```

1. Initial Set-up

This section describes the extended functionality of the BSAvis package, which allows the user to perform interactive BSA analysis using a user-friendly R-Shiny application.

1.1. Installing Shiny Libraries

Before running interactive version of BSAvis, the user is required to manually install and load the required libraries on RStudio, following two steps:

1. **Install the required libraries** using the following command:

- ```
  install.packages(c("shiny", "shinycssloaders", "shinyalert")
  ```

   Proceed only after the installation is completed: a message will show up on the console to warn you when the libraries have finished downloading. **(Figure 6)**
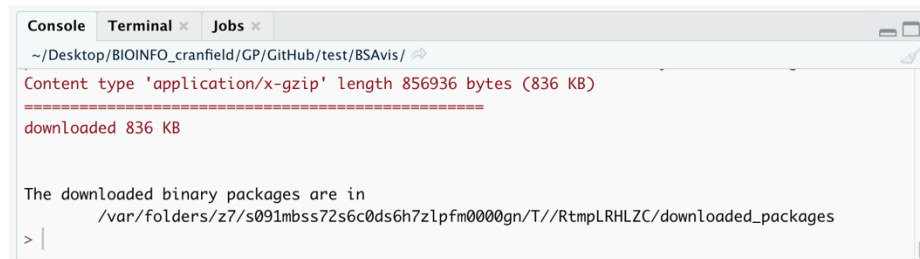
Figure 6. Screenshot of the final message printed on the console of RStudio, after all the required libraries have finished installing.

2. **Load the libraries** running the following commands:

   - ```
     library(shiny)
     ```
   - ```
     library(shinycssloaders)
     ```
   - ```
     library(shinyalert)
     ```

1.2. Running BSAvis R-Shiny Application

After loading the libraries, you will be able to run the BSAvis interactive tool by calling the following function:

- `BSAvis_shiny(`**`vcf_list`**`)`

Important – be sure to have run the `readBSA_vcf()` function, as recommended at the beginning, since its output (vcf_list) is needed for the `BSAvis_shiny()` function (section **D.** on page 5).

After a few seconds, the following window will pop up on your screen: **(Figure 7)**
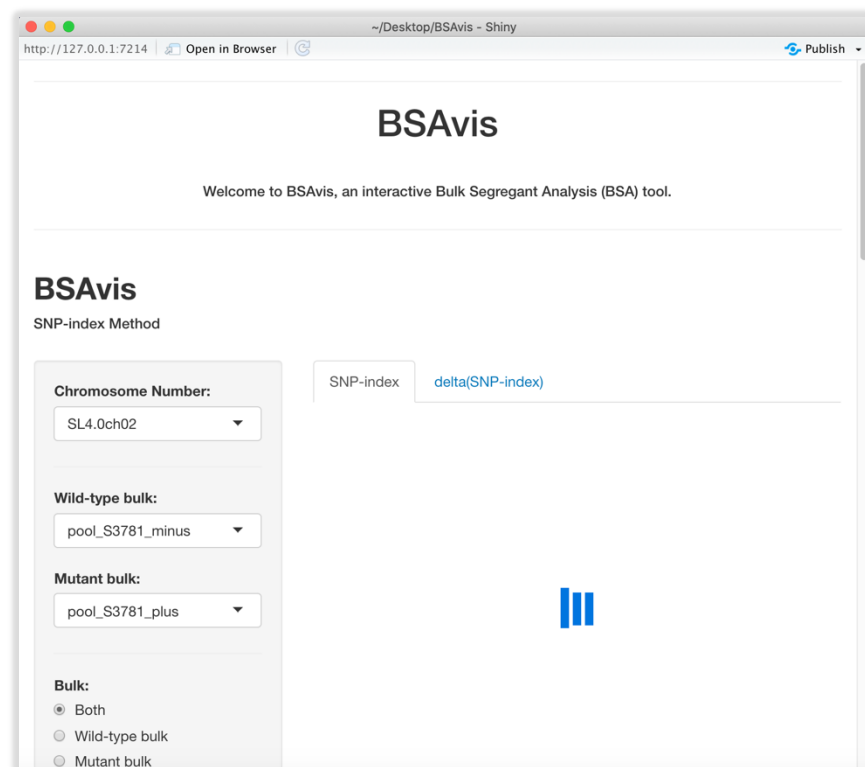
Figure 7. Screenshot of BSAvis R-Shiny App. In blue, an animated loader will show up, waiting for the plot to get generated.

2. BSAvis R-Shiny Application

This section describes all the functionalities included in the BSAvis R-Shiny dashboard.

2.1. SNP-index Method

The first panel refers to the SNP-index method.

The sidebar panel on the left allows the user to select the desired parameters to plot the results. **(Figures 8, 9)**

Figure 7. Screenshot of the first part of the SNP-index method sidebar panel.

Figure 8. Continuation of **Figure 7**. Screenshot of the "filtering parameters" section of the SNP-index method sidebar panel.

On the right of the sidebar panel, the user can switch between the *SNP-index* and *delta(SNP-index* panels to visualise the corresponding plots. **(Figure 9)**
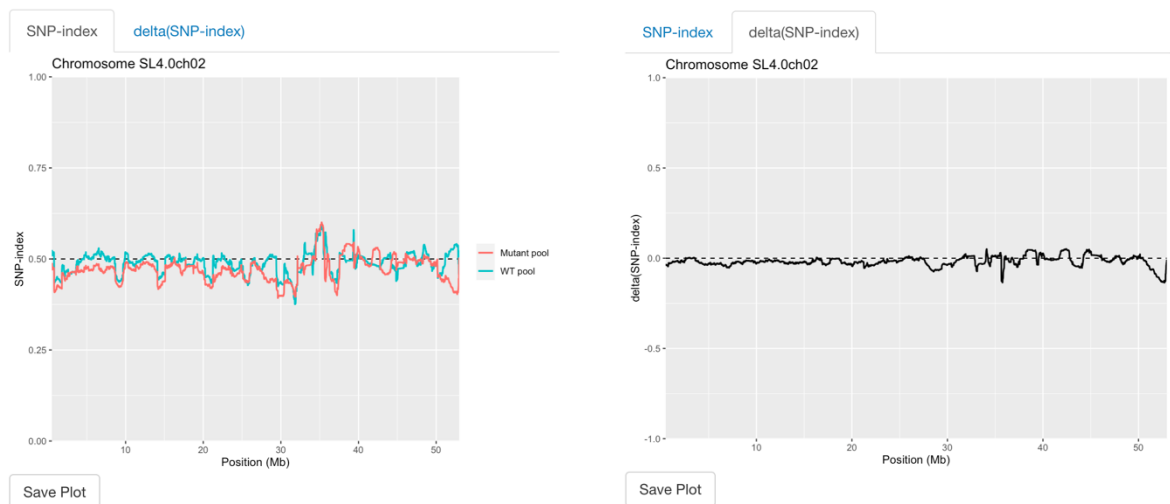
Figure 9. Screenshots of the BSAvis plotting panels for SNP-index (on the left) and delta(SNP-index) (on the right) for chromosome 2 .

2.2. SNP-ratio Method

The second section, following the SNP-index method, refers to the SNP-ratio method.

The sidebar panel on the left allows the user to select the desired parameters of the implemented SNP-ratio method to generate the plots. **(Figure 10)**

Figure 10. Screenshot of the SNP-ratio method sidebar panel.

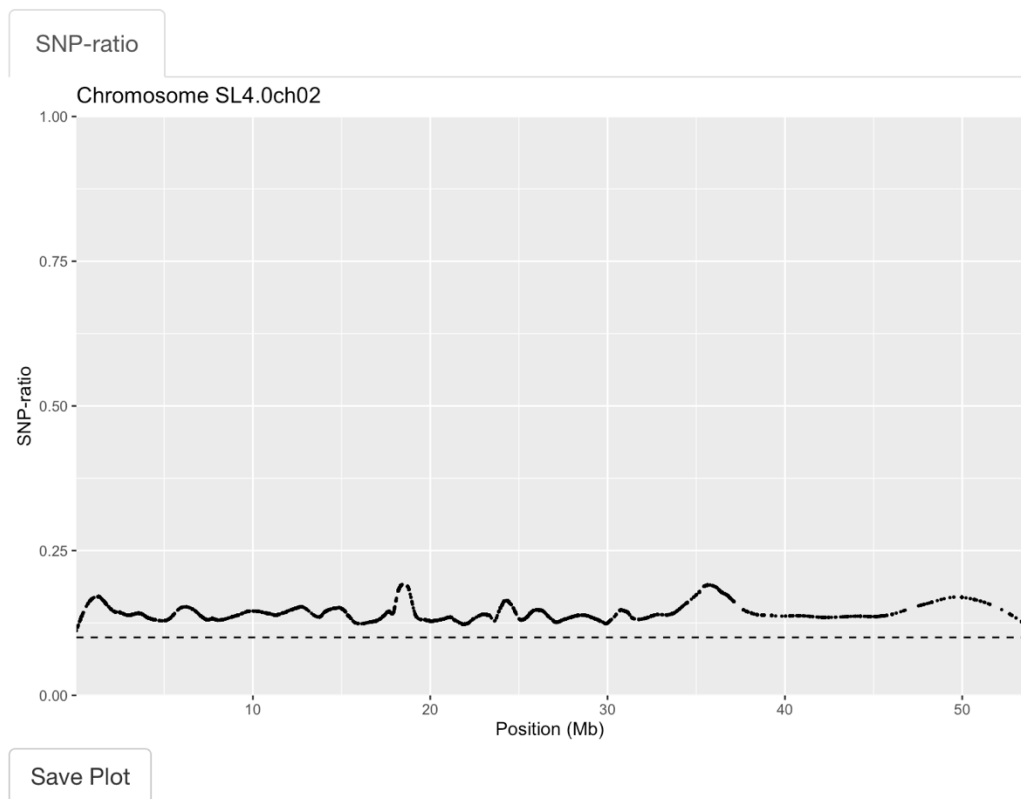SNP-ratio plots will be generated on the right-side of the SNP-ratio sider panel. **(Figure 11)**

Figure 11. Screenshot of the SNP-ratio plotting panel, for chromosome 2.

2.3. Saving Plots

All generated plots can be interactively saved by clicking on the "Save Plot" button, found below the plotting panels. **(Figure 12)**
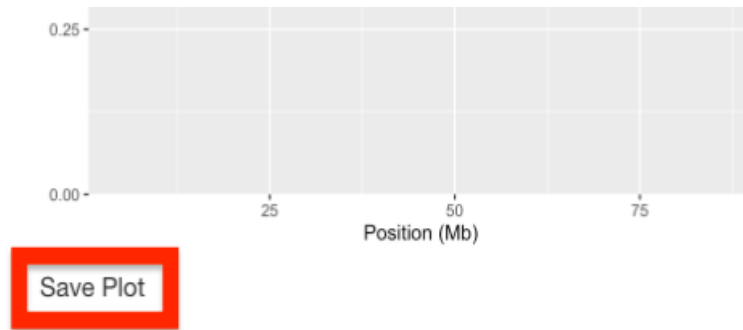

Figure 12. Screenshot of the bottom-part of the plotting panel. Highlighted in red is shown the "Save Plot" button.

After clicking the latter, a saving window will pop-up to enable customizing the saving options. Click the "OK" button when you are happy with the chosen parameters, to proceed saving the plot (in TIFF format). **(Figure 13)**
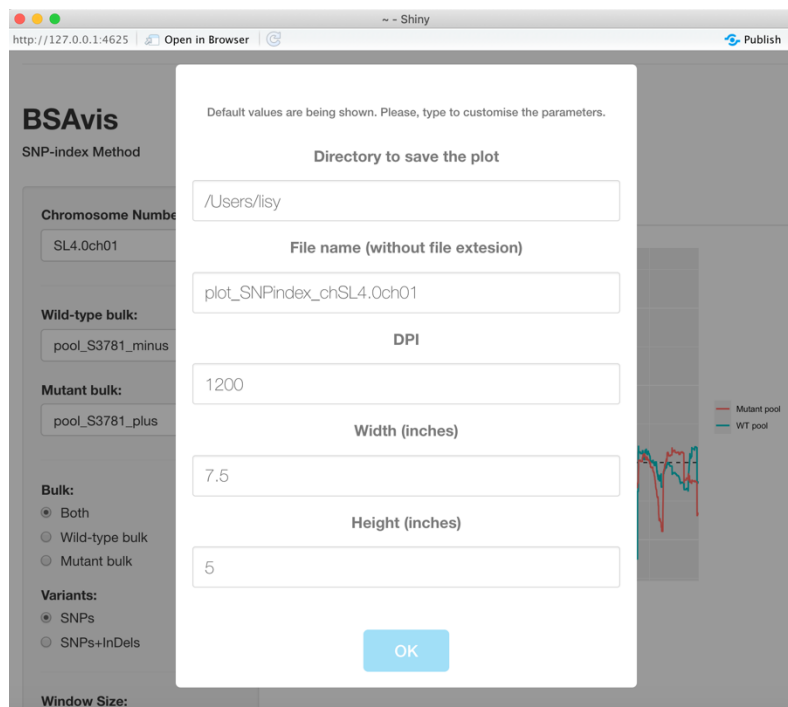

Figure 13. Screenshot of the saving window.

2.4. Zooming Functionality

An additional plotting functionality includes zooming in the plot.

This can be down by dragging, releasing, and <u>double-clicking</u> on the selected area. **(Figure 14)**
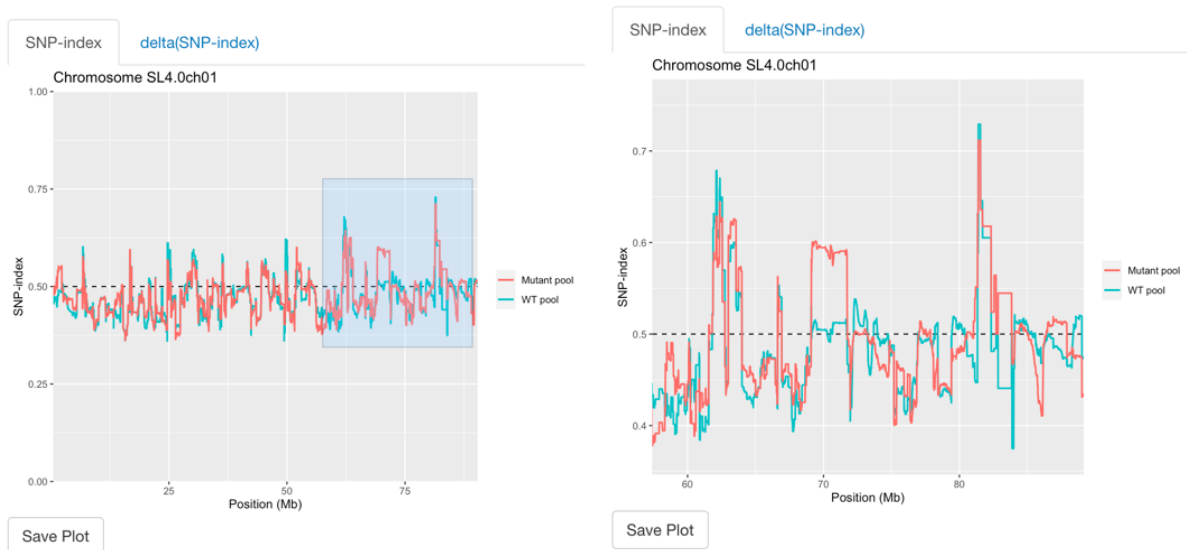

Figure 14. BSAvis screenshot, showing the zooming functionality. The selected area is shown on the left, whereas the zoomed-in area is shown on the right.

Note that this can be done with **any of the plots** found inside the BSAvis tool.

Glossary

MacOS Mac Operating System

BSA Bulk Segregant Analysis

SNP Single Nucleotide Polymorphism

VCF Variant Calling Format file

RData file format for storing and sharing R workspaces

TIFF Tagged Image File Format required for saving publication-quality plots