BSAvis

BSA VISUALISATION TOOL

User Manual

BSAvis Version 1.0

Group Project (2021)
Bioinformatics Team, Cranfield University, Cranfield, UK
Authors: *Rey-Carrascosa, C., Galatola, E., Jajoriya, S., Guy, R., Huang W.*
Supervisors: *Molitor, C., Rezwan, F., Mohareb, F. and Kurowski, T.*

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

Prerequisites

RStudio integrated development environment (IDE) is required, which can be downloaded from the following link: **https://www.rstudio.com/products/rstudio/download/**

BSAvis package requires merged Varant Calling Format (VCF) files as input files, generated by using the GATK variant calling functions. For best results from the BSAvis package, the following Next Generation Sequencing (NGS) data analysis pipeline is recommended ***(add in list of pipeline tools that we used) (see...repository?? To link the pipeline – maybe – or wherever we have the pipeline)***

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.

Be sure to follow steps **A. B. C.** and **D.** on the next page before proceeding with the anlyses.

A. Testing Files

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

- `test_userManual.RData`
- `dataset1_pools.vcf`

Please follow the steps below to download the testing files and set your working directory:

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

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

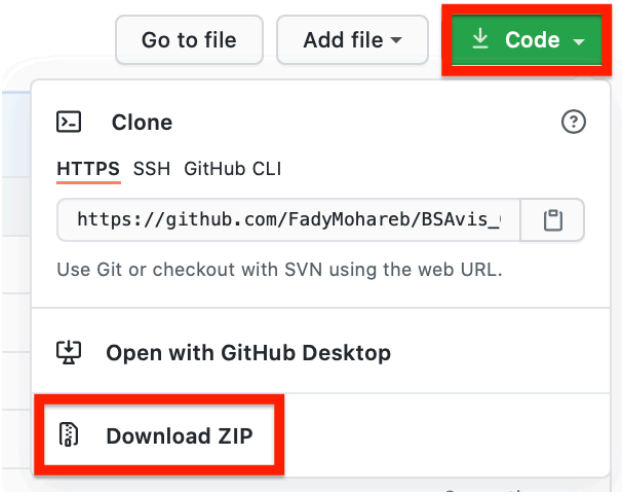

Figure 1. GitHub screenshot, showing the required buttons to click.

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

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

   - running the following command:
     `setwd("~/...yourPath.../BSAvis")`

- or manually, by searching the folder inside the bottom-right window in RStudio (inside the "Files" panel), clicking on the gear button and selecting “Set as working directory” **(Figure 2)**

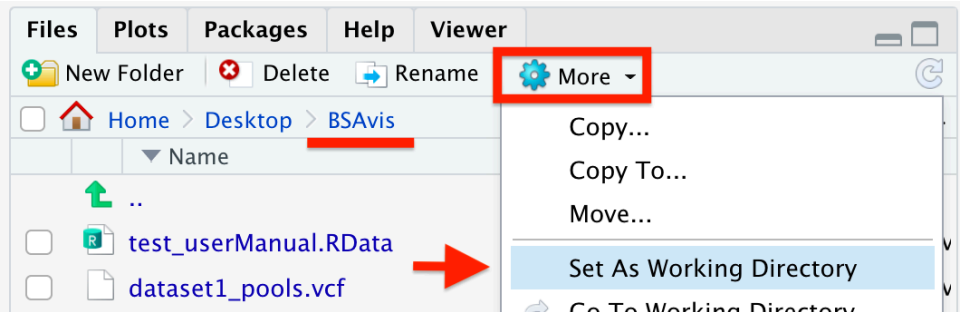

Figure 2. Screenshot of RStudio, showing how to manually set the working directory.

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

B. Installing BSAvis Dependencies

To avoid encountering issues, it is strongly recommended to manually install and load all the required dependencies, using the following commands on RStudio:

1. Install required packages:
    - `install.packages(c("vcfR", "ggplot2", "dplyr", "tidyr")`
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 three simple steps, on RStudio:

1 **Install devtools package** using the following command:

- `install.packages("devtools")`

2 **Install the BSAvis package** using the following command:

- `devtools::install_github("EG-lisy/BSAvis", dependencies=T, force=T)`

3 **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 needs to be read and loaded inside the working directory. 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 merged VCF file, proceed running the following command:

- `vcf_list <- readBSA_vcf("dataset1_pools.vcf")`

Important – 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, until the red button (found at the top-right corner of the console) disappears. **(Figure 3)**

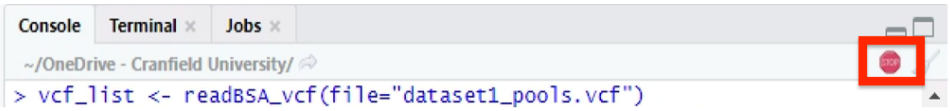

Figure 3. Screenshot of the console in RStudio. Highlighted in red can be found the "STOP" button, which will disappear once the `vcf_list` object gets loaded inside the working environment.

Alternatively, you can direcly load the `vcf_list` object using the provided .RData file (`test_userManual.RData`) to skip having to run the `readBSA_function()` and save time.

To do so, simply click on the file to open it on RStudio, or run the following command:

- `load("test_userManual.RData")`

After the VCF gets loaded in the environment **(Figure 4),** you can proceed following the steps to apply the desired methods described in **Part 1** (running BSAvis package scripts) or **Part 2** (running the BSAvis interactive R-Shiny application).

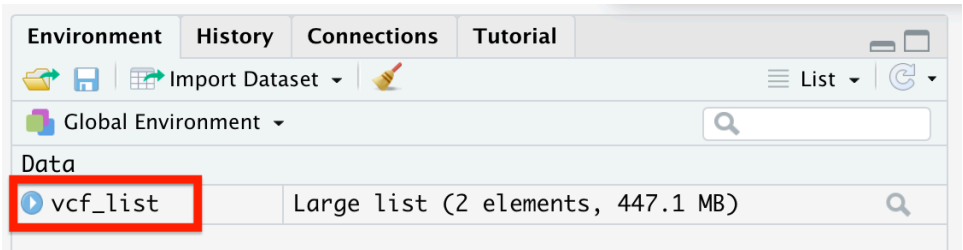

Figure 4. Screenshot of the loaded vcf_list object inside the RStudio working environment.

Important note – if you intend to use your own merged VCF file, you will need to run the `readBSA_vcf()` function, mentioned at the beginning of this section:

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

Part 1 – BSAvis Package Functions

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 x**).

Important note – be sure to have run the `readBSA_vcf()` function before continuing (see previous section, **B.**). **All of the included 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. In case you are happy with these, you do not need to specify them inside the function
- 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. In case you are happy with these, you do not need to specify them inside the function
- 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. In case you are happy with these, you do not need to specify them inside the function
- 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 run the `readBSA_vcf()` function before running a method (see section **B.,** page X). **All of the included examples refer have been applied 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 seven steps, which are 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.

- **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")
```

- **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)
```

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

Example:

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

- **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)
```

- **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")
```

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

Example:

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

- **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 ΔSNP-index 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.

- **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")
```

- **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)
```

- **Extract chromosome IDs** using the `extract_chrIDs()` function

  Example:

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

- **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")
```

Part 2 – Interactive BSA dashboard

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")`

  Be patient and 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 5)**

Figure 5. 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()`, as recommended at the beginning, since its output (vcf_list) is needed for the `BSAvis_shiny()` function (see section **B.**, page X).

After a few seconds, the following window will pop up on your screen: **(Figure 6)**

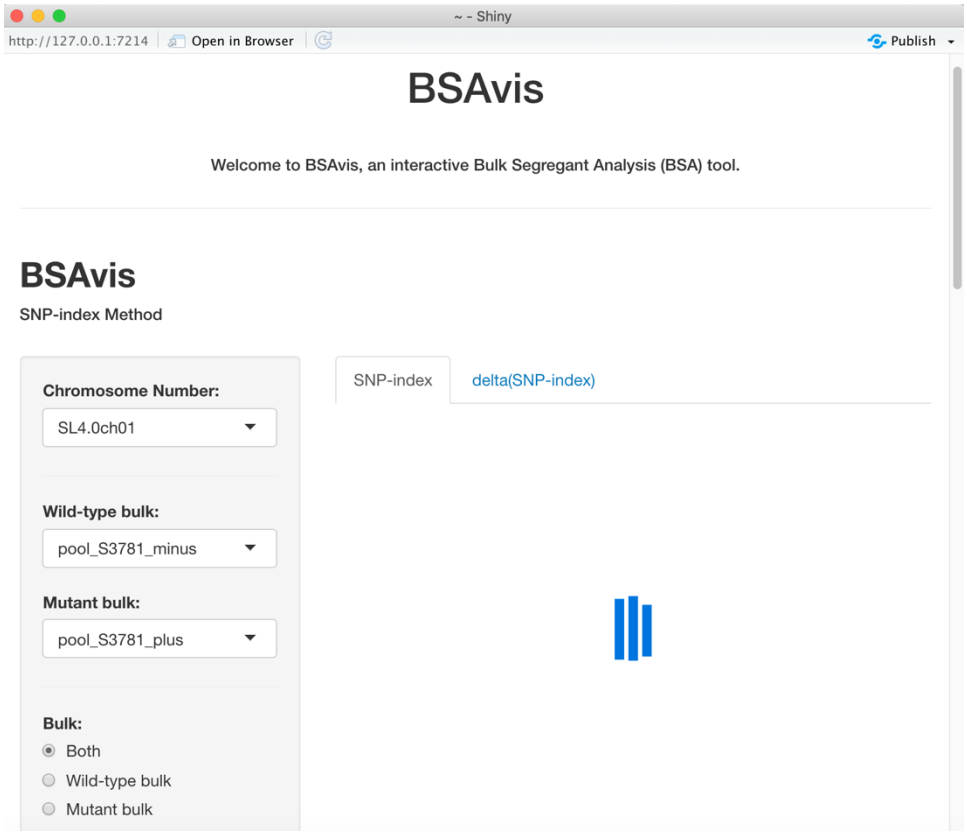

Figure 6. Screenshot of BSAvis R-Shiny App. In blue, an animated loader will show up, waiting for the plot to get generated.

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 7 and 8)**

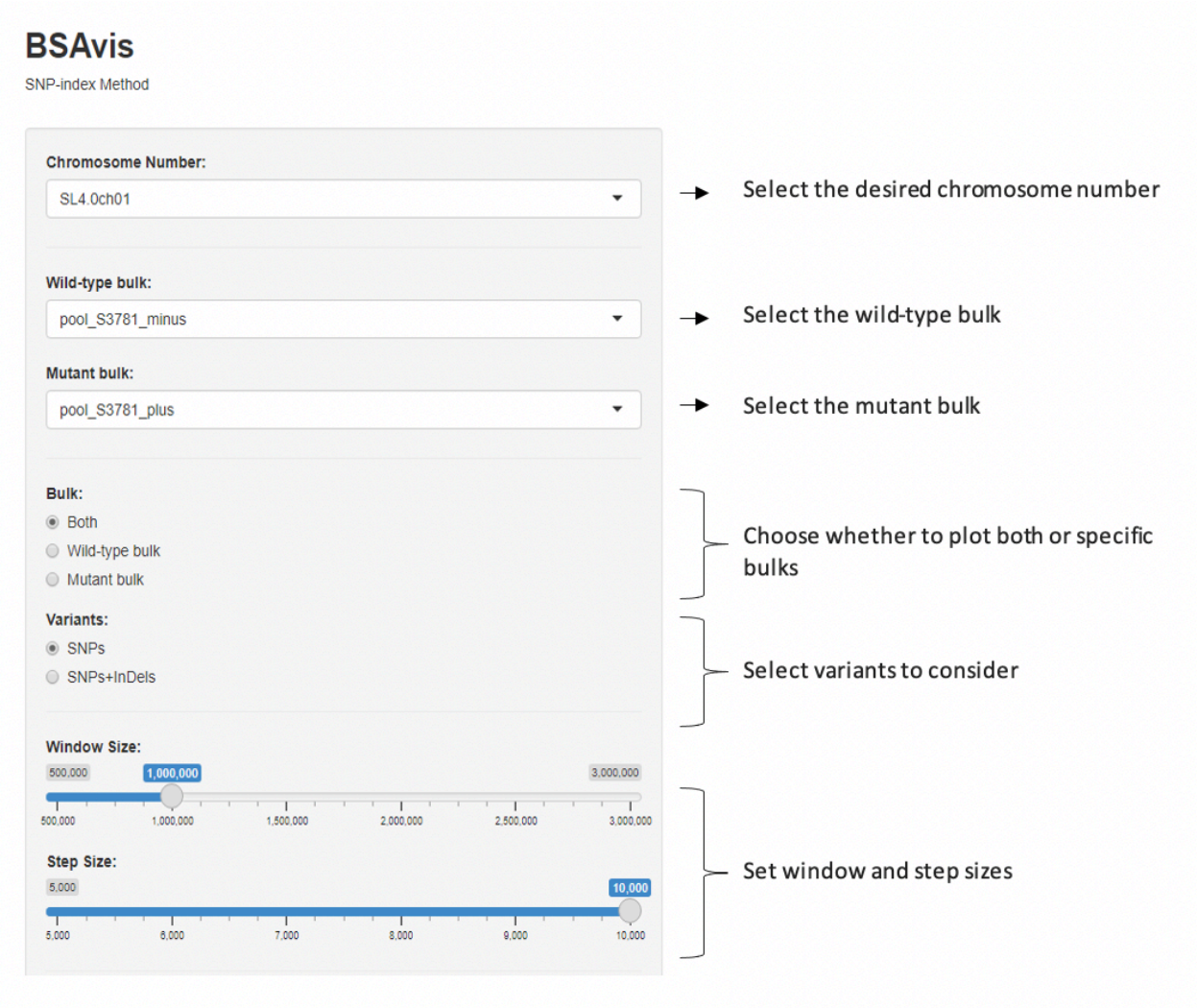

Figure 7. Screenshot of the first part of the SNP-index method sidebar panel.

Figure 8. 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)**

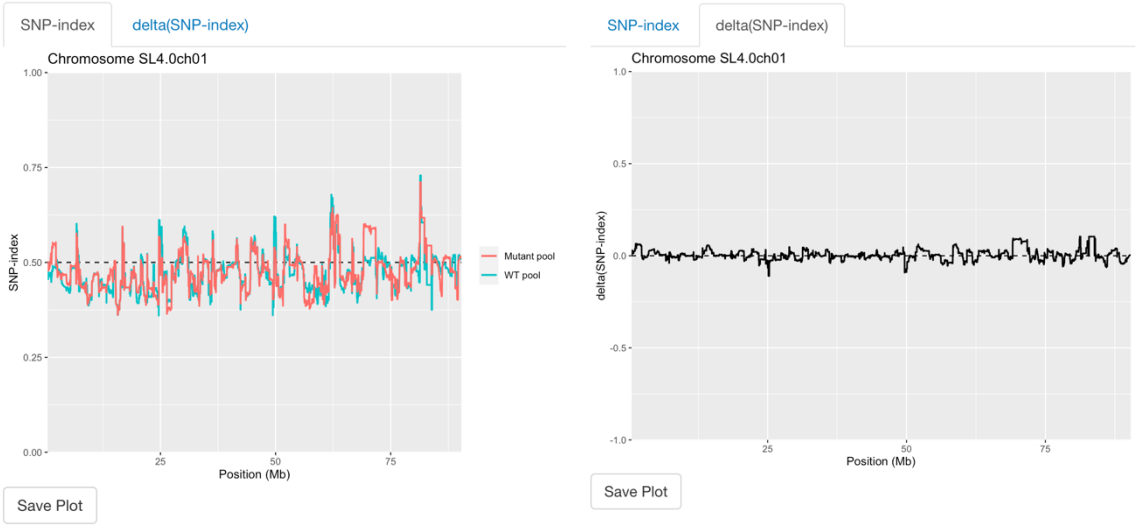

Figure 9. Screenshots of the BSAvis plotting panels for SNP-index (on the left) and delta(SNP-index) (on the right).

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)**

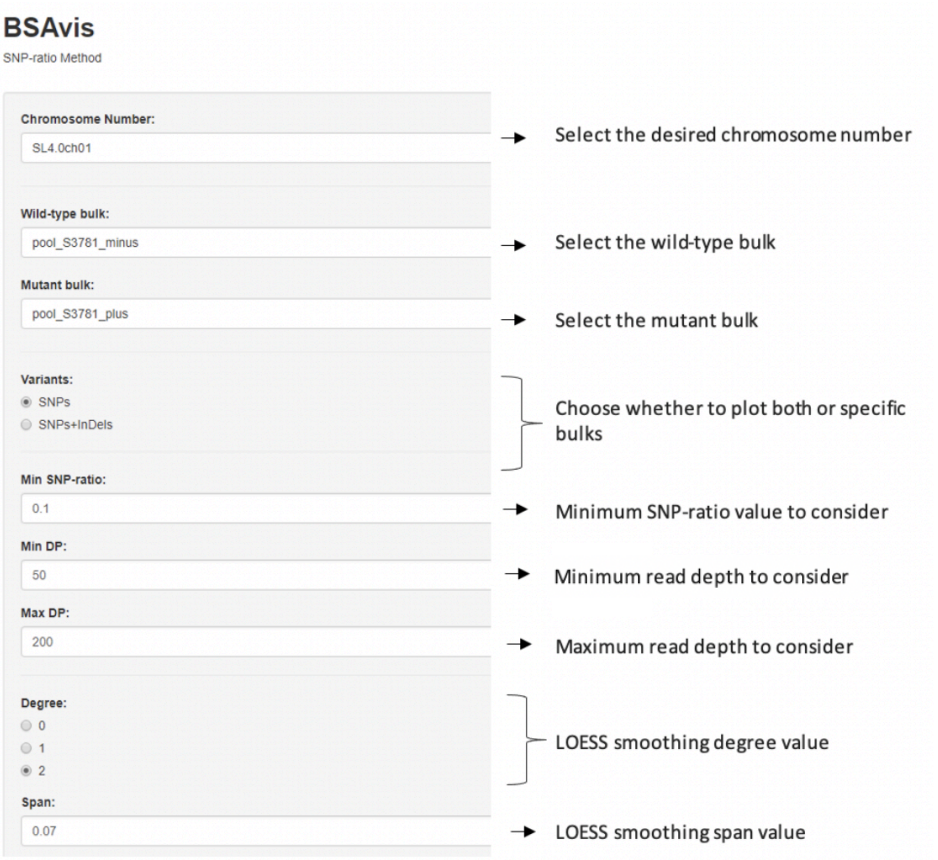

Figure 10. Screenshot of the SNP-ratio method sidebar panel.

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

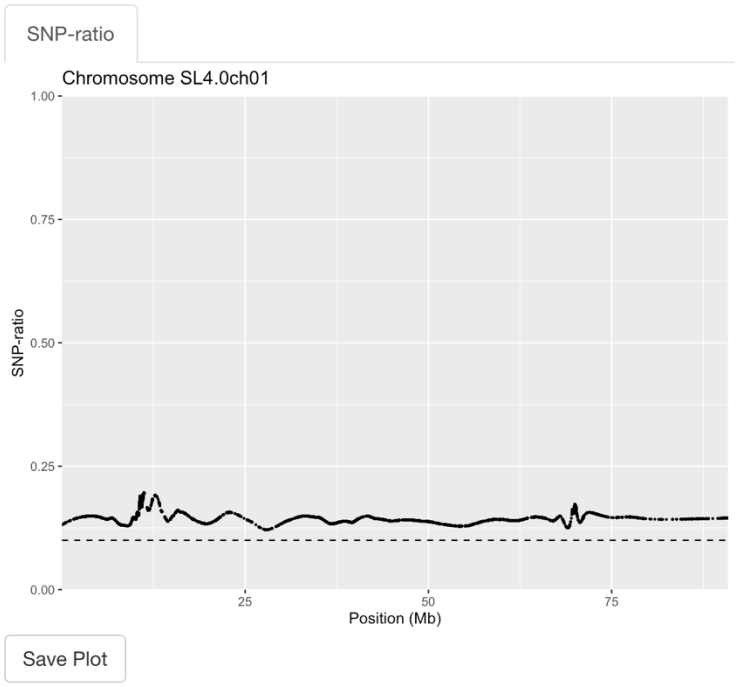

Figure 11. Screenshot of the SNP-ratio plotting panel.

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 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, to save the plot in .TIF format. **(Figure 13)**

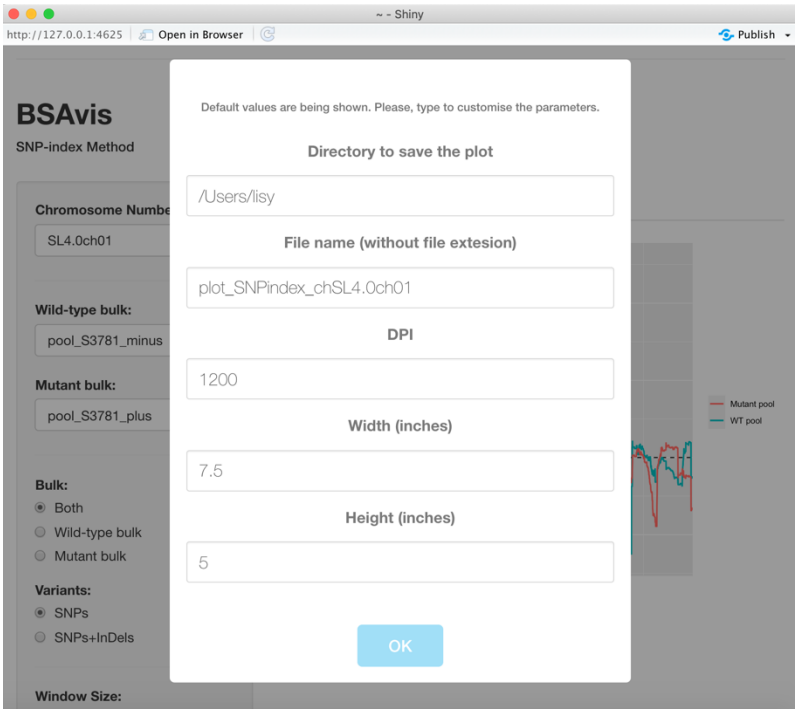

Figure 13. Screenshot of the saving window.

2.4. Zooming Functionality

An additional plotting functionality includes zooming in the plot.

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

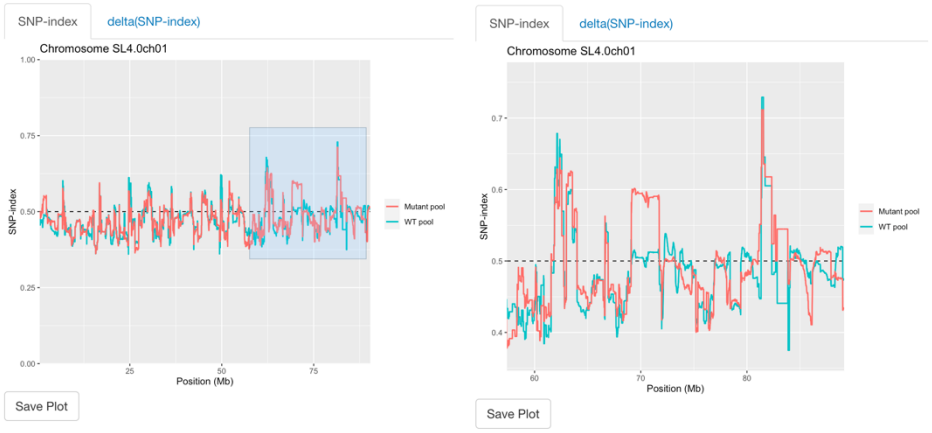

Figure 14. BSAvis screenshot, showing the zooming functionality. The selected area is shown on the left, whereas the zoomed-in area is shown on the right.

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

Glossary

BSA Bulk Segregant Analysis

VCF Variant Calling Format file

RData file format for storing and sharing R workspaces