BSAvis
BSA VISUALISATION TOOL

Technical Documentation

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

1. Introduction

BSAvis is an R package created for identifying genetic loci for phenotypes of interest by applying bulk segregant analysis (BSA) and generating publication-quality plots. Significant peaks in the plots will suggest possible causal regions for the observed phenotype, and therefore, for the underlying mutations.

The only input BSAvis requires is a single Variant Call Format (VCF) file containing variant information (single nucleotide polymorphisms (SNPs) and insertions/deletions (InDels)) of two bulks showing contrasting wild-type and mutant phenotypes. To run BSAvis, it is recommended that the VCF file is generated through Genome Analysis Toolkit version 4 (GATK4) software, having performed joint genotyping to obtain the information from both bulks in a single file.

Alternative variant calling software can be used if the resulting VCF file contains an unfiltered allele depth (AD) column and the data pertaining to variants of both samples (bulks in our case) are contained in the same file, meaning that joint genotyping was performed.

2. Program design

The R package was designed to construct plots by applying one of the two BSA methods currently available in this package (referred to as Δ(SNP-index) and SNP-ratio). This is achieved through a series of connected functions aimed at a specific method (Figure 1).

Moreover, BSAvis was created to be run through R in two different modes: using package functions or as an interactive R-shiny application.

When run in the first mode, the package functions are designed to be sequentially called in the correct order to construct the plot of a chosen BSA method for a specific chromosome.

When using the second mode, the R-shiny application is designed to allow customisation of different parameters using buttons and widgets prior to generating the plots.

3. Program structure

The BSAvis package is composed of several functions that are managed to implement two BSA methods. The functions that make up the package can be classified into those intended to be run by the user and those created to run the interactive R-shiny application.

Within this classification, functions can be sub-classified into individual functions and wrapper functions (Figure 2). The latter type of function, one for each method, calls a series of individual functions in a specific order to apply the chosen BSA method in one step. The functions that form the package are described below.

4. BSAvis package: user accessible functions

4.1. Individual functions

`readBSA_vcf` *Read VCF file and convert it to data frame*

Description

Reads a VCF file and converts it into a data frame. VCF format fields get separated as follows: reference AD, alternate AD, DP, GQ, GT, and GT_alleles.

Usage

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

Arguments

`file`	VCF file containing wild-type and mutant pools as samples. Must include AD column

Return

List containing two elements: character vector with meta information and data frame with variant information.

Details

Using `vcfR` package, a VCF file is read and converted into a data frame containing the following columns: chromosome, position, individual (sample ID in the VCF file), reference AD, alternate AD, DP, GQ, GT, and GT_alleles. The meta information of the VCF file is also extracted. The output is a list containing two elements: meta information and the mentioned data frame.

`calc_SNPindex` *Calculate SNP-index*

Description

Calculates, for both wild-type and mutant bulks, the SNP-index value of each variant recorded in the data frame returned by `readBSA_vcf()`.

Usage

```
vcf_df_SNPindex <- calc_SNPindex(
                       vcf.df = vcf_list$df,
                       wtBulk = "pool_S3781_minus",
                       mBulk = "pool_S3781_plus",
                       variants = "all"
                     )
```

Arguments

`vcf.df`	data frame returned by `readBSA_vcf()`
`wtBulk`	ID of the wild-type bulk
`mBulk`	ID of the mutant bulk
`variants`	variant type to be considered (**default** = `"SNP"`, **allowed**: `"SNP"` or `"all"`)

Return

Data frame containing variant and SNP-index information for each bulk.

Details

The data frame returned by `readBSA_vcf()` is filtered by bulk ID to create two separate data frames: one specific to the wild-type bulk variants information and other one specific to the mutant bulk variants information. For each one of the variants in each data frame, SNP-index is calculated and added as a new column.

In the instance that the type of variants to be considered is set to `"SNP"`, the variants corresponding to InDels are discarded. The rows containing more than three characters in the GT_alleles column (e.g., "T/AT" corresponds to an insertion and contains 4 characters) or containing three characters but one of them being "*", meaning a deletion, are removed from the data frame.

Both data frames are then joined by the chromosome and position fields to ensure that merged rows contain information on the same genomic position. This data frame is returned by the function.

`filter_SNPindex` *Filter variants from SNP-index data frame*

Description

Filters out variants in the data frame returned by `calc_SNPindex()` which do not fall between minimum DP, maximum DP, minimum SNP-index, maximum SNP-index, and minimum GQ for both wild-type and mutant bulks.

Usage

```
vcf_df_SNPindex_filt <- filter_SNPindex(
                           vcf.df.SNPindex=vcf_df_SNPindex,
                           min.SNPindex = 0.25,
                           max.SNPindex = 0.8,
                           min.DP = 60,
                           max.DP = 250,
                           min.GQ = 98
                         )
```

Arguments

`vcf.df.SNPindex`	data frame returned by `calc_SNPindex()`
`min.SNPindex`	min. SNP-index value required for each variant (**default**=`0.3`)
`max.SNPindex`	max SNP-index value required for each variant (**default**=`0.9`)
`min.DP`	min. read depth value required for each variant (**default**=`50`)
`max.DP`	max. read depth value required for each variant (**default**=`200`)
`min.GQ`	min. genotype quality value required for each variant (**default**=`99`)

Return

Data frame containing variant and SNP-index information on variants that passed the filters.

Details

Variants from the data frame returned by `calc_SNPindex()` are filtered out based on criteria which are: minimum DP, maximum DP, minimum SNP-index, maximum SNP-index and minimum GQ.

Variants must pass these filters in both bulks to be retained.

`extract_chrIDs` *Extract chromosome IDs*

Description

Extracts chromosome IDs from the VCF file, based on the meta information contained in the file.

Usage

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

Arguments

`meta`	meta information contained in the first element of the list returned by `readBSA_vcf()`

Return

Character vector containing chromosome IDs from VCF file.

Details

The meta information from the VCF file (which is stored in the first element of the list generated by `readBSA_vcf()`) is taken as input.

The lines in the meta information which contain a sequence of key words (including `ID` and `length`) that make those lines unique, are extracted into a character vector.

For each element of that vector, the characters before and after the chromosome ID are removed, resulting in each element of the vector containing only the characters corresponding to a chromosome ID.

The final character vector contains all chromosomes IDs in the way they are named and ordered within the VCF file.

`slidingWindow` *Apply sliding window to calculate mean SNP-indexes*

Description

Applies the sliding window technique across the length of a specific chromosome to calculate mean SNP-index in each window of a specific size (in base pairs), taking steps of a particular size.

Usage

```
SNPindex_windows <- slidingWindow(
                      meta = vcf_list$meta,
                      chrList = chromList,
                      chrID = "SL4.0ch03",
                      windowSize = 2000000,
                      windowStep = 20000,
                      vcf.df.SNPindex.filt =
                           vcf_df_SNPindex_filt
                    )
```

Arguments

`meta`	meta information contained in the first element of the list returned by `readBSA_vcf()`
`chrList`	all chromosome IDs as returned by `extract_chrIDs()`
`chrID`	ID of chromosome of interest
`windowSize`	window size (**default**=`1000000`)
`windowStep`	window step (**default**=`10000`)
`vcf.df.SNPindex.filt`	filtered SNP-index data frame returned by `filter_SNPindex()`

Return

Data frame containing start, mid and stop positions of each window as well as the corresponding mean SNP-index value for each of the bulks.

Details

Firstly, the length of the chosen chromosome is extracted from the meta information in the VCF file. To this end, the lines from the meta information which contain a sequence of key words (including `ID` and `length`) that make those lines unique from the others, are extracted into a character vector.

For each element of that vector, the characters before and after the chromosome length are removed, so the final character vector contains all chromosomes lengths

in the way they are named and ordered in the VCF file (which matches the order of the elements in `chrList`).

The length of the chosen chromosome is then found by extracting the index of the `chrID` in `chrList` since that index will be equal to the index of the length of the chosen chromosome in the vector of lengths.

Once the length of the chromosome has been extracted, the start, mid and stop positions of each window are calculated across the chromosome length. The stop position of each window is calculated by adding the window size (either the default value or the specified by the user) to the start position of the window. The start position of the first window is 1 and the start position of the following window is calculated by adding the step size to the start position of the previous window.

The windows where the stop position falls past the chromosome length are removed. The start, mid and stop positions of each window in the chromosome are stored in a data frame.

Then, the input data frame returned by `filter_SNPindex()`, is filtered by the chosen chromosome to restrict the data frame to variants specific to that chromosome.

Next, for each window, only the SNP-indexes of the variants comprised between the initial and final position of the window are considered and the mean SNP-index of the variants between those positions is calculated for both wild-type and mutant bulks. The mean SNP-index of both the wild-type bulk and the mutant bulk in each window are added in separate columns to the data frame containing the start, mid and stop positions.

In the case that no variants were found in a specific window, 0.5 will be the value added to the data frame as the mean SNP-index in order to avoid gaps in the plot corresponding to "Not a Number" (NaN) values. The final data frame with the window positions and mean SN-indexes is returned.

`plot_SNPindex` *Plot SNP-index across chromosome positions*

Description

Creates plot showing mean SNP-index values from each bulk against the mid position of the corresponding window in a specific chromosome. Plot can be saved if the `dpi` argument is called.

Usage

```
plot_SNPindex(
      SNPindex.windows = SNPindex_windows,
      chr = 3,
      filename = "SNPindex_ch03",
      path = /setPath,
      dpi = 200,
      width = 20,
      height = 12,
      units = "cm"
)
```

Arguments

`SNPindex.windows`	data frame returned by `slidingWindow()`
`chr`	number of the chromosome to be plotted
`filename`	file name under which the plot will be saved (**default**=`"plot_SNPindex_chX"`)
`path`	path where the file will be saved (**default**=current working directory)
`dpi`	dpi resolution value of saved plot. If no value is given, plots will be generated but not saved
`width`	width value of saved plot (**default**=`5`)
`height`	height value of saved plot (**default**=`7.5`)
`units`	size units (**default**=`"in"`)

Return

`ggplot2` plot / tiff file.

Details

The mean SNP-index from each bulk is plotted in a different colour against the corresponding window mid position in a `ggplot2` line plot.

If the `dpi` argument is not passed, the plot will be shown in the plot panel; however, if a value is added to the `dpi` argument, the graph will be saved in tiff format. Additionally, the name with which to save the file, the directory, the height

and width of the plot and their units can be specified in the arguments. The plot will be saved with default values if different ones are not specified.

`calc_deltaSNPindex` *Calculate Δ(SNP-index)*

Description

Calculates for each variant the Δ(SNP-index) by subtracting the SNP-index value of the wild-type bulk from the SNP-index value of the mutant bulk.

Usage

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

Arguments

`SNPindex.windows`	data frame returned by `slidingWindow()`

Return

Data frame containing start, mid and stop positions of each window in the chromosome as well as the corresponding mean SNP-index value for each bulk and Δ(SNP-index) value.

Details

For each row in the data frame returned by `slidingWindow()`, corresponding to a different window, Δ(SNP-index) is calculated by subtracting the SNP-index value of the reference bulk from the SNP-index value of the mutant bulk. The values of Δ(SNP-index) in each window are added to the data frame as a new column.

`plot_deltaSNPindex` *Plot Δ(SNP-index) across chromosome positions*

Description

Creates plot of Δ(SNP-index) values against the mid position of the corresponding window in a specific chromosome. Plot can be saved if the `dpi` argument is called.

Usage

```
plot_deltaSNPindex(
      deltaSNPindex.windows = deltaSNPindex_windows,
      chr = 3,
      filename = "deltaSNPindex_ch03",
      path = /setPath,
      dpi = 200,
      width = 20,
      height = 12,
      units = "cm"
)
```

Arguments

`deltaSNPindex.windows`	data frame returned by `calc_deltaSNPindex()`
`chr`	number of the chromosome to be plotted
`filename`	file name under which the plot will be saved (**default**=`"plot_deltaSNPindex_chX"`)
`path`	path where the file will be saved (**default**=current working directory)
`dpi`	dpi resolution value of saved plot. If no value is given, plots will be generated but not saved
`width`	width value of saved plot (**default**=`5`)
`height`	height value of saved plot (**default**=`7.5`)
`units`	size units (**default**=`"in"`)

Return

`ggplot2` plot / tiff file.

Details

Δ(SNP-index) values are plotted against the window mid position in a `ggplot2` line plot. If the `dpi` argument is not passed, the plot will be shown in the plot panel; however, if a value is added to the `dpi` argument, the graph will be saved in tiff format. Additionally, the name with which to save the file, the directory, the height and width of the plot and their units can be specified in the arguments. The plot will be saved with default values if different ones are not specified.

`calc_SNPratio` *Calculate SNP-ratio*

Description

Calculates the SNP-ratio between the wild-type and the mutant bulk and adds it as a new column in the data frame returned by `readBSA_vcf()`.

Usage

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

Arguments

`vcf.df`	data frame returned by `readBSA_vcf()`
`wtBulk`	ID of the wild-type bulk
`mBulk`	ID of the mutant bulk
`variants`	variant type to be considered (**default** = `"SNP"`, **allowed**: `"SNP"` or `"all"`)

Return

Data frame containing variant and SNP-ratio information.

Details

The data frame returned by `readBSA_vcf()` is filtered by bulk ID to create two separate data frames: one specific to the wild-type bulk variants information and other one specific to the mutant bulk variants information.

In the instance that the type of variants is set to `"SNP"`, the variants corresponding to InDels are discarded by removing the rows in the data frame containing more than three characters in the GT_alleles column (e.g., "T/AT" corresponds to an insertion and contains 4 characters). This also applies for deletions, where one of the three characters is a "*".

Both data frames (wild-type bulk and mutant bulk) are then joined by the chromosome and position fields. The SNP-ratio is then calculated for each of the variants and the values are added in a new column of the joint data frame, which will be returned by the function.

`filter_SNPratio` *Filter variants from SNP-ratio data frame*

Description

Filters out variants in the data frame returned by `calc_SNPratio()` which do not fall between the minimum SNP-index, minimum DP and maximum DP, for both wild-type and mutant bulks.

Usage

```
vcf_df_SNPratio_filt <- filter_SNPratio(
                           vcf.df.SNPratio = vcf_df_SNPratio,
                           min.SNPratio = 0.3,
                           min.DP = 60,
                           max.DP = 250
                        )
```

Arguments

`vcf.df.SNPratio`	data frame returned by `calc_SNPratio()`
`min.SNPratio`	min. SNP-ratio value required for each variant (**default**=`0.1`)
`min.DP`	min. read depth value required for each variant (**default**=`50`)
`max.DP`	max. read depth value required for each variant (**default**=`200`)

Return

Data frame containing variant information on variants that passed the filters.

Details

Variants in the data frame returned by `calc_SNPratio()` with SNP-ratio values less than the supplied (or default) minimum SNP-ratio value, as well as variants which do not fall between the provided (or default) minimum and maximum DP values in both bulks, are discarded and removed from the final data frame returned by the function.

`plot_SNPratio` *Plot SNP-ratio across chromosome positions*

Description

Creates a plot of SNP-ratio values across positions of a specified chromosome. Plot can be saved if the `dpi` argument is called.

Usage

```
plot_SNPratio(
      vcf.df.SNPratio.filt = vcf_df_SNPratio_filt,
      chrList = chromList,
      chrID = "SL4.0ch03",
      chr = 3,
      min.SNPratio = 0.3,
      degree = 2,
      span = 0.3,
      filename = "SNPratio_ch03",
      path = /setPath,
      dpi = 200,
      width = 20,
      height = 12,
      units = "cm"
)
```

Arguments

`vcf.df.SNPratio.filt`	data frame returned by `filter_SNPratio()`
`chrList`	chromosome IDs as returned by `extract_chrIDs()`
`chrID`	ID of chromosome of interest
`chr`	number of the chromosome to be plotted
`min.SNPratio`	min. SNP-ratio value required for each variant (**default**=`0.1`)
`degree`	degree value for LOESS smoothing (**default**=`2`)
`span`	span value for LOESS smoothing (**default**=`0.07`)
`filename`	file name under which the plot will be saved (**default**=`"plot_SNPratio_chX"`)
`path`	path where the file will be saved (**default**=current working directory)
`dpi`	dpi resolution value of saved plot. If no value is given, plots will be generated but not saved
`width`	width value of saved plot (**default**=`5`)
`height`	height value of saved plot (**default**=`7.5`)
`units`	size units (**default**=`"in"`)

Return

`ggplot2` plot / tiff file.

Details

The data frame returned by `filter_SNPratio()` is filtered by the input chromosome to restrict the data frame to the variants specific to the chosen chromosome.

LOESS regression is then applied to the chromosome positions and SNP-ratio values in order to smooth the resulting line. Degree and span parameters of the LOESS regression can be specified in the function arguments. If no values are specified, the default degree and span values will be used.

The smoothed SNP-ratio values are plotted against the chromosome position in a line plot. A cut-off line is included in the plot (equivalent to the minimum SNP-ratio required).

If the `dpi` argument is not passed, the plot will be shown in the plot panel; however, if a value is added to the `dpi` argument, the graph will be saved in tiff format. Additionally, the name with which to save the file, the directory, the height and width of the plot and their units can be specified in the arguments. The plot will be saved with default values if different ones are not specified.

4.2. Wrapper functions

`SNPindex_plot`	*Call all functions involved in SNP-index plots*

Description

Wrapper function for SNP-index plots (Δ(SNP-index) method). Calls all functions involved in plotting mean SNP-index values from each bulk against the mid position of the corresponding window in a specific chromosome.

Usage

```
SNPindex_plot(
      vcf.list = vcf_list,
      wtBulk = "pool_S3781_minus",
      mBulk = "pool_S3781_plus",
      variants = "all",
      min.SNPindex = 0.25,
      max.SNPindex = 0.8,
      min.DP = 60,
      max.DP = 250,
      min.GQ = 98,
      chrID = "SL4.0ch03",
      chr = 3,
      windowSize = 2000000,
      windowStep = 20000,
      filename = "SNPindex_ch03",
      path = /setPath,
      dpi = 200,
      width = 20,
      height = 12,
      units = "cm"
)
```

Arguments

`vcf.list`	object returned by `readBSA_vcf()`
`wtBulk`	ID of the wild-type bulk
`mBulk`	ID of the mutant bulk
`variants`	variant type to be considered (**default** = `"SNP"`, **allowed**: `"SNP"` or `"all"`)
`min.SNPindex`	min. SNP-index value required for each variant (**default**=`0.3`)
`max.SNPindex`	max. SNP-index value required for each variant (**default**=`0.9`)
`min.DP`	min. read depth value required for each variant (**default**=`50`)
`max.DP`	max. read depth value required for each variant (**default**=`200`)
`min.GQ`	min. genotype quality value required for each variant (**default**=`99`)

`chrID`	ID of chromosome of interest
`chr`	number of the chromosome to be plotted
`windowSize`	window size(**default**=`1000000`)
`windowStep`	window step (**default**=`10000`)
`filename`	file name under which the plot will be saved (**default**=`"plot_SNPindex_chX"`)
`path`	path where the file will be saved (**default**=current working directory)
`dpi`	dpi resolution value of saved plot. If no value is given, plots will be generated but not saved
`width`	width value of saved plot (**default**=`5`)
`height`	height value of saved plot (**default**=`7.5`)
`units`	size units (**default**=`"in"`)

Return

`ggplot2` plot / tiff file.

Details

Wrapper function that sequentially calls in the appropriate order all the functions involved in creating a SNP-index plot: `calc_SNPindex()`, `filter_SNPindex()`, `extract_chrIDs()`, `slidingWindow()` and `plot_SNPindex()`.

The result is a plot showing for each bulk mean SNP-index values against the mid position of the corresponding window in a specific chromosome.

`deltaSNPindex_plot` *Call all functions involved in Δ(SNP-index) plots*

Description

Wrapper function for Δ(SNP-index) method. Calls all functions involved in plotting Δ(SNP-index) values against the mid position of the corresponding window in a specific chromosome.

Usage

```
deltaSNPindex_plot(
      vcf.list = vcf_list,
      wtBulk = "pool_S3781_minus",
      mBulk = "pool_S3781_plus",
      variants = "all",
      min.SNPindex = 0.25,
      max.SNPindex = 0.8,
      min.DP = 60,
      max.DP = 250,
      min.GQ = 98,
      chrID = "SL4.0ch03",
      chr = 3,
      windowSize = 2000000,
      windowStep = 20000,
      filename = "deltaSNPindex_ch03",
      path = /setPath,
      dpi = 200,
      width = 20,
      height = 12,
      units = "cm"
)
```

Arguments

`vcf.list`	object returned by `readBSA_vcf()`
`wtBulk`	ID of the wild-type bulk
`mBulk`	ID of the mutant bulk
`variants`	variant type to be considered (**default** = `"SNP"`, **allowed**: `"SNP"` or `"all"`)
`min.SNPindex`	min. SNP-index value required for each variant (**default**=`0.3`)
`max.SNPindex`	max. SNP-index value required for each variant (**default**=0.9)
`min.DP`	min. read depth value required for each variant (**default**=50)
`max.DP`	max. read depth value required for each variant (**default**=200)
`min.GQ`	min. genotype quality value required for each variant (**default**=`99`)
`chrID`	ID of chromosome of interest

`chr`	number of the chromosome to be plotted
`windowSize`	window size(**default**=`1000000`)
`windowStep`	window step (**default**=`10000`)
`filename`	file name under which the plot will be saved (**default**=`"plot_deltaSNPindex_chX"`)
`path`	path where the file will be saved (**default**=current working directory)
`dpi`	dpi resolution value of saved plot. If no value is given, plots will be generated but not saved
`width`	width value of saved plot (**default**=`5`)
`height`	height value of saved plot (**default**=`7.5`)
`units`	size units (**default**=`"in"`)

Return

`ggplot2` plot / tiff file.

Details

Wrapper function for the Δ(SNP-index) method which sequentially calls in the appropriate order all the functions involved in this method: `calc_SNPindex()`, `filter_SNPindex()`, `extract_chrIDs()`, `slidingWindow()`, `calc_deltaSNPindex()` and `plot_deltaSNPindex()`.

The result is a plot showing Δ(SNP-index) against the mid position of the corresponding window in a specific chromosome.

`SNPratio_plot` *Call all functions involved in SNP-ratio plots*

Description

Wrapper function for SNP-ratio method. Calls all functions involved in plotting SNP-ratio values across positions of a specific chromosome.

Usage

```
SNPratio_plot(
      vcf.list = vcf_list,
      wtBulk = "pool_S3781_minus",
      mBulk = "pool_S3781_plus",
      variants = "all",
      min.SNPratio = 0.25,
      min.DP = 60,
      max.DP = 250,
      chrID = "SL4.0ch03",
      chr = 3,
      degree = 2,
      span = 0.3,
      filename = "SNPratio_ch03",
      path = /setPath,
      dpi = 200,
      width = 20,
      height = 12,
      units = "cm"
)
```

Arguments

`vcf.list`	object returned by `readBSA_vcf()`
`wtBulk`	ID of the wild-type bulk
`mBulk`	ID of the mutant bulk
`variants`	variant type to be considered. **Default** = `"SNP"` (allowed: `"SNP"` or `"all"`)
`min.SNPratio`	min. SNP-ratio value required for each variant (**default**=`0.1`)
`min.DP`	min. read depth value required for each variant (**default**=`50`)
`max.DP`	max. read depth value required for each variant (**default**=`200`)
`chrID`	ID of chromosome of interest
`chr`	number of the chromosome to be plotted
`degree`	degree value for LOESS smoothing (**default**=`2`)
`span`	span value for LOESS smoothing (**default**=`0.07`)
`filename`	file name under which the plot will be saved (**default**=`"plot_SNPratio_chX"`)

`path`	path where the file will be saved (**default**=current working directory)
`dpi`	dpi resolution value of saved plot. If no value is given, plots will be generated but not saved
`width`	width value of saved plot (**default**=`5`)
`height`	height value of saved plot (**default**=`7.5`)
`units`	size units (**default**=`"in"`)

Return

`ggplot2` plot / tiff file.

Details

Wrapper function to sequentially call in the appropriate order all the functions involved in the SNP-ratio method: `calc_SNPratio()`, `filter_SNPratio()`, `extract_chrIDs()` and `plot_SNPratio()`.

The result is a plot showing SNP-ratio between both bulks (wild-type and mutant) across chromosome positions.

5. BSAvis package: R-shiny application underlying functions

The following functions were specifically created for R-shiny purposes and thus to be called by `BSAvis_shiny()` and not by the user.

5.1. Specific plot functions to R-shiny

`shinyPlot_SNPindex`	*R-shiny: Plot SNP-index across chromosome positions* ***(not intended to be run by the user)***

Description

Creates plot specific to R-shiny application showing for each bulk (or just one of them) the mean SNP-index versus the mid position of the corresponding window in a specific chromosome.

Usage

Function not intended to be run by the user.

Arguments

`vcf.df.window.SNPindex`	data frame returned by `slidingWindow()`
`chr`	ID of the chromosome to be plotted
`bulk`	bulk whose SNP-index to plot (`0` for both bulks, `1` for wild-type bulk and `2` for mutant bulk)
`ranges`	range for x-axis and y-axis

Return

`ggplot2` plot

Details

This function is a variant of the `plot_SNPindex()` function created to meet the needs of the R-shiny application.

Firstly, a `ggplot` object is created with all necessary elements in the plot except the lines, each for one bulk, corresponding to mean SNP-index against the mid position of the corresponding window in a specific chromosome.

Based on the value of the `bulk` argument, the appropriate SNP-index line(s) will be added to the previously created `ggplot` object. If `bulk=0` two lines corresponding to the SNP-indexes of both wild-type and mutant bulks will be added, while if `bulk=1` or `bulk=2` a single line will be added to the plot

corresponding to the wild-type bulk SNP-indexes or mutant bulk SNP-indexes, respectively.

The value of the `bulk` argument will be set inside the R-shiny application function (`BSAvis_shiny()`) depending on the user input.

Additionally, this function includes the `ranges` argument that allows setting the limits of the x and y-axis to show a zoomed area of the SNP-index plot when that area is selected to be enlarged from the R-shiny application. If no area is selected, the x and y axis limits will be set to null, showing the entire plot.

This function does not include any arguments related to plot saving because this functionality is accomplished with a specific button in the R-shiny application.

`shinyPlot_deltaSNPindex`	*R-shiny: Plot Δ(SNP-index) across chromosome positions* ***(not intended to be run by the user)***

Description

Creates plot specific to R-shiny application showing Δ(SNP-index) versus the mid position of the corresponding window in a specific chromosome.

Usage

Function not intended to be run by the user.

Arguments

`vcf.df.window.delta`	data frame returned by `calc_deltaSNPindex()`
`chr`	ID of the chromosome to be plotted
`ranges`	range for x-axis and y-axis

Return

`ggplot2` plot

Details

This function is a variant of the `plot_deltaSNPindex()` function created to meet the needs of the R-shiny application. The difference with the original is that it includes the `ranges` argument that allows setting the limits of the x and y-axis to show a zoomed area of the Δ(SNP-index) plot when that area is selected to be enlarged from the R-shiny application. If no area is selected, the x and y-axis limits will be set to null, showing the entire plot.

This function does not include any arguments related to plot saving because this functionality is accomplished with a specific button in the R-shiny application.

shinyPlot_SNPratio	*R-shiny: Plot SNP-ratio across chromosome positions* ***(not intended to be run by the user)***

Description

Creates plot specific to R-shiny application showing SNP-ratio versus position in a specific chromosome.

Usage

Function not intended to be run by the user.

Arguments

`vcf.df.SNPratio.filt`	data frame returned by `calc_deltaSNPindex()`
`chrList`	chromosome IDs as returned by `extract_chrIDs()`
`chrID`	ID of chromosome of interest
`chr`	number of the chromosome to be plotted
`min.SNPratio`	min. SNP-ratio value required for each variant (**default**=`0.1`)
`degree`	degree value for LOESS smoothing (**default**=`2`)
`span`	span value for LOESS smoothing (**default**=`0.07`)
`ranges`	range for x-axis and y-axis

Return

`ggplot2` plot

Details

This function is a variant of the `plot_SNPratio()` function created to meet the needs of the R-shiny application. The difference with the original is that it includes the `ranges` argument that allows setting the limits of the x and y-axis to show a zoomed area of the SNP-ratio plot when that area is selected to be enlarged from the R-shiny application. If no area is selected, the x and y-axis limits will be set to null, showing the entire plot.

This function does not include any arguments related to plot saving because this functionality is accomplished with a specific button in the R-shiny application.

5.2. Specific wrapper functions to R-shiny

`shiny_SNPindex`	*R-shiny: Call all functions involved in SNP-index plots* ***(not intended to be run by the user)***

Description

Wrapper function for SNP-index plots (Δ(SNP-index) method) specific to R-shiny application. Calls all functions involved in plotting mean SNP-index values from both bulks (or just one of them) against the mid position of the corresponding window in a specific chromosome.

Usage

Function not intended to be run by the user.

Arguments

`vcf.list`	object returned by `readBSA_vcf()`
`wtBulk`	ID of the wild-type bulk
`mBulk`	ID of the mutant bulk
`variants`	variant type to be considered (allowed: `"SNP"` or `"all"`)
`min.SNPindex`	min. SNP-index value required for each variant
`max.SNPindex`	max. SNP-index value required for each variant
`min.DP`	min. read depth value required for each variant
`max.DP`	max. read depth value required for each variant
`min.GQ`	min. genotype quality value required for each variant
`chrID`	ID of chromosome of interest
`chr`	number of the chromosome to be plotted
`windowSize`	window size
`windowStep`	window step
`bulk`	bulk whose SNP-index to plot (`0` for both bulks, `1` for wild-type bulk and `2` for mutant bulk)
`ranges`	range for x-axis and y-axis

Return

`ggplot2` plot

Details

Wrapper function for creating SNP-index plots in the R-shiny application. The functions involved are sequentially called in the appropriate order:

`calc_SNPindex()`,`filter_SNPindex()`,`extract_chrIDs()`,`slidingWindow()` and `shinyPlot_SNPindex()`.

The function was created to be uniquely used by `BSAvis_shiny()`.

shiny_deltaSNPindex	*R-shiny: Call all functions involved in Δ(SNP-index) plots* ***(not intended to be run by the user)***

Description

Wrapper function for Δ(SNP-index) method) specific to R-shiny application. Calls all functions involved in plotting Δ(SNP-index) values against the mid position of the corresponding window in a specific chromosome.

Usage

Function not intended to be run by the user.

Arguments

`vcf.list`	object returned by `readBSA_vcf()`
`wtBulk`	ID of the wild-type bulk
`mBulk`	ID of the mutant bulk
`variants`	variant type to be considered (allowed: `"SNP"` or `"all"`)
`min.SNPindex`	min. SNP-index value required for each variant
`max.SNPindex`	max. SNP-index value required for each variant
`min.DP`	min. read depth value required for each variant
`max.DP`	max. read depth value required for each variant
`min.GQ`	min. genotype quality value required for each variant
`chrID`	ID of chromosome of interest
`Chr`	number of the chromosome to be plotted
`windowSize`	window size
`windowStep`	window step
`ranges`	range for x-axis and y-axis

Return

`ggplot2` plot

Details

Wrapper function for creating Δ(SNP-index) plots in the R-shiny application. The functions involved are sequentially called in the appropriate order: `calc_SNPindex()`, `filter_SNPindex()`, `extract_chrIDs()`, `slidingWindow()`, `calc_deltaSNPindex()` and `shinyPlot_deltaSNPindex()`.

The function was created to be uniquely used by `BSAvis_shiny()`.

shiny_SNPratio	*R-shiny: Call all functions involved in SNP-ratio plots* ***(not intended to be run by the user)***

Description

Wrapper function for SNP-ratio method specific to R-shiny application. Calls all functions involved in plotting SNP-ratio values across positions of a specific chromosome.

Usage

Function not intended to be run by the user.

Arguments

vcf.list	object returned by readBSA_vcf()
wtBulk	ID of the wild-type bulk
mBulk	ID of the mutant bulk
variants	variant type to be considered (allowed: "SNP" or "all")
min.SNPratio	min. SNP-ratio value required for each variant
min.DP	min. read depth value required for each variant
max.DP	max. read depth value required for each variant
chrID	ID of chromosome of interest
chr	number of the chromosome to be plotted
degree	degree value for LOESS smoothing
span	span value for LOESS smoothing
ranges	range for x-axis and y-axis

Return

ggplot2 plot

Details

Wrapper function for creating SNP-index plots in the R-shiny application. The functions involved are sequentially called in the appropriate order: calc_SNPratio(), filter_SNPratio(), extract_chrIDs() and shiny Plot_SNPratio().

The function was created to be uniquely used by BSAvis_shiny().

5.3. R-shiny function – user accessible

`BSAvis_shiny`	*Launch BSAvis R-shiny application*

Description

Launches R-shiny application to create and save SNP-index, Δ(SNP-index) and SNP-ratio plots while exploring and customising the different parameters involved.

Usage

```
BSAvis_shiny(vcf.list = vcf_list)
```

Arguments

`vcf.list`	Object returned by `readBSA_vcf()`

Return

R-shiny application

Details

This function was implemented to launch a R-shiny application allowing the user to make an interactive use of the BSAvis package.

The body of the `BSAvis_shiny()` function is composed of two elements: the user interface (ui) and the server. The ui code adds buttons, widgets and plots panels in order to build a single screen with the upper part intended to show the plots related to the Δ(SNP-index) method and the widgets to modify it, while the lower part showing the SNP-ratio method plot and the widgets to customise it.

The server's code makes use of reactive expressions and calls to the `shiny_SNPindex()`, `shiny_deltaSNPindex()` and `shiny_SNPratio()` functions to rebuild and display the plots based on the parameters value selected by the user through the widgets. This part also includes the code that allows saving the plots by using the appropriate button which launches a save configuration window. Furthermore, the plots are rebuilt in the event that the user selects an area of the plot with the mouse and double-clicks to zoom in on that area.

6. Figures

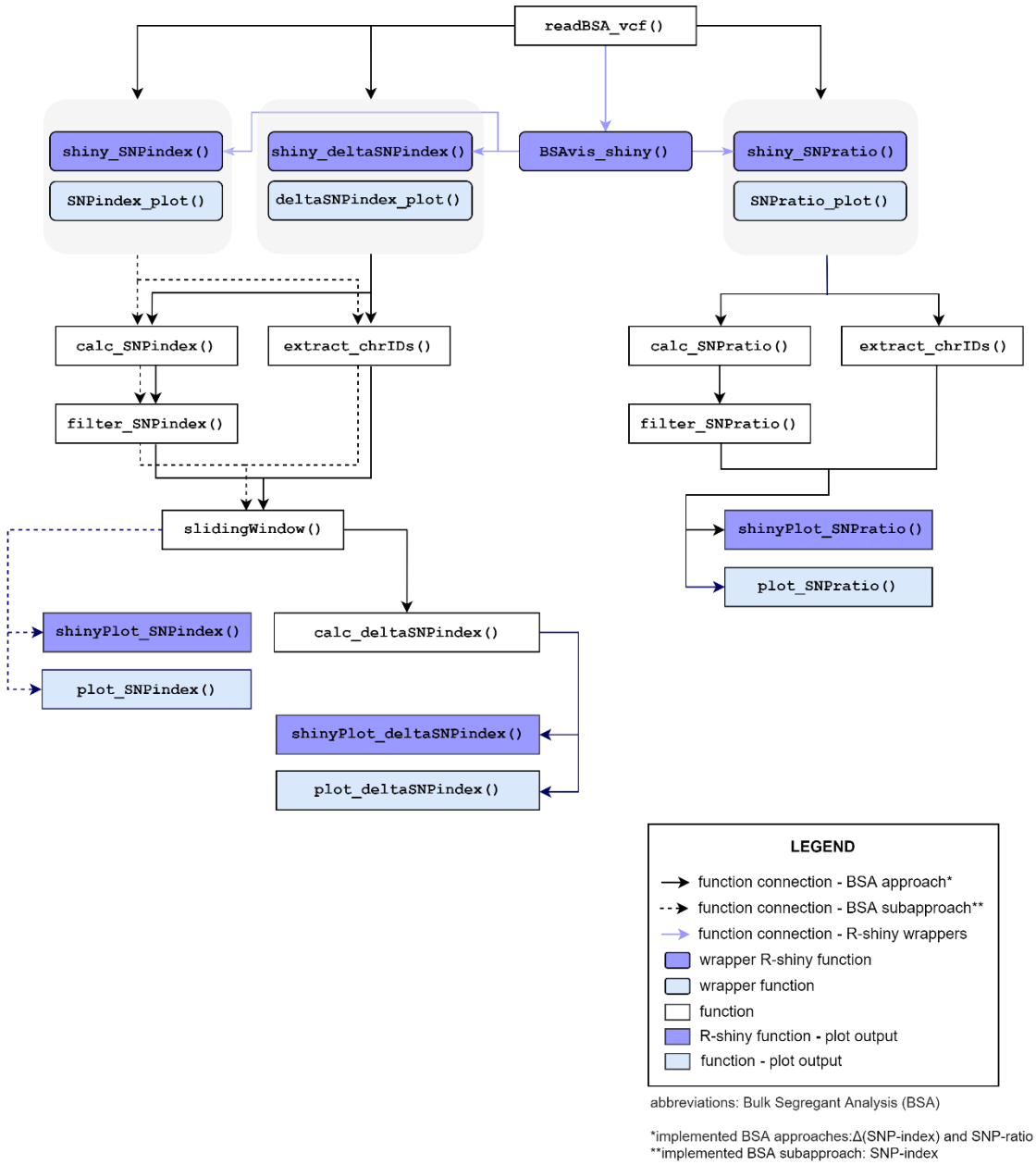

Figure 1. Connection between BSAvis functions. All the functions that make up the BSAvis package are shown here. The connections between them are indicated by arrows, showing how outputs become inputs of functions called later in the process.

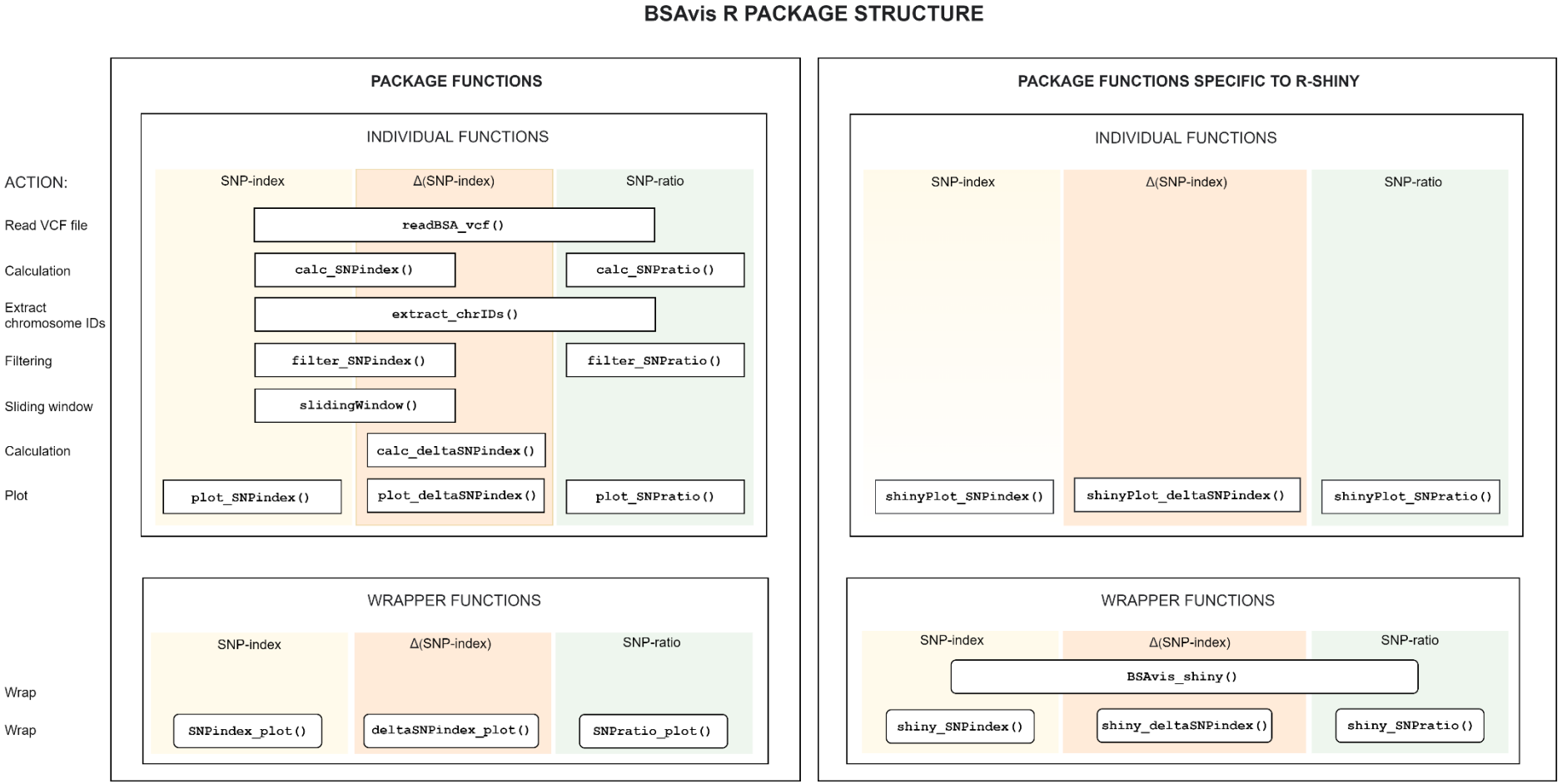

Figure 2. Classification of BSAvis functions. The classification and sub-classifications of each of the functions that make up the BSAvis package are shown above. The actions carried out by the functions are indicated on the left.