The Shiny Variant Explorer

Kévin Rue-Albrecht *[1,2]

[1]Department of Medicine, Imperial College London, UK
[2]Nuffield Department of Medicine, University of Oxford, UK

*kevinrue67@gmail.com

2 May 2019

Abstract

Shiny web-application that demonstrates the functionalities of the *TVTB* package integrated in a programming-free environment.

Package

BiocStyle 2.12.0

Contents

1 Preliminary notes

The Shiny Variant Explorer (*tSVE*) was primarily developped to demonstrate features implemented in the *TVTB*, **not** as a production environment. As a result, a few important considerations should be made to clarify what should and should **not** be expected from the web-application:

- Bug fixes will be treated with a much lower priority relative to those related to package methods.
- It is technically not feasible to offer in a web-interface the same degree of flexibility as the command-line environment (*e.g.* `...`[1]).
- Greater control over the input and output data is possible at the command-line (*e.g.* refinement of `ggplot` objects, definition of custom genomic ranges).
- Requests for new features should apply to the package first. Only features relevant to the package functionalities may be made available in the web-application.
- Figures (`ggplot`) are currently the only output that can be exported from the web-application (using the web browser "Download image", or equivalent context menu item). In the future, action buttons may be added to export tables (*e.g.* CSV format) and figures (*e.g.* PDF format).
- This vignette is largely static as the web-application may only be used in an interactive session.
- First-time users are encouraged to follow this vignette sequentially (*i.e* in order, without skipping sections), as it takes readers through the sequence of actions of a typical analysis.
  - This vignette was designed to be read beside an open *R* session with the *TVTB* package installed, so that users may follow the instructions marked by the word **Action** and bulleted points in the following sections.

[1] The `...` argument is called "ellipsis".

2 Pre-requisites

The *Shiny Variant Explorer* suggests a few additional package dependencies compared to the package, to support certain forms of data input and display.

Input

- The *ensembldb* package and relevant `EnsDb`[2] annotation packages are required if that interface is used to query genomic ranges (demonstrated in this section).
- The *EnsDb.Hsapiens.v75* is required to query genomic ranges associated by gene names for the demonstration data[3].
- The *rtracklayer* package is required if a BED file is used to provide genomic ranges (demonstrated in this section).

Display

- The latest version of the *DT* package is recommended to benefit from the latest developments (*e.g.* column filters inactivated if a single value exist in that column; version $>= 0.2.2$).
- The *shiny* package is required for all Shiny web-applications.

[2] In the future, the web-application may also support `TxDb` and `OrganismDb` annotation packages.

[3] In the future, the web-application may also use annotation packages to facet statistics and figures by genomic range(s).

3 Launching the Shiny Variant Explorer

The `TVTB::tSVE()` method launches the web-application.

4 Overall layout of the web-application

Overall, the web-application is implemented as a web-page with a top level navigation bar organised from left to right to reflect progression through a typical analysis, with the exception of the last two menu items **Settings** and **Session**, which may be useful to check and update at any point.

Here is a brief overview of the menu items:

- **Input**
  - Control which samples, phenotypes, genomic ranges, and VCF fields must be imported.
  - An `EnsDb` annotation package may be selected to use the associated database interface.
- **Frequencies**
  - Add and remove INFO fields that contain calculated genotype counts and allele frequencies.
  - Add and remove genotype counts and allele frequencies across all samples, or within individual phenotype levels.
- **Filters**
  - Define and apply *VCF filter rules* (detailed in a separate vignette).
- **Views**
  - Display and examine major objects of the analysis and their slots.
- **Plots**
  - Display data plots and associated data tables.
- **Settings**
  - Control advanced parameters of the analysis and web-application.
- **Session**
  - Display session information and other relevant information.

5 Input panel

The **Input** panel controls the major input parameters of the analysis, including phenotypes (and therefore samples), genomic ranges, and fields to import from VCF file(s). Those inputs are useful to import only data of interest, as well as to limit memory usage and duration of calculations.

5.1 Phenotypes

Phenotypes are critical to define groups of samples that may be compared in summary statistics, tables, and plots. Moreover, phenotypes also implicitely define the set of samples required in the analysis (unique sample identifiers usually set as `rownames` of the phenotypes).

The web-application accepts phenotypes stored in a text file, with the following requirements:

- Fields must be delimited by "white space" (default separator for the `read.table` function).
- The first column of the file must contain unique sample identifiers, as syntactically valid `rownames`.
- The first row of the file must phenotype names, as syntactically valid `colnames`.

When provided, phenotypes will be used to import from VCF file(s) only genotypes for the corresponding samples identifiers. Moreover, an error message will be displayed if any of the sample identifiers present in the phenotypes is absent from the VCF file(s).

Note that the web-application does not *absolutely* require phenotype information. In the absence of phenotype information, all samples are imported from VCF file(s).

Action:

- Click on the *Browse* action button
- Navigate to the `extdata` folder of the *TVTB* installation directory
- Select the file `integrated_samples.txt`

Alternatively: click the *Sample file* button

Notes

- The *TVTB* installation directory can be identified using the following command in an *R* session:

```
system.file("extdata", package = "TVTB")
```

- The file selection pop-up window that is open by the action button browses files on the server side. *This point is only relevant if the package/web-application is run on a remote server.*

5.2 Genomic ranges

Genomic ranges are critical to import only variants in targeted genomic regions or features (*e.g.* genes, transcripts, exons), as well as to limit memory usage and duration of calculations.

The Shiny Variant Explorer currently supports three types of input to define genomic ranges:

- BED file
- UCSC-style text input
- `EnsDb` annotation packages

Currently, the web-application uses genomic ranges solely to query the corresponding variants from VCF file(s). In the future, those genomic ranges may also be used to produce faceted summary statistics and plots.

Notes:

- The web-application does not *absolutely* require genomic ranges. In the absence of genomic ranges, all variants are imported from VCF file(s). *Caution recommended with large files!*
- When VCF file(s) are parsed (in a later section), only the genomic ranges from the currently selected input mode are
- The active genomic ranges are only taken from the currently selected input mode

5.2.1 BED file

If a BED file is supplied, the web-application parses it using the *rtracklayer* `import.bed` method. Therefore the file must respect the BED file format guidelines.

Action:

- Click on the *Browse* action button
- Navigate to the `extdata` folder of the *TVTB* installation directory
- Select the file `SLC24A5.bed`

Alternatively: click the *Sample file* button

Notes:

- The BED file defines the same genomic range that was used to extract variant from the 1000 Genomes Project Phase 3 release VCF file used in this vignette.
- The file selection window that is open by the action button browses files on the server side (see Phenotypes section above).

5.2.2 UCSC format

Sequence names (*i.e.* chromosomes), start, and end positions of one or more genomic ranges may be defined in the text field, with individual regions separated by `";"`.

Action:

- Paste `15:48,413,169-48,434,869` in the text field

Alternatively: click the *Sample input* button

Notes:

- The web-application automatically trims `","` characters from the text input, before coercing the start and end positions to `numeric`
- Multiple genomic ranges may be supplied (*e.g.* `1:123-456;2:234-345;2:456-789`)

5.2.3 Ensembl-based annotation packages

Currently, genomic ranges encoding only gene-coding regions may be retrieved from an Ensembl-based database. This feature was adapted from the web-application implemented in the *ensembldb* package.

Comment: In the future, the interface to query transcripts and exons annotations may be added to the web-application.

Action:

- Paste `SLC24A5` in the text field

Alternatively: click the *Sample input* button

FixMe: Genomic feature located on contigs may cause problems when working with one VCF per chromosome. In the future, an option may be added to ignore contigs.

5.3 Variants

At the core of the *TVTB* package, variants must be imported from one or more VCF file(s) annotated by the Ensembl Variant Effect Predictor (VEP) script (McLaren et al. 2010).

Considering the large size of most VCF file(s), it is common practice to split genetic variants into multiple files, each file used to store variants located on a single chromosome (more generally; a single sequence). The Shiny Variant Explorer supports two situations:

- All variants are stored in a single VCF file ("Single-VCF" mode).
- Variants are split into one file per sequence, with the requirement that files be named with a pattern including the sequence name (must match the `seqnames` slot of the genomic ranges described above ("Multi-VCF mode").

In addition, VCF files can store a plethora of information in their various fields. It is often useful to select only a subset of fields relevant for a particular analysis, to limit memory usage. The web-application uses the *VariantAnnotation* `scanVcfHeader` to parse the header of the VCF file (*Single-VCF* mode) or the first VCF file (*Multi-VCF* mode), to display the list of available fields that users may choose to import. A few considerations must be made:

- The web-application requires that Ensembl VEP predictions be present in the INFO field.
- The web-application requires that the `"GT"` key be present in the FORMAT field.

5.3.1 Single-VCF mode

This mode display an action button that must be used to select the VCF file from which to import variants.

Action:

- Click on the *Browse* action button
- Navigate to the `extdata` folder of the *TVTB* installation directory
- Select the file `chr15.phase3_integrated.vcf.gz`

Alternatively: click the *Sample file* button

5.3.2 Multi-VCF mode

This mode requires two pieces of information:

- The path to the folder that contains one or more VCF file(s).
- The naming pattern of VCF file(s), with the following requirement:
  - The pattern must include `"%s"` to declare the emplacement of the sequence (*i.e.* chromosome) name in the pattern.

Note that a summary of VCF file(s) detected using the given the folder and pattern is displayed on the right, to help users determine whether the parameters are correct. In addition, the content of the given folder is displayed at the bottom of the page, beside the same content filtered for the VCF file naming pattern.

Action:

None. The text fields should already be filled with default values, pointing to the single example VCF file (`chr15.phase3_integrated.vcf.gz`).

5.3.3 VCF scan parameters

This panel allows users to select the INFO and FORMAT fields to import (in the `info` and `geno` slots of the `VCF` object, respectively).

It is important to note that the FORMAT/GT and INFO/ fields—where `<vep>` stands for the INFO key where Ensembl VEP predictions are stored—are implicitely imported from the VCF. Similarly, the mandatory FIXED fields `CHROM`, `POS`, `ID`, `REF`, `ALT`, `QUAL`, and `FILTER` are automatically imported to populate the `rowRanges` slot of the `VCF` object.

Action:

- Click the *Deselect all* action button under the *INFO fields* selection input to import only the INFO/CSQ and FORMAT/GT fields.
- Click the *Import variants* action button

A summary of variants, phenotypes, and samples imported will appear beside the action button.

5.4 Annotations

This panel allows users to select a pre-installed annotation package. Currently, only `EnsDb` annotation packages are supported, and only **gene**-coding regions may be queried.

Action:

- If none of the `EnsDb` packages are installed, it will simply **not** be possible to use the `ensembl` interface of the *Genomic ranges* input tab.
- If the `EnsDb.Hsapiens.v75` package is the only `EnsDb` packages installed, no action is required; the package should already be pre-selected.
- If the `EnsDb.Hsapiens.v75` package is **not** the only `EnsDb` packages installed, users should select it in the list of choices.

6 Frequencies panel

This panel demonstrates the use of three methods implemented in the *TVTB* package, namely `addFrequencies`, `addOverallFrequencies`, and `addPhenoLevelFrequencies`.

6.1 Overall frequencies

This panel allows users to *Add* and *Remove* INFO fields that contain genotype counts (*i.e.* homozygote reference, heterozygote, homozygote alternate) and allele frequencies (*i.e.* alternate allele frequency, minor allele frequency) calculated across all the samples and variants imported. The web-application uses the homozygote reference, heterozygote, and homozygote alternate genotypes defined in the Advanced settings panel.

Importantly, the name of the INFO keys that are used to store the calculated values can be defined in the Advanced settings panel.

Action:

- Click the *Add* action button

- See the *Latest changes* message update at the top of the screen.
- Optionally, the Views panel can be used to examine the new fields

6.2 Phenotype-level frequencies

This panel allows users to *Refresh* the list of INFO fields that contain genotype counts and allele frequencies calculated within *groups of samples* associated with various levels of a given phenotype.

Action:

- Select `super_pop` in the list of phenotypes
- Click the *Select all* action button
- Click the *Refresh* action button
- See the *Latest changes* message update at the top of the screen.
- Optionally, the Views panel can be used to examine the new fields

7 Filters panel

One of the flagship features of the *TVTB* package are the *VCF filter rules*, extending the *S4Vectors* `FilterRules` class to new classes of filter rules that can be evaluated within environments defined by the various slots of `VCF` objects.

Generally speaking, `FilterRules` greatly facilitate the design and combination of powerful filter rules for table-like objects, such as the `fixed` and `info` slots of *VariantAnnotation* `VCF` objects, as well as Ensembl VEP predictions stored in the meta-columns of `GRanges` returned by the *ensemblVEP* `parseCSQToGRanges` method.

A separate vignette describes in greater detail the use of classes that contain *VCF filter rules*. A simple example is shown below.

Action:

- Select `VEP` as the *Type* of filter
- Paste `grepl("missense",Consequence)` in the text field
- Leave the *Active?* checkbox ticked
- Click the *Add filter* action button
- See the list of rules update at the bottom of the screen
- Click the *Apply filters* action button
- See the summary of filtered variants update beside the action button
- Optionally, the Views panel can be used to examine the new fields

Alternatively: click the *Sample input* button

8 Views panel

This panel offers the chance to examine the main objects of the session, namely:

- The active genomic ranges
- The `rowRanges` and selected meta-columns of the filtered variants.

- Selected field of the `info` slot (of the filtered variants).
- Selected Ensembl VEP predictions (of the filtered variants).
- Selected phenotypes attached to the variants.
- Subset of genotypes (among the filtered variants).
  - Genotypes for all filtered variants may be displayed as a heatmap (`ggplot`).

Action:

- In the various panels, select fields to examine each object
  - In particular, note the INFO fields that contain genotype counts and allele frequencies calculated earlier
- Go to the *Heatmap* tab of the *Genotypes* panel
- Click the *Go!* action button to calculate and display the heatmap

9 Plots panel

This panel demonstrates the use of two methods implemented in the *TVTB* package, namely `tabulateVepByPhenotype` and `densityVepByPhenotype`.

10 Settings panel

This panel stores more advanced settings that users may not need to edit as frequently, if at all. Those settings are divided in two sub-panels:

- **Advanced**
  - Genotypes, INFO key suffixes, and VCF yield size
- **Parallel**
  - Use of multiple CPUs to accelerate calculations

10.1 Advanced settings

10.1.1 Genotypes

It is critical to accurately identify and define how the different genotypes—homozygote reference, heterozygote, and homozygote alternate—are encoded in the VCF file, to produce accurate genotypes counts and frequencies, for instance. This generally requires examining the content of the FORMAT/GT field outside of the web-application. For instance, the functions `unique` and `table` may be used to identify (and count) all the distinct genotype codes in the `geno` slot (`"GT"` key) of a `VCF` object.

The default selected values are immediately compatible with the demonstration data set. Users who wish to select genotypes codes not yet available among the current choices may either contact the package maintainer to add them in a future release, or edit the Global configuration file of the web-application locally.

10.1.2 INFO key suffixes

Currently, the three calculated genotypes counts and two allele frequencies require five INFO fields to store their respective values.

Considering that *TVTB* offers the possibility to calculate counts and frequencies for the overall data set, and for each level of each phenotype, it is important to define a clear and consistent naming mechanism that does not conflict with INFO keys imported from the VCF file(s). In the *TVTB* package, a suffix is required for each type of genotype and frequency calculated, to generate INFO as follows:

- Overall counts and frequencies are stored in INFO keys named `<suffix>`
- Counts and frequencies calculated for individual levels of selected phenotypes are stored under INFO keys formed as `<phenotype>_<level>_<suffix>`

Again, the default values are immediately compatible with the demonstration data set. For other data sets, it may be necessary to change those values, either by preference, or to avoid conflict with INFO keys imported from the VCF file(s).

10.1.3 Miscellaneous settings

Other rarely used settings in this panel include:

- VCF yield size
  - Only applicable when VCF file(s) are parsed without defined genomic ranges. See the *Rsamtools* documentation.

10.2 Parallel settings

Several functionalities of the *TVTB* package are applied to independent subsets of data (*e.g.* counting genotypes in various levels of a given phenotype). Such processes can benefit from multi-threaded calculations. Multi-threading settings in the Shiny web-application are somewhat experimental, as they have been validated only on a small set of operating systems, while some issues have been reported for others.

Report	Operating System	Cluster Class	Cluster type	# Cores
OK	Ubuntu 14.04	Multicore	FORK	2
OK	Scientific Linux 6.7	Multicore	FORK	2
Hang[1]	OS X El Capitan	Snow	SOCK	2

1. Application hangs while CPUs work infinitely at full capacity.

Comment: Users are welcome to send feedback to report additional successful configuration, as well as newly identified issues.

11 Session information

The last panel of the Shiny Variant Explorer offers detailed views of objects and settings in the current session, including:

- **Session info**
  - The `sessionInfo()` value
- **TVTB settings**
  - See the vignette *Introduction to TVTB* for more information
- **General settings**
  - Including the current value of various input widgets
- **Advanced settings**
  - including the current value of more input widgets
- **BED**
  - Structural view of the active genomic ranges
- **Variants**
  - Overview of the raw `VCF` object
- **VEP**
  - Structural view of the `GRanges` that store the Ensembl VEP predictions
- **Phenotypes**
  - Structural view of the phenotype information attached to the variants
- **Genotypes**
  - Structural view of the `geno` slot (`"GT"` key) of the raw variants

12 Global configuration

Most default values are stored in the `global.R` file of the web-application. All the files of the web-application are stored in the `extdata/shinyApp` folder of the *TVTB* installation directory (see an earlier section to identify this directory).

Users who wish to change the default values of certain input widgets (*e.g.* genotype codes) may edit the `global.R` file accordingly. However, the file will be reset at each package update.

Comment: In the future, a mechanism may be implemented to override global settings locally, without risk of seeeing this custom configuration overwritten at the next package update (e.g. a file in the user home folder that would be parsed to overwrite certain settings).

13 Vignette session

Here is the output of `sessionInfo()` on the system on which this document was compiled:

```
sessionInfo()
## R version 3.6.0 (2019-04-26)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: OS X El Capitan 10.11.6
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
```

```
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] TVTB_1.10.0      knitr_1.22       BiocStyle_2.12.0
##
## loaded via a namespace (and not attached):
##   [1] ProtGenerics_1.16.0         bitops_1.0-6
##   [3] matrixStats_0.54.0          EnsDb.Hsapiens.v75_2.99.0
##   [5] bit64_0.9-7                 RColorBrewer_1.1-2
##   [7] progress_1.2.0              httr_1.4.0
##   [9] GenomeInfoDb_1.20.0         backports_1.1.4
##  [11] tools_3.6.0                 R6_2.4.0
##  [13] rpart_4.1-15                Hmisc_4.2-0
##  [15] DBI_1.0.0                   lazyeval_0.2.2
##  [17] BiocGenerics_0.30.0         Gviz_1.28.0
##  [19] colorspace_1.4-1            nnet_7.3-12
##  [21] tidyselect_0.2.5            gridExtra_2.3
##  [23] prettyunits_1.0.2           GGally_1.4.0
##  [25] curl_3.3                    bit_1.1-14
##  [27] compiler_3.6.0              Biobase_2.44.0
##  [29] htmlTable_1.13.1            DelayedArray_0.10.0
##  [31] labeling_0.3                rtracklayer_1.43.3
##  [33] bookdown_0.9                scales_1.0.0
##  [35] checkmate_1.9.1             stringr_1.4.0
##  [37] digest_0.6.18               Rsamtools_2.0.0
##  [39] foreign_0.8-71              rmarkdown_1.12
##  [41] XVector_0.24.0              dichromat_2.0-0
##  [43] base64enc_0.1-3             pkgconfig_2.0.2
##  [45] htmltools_0.3.6             ensembldb_2.8.0
##  [47] limma_3.40.0                BSgenome_1.52.0
##  [49] htmlwidgets_1.3             rlang_0.3.4
##  [51] rstudioapi_0.10             RSQLite_2.1.1
##  [53] BiocParallel_1.18.0         acepack_1.4.1
##  [55] dplyr_0.8.0.1               VariantAnnotation_1.30.0
##  [57] RCurl_1.95-4.12             magrittr_1.5
##  [59] GenomeInfoDbData_1.2.1      Formula_1.2-3
##  [61] Matrix_1.2-17               Rcpp_1.0.1
##  [63] munsell_0.5.0               S4Vectors_0.22.0
##  [65] ensemblVEP_1.26.0           stringi_1.4.3
##  [67] yaml_2.2.0                  SummarizedExperiment_1.14.0
##  [69] zlibbioc_1.30.0             plyr_1.8.4
##  [71] grid_3.6.0                  blob_1.1.1
##  [73] parallel_3.6.0              crayon_1.3.4
##  [75] lattice_0.20-38             Biostrings_2.52.0
##  [77] splines_3.6.0               GenomicFeatures_1.36.0
##  [79] pander_0.6.3                hms_0.4.2
##  [81] pillar_1.3.1                GenomicRanges_1.36.0
##  [83] reshape2_1.4.3              biomaRt_2.40.0
##  [85] stats4_3.6.0                XML_3.98-1.19
##  [87] glue_1.3.1                  evaluate_0.13
##  [89] biovizBase_1.32.0           latticeExtra_0.6-28
```

```
##  [91] data.table_1.12.2          BiocManager_1.30.4
##  [93] gtable_0.3.0               purrr_0.3.2
##  [95] reshape_0.8.8              assertthat_0.2.1
##  [97] ggplot2_3.1.1              xfun_0.6
##  [99] AnnotationFilter_1.8.0     survival_2.44-1.1
## [101] tibble_2.1.1               GenomicAlignments_1.20.0
## [103] AnnotationDbi_1.46.0       memoise_1.1.0
## [105] IRanges_2.18.0             cluster_2.0.9
```

References

McLaren, W., B. Pritchard, D. Rios, Y. Chen, P. Flicek, and F. Cunningham. 2010. "Deriving the Consequences of Genomic Variants with the Ensembl API and SNP Effect Predictor." Journal Article. *Bioinformatics* 26 (16): 2069–70. https://doi.org/10.1093/bioinformatics/btq330.