

# Package ‘vivaldi’

December 15, 2022

**Type** Package

**Title** Viral Variant Location and Diversity

**Version** 1.0.0

**Description** Analysis of minor alleles in Illumina sequencing data of viral genomes. Functions in 'vivaldi' primarily operate on vcf files.

**License** MIT + file LICENSE

**URL** <https://github.com/GreshamLab/vivaldi>

**BugReports** <https://github.com/GreshamLab/vivaldi/issues>

**Imports** dplyr (>= 1.0.2), ggplot2 (>= 3.3.2), glue (>= 1.4.2),  
magrittr (>= 2.0.1), plotly (>= 4.10.0), seqinr (>= 4.2-8),  
tidyr (>= 1.1.2), tidyselect (>= 1.1.2), tidyverse (>= 1.3.0),  
vcfR (>= 1.12.0)

**Suggests** kableExtra, knitr, rmarkdown, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.2.1

**Config/testthat/edition** 3

**Depends** R (>= 2.10)

**NeedsCompilation** no

**Author** Marissa Knoll [aut],  
Katherine Johnson [aut],  
Megan Hockman [aut],  
Eric Borenstein [aut],  
Mohammed Khalfan [aut],  
Elodie Ghedin [aut],  
David Gresham [aut, cre, cph]

**Maintainer** David Gresham <dg107@nyu.edu>

**Repository** CRAN

**Date/Publication** 2022-12-15 11:00:02 UTC

**R topics documented:**

add_metadata . . . . .	2
af_distribution . . . . .	3
arrange_data . . . . .	4
dNdS_segment . . . . .	5
example_filtered_SNV_df . . . . .	5
filter_variants . . . . .	6
merge_replicates . . . . .	7
plot_shannon . . . . .	8
position_allele_freq . . . . .	9
prepare_annotations . . . . .	10
read_reference_fasta_dna . . . . .	10
shannon_entropy . . . . .	11
shared_snv_plot . . . . .	12
shared_snv_table . . . . .	13
snpeff_info . . . . .	14
snv_genome . . . . .	14
snv_location . . . . .	15
snv_segment . . . . .	16
tally_it . . . . .	17
tstv_plot . . . . .	18
tstv_ratio . . . . .	18

<b>Index</b>	<b>19</b>
--------------	-----------

---

add_metadata	<i>add_metadata</i>
--------------	---------------------

---

**Description**

Adds metadata information to the vcf dataframe

**Usage**

```
add_metadata(df, metadf, by_vcf, by_meta)
```

**Arguments**

df	A rearranged vcf dataframe (arrange_data)
metadf	A metadata dataframe
by_vcf	A vector of column names in the vcf dataframe that should be used to merge the vcf data with the metadata
by_meta	A vector of column names in the metadata dataframe that should be used to merge the metadata with the vcf data

**Value**

A vcf dataframe with metadata included

**Examples**

```
df <- data.frame(CHROM = c("A", "B"),
                 POS = c(234, 240),
                 REF = c("G", "A"),
                 ALT = c("A", "G")
                )

sizes <- data.frame(segment = c("A", "B"),
                   SegmentSize = c(2280, 2274)
                  )

df

sizes

# Add a new column of sizes of the segments which are necessary for
# downstream calculations such as transition:transversion (tstv) and dNdS.
add_metadata(df, sizes, c('CHROM'), c('segment'))
```

---

af_distribution	<i>af_distribution</i>
-----------------	------------------------

---

**Description**

Plots distribution of all minor variants

**Usage**

```
af_distribution(df)
```

**Arguments**

df                    A dataframe that has been arranged (arrange\_data) and filtered (filter\_variants)

**Value**

plots with the distribution of all minor variants

**Examples**

```
# Example 1:
df <- data.frame(sample = c("m1", "m2", "m1", "m2", "m1"),
  CHROM = c("PB1", "PB1", "PB2", "PB2", "NP"),
  POS = c(234, 234, 240, 240, 254),
  REF = c("G", "G", "A", "A", "C"),
  ALT = c("A", "A", "G", "G", "T"),
  minorfreq = c(0.010, 0.022, 0.043, 0.055, 0.011),
  majorfreq = c(0.990, 0.978, 0.957, 0.945, 0.989),
  minorcount = c(7, 15, 26, 32, 7),
  majorcount = c(709, 661, 574, 547, 610),
  gt_DP = c(716, 676, 600, 579, 617)
)

af_distribution(df)

# Example 2:
af_distribution(example_filtered_SNV_df)
```

---

arrange\_data

*arrange\_data*


---

**Description**

Reads in a directory of VCF files and converts them into a single dataframe

**Usage**

```
arrange_data(
  vardir,
  reference_fasta,
  annotated = "yes",
  ntlist = c("A", "G", "T", "C", "-"),
  verbose = FALSE
)
```

**Arguments**

vardir	Directory path containing vcf files
reference_fasta	Reference fasta file used for alignment
annotated	Whether the VCF files are annotated using snpeff "yes" or "no" (default "yes")
ntlist	Nucleotides (default A, T, G, C) used for finding multiple alt alleles
verbose	set verbosity of the vcfR commands

**Value**

A large dataframe containing information from all input VCF files

---

dNdS_segment	<i>dNdS_segment</i>
--------------	---------------------

---

### Description

Reads in a dataframe that has been arranged (`arrange_data`), filtered (`filter_variants`), and annotated (`prepare_annotations`), calculates dNdS, and outputs plots

### Usage

```
dNdS_segment(annotation_df, SPLICEFORMS)
```

### Arguments

`annotation_df` A rearranged, filtered, and annotated vcf dataframe - must be for amino-acid specific calculations, cannot be the same as the dataframe used for SNP calculations

`SPLICEFORMS` A character vector of isoform names

### Value

A plot showing the dN/dS ratio for each splice form (rather than segment) for each sample

### Examples

```
# Sample Data
head(example_filtered_SNV_df)
dim(example_filtered_SNV_df)

# Plot showing the dN/dS ratio for each splice form
SPLICEFORMS = c("H1N1_PB2.1", "H1N1_PB1.1", "H1N1_NS.2")
dNdS_segment(example_filtered_SNV_df, SPLICEFORMS)
```

---

```
example_filtered_SNV_df
```

*Example Dataframe The DF\_filt\_SNVs dataframe created in the vignette*

---

### Description

Example Dataframe The DF\_filt\_SNVs dataframe created in the vignette

### Usage

```
example_filtered_SNV_df
```

**Format**

## 'example\_filtered\_SNV\_df' A data frame with 735 rows and 57 columns:

---

filter_variants	<i>filter_variants</i>
-----------------	------------------------

---

**Description**

Filters single-nucleotide variants using a coverage and frequency cutoff

**Usage**

```
filter_variants(df, coverage_cutoff = 200, frequency_cutoff = 0.03)
```

**Arguments**

df	A rearranged VCF dataframe (rearranged using the arrange_data function)
coverage_cutoff	The coverage cutoff for calling a SNV (default: 200x)
frequency_cutoff	Frequency cutoff for calling a SNV (default: 3%)

**Value**

A filtered VCF dataframe

**Examples**

```
df <- data.frame(CHROM = c("A", "B", "C"),
                 POS = c(234, 240, 255),
                 ALT_FREQ = c(0.016, 0.049, 0.031),
                 gt_DP = c(716, 600, 187)
                )

df

# Default: filter by 3% frequency threshold and 200 coverage cutoff
filter_variants(df)

# Example 1: A 1% allele frequency threshold and 200 coverage cutoff
filter_variants(df, coverage_cutoff = 200, frequency_cutoff = 0.01)

# Example 2: A 2% allele frequency threshold and 100 coverage cutoff
filter_variants(df, coverage_cutoff = 100, frequency_cutoff = 0.02)
```

---

```
merge_replicates      merge_replicates
```

---

**Description**

Merges replicate VCF files into a single dataframe

**Usage**

```
merge_replicates(vardf, repdata, nameofrep1, nameofrep2, commoncols)
```

**Arguments**

vardf	Data frame of variants
repdata	Data frame of replicate information
nameofrep1	Name of variable representing the first replicate, must be written with quotes
nameofrep2	Name of variable representing the second replicate
commoncols	List of columns to merge the replicates by

**Value**

a data frame containing replicate information

**Examples**

```
df <- data.frame(sample = c("m1", "m2", "m1", "m2", "m1"),
  CHROM = c("PB1", "PB1", "PB2", "PB2", "NP"),
  POS = c(234, 234, 240, 240, 254),
  REF = c("G", "G", "A", "A", "C"),
  ALT = c("A", "A", "G", "G", "T"),
  minorfreq = c(0.010, 0.022, 0.043, 0.055, 0.011),
  majorfreq = c(0.990, 0.978, 0.957, 0.945, 0.989),
  minorcount = c(7, 15, 26, 32, 7),
  majorcount = c(709, 661, 574, 547, 610),
  gt_DP = c(716, 676, 600, 579, 617)
)

# Dataframe shows a pair of replicates and their variants at 3 positions.
df

replicates <- data.frame(filename = c("m1", "m2"),
  replicate = c("rep1", "rep2"),
  sample = c("a_2_iv", "a_2_iv")
)

# Dataframe showing relationship between filename, replicate, and sample name
replicates
```

```
# Merge by the following columns
cols = c("sample", "CHROM", "POS", "REF", "ALT")

merge_replicates(df, replicates, "rep1", "rep2", cols)
# The dataframe now contains the 2 variants at positions 234 & 240 that were
# detected in both sequencing replicates whereas the variant at position 254
# was only in a single replicate so it was removed during the merge.
```

---

plot\_shannon

*plot\_shannon*


---

### Description

Reads in a dataframe that has been arranged (`arrange_data`), filtered (`filter_variants`), and piped through the Shannon calculations (`shannon_entropy`) and outputs plots

### Usage

```
plot_shannon(shannon_df)
```

### Arguments

`shannon_df` A dataframe that has been arranged (`arrange_data`), filtered (`filter_variants`), and piped through the Shannon calculations (`shannon_entropy`)

### Details

The `plot_shannon()` function takes the variant dataframe and generates three plots. 1. The Shannon entropy, or amount of diversity, at each position in the genome at which a variant was found. 2. The Shannon entropy summed over each segment 3. The Shannon entropy summed over each genome. A higher value indicates more diversity.

### Value

Three plots showing the nt Shannon, chrom Shannon, and full genome Shannon calculations

### Examples

```
# Sample dataframe
df <- data.frame(sample = c("m1", "m2", "m1", "m2", "m1"),
  CHROM = c("PB1", "PB1", "PB2", "PB2", "NP"),
  POS = c(234, 234, 240, 240, 254),
  minorfreq = c(0.010, 0.022, 0.043, 0.055, 0.011),
  majorfreq = c(0.990, 0.978, 0.957, 0.945, 0.989),
  SegmentSize = c(2280, 2280, 2274, 2274, 1809)
)
df
```



```
genome_size = 13133

# Modify the dataframe to add 5 new columns of shannon entropy data:
# 1. shannon_ntpos
# 2. chrom_shannon
# 3. genome_shannon
# 4. shannon_chrom_perkb
# 5. genome_shannon_perkb
shannon_df = shannon_entropy(df, genome_size)

# Plot
plot_shannon(shannon_df)
```

---

position\_allele\_freq    *position\_allele\_freq*

---

## Description

Reads in a dataframe that has been arranged (`arrange_data`) and filtered (`filter_variants`) and outputs plots

## Usage

```
position_allele_freq(vardf, segment, nt)
```

## Arguments

<code>vardf</code>	A rearranged ( <code>arrange_data</code> ) and filtered ( <code>filtered_variants</code> ) vcf dataframe
<code>segment</code>	Name of segment (must be in quotes)
<code>nt</code>	Position on segment (must be in quotes)

## Value

A plot showing the the frequencies of the major and minor allele at the given position across all samples

## Examples

```
position_allele_freq(example_filtered_SNV_df, "H1N1_NP", "1247")
```

prepare\_annotations    *prepare\_annotations*

---

### Description

Separates the SNPeff annotations found in an annotated and rearranged VCF dataframe (arranged using `arrange_data`)

### Usage

```
prepare_annotations(df)
```

### Arguments

`df`                    A rearranged and annotated VCF dataframe

### Value

A dataframe containing each annotation on a separate column

### Examples

```
# Example 1: Shows the separation of the ANN column based on | delimiter.
test <- data.frame( ANN = c("A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P"))

# The ANN column will be split based on the strings in `snpeff_info()` and
# an additional "error" column.
snpeff_info()

# Split the SNPeff annotations in "ANN" column and save to dataframe `df`
df <- prepare_annotations(df)

# The one "ANN" column is split into 16 columns
dim(test)
dim(df)
```

---

read\_reference\_fasta\_dna  
                          *read\_reference\_fasta\_dna*

---

### Description

Imports reference fasta, generates a dataframe with chroms and chrom lengths

### Usage

```
read_reference_fasta_dna(reference_fasta)
```

**Arguments**

reference\_fasta  
The name and location of the reference fasta file used for alignment

**Value**

A dataframe containing the chroms and chrom lengths of a reference fasta

---

shannon_entropy	<i>shannon_entropy</i>
-----------------	------------------------

---

**Description**

Takes a rearranged vcf dataframe and calculates the Shannon entropy

**Usage**

```
shannon_entropy(df, genome_size)
```

**Arguments**

df                   A rearranged vcf dataframe (arrange\_data)  
genome\_size        Size of whole genome being used

**Details**

Shannon entropy is a commonly used metric to describe the amount of genetic diversity in sequencing data. It is calculated by considering the frequency of the ALT and REF allele at every position and then summing those values over 1) a segment or 2) the entire genome. These values can then be normalized by sequence length (kb) in order to compare across different segments or samples.

**Value**

A dataframe with Shannon entropy/kb calculations for the chroms and entire genome

**Examples**

```
# Sample dataframe
df <- data.frame(sample = c("m1", "m2", "m1", "m2", "m1"),
                 CHROM = c("PB1", "PB1", "PB2", "PB2", "NP"),
                 minorfreq = c(0.010, 0.022, 0.043, 0.055, 0.011),
                 majorfreq = c(0.990, 0.978, 0.957, 0.945, 0.989),
                 SegmentSize = c(2280, 2280, 2274, 2274, 1809)
                )

df

genome_size = 13133
```

```
# Modify the dataframe to add 5 new columns of shannon entropy data:  
# 1. shannon_ntpos  
# 2. chrom_shannon  
# 3. genome_shannon  
# 4. shannon_chrom_perkb  
# 5. genome_shannon_perkb  
shannon_entropy(df, genome_size)
```

---

shared_snv_plot	<i>shared_snv_plot</i>
-----------------	------------------------

---

## Description

Reads in a dataframe that has been arranged (`arrange_data`) and filtered (`filter_variants`) and outputs plots

## Usage

```
shared_snv_plot(vardf, samples = unique(DF_filt$sample))
```

## Arguments

<code>vardf</code>	A rearranged ( <code>arrange_data</code> ) and filtered ( <code>filtered_variants</code> ) vcf dataframe
<code>samples</code>	A vector of samples to be compared (default:all samples in <code>DF_filt</code> )

## Value

A plot showing the location of variants and the number of samples that contain each variant

## Examples

```
samples = c("a_1_fb", "a_1_iv", "a_2_fb", "a_2_iv", "a_3_fb", "a_3_iv", "b_1_fb", "b_1_iv")  
shared_snv_plot(example_filtered_SNV_df, samples)
```

---

shared_snv_table	<i>shared_snv_table</i>
------------------	-------------------------

---

## Description

Reads in a dataframe that has been arranged (`arrange_data`) and filtered (`filter_variants`) and outputs a table

## Usage

```
shared_snv_table(vardf)
```

## Arguments

`vardf` A rearranged (`arrange_data`) and filtered (`filtered_variants`) vcf dataframe

## Details

The `'shared_snv_table()'` function takes the variant dataframe and creates a new table, listing the variants in descending order of frequency how many samples they are found in. This function is meant to simplify further investigation of visual patterns in the previous plot.

## Value

A table listing variants in order by how many samples they are found in

## Examples

```
# Sample dataframe has 57 columns
dim(example_filtered_SNV_df)

# Simplify sample dataframe
df <- shared_snv_table(example_filtered_SNV_df)

# Dataframe created has 15 columns
df
dim(df)
```

---

`snpeff_info`*snpeff\_info*

---

**Description**

Returns vector containing information in snpeff annotations

**Usage**

```
snpeff_info()
```

**Value**

Returns vector containing information in snpeff annotations

**Examples**

```
snpeff_info()
```

---

`snv_genome`*snv\_genome*

---

**Description**

Reads in a dataframe that has been arranged (`arrange_data`) and filtered (`filter_variants`) and outputs plots

**Usage**

```
snv_genome(vardf)
```

**Arguments**

`vardf` A rearranged (`arrange_data`) and filtered (`filtered_variants`) vcf dataframe

**Value**

A bar plot showing the number of variants per sample colored by their SNPEff annotation

**Examples**

```
# Example 1: Simple dataframe
df <- data.frame(sample = c("m1", "m1", "m1", "m1", "m1",
                           "m2", "m2", "m2", "m2", "m2"),
                 annotation = c("downstream_gene_variant", "synonymous_variant",
                                "synonymous_variant", "stop_gained",
                                "missense_variant", "downstream_gene_variant",
                                "downstream_gene_variant", "synonymous_variant",
                                "stop_gained", "missense_variant")
                 )

df

snv_genome(df)

# Example 2: Sample dataframe
snv_genome(example_filtered_SNV_df)
```

---

<code>snv_location</code>	<i>snv_location</i>
---------------------------	---------------------

---

**Description**

Reads in the vcf dataframe and generates a plot showing the frequency and location of SNVs

**Usage**

```
snv_location(df)
```

**Arguments**

`df`                    A rearranged dataframe

**Value**

A plot showing the location and frequency of SNVs found across samples

**Examples**

```
# Example 1:
df <- data.frame(sample = c("m1", "m1", "m1", "m1", "m1",
                           "m2", "m2", "m2", "m2", "m2"),
                 CHROM = c("PB1", "PB1", "PB2", "PB2", "PB2",
                           "PB1", "PB1", "PB2", "PB2", "PB2"),
                 POS = c(234, 266, 117, 134, 180,
                        234, 266, 199, 88, 180),
                 major = c("G", "G", "A", "A", "C",
                           "G", "G", "A", "G", "C"),
                 )
```

```

      minor = c("A", "A", "G", "G", "T",
                "A", "A", "G", "A", "T"),
      ALT_TYPE = c("minor", "minor", "minor", "minor", "minor",
                  "minor", "minor", "minor", "major", "minor"),
      minorfreq = c(0.010, 0.022, 0.043, 0.055, 0.011,
                   0.010, 0.022, 0.043, 0.055, 0.011),
      majorfreq = c(0.990, 0.978, 0.957, 0.945, 0.989,
                   0.990, 0.978, 0.957, 0.945, 0.989)
    )

df

snv_location(df)

# Example 2:
snv_location(head(example_filtered_SNV_df, n=20))

```

---

snv\_segment

*snv\_segment*

---

## Description

Reads in a dataframe that has been arranged (`arrange_data`) and filtered (`filter_variants`) and outputs plots

## Usage

```
snv_segment(vardf)
```

## Arguments

`vardf` A rearranged (`arrange_data`) and filtered (`filtered_variants`) vcf dataframe

## Value

A bar plot showing the number of variants colored by their SNPEff annotation

## Examples

```

# Example 1: Simple data
df <- data.frame(sample = c("m1", "m1", "m1", "m1", "m1",
                           "m2", "m2", "m2", "m2", "m2"),
                 CHROM = c("PB1", "PB1", "PB2", "PB2", "PB2",
                           "PB1", "PB1", "PB2", "PB2", "PB2"),
                 annotation = c("downstream_gene_variant", "synonymous_variant",
                                "synonymous_variant", "stop_gained", "missense_variant",
                                "downstream_gene_variant", "downstream_gene_variant",
                                "synonymous_variant", "stop_gained", "missense_variant")
                )

```



```
df
snv_segment(df)

# Example 2: Sample data
snv_segment(example_filtered_SNV_df)
```

---

tally_it	<i>tally_it</i>
----------	-----------------

---

## Description

Groups the input vcf data frame using a list of variables and tallies the number of occurrences

## Usage

```
tally_it(df, groupit, new_colname)
```

## Arguments

df	A rearranged vcf dataframe (arrange_data)
groupit	A vector containing column names that data should be grouped by
new_colname	The name of the count column

## Value

A dataframe with columns from the 'groupit' vector and the number of times each unique grouping occurs in the data

## Examples

```
# Sample dataframe of 7 variants across 2 samples
df <- data.frame(
  sample = c("sample1", "sample1", "sample1", "sample2",
            "sample2", "sample2", "sample2"),
  CHROM = c("PB1", "PB2", "PB2", "LEO", "LEO", "LEO", "ALE"),
  SegmentSize = c(2280, 2274, 2274, 1701, 1701, 1701, 1888),
  minorfreq = c(0.04422785, 0.03738175, 0.01390202, 0.02927786,
               0.03071955, 0.02626025, 0.02875321)
)

# Example 1: to get the sum of variants on every segment:
groupit = c('sample','CHROM', "SegmentSize")
tally_it(df, groupit, "snv_count")

# Example 2: to get the count across genomes:
groupit = c('sample')
```

```
tally_it(df, groupit, "snv_count")
```

---

```
tstv_plot          tstv_plot
```

---

### Description

Plots Ts/Tv ratios

### Usage

```
tstv_plot(df)
```

### Arguments

df                    TsTv dataframe generated using the tstv\_ratio function

### Value

two plots showing the K2P and simple Ts/Tv ratios

### Examples

```
df <- tstv_ratio(example_filtered_SNV_df, 1300)
tstv_plot(df)
```

---

```
tstv_ratio          tstv_ratio
```

---

### Description

Inputs a filtered and rearranged vcf dataframe and calculates the transition/transversion ratio

### Usage

```
tstv_ratio(df, genome_size)
```

### Arguments

df                    The filtered and rearranged variant dataframe  
genome\_size        Size of whole genome being used

### Value

A dataframe containing the calculated transition/transversion ratio (R or basic\_tstv)

### Examples

```
tstv_ratio(example_filtered_SNV_df, 13000)
```

# Index

## \* datasets

- example\_filtered\_SNV\_df, 5
- add\_metadata, 2
- af\_distribution, 3
- arrange\_data, 4
- dNdS\_segment, 5
- example\_filtered\_SNV\_df, 5
- filter\_variants, 6
- merge\_replicates, 7
- plot\_shannon, 8
- position\_allele\_freq, 9
- prepare\_annotations, 10
- read\_reference\_fasta\_dna, 10
- shannon\_entropy, 11
- shared\_snv\_plot, 12
- shared\_snv\_table, 13
- snpeff\_info, 14
- snv\_genome, 14
- snv\_location, 15
- snv\_segment, 16
- tally\_it, 17
- tstv\_plot, 18
- tstv\_ratio, 18