Package 'immuneSIM'

October 13, 2022

Type Package

Title Tunable Simulation of B- And T-Cell Receptor Repertoires

Version 0.8.7

Author Cédric R. Weber [aut, cre],
Victor Greiff [aut]

Maintainer Cédric R. Weber <cedric.weber@bsse.ethz.ch>

Description Simulate full B-cell and T-cell receptor repertoires using an in silico
recombination process that includes a wide variety of tunable parameters to introduce noise and biases.
Additional post-
simulation modification functions allow the user to implant motifs or codon biases as
well as remodeling sequence similarity architecture. The output repertoires contain records of all
relevant repertoire dimensions and can be analyzed using provided repertoire analysis functions.
Preprint is available at bioRxiv (Weber et al., 2019 <doi:10.1101/759795>).

Depends R (>= 3.4.0)

Imports poweRlaw, stringdist, Biostrings, igraph, stringr, data.table,
plyr, reshape2, ggplot2, grid, ggthemes, RColorBrewer, Metrics,
repmis

License GPL-3

URL https://immuneSIM.readthedocs.io

BugReports https://github.com/GreiffLab/immuneSIM/issues

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-09-27 10:30:06 UTC

R topics documented:

---

codon_replacement *Replaces codons with synonymous codons*

Description

Replaces codons with synonymous codons

Usage

```
codon_replacement(repertoire, mode = "both", codon_replacement_list,
  skip_probability = 0)
```

Arguments

`repertoire` An annotated AIRR compliant immuneSIM repertoire.
(http://docs.airr-community.org/en/latest/)

`mode` Defines whether codons should be replaced in the nt or AA sequence or in both ("nt","AA","both")

`codon_replacement_list`
List containing instructions for which codons should be replaced and how

`skip_probability`
Probability with which a sequence gets skipped in the codon replacement process between 0,1

Value

immuneSIM repertoire with replaced codons

Examples

```
repertoire <- list_example_repertoires[["example_repertoire_A"]]
rep_codon_repl <- codon_replacement(repertoire, "both",
list(tat = "tac", agt = "agc", gtt = "gtg"), 0)
```

---

codon_replacement_reconstruction

Decodes immuneSIM repertoire codon replacements events.

Description

Decodes immuneSIM repertoire codon replacements events.

Usage

```
codon_replacement_reconstruction(codon_replacement_vec)
```

Arguments

`codon_replacement_vec`

An vector containing strings describing codon replacement events as generated by codon_replacement() function. The string contains information on every replacement event in the form:

"initial_codon:replacement_codon:number_of_occurrences"

which is combined into: "Replacement1|Replacement2|Replacement3".

(For example: "tac,tat:3|agc,agt:1|gtg,gtt:0".)

Value

List of dataframes. Each entry contains replacement info including count of occurrences for each simulated sequence.

Examples

```
codon_replacement_example <- c("tat,tac:3|agt,agc:3|gtt,gtg:0", "tat,tac:1|agt,agc:1|gtt,gtg:1")
codon_replacement_list <- codon_replacement_reconstruction(codon_replacement_example)
```

`combine_into_paired` *Generates a dataframe from separate heavy and light or beta and alpha chain dataframes*

Description

Generates a dataframe from separate heavy and light or beta and alpha chain dataframes

Usage

```
combine_into_paired(repertoire_heavy, repertoire_light)
```

Arguments

`repertoire_heavy`
A repertoire containing heavy/beta chain data

`repertoire_light`
A repertoire containing light/alpha chain data

Value

immuneSIM repertoire containing heavy/beta and light/alpha chain data.

Examples

```
repertoire_heavy <- immuneSIM(number_of_seqs = 5,species = "mm",receptor = "ig", chain = "h")
repertoire_light <- immuneSIM(number_of_seqs = 5,species = "mm",receptor = "ig", chain = "kl")
paired_repertoire <- combine_into_paired(repertoire_heavy,repertoire_light)
```

`gen_code` *Translation dictionary amino acid <-> nucleotide codon*

Description

A dataframe containing a mapping from each of 64 codons to amino acids.

Usage

```
gen_code
```

Format

A data frame with 64 rows and variables:

aa amino acid

codon nucleotide codon

Source

https://www.genscript.com/tools/codon-table

`hotspot_df` *Hotspot dataframe for SHM*

Description

A dataframe containing mutation probabilities for every possible 5mer pattern

Usage

```
hotspot_df
```

Format

A data frame with 1024 rows and variables:

pattern amino acid

toA probability of mutation to adenine

toC probability of mutation to cytosine

toG probability of mutation to guanine

toT probability of mutation to thymine

Source source of probability

Source

https://cran.r-project.org/package=AbSim

`hub_seqs_exclusion` *Deletes top hub sequences from repertoire, changing the network architecture.*

Description

Deletes top hub sequences from repertoire, changing the network architecture.

Usage

```
hub_seqs_exclusion(repertoire, top_x = 0.005, report = FALSE,
  output_dir = "", verbose = TRUE)
```

Arguments

`repertoire` An annotated AIRR compliant repertoire.
(http://docs.airr-community.org/en/latest/)

`top_x` Determines what percentage of hub sequences get excluded
(Default: 0.005, i.e. Top 0.5 percent)

`report` The user can choose to output a report csv file containing the excluded sequences. (Default: FALSE)

`output_dir` If user specifies and output directory a csv file containing the excluded sequences is saved at that path, otherwise it will be saved in tempdir().

`verbose` Determines whether messages on plot locations are output to user. (Default: TRUE)

Value

Repertoire reduced by hub sequence (new network architecture)

Examples

```
repertoire <- list_example_repertoires[["example_repertoire_A"]]
rep_excluded_hubs <- hub_seqs_exclusion(repertoire, top_x = 0.005, output_dir = "")
```

immuneSIM *Simulates an immune repertoire based on user-defined parameters*

Description

Simulates an immune repertoire based on user-defined parameters

Usage

```
immuneSIM(number_of_seqs = 1000,
  vdj_list = list_germline_genes_allele_01, species = "mm",
  receptor = "ig", chain = "h",
  insertions_and_deletion_lengths = insertions_and_deletion_lengths_df,
  user_defined_alpha = 2, name_repertoire = "sim_rep",
  length_distribution_rand = length_dist_simulation, random = FALSE,
  shm.mode = "none", shm.prob = 15/350, vdj_noise = 0,
  vdj_dropout = c(V = 0, D = 0, J = 0), ins_del_dropout = c(""),
  equal_cc = FALSE, freq_update_time = round(0.5 * number_of_seqs),
  max_cdr3_length = 100, min_cdr3_length = 6, verbose = TRUE,
  airr_compliant = TRUE)
```

Arguments

`number_of_seqs` Integer defining the number of sequences that should be simulated

`vdj_list` List containing germline genes and their frequencies

`species` String defining species for which repertoire should be simulated ("mm": mouse, "hs": human. Default: "mm").

`receptor` String defining receptor type ("ig" or "tr". Default: "ig")

`chain` String defining chain (for ig: "h","k","l", for tr: "b" or "a". Default: "h")

`insertions_and_deletion_lengths`
Data.frame containing np1, np2 sequences as well as deletion lengths. (Pooled from murine repertoire data, Greiff,2017) Note: This is a subset of 500000 observations of the dataframe used in the paper. The full dataframe which can be introduced here can be found on: (Git-Link)

`user_defined_alpha`
Numeric. Scaling parameter used for the simulation of powerlaw distribution (recommended range 2-5. Default: 2, https://en.wikipedia.org/wiki/Power_law)

`name_repertoire`
String defining chosen repertoire name recorded in the name_repertoire column of the output for identification.

`length_distribution_rand`
Vector containing lengths of immune receptor sequences based on immune repertoire data (Greiff, 2017).

`random` Boolean. If TRUE repertoire will consist of fully random sequences, independent of germline genes.

`shm.mode` String defining mode of somatic hypermutation simulation based on AbSim (options: 'none', 'data','poisson', 'naive', 'motif', 'wrc'. Default: 'none'). See AbSim documentation.

`shm.prob` Numeric defining probability of a SHM (somatic hypermutation) occurring at each position.

`vdj_noise` Numeric between 0,1, setting noise level to be introduced in provided V,D,J germline frequencies. 0 denotes no noise. (Default: 0)

`vdj_dropout` Named vector containing entries V,D,J setting the number of germline genes to be dropped out. (Default: c("V"=0,"D"=0,"J"=0))

`ins_del_dropout`
String determining whether insertions and deletions should occur. Options: "", "no_insertions", "no_insertions_n1", "no_insertions_n2", "no_deletions_v", "no_deletions_d_5", "no_deletions_d_3", "no_deletions_j", "no_deletions_vd", "no_deletions". Default: "")

`equal_cc` Boolean that if set TRUE will override user_defined_alpha and generate a clone count distribution that is equal for all sequences. Default: FALSE.

`freq_update_time`
Numeric determining whether simulated VDJ frequencies agree with input after set amount of sequences to correct for VDJ bias. Default: Update after 50 percent of sequences.

`max_cdr3_length`
: Numeric defining maximal length of cdr3. (Default: 100)

`min_cdr3_length`
: Numeric defining minimal length of cdr3. (Default: 6)

`verbose` Boolean toggling printing of progress on and off (Default: FALSE)

`airr_compliant` Boolean determining whether output repertoire should be named in an AIRR compliant manner (Default: TRUE). (http://docs.airr-community.org/en/latest/)

Value

An annotated AIRR-compliant immuneSIM repertoire. (http://docs.airr-community.org/en/latest/)

Examples

```
sim_rep <- immuneSIM(number_of_seqs = 10, vdj_list = list_germline_genes_allele_01,
species = "mm", receptor = "ig", chain = "h",
insertions_and_deletion_lengths = insertions_and_deletion_lengths_df,
user_defined_alpha = 2,name_repertoire = "mm_igh_sim",
shm.mode = "data",shm.prob=15/350,vdj_noise = 0, vdj_dropout = c(V=0,D=0,J=0),
ins_del_dropout = "",min_cdr3_length = 6)
```

insertions_and_deletion_lengths_df

Dataframe containing insertion sequences and deletion lengths

Description

A dataframe containing all insertions and deletions observed in experimental data (pooled across all samples, Greiff, 2017) This dataframe is a subset of the dataframe used in the application note. The original dataframe which contains 11363603 rows can be downloaded from:

Usage

```
insertions_and_deletion_lengths_df
```

Format

A data frame with 500000 rows and variables:

n1 np1 insertions

n2 np2 insertions

del_v lengths of V gene deletions

del_d_5 lengths of 5’ end D gene deletions

del_d_3 lengths of 3’ end D gene deletions

del_j lengths of J gene deletions

Details

https://github.com/GreiffLab/immuneSIM or using the provided function: load_insdel_data()

Source

`https://doi.org/10.1016/j.celrep.2017.04.054`

`length_dist_simulation` *Vector containing VDJ length distributions*

Description

A vector containing 10000 VDJ lengths for simulating of fully random sequences (independent of germline genes)

Usage

```
length_dist_simulation
```

Format

A vector with 10000 entries:

length VDJ nucleotide lengths sampled from murine naive follicular B-cell data, Greiff 2017

Source

`https://doi.org/10.1016/j.celrep.2017.04.054`

`list_example_repertoires` *Example repertoires*

Description

A list containing two example repertoires (100 sequences each) simulated with immuneSIM using default parameters. These repertoires are used in the examples.

Usage

```
list_example_repertoires
```

Format

A list with 2 entries:

example_repertoire_A Repertoire simulated using standard parameters (A)

example_repertoire_A Repertoire simulated using standard parameters (B)

Source

https://immunesim.readthedocs.io

`list_germline_genes_allele_01`

Collection of germline genes and frequencies

Description

A list containing sublists for species ("hs","mm") which in turn contain sublists for receptors ("ig","tr") which are subset in chains ("h", "k", "l" and "b", "a", respectively). Each entry contains a list of three dataframes ("V","D" and "J") with the major IMGT annotated germline genes including name, sequence based on IMGT and frequencies based on experimental data from DeWitt(2017), Emerson (2017), Greiff (2017) and Madi (2017)

Usage

```
list_germline_genes_allele_01
```

Format

A list of lists containing dataframes with up to 126 entries:

gene name of germline gene

allele allele number (presently restricted to allele 01)

sequence nucleotide sequence of germline gene

species name of species

frequency Frequencies of germline genes based on experimental data

Source

http://www.imgt.org/vquest/refseqh.html

https://doi.org/10.1371/journal.pone.0160853

https://doi.org/10.1038/ng.3822

https://doi.org/10.1016/j.celrep.2017.04.054

https://doi.org/10.7554/eLife.22057

`load_insdel_data` *Loads full insertion/deletion data from GitHub*

Description

Loads full insertion/deletion data from GitHub

Usage

```
load_insdel_data()
```

Value

Dataframe containing insertions and deletions (11363603 rows, 6 columns)

Examples

```
full_insertions_and_deletion_df <- load_insdel_data()
```

`motif_implantation` *Implant random or predefined motifs into CDR3*

Description

Implant random or predefined motifs into CDR3

Usage

```
motif_implantation(sim_repertoire, motif, fixed_pos = 0)
```

Arguments

`sim_repertoire`	An annotated AIRR compliant immuneSIM repertoire.
`motif`	Either a list that contains number, length and frequencies of motifs or dataframe that contains predefined motifs and their frequencies
`fixed_pos`	defines position at which motif is to be introduced. if 0 motif will be introduced at random position

Value

Repertoire with modified sequences containing implanted motifs in CDR3.

Examples

```
sim_repertoire <- list_example_repertoires[["example_repertoire_A"]]
sim_rep_motifs <- motif_implantation(sim_repertoire,list("n"=2,"k"=3,"freq"=c(0.1,0.1)),0)
```

one_spot_df *One Spot*

Description

A dataframe containing a mutation probabilities to base per 5mer (inherited from AbSim package)

Usage

```
one_spot_df
```

Format

A dataframe with 32 entries:

pattern amino acid

toA probability of mutation to adenine

toC probability of mutation to cytosine

toG probability of mutation to guanine

toT probability of mutation to thymine

Source source of probability

Source

https://cran.r-project.org/package=AbSim

https://doi.org/10.1093/bioinformatics/btx533

plot_repertoire_A_vs_B *Comparative plots of main repertoire features of two input repertoires (length distribution, amino acid frequency, VDJ usage, kmer occurrence)*

Description

Comparative plots of main repertoire features of two input repertoires (length distribution, amino acid frequency, VDJ usage, kmer occurrence)

Usage

```
plot_repertoire_A_vs_B(repertoire_A, repertoire_B,
  names_repertoires = c("Repertoire_A", "Repertoire_B"),
  length_aa_plot = 14, output_dir = "", verbose = TRUE)
```

Arguments

`repertoire_A` An annotated AIRR-compliant immuneSIM repertoire. (http://docs.airr-community.org/en/latest/)

`repertoire_B` An annotated AIRR-compliant immuneSIM repertoire.

`names_repertoires` A vector containing two strings denoting the names of the repertoires / repertoire descriptions.

`length_aa_plot` Defines sequence length for which the amino acid frequency plot will be made.

`output_dir` String containing full path of desired output folder. If empty, figures will be output in tempdir().

`verbose` Determines whether messages on plot locations are output to user. (Default: TRUE)

Value

TRUE (plots saved as pdfs into subfolder 'figures')

Examples

```
repertoire_A <- list_example_repertoires[["example_repertoire_A"]]
repertoire_B <- list_example_repertoires[["example_repertoire_B"]]
plot_repertoire_A_vs_B(
repertoire_A,
repertoire_B,
c("Sim_repertoire_1","Sim_repertoire_2"),
length_aa_plot = 14,
output_dir="")
```

plot_report_repertoire *Plots main repertoire features (length distribution,amino acid frequencies and VDJ usage)*

Description

Plots main repertoire features (length distribution,amino acid frequencies and VDJ usage)

Usage

```
plot_report_repertoire(repertoire, output_dir = "", verbose = TRUE)
```

Arguments

`repertoire`	An annotated AIRR-compliant immuneSIM repertoire. (http://docs.airr-community.org/en/latest/)
`output_dir`	String containing full path of desired output folder. If empty figures will be output in tempdir().
`verbose`	Determines whether messages on plot locations are output to user. (Default: TRUE)

Value

TRUE (plots saved as pdfs into subfolder 'figures')

Examples

```
repertoire <- list_example_repertoires[["example_repertoire_A"]]
plot_report_repertoire(repertoire,output_dir="")
```

`shm_event_reconstruction` *Decodes immuneSIM repertoire shm_events column.*

Description

Decodes immuneSIM repertoire shm_events column.

Usage

```
shm_event_reconstruction(shm_event_vec)
```

Arguments

`shm_event_vec`	An vector containing strings describing SHM events as output in shm_events column of immuneSIM repertoires. The string contains information on every mutation event in the form: "Position:pre_mutation_nucleotide,post_mutation_nucleotide" combined as: "Mutation1|Mutation2|Mutation3". For example: "171:t,a|186:g,a".

Value

List of dataframes. Each entry contains location and shm mutation info for a simulated sequence

Examples

```
shm_events_example<-c("171:t,a|186:g,a|287:g,a|310:t,c","","294:c,g|316:t,c|330:c,t")
shm_list<-shm_event_reconstruction(shm_events_example)
```

Index