Package 'immGLIPH'

June 20, 2026

Title Grouping of Lymphocyte Interactions by Paratope Hotspots

Version 0.99.5

Description An R implementation of the GLIPH and GLIPH2 algorithms for clustering T cell receptors (TCRs) predicted to bind the same HLA-restricted peptide antigen. Identifies specificity groups based on local (motif-based) and global (sequence-based) CDR3 similarities. Integrates with the scRepertoire ecosystem via immApex for single-cell immune repertoire analysis. Users should cite the original GLIPH algorithm papers: Glanville et al. (2017) <doi:10.1038/nature22976> and Huang et al. (2020) <doi:10.1038/s41587-020-0505-4>.

License MIT + file LICENSE

biocViews Software, ImmunoOncology, Clustering, SingleCell, Sequencing, Visualization

Depends R (>= 4.5.0)

Imports stringdist, igraph, BiocParallel, parallel, stringr, stats, utils, graphics, grDevices, viridis, visNetwork, plotfunctions, immApex

Suggests BiocFileCache, scRepertoire, SeuratObject, Seurat, SingleCellExperiment, testthat (>= 3.0.0), BiocStyle, knitr, rmarkdown

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/BorchLab/immGLIPH, https://github.com/BorchLab/scRepertoire, https://github.com/BorchLab/immApex

BugReports https://github.com/BorchLab/immGLIPH/issues

git_url https://git.bioconductor.org/packages/immGLIPH

git_branch devel

git_last_commit 17636de

git_last_commit_date 2026-05-20

Repository Bioconductor 3.24

Date/Publication 2026-06-19

Author Nick Borcherding [aut, cre]

Maintainer Nick Borcherding <ncborch@gmail.com>

Contents

`immGLIPH-package` *immGLIPH: Grouping of Lymphocyte Interactions by Paratope Hotspots*

Description

An R implementation of the GLIPH and GLIPH2 algorithms for clustering T cell receptors (TCRs) predicted to bind the same HLA-restricted peptide antigen. Identifies specificity groups based on local (motif-based) and global (sequence-based) CDR3 similarities. Integrates with the scRepertoire ecosystem via immApex for single-cell immune repertoire analysis. Users should cite the original GLIPH algorithm papers: Glanville et al. (2017) doi:10.1038/nature22976 and Huang et al. (2020) doi:10.1038/s4158702005054.

Author(s)

Maintainer: Nick Borcherding <ncborch@gmail.com>

References

Glanville, J. et al. (2017). Identifying specificity groups in the T cell receptor repertoire. *Nature*, 547, 94–98. doi:10.1038/nature22976

Huang, H. et al. (2020). Analyzing the Mycobacterium tuberculosis immune response by T-cell receptor clustering with GLIPH2 and genome-wide antigen screening. *Nature Biotechnology*, 38, 1194–1202. doi:10.1038/s4158702005054

See Also

Useful links:

- https://github.com/BorchLab/immGLIPH
- https://github.com/BorchLab/scRepertoire
- https://github.com/BorchLab/immApex
- Report bugs at https://github.com/BorchLab/immGLIPH/issues

`.check_existing_files` *Check for existing output files*

Description

Check for existing output files

Usage

```
.check_existing_files(result_folder, filenames)
```

Arguments

`result_folder`	Path to output folder
`filenames`	Character vector of filenames to check

Value

Logical indicating whether saving should proceed

`.cluster_gliph1` *GLIPH1-style clustering via igraph connected components*

Description

Builds a clone network from local and global edges, optionally filters by V-gene and donor constraints, constructs an igraph graph, and extracts connected components as convergence groups. Each component becomes one cluster named `CRG-<first_CDR3b>`.

Usage

```
.cluster_gliph1(
  clone_network,
  sequences,
  not_in_global_ids,
  seqs,
  vgene.info,
  patient.info,
  global_vgene,
  public_tcrs,
  cluster_min_size,
  verbose,
  BPPARAM
)
```

Arguments

`clone_network`	Data frame. Edge list with columns `V1`, `V2`, `type` (and optionally `tag`).
`sequences`	Data frame. Full sample sequences with at least `CDR3b` and optionally `TRBV`, `patient`.
`not_in_global_ids`	Integer vector. Indices of sequences without global neighbours (used to add singletons).
`seqs`	Character vector. Unique CDR3b sequences.
`vgene.info`	Logical. Whether V-gene info is available.
`patient.info`	Logical. Whether patient info is available.
`global_vgene`	Logical. Whether global edges require V-gene match.
`public_tcrs`	Logical. If `FALSE`, restrict edges to same donor.
`cluster_min_size`	Integer. Minimum cluster size to retain.
`verbose`	Logical. Print progress messages.
`BPPARAM`	A `BiocParallelParam` object for parallel evaluation.

Value

A list with:

cluster_properties Data frame with cluster_size, tag, members.

cluster_list Named list of data frames (member details).

clone_network Data frame of the final edge list.

save_cluster_list_df Data frame for saving cluster members.

`.cluster_gliph2` *GLIPH2-style clustering by individual motif/struct tags*

Description

Each local motif or global struct tag defines its own cluster. Edges are restricted by motif distance for local connections and by BLOSUM62 for global connections. Clusters are built as igraph components within each tag, and enrichment is reassessed per cluster.

Usage

```
.cluster_gliph2(
  local_res,
  global_res,
  sequences,
  local_similarities,
  global_similarities,
  global_vgene,
  all_aa_interchangeable,
  structboundaries,
  boundary_size,
  motif_distance_cutoff,
  cluster_min_size,
  boost_local_significance,
  verbose,
  BPPARAM
)
```

Arguments

`local_res`	Data frame. Selected motifs from local enrichment (must have columns `motif`, `start`, `stop`, `num_in_sample`, `num_in_ref`, `num_fold`, `fisher.score`, `members`).
`global_res`	Data frame. Global cluster list from `.global_fisher()` (columns: `cluster_tag`, `cluster_size`, `unique_CDR3b`, `num_in_ref`, `fisher.score`, `aa_at_position`, `TRBV`, `CDR3b`).
`sequences`	Data frame. Full sample data.
`local_similarities`	Logical. Whether local similarities were run.
`global_similarities`	Logical. Whether global similarities were found.
`global_vgene`	Logical. Restrict global edges to matching V-gene.

`all_aa_interchangeable`
: Logical. BLOSUM62 filtering.

`structboundaries`
: Logical. Whether boundary trimming is active.

`boundary_size`
: Integer. Boundary trim size.

`motif_distance_cutoff`
: Integer. Max positional distance for local motifs.

`cluster_min_size`
: Integer. Minimum cluster size.

`boost_local_significance`
: Logical. Whether to boost local p-values using germline N-nucleotide information.

`verbose`
: Logical. Print progress messages.

`BPPARAM`
: A `BiocParallelParam` object for parallel evaluation.

Value

A list with:

merged_clusters Data frame of cluster properties (type, tag, cluster_size, unique_cdr3_sample, unique_cdr3_ref, OvE, fisher.score, members).

cluster_list Named list of data frames with member details.

clone_network Data frame of edges (V1, V2, type, cluster_tag).

save_cluster_list_df Data frame for saving cluster member details.

`.coerce_numeric_cols` *Coerce character columns to numeric where possible*

Description

Coerce character columns to numeric where possible

Usage

```
.coerce_numeric_cols(df)
```

Arguments

`df`
: A data frame

Value

Data frame with numeric columns where appropriate

.extract_input *Extract TCR input data from various sources*

Description

Handles Seurat objects, SingleCellExperiment objects, combineTCR/combineBCR list output, data frames, and character vectors. Uses immApex::getIR() when single-cell objects are detected.

Usage

```
.extract_input(input, chains = "TRB")
```

Arguments

input	Input data (data frame, vector, Seurat, SCE, or list)
chains	Chain type for getIR extraction. Default "TRB".

Value

A data frame with standardized column names

.get_blosum_vec *Get BLOSUM62-compatible amino acid pairs*

Description

Returns a character vector of two-letter amino acid pairs whose BLOSUM62 substitution score is >= 0, derived from the full BLOSUM62 matrix in `immApex::immapex_blosum.pam.matrices`.

Usage

```
.get_blosum_vec()
```

Value

Character vector of AA pair strings (e.g. `"AA"`, `"AS"`).

.get_reference_list *Load reference list (internal, with caching)*

Description

Loads the reference list, using a package-level cache to avoid repeated disk reads or downloads within a single session.

Usage

```
.get_reference_list()
```

Value

The reference list.

.global_cutoff *Global similarity search using Hamming distance cutoff (GLIPH1.0 method)*

Description

Global similarity search using Hamming distance cutoff (GLIPH1.0 method)

Usage

```
.global_cutoff(
  seqs,
  motif_region,
  sequences,
  gccutoff,
  global_vgene,
  BPPARAM,
  verbose
)
```

Arguments

seqs	character vector. Unique CDR3b sequences filtered by C/F start/end (if applicable) and minimum length.
motif_region	character vector. The motif region of each sequence in seqs (i.e. with boundaries stripped if structboundaries is active).
sequences	data.frame. The full input data frame containing at least columns CDR3b and optionally TRBV.
gccutoff	numeric. Maximum Hamming distance for two sequences to be considered globally similar.
global_vgene	logical. If TRUE, global connections are restricted to sequence pairs that share a V gene.
BPPARAM	A BiocParallelParam object controlling parallel evaluation (e.g. BiocParallel::MulticoreParam
verbose	logical. If TRUE, progress messages are printed.

Value

A list with two elements:

edges A `data.frame` with columns `V1`, `V2`, and `type` (always `"global"`). Each row represents a pair of CDR3b sequences that are globally similar.

not_in_global_ids An integer vector of indices (into `seqs`) for sequences that have no global neighbour.

.global_cutoff_immapex *immApex-accelerated global cutoff via buildNetwork()*

Description

immApex-accelerated global cutoff via buildNetwork()

Usage

```
.global_cutoff_immapex(
  seqs,
  motif_region,
  sequences,
  gccutoff,
  global_vgene,
  verbose
)
```

Value

A list with edge data and excluded sequence IDs.

.global_cutoff_stringdist *stringdist + BiocParallel fallback for global cutoff*

Description

stringdist + BiocParallel fallback for global cutoff

Usage

```
.global_cutoff_stringdist(
  seqs,
  motif_region,
  sequences,
  gccutoff,
  global_vgene,
  BPPARAM,
  verbose
)
```

Value

A list with edge data and excluded sequence IDs.

`.global_fisher` *Global similarity search using structural matching and Fisher's test (GLIPH2)*

Description

Identifies globally similar CDR3b sequences by generating "struct" tags (motif region with one variable position replaced by `"%"`), then testing for enrichment of each struct in the sample set vs. a naive reference database using the hypergeometric distribution (one-sided Fisher's exact test). Optionally filters for BLOSUM62-compatible amino acid substitutions at the variable position.

Usage

```
.global_fisher(
  seqs,
  motif_region,
  sequences,
  refseqs,
  refseqs_motif_region,
  structboundaries,
  boundary_size,
  global_vgene,
  all_aa_interchangeable,
  BPPARAM,
  verbose
)
```

Arguments

`seqs`	Character vector. Unique CDR3b sequences from the sample.
`motif_region`	Character vector. Motif regions of the sample sequences (boundaries already stripped if applicable).
`sequences`	Data frame. Full sample data frame with at least columns `CDR3b` and (if applicable) `TRBV`.
`refseqs`	Data frame. Reference database with at least a `CDR3b` column.
`refseqs_motif_region`	Character vector. Motif regions of the reference sequences.
`structboundaries`	Logical. Whether structural boundaries are applied.
`boundary_size`	Integer. Number of AAs trimmed from each end.
`global_vgene`	Logical. If `TRUE`, restrict edges to pairs sharing a V-gene.
`all_aa_interchangeable`	Logical. If `FALSE`, only pairs whose variable-position amino acids have BLOSUM62 >= 0 are kept.
`BPPARAM`	A `BiocParallelParam` object controlling parallel evaluation.
`verbose`	Logical. Print progress messages.

Value

A list with elements:

cluster_list Data frame of struct clusters with columns `cluster_tag`, `cluster_size`, `unique_CDR3b`, `num_in_ref`, `fisher.score`, `aa_at_position`, `TRBV`, `CDR3b`.

global_similarities Logical indicating whether any global similarities were found.

`.load_reference` *Load and prepare reference database*

Description

Handles both named reference databases and user-provided data frames.

Usage

```
.load_reference(
  refdb_beta,
  accept_CF = TRUE,
  min_seq_length = 8,
  global_vgene = FALSE,
  vgene_stratify = FALSE,
  structboundaries = TRUE,
  boundary_size = 3,
  verbose = TRUE
)
```

Arguments

`refdb_beta`	Character name or data frame
`accept_CF`	Filter for C/F start/end
`min_seq_length`	Minimum sequence length
`global_vgene`	Whether V-gene info is needed
`vgene_stratify`	Whether V-gene stratification is needed
`structboundaries`	Whether to use structural boundaries
`boundary_size`	Boundary size
`verbose`	Print messages

Value

A list with refseqs (character vector of CDR3b), ref_vgenes, refseqs_motif_region, and the full reference data frame

`.local_fisher` *Local similarity detection using Fisher's exact test*

Description

Implements the GLIPH2-style Fisher's exact test approach for detecting locally enriched CDR3 motifs in a sample set compared to a reference database.

Usage

```
.local_fisher(
  motif_region,
  refseqs_motif_region,
  seqs,
  refseqs,
  sequences,
  motif_length,
  kmer_mindepth,
  lcminp,
  lcminove,
  discontinuous_motifs,
  motif_distance_cutoff,
  BPPARAM,
  verbose
)
```

Arguments

`motif_region`	Character vector. Motif regions extracted from sample sequences (e.g. CDR3b with structural boundaries trimmed).
`refseqs_motif_region`	Character vector. Motif regions extracted from reference sequences.
`seqs`	Character vector. Unique sample CDR3b sequences.
`refseqs`	Data frame. Reference database containing at least a `CDR3b` column.
`sequences`	Data frame. Sample data containing at least a `CDR3b` column.
`motif_length`	Numeric vector. Lengths of motifs to search for (e.g. `2:4`).
`kmer_mindepth`	Numeric. Minimum number of times a motif must be observed in the sample set to be considered.
`lcminp`	Numeric. Maximum p-value threshold for a motif to be considered significant.
`lcminove`	Numeric or numeric vector. Minimum fold change threshold(s) for enrichment filtering. If a vector, each element corresponds to the matching entry in `motif_length`.
`discontinuous_motifs`	Logical. Whether to include discontinuous motifs in the search.
`motif_distance_cutoff`	Numeric. Maximum positional distance for motifs to be grouped together. Not used directly in this function but kept for interface consistency.
`BPPARAM`	A `BiocParallelParam` object specifying the parallel backend. Defaults to `BiocParallel::SerialP`
`verbose`	Logical. If `TRUE`, print status messages via `message()`.

Value

A list with two elements:

selected_motifs Data frame of motifs passing significance and enrichment filters.

all_motifs Data frame of all motifs found in the sample set with associated statistics.

`.local_rrs` *Local similarity detection via repeated random sampling (GLIPH1-style)*

Description

Identifies enriched motifs in TCR CDR3b sequences by comparing motif frequencies in the sample set against repeated random subsamples drawn from a naive reference database.

Usage

```
.local_rrs(
  motif_region,
  refseqs_motif_region,
  seqs,
  sequences,
  motif_length,
  sim_depth,
  kmer_mindepth,
  lcminp,
  lcminove,
  discontinuous_motifs,
  cdr3_len_stratify,
  vgene_stratify,
  BPPARAM,
  verbose,
  motif_lengths_list,
  ref_motif_lengths_id_list,
  motif_region_vgenes_list,
  ref_motif_vgenes_id_list,
  lengths_vgenes_list,
  ref_lengths_vgenes_list
)
```

Arguments

`motif_region`	Character vector. Motif regions extracted from sample sequences.
`refseqs_motif_region`	Character vector. Motif regions extracted from reference sequences.
`seqs`	Character vector. Unique sample CDR3b sequences.
`sequences`	Data frame. Must contain columns `CDR3b` and `TRBV`.
`motif_length`	Numeric vector. Lengths of motifs to search for.
`sim_depth`	Numeric. Number of repeated random sampling iterations.

`kmer_mindepth` Numeric. Minimum number of times a motif must appear in the sample set to be considered.

`lcminp` Numeric. Maximum p-value threshold for a motif to be selected.

`lcminove` Numeric vector. Minimum fold-change (observed / expected) threshold(s). If a single value, the same threshold is applied to all motif lengths.

`discontinuous_motifs` Logical. Whether to include discontinuous motifs.

`cdr3_len_stratify` Logical. Whether to stratify random subsamples by CDR3 length distribution.

`vgene_stratify` Logical. Whether to stratify random subsamples by V-gene usage distribution.

`BPPARAM` A `BiocParallelParam` object controlling parallel execution (default: `BiocParallel::bpparam()`).

`verbose` Logical. If `TRUE`, emit progress messages.

`motif_lengths_list` List. Pre-computed CDR3 length counts from the sample.

`ref_motif_lengths_id_list` List. Pre-computed reference indices by CDR3 length.

`motif_region_vgenes_list` List. Pre-computed V-gene counts from the sample.

`ref_motif_vgenes_id_list` List. Pre-computed reference indices by V-gene.

`lengths_vgenes_list` List. Pre-computed joint CDR3-length / V-gene counts from the sample.

`ref_lengths_vgenes_list` List. Pre-computed reference indices by joint CDR3-length / V-gene.

Value

A list with three elements:

sample_log Data frame. Rows are `"Discovery"` followed by one row per simulation; columns are motifs. Values are motif counts.

selected_motifs Data frame of motifs passing enrichment filters with columns `motif`, `counts`, `num_in_ref`, `avgRef`, `topRef`, `OvE`, `p.value`.

all_motifs Data frame with the same columns as `selected_motifs` but for every motif found.

`.parse_sequences` *Parse and filter sequences data frame*

Description

Consolidates the input preparation logic into a single function.

Usage

```
.parse_sequences(
  cdr3_sequences,
  accept_CF = TRUE,
  min_seq_length = 8,
  global_vgene = FALSE,
  vgene_stratify = FALSE,
  verbose = TRUE
)
```

Arguments

cdr3_sequences	Data frame or vector of sequences
accept_CF	Accept only sequences starting with C and ending with F
min_seq_length	Minimum sequence length
global_vgene	Whether global vgene matching is required
vgene_stratify	Whether vgene stratification is required
verbose	Print notifications

Value

A list with sequences data frame and info flags

.prepare_motif_region *Prepare motif regions from sequences*

Description

Extracts the motif region by removing boundary amino acids.

Usage

```
.prepare_motif_region(seqs, structboundaries, boundary_size)
```

Arguments

seqs	Character vector of sequences
structboundaries	Whether to trim boundaries
boundary_size	Number of AAs to trim from each end

Value

Character vector of motif regions

.prepare_result_folder *Prepare result folder path*

Description

Prepare result folder path

Usage

```
.prepare_result_folder(result_folder)
```

Arguments

result_folder	Path string

Value

Normalized path or "" if no saving

.save_parameters *Save parameter list to file*

Description

Save parameter list to file

Usage

```
.save_parameters(parameters, result_folder)
```

Arguments

parameters	Named list of parameters
result_folder	Path to output folder

Value

NULL (invisibly). Called for side effect of writing file.

.setup_parallel *Create a BiocParallel backend parameter object*

Description

Returns a BiocParallelParam suitable for the current platform and requested number of cores. On Unix-like systems (macOS, Linux) with more than one core, a MulticoreParam is returned. On Windows or when only one core is requested, a SerialParam is used instead.

Usage

```
.setup_parallel(n_cores)
```

Arguments

n_cores	Number of cores. NULL auto-detects.

Details

The core count is clamped to 2 when the _R_CHECK_LIMIT_CORES_ environment variable is set (as during R CMD check).

Value

A BiocParallelParam object.

.standardize_colnames *Standardize column names from various input formats*

Description

Maps alternative column names to canonical GLIPH names. Supports scRepertoire, immApex getIR, and native GLIPH formats.

Usage

```
.standardize_colnames(df)
```

Arguments

df	A data frame

Value

Data frame with standardized column names

`.validate_params` *Validate parameters for runGLIPH*

Description

Consolidates all parameter validation into one function.

Usage

```
.validate_params(
  refdb_beta = "human_v2.0_CD48",
  v_usage_freq = NULL,
  cdr3_length_freq = NULL,
  ref_cluster_size = "original",
  sim_depth = 1000,
  lcminp = 0.01,
  lcminove = c(1000, 100, 10),
  kmer_mindepth = 3,
  accept_CF = TRUE,
  min_seq_length = 8,
  gccutoff = NULL,
  structboundaries = TRUE,
  boundary_size = 3,
  motif_length = c(2, 3, 4),
  local_similarities = TRUE,
  global_similarities = TRUE,
  cluster_min_size = 2,
  hla_cutoff = 0.1,
  n_cores = 1,
  motif_distance_cutoff = 3,
  discontinuous_motifs = FALSE,
  all_aa_interchangeable = FALSE,
  boost_local_significance = FALSE
)
```

Arguments

`refdb_beta`	Reference database
`v_usage_freq`	V-gene usage frequency data frame
`cdr3_length_freq`	CDR3 length frequency data frame
`ref_cluster_size`	Cluster size reference type
`sim_depth`	Simulation depth
`lcminp`	Local convergence min p-value
`lcminove`	Local convergence min OvE
`kmer_mindepth`	Minimum kmer observations
`accept_CF`	Accept C/F start/end only

`min_seq_length` Minimum sequence length

`gccutoff` Global convergence cutoff

`structboundaries` Use structural boundaries

`boundary_size` Boundary size in AAs

`motif_length` Motif lengths to search

`local_similarities` Search local similarities

`global_similarities` Search global similarities

`cluster_min_size` Minimum cluster size

`hla_cutoff` HLA significance cutoff

`n_cores` Number of cores

`motif_distance_cutoff` Motif distance cutoff (GLIPH2)

`discontinuous_motifs` Allow discontinuous motifs

`all_aa_interchangeable` BLOSUM62 filtering

`boost_local_significance` Germline boost

Value

List of validated (and possibly adjusted) parameter values

.valid_reference_names *Valid built-in reference database names*

Description

Valid built-in reference database names

Usage

```
.valid_reference_names()
```

Value

Character vector of valid reference database names.

clusterScoring *Score CDR3 clusters using the GLIPH or GLIPH2 algorithm*

Description

Calculates scores for CDR3 clusters following the GLIPH and GLIPH2 scoring procedures. Depending on the information provided, a final score is computed from up to five cluster properties: cluster size, enrichment of CDR3 lengths, enrichment of V genes, enrichment of clonal expansions, and enrichment of common HLA alleles.

Usage

```
clusterScoring(
  cluster_list,
  cdr3_sequences,
  refdb_beta = "human_v2.0_CD48",
  v_usage_freq = NULL,
  cdr3_length_freq = NULL,
  ref_cluster_size = "original",
  gliph_version = 1,
  sim_depth = 1000,
  hla_cutoff = 0.1,
  n_cores = 1
)
```

Arguments

cluster_list	A list where each element contains a data.frame of CDR3b sequences and additional information needed for scoring. Corresponds to the $cluster_list element returned by runGLIPH.
cdr3_sequences	A vector or data.frame of CDR3 sequences and optional metadata. The columns must be named as specified below in arbitrary order: "CDR3b" CDR3 sequences of beta chains. "TRBV" Optional. V-genes of beta chains. "patient" Optional. Donor index for the corresponding sequence, composed of a donor identifier and an optional experimental condition separated by a colon (e.g., 09/0410:MtbLys). Only the identifier before the colon is used for HLA scoring. "HLA" Optional. Comma-separated HLA alleles for the corresponding donor in standard notation (e.g., DPA1*01:03). Information after the colon in each allele is ignored during HLA scoring. "counts" Optional. Clone frequency.
refdb_beta	A character string or data.frame specifying the reference database. When a data.frame is supplied, CDR3b sequences must be in the first column and V-gene information (if available) in the second column. Built-in databases include "human_v1.0_CD4", "human_v1.0_CD8", "human_v1.0_CD48", "human_v2.0_CD4", "human_v2.0_CD8", "human_v2.0_CD48", "mouse_v1.0_CD4", "mouse_v1.0_CD8", "mouse_v1.0_CD48", and the legacy alias "gliph_reference" (= "human_v1.0_CD48"). See reference_list for details. **Default:** "human_v2.0_CD48"

v_usage_freq A data.frame with V-gene alleles in the first column and their naive-repertoire frequencies in the second column. **Default:** NULL

cdr3_length_freq
A data.frame with CDR3 lengths in the first column and their naive-repertoire frequencies in the second column. **Default:** NULL

ref_cluster_size
A character string defining which cluster-size probabilities to use for scoring.

"original" Standard probabilities from the original algorithm, constant across sample sizes.

"simulated" Probabilities estimated for different sample sizes via a 500-step simulation using random sequences from the reference database.

Default: "original"

gliph_version A numeric value indicating the algorithm version.

1 GLIPH scoring (product of individual scores multiplied by 0.064).

2 GLIPH2 scoring (product of individual scores only).

Default: 1

sim_depth A numeric value for simulated resampling depth in non-parametric convergence significance tests. Higher values increase runtime but improve reproducibility. **Default:** 1000

hla_cutoff A numeric threshold below which HLA probability scores are considered significant. **Default:** 0.1

n_cores A numeric value for the number of cores to use. When NULL, it is set to the number of available cores minus two. **Default:** 1

Value

A data.frame of cluster scoring results. The first column contains the total score and additional columns contain up to five individual scores (cluster size, CDR3 length enrichment, V-gene enrichment, clonal expansion enrichment, and common HLA enrichment).

References

Glanville, Jacob, et al. "Identifying specificity groups in the T cell receptor repertoire." Nature 547.7661 (2017): 94.

https://github.com/immunoengineer/gliph

Examples

```
utils::data("gliph_input_data")
ref_df <- gliph_input_data[, c("CDR3b", "TRBV")]

res <- runGLIPH(cdr3_sequences = gliph_input_data[seq_len(200), ],
                refdb_beta = ref_df,
                sim_depth = 100,
                n_cores = 1)

scoring_results <- clusterScoring(
    cluster_list = res$cluster_list,
    cdr3_sequences = gliph_input_data[seq_len(200), ],
    refdb_beta = ref_df,
    gliph_version = 1,
```

```
    sim_depth = 100,
    n_cores = 1)
```

---

deNovoTCRs *Generate de novo TCR sequences*

Description

De novo generation of CDR3 sequences based on GLIPH or GLIPH2 clustering results. Using the position-specific abundance of amino acids in the CDR3 region of sequences within a convergence group, artificial sequences are simulated following the approach established in Glanville et al. The generated sequences are scored by a positional weight matrix (PWM) derived from the convergence group, and optionally normalized against a reference database. The top-scoring sequences are returned.

Usage

```
deNovoTCRs(
  convergence_group_tag,
  result_folder = "",
  clustering_output = NULL,
  refdb_beta = "gliph_reference",
  normalization = FALSE,
  accept_sequences_with_C_F_start_end = TRUE,
  sims = 1e+05,
  num_tops = 1000,
  min_length = 10,
  make_figure = FALSE,
  n_cores = 1
)
```

Arguments

`convergence_group_tag`
: Character. Tag of the convergence group to use for prediction.

`result_folder`
: Character. Path to the folder containing clustering output files and where results will be saved. If the value is `""`, results are not saved to disk and the clustering output must be provided via `clustering_output`. **Default:** `""`

`clustering_output`
: List. The output list from `runGLIPH`. Required when `result_folder` is `""`. **Default:** `NULL`

`refdb_beta`
: Character or data.frame. Specifies the reference database to use. When a data.frame is provided, the first column should contain CDR3b sequences and the second column (optional) should contain V genes. The following keyword can be used to select a built-in database:

  - `"gliph_reference"`: 162,165 CDR3b sequences of naive human CD4+ or CD8+ T cells from two individuals (GLIPH paper).

  Default: `"gliph_reference"`

`normalization` Logical. If `TRUE`, calculated scores are normalized to the reference database. The returned value represents the probability that a reference sequence has a score greater than or equal to the sample sequence score. When V gene information is available, only sequences with identical V genes are compared. **Default:** `FALSE`

`accept_sequences_with_C_F_start_end` Logical. If `TRUE`, only sequences beginning with cysteine (C) and ending with phenylalanine (F) are accepted. **Default:** `TRUE`

`sims` Numeric. Number of de novo CDR3 sequences to generate. **Default:** `100000`

`num_tops` Numeric. Number of top-scoring de novo sequences to return. **Default:** `1000`

`min_length` Numeric. Minimum CDR3 sequence length; also determines the number of N-terminal positions used for PWM scoring. **Default:** `10`

`make_figure` Logical. Whether to plot the `num_tops` best-scoring de novo sequences as a function of rank. **Default:** `FALSE`

`n_cores` Numeric. Number of cores for parallel computation. If `NULL`, the number of available cores minus two is used. **Default:** `1`

Value

A `list` with the following elements:

de_novo_sequences A `data.frame` of the `num_tops` best-scoring generated sequences and their corresponding scores.

sample_sequences_scores A `data.frame` of the convergence group sequences and their corresponding scores.

cdr3_length_probability A `data.frame` with each observed CDR3 length and its probability of occurrence in the convergence group. The length distribution of generated sequences mirrors this distribution.

PWM_Scoring A `data.frame` containing the positional weight matrix used for scoring. Columns represent amino acids and rows represent positions relative to the N-terminus.

PWM_Prediction A `list` of `data.frames` containing the positional weight matrices used for sequence generation, one per observed CDR3 length. Columns represent amino acids and rows represent positions relative to the N-terminus.

If `result_folder` is specified, a tab-delimited file named `<convergence_group_tag>_de_novo.txt` is also written to disk.

References

Glanville, Jacob, et al. "Identifying specificity groups in the T cell receptor repertoire." Nature 547.7661 (2017): 94.

https://github.com/immunoengineer/gliph

Examples

```
# Build a minimal clustering output to demonstrate deNovoTCRs
fake_cluster <- data.frame(
  CDR3b = c("CASSLAPGATNEKLFF", "CASSLAPGGTNEKLFF",
            "CASSLAPGDTNEKLFF", "CASSLAPGETNEKLFF",
            "CASSLAPGANEKLFF",  "CASSLAPGVTNEKLFF"),
  TRBV  = rep("TRBV5-1", 6),
  stringsAsFactors = FALSE
```

```
)
fake_output <- list(cluster_list = list("motif-LAP" = fake_cluster))
ref_seqs <- fake_cluster[, c("CDR3b", "TRBV")]
new_seqs <- deNovoTCRs(
  convergence_group_tag = "motif-LAP",
  clustering_output = fake_output,
  refdb_beta = ref_seqs,
  sims = 100,
  num_tops = 10,
  min_length = 8,
  make_figure = FALSE,
  n_cores = 1
)
```

`findMotifs` *Find continuous and discontinuous sequence motifs*

Description

Searches a character vector of amino acid sequences for k-mer motifs and returns their frequencies. Both continuous and discontinuous (gapped) motifs are supported. When **immApex** (>= 2.0.0) is installed, the C++-accelerated `immApex::calculateMotif()` backend is used automatically for improved performance; otherwise the function falls back to a pure-R implementation based on `qgrams`.

Usage

```
findMotifs(seqs, q = 2:4, kmer_mindepth = NULL, discontinuous = FALSE)
```

Arguments

`seqs`	A character vector of amino acid sequences in which motifs will be identified and counted.
`q`	A numeric vector of motif lengths to search for. **Default:** `2:4`.
`kmer_mindepth`	The minimum number of times a k-mer must be observed in `seqs` for it to be included in the output. **Default:** `NULL` (no filtering).
`discontinuous`	Whether to include discontinuous (gapped) motifs in the search. **Default:** `FALSE`.

Value

A `data.frame` with two columns: `motif` (the k-mer string) and `V1` (the observed frequency).

Examples

```
utils::data("gliph_input_data")
sample_seqs <- as.character(gliph_input_data$CDR3b)
res <- findMotifs(seqs = sample_seqs)
```

getGLIPHreference *Get or download the immGLIPH reference list*

Description

Downloads the reference repertoire data from Zenodo on first use and caches locally via **BiocFileCache**. Subsequent calls load from the cache without re-downloading.

Usage

```
getGLIPHreference(force_download = FALSE, verbose = TRUE)
```

Arguments

`force_download`	Logical. If `TRUE`, re-download even if cached. **Default:** `FALSE`
`verbose`	Logical. Print messages. **Default:** `TRUE`

Details

The cached file contains a named `list` with entries for each built-in reference database (see `.valid_reference_names`).

Value

A named `list` of reference databases. Each element is a list with `refseqs`, `vgene_frequencies`, and `cdr3_length_frequencies`.

Examples

```
# Available reference database names
c("human_v1.0_CD4", "human_v1.0_CD8", "human_v1.0_CD48",
  "human_v2.0_CD4", "human_v2.0_CD8", "human_v2.0_CD48",
  "mouse_v1.0_CD4", "mouse_v1.0_CD8", "mouse_v1.0_CD48",
  "gliph_reference")


ref <- getGLIPHreference()
names(ref)
head(ref[["human_v2.0_CD48"]]$refseqs)
```

getRandomSubsample *Draw a stratified random subsample from the reference repertoire*

Description

Draws a random subset of reference motif regions with the same size as the sample set. When `cdr3_len_stratify` and/or `vgene_stratify` are enabled, the function preserves the CDR3 length and/or V-gene distribution of the sample in the subsample. This is used internally by the repeated random sampling (RRS) local-similarity method in `runGLIPH`.

Usage

```
getRandomSubsample(
  cdr3_len_stratify = FALSE,
  vgene_stratify = FALSE,
  refseqs_motif_region,
  motif_region,
  motif_lengths_list,
  ref_motif_lengths_id_list,
  motif_region_vgenes_list,
  ref_motif_vgenes_id_list,
  ref_lengths_vgenes_list,
  lengths_vgenes_list
)
```

Arguments

`cdr3_len_stratify`
: Whether to preserve the CDR3 length distribution. **Default:** `FALSE`

`vgene_stratify`
: Whether to preserve the V-gene distribution. **Default:** `FALSE`

`refseqs_motif_region`
: Character vector of reference motif regions.

`motif_region`
: Character vector of sample motif regions.

`motif_lengths_list`
: Named list mapping CDR3 lengths to their frequency in `motif_region`. Required when `cdr3_len_stratify = TRUE`.

`ref_motif_lengths_id_list`
: Named list mapping CDR3 lengths to indices in `refseqs_motif_region`. Required when `cdr3_len_stratify = TRUE`.

`motif_region_vgenes_list`
: Named list mapping V-genes to their frequency in `motif_region`. Required when `vgene_stratify = TRUE`.

`ref_motif_vgenes_id_list`
: Named list mapping V-genes to indices in `refseqs_motif_region`. Required when `vgene_stratify = TRUE`.

`ref_lengths_vgenes_list`
: Nested list mapping CDR3 length x V-gene combinations to indices in `refseqs_motif_region`. Required when both stratification flags are `TRUE`.

`lengths_vgenes_list`
: Nested list mapping CDR3 length x V-gene combinations to their frequency in the sample. Required when both stratification flags are `TRUE`.

Value

A character vector of length `length(motif_region)` drawn from `refseqs_motif_region`.

Examples

```
ref_seqs <- c("ASSG", "ASSD", "ASSE", "ASSF", "ASSK", "ASSL")
sample_seqs <- c("ASSG", "ASSF", "ASSL")
sub <- getRandomSubsample(
  refseqs_motif_region = ref_seqs,
```

```
  motif_region = sample_seqs
)
```

`gliph_input_data` *Example TCR input data*

Description

A `data.frame` of 365 TRB CDR3 sequences with V-gene and patient annotations, derived from the **scRepertoire** example dataset (Yost et al. 2021). The data were extracted from the `gliph_sce` SingleCellExperiment object using `immApex::getIR()`.

Usage

```
data(gliph_input_data)
```

Format

A data.frame with 365 rows and 3 columns (CDR3b, TRBV, patient).

Details

`CDR3b` Amino acid sequence of the TRB CDR3 region.

`TRBV` TRBV gene name (e.g. `"TRBV9"`).

`patient` Patient/sample identifier (e.g. `"P17B"`, `"P19L"`).

Source

Yost, K. E. et al. Clonal replacement of tumor-specific T cells following PD-1 blockade. *Nature Medicine* 25, 1251–1259 (2019).

Built from **scRepertoire** example data; see `data-raw/build_example_data.R`.

See Also

`gliph_sce` for the parent SingleCellExperiment object, `runGLIPH` for the main analysis function.

`gliph_sce` *Example SingleCellExperiment with TCR clonal information*

Description

A `SingleCellExperiment` object containing 2,000 genes across 500 cells, with T-cell receptor clonotype information stored in the `colData`. Built from the **scRepertoire** example dataset using `combineTCR()` and `combineExpression()`.

Usage

```
data(gliph_sce)
```

Format

A SingleCellExperiment with 2000 genes and 500 cells.

Details

The `colData` includes scRepertoire columns such as `CTaa` (amino acid clonotype), `CTgene` (gene-level clonotype), `CTnt` (nucleotide clonotype), `CTstrict` (strict clonotype), and clone frequency/proportion columns. These can be parsed by `immApex::getIR()` to extract chain-specific TCR data.

This object demonstrates how to pass a SingleCellExperiment directly to `runGLIPH`.

Source

Yost, K. E. et al. Clonal replacement of tumor-specific T cells following PD-1 blockade. *Nature Medicine* 25, 1251–1259 (2019).

Built from **scRepertoire** example data; see `data-raw/build_example_data.R`.

See Also

`gliph_input_data` for a plain data.frame extracted from this object, `runGLIPH` for the main analysis function.

---

gTRB *Germline TCR-beta CDR3 fragments*

Description

A `list` of three `data.frames` containing germline-encoded fragments of V (`gTRV`), D (`gTRD`), and J (`gTRJ`) gene segments that may appear in the CDR3 region. These fragments are used by the GLIPH2 algorithm to identify germline-encoded sequence segments.

Usage

```
data(gTRB)
```

Format

A list of 3 data.frames: gTRV, gTRD, and gTRJ.

Source

Lefranc, M.-P. IMGT, the international ImMunoGeneTics database. *Nucl. Acids Res.* 29, 207–209 (2001).

`loadGLIPH` *Load saved GLIPH results from disk*

Description

Reads the tab-delimited output files produced by `runGLIPH` (when `result_folder` was specified) and reconstructs the same list structure that `runGLIPH()` returns.

Usage

```
loadGLIPH(result_folder = "")
```

Arguments

`result_folder`	Path to the folder containing the saved GLIPH output files.

Value

A `list` with the same structure as the return value of `runGLIPH`, including elements such as `cluster_list`, `cluster_properties`, `motif_enrichment`, `connections`, and `parameters`.

Examples

```
utils::data("gliph_input_data")
ref_df <- gliph_input_data[, c("CDR3b", "TRBV")]
tmp_dir <- tempfile("gliph_out_")
res <- runGLIPH(
  cdr3_sequences = gliph_input_data[seq_len(200), ],
  method = "gliph1",
  refdb_beta = ref_df,
  result_folder = tmp_dir,
  sim_depth = 50,
  n_cores = 1
)
reloaded <- loadGLIPH(result_folder = tmp_dir)
unlink(tmp_dir, recursive = TRUE)
```

`plotNetwork` *Visualize TCR convergence group network*

Description

Uses the visNetwork package to build an interactive network graph from the clustering results produced by `runGLIPH`. Nodes represent individual CDR3b sequences and edges encode local or global sequence similarities. The resulting visualization is fully interactive: scroll to zoom, hover over a node for details, and click a node to highlight its direct neighbors.

Usage

```
plotNetwork(
  clustering_output = NULL,
  result_folder = "",
  show_additional_columns = NULL,
  color_info = "total.score",
  color_palette = viridis::viridis,
  local_edge_color = "orange",
  global_edge_color = "#68bceb",
  size_info = NULL,
  absolute_size = FALSE,
  cluster_min_size = 3,
  n_cores = 1
)
```

Arguments

`clustering_output`
: Output list returned by `runGLIPH`. **Default:** `NULL`

`result_folder`
: Path to the folder containing saved GLIPH output files. When a non-empty path is supplied the results are loaded from disk and `clustering_output` is ignored. **Default:** `""`

`show_additional_columns`
: Character vector of extra column names whose values should be displayed in the node tooltips. Column names from the original `cdr3_sequences` data frame and from `clustering_output$cluster_properties` are accepted. **Default:** `NULL`

`color_info`
: Column name used to colour the nodes. Accepts any column from the input `cdr3_sequences` or `clustering_output$cluster_properties`. Set to `"none"` to colour all nodes grey, or `"color"` to use pre-assigned colour values stored in that column. For numeric columns the viridis palette is applied automatically (purple = low, yellow = high). **Default:** `"total.score"`

`color_palette`
: A function that accepts a single integer `n` and returns `n` colour values. **Default:** `viridis::viridis`

`local_edge_color`
: Colour applied to edges representing local similarities. **Default:** `"orange"`

`global_edge_color`
: Colour applied to edges representing global similarities. **Default:** `"#68bceb"`

`size_info`
: Column name whose numeric values determine node sizes. Accepts columns from `cdr3_sequences` or `clustering_output$cluster_properties`. **Default:** `NULL`

`absolute_size`
: If `TRUE` the raw values from the `size_info` column are used as node sizes; otherwise the values are linearly scaled to the range 12–20. **Default:** `FALSE`

`cluster_min_size`
: Minimum number of members a cluster must contain to be included in the plot. **Default:** `3`

`n_cores`
: Number of cores for parallel processing. When `NULL` the number of available cores minus two is used. **Default:** `1`

Value

A visNetwork object containing the interactive network graph.

Examples

```
utils::data("gliph_input_data")
ref_df <- gliph_input_data[, c("CDR3b", "TRBV")]
res <- runGLIPH(cdr3_sequences = gliph_input_data[seq_len(200),],
                method = "gliph1",
                refdb_beta = ref_df,
                sim_depth = 100,
                n_cores = 1)

plotNetwork(clustering_output = res,
            n_cores = 1)
```

---

reference_list *GLIPH reference repertoire list (external data)*

Description

A named list of naive TCR repertoire reference databases used for motif enrichment testing and cluster scoring. The data is **not** bundled with the package; it is downloaded on first use from Zenodo and cached locally via **BiocFileCache** (see getGLIPHreference).

Format

NULL. Data is downloaded on first use via getGLIPHreference.

Details

Each element is itself a list with three components:

refseqs A data.frame with columns CDR3b (amino acid sequence) and TRBV (V-gene name).

vgene_frequencies A data.frame with columns vgene and freq giving the relative frequency of each V gene in the reference repertoire.

cdr3_length_frequencies A data.frame with columns len and freq giving the relative frequency of each CDR3 length in the reference repertoire.

The following named entries are available:

- "human_v1.0_CD4", "human_v1.0_CD8", "human_v1.0_CD48" – Glanville et al. (2017)
- "human_v2.0_CD4", "human_v2.0_CD8", "human_v2.0_CD48" – Huang et al. (2020)
- "mouse_v1.0_CD4", "mouse_v1.0_CD8", "mouse_v1.0_CD48" – Glanville et al. (2017)
- "gliph_reference" – Legacy alias for "human_v1.0_CD48"

Value

No return value. This documents the reference_list object which is downloaded at runtime by getGLIPHreference.

Source

Glanville, J. et al. Identifying specificity groups in the T cell receptor repertoire. *Nature* 547, 94–98 (2017).

Huang, H. et al. Analyzing the *Mycobacterium tuberculosis* immune response by T-cell receptor clustering with GLIPH2 and genome-wide antigen screening. *Nature Biotechnology* 38, 1194–1202 (2020).

Raw data downloaded from http://50.255.35.37:8080/tools.

See Also

`getGLIPHreference` to download or load the data, `runGLIPH` and `clusterScoring` which use the reference internally via the `refdb_beta` parameter.

`ref_cluster_sizes` *Cluster size probabilities in naive reference repertoire*

Description

A `list` with two elements providing expected cluster-size probabilities under the null model (no true convergence):

`original` Probabilities from the original GLIPH algorithm, applied uniformly across all sample sizes.

`simulated` Probabilities estimated from 500-step simulations at sample sizes of 125, 250, 500, 1000, 2000, 4000, 6000, 8000, and 10000 random reference sequences. During scoring the row closest to the actual sample size is used.

Usage

```
data(ref_cluster_sizes)
```

Format

A list with 2 elements: original and simulated.

Source

Glanville, J. et al. Identifying specificity groups in the T cell receptor repertoire. *Nature* 547, 94–98 (2017).

runGLIPH *Run the GLIPH or GLIPH2 TCR clustering algorithm*

Description

Unified entry point for the GLIPH/GLIPH2 algorithm for grouping T cell receptors by antigen specificity. The function identifies locally and globally similar CDR3b sequences, clusters them into convergence groups, and scores each group for biological relevance.

Usage

```
runGLIPH(
  cdr3_sequences,
  method = c("gliph2", "gliph1", "custom"),
  chains = "TRB",
  result_folder = "",
  refdb_beta = "human_v2.0_CD48",
  v_usage_freq = NULL,
  cdr3_length_freq = NULL,
  ref_cluster_size = "original",
  sim_depth = 1000,
  lcminp = 0.01,
  lcminove = c(1000, 100, 10),
  kmer_mindepth = 3,
  accept_CF = TRUE,
  min_seq_length = 8,
  gccutoff = NULL,
  structboundaries = TRUE,
  boundary_size = 3,
  motif_length = c(2, 3, 4),
  local_similarities = TRUE,
  global_similarities = TRUE,
  local_method = NULL,
  global_method = NULL,
  clustering_method = NULL,
  scoring_method = NULL,
  cluster_min_size = 2,
  hla_cutoff = 0.1,
  n_cores = 1,
  motif_distance_cutoff = 3,
  discontinuous_motifs = FALSE,
  all_aa_interchangeable = FALSE,
  boost_local_significance = FALSE,
  global_vgene = FALSE,
  cdr3_len_stratify = FALSE,
  vgene_stratify = FALSE,
  public_tcrs = TRUE,
  vgene_match = "none",
  scoring_sim_depth = 1000,
  verbose = TRUE
)
```

Arguments

cdr3_sequences	Input data containing CDR3b amino acid sequences. Accepts a character vector, a `data.frame` with columns described below, a `Seurat` object, a `SingleCellExperiment` object, or a list returned by `scRepertoire::combineTCR()/combineBCR()`. When a `data.frame` is supplied, the following column names are recognized (alternative names in parentheses are mapped automatically): **CDR3b** (cdr3, cdr3_aa, CDR3.beta, junction_aa) Required. CDR3 beta-chain amino acid sequences. **TRBV** (v_gene, v.gene, Vgene, v_call) Optional. V-gene usage. **patient** (sample, donor, sample_id) Optional. Donor index. **HLA** (hla, HLA_alleles) Optional. HLA alleles, comma-separated. **counts** (frequency, clone_count, cloneCount) Optional. Clone frequency.
method	Character. Algorithm preset to use. `"gliph2"` Fisher-based local and global similarity, GLIPH2-style isolated clustering and scoring. `"gliph1"` Repeated random sampling for local similarity, Hamming distance cutoff for global similarity, GLIPH1-style connected-component clustering. `"custom"` All parameters can be set independently. **Default:** `"gliph2"`
chains	Character. Chain type for extraction from `Seurat` or `SingleCellExperiment` objects via `immApex::getIR()`. **Default:** `"TRB"`
result_folder	Character. Path to output folder. If `""`, results are not saved to disk. **Default:** `""`
refdb_beta	Character or `data.frame`. Reference database for motif enrichment testing. Built-in databases include `"human_v1.0_CD4"`, `"human_v1.0_CD8"`, `"human_v1.0_CD48"`, `"human_v2.0_CD4"`, `"human_v2.0_CD8"`, `"human_v2.0_CD48"`, `"mouse_v1.0_CD4"`, `"mouse_v1.0_CD8"`, `"mouse_v1.0_CD48"`, and the legacy alias `"gliph_reference"` (= `"human_v1.0_CD48"`). Alternatively, supply a `data.frame` with CDR3b in the first column and optional V-gene in the second. See `reference_list` for details. **Default:** `"human_v2.0_CD48"`
v_usage_freq	`data.frame` or `NULL`. V-gene frequencies for scoring. If `NULL`, built-in defaults are used. **Default:** `NULL`
cdr3_length_freq	`data.frame` or `NULL`. CDR3 length frequencies for scoring. If `NULL`, built-in defaults are used. **Default:** `NULL`
ref_cluster_size	Character. Reference cluster size strategy. `"original"` Use the original sample size. `"simulated"` Use simulated cluster sizes. **Default:** `"original"`
sim_depth	Integer. Simulation depth for repeated random sampling (local method `"rrs"`) or cluster scoring. **Default:** `1000`
lcminp	Numeric. Local convergence maximum p-value threshold. **Default:** `0.01`
lcminove	Numeric vector. Local convergence minimum fold-change per motif length (lengths 2, 3, and 4 respectively). **Default:** `c(1000, 100, 10)`
kmer_mindepth	Integer. Minimum number of kmer observations required to consider a motif. **Default:** `3`

accept_CF Logical. If TRUE, accept only sequences starting with C and ending with F. **Default:** TRUE

min_seq_length Integer. Minimum CDR3b sequence length to retain. **Default:** 8

gccutoff Numeric or NULL. Global convergence Hamming distance cutoff (used when global_method = "cutoff"). If NULL, the cutoff is auto-selected based on sample size. **Default:** NULL

structboundaries Logical. If TRUE, trim structural boundaries from CDR3b sequences before motif search. **Default:** TRUE

boundary_size Integer. Number of positions to trim from each end when structboundaries = TRUE. **Default:** 3

motif_length Numeric vector. Motif lengths to search. **Default:** c(2, 3, 4)

local_similarities Logical. If TRUE, search for locally similar CDR3b sequences. **Default:** TRUE

global_similarities Logical. If TRUE, search for globally similar CDR3b sequences. **Default:** TRUE

local_method Character or NULL. Method for local similarity detection. If NULL, set by the method preset.

"fisher" Fisher exact test for motif enrichment.

"rrs" Repeated random sampling.

Default: NULL

global_method Character or NULL. Method for global similarity detection. If NULL, set by the method preset.

"fisher" Fisher exact test for struct enrichment.

"cutoff" Hamming distance cutoff.

Default: NULL

clustering_method Character or NULL. Clustering strategy. If NULL, set by the method preset.

"GLIPH1.0" Connected-component clustering.

"GLIPH2.0" Isolated clustering with merging.

Default: NULL

scoring_method Character or NULL. Scoring strategy. If NULL, set by the method preset.

"GLIPH1.0" GLIPH1-style scoring.

"GLIPH2.0" GLIPH2-style scoring.

Default: NULL

cluster_min_size Integer. Minimum number of unique CDR3b sequences required to retain a convergence group. **Default:** 2

hla_cutoff Numeric. Significance cutoff for HLA enrichment testing. **Default:** 0.1

n_cores Integer or NULL. Number of cores for parallel processing. If NULL, the number of available cores is auto-detected. **Default:** 1

motif_distance_cutoff Integer. Maximum positional distance between shared motifs for two CDR3b sequences to be linked (GLIPH2). **Default:** 3

discontinuous_motifs Logical. If TRUE, allow discontinuous motif patterns during local similarity search. **Default:** FALSE

all_aa_interchangeable
: Logical. If FALSE, BLOSUM62 filtering is applied to global similarities, restricting substitutions to biochemically similar amino acids. **Default:** FALSE

boost_local_significance
: Logical. If TRUE, boost local p-values using germline N-nucleotide insertion information. **Default:** FALSE

global_vgene
: Logical. If TRUE, restrict global similarity edges to pairs sharing the same V-gene. **Default:** FALSE

cdr3_len_stratify
: Logical. If TRUE, stratify random subsamples by CDR3 length (used with local_method = "rrs"). **Default:** FALSE

vgene_stratify
: Logical. If TRUE, stratify random subsamples by V-gene usage (used with local_method = "rrs"). **Default:** FALSE

public_tcrs
: Logical or character. Controls cross-donor edge filtering. For method = "gliph1" or "gliph2": if FALSE, restrict edges to same donor. For method = "custom":

  "all" Allow cross-donor edges for all similarity types.

  "local" Allow cross-donor edges for local only.

  "global" Allow cross-donor edges for global only.

  "none" Restrict all edges to same donor.

  Default: TRUE

vgene_match
: Character. V-gene matching requirement for custom clustering.

  "none" No V-gene matching required.

  "local" Require V-gene match for local edges.

  "global" Require V-gene match for global edges.

  "all" Require V-gene match for all edges.

  Default: "none"

scoring_sim_depth
: Integer. Simulation depth used specifically for convergence group scoring. **Default:** 1000

verbose
: Logical. If TRUE, print progress messages to the console. **Default:** TRUE

Value

A list with the following elements:

sample_log data.frame. Motif counts per simulation iteration (only present when local_method = "rrs").

motif_enrichment list with two elements:

- selected_motifs data.frame of significantly enriched motifs passing all thresholds.
- all_motifs data.frame of all evaluated motifs with enrichment statistics.

global_enrichment list. Global struct enrichment results (GLIPH2 only; NULL otherwise).

connections data.frame. Edge list representing the clone network.

cluster_properties data.frame. Convergence group properties and scores.

cluster_list Named list of data.frame objects with per-cluster member details.

parameters list. All input parameters used for the run.

References

Glanville, J. et al. (2017). Identifying specificity groups in the T cell receptor repertoire. *Nature*, 547, 94–98. doi:10.1038/nature22976

Huang, H. et al. (2020). Analyzing the Mycobacterium tuberculosis immune response by T-cell receptor clustering with GLIPH2 and genome-wide antigen screening. *Nature Biotechnology*, 38, 1194–1202. doi:10.1038/s4158702005054

Examples

```
utils::data("gliph_input_data")
ref_df <- gliph_input_data[, c("CDR3b", "TRBV")]
res <- runGLIPH(
  cdr3_sequences = gliph_input_data[seq_len(200), ],
  method = "gliph2",
  refdb_beta = ref_df,
  sim_depth = 50,
  n_cores = 1
)
```

Index