Package 'immunogenetr'

May 26, 2026

Type Package

Title A Comprehensive Toolkit for Clinical HLA Informatics

Version 1.3.0

Maintainer Nicholas Brown <nicholas.brown@pennmedicine.upenn.edu>

Description A comprehensive toolkit for clinical Human Leukocyte Antigen (HLA) informatics, built on 'tidyverse' <https://tidyverse.tidyverse.org/> principles and making use of genotype list string (GL string, Mack et al. (2023) <doi:10.1111/tan.15126>) for storing and computing HLA genotype data. Specific functionalities include: coercion of HLA data in tabular format to and from GL string; calculation of matching and mismatching in all directions, with multiple output formats; automatic formatting of HLA data for searching within a GL string; truncation of molecular HLA data to a specific number of fields; and reading HLA genotypes in HML files and extracting the GL string. This library is intended for research use. Any application making use of this package in a clinical setting will need to be independently validated according to local regulations.

License GPL (>= 3)

URL https://immunogenetr.org,
https://github.com/k96nb01/immunogenetr_package,
https://doi.org/10.1016/j.humimm.2025.111619

BugReports https://github.com/k96nb01/immunogenetr_package/issues

Depends R (>= 4.1.0)

Imports cli (>= 3.0.0), dplyr (>= 1.1.1), purrr (>= 1.0.2), rlang (>= 1.1.2), stringi (>= 1.7.0), stringr (>= 1.5.0), tibble (>= 3.1.3), tidyr (>= 1.3.0), tidyselect (>= 1.2.0), xml2 (>= 1.3.6)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

LazyData true

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Nicholas Brown [cre, aut] (ORCID:
<https://orcid.org/0000-0002-0046-2315>),
Busra Coskun [aut] (ORCID: <https://orcid.org/0009-0008-6828-3453>)

Repository CRAN

Date/Publication 2026-05-26 08:50:10 UTC

Contents

ambiguity_table_to_GLstring *ambiguity_table_to_GLstring*

Description

A function that converts a data table of HLA allele ambiguities (e.g. as created by 'GLstring_expand_longer' or 'GLstring_to_ambiguity_table') into a GL string format. The function processes the table by combining allele ambiguities, haplotypes, gene copies, and loci into a structured GL string.

Usage

```
ambiguity_table_to_GLstring(data, remove_duplicates = FALSE)
```

Arguments

data	A data frame containing columns that represent possible gene locations, loci, genotype ambiguities, genotypes, and haplotypes.
remove_duplicates	A logical value indicating if the function will check for duplicate entries at each step and remove them before assembling the final GL string. Useful if the ambiguity table has been altered, for example by truncating allele designations. Default is FALSE.

Value

A GL string representing the combined gene locations, loci, genotype ambiguities, genotypes, and haplotypes.

Examples

```
# Example data frame input
data <- tibble::tribble(
  ~value, ~entry, ~possible_gene_location,
  ~locus, ~genotype_ambiguity, ~genotype, ~haplotype, ~allele,
  "HLA-A*01:01:01:01", 1, 1,
  1, 1, 1, 1, 1,
  "HLA-A*01:02", 1, 1,
  1, 1, 1, 1, 2,
  "HLA-A*01:03", 1, 1,
  1, 1, 1, 1, 3,
  "HLA-A*01:95", 1, 1,
  1, 1, 1, 1, 4,
  "HLA-A*24:02:01:01", 1, 1,
  1, 1, 2, 1, 1,
  "HLA-A*01:01:01:01", 1, 1,
  1, 2, 1, 1, 1,
  "HLA-A*01:03", 1, 1,
  1, 2, 1, 1, 2,
```

```
  "HLA-A*24:03:01:01", 1, 1,
  1, 2, 2, 1, 1,
  "HLA-B*07:01:01", 1, 1,
  2, 1, 1, 1, 1,
  "B*15:01:01", 1, 1,
  2, 1, 2, 1, 1,
  "B*15:02:01", 1, 1,
  2, 1, 2, 1, 2,
  "B*07:03", 1, 1,
  2, 2, 1, 1, 1,
  "B*15:99:01", 1, 1,
  2, 2, 2, 1, 1,
  "HLA-DRB1*03:01:02", 1, 1,
  3, 1, 1, 1, 1,
  "HLA-DRB5*01:01:01", 1, 1,
  3, 1, 1, 2, 1,
  "HLA-KIR2DL5A*0010101", 1, 1,
  3, 1, 2, 1, 1,
  "HLA-KIR2DL5A*0010201", 1, 1,
  3, 1, 3, 1, 1,
  "HLA-KIR2DL5B*0010201", 1, 2,
  1, 1, 1, 1, 1,
  "HLA-KIR2DL5B*0010301", 1, 2,
  1, 1, 2, 1, 1
)

ambiguity_table_to_GLstring(data)
```

`GLstring_expand_longer` *GLstring_expand_longer*

Description

A function that expands a GL string into a longer, more detailed format (also known as an ambiguity table) by separating the string into its components resulting from its hierarchical set of operators, including gene locations, loci, genotypes, haplotypes, and alleles. The function processes each level of the GL string and assigns identifiers for each hierarchical component. The resulting table can be assembled back into a GL string using the function ‘ambiguity_table_to_GLstring‘.

Usage

```
GLstring_expand_longer(GL_string)
```

Arguments

`GL_string`	A GL string that encodes HLA alleles and their potential ambiguities

Value

A tibble that contains the expanded GL string with separate columns for possible gene locations, loci, genotype ambiguities, genotypes, haplotypes, and alleles, each with associated identifiers

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())
result <- GLstring_expand_longer(GL_string[1])
print(result)
```

`GLstring_genes` *GLstring_genes*

Description

This function processes a specified column in a data frame that contains GL strings. It separates the GL strings, identifies the HLA loci, and transforms the data into a wider format with loci as column names.

Usage

```
GLstring_genes(data, gl_string)
```

Arguments

`data`	A data frame
`gl_string`	The name of the column in the data frame that contains GL strings

Value

A data frame with GL strings separated, loci identified, and data transformed to a wider format with loci as columns.

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- data.frame("GL_string" = HLA_columns_to_GLstring(
  file,
  HLA_typing_columns = everything()
))
GL_string <- GL_string[1, , drop = FALSE] # When considering first patient
result <- GLstring_genes(GL_string, "GL_string")
print(result)
```

GLstring_genes_expanded *GLstring_genes_expanded*

Description

This function processes a specified column in a data frame that contains GL strings. It separates the GL strings, identifies the HLA loci, and transforms the data into a wider format with loci as column names. It also creates multiple rows to separate each locus in the allele.

Usage

```
GLstring_genes_expanded(data, gl_string)
```

Arguments

data	A data frame containing GL strings for HLA data.
gl_string	The name of the column in the data frame that contains GL strings.

Value

A data frame with expanded columns, where each row has a single allele for a specific locus.

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- data.frame("GL_string" = HLA_columns_to_GLstring(
  file,
  HLA_typing_columns = everything()
))
GL_string <- GL_string[1, , drop = FALSE] # When considering first patient
result <- GLstring_genes_expanded(GL_string, "GL_string")
print(result)
```

GLstring_gene_copies_combine *GLstring_gene_copies_combine*

Description

A function for combining two columns of typing from the same locus into a single column in the appropriate GL string format.

Usage

```
GLstring_gene_copies_combine(.data, columns, sample_column = "sample")
```

Arguments

.data	A data frame
columns	The names of the columns in the data frame that contain typing information to be combined
sample_column	The name of the column that identifies samples in the data frame. Default is "sample".

Value

A data frame with the specified columns combined into a single column for each locus, in GL string format.

Examples

```
library(dplyr)
HLA_typing_1 %>%
  mutate(across(A1:B2, ~ HLA_prefix_add(.))) %>%
  GLstring_gene_copies_combine(c(A1:B2), sample_column = patient)
```

GLstring_genotype_ambiguity

GLstring_genotype_ambiguity

Description

This function processes GL strings in the specified columns of a data frame to retain only the first genotype ambiguity, optionally retaining the remaining ambiguities in a separate column with "_ambiguity" appended. The function ensures that genes have been separated from the GL strings prior to execution; otherwise, an error will be thrown if a "^" is detected in the GL strings.

Usage

```
GLstring_genotype_ambiguity(data, columns, keep_ambiguities = FALSE)
```

Arguments

data	A data frame
columns	The names of the columns in the data frame that contain GL strings
keep_ambiguities	A logical value indicating whether to retain the remaining ambiguities in separate columns with "_genotype_ambiguity" appended to the original column names. Default is FALSE.

Value

A data frame with the first genotype ambiguity retained in the original columns. If keep_ambiguities is TRUE, the remaining ambiguities are placed in separate columns.

Examples

```
HLA_type <- data.frame(
  sample = c("sample1", "sample2"),
 HLA_A = c("A*01:01+A*68:01|A*01:02+A*68:55|A*01:99+A*68:66", "A*02:01+A*03:01|A*02:02+A*03:03"),
  HLA_B = c("B*07:02+B*58:01|B*07:03+B*58:09", "B*08:01+B*15:01|B*08:02+B*15:17"),
  stringsAsFactors = FALSE
)

GLstring_genotype_ambiguity(HLA_type, columns = c("HLA_A", "HLA_B"), keep_ambiguities = TRUE)
```

GLstring_regex *GLstring_regex*

Description

This function will format an HLA allele (e.g. "HLA-A*02:01") to a regex pattern for searching within a GL string. Note that in order for this function to work properly, the full HLA allele name, including prefixes, is required. Allele values of "A*02:01" will need to be updated to "HLA-A*02:01", and "A2" will need to be updated to "HLA-A2". The ‘HLA_prefix_add‘ function is useful in these situations.

Usage

```
GLstring_regex(data)
```

Arguments

data	A string containing an HLA allele.

Value

A string with the HLA allele formatted as a regex pattern.

Examples

```
# To understand how the function works we can see how it alters the allele "HLA-A*02:01":

GLstring_regex("HLA-A*02:01")

# The result is the same allele with extra formatting to escape special characters found
# in a GL string, as well as the ability to accurately search for an allele in a GL string.
# For example, we would not want the allele "HLA-A*02:14" to match to "HLA-A*02:149:01",
# which would happen if we simply escaped the special characters:

library(stringr)
str_view("HLA-A*02:149:01", str_escape("HLA-A*02:14"), match = NA)

# Using `GLstring_regex` prevents this:
```

```
str_view("HLA-A*02:149:01", GLstring_regex("HLA-A*02:14"), match = NA)

# Using a longer GL string with multiple alleles and loci:

GL_string <- "HLA-A*02:01:01+HLA-A*68:01^HLA-B*07:01+HLA-B*15:01"

# We can match any allele accurately:

str_view(GL_string, GLstring_regex("HLA-A*68:01"), match = NA)

# Note that alleles supplied with fewer fields than in the GL string will also match:

str_view(GL_string, GLstring_regex("HLA-A*02:01"), match = NA)
```

`GLstring_to_ambiguity_table` *GLstring_to_ambiguity_table*

Description

A function that expands a GL string into a longer, more detailed format (also known as an ambiguity table) by separating the string into its components resulting from its hierarchical set of operators, including gene locations, loci, genotypes, haplotypes, and alleles. The function processes each level of the GL string and assigns identifiers for each hierarchical component. The resulting table can be assembled back into a GL string using the function 'ambiguity_table_to_GLstring'. This function is an alias of 'GLstring_expand_longer'.

Usage

```
GLstring_to_ambiguity_table(GL_string)
```

Arguments

`GL_string`	A GL string that encodes HLA alleles and their potential ambiguities

Value

A tibble that contains the expanded GL string with separate columns for possible gene locations, loci, genotype ambiguities, genotypes, haplotypes, and alleles, each with associated identifiers

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())
result <- GLstring_to_ambiguity_table(GL_string[1])
print(result)
```

`Haplotype_frequencies` *Ten HLA-A~C~B~DRB3/4/5~DRB1~DQA1~DQB1~DPA1~DPB1 haplotypes and their frequencies in the global population.*

Description

Ten HLA-A~C~B~DRB3/4/5~DRB1~DQA1~DQB1~DPA1~DPB1 haplotypes and their frequencies in the global population.

Usage

```
data(Haplotype_frequencies)
```

Format

A tibble:

HLA-A:HLA-DPB1 HLA allele names

Global_Fq haplotype frequencies

Global_n haplotype observations

Global_rank haplotype rank ...

Source

<https://doi.org/10.1016/j.humimm.2021.04.007>

`HLA_columns_to_GLstring` *HLA_columns_to_GLstring*

Description

A function to take HLA typing data spread across different columns, as is often found in wild-caught data, and transform it to a GL string. If column names have anything besides the locus name and a number (e.g. "mA1Cd" instead of just "A1"), the function will have trouble determining the locus from the column name. The ‘prefix_to_remove‘ and ‘suffix_to_remove‘ arguments can be used to clean up the column names. See the example for how these arguments are used.

Usage

```
HLA_columns_to_GLstring(
  data,
  HLA_typing_columns,
  prefix_to_remove = "",
  suffix_to_remove = ""
)
```

Arguments

data	A data frame with each row including an HLA typing result, with individual columns containing a single allele.
HLA_typing_columns	A list of columns containing the HLA alleles. Tidyselect is supported.
prefix_to_remove	An optional string of characters to remove from the locus names. The goal is to get the column names to the locus and a number. For example, columns named "mDRB11Cd" and "mDRB12Cd" should use the ‘prefix_to_remove‘ value of "m".
suffix_to_remove	An optional string of characters to remove from the locus names. Using the example above, the ‘suffix_to_remove‘ value will be "Cd".

Value

A list of GL strings in the order of the original data frame.

Examples

```
# The HLA_typing_LIS dataset contains a table as might be found in a clinical laboratory
# information system:
print(HLA_typing_LIS)

# The `HLA_columns_to_GLString` function can be used to coerce typing spread across
# multiple columns into a GL string:
library(dplyr)
HLA_typing_LIS %>%
  mutate(
    GL_string = HLA_columns_to_GLstring(
      ., # Note that if this function is used inside a `mutate` call "." will have to be
      # used as the first argument to extract data from the working data frame.
      HLA_typing_columns = mA1Cd.recipient:mDPB12cd.recipient,
      prefix_to_remove = "m",
      suffix_to_remove = "Cd.recipient"
    ),
    .after = patient
  ) %>%
  select(patient, GL_string)
```

HLA_column_repair *HLA_column_repair*

Description

This function will change column names that have the official HLA nomenclature (e.g. "HLA-A*" or "HLA-A") to a format more easily selected in tidyverse functions (e.g. "HLA_A"). The dash and asterisk are special characters in R, and makes selecting columns by name difficult. This function will also allow for easily changing back to WHO-compliant nomenclature (e.g. "HLA-A*").

Usage

```
HLA_column_repair(data, format = "tidyverse", asterisk = FALSE)
```

Arguments

data	A data frame
format	Either "tidyverse" or "WHO".
asterisk	Logical value to return column with an asterisk.

Value

A data frame object with column names renamed in the specified format.

Examples

```
HLA_type <- data.frame(
  "HLA-A*" = c("01:01", "02:01"),
  "HLA-B*" = c("07:02", "08:01"),
  "HLA-C*" = c("03:04", "04:01"),
  stringsAsFactors = FALSE
)

HLA_column_repair(HLA_type, format = "tidyverse")
```

HLA_dictionary *Data on HLA alleles in the 2008 HLA dictionary*

Description

Data on HLA alleles in the 2008 HLA dictionary

Usage

```
data(HLA_dictionary)
```

Format

A tibble:

IPD.Accession IPD-IMGT/HLA Accession number

HLA.Allele HLA allele name

WHO.Assigned.Type WHO-assigned serologic equivalent ...

Source

<https://doi.org/10.1111/j.1399-0039.2008.01183.x>

HLA_match_number *HLA_match_number*

Description

Calculates the number of HLA matches as two minus the number of mismatches from ‘HLA_mismatch_number‘. Homozygous mismatches are counted twice. Supports match calculations for host-vs-graft (HvG), graft-vs-host (GvH), or bidirectional. Bidirectional matching is the default, but can be overridden using the "direction" argument.

Usage

```
HLA_match_number(
  GL_string_recip,
  GL_string_donor,
  loci,
  direction = "bidirectional"
)
```

Arguments

GL_string_recip	A GL string representing the recipient's HLA genotype.
GL_string_donor	A GL string representing the donor's HLA genotype.
loci	A character vector specifying the loci to be considered for mismatch calculation. HLA-DRB3/4/5 (and their serologic equivalents DR51/52/53) are considered once locus for this function, and should be called in this argument as "HLA-DRB3/4/5" or "HLA-DR51/52/53", respectively.
direction	A character string indicating the direction of match. Options are "HvG" (host vs. graft), "GvH" (graft vs. host), "bidirectional" (the minimum value of "HvG" and "GvH").

Value

An integer value or a character string: - If ‘loci‘ includes only one locus, the function returns an integer mismatch count for that locus. - If ‘loci‘ includes multiple loci, the function returns a character string in the format "Locus1=Count1, Locus2=Count2, ...".

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())
GL_string_recip <- GL_string[1]
GL_string_donor <- GL_string[2]

loci <- c("HLA-A", "HLA-B")
```

```
# Calculate mismatch numbers (Host vs. Graft)
HLA_match_number(GL_string_recip, GL_string_donor, loci, direction = "HvG")

# Calculate mismatch numbers (Graft vs. Host)
HLA_match_number(GL_string_recip, GL_string_donor, loci, direction = "GvH")

# Calculate mismatch numbers (Bidirectional)
HLA_match_number(GL_string_recip, GL_string_donor,
  loci,
  direction = "bidirectional"
)
```

`HLA_match_summary_HCT` *HLA_match_summary_HCT*

Description

Calculates the match summary for either the HLA-A, B, C and DRB1 loci (out-of-8 matching) or the HLA-A, B, C, DRB1 and DQB1 loci (out-of-10 matching), as is commonly used for hematopoietic cell transplantation (HCT). Homozygous mismatches are counted twice. Bidirectional matching is the default, but can be overridden with the "direction" argument.

Usage

```
HLA_match_summary_HCT(
  GL_string_recip,
  GL_string_donor,
  direction = "bidirectional",
  match_grade,
  scope = "locus"
)
```

Arguments

`GL_string_recip`	A GL string representing the recipient's HLA genotype, and minimally containing the HLA-A, B, C and DRB1 loci (for Xof8 matching) or the HLA-A, B, C, DRB1 and DQB1 loci (for Xof10 matching).
`GL_string_donor`	A GL string representing the donor's HLA genotype, and minimally containing the HLA-A, B, C and DRB1 loci (for Xof8 matching) or the HLA-A, B, C, DRB1 and DQB1 loci (for Xof10 matching).
`direction`	"GvH", "HvG" or "bidirectional". Default is "bidirectional".
`match_grade`	"Xof8" for HLA-A, B, C and DRB1 matching or "Xof10" for HLA-A, B, C, DRB1 and DQB1 matching.

scope "locus" or "genotype". Default is "locus". When "locus" (the default), bidirectional matching takes the minimum match count at each locus independently before summing. When "genotype", the GvH and HvG match summaries are calculated separately across all loci, and the maximum of the two totals is returned. Only affects results when direction is "bidirectional"; ignored otherwise.

Value

An integer value of the match grade summary.

Examples

```
# Example recipient and donor GL strings
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())

GL_string_recip <- GL_string[1]
GL_string_donor <- GL_string[2]

# Calculate mismatch numbers
HLA_match_summary_HCT(GL_string_recip, GL_string_donor,
  direction = "bidirectional", match_grade = "Xof8"
)

# Genotype-level bidirectional matching (max of GvH and HvG totals)
HLA_match_summary_HCT(GL_string_recip, GL_string_donor,
  direction = "bidirectional", match_grade = "Xof8", scope = "genotype"
)
```

HLA_mismatched_alleles

HLA_mismatched_alleles

Description

A function to return a string of mismatches between recipient and donor HLA genotypes represented as GL strings. The function finds mismatches based on the direction of comparison specified in the inputs and also handles homozygosity.

'HLA_mismatch_alleles' and 'HLA_mismatched_alleles' are synonyms.

Usage

```
HLA_mismatched_alleles(
  GL_string_recip,
  GL_string_donor,
  loci,
  direction,
  homozygous_count = 2
)
```

Arguments

GL_string_recip	A GL strings representing the recipient's HLA genotypes.
GL_string_donor	A GL strings representing the donor's HLA genotypes.
loci	A character vector specifying the loci to be considered for mismatch calculation. HLA-DRB3/4/5 (and their serologic equivalents DR51/52/53) are considered once locus for this function, and should be called in this argument as "HLA-DRB3/4/5" or "HLA-DR51/52/53", respectively.
direction	A character string indicating the direction of mismatch. Options are "HvG" (host vs. graft), "GvH" (graft vs. host), "bidirectional" (the max value of "HvG" and "GvH"), or "SOT" (host vs. graft, as is used for mismatching in solid organ transplantation).
homozygous_count	An integer specifying how to handle homozygosity. Defaults to 2, where homozygous alleles are treated as duplicated for mismatch calculations. Can be specified as 1, in which case homozygous alleles are treated as single occurrences without duplication (in other words, homozygous mismatches are only "counted" once).

Value

A character vector, where each element is a string summarizing the mismatches for the specified loci. The strings are formatted as comma-separated locus mismatch entries if multiple loci were supplied, or as simple GL strings if a single locus was supplied.

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())

GL_string_recip <- GL_string[1]
GL_string_donor <- GL_string[2]

loci <- c("HLA-A", "HLA-DRB3/4/5", "HLA-DPB1")
mismatches <- HLA_mismatched_alleles(GL_string_recip, GL_string_donor, loci, direction = "HvG")
print(mismatches)
```

HLA_mismatch_alleles *HLA_mismatch_alleles*

Description

A function to return a string of mismatches between recipient and donor HLA genotypes represented as GL strings. The function finds mismatches based on the direction of comparison specified in the inputs and also handles homozygosity.

'HLA_mismatch_alleles' and 'HLA_mismatched_alleles' are synonyms.

Usage

```
HLA_mismatch_alleles(
  GL_string_recip,
  GL_string_donor,
  loci,
  direction,
  homozygous_count = 2
)
```

Arguments

GL_string_recip	A GL strings representing the recipient's HLA genotypes.
GL_string_donor	A GL strings representing the donor's HLA genotypes.
loci	A character vector specifying the loci to be considered for mismatch calculation. HLA-DRB3/4/5 (and their serologic equivalents DR51/52/53) are considered once locus for this function, and should be called in this argument as "HLA-DRB3/4/5" or "HLA-DR51/52/53", respectively.
direction	A character string indicating the direction of mismatch. Options are "HvG" (host vs. graft), "GvH" (graft vs. host), "bidirectional" (the max value of "HvG" and "GvH"), or "SOT" (host vs. graft, as is used for mismatching in solid organ transplantation).
homozygous_count	An integer specifying how to handle homozygosity. Defaults to 2, where homozygous alleles are treated as duplicated for mismatch calculations. Can be specified as 1, in which case homozygous alleles are treated as single occurrences without duplication (in other words, homozygous mismatches are only "counted" once).

Value

A character vector, where each element is a string summarizing the mismatches for the specified loci. The strings are formatted as comma-separated locus mismatch entries if multiple loci were supplied, or as simple GL strings if a single locus was supplied.

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())

GL_string_recip <- GL_string[1]
GL_string_donor <- GL_string[2]

loci <- c("HLA-A", "HLA-DRB3/4/5", "HLA-DPB1")
mismatches <- HLA_mismatch_alleles(GL_string_recip, GL_string_donor, loci, direction = "HvG")
print(mismatches)
```

HLA_mismatch_base *HLA_mismatch_base*

Description

A function to return a string of mismatches between recipient and donor HLA genotypes represented as GL strings. The function finds mismatches based on the direction of comparison specified in the inputs and also handles homozygosity. As the name implies, this function is the base for all other mismatch (and matching) functions. This function is not meant to be called directly; it is better to use one of the derivative functions.

Usage

```
HLA_mismatch_base(
  GL_string_recip,
  GL_string_donor,
  loci,
  direction,
  homozygous_count = 2
)
```

Arguments

GL_string_recip	A GL string representing the recipient's HLA genotype.
GL_string_donor	A GL string representing the donor's HLA genotype.
loci	A character vector specifying the loci to be considered for mismatch calculation. HLA-DRB3/4/5 (and their serologic equivalents DR51/52/53) are considered once locus for this function, and should be called in this argument as "HLA-DRB3/4/5" or "HLA-DR51/52/53", respectively.
direction	A character string indicating the direction of mismatch. Options are "HvG" (host vs. graft) or "GvH" (graft vs. host).
homozygous_count	An integer specifying how to handle homozygosity. Defaults to 2, where homozygous alleles are treated as duplicated for mismatch calculations. Can be specified to be 1, in which case homozygous alleles are treated as single occurrences without duplication.

Value

A character vector, where each element is a string summarizing the mismatches for the specified loci. The strings are formatted as comma-separated locus mismatch entries if multiple loci are supplied, or simple GL strings if a single locus is supplied.

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())

GL_string_recip <- GL_string[1]
GL_string_donor <- GL_string[2]

loci <- c("HLA-A", "HLA-DRB3/4/5", "HLA-DPB1")
mismatches <- HLA_mismatch_base(GL_string_recip, GL_string_donor, loci, direction = "HvG")
print(mismatches)
```

HLA_mismatch_logical *HLA_mismatch_logical*

Description

Determines if there are any mismatches between recipient and donor HLA alleles for the specified loci. Returns ‘TRUE‘ if mismatches are present, and ‘FALSE‘ otherwise.

Usage

```
HLA_mismatch_logical(GL_string_recip, GL_string_donor, loci, direction)
```

Arguments

GL_string_recip	A GL strings representing the recipient’s HLA genotypes.
GL_string_donor	A GL strings representing the donor’s HLA genotypes.
loci	A character vector specifying the loci to be considered for mismatch calculation. HLA-DRB3/4/5 (and their serologic equivalents DR51/52/53) are considered once locus for this function, and should be called in this argument as "HLA-DRB3/4/5" or "HLA-DR51/52/53", respectively.
direction	A character string indicating the direction of mismatch. Options are "HvG" (host vs. graft), "GvH" (graft vs. host), "bidirectional" (if either "HvG" or "GvH" is TRUE), or "SOT" (host vs. graft, as is used for mismatching in solid organ transplantation).

Value

A logical value (‘TRUE‘ or ‘FALSE‘): - ‘TRUE‘ if there are mismatches between recipient and donor HLA alleles. - ‘FALSE‘ if there are no mismatches.

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())

GL_string_recip <- GL_string[1]
GL_string_donor <- GL_string[2]

loci <- c("HLA-A", "HLA-DRB3/4/5", "HLA-DPB1")
mismatches <- HLA_mismatch_logical(GL_string_recip, GL_string_donor, loci, direction = "HvG")
print(mismatches)
```

HLA_mismatch_number *HLA_mismatch_number*

Description

Calculates the number of mismatched HLA alleles between a recipient and a donor across specified loci. Supports mismatch calculations for host-vs-graft (HvG), graft-vs-host (GvH), or bidirectional.

Usage

```
HLA_mismatch_number(
  GL_string_recip,
  GL_string_donor,
  loci,
  direction,
  homozygous_count = 2
)
```

Arguments

GL_string_recip	A GL string representing the recipient's HLA genotype.
GL_string_donor	A GL string representing the donor's HLA genotype.
loci	A character vector specifying the loci to be considered for mismatch calculation. HLA-DRB3/4/5 (and their serologic equivalents DR51/52/53) are considered once locus for this function, and should be called in this argument as "HLA-DRB3/4/5" or "HLA-DR51/52/53", respectively.
direction	A character string indicating the direction of mismatch. Options are "HvG" (host vs. graft), "GvH" (graft vs. host), "bidirectional" (the max value of "HvG" and "GvH"), or "SOT" (host vs. graft, as is used for mismatching in solid organ transplantation).

homozygous_count
: An integer specifying how to count homozygous mismatches. Defaults to 2, where homozygous mismatches are treated as two mismatches, regardless if one or two alleles are supplied in the GL string (in cases where one allele is supplied, it is duplicated by the function). If specified as 1, homozygous mismatches are only counted once, regardless of whether one or two alleles are supplied in the GL string (in cases where two alleles are supplied, the second identical allele is deleted).

Value

An integer value or a character string: - If 'loci' includes only one locus, the function returns an integer mismatch count for that locus. - If 'loci' includes multiple loci, the function returns a character string in the format "Locus1=Count1, Locus2=Count2, ...".

Examples

```
file <- HLA_typing_1[, -1]
GL_string <- HLA_columns_to_GLstring(file, HLA_typing_columns = everything())

GL_string_recip <- GL_string[1]
GL_string_donor <- GL_string[2]

loci <- c("HLA-A", "HLA-DRB3/4/5", "HLA-DPB1")

# Calculate mismatch numbers (Host vs. Graft)
HLA_mismatch_number(GL_string_recip, GL_string_donor, loci, direction = "HvG")

# Calculate mismatch numbers (Graft vs. Host)
HLA_mismatch_number(GL_string_recip, GL_string_donor, loci, direction = "GvH")

# Calculate mismatch numbers (Bidirectional)
HLA_mismatch_number(GL_string_recip, GL_string_donor,
  loci,
  direction = "bidirectional"
)
```

HLA_prefix_add *HLA_prefix_add*

Description

This function adds a specified prefix to the beginning of each HLA type, and works on a single allele or all alleles in a GL string. Useful for adding HLA or gene prefixes.

Usage

```
HLA_prefix_add(data, prefix = "HLA-")
```

Arguments

data	A string with a single HLA allele, a GL string of HLA alleles, or a character vector containing either of the previous.
prefix	A character string to be added as a prefix to the alleles. Default is "HLA-".

Value

A vector with the specified prefix added to the values.

Examples

```
# The HLA_typing_LIS dataset contains a table as might be found in a clinical
# laboratory information system:
print(HLA_typing_LIS)

# The `HLA_prefix_add` function can be used to add the correct HLA prefixes to the table:
library(dplyr)
HLA_typing_LIS %>% mutate(
  across(
    mA1Cd.recipient:mA2Cd.recipient,
    ~ HLA_prefix_add(., "HLA-A*")
  )
)
```

HLA_prefix_remove *HLA_prefix_remove*

Description

This function removes HLA and optionally locus prefixes from a string of HLA typing: "HLA-A2" changes to "A2" or "2". By default, HLA and locus prefixes are removed. This function also works on each allele in a GL string.

Usage

```
HLA_prefix_remove(data, keep_locus = FALSE)
```

Arguments

data	A string with a single HLA allele, a GL string of HLA alleles, or a character vector containing either of the previous.
keep_locus	A logical value indicating whether to retain any locus values. The default value is FALSE.

Value

A vector modified to remove HLA and optionally locus prefixes.

Examples

```
# The HLA_typing_1 dataset contains a table with HLA typing spread across multiple columns:
print(HLA_typing_1)

# The `HLA_prefix_remove` function can be used to get each column to have only the
# colon-separated fields:
library(dplyr)
HLA_typing_1 %>% mutate(
  across(
    A1:DPB1_2,
    ~ HLA_prefix_remove(.)
  )
)
```

`HLA_truncate` *HLA_truncate*

Description

This function truncates HLA typing values in molecular nomenclature (for example from 4 fields to 2 fields). The truncation is based on the number of fields specified and optionally retains any WHO-recognized suffixes (L, S, C, A, Q, or N) or G and P group designations (G or P). This function will work on individual alleles (e.g. "HLA-A*02:01:01:01") or on all alleles in a GL string (e.g. "HLA-A*02:01:01:01+HLA-A*68:01:01^HLA-DRB1*01:01:01+HLA-DRB1*03:01:01").

Note: depending on arguments used, this function can output HLA alleles that do not exist in the IPD-IMGT/HLA database. For example, truncating the allele "DRB4*01:03:01:02N" to 2 fields would result in "DRB4*01:03N," which does not exist in the IPD-IMGT/HLA database. Users should take care in setting the parameters for this function.

Usage

```
HLA_truncate(
  data,
  fields = 2,
  keep_suffix = TRUE,
  keep_G_P_group = FALSE,
  remove_duplicates = FALSE
)
```

Arguments

`data`	A string containing an HLA allele or a GL string.
`fields`	An integer specifying the number of fields to retain in the truncated values. Default is 2.
`keep_suffix`	A logical value indicating whether to retain any WHO-recognized suffixes. Default is TRUE.

keep_G_P_group A logical value indicating whether to retain any G or P group designations. Default is FALSE.

remove_duplicates A logical value indicating whether to remove duplicated values from a GL string after truncation. Default is FALSE.

Value

A string with the HLA typing truncated according to the specified number of fields and optional suffix retention.

Examples

```
# The Haplotype_frequencies dataset contains a table with HLA typing spread across multiple columns:
print(Haplotype_frequencies)

# The `HLA_truncate` function can be used to truncate the typing results to 2 fields:
library(dplyr)
Haplotype_frequencies %>% mutate(
  across(
    "HLA-A":"HLA-DPB1",
    ~ HLA_truncate(
      .,
      fields = 2,
      keep_suffix = TRUE,
      keep_G_P_group = FALSE
    )
  )
)
```

HLA_typing_1 *Synthetic HLA typing data for 10 individuals for the HLA-A, B, C, DRB1, DRB3/4/5, DQB1, DQA1, DPB1 and DPA1 loci.*

Description

Synthetic HLA typing data for 10 individuals for the HLA-A, B, C, DRB1, DRB3/4/5, DQB1, DQA1, DPB1 and DPA1 loci.

Usage

```
data(HLA_typing_1)
```

Format

A tibble:

patient Patients 1-10

A1:DPB1_2 HLA allele names ...

Source

Synthetic data from Nicholas K. Brown.

`HLA_typing_LIS` *Synthetic HLA typing data for 10 individuals for the HLA-A, B, C, DRB1, DRB3, DRB4, DRB5, DQB1, DQA1, DPB1 and DPA1 loci. Data formatted to resemble a HistoTrac table.*

Description

Synthetic HLA typing data for 10 individuals for the HLA-A, B, C, DRB1, DRB3, DRB4, DRB5, DQB1, DQA1, DPB1 and DPA1 loci. Data formatted to resemble a HistoTrac table.

Usage

```
data(HLA_typing_LIS)
```

Format

A tibble:

patient Patients 1-10

mA1Cd.recipient:mDPB12cd.recipient HLA allele names ...

Source

Synthetic data from Nicholas K. Brown.

`HLA_validate` *HLA_validate*

Description

Returns only HLA alleles in valid nomenclature, either serologic or molecular. Simple numbers, such as "2" or "27" will be returned as-is. Suffixes that are not WHO-recognized suffixes (L, S, C, A, Q, N) or G or P group designations will be removed. For example "novel" at the end of the allele will be removed, while "n" at the end of the allele will be retained. Other values, such as "blank" or "-" will be converted to NA values. This function is helpful for cleaning up the typing of an entire table of HLA values.

Usage

```
HLA_validate(data)
```

Arguments

data	A string containing an HLA allele.

Value

A string with a valid HLA allele or NA if no valid allele was present.

Examples

```
HLA_validate("HLA-A2")
HLA_validate("A*02:01:01:01N")
HLA_validate("A*02:01:01N")
HLA_validate("HLA-DRB1*02:03novel")
HLA_validate("HLA-DQB1*03:01v")
HLA_validate("HLA-DRB1*02:03P")
HLA_validate("HLA-DPB1*04:01:01G")
HLA_validate("2")
HLA_validate(2)
HLA_validate("B27")
HLA_validate("A*010101")
HLA_validate("-")
HLA_validate("blank")

# The HLA_typing_LIS dataset contains a table with HLA typing spread across multiple columns:
print(HLA_typing_LIS)

# Cleaning up the entire table. Note that blank values will be converted to "NA".
library(dplyr)
HLA_typing_LIS %>% mutate(
  across(
    mA1Cd.recipient:mDPB12cd.recipient,
    ~ HLA_validate(.)
  )
)
```

---

mismatch_table_2010 *Consensus mismatch numbers for every possible allele combination at a single locus, from the 2010 publication.*

Description

Consensus mismatch numbers for every possible allele combination at a single locus, from the 2010 publication.

Usage

```
data(mismatch_table_2010)
```

Format

A tibble:

patient:donor allele combinations for patient and donor

#GvH:#Max consensus mismatch number for each direction ...

Source

<https://www.nature.com/articles/bmt2010132>

`mismatch_table_2016` *Consensus mismatch numbers for every possible allele combination at a single locus, from the 2016 publication.*

Description

Consensus mismatch numbers for every possible allele combination at a single locus, from the 2016 publication.

Usage

```
data(mismatch_table_2016)
```

Format

A tibble:

Patient:Donor allele combinations for patient and donor

#GvH:#Max consensus mismatch number for each direction ...

Source

<https://www.nature.com/articles/bmt2010132>

read_HML *read_HML*

Description

Reads the GL strings of HML files and returns a tibble with the full genotype for each sample.

Usage

```
read_HML(HML_file)
```

Arguments

HML_file	The path to an HML file.

Value

A tibble with the sample name and the GL string.

Examples

```
HML_1 <- system.file("extdata", "HML_1.hml", package = "immunogenetr")
HML_2 <- system.file("extdata", "hml_2.hml", package = "immunogenetr")

read_HML(HML_1)
read_HML(HML_2)
```

Index