Package 'immunogenetr'

January 23, 2025

Type Package

Title Data Wrangling for HLA Data

Version 0.1.0

Description Makes wrangling HLA data to WHO-approved nomenclature easier. Uses GL string for encoding genotypes.

License GPL-3 # Should probably think about this.

URL https://github.com/k96nb01/immunogenetr_package

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

Encoding UTF-8

RoxygenNote 7.3.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Imports cli, dplyr, glue, purrr, rlang, stringr, tibble, tidyr, tidyselect, xml2, magrittr

NeedsCompilation no

Author Nicholas Brown [cre, aut], Busra Coskun [aut]

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

Depends R (>= 3.5.0)

Contents

`ambiguity_table_to_GLstring` *ambiguity_table_to_GLstring*

Description

A function that converts a data table of HLA allele ambiguities 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)
```

Arguments

`data`	A data frame containing columns that represent possible gene locations, loci, genotype ambiguities, genotypes, and haplotypes.

Value

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

Examples

```
# Example data frame input
data <- tibble(
  value = c(
    "HLA-A*01:01:01:01", "HLA-A*01:02", "HLA-A*01:03", "HLA-A*01:95",
    "HLA-A*24:02:01:01", "HLA-A*01:01:01:01", "HLA-A*01:03",
    "HLA-A*24:03:01:01", "HLA-B*07:01:01", "B*15:01:01"),
  possible_gene_location = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
  locus = c("HLA-A", "HLA-A", "HLA-A", "HLA-A", "HLA-A", "HLA-A", "HLA-A",
    "HLA-A", "HLA-B", "HLA-B"),
  genotype_ambiguity = c(1, 1, 1, 1, 1, 2, 2, 2, 1, 1),
  genotype = c(1, 1, 1, 1, 2, 1, 1, 2, 1, 2),
  haplotype = c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1),
  allele = c(1, 2, 3, 4, 1, 1, 2, 1, 1, 1)
)
result <- ambiguity_table_to_GLstring(data)
print(result)
```

GLstring_expand_longer *GLstring_expand_longer*

Description

A function that expands a GL string into a longer, more detailed format 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.

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

```
GL_string <- "HLA-A*01:01:01:01/HLA-A*01:02/HLA-A*01:03/HLA-A
  *01:95+HLA-A*24:02:01:01|HLA-A*01:01:01:01/HLA-A*01:03+HLA-A*24:03:01:01
  ^HLA-B*07:01:01+B*15:01:01/B*15:02:01|B*07:03+B*15:99:01^HLA-DRB1*03:01:02
  ~HLA-DRB5*01:01:01+HLA-KIR2DL5A*0010101+HLA-KIR2DL5A*0010201?
  HLA-KIR2DL5B*0010201+HLA-KIR2DL5B*0010301"
result <- GLstring_expand_longer(GL_string)
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

```
table <- tibble(GL_string = "HLA-A*29:02+HLA-A*30:02^HLA-C*06:02+HLA-C*07:01^
HLA-B*08:01+HLA-B*13:02^HLA-DRB4*01:03+HLA-DRB4*01:03^HLA-DRB1*04:01+HLA-DRB1*07:01")

table %>% GLstring_genes("GL_string")
```

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

```
table <- tibble(GL_string = "HLA-A*29:02+HLA-A*30:02^HLA-C*06:02+HLA-C*07:01^
HLA-B*08:01+HLA-B*13:02^HLA-DRB4*01:03+HLA-DRB4*01:03^HLA-DRB1*04:01+HLA-DRB1*07:01")

table %>% GLstring_genes_expanded("GL_string")
```

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

```
HLA_type <- tibble(
  sample = c("sample1", "sample2"),
  HLA_A1 = c("HLA-A*01:01", "HLA-A*02:01"),
  HLA_A2 = c("HLA-A*01:02", "HLA-A*02:02"),
  stringsAsFactors = FALSE
  )

HLA_type %>% GLstring_gene_copies_combine(columns = c("HLA_A1", "HLA_A2"))
```

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 <- tibble(
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")
)

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

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

```
typing_table <- tibble(
patient = c("patient1", "patient2", "patient3"),
mA1cd = c("A*01:01", "A*02:01", "A*03:01"),
mA2cd = c("A*11:01", "blank", "A*26:01"),
mB1cd = c("B*07:02", "B*08:01", "B*15:01"),
mB2cd = c("B*44:02", "B*40:01", "-"),
mC1cd = c("C*03:04", "C*04:01", "C*05:01"),
mC2cd = c("C*07:01", "C*07:02", "C*08:01")
)

typing_table %>% mutate(GL_string = HLA_columns_to_GLstring(., HLA_typing_columns =
c(mA1cd:mC2cd), prefix_to_remove = "m", suffix_to_remove = "cd"))
```

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 <- tibble(
"HLA-A*" = c("01:01", "02:01"),
"HLA-B*" = c("07:02", "08:01"),
"HLA-C*" = c("03:04", "04:01")
)
HLA_type %>% HLA_column_repair(format = "tidyverse")
```

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.

Usage

```
HLA_mismatched_alleles(
  GL_string_recip,
  GL_string_donor,
  loci,
  direction = c("HvG", "GvH", "bidirectional"),
  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.
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.

Examples

```
GL_string_recip <- "HLA-A2+HLA-A68^HLA-Cw1+HLA-Cw17^HLA-DR1+HLA-DR17^HLA-DR52
^HLA-DPB1*04:01"
GL_string_donor <- "HLA-A3+HLA-A69^HLA-Cw10+HLA-Cw9^HLA-DR4+HLA-DR17^HLA-DR52
+HLA-DR53^HLA-DPB1*04:01+HLA-DPB1*04:02"
loci <- c("HLA-A", "HLA-Cw", "HLA-DR51/52/53", "HLA-DPB1")
mismatches <- HLA_mismatch_base(GL_string_recip, GL_string_donor, loci, direction = "HvG")
print(mismatches)

# Output
# "HLA-A=HLA-A3+HLA-A69, HLA-Cw=HLA-Cw10+HLA-Cw9, HLA-DR51/52/53=HLA-DR53, HLA-DPB1=HLA-DPB1*04:02"
```

`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.

Usage

```
HLA_mismatch_base(
  GL_string_recip,
  GL_string_donor,
  loci,
  direction = c("HvG", "GvH"),
  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.
`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.

Examples

```
GL_string_recip <- "HLA-A2+HLA-A68^HLA-Cw1+HLA-Cw17^HLA-DR1+HLA-DR17^HLA-DR52
^HLA-DPB1*04:01"
GL_string_donor <- "HLA-A3+HLA-A69^HLA-Cw10+HLA-Cw9^HLA-DR4+HLA-DR17^HLA-DR52
+HLA-DR53^HLA-DPB1*04:01+HLA-DPB1*04:02"
loci <- c("HLA-A", "HLA-Cw", "HLA-DR51/52/53", "HLA-DPB1")
mismatches <- HLA_mismatch_base(GL_string_recip, GL_string_donor, loci, direction = "HvG")
print(mismatches)

# Output
# "HLA-A=HLA-A3+HLA-A69, HLA-Cw=HLA-Cw10+HLA-Cw9, HLA-DR51/52/53=HLA-DR53, HLA-DPB1=HLA-DPB1*04:02"
```

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 = c("HvG", "GvH", "bidirectional"),
  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.
direction	A character string indicating the direction of mismatch. Options are "HvG" (host vs. graft), "GvH" (graft vs. host), or "bidirectional" (max of "HvG" and "GvH").
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 logical value (‘TRUE‘ or ‘FALSE‘): - ‘TRUE‘ if there are mismatches between recipient and donor HLA alleles. - ‘FALSE‘ if there are no mismatches.

Examples

```
# Example recipient and donor GL strings
GL_string_recip <- "HLA-A*03:01+HLA-A*74:01^HLA-DRB3*03:01^HLA-DRB5*02:21"
GL_string_donor <- "HLA-A*03:02+HLA-A*20:01^HLA-DRB3*03:01"

# Check if there are mismatches for HLA-A (Graft vs. Host)
has_mismatch <- HLA_mismatch_logical(GL_string_recip, GL_string_donor, loci =
"HLA-A", direction = "GvH")
print(has_mismatch)
# Output: TRUE
```

`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 = c("HvG", "GvH", "bidirectional"),
  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.
`direction`	A character string indicating the direction of mismatch. Options are "HvG" (host vs. graft), "GvH" (graft vs. host), or "bidirectional" (max of "HvG" and "GvH").
`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

```
# Example recipient and donor GL strings
GL_string_recip <- "HLA-A*01:01+HLA-A*02:01^HLA-B*07:02+HLA-B*08:01"
GL_string_donor <- "HLA-A*01:01+HLA-A*03:01^HLA-B*07:02+HLA-B*44:02"
loci <- c("HLA-A", "HLA-B")

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

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

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

HLA_prefix_add *HLA_prefix_add*

Description

This function adds a specified prefix to the beginning of each value in the identified columns of the given data frame. Useful for adding HLA or gene prefixes.

Usage

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

Arguments

data	A string with a single HLA allele.
prefix	A character string to be added as a prefix to the column values. Default is "HLA-".

Value

A data frame with the specified prefix added to the values in the selected columns.

Examples

```
df <- data.frame(
  A1 = c("01:01", "02:01"),
  A2 = c("03:01", "11:01"),
  B1 = c("07:02", "08:01"),
  B2 = c("15:01", "44:02"),
  stringsAsFactors = FALSE
)

# Add "HLA-A*" prefix to columns A1 and A2
df %>% mutate(across(A1:A2, ~HLA_prefix_add(., "HLA-A*")))
```

`HLA_prefix_remove` *HLA_prefix_remove*

Description

This function removes HLA and locus prefixes from a string of HLA typing: "HLA-A2" changes to "2".

Usage

```
HLA_prefix_remove(data)
```

Arguments

`data`	A string with a single HLA allele.

Value

A string modified to remove HLA and locus prefixes.

Examples

```
df <- data.frame(
  A1 = c("HLA-A2", "A2", "A*11:01", "A66", "HLA-DRB3*15:01"),
  A2 = c("HLA-A1", "A1", "A*02:01", "A68", "HLA-DRB4*14:01"),
  stringsAsFactors = FALSE
)

df %>% mutate(A1 = HLA_prefix_remove(A1))
```

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").

Usage

```
HLA_truncate(data, fields = 2, keep_suffix = TRUE, keep_G_P_group = 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 indicated whether to retain any G or P group desingations. Default is FALSE.

Value

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

Examples

```
typing <- "A*01:01:01:02N"
HLA_truncate(typing) # "A*01:01"
```

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.

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")
```

---

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

```
read_HML("HML_1.hml")
read_HML("HML_2.hml")
```

Index