Package 'tigger'

July 19, 2019

Type Package

Version 0.4.0

Date 2019-07-18

Title Infers Novel Immunoglobulin Alleles from Sequencing Data

Description Infers the V genotype of an individual from immunoglobulin (Ig) repertoire sequencing data (AIRR-Seq, Rep-Seq). Includes detection of any novel alleles. This information is then used to correct existing V allele calls from among the sample sequences.
Citations:
Gadala-Maria, et al (2015) <doi:10.1073/pnas.1417683112>.

License CC BY-SA 4.0

URL `http://tigger.readthedocs.io`

BugReports `https://bitbucket.org/kleinstein/tigger/issues`

LazyData true

BuildVignettes true

VignetteBuilder knitr

Encoding UTF-8

Depends R (>= 3.2.5), ggplot2 (>= 3.1.1)

Imports alakazam (>= 0.3.0), dplyr (>= 0.8.1), doParallel, foreach, graphics, gridExtra, gtools, iterators, lazyeval, parallel, rlang, shazam (>= 0.2.0), stats, stringi, tidyr

Suggests knitr, testthat

RoxygenNote 6.1.1

NeedsCompilation no

Author Daniel Gadala-Maria [aut],
Susanna Marquez [aut],
Moriah Cohen [aut],
Gur Yaari [aut],
Jason Vander Heiden [ctb, cre],
Steven Kleinstein [aut, cph]

Maintainer Jason Vander Heiden <jason.vanderheiden@yale.edu>

Repository CRAN

Date/Publication 2019-07-19 05:40:03 UTC

R topics documented:

`airrDb` *Example human immune repertoire data*

Description

A `data.frame` of example V(D)J immunoglobulin sequences derived from a single individual (PGP1), sequenced on the Roche 454 platform, and assigned by IMGT/HighV-QUEST to IGHV1 family alleles.

Format

A `data.frame` where rows correspond to unique V(D)J sequences and columns include:

- `"sequence_alignment"`: IMGT-gapped V(D)J nucleotide sequence.
- `"v_call"`: IMGT/HighV-QUEST V segment allele calls.
- `"d_call"`: IMGT/HighV-QUEST D segment allele calls.
- `"j_call"`: IMGT/HighV-QUEST J segment allele calls.
- `"junction_length"`: Junction region length.

References

1. Gadala-Maria, et al. (2015) Automated analysis of high-throughput B cell sequencing data reveals a high frequency of novel immunoglobulin V gene segment alleles. PNAS. 112(8):E862-70.

See Also

See SampleDb for Change-O formatted version of `airrDb`.

`cleanSeqs` *Clean up nucleotide sequences*

Description

`cleanSeqs` capitalizes nucleotides and replaces all characters besides `c("A","C","G","T","-",".")` with `"N"`.

Usage

```
cleanSeqs(seqs)
```

Arguments

`seqs`	a vector of nucleotide sequences.

Value

A modified vector of nucleotide sequences.

See Also

sortAlleles and updateAlleleNames can help format a list of allele names.

Examples

```
# Clean messy nucleotide sequences
seqs <- c("AGAT.taa-GAG...ATA", "GATACAGTXXZZAGNNPPACA")
cleanSeqs(seqs)
```

`findNovelAlleles` *Find novel alleles from repertoire sequencing data*

Description

`findNovelAlleles` analyzes mutation patterns in sequences thought to align to each germline allele in order to determine which positions might be polymorphic.

Usage

```
findNovelAlleles(data, germline_db, v_call = "V_CALL",
  j_call = "J_CALL", seq = "SEQUENCE_IMGT", junction = "JUNCTION",
  junction_length = "JUNCTION_LENGTH", germline_min = 200,
  min_seqs = 50, auto_mutrange = TRUE, mut_range = 1:10,
  pos_range = 1:312, y_intercept = 0.125, alpha = 0.05,
  j_max = 0.15, min_frac = 0.75, nproc = 1)
```

Arguments

`data`	a `data.frame` in Change-O format. See details.
`germline_db`	a vector of named nucleotide germline sequences matching the V calls in `data`.
`v_call`	name of the column in `data` with V allele calls. Default is V_CALL.
`j_call`	name of the column in `data` with J allele calls. Default is J_CALL.
`seq`	name of the column in `data` with the aligned, IMGT-numbered, V(D)J nucleotide sequence. Default is SEQUENCE_IMGT.
`junction`	Junction region nucleotide sequence, which includes the CDR3 and the two flanking conserved codons. Default is JUNCTION.
`junction_length`	Number of junction nucleotides in the junction sequence. Default is JUNCTION_LENGTH.
`germline_min`	the minimum number of sequences that must have a particular germline allele call for the allele to be analyzed
`min_seqs`	the minimum number of total sequences (within the desired mutational range and nucleotide range) required for the samples to be considered
`auto_mutrange`	if `TRUE`, the algorithm will attempt to determine the appropriate mutation range automatically using the mutation count of the most common sequence assigned to each allele analyzed
`mut_range`	the range of mutations that samples may carry and be considered by the algorithm
`pos_range`	the range of IMGT-numbered positions that should be considered by the algorithm
`y_intercept`	the y-intercept threshold above which positions should be considered potentially polymorphic

alpha	the alpha value used for determining whether the fit y-intercept is greater than the y_intercept threshold
j_max	the maximum fraction of sequences perfectly aligning to a potential novel allele that are allowed to utilize to a particular combination of junction length and J gene
min_frac	the minimum fraction of sequences that must have usable nucleotides in a given position for that position to considered
nproc	the number of processors to use

Details

The TIgGER allele-finding algorithm, briefly, works as follows: Mutations are determined through comparison to the provided germline. Mutation frequency at each *position* is determined as a function of *sequence-wide* mutation counts. Polymorphic positions exhibit a high mutation frequency despite sequence-wide mutation count. False positive of potential novel alleles resulting from clonally-related sequences are guarded against by ensuring that sequences perfectly matching the potential novel allele utilize a wide range of combinations of J gene and junction length.

Value

A data.frame with a row for each known allele analyzed. Besides metadata on the the parameters used in the search, each row will have either a note as to where the polymorphism-finding algorithm exited or a nucleotide sequence for the predicted novel allele, along with columns providing additional evidence.

The output contains the following columns:

- GERMLINE_CALL: The input (uncorrected) V call.
- NOTE: Comments regarding the inferrence.
- POLYMORPHISM_CALL: The novel allele call.
- NT_SUBSTITUTIONS: Mutations identified in the novel allele, relative to the reference germline (GERMLINE_CALL)
- NOVEL_IMGT: The novel allele sequence.
- NOVEL_IMGT_COUNT: The number of times the sequence NOVEL_IMGT is found in the input data. Considers the subsequence of NOVEL_IMGT in the pos_range.
- NOVEL_IMGT_UNIQUE_J: Number of distinct J calls associated to NOVEL_IMGT in the input data. Considers the subsequence of NOVEL_IMGT in the pos_range.
- NOVEL_IMGT_UNIQUE_CDR3: Number of distinct CDR3 sequences associated with NOVEL_IMGT in the input data. Considers the subsequence of NOVEL_IMGT in the pos_range.
- PERFECT_MATCH_COUNT: Final number of sequences retained to call the new allele. These are unique sequences that have V segments that perfectly match the predicted germline in the pos_range.
- PERFECT_MATCH_FREQ: PERFECT_MATCH_COUNT / GERMLINE_CALL_COUNT
- GERMLINE_CALL_COUNT: The number of sequences with the GERMLINE_CALL in the input data that were initially considered for the analysis.

- GERMLINE_CALL_FREQ: The fraction of sequences with the GERMLINE_CALL in the input data initially considered for the analysis.
- GERMLINE_IMGT: Germline sequence for GERMLINE_CALL.
- GERMLINE_IMGT_COUNT: The number of times the GERMLINE_IMGT sequence is found in the input data.
- MUT_MIN: Minimum mutation considered by the algorithm.
- MUT_MAX: Maximum mutation considered by the algorithm.
- MUT_PASS_COUNT: Number of sequences in the mutation range.
- POS_MIN: First position of the sequence considered by the algorithm (IMGT numbering).
- POS_MAX: Last position of the sequence considered by the algorithm (IMGT numbering).
- Y_INTERCEPT: The y-intercept above which positions were considered potentially polymorphic.
- Y_INTERCEPT_PASS: Number of positions that pass the Y_INTERCEPT threshold.
- SNP_PASS: Number of sequences that pass the Y_INTERCEPT threshold and are within the desired nucleotide range (min_seqs).
- UNMUTATED_COUNT: Number of unmutated sequences.
- UNMUTATED_FREQ: Number of unmutated sequences over GERMLINE_IMGT_COUNT.
- UNMUTATED_SNP_J_GENE_LENGTH_COUNT: Number of distinct combinations of SNP, J gene, and junction length.
- SNP_MIN_SEQS_J_MAX_PASS: Number of SNPs that pass both the min_seqs and j_max thresholds.
- ALPHA: Significance threshold to be used when constructing the confidence interval for the y-intercept.
- MIN_SEQS: Input min_seqs. The minimum number of total sequences (within the desired mutational range and nucleotide range) required for the samples to be considered.
- J_MAX: Input j_max. The maximum fraction of sequences perfectly aligning to a potential novel allele that are allowed to utilize to a particular combination of junction length and J gene.
- MIN_FRAC: Input min_frac. The minimum fraction of sequences that must have usable nucleotides in a given position for that position to be considered.

The following comments can appear in the NOTE column:

- *Novel allele found*: A novel allele was detected.
- *Plurality sequence too rare*: No sequence is frequent enough to pass the J test (j_max).
- *A J-junction combination is too prevalent*: Not enough J diversity (j_max).
- *No positions pass y-intercept test*: No positions above y_intercept.
- *Insufficient sequences in desired mutational range*: mut_range and pos_range.
- *Not enough sequences*: Not enough sequences in the desired mutational range and nucleotide range (min_seqs).
- *No unmutated versions of novel allele found*: All observed variants of the allele are mutated.

See Also

selectNovel to filter the results to show only novel alleles. plotNovel to visualize the data supporting any novel alleles hypothesized to be present in the data and inferGenotype to determine if the novel alleles are frequent enought to be included in the subject's genotype.

Examples

```
# Find novel alleles and return relevant data
novel <- findNovelAlleles(SampleDb, SampleGermlineIGHV)
selectNovel(novel)
# Note: In this example, with SampleGermlineIGHV,
# which contains reference germlines retrieved on August 2018,
# TIgGER finds the allele IGHV1-8*02_G234T. This allele
# was added to IMGT as IGHV1-8*03 on March 28, 2018.
```

---

findUnmutatedCalls *Determine which calls represent an unmutated allele*

Description

`findUnmutatedCalls` determines which allele calls would represent a perfect match with the germline sequence, given a vector of allele calls and mutation counts. In the case of multiple alleles being assigned to a sequence, only the subset that would represent a perfect match is returned.

Usage

```
findUnmutatedCalls(allele_calls, sample_seqs, germline_db)
```

Arguments

`allele_calls`	a vector of strings respresenting Ig allele calls, where multiple calls are separated by a comma.
`sample_seqs`	V(D)J-rearranged sample sequences matching the order of the given `allele_calls`.
`germline_db`	a vector of named nucleotide germline sequences

Value

A vector of strings containing the members of `allele_calls` that represent unmutated sequences.

Examples

```
# Find which of the sample alleles are unmutated
calls <- findUnmutatedCalls(SampleDb$V_CALL, SampleDb$SEQUENCE_IMGT,
         germline_db=SampleGermlineIGHV)
```

generateEvidence *Generate evidence*

Description

generateEvidence builds a table of evidence metrics for the final novel V allele detection and genotyping inferrences.

Usage

```
generateEvidence(data, novel, genotype, genotype_db, germline_db,
  j_call = "J_CALL", junction = "JUNCTION", fields = NULL)
```

Arguments

data	a data.frame containing sequence data that has been passed through reassignAlleles to correct the allele assignments.
novel	the data.frame returned by findNovelAlleles.
genotype	the data.frame of alleles generated with inferGenotype denoting the genotype of the subject.
genotype_db	a vector of named nucleotide germline sequences in the genotype. Returned by genotypeFasta.
germline_db	the original uncorrected germline database used to by findNovelAlleles to identify novel alleles.
j_call	name of the column in data with J allele calls. Default is J_CALL.
junction	Junction region nucleotide sequence, which includes the CDR3 and the two flanking conserved codons. Default is JUNCTION
fields	character vector of column names used to split the data to identify novel alleles, if any. If NULL then the data is not divided by grouping variables.

Value

Returns the genotype input data.frame with the following additional columns providing supporting evidence for each inferred allele:

- FIELD_ID: Data subset identifier, defined with the input paramter fields.
- A variable number of columns, specified with the input parameter fields.
- POLYMORPHISM_CALL: The novel allele call.
- NOVEL_IMGT: The novel allele sequence.
- CLOSEST_REFERENCE: The closest reference gene and allele in the germline_db database.
- CLOSEST_REFERENCE_IMGT: Sequence of the closest reference gene and allele in the germline_db database.
- GERMLINE_CALL: The input (uncorrected) V call.

- GERMLINE_IMGT: Germline sequence for GERMLINE_CALL.
- NT_DIFF: Number of nucleotides that differ between the new allele and the closest reference (CLOSEST_REFERENCE) in the germline_db database.
- NT_SUBSTITUTIONS: A comma separated list of specific nucleotide differences (e.g. 112G>A) in the novel allele.
- AA_DIFF: Number of amino acids that differ between the new allele and the closest reference (CLOSEST_REFERENCE) in the germline_db database.
- AA_SUBSTITUTIONS: A comma separated list with specific amino acid differences (e.g. 96A>N) in the novel allele.
- SEQUENCES: Number of sequences unambiguosly assigned to this allele.
- UNMUTATED_SEQUENCES: Number of records with the unmutated novel allele sequence.
- UNMUTATED_FREQUENCY: Proportion of records with the unmutated novel allele sequence (UNMUTATED_SEQUENCES / SEQUENCE).
- ALLELIC_PERCENTAGE: Percentage at which the (unmutated) allele is observed in the sequence dataset compared to other (unmutated) alleles.
- UNIQUE_JS: Number of unique J sequences found associated with the novel allele. The sequences are those who have been unambiguously assigned to the novel allelle (POLYMORPHISM_CALL).
- UNIQUE_CDR3S: Number of unique CDR3s associated with the inferred allele. The sequences are those who have been unambiguously assigned to the novel allelle (POLYMORPHISM_CALL).
- MUT_MIN: Minimum mutation considered by the algorithm.
- MUT_MAX: Maximum mutation considered by the algorithm.
- POS_MIN: First position of the sequence considered by the algorithm (IMGT numbering).
- POS_MAX: Last position of the sequence considered by the algorithm (IMGT numbering).
- Y_INTERCEPT: The y-intercept above which positions were considered potentially polymorphic.
- ALPHA: Significance threshold to be used when constructing the confidence interval for the y-intercept.
- MIN_SEQS: Input min_seqs. The minimum number of total sequences (within the desired mutational range and nucleotide range) required for the samples to be considered.
- J_MAX: Input j_max. The maximum fraction of sequences perfectly aligning to a potential novel allele that are allowed to utilize to a particular combination of junction length and J gene.
- MIN_FRAC: Input min_frac. The minimum fraction of sequences that must have usable nucleotides in a given position for that position to be considered.
- NOTE: Comments regarding the novel allele inferrence.

See Also

See findNovelAlleles, inferGenotype and genotypeFasta for generating the required input.

Examples

```
# Generate input data
novel <- findNovelAlleles(SampleDb, SampleGermlineIGHV)
genotype <- inferGenotype(SampleDb, find_unmutated=TRUE,
                          germline_db=SampleGermlineIGHV,
                          novel=novel)
genotype_db <- genotypeFasta(genotype, SampleGermlineIGHV, novel)
data_db <- reassignAlleles(SampleDb, genotype_db)

# Assemble evidence table
evidence <- generateEvidence(data_db, novel, genotype,
                             genotype_db, SampleGermlineIGHV)
```

---

genotypeFasta *Return the nucleotide sequences of a genotype*

Description

genotypeFasta converts a genotype table into a vector of nucleotide sequences.

Usage

```
genotypeFasta(genotype, germline_db, novel = NA)
```

Arguments

genotype	a data.frame of alleles denoting a genotype, as returned by inferGenotype.
germline_db	a vector of named nucleotide germline sequences matching the alleles detailed in genotype.
novel	an optional data.frame containing putative novel alleeles of the type returned by findNovelAlleles.

Value

A named vector of strings containing the germline nucleotide sequences of the alleles in the provided genotype.

See Also

inferGenotype

Examples

```
# Find the sequences that correspond to the genotype
genotype_db <- genotypeFasta(SampleGenotype, SampleGermlineIGHV, SampleNovel)
```

GermlineIGHV *Human IGHV germlines*

Description

A `character` vector of all human IGHV germline gene segment alleles in IMGT/GENE-DB (2019-06-01, 372 alleles). See IMGT data updates: http://www.imgt.org/IMGTgenedbdoc/dataupdates.html.

Format

Values correspond to IMGT-gaped nuceltoide sequences (with nucleotides capitalized and gaps represented by ".") while names correspond to stripped-down IMGT allele names (e.g. "IGHV1-18*01").

References

1. Xochelli, et al. (2014) Immunoglobulin heavy variable (IGHV) genes and alleles: new entities, new names and implications for research and prognostication in chronic lymphocytic leukaemia. Immunogenetics. 67(1):61-6.

getMutatedPositions *Find the location of mutations in a sequence*

Description

`getMutatedPositions` takes two vectors of aligned sequences and compares pairs of sequences. It returns a list of the nucleotide positions of any differences.

Usage

```
getMutatedPositions(samples, germlines, ignored_regex = "[\\.N-]",
  match_instead = FALSE)
```

Arguments

`samples`	a vector of strings respresenting aligned sequences
`germlines`	a vector of strings respresenting aligned sequences to which `samples` will be compared. If only one string is submitted, it will be used for all `samples`.
`ignored_regex`	a regular expression indicating what characters should be ignored (such as gaps and N nucleotides).
`match_instead`	if `TRUE`, the function returns the positions that are the same instead of those that are different.

Value

A list of the nucleotide positions of any differences between the input vectors.

Examples

```
# Create strings to act as a sample sequences and a reference sequence
seqs <- c("----GATA", "GAGAGAGA", "TANA")
ref <- "GATAGATA"

# Find the differences between the two
getMutatedPositions(seqs, ref)
```

---

getMutCount *Determine the mutation counts from allele calls*

Description

`getMutCount` takes a set of nucleotide sequences and their allele calls and determines the distance between that seqeunce and any germline alleles contained within the call

Usage

```
getMutCount(samples, allele_calls, germline_db)
```

Arguments

`samples`	a vector of IMGT-gapped sample V sequences
`allele_calls`	a vector of strings respresenting Ig allele calls for the sequences in `samples`, where multiple calls are separated by a comma
`germline_db`	a vector of named nucleotide germline sequences matching the calls detailed in `allele_calls`

Value

A list equal in length to `samples`, containing the Hamming distance to each germline allele contained within each call within each element of `samples`

Examples

```
# Insert a mutation into a germline sequence
s2 <- s3 <- SampleGermlineIGHV[1]
stringi::stri_sub(s2, 103, 103) <- "G"
stringi::stri_sub(s3, 107, 107) <- "C"

sample_seqs <- c(SampleGermlineIGHV[2], s2, s3)

# Pretend that one sample sequence has received an ambiguous allele call
```

```
sample_alleles <- c(paste(names(SampleGermlineIGHV[1:2]), collapse=","),
                    names(SampleGermlineIGHV[2]),
                    names(SampleGermlineIGHV[1]))

# Compare each sequence to its assigned germline(s) to determine the distance
getMutCount(sample_seqs, sample_alleles, SampleGermlineIGHV)
```

`getPopularMutationCount` *Find mutation counts for frequency sequences*

Description

`getPopularMutationCount` determines which sequences occur frequently for each V gene and returns the mutation count of those sequences.

Usage

```
getPopularMutationCount(data, germline_db, v_call = "V_CALL",
  seq = "SEQUENCE_IMGT", gene_min = 0.001, seq_min = 50,
  seq_p_of_max = 1/8, full_return = FALSE)
```

Arguments

`data`	a `data.frame` in the Change-O format. See findNovelAlleles for a list of required columns.
`germline_db`	A named list of IMGT-gapped germline sequences.
`v_call`	name of the column in `data` with V allele calls. Default is V_CALL.
`seq`	name of the column in `data` with the aligned, IMGT-numbered, V(D)J nucleotide sequence. Default is SEQUENCE_IMG
`gene_min`	The portion of all unique sequences a gene must constitute to avoid exclusion.
`seq_min`	The number of copies of the V that must be present for to avoid exclusion.
`seq_p_of_max`	For each gene, fraction of the most common V sequence's count that a sequence must meet to avoid exclusion.
`full_return`	If `TRUE`, will return all `data` columns and will include sequences with mutation count < 1.

Value

A data frame of genes that have a frequent sequence mutation count above 1.

See Also

getMutatedPositions can be used to find which positions of a set of sequences are mutated.

Examples

```
getPopularMutationCount(SampleDb, SampleGermlineIGHV)
```

`inferGenotype` *Infer a subject-specific genotype using a frequency method*

Description

`inferGenotype` infers an subject's genotype using a frequency method. The genotype is inferred by finding the minimum number set of alleles that can explain the majority of each gene's calls. The most common allele of each gene is included in the genotype first, and the next most common allele is added until the desired fraction of alleles can be explained. In this way, mistaken allele calls (resulting from sequences which by chance have been mutated to look like another allele) can be removed.

Usage

```
inferGenotype(data, germline_db = NA, novel = NA, v_call = "V_CALL",
  seq = "SEQUENCE_IMGT", fraction_to_explain = 0.875,
  gene_cutoff = 1e-04, find_unmutated = TRUE)
```

Arguments

`data`	a `data.frame` containing V allele calls from a single subject. If `find_unmutated` is `TRUE`, then the sample IMGT-gapped V(D)J sequence should
`germline_db`	named vector of sequences containing the germline sequences named in `allele_calls`. Only required if `find_unmutated` is `TRUE`.
`novel`	an optional `data.frame` of the type novel returned by findNovelAlleles containing germline sequences that will be utilized if `find_unmutated` is `TRUE`. See Details.
`v_call`	column in `data` with V allele calls. Default is `"V_CALL"`.
`seq`	name of the column in `data` with the aligned, IMGT-numbered, V(D)J nucleotide sequence. Default is SEQUENCE_IMGT.
`fraction_to_explain`	the portion of each gene that must be explained by the alleles that will be included in the genotype.
`gene_cutoff`	either a number of sequences or a fraction of the length of `allele_calls` denoting the minimum number of times a gene must be observed in `allele_calls` to be included in the genotype.
`find_unmutated`	if `TRUE`, use `germline_db` to find which samples are unmutated. Not needed if `allele_calls` only represent unmutated samples.

Details

Allele calls representing cases where multiple alleles have been assigned to a single sample sequence are rare among unmutated sequences but may result if nucleotides for certain positions are not available. Calls containing multiple alleles are treated as belonging to all groups. If `novel` is provided, all sequences that are assigned to the same starting allele as any novel germline allele will have the novel germline allele appended to their assignent prior to searching for unmutated sequences.

Value

A `data.frame` of alleles denoting the genotype of the subject containing the following columns:

- `GENE`: The gene name without allele.
- `ALLELES`: Comma separated list of alleles for the given `GENE`.
- `COUNTS`: Comma separated list of observed sequences for each corresponding allele in the `ALLELES` list.
- `TOTAL`: The total count of observed sequences for the given `GENE`.
- `NOTE`: Any comments on the inferrence.

Note

This method works best with data derived from blood, where a large portion of sequences are expected to be unmutated. Ideally, there should be hundreds of allele calls per gene in the input.

See Also

plotGenotype for a colorful visualization and genotypeFasta to convert the genotype to nucleotide sequences. See inferGenotypeBayesian to infer a subject-specific genotype using a Bayesian approach.

Examples

```
# Infer IGHV genotype, using only unmutated sequences, including novel alleles
inferGenotype(SampleDb, germline_db=SampleGermlineIGHV, novel=SampleNovel,
              find_unmutated=TRUE)
```

`inferGenotypeBayesian` *Infer a subject-specific genotype using a Bayesian approach*

Description

inferGenotypeBayesian infers an subject's genotype by applying a Bayesian framework with a Dirichlet prior for the multinomial distribution. Up to four distinct alleles are allowed in an individual's genotype. Four likelihood distributions were generated by empirically fitting three high coverage genotypes from three individuals (Laserson and Vigneault et al, 2014). A posterior probability is calculated for the four most common alleles. The certainty of the highest probability model was calculated using a Bayes factor (the most likely model divided by second-most likely model). The larger the Bayes factor (K), the greater the certainty in the model.

Usage

```
inferGenotypeBayesian(data, germline_db = NA, novel = NA,
  v_call = "V_CALL", seq = "SEQUENCE_IMGT", find_unmutated = TRUE,
  priors = c(0.6, 0.4, 0.4, 0.35, 0.25, 0.25, 0.25, 0.25, 0.25))
```

Arguments

data	a data.frame containing V allele calls from a single subject. If find_unmutated is TRUE, then the sample IMGT-gapped V(D)J sequence should be provided in column sequence_alignment
germline_db	named vector of sequences containing the germline sequences named in allele_calls. Only required if find_unmutated is TRUE.
novel	an optional data.frame of the type novel returned by findNovelAlleles containing germline sequences that will be utilized if find_unmutated is TRUE. See Details.
v_call	column in data with V allele calls. Default is "V_CALL".
seq	name of the column in data with the aligned, IMGT-numbered, V(D)J nucleotide sequence. Default is SEQUENCE_IMGT.
find_unmutated	if TRUE, use germline_db to find which samples are unmutated. Not needed if allele_calls only represent unmutated samples.
priors	a numeric vector of priors for the multinomial distribution. The priors vector must be nine values that defined the priors for the heterozygous (two allele), trizygous (three allele), and quadrozygous (four allele) distributions. The first two values of priors define the prior for the heterozygous case, the next three values are for the trizygous case, and the final four values are for the quadrozygous case. Each set of priors should sum to one. Note, each distribution prior is actually defined internally by set of four numbers, with the unspecified final values assigned to 0; e.g., the heterozygous case is c(priors[1],priors[2],0,0). The prior for the homozygous distribution is fixed at c(1,0,0,0).

Details

Allele calls representing cases where multiple alleles have been assigned to a single sample sequence are rare among unmutated sequences but may result if nucleotides for certain positions are not available. Calls containing multiple alleles are treated as belonging to all groups. If novel is provided, all sequences that are assigned to the same starting allele as any novel germline allele

will have the novel germline allele appended to their assignent prior to searching for unmutated sequences.

Value

A `data.frame` of alleles denoting the genotype of the subject with the log10 of the likelihood of each model and the log10 of the Bayes factor. The output contains the following columns:

- `GENE`: The gene name without allele.
- `ALLELES`: Comma separated list of alleles for the given `GENE`.
- `COUNTS`: Comma separated list of observed sequences for each corresponding allele in the `ALLELES` list.
- `TOTAL`: The total count of observed sequences for the given `GENE`.
- `NOTE`: Any comments on the inferrence.
- `KH`: log10 likelihood that the `GENE` is homozygous.
- `KD`: log10 likelihood that the `GENE` is heterozygous.
- `KT`: log10 likelihood that the `GENE` is trizygous
- `KQ`: log10 likelihood that the `GENE` is quadrozygous.
- `K_DIFF`: log10 ratio of the highest to second-highest zygosity likelihoods.

Note

This method works best with data derived from blood, where a large portion of sequences are expected to be unmutated. Ideally, there should be hundreds of allele calls per gene in the input.

References

1. Laserson U and Vigneault F, et al. High-resolution antibody dynamics of vaccine-induced immune responses. PNAS. 2014 111(13):4928-33.

See Also

plotGenotype for a colorful visualization and genotypeFasta to convert the genotype to nucleotide sequences. See inferGenotype to infer a subject-specific genotype using a frequency method

Examples

```
# Infer IGHV genotype, using only unmutated sequences, including novel alleles
inferGenotypeBayesian(SampleDb, germline_db=SampleGermlineIGHV, novel=SampleNovel,
                      find_unmutated=TRUE)
```

insertPolymorphisms *Insert polymorphisms into a nucleotide sequence*

Description

`insertPolymorphisms` replaces nucleotides in the desired locations of a provided sequence.

Usage

```
insertPolymorphisms(sequence, positions, nucleotides)
```

Arguments

`sequence`	starting nucletide sequence.
`positions`	numeric vector of positions which to be changed.
`nucleotides`	character vector of nucletides to which to change the positions.

Value

A sequence with the desired nucleotides in the provided locations.

Examples

```
insertPolymorphisms("HUGGED", c(1, 6, 2), c("T", "R", "I"))
```

plotGenotype *Show a colorful representation of a genotype*

Description

`plotGenotype` plots a genotype table.

Usage

```
plotGenotype(genotype, facet_by = NULL, gene_sort = c("name",
  "position"), text_size = 12, silent = FALSE, ...)
```

Arguments

genotype	a data.frame of alleles denoting a genotype, as returned by inferGenotype.
facet_by	a column name in genotype to facet the plot by. If NULL, then do not facet the plot.
gene_sort	a string defining the method to use when sorting alleles. If "name" then sort in lexicographic order. If "position" then sort by position in the locus, as determined by the final two numbers in the gene name.
text_size	the point size of the plotted text.
silent	if TRUE do not draw the plot and just return the ggplot object; if FALSE draw the plot.
...	additional arguments to pass to ggplot2::theme.

Value

A ggplot object defining the plot.

See Also

inferGenotype

Examples

```
# Plot genotype
plotGenotype(SampleGenotype)

# Facet by subject
genotype_a <- genotype_b <- SampleGenotype
genotype_a$SUBJECT <- "A"
genotype_b$SUBJECT <- "B"
geno_sub <- rbind(genotype_a, genotype_b)
plotGenotype(geno_sub, facet_by="SUBJECT", gene_sort="pos")
```

---

plotNovel *Visualize evidence of novel V alleles*

Description

plotNovel is be used to visualize the evidence of any novel V alleles found using findNovelAlleles. It can also be used to visualize the results for alleles that did

Usage

```
plotNovel(data, novel_row, v_call = "V_CALL", j_call = "J_CALL",
  seq = "SEQUENCE_IMGT", junction = "JUNCTION",
  junction_length = "JUNCTION_LENGTH", ncol = 1)
```

Arguments

data	a data.frame in Change-O format. See findNovelAlleles for details.
novel_row	a single row from a data frame as output by findNovelAlleles that contains a polymorphism-containing germline allele
v_call	name of the column in data with V allele calls. Default is "V_CALL".
j_call	name of the column in data with J allele calls. Default is J_CALL.
seq	name of the column in data with the aligned, IMGT-numbered, V(D)J nucleotide sequence. Default is SEQUENCE_IMGT.
junction	Junction region nucleotide sequence, which includes the CDR3 and the two flanking conserved codons. Default is JUNCTION.
junction_length	Number of junction nucleotides in the junction sequence. Default is JUNCTION_LENGTH.
ncol	number of columns to use when laying out the plots

Details

The first panel in the plot shows, for all sequences which align to a particular germline allele, the mutation frequency at each postion along the aligned sequece as a function of the sequence-wide mutation. Sequences that pass the novel allele test are colored red, while sequences that don't pass the test are colored yellow. The second panel shows the nucleotide usage at the positions as a function of sequence-wide mutation count.

To avoid cases where a clonal expansion might lead to a false positive, tigger examines the combinations of J gene and junction length among sequences which perfectly match the proposed germline allele.

Examples

```
# Plot the evidence for the first (and only) novel allele in the example data
novel <- selectNovel(SampleNovel)
plotNovel(SampleDb, novel[1, ])
```

---

readIgFasta *Read immunoglobulin sequences*

Description

readIgFasta reads a fasta-formatted file of immunoglobulin (Ig) sequences and returns a named vector of those sequences.

Usage

```
readIgFasta(fasta_file, strip_down_name = TRUE, force_caps = TRUE)
```

Arguments

fasta_file	fasta-formatted file of immunoglobuling sequences.
strip_down_name	if TRUE, will extract only the allele name from the strings fasta file's sequence names.
force_caps	if TRUE, will force nucleotides to uppercase.

Value

Named vector of strings respresenting Ig alleles.

See Also

writeFasta to do the inverse.

---

reassignAlleles *Correct allele calls based on a personalized genotype*

Description

reassignAlleles uses a subject-specific genotype to correct correct preliminary allele assignments of a set of sequences derived from a single subject.

Usage

```
reassignAlleles(data, genotype_db, v_call = "V_CALL",
  seq = "SEQUENCE_IMGT", method = "hamming", path = NA,
  keep_gene = c("gene", "family", "repertoire"))
```

Arguments

data	a data.frame containing V allele calls from a single subject and the sample IMGT-gapped V(D)J sequences under "SEQUENCE_IMGT".
genotype_db	a vector of named nucleotide germline sequences matching the calls detailed in allele_calls and personalized to the subject
v_call	name of the column in data with V allele calls. Default is "V_CALL".
seq	name of the column in data with the aligned, IMGT-numbered, V(D)J nucleotide sequence. Default is SEQUENCE_IMGT
method	the method to be used when realigning sequences to the genotype_db sequences. Currently, only "hammming" (for Hamming distance) is implemented.
path	directory containing the tool used in the realignment method, if needed. Hamming distance does not require a path to a tool.
keep_gene	a string indicating if the gene ("gene"), family ("family") or complete repertoire ("repertoire") assignments should be performed. Use of "gene" increases speed by minimizing required number of alignments, as gene level assignments will be maintained when possible.

Details

In order to save time, initial gene assignments are preserved and the allele calls are chosen from among those provided in `genotype_db`, based on a simple alignment to the sample sequence.

Value

A modifed input `data.frame` containing the best allele call from among the sequences listed in `genotype_db` in the `V_CALL_GENOTYPED` column.

Examples

```
# Extract the database sequences that correspond to the genotype
genotype_db <- genotypeFasta(SampleGenotype, SampleGermlineIGHV, novel=SampleNovel)

# Use the personlized genotype to determine corrected allele assignments
output_db <- reassignAlleles(SampleDb, genotype_db)
```

SampleDb *Example human immune repertoire data*

Description

A `data.frame` of example V(D)J immunoglobulin sequences derived from a single individual (PGP1), sequenced on the Roche 454 platform, and assigned by IMGT/HighV-QUEST to IGHV1 family alleles.

Format

A `data.frame` where rows correspond to unique V(D)J sequences and columns include:

- `"SEQUENCE_IMGT"`: IMGT-gapped V(D)J nucleotide sequence.
- `"V_CALL"`: IMGT/HighV-QUEST V segment allele calls.
- `"D_CALL"`: IMGT/HighV-QUEST D segment allele calls.
- `"J_CALL"`: IMGT/HighV-QUEST J segment allele calls.
- `"JUNCTION_LENGTH"`: Junction region length.

References

1. Gadala-Maria, et al. (2015) Automated analysis of high-throughput B cell sequencing data reveals a high frequency of novel immunoglobulin V gene segment alleles. PNAS. 112(8):E862-70.

See Also

See airrDb for an AIRR formatted version of `SampleDb`.

SampleGenotype *Example genotype inferrence results*

Description

A `data.frame` of genotype inference results from inferGenotype after novel allele detection via findNovelAlleles. Source data was a collection of V(D)J immunoglobulin sequences derived from a single individual (PGP1), sequenced on the Roche 454 platform, and assigned by IMGT/HighV-QUEST to IGHV1 family alleles.

Format

A `data.frame` where rows correspond to genes carried by an individual and columns lists the alleles of those genes and their counts.

References

1. Gadala-Maria, et al. (2015) Automated analysis of high-throughput B cell sequencing data reveals a high frequency of novel immunoglobulin V gene segment alleles. PNAS. 112(8):E862-70.

See Also

See inferGenotype for detailed column descriptions.

SampleGermlineIGHV *Example Human IGHV germlines*

Description

A `character` vector of all 344 human IGHV germline gene segment alleles in IMGT/GENE-DB release 201408-4.

Format

Values correspond to IMGT-gaped nuceltoide sequences (with nucleotides capitalized and gaps represented by ".") while names correspond to stripped-down IMGT allele names (e.g. "IGHV1-18*01").

References

1. Xochelli, et al. (2014) Immunoglobulin heavy variable (IGHV) genes and alleles: new entities, new names and implications for research and prognostication in chronic lymphocytic leukaemia. Immunogenetics. 67(1):61-6.

SampleNovel *Example novel allele detection results*

Description

A `data.frame` of novel allele detection results from findNovelAlleles. Source data was a collection of V(D)J immunoglobulin sequences derived from a single individual (PGP1), sequenced on the Roche 454 platform, and assigned by IMGT/HighV-QUEST to IGHV1 family alleles.

Format

A `data.frame` where rows correspond to alleles checked for polymorphisms and columns give results as well as paramaters used to run the test.

References

1. Gadala-Maria, et al. (2015) Automated analysis of high-throughput B cell sequencing data reveals a high frequency of novel immunoglobulin V gene segment alleles. PNAS. 112(8):E862-70.

See Also

See findNovelAlleles for detailed column descriptions.

selectNovel *Select rows containing novel alleles*

Description

`selectNovel` takes the result from findNovelAlleles and selects only the rows containing unique, novel alleles.

Usage

```
selectNovel(novel, keep_alleles = FALSE)
```

Arguments

`novel`	a `data.frame` of the type returned by findNovelAlleles.
`keep_alleles`	a `logical` indicating if different alleles leading to the same novel sequence should be kept. See Details.

Details

If, for instance, subject has in his genome IGHV1-2*02 and a novel allele equally close to IGHV1-2*02 and IGHV1-2*05, the novel allele may be detected by analyzing sequences that best align to either of these alleles. If keep_alleles is TRUE, both polymorphic allele calls will be retained. In the case that multiple mutation ranges are checked for the same allele, only one mutation range will be kept in the output.

Value

A data.frame containing only unique, novel alleles (if any) that were in the input.

Examples

```
novel <- selectNovel(SampleNovel)
```

---

sortAlleles *Sort allele names*

Description

sortAlleles returns a sorted vector of strings respresenting Ig allele names. Names are first sorted by gene family, then by gene, then by allele. Duplicated genes have their alleles are sorted as if they were part of their non-duplicated counterparts (e.g. IGHV1-69D*01 comes after IGHV1-69*01 but before IGHV1-69*02), and non-localized genes (e.g. IGHV1-NL1*01) come last within their gene family.

Usage

```
sortAlleles(allele_calls, method = c("name", "position"))
```

Arguments

allele_calls	a vector of strings respresenting Ig allele names.
method	a string defining the method to use when sorting alleles. If "name" then sort in lexicographic order. If "position" then sort by position in the locus, as determined by the final two numbers in the gene name.

Value

A sorted vector of strings respresenting Ig allele names.

See Also

Like sortAlleles, updateAlleleNames can help format a list of allele names.

Examples

```
# Create a list of allele names
alleles <- c("IGHV1-69D*01","IGHV1-69*01","IGHV1-2*01","IGHV1-69-2*01",
             "IGHV2-5*01","IGHV1-NL1*01", "IGHV1-2*01,IGHV1-2*05",
             "IGHV1-2", "IGHV1-2*02", "IGHV1-69*02")

# Sort the alleles by name
sortAlleles(alleles)

# Sort the alleles by position in the locus
sortAlleles(alleles, method="pos")
```

---

subsampleDb *Subsample repertoire*

Description

`subsampleDb` will sample the same number of sequences for each gene, family or allele (specified with `mode`) in `data`. Samples or subjects can be subsampled indepently by setting `group`.

Usage

```
subsampleDb(data, gene = "V_CALL", mode = c("gene", "allele",
  "family"), min_n = 1, max_n = NULL, group = NULL)
```

Arguments

`data`	a `data.frame` in Change-O format.
`gene`	name of the column in `data` with allele calls
`mode`	one of c("gene", "family", "allele") defining the degree of specificity regarding allele calls when subsetting sequences. Determines how `data` will be split into subsets from which the same number of sequences will be subsampled. See also `group`.
`min_n`	minimum number of observations to sample from each groupe. A group with less observations than the minimum is excluded.
`max_n`	maximum number of observations to sample for all `mode` groups. If `NULL`, it will be set automatically to the size of the smallest group. If `max_n` is larger than the availabe number of sequences for any `mode` group, if will be automatically adjusted and the efective `max_n` used will be the size of the smallest `mode` group.
`group`	columns containing additional grouping variables, e.g. sample_id. These groups will be subsampled independently. If `max_n` is `NULL`, a `max_n` will be automatically set for each `group`.

Details

data will be split into gene, allele or family subsets (mode) from which the same number of sequences will be subsampled. If mode=gene, for each gene in the field gene from data, a maximum of max_n sequences will be subsampled. Input sequences that have multiple gene calls (ties), can be subsampled from any of their calls, but these duplicated samplings will be removed, and the final subsampled data will contain unique rows.

Value

A data.frame, subsampled from data.

See Also

selectNovel

Examples

```
# subsampleDb(SampleDb)
```

---

tigger *tigger*

Description

Here we provide a **T**ool for **I**mmuno**g**lobulin **G**enotype **E**lucidation via **R**ep-Seq (TIgGER). TIgGER inferrs the set of Ig alleles carried by an individual (including any novel alleles) and then uses this set of alleles to correct the initial assignments given to sample sequences by existing tools.

Details

Immunoglobulin repertoire sequencing (AIRR-Seq, Rep-Seq) data is currently the subject of much study. A key step in analyzing these data involves assigning the closest known V(D)J germline alleles to the (often somatically mutated) sample sequences using a tool such as IMGT/HighV-QUEST. However, if the sample utilizes alleles not in the germline database used for alignment, this step will fail. Additionally, this alignment has an associated error rate of ~5 mutations. The purpose of TIgGER is to address these issues.

Allele detection and genotyping

- findNovelAlleles: Detect novel alleles.
- plotNovel: Plot evidence of novel alleles.
- inferGenotype: Infer an Ig genotype using a frequency approach.
- inferGenotypeBayesian: Infer an Ig genotype using a Bayesian approach.
- plotGenotype: A colorful genotype visualization.
- genotypeFasta: Convert a genotype to sequences.
- reassignAlleles: Correct allele calls.
- generateEvidence: Generate evidence for the genotype and allele detection inferrence.

Mutation handling

- getMutatedPositions: Find mutation locations.
- getMutCount: Find distance from germline.
- findUnmutatedCalls: Subset unmutated sequences.
- getPopularMutationCount: Find most common sequence's mutation count.
- insertPolymorphisms: Insert SNPs into a sequence.

Input, output and formatting

- readIgFasta: Read a fasta file of Ig sequences.
- updateAlleleNames: Correct outdated allele names.
- sortAlleles: Sort allele names intelligently.
- cleanSeqs: Standardize sequence format.

References

1. Gadala-Maria, et al. (2015) Automated analysis of high-throughput B cell sequencing data reveals a high frequency of novel immunoglobulin V gene segment alleles. PNAS. 112(8):E862-70.

`updateAlleleNames` *Update IGHV allele names*

Description

`updateAlleleNames` takes a set of IGHV allele calls and replaces any outdated names (e.g. IGHV1-f) with the new IMGT names.

Usage

```
updateAlleleNames(allele_calls)
```

Arguments

`allele_calls` a vector of strings respresenting IGHV allele names.

Value

Vector of strings respresenting updated IGHV allele names.

Note

IGMT has removed `IGHV2-5*10` and `IGHV2-5*07` as it has determined they are actually alleles `02` and `04`, respectively. The updated allele names are based on IMGT release 201408-4.

References

1. Xochelli et al. (2014) Immunoglobulin heavy variable (IGHV) genes and alleles: new entities, new names and implications for research and prognostication in chronic lymphocytic leukaemia. Immunogenetics. 67(1):61-6

See Also

Like `updateAlleleNames`, sortAlleles can help format a list of allele names.

Examples

```
# Create a vector that uses old gene/allele names.
alleles <- c("IGHV1-c*01", "IGHV1-f*02", "IGHV2-5*07")

# Update the alleles to the new names
updateAlleleNames(alleles)
```

---

writeFasta *Write to a fasta file*

Description

`writeFasta` writes a named vector of sequences to a file in fasta format.

Usage

```
writeFasta(named_sequences, file, width = 60, append = FALSE)
```

Arguments

`named_sequences`	a vector of named string representing sequences
`file`	the name of the output file.
`width`	the number of characters to be printed per line. if not between 1 and 255, width with be infinite.
`append`	`logical` indicating if the output should be appended to `file` instead of overwriting it

Value

A named vector of strings respresenting Ig alleles.

See Also

readIgFasta to do the inverse.