Package ‘MADGiC’

January 19, 2015

Type Package

Title Fits an empirical Bayesian hierarchical model to obtain
posterior probabilities that each gene is a driver.

Description The model accounts for (1) frequency of mutation compared to a
sophisticated background model that accounts for gene-specific factors in
addition to mutation type and nucleotide context, (2) predicted
functional impact (in the form of SIFT scores) of each specific change,and (3) oncogenic patterns in the position of mutations that have been
deposited into the COSMIC (Catalogue of Somatic Mutations in Cancer) database.

Depends R (>= 2.15.2), abind, biomaRt, data.table, rtracklayer,splines, stats

Version 0.3

Author Keegan Korthauer

Maintainer Keegan Korthauer <kdkorthauer@wisc.edu>

License GPL-3

LazyData false

Collate 'MADGiC.R'

R topics documented:

`calculate.bij` *Calculate the background mutation probability for every gene/sample combination*

Description

This function calcuates the empirical Bayes estimate of a mutation occurring in every gene and sample according to the background mutation model

Usage

```
calculate.bij(gid, gene, p, s, epsilon, delta,
  exome.constants.mult, sample.name, muttable,
  nonsil.mut.type, a, b, uniqueA, tableA)
```

Arguments

`gid` a character vector containing the Ensembl gene ids of all genes

`exome.constants.mult`
an object returned by `multiply.p.s.epsilon`

`sample.name` a character vector containing the names of all samples `muttable` a matrix containing the likelihood due to existing background mutations, see `mut.lik.change`

`nonsil.mut.type` a matrix with one row for every observed mutation, with 10 columns in this order: Ensemble gene id, chromosome, position, mutation type (SNP or In_frame or Frame_shift), reference allele, tumor allele 1, tumor allele 2, sample id, mutation type (1=transition or 2=transversion), and SIFT score

`a` numeric value for the maximum likelihood estimate of the hyperparameter a representing the prior for q_j (sample-specific mutation rate, q_j~Unif(a,b))

`b` numeric value for the maximum likelihood estimate of the hyperparameter b representing the prior for q_j (sample-specific mutation rate, q_j~Unif(a,b))

`uniqueA` a numeric vector containing all unique values of A (see `getA`)

`tableA` a numeric vector of the same length of `uniqueA` that contains the number of instances of each value in `uniqueA`

`gene` a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).

`p` a vector of length 7 containing the mutation type relative rate parameters in the following order: A/T transition, A/T transversion, non-CpG transition, non-CpG transversion, CpG transition, CpG transversion, and Indel.

`s` a vector of length 3 containing the relative rate parameters for the three replication timing regions (1=Early, 2=Middle, 3=Late)

`epsilon` a vector of length 3 containing the relative rate parameters for the three expression levels (1=Low, 2=Medium, 3=High)

`delta` vector of length 2 where the second element represents the relative rate of mutation in olfactory receptor (OR) genes compared to all others within a similar replication timing and expression level category. First element set to 1 (reference category).

Details

the return value is only calculated for genes with two or more mutations to save on computation time.

Value

a list with one entry for each gene where each entry contains a vector with the probability of mutation for each sample

Note

This internal function is not intended to be called by the user.

calculate.post.probs *Calculate posterior probability of being a driver*

Description

This function calculates the posterior probability for every gene being a driver

Usage

```
calculate.post.probs(bij, sample.name, nonsil.mut.type,
  gid, exome.SIFT, f0, f1, alpha = 0.2, beta = 6, p0)
```

Arguments

bij	list object with one entry per gene, where each entry is a numeric vector containing 1-the probability of a mutation in each sample under the background mutation model
nonsil.mut.type	table of mutations, object obtained from mut.type.converter
f0	numeric vector containing the estimated null density of FI scores obtained from locfdr
f1	numeric vector containing the estimated non-null density of FI scores obtained from locfdr
alpha	numeric value of first shape parameter of the prior Beta distribution on the probability of mutation for driver genes. Default value of 0.2 is chosen as a compromise between a cancer type with a relatively low mutation rate (Ovarian cancer, fitted value from COSMIC of 0.15) and one with a comparatively high mutation rate (Squamous cell lung, fitted value from COSMIC of 0.27), but results are robust to changes in this parameter. Note that intuitively (and empirically), a higher mutation rate overall leads to a higher driver mutation rate overall - and thus less mass is concentrated in the left tail of the distribution.
beta	numeric value of second shape parameter of the prior Beta distribution on the probability of mutation for driver genes. Default value of 6 is chosen as a compromise between a cancer type with a relatively low mutation rate (Ovarian cancer, fitted value from COSMIC of 6.6) and one with a comparatively high mutation rate (Squamous cell lung, fitted value from COSMIC of 5.83), but results are robust to changes in this parameter. Note that intuitively (and empirically), a higher mutation rate overall leads to a higher driver mutation rate overall - and thus less mass is concentrated in the left tail of the distribution.
exome.SIFT	list object with one item per chromosome where each item contains matrix with one row per coding base pair and 7 columns: position, nucleotide, CpG context, FI score for mutation to "A", FI score for mutation to "C", FI score for mutation to "G", and FI score for mutation to "T".
p0	vector of prior probabilities for each gene being a driver. This is precomputed from COSMIC and loaded into package. See paper for details.

`sample.name` a character vector containing the names of all samples

`gid` a character vector containing the Ensembl gene ids of all genes

Value

named vector of posterior probabilities that each gene is a driver.

Note

This internal function is not intended to be called by the user.

`compRatio` *Cubic spline logistic regression*

Description

Performs the cubic spline logistic regression for use in the `locfdr` function

Usage

```
compRatio(center, succ, fail, df = 5)
```

Arguments

`center` numeric vector of the midpoints of the bins

`succ` numeric vector of the same length of `center` that corresponds to the number of 'successes' (fulls) in each bin

`fail` umeric vector of the same length of `center` that corresponds to the number of 'failures' (nulls) in each bin

`df` the number of degrees of freedom for the spline regression (defaults to 5)

Value

list object with four elements:

`z` numeric vector of midpoints

`probs` numeric vector of fitted probabilities of success

`ratio` numeric vector of the ratio of successes to failures, accounting for total numbers of successes and failures

`df` the number of degrees of freedom used for the spline regression

Note

This internal function is not intended to be called by the user.

`convert.hgnc.to.ensembl` *Convert HGNC symbols to Ensembl identifiers using biomaRt*

Description

This function takes an MAF table and returns a vector of Ensembl identifiers corresponding to the HGNC names in the "Hugo_Symbol" column.

Usage

```
convert.hgnc.to.ensembl(table, ensembl)
```

Arguments

`table`	a matrix of MAF data that contains a column named "Hugo_Symbol" with the HGNC names
`ensembl`	an object created by `useMart` in the `biomaRt` package which tells the function which version of Ensembl to use

Value

a vector of Ensembl identifiers corresponding to the HGNC symbols, obtained from `biomaRt`.

Note

This internal function is not intended to be called by the user.

`convert.seq.to.num` *Convert nucleotide sequence to numbers*

Description

This function converts nucleotide sequence to numbers for the efficiency of calculation

Usage

```
convert.seq.to.num(x)
```

Arguments

`x`	a character vector containing any number of "A", "T", "G", and "C".

Value

a vector of integers where 1 corresponds to "A", 2 corresponds to "T", 3 corresponds to "G", and 4 corresponds to "C". A 0 is put in place of characters that are missing or not one of the four nucleotides.

Note

This internal function is not intended to be called by the user.

`dCG.type` *Get mutation type (5-6) of CpG mutations*

Description

Given a sequence of two characters, the function returns the type of mutation: transition from CpG G/C (5), or transversion from CpG G/C (6).

Usage

```
dCG.type(x)
```

Arguments

`x`	a character of length two where each character is either "A","T","G", or "C". Only the 6 two letter codes that begin with "C" or "G" are allowed. For example, x could be "CT".

Details

used internally in `mut.type.converter`

Value

an integer indicator of mutation type, where 1=transition from CpG C/G and 2=transversion from CpG C/G. For example x="AG" would return 1 (for transition).

Note

This internal function is not intended to be called by the user.

exome *Exome annotation for human genome build NCBI 36/hg18*

Description

This data set contains a list with an item for each chromosome where each item is a matrix with a row for each position and 15 columns that contain information about how many of each type of mutation are possible, what their FI (functional impact, here we use SIFT scores) are, and whether each type of change is nonsilent. This object is broken down into 3 objects in the function `get.post.probs`, each containing columns 1, 2, and 7: `exome.constants` (and columns 3-6), `exome.SIFT` (and columns 8:11), and `exome.nonsil` (and columns 12-15).

Format

Each list item (one per chromosome) contains a matrix with one row per position and the following 15 columns: 1 - base pair position, 2 -ame nucleotide (integer representation, see `convert.seq.to.num`), 3 - number of possible nonsilent transitions, 4 - number of possible nonsilent transversions, 5 - number of possible silent transitions, 6 - number of possible silent transversions, 7 - indicator of whether position is in a CpG dinucleotide, 8 - FI score for mutation to "A", 9 -FI score for mutation to "C", 10 - FI score for mutation to "G", 11 - FI score for mutation to "T", 12 - nonsilent indicator (1=nonsilent, 0=silent) for mutation to "A", 13 - nonsilent indicator for mutation to "C", 14 - nonsilent indicator for mutation to "G", and 15 - nonsilent indicator for mutation to "T".

find.prior.param *Find the maximum likelihood values for the hyperparameters* `a` *and* `b`

Description

Calculates the value of (a,b) that maximizes the likelihood of q_j (sample-specific background mutation rate) given the relative rate parameters.

Usage

```
find.prior.param(x, muttable, uniqueA, tableA, S)
```

Arguments

`x`	a vector of length two with the initial guess for `a` and `b`. Note that these values are on the log scale, i.e. a lower bound of 5E-8 would correspond to a=-16.8.
`S`	an integer corresponding to the number of samples in the dataset.
`muttable`	a matrix containing the likelihood due to existing background mutations, see `mut.lik.change`
`uniqueA`	a numeric vector containing all unique values of A (see `getA`)
`tableA`	a numeric vector of the same length of `uniqueA` that contains the number of instances of each value in `uniqueA`

Value

a list containing two items:

a	numeric value for the maximum likelihood estimate of the hyperparameter a representing the prior for q_j (sample-specific mutation rate, q_j~Unif(a,b))
b	numeric value for the maximum likelihood estimate of the hyperparameter a representing the prior for q_j (sample-specific mutation rate, q_j~Unif(a,b))

Note

This internal function is not intended to be called by the user.

fit.background.model *Fit background mutation model*

Description

This function calculates the relative rate parameters of the background mutation model, estimated by method of moments

Usage

```
fit.background.model(mutab, nonsil.mut.type.sampl.sum,
  sil.mut.type.sampl.sum, nonsil.type.const,
  sil.type.const, gene)
```

Arguments

mutab	a matrix containing one row per mutation and 8 columns (Ensembl gene name, chromosome, position, variant type (SNP, In_frame, Frame_shift), reference allele, tumor allele 1, tumor allele 2, and sample id.
nonsil.mut.type.sampl.sum	a 3 (expression category) by 3(replication timing category) by 6 (mutation type) matrix containing the total number of base pairs eligible for a nonsilent mutation in each category (2nd item obtained from mut.type.converter)
sil.mut.type.sampl.sum	a 3 (expression category) by 3(replication timing category) by 6 (mutation type) matrix containing the total number of base pairs eligible for a silent mutation in each category (2nd item obtained from mut.type.converter)
nonsil.type.const	list of 3 objects with information about nonsilent coding sequences (see preprocess.BM)
sil.type.const	list of 3 objects with information about silent coding sequences (see preprocess.BM)
gene	a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).

Value

a list object with four elements:

`p` a vector of length 7 containing the mutation type relative rate parameters in the following order: A/T transition, A/T transversion, non-CpG transition, non-CpG transversion, CpG transition, CpG transversion, and Indel.

`r` numeric value representing the relative rate parameter estimate for the ratio of mutations in genes with nonsilent vs only silent mutations (selection bias)

`s` a vector of length 3 containing the relative rate parameters for the three replication timing regions (1=Early, 2=Middle, 3=Late)

`epsilon` a vector of length containing the relative rate parameters for the three expression levels (1=Low, 2=Medium, 3=High)

`delta` a vector of length 2 where the second element represents the relative rate of mutation in olfactory receptor (OR) genes compared to all others within a similar replication timing and expression level category. First element set to 1 (reference category).

Note

This internal function is not intended to be called by the user.

`gene.rep.expr` *Gene annotation list*

Description

This dataset contains a list with an item for each gene (18,926) that contains information about the basepairs, length, replication timing and expression level.

Format

Each gene is a 5 item list: 1. Ensembl ID, 2. chromosome, 3. coding base pairs, 4. replication timing category (1=Early, 2=Middle, 3=Late), and 5. expression level category (1=Low, 2=Medium, 3=High).

`generate.sel.exome` *Select the part of exome.constants that belongs to a list of genes*

Description

A function to pull only those base pairs that reside within a list of genes from the object `exome.constants`

Usage

```
generate.sel.exome(genelist, gene, exome.constants)
```

Arguments

`genelist`	a vector containing gene names to subset on
`gene`	a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).
`exome.constants`	a list with one entry for each chromosome, where each entry is a matrix containing a row for each coding base pair and the following columns: 1. base pair position, 2. nucleotide number (see `convert.seq.to.num`), 3. number of possible nonsilent transitions, 4. number of possible nonsilent transversions, 5. number of possible silent transitions, 6. number of possible silent transversions, and 7. whether the position is in a CpG dinucleotide.

Value

an object of the same structure as `exome.constants`, but only containing positions that reside in the genes of `genelist`

Note

This internal function is not intended to be called by the user.

`gene_names` *Ensembl names for all genes in the exome*

Description

This data set is a text file that contains all the Ensembl names of coding genes in build 36.

Format

one Ensembl name per line ("ENSG###########") for 19,238 lines.

Source

Distributed with code of Youn and Simon 2011.

get.post.probs *Main posterior probability calculation*

Description

This function reads in an MAF data file, exome annotation, and pre-computed prior information and then fits a hierarchical emprical Bayesian model to obtain posterior probabilities that each gene is a driver.

Usage

```
get.post.probs(maf.file,
  exome.file = system.file("data/exome_36.RData", package = "MADGiC"),
 gene.rep.expr.file = system.file("data/gene.rep.expr.RData", package = "MADGiC"),
  gene.names.file = system.file("data/gene_names.txt", package = "MADGiC"),
  prior.file = system.file("data/prior.RData", package = "MADGiC"),
  alpha = 0.2, beta = 6, N = 20, replication.file = NULL,
  expression.file = NULL)
```

Arguments

`maf.file` name of an MAF (Mutation Annotation Format) data file containing the somatic mutations. Currently, NCBI builds 36 and 37 are supported.

`exome.file` name of an .RData file that annotates every position of the exome for how many transitions/transversions are possible, whether each change is silent or nonsilent, and the SIFT scores for each possible change

`gene.rep.expr.file` name of an .RData file that annotates every gene for its Ensembl name, chromosome, base pair positions, replication timing region, and expression level.

`gene.names.file` name of a text file containing the Ensembl names of all genes.

`prior.file` name of an .RData file containing a named vector of prior probabilities that each gene is a driver, obtained from positional information in the COSMIC database.

`N` integer number of simulated datasets to be used in the estimation of the null distribution of functional impact scores. The default value is 20 (see `shuffle.muts`).

`expression.file` (optional) name of a .txt file containing gene expression data if user wishes to supply one (default is to use an average expression signal of the CCLE). The .txt file should have two columns and no header. The first column should contain the Ensembl Gene ID (using Ensembl 54 for hg18) and the second column should contain the expression measurements. These can be raw or log-scaled but should be normalized if normalization is desired.

`replication.file` (optional) name of a .txt file containing replication timing data if user wishes to supply one (default is to use data from Chen et al. (2010)). The .txt file should have two columns and no header. The first column should contain the Ensembl Gene ID (using Ensembl 54 for hg18) and the second column should contain the replication timing measurements.

`alpha`	numeric value of first shape parameter of the prior Beta distribution on the probability of mutation for driver genes. Default value of 0.2 is chosen as a compromise between a cancer type with a relatively low mutation rate (Ovarian cancer, fitted value from COSMIC of 0.15) and one with a comparatively high mutation rate (Squamous cell lung, fitted value from COSMIC of 0.27), but results are robust to changes in this parameter. Note that intuitively (and empirically), a higher mutation rate overall leads to a higher driver mutation rate overall - and thus less mass is concentrated in the left tail of the distribution.
`beta`	numeric value of second shape parameter of the prior Beta distribution on the probability of mutation for driver genes. Default value of 6 is chosen as a compromise between a cancer type with a relatively low mutation rate (Ovarian cancer, fitted value from COSMIC of 6.6) and one with a comparatively high mutation rate (Squamous cell lung, fitted value from COSMIC of 5.83), but results are robust to changes in this parameter. Note that intuitively (and empirically), a higher mutation rate overall leads to a higher driver mutation rate overall - and thus less mass is concentrated in the left tail of the distribution.

Details

The typical user only need specify the MAF file they wish to analyze. The other fields (exome annotation, gene annotation, gene names, and prior probabilities) have been precomputed and distributed with this package.

Value

a named vector of posterior probabilities that each gene is a driver

Examples

```
## Not run:

# pointer to the MAF file to be analyzed
maf.file <- system.file("data/OV.maf",package="MADGiC")

# calculation of posterior probabilities that each gene is a driver
post.probs <- get.post.probs(maf.file)

# Modify default settings to match TCGA ovarian analysis in paper
post.probs <- get.post.probs(maf.file, N=100, alpha=0.15, beta=6.6)

## End(Not run)
```

getA *Compute A - the likelihood of a mutation at every position*

Description

This function computes the likelihood of a mutation occurring at each position in the exome according to the background mutation model

Usage

```
getA(nonsil.exclus, sil.exclus, both, p, r, s, epsilon,
  delta, gene)
```

Arguments

`nonsil.exclus`	oject obtained from `generate.sel.exome` representing the exclusively nonsilent genes
`sil.exclus`	oject obtained from `generate.sel.exome` representing the exclusively silent genes
`both`	oject obtained from `generate.sel.exome` representing the genes with nonsilent mutations also used for silent mutation detection
`r`	numeric value representing the relative rate parameter estimate for the ratio of mutations in genes with nonsilent vs only silent mutations (selection bias)
`p`	a vector of length 7 containing the mutation type relative rate parameters in the following order: A/T transition, A/T transversion, non-CpG transition, non-CpG transversion, CpG transition, CpG transversion, and Indel.
`s`	a vector of length 3 containing the relative rate parameters for the three replication timing regions (1=Early, 2=Middle, 3=Late)
`epsilon`	a vector of length 3 containing the relative rate parameters for the three expression levels (1=Low, 2=Medium, 3=High)
`delta`	vector of length 2 where the second element represents the relative rate of mutation in olfactory receptor (OR) genes compared to all others within a similar replication timing and expression level category. First element set to 1 (reference category).
`gene`	a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).

Value

a numeric vector containting the likelihood of a mutation at every position.

Note

This internal function is not intended to be called by the user.

locfdr *Estimation of non-null density*

Description

This function utilizes the algorithm for local false discovery rate (locfdr) estimation from Efron (2001) to estimate the non-null density of SIFT scores. Null scores are obtained from the simulated datasets created using `shuffle.muts`

Usage

```
locfdr(nonsil.mutab, exome.SIFT, dir = "./simulated",
  N = 100)
```

Arguments

`dir`	a character string that contains the directory path to the simulated datasets
`N`	a numeric value that corresponds to how many simulated datasets are contained in the directory `dir`
`nonsil.mutab`	a matrix containing one row per nonsilent mutation and 8 columns (Ensembl gene name, chromosome, position, variant type (SNP, In_frame, Frame_shift), reference allele, tumor allele 1, tumor allele 2, and sample id.
`exome.SIFT`	list object with one item per chromosome where each item contains matrix with one row per coding base pair and 7 columns: position, nucleotide, CpG context, FI score for mutation to "A", FI score for mutation to "C", FI score for mutation to "G", and FI score for mutation to "T".

Value

a list object with three elements:

`f0`	numeric vector containing the estimated null density of FI scores over `nbins` bins
`f1`	numeric vector containing the estimated non-null density of FI scores over `nbins` bins
`nbins`	numeric value that represents how many bins were used for the spline regression

Note

This internal function is not intended to be called by the user.

lupost.vec *Likelihood of mutations in a single gene and sample given the background model*

Description

Calculates, for a given gene and sample, the likelihood of mutations in a given gene under the background mutation model and current value of the sample-specific mutation rate

Usage

```
lupost.vec(t, mutationexist, C, D, uniqueB, tableB,
  uniqueA, tableA, base)
```

Arguments

`t`	numeric value of sample-specific mutation rate (log-scale)
`mutationexist`	logical value indicating whether to include observed mutations in the current gene
`C`	subset of matrix `muttable` (see `calculate.bij`) containing only those rows corresponding to the current sample
`D`	numeric value corresponding to the sum of the logarithms of the first column of `C`
`uniqueB`	similar to `uniqueA` except restricted to the current gene
`tableB`	similar to `tableA` except restricted to the current gene
`base`	minimum value of the log likelihood of the sample-specific mutation rate q_j given the background model
`uniqueA`	a numeric vector containing all unique values of A (see `getA`)
`tableA`	a numeric vector of the same length of `uniqueA` that contains the number of instances of each value in `uniqueA`

Value

likelihood of mutations in a single gene and sample given the background model and sample-specific mutation rate `t`

Note

This internal function is not intended to be called by the user.

lupost2.vec *Likelihood of sample specific mutation rate given the background model*

Description

Calculates the likelihood of the sample specific mutation rate under the background mutation model.

Usage

```
lupost2.vec(t, mutationexist, C, D, uniqueA, tableA,
  base)
```

Arguments

`uniqueA`	a numeric vector containing all unique values of A (see `getA`)
`tableA`	a numeric vector of the same length of `uniqueA` that contains the number of instances of each value in `uniqueA`
`t`	numeric value of sample-specific mutation rate (log-scale)
`mutationexist`	logical value indicating whether to include observed mutations in the current gene
`C`	subset of matrix `muttable` (see `calculate.bij`) containing only those rows corresponding to the current sample
`D`	numeric value corresponding to the sum of the logarithms of the first column of `C`
`base`	minimum value of the log likelihood of the sample-specific mutation rate q_j given the background model

Value

likelihood of the sample-specific mutation rate q_j under the background mutation model given the observed mutations.

Note

This internal function is not intended to be called by the user.

lupost3.vec *Likelihood of sample specific mutation rate given the background model*

Description

Calculates the log likelihood of the sample specific mutation rate under the background mutation model. Similar to lupost2.vec except that it returns the raw value of the likelihood value instead of the logarithm and it is not normalized with base.

Usage

```
lupost3.vec(t, mutationexist, C, D, uniqueA, tableA)
```

Arguments

t	numeric value of sample-specific mutation rate (log-scale)
mutationexist	logical value indicating whether to include observed mutations in the current gene
C	subset of matrix muttable (see calculate.bij) containing only those rows corresponding to the current sample
D	numeric value corresponding to the sum of the logarithms of the first column of C
uniqueA	a numeric vector containing all unique values of A (see getA)
tableA	a numeric vector of the same length of uniqueA that contains the number of instances of each value in uniqueA

Value

log likelihood of the sample-specific mutation rate q_j under the background mutation model given the observed mutations.

Note

This internal function is not intended to be called by the user.

lupost4.vec *Posterior mean of sample-specific mutation rate given the background model*

Description

Calculates the posterior mean of the sample specific mutation rate under the background mutation model. Similar to lupost2.vec except that it returns the un-normalized value for the posterior mean of the sample-specific rate.

Usage

```
lupost4.vec(t, mutationexist, C, D, uniqueA, tableA,
  base)
```

Arguments

t	numeric value of sample-specific mutation rate (log-scale)
mutationexist	logical value indicating whether to include observed mutations in the current gene
C	subset of matrix muttable (see calculate.bij) containing only those rows corresponding to the current sample
D	numeric value corresponding to the sum of the logarithms of the first column of C
uniqueA	a numeric vector containing all unique values of A (see getA)
tableA	a numeric vector of the same length of uniqueA that contains the number of instances of each value in uniqueA
base	minimum value of the log likelihood of the sample-specific mutation rate q_j given the background model

Value

unormalized posterior mean of the sample-specific mutation rate q_j under the background mutation model given the observed mutations.

Note

This internal function is not intended to be called by the user.

multiply.p.s.epsilon *Multiply the constants in* `exome.constants` *by the relative rate parameters of the background mutation model*

Description

A function to multiply the constants e, f, c, and d in `exome.constants` by the relative rate parameters of the background mutation model. The parameters used depend on the mutation type, nucleotide context of the position, and the replication timing region and expression level of the gene that the position resides in.

Usage

```
multiply.p.s.epsilon(X, p, s, epsilon, delta, gene)
```

Arguments

`X`	a list with one entry for each chromosome, where each entry is a matrix containing a row for each coding base pair and the following columns: 1. base pair position, 2. nucleotide number (see `convert.seq.to.num`), 3. number of possible nonsilent transitions, 4. number of possible nonsilent transversions, 5. number of possible silent transitions, 6. number of possible silent transversions, and 7. whether the position is in a CpG dinucleotide.
`p`	a vector of length 7 containing the mutation type relative rate parameters in the following order: A/T transition, A/T transversion, non-CpG transition, non-CpG transversion, CpG transition, CpG transversion, and Indel.
`s`	a vector of length 3 containing the relative rate parameters for the three replication timing regions (1=Early, 2=Middle, 3=Late)
`epsilon`	a vector of length 3 containing the relative rate parameters for the three expression levels (1=Low, 2=Medium, 3=High)
`delta`	vector of length 2 where the second element represents the relative rate of mutation in olfactory receptor (OR) genes compared to all others within a similar replication timing and expression level category. First element set to 1 (reference category).
`gene`	a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).

Value

an object of the same structure as exome.constants, but columns 3-6 (e, f, c, d) have been multiplied by relative rate parameters `p`, `s`, and `epsilon`.

Note

This internal function is not intended to be called by the user.

multiplyr — *Multiply the nonsilent constants in a subsetted* `exome.constants` *object by the relative rate parameters of the background mutation model*

Description

A function to multiply the constants e and f in a subsetted `exome.constants` object by the relative rate parameter for selection bias of the background mutation model. This parameter represents the ratio of nonsilent to silent mutations

Usage

```
multiplyr(res, r)
```

Arguments

`res`	a list with one entry for each chromosome, where each entry is a matrix containing a row for each coding base pair and the following columns: 1. base pair position, 2. nucleotide number (see `convert.seq.to.num`), 3. number of possible nonsilent transitions, 4. number of possible nonsilent transversions, 5. number of possible silent transitionsget, 6. number of possible silent transversions, and 7. whether the position is in a CpG dinucleotide. Is subsetted by a particular gene list to include only genes used for nonsilent rate estimation (see `generate.sel.exome`)
`r`	numeric value representing the relative rate parameter estimate for the ratio of mutations in genes with nonsilent vs only silent mutations (selection bias)

Value

an object of the same structure as `res`, but columns 3 and 4 (e, and f) have been multiplied by the nonsilent relative rate parameter `r`.

Note

This internal function is not intended to be called by the user.

mut.base — *Convert nucleotide sequence to column numbers in* `exome.SIFT`

Description

This function converts nucleotide sequence to column numbers in the `exome.SIFT` object that they correspond to

Usage

```
mut.base(x)
```

Arguments

x a character vector containing any number of "A", "T", "G", and "C".

Value

a vector of integers where 4 corresponds to "A", 5 corresponds to "T", 6 corresponds to "G", and 7 corresponds to "C".

Note

This internal function is not intended to be called by the user.

mut.lik.change *Calculate change in likelihood due to background mutations*

Description

This function calculates the change in likelihood for each existing mutation in the reformatted tables obtained from mut.type.converter given estimates of the relative rate parameters from fit.background.model.

Usage

```
mut.lik.change(nonsil.mut.type, res, nonsilent, both_log,
  p_inframe, p_frameshift, p, r, s, epsilon, delta, gene)
```

Arguments

nonsil.mut.type
: a reformatted mutation table that contains an extra two columns: 1. a mutation type indicator (1= transition, 2=transversion, and 3=indel), 2. SIFT score for the mutation. Second item returned from mut.type.converter.

res
: a gene list obtained from generate.sel.exome and run through multiplyr for genes used for nonsilent rate estimation

nonsilent
: logical value indicating whether genes are used for nonsilent rate estimation

both_log
: logical value indicating whether genes are used for both silent and nonsilent rate estimation

p_inframe
: numerical value representing the rate if in-frame indel mutations to A/T transitions

p_frameshift
: numerical value representing the rate if frameshift indel mutations to A/T transitions

`p` a vector of length 7 containing the mutation type relative rate parameters in the following order: A/T transition, A/T transversion, non-CpG transition, non-CpG transversion, CpG transition, CpG transversion, and Indel.

`s` a vector of length 3 containing the relative rate parameters for the three replication timing regions (1=Early, 2=Middle, 3=Late)

`epsilon` a vector of length 3 containing the relative rate parameters for the three expression levels (1=Low, 2=Medium, 3=High)

`delta` vector of length 2 where the second element represents the relative rate of mutation in olfactory receptor (OR) genes compared to all others within a similar replication timing and expression level category. First element set to 1 (reference category).

`gene` a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).

`r` numeric value representing the relative rate parameter estimate for the ratio of mutations in genes with nonsilent vs only silent mutations (selection bias)

Value

a matrix containing the likelihood due to existing background mutations, to be utilized in `calculate.bij`

Note

This internal function is not intended to be called by the user.

`mut.type` *Get mutation type (transition or transversion)*

Description

Given a sequence of two characters, the function returns whether the mutation is a transition (1) or a transversion (2)

Usage

```
mut.type(x)
```

Arguments

`x` a character of length two where each character is either "A","T","G", or "C". For example, x could be "AG". 12 possible two-letter codes.

Details

see `mut.type.converter`

Value

an integer indicator of mutation type, where 1=transition and 2=transversion. For example x="AG" would return 1 (for transition).

Note

This internal function is not intended to be called by the user.

---

mut.type.converter — *Get total counts of mutations for each replication timing, expression, and mutation type category and reformatted mutation table*

Description

This function reads in a mutation table and returns the total counts of mutations in each of the replication timing regions, expression level categories, and mutation types. It also returns a reformatted mutation table.

Usage

```
mut.type.converter(mutab, exome.SIFT, seq.in.chrom,
  dCG.in.chrom, gene)
```

Arguments

`seq.in.chrom`	a list with an item for each chromosome that contains a matrix whose first column is the position and the second column is the nucleotide of a base pair within that chromosomes (first item returned from `preprocess.BM`)
`dCG.in.chrom`	a list with an item for each chromosome that contains a vector of the position of the CpG dinucleotide coding sequences (second item returned from `preprocess.BM`)
`mutab`	a matrix containing one row per mutation and 8 columns (Ensembl gene name, chromosome, position, variant type (SNP, In_frame, Frame_shift), reference allele, tumor allele 1, tumor allele 2, and sample id.
`gene`	a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).
`exome.SIFT`	list object with one item per chromosome where each item contains matrix with one row per coding base pair and 7 columns: position, nucleotide, CpG context, FI score for mutation to "A", FI score for mutation to "C", FI score for mutation to "G", and FI score for mutation to "T".

Value

a list object with two elements:

`mut.per.type.sum`
: a 3 by 3 by 6 matrix with the total counts of mutations in each of the 3 replication timing regions, 3 expression level categories, and 6 mutation types (indels counted separately since every base pair is at risk for this type of mutation).

`mutab`
: a reformatted mutation table that contains an extra two columns: 1. a mutation type indicator (1= transition, 2=transversion, and 3=indel), 2. SIFT score for the mutation.

Note

This internal function is not intended to be called by the user.

`no.dCG.type` *Get mutation type (1-4) of non-CpG mutations*

Description

Given a sequence of two characters, the function returns the type of mutation: transition from A/T (1), transversion from A/T (2), transition from non-CpG G/C (3), or transversion from non-CpG G/C (4).

Usage

```
no.dCG.type(x)
```

Arguments

`x`
: a character of length two where each character is either "A","T","G", or "C". For example, x could be "AG". 12 possible two-letter codes.

Details

used internally in `mut.type.converter`

Value

an integer indicator of mutation type, where 1=transition from A/T, 2=transversion from A/T, 3=transition from non-CpG C/G and 4=transversion from non-CpG C/G. For example x="AG" would return 1 (for transition).

Note

This internal function is not intended to be called by the user.

number *Convert chromosome names (character) to numeric*

Description

Function that converts character chromosome names (including those using "X" and "Y") to numeric values 1-24

Usage

```
number(x)
```

Arguments

x	a character, one of "1", "2", ..., "24", "X", "Y"

Value

an integer between 1 and 24. "X" is converted to 23 and "Y" is converted to 24. All other values just change classes from character to numeric ("1" -> 1).

Note

This internal function is not intended to be called by the user.

OV.maf *TCGA Ovarian MAF (Mutation Annotation Format) file*

Description

Contains all somatic mutation data from TCGA Ovarian project, downloaded from the TCGA data portal on October 1, 2013 and collected into one file

Format

See `https://wiki.nci.nih.gov/display/TCGA/Mutation+Annotation+Format+(MAF)+Specification` for details about MAF files.

`preprocess.BM`	*Pull out sequences, CpG dinucleotide positions, and base pairs at risk for mutation subsetted by categories in the background model.*

Description

A function to gather information about a given gene list that will be needed to fit the background mutation model. It pulls out the sequence of the genes in the list `X`, as well as indicates which of the positions resides in a CpG dinucleotide pair, and counts how many base pairs are at risk for mutation in the 108 combinations of 6 mutation types x 2 silent/nonsilent status x 3 replication timing categories x 3 expression level categories.

Usage

```
preprocess.BM(X, gene)
```

Arguments

`X`	a list with one entry for each chromosome, where each entry is a matrix containing a row for each coding base pair and the following columns: 1. base pair position, 2. nucleotide number (see `convert.seq.to.num`), 3. number of possible nonsilent transitions, 4. number of possible nonsilent transversions, 5. number of possible silent transitions, 6. number of possible silent transversions, and 7. whether the position is in a CpG dinucleotide. May be subsetted by a particular gene list (see `generate.sel.exome`)
`gene`	a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).

Value

a list with three items:

`seq.in.chrom`	a list with an item for each chromosome that contains a matrix whose first column is the position and the second column is the nucleotide of a base pair within that chromosomes
`dCG.in.chrom`	position of the CpG dinucleotide coding sequences within chromosomes
`type.const`	a 3 by 12 matrix containing E_1,n,h,F_2,n,h,E_3,n,h,F_4,n,h,E_5,n,h,F_6,n,h, C_1,n,h,D_2,n,h,C_3,n,h,D_4,n,h,C_5,n,h,D_6,n,h for n=1,2,3, and h=1,2,3

Note

This internal function is not intended to be called by the user.

prior *Prior probabilities that each gene is a passenger*

Description

This data set is a named vector of prior probabilities that each gene is a passenger. It was obtained using positional data in COSMIC (v66), such that the baseline prior probability was set to 0.99 and this probability was decreased for genes that showed evidence of either tumor suppressor activity (mutations clustering at the same amino acid positions) or oncogenic activity (a higher proportion of truncating mutations than the overall proportion).

Format

A vector of length 18,926 where each element corresponds to a gene and the name of the element is the Ensembl id. The numeric value lies between 0.5 and 0.99 and represents the prior probability that each gene is a passenger.

shuffle.muts *Simulate new dataset*

Description

Shuffle the observed mutations restricted to same mutation type, replication region and expression level

Usage

```
shuffle.muts(N, sil.mutab, nonsil.mutab, exome.constants,
  gene, exome.nonsil, exome.SIFT, SEED = NULL, dir = ".",
  allF = TRUE)
```

Arguments

`N`	integer number of simulated datasets to create
`sil.mutab`	a matrix containing one row per silent mutation and 8 columns (Ensembl gene name, chromosome, position, variant type (SNP, In_frame, Frame_shift), reference allele, tumor allele 1, tumor allele 2, and sample id.
`nonsil.mutab`	a matrix containing one row per nonsilent mutation and 8 columns (Ensembl gene name, chromosome, position, variant type (SNP, In_frame, Frame_shift), reference allele, tumor allele 1, tumor allele 2, and sample id.
`exome.nonsil`	list object with one item per chromosome where each item contains matrix with one row per coding base pair and 7 columns: position, nucleotide, CpG context, nonsilent indicator (1=nonsilent, 0=silent) for mutation to "A", nonsilent indicator for mutation to "C", nonsilent indicator for mutation to "G", and nonsilent indicator for mutation to "T".

`SEED`	random seed for reproducible results
`dir`	directory path where to save the simulated datasets for use in `locfdr`
`allF`	logical value indicating whether or not the sample consists of all females whether or not to shuffle the observed mutations over the Y chromosome
`exome.constants`	a list with one entry for each chromosome, where each entry is a matrix containing a row for each coding base pair and the following columns: 1. base pair position, 2. nucleotide number (see `convert.seq.to.num`), 3. number of possible nonsilent transitions, 4. number of possible nonsilent transversions, 5. number of possible silent transitions, 6. number of possible silent transversions, and 7. whether the position is in a CpG dinucleotide.
`gene`	a list with one entry for each gene, each entry is another list of 5 elements: Ensembl name, chromosome, base pairs, replication timing region (1=Early, 2=Middle, 3=Late), and expression level (1=Low, 2=Medium, 3=High).
`exome.SIFT`	list object with one item per chromosome where each item contains matrix with one row per coding base pair and 7 columns: position, nucleotide, CpG context, FI score for mutation to "A", FI score for mutation to "C", FI score for mutation to "G", and FI score for mutation to "T".

Value

NULL

Note

This internal function is not intended to be called by the user.

`sift.col` *Retrieve the column from SIFT oject correspoinding to mutated base number*

Description

function to match the mutated base to the appropriate column of `exome.SIFT` (1:4, 2:7, 3:6, 4:5)

Usage

```
sift.col(mutbase)
```

Arguments

`mutbase`	an integer 1, 2, 3 or 4

Details

see `convert.seq.to.num` for correspondence of base letters to numbers and `mut.type.converter` for details of the structure of `exome.SIFT`.

Value

an integer 4, 5, 6 or 7

Note

This internal function is not intended to be called by the user.

`sift.colA` *Retrieve the column from SIFT oject correspoinding to mutated base letter*

Description

function to match the mutated base to the appropriate column of `exome.SIFT` ("A":4, "C":7, "G":6, "T":5)

Usage

```
sift.colA(mutbase)
```

Arguments

`mutbase`	a character "A", "C", "G", or "T"

Details

Similar to `sift.col` except uses the character of bases instead of numerical indicator. See `mut.type.converter` for details of the structure of `exome.SIFT`.

Value

an integer 4, 5, 6 or 7

Note

This internal function is not intended to be called by the user.

Index