Package 'mSigAct'

November 10, 2020

Title mutational Signature Activity analysis ('mSigAct')

Version 2.0.5

Author Steve Rozen, Alvin Wei Tian Ng, Arnoud Boot

Maintainer Steve Rozen <steverozen@gmail.com>

Description Analyze the ``activities'' of mutational signatures in one or more mutational spectra. 'mSigAct' stands for mutational Signature Activity. mSigAct can estimate (conservatively) whether there is evidence that a particular set of mutational signatures is present in a spectrum. It can also determine a *minimal* subset of signatures needed to plausibly reconstruct an observed spectrum. This sparse assign signatures functionality is *deliberately biased* toward using as few signatures as possible.This package does not provide all-purpose estimation for signature attribution.

License GPL-3

URL https://github.com/steverozen/mSigAct

BugReports https://github.com/steverozen/mSigAct/issues

Encoding UTF-8

LazyData true

Language en-US

Remotes github::steverozen/PCAWG7,
github::steverozen/ICAMS@master,
github::steverozen/mSigBG

Depends R (>= 3.5),

RoxygenNote 7.1.1

VignetteBuilder knitr

biocViews

Imports ICAMS,
nloptr,
stats,
lsa,
sets

Suggests BSgenome.Hsapiens.1000genomes.hs37d5,
devtools,
htmlwidgets,
knitr,
PCAWG7,

profvis,
rmarkdown,
testthat (>= 2.1.0),
usethis,
utils

R topics documented:

`AnySigSubsetPresent`

For each combination of several signatures, determine if the combination is plausibly needed to reconstruct a spectrum.

Description

Please see **Details**.

Usage

```
AnySigSubsetPresent(
  spect,
  all.sigs,
  Ha.sigs.indices,
  eval_f = mSigAct::ObjFnBinomMaxLHNoRoundOK,
  m.opts,
  max.mc.cores = NULL
)
```

Arguments

`spect`	The spectrum to be reconstructed, as single column matrix or `ICAMS` catalog.
`all.sigs`	The matrix or catalog of all signatures of possible interest, which consist of the signatures for H_0 and for the alternative hypotheses.
`Ha.sigs.indices`	An integer vector of the indices of the signatures that are in the various H_a's.
`eval_f`	Usually one of `ObjFnBinomMaxLHNoRoundOK` or `ObjFnBinomMaxLHMustRound`. For background see `nloptr`.
`m.opts`	Controls the numerical search for maximum likelihood reconstructions of `spect` plus some additional flags; see `DefaultManyOpts`.

`max.mc.cores` The maximum number of cores to use. If `NULL` defaults to $2^{n_a} - 1$, where n_a is the length of `Ha.sigs.indices` – except on MS Windows machines, where it defaults to 1.

Details

Let H_0 be the likelihood that the signatures specified by `all.sigs[,-Ha.sigs.indicies,drop = FALSE]` generated the observed spectrum, `spect`. For each non-empty subset, S, of `Ha.sigs.indices` let H_a be the likelihood that all the signatures in H_0 plus the signatures specified by S generated `spect`. Return a list with the results of likelihood ratio tests of all H_a's against H_0.

Value

A list with two elements:

`H0.info` contains the sub-elements

- `loglh` The log likelihood associated with H_0.
- `exposure` The signature attributions (exposures) corresponding to the H_0 log likelihood.
- `everything.else` A sub-list with information on the output of the numerical optimization that provided `loglh`.

`all.Ha.info` A list with one sub-element for each non-empty subset of `Ha.sigs.indices`. Each sub-element is a list with elements that include

- `sigs.added` The identifiers of the (additional) signatures tested.
- `p` The p value for the likelihood-ratio test.This this p value can be `NaN` when the likelihoods of (H_0 and H_a) are both `-Inf`. This can occur if there are are mutation classes in the spectra that are > 0 but that have 0 probability in all the available input signatures. This is unlikely to occur, since most spectra have non-0 (albeit very small) probabilities for most mutation classes. This is not an error is using `eval_f = ObjFnBinomMaxLHNoRoundOK`. However, if `p == NaN` when using `eval_f = ObjFnBinomMaxLHMustRound`, switch to `ObjFnBinomMaxLHNoRoundOK`.
- `df` The degrees of freedom of the likelihood-ratio test (equal to the number of signatures in `sigs.added`).

WARNING: tests all non-empty subsets of `Ha.sigs.indices`, so will get very slow for large numbers of `Ha.sigs.indices`.

`cossim` *Cosine similarity with useful argument types..*

Description

Calls `cosine`.

Usage

```
cossim(v1, v2)
```

Arguments

`v1`	A vector or single-column matrix
`v2`	A vector or single-column matrix

DefaultManyOpts *Set default options for many functions, especially* nloptr.

Description

Set default options for many functions, especially nloptr.

Usage

```
DefaultManyOpts()
```

Value

A list with the following elements

global.opts Options for nloptr, q.v., for the global optimization phase.

local.opts Options for nloptr, q.v., for the local optimization phase.

nbinom.size The dispersion parameter for the negative binomial distribution; smaller is more dispersed. See NegBinomial.

trace If > 0 print progress messages.

LLHSpectrumNegBinom *Likelihood that 1 observed spectrum was generated from a vector of expected mutation counts.*

Description

Likelihood that 1 observed spectrum was generated from a vector of expected mutation counts.

Usage

```
LLHSpectrumNegBinom(spectrum, expected.counts, nbinom.size, verbose = FALSE)
```

Arguments

spectrum	An observed spectrum (a numeric vector)
expected.counts	A vector of (integer) expected mutation counts, one expected count for each mutation type. We want to know the likelihood that this model generated the observed spectrum, assuming each mutational types generates counts according to a negative binomial distribution with the given expected.counts (argument mu to NegBinomial) and dispersion parameter nbinom.size.
nbinom.size	The dispersion parameter for the negative binomial distribution; smaller is more dispersed. See NegBinomial.
verbose	If TRUE print messages under some circumstances.

Value

`log(likelihood(spectrum | expected.counts))`, or, in more detail, the sum of the negative binomial likelihoods that each each element of the spectrum (i.e., the count for each mutation type e.g. ACT > AAT) was generated from the expected count for that mutation type.

`ObjFnBinomMaxLHMustRound` *A deprecated negative binomial maximum likelihood objective function.*

Description

Use `ObjFnBinomMaxLHNoRoundOK` instead.

Usage

```
ObjFnBinomMaxLHMustRound(exp, spectrum, sigs, nbinom.size)
```

Arguments

`exp`	The matrix of exposures ("activities").
`spectrum`	The spectrum to assess.
`sigs`	The matrix of signatures.
`nbinom.size`	The dispersion parameter for the negative binomial distribution; smaller is more dispersed. See `NegBinomial`.

Details

This function will lead to errors in some situations when the rounded reconstructed signature contains 0s for mutations classes for which the target spectrum is > 0.

`ObjFnBinomMaxLHNoRoundOK` *The preferred negative binomial maximum likelihood objective function.*

Description

Can be used as the objective function for `SparseAssignActivity`, `SparseAssignActivity1`, and `SignaturePresenceTest1`. (Internally used by by `nloptr`.)

Usage

```
ObjFnBinomMaxLHNoRoundOK(exp, spectrum, sigs, nbinom.size)
```

Arguments

exp	The matrix of exposures ("activities").
spectrum	The spectrum to assess.
sigs	The matrix of signatures.
nbinom.size	The dispersion parameter for the negative binomial distribution; smaller is more dispersed. See NegBinomial.

Value

-1 * log(likelihood(spectrum | reconstruction))

nloptr minimizes the objective function, so the lower the objective function, the better.

OptimizeExposure *Optimize the reconstruction of a spectrum from a set of signatures.*

Description

Optimize the reconstruction of a spectrum from a set of signatures.

Usage

```
OptimizeExposure(spectrum, sigs, m.opts, eval_f, ...)
```

Arguments

spectrum	The spectrum to be reconstructed.
sigs	The available signatures.
m.opts	Options that govern the numerical optimization. For documentation see DefaultManyOpts.
eval_f	The objective function for nloptr. We have only tested ObjFnBinomMaxLHNoRoundOK and ObjFnBinomMaxLHMustRound.
...	Additional arguments for eval_f.

Returns a list with elements

loglh The log likelihood of the best solution (set of exposures) found. For a more general objective function this might be NULL.

exposure The vector of exposures that generate loglh, i.e. the number of mutations ascribed to each signature.

obj.fn.value The objective function value associated with exposure.

everything.else Everything returned by the function Nloptr1Tumor.

SignaturePresenceTest

Test whether a given signature is plausibly present in a catalog

Description

Test whether a given signature is plausibly present in a catalog

Usage

```
SignaturePresenceTest(
  spectra,
  sigs,
  target.sig.index,
  m.opts = NULL,
  eval_f,
  mc.cores = 10
)
```

Arguments

spectra	The catalog (matrix) to analyze. This could be an ICAMS catalog or a numerical matrix.
sigs	A catalog of signatures from which to choose. This could be and ICAMS catalog or a numerical matrix.
target.sig.index	The index of the signature the presence of which we want to test.
m.opts	If NULL use the return from calling DefaultManyOpts. For documentation see DefaultManyOpts.
eval_f	See nloptr.
mc.cores	Number of cores to use. Always silently changed to 1 on Microsoft Windows.

SparseAssignActivity

Find known signatures that can most sparsely reconstruct each spectrum in a catalog.

Description

Find known signatures that can most sparsely reconstruct each spectrum in a catalog.

Usage

```
SparseAssignActivity(
  spectra,
  sigs,
  max.level = 5,
  p.thresh = 0.05,
  eval_f = ObjFnBinomMaxLHNoRoundOK,
  m.opts = NULL,
  num.parallel.samples = 5,
  mc.cores.per.sample = NULL
)
```

Arguments

`spectra`	The spectra (multiple spectra) to be reconstructed.
`sigs`	The known signatures to use in reconstruction.
`max.level`	The largest number of signatures to consider discarding in the reconstruction.
`p.thresh`	The maximum p value based on which it is decided to retain a signature in a reconstruction.
`eval_f`	The objective function for `nloptr`.
`m.opts`	For documentation see `DefaultManyOpts`.
`num.parallel.samples`	The (maximum) number of samples to run in parallel; each sample in turn can require multiple cores, as governed by `mc.cores.per.sample`.
`mc.cores.per.sample`	The maximum number of cores to use for each sample. If `NULL` defaults to `min(20,2^max.level)` – except on MS Windows machines, where it defaults to 1.

Value

A list with the inferred exposure matrix as element `exposure`.

Index