# Package 'mSigAct'

May 15, 2020

**Title** mutational Signature Activity analysis ('mSigAct')

**Version** 2.0.0.9018

**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. This package can also separate out the activity of a background mutational signature from spectra generated in experiments (usually cell culture experiments).

**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

**Depends** R (>= 3.5),

**RoxygenNote** 7.1.0

**VignetteBuilder** knitr

**biocViews**

**Imports** ICAMS,
nloptr,
stats,
lsa,
sets

**Suggests** devtools,
testthat (>= 2.1.0),
usethis,
utils,

knitr,
BSgenome.Hsapiens.1000genomes.hs37d5,
rmarkdown,
PCAWG7

# 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`.

---

# `background.info` *Specifications of background signatures*

---

## Description

Specifications of background signatures

Background information for MCF-10A cells.

## Usage

```
HepG2.background.info

MCF10A.background.info
```

## Format

A list with the elements

`background.sig` The background signature profile.

**codesig.nbinom.size** The dispersion parameter for the negative binomial distribution for sampling error around the components of `background.sig`. Smaller is more dispersed. See `NegBinomial`.

**codecount.nbinom.mu** The `mu` argument for `NegBinomial` for the distribution of total counts due to `background.sig` across replicate exposed clones.

`count.nbinom.size` The dispersion parameter for the negative binomial distribution of sampling error for total counts due to `background.sig` across replicate exposed clones. Smaller is more dispersed. See `NegBinomial`

An object of class `list` of length 5.

## Source

`HepG2.background.info` was estimated from `HepG2.background.spectra`.

`MCF10A.background.info` was estimated from `MCF10A.background.spectra`

## Examples

```
HepG2.background.info$count.nbinom.mu
HepG2.background.info$count.nbinom.size
HepG2.background.info$sig.nbinom.size
HepG2.background.info$background.sig[1:3, ]
```

---

# `background.spectra` *Background spectra for HepG2 and MCF-10A*

---

## Description

Background spectra for HepG2 and MCF-10A

Background spectra for MCF-10A cells

## Usage

```
HepG2.background.spectra

MCF10A.background.spectra
```

### Format

An ICAMS counts catalog.

An object of class SBS96Catalog (inherits from matrix) with 96 rows and 3 columns.

---

## 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.

`EstimateSignatureFromSpectraLH` *Build a signature for background extraction from a matrix of spectra.*

## Description

This function not only produces a signature, but also an estimate of the number of mutations usually generated by the signature and an indication of variability around that estimate.

## Usage

```
EstimateSignatureFromSpectraLH(
  spectra,
  algorithm = "NLOPT_LN_COBYLA",
  maxeval = 1000,
  print_level = 0,
  xtol_rel = 0.001,
  xtol_abs = 1e-04
)
```

## Arguments

| | |
|---|---|
| `spectra` | An `ICAMS` `catalog` with `catalog.type = "counts"`. |
| `algorithm` | See `nloptr`. |
| `maxeval` | See `nloptr`. |
| `print_level` | See `nloptr`. |
| `xtol_rel` | See `nloptr`. |
| `xtol_abs` | See `nloptr`. |

## Details

Only works on SBS 96 signatures.

## Value

A list with the elements

1. `signature` An `ICAMS` `catalog` with `catalog.type == "counts.signature"`.
2. `log10.counts` Mean log base 10 of the total counts in `spectra`
3. `sd.log10.counts.per.base` Standard deviation of `log10.counts.per.base`.

## MeanOfSpectraAsSig *Return the mean of multiple spectra as a signature.*

### Description

Return the mean of multiple spectra as a signature.

### Usage

```
MeanOfSpectraAsSig(spectra)
```

### Arguments

| | |
|---|---|
| spectra | Convert each spectrum to a signature and then compute the mean of all signatures. |

## nitrosamine.examples *Example nitrosamine data for background subtraction*

### Description

Example nitrosamine data for background subtraction

### Usage

```
nitrosamine.examples
```

### Format

A list of spectra catalogs for nitrosamines. The names of the catalogs are self-explanatory. Each catalog has 2 spectra from each of four different nitrosamines, "NDEA", "NDMA", "NPIP", "NPYR". The samples names also should be self-explanatory.

## 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 reconsructed. |
| `sigs` | The available signatures. |
| `m.opts` | Options that govern the numerical optimizaiton. 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`.

---

`SeparateSignatureFromBackground`

*Estimate a signature from experimentally exposed spectra minus a background signature.*

---

## Description

We index mutation channels (e.g. `ACA > AAA, ACC > AAC, ...`) by $j$, $j \in 1...96$.

We index input mutational spectra by $i$.

Let

$g = g_1, g_2, \ldots, g_{96}$, with $\Sigma g_j = 1$, be the previously determined, input background signature profile,

$s^i, i \in 1, 2, \ldots$ be the input spectra, from exposed samples, usually only 2 or 3,

$b^i, i \in 1, 2, \ldots$ be the (to-be-estimated) numbers of mutations due to the background signature in each $s^i$, and

$t = t_1, t_2, \ldots, t_{96}$, with $\Sigma t_j = 1$, be the (to-be-estimated) target signature due to an exposure.

We want to maximize $\Pi^i P(s^i|b^i, t)P(b^i)$ over $b^1, b^2, \ldots$ and $t$. (Note that the code actually minimizes the additive inverse of this.)

$P(b^i)$ is estimated from the distribution of previously observed numbers of mutations in untreated samples, with the additional constraint that $b^i \leq |s^i|$), where $|s^i|$ is defined as the total number of mutations in spectrum $s^i$, i.e. $|s^i| = \Sigma_j s^i_j$, $j \in 1...96$.

$P(s^i|b^i, t)$ is estimated as follows:

The expected number of mutations in each mutation category, $j$, is estimated as

$e^i_j = g_j b^i + t_j(|s^i| - b^i)$.

Then $P(s^i|e^i)$ is estimated as $\Pi_j P(s^i_j|e^i_j)$.

$P(s^i_j|e^i_j)$ is estimated from a negative binomial distribution centered on each $e^i_j$; these distributions all have a dispersion parameter of 10 (hard coded in `ObjFn1`), a value chosen based on tests with synthetic data.

## Usage

```
SeparateSignatureFromBackground(
  spectra,
  bg.sig.info,
  m.opts = NULL,
  start.b.fraction = 0.1
)
```

## Arguments

| | |
|---|---|
| `spectra` | The spectra from which to subtract the background, as a matrix or `ICAMS` catalog. |
| `bg.sig.info` | Information about the background signature. See for example `HepG2.background.info`. |
| `m.opts` | Options to pass to `nloptr`. |
| `start.b.fraction` | The estimated fraction of the mutations in `spectra` due to the background signature. |

## Details

See `ObjFn1`.

---

`SeparateSignatureFromBackgroundOptions` *Return a default value to pass as the* `m.opts` *argument to* `SeparateSignatureFromBackground`.

---

## Description

Return a default value to pass as the `m.opts` argument to `SeparateSignatureFromBackground`.

## Usage

```
SeparateSignatureFromBackgroundOptions()
```

## 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 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