Package 'SynSigGen'

December 3, 2022

Type Package

Title Create Catalogs of Synthetic Mutational Spectra

Version 1.2.1

Author Steven G. Rozen, Yang Wu, Nanhai Jiang

Maintainer Steven G. Rozen <steverozen@gmail.com>

Description Create catalogs of synthetic mutational spectra for assessing the performance of mutational signature analysis programs. 'SynSigGen' stands for Synthetic Signature Generation.

License GPL-3

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

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

Language en-US

Encoding UTF-8

LazyData true

biocViews

Imports data.table,
fitdistrplus,
ICAMS,
ICAMSxtra,
mSigTools,
PCAWG7

Remotes github::steverozen/ICAMS@*release,
github::steverozen/ICAMSxtra@*release,
github::steverozen/PCAWG7@*release

Depends R (>= 4.0)

RoxygenNote 7.2.2

Suggests cosmicsig,
testthat,
knitr,
rmarkdown,
DelayedArray,
lsa

R topics documented:

`AddNoise` *Exposures and spectra with Poisson or negative binomial noise in exposures.*

Description

Exposures and spectra with Poisson or negative binomial noise in exposures.

Usage

```
AddNoise(input.exposure, signatures, n.binom.size = 100)
```

Arguments

`input.exposure`	The exposures to which to add noise; a numeric matrix or data frame in which the rows are signatures and the columns are samples. Each cell indicates the number of mutations due to a particular signature in a particular sample.
`signatures`	The signatures in the exposure; the column names of `signatures` have to include all row names in `input.exposure`; can be an `ICAMS` catalog or a numerical matrix or data frame.
`n.binom.size`	If non `NULL`, use negative binomial noise with this size parameter; see `NegBinomial`. If `NULL`, use Poisson noise.

Value

A list with the elements

expsoures The numbers of mutations due to each signature after adding noise

spectra The spectra based on the noisy signature exposures.

`BladderSkin1000` *Generate synthetic data sets modeled on bladder TCC and skin melanoma.*

Description

Creates spectra dataset consists of 500 synthetic bladder transitional cell carcinoma with high prevalence and mutation load from SBS2, and 500 synthetic skin melanoma with high prevalence and mutation load from SBS7a and SBS7b. This dataset challenges the computational approaches as SBS2 has a similar pattern to the mixture of SBS7a and SBS7b, thus the existence of these signatures may interfere computational approaches from accurately extracting these signatures.

Usage

```
BladderSkin1000(
  seed = 191906,
 regress.dir = "data-raw/long.test.regression.data/syn.2.7a.7b.bladder.and.melanoma/",
  top.level.dir = "../2.7a.7b.bladder.and.melanoma.191906",
  unlink = FALSE
)
```

Arguments

seed	A random seed to use.
regress.dir	If not NULL, compare the result to the contents of this directory with a diff.
top.level.dir	The directory in which to put the output; will be created if necessary.
unlink	If TRUE and !is.null(regress.dir), then unlink the result directory if there are no differences.

Details

This function replaces the first part of data-raw/Create.2.7a.7b.Rmd in GitHub repository steverozen/SynSig. With default arguments, this function generates the same results as the first part of data-raw/Create.2.7a.7b.Rmd.

#' The second half of data-raw/Create.2.7a.7b.Rmd is replaced by Create.2.7a.7b.Abstract.

Data set generated by this function can be found at Synapse with Synapse ID: syn18500217.

Create.3.5.40.Abstract

Create synthetic spectra based on SBS3 SBS5 SBS40

Description

This function generates synthetic spectra with mutation loads of SBS3 (signature prevalent in ovarian adenocarcinoma), SBS5 and SBS40 (signatures prevalent in renal cell carcinoma). This dataset challenges the computational approaches as these three signatures are "flat" signatures hard to be extracted accurately.

Usage

```
Create.3.5.40.Abstract(
  seed = 44,
  overwrite = TRUE,
  regress.dir = "data-raw/long.test.regression.data/syn.3.5.40.abst/",
  num.syn.tumors = 1000,
  top.level.dir = paste0("../syn.3.5.40.abst.", seed),
  unlink = FALSE
)
```

Arguments

seed	A random seed to use.
overwrite	If TRUE, overwrite existing directories / files.
regress.dir	If not NULL, compare the result to the contents of this directory with a diff.
num.syn.tumors	The number of tumors to create **for each cancer type** in cancer.types.
top.level.dir	The directory in which to put the output; will be created if necessary.
unlink	If TRUE and !is.null(regress.dir), then unlink the result directory if there are no differences.

Details

This function supersedes the second part of `data-raw/Create.3.5.40.Rmd` in GitHub repository `steverozen/SynSig`. With default arguments, this function generates the same results as the second half of `data-raw/Create.3.5.40.Rmd`.

Data set generated by this function can be found at Synapse with Synapse ID: syn18500215.

`CreateAndWriteCatalog` *Create and write a mutational spectra catalog*

Description

Create and write a mutational spectra catalog

Usage

```
CreateAndWriteCatalog(
  sigs,
  exp,
  dir = NULL,
  write.cat.fn = ICAMS::WriteCatalog,
  extra.file.suffix = "",
  overwrite = FALSE,
  my.dir = NULL
)
```

Arguments

`sigs`	Signatures to use.
`exp`	(Synthetic) exposures.
`dir`	Deprecated, maintained only to avoid breaking old code. A subdirectory based on the deprecated global variable `OutDir`.
`write.cat.fn`	Function to write catalogs **or** spectra to files.
`extra.file.suffix`	Extra string to put before ".csv".
`overwrite`	If TRUE, overwrite existing directory; useful for debugging / testing.
`my.dir`	The directory in which to write the catalog and several additional files.

Details

Create a file with the catalog `syn.data.csv` and writes `sigs` to `input.sigs.csv`.

Value

Invisibly, the generated catalog.

`CreateFromReal` *Create a specific synthetic data set based on real exposures in one or more cancer types.*

Description

Create a full SignatureAnalyzer / SigProfiler test data set for a set of various tumor types.

Usage

```
CreateFromReal(
  seed,
  top.level.dir = NULL,
  enclosing.dir = NULL,
  num.syn.tumors,
  cancer.types,
  data.suite.name = NULL,
  sa.exp = SynSigGen::sa.all.real.exposures,
  sp.exp = SynSigGen::sp.all.real.exposures,
  overwrite = TRUE,
  regress.dir = NULL,
  unlink = FALSE,
  verbose = FALSE,
  bladder.regress.hack = FALSE
)
```

Arguments

`seed`	A random seed to use.
`top.level.dir`	The directory in which to put the output; will be created if necessary.
`enclosing.dir`	Deprecated; create the output in a subdirectory of this directory.
`num.syn.tumors`	The number of tumors to create **for each cancer type** in `cancer.types`.
`cancer.types`	Search `sa.exp` and `sp.exp` for exposures from tumors matching these strings. Each string should identify one tumor type, for some definition of tumor type. Probably the tumors in each type should be non-overlapping, but the code does not enforce this and does not care.
`data.suite.name`	Deprecated; the directory created will be `file.path(enclosing.dir,paste0(data.suite.name,"`
`sa.exp`	A matrix of exposures; this function will use the columns with column names beginning `paste0(cancer.type,"::")`.
`sp.exp`	A matrix of exposures; this function will use the columns with column names beginning `paste0(cancer.type,"::")`.
`overwrite`	If TRUE, overwrite existing directories and files.
`regress.dir`	If not `NULL`, compare the result to the contents of this directory with a `diff`.
`unlink`	If `TRUE` and `!is.null(regress.dir)`, then unlink the result directory if there are no differences.
`verbose`	If `TRUE` print various informative messages.

`bladder.regress.hack`
: Set this to `TRUE` to handle mixed "all" and "no hyper" signature sets for the regression test for `BladderSkin1000`.

`CreateMixedTumorTypeSyntheticData` *Create a test data set based on >= 1 tumor types.*

Description

Create a test data set based on >= 1 tumor types.

Usage

```
CreateMixedTumorTypeSyntheticData(
  top.level.dir,
  cancer.type.strings,
  num.syn.tumors,
  overwrite = FALSE,
  sa.exp = sa.all.real.exposures,
  sp.exp = sp.all.real.exposures,
  verbose = FALSE,
  bladder.regress.hack = FALSE
)
```

Arguments

`top.level.dir`	Path to top level of directory structure to be created.
`cancer.type.strings`	Search the PCAWG data for tumors matching these strings. Each string should identify one tumor type, for some definition of tumor type. Probably the tumors in each type should be non-overlapping, but the code does not enforce this and does not care.
`num.syn.tumors`	Number of synthetic tumors to create for each cancer type.
`overwrite`	If TRUE, overwrite existing directories / files.
`sa.exp`	SignatureAnalyzer exposures from which to select cancer types specified by `cancer.type.strings`. In the column names of `sa.exp` the cancer type string should be separated from the sample identifier by two colons (::).
`sp.exp`	SigProfiler exposures from which to select cancer types specified by `cancer.type.strings`. In the column names of `sp.exp` the cancer type string should be separated from the sample identifier by two colons (::).
`verbose`	If > 0, cat various messages.
`bladder.regress.hack`	For use by `BladderSkin1000`. Forces use of non-hyper-mutated exposures for bladder-TCC even if `sa.exp` and `sp.exp` include hyper-mutated exposures.

`CreateRandomSyn` *This is the top-level function to create a set of spectra from random signatures.*

Description

This is the top-level function to create a set of spectra from random signatures.

Usage

```
CreateRandomSyn(
  top.level.dir,
  seed = 1443196,
  regress.dir = "data-raw/long.test.regression.data/syn.30.random.sigs/",
  num.syn.tumors = 1000,
  overwrite = FALSE,
  unlink = FALSE,
  verbose = FALSE
)
```

Arguments

`top.level.dir`	Directory in which to put all results. It will be created if necessary.
`seed`	Use default for regression testing.
`regress.dir`	If not `NULL` compare the known results in this directory with the created results in `top.level.dir`.
`num.syn.tumors`	Total number of synthetic tumors to create. Use the default for regression testing.
`overwrite`	If `TRUE` overwrite existing files and directories.
`unlink`	If `TRUE` unlink the created directory after the regression test.
`verbose`	If `TRUE` print a few informative messages.

`CreateSBS1SBS5CorrelatedSyntheticData` *Function to generate 20 SBS1-SBS5-correlated Synthetic datasets used in testing.*

Description

This function is a wrapper around `CreateSBS1SBS5CorrelatedSyntheticDataOneDataset`. It will use the default parameters to repeat the results.

Usage

```
CreateSBS1SBS5CorrelatedSyntheticData(
  top.level.dir = "./",
  regress.dir = NULL,
  overwrite = FALSE,
  add.info = TRUE,
  unlink = FALSE
)
```

Arguments

top.level.dir	Top-level-folder to place 20 spectra datasets generated by this function. Default: ./ (Current working directory)
regress.dir	If not NULL, compare the result to the contents of this directory with a diff.
overwrite	Whether to overwrite (Default: FALSE)
add.info	Whether to generate additional information. You should set it to FALSE when you want to make a diff (i.e. regressdir is not NULL). This is because Additional information may differ on different OS or R sessions, thus may prevent the dataset from passing the NewDiff4SynDatasets check. (Default: TRUE)
unlink	Whether to delete temporary dataset folder top.level.dir. (Set to TRUE for testing)

Details

This function will generate 20 datasets, each with files listed below:

ground.truth.syn.catalog.csv: Generated tumor spectra in ICAMS SBS96 CSV format.

ground.truth.syn.exposures.csv: Mutation burdens of SBS1 and SBS5 in generated tumor spectra in ICAMS CSV format.

ground.truth.syn.sigs.csv: Ground-truth SBS1 and SBS5 signatures in ICAMS SBS96 CSV format.

parameters.txt: Parameters used to generate the exposures and tumor spectra.

scatterplot.pdf: scatterplot illustrating correlation of exposures of two signatures in generated spectra

seedInUse.txt, RNGInUse.txt: seed and Random Number Generator used in generation. (For better reproducibility)

sessionInfo.txt: information related to R versions, platforms, loaded or imported packages, etc. (For better reproducibility)

CreateSBS1SBS5CorrelatedSyntheticDataOneDataset

Wrapper function for generating SBS1-SBS5-correlated Synthetic data

---

Description

This function will use SigProfiler-SBS96 mutational signatures to generate imaginary tumor spectra with mutation burdens only from SBS1 and SBS5, and mutation burdens of both signatures are highly correlated.

Usage

```
CreateSBS1SBS5CorrelatedSyntheticDataOneDataset(
  dir.name = "./S.0.5.Rsq.0.3",
  dataset.name = NULL,
  overwrite = FALSE,
  seed = 1,
  parameter.df = SynSigGen::SBS1SBS5parameter["S.0.5.Rsq.0.3", ],
  add.info = TRUE,
  verbose = FALSE
)
```

Arguments

`dir.name`	Folder to place the generated tumor spectra and other output files. Default: ./S.0.5.Rsq.0.3
`dataset.name`	The dataset.name encodes the parameters for the synthetic data, but this is just a convention. If NULL, it will be changed to the last part of the `dir.name` (Default: NULL)
`overwrite`	Whether to overwrite (Default: FALSE)
`seed`	The seed number used to initialize pseudo-random number generator (RNG). This makes the generation of the correlated datasets repeatable. (Default: 1)
`parameter.df`	a named 1*14 data.frame containing the following items:

1. `main.signature` The name of the main signature whose exposure can vary freely. (Default: SBS5)
2. `correlated.signature` The name of the correlated signature whose exposure is influenced by and co-varies with the exposure of main.signature. In this study, it defaults as "SBS1".
3. `name.prefix` Default: `TwoCorreSigsGen`
4. `sample.number` The number of synthetic tumors you want to generate. Default: 500
5. `main.mean.log` The mean of log(count(SBS5),base = 10) Default: 2.5
6. `main.stdev.log` The standard deviation of log(count(SBS5),base = 10) Default: 0.3
7. `correlated.stdev.log` The ADDED standard deviation of log(count(SBS1),base = 10). This parameter is ADDED stdev because based on the mechanism to generate the count, log10(count(SBS1)) inherently has a stdev = slope * main.stdev.log Default: 0.4
8. `slope.linear` The ratio for: (Correlated exposure) / (Main exposure) IN LINEAR SPACE! Default: 0.5
9. `main.signature.lower.thres` This program will force the exposure count of main.signature to be greater than this threshold. Default: 100
10. `correlated.signature.lower.thres` This program will force the exposure count of correlated.signature to be greater than this threshold. Default: 1
11. `pearson.r.2.lower.thres` Lower boundary of Pearson's R^2 (Default: 0.29)
12. `pearson.r.2.higher.thres` Upper boundary of Pearson's R^2 (Default: 0.31)

13. `min.main.to.correlated.ratio.linear` The lower ratio for count(SBS5) / count(SBS1) in LINEAR SPACE! (Default: 1/3)
14. `max.main.to.correlated.ratio.linear` The upper ratio for count(SBS5) / count(SBS1) in LINEAR SPACE! (Default: Inf)

`add.info` Whether to generate additional information.

`verbose` If `TRUE` cat progress messages. You should set it to `FALSE` when you want to make a `diff` using `CreateSBS1SBS5CorrelatedSyntheticData` (i.e. parameter `regressdir` is not `NULL`). This is because Additional information may differ on different OS or R sessions, thus may prevent the dataset from passing the `NewDiff4SynDatasets` check. (Default: TRUE)

Warning

Exposure generation function will repeat generating exposure counts using mean and stdev parameters, until the dataset has a Pearson's R^2 which falls between two boundaries of Pearson's R^2. Below are a group of parameters which have been tested successfully. If you intend to lower the Pearson's R^2, do remember to increase the main.stdev.log and correlated.stdev.log. Otherwise, the exposure generation will keep generating and discarding datasets!

Details

If you want to customize Pearson R^2 of the dataset, you need to change the standard deviations of two signatures. i.e., main.stdev.log and correlated.stdev.log.

This function will generate files listed below:

ground.truth.syn.catalog.csv: Generated tumor spectra in ICAMS SBS96 CSV format.

ground.truth.syn.exposures.csv: Mutation burdens of SBS1 and SBS5 in generated tumor spectra in ICAMS CSV format.

ground.truth.syn.sigs.csv: Ground-truth SBS1 and SBS5 signatures in ICAMS SBS96 CSV format.

parameters.txt: Parameters used to generate the exposures and tumor spectra.

scatterplot.pdf: scatterplot illustrating correlation of exposures of two signatures in generated spectra

seedInUse.txt, RNGInUse.txt: seed and Random Number Generator used in generation. (For better reproducibility)

sessionInfo.txt: information related to R versions, platforms, loaded or imported packages, etc. (For better reproducibility)

`CreateSynCatalogs` *Generate synthetic spectra catalogs given signature profiles and synthetic exposures.*

Description

Generate synthetic spectra catalogs given signature profiles and synthetic exposures.

Usage

```
CreateSynCatalogs(signatures, exposures, sample.id.suffix = NULL)
```

Arguments

signatures	The signature profiles.
exposures	The synthetic exposures.
sample.id.suffix	A string for adding a suffix to sample ID. For example, if sample.id.suffix is "abc", then SomeCancerType::s1.33 is changed to SomeCancerType::s1-abc.33. Actually, this just replaces the first "." in the sample id with "-" concatenated to sample.id.suffix. TODO(Steve): probably drop this

Value

A list of three elements that comprise the synthetic data:

1. ground.truth.catalog: Spectra catalog for the software input.
2. ground.truth.signatures: Signatures active in ground.truth.catalog.
3. ground.truth.exposures: Exposures of ground.truth.signatures in ground.truth.catalog.

Diff4SynDataSets *diff new directory / files against regression data for testing.*

Description

diff new directory / files against regression data for testing.

Usage

```
Diff4SynDataSets(dirname, unlink)
```

Arguments

dirname	the root name of the directories to diff.
unlink	if TRUE unlink tmpdirname, but do not unlink if there are diffs.

Value

The output of the diff command.

`GenerateListOfSigParams` *Generate a list of signature parameters for different cancer types from real exposure*

Description

Generate a list of signature parameters for different cancer types from real exposure

Usage

```
GenerateListOfSigParams(
  real.exposures,
  cancer.types,
  distribution = NULL,
  verbose = 0,
  sig.params = NULL
)
```

Arguments

`real.exposures`	A matrix of real exposures.
`cancer.types`	A vector of character strings denoting different cancer types. This function will search `real.exposures` for exposures from tumors matching these strings. See `PCAWG7::CancerTypes()` for example.
`distribution`	Probability distribution used to generate synthetic exposures due to active mutational signatures. Can be `neg.binom` which stands for negative binomial distribution. If `NULL` (Default), then this function uses log normal distribution with base 10.
`verbose`	If > 0 cat various messages.
`sig.params`	Empirical signature parameters generated using real exposures irrespective of their cancer types. If there is only one tumor having a signature in a cancer type in `real.exposures`, we cannot fit the `distribution` to only one data point. Instead, we will use the empirical parameter `size` from `sig.params`. Users can use `SynSigGen:::GetSynSigParamsFromExposuresOld` to generate their own signature parameters. If `NULL`(default), this function uses the PCAWG7 empirical signature parameters. See `signature.params` for more details.

Note

This function calls `GetSynSigParamsFromExposures`.

Examples

```
# Generate a list of signature parameters for Indel (ID) using negative binomial distribution
real.exposures.ID <- PCAWG7::exposure$PCAWG$ID
cancer.types <- PCAWG7::CancerTypes()[1:5]
sig.params <- SynSigGen::signature.params$ID
ID.sig.params <-
  GenerateListOfSigParams(real.exposures = real.exposures.ID,
                          cancer.types = cancer.types,
```

```
                          distribution = "neg.binom",
                          sig.params = sig.params
  )
```

`GenerateNoisyTumors` *Generate noisy tumors from available exposures*

Description

Generate noisy tumors from available exposures

Usage

```
GenerateNoisyTumors(
  seed,
  dir,
  input.exposure,
  signatures,
  n.binom.size = NULL,
  overwrite = TRUE
)
```

Arguments

`seed`	A random seed to use.
`dir`	The directory in which to put the output; will be created if necessary.
`input.exposure`	A matrix of exposures.
`signatures`	A matrix of signatures.
`n.binom.size`	If non `NULL`, use negative binomial noise with this size parameter; see `NegBinomial`. If `NULL`, then use Poisson distribution to do the resampling.
`overwrite`	If TRUE, overwrite existing directories and files.

Value

A list with the elements

exposures The numbers of mutations due to each signature after adding noise

spectra The spectra based on the noisy signature exposures.

Examples

```
# Generate synthetic tumors for Indel (ID) using negative binomial distribution
input.sigs.ID <- cosmicsig::COSMIC_v3.2$signature$GRCh37$ID
real.exposures.ID <- PCAWG7::exposure$PCAWG$ID
cancer.types <- PCAWG7::CancerTypes()[1:5]
ID.synthetic.tumors <-
  GenerateSyntheticTumors(seed = 191906,
                          dir = file.path(tempdir(), "ID.synthetic.tumors"),
                          cancer.types = cancer.types,
                          samples.per.cancer.type = 30,
                          input.sigs = input.sigs.ID,
```

```
                          real.exposures = real.exposures.ID,
                          distribution = "neg.binom",
                          sample.prefix.name = "SP.Syn."
  )

# Add noise to the exposures
ID.noisy.tumors <-
  GenerateNoisyTumors(seed = 892513,
                      dir = file.path(tempdir(), "ID.noisy.tumors"),
                      input.exposure = ID.synthetic.tumors$ground.truth.exposures,
                      signatures = ID.synthetic.tumors$ground.truth.signatures,
                      n.binom.size = 1)

# Plot the synthetic and noisy catalog and exposures
ICAMS::PlotCatalogToPdf(catalog = ID.synthetic.tumors$ground.truth.catalog,
                        file = file.path(tempdir(), "ID.synthetic.catalog.pdf"))
mSigTools::plot_exposure_to_pdf(exposure = ID.synthetic.tumors$ground.truth.exposures,
                                file = file.path(tempdir(), "ID.synthetic.exposures.pdf"),
                                  cex.xaxis = 0.7)
ICAMS::PlotCatalogToPdf(catalog = ID.noisy.tumors$spectra,
                        file = file.path(tempdir(), "ID.noisy.catalog.pdf"))
mSigTools::plot_exposure_to_pdf(exposure = ID.noisy.tumors$exposures,
                                  file = file.path(tempdir(), "ID.noisy.exposures.pdf"),
                                  cex.xaxis = 0.7)
```

`GenerateSynFromReal` *Generate synthetic exposures from real exposures.*

Description

Checkpoints the parameters and the synthetic exposures to files. It also checks that the parameters inferred from the synthetic data approximate those inferred from `real.exp`.

Usage

```
GenerateSynFromReal(
  real.exp,
  num.syn.tumors,
  file.prefix,
  sample.id.prefix,
  top.level.dir = NULL
)
```

Arguments

`real.exp`	The actual (real) exposures upon which to base the parameters and synthetic exposures.
`num.syn.tumors`	Generate this number of synthetic tumors.
`file.prefix`	Prepend this to output filenames to indicate the organization of the data.
`sample.id.prefix`	Prefix for sample identifiers for the synthetic samples.
`top.level.dir`	Directory in which to create several files. This directory must already exist.

Value

A list with elements:

1. `parms` The parameters inferred from `real.exp`.
2. `syn.exp` The synthetic exposures generated from `parms`.

`GenerateSyntheticExposures` *Create synthetic exposures based given parameters*

Description

Create synthetic exposures based given parameters

Usage

```
GenerateSyntheticExposures(
  sig.params,
  num.samples = 10,
  name = "synthetic",
  sig.matrix = NULL,
  distribution = NULL,
  round.exposure = TRUE
)
```

Arguments

`sig.params`	Parameters from `GetSynSigParamsFromExposures` or another source. Should be a matrix or data frame with one column for each signature and the following rows: • For log normal distribution, **prob** The proportion of tumors with the signature. **mean** The mean(log_10(number of mutations)). **stdev** The stdev(log_10(number of mutations)). • For negative binomial distribution, **prob** The proportion of tumors with the signature. **size** Dispersion parameter. **mu** Mean. The rownames need to be the column names of a signature catalog.
`num.samples`	Number of samples to generate
`name`	Prefix for sample identifiers in the simulated dataset
`sig.matrix`	Signature matrix to construct synthetic tumors
`distribution`	Probability distribution used to generate synthetic exposures due to active mutational signatures. Can be `neg.binom` which stands for negative binomial distribution. If `NULL` (Default), then this function uses log normal distribution with base 10.
`round.exposure`	Whether the exposures should be rounded to an integer. Set as `FALSE` only when reproducing legacy data sets.

Value

A matrix with the rows being each signature and the columns being generated samples. Each entry is the count of mutations due to one signature in one sample.

`GenerateSyntheticTumors` *Generate synthetic tumors based on real exposures in one or more cancer types*

Description

Generate synthetic tumors based on real exposures in one or more cancer types

Usage

```
GenerateSyntheticTumors(
  seed,
  dir,
  cancer.types,
  samples.per.cancer.type,
  input.sigs,
  real.exposures,
  distribution = NULL,
  sample.prefix.name = "SP.Syn.",
  tumor.marker.name = NULL,
  overwrite = TRUE,
  verbose = 0,
  sig.params = NULL
)
```

Arguments

`seed`	A random seed to use.
`dir`	The directory in which to put the output; will be created if necessary.
`cancer.types`	A vector of character strings denoting different cancer types. This function will search `real.exposures` for exposures from tumors matching these strings. See `PCAWG7::CancerTypes()` for example.
`samples.per.cancer.type`	Number of synthetic tumors to create for each cancer type. If it is **one** number, then generate the **same** number of synthetic tumors for each `cancer.types`. Or if it is a **vector** of numbers, then generate synthetic tumors for each `cancer.type` accordingly to the number specified in the vector. The length and order of `samples.per.cancer.type` should match that in `cancer.types`.
`input.sigs`	A matrix of signatures.
`real.exposures`	A matrix of real exposures.
`distribution`	Probability distribution used to generate synthetic exposures due to active mutational signatures. Can be `neg.binom` which stands for negative binomial distribution. If `NULL` (Default), then this function uses log normal distribution with base 10.

`sample.prefix.name`
: Prefix name to add to the synthetic tumors.

`tumor.marker.name`
: Tumor marker name to add to the synthetic tumors. E.g. "MSI-H", "POLE".

`overwrite`
: If TRUE, overwrite existing directories and files.

`verbose`
: If > 0 cat various messages.

`sig.params`
: Empirical signature parameters generated using real exposures irrespective of their cancer types. If there is only one tumor having a signature in a cancer type in `real.exposures`, we cannot fit the `distribution` to only one data point. Instead, we will use the empirical parameter `size` from `sig.params`. Users can use `SynSigGen:::GetSynSigParamsFromExposuresOld` to generate their own signature parameters. If `NULL`(default), this function uses the PCAWG7 empirical signature parameters. See `signature.params` for more details.

Value

A list of three elements that comprise the synthetic data:

1. `ground.truth.catalog`: Spectra catalog with rows denoting mutation types and columns denoting sample names.
2. `ground.truth.signatures`: Signatures active in `ground.truth.catalog`.
3. `ground.truth.exposures`: Exposures of `ground.truth.signatures` in `ground.truth.catalog`.

Examples

```
# Generate synthetic tumors for DBS78
input.sigs.DBS78 <- cosmicsig::COSMIC_v3.2$signature$GRCh37$DBS78
real.exposures.DBS78 <- PCAWG7::exposure$PCAWG$DBS78
cancer.types <- PCAWG7::CancerTypes()[1:5]
DBS78.synthetic.tumors <-
  GenerateSyntheticTumors(seed = 191906,
                          dir = file.path(tempdir(), "DBS78.synthetic.tumors"),
                          cancer.types = cancer.types,
                          samples.per.cancer.type = 30,
                          input.sigs = input.sigs.DBS78,
                          real.exposures = real.exposures.DBS78,
                          sample.prefix.name = "SP.Syn."
  )

# Generate synthetic tumors for Indel (ID) using negative binomial distribution
input.sigs.ID <- cosmicsig::COSMIC_v3.2$signature$GRCh37$ID
real.exposures.ID <- PCAWG7::exposure$PCAWG$ID
cancer.types <- PCAWG7::CancerTypes()[1:5]
ID.synthetic.tumors <-
  GenerateSyntheticTumors(seed = 191906,
                          dir = file.path(tempdir(), "ID.synthetic.tumors"),
                          cancer.types = cancer.types,
                          samples.per.cancer.type = 30,
                          input.sigs = input.sigs.ID,
                          real.exposures = real.exposures.ID,
                          distribution = "neg.binom",
                          sample.prefix.name = "SP.Syn."
  )
```

```
# Plot the synthetic catalog and exposures
ICAMS::PlotCatalogToPdf(catalog = DBS78.synthetic.tumors$ground.truth.catalog,
                        file = file.path(tempdir(), "DBS78.synthetic.catalog.pdf"))
mSigTools::plot_exposure_to_pdf(exposure = DBS78.synthetic.tumors$ground.truth.exposures,
                            file = file.path(tempdir(), "DBS78.synthetic.exposures.pdf"),
                                 cex.xaxis = 0.7)
```

`GenerateSyntheticTumorsFromSigParams` *Generate synthetic tumors based on signature parameters in one or more cancer types*

Description

Generate synthetic tumors based on signature parameters in one or more cancer types

Usage

```
GenerateSyntheticTumorsFromSigParams(
  seed,
  dir,
  cancer.types,
  samples.per.cancer.type,
  input.sigs,
  sig.params,
  distribution = NULL,
  sample.prefix.name = "SP.Syn.",
  tumor.marker.name = NULL,
  overwrite = TRUE,
  verbose = 0
)
```

Arguments

`seed`	A random seed to use.
`dir`	The directory in which to put the output; will be created if necessary.
`cancer.types`	A vector of character strings denoting different cancer types. See `PCAWG7::CancerTypes()` for example.
`samples.per.cancer.type`	Number of synthetic tumors to create for each cancer type. If it is **one** number, then generate the **same** number of synthetic tumors for each `cancer.types`. Or if it is a **vector** of numbers, then generate synthetic tumors for each `cancer.type` accordingly to the number specified in the vector. The length and order of `samples.per.cancer.type` should match that in `cancer.types`.
`input.sigs`	A matrix of signatures.
`sig.params`	A list of empirical signature parameters generated using real exposures for each `cancer.types`. This list should have names that match `cancer.types`. Users can use `GenerateListOfSigParams` to generate their own signature parameters.

`distribution` Probability distribution used to generate synthetic exposures due to active mutational signatures. Can be `neg.binom` which stands for negative binomial distribution. If `NULL` (Default), then this function uses log normal distribution with base 10.

`sample.prefix.name`
Prefix name to add to the synthetic tumors.

`tumor.marker.name`
Tumor marker name to add to the synthetic tumors. E.g. "MSI", "POLE".

`overwrite` If TRUE, overwrite existing directories and files.

`verbose` If > 0 cat various messages.

Value

A list of three elements that comprise the synthetic data:

1. `ground.truth.catalog`: Spectra catalog with rows denoting mutation types and columns denoting sample names.
2. `ground.truth.signatures`: Signatures active in `ground.truth.catalog`.
3. `ground.truth.exposures`: Exposures of `ground.truth.signatures` in `ground.truth.catalog`.

Note

See also `GenerateSyntheticTumors` which uses real exposure matrix instead of list of signature parameters to generate synthetic tumor.

Examples

```
# Generate synthetic tumors for DBS78 using log normal distribution
input.sigs.DBS78 <- cosmicsig::COSMIC_v3.2$signature$GRCh37$DBS78
real.exposures.DBS78 <- PCAWG7::exposure$PCAWG$DBS78
cancer.types <- PCAWG7::CancerTypes()[1:5]
sig.params <- SynSigGen::signature.params$DBS78
DBS78.sig.params <-
  GenerateListOfSigParams(real.exposures = real.exposures.DBS78,
                          cancer.types = cancer.types,
                          sig.params = sig.params)

DBS78.synthetic.tumors <-
  GenerateSyntheticTumorsFromSigParams(seed = 191906,
                                       dir = file.path(tempdir(), "DBS78.synthetic.tumors"),
                                           cancer.types = cancer.types,
                                           samples.per.cancer.type = 30,
                                           input.sigs = input.sigs.DBS78,
                                           sig.params = DBS78.sig.params,
                                           sample.prefix.name = "SP.Syn.")

# Generate synthetic tumors for Indel (ID) using negative binomial distribution
input.sigs.ID <- cosmicsig::COSMIC_v3.2$signature$GRCh37$ID
real.exposures.ID <- PCAWG7::exposure$PCAWG$ID
cancer.types <- PCAWG7::CancerTypes()[1:5]
sig.params <- SynSigGen::signature.params$ID
ID.sig.params <-
  GenerateListOfSigParams(real.exposures = real.exposures.ID,
                          cancer.types = cancer.types,
```

```
                          distribution = "neg.binom",
                          sig.params = sig.params)

ID.synthetic.tumors <-
  GenerateSyntheticTumorsFromSigParams(seed = 191906,
                                       dir = file.path(tempdir(), "ID.synthetic.tumors"),
                                        cancer.types = cancer.types,
                                        samples.per.cancer.type = 30,
                                        input.sigs = input.sigs.ID,
                                        sig.params = ID.sig.params,
                                        distribution = "neg.binom",
                                        sample.prefix.name = "SP.Syn.")

# Plot the synthetic catalog and exposures
ICAMS::PlotCatalogToPdf(catalog = DBS78.synthetic.tumors$ground.truth.catalog,
                        file = file.path(tempdir(), "DBS78.synthetic.catalog.pdf"))
mSigTools::plot_exposure_to_pdf(exposure = DBS78.synthetic.tumors$ground.truth.exposures,
                            file = file.path(tempdir(), "DBS78.synthetic.exposures.pdf"),
                                cex.xaxis = 0.7)
```

`GenSBS1SBS5Exposure` *Generate correlated exposures for multiple tumors*

Description

Generate correlated exposures for multiple tumors

Usage

```
GenSBS1SBS5Exposure(
  main.signature = "SBS5",
  correlated.signature = "SBS1",
  sample.number = 500,
  name.prefix = "TwoCorreSigsGen",
  main.mean.log = 2.5,
  main.stdev.log = 0.25,
  correlated.stdev.log = 0.25,
  slope.linear = 1,
  main.signature.lower.thres = 50,
  correlated.signature.lower.thres = 30,
  pearson.r.2.lower.thres = 0.1,
  pearson.r.2.higher.thres = 1,
  min.main.to.correlated.ratio.linear = 1/3,
  max.main.to.correlated.ratio.linear = Inf
)
```

Arguments

`main.signature`	Name of a signature with smaller variance in the log10 space. (Default: "SBS5")
`correlated.signature`	Name of a signature with larger variance in the log10 space. (Default: "SBS1")
`sample.number`	Number of tumors whose mutation burdens will be generated. (Default: 500)

`name.prefix` Prefix of tumor name. (Default: `"TwoCorreSigsGen"`) By default, the name of tumors to be created will be: TwoCorreSigGen::1, TwoCorreSigGen::2, TwoCorreSigGen::3...

`main.mean.log` Mean of log10(mutation burden of `main.signature`)

`main.stdev.log` Standard deviation of log10(mutation burden of `main.signature`)

`correlated.stdev.log`
Contribute to part of the standard deviation of log10(mutation burden of correlated.signature). In this script, the s.d. of log10(mutation burden of correlated.signature) = main.stdev.log + correlated.stdev.log

`slope.linear` Average ratio of mutation burden of `correlated.signature` over mutation burden of `main.signature`

`main.signature.lower.thres`
Minimum mutation burden (number of mutations) induced by `main.signature` in each tumor.

`correlated.signature.lower.thres`
Minimum mutation burden (number of mutations) induced by `correlated.signature` in each tumor.

`pearson.r.2.lower.thres`
Minimum Pearson's R^2 of mutation burdens of two signatures in `sample.number` tumors.

`pearson.r.2.higher.thres`
Maximum Pearson's R^2 of mutation burdens of two signatures in `sample.number` tumors.

`min.main.to.correlated.ratio.linear`
Minimum ratio of `main.signature` over mutation burden of `correlated.signature` in each tumor.

`max.main.to.correlated.ratio.linear`
Maximum ratio of `main.signature` over mutation burden of `correlated.signature` in each tumor.

Details

Wrapper function around `GenSBS1SBS5ExposureOneTumor`: A function to generate exposure of two correlated signatures (Example: SBS1 and SBS5) for `sample.number` (e.g. 500) synthetic tumors.

NOTE: `pearson.r.2.lower.thres` and `pearson.r2.higher.thres` are used to constraint the Pearson's R^2 of mutation burdens of two signatures in multiple tumors.

`GetNonZeroNoisySample` *Add noise to one sample until the final spectrum has mutation > 0*

Description

Add noise to one sample until the final spectrum has mutation > 0

Usage

```
GetNonZeroNoisySample(
  one.sample,
  input.exposure,
  signatures,
  n.binom.size = 100
)
```

Arguments

one.sample	Name of one sample to add noise.
input.exposure	The exposures to which to add noise; a numeric matrix or data frame in which the rows are signatures and the columns are samples. Each cell indicates the number of mutations due to a particular signature in a particular sample.
signatures	The signatures in the exposure; the column names of signatures have to include all row names in input.exposure; can be an ICAMS catalog or a numerical matrix or data frame.
n.binom.size	If non NULL, use negative binomial noise with this size parameter; see NegBinomial. If NULL, use Poisson noise.

GetSynSigParamsFromExposures

Empirical estimates of key parameters describing exposures due to signatures.

Description

Empirical estimates of key parameters describing exposures due to signatures.

Usage

```
GetSynSigParamsFromExposures(
  exposures,
  verbose = 0,
  distribution = NULL,
  cancer.type = NULL,
  sig.params = NULL
)
```

Arguments

exposures	A matrix in which each column is a sample and each row is a mutation signature, with each element being the "exposure", i.e. mutation count attributed to a (sample, signature) pair.
verbose	If > 0 cat various messages.
distribution	Probability distribution used to fit exposures due to one mutational signature. Can be neg.binom which stands for negative binomial distribution. If NULL (Default), then this function uses log normal distribution with base 10.
cancer.type	Optional argument specifying the cancer type of the samples being analyzed.

`sig.params` Empirical signature parameters generated using real exposures irrespective of their cancer types. If there is only one tumor having a signature in a cancer type in `exposures`, we cannot fit the `distribution` to only one data point. Instead, we will use the empirical parameter `size` from `sig.params`. Users can use `SynSigGen:::GetSynSigParamsFromExposuresOld` to generate their own signature parameters. If `NULL`(default), this function uses the PCAWG7 empirical signature parameters. See `signature.params` for more details.

Value

- For log normal distribution, a data frame with one column for each of a subset of the input signatures and the following rows

  prob The proportion of tumors with the signature.

  mean The mean(log_10(number of mutations)).

  stdev The stdev(log_10(number of mutations)).

  Signatures not present in `exposures` or present only in a single tumor in `exposures` are removed.

- For negative binomial distribution, a data frame with one column for each of a subset of the input signatures and the following rows

  prob The proportion of tumors with the signature.

  size Dispersion parameter.

  mu Mean.

`ManyTypes2700` *Create a specific synthetic data set of 2,700 tumors.*

Description

Data set generated by this function can be found at Synapse with Synapse ID: syn18500213.

Usage

```
ManyTypes2700(
  seed = 191906,
  regress.dir = "data-raw/long.test.regression.data/syn.many.types/",
  top.level.dir = "../Many.types.191906",
  unlink = FALSE
)
```

Arguments

`seed`	A random seed to use.
`regress.dir`	If not `NULL`, compare the result to the contents of this directory with a `diff`.
`top.level.dir`	The directory in which to put the output; will be created if necessary.
`unlink`	If `TRUE` and `!is.null(regress.dir)`, then unlink the result directory if there are no differences.

`MapSPToSASignatureNamesInExposure` *With the signatures represented in a matrix of exposures, find the nearest SignatureAnalyzer exposure.*

Description

With the signatures represented in a matrix of exposures, find the nearest SignatureAnalyzer exposure.

Usage

```
MapSPToSASignatureNamesInExposure(
  sp.exposures,
  sa.sig.names.to.consider = colnames(sa.96.sigs)
)
```

Arguments

`sp.exposures`	The exposures
`sa.sig.names.to.consider`	A subset of the colnames of `sa.96.sigs`

Details

IMPORTANT: uses the package global variables `sa.96.sigs` and `sp.sigs`.

Value

A list with

1. `exp2` Copy of `sp.exposures` with the rownames(signature names) updated according to the match.
2. `sp.to.sa.sig.match`
3. `sa.to.sp.sig.match` Best matches in the opposite direction

`MergeExposures` *Merge all exposure matrices in a list of matrices*

Description

Merge all exposure matrices in a list of matrices

Usage

```
MergeExposures(list.of.exposures)
```

Arguments

`list.of.exposures`
: A list of exposure matrices

Value

The column-wise merge of all the input matrices with all rownames from all matrices preserved and corresponding entries filled with 0s.

`MutationalSignatures` *Reference mutational signature profiles from PCAWG7.*

Description

Reference mutational signature profiles from PCAWG7.

Usage

```
sa.96.sigs

sa.COMPOSITE.sigs

sa.DBS.sigs

sa.ID.sigs

sp.sigs
```

Format

Numerical matrix with rows indicating mutation types and columns indicating signatures.

An object of class `matrix` (inherits from `array`) with 96 rows and 60 columns.

An object of class `matrix` (inherits from `array`) with 1697 rows and 60 columns.

An object of class `matrix` (inherits from `array`) with 78 rows and 15 columns.

An object of class `matrix` (inherits from `array`) with 83 rows and 29 columns.

An object of class `matrix` (inherits from `array`) with 96 rows and 65 columns.

Details

`sa.96.sigs` provides SignatureAnalyzer mutational signature profiles collapsed from COMPOSITE to 96-channel SNS signatures.

`sa.COMPOSITE.sigs` provides COMPOSITE mutational signature profiles extracted by SignatureAnalyzer. `sa.COMPOSITE.sigs` are an `rbind` of the contents of https://www.synapse.org/#!Synapse:syn11738311 (SBS 1536), https://www.synapse.org/#!Synapse:syn11738308 (DBS), and https://www.synapse.org/#!Synapse:syn11738309 (ID).

`sa.DBS.sigs` provides the DBS signatures extracted by SignatureAnalyzer, from https://www.synapse.org/#!Synapse:syn11738312. These are not the DBS signatures that are part of `sa.COMPOSITE.sigs`; these were extracted from the ID catalogs alone.

sa.ID.sigs provides the ID signatures extracted by SignatureAnalyzer, from https://www.synapse.org/#!Synapse:syn11738313.These are not the ID signatures that are part of sa.COMPOSITE.sigs; these were extracted from the ID catalogs alone.

sp.sigs provides signatures extracted by SigProfiler.

Source

https://www.synapse.org/#!Synapse:syn11738310

https://www.synapse.org/#!Synapse:syn11738311

https://www.synapse.org/#!Synapse:syn11738308

https://www.synapse.org/#!Synapse:syn11738309

https://www.synapse.org/#!Synapse:syn11738312

https://www.synapse.org/#!Synapse:syn11738313

https://www.synapse.org/#!Synapse:syn11738319

NewCreateAndWriteCatalog

Create and write a mutational spectra catalog

Description

Create and write a mutational spectra catalog

Usage

```
NewCreateAndWriteCatalog(
  sigs,
  exp,
  dir,
  extra.file.suffix = "",
  overwrite = FALSE
)
```

Arguments

sigs	Signatures to use.
exp	(Synthetic) exposures.
dir	Directory in which to put the signatures; NOTE: this will be a subdirectory based on OutDir.
extra.file.suffix	Extra string to put before ".csv".
overwrite	If TRUE, overwrite existing directory; useful for debugging / testing.

Details

Create a file with the catalog syn.data.csv and writes sigs to input.sigs.csv.

Value

Invisibly, the generated catalog.

NewDiff4SynDataSets *Diff two directories or files.*

Description

Diff two directories or files.

Usage

```
NewDiff4SynDataSets(
  newdir,
  regressdirname,
  unlink,
  verbose = FALSE,
  long.diff = FALSE
)
```

Arguments

newdir	the path of dir2 for a folder to be recursively compared with dir1; it can also be the path of a single file file2 to diff with file1.
regressdirname	the path of dir2 for a folder to be recursively compared with dir1; it can also be the path of a single file file2 to diff with file1.
unlink	if TRUE unlink newdir, but do not unlink if there are diffs.
verbose	Whether to display additional R messages.
long.diff	If TRUE, invoke diff -r (detailed text information even if the two files/folders are the same); if FALSE, invoke diff -rq (detailed text information only if two files/folders are different). (Default: FALSE)

Value

The output of the diff command.

OLD.SplitCatCOMPOSITE *Split COMPOSITE (SNS1536+DBS78+ID83) catalogs in ICAMS format into 3 individual catalogs.*

Description

Split COMPOSITE (SNS1536+DBS78+ID83) catalogs in ICAMS format into 3 individual catalogs.

Usage

```
OLD.SplitCatCOMPOSITE(catalog)
```

Arguments

catalog	Input catalog, can be a .csv file or matrix in ICAMS COMPOSITE format.

Value

a list, containing 3 catalog matrices in MultiModalMuSig format. Each matrix contains SNS1536, DBS78 and ID83 information, respectively.

`OutDir` *Create file names in a given directory*

Description

The directory is provided by the global variable `OutDir.dir`, which **must** be set by the user. If `OutDir.dir` is NULL then just return `file.name`.

Usage

```
OutDir(file.name)
```

Arguments

`file.name`	The name of the that will be prefixed by `OutDir.dir`.

Value

`file.name` prefixed by `OutDir.dir`.

`PancAdenoCA1000` *Create 1000 synthetic pancreatic adenocarcinoma spectra.*

Description

This function generates synthetic tumor spectra with mutational signature prevalence and mutation load similar to pancreatic adenocarcinoma in PCAWG cohort.

Usage

```
PancAdenoCA1000(
  seed = 191907,
  regress.dir = "data-raw/long.test.regression.data/syn.pancreas/",
  num.syn.tumors = 1000,
  top.level.dir = "../Pan-AdenoCA",
  unlink = FALSE
)
```

Arguments

`seed`	A random seed to use.
`regress.dir`	If not `NULL`, compare the result to the contents of this directory with a `diff`.
`num.syn.tumors`	Generate this number of synthetic tumors.
`top.level.dir`	The directory in which to put the output; will be created if necessary.
`unlink`	If `TRUE` and `!is.null(regress.dir)`, then unlink the result directory if there are no differences.

Details

This function replaces `data-raw/Create.pancreas.Rmd` in GitHub repository `steverozen/SynSig`. With default arguments, this function generates the same results as `data-raw/Create.pancreas.Rmd`.

Data set generated by this function can be found at Synapse with Synapse ID: syn18500212.

`PlotCorrelationScatterplot` *Plot scatter plot for correlation between two vectors.*

Description

Plot scatter plot for correlation between two vectors.

Usage

```
PlotCorrelationScatterplot(
  x,
  y,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  optional.remarks = "",
  ...
)
```

Arguments

`x`	vector of exposures of `main.signature` (SBS5 in the paper). The exposures of `main.siganture` will be aligned onto x axis.
`y`	vector of exposures of `correlated.signature` (SBS1 in the paper). The exposures of `correlated.siganture` will be aligned onto y axis.
`xlab`	Label below x axis.
`ylab`	Label below y axis.
`main`	Title on the scatterplot. Default: NULL
`optional.remarks`	Remarks added below the title.
`...`	Other parameters provided to the function `graphics::plot()`.

Details

`PlotCorrelationScatterplot` is a wrapper around `graphics::plot()`, and a function to plot the correlation between two vectors, `x` and `y`. These vectors are expected to be exposures of two signatures.

It will draw a scatterplot, and it will also print information onto the plot, including correlation between `x` and `y`, mean and stdev of `x` and `y`, etc.

`PlotCorrelationScatterplotForExposures` *Plot scatter plot for correlation between exposures of two signatures*

Description

Plot scatter plot for correlation between exposures of two signatures

Usage

```
PlotCorrelationScatterplotForExposures(
  pdf.filename,
  main.signature = "SBS5",
  correlated.signature = "SBS1",
  slope.linear,
  exposure.counts,
  xlim = c(0, 4),
  ylim = c(0, 4),
  ...
)
```

Arguments

`pdf.filename`	Name of the PDF to contain the scatterplots.
`main.signature`	Name of a signature with smaller variance in the log10 space. (Default: "SBS5")
`correlated.signature`	Name of a signature with larger variance in the log10 space. (Default: "SBS1")
`slope.linear`	Average ratio of mutation burden of `correlated.signature` over mutation burden of `main.signature`
`exposure.counts`	Data.frame or matrix storing exposures of two signatures. The exposure.counts object is usually obtained from `SynSig::ReadExposure()`.
`xlim, ylim`	numeric vectors of length 2, giving the x and y coordinates ranges. Default: c(0,4)
`...`	Other parameters provided to the function `graphics::plot()`.
`main`	Title on the scatterplot. Default: NULL

Details

Plot scatter plot for correlation between exposures of two signatures, SBS1 and SBS5 in this study.

`PlotCorrelationScatterplotForExposures` is a wrapper around `PlotCorrelationScatterplot`. It lets `exposure.counts <-` the exposure matrix, and will draw a scatterplot for exposures of two signatures.

RCCOvary1000 *Create synthetic spectra based on renal cell carcinoma and ovarian adenocarcinoma*

Description

Creates spectra dataset consists of 500 synthetic renal cell carcinoma (RCC) with high prevalence and mutation load from SBS5 and SBS40 signatures, and 500 synthetic ovarian adenocarcinoma with high prevalence and mutation load from SBS3. This dataset challenges the computational approaches as these three signatures are "flat" signatures hard to be extracted accurately.

Usage

```
RCCOvary1000(
  seed = 191905,
  unlink = FALSE,
  regress.dir = NULL,
  top.level.dir = "tmp.3.5.40.RCC.and.ovary"
)
```

Arguments

seed	A random seed to use.
unlink	The directory in which to put the output; will be created if necessary.
regress.dir	If not NULL, compare the result to the contents of this directory with a diff.
top.level.dir	The directory in which to put the output; will be created if necessary.

Details

This function Replaces the first part of data-raw/Create.3.5.40.Rmd in GitHub repository steverozen/SynSig. With default arguments, this function generates the same results as the first part of data-raw/Create.3.5.40.Rmd.

The second half of data-raw/Create.3.5.40.Rmd in steverozen/SynSig is replaced by Create.3.5.40.Abstract.

Data set generated by this function can be found at Synapse with Synapse ID: syn18500214.

ReadCatCOMPOSITE *Read a COMPOSITE catalog*

Description

A COMPOSITE catalog is an rbind of a 1536 catalog, a DBS catalog, and an ID catalog. This function does not read SignatureAnalyzer signatures as found on the PCAWG7 Synapse web site, but rather as generated by this package for analysis by SignatureAnalyzer.

Usage

```
ReadCatCOMPOSITE(path, strict = FALSE)
```

Arguments

path	Path of the file to read from.
strict	For compatibility with other ReadCat functions; ignored.

Value

An in memory matrix corresponding to the contents of the file at path.

See Also

WriteCatCOMPOSITE

ReadExposure *Read an exposure matrix from a file*

Description

Read an exposure matrix from a file

Usage

```
ReadExposure(file, check.names = TRUE)
```

Arguments

file	CSV file containing an exposure matrix
check.names	Passed to utils::read.csv. IMPORTANT: If TRUE this will replace the double colon in identifiers of the form <tumor_type>::<sample_id> with two periods (i.e. <tumor_type>..<sample_id>. If check.names is true, generate a warning if double colons were present.

Value

Matrix of exposures

ReadSynapseExposure *Read an exposure matrix from a Synapse file*

Description

Read an exposure matrix from a Synapse file

Usage

```
ReadSynapseExposure(file)
```

Arguments

file	CSV file containing an exposure matrix

Value

Matrix of exposures

`RealExposures` *Real exposure (signature attributions) from SignatureAnalyzer and SigProfiler*

Description

Real exposure (signature attributions) from SignatureAnalyzer and SigProfiler

Usage

```
sa.all.real.exposures

sp.all.real.exposures

sa.no.hyper.real.exposures

sp.no.hyper.real.exposures
```

Format

Numerical matrix with rows indicating signatures and columns indicating (tumor) samples.

An object of class `matrix` (inherits from `array`) with 60 rows and 2780 columns.

An object of class `matrix` (inherits from `array`) with 65 rows and 2780 columns.

An object of class `matrix` (inherits from `array`) with 35 rows and 2624 columns.

An object of class `matrix` (inherits from `array`) with 65 rows and 2624 columns.

Note

Prefix `sa` indicates SignatureAnalyzers, `sp` indicates SigProfiler; `all` indicates all samples, `no.hyper` means that hypermutated tumors as defined for SignatureAnalyzer have been removed.

Source

https://dx.doi.org/10.7303/syn11761237.4

https://dx.doi.org/10.7303/syn11738669.5

https://dx.doi.org/10.7303/syn11761198.4

https://dx.doi.org/10.7303/syn11761237.4

SAAndSPSynDataOneCAType *Generate parallel synthetic exposures from real SA and SP exposures and signatures*

Description

Generate parallel synthetic exposures from real SA and SP exposures and signatures

Usage

```
SAAndSPSynDataOneCAType(
  sa.real.exp,
  sp.real.exp,
  ca.type,
  num.syn.tumors,
  file.prefix,
  top.level.dir = NULL
)
```

Arguments

sa.real.exp	Exposure matrix from SignatureAnalyzer.
sp.real.exp	Exposure matrix from SigProfiler.
ca.type	The type the cancer, which is used in sample identifiers, which SigProfiler expects.
num.syn.tumors	Number of synthetic tumors to generate.
file.prefix	To explain later.
top.level.dir	Specifies the location to generate files.

Value

A list with the following elements:

1. sa.parms The parameters computed from sa.real.exp. This a matrix with a column for each signature and 3 rows:
   (a) The proportion of tumors with given signature (in sa.real.exp).
   (b) The mean of the log10 of the number of mutations for a given signature.
   (c) The standard deviation of log10 of the number of mutations for a given signature.
2. sa.syn.exp The synthetic exposures computed from sa.parms.
3. sp.parms The parameters computed from sp.real.exp, with rows analogous to the rows in sa.parms.
4. sp.syn.exp The synthetic exposures computed from sp.parms.

@details Creates a bunch of files in location governed by top.level.dir. The main rationale for packaging this as one function is to ensure that some conventions regarding file naming are followed.

This function does **not** create the synthetic mutational spectra catalogs but **does** generate the synthetic exposures.

SBS1SBS5datasetNames *Names of datasets generated by* CreateSBS1SBS5CorrelatedSyntheticData.

Description

Names of datasets generated by CreateSBS1SBS5CorrelatedSyntheticData.

Usage

```
SBS1SBS5datasetNames
```

Format

Character vector specifying names of datasets.

Note

S is the initial of "slope", which refers to **SBS1-SBS5 Exposure Ratio**. Rsq is the abbreviation for "Pearson's R squared", which refers to **SBS1-SBS5 Correlation**.

SBS1SBS5parameter *Parameters used to generate synthetic spectra with correlated SBS1 and SBS5 exposures.*

Description

Parameters used to generate synthetic spectra with correlated SBS1 and SBS5 exposures.

Usage

```
SBS1SBS5parameter
```

Format

A data.frame with parameters for generating the synthetic data. The row names of the data.frame refers to the names of datasets, equals to SBS1SBS5datasetNames.

signature.params — *Empirical signature parameters of PCAWG7 platinum tumor exposures using negative binomial distribution*

Description

Empirical signature parameters of PCAWG7 platinum tumor exposures using negative binomial distribution

Usage

```
signature.params
```

Format

An object of class `list` of length 3.

Note

These parameters were generated using all platinum tumors in PCAWG7 exposures irrespective of their cancer types.

SynSigGen — *SynSigGen*

Description

Create catalogs of synthetic mutational spectra for assessing the performance of mutational-signature analysis programs.

Overview

The main focus is generating synthetic catalogs of mutational spectra (mutations in tumors) based on known mutational signature profiles and software-inferred exposures (software's estimate on number of mutations induced by mutational signatures in tumors) in the PCAWG7 data. We call this kind of synthetic data broadly "reality-based" synthetic data. The package also has a set of functions that generate random mutational signature profiles and then create synthetic mutational spectra based on these random signature profiles. We call this kind of synthetic data "random" synthetic data, while pointing out that much depends on the distributions from which the random signature profiles and attributions are generated.

Workflow for generating "reality-based" synthetic mutational spectra

Typical workflow for generating synthetic mutational spectra is as follows.

1. Input (based on SignatureAnalyzer or SigProfiler analysis of PCAWG tumors) `E`, matrix of software-inferred exposures of mutational signatures (signatures x samples) `S`, mutational signature profiles (mutation types x signatures)
2. Obtain distribution parameters from software-inferred exposures

```
P <- GetSynSigParamsFromExposures(E, ...)
```

3. Generate exposures for synthetic mutational spectra based on `P`

```
synthetic.exposures <- GenerateSyntheticExposures(P, ...)
```

4. Generate synthetic mutational spectra by multiplying `S` and `synthetic.exposures`, and round the product to the nearest unit:

```
synthetic.spectra <- CreateAndWriteCatalog(S, synthetic.exposures, ...)
```

Workflow for generating "random" synthetic mutational spectra

The top-level function for generating "random" synthetic mutational spectra is `CreateRandomSyn`. It adopts the following steps to generate catalogs of "random" synthetic mutational spectra.

1. Create random mutational signature profiles:

```
S <- CreateRandomMutSigProfiles(...)
```

2. Generate distribution parameters for exposures of random signatures:

```
P <- CreateMeanAndStdevForSigs(sig.names = colnames(S),...)
```

3. Create exposures for mutational signatures based on `P` and other parameters:

```
synthetic.exposures <- CreateRandomExposures(sigs = S, per.sig.mean.and.sd = P)
```

4. Generate synthetic mutational spectra by multiplying `S` and `synthetic.exposures` and round the product to the nearest unit:

```
synthetic.spectra <- NewCreateAndWriteCatalog(S, synthetic.exposures, ...)
```

Function for generating "SBS1-SBS5-correlated" synthetic mutational spectra

`CreateSBS1SBS5CorrelatedSyntheticData` is the top-level function for generating 20 data sets which only have 2 active signatures (SBS1 and SBS5) with positively-correlated exposures.

This function is used for generating synthetic mutational spectra used in paper "Performance of Mutational Signature Software on Correlated Signatures".

Functions for generating synthetic tumor spectra used in paper *The repertoire of mutational signatures in human cancer*

The repertoire of mutational signatures in human cancer (https://doi.org/10.1038/s41586-020-1943-3) involves evaluation of performances on two computational approaches (`SigProfiler` and `SignatureAnalyzer`) on 11 synthetic data sets (Synapse ID: syn18497223).

1. Function `PancAdenoCA1000` creates 1000 pancreatic adenocarcinoma spectra data set (syn18500212).
2. Script

creates 2,700 synthetic spectra (syn18500213). This data set consists of 9 cancer types each with 300 synthetic tumors:

- bladder transitional cell carcinoma,
- oesophageal adenocarcinoma,
- breast adenocarcinoma,
- lung squamous cell carcinoma,

- renal cell carcinoma,
- ovarian adenocarcinoma,
- osteosarcoma,
- cervical adenocarcinoma and
- stomach adenocarcinoma.

3. Function `RCCOvary1000` creates spectra dataset consists of 500 synthetic kidney (RCC) with high prevalence and mutation load from SBS5 and SBS40 signatures, and 500 synthetic ovarian adenocarcinoma with high prevalence and mutation load from SBS3.

   Notes:

   - Mutation loads from other mutational signatures (besides SBS3, SBS5, SBS30) also exist in the spectra dataset created by function RCCOvary1000;
   - SBS3, SBS5, SBS40 are flat signatures. This dataset challenges the computational approaches on accurately separating these 3 mutational signatures, as mixing SBS5 and SBS40 can get a mutational signature similar to SBS3.

4. Function `Create.3.5.40.Abstract` creates 1000 synthetic spectra all constructed entirely from SBS3, SBS5, and SBS40, using mutational loads modelled on kidney-RCC (SBS5 and SBS40) and ovarian adenocarcinoma (SBS3). Most synthetic spectra have contributions from all three signatures.

`WriteCatCOMPOSITE` *Write a COMPOSITE catalog or signature matrix to disk*

Description

Write a COMPOSITE catalog or signature matrix to disk

Usage

```
WriteCatCOMPOSITE(ct, path)
```

Arguments

`ct`	A catalog or signature matrix
`path`	Path to file to write

See Also

`ReadCatCOMPOSITE`

WriteExposure *Write exposure matrix to a file*

Description

Write exposure matrix to a file

Usage

```
WriteExposure(exposure.matrix, file)
```

Arguments

exposure.matrix	Matrix of exposures
file	File to which to write the exposure matrix (as a CSV file)

WriteSynSigParams *Write key parameters describing exposures due to a signature to a file.*

Description

Write key parameters describing exposures due to a signature to a file.

Usage

```
WriteSynSigParams(
  params,
  file,
  append = FALSE,
  col.names = ifelse(append, FALSE, NA)
)
```

Arguments

params	The parameters to write.
file	The path to the file to write.
append	Whether to append to or overwrite `file` if it already exists.
col.names	If NA, add column names.

Details

The parameters written are prevalence, mean(log(exposure)), and sd(log(exposure)).

Index