Package 'SynSigRun'

May 5, 2020

Type Package

Title Run Mutational Signature Analysis Software Packages Using Mutational Spectra generated by SynSigGen.

Version 0.0.1.9010

Author Steven G. Rozen, Yang Wu

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

Description Create catalogs of synthetic mutational spectra and assess the performance of mutational signature analysis programs on these.

License GPL-3

Language en-US

Encoding UTF-8

LazyData true

Imports lsa,
data.table,
ICAMS,
SynSigGen,
stats,
devtools,
utils,
rlang,
graphics,
grDevices,
ggplot2,
ggpubr,
ggbeeswarm,
gtools

Depends R (>= 3.5)

RoxygenNote 7.1.0

Suggests testthat,
knitr,
rmarkdown,
DelayedArray,
BiocManager,
NMF,
SparseSignatures,
YAPSA,

decompTumor2Sig,
deconstructSigs,
hdp,
maftools,
MutationalPatterns,
mutSignatures,
mSigAct,
rstan,
sigfit,
SignatureEstimation,
signeR

R topics documented:

`CopyBestSignatureAnalyzerResult` *Find the SignatureAnalyzer results directory with the best results and make a copy of it as* `sa.results.dir/best.run/`

Description

Find the SignatureAnalyzer results directory with the best results and make a copy of it as `sa.results.dir/best.run/`

Usage

```
CopyBestSignatureAnalyzerResult(
  sa.results.dir,
  verbose = FALSE,
  overwrite = FALSE
)
```

Arguments

sa.results.dir	See BestSignatureAnalyzerResult
verbose	See BestSignatureAnalyzerResult
overwrite	If TRUE overwrite existing "best.run"

Value

The path of the best directory that was copied as a string, with the list directories examined as the attribute run.directories.

CreateEMuOutput *Prepare input file for EMu from a EMu formatted catalog file.*

Description

Prepare input file for EMu from a EMu formatted catalog file.

Usage

```
CreateEMuOutput(
  catalog,
  out.dir = paste0(dirname(catalog), "/ExtrAttr/EMu.results"),
  overwrite = FALSE
)
```

Arguments

catalog	a catalog in ICAMS format. It can be a .csv file, or a matrix or data.frame. Usually, it refers to "ground.truth.syn.catalog.csv".
out.dir	Directory that will be created for the output; abort if it already exists. Usually, the out.dir will be a EMu.results folder directly under the folder storing catalog.
overwrite	If TRUE, overwrite existing output

Details

Creates folder named EMu.results containing catalogs in EMu-formatted catalogs: Rows are signatures; the first column is the name of the mutation type, while the remaining columns are samples (tumors). These EMu-formated catalogs will the input when running EMu program later on compiled binary.

Value

invisible(catalog), original catalog in EMu format

CreatehelmsmanOutput *Prepare input file for helmsman from a helmsman formatted catalog file.*

Description

Prepare input file for helmsman from a helmsman formatted catalog file.

Usage

```
CreatehelmsmanOutput(
  catalog,
  out.dir = paste0(dirname(catalog), "/ExtrAttr/helmsman.results"),
  overwrite = FALSE
)
```

Arguments

catalog	a catalog in ICAMS format. It can be a .csv file, or a matrix or data.frame. Usually, it refers to "ground.truth.syn.catalog.csv".
out.dir	Directory that will be created for the output; abort if it already exists. Usually, the out.dir will be a helmsman.results folder directly under the folder storing catalog.
overwrite	If TRUE, overwrite existing output

Details

Creates folder named helmsman.results containing catalogs in helmsman-formatted catalogs: Rows are signatures; the first column is the name of the mutation type, while the remaining columns are samples (tumors). These helmsman-formated catalogs will the input when running helmsman program later on Python platform.

Value

invisible(catMatrix), original catalog in helmsman format

CreateMultiModalMuSigOutput *Prepare input file for MultiModalMuSig from a MultiModalMuSig formatted catalog file.*

Description

Prepare input file for MultiModalMuSig from a MultiModalMuSig formatted catalog file.

Usage

```
CreateMultiModalMuSigOutput(
  catalog,
  out.dir = paste0(dirname(catalog), "/ExtrAttr/MultiModalMuSig.results"),
  overwrite = FALSE
)
```

Arguments

catalog	a catalog in ICAMS format. It can be a .csv file, or a matrix or data.frame. Usually, it refers to "ground.truth.syn.catalog.csv".
out.dir	Directory that will be created for the output; abort if it already exists. Usually, the out.dir will be a MultiModalMuSig.results folder directly under the folder storing catalog.
overwrite	If TRUE, overwrite existing output

Details

Creates folder named MultiModalMuSig.results containing catalogs in MultiModalMuSig-formatted catalogs: Rows are signatures; the first column is the name of the mutation type, while the remaining columns are samples (tumors). These MM-formated catalogs will the input when running MultiModalMuSig program later on Julia platform.

Value

invisible(catMatrix), original catalog in MultiModalMuSig format

---

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.

FixSASigNames *Standardize SignatureAnalyzer signature names*

Description

For example, change BI_COMPOSITE_SNV_SBS83_P to BI_COMPOSITE_SBS83_P

Usage

```
FixSASigNames(sig.names)
```

Arguments

sig.names	Vector of signature names

Details

This is necessary because for COMPOSITE signatures we rbind coordinated "SNV", "DNP", and "INDEL" signatures.

Value

Vector of signatures names with "_SNV" removed.

helmsmanCatalog2ICAMS *Read Catalog files or matrices in helmsman format.*

Description

Read Catalog files or matrices in helmsman format.

Usage

```
helmsmanCatalog2ICAMS(
  cat,
  region = "unknown",
  catalog.type = "counts.signature"
)
```

Arguments

cat	Input catalog, can be a tab-delimited text file in helmsman format, or a matrix/data.frame object.
region	Catalog region. Can be a specific genomic or exonic region, or "unknown". Default: "unknown"
catalog.type	Is the catalog a signature catalog, or a spectrum catalog? Default: "counts.signature"

Value

a catalog matrix in ICAMS format.

ICAMSCatalog2EMu *Convert Catalogs from ICAMS format to EMu format*

Description

Convert Catalogs from ICAMS format to EMu format

Usage

```
ICAMSCatalog2EMu(catalog)
```

Arguments

catalog	A catalog matrix in ICAMS format. (SNS only!)

Value

a matrix without any dimnames, but the values are the transposition of the values in `catalog`.

ICAMSCatalog2helmsman *Convert Catalogs from ICAMS format to helmsman format*

Description

Convert Catalogs from ICAMS format to helmsman format

Usage

```
ICAMSCatalog2helmsman(catalog, type = "spectra")
```

Arguments

catalog	A catalog matrix in ICAMS format. (SNS only!)
type	Whether it is a spectra catalog ("spectra") or a signature catalog ("signature").

Value

a catalog matrix in helmsman format.

`ICAMSCatalog2MM` *Convert Catalogs from ICAMS format to MM format*

Description

Convert Catalogs from ICAMS format to MM format

Usage

```
ICAMSCatalog2MM(catalog)
```

Arguments

`catalog`	A catalog matrix in ICAMS format. (SNS/DNS/ID)

Value

a catalog matrix in MultiModalMuSig format.

`InstalldecompTumor2Sig` *Install decompTumor2Sig from Bioconductor*

Description

Install decompTumor2Sig from Bioconductor

Usage

```
InstalldecompTumor2Sig()
```

`InstalldeconstructSigs` *Install deconstructSigs from CRAN*

Description

Install deconstructSigs from CRAN

Usage

```
InstalldeconstructSigs()
```

`InstallmutSignatures` *Install mutSignatures from github*

Description

Install mutSignatures from github

Usage

```
InstallmutSignatures()
```

`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

Match1Sig — *Find the signature in other.sigs that is nearest (by cosine similarity) to query.sig.*

Description

Find the signature in other.sigs that is nearest (by cosine similarity) to query.sig.

Usage

```
Match1Sig(query.sig, other.sigs)
```

Arguments

query.sig	A single signature
other.sigs	Matrix with each column being one signature

Value

The maximum similarity between query.sig and any signature in other.sigs

See Also

Other signature matching functions: MatchSigs1Direction(), MatchSigs2Directions()

MatchSigs1Direction — *Find the closest match in* other.sigs *for each signature in* query.sigs

Description

Find the closest match in other.sigs for each signature in query.sigs

Usage

```
MatchSigs1Direction(query.sigs, other.sigs)
```

Arguments

query.sigs	A signature matrix; signatures for which to find the closest match in other.sigs. The colnames are used as the identifiers of the signatures.
other.sigs	A signature matrix; find the closest matches to a signature in this matrix. The colnames are used as the identifiers of the signatures.

Value

A list with one element for each signature in query.sigs. The names of the list elements are the colnames of query.sigs. Each list element is a vector of length 1, and the name of the vector element is the name of the closest matching signature in other.sigs, and the value is the cosine similarity between the given signature in query.sigs and the matching signature in other.sigs.

See Also

Other signature matching functions: `Match1Sig()`, `MatchSigs2Directions()`

`MatchSigs2Directions` *Calculate bidirectional closest similarities between two sets of signatures and the average of the similarities.*

Description

Calculate bidirectional closest similarities between two sets of signatures and the average of the similarities.

Usage

```
MatchSigs2Directions(sigs1, sigs2)
```

Arguments

`sigs1`	Matrix of signatures; colnames are used as signature identifiers, and the colnames in `sigs1` should be distinguishable from those in `sigs2`.
`sigs2`	Matrix of signatures; colnames are used as signature identifiers.

Value

A list with the elements:

`averCosSim`: the average of the cosine similarities between each signature in `sigs1` and its closest match in `sigs2` and the closest match between each signature in `sigs2` and its closest match in `sigs1`.

`match1`: a data frame with rownames being signature identifiers from `sigs1`, the signature identifier of the closest match in `sigs1` in the 1st column, and the cosine similarity between them in the 2nd column.

`match2`: a data frame with the rownames being signature identifiers from `sigs2`, the signature identifier of the closest match in `sigs1` in the 1st column, and the cosine similarity between them in the 2nd column.

`match1` and `match2` might not have the same number of rows.

See Also

Other signature matching functions: `Match1Sig()`, `MatchSigs1Direction()`

MMCatalog2ICAMS *Convert Catalogs (File or Matrix) from MM format to ICAMS format*

Description

Convert Catalogs (File or Matrix) from MM format to ICAMS format

Usage

```
MMCatalog2ICAMS(cat, region = "unknown", catalog.type = "counts.signature")
```

Arguments

cat	Input catalog, can be a tab-delimited file or matrix in MultiModalMuSig format.
region	Catalog region. Can be a specific genomic or exonic region, or "unknown". Default: "unknown"
catalog.type	Is the catalog a signature catalog, or a spectrum catalog? Default: "counts.signature"

Value

a catalog matrix in ICAMS format.

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

NumFromId *Get the numerical parts of signature ids*

Description

Get the numerical parts of signature ids

Usage

```
NumFromId(s)
```

Arguments

`s`	A character vector

Value

A vector, each element of which is the integer corresponding to the first string of digits of an element of s

`ReadEMuCatalog` *Read Catalog files in EMu format.*

Description

Read Catalog files in EMu format.

Usage

```
ReadEMuCatalog(
  cat,
  mutTypes,
  sigOrSampleNames,
  region = "unknown",
  catalog.type = "counts.signature"
)
```

Arguments

`cat`	A tab-delimited catalog text file in EMu format; or a EMu formatted matrix or data.frame.
`mutTypes`	Types of mutations. They are usually from an `ICAMS:::catalog.row.header` object.
`sigOrSampleNames`	If input file is a counts signature file (`catalog.type == "counts.signature"`), signature names should be provided. If input file is a counts spectra file (`catalog.type == "counts"`), names of samples should be provided.
`region`	Catalog region. Can be a specific genomic or exonic region, or "unknown". Default: "unknown"
`catalog.type`	Is the catalog a signature catalog, or a spectrum catalog? Default: "counts"

Value

a catalog matrix in ICAMS format.

`ReadEMuExposureFile` *Read Exposure files in EMu format.*

Description

Read Exposure files in EMu format.

Usage

```
ReadEMuExposureFile(exposureFile, sigNames, sampleNames)
```

Arguments

exposureFile	Exposure file generated by EMu. Usually, it is called "W_components.txt".
sigNames	Names of signatures. These will be served as the rownames of the exposure matrix.
sampleNames	Names of samples in exposure file.
	Return ICAMS/SynSigEval formatted exposure matrix.

ReadExposureMM *Read Catalog files in MM format*

Description

Read Catalog files in MM format

Usage

```
ReadExposureMM(exposureFile)
```

Arguments

exposureFile	Input exposure file, can be a tab-delimited text file in MultiModalMuSig format.

Value

a exposure matrix in ICAMS format.

ReadhelmsmanExposure *Read Exposure files in helmsman format.*

Description

Read Exposure files in helmsman format.

Usage

```
ReadhelmsmanExposure(exposure, check.names = TRUE)
```

Arguments

exposure	Exposure file generated by helmsman. Usually, it is called "W_components.txt".
check.names	logical. If TRUE then the names of the variables in the data frame are checked to ensure that they are syntactically valid variable names. If necessary they are adjusted (by make.names) so that they are, and also to ensure that there are no duplicates.
	Return ICAMS/SynSigEval formatted exposure matrix.

`ReadSigProfilerExposure` *Read a file containing exposures attributed by SigProfiler/Python*

Description

Read a file containing exposures attributed by SigProfiler/Python

Usage

```
ReadSigProfilerExposure(file)
```

Arguments

`file`	The name of the file to read.

Value

The corresponding signature matrix in standard internal representation.

`ReadSigProfilerSig96` *Read a file containing SBS96 signatures extracted by SigProfiler/Python*

Description

Read a file containing SBS96 signatures extracted by SigProfiler/Python

Usage

```
ReadSigProfilerSig96(file)
```

Arguments

`file`	The name of the file to read.

Value

The corresponding signature matrix in standard internal representation.

`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

RunAndEvalHdp *Run and evaluate hdp*

Description

Run and evaluate hdp

Usage

```
RunAndEvalHdp(
  input.catalog.file,
  ground.truth.exposure.file,
  ground.truth.sig.file = NULL,
  ground.truth.sig.catalog = NULL,
  out.dir,
  CPU.cores = 1,
  seedNumber = 1,
  K.guess,
  multi.types = FALSE,
  remove.noise = FALSE,
  test.only = 0,
  overwrite = FALSE,
  verbose = TRUE,
  num.posterior = 4,
  post.burnin = 4000,
  post.n = 50,
  post.space = 50,
  post.cpiter = 3,
  post.verbosity = 0,
  cos.merge = 0.9,
  min.sample = 1
)
```

Arguments

input.catalog.file	File containing a spectra catalog in ICAMS format.
ground.truth.exposure.file	Path to file with ground truth exposures.
ground.truth.sig.file	Path to file with ground truth signatures.
ground.truth.sig.catalog	ICAMS catalog with signatures used to construct the ground truth spectra. Specify only one of ground.truth.sig.file.path or ground.truth.sig.catalog.
out.dir	Directory that will be created for the output; if overwrite is FALSE then abort if out.dir already exits.
CPU.cores	Number of CPUs to use in running hdp_posterior; this is used to parallize running the posterior sampling chains, so there is no point in making this larger than num.posterior.

seedNumber	An integer that is used to generate separate random seeds for each call to dp_activate, and each call of hdp_posterior; please see the code on how this is done. But repeated calls with same value of seedNumber and other inputs should produce the same results.
K.guess	Suggested initial value of the number of signatures, passed to dp_activate as initcc.
multi.types	A logical scalar or a character vector. If FALSE, hdp will regard all input spectra as one tumor type. If TRUE, hdp will infer tumor types based on the string before "::" in their names. e.g. tumor type for "SA.Syn.Ovary-AdenoCA::S.500" would be "SA.Syn.Ovary-AdenoCA" If multi.types is a character vector, then it should be of the same length as the number of columns in input.catalog, and each value is the name of the tumor type of the corresponding column in input.catalog, e.g. c("SA.Syn.Ovary-AdenoCA","SA.Syn.O
remove.noise	Deprecated; ignored
test.only	If > 0, only analyze the first test.only columns in input.catalog.file.
verbose	If TRUE then message progress information.
num.posterior	Number of posterior sampling chains; can set to 1 for testing.
post.burnin	Pass to hdp_posterior burnin.
post.n	Pass to hdp_posterior n.
post.space	Pass to hdp_posterior space.
post.cpiter	Pass to hdp_posterior cpiter.
post.verbosity	Pass to hdp_posterior verbosity.
cos.merge	The cosine similarity threshhold for merging raw clusters from the posterior sampling chains into "components" i.e. signatures; passed to hdp_extract_components.
min.sample	A "component" (i.e. signature) must have at least this many samples; passed to hdp_extract_components.

RundecompTumor2SigAttributeOnly *Run decompTumor2Sig attribution on a spectra catalog file and known signatures.*

Description

Run decompTumor2Sig attribution on a spectra catalog file and known signatures.

Usage

```
RundecompTumor2SigAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  seedNumber = 1,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
gt.sigs.file	File containing input mutational signatures. Columns are signatures, rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
seedNumber	Specify the pseudo-random seed number used to run deconstructSigs. Setting seed can make the attribution of deconstructSigs repeatable. Default: 1.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in paste0(out.dir,"/sa.output.rdata"). These are TODO(Steve): list the files

Value

The inferred exposure of deconstructSigs, invisibly.

RundeconstructSigsAttributeOnly

Run deconstructSigs attribution on a spectra catalog file and known signatures.

Description

Run deconstructSigs attribution on a spectra catalog file and known signatures.

Usage

```
RundeconstructSigsAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  seedNumber = 1,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
gt.sigs.file	File containing input mutational signatures. Columns are signatures, rows are mutation types.

out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
seedNumber	Specify the pseudo-random seed number used to run deconstructSigs. Setting seed can make the attribution of deconstructSigs repeatable. Default: 1.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in paste0(out.dir,"/sa.output.rdata"). These are TODO(Steve): list the files

Value

The inferred exposure of deconstructSigs, invisibly.

Runhdp *Run hdp extraction and attribution on a spectra catalog file*

Description

Run hdp extraction and attribution on a spectra catalog file

Usage

```
Runhdp(
  input.catalog,
  out.dir,
  CPU.cores = 1,
  seedNumber = 1,
  K.guess,
  multi.types = FALSE,
  remove.noise = FALSE,
  test.only = FALSE,
  overwrite = FALSE,
  verbose = TRUE
)
```

Arguments

input.catalog	File containing a spectra catalog in ICAMS format.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
CPU.cores	Number of CPUs to use in running hdp_posterior.
seedNumber	Specify the pseudo-random seed number used to run hdp. Setting seed can make the attribution of hdp repeatable. Default: 1.
K.guess	Suggested initial value of the number of signatures, passed to dp_activate as initcc.

multi.types	A logical scalar or a character vector. If FALSE, hdp will regard all input spectra as one tumor type, and will allocate them to one single dirichlet process node. If TRUE, hdp will infer tumor types based on the string before "::" in their names. e.g. Tumor type for "SA.Syn.Ovary-AdenoCA::S.500" would be "SA.Syn.Ovary-AdenoCA" If it is a character vector, it should be a vector of case-sensitive tumor types. e.g. c("SA.Syn.Ovary-AdenoCA","SA.Syn.Ovary-AdenoCA","SA.Syn.Kidney-RCC").
remove.noise	Whether to remove noise signature "hdp.0"? In normal cases scenarios, only few mutations will be assigned to noise signature. For result visualization and assessment of hdp package, select TRUE; for diagnostic purposes, select FALSE.
test.only	If TRUE, only analyze the first 10 columns in input.catalog.
overwrite	If TRUE, overwrite existing output.
verbose	If TRUE then message progress information.

Details

Creates several files in out.dir. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The attributed exposure of hdp, invisibly.

Runhdp2 *Run hdp extraction and attribution on a spectra catalog file*

Description

Run hdp extraction and attribution on a spectra catalog file

Usage

```
Runhdp2(
  input.catalog.file,
  out.dir,
  CPU.cores = 1,
  seedNumber = 1,
  K.guess,
  multi.types = FALSE,
  remove.noise = FALSE,
  test.only = 0,
  overwrite = FALSE,
  verbose = TRUE,
  num.posterior = 4,
  post.burnin = 4000,
  post.n = 50,
  post.space = 50,
  post.cpiter = 3,
```

```
  post.verbosity = 0,
  cos.merge = 0.9,
  min.sample = 1
)
```

Arguments

input.catalog.file	File containing a spectra catalog in ICAMS format.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
CPU.cores	Number of CPUs to use in running hdp_posterior; this is used to parallize running the posterior sampling chains, so there is no point in making this larger than num.posterior.
seedNumber	Specify the random seed for repeatable results.
K.guess	Suggested initial value of the number of signatures, passed to dp_activate as initcc.
multi.types	A logical scalar or a character vector. If FALSE, hdp will regard all input spectra as one tumor type. If TRUE, hdp will infer tumor types based on the string before "::" in their names. e.g. tumor type for "SA.Syn.Ovary-AdenoCA::S.500" would be "SA.Syn.Ovary-AdenoCA" If multi.types is a character vector it should be of the same length as the number of columns in input.catalog, and each value is the name of the tumor type of the corresponding column in input.catalog, e.g. c("SA.Syn.Ovary-AdenoCA","SA.Syn.Ovary
remove.noise	Deprecated; ignored For result visualization and assessment of hdp package, select TRUE; for diagnostic purposes, select FALSE.
test.only	If > 0, only analyze the first test.only columns in input.catalog.file.
verbose	If TRUE then message progress information.
num.posterior	Number of posterior sampling chains; can set to 1 for testing.
post.burnin	Pass to hdp_posterior burnin.
post.n	Pass to hdp_posterior n.
post.space	Pass to hdp_posterior space.
post.cpiter	Pass to hdp_posterior cpiter.
post.verbosity	Pass to hdp_posterior verbosity.
cos.merge	The cosine similarity threshhold for merging raw clusters from the posterior sampling chains into "components" i.e. signatures; passed to hdp_extract_components.
min.sample	A "component" (i.e. signature) must have at least this many samples; passed to hdp_extract_components.

Details

Creates several files in out.dir. These are: TODO(Steve): list the files

Value

The same list as returned by RunhdpInternal.

Runhdp3 *Run hdp extraction and attribution on a spectra catalog file*

Description

Run hdp extraction and attribution on a spectra catalog file

Usage

```
Runhdp3(
  input.catalog.file,
  out.dir,
  CPU.cores = 1,
  seedNumber = 1,
  K.guess,
  multi.types = FALSE,
  remove.noise = FALSE,
  test.only = 0,
  overwrite = FALSE,
  verbose = TRUE,
  num.posterior = 4,
  post.burnin = 4000,
  post.n = 50,
  post.space = 50,
  post.cpiter = 3,
  post.verbosity = 0,
  cos.merge = 0.9,
  min.sample = 1
)
```

Arguments

input.catalog.file	File containing a spectra catalog in ICAMS format.
out.dir	Directory that will be created for the output; if overwrite is FALSE then abort if out.dir already exits.
CPU.cores	Number of CPUs to use in running hdp_posterior; this is used to parallize running the posterior sampling chains, so there is no point in making this larger than num.posterior.
seedNumber	An integer that is used to generate separate random seeds for each call to dp_activate, and each call of hdp_posterior; please see the code on how this is done. But repeated calls with same value of seedNumber and other inputs should produce the same results.
K.guess	Suggested initial value of the number of signatures, passed to dp_activate as initcc.
multi.types	A logical scalar or a character vector. If FALSE, hdp will regard all input spectra as one tumor type. If TRUE, hdp will infer tumor types based on the string before "::" in their names. e.g. tumor type for "SA.Syn.Ovary-AdenoCA::S.500" would be "SA.Syn.Ovary-AdenoCA"

	If multi.types is a character vector, then it should be of the same length as the number of columns in input.catalog, and each value is the name of the tumor type of the corresponding column in input.catalog, e.g. c("SA.Syn.Ovary-AdenoCA","SA.Syn.O
remove.noise	Deprecated; ignored
test.only	If > 0, only analyze the first test.only columns in input.catalog.file.
verbose	If TRUE then message progress information.
num.posterior	Number of posterior sampling chains; can set to 1 for testing.
post.burnin	Pass to hdp_posterior burnin.
post.n	Pass to hdp_posterior n.
post.space	Pass to hdp_posterior space.
post.cpiter	Pass to hdp_posterior cpiter.
post.verbosity	Pass to hdp_posterior verbosity.
cos.merge	The cosine similarity threshhold for merging raw clusters from the posterior sampling chains into "components" i.e. signatures; passed to hdp_extract_components.
min.sample	A "component" (i.e. signature) must have at least this many samples; passed to hdp_extract_components.

Details

Creates several files in out.dir. These are: TODO(Steve): list the files

Value

The same list as returned by RunhdpInternal.

RunhdpInternal *Run hdp extraction and attribution on a spectra catalog file*

Description

Run hdp extraction and attribution on a spectra catalog file

Usage

```
RunhdpInternal(
  input.catalog,
  CPU.cores = 1,
  seedNumber = 1,
  K.guess,
  multi.types = FALSE,
  verbose = TRUE,
  num.posterior = 4,
  post.burnin = 4000,
  post.n = 50,
  post.space = 50,
  post.cpiter = 3,
  post.verbosity = 0,
  cos.merge = 0.9,
  min.sample = 1
)
```

Arguments

input.catalog	A catalog of spectra catalog in ICAMS format.
CPU.cores	Number of CPUs to use in running hdp_posterior; this is used to parallize running the posterior sampling chains, so there is no point in making this larger than num.posterior.
seedNumber	Specify the random seed for repeatable results.
K.guess	Suggested initial value of the number of signatures, passed to dp_activate as initcc.
multi.types	A logical scalar or a character vector. If FALSE, hdp will regard all input spectra as one tumor type. If TRUE, hdp will infer tumor types based on the string before "::" in their names. e.g. tumor type for "SA.Syn.Ovary-AdenoCA::S.500" would be "SA.Syn.Ovary-AdenoCA" If multi.types is a character vector it should be of the same length as the number of columns in input.catalog, and each value is the name of the tumor type of the corresponding column in input.catalog, e.g. c("SA.Syn.Ovary-AdenoCA","SA.Syn.Ovary
verbose	If TRUE then message progress information.
num.posterior	Number of posterior sampling chains; can set to 1 for testing.
post.burnin	Pass to hdp_posterior burnin.
post.n	Pass to hdp_posterior n.
post.space	Pass to hdp_posterior space.
post.cpiter	Pass to hdp_posterior cpiter.
post.verbosity	Pass to hdp_posterior verbosity.
cos.merge	The cosine similarity threshhold for merging raw clusters from the posterior sampling chains into "components" i.e. signatures; passed to hdp_extract_components.
min.sample	A "component" (i.e. signature) must have at least this many samples; passed to hdp_extract_components.

Value

A list with the following elements:

signature The extracted signature profiles as a matrix; rows are mutation types, columns are samples (e.g. tumors).

exposure The inferred exposures as a matrix of mutation counts; rows are signatures, columns are samples (e.g. tumors).

exposure.p exposure converted to proportions.

multi.chains A hdpSampleMulti-class object.

RunhdpInternal3 *Run hdp extraction and attribution on a spectra catalog file*

Description

Run hdp extraction and attribution on a spectra catalog file

Usage

```
RunhdpInternal3(
  input.catalog,
  CPU.cores = 1,
  seedNumber = 1,
  K.guess,
  multi.types = FALSE,
  verbose = TRUE,
  num.posterior = 4,
  post.burnin = 4000,
  post.n = 50,
  post.space = 50,
  post.cpiter = 3,
  post.verbosity = 0,
  cos.merge = 0.9,
  min.sample = 1
)
```

Arguments

input.catalog	A catalog of spectra catalog in ICAMS format.
CPU.cores	Number of CPUs to use in running hdp_posterior; this is used to parallize running the posterior sampling chains, so there is no point in making this larger than num.posterior.
seedNumber	An integer that is used to generate separate random seeds for each call to dp_activate, and each call of hdp_posterior; please see the code on how this is done. But repeated calls with same value of seedNumber and other inputs should produce the same results.
K.guess	Suggested initial value of the number of signatures, passed to dp_activate as initcc.
multi.types	A logical scalar or a character vector. If FALSE, hdp will regard all input spectra as one tumor type. If TRUE, hdp will infer tumor types based on the string before "::" in their names. e.g. tumor type for "SA.Syn.Ovary-AdenoCA::S.500" would be "SA.Syn.Ovary-AdenoCA" If multi.types is a character vector, then it should be of the same length as the number of columns in input.catalog, and each value is the name of the tumor type of the corresponding column in input.catalog, e.g. c("SA.Syn.Ovary-AdenoCA","SA.Syn.O
verbose	If TRUE then message progress information.
num.posterior	Number of posterior sampling chains; can set to 1 for testing.
post.burnin	Pass to hdp_posterior burnin.

post.n	Pass to hdp_posterior n.
post.space	Pass to hdp_posterior space.
post.cpiter	Pass to hdp_posterior cpiter.
post.verbosity	Pass to hdp_posterior verbosity.
cos.merge	The cosine similarity threshhold for merging raw clusters from the posterior sampling chains into "components" i.e. signatures; passed to hdp_extract_components.
min.sample	A "component" (i.e. signature) must have at least this many samples; passed to hdp_extract_components.

Value

A list with the following elements:

signature The extracted signature profiles as a matrix; rows are mutation types, columns are samples (e.g. tumors).

exposure The inferred exposures as a matrix of mutation counts; rows are signatures, columns are samples (e.g. tumors).

exposure.p exposure converted to proportions.

multi.chains A hdpSampleMulti-class object. This object has the method chains which returns a list of hdpSampleChain-class objects. Each of these sample chains objects has a method final_hdpState (actually the methods seems to be just hdp) that returns the hdpState from which it was generated.

Runmaftools *Run maftools extraction ONLY on a spectra catalog file*

Description

WARNING: maftools can only do signature extraction!

Usage

```
Runmaftools(
  input.catalog,
  out.dir,
  CPU.cores = NULL,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
CPU.cores	Number of CPUs to use in running maftools. For a server, 30 cores would be a good choice; while for a PC, you may only choose 2-4 cores. By default (CPU.cores = NULL), the CPU.cores would be equal to (parallel::detectCores())/2, total number of CPUs divided by 2.
seedNumber	Specify the pseudo-random seed number used to run maftools. Setting seed can make the attribution of maftools repeatable. Default: 1.
K.exact, K.range	K.exact is the exact value for the number of signatures active in spectra (K). Specify K.exact if you know exactly how many signatures are active in the input.catalog, which is the ICAMS-formatted spectra file. K.range is A numeric vector (K.min,K.max) of length 2 which tell maftools to search the best signature number active in spectra, K, in this range of Ks. Specify K.range if you don't know how many signatures are active in the input.catalog. WARNING: You must specify only one of K.exact or K.range! Default: NULL
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in out.dir. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The extracted signatures of maftools, invisibly.

RunmSigActAttributeOnly

Run mSigAct attribution on a spectra catalog file and known signatures.

Description

Run mSigAct attribution on a spectra catalog file and known signatures.

Usage

```
RunmSigActAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  CPU.cores = NULL,
  seedNumber = 1,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
gt.sigs.file	File containing input mutational signatures. Columns are signatures, rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
CPU.cores	Number of CPUs to use in running sigfit. For a server, 30 cores would be a good choice; while for a PC, you may only choose 2-4 cores. By default (CPU.cores = NULL), the CPU.cores would be equal to (parallel::detectCores())/2, total number of CPUs divided by 2.
seedNumber	Specify the pseudo-random seed number used to run mSigAct. Setting seed can make the attribution of mSigAct repeatable. Default: 1.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in paste0(out.dir,"/sa.output.rdata"). These are TODO(Steve): list the files

Value

The inferred exposure of mSigAct, invisibly.

RunMutationalPatterns *Run MutationalPatterns extraction and attribution on a spectra catalog file*

Description

WARNING: MutationalPatterns can only do exposure attribution using SBS96 spectra catalog and signature catalog!

Usage

```
RunMutationalPatterns(
  input.catalog,
  out.dir,
  CPU.cores = NULL,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
CPU.cores	Number of CPUs to use in running MutationalPatterns. For a server, 30 cores would be a good choice; while for a PC, you may only choose 2-4 cores. By default (CPU.cores = NULL), the CPU.cores would be equal to (parallel::detectCores())/2, total number of CPUs divided by 2.
seedNumber	Specify the pseudo-random seed number used to run MutationalPatterns. Setting seed can make the attribution of MutationalPatterns repeatable. Default: 1.
K.exact, K.range	K.exact is the exact value for the number of signatures active in spectra (K). Specify K.exact if you know exactly how many signatures are active in the input.catalog, which is the ICAMS-formatted spectra file. K.range is A numeric vector (K.min,K.max) of length 2 which tell MutationalPatterns to search the best signature number active in spectra, K, in this range of Ks. Specify K.range if you don't know how many signatures are active in the input.catalog. WARNING: You must specify only one of K.exact or K.range! Default: NULL
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in out.dir. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The inferred exposure of MutationalPatterns, invisibly.

RunMutationalPatternsAttributeOnly *Run MutationalPatterns attribution on a spectra catalog file and known signatures.*

Description

Run MutationalPatterns attribution on a spectra catalog file and known signatures.

Usage

```
RunMutationalPatternsAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  seedNumber = 1,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
gt.sigs.file	File containing input mutational signatures. Columns are signatures, rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
seedNumber	Specify the pseudo-random seed number used to run MutationalPatterns. Setting seed can make the attribution of MutationalPatterns repeatable. Default: 1.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in paste0(out.dir,"/sa.output.rdata"). These are TODO(Steve): list the files

Value

The inferred exposure of MutationalPatterns, invisibly.

RunmutSignatures *Run mutSignatures extraction and attribution on a spectra catalog file*

Description

Run mutSignatures extraction and attribution on a spectra catalog file

Usage

```
RunmutSignatures(
  input.catalog,
  out.dir,
  algorithm = "alexa",
  CPU.cores = NULL,
  iterations = 1000,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
algorithm	NMF implementation used to to extract signatures and attribute exposures. Only "alexa", "brunet" or "lin" is valid. "alexa" or "brunet": Jean-Philippe Brunet's implementation. This is the most widely used NMF implementation for signature extraction. DOI: 10.1073/pnas.0308531101 "lin": Chih-Jen Lin's implementation. DOI:10.1109/TNN.2007.895831 Default: "alexa".
CPU.cores	Number of CPUs to use in running sigfit. For a server, 30 cores would be a good choice; while for a PC, you may only choose 2-4 cores. By default (CPU.cores = NULL), the CPU.cores would be equal to (parallel::detectCores())/2, total number of CPUs divided by 2.
iterations	Number of iterations in signature extraction. Default: 1000.
seedNumber	Specify the pseudo-random seed number used to run sigfit. Setting seed can make the attribution of sigfit repeatable. Default: 1.
K.exact, K.range	K.exact is the exact value for the number of signatures active in spectra (K). Specify K.exact if you know exactly how many signatures are active in the input.catalog, which is the ICAMS-formatted spectra file. K.range is A numeric vector (K.min,K.max) of length 2 which tell sigfit to search the best signature number active in spectra, K, in this range of Ks. Specify K.range if you don't know how many signatures are active in the input.catalog. K.max - K.min >= 3, otherwise an error will be thrown.

	WARNING: You must specify only one of `K.exact` or `K.range`! Default: NULL
`test.only`	If TRUE, only analyze the first 10 columns read in from `input.catalog`. Default: FALSE
`overwrite`	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in `out.dir`. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The inferred exposure of `mutSignatures`, invisibly.

`RunmutSignaturesAttributeOnly` *Run mutSignatures attribution on a spectra catalog file and known signatures.*

Description

Run mutSignatures attribution on a spectra catalog file and known signatures.

Usage

```
RunmutSignaturesAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  seedNumber = 1,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

`input.catalog`	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
`gt.sigs.file`	File containing input mutational signatures. Columns are signatures, rows are mutation types.
`out.dir`	Directory that will be created for the output; abort if it already exits. Log files will be in `paste0(out.dir,"/tmp")`.
`seedNumber`	Specify the pseudo-random seed number used to run mutSignatures. Setting seed can make the attribution of mutSignatures repeatable. Default: 1.
`test.only`	If TRUE, only analyze the first 10 columns read in from `input.catalog`. Default: FALSE
`overwrite`	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in `paste0(out.dir,"/sa.output.rdata")`. These are TODO(Steve): list the files

Value

The inferred exposure of `mutSignatures`, invisibly.

`RunmutSpec` *Run mutSpec extraction and attribution on a spectra catalog file*

Description

NOTE: mutSpec can only do exposure attribution using SBS96 spectra catalog and signature catalog!

Usage

```
RunmutSpec(
  input.catalog,
  out.dir,
  CPU.cores = NULL,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

`input.catalog`	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
`out.dir`	Directory that will be created for the output; abort if it already exits. Log files will be in `paste0(out.dir,"/tmp")`.
`CPU.cores`	Number of CPUs to use in running mutSpec. For a server, 30 cores would be a good choice; while for a PC, you may only choose 2-4 cores. By default (CPU.cores = NULL), the CPU.cores would be equal to `(parallel::detectCores())/2`, total number of CPUs divided by 2.
`seedNumber`	Specify the pseudo-random seed number used to run mutSpec. Setting seed can make the attribution of mutSpec repeatable. Default: 1.
`K.exact, K.range`	`K.exact` is the exact value for the number of signatures active in spectra (K). Specify `K.exact` if you know exactly how many signatures are active in the `input.catalog`, which is the `ICAMS`-formatted spectra file. `K.range` is A numeric vector `(K.min,K.max)` of length 2 which tell mutSpec to search the best signature number active in spectra, K, in this range of Ks. Specify `K.range` if you don't know how many signatures are active in the `input.catalog`. WARNING: You must specify only one of `K.exact` or `K.range`! Default: NULL

test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in out.dir. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The inferred exposure of mutSpec, invisibly.

Runsigfit *Run sigfit extraction and attribution on a spectra catalog file*

Description

WARNING: sigfit can only do exposure attribution using SBS96 spectra catalog and signature catalog!

Usage

```
Runsigfit(
  input.catalog,
  out.dir,
  model = "nmf",
  CPU.cores = NULL,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
model	Algorithm to be used to extract signatures and attribute exposures. Only "nmf" or "emu" is valid. Default: "nmf".
CPU.cores	Number of CPUs to use in running sigfit. For a server, 30 cores would be a good choice; while for a PC, you may only choose 2-4 cores. By default (CPU.cores = NULL), the CPU.cores would be equal to (parallel::detectCores())/2, total number of CPUs divided by 2.
seedNumber	Specify the pseudo-random seed number used to run sigfit. Setting seed can make the attribution of sigfit repeatable. Default: 1.

`K.exact, K.range`
: `K.exact` is the exact value for the number of signatures active in spectra (K). Specify `K.exact` if you know exactly how many signatures are active in the `input.catalog`, which is the `ICAMS`-formatted spectra file.

  `K.range` is A numeric vector (`K.min,K.max`) of length 2 which tell sigfit to search the best signature number active in spectra, K, in this range of Ks. Specify `K.range` if you don't know how many signatures are active in the `input.catalog`. K.max - K.min >= 3, otherwise an error will be thrown.

  WARNING: You must specify only one of `K.exact` or `K.range`!

  Default: NULL

`test.only`
: If TRUE, only analyze the first 10 columns read in from `input.catalog`. Default: FALSE

`overwrite`
: If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in `out.dir`. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The inferred exposure of `sigfit`, invisibly.

`RunsigfitAttributeOnly` *Run sigfit attribution on a spectra catalog file and known signatures.*

Description

Run sigfit attribution on a spectra catalog file and known signatures.

Usage

```
RunsigfitAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  model = "nmf",
  seedNumber = 1,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

`input.catalog`
: File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.

`gt.sigs.file`
: File containing input mutational signatures. Columns are signatures, rows are mutation types.

out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
model	Algorithm to be used to extract signatures and attribute exposures. Only "nmf" or "emu" is valid. Default: "nmf".
seedNumber	Specify the pseudo-random seed number used to run sigfit. Setting seed can make the attribution of sigfit repeatable. Default: 1.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in paste0(out.dir,"/sa.output.rdata"). These are TODO(Steve): list the files

Value

The inferred exposure of sigfit, invisibly.

RunSignatureAnalyzerAttribution *Run SignatureAnalyzer attribution on a catalog file and output by RunSignatureAnalyzerOnFile().*

Description

Normally, please call SignatureAnalyzerOneRun instead of this function.

Usage

```
RunSignatureAnalyzerAttribution(
  input.catalog,
  read.catalog.function,
  extracted.signature.file,
  raw.exposures.file,
  write.signature.function,
  out.dir,
  test.only = FALSE,
  input.exposures = NULL,
  delete.tmp.files = TRUE,
  overwrite = FALSE,
  verbose = FALSE
)
```

Arguments

input.catalog	File containing input catalog. Columns are samples (tumors), rows are signatures. SignatureAnalyzer does not care about the row names (I think) TODO(Steve): check this.

`read.catalog.function`
: Function taking a file path as its only argument and returning a catalog as a numeric matrix.

`extracted.signature.file`
: A .csv file containing extracted signatures. Normally, this file is named `"sa.output.sigs.csv"` and is generated by function `RunSignatureAnalyzerOnFile()`. It expects to have the same format as the `input.catalog`, thus it will be read by `read.catalog.function` too.

`raw.exposures.file`
: A .csv file containing raw attributions of exposures. Normally, this file is named `"sa.output.raw.exp.csv"` and is generated by function `RunSignatureAnalyzerOnFile()`.

`write.signature.function`
: Function with first argument the signatures generated by SignatureAnalyzer and second argument the file to write to.

`out.dir`
: Directory that will be created for the output; abort if it already exits. Log files will be in `paste0(out.dir,"/tmp")`.

`test.only`
: If TRUE, only analyze the first 10 columns read in from `input.catalog`.

`input.exposures`
: A file with the synthetic exposures used to generate `input.catalog`; if provided here, this is copied over to the output directory for downstream analysis.

`delete.tmp.files`
: If TRUE delete the many temporary files generated by SignatureAnalyzer.

`overwrite`
: If TRUE, overwrite existing output.

`verbose`
: If `TRUE` cat a message regarding progress.

Details

Save the final attribution of a catalog matrix into a file named `"sa.output.fine.exp.csv"` under the folder `out.dir`.

Value

The final attribution matrix. (i.e. `exp.fine.tuned`)

`RunSignatureAnalyzerOnFile`
: *Run SignatureAnalyzer on a file containing a catalog AFTER the SignatureAnalyzer code has been source'ed.*

Description

Normally, please call `SignatureAnalyzerOneRun` instead of this function.

Usage

```
RunSignatureAnalyzerOnFile(
  input.catalog,
  out.dir,
  input.exposures = NULL,
  maxK = 30,
  tol = 1e-07,
  test.only = FALSE,
  delete.tmp.files = TRUE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input catalog. Columns are samples (tumors), rows are signatures. SignatureAnalyzer does not care about the row names (I think) TODO(Steve): check this.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
input.exposures	A file with the synthetic exposures used to generate input.catalog; if provided here, this is copied over to the output directory for downstream analysis.
maxK	The maximum number of signatures to consider extracting.
tol	Controls when SignatureAnalyzer will terminate its search; tol was 1.e-05 for the PCAWG7 analysis.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog.
delete.tmp.files	If TRUE delete the many temporary files generated by SignatureAnalyzer.
overwrite	If TRUE, overwrite existing output

Details

Creates several files in out.dir:

1. sa.output.sigs.csv Normalized signatures (no all-0 signatures, column sums all 0)
2. sa.output.raw.exp.csv Raw exposures (attributions)
3. sa.output.exp.csv Same as sa.output.raw.exp.csv
4. sa.output.other.data.csv, contains a summary of important information, including the number of signatures extracted.
5. input.syn.exp.csv Optional, a copy of input.exposures, if it was provided.

Value

A list with the following elements:

1. signatures.W The raw signature matrix, *including* columns of all zeros.
2. exposures.H The raw exposure matrix, *excluding* rows of all zeros. The matrix product of the non-zero columns of signatures.w and exposures.H approximates the input spectrum matrix.
3. likelihood The likelihood as returned by SignatureAnalyzer.

4. `evidence` -1 * the posterior probability as returned by SignatureAnalyzer.
5. `relevance` One for each column of the `signatures.W`, as returned by SignatureAnalyzer.
6. `error` A measure of reconstruction error (?) as returned by SignatureAnalyzer
7. `normalized.sigs` The non-0 columns of `signatures.W` normalized so that each column sum is 1.

`RunSignatureEstimationQPAttributeOnly` *Run SignatureEstimation Quadratic Programming (QP) attribution on a spectra catalog file and known signatures.*

Description

Run SignatureEstimation Quadratic Programming (QP) attribution on a spectra catalog file and known signatures.

Usage

```
RunSignatureEstimationQPAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  seedNumber = 1,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

`input.catalog`	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
`gt.sigs.file`	File containing input mutational signatures. Columns are signatures, rows are mutation types.
`out.dir`	Directory that will be created for the output; abort if it already exits. Log files will be in `paste0(out.dir,"/tmp")`.
`seedNumber`	Specify the pseudo-random seed number used to run SignatureEstimation. Setting seed can make the attribution of SignatureEstimation repeatable. Default: 1.
`test.only`	If TRUE, only analyze the first 10 columns read in from `input.catalog`. Default: FALSE
`overwrite`	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in `paste0(out.dir,"/sa.output.rdata")`. These are TODO(Steve): list the files

Value

The inferred exposure of `SignatureEstimation`, invisibly.

RunSignatureEstimationSAAttributeOnly
Run SignatureEstimation Simulated Annealing (SA) attribution on a spectra catalog file and known signatures.

Description

Run SignatureEstimation Simulated Annealing (SA) attribution on a spectra catalog file and known signatures.

Usage

```
RunSignatureEstimationSAAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  seedNumber = 1,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
gt.sigs.file	File containing input mutational signatures. Columns are signatures, rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
seedNumber	Specify the pseudo-random seed number used to run SignatureEstimation. Setting seed can make the attribution of SignatureEstimation repeatable. Default: 1.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in paste0(out.dir,"/sa.output.rdata"). These are TODO(Steve): list the files

Value

The inferred exposure of SignatureEstimation, invisibly.

RunsigneR *Run signeR extraction and attribution on a spectra catalog file*

Description

Run signeR extraction and attribution on a spectra catalog file

Usage

```
RunsigneR(
  input.catalog,
  out.dir,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
seedNumber	Specify the pseudo-random seed number used to run signeR. Setting seed can make the attribution of signeR repeatable. Default: 1.
K.exact, K.range	K.exact is the exact value for the number of signatures active in spectra (K). Specify K.exact if you know exactly how many signatures are active in the input.catalog, which is the ICAMS-formatted spectra file. K.range is A numeric vector (K.min,K.max) of length 2 which tell signeR to search the best signature number active in spectra, K, in this range of Ks. Specify K.range if you don't know how many signatures are active in the input.catalog. WARNING: You must specify only one of K.exact or K.range! Default: NULL
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in out.dir. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The inferred exposure of signeR, invisibly.

`RunSomaticSignatures` *Run SomaticSignatures extraction and attribution on a spectra catalog file*

Description

Run SomaticSignatures extraction and attribution on a spectra catalog file

Usage

```
RunSomaticSignatures(
  input.catalog,
  out.dir,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

`input.catalog`	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
`out.dir`	Directory that will be created for the output; abort if it already exits. Log files will be in `paste0(out.dir,"/tmp")`.
`seedNumber`	Specify the pseudo-random seed number used to run SomaticSignatures. Setting seed can make the attribution of SomaticSignatures repeatable. Default: 1.
`K.exact, K.range`	`K.exact` is the exact value for the number of signatures active in spectra (K). Specify `K.exact` if you know exactly how many signatures are active in the `input.catalog`, which is the `ICAMS`-formatted spectra file. `K.range` is A numeric vector `(K.min,K.max)` of length 2 which tell SomaticSignatures to search the best signature number active in spectra, K, in this range of Ks. Specify `K.range` if you don't know how many signatures are active in the `input.catalog`. WARNING: You must specify only one of `K.exact` or `K.range`! Default: NULL
`test.only`	If TRUE, only analyze the first 10 columns read in from `input.catalog`. Default: FALSE
`overwrite`	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in `out.dir`. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The attributed exposure of `SomaticSignatures`, invisibly.

`RunSparseSignatures` *Run SparseSignatures extraction and attribution on a spectra catalog file*

Description

Run SparseSignatures extraction and attribution on a spectra catalog file

Usage

```
RunSparseSignatures(
  input.catalog,
  out.dir,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

`input.catalog`	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
`out.dir`	Directory that will be created for the output; abort if it already exits. Log files will be in `paste0(out.dir,"/tmp")`.
`seedNumber`	Specify the pseudo-random seed number used to run SparseSignatures. Setting seed can make the attribution of SparseSignatures repeatable. Default: 1.
`K.exact, K.range`	`K.exact` is the exact value for the number of signatures active in spectra (K). Specify `K.exact` if you know exactly how many signatures are active in the `input.catalog`, which is the `ICAMS`-formatted spectra file. `K.range` is A numeric vector (`K.min,K.max`) of length 2 which tell SparseSignatures to search the best signature number active in spectra, K, in this range of Ks. Specify `K.range` if you don't know how many signatures are active in the `input.catalog`. WARNING: You must specify only one of `K.exact` or `K.range`! Default: NULL
`test.only`	If TRUE, only analyze the first 10 columns read in from `input.catalog`. Default: FALSE
`overwrite`	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in `out.dir`. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The inferred exposure of `SparseSignatures`, invisibly.

Runtcsm *Run tcsm extraction and attribution on a spectra catalog file*

Description

WARNING: tcsm can only do exposure attribution using SBS96 spectra catalog and signature catalog!

Usage

```
Runtcsm(
  input.catalog,
  out.dir,
  CPU.cores = NULL,
  seedNumber = 1,
  K.exact = NULL,
  K.range = NULL,
  covariates = NULL,
  test.only = FALSE,
  overwrite = FALSE,
  feature.file = NULL,
  effect.output.file = NULL,
  sigma.output.file = NULL,
  gamma.output.file = NULL
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
CPU.cores	Number of CPUs to use in running tcsm. For a server, 30 cores would be a good choice; while for a PC, you may only choose 2-4 cores. By default (CPU.cores = NULL), the CPU.cores would be equal to (parallel::detectCores())/2, total number of CPUs divided by 2.
seedNumber	Specify the pseudo-random seed number used to run tcsm. Setting seed can make the attribution of tcsm repeatable.
K.exact, K.range	K.exact is the exact value for the number of signatures active in spectra (K). Specify K.exact if you know exact how many signatures are active in the input.catalog, which is the ICAMS-formatted spectra file. K.range is A numeric vector (K.min,K.max) of length 2 which tell tcsm to search the best signature number active in spectra, K, in this range of Ks. Specify K.range if you don't know how many signatures are active in the input.catalog. K.max - K.min >= 3, otherwise an error will be thrown. WARNING: You must specify only one of K or K.range!
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog.
overwrite	If TRUE, overwrite existing output.

model	Algorithm to be used to extract signatures and attribute exposures. Only "nmf" or "emu" is valid. Default: "nmf".

Details

Creates several files in out.dir. These are: TODO(Steve): list the files

TODO(Wuyang)

Value

The inferred exposure of tcsm, invisibly.

RunYAPSAAttributeOnly *Run YAPSA attribution on a spectra catalog file and known signatures.*

Description

Run YAPSA attribution on a spectra catalog file and known signatures.

Usage

```
RunYAPSAAttributeOnly(
  input.catalog,
  gt.sigs.file,
  out.dir,
  seedNumber = 1,
  signature.cutoff = NULL,
  test.only = FALSE,
  overwrite = FALSE
)
```

Arguments

input.catalog	File containing input spectra catalog. Columns are samples (tumors), rows are mutation types.
gt.sigs.file	File containing input mutational signatures. Columns are signatures, rows are mutation types.
out.dir	Directory that will be created for the output; abort if it already exits. Log files will be in paste0(out.dir,"/tmp").
seedNumber	Specify the pseudo-random seed number used to run YAPSA. Setting seed can make the attribution of YAPSA repeatable. Default: 1.
signature.cutoff	A numeric vector of values less than 1. Signatures from within W with an overall exposure less than the respective value in in_cutoff_vector will be discarded. Default: vector length of number of sigs with all zeros
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog. Default: FALSE
overwrite	If TRUE, overwrite existing output. Default: FALSE

Details

Creates several files in `paste0(out.dir,"/sa.output.rdata")`. These are TODO(Steve): list the files

Value

The inferred exposure of `YAPSA`, invisibly.

`SAMultiRunOneCatalog` *Run SignatureAnalyzer many times on one catalog and put results in specified location.*

Description

Run SignatureAnalyzer many times on one catalog and put results in specified location.

Usage

```
SAMultiRunOneCatalog(
  num.runs,
  signatureanalyzer.code.dir,
  input.catalog,
  out.dir,
  maxK = 30,
  tol = 1e-07,
  test.only = FALSE,
  delete.tmp.files = TRUE,
  overwrite = FALSE,
  mc.cores = 1,
  verbose = FALSE,
  seed = NULL
)
```

Arguments

`num.runs`	The number of times run SignatureAnalyzer on each catalog (matrix of mutational spectra).
`signatureanalyzer.code.dir`	The directory holding the SignatureAnalyzer code.
`input.catalog`	The catalog to analyze.
`out.dir`	Root of directory tree that will contain the results.
`maxK`	The maximum number of signatures to consider extracting.
`tol`	Controls when SignatureAnalyzer will terminate its search; `tol` was 1.e-05 for the PCAWG7 analysis.
`test.only`	If TRUE, only analyze the first 10 columns read in from `input.catalog`.
`delete.tmp.files`	If TRUE delete the many temporary files generated by SignatureAnalyzer.
`overwrite`	If TRUE overwrite previous results in same directory tree.

mc.cores	Number of cores to use for mclapply; ignored on Windows.
verbose	If TRUE cat a message regarding progress.
seed	If not NULL call RNGkind(kind = "L'Ecuyer-CMRG"); set.seed(seed).

SignatureAnalyzer4MatchedCatalogs

Run SignatureAnalyzer on 4 coordinated data sets and put results in specified location.

Description

Run SignatureAnalyzer on 4 coordinated data sets and put results in specified location.

Usage

```
SignatureAnalyzer4MatchedCatalogs(
  num.runs = 20,
  signatureanalyzer.code.dir,
  dir.root,
  maxK = 30,
  tol = 1e-07,
  test.only = FALSE,
  delete.tmp.files = TRUE,
  slice = 1:4,
  overwrite = FALSE,
  mc.cores = 1
)
```

Arguments

num.runs	Number of SignatureAnalyzer runs per data set.
signatureanalyzer.code.dir	The directory holding the SignatureAnalyzer code.
dir.root	Root of directory tree that contains the input data and to which the results will be written.
maxK	The maximum number of signatures to consider extracting.
tol	Controls when SignatureAnalyzer will terminate its search; tol was 1.e-05 for the PCAWG7 analysis.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog.
delete.tmp.files	If TRUE delete the many temporary files generated by SignatureAnalyzer.
slice	Vector of integers from 1:4. Only run on the corresponding data set (see Details).
overwrite	if TRUE overwrite preexisting results.
mc.cores	The number of cores to use with mclapply; automatically overridden to 1 on Windows.

Details

The 4 coordinated data sets are

1. `sa.sa.96`
2. `sp.sp`
3. `sa.sa.COMPOSITE`
4. `sp.sa.COMPOSITE`

which are described elsewhere.

`SignatureAnalyzerOneRun` *Source SignatureAnalyzer and run it once on a single data set and put results in specified location.*

Description

Source SignatureAnalyzer and run it once on a single data set and put results in specified location.

Usage

```
SignatureAnalyzerOneRun(
  signatureanalyzer.code.dir,
  input.catalog,
  out.dir,
  seedNumber = NULL,
  input.exposures = NULL,
  maxK = 30,
  tol = 1e-07,
  test.only = FALSE,
  delete.tmp.files = TRUE,
  verbose = 0,
  overwrite = FALSE
)
```

Arguments

`signatureanalyzer.code.dir`
: The directory holding the SignatureAnalyzer code.

`input.catalog`
: File containing input catalog. Columns are samples (tumors), rows are signatures. SignatureAnalyzer does not care about the row names (I think) TODO(Steve): check this.

`out.dir`
: Directory that will be created for the output; abort if it already exits. Log files will be in `paste0(out.dir,"/tmp")`.

`seedNumber`
: Specify the pseudo-random seed number used to run SignatureAnalyzer. Setting seed can make the attribution of SignatureAnalyzer repeatable. If NULL, this function will not specify seed number. Default: NULL.

`input.exposures`
: A file with the synthetic exposures used to generate `input.catalog`; if provided here, this is copied over to the output directory for downstream analysis.

maxK	The maximum number of signatures to consider extracting.
tol	Controls when SignatureAnalyzer will terminate its search; tol was 1.e-05 for the PCAWG7 analysis.
test.only	If TRUE, only analyze the first 10 columns read in from input.catalog.
delete.tmp.files	If TRUE delete the many temporary files generated by SignatureAnalyzer.
verbose	If TRUE, then print various messages.
overwrite	If TRUE, overwrite existing output

Details

Creates several files in out.dir:

1. sa.output.sigs.csv Normalized signatures (no all-0 signatures, column sums all 0)
2. sa.output.raw.exp.csv Raw exposures (attributions)
3. sa.output.exp.csv Same as sa.output.raw.exp.csv
4. sa.output.other.data.csv, contains a summary of important information, including the number of signatures extracted.
5. input.syn.exp.csv Optional, a copy of input.exposures, if it was provided.

Value

A list with the following elements:

1. signatures.W The raw signature matrix, *including* columns of all zeros.
2. exposures.H The raw exposure matrix, *excluding* rows of all zeros. The matrix product of the non-zero columns of signatures.w and exposures.H approximates the input spectrum matrix.
3. likelihood The likelihood as returned by SignatureAnalyzer.
4. evidence -1 * the posterior probability as returned by SignatureAnalyzer.
5. relevance One for each column of the signatures.W, as returned by SignatureAnalyzer.
6. error A measure of reconstruction error (?) as returned by SignatureAnalyzer
7. normalized.sigs The non-0 columns of signatures.W normalized so that each column sum is 1.

SignatureAnalyzerPrepHyper1Secondary

Prepare the "hypermutated" segment (a.k.a "Secondary" segment of a split non-hyper and hyper data set.)

Description

Prepare the "hypermutated" segment (a.k.a "Secondary" segment of a split non-hyper and hyper data set.)

Usage

```
SignatureAnalyzerPrepHyper1Secondary(
  non.hyper.results,
  primary.catalog,
  hyper.catalog,
  secondary.catalog,
  read.fn,
  write.fn,
  overwrite = TRUE
)
```

Arguments

non.hyper.results	The directory containing the the results of the analysis of the non-hyper-mutated (a.k.a "PRIMARY") mutational spectra.
primary.catalog	The catalog of non-hyper-mutated mutational spectra from which the results in non.hyper.results were derived.
hyper.catalog	The catalog of hyper-mutated mutational spectra which will be part of the input for the secondary analysis.
secondary.catalog	The final output catalog on which the secondary analysis will be performed; this is a cbind of pseudo-spectra generated from the PRIMARY signatures with the hyper.catalog.
read.fn	Function to use for reading signatures.
write.fn	Function to use for writing signatures.
overwrite	If TRUE overwrite possible previously computed files and/or directories.

SignatureAnalyzerPrepHyper4

Prepare the "hypermutated" segment (a.k.a "Secondary" segment of a split non-hyper and hyper data set.)

Description

Prepare the "hypermutated" segment (a.k.a "Secondary" segment of a split non-hyper and hyper data set.)

Usage

```
SignatureAnalyzerPrepHyper4(parent.dir, overwrite = FALSE)
```

Arguments

parent.dir	A directory that must contain subdirectories syn.SA.hyper.low and syn.SA.hyper.mixed. syn.SA.hyper.low must contain the synthetic non-hypermutated data and the results of running SignatureAnalyzer on the non-hyper segment, with subdirectories sa.sa.96, sa.sa.COMPOSITE, sp.sa.COMPOSITE, and sp.sp. syn.SA.hyper.mixed

must contain the synthetic hypermutated data. The results of the initial SignatureAnalyzer run will be placed here to prepare this directory for the second SignatureAnalyzer run.

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

`SignatureAnalyzerSummarizeSBS1SBS5`

Summarize all subdirectories of Signatureanalyzer results on the correlated SBS1 / SBS5.

Description

This is special-purpose function to summarize results from one in-silico experiment that examines how well signatures can be extracted from synthetic tumors with correlated SBS1 and SBS5.

Usage

```
SignatureAnalyzerSummarizeSBS1SBS5(top.level.dir, overwrite = FALSE)
```

Arguments

`top.level.dir`	Path to top level directory.
`overwrite`	If TRUE overwrite existing directories and files.

`SignatureAnalyzerSummarizeTopLevel`

Summarize all subdirectories of SignatureAnalyzer results on a major dataset.

Description

This function depends on a particular directory structure: see argument `top.level.dir`. This function finds the best of multiple SignatureAnalyzer extraction runs and summarizes the comparison of the best run with the ground truth.

Usage

```
SignatureAnalyzerSummarizeTopLevel(top.level.dir, overwrite = FALSE)
```

Arguments

`top.level.dir`	Path to top level directory, which must contain the following subdirectories: • `sa.sa.96/sa.results/` • `sp.sp/sa.results/` • `sa.sa.COMPOSITE/sa.results/` • `sp.sa.COMPOSITE/sa.results/` Each of the directories must contain additional subdirectories, one for each SignatureAnalyzer run, names `sa.run.<n>`, where `<n>` is an integer (string of digits).
`overwrite`	If TRUE overwrite existing summary files.

SourceSignatureAnalyzerCode

Source SignatureAnalyzer Codes.

Description

Source SignatureAnalyzer Codes.

Usage

```
SourceSignatureAnalyzerCode(signatureanalyzer.code.dir)
```

Arguments

signatureanalyzer.code.dir	The directory which stores SignatureAnalyzer program files. It must include a folder named INPUT_SignatureAnalyzer and a R script named SignatureAnalyer.PCAWG.functi

SummarizeMultiRuns

Assess/evaluate multiple summarized runs for one dataset from one software package.

Description

Summarize results from each software package in tool.dir/run.names (generated by running a software package), combine them into tool.dir.

Usage

```
SummarizeMultiRuns(datasetName, toolName, tool.dir, run.names)
```

Arguments

datasetName	Name of the dataset. (e.g. "S.0.1.Rsq.0.1"). Usually, it is has the same name as basename(top.dir).
toolName	Name of software package. (e.g. "sigproextractor")
tool.dir	Fourth level path from the top.dir. Expected to have multiple runs with different names (e.g. "seed.1") That is, top.dir/sp.sp/ExtrAttr/sa.results/. or top.dir/sa.sa.96/Attr/deconstructSigs.results/ Here, top.dir refers to a top-level directory which contains the full information of a synthetic dataset. (e.g. syn.2.7a.7b.abst.v8) This code depends on a conventional directory structure documented elsewhere. However there should be a directory within the tool.dir which stores the software output.
run.names	A character vector records the list of run.dir, or fifth level directories from the dataset top-level folder. E.g., c("seed.1","seed.691")

Details

Also writes multiple files into folder tool.dir:

Value

A list contain c(mean,sd) of multiple runs: Cosine similarity True Positives(TP): Ground-truth signatures which are active in the spectra, and extracted. False Negatives(FN): Ground-truth signatures not extracted. False Positives(FP): Signatures wrongly extracted, not resembling any ground-truth signatures. True Positive Rate (TPR, Sensitivity): TP / (TP + FN) False Discovery Rate (FDR): FP / (FP + TP)

SummarizeMultiToolsMultiDatasets

Combine results for multiple datasets, from different software packages.

Description

Summarize results from each software package in `third.level.dir/tool.dirnames` (generated by SummarizeMultiRuns), combine them into `third.level.dir`.

Usage

```
SummarizeMultiToolsMultiDatasets(
  dataset.dirs,
  second.third.level.dirname,
  out.dir,
  overwrite = FALSE
)
```

Arguments

`dataset.dirs`	Paths of top-level dataset directories trees you want to investigate. E.g. "./S.0.1.Rsq.0.1"
`second.third.level.dirname`	Name of the second.level.dir (e.g. "sp.sp") and the third.level.dir (e.g. "ExtrAttr") to be investigated. Examples are: "sp.sp/ExtrAttr", "sa.sa.96/Attr" Note: `multiTools.RDa` are expected to be exist under `dataset.dirs/second.third.level.dirnam`
`out.dir`	Path of the output directory.
`overwrite`	Whether to overwrite the contents in out.dir if it already exists. (Default: FALSE)

SummarizeMultiToolsOneDataset

Combine results for a single dataset, from different software packages.

Description

Summarize results from each software package in `third.level.dir/tool.dirnames` (generated by SummarizeMultiRuns), combine them into `third.level.dir`.

Usage

```
SummarizeMultiToolsOneDataset(
  third.level.dir,
  toolNames,
  tool.dirnames,
  datasetGroup,
  datasetGroupName,
  datasetSubGroup,
  datasetSubGroupName
)
```

Arguments

third.level.dir	Third level path distinguishing de-novo extraction + attribution packages from attribution-only packages. Examples: top.dir/sp.sp/ExtrAttr/ top.dir/sa.sa/Attr/
toolNames	Names of software package. (e.g. "sigproextractor")
tool.dirnames	Third level path from the top.dir. Expected to have summarized results generated by SummarizeMultiRuns. (multiRun.RDa, ManhattanDist.csv, meanSD.csv, meanSD.Manhattan.dist.csv) Examples: "signeR.results" (Under third.level.dir "ExtrAttr") "deconstructSigs.results" (Under third.level.dir "Attr") Here, top.dir refers to a top-level directory which contains the full information of a synthetic dataset. (e.g. syn.2.7a.7b.abst.v8) This code depends on a conventional directory structure documented elsewhere. However there should be a directory within the tool.names which stores the software output.
datasetGroup	Numeric or character vector specifying the group each dataset belong to. E.g. For SBS1-SBS5 correlated datasets, we can consider slope as the group: c("slope=0.1","slope=0.5","s Default: "Default"
datasetGroupName	Meaning or label of all datasetGroup. E.g. For SBS1-SBS5 correlated datasets, we can consider "SBS1:SBS5 mutation count ratio" as the label of the datasetGroup slope.
datasetSubGroup	Numeric or character vector differentiating datasets within each group. E.g. For SBS1-SBS5 correlated datasets, we can consider Pearson's R^2 as the subgroup: c("Rsq=0.1","Rsq=0.2","Rsq=0.3","Rsq=0.6") Default: Names of datasets, which are basename(dataset.dirs)
datasetSubGroupName	Meaning or label of all datasetSubGroup. E.g. For SBS1-SBS5 correlated datasets, we can consider "Pearson's R squared" as the label of the datasetSubGroup Pearson's R^2.

Value

A list contain c(mean,sd) of multiple runs: Cosine similarity True Positives(TP): Ground-truth signatures which are active in the spectra, and extracted. False Negatives(FN): Ground-truth signatures not extracted. False Positives(FP): Signatures wrongly extracted, not resembling any ground-truth signatures. True Positive Rate (TPR, Sensitivity): TP / (TP + FN) False Discovery Rate (FDR): FP / (FP + TP)

SummarizeOneToolMultiDatasets *Combine results for multiple datasets, from one software packages.*

Description

Summarize results from each software package in `third.level.dir/tool.dirnames` (generated by `SummarizeMultiRuns`), combine them into `third.level.dir`.

Usage

```
SummarizeOneToolMultiDatasets(
  dataset.dirs,
  datasetGroup = NULL,
  datasetGroupName,
  datasetSubGroup = NULL,
  datasetSubGroupName,
  toolName,
  tool.dirname,
  out.dir,
  overwrite = FALSE
)
```

Arguments

`dataset.dirs`	Paths of top-level dataset directories trees you want to investigate. E.g. "./S.0.1.Rsq.0.1"
`datasetGroup`	Numeric or character vector specifying the group each dataset belong to. E.g. For SBS1-SBS5 correlated datasets, we can consider slope (SBS1:SBS5 count ratio) as the group: `c(0.1,0.5,1,2,5,10)` Default: "Default"
`datasetGroupName`	Meaning or label of all datasetGroup. E.g. For SBS1-SBS5 correlated datasets, we can consider `"SBS1:SBS5 mutation count ratio"` as the label of the `datasetGroup` slope.
`datasetSubGroup`	Numeric or character vector differentiating datasets within each group. E.g. For SBS1-SBS5 correlated datasets, we can consider Pearson's R^2 as the subgroup: c(0.1,0.2,0.3,0.6) Default: Names of datasets, which are `basename(dataset.dirs)`
`datasetSubGroupName`	Meaning or label of all datasetSubGroup. E.g. For SBS1-SBS5 correlated datasets, we can consider `"Pearson's R squared"` as the label of the `datasetSubGroup` Pearson's R^2.
`toolName`	Name of software package to be investigated (e.g. "sigproextractor")
`tool.dirname`	Name of the second.level.dir (e.g. "sp.sp"), third.level.dir (e.g. "ExtrAttr") and tool.dir (e.g. "sigproextractor.results") to be investigated. One example: "sp.sp/ExtrAttr/sigproextractor.results" Note: this function expects the summary generated by `SummarizeSigOneSubdir` under `dataset.dirs/tool.dirname`
`out.dir`	Path of the output directory.
`overwrite`	Whether to overwrite the contents in out.dir if it already exists. (Default: FALSE)

SummarizeSigOneAttrSubdir

Assess/evaluate results from packages which can ONLY do exposure attribution.

Description

Packages including but not limited to: deconstructSigs, YAPSA.

Usage

```
SummarizeSigOneAttrSubdir(
  run.dir,
  ground.truth.exposure.dir = paste0(run.dir, "/../../../"),
  overwrite = FALSE
)
```

Arguments

run.dir	Lowest level path to results, e.g. <top.dir>/sa.sa.96/Attr/YAPSA.results/seed.1/ Here, <top.dir> refers to a top-level directory which contains the full information of a synthetic dataset. (e.g. syn.2.7a.7b.abst.v8) This code depends on a conventional directory structure documented elsewhere. For packages which can do both extraction and attribution, we expect two files, ground.truth.signatures.csv and attributed.exposures.csv are in the folder.
ground.truth.exposure.dir	Folder which stores ground-truth exposures. It defaults to be sub.dir, i.e. run.dir/../../
overwrite	If TRUE overwrite existing directories and files.

Details

Here, we excluded SignatureEstimation. Although it is also a package with only attribution, but it has two attribution algorithms. Therefore the naming of the results are slightly different from the other two packages.

SummarizeSigOneExtrAttrSubdir

Assess/evaluate results from packages which can do BOTH extraction and attribution, excluding SigProfiler-Python and SignatureAnalyzer.

Description

Packages including but not limited to: HDP, MutationalPatterns, sigfit, SigneR, SomaticSignatures.

Usage

```
SummarizeSigOneExtrAttrSubdir(
  run.dir,
  ground.truth.exposure.dir = paste0(run.dir, "/../../../"),
  overwrite = FALSE
)
```

Arguments

run.dir	Lowest level path to result of a run. E.g. <top.dir>/sa.sa.96/ExtrAttr/SomaticSignatures.res Here, <top.dir> refers to a top-level directory which contains the full information of a synthetic dataset. (e.g. syn.2.7a.7b.abst.v8) This code depends on a conventional directory structure documented elsewhere. For packages which can do both extraction and attribution, we expect two files, extracted.signatures.csv and attributed.exposures.csv are in the folder.
ground.truth.exposure.dir	Folder which stores ground-truth exposures. It defaults to be sub.dir, i.e. run.dir/../../../
overwrite	If TRUE overwrite existing directories and files.

SummarizeSigOnehelmsmanSubdir

Assess/evaluate results from SigProfiler-python (a.k.a. sigproextractor) Assessment is restricted to v0.0.5.43, because different version has different folder structure.

Description

Assess/evaluate results from SigProfiler-python (a.k.a. sigproextractor) Assessment is restricted to v0.0.5.43, because different version has different folder structure.

Usage

```
SummarizeSigOnehelmsmanSubdir(
  run.dir,
  ground.truth.exposure.dir = paste0(run.dir, "/../../../"),
  overwrite = FALSE,
  hierarchy = FALSE
)
```

Arguments

run.dir	Lowest level path to results, e.g. <top.dir>/sa.sa.96/ExtrAttr/sigproextractor.results/see Here, <top.dir> refers to a top-level directory which contains the full information of a synthetic dataset. (e.g. syn.2.7a.7b.abst.v8) This code depends on a conventional directory structure documented elsewhere. However there should be a directory <run.dir>/SBS96 which stores SigProfiler results.
ground.truth.exposure.dir	Folder which stores ground-truth exposures. Usually, it refers to sub.dir, i.e. run.dir/../../../

overwrite — If TRUE overwrite existing directories and files.

hierarchy — Whether the user have enabled hierarchy = True when running sigproextractor. specifying True or False into sigproextractor will cause the program to generate different folder structure. (Default: FALSE)

SummarizeSigOneSigProExtractorSubdir

Assess/evaluate results from SigProExtractor SigProFiler-python de novo extraction and attribution package. Assessment is restricted to v0.0.5.43+, because different version has different folder structure.

Description

Assess/evaluate results from SigProExtractor SigProFiler-python de novo extraction and attribution package. Assessment is restricted to v0.0.5.43+, because different version has different folder structure.

Usage

```
SummarizeSigOneSigProExtractorSubdir(
  run.dir,
  ground.truth.exposure.dir = paste0(run.dir, "/../../../"),
  overwrite = FALSE,
  hierarchy = FALSE
)
```

Arguments

run.dir — Lowest level path to results, e.g. <top.dir>/sa.sa.96/ExtrAttr/sigproextractor.results/see
Here, <top.dir> refers to a top-level directory which contains the full information of a synthetic dataset. (e.g. syn.2.7a.7b.abst.v8) This code depends on a conventional directory structure documented elsewhere. However there should be a directory <run.dir>/SBS96 which stores SigProfiler results.

ground.truth.exposure.dir — TODO(Wu Yang): Fix this File name which stores ground-truth exposures; defaults to "ground.truth.syn.exposures.csv". This file can be found in the sub.dir, i.e. <run.dir>/../../../

overwrite — If TRUE overwrite existing directories and files.

hierarchy — Whether the user have enabled hierarchy = True when running sigproextractor. specifying True or False into sigproextractor will cause the program to generate different folder structure. (Default: FALSE)

SummarizeSigOneSigProSSSubdir *Assess/evaluate results from sigproSS (a.k.a. SigProfiler Python attribution package)*

Description

Assess/evaluate results from sigproSS (a.k.a. SigProfiler Python attribution package)

Usage

```
SummarizeSigOneSigProSSSubdir(
  run.dir,
  ground.truth.exposure.dir = paste0(run.dir, "/../../../"),
  overwrite = FALSE
)
```

Arguments

run.dir	Lowest level path to results, e.g. <top.dir>/sa.sa.96/ExtrAttr/sigproextractor.results/see Here, <top.dir> refers to a top-level directory which contains the full information of a synthetic dataset. (e.g. syn.2.7a.7b.abst.v8) This code depends on a conventional directory structure documented elsewhere. However there should be a directory <run.dir>/SBS96 which stores SigProfiler results.
ground.truth.exposure.dir	TODO(Wu Yang): Fix this File name which stores ground-truth exposures; defaults to "ground.truth.syn.exposures.csv". This file can be found in the sub.dir, i.e. <run.dir>/../../../
overwrite	If TRUE overwrite existing directories and files.

SummarizeSigProExtractor *Summarize SigProfiler results in the sa.sa.96 and/or sp.sp subdirectories.*

Description

Summarize SigProfiler results in the sa.sa.96 and/or sp.sp subdirectories.

Usage

```
SummarizeSigProExtractor(
  top.dir,
  sub.dir = c("sa.sa.96", "sp.sp"),
  overwrite = FALSE
)
```

Arguments

top.dir	The top directory of a conventional data structure containing at least one of the subdirectories: sa.sa.96/sp.results and sp.sp/sp.results; see further documentation elsewhere.
sub.dir	The subdirectory under top.dir, and containing a folder named sp.results. By default, it contains both c("sa.sa","sp.sp"). But you should specify sub.dir = "sp.sp" for top.dir with only the sp.sp subdirectory (as is the case for the correlated SBS1-and-SBS5-containing data sets).
overwrite	whether to overwrite the existing run.dir/summary folder? If chosen to be FALSE and there is an existing summary folder, an error will be raised.

Details

Results are put in standardized subdirectories of top.dir.

SynSigRun *SynSigRun: An easy-to-use package for non-experts which runs software packages reproducibly with synthetic tumors generated by* SynSigGen.

Description

SynSigRun gives necessary information to mutational-signature analysis programs. These programs used catalogs of synthetic mutational spectra created by package SynSigGen, and results were assessed by SynSigEval.

Overview

The main focus is generating synthetic catalogs of mutational spectra (mutations in tumors) based on known mutational signature profiles and attributions (assignment of exposures to 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 catalogs 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.

Typical workflow for generating catalogs of "reality-based" synthetic mutational spectra is as follows.

```
In \code{SynSigGen}:

Input (based on SignatureAnalyzer or SigProfiler analysis of PCAWG tumors)
  A, matrix of attributions (signatures x samples)
  S, mutational signature profiles (mutation type x signature)

P <- GetSynSigParamsFromExposures(A, ...)

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

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

```
In \code{SynSigEval}:

T <- Signatures extracted by SignatureAnalzer or SigProfiler on synthetic.spectra

SummarizeResults(T, S, synthetic.exposures, ...)
```

(In SynSigGen) Creating Synthetic Mutational Catalogs

These functions create synthetic mutational catalogs based on parameters derived from signature profiles and attributions (exposures).

(In SynSigEval) Summarize results (of signature extraction)

Relevant functions are:

1. SummarizeSigProExtractor
2. SignatureAnalyzerSummarizeTopLevel
3. SignatureAnalyzerSummarizeSBS1SBS5

Comparing two sets of mutational signatures

Functions for comparing mutational signatures and sets of mutational signatures. Often we will be interested in comparing signature profiles extracted from synthetic data to the ground-truth signature profiles.

Match1Sig, MatchSigs1Direction, MatchSigs2Directions, MatchSigsAndRelabel

Folder structure for SynSigEval v0.2

Summary function will fit to the new 5-level folder structure:

First Level - `top.level.dir`: dataset folder (e.g. "S.0.1.Rsq.0.1", "syn.prancreas"). All spectra datasets under any top.level.dir have the same exposure.

Second Level - `ground.truth.exposure.dir`: spectra folder: (e.g. "sp.sp", "sa.sa.96"). All spectra datasets under any second.level.dir have the same signature and the same exposure counts.

Third Level - `third.level.dir`: It can be ("Attr") for storing results of packages which can only do exposure attribution of known signatures ("Attr"); it can also be ("ExtrAttr"), folder to store results of software packages which can do de-novo extraction and following attribution.

Fourth Level - `tool.dir`: The results of a software package (e.g. "sigproextractor.results","SignatureEstimation.QP.result
Under this level, `tool.dir` may contain multiple `run.dir`, each is a run of the software package using a specific number of seed.

Fifth level - `run.dir`: contains results from a run of the software package using a specific number of seed. (e.g. "seed.1")

Index