Package 'ICAMS'

January 23, 2019

Type Package

Title In-depth Characterization and Analysis of Mutational Signatures

Version 0.0.0.9001

Author Steve Rozen, Nanhai Jiang, Arnoud Boot

Maintainer Steve Rozen <steverozen@gmail.com>

Description This package has functions to read in VCF files from Strelka and GATK, create SNS, DNS, ID catalogs and do different types of plotting.

License GPL-3

Encoding UTF-8

LazyData true

Imports Biostrings (>= 2.50.2),
BSgenome (>= 1.50.0),
BSgenome.Hsapiens.1000genomes.hs37d5 (>= 0.99.1),
data.table,
dplyr,
GenomicRanges (>= 1.34.0),
graphics,
grDevices,
methods,
RColorBrewer,
RCurl,
stringr,
utils

Depends R (>= 3.5),

RoxygenNote 6.1.1

Suggests knitr,
rmarkdown,
testthat

VignetteBuilder knitr

Collate 'ICAMS.R'
'INDELS_related_functions.R'
'utility_functions.R'
'VCF_to_catalog_functions.R'
'plot_SNS_catalog.R'
'plot_DNS_catalog.R'
'plot_INDELS_catalog.R'

'read_write_catalog.R'
'test_functions.R'

R topics documented:

AddSequenceID . . . 2
CatalogToPdf . . . 3
CollapseCatalog . . . 4
CreateOneColIDCatalog . . . 5
FindDelMH . . . 6
FindMaxRepeatDel . . . 6
FindMaxRepeatIns . . . 7
GetStrelkaVAF . . . 8
ICAMS . . . 8
MakeVCFDNSdf . . . 9
PlotCatalog . . . 9
ReadCatalog . . . 11
ReadListOfStrelkaVCFs . . . 12
revc . . . 12
SplitSNSVCF . . . 13
TestMakeCatalogFromSNSVCFs . . . 13
TestSNSandDNSCat . . . 13
VCFFilesToCatalog . . . 14
VCFsToDNSCatalogs . . . 14
VCFsToSNSCatalogs . . . 15
WriteCatalog . . . 15

Index **17**

`AddSequenceID` *Add sequence context to a data frame with ID (insertion/deletion) mutation records*

Description

Add sequence context to a data frame with ID (insertion/deletion) mutation records

Usage

```
AddSequenceID(df, seq = BSgenome.Hsapiens.1000genomes.hs37d5)
```

Arguments

`df`	A data frame storing mutation records of a VCF file containing only insertions and deletions. IMPORTANT: The representation of indels in df must have been canonicalized, so that context bases (which are added by some indel callers) are placed in a column "Left.context.base" and so that, for deletions, ALT is the empty string, and, for insertions, REF is the empty string.
`seq`	A particular reference genome.

Value

A data frame with 2 new columns added to the input data frame. One column contains sequence context information and the other column contains the length of the "context" string to the left of the site of the variant.

`CatalogToPdf` *Catalog to Pdf Functions*

Description

Plot the mutation catalog of different samples to a PDF file

Usage

```
Cat96ToPdf(catalog, name, id = colnames(catalog), type = "density",
  grid = FALSE, upper = TRUE, xlabels = TRUE, abundance = NULL)

Cat192ToPdf(catalog, name, id = colnames(catalog), type = "counts",
  cex = 0.8, abundance = NULL)

Cat192StrandToPdf(catalog, name, id = colnames(catalog),
  type = "counts", cex = 1, abundance = NULL)

Cat1536ToPdf(catalog, name, id = colnames(catalog), abundance)

CatDNS78ToPdf(catalog, name, id = colnames(catalog), type = "density",
  abundance = NULL)

CatDNS144ToPdf(catalog, name, id = colnames(catalog), type = "counts",
  cex = 1, abundance = NULL)

CatIDToPdf(catalog, name, id = colnames(catalog), type = "counts")
```

Arguments

`catalog`	A matrix whose rownames indicate the mutation types while its columns contain the counts of each mutation type from different samples.
`name`	The name of the PDF file to be produced.
`id`	A vector containing the ID information of different samples.
`type`	A vector of values indicating the type of plot for each sample. If type = "counts", the graph will plot the occurrences of the mutation types in the sample. If type = "signature", the graph will plot mutation signatures of the sample. If type = "density", the graph will plot the rates of mutations per million nucleotides for each mutation type. (Please take note there is no "density" type for CatIDtoPdf function and the option of type = "density" is not implemented for function Cat192ToPdf, Cat192StrandToPdf and CatDNS144ToPdf at the current stage.)
`grid`	A logical value indicating whether to draw the grid lines in the graph.
`upper`	A logical value indicating whether to draw horizontal lines and names of major mutation class on top of graph.

xlabels	A logical value indicating whether to draw x axis labels.
abundance	A matrix containing nucleotide abundance information, to be used only when type = "density".
cex	A numerical value giving the amount by which mutation class labels, y axis labels, sample name and legend(if there exists) should be magnified relative to the default.

Details

`Cat96ToPdf` Plot the SNS 96 mutation catalog of different samples to a PDF file.

`Cat192ToPdf` Plot the SNS 192 mutation catalog of different samples to a PDF file.

`Cat192StrandToPdf` Plot the transcription strand bias graph of 6 SNS mutation types ("C>A", "C>G", "C>T", "T>A", "T>C", "T>G") of different samples to a PDF file.

`Cat1536ToPdf` Plot the 1536 mutation catalog of >= 1 samples to a PDF file. The mutation types are in six-letters like CATTAT, first 2-letters CA refers to (-2, -1) position, third letter T refers to the base which has mutation, next second 2-letters TA refers to (+1, +2) position, last letter T refers to the base after mutation.

`CatDNS78ToPdf` Plot the DNS 78 mutation catalog of different samples to a PDF file.

`CatDNS144ToPdf` Plot the transcription strand bias graph of 10 major DNS mutation types ("AC>NN", "AT>NN", "CC>NN", "CG>NN", "CT>NN", "GC>NN", "TA>NN", "TC>NN", "TG>NN", "TT>NN") of different samples to a PDF file.

`CatIDToPdf` Plot the insertion and deletion catalog of different samples to a PDF file. (Please take note that the deletions Repeat Size ranges from 0 to 5+ in the catalog, but for plotting and end user documentation it ranges from 1 to 6+.)

Value

invisible(TRUE)

`CollapseCatalog` *Collapse Catalog Functions*

Description

Collapse a catalog matrix to a canonical one

Usage

```
Collapse192To96(catalog)

Collapse1536To96(catalog)

Collapse144To78(catalog)
```

Arguments

catalog	A catalog matrix to be collapsed whose row names indicate the mutation types while its columns show the occurrences of each mutation type of different samples.

Details

`Collapse192To96` Collapse a SNS 192 catalog matrix to a SNS 96 catalog matrix.

`Collapse1536To96` Collapse a SNS 1536 catalog matrix to a SNS 96 catalog matrix.

`Collapse144To78` Collapse a DNS 144 catalog matrix to a DNS 78 catalog matrix.

Value

A canonical catalog matrix whose row names indicate the mutation types while its columns show the occurrences of each mutation type of different samples.

`CreateOneColIDCatalog` *Create an indel (ID) mutation catalog for *one* sample from a Variant Call Format (VCF) file*

Description

Create an indel (ID) mutation catalog for *one* sample from a Variant Call Format (VCF) file

Usage

```
CreateOneColIDCatalog(ID.vcf, SBS.vcf)
```

Arguments

`ID.vcf`	An in-memory VCF as a data.frame annotated by the AddSequence and AddTranscript functions. It must only contain indels and must *not* contain SBS (single base substituions), DBS, or triplet base substituions etc. * Sequence must already have been added to ID.vcf One design decision for variant callers is the representation of "complex indels", e.g. mutations e.g. CAT > GC. Some callers represent this as C>G, A>C, and T>_. Others might represent it as CAT > CG. Multiple issues can arise. In PCAWG, overlapping indel/SBS calls from different callers were included in the indel VCFs.
`SBS.vcf`	An in-memory VCF as a data frame. Because we have to work with some PCAWG data, we will look for neigboring indels and indels adjoining SBS. That means this functions takes an SBS VCF and an ID VCF from the same sample.

Value

A list with two elemsents: ID.cat: A 1-column matrix containing the mutation catalog information. problems: Locations of neighboring indels or indels neighboring SBS. In the future we might handle these depending on what we find in the indel calls from different variant callers. TODO(steve) Is problems implemented?

`FindDelMH` *FindDelMH TODO(steve):not finished*

Description

Microhomology can be alligned in multiple equivalent ways. Example:

Usage

```
FindDelMH(context, q, pos)
```

Arguments

`context`	TODO
`q`	TODO
`pos`	TODO

Details

GGCTAGTT aligned to

GGCTAGAACTAGTT GG——CTAGTT GGCTAGTT GG[CTAGAA]CTAGTT —- —- GGC——TAGTT GGCTAGTT GGC[TAGAAC]TAGTT * — * — GGCT——AGTT GGCTAGTT GGCTA——GTT GGCTAGTT GGCTAG——TT GGCTAGTT

All the same pairs of sequence, aligned 5 different ways. 4 bp of microhomology.

Need to find:

(1) The maxium match of undeleted sequence on left that is identical to the right end of deleted sequence, and

(2) The maxium match of undeleted sequence on right that is identical to the left end of deleted sequence.

The microhomology sequence is the concatenation of items (1) and (2).

Value

TODO

`FindMaxRepeatDel` *Return the number of repeat units in which a deletion is embedded. TODO(Steve): check this statement; what if there is no repeat?*

Description

e.g. rep.unit.seq = ac pos = 3 context = xyaczt pos ^ Return 0

Usage

```
FindMaxRepeatDel(context, rep.unit.seq, pos)
```

Arguments

context	A string that embeds rep.unit.seq at position pos
rep.unit.seq	A substring of context at pos to pos + nchar(rep.unit.seq) - 1, which is the repeat unit sequence.
pos	The position of rep.unit.seq.

Details

If substr(context, pos, pos + nchar(rep.unit.seq) - 1) != rep.unit.seq then stop.

Value

The number of repeat units in which rep.unit.seq is embedded, not include the input rep.unit.seq in the count.

FindMaxRepeatIns *FindMaxRepeatIns TODO(steve):finish this*

Description

If q is an insertion into context between pos and pos+1 if q is repeated in context it might start at pos+1:

Usage

```
FindMaxRepeatIns(context, q, pos)
```

Arguments

context	TODO
q	TODO
pos	TODO

Details

e.g. q = ac pos = 4 context = abxyac pos ^ start ^

or q might start at pos + 1 - len(q)

e.g. q = ac pos = 4 context = xyaczz pos ^ start ^

Value

TODO

`GetStrelkaVAF` *Extract the VAFs (variant allele frequencies) from a VAF created by Strelka version 1*

Description

Extract the VAFs (variant allele frequencies) from a VAF created by Strelka version 1

Usage

```
GetStrelkaVAF(strelka.vcf)
```

Arguments

`strelka.vcf`	said VCF as a data.frame

Value

A vector of VAFs, one for each row of strelka.vcf

`ICAMS` *ICAMS: In-depth Characterization and Analysis of Mutational Signatures*

Description

This package has functions to read in VCF files from Strelka and GATK, create SNS, DNS, ID catalogs and do different types of plotting.

Reading catalogs

Functions for reading a catalog in PCAWG7 format from path: `ReadCatalog`

Writing catalogs

Functions for writting a mutation catalog to a file on disk: `WriteCatalog`

Collapsing catalogs

Functions for collapsing a mutation catalog to a canonical one: `CollapseCatalog`

Plotting catalogs

Functions for plotting the mutation catalog of one sample: `PlotCatalog`

Functions for plotting mutation catalog of different samples to a PDF file: `CatalogToPdf`

`MakeVCFDNSdf` *MakeVCFDNSdf TODO(steve) add average VAF*

Description

Take DNS ranges and the original VCF and generate a VCF with dinucleotide REF and ALT alleles. The output VCF has minimal columns: just CHROM, POS, ID, REF, ALT.

Usage

```
MakeVCFDNSdf(DNS.range.df, SNS.vcf.dt)
```

Arguments

`DNS.range.df`	Data frame with columns CHROM, LOW, HIGH
`SNS.vcf.dt`	TODO

Value

TODO

`PlotCatalog` *Plot Catalog Functions*

Description

Plot the mutation catalog of one sample

Usage

```
PlotCat96(catalog, id, type = "density", cex = 0.8, grid = TRUE,
  upper = TRUE, xlabels = TRUE, abundance = NULL)

PlotCat192(catalog, id, type = "counts", cex = 0.8, abundance = NULL)

PlotCat192Strand(catalog, id, type = "counts", cex = 1,
  abundance = NULL)

PlotCat1536(catalog, id, abundance)

PlotCatDNS78(catalog, id, type = "density", abundance = NULL)

PlotCatDNS144(catalog, id, type = "counts", cex = 1,
  abundance = NULL)

PlotCatID(catalog, id, type = "counts")
```

Arguments

catalog	A matrix whose rownames indicate the mutation types while its columns contain the counts of each mutation type.
id	The ID information of the sample which has mutations.
type	A value indicating the type of graph. If type = "counts", the graph will plot the occurrences of the mutation types in the sample. If type = "signature", the graph will plot mutation signatures of the sample. If type = "density", the graph will plot the rates of mutations per million nucleotides for each mutation type. (Please take note there is no "density" type for PlotCatID function and the option of type = "density" is not implemented for function PlotCat192, PlotCat192Strand and PlotCatDNS144 at the current stage.)
cex	A numerical value giving the amount by which mutation class labels, mutation counts(if there exists), y axis and its labels, x axis labels and its annotations(if there exists) sample name and legend(if there exists) should be magnified relative to the default.
grid	A logical value indicating whether to draw the grid lines in the graph.
upper	A logical value indicating whether to draw horizontal lines and names of major mutation class on top of graph.
xlabels	A logical value indicating whether to draw x axis labels.
abundance	A matrix containing nucleotide abundance information and strand information(if there exists), to be used only when type = "density".

Details

`PlotCat96` Plot the SNS 96 mutation catalog of one sample.

`PlotCat192` Plot the SNS 192 mutation catalog of one sample.

`PlotCat192Strand` Plot the transcription strand bias graph of 6 SNS mutation types ("C>A", "C>G", "C>T", "T>A", "T>C", "T>G") in one sample.

`PlotCat1536` Plot the pentanucleotide sequence contexts for one sample, normalized by pentanucleotide occurrence in the genome. The mutation types are in six-letters like CATTAT, first 2-letters CA refers to (-2, -1) position, third letter T refers to the base which has mutation, next second 2-letters TA refers to (+1, +2) position, last letter T refers to the base after mutation.

`PlotCatDNS78` Plot the DNS 78 mutation catalog of one sample.

`PlotCatDNS144` Plot the transcription strand bias graph of 10 major DNS mutation types ("AC>NN", "AT>NN", "CC>NN", "CG>NN", "CT>NN", "GC>NN", "TA>NN", "TC>NN", "TG>NN", "TT>NN") in one sample.

`PlotCatID` Plot the insertion and deletion catalog of one sample. (Please take note that the deletions Repeat Size ranges from 0 to 5+ in the catalog, but for plotting and end user documentation it ranges from 1 to 6+.)

Value

invisible(TRUE)

ReadCatalog *Read Catalog Functions*

Description

Read a catalog in PCAWG7 format from path

Usage

```
ReadCat96(path, strict = TRUE)

ReadCat192(path, strict = TRUE)

ReadCat1536(path, strict = TRUE)

ReadCatDNS78(path, strict = TRUE)

ReadCatDNS144(path, strict = TRUE)

ReadCatQUAD136(path, strict = TRUE)

ReadCatID(path, strict = TRUE)
```

Arguments

path	Path to a catalog on disk in the "PCAWG7" format.
strict	If TRUE, do additional checks on the input, and stop if the checks fail.

Details

ReadCat96 Read a 96 SNS catalog from path

ReadCat192 Read a 192 SNS catalog from path

ReadCat1536 Read a 1536 SNS catalog from path

ReadCatDNS78 Read a 78 DNS catalog from path

ReadCatDNS144 Read a 144 DNS catalog from path

ReadCatQUAD136 Read a 136 QUAD catalog from path

ReadCatID Read a ID (insertion/deletion) catalog from path Please take note that the deletions Repeat Size ranges from 0 to 5+ in the catalog, but for plotting and end user documentation it ranges from 1 to 6+.

See also WriteCatalog

Value

A catalog in canonical in-memory format.

ReadListOfStrelkaVCFs *Read a list of Strelka VCF files from path*

Description

Read a list of Strelka VCF files from path

Usage

```
ReadListOfStrelkaVCFs(vector.of.file.paths)
```

Arguments

vector.of.file.paths	A vector containing the paths of the VCF files.

Value

A list of vcfs from vector.of.file.paths.

revc *Reverse complement every string in string.vec*

Description

Reverse complement every string in string.vec

Usage

```
revc(string.vec)
```

Arguments

string.vec	a vector of type character.

Value

A vector of type characters with the reverse complement of of every string in string.vec.

`SplitSNSVCF` *Split an in-memory VCF into SNS, DNS, and variants involving > 2 consecutive bases*

Description

SNSs are single nucleotide substitutions, eg C>T, A<G,.... DNSs are double nucleotide substitutions, eg CC>TT, AT>GG, ... Variants involving > 2 consecutive bases are rare, so this function just records them. These would be variants such ATG>CCT, AGAT > TCTA, ...

Usage

```
SplitSNSVCF(vcf.df, max.vaf.diff = 0.02)
```

Arguments

`vcf.df`	An in-memory data frame containing a VCF file contents.
`max.vaf.diff`	The maximum difference of VAF, default value is 0.02.

Value

A list of 3 in-memory objects with the elements:

`TestMakeCatalogFromSNSVCFs` *This function is to make catalogs from the sample VCF files to compare with the expected catalog information.*

Description

This function is to make catalogs from the sample VCF files to compare with the expected catalog information.

Usage

```
TestMakeCatalogFromSNSVCFs()
```

`TestSNSandDNSCat` *This function is to test whether the predefined functions are working correctly to produce the desired SNS and DNS catalogs.*

Description

This function is to test whether the predefined functions are working correctly to produce the desired SNS and DNS catalogs.

Usage

```
TestSNSandDNSCat()
```

VCFFilesToCatalog *Create SNS and DNS catalogs from VCF files*

Description

Create 3 SNS catalogs (96, 192, 1536) and 3 DNS catalogs (78, 136, 144) from the VCFs specified by vector.of.file.paths

Usage

```
VCFFilesToCatalog(vector.of.file.paths, genome, trans.ranges)
```

Arguments

vector.of.file.paths	A vector containing the paths of the VCF files.
genome	Name of a particular reference genome (without quotations marks).
trans.ranges	A data.table which contains transcript range and strand information.

Details

This function calls VCFsToSNSCatalogs and VCFsToDNSCatalogs

Value

A list of 3 SNS catalogs (one each for 96, 192, and 1536) and 3 DNS catalogs (one each for 78, 136, and 144)

VCFsToDNSCatalogs *Create DNS catalogs from VCFs*

Description

Create a list of 3 catalogs (one each for DNS78, DNS144 and QUAD136) out of the contents of the VCFs in list.of.vcfs

Usage

```
VCFsToDNSCatalogs(list.of.vcfs, genome, trans.ranges)
```

Arguments

list.of.vcfs	List vector of in-memory VCFs. The list names will be the sample ids in the output catalog.
genome	Name of a particular reference genome (without quotations marks).
trans.ranges	A data frame containing transcript ranges.

Value

A list of 3 catalogs, one each for DNS78, DNS144, QUAD136: catDNS78 catDNS144 catQUAD136

VCFsToSNSCatalogs *Create SNS catalogs from VCFs*

Description

Create a list of 3 catalogs (one each for 96, 192, 1536) out of the contents of the VCFs in list.of.vcfs

Usage

```
VCFsToSNSCatalogs(list.of.vcfs, genome, trans.ranges)
```

Arguments

list.of.vcfs	List vector of in-memory VCFs. The list names will be the sample ids in the output catalog.
genome	Name of a particular reference genome (without quotations marks).
trans.ranges	A data frame containing transcript ranges.

Value

A list of 3 catalogs, one each for 96, 192, 1536: cat96 cat192 cat1536

WriteCatalog *Write Catalog Functions*

Description

Write a mutation catalog to a file on disk

Usage

```
WriteCat96(ct, path, strict = TRUE)

WriteCat192(ct, path, strict = TRUE)

WriteCat1536(ct, path, strict = TRUE)

WriteCatDNS78(ct, path, strict = TRUE)

WriteCatDNS144(ct, path, strict = TRUE)

WriteCatQUAD136(ct, path, strict = TRUE)

WriteCatID(ct, path, strict = TRUE)
```

Arguments

ct	A matrix of mutation catalog.
path	The path of the file to be written on disk.
strict	If TRUE, do additional checks on the input, and stop if the checks fail.

Details

`WriteCat96` Write a SNS 96 mutation catalog to a file on disk

`WriteCat192` Write a SNS 192 mutation catalog to a file on disk

`WriteCat1536` Write a SNS 1536 mutation catalog to a file on disk

`WriteCatDNS78` Write a DNS 78 mutation catalog to a file on disk

`WriteCatDNS144` Write a DNS 144 mutation catalog to a file on disk

`WriteCatQUAD136` Write a 136 QUAD catalog from path

`WriteCatID` Write a ID (insertion/deletion) catalog to a file on disk Please take note that the deletions Repeat Size ranges from 0 to 5+ in the catalog, but for plotting and end user documentation it ranges from 1 to 6+.

See also `ReadCatalog`

Index

AddSequenceID, 2

Cat1536ToPdf *(CatalogToPdf)*, 3
Cat192StrandToPdf *(CatalogToPdf)*, 3
Cat192ToPdf *(CatalogToPdf)*, 3
Cat96ToPdf *(CatalogToPdf)*, 3
CatalogToPdf, 3, *8*
CatDNS144ToPdf *(CatalogToPdf)*, 3
CatDNS78ToPdf *(CatalogToPdf)*, 3
CatIDToPdf *(CatalogToPdf)*, 3
Collapse144To78 *(CollapseCatalog)*, 4
Collapse1536To96 *(CollapseCatalog)*, 4
Collapse192To96 *(CollapseCatalog)*, 4
CollapseCatalog, 4, *8*
CreateOneColIDCatalog, 5

FindDelMH, 6
FindMaxRepeatDel, 6
FindMaxRepeatIns, 7

GetStrelkaVAF, 8

ICAMS, 8
ICAMS-package *(ICAMS)*, 8

MakeVCFDNSdf, 9

PlotCat1536 *(PlotCatalog)*, 9
PlotCat192 *(PlotCatalog)*, 9
PlotCat192Strand *(PlotCatalog)*, 9
PlotCat96 *(PlotCatalog)*, 9
PlotCatalog, *8*, 9
PlotCatDNS144 *(PlotCatalog)*, 9
PlotCatDNS78 *(PlotCatalog)*, 9
PlotCatID *(PlotCatalog)*, 9

ReadCat1536 *(ReadCatalog)*, 11
ReadCat192 *(ReadCatalog)*, 11
ReadCat96 *(ReadCatalog)*, 11
ReadCatalog, *8*, 11, *16*
ReadCatDNS144 *(ReadCatalog)*, 11
ReadCatDNS78 *(ReadCatalog)*, 11
ReadCatID *(ReadCatalog)*, 11
ReadCatQUAD136 *(ReadCatalog)*, 11
ReadListOfStrelkaVCFs, 12

revc, 12

SplitSNSVCF, 13

TestMakeCatalogFromSNSVCFs, 13
TestSNSandDNSCat, 13

VCFFilesToCatalog, 14
VCFsToDNSCatalogs, *14*, 14
VCFsToSNSCatalogs, *14*, 15

WriteCat1536 *(WriteCatalog)*, 15
WriteCat192 *(WriteCatalog)*, 15
WriteCat96 *(WriteCatalog)*, 15
WriteCatalog, *8*, *11*, 15
WriteCatDNS144 *(WriteCatalog)*, 15
WriteCatDNS78 *(WriteCatalog)*, 15
WriteCatID *(WriteCatalog)*, 15
WriteCatQUAD136 *(WriteCatalog)*, 15