Package ‘scphaser’

July 2, 2016

Title Phase Variants Within Genes Using Allele Counts

Version 1.0.0

Description Phase variants within genes using allele counts from single-cell RNA-seq data.

URL `https://github.com/edsgard/scphaser`

BugReports `https://github.com/edsgard/scphaser/issues`

Depends R (>= 3.2.2)

License GPL-3

LazyData true

Imports cluster,
Hmisc,
R.utils,
BiocParallel

Suggests testthat,
knitr,
rmarkdown

VignetteBuilder knitr

RoxygenNote 5.0.1

R topics documented:

call_gt 2
filter_acset 3
filter_feat_nminvar 4
filter_homovars 5
filter_var_gt 6
filter_zerorow 7
marinov 8
mousehybrid 9
new_acset 9
phase 11
plot_conc 12
racset 13
set_gt_conc 14
subset_cols 15
subset_feat 16
subset_rows 17

Index 19

`call_gt` *Call transcribed genotypes*

Description

`call_gt` calls transcribed genotypes using allele RNA-seq read counts

Usage

```
call_gt(acset, min_acount = 3, fc = 3)
```

Arguments

`acset`	An acset list created by the function `new_acset`. The acset must contain a refcount and altcount matrix with allele counts.
`min_acount`	An integer specifying the minimum number of reads required to be present for at least one of the two alleles as to make a call. If not fulfilled the call is set to NA.
`fc`	An integer specifying the fold-change cutoff between the expression of the alternative and reference allele (fc.observed = alternative allele count / reference allele count). The transcribed genotype call is set to 0 if fc.observed <= fc, 2 if fc.observed >= fc and 1 otherwise (if there are enough reads present, see the "min_acount" parameter).

Details

This is a simplistic transcribed genotype caller which is used to discretize which allele is the most expressed. For each variant and cell one of four possible discrete values is set depending on the expression of the two alleles. 0: reference allele most highly expressed, 1: bi-allelic expression with similar degree of expression from the two alleles, 2: alternative allele most highly expressed, or, NA if there are not enough reads to make a call.

Value

acset An acset list where the "gt" element is set or updated. "gt" is a matrix with integer values representing transcribed genotype calls. 0: reference allele most highly expressed, 1: bi-allelic expression with similar degree of expression from the two alleles, 2: alternative allele most highly expressed. NA's are used to represent entries where no call could be made. The rownames are set to the variant column, "var", of featdata. The colnames are set to the colnames of "refcount". The elements [['args']][['filter']][c('min_acount', 'fc')] are added or updated to the acset as to record the filter arguments used.

Examples

```
##load dataset
invisible(marinov)
acset = new_acset(featdata = marinov[['featdata']], refcount =
marinov[['refcount']], altcount = marinov[['altcount']], phenodata =
marinov[['phenodata']])

##Call transcribed genotypes
```

```
min_acount = 3
fc = 3
acset = call_gt(acset, min_acount, fc)
```

`filter_acset` *Filter variants and features*

Description

`filter_acset` removes variants and features with too little data

Usage

```
filter_acset(acset, nmincells, nminvar = 2, feat_filter = FALSE)
```

Arguments

`acset`	An acset list created by `new_acset`. It must contain a "gt" element with transcribed genotype calls, see `call_gt`.
`nmincells`	An integer specifying the minimum number of cells with imbalanced allelic expression.
`nminvar`	An integer specifying the minimum number of variants within a feature.
`feat_filter`	Boolean specifying if filtering should be done per feature including an initial cell-filter removing cells with less than two variants with imbalanced expression within the feature.

Details

The function removes variants which have less than "nmincells" cells with imbalanced allelic expression. A cell is deemed to have imbalanced allelic expression if its transcribed genotype is set to 0 or 2, see `call_gt`. Subsequently it removes features with less than "nminvar" variants, see `filter_feat_nminvar`.

Value

acset An acset list.

Examples

```
##create a small artificial genotype matrix
ncells = 10
paternal = c(0, 2, 0, 0, 2)
maternal = c(2, 0, 2, 2, 0)
gt = as.matrix(as.data.frame(rep(list(paternal, maternal), ncells / 2)))
vars = 1:nrow(gt)
colnames(gt) = 1:ncells
rownames(gt) = vars

##create a feature annotation data-frame
nvars = nrow(gt)
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, gt = gt)

##Remove variants with imbalanced expression in less than nmincells cells and
##features with less than nminvar variants
nmincells = 5
nminvar = 2
acset_filt = filter_acset(acset, nmincells, nminvar)
```

`filter_feat_nminvar` *Filter features on minimal number of required variants*

Description

`filter_feat_nminvar` removes features with less than "nmin_var" variants.

Usage

```
filter_feat_nminvar(acset, nmin_var = 2)
```

Arguments

`acset`	An acset list created by the function `new_acset`.
`nmin_var`	An integer specifying the minimum number of variants within a feature

Details

The function takes an acset and an integer, "nmin_var", as input and filter out features that have less than "nmin_var" variants.

Value

acset An acset list. The element [['filter']][['nmin_var']] is added or updated to the acset to record the filter argument used.

Examples

```
##create a small artificial genotype matrix
ncells = 10
paternal = c(0, 2, 0, 0, 2)
maternal = c(2, 0, 2, 2, 0)
gt = as.matrix(as.data.frame(rep(list(paternal, maternal), ncells / 2)))
vars = 1:nrow(gt)
colnames(gt) = 1:ncells
rownames(gt) = vars

##create a feature annotation data-frame
nvars = nrow(gt)
```

```
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, gt = gt)

##Remove features with less than nminvar variants
nminvar = 2
acset = filter_feat_nminvar(acset, nminvar)
```

`filter_homovars` *Filter homozygous variants*

Description

`filter_homovars` removes variants for which a large proportion of cells with imbalanced expression is imbalanced towards the same allele

Usage

```
filter_homovars(acset, alpha = 0.1, mono_ase = 0.1)
```

Arguments

`acset`	An acset list which must contain "refcount" and "altcount" elements with allele counts, see `new_acset`.
`alpha`	A numeric specifying the significance level of the binomial test, where the test compares the number of cells expressing each allele in an imbalanced "monoallelic" manner.
`mono_ase`	A numeric between 0 and 1 specifying the allele specific expression level at which to deem an allele monoallelically expressed.

Details

The function removes variants which tend to express the same allele in a large proportion of cells, that is, the allelic expression is stable across cells rather than random with respect to which allele is the most highly expressed. The purpose of this filter is to reduce the number of variants that have falsely been called as heterozygous variants but are actually homozygous. Use with caution as a heterozygous variants can indeed have imbalanced expression towards the same allele in the majority of cells. A cell is deemed to have imbalanced allelic expression if its allele-specific expression, ase, is < mono_ase or >(1 - mono_ase), where ase = alternative allele count / (alternative allele count + reference allele count). If the number of cells expressing one allele in such an imbalanced manner is significantly greater than the number of cells expressing the other allele the variant is removed (binomial test).

Value

acset An acset list subsetted on variants that pass the filter.

Examples

```
##load dataset
invisible(marinov)
acset = new_acset(featdata = marinov[['featdata']], refcount =
marinov[['refcount']], altcount = marinov[['altcount']], phenodata =
marinov[['phenodata']])

##Remove variants having monoallelic expression of the same allele in a
##large proportion of cells
alpha = 0.1
mono_ase = 0.1
acset_filt = filter_homovars(acset, alpha, mono_ase)
```

---

filter_var_gt *Filter variants*

Description

filter_var_gt removes variants based on the number of cells with imbalanced expression

Usage

```
filter_var_gt(acset, nmincells = 3)
```

Arguments

acset	An acset list created by new_acset. It must contain a "gt" element with transcribed genotype calls, see call_gt.
nmincells	An integer specifying the minimum number of cells with imbalanced allelic expression.

Details

The function removes variants which have less than "nmincells" cells with imbalanced allelic expression. A cell is deemed to have imbalanced allelic expression if its transcribed genotype is set to 0 or 2, see call_gt.

Value

acset An acset list subsetted on variants that pass the filter.

Examples

```
##create a small artificial genotype matrix
ncells = 10
paternal = c(0, 2, 0, 0, 2)
maternal = c(2, 0, 2, 2, 0)
gt = as.matrix(as.data.frame(rep(list(paternal, maternal), ncells / 2)))
vars = 1:nrow(gt)
colnames(gt) = 1:ncells
rownames(gt) = vars
```

```
##create a feature annotation data-frame
nvars = nrow(gt)
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, gt = gt)

##Remove variants with imbalanced expression in less than 3 cells
nmincells = 3
acset_filt = filter_var_gt(acset, nmincells)
```

`filter_zerorow` *Filter variants with zero allele counts in all cells*

Description

`filter_zerorow` removes variants which do no have any read count

Usage

```
filter_zerorow(acset)
```

Arguments

`acset`	An acset list which must contain "refcount" and "altcount" elements with allele counts, see `new_acset`.

Value

acset An acset list subsetted on variants that pass the filter.

Examples

```
##create small artifical allele count matrixes where first row is zero
ncells = 10
ref = c(0, 10, 1, 5, 2, 9)
alt = c(0, 2, 5, 5, 6, 3)
refcount = as.matrix(as.data.frame(rep(list(ref, alt), ncells / 2)))
altcount = as.matrix(as.data.frame(rep(list(alt, ref), ncells / 2)))

vars = 1:nrow(refcount)
samples = 1:ncells
colnames(refcount) = samples
rownames(refcount) = vars
colnames(altcount) = samples
rownames(altcount) = vars

##create a feature annotation data-frame
nvars = length(vars)
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, altcount = altcount, refcount = refcount)

##Remove variants having no allele counts
acset_filt = filter_zerorow(acset)
```

---

marinov *Single-cell RNA-seq allele counts for the Marinov et al dataset*

Description

A dataset containing alternative and reference allele read counts per cell and heterozygous variant, derived from a single-cell RNA-seq dataset of the lymphoblastoid cell-line of HapMap individual NA12878.

Usage

```
data(marinov)
```

Format

An acset with four elements, featdata, refcount, altcount and phenodata, see `new_acset` for a description of these elements. The acset contains data for 2809 variants and 28 single cells.

Details

Allele counts were generated by alignment of the RNA-seq data to the haplo- genomes of NA12878 and subsequently running samtools mpileup using variants called as heterozygous within the DNA-seq data of the individual. Variants were filtered on being within RefSeq genes, in the dbSNP database and successfully phased (via transmission from the parental genome data). Variants were further filtered using the allele count data to not monoallelically express the same allele across cells (see `filter_homovars`) and on having imbalanced allelic expression in at least 3 cells (see `filter_var_gt`. Features were filtered on having at least two such variants (see `filter_feat_nminvar`). For further details on the generation of the allele count data see the Supplemental Data in Edsgard et al, scphaser: Haplotype Inference Using Single-Cell RNA-Seq Data, Bioinformatics, 2016.

Source

- RNA-seq fastq files `ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByStudy/sra/SRP/SRP018/SRP018838/`
- RNA-seq meta-info`ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE44nnn/GSE44618`
- Haplo-genomes `http://sv.gersteinlab.org/NA12878_diploid`
- DNA-seq variants `http://sv.gersteinlab.org/NA12878_diploid/CEUTrio.HiSeq.WGS.b37.bestPractices.phased.hg19.vcf`

`mousehybrid` *Single-cell RNA-seq allele counts for a mouse-hybrid dataset*

Description

A dataset containing alternative and reference allele read counts per cell and heterozygous variant, derived from a single-cell RNA-seq dataset of fibroblast and liver cells from crossed CAST/EiJ x C57BL/6J mouse strains. The dataset has been subsetted to 300 genes to restrict its size.

Usage

```
mousehybrid
```

Format

An acset with four elements, featdata, refcount, altcount and phenodata, see `new_acset` for a description of these elements. The acset contains data for 3313 variants and 336 single cells.

Details

Allele counts were generated by alignment of the RNA-seq data to each of the genomes of the two mouse strains and subsequently running samtools mpileup using variants that were homozygous within each strain and differed between the strain-genomes. Variants were filtered on being within RefSeq genes. Variants were further filtered using the allele count data to not monoallelically express the same allele across cells (see `filter_homovars`) and on having imbalanced allelic expression in at least 3 cells (see `filter_var_gt`. Features were filtered on having at least two such variants (see `filter_feat_nminvar`). For additional filters and further details on the generation of the allele count data see the Supplemental Data in Edsgard et al, scphaser: Haplotype Inference Using Single-Cell RNA-Seq Data, Bioinformatics, 2016.

Source

RNA-seq data can be found at `http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE75659`

`new_acset` *Construct an allele count set*

Description

`new_acset` constructs a list which is the data-structure used by the scphaser phasing functions.

Usage

```
new_acset(featdata, refcount = NA, altcount = NA, phenodata = NA,
  gt = NA)
```

Arguments

`featdata`	Data-frame with four required columns and arbitrary additional columns. The purpose of the featdata data-frame is to map variants to features and specify the two alleles of each variant. The four required columns must be named 'feat', 'var', 'ref' and 'alt'. The rownames must also be set to be identical to the var column. 'feat' is a character vector specifying feature names, such as gene names. 'var' is a character vector specifying variant names, such as dbSNP rs id. 'ref' and 'alt' are character vectors specifying the alleles of each variant, such as the reference and alternative allele.
`refcount`	Matrix with allelic counts for the reference allele with variants as rows and cells as columns. The rownames have to match the values in the 'var' column in the featdata, and the colnames the values in the phenodata 'sample' column. If refcount is provided altcount must also be provided. If the gt argument is provided then refcount and altcount are not required arguments.
`altcount`	Matrix with allelic counts for the alternative allele, where the row- and colnames must match those of refcount.
`phenodata`	Data-frame which annotates the cells. It must contain a column named 'sample'. If the phenodata argument is not provided it is created by the `link{new_acset}` function with the sample column set to be identical to the column names of refcount or gt.
`gt`	Matrix with integer values representing transcribed genotype calls. 0: reference allele most highly expressed, 1: bi-allelic expression with similar degree of expression from the two alleles, 2: alternative allele most highly expressed. NA's are allowed and can be used to indicate entries where no call could be made. The rownames have to match the values in the 'var' column in the featdata, and the colnames the values in the phenodata 'sample' column. If refcount and altcount are provided then the gt argument does not need to be provided.

Details

The function performs a number of error-checks to ensure that the constructed acset-list elements satisfy the data-format used by the phasing functions.

Value

acset An acset list which contains elements required to apply the phasing functions.

Examples

```
##create a small artificial genotype matrix
ncells = 10
paternal = c(0, 2, 0, 0, 2)
maternal = c(2, 0, 2, 2, 0)
gt = as.matrix(as.data.frame(rep(list(paternal, maternal), ncells / 2)))
vars = 1:nrow(gt)
colnames(gt) = 1:ncells
rownames(gt) = vars

##create a feature annotation data-frame
nvars = nrow(gt)
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, gt = gt)
```

`phase` *Phasing of alleles*

Description

`phase` phases alleles using allelic counts or genotype calls from single-cell RNA-seq data.

Usage

```
phase(acset, input = "gt", weigh = FALSE, method = "exhaust",
  nvars_max = 10, verbosity = -1, bp_param = BiocParallel::SerialParam())
```

Arguments

`acset`	An acset list created by the `new_acset` function. It must contain elements with either a genotype matrix or two matrixes containing the reference and alternative allele counts, respectively.
`input`	A character string specifying if allele counts or genotype calls should be used for phasing. Two values are allowed, 'gt' or 'ac'. 'gt' specifies that genotype calls should be used. 'ac' specifies that allele counts should be used.
`weigh`	A logical specifying if the sample-size, that is, the scale of the allele counts at a variant, should be taken into account. Variants with high counts will be given a greater weight as they are more reliable.
`method`	A character string specifying the clustering method to be used for the phasing. Two values are allowed, 'exhaust' or 'pam'.
`nvars_max`	An integer specifying the number of variants within a feature (e.g. gene), above which 'pam' clustering will be used even if the method argument was set to 'exhaust'.
`verbosity`	An integer specifying the verbosity level. Higher values increase the verbosity.
`bp_param`	A BiocParallelParam instance, see `bplapply`.

Details

The function phases alleles within each feature. As phasing is not done between features inferred haplotypes should only be used within features.

Value

An acset list with the following elements added by the phasing function: 'phasedfeat': Data-frame with six columns. Four first columns, 'feat', 'var', 'ref' and 'alt' are taken from the featdata data-frame of the input acset. The last two columns 'hapA' and 'hapB' contains the haplotype sequence of alleles. 'args': List where each element corresponds to argument values supplied to the phase function. 'varflip': Character vector with names of variants where the alleles were swapped. 'score': Numeric vector with a variability score per feature after phasing. 'gt_phased': Character matrix of genotypes after having swapped alleles according to the inferred phase. 'weights': Numeric matrix with a weight per variant and cell. Only added if arguments set as weigh == TRUE and input == 'gt'.

Examples

```
##create a small artificial genotype matrix
ncells = 10
paternal = c(0, 2, 0, 0, 2)
maternal = c(2, 0, 2, 2, 0)
gt = as.matrix(as.data.frame(rep(list(paternal, maternal), ncells / 2)))
vars = 1:nrow(gt)
colnames(gt) = 1:ncells
rownames(gt) = vars

##feature annotation data-frame
nvars = nrow(gt)
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, gt = gt)

##phase
acset = phase(acset, input = 'gt', weigh = FALSE, method = 'exhaust',
verbosity = 0)

##' The haplotype output is contained in an element of an acset list that was
##added by the phasing function and is named "phasedfeat":
head(acset[['phasedfeat']])
```

---

plot_conc *Phasing concordance plot*

Description

plot_conc plots the phasing concordance, see set_gt_conc

Usage

```
plot_conc(acset, feats = NA, cex = 0.5)
```

Arguments

acset	An acset list created by new_acset. The acset must contain the elements 'gt_conc' and 'gt_phased_conc' which contain the concordance and inconcordance for every gene before (gt_conc) and after phasing (gt_phased_conc), see set_gt_conc.
feats	A character vector of feature names to include in the plot.
cex	A numerical value giving the amount by which plotting text and symbols should be magnified relative to the default, see par.

Details

To assess the success of the phasing one can calculate the degree of variability remaining if all cells with haplotype 2 are set to haplotype 1. As a rough measure of this this function calculates the variability as the number of cells that differ from the inferred haplotype for every gene with two variants. The differing number of cells per gene we denote as the inconcordance and the number of cells with identical haplotype to the inferred haplotype as concordance, see `set_gt_conc` for further details.

Examples

```
##load dataset
invisible(marinov)
acset = new_acset(featdata = marinov[['featdata']], refcount =
marinov[['refcount']], altcount = marinov[['altcount']], phenodata =
marinov[['phenodata']])

##Call gt
acset = call_gt(acset, min_acount = 3, fc = 3)

##Filter variants and genes
acset = filter_acset(acset, nmincells = 5, nminvar = 2)

##Phase
acset = phase(acset, input = 'gt', weigh = FALSE, method = 'exhaust')

##Calculate the genotype concordance before and after phasing
acset = set_gt_conc(acset)

##Plot the genotype concordance before and after phasing
acset = plot_conc(acset)
```

`racset` *Randomize an acset*

Description

`racset` randomizes the entries in the allele count matrixes or the genotype matrix

Usage

```
racset(acset, type = "ac", fixedrowmargin = FALSE)
```

Arguments

`acset`	An acset list created by `new_acset`.
`type`	A character string with two allowed values. 'ac' specifies that the allele count matrixes should be randomized and 'gt' specifies that the genotype matrix should be randomized.
`fixedrowmargin`	Boolean specifying if the row-sums should be kept unchanged.

Details

The function scrambles all entries within each allele count matrix or within the genotype matrix.

Value

acset A randomized acset list.

Examples

```
##create a small artificial genotype matrix
ncells = 10
paternal = c(0, 2, 0, 0, 2)
maternal = c(2, 0, 2, 2, 0)
gt = as.matrix(as.data.frame(rep(list(paternal, maternal), ncells / 2)))
vars = 1:nrow(gt)
colnames(gt) = 1:ncells
rownames(gt) = vars

##create a feature annotation data-frame
nvars = nrow(gt)
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, gt = gt)

##Randomize the genotype matrix
type = 'gt'
acset_rand = racset(acset, type)
```

---

set_gt_conc *Cell-distribution variability upon phasing*

Description

set_gt_conc calculates an approximate measure of the spread of the cells in the allele-specific expression variant-space per gene

Usage

```
set_gt_conc(acset)
```

Arguments

acset	An acset list created by new_acset. The acset must contain an element 'gt', see call_gt and an element 'gt_phased', see function phase.

Details

To assess the success of the phasing one can calculate the degree of variability remaining if all cells with haplotype 2 are set to haplotype 1. As a rough measure of this this function calculates the variability as the number of cells that differ from the inferred haplotype for every gene with two variants. The differing number of cells per gene we denote as the inconcordance and the number of cells with identical haplotype to the inferred haplotype as concordance. This can be understood by viewing each cell as a point in a transcribed genotype variant space, where each dimension is a variant (within a gene), with values in the transcribed genotype domain. The expression of the alleles within a gene from a single cell will then tend to cluster towards the haplotype state, and the remaining variability of the cells in that space after phasing can be used to measure how well the cells conform to the haplotype.

Value

acset An acset list where two elements have been added or updated, 'gt_conc' and 'gt_phased_conc'. The elements contain the concordance and inconcordance for every gene before (gt_conc) and after phasing (gt_phased_conc).

Examples

```
##load dataset
invisible(marinov)
acset = new_acset(featdata = marinov[['featdata']], refcount =
marinov[['refcount']], altcount = marinov[['altcount']], phenodata =
marinov[['phenodata']])

##Call gt
acset = call_gt(acset, min_acount = 3, fc = 3)

##Filter variants and genes
acset = filter_acset(acset, nmincells = 5, nminvar = 2)

##Phase
acset = phase(acset, input = 'gt', weigh = FALSE, method = 'exhaust')

##Get genotype concordance before and after phasing
acset = set_gt_conc(acset)
```

`subset_cols` *Subset cells*

Description

`subset_cols` subsets the cells of an acset

Usage

```
subset_cols(acset, sel.ind)
```

Arguments

`acset`	An acset list created by the `new_acset` function.
`sel.ind`	A vector with either integer column indices or a character vector to subset the colnames.

Value

acset An acset list where every element containing cell-related information has been subsetted

Examples

```
##create a small artificial genotype matrix
ncells = 10
paternal = c(0, 2, 0, 0, 2)
maternal = c(2, 0, 2, 2, 0)
gt = as.matrix(as.data.frame(rep(list(paternal, maternal), ncells / 2)))
vars = 1:nrow(gt)
colnames(gt) = 1:ncells
rownames(gt) = vars

##create a feature annotation data-frame
nvars = nrow(gt)
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, gt = gt)

##subset variants
sel_ind = c(1, 3)
acset = subset_cols(acset, sel_ind)
```

`subset_feat` *Subset features*

Description

`subset_feat` subsets the features of an acset

Usage

```
subset_feat(acset, feat)
```

Arguments

`acset`	An acset list created by the `new_acset` function.
`feat`	A character vector with features to subset.

Value

acset An acset list where every element containing feature-related information has been subsetted

Examples

```
##load dataset
invisible(marinov)
acset = new_acset(featdata = marinov[['featdata']], refcount =
marinov[['refcount']], altcount = marinov[['altcount']], phenodata =
marinov[['phenodata']])

##subset features
sel_feat = c('HMGB1', 'ITGA4')
acset_filt = subset_feat(acset, sel_feat)
```

`subset_rows` *Subset variants*

Description

`subset_rows` subsets the variants of an acset

Usage

```
subset_rows(acset, sel.ind)
```

Arguments

`acset`	An acset list created by the `new_acset` function.
`sel.ind`	A vector with either integer row indices or a character vector to subset the rownames.

Value

acset An acset list where every element containing variant-related information has been subsetted

Examples

```
##create a small artificial genotype matrix
ncells = 10
paternal = c(0, 2, 0, 0, 2)
maternal = c(2, 0, 2, 2, 0)
gt = as.matrix(as.data.frame(rep(list(paternal, maternal), ncells / 2)))
vars = 1:nrow(gt)
colnames(gt) = 1:ncells
rownames(gt) = vars

##create a feature annotation data-frame
nvars = nrow(gt)
featdata = as.data.frame(matrix(cbind(rep('jfeat', nvars),
as.character(1:nvars), rep('dummy', nvars), rep('dummy', nvars)), ncol = 4,
dimnames = list(vars, c('feat', 'var', 'ref', 'alt'))), stringsAsFactors =
FALSE)

##create acset
acset = new_acset(featdata, gt = gt)

##subset variants
sel_ind = c(1, 3)
acset = subset_rows(acset, sel_ind)
```

Index

∗Topic **datasets**
    marinov, 8
    mousehybrid, 9

bplapply, *11*

call_gt, 2, *3*, *6*, *14*

filter_acset, 3
filter_feat_nminvar, *3*, 4, *8*, *9*
filter_homovars, 5, *8*, *9*
filter_var_gt, 6, *8*, *9*
filter_zerorow, 7

marinov, 8
mousehybrid, 9

new_acset, *2–9*, 9, *11–14*, *16*, *17*

par, *12*
phase, 11, *14*
plot_conc, 12

racset, 13

set_gt_conc, *12*, *13*, 14
subset_cols, 15
subset_feat, 16
subset_rows, 17