Package 'scquantum'

March 23, 2022

Title Estimate Ploidy and Absolute Copy Number from Single Cell Sequencing

Version 1.0.0

Description Given bincount data from single-cell copy number profiling (segmented or unsegmented), estimates ploidy, and uses the ploidy estimate to scale the data to absolute copy numbers. Uses the modular quantogram proposed by Kendall (1986) <doi:10.1002/0471667196.ess2129.pub2>, modified by weighting segments according to confidence, and quantifying confidence in the estimate using a theoretical quantogram. Includes optional fused-lasso segmentation with the algorithm in Johnson (2013) <doi:10.1080/10618600.2012.681238>, using the implementation from glmgen by Arnold, Sadhanala, and Tibshirani.

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.1.2

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Alexander Davis [aut, cre],
Taylor Arnold [cph] (Author of segmentation function),
Veeranjaneyulu Sadhanala [cph] (Author of segmentation function),
Ryan Tibshirani [cph] (Author of segmentation function)

Maintainer Alexander Davis <alexlovesmolecules@gmail.com>

Repository CRAN

Date/Publication 2022-03-23 09:00:06 UTC

R topics documented:

`ploidy.inference` *Ploidy inference with quantogram*

Description

Infer ploidy of a cell, given a copy number profile. Constructs a quantogram (either modular or cosine, depending on parameters). The maximum of the quantogram is the estimated ploidy. If unsegmented bincounts are given, segmentation will be performed using the fused lasso.

Usage

```
ploidy.inference(
  x,
  chrom = NULL,
  start = NULL,
  end = NULL,
  penalty = 25,
  do_segmentation = TRUE,
  seg_length = NULL,
  iod = NULL,
  mean_bincount = NULL
)
```

Arguments

`x`	Bincounts or segment means
`chrom`	Optional chromosome numbers
`start`	Optional bin/segment start positions
`end`	Optional bin/segment end positions
`penalty`	If segmenting, penalty parameter for the fused lasso (higher penalty, fewer segments)
`do_segmentation`	Boolean, whether to do segmentation (set this to TRUE if giving unsegmented bincounts)
`seg_length`	If giving already segmented data, length of each segment
`iod`	If giving already segmented data, the index of dispersion of the bincount distribution (that is, within segments, not including between-segment variance)
`mean_bincount`	If giving already segmented ratio values, the original mean bincount

Value

A ploidy inference object

penalty The segmentation penalty given as an argument, if any

multiply_ratios_by To convert ratios to (unrounded) copy number estimates, multiply by this number

subtract_from_scaled_ratios To convert ratios to (unrounded) copy numbers, after multiplying, subtract this number. Only required if the count data have some extra reads even at copy number 0, generally due to mapping problems

ploidy The estimated ploidy

peak_height The height of the quantogram peak at the estimated ploidy. Between 0 and 1. Higher values indicate a stronger signal

segmentation The segmented values (either given as an argument, or produced interally by segmentation)

polar_quantogram The complex-valued quantogram, whose absolute values measure consistency with each possible ploidy

bincounts The raw bincounts given as an argument (if a segmentation was not given directly)

theoretical_quantogram Based on the inferred copy numbers and index of dispersion, what the absolute value of the quantogram should look like. Deviation of this theoretical quantogram from the real one indicate that the ploidy estimate may be wrong

theoretical_peak_height Height of the peak in the theoretical quantogram, measuring the expected strength of signal for the ploidy value

confidence_ratio Ratio of actual to theoretical peak height. Values near (or above) 1 indicate the signal was as strong as would be expected gievn this data quality and ploidy; low values indicate that the ploidy inference may be wrong or that there are unexpected quality issues with the data

Examples

```
# Generating a random copy number profile
set.seed(705)
cns <- rpois(30, 3) + 1
x <- unlist(lapply(cns, function(cn) rpois(100, 25 * cn)))
annotations <- data.frame(chrom = 1, start = 1:length(x), end = 1:length(x))

# Inferring ploidy
# Annotations and penalty are optional
estimate.from.bincounts <- ploidy.inference(x, annotations$chrom, annotations$start, penalty = 25)

# Using scquantum internal functions to segment the data and estimate index
# of git@github.com:navinlabcode/scquantum.git
# dispersion
mu.est <- mean(x)
iod.est <- timeseries.iod(x)
seg <- prof2invals(x, 25, annotations, "chrom", "start", "end")
mean.est <- mean(x)
iod.est <- timeseries.iod(x)
estimate.from.segmentation <-
  ploidy.inference(
    seg$mean,
    seg$chrom,
    seg$start,
    seg$end,
    iod = iod.est,
    mean_bincount = mean.est,
    do_segmentation = FALSE
  )
```

Index