Package: HaarSeg (via r-universe)

September 7, 2024

Version 0.0.4

Title Fast and Flexible Microarray Segmentation

Author Erez Ben-Yaacov, Yonina C. Eldar, and Henrik Bengtsson (R package)

Maintainer Henrik Bengtsson <henrikb@braju.com>

Description A fast and flexible method for the segmentation of aCGH data using the HaarSeg method by Ben-Yaacov and Eldar (2008) <doi:10.1093/bioinformatics/btn272>.

URL https://github.com/HenrikBengtsson/HaarSeg

BugReports https://github.com/HenrikBengtsson/HaarSeg/issues

License LGPL (== 2.1)

Depends R (>= 2.6.0)

LazyLoad yes

Repository https://henrikbengtsson.r-universe.dev

RemoteUrl https://github.com/HenrikBengtsson/HaarSeg

RemoteRef main

RemoteSha 20016d082abf6f8459564104f29a28aa42c39ead

Contents

HaarSeg-package *Package HaarSeg*

Description

A fast and flexible method for the segmentation of aCGH data using the HaarSeg method by Ben-Yaacov and Eldar (2008) <doi:10.1093/bioinformatics/btn272>..

Dependancies and other requirements

None.

To get started

haarSeg()

Author(s)

Erez Ben-Yaacov. R package created by Henrik Bengtsson.

References

[1] Ben-Yaacov E. and Eldar YC. *A fast and flexible method for the segmentation of aCGH data*, Bioinformatics, 2008. https://www.ee.technion.ac.il/Sites/People/YoninaEldar/Info/software/HaarSeg.htm

haarSeg *Performs segmentation according to the HaarSeg algorithm*

Description

Performs segmentation according to the HaarSeg algorithm. HaarSeg segmentation is based on detecting local maxima in the wavelet domain, using Haar wavelet. The main algorithm parameter is breaksFdrQ, which controls the sensitivity of the segmentation result. This function includes several optional extentions, supporting the use of weights (also known as quality of measurments) and raw measurments. We recommend using both extentions where possible, as it greatly improves the segmentation result. Raw red / green measurments are used to detect low value probes, which are more sensitive to noise.

Usage

```
haarSeg(I, W=vector(), rawI=vector(), chromPos=matrix(c(1, length(I)),
  nrow = 1, ncol = 2), breaksFdrQ=0.001, haarStartLevel=1, haarEndLevel=5)
```

Arguments

I	a single array of log(R/G) measurements, sorted according to their genomic location.
W	Weight matrix, corresponding to quality of measurment. Insert $1/(\sigma^2)$ as weights if your platform output σ as the quality of measurment. W must have the same size as I.
rawI	The mininum between the raw red and raw green measurment (before applying log ratio, but after any background reduction and/or normalization). rawI is used for the non-stationary variance compensation. rawI must have the same size as I.
chromPos	A matrix of two columns. The first column is the start index of each chromosome. The second column is the end index of each chromosome.
breaksFdrQ	The FDR q parameter. This value should lie between 0 and 0.5. The smaller this value is, the less sensitive the segmentation result will be. For example, we will detect less breaks in the segmentation result when using Q = 1e-4, compared to the amounts of breaks when using Q = 1e-3. Common used values are 1e-2, 1e-3, 1e-4. Default value is 1e-3.
haarStartLevel	The detail subband from which we start to detect peaks. The higher this value is, the less sensitive we are to short segments. The default is value is 1, corresponding to segments of 2 probes.
haarEndLevel	The detail subband until which we use to detect peaks. The higher this value is, the more sensitive we are to large trends in the data. This value DOES NOT indicate the largest possible segment that can be detected. The default is value is 5, corresponding to step of 32 probes in each direction.

Value

A list containing two elements:

SegmentsTable	Segments result table: (segment start index, segment size, segment value)
Segmented	The complete segmented signal (same size as I).

Author(s)

Erez Ben-Yaacov

Examples

```
real.data = c(rep.int(0,2000),rep.int(1,100),rep.int(0,2000));
noisy.data = real.data + rnorm(length(real.data),sd = 0.2);
plot(noisy.data)

# using default parameters
seg.data = haarSeg(noisy.data);

#segments result table: segment start index | segment size | segment value
print(seg.data$SegmentsTable)
```

```
# the complete segmented signal
lines(seg.data$Segmented, col="red", lwd=3)
```

Index