Package 'ArrayTV'

April 22, 2016

Version 1.8.0

Title Implementation of wave correction for arrays

Date Thu Sep 19 10:45:22 EDT 2013

Author Eitan Halper-Stromberg

Maintainer Eitan Halper-Stromberg <Eitan177@gmail.com>

Description Wave correction for genotyping and copy number arrays

License GPL (>= 2)

LazyLoad yes

Depends R (>= 2.14)

Imports foreach, DNAcopy, methods, oligoClasses (>= 1.21.3)

Suggests RColorBrewer, crlmm, ff,
BSgenome.Hsapiens.UCSC.hg18,BSgenome.Hsapiens.UCSC.hg19,
lattice, latticeExtra, RUnit, BiocGenerics

Enhances doMC, doSNOW, doParallel

Collate AllGenerics.R calctvScore.R computeGC.R correctionTVscore.R
CorrectM.R gcCorrectMain.R gcFracAllWin.R gcFracOneRange.R
getGCinBestWindowGenome.R methods-gcCorrect.R priorFracs.R
priorFracsRestOfGenome.R

biocViews CopyNumberVariation

NeedsCompilation no

R topics documented:

ArrayTV-package *ArrayTV-implementation of GC correction for arrays*

Description

Correct gc biases in array data. The user must provide a range of windows and this function will compute the tv score for each window and correct the arrays based upon the window with the highest tv score (i.e with window showing the most gc bias)

Details

Package:	ArrayTV
Type:	Package
Version:	1.0
Date:	2012-08-02
License:	Johsn Hopkins School of Medicine

gcCorrectMain takes a matrix of array signal intensities, each column representing one array, along with vectors corresponding to probe positions and chromosomes. The gcCorrect methods enables array input for objects of class matrix, BeadStudioSet,BafLrrSet, and BafLrrSetList

Author(s)

Eitan Halper-Stromberg Maintainer: Eitan Halper-Stromberg

References

Yuval Benjamini and Terence P. Speed. Summarizing and correcting the GC content bias in high-throughput sequencing. Nucleic Acids Res, 40(10):e72, May 2012.

Eitan Halper-Stromberg, Laurence Frelin, Ingo Ruczinski, Robert Scharpf, Chunfa Jie, Benilton Carvalho, Haiping Hao, Kurt Hetrick, Anne Jedlicka, Amanda Dziedzic, Kim Doheny, Alan F. Scott, Steve Baylin, Jonathan Pevsner, Forrest Spencer, and Rafael A. Irizarry. Performance assessment of copy number microarray platforms using a spike-in experiment. Bioinformatics, 27(8):1052

Adam B Olshen, E. S. Venkatraman, Robert Lucito, and Michael Wigler. Circular binary segmentation for the analysis of array-based DNA copy number data. Biostatistics, 5(4):557

See Also

type ?gcCorrectMain.Rd at the command prompt for an example

computeGC-methods *Compute the GC composition for selected windows*

Description

Compute the GC composition for selected windows around a vector of genomic positions (e.g., locations of markers on an array)

Methods

computeGC(x, maxwins, increms, chr, build, ...):

x can be numeric, in which case x should be a vector of physical position along the genome. Alternatively, x can be an instance of the BafLrrSetList class. When x is numeric, arguments chr and build are required. Arguments chr and build are ignored for BafLrrSetList objects.

Arguments maxwins, increms, build, and chr are as described in the documentation for gcCorrectMain

See Also

BafLrrSetList, gcCorrectMain

Examples

```
    if(require(crlmm) && require(ff) && require(doParallel)){
## nodes may be set to the number of cpus available
        nodes<-3
        cl <- makeCluster(nodes)
        registerDoParallel(cl)

data(cnSetExample, package="crlmm")
brList <- BafLrrSetList(cnSetExample)
pos <- unlist(position(brList))
chr <- paste("chr",
     rep(chromosome(brList), elementLengths(brList)),
     sep="")
gc.matrix1 <- ArrayTV:::computeGC(x=pos, maxwins=c(12,10e3),
              increms=c(12,10e3),
            chr=chr,
          build="hg19")
gc.matrix2 <- ArrayTV:::computeGC(brList, c(12, 10e3),
                    c(12, 10e3))
        identical(gc.matrix1, gc.matrix2)
        stopCluster(cl)
}
```

gcCorrect-methods *Methods for* gcCorrect *in Package* **ArrayTV**

Description

Correct gc biases in array data for arrays stored as objects of an acceptable class. The user provides the array data and a range of windows and these methods will format the data and call gcCorrectMain. This function computes the tv score for each window and corrects the arrays based upon the window with the highest tv score (i.e with window showing the most gc bias)

Methods

gcCorrect(object, ...):

Methods have been defined for objects of class matrix, BeadStudioSet, BafLrrSet, and BafLrrSetList.

Objects of class BafLrrSetList are containers for log R ratios and B allele frequencies (BAFs) stored by chromosome. In particular, each element in this list class contains BAFs and log R ratios as assayData and genomic annotation of the markers (featureData) for one chromosome.

The gcCorrect method for objects of this class extracts the log R ratios for all elements (chromosomes) in the list, and then combines these into a single matrix. If the log R ratios are stored as ff objects, the samples will be GC-corrected in chunks determined by the function ocSamples(). For example, if object contains 100 samples and ocSamples(10) was specified prior to calling gcCorrect, the wave correction will be performed on 10 samples at a time in order to keep RAM at an acceptable level. When processing large numbers of samples, it can be helpful to evaluate the tv score on a subset of the samples, pick one or two windows for the correction, and precompute the GC content for these windows. See the examples below for the implementation with BafLrrSetList objects for details. When the assay data is represented as ff objects, the gcCorrect method returns the value NULL. Note that the assay data stored on disk will have changed as a result of calling this method.

When object is a matrix of log R ratios, the gcCorrect method returns a matrix of wave-adjusted log R ratios.

Additional arguments are passed to gcCorrectMain through the ... operator.

References

Benjamini Y, Speed TP. Summarizing and correcting the GC content bias in high-throughput sequencing. Nucleic Acids Res 2012;40:e72

See Also

BeadStudioSet, BafLrrSetList, BafLrrSet gcCorrectMain

Examples

```
        ## Example with 2 iterations of correction
path <- system.file("extdata", package="ArrayTV")
load(file.path(path, "array_logratios.rda"))
nimblegen[, "M"] <- nimblegen[, "M"]/1000
increms <- c(10,1000,100e3)
wins <- c(100,10e3,1e6)
        if(require(doParallel)){
  ## nodes may be set to the number of cpus available
          nodes<-3
          cl <- makeCluster(nodes)
          registerDoParallel(cl)
        }

## calculate tv scores in many windows and correct M values using the best window
nimcM1List <- gcCorrectMain(nimblegen[, "M",
drop=FALSE],
chr=rep("chr15",
nrow(nimblegen)),
starts=nimblegen[,"position"],
increms=increms,
maxwins=wins,build='hg18')
tvScores <- nimcM1List[['tvScore']]
winsorted <- as.numeric(rownames(tvScores)[order(tvScores,decreasing=TRUE)])
logwinsorted <- log(as.numeric(winsorted),10)
logwinsortdiff <- abs(logwinsorted[1]-logwinsorted)
## correct M values a 2nd time using a different window size
nimcM2List <- gcCorrect(nimcM1List[['correctedVals']],
chr=rep("chr15",
nrow(nimblegen)),
starts=nimblegen[,"position"],
maxwins=winsorted[logwinsortdiff>=1][1],
build='hg18')
## Refer to the vignette for details

## Example using a list container (BafLrrSetList) containing log R
## ratios and B allele frequencies

if(require(crlmm) && require(ff)){
data(cnSetExample, package="crlmm")
brList <- BafLrrSetList(cnSetExample)
is(lrr(brList)[[1]], "ff_matrix")
rold <- lrr(brList)[[1]][, 1]/100
##
## To avoid having too much data in RAM it might be
## useful to process the samples n at a time
##
##  The number of samples to be processed at a time is
##  set by the ocSamples function.  For example, to
##  process 20 samples at a time one would do
ocSamples(20)
```

```
##
## When assay data elements are ff objects, the wave
## corrected values are updated on disk.  Currently,
## the  data is stored is here:
filename(lrr(brList)[[1]])
##
## To avoid permanently changing the log R ratio
## values for the brList object, we copy the ff files
## to a different path and create a new BafLrrSetList
## object.  This is done by the non-exported function
## "duplicateBLList". The path for the new ff objects
## is given by the function ldPath().  Here, we use a
## temp directory
ldPath(tempdir())
brList.copy <- oligoClasses:::duplicateBLList(brList, ids=sampleNames(brList))
filename(lrr(brList.copy)[[1]])
##
## wave correct the log R ratios
##
gcCorrect(brList.copy, increms=c(12, 10e3),
  maxwins=c(12, 10e3))
##
##
rupdated <- lrr(brList.copy)[[1]][, 1]/100
## note that rold and rupdated are no longer the same
## since the log R ratios were updated in the
## brList.copy container
plot(rupdated, rold, pch=".")
##
## Remarks on efficiency
##
## If a large number of samples are to be processed,
## the most efficient procedure is to settle on an
## appropriate window size for gc correction using a
## subset of the dataset (e.g., 20 samples).  See the
## vignette for details.  Here, we assume that two
## windows were already selected using such a
## procedure (here, 12 bp and 10,000 bp) and the goal
## is to efficiently do wave correction using these
## two windows for all the samples in a large study.
## Currently, our recommended approach is to first
## calculate the gc content for these windows:
gc.matrix <- ArrayTV:::computeGC(brList.copy, c(12, 10e3),
 c(12, 10e3))
## The number of columns in gc.matrix will correspond
## to the number of windows
ncol(gc.matrix)
## Having calculated the gc content for these two
## windows, we pass the gc matrix to the method
## gcCorrect.  This function will iteratively update
## the log R ratios for the gc content given by the
## columns in gc.matrix (the log R ratios are updated
## for each column in gc.matrix).
```

```
ArrayTV:::gcCorrect(brList.copy, increms=c(12, 10e3),
maxwins=c(12, 10e3),
providedGC=gc.matrix)
                       stopCluster(cl)
}
```

---

gcCorrectMain *gcCorrectMain*

Description

Correct gc biases in array data. The user must provide a range of windows and this function will compute the tv score for each window and correct the arrays based upon the window with the highest tv score (i.e with window showing the most gc bias)

Usage

```
gcCorrectMain(Ms, chr, starts, samplechr, increms,
                      maxwins, jittercorrection=FALSE,
                  returnOnlyTV=FALSE, onlyGC=FALSE,providedGC=NULL,build, verbose=FALSE)
```

Arguments

Ms	A matrix holding the array values
chr	a vector indicating the chromosome of each probe (should be the same length as the number of rows in the array matrix)
starts	a vector holding the genomic coordinates of the start positions the probes (should be the same length as the number of rows inthe array matrix)
samplechr	a subset (or the entire set) of the chromosomes represented in the arrays. The probes on these chromosomes will be used to compute the tvScore
increms	a vector of integers
maxwins	a vector ofintegers with each value >= the corresponding value in increms. Corresponding values of maxwins and increms must have the same quotient. For instance, if increms is c(10,1000) then an acceptable value of maxwins would be(500,5000) because the quotients of the corresponding elements are both 5.
jittercorrection	if TRUE amplify the correction values by a small amount to decrease the autocorrelation of proximal probes
returnOnlyTV	if TRUE only return the tvScores rather than applying corrections
onlyGC	if TRUE return the gc fraction of each window (the result may be subsequently passed into the providedGC argument to be used in correction)
providedGC	a vector of gc values (fractions from zero to 1) to use for computing corrections. This should be the same length as the number of rows in the array, if provided

build	the build matching the array. This is used to load the BSgenome object. If the arrays are annotated based on hg18 this would be 'hg18', if the annotation is based on hg19 this would be 'hg19'.
verbose	provide output throughout computation

Details

see example

Value

The default is to return corrected values for the arrays passed in. This result will have the same dimensions as the first argument (the uncorrected values).if returnOnlyTV is TRUE this will be a matrix of tv score values, one row per window, one column per array. The rownames of the matrix will indicate the size of the window. If onlyGC=TRUE the gc fraction of each window will be returned.

Author(s)

Eitan Halper-Stromberg

References

Optimal window finding was inspired by a similar concept applied to gc correction for next generation sequencing experiments:Benjamini Y, Speed TP. Summarizing and correcting the GC content bias in high-throughput sequencing. Nucleic Acids Res 2012;40:e72

Examples

```
## Example with parallelization, setting the number of processors to 3

path <- system.file("extdata", package="ArrayTV")
load(file.path(path, "array_logratios.rda"))
nimblegen[, "M"] <- nimblegen[, "M"]/1000

increms <- c(10,1000,100e3)
wins <- c(100,10e3,1e6)
        if(require(doParallel)){
  ## nodes may be set to the number of cpus available
          nodes<-3
          cl <- makeCluster(nodes)
          registerDoParallel(cl)
        }

## calculate tv scores in many windows and correct M values
## using the best window
nimcM1List <- gcCorrectMain(nimblegen[, "M", drop=FALSE],
    chr=rep("chr15",nrow(nimblegen)),
    starts=nimblegen[, "position"],
    increms=increms,
```

```
    maxwins=wins,
    build='hg18')
tvScores <- nimcM1List[['tvScore']]
winsorted <- as.numeric(rownames(tvScores)[order(tvScores,decreasing=TRUE)])
logwinsorted <- log(as.numeric(winsorted),10)
logwinsortdiff <- abs(logwinsorted[1]-logwinsorted)
## correct M values a 2nd time using a different window size
nimcM2List <- gcCorrect(nimcM1List[['correctedVals']],
chr=rep("chr15",nrow(nimblegen)),
starts=nimblegen[, "position"],
maxwins=winsorted[logwinsortdiff>=1][1],
build='hg18')
        stopCluster(cl)
## Refer to the vignette for details
```

Index