Package 'bPeaks'

February 19, 2015

Type Package

Title bPeaks: an intuitive peak-calling strategy to detect transcription factor binding sites from ChIP-seq data in small eukaryotic genomes

Version 1.2

Date 2014-02-28

Author Jawad MERHEJ and Gaelle LELANDAIS

Maintainer Gaelle LELANDAIS <gaelle.lelandais@univ-paris-diderot.fr>

Description bPeaks is a simple approach to identify transcription factor binding sites from ChIP-seq data. Our general philosophy is to provide an easy-to-use tool, well-adapted for small eukaryotic genomes (< 20 Mb). bPeaks uses a combination of 4 cut-offs (T1, T2, T3 and T4) to mimic ``good peak" properties as described by biologists who visually inspect the ChIP-seq data on a genome browser. For yeast genomes, bPeaks calculates the proportion of peaks that fall in promoter sequences. These peaks are good candidates as transcription factor binding sites.

License GPL

Depends R (>= 2.10)

NeedsCompilation no

Repository CRAN

Date/Publication 2014-02-28 17:55:13

R topics documented:

bPeaks-package *bPeaks: an intuitive peak-calling strategy to detect transcription factor binding sites from ChIP-seq data in small eukaryotic genomes*

Description

bPeaks is a simple approach to identify transcription factor binding sites from ChIP-seq data. Our general philosophy is to provide an easy-to-use tool, well-adapted for small eukaryotic genomes (< 20 Mb). bPeaks uses a combination of 4 cutoffs (T1, T2, T3 and T4) to mimic "good peak" properties as described by biologists who visually inspect the ChIP-seq data on a genome browser. For yeast genomes, bPeaks calculates the proportion of peaks that fall in promoter sequences. These peaks are good candidates as transcription factor binding sites.

Details

Package:	bPeaks
Type:	Package
Version:	1.2
Date:	2014-02-28
License:	GPL

Author(s)

Jawad MERHEJ and Gaelle LELANDAIS Maintainer: Gaelle LELANDAIS <gaelle.lelandais@univ-paris-diderot.fr>

References

More information can be found online: `http://bpeaks.gene-networks.net/`

See Also

`http://bpeaks.gene-networks.net/`

Examples

```
# get library
library(bPeaks)

# STEP 1: get PDR1 data (ChIP-seq experiments, IP and control samples,
#  related to the transcription factor Pdr1 in yeast Saccharomyces
#  cerevisiae)
data(dataPDR1)
```

```
# STEP 2 : bPeaks analysis (only 10 kb of chrIV are analyzed here,
#          as an illustration)
bPeaksAnalysis(IPdata = dataPDR1$IPdata[40000:50000,],
               controlData  = dataPDR1$controlData[40000:50000,],
       cdsPositions = dataPDR1$cdsPositions,
               windowSize = 150, windowOverlap = 50,
               IPcoeff = 4, controlCoeff = 2,
       log2FC  = 1, averageQuantiles = 0.5,
               resultName = "bPeaks_example")

# --> Result files (PDF and BED) are written in the working directory.

## Not run:
# -> bPeaks analysis, all chromosome IV and default parameters (optimized for yeasts)

# STEP 1: get PDR1 data (ChIP-seq experiments, IP and control samples,
#  related to the transcription factor Pdr1 in yeast Saccharomyces
#  cerevisiae)
data(dataPDR1)

# STEP 2: bPeaks analysis
bPeaksAnalysis(IPdata = dataPDR1$IPdata,
               controlData  = dataPDR1$controlData,
       cdsPositions = dataPDR1$cdsPositions,
               windowSize = 150, windowOverlap = 50,
               IPcoeff = 2, controlCoeff = 2,
               log2FC = 2, averageQuantiles = 0.9,
               resultName = "bPeaks_PDR1",
               peakDrawing = TRUE)

# STEP 3 : procedure to locate peaks according to
#     gene positions
peakLocation(bedFile = "bPeaks_PDR1_bPeaks_allGenome.bed",
            cdsPositions = yeastCDS$Saccharomyces.cerevisiae,
            outputName = "bPeakLocation_finalPDR1", promSize = 800)

# -> Note that cds (genes) positions are stored in bPeaks package for several yeast
# species
data(yeastCDS)

summary(yeastCDS)
#                         Length Class      Mode
# Debaryomyces.hansenii    31370  -none-     character
# Eremothecium.gossypii    23615  -none-     character
# Kluyveromyces.lactis     25380  -none-     character
# Pichia.sorbitophila      55875  -none-     character
# Saccharomyces.kluyveri   27790  -none-     character
# Yarrowia.lipolytica      32235  -none-     character
# Zygosaccharomyces.rouxii 24955  -none-     character
# Saccharomyces.cerevisiae     5  data.frame list
# Candida.albicans             5  data.frame list
# Candida.glabrata             5  data.frame list
```

```
## End(Not run)
```

`baseLineCalc` *Function to calculate the average number of reads mapped on each nucleotide in a genome*

Description

This function calculates the mean genome-wide read depth.

Usage

```
baseLineCalc(covData)
```

Arguments

`covData`	A vector with the numbers of sequences aligned at each genomic position to be considered in the analysis

Details

This function adds the numbers of sequences observed at each position and divides this number by the genome size (total number of nucleotides).

Value

The average number of reads mapped on each nucleotide in the genome.

Note

Detailed information and tutorials can be found online `http://bpeaks.gene-networks.net/`.

Author(s)

Gaelle LELANDAIS

References

`http://bpeaks.gene-networks.net/`

See Also

`bPeaksAnalysis`

Examples

```
# get library
library(bPeaks)

# get PDR1 data
data(dataPDR1)

# mean genome-wide read depth in IP data
meanIPcov = baseLineCalc(dataPDR1$IPdata[,3])
print(meanIPcov)

# mean genome-wide read depth in control data
meanContCov = baseLineCalc(dataPDR1$controlData[,3])
print(meanContCov)
```

---

bPeaksAnalysis *Function to run the entire bPeaks procedure*

Description

This function allows to detect basic peaks (bPeaks) using the procedure described in the function peakDetection. Chromosomes are analyzed successively. Several values(regarding thresholds T1, T2, T3 and T4 and other parameters) can be specified simultaneously in order to rapidly compare the obtained results and evaluate parameter relevance.

Usage

```
bPeaksAnalysis(IPdata, controlData, cdsPositions = NULL,
smoothingValue = 20,
                windowSize = 150, windowOverlap = 50,
IPcoeff = 2, controlCoeff = 2,
                log2FC = 2, averageQuantiles = 0.9,
resultName = "bPeaks",
peakDrawing = TRUE, promSize = 800, withoutOverlap = FALSE)
```

Arguments

IPdata	A dataframe with sequencing results of the IP sample. This dataframe has three columns (chromosome, position, number of sequences) and should have been created with the dataReading function
controlData	A dataframe with sequencing results of the control sample. This dataframe has three columns (chromosome, position, number of sequences) and should have been created with the dataReading function
cdsPositions	Not mandatory. A table (matrix) with positions of CDS (genes). Four columns are required (chromosome, starting position, ending position, strand (W or C), description). CDS positions for several yeast species are stored in bPeaks package (see the dataset yeastCDS and also peakLocation function)

`smoothingValue` The number (n/2) of surrounding positions to use for mean calculation in the `dataSmoothing` function

`windowSize` Size of the sliding windows to scan chromosomes

`windowOverlap` Size of the overlap between two successive windows

`IPcoeff` Threshold T1. Value for the multiplicative parameter that will be combined with the value of the mean genome-wide read depth (see `baseLineCalc`). As an illustration, if the IPcoeff = 6, it means that to be selected, the IP signal should be GREATER than 6 * (the mean genome-wide read depth). Note that a vector with different values can be specified, the bPeaks analysis will be therefore repeated using successively each value for peak detection

`controlCoeff` Threshold T2. Value for the multiplicative parameter that will be combined with the value of the mean genome-wide read depth (see `baseLineCalc`). As an illustration, if the controlCoeff = 2, it means that to be selected, the control signal should be LOWER than 2 * (the mean genome-wide read depth). Note that a vector with different values can be specified, the bPeaks analysis will be therefore repeated using successively each value for peak detection

`log2FC` Threshold T3. Threshold to consider log2(IP/control) values as sufficiently important to be interesting. Note that a vector with different values can be specified, the bPeaks analysis will be therefore repeated using successively each value for peak detection

`averageQuantiles`
Threshold T4. Threshold to consider (log2(IP) + log2(control)) / 2 as sufficiently important to be interesting. This parameter ensures that the analyzed genomic region has enough sequencing coverage to be reliable. These threshold should be between [0, 1] and refers to the quantile value of the global distribution observed with the analyzed chromosome

`resultName` Name for output files created during bPeaks procedure

`peakDrawing` TRUE or FLASE. If TRUE, the function `peakDrawing` is called and PDF files with graphical representations of detected peaks are created.

`promSize` Size of the genomic regions to be considered as "upstream" to the annotated genomic features (see documentation of the function `peakLocation` for more information).

`withoutOverlap` If TRUE, this option allows to filter peak that are located in a promoter AND a CDS.

Details

More information together with tutorials can be found online `http://bpeaks.gene-networks.net/`.

Value

BED files for each chromosomes and a final BED file combining all the results with information regarding detected peaks (genomic positions, mean IP signal, etc.). These files are all saved in the R working directory. Summaries of parameter calculations and peak detection criteria are shown in PDF files (saved in the working directory).

Note

Detailed information and tutorials can be found online http://bpeaks.gene-networks.net/

Author(s)

Gaelle LELANDAIS

References

http://bpeaks.gene-networks.net/

See Also

peakDetection dataReading dataSmoothing baseLineCalc peakDrawing peakLocation

Examples

```
# get library
library(bPeaks)

# STEP 1: get PDR1 data
data(dataPDR1)

# STEP 2 : bPeaks analysis (only 10 kb of chrIV are analyzed here,
#          as an illustration)
bPeaksAnalysis(IPdata = dataPDR1$IPdata[40000:50000,],
               controlData = dataPDR1$controlData[40000:50000,],
               windowSize = 150, windowOverlap = 50,
               IPcoeff = 4, controlCoeff = 2, log2FC = 1,
               averageQuantiles = 0.5,
               resultName = "bPeaks_example",
               peakDrawing = TRUE, promSize = 800)

## Not run:
# STEP 2 : bPeaks analysis (all chromosome)
bPeaksAnalysis(IPdata = dataPDR1$IPdata, controlData = dataPDR1$controlData,
                cdsPositions = dataPDR1$cdsPositions,
                smoothingValue = c(20),
                windowSize = c(150), windowOverlap = 50,
                IPcoeff = c(2), controlCoeff = c(2), log2FC = c(2),
                averageQuantiles = c(0.9),
                resultName = "bPeaks_PDR1_chr4",
                peakDrawing = TRUE, promSize = 800)

# To repeat the bPeaks analysis with different parameters
bPeaksAnalysis(IPdata = dataPDR1$IPdata, controlData = dataPDR1$controlData,
                cdsPositions = dataPDR1$cdsPositions,
                smoothingValue = c(20),
                windowSize = c(150), windowOverlap = 50,
                IPcoeff = c(2, 4, 6), controlCoeff = c(2, 4, 6), log2FC = c(2, 3),
                averageQuantiles = c(0.7, 0.9),
                resultName = "bPeaks_PDR1_chr4_paremeterEval",
```

```
                peakDrawing = FALSE, promSize = 800)

# -> Summary table is created and saved as "peakStats.Robject" in the working directory
# as well as a text file named "_bPeaks_parameterSummary.txt"...
load("peakStats.Robject")
# This table comprises different information regarding peak detection (number of peaks,
# mean size of peaks, mean IP signal, mean control signal, etc.)
peakStats[1:2,]

#     smoothingValue windowSize windowOverlap IPcoeff controlCoeff log2FC
# [1,]             20        150            50       1            1      1
# [2,]             20        150            50       1            1      1
#     averageQuantiles bPeakNumber meanSize meanIPsignal meanControlSignal
# [1,]              0.5         308  209.091      276.047            71.534
# [2,]              0.7         294  205.782      287.808            74.002
#     meanLog2FC bPeakNumber_beforeFeatures bPeakNumber_afterFeatures
# [1,]      1.571                         99                        80
# [2,]      1.589                         94                        77
#     bPeakNumber_inFeatures
# [1,]                     52
# [2,]                     53


## End(Not run)
```

---

dataPDR1 *ChIP-seq results (IP and control samples) obtained with the transcription factor Pdr1 in yeast Saccharomyces cerevisiae*

Description

ChIP-seq experiments were performed in order to identify the genomic regions that interact with the transcription factor Pdr1, in yeast Saccharomyces cerevisiae. Two samples (IP and control) were sequenced simultaneously using the Illumina technology (ENS transcriptome platform). Only the data for chrIV are available here, but complete datasets can be found online: `http://bpeaks.gene-networks.net`

Usage

```
data(dataPDR1)
```

Format

`dataPDR1$IPdata`: IPdata and controlData are dataframes with three columns. The first column comprises chromosome names, the second column comprises base positions and the third column comprises the numbers of sequences mapped at the considered position. `dataPDR1$controlData`: IPdata and controlData are dataframes with three columns. The first column comprises chromosome names, the second column comprises base positions and the third column comprises the numbers of sequences mapped at the considered position. `dataPDR1$cdsPositions`: A table with annotated positions of genes in yeast S. cerevisiae. The first column indicates chromosome names, the

second and third columns indicate respectively "start" and "end" positions of genes, and the fourth column indicates the gene annotation (according to the Saccharomyces Genome Database (SGD http://www.yeastgenome.org/)).

Details

Complete procedure to analyze sequencing data (intial FASTQ files) can be found online:

http://bpeaks.gene-networks.net. Initial read length was 50 bases. After quality controls and filtering of low quality bases, around 30.000.000 of sequences (IP sample) and around 88.000.000 of sequences (control sample) were independantly mapped on the genome using the bowtie algorithm [1]. Output files (SAM format) were converted into BAM files and indexed using the SAMTOOLS suite [2]. Numbers of sequences mapped on each nucleotide in the reference genome were finally calculated using the "genomeCoverageBed" tool available from the BEDTOOLS suite [3].

Source

Sample sequencing was performed at the "transcriptome platform", ENS institute in Paris (France), http://www.transcriptome.ens.fr.

References

More information concerning this dataset can found online : http://bpeaks.gene-networks.net.

[1] Langmead B, Trapnell C, Pop M, Salzberg SL. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol 10:R25.

[2] Li H.*, Handsaker B.*, Wysoker A., Fennell T., Ruan J., Homer N., Marth G., Abecasis G., Durbin R. and 1000 Genome Project Data Processing Subgroup (2009) The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics, 25, 2078 9. [PMID: 19505943]

[3] Quinlan AR and Hall IM, 2010. BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics. 26, 6, pp. 841 842.

Examples

```
# get library
library(bPeaks)

# get data
data(dataPDR1)

summary(dataPDR1)
#             Length Class      Mode
# IPdata       3      data.frame list
# controlData  3      data.frame list
# cdsPositions 5      data.frame list

# run peak calling, comparing IP and control samples
# (only 10 kb of chrIV are analyzed here, as an illustration)
bPeaksAnalysis(IPdata = dataPDR1$IPdata[40000:50000,],
               controlData = dataPDR1$controlData[40000:50000,],
               windowSize = 150, windowOverlap = 50,
               IPcoeff = 4, controlCoeff = 2,
```

```
       log2FC = 1, averageQuantiles = 0.5,
               resultName = "bPeaks_example",
               peakDrawing = TRUE, promSize = 800)

## Not run:
# -> bPeaks analysis, all chromosome IV and default parameters (optimized for yeasts)

# STEP 1: get PDR1 data (ChIP-seq experiments, IP and control samples,
#  related to the transcription factor Pdr1 in yeast Saccharomyces
#  cerevisiae)
data(dataPDR1)

# STEP 2: bPeaks analysis
bPeaksAnalysis(IPdata = dataPDR1$IPdata,
               controlData  = dataPDR1$controlData,
       cdsPositions = dataPDR1$cdsPositions,
               windowSize = 150, windowOverlap = 50,
               IPcoeff = 2, controlCoeff = 2,
               log2FC = 2, averageQuantiles = 0.9,
               resultName = "bPeaks_PDR1",
               peakDrawing = TRUE)

# STEP 3 : procedure to locate peaks according to
#     gene positions
peakLocation(bedFile = "bPeaks_PDR1_bPeaks_allGenome.bed",
            cdsPositions = yeastCDS$Saccharomyces.cerevisiae,
            outputName = "bPeakLocation_finalPDR1", promSize = 800)

# -> Note that cds (genes) positions are stored in bPeaks package for several yeast
# species
data(yeastCDS)

summary(yeastCDS)
#                          Length Class      Mode
# Debaryomyces.hansenii    31370  -none-     character
# Eremothecium.gossypii    23615  -none-     character
# Kluyveromyces.lactis     25380  -none-     character
# Pichia.sorbitophila      55875  -none-     character
# Saccharomyces.kluyveri   27790  -none-     character
# Yarrowia.lipolytica      32235  -none-     character
# Zygosaccharomyces.rouxii 24955  -none-     character
# Saccharomyces.cerevisiae     5  data.frame list
# Candida.albicans             5  data.frame list
# Candida.glabrata             5  data.frame list

# number of detected peaks
print(resLocation$numPeaks)

# number of peaks "upstream" annotated genes (W strand)
print(resLocation$beforeFeatures)

# number of peaks "in" annotated genes
print(resLocation$inFeatures)
```

```
# number of peaks "upstream" annotated genes (C strand)
print(resLocation$afterFeatures)

## End(Not run)
```

`dataReading` *Function to import sequencing results in R*

Description

Sequencing results should be converted to datafiles with numbers of sequences mapped on each nucleotide in the reference genome. These files can be generated from indexed BAM files (mapping results) using the "genomeCoverageBed" tool available from the BEDTOOLS suite [1]. More information concerning file descriptions can be found online:

`http://bpeaks.gene-networks.net/`.

File format should be as follows (chromosome, position, number of sequences):

chrI	1	4
chrI	2	4
chrI	3	4
chrI	4	4
chrI	5	7
chrI	6	7
chrI	7	9
chrI	8	9
chrI	9	10
chrI	10	13

Usage

```
dataReading(IPfile, controlFile, yeastSpecies = NULL)
```

Arguments

`IPfile`	Name of the file with sequencing results related to IP sample
`controlFile`	Name of the file with sequencing results related to control sample
`yeastSpecies`	Not mandatory. Annotations to be used for locations of peaks in promoters. Annotations of CDS are available in bPeaks for yeasts: Debaryomyces.hansenii, Eremothecium.gossypii, Kluyveromyces.lactis, Pichia.sorbitophila, Saccharomyces.kluyveri, Yarrowia.lipolytica, Zygosaccharomyces.rouxii, Saccharomyces.cerevisiae, Candida.albicans, Candida.glabrata (see data yeastCDS)

Details

To obtain a required file from a BAM file (resultFile.bam), the command line is (SHELL):

genomeCoverageBed -ibam resultFile.bam -d > resultFile.txt

More information concerning file conversions can be found online:

`http://bpeaks.gene-networks.net/`.

Value

A list with three elements ($IPdata, $controlData, $cdsPositions): IP data, control data and (if specified by user) CDS positions for locations of peaks in promoters.

Note

Conversion of file formats regarding sequencing results can be a tricky task. Detailed information can be found online `http://bpeaks.gene-networks.net/`. Don't hesitate to contact us for further discussions.

Author(s)

Gaelle LELANDAIS

References

[1] Quinlan AR and Hall IM, 2010. BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics. 26, 6, pp. 841 842.

See Also

`peakLocation` `dataPDR1` `yeastCDS`

Examples

```
# get library
library(bPeaks)

## Not run:
# Sequencing result files associated to PDR1 datasets (IP and control files)
# can be downloaded from our website http://bpeaks.gene-networks.net/.
# They are respectively named "IP_genomeCoverage.txt" (IP sample) and
# "INPUT_genomeCoverage.txt" (control sample).

# Import in R the sequencing results using S. cerevisiae CDS annotations.
data(yeastCDS)
seqResult = dataReading("IP_genomeCoverage.txt", "INPUT_genomeCoverage.txt",
yeastSpecies = yeastCDS$Saccharomyces.cerevisiae)

# IP data
seqResult$IPdata

# control data
seqResult$controlData

# run peak detection from IP and control data (with default parameters)
```

```
bPeaksAnalysis(IPdata = seqResult$IPdata,
       controlData  = seqResult$controlData,
       cdsPositions = seqResult$cdsPositions)

## End(Not run)
```

`dataSmoothing` *Function to smooth sequencing coverage along a chromosome*

Description

This function allows to obtain a smoothed signal of the genome-wide read depth. Simple moving average (SMA) procedure is used. At each genomic position, the sequencing coverage is replaced by the unweighted mean of the n surrounding positions (n/2 before and n/2 after).

Usage

```
dataSmoothing(vecData, widthValue = 20)
```

Arguments

`vecData`	A vector with the numbers of sequences aligned at each genomic position to be considered in the analysis
`widthValue`	The number (n/2) of surrounding positions to use for mean calculation

Details

Detailed information and tutorials can be found online `http://bpeaks.gene-networks.net/`.

Value

A vector with the smoothed signal. Note that the SMA procedure creates missing values at the beginning and at the end of the vector with the smoothed signal.

Note

Detailed information and tutorials can be found online `http://bpeaks.gene-networks.net/`.

Author(s)

Gaelle LELANDAIS

References

`http://bpeaks.gene-networks.net/`

See Also

`bPeaksAnalysis` `baseLineCalc`

Examples

```
# get data
data(dataPDR1)

# inital IP signal
iniIPsignal = dataPDR1$IPdata[,3]

par(mfrow = c(2,2))

# plot initial IP signal
plot(iniIPsignal[416900:417400], type = "h",
    xlab = "genomic position", ylab = "sequencing coverage",
    main = "IP sample (PDR1 data)\nno smoothing",
    col = "red")

# calculate and plot smoothed signal (widthValue = 5)
smoothedIPsignal = dataSmoothing(vecData = iniIPsignal, widthValue = 5)
plot(smoothedIPsignal[416900:417400], type = "h",
    xlab = "genomic position", ylab = "sequencing coverage",
    main = "IP sample (PDR1 data)\nsmoothing (widthValue = 5)",
    col = "pink")

# calculate and plot smoothed signal (widthValue = 10)
smoothedIPsignal = dataSmoothing(vecData = iniIPsignal, widthValue = 10)
plot(smoothedIPsignal[416900:417400], type = "h",
    xlab = "genomic position", ylab = "sequencing coverage",
    main = "IP sample (PDR1 data)\nsmoothing (widthValue = 10)",
    col = "pink")

# calculate and plot smoothed signal (widthValue = 20)
smoothedIPsignal = dataSmoothing(vecData = iniIPsignal, widthValue = 20)
plot(smoothedIPsignal[416900:417400], type = "h",
    xlab = "genomic position", ylab = "sequencing coverage",
    main = "IP sample (PDR1 data)\nsmoothing (widthValue = 20)",
    col = "pink")
```

`peakDetection` *Peak calling method, i. e. identification of genomic regions with a high density of sequences (reads)*

Description

bPeaks uses a sliding window to scan the genomic sequence. Four criterion define interesting regions: 1) a high number of reads in the IP sample (T1 = IPthreshold); 2) a low number of reads in the control sample (T2 = controlThreshold); 3) a high value of log(IP/control) (T3 = ratioThreshold) and 4) a good sequencing coverage (IP and control samples) (T4 = averageThreshold). Parameters for peak detection, therefore, rely on four threshold values (T1, T2, T3 and T4). Graphical outputs and BED files are provided allowing the user to rapidly assess the relevance of the chosen parameters.

Usage

```
peakDetection(IPdata, controlData, chrName, windowSize = 150, windowOverlap = 50,
              outputName = "bPeaks_results",
              baseLineIP = NULL, baseLineControl = NULL,
              IPthreshold = 6, controlThreshold = 4, ratioThreshold = 2,
              averageThreshold = 0.7, peakDrawing = TRUE)
```

Arguments

IPdata	A dataframe with sequencing results of the IP sample. This dataframe has three columns (chromosome, position, number of sequences) and should have been created with the dataReading function
controlData	A dataframe with sequencing results of the control sample. This dataframe has three columns (chromosome, position, number of sequences) and should have been created with the dataReading function
chrName	Name of the chromosome to be scanned with the sliding windows (to compare IP and control signals and detect interesting regions)
windowSize	Size of the sliding window to scan chromosomes
windowOverlap	Size of the overlap between two successive windows
outputName	Name for output files created during bPeaks procedure
baseLineIP	Value of the mean genome-wide read depth (IP sample). This value is calculated using the baseLineCalc function
baseLineControl	Value of the mean genome-wide read depth (control sample). This value is calculated using the baseLineCalc function
IPthreshold	Threshold T1. Threshold to consider IP signal as sufficiently important to be interesting. Note that these threshold is a multiplicative parameter that will be combined with the calculated "baseLineIP" (see before) value. As an illustration, if the IPthreshold = 6, it means that to be selected, the IP signal should be GREATER than 6 * baseLineIP
controlThreshold	Threshold T2. Threshold to consider control signal as sufficiently low to be interesting. Note that these threshold is a multiplicative parameter that will be combined with the calculated "baseLineControl" (see before) value. As an illustration, if the controlThreshold = 2, it means that to be selected, the control signal should remain LOWER than 2 * baseLineControl
ratioThreshold	Threshold T3. Threshold to consider log2(IP/control) values as sufficiently important to be interesting
averageThreshold	Threshold T4. Threshold to consider (log2(IP) + log2(control)) / 2 as sufficiently important to be interesting. These parameter is important to ensure that the analyzed genomic region has enough sequencing coverage to be reliable. These threshold should be between [0, 1] and refers to the quantile value of the global distribution observed with the analyzed chromosome
peakDrawing	TRUE or FLASE. If TRUE, the function peakDrawing is called and PDF files with graphical representations of detected peaks are created.

Details

Detailed description of the bPeaks procedure together with tutorials can be found online:

`http://bpeaks.gene-networks.net/`.

Value

A matrix with genomic positions of the detected peaks. Summaries of parameter calculations and peak detection criteria are shown in PDF files (saved in the working directory).

Note

Detailed information and tutorials can be found online `http://bpeaks.gene-networks.net/`. Don't hesitate to contact us for further discussions.

Author(s)

Gaelle LELANDAIS

References

`http://bpeaks.gene-networks.net/`

See Also

`dataReading` `baseLineCalc` `peakDrawing` `bPeaksAnalysis`

Examples

```
# get library
library(bPeaks)

# get PDR1 data
data(dataPDR1)

# combine IP and control data
allData = cbind(dataPDR1$IPdata, dataPDR1$controlData)
colnames(allData) = c("chr", "pos", "IPsignal", "chr", "pos", "controlSignal")

print("*********************************************")
# calculate baseline IP and control values
lineIP    = baseLineCalc(allData$IPsignal)
print(paste("Baseline coverage value in IP sample : ", round(lineIP, 3)))
lineControl = baseLineCalc(allData$controlSignal)
print(paste("Baseline coverage value in control sample : ", round(lineControl, 3)))
print("*********************************************")
print("")

# get list of chromosomes
chromNames = unique(allData[,1])

# start peak detection on the first chromosome
```

```
print("*******************************************")
print(paste("Starting analysis of chromosome ", chromNames[1]))

# information for one chromosome
subData = allData[allData[,1] == chromNames[1],]

# only 10 kb are analyzed here (as an illustration)
vecIP      = subData[40000:50000,3]
vecControl = subData[40000:50000,6]

# smooth of the data
smoothedIP    = dataSmoothing(vecData = vecIP, widthValue = 20)
smoothedControl = dataSmoothing(vecData = vecControl, widthValue = 20)

# peak detection
detectedPeaks = peakDetection(IPdata = smoothedIP, controlData = smoothedControl,
                chrName = as.character(chromNames[1]),
                windowSize = 150, windowOverlap = 50,
                outputName = paste("bPeaks_example_", chromNames[1], sep = ""),
                baseLineIP = lineIP, baseLineControl = lineControl,
                IPthreshold = 4, controlThreshold = 2,
                ratioThreshold = 1, averageThreshold = 0.5,
                peakDrawing = TRUE)

# print detected genomic positions
print(detectedPeaks)
```

---

peakDrawing *Function to draw graphical representations of genomic regions detected using bPeaks methodology*

Description

This function allows to create PDF files, with graphical representations of the detected basic peaks (bPeaks). Genomic regions are shown together with the values of the parameters used to detect the region.

Usage

```
peakDrawing(vecIP, vecControl, lineIP, lineControl, lineFC, lineAverage,
            posInf = 1, posSup = NULL, add = 10, title = "")
```

Arguments

vecIP	Vector with sequencing depth at each nucleotide (from start pos = 1 to end)
vecControl	Vector with sequencing depth at each nucleotide (from start pos = 1 to end)
lineIP	Threshold values used for peak detection (IP signal)
lineControl	Threshold values used for peak detection (control signal)

lineFC	Threshold values used for peak detection (log2(IP/control) values)
lineAverage	Threshold values used for peak detection (average log2(IP) and log2(control) values)
posInf	Genomic position to start the representation
posSup	Genomic position to end the representation
add	Number of bases before and after posInf and posSup to add
title	Graphic main title

Details

More information can be found online: http://bpeaks.gene-networks.net/.

Value

Image in x11() terminal

Note

Detailed information and tutorials can be found online http://bpeaks.gene-networks.net/.

Author(s)

Gaelle LELANDAIS

References

http://bpeaks.gene-networks.net/

See Also

bPeaksAnalysis

Examples

```
# get library
library(bPeaks)

# get PDR1 data
data(dataPDR1)

# IP signal (smoothed) - Chromosome IV
IPsignal = dataSmoothing(dataPDR1$IPdata[dataPDR1$IPdata[,1] == "chrIV",3], 20)
# control signal (smoothed)
controlSignal = dataSmoothing(dataPDR1$controlData[dataPDR1$controlData[,1] == "chrIV",3], 20)

# draw all chromosome
peakDrawing(vecIP = IPsignal, vecControl = controlSignal,
            lineIP = 0, lineControl = 0, lineFC = 0, lineAverage = 0,
            posInf = 465000, posSup = 465550,
            add = 10, title = "PDR1 data - chromosome #4")
```

peakLocation *Function to locate detected basic peaks (bPeaks) according to predifined chromosomal features*

Description

Starting from a BED file with positions of detected peaks and a table with positions of CDS (genes), this function allows to dentify the peaks that are located "upstream" or "in" annotated CDS. Annotations of CDS for different yeast species are available in bPeaks package (see data yeastCDS).

Usage

```
peakLocation(bedFile, cdsPositions, withoutOverlap = FALSE,
            outputName = "bPeaksLocation", promSize = 800)
```

Arguments

bedFile	Name of a BED file with positions of detected peaks (using bPeaks or another program)
cdsPositions	A table (matrix) with positions of CDS (genes). Four columns are required (chromosome, starting position, ending position, strand (W or C), description)
withoutOverlap	If TRUE, this option allows to filter peak that are located in a promoter AND a CDS.
outputName	Name for output files
promSize	Genomic size to be considered as promoter (upstream to CDS)

Details

More information can be found online http://bpeaks.gene-networks.net/.

Value

Graphics and text files (saved in the R working directory).

Note

Detailed information and tutorials can be found online http://bpeaks.gene-networks.net/.

Author(s)

Gaelle LELANDAIS

References

http://bpeaks.gene-networks.net/

See Also

bPeaksAnalysis dataReading yeastCDS

Examples

```
## Not run:
# -> bPeaks analysis with (all chromosome and default parameters optimized for yeasts)

# STEP 1: get PDR1 data and annotations in yeasts
data(dataPDR1)
data(yeastCDS)

# STEP 2: bPeaks analysis
bPeaksAnalysis(IPdata = dataPDR1$IPdata,
               controlData = dataPDR1$controlData,
               windowSize = 150, windowOverlap = 50,
               IPcoeff = 6, controlCoeff = 4,
               log2FC = 2, averageQuantiles = 0.9,
               resultName = "bPeaks_PDR1",
               peakDrawing = TRUE)

# STEP 3 : procedure to locate peaks according to
# predefined chromosomal features
peakLocation(bedFile = "bPeaks_PDR1_bPeaks_allGenome.bed",
             cdsPositions = yeastCDS$Saccharomyces.cerevisiae,
    withoutOverlap = FALSE,
             outputName = "bPeakLocation_finalPDR1", promSize = 800)

## End(Not run)
```

---

yeastCDS *Annotations of CDS for different yeast species*

Description

Annotations of CDS for different yeast species. Data were collected (July 2013) from: Genolevures http://genolevures.org/, SGD http://www.yeastgenome.org/, CGD http://www.candidagenome.org/ databases.

Usage

```
data(yeastCDS)
```

Format

yeastCDS$Saccharomyces.cerevisiae: A table (matrix) with positions of CDS (genes) in yeast S. cerevisiae. Four columns are required (chromosome, starting position, ending position, strand (W or C), description).

Details

This R object is a list composed of multiple tables (matrices) - one for each species - with positions of CDS (genes).

Source

Data were collected (July 2013) from Genolevures http://genolevures.org/, SGD http://www.yeastgenome.org/ and CGD http://www.candidagenome.org/ databases.

References

Genolevures database http://genolevures.org/

SGD database http://www.yeastgenome.org/

CGD database http://www.candidagenome.org/

Examples

```
# get library
library(bPeaks)

# get data
data(yeastCDS)

# different species for wich information is available
summary(yeastCDS)
#                          Length Class      Mode
# Debaryomyces.hansenii    31370  -none-     character
# Eremothecium.gossypii    23615  -none-     character
# Kluyveromyces.lactis     25380  -none-     character
# Pichia.sorbitophila      55875  -none-     character
# Saccharomyces.kluyveri   27790  -none-     character
# Yarrowia.lipolytica      32235  -none-     character
# Zygosaccharomyces.rouxii 24955  -none-     character
# Saccharomyces.cerevisiae     5  data.frame list
# Candida.albicans             5  data.frame list
# Candida.glabrata             5  data.frame list

# CDS annotations for yeast Debaryomyces hansenii
yeastCDS$Debaryomyces.hansenii[1:10,]
#      chrm     start   end     strand geneName
# [1,] "Deha2A" "2023"  "6370"  "C"    "DEHA2A00110g"
# [2,] "Deha2A" "6587"  "7810"  "C"    "DEHA2A00132g"
# [3,] "Deha2A" "8314"  "9354"  "W"    "DEHA2A00154g"
# [4,] "Deha2A" "9632"  "9844"  "C"    "DEHA2A00176g"
# [5,] "Deha2A" "13806" "14132" "W"    "DEHA2A00198g"
# [6,] "Deha2A" "14558" "16519" "C"    "DEHA2A00220g"
# [7,] "Deha2A" "17520" "19442" "W"    "DEHA2A00242g"
# [8,] "Deha2A" "22619" "23977" "C"    "DEHA2A00264g"
# [9,] "Deha2A" "24949" "25434" "W"    "DEHA2A00286g"
#[10,] "Deha2A" "26440" "26640" "C"    "DEHA2A00308g"
```

Index