ACME
April 19, 2009

aGFF-class  

Class for storing GFF-like data

Description
The GFF format is quite versatile while remaining simple. This class simply stores the annotation associated with a set of GFF files from the same regions of the genome along with some information about the samples from which the data came and the data (from the "score" column of the GFF file) themselves.

Objects from the Class
Objects can be created by calls of the form `new("aGFF", ...)`. Also, the `\code{read.resultsGFF()}` function returns aGFF objects.

Slots
- **annotation**: Object of class "\code{data.frame}" with two columns absolutely necessary, "Chromosome" and "Location". Other columns can be included.
- **data**: Object of class "\code{matrix}" of the same number of rows as the annotation slot and the same number of columns as the number of rows in the samples slot, containing data for later analysis
- **samples**: Object of class "\code{data.frame}" for describing the samples, one row per sample

Methods
- **\code{plot} signature(x = "aGFF")**: to plot a region along the genome.
- **\code{print} signature(x = "aGFF")**: simple method to display summary of aGFF object
- **\code{show} signature(object = "aGFF")**: simple method to display summary of aGFF object

Author(s)
Sean Davis

See Also
- `\code{read.resultsGFF}` and `\code{aGFFCalc-class}`
# Load an example
data(example.agff)
example.agff

## aGFFCalc-class

### Class "aGFFCalc"

#### Description

Store results of ACME calculations

#### Objects from the Class

Objects can be created by calls of the form `new("aGFFCalc", ...)`.

#### Slots

- **call**: Object of class "call", contains the exact call to `do.aGFF.calc`, for historical purposes
- **threshold**: Object of class "numeric", the threshold used in the calculation
- **cutpoints**: Object of class "numeric", the data value above which probes were considered positive
- **vals**: Object of class "matrix", equivalent in size to the original data matrix, containing the calculated p-values from the ACME algorithm
- **annotation**: Object of class "data.frame", currently a copy of the original annotation, possibly reordered in chromosome order
- **data**: Object of class "matrix", the original data, possibly reordered
- **samples**: Object of class "data.frame", sample metadata

#### Extends

Class "aGFF", directly.

#### Methods

- **plot** signature(x = "aGFFCalc", ask=FALSE): plot the results of an ACME calculation
- **print** signature(x = "aGFFCalc"): brief overview of the object
- **show** signature(object = "aGFFCalc"): brief overview of the object

#### Author(s)

Sean Davis <sdavis2@mail.nih.gov>

#### See Also

`do.aGFF.calc`, `aGFF-class`
**do.aGFF.calc**

**Examples**

```r
data(example.agff)
example.agffcalc <- do.aGFF.calc(example.agff, window=1000, thresh=0.9)
example.agffcalc
```

---

**Description**

This function performs the moving window chi-square calculation. It is written in C, so is quite fast.

**Usage**

```r
doaGFF.calc(x, window, thresh)
```

**Arguments**

- `x`: An aGFF class object
- `window`: An integer value, representing the number of basepairs to include in the windowed chi-square calculation
- `thresh`: The quantile of the data distribution for each sample that will be used to classify a probe as positive

**Details**

A window size on the order of 2-3 times the average size of fragments from sonication, digestion, etc. and containing at least 8-10 probes is the recommended size. Larger size windows are probably more sensitive, but obviously reduce the accuracy with which boundaries of signal can be called.

A threshold of between 0.9 and 0.99 seems empirically to be adequate. If one plots the histogram of data values and there is an obvious better choice (such as a bimodal distribution, with one peak representing enrichment), a more data-driven approach may yield better results.

**Value**

An object of class `aGFFCalc`

**Author(s)**

Sean Davis <sdavis2@mail.nih.gov>

**Examples**

```r
data(example.agff)
example.agffcalc <- do.aGFF.calc(example.agff, window=1000, thresh=0.9)
example.agffcalc
```
example.agff  

An example ACME data structure of class aGFF

Description

An aGFF data structure from two Nimblegen arrays, custom tiled to include multiple HOX genes.

Usage

data(example.agff)

Format

The format is: chr 'example.agff'

Source

From Scacheri et al., Plot Genet, 2006. Pubmed ID 16604156

Examples

data(example.agff)
example.agff

findClosestGene  

Find closest refseq gene

Description

This function is used to find the nearest refseq transcript(s) to a point in the genome specified. Note that it is limited to the refseq transcripts listed at genome.ucsc.edu, where this function goes for information.

Usage

findClosestGene(chrom, pos, genome = "hg17", position = "txStart")

Arguments

chrom         Usually specified like 'chr1', 'chr2', etc.
pos          A position in base pairs in the genome
genome        Something like 'hg16', 'hg17', 'mm6', etc.
position      The location to measure distance from: one of 'txStart', 'txEnd', 'cdsStart', 'cdsEnd'

Details

The first time the function is run, it checks to see if the refflat table for the given genome is present in the package environment. If not, it downloads it to the /tmp directory and gunzips it (using getRefflat). It is then stored so that in future calls, there is no re-download required.
findRegions

Value

A data frame with the gene name, refseq id(s), txStart, txEnd, cdsStart, cdsEnd, exon count, and distance. Note that distance is measured as pos-position, so negative values mean that the point in the gene is to the left of the point specified in the function call (with the p-tel on the left).

Note

The function may return more than one transcript, as several transcripts may have the same start site

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

Examples

findClosestGene('chr1',100000000,'hg17')

findRegions

Find all regions in data above p-value threshold

Description

After the ACME calculation, each probe is associated with a p-value of enrichment. However, one often wants the contiguous regions associated with runs of p-values above a given p-value threshold.

Usage

findRegions(calcobj, thresh = 1e-04)

Arguments

calcobj An aGFFCalc object
thresh The p-value threshold

Details

Runs of p-values above the p-value threshold will be reported as one "region". These can be used for downstream analyses, export to browsers, submitted for transcription factor binding enrichment analyses, etc.

Value

A data frame with these columns:

Length The length of the region in probes
TF Either TRUE or FALSE; TRUE regions represent regions of enrichment while FALSE regions are the regions between the TRUE regions
StartInd The starting Index of the region
EndInd The ending Index of the region
Sample The sample containing the region
getRefflat

<table>
<thead>
<tr>
<th>Chromosome</th>
<th>The Chromosome of the region</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>The starting basepair of the region</td>
</tr>
<tr>
<td>End</td>
<td>The ending basepair of the region</td>
</tr>
</tbody>
</table>

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

See Also

do.aGFF.calc, findClosestGene

Examples

data(example.agff)
exampagffcalc <- do.aGFF.calc(example.agff, window=1000, thresh=0.9)
foundregions <- findRegions(example.agffcalc, thresh=0.001)
foundregions[1:6,]

grefflat

Get the refflat table from ucsc for the given genome

Description

Fetches the refflat table from ucsc, stores in temp dir and then gunzips it and reads it in.

Usage

grefflat(genome = "hg17")

Arguments

genome The genome code from ucsc, like 'hg16', 'mm6', etc.

Value

A data frame mirroring the UCSC table structure.

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

References

http://genome.ucsc.edu

See Also

findClosestGene

Examples

rf <- getRefflat('hg17')
read.resultsGFF  

Read Nimblegen GFF files

Description

A GFF format file is a quite flexible format for storing genomic data. Nimblegen uses these format files as one format for making chip-chip data available. This function reads these files, one per experiment and creates a resulting aGFF-class object.

Usage

read.resultsGFF(fnames, path = ".", samples = NULL, notes = NULL, skip = 0, sep = \t, quote = \""

Arguments

fnames  
A vector of filenames

path  
The path to the filenames

samples  
A data.frame containing sample information, one row per sample, in the same order as the files in fnames

notes  
A character vector for notes–not currently stored

skip  
Number of lines to skip if the file contains a header

sep  
The field separator–should be a tab character for gff files, but can be set if necessary.

quote  
The text quote character–again not used for gff file, typically

Details

The output is an aGFF object.

Value

A single aGFF object.

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

References

http://www.sanger.ac.uk/Software/formats/GFF/

See Also

aGFF-class

Examples

datdir <- system.file('extdata',package='ACME')
fnames <- dir(datdir)
exdir <- read.resultsGFF(fnames,path=datdir)
write.sgr  

Write Affy IGB .sgr format files

Description

The affy Integrated Genome Browser (IGB) is a powerful, fast browser for genomic data. The file format is simple (three columns: chromosome, location, and score) to generate. This function will write the sgr files associated with a aGFFcalc object. There will be either one or two files (default two) representing the raw data and the calculated data (which is output as -log10(val) for visualization purposes).

Usage

```
write.sgr(agff, raw = TRUE, vals = TRUE, directory = ".")
```

Arguments

- `agff`: An aGFFCalc object obtained after running do.aGFF.calc
- `raw`: Create a file for the raw data?
- `vals`: Create a file for the calculated p-values?
- `directory`: Give a directory for storing the files

Author(s)

Sean Davis

Examples

```
data(example.agff)
write.sgr(example.agff)
```
# Index

*Topic IO
- findClosestGene, 4
- getRefflat, 6
- read.resultsGFF, 7
- write.sgr, 8

*Topic classes
- aGFF-class, 1
- aGFFCalc-class, 2

*Topic datasets
- example.agff, 4

*Topic htest
- do.aGFF.calc, 3

*Topic manip
- findRegions, 5
- read.resultsGFF, 7

aGFF-class, 2, 7
aGFF-class, 1
aGFFCalc-class, 1
aGFFCalc-class, 2

do.aGFF.calc, 2, 3, 6
example.agff, 4
findClosestGene, 4, 6
findRegions, 5
getRefflat, 4, 6

plot, aGFF-method(aGFF-class), 1
plot, aGFFCalc-method(aGFFCalc-class), 2
print, aGFF-method(aGFF-class), 1
print, aGFFCalc-method(aGFFCalc-class), 2

read.resultsGFF, 1, 7
show, aGFF-method(aGFF-class), 1
show, aGFFCalc-method(aGFFCalc-class), 2

write.sgr, 8