

# Package ‘davidTiling’

April 4, 2014

**Version** 1.2.19

**Title** Data and analysis scripts for David, Huber et al. yeast tiling array paper

**Author** Wolfgang Huber <huber@ebi.ac.uk>, Joern Toedling <toedling@ebi.ac.uk>

**Maintainer** Wolfgang Huber <huber@ebi.ac.uk>

**Depends** R (>= 2.10), Biobase (>= 2.5.5), tilingArray, GO.db

**Description** This package contains the data for the paper by L. David et al. in PNAS 2006 (PMID 16569694): 8 CEL files of Affymetrix genechips, an ExpressionSet object with the raw feature data, a probe annotation data structure for the chip and the yeast genome annotation (GFF file) that was used. In addition, some custom-written analysis functions are provided, as well as R scripts in the scripts directory.

**biocViews** ExperimentData, Yeast

**Reference** A high-resolution map of transcription in the yeast genome.  
David L, Huber W, Granovskaia M, Toedling J, Palm CJ, Bofkin L, Jones T, Davis RW, Steinmetz LM. Proc Natl Acad Sci U S A. 2006  
Apr 4;103(14):5320-5.

**License** LGPL

**URL** <http://www.ebi.ac.uk/huber>

## R topics documented:

data:davidTiling . . . . .	2
data:gff . . . . .	2
data:probeAnno . . . . .	3
getAllGO . . . . .	4
getAttributeField . . . . .	5
GOHyperG . . . . .	6
scatterWithHist . . . . .	7
scoreSegments . . . . .	8
showDens . . . . .	9
yeastFeatures . . . . .	10

**Index****11**

---

data:davidTiling      *Dataset of class eSet with the raw 'CEL file' intensities*

---

**Description**

The data are from an experiment that used Affymetrix Scerevisiaetiling chips from 2004, which were custom-made for the Stanford Genome Center. The chips tile the complete genome of *S. cerevisiae* in steps of 8 bases, separately for each strand of each chromosome. The two tiles for one chromosome (Watson and Crick strands) are offset by 4 bases.

Note that the class `eSet` was used instead of `AffyBatch` since the additional overhead of 'CDF environments' in the latter is not needed here.

**Usage**

```
data("davidTiling")
```

**Format**

Intensity data for 8 arrays. The `phenoData` slot contains the file names and the nucleic acid type.

**Author(s)**

W. Huber <huber@ebi.ac.uk>

**Source**

Lior David and Lars Steinmetz, both from the Stanford Genome Center. Lars Steinmetz is also at EMBL Heidelberg.

**Examples**

```
data("davidTiling")
dim(exprs(davidTiling))
```

---

data:gff      *Genomic features of Saccharomyces cerevisiae*

---

**Description**

A data frame with genomic features of *Saccharomyces cerevisiae*.

**Usage**

```
data("gff")
```

**Format**

Object of class `data.frame`. GFF is a file format for annotating genomes, see <insert the URL to the documentation page for GFF at Sanger here>. The format is essentially a rectangular table, and here it is represented as a data frame.

**Author(s)**

W. Huber <huber@ebi.ac.uk>

**Source**

Two GFF files were downloaded: `saccharomyces_cerevisiae.gff` from `ftp://genome-ftp.stanford.edu/pub/yeast/` on 7 Aug 2005, 18:16 BST, and `IGR_v24.2.p001.allowoverlap.GFF` from `http://jura.wi.mit.edu/fraenkel/download/` upon suggestion from the SGD curators on 30 Aug 2005. (Future versions of SGD's GFF files are likely to include the latter as well). They were parsed, combined and written into the `gff` `data.frame` with the script `makeProbeAnno.R` in the `inst/scripts` directory of this package.

**Examples**

```
data("gff")
str(gff)
```

---

data:probeAnno	<i>An environment with probe mapping information for the Scerevisiae tiling array</i>
----------------	---

---

**Description**

The environment contains probe mapping information for the Affymetrix Scerevisiaetiling chip from 2004, which was custom-made for the Stanford Genome Center. The chips tile the complete genome of *S. cerevisiae* in steps of 8 bases, separately for each strand of each chromosome. The two tiles for one chromosome (Watson and Crick strands) are offset by 4 bases.

In the following a brief description of the 138 elements of the `probeAnno` environment.

`probeReverse`: a list of 8 factors, each of length 6553600, corresponding to the rows of [davidTiling](#). For example, if the probe corresponding to the *j*-th row in [davidTiling](#) maps to the coding sequence of a gene, then the factor level of `probeReverse$CDS[j]` is the name of the gene, and the empty string "" otherwise. This applies to samples that were hybridized to the chip after a reverse transcription step.

`probeDirect`: analogous to `probeReverse`, but for samples that were hybridized to the chip without a reverse transcription step. The probes map to the opposite chromosomal strand compared to experiments with reverse transcription.

`1.+ .index`: indices (from 1..6553600, corresponding to the rows of [davidTiling](#)) of probes mapping to the Watson strand of chromosome 1.

`1.+ .start`, `1.+ .end`: start and end positions in genomic coordinates of the alignments of the probes (in the same order as in `1.+ .index`) to the Watson strand of chromosome 1. For 25-mers, the values in `1.+ .end` are those in `1.+ .start` plus 24, but not all probes on the array are 25-mers.

1.-.unique: specificity of the probe:

**0** has exactly one perfect match (PM) and no near-matches in the genome

**1** has exactly one PM and some near-matches

**2** has no PM but one or more near-matches

**3** has multiple PMs in the genome

1.-.index, 1.-.start, 1.-.end, 1.-.unique: analogous to the above, but for the Crick strand of chromosome 1. 2.+ .index, 2.+ .start, 2.+ .end, 2.-.unique: analogous to the above, but for the Watson strand of chromosome 2; and so forth. "Chromosome 17" is mitochondrial DNA.

### Usage

```
data("probeAnno")
```

### Author(s)

W. Huber <huber@ebi.ac.uk>

### Source

Probe sequences were obtained from Affymetrix in a file called `S.cerevisiae_tiling.11q`. The genomic sequences of the *S. cerevisiae* chromosomes were downloaded from `ftp://genome-ftp.stanford.edu/pub/yeast` on 7 Aug 2005, 18:16 BST in 17 files `chr01.fsa`–`chr16.fsa`, and `chrmt.fsa`. The probe sequences were matched against the chromosomal sequences with the program MUMmer, see the script `mapProbesToGenome.sh` (in the `inst/scripts` directory of this package). MUMmer results were parsed and processed into the `probeAnno` environment with the script `makeProbeAnno.R` (in the `inst/scripts` directory of this package).

### Examples

```
data("probeAnno")
ls(probeAnno)
str(probeAnno$"1.+ .start")
```

---

getAllGO

*Get all GO categories for a list of genes.*

---

### Description

The function uses the GO categories in the data.frame `gff` to obtain annotated GO categories, then the `GO**ANCESTOR` data in the `GO` package to add all parent terms as well.

### Usage

```
getAllGO(x, gff)
```

**Arguments**

x	character vector.
gff	data.frame with columns feature, Name, and (Ontology_term or attributes), see details

**Details**

The elements of x are matched against the column gene in gff. All are required to match. A list of GO terms is then extracted from the corresponding rows in the Ontology\_term column. A gene may be annotated by several terms, separated by ",". Then the GO package is used to augment this by all ancestor terms.

**Value**

List of character vectors.

**Author(s)**

W. Huber <huber@ebi.ac.uk>

---

getAttributeField      *Extract the value of a certain field out of a character vector such as in the "attributes" column of a GFF table.*

---

**Description**

Extract the value of a certain field out of a character vector such as in the "attributes" column of a GFF table.

**Usage**

```
getAttributeField(x, field, attrsep=";")
```

**Arguments**

x	character vector.
field	character vector of length 1, containing the field name.
attrsep	character vector of length 1, containing the separator name.

**Details**

See example.

**Value**

Character vector.

**Author(s)**

W. Huber <huber@ebi.ac.uk>

**Examples**

```
acol = c("ID=46891;Name=TEL01L-TR;Note=Bla",
        "ID=46892;Name=TEL01L;Note=Di",
        "ID=46893;Name=TEL01L-XR;Note=Bla")

getAttributeField(acol, "Name")
getAttributeField(acol, "ID")
```

---

GOHyperG

*Hypergeometric test for GO category enrichment.*

---

**Description**

This function is adapted from the function of the same name in G0stats package. Main difference is that it draws the GO annotations of the genes in candidates from the data.frame gff. It also draws a plot.

**Usage**

```
GOHyperG(candidates, gff, plotmain)
```

**Arguments**

candidates	character vector
gff	see <a href="#">getAllGO</a>
plotmain	character, plot title

**Details**

The elements of `x` are matched against the column `gene` in `gff`. All are required to match. A list of GO terms is then extracted from the corresponding rows in the `Ontology_term` column. A gene may be annotated by several terms, separated by `","`. Then the GO package is used to augment this by all ancestor terms.

**Value**

List of character vectors.

**Author(s)**

W. Huber <huber@ebi.ac.uk>

**See Also**

[getAllGO](#)

---

scatterWithHist	<i>Scatterplot with histograms of marginal distributions</i>
-----------------	--

---

### Description

Scatterplot with histograms of marginal distributions.

### Usage

```
scatterWithHist(x, breaks, barcols, xlab, ylab, ...)
```

### Arguments

x	numeric matrix with 2 columns.
breaks	numeric vector with histogram breaks, see <a href="#">hist</a> .
barcols	character vector of length 2, colors for the histogram filling.
xlab	character of length 1, label for x-axis.
ylab	character of length 1, label for y-axis.
...	further arguments that get passed on to <a href="#">plot</a> .

### Value

The function is called for its side effect.

### Author(s)

W. Huber <huber@ebi.ac.uk>

### Examples

```
x = rexp(100)
x = cbind(x, x+0.6*rnorm(length(x)))
scatterWithHist(x,
  breaks=seq(min(x),max(x),length=20),
  barcols=c("mistyrose", "lightblue"),
  xlab="Daffodil", ylab="Petunia", pch=16)
```

---

scoreSegments	<i>Score segments</i>
---------------	-----------------------

---

## Description

Score the segments found by a previous call to `findSegments` by comparing to genome annotation

## Usage

```
scoreSegments(s, gff,  
             nrBasePerSeg = 1500,  
             probeLength = 25,  
             params = c(overlapFraction = 0.5, oppositeWindow = 100, flankProbes=10),  
             verbose = TRUE)
```

## Arguments

<code>s</code>	environment. See details.
<code>gff</code>	GFF dataframe.
<code>nrBasePerSeg</code>	Numeric of length 1. This parameter determines the number of segments.
<code>probeLength</code>	Numeric of length 1.
<code>params</code>	vector of additional parameters, see details.
<code>verbose</code>	Logical.

## Details

This function scores segments. It is typically called after a *segmentation*. For an example segmentation script, see the script `segment.R` in the `scripts` directory of this package. For an example scoring script, which loads the data and then calls this function, see the script `scoreSegments.R`.

## Value

A dataframe with columns as described in the details section.

## Author(s)

W. Huber <huber@ebi.ac.uk>



---

`showDens`*Plot function for more than one density*

---

**Description**

Plot function for more than one density

**Usage**

```
showDens(z, breaks, xat, xtickLabels=paste(xat), col, ylab = "", ...)
```

**Arguments**

<code>z</code>	List: numeric vectors for computing histograms for
<code>breaks</code>	Numeric vector: breaks of the histogram
<code>xat</code>	Numeric vector: where to put the x-axis ticks
<code>xtickLabels</code>	Character vector: what to write underneath them
<code>col</code>	Character vector: colours of the histograms
<code>ylab</code>	Character scalar: y-axis label
<code>...</code>	further arguments passed on to plot

**Details**

...

**Value**

returns scale factor

**Author(s)**

Wolfgang Huber <huber@ebi.ac.uk>

**See Also**

[hist](#)

**Examples**

```
showDens(list(x1=runif(100), x2=exp(runif(50))-1, x3=runif(20)),
  breaks=seq(0, 2, 0.2), xat=seq(0, 2, 0.5), col=rainbow(3), xlab="Random Numbers")
```

---

`yeastFeatures`*A data.frame with metadata about the features in SGD GFF files*

---

**Description**

The rows of the data frame correspond to feature categories, such as "gene", "CDS", "telomere". The column `isTranscribed` is a logical vector that denotes whether this feature category is considered to be transcribable.

**Usage**

```
data(yeastFeatures)
```

**Format**

A `data.frame`

**Author(s)**

W. Huber <huber@ebi.ac.uk>

**Examples**

```
data(yeastFeatures)
```

# Index

## \*Topic **datasets**

data:davidTiling, 2

data:gff, 2

data:probeAnno, 3

yeastFeatures, 10

## \*Topic **hplot**

scatterWithHist, 7

showDens, 9

## \*Topic **manip**

getAllGO, 4

getAttributeField, 5

GOHyperG, 6

scoreSegments, 8

AffyBatch, 2

data:davidTiling, 2

data:gff, 2

data:probeAnno, 3

davidTiling, 3

davidTiling (data:davidTiling), 2

eSet, 2

getAllGO, 4, 6

getAttributeField, 5

gff (data:gff), 2

GOHyperG, 6

hist, 7, 9

phenoData, 2

plot, 7

probeAnno (data:probeAnno), 3

scatterWithHist, 7

scoreSegments, 8

showDens, 9

yeastFeatures, 10