

# Package ‘harbChIP’

October 7, 2015

**Title** Experimental Data Package: harbChIP

**Description** data from a yeast ChIP-chip experiment

**Version** 1.6.0

**Author** VJ Carey

**Maintainer** VJ Carey <stvjc@channing.harvard.edu>

**Depends** R (>= 2.10.0), tools, utils, IRanges, Biobase (>= 2.5.5),  
Biostrings

**Imports** methods, stats

**License** Artistic-2.0

**biocViews** ExperimentData, Saccharomyces\_cerevisiae\_Data,  
SequencingData

**NeedsCompilation** no

## R topics documented:

allhex . . . . .	1
buildUpstreamSeqs2 . . . . .	2
chkMotif4TF . . . . .	3
harbChIP . . . . .	4
sceUpstr . . . . .	5
upstreamSeqs-class . . . . .	5

<b>Index</b>	<b>7</b>
--------------	----------

---

allhex	<i>utility function: get all hexamers in upstream sequence for an ORF</i>
--------	---

---

## Description

utility function: get all hexamers in upstream sequence for an ORF

**Usage**

```
allhex(orf, usobj)
```

**Arguments**

```
orf          character string, ORF name
usobj        upstreamSeqs object
```

**Details**

computes Biostrings Views

**Value**

computes Biostrings Views

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**Examples**

```
data(sceUpstr)
allhex("YAL001C", sceUpstr)
```

---

buildUpstreamSeqs2     *workflow component – build an upstreamSeqs instance from a FASTA read*

---

**Description**

workflow component – build an upstreamSeqs instance from a FASTA read

**Usage**

```
buildUpstreamSeqs2(fastaRead, organism="sce", provenance="harmen")
```

**Arguments**

```
fastaRead    results of a readFASTA from Biostrings
organism     string naming organism
provenance   string or structure describing provenance
```

**Details**

generates an instance of upstreamSeqs

**Value**

generates an instance of upstreamSeqs

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**Examples**

```
# x = readFASTA(...)
# y = buildUpstreamSeqs2(x)
```

---

chkMotif4TF	<i>analyze relationship between motif frequency and binding intensity for selected motif and TF</i>
-------------	---

---

**Description**

analyze relationship between motif frequency and binding intensity for selected motif and TF

**Usage**

```
chkMotif4TF(motif, TF, chset, upstr, bthresh=2, countthresh=0)
```

**Arguments**

motif	character string in alphabet known to Biostrings
TF	name of a TF (sample name in the ChIP-chip data structure chset)
chset	an ExpressionSet instance harboring ChIP-chip data
upstr	an instance of upstreamSeqs
bthresh	threshold for binding intensity results to declare TF 'bound' to the upstream region
countthresh	threshold for motif count to be considered 'present' in upstream region

**Details**

Uses [countPattern](#) to perform motif count.

**Value**

a list with elements `call`, `table`, and `test`, the latter providing the result of [fisher.test](#)

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

## Examples

```
# slow
## Not run:
data(harbChIP)
data(sceUpstr)
chkMotif4TF("CGGCCG", "RDS1", harbChIP, sceUpstr)

## End(Not run)
```

---

harbChIP

*Experimental Data Package: harbChIP*

---

## Description

binding ratios and intergenic region data from a ChIP-chip experiment in yeast

## Usage

```
data(harbChIP)
```

## Format

The format is: An ExpressionSetObject with covariates:

- txFac: transcription factor symbol from Harbison website CSV file columnnames

## Note

derived from web site [jura.wi.mit.edu/young\\_public/regulatory\\_code/GWLD.html](http://jura.wi.mit.edu/young_public/regulatory_code/GWLD.html), binding ratios

## Examples

```
data(harbChIP)
harbChIP
experimentData(harbChIP)
exprs(harbChIP)[1:6,1:7]
dim(exprs(harbChIP))
pData(featureData(harbChIP))[1:6,]
```

---

`sceUpstr`*Biostrings representations of S. cerevisiae upstream regions*

---

**Description**

Biostrings representations of S. cerevisiae upstream regions

**Usage**

```
data(sceUpstr)
```

**Details**

environment-based S4 object with DNAstring elements

**Value**

environment-based S4 object with DNAstring elements

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**Examples**

```
data(sceUpstr)
sceUpstr
getUpstream("YAL001C", sceUpstr)
```

---

`upstreamSeqs-class`*Class "upstreamSeqs"*

---

**Description**

Container for a collection of upstream sequences

**Objects from the Class**

Objects can be created by calls of the form `new("upstreamSeqs", ...)`. Environments are used to store collections of DNAstrings.

**Slots**

seqs: Object of class "environment" ~~  
chrom: Object of class "environment" ~~  
revComp: Object of class "environment" ~~  
type: Object of class "environment" ~~  
organism: Object of class "character" ~~  
provenance: Object of class "ANY" ~~

**Methods**

**Nmers** signature(n = "numeric", orf = "character", usobj = "upstreamSeqs"): obtain  
all subsequences of length n as view elements of a DNA string  
**keys** signature(x = "upstreamSeqs"): ...  
**organism** signature(x = "upstreamSeqs"): ...  
**seqs** signature(x = "upstreamSeqs"): ...  
**show** signature(object = "upstreamSeqs"): ...

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**Examples**

```
showClass("upstreamSeqs")  
data(sceUpstr)  
sceUpstr  
keys(sceUpstr)[1:5]
```

# Index

## \*Topic **classes**

upstreamSeqs-class, 5

## \*Topic **datasets**

harbChIP, 4

## \*Topic **models**

allhex, 1

buildUpstreamSeqs2, 2

chkMotif4TF, 3

sceUpstr, 5

allhex, 1

buildUpstreamSeqs2, 2

chkAllUS (chkMotif4TF), 3

chkMotif4TF, 3

countPattern, 3

fisher.test, 3

getUpstream (upstreamSeqs-class), 5

harbChIP, 4

keys (upstreamSeqs-class), 5

keys, upstreamSeqs-method  
(upstreamSeqs-class), 5

Nmers (upstreamSeqs-class), 5

Nmers, numeric, character, upstreamSeqs-method  
(upstreamSeqs-class), 5

organism (upstreamSeqs-class), 5

organism, upstreamSeqs-method  
(upstreamSeqs-class), 5

sceUpstr, 5

seqs (upstreamSeqs-class), 5

seqs, upstreamSeqs-method  
(upstreamSeqs-class), 5

show, upstreamSeqs-method  
(upstreamSeqs-class), 5

upstreamSeqs-class, 5