The *TFBSTools* package overview

Ge Tan *

Edited: Jan 2015; Compiled: March 21, 2015

Contents

1 Introduction 1

2 S4 classes in TFBSTools 2
   2.1 XMatrix and its subclasses ........................................... 2
   2.2 XMatrixList and its subclasses ..................................... 4
   2.3 SiteSet, SiteSetList, SitePairSet and SitePairSetList .......... 4
   2.4 MotifSet .................................................................. 5

3 Database interfaces for JASPAR2014 database 5
   3.1 Search JASPAR2014 database .......................................... 5
   3.2 Store, delete and initialize JASPAR2014 database .............. 6

4 PFM, PWM and ICM methods 6
   4.1 PFM to PWM .................................................................. 6
   4.2 PFM to ICM .................................................................. 8
   4.3 Align PFM to a custom matrix or IUPAC string ................. 9
   4.4 PWM similarity ................................................................ 10
   4.5 Dynamic random profile generation ............................... 11

5 Scan sequence and alignments with PWM pattern 13
   5.1 searchSeq .................................................................. 13
   5.2 searchAln .................................................................. 14
   5.3 searchPairBSgenome ..................................................... 17

6 Use de novo motif discovery software 17
   6.1 MEME .................................................................... 17

7 Session info 18

1 Introduction

Eukaryotic regulatory regions are characterized based a set of discovered transcription factor binding sites (TFBSs), which can be represented as sequence patterns with various degree of degeneracy.

This *TFBSTools* package is designed to be a computational framework for TFBSs analysis. Based on the famous perl module TFBS [1], we extended the class definitions and enhanced implementations in

*ge.tan09@imperial.ac.uk
The TFBSTools package overview

an interactive environment. So far this package contains a set of integrated \textit{R} \textit{S4} style classes, tools, JASPAR database interface functions. Most approaches can be described in three sequential phases. First, a pattern is generated for a set of target sequences known to be bound by a specific transcription factor. Second, a set of DNA sequences are analyzed to determine the locations of sequences consistent with the described binding pattern. Finally, in advanced cases, predictive statistical models of regulatory regions are constructed based on multiple occurrences of the detected patterns.

\textit{TFBSTools} aims to support all these functionalities in the environment \textit{R}, except the external motif finding software, such as \textit{MEME} [2].

2 \textit{S4} classes in TFBSTools

The package is built around a number of \textit{S4} class of which the \textit{XMatrix}, \textit{SiteSet} classes are the most important. The section will briefly explain most of them defined in \textit{TFBSTools}.

2.1 \textit{XMatrix} and its subclasses

\textit{XMatrix} is a virtual class, which means no concrete objects can be created directly from it. The subclass \textit{PFMatrix} is designed to store all the relevant information for one raw position frequency matrix (PFM). This object is compatible with one record from JASPAR database. \textit{PWMMatrix} is used to store a position weight matrix (PWM). Compared with \textit{PFMatrix}, it has one extra slot pseudocounts. \textit{ICMatrix} is used to store a information content matrix (ICM). Compared with \textit{PWMMatrix}, it has one extra slot \textit{schneider}.

The following examples demonstrate the creation of \textit{PFMatrix}, the conversions between these matrices and some associated methods defined for these classes.

```r
## PFMatrix construction; Not all of the slots need to be initialised.

pfm = PFMatrix(ID="MA0004.1", name="Arnt", matrixClass="Zipper-Type", strand="+",
bg=c(A=0.25, C=0.25, G=0.25, T=0.25),
tags=list(family="Helix-Loop-Helix", species="10090",
tax_group="vertebrates",medline="7592839", type="SELEX",
ACC="P53762", pazar_tf_id="TF000003",
TFBSshape_ID="11", TFencyclopedia_ID="580"),
profileMatrix=matrix(c(4L, 19L, 0L, 0L, 0L, 0L,
16L, 0L, 20L, 0L, 0L, 0L,
0L, 1L, 0L, 20L, 0L, 0L,
0L, 0L, 0L, 20L, 0L, 20L),
byrow=TRUE, nrow=4,
dimnames=list(c("A", "C", "G", "T")))

pfm
```

```
## An object of class PFMatrix
## ID: MA0004.1
## Name: Arnt
## Matrix Class: Zipper-Type
## strand: +
## Tags:
## $family
## [1] "Helix-Loop-Helix"
```
## $species
## [1] "10090"
##
## $tax_group
## [1] "vertebrates"
##
## $medline
## [1] "7592839"
##
## $type
## [1] "SELEX"
##
## $ACC
## [1] "P53762"
##
## $pazar_tf_id
## [1] "TF0000003"
##
## $TFBSshape_ID
## [1] "11"
##
## $TFencyclopedia_ID
## [1] "580"
##
## Background:
## A     C     G     T
## 0.25  0.25  0.25  0.25
## Matrix:
## A   4   19   0   0   0   0
## C  16   0  20   0   0   0
## G   0   1  20   0  20   0
## T   0   0   0  20   0   0
## coerced to matrix
## as.matrix(pfm)
## A   4   19   0   0   0   0
## C  16   0  20   0   0   0
## G   0   1  20   0  20   0
## T   0   0   0  20   0   0
## access the slots of pfm
## ID(pfm)
## [1] "MA0004.1"
##
## name(pfm)
## [1] "Arnt"
##
## Matrix(pfm)
## 

```
## A  4  19  0  0  0  0
## C 16  0  20  0  0  0
## G  0  1  0  20 0  20
## T  0  0  0  0  20  0

ncol(pfm)
## [1] 6

length(pfm)
## [1] 6
```

```
# convert a FPM to PWM, ICM
pwm = toPWM(pfm, type="log2probratio", pseudocounts=0.8,
            bg=c(A=0.25, C=0.25, G=0.25, T=0.25))

icm = toICM(pfm, pseudocounts=sqrt(rowSums(pfm)[1]), schneider=FALSE,
            bg=c(A=0.25, C=0.25, G=0.25, T=0.25))
```

```
## get the reverse compliment matrix with all the same information except the strand.
pwmRevComp <- reverseComplement(pwm)
```

### 2.2 XMatrixList and its subclasses

`XMatrixList` is used to store a set of `XMatrix` objects. Basically it is a SimpleList for easy manipulation the whole set of `XMatrix`. The concrete objects can be `PFMatrix`, `PWMMatrix` and `ICMatrix`.  

```
pfm2 = pfm
pfmList = PFMatrixList(pfm1=pfm, pfm2=pfm2, use.names=TRUE)
pfmList
```

```
## PFMatrixList of length 2
## names(2): pfm1 pfm2

names(pfmList)
## [1] "pfm1" "pfm2"
```

### 2.3 SiteSet, SiteSetList, SitePairSet and SitePairSetList

The `SiteSet` class is a container for storing a set of putative transcription factor binding sites on a nucleotide sequence (start, end, strand, score, pattern as a `PWMMatrix`, etc.) from scaning a nucleotide sequence with the corresponding `PWMMatrix`. Similarly, `SiteSetList` stores a set of `SiteSet` objects.  

For holding the results returned from a pairwise alignment scaning, `SitePairSet` and `SitePairSetList` are provided.

More detailed examples of using these classes will be given in Section 5;
2.4 MotifSet

This MotifSet class is used to store the generated motifs from de novo motif discovery software, such as MEME [2].

3 Database interfaces for JASPAR2014 database

This section will demonstrate how to operate on the JASPAR 2014 database. JASPAR is a collection of transcription factor DNA-binding preferences, modeled as matrices. These can be converted into PWMs, used for scanning genomic sequences. JASPAR is the only database with this scope where the data can be used with no restrictions (open-source). A Bioconductor experiment data package JASPAR2014 is provided with the latest release of JASPAR.

3.1 Search JASPAR2014 database

This search function fetches matrix data for all matrices in the database matching criteria defined by the named arguments and returns a PFMatrixList object. For more search criteria, please see the help page for getMatrixSet.

```r
library(JASPAR2014)
```

## Loading required package: Biostrings
## Loading required package: BiocGenerics
## Loading required package: parallel
##
## Attaching package: 'BiocGenerics'
##
## The following objects are masked from 'package:parallel':
##
## clusterApply, clusterApplyLB, clusterCall, clusterEvalQ, clusterExport,
## clusterMap, parApply, parCapply, parLapply, parLapplyLB, parRapply,
## parSapply, parSapplyLB
##
## The following object is masked from 'package:TFBSTools':
##
## as.data.frame
##
## The following object is masked from 'package:stats':
##
## xtabs
##
## The following objects are masked from 'package:base':
##
## Filter, Find, Map, Position, Reduce, anyDuplicated, append, as.data.frame,
## as.vector, cbind, colnames, do.call, duplicated, eval, evalq, get,
## intersect, is.unsorted, lapply, mapply, match, mget, order, paste, pmax,
## pmax.int, pmin, pmin.int, rank, rbinding, rep.int, rownames, sapply, setdiff,
## sort, table, tapply, union, unique, unlist, unsplit
##
## Loading required package: S4Vectors
The **TFBSTools** package overview

```r
## Loading required package: stats4
## Loading required package: IRanges
## Loading required package: XVector
## Note: the specification for class "JASPAR2014" in package 'JASPAR2014' seems equivalent
to one from package 'TFBSTools': not turning on duplicate class definitions for this
## class.

opts = list()
opts["species"] = 9606
opts["name"] = "RUNX1"
#opts["class"] = "Ig-fold"
opts["type"] = "SELEX"
opts["all_versions"] = TRUE
PFMatrixList = getMatrixSet(JASPAR2014, opts)
PFMatrixList

opts2 = list()
opts2["type"] = "SELEX"
PFMatrixList2 = getMatrixSet(JASPAR2014, opts2)
PFMatrixList2
```

### 3.2 Store, delete and initialize JASPAR2014 database

We also provide some functions to initialize an empty JASPAR2014 style database, store new **PFMatrix** or **PFMatrixList** into it, or delete some records based on ID. The backend of the database is **SQLite**.

```r
db = "myMatrixDb.sqlite"
initializeJASPARDB(db)
storeMatrix(db, pfm)
deleteMatrixHavingID(db, "MA0003")
```

### 4 PFM, PWM and ICM methods

This section will give an introduction of matrix operations, including conversion from PFM to PWM and ICM, profile matrices comparison, dynamic random profile generation.

#### 4.1 PFM to PWM

The method `toPWM` can convert PFM to PWM [3]. Optional parameters include `type`, `pseudocounts`, `bg`. The implementation in this package is a bit different from that in `Biostrings`.

First of all, `toPWM` allows the input matrix to have different column sums, which means the count matrix can have an unequal number of sequences contributing to each column. This scenario is rare, but exists in JASPAR SELEX data.
Second, we can specify customized pseudocounts. Pseudocounts is necessary for correcting the small number of counts or eliminating the zero values before log transformation. In TFBS perl module, the square root of the number of sequences contributing to each column. However, it has been shown to too harsh [4]. Hence, a default value of 0.8 is used. Of course, it can be changed to other customized value or even different values for each column.

```r
pwm = toPWM(pfm, pseudocounts=0.8)
pwm
```

```
# An object of class PWMatrix
# ID: MA0004.1
# Name: Arnt
# Matrix Class: Zipper-Type
# strand: +
# Pseudocounts: 0.8
# Tags:
# $family
# [1] "Helix-Loop-Helix"
#
# $species
# [1] "10090"
#
# $tax_group
# [1] "vertebrates"
#
# $medline
# [1] "7592839"
#
# $type
# [1] "SELEX"
#
# $ACC
# [1] "P53762"
#
# $pazar_tf_id
# [1] "TF0000003"
#
# $TFBSshape_ID
# [1] "11"
#
# $TFencyclopedia_ID
# [1] "580"
#
# Background:
# A C G T
# 0.25 0.25 0.25 0.25
# Matrix:
# A -0.3081 1.885 -4.700 -4.700 -4.700 -4.700
# C 1.6394 -4.700 1.958 -4.700 -4.700 1.958
# G -4.7004 -2.115 -4.700 1.958 -4.700 1.958
# T -4.7004 -4.700 -4.700 -4.700 1.958 -4.700
```
4.2 PFM to ICM

The method toICM can convert PFM to ICM [5]. Besides the similar pseudocounts, bg, you can also choose to do the schneider correction.

The information content matrix has a column sum between 0 (no base preference) and 2 (only 1 base used). Usually this information is used to plot sequence log.

How a PFM is converted to ICM: we have the PFM matrix $x$, base background frequency $bg$, pseudocounts for correction.

\[
Z[j] = \sum_{i=1}^{4} x[i, j]
\]  
(1)

\[
p[i, j] = \frac{(x[i, j] + bg[i] \times \text{pseudocounts}[j])}{(Z[j] + \sum_{i=1}^{4} bg[i] \times \text{pseudocounts}[j])}
\]  
(2)

\[
D[j] = \log_2 4 + \sum_{i=1}^{4} p[i, j] * \log p[i, j]
\]  
(3)

\[
ICM[i, j] = p[i, j] \times D[j]
\]  
(4)

```r
icm = toICM(pfm, pseudocounts=0.8, schneider=TRUE)
```

```r
icm
## An object of class ICMatrix
## ID: MA0004.1
## Name: Arnt
## Matrix Class: Zipper-Type
## strand: +
## Pseudocounts: 0.8
## Schneider correction: TRUE
## Tags:
## $family
## [1] "Helix-Loop-Helix"
##
## $species
## [1] "10090"
##
## $tax_group
## [1] "vertebrates"
##
## $medline
## [1] "7592839"
##
## $type
## [1] "SELEX"
##
## $ACC
## [1] "P53762"
##
## $pazar_tf_id
## [1] "TF0000003"
```
To plot the sequence logo, we use the package `seqlogo`.

```r
seqLogo(icm)
```

### 4.3 Align PFM to a custom matrix or IUPAC string

In some cases, it is beneficial to assess similarity of existing profile matrices, such as JASPAR, to a newly discovered matrix (as with using BLAST for sequence data comparison when using Genbank).

`TFBSTools` provides tools for comparing pairs of PFMs, or a PFM with IUPAC string, using a modified Needleman-Wunsch algorithm [6].
## one to one comparison

```r
data(MA0003.2)
data(MA0004.1)
pfmSubject = MA0003.2
pfmQuery = MA0004.1
PFMSimilarity(pfmSubject, pfmQuery)
## score relScore
## 8.104 27.012
```

## one to several comparison

```r
PFMSimilarity(pfmList, pfmQuery)
## $pfm1
## score relScore
## 12 100
##
## $pfm2
## score relScore
## 12 100
```

## align IUPAC string

```r
IUPACString = "ACGMRWSYKVHDBN"
PFMSimilarity(pfmList, IUPACString)
## $pfm1
## score relScore
## 8.815 29.383
##
## $pfm2
## score relScore
## 8.815 29.383
```

### 4.4 PWM similarity

To measure the similarity of two PWM matrix in three measurements: “normalised Euclidean distance”, “Pearson correlation” and “Kullback Leibler divergence” [7]. Given two PWMs, $P^1$ and $P^2$, where $l$ is the length. $P_{i,b}$ is the values in column $i$ with base $b$. The normalised Euclidean distance is computed in

$$
D(a, b) = \frac{1}{\sqrt{2l}} \cdot \sum_{i=1}^{l} \sqrt{\sum_{b \in \{A,C,G,T\}} (P_{i,b}^1 - P_{i,b}^2)^2}
$$

This distance is between 0 (perfect identity) and 1 (complete dis-similarity).

The pearson correlation coefficient is computed in

$$
r(P^1, P^2) = \frac{1}{l} \cdot \sum_{i=1}^{l} \frac{\sum_{b \in \{A,C,G,T\}} (P_{i,b}^1 - 0.25)(P_{i,b}^2 - 0.25)}{\sqrt{\sum_{b \in \{A,C,G,T\}} (P_{i,b}^1 - 0.25)^2} \cdot \sum_{b \in \{A,C,G,T\}} (P_{i,b}^2 - 0.25)^2}
$$
The Kullback-Leibler divergence is computed in

\[
KL(P^1, P^2) = \frac{1}{2l} \sum_{i=1}^{l} \sum_{b \in \{A,C,G,T\}} (P^1_{i,b} \log \frac{P^1_{i,b}}{P^2_{i,b}} + P^2_{i,b} \log \frac{P^2_{i,b}}{P^1_{i,b}}).
\]

(7)

```r
data(MA0003.2)
data(MA0004.1)
pwm1 = toPWM(MA0003.2, type="prob")
pwm2 = toPWM(MA0004.1, type="prob")
PWMSimilarity(pwm1, pwm2, method="Euclidean")
## [1] 0.5135
PWMSimilarity(pwm1, pwm2, method="Pearson")
## [1] 0.2829
PWMSimilarity(pwm1, pwm2, method="KL")
## [1] 2.386
```

### 4.5 Dynamic random profile generation

In this section, we will demonstrate the capability of random profile matrices generation with matrix permutation and probabilistic sampling. In many computational/simulation studies, it is particularly desired to have a set of random matrices. Some cases includes the estimation of distance between putative TFBS and transcription start site, the evaluation of comparison between matrices [8]. These random matrices are expected to have same statistical properties with the selected profiles, such as nucleotide content or information content.

The permutation method is relatively easy. It simply shuffles the columns either constrained in each matrix, or columns among all selected matrices. The probabilistic sampling is more complicated and can be done in two steps:

1. A Dirichlet multinomial mixture model is trained on all available matrices in JASPAR.
2. Random columns are sampled from the posterior distribution of the trained Dirichlet model based on selected profiles.

```r
## Matrix permutation
permuteMatrix(pfmQuery)
```

## An object of class PFMatrix
## ID: MA0004.1
## Name: Arnt
## Matrix Class: Zipper-Type
## strand: +
## Tags:
## $comment
## [1] "-
##
## $family
## [1] "Helix-Loop-Helix"
##
## $medline
## [1] "7592839"
## $pazar_tf_id
## [1] "TF0000003"
##
## $tax_group
## [1] "vertebrates"
##
## $tfbs_shape_id
## [1] "11"
##
## $tfe_id
## [1] "580"
##
## $type
## [1] "SELEX"
##
## $collection
## [1] "CORE"
##
## $species
## [1] "10090"
##
## $acc
## [1] "P53762"
##
## Background:
## A   C   G   T
## 0.25 0.25 0.25 0.25
##
## Matrix:
## A   0   0  19   4   0   0
## C   0   0   0  16  20   0
## G   0  20   1   0   0  20
## T  20   0   0   0   0   0

permuteMatrix(pfmList, type="intra")
## PFMMatrixList of length 2
## names(2): pfm1 pfm2

permuteMatrix(pfmList, type="inter")
## PFMMatrixList of length 2
## names(2): pfm1 pfm2

## Dirichlet model training
data(MA0003.2)
data(MA0004.1)
pfmList <- PFMatrixList(pfm1=MA0003.2, pfm2=MA0004.1, use.names=TRUE)
dmmParameters <- dmmEM(pfmList, K=6, alg="C")
## Matrice sampling from trained Dirichlet model
pwmSampled <- rPWMDmm(MA0003.2, dmmParameters$alpha0, dmmParameters$pmix,
                      N=1, W=6)
5 Scan sequence and alignments with PWM pattern

5.1 searchSeq

searchSeq scans a nucleotide sequence with the pattern represented in the PWM. The strand argument controls which strand of the sequence will be searched. When it is "*", both strands will be scanned.

A siteset object will be returned which can be exported into GFF3 or GFF2 format. Empirical p-values for the match scores can be calculated by an exact method from TFMPvalue or the distribution of sampled scores.

```r
library(Biostrings)
data(MA0003.2)
data(MA0004.1)
pwmList = PWMMatrixList(MA0003.2=toPWM(MA0003.2), MA0004.1=toPWM(MA0004.1), use.names=TRUE)
subject = DNAString("GAATTCTCTCTTGTTGTAGTCTCTTGACAAAATG")
siteset = searchSeq(pwm, subject, seqname="seq1", min.score="60%", strand="*")
sitesetList = searchSeq(pwmList, subject, seqname="seq1", min.score="60%", strand="*")

## generate gff2 or gff3 style output
head(writeGFF3(siteset))
## seqname source feature start end score strand frame
## 1 seq1 TFBS TFBS 8 13 -1.888 + .
## 2 seq1 TFBS TFBS 21 26 -1.888 + .
## 3 seq1 TFBS TFBS 29 34 -3.909 + .
## 4 seq1 TFBS TFBS 8 13 -1.961 - .
## 5 seq1 TFBS TFBS 10 15 -3.909 - .
## 6 seq1 TFBS TFBS 21 26 -1.961 - .
## attributes
## MA0003.2 TF=Arnt;class=Zipper-Type;sequence=CTCTTG
## MA0004.1.1 TF=Arnt;class=Zipper-Type;sequence=CTCTTG
## MA0004.1.2 TF=Arnt;class=Zipper-Type;sequence=AAAATG
## MA0004.1.3 TF=Arnt;class=Zipper-Type;sequence=AAAGAG
## MA0004.1.4 TF=Arnt;class=Zipper-Type;sequence=AAAGAG

head(writeGFF3(sitesetList))
## seqname source feature start end score strand frame
## MA0003.2 seq1 TFBS TFBS 18 32 -16.438 - .
## MA0004.1.1 seq1 TFBS TFBS 8 13 -1.888 + .
## MA0004.1.2 seq1 TFBS TFBS 21 26 -1.888 + .
## MA0004.1.3 seq1 TFBS TFBS 29 34 -3.909 + .
## MA0004.1.4 seq1 TFBS TFBS 8 13 -1.961 - .
## MA0004.1.5 seq1 TFBS TFBS 10 15 -3.909 - .
## attributes
## MA0003.2 TF=TFAP2A;class=Zipper-Type;sequence=TTTTGTCAAGAGACT
## MA0004.1.1 TF=Arnt;class=Zipper-Type;sequence=CTCTTG
```
The **TFBSTools** package overview

## MA0004.1.2
TF=Arnt; class=Zipper-Type; sequence=CTCTTG

## MA0004.1.3
TF=Arnt; class=Zipper-Type; sequence=AAAATG

## MA0004.1.4
TF=Arnt; class=Zipper-Type; sequence=CAAGAG

## MA0004.1.5
TF=Arnt; class=Zipper-Type; sequence=AACAAG

```r
head(writeGFF2(siteset))
```

<table>
<thead>
<tr>
<th>seqname</th>
<th>source</th>
<th>feature</th>
<th>start</th>
<th>end</th>
<th>score</th>
<th>strand</th>
<th>frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>8</td>
<td>13</td>
<td>-1.888</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>21</td>
<td>26</td>
<td>-1.888</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>29</td>
<td>34</td>
<td>-3.909</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>8</td>
<td>13</td>
<td>-1.961</td>
<td>-</td>
<td>.</td>
</tr>
<tr>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>10</td>
<td>15</td>
<td>-3.909</td>
<td>-</td>
<td>.</td>
</tr>
<tr>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>21</td>
<td>26</td>
<td>-1.961</td>
<td>-</td>
<td>.</td>
</tr>
</tbody>
</table>

```r
attributes
```

```
## MA0003.2
[1] 0.6197

## MA0004.1
[1] 0.6652 0.6652 0.6141 0.6634 0.6141 0.6634
```

## 5.2 searchAln

searchAln scans a pairwise alignment with the pattern represented by the PWM. It reports only those hits that are present in equivalent positions of both sequences and exceed a specified threshold score in both, AND are found in regions of the alignment above the specified.

```r
library(Biostrings)
data(MA0003.2)
pwm <- toPWM(MA0003.2)
aln1 <- DNAString("ACTTCACCAGCTCCCTGGCGGTAAGTTGATC---AAAGG---AAACGCAAAGTTTTCAAG")
aln2 <- DNAString("GTTTCACTACTTCCTTTCGGGTAAGTAAATATATAAATATATAAAAATATAATTTTCATC")
sitePairSet <- searchAln(pwm, aln1, aln2, seqname1="seq1", seqname2="seq2",
The **TFBSTools** package overview

```r
min.score="50\%", cutoff=0.5, strand="\*", type="any"
```

## generate gff style output

```r
head(writeGFF3(sitePairSet))
```

<table>
<thead>
<tr>
<th>#</th>
<th>seqname</th>
<th>source</th>
<th>feature</th>
<th>start</th>
<th>end</th>
<th>score</th>
<th>strand</th>
<th>frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>6</td>
<td>20</td>
<td>-9.515</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>7</td>
<td>21</td>
<td>-13.349</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>8</td>
<td>22</td>
<td>-13.182</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>4</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>9</td>
<td>23</td>
<td>-3.730</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>10</td>
<td>24</td>
<td>-7.678</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>6</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>14</td>
<td>28</td>
<td>-20.775</td>
<td>+</td>
<td>.</td>
</tr>
</tbody>
</table>

### attributes

1. TF=TFAP2A; class=Zipper-Type; sequence=ACCAGCTCCTGGCGG
2. TF=TFAP2A; class=Zipper-Type; sequence=CCAGCTCCTGGCGG
3. TF=TFAP2A; class=Zipper-Type; sequence=CAGCTCCTGGCGGT
4. TF=TFAP2A; class=Zipper-Type; sequence=AGCTCCTGGCGGT
5. TF=TFAP2A; class=Zipper-Type; sequence=GCTCCTGGCGGT
6. TF=TFAP2A; class=Zipper-Type; sequence=CCTGGCGGT

```r
head(writeGFF2(sitePairSet))
```

<table>
<thead>
<tr>
<th>#</th>
<th>seqname</th>
<th>source</th>
<th>feature</th>
<th>start</th>
<th>end</th>
<th>score</th>
<th>strand</th>
<th>frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>6</td>
<td>20</td>
<td>-9.515</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>7</td>
<td>21</td>
<td>-13.349</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>8</td>
<td>22</td>
<td>-13.182</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>4</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>9</td>
<td>23</td>
<td>-3.730</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>5</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>10</td>
<td>24</td>
<td>-7.678</td>
<td>+</td>
<td>.</td>
</tr>
<tr>
<td>6</td>
<td>seq1</td>
<td>TFBS</td>
<td>TFBS</td>
<td>14</td>
<td>28</td>
<td>-20.775</td>
<td>+</td>
<td>.</td>
</tr>
</tbody>
</table>

### attributes

1. TF "TFAP2A"; class "Zipper-Type"; sequence "ACCAGCTCCTGGCGG"
2. TF "TFAP2A"; class "Zipper-Type"; sequence "CCAGCTCCTGGCGG"
3. TF "TFAP2A"; class "Zipper-Type"; sequence "CAGCTCCTGGCGGT"
4. TF "TFAP2A"; class "Zipper-Type"; sequence "AGCTCCTGGCGGT"
5. TF "TFAP2A"; class "Zipper-Type"; sequence "GCTCCTGGCGGT"
6. TF "TFAP2A"; class "Zipper-Type"; sequence "CCTGGCGGT"

## search the Axt alignment

```r
library(CNEr)
```

```r
axtFilesHg19DanRer7 <- file.path(system.file("extdata", package="CNEr"), "hg19.danRer7.net.axt")
```

```r
axtHg19DanRer7 <- readAxt(axtFilesHg19DanRer7)
```

```r
## The number of axt files 1
```

```r
## The number of axt alignments is 133
```

```r
sitePairSet <- searchAln(pwm, axtHg19DanRer7, min.score="80\%", windowSize=51L, cutoff=0.7, strand="\*", type="any", conservation=NULL, mc.cores=2)
```

```r
GRangesTFBS <- toGRangesList(sitePairSet, axtHg19DanRer7)
```

```r
GRangesTFBS$targetTFBS
```

## GRanges object with 33 ranges and 5 metadata columns:

```r
## sequeances ranges strand | matrix.ID matrix.strand abs.score
```
## The TFBSTools package overview

<table>
<thead>
<tr>
<th></th>
<th>&lt;Rle&gt;</th>
<th>&lt;IRanges&gt;</th>
<th>&lt;Rle&gt;</th>
<th>&lt;character&gt;</th>
<th>&lt;character&gt;</th>
<th>&lt;numeric&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>chr11</td>
<td>[31128026, 31128040]</td>
<td>+</td>
<td>MA0003.2</td>
<td>+</td>
<td>2.089</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>[29]</td>
<td>chr11</td>
<td>[32198154, 32198168]</td>
<td>+</td>
<td>MA0003.2</td>
<td>+</td>
<td>2.512</td>
</tr>
<tr>
<td>[31]</td>
<td>chr11</td>
<td>[32221918, 32221932]</td>
<td>+</td>
<td>MA0003.2</td>
<td>-</td>
<td>10.610</td>
</tr>
<tr>
<td>[33]</td>
<td>chr11</td>
<td>[32456355, 32456369]</td>
<td>+</td>
<td>MA0003.2</td>
<td>-</td>
<td>1.177</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>rel.score</th>
<th>sitesSeq</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>0.8130</td>
<td>GGCCCCCCTGGGGT</td>
</tr>
<tr>
<td>[2]</td>
<td>0.8059</td>
<td>CCCCCCCTGGGGT</td>
</tr>
<tr>
<td>[3]</td>
<td>0.8725</td>
<td>ACTGCCCCCTGGGACA</td>
</tr>
<tr>
<td>[4]</td>
<td>0.8726</td>
<td>TGGCCCTAGGGACAGG</td>
</tr>
<tr>
<td>[5]</td>
<td>0.8146</td>
<td>TCTGACCCAGGGACA</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>[29]</td>
<td>0.8174</td>
<td>ACCGTCTTTAGGGGA</td>
</tr>
<tr>
<td>[30]</td>
<td>0.8400</td>
<td>TCTACCCCTGGGGAC</td>
</tr>
<tr>
<td>[31]</td>
<td>0.9020</td>
<td>TACCCCTAGGGACAGG</td>
</tr>
<tr>
<td>[32]</td>
<td>0.8351</td>
<td>AAGGCCCCGTAAGGCA</td>
</tr>
<tr>
<td>[33]</td>
<td>0.8035</td>
<td>GGGCCCTAGGGACAGG</td>
</tr>
</tbody>
</table>

---

seqinfo: 3 sequences from an unspecified genome; no seqlengths

---

## GRangesTFBS$queryTFBS

<table>
<thead>
<tr>
<th></th>
<th>&lt;Rle&gt;</th>
<th>&lt;IRanges&gt;</th>
<th>&lt;Rle&gt;</th>
<th>&lt;character&gt;</th>
<th>&lt;character&gt;</th>
<th>&lt;numeric&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>[29]</td>
<td>chr25</td>
<td>[15359169, 15359183]</td>
<td>+</td>
<td>MA0003.2</td>
<td>+</td>
<td>3.135</td>
</tr>
<tr>
<td>[31]</td>
<td>chr25</td>
<td>[15472024, 15472038]</td>
<td>+</td>
<td>MA0003.2</td>
<td>-</td>
<td>5.674</td>
</tr>
<tr>
<td>[32]</td>
<td>chr25</td>
<td>[15570786, 15570800]</td>
<td>+</td>
<td>MA0003.2</td>
<td>+</td>
<td>1.605</td>
</tr>
<tr>
<td>[33]</td>
<td>chr25</td>
<td>[15570789, 15570803]</td>
<td>+</td>
<td>MA0003.2</td>
<td>-</td>
<td>5.262</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>rel.score</th>
<th>sitesSeq</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>0.8190</td>
<td>GGCCCCCCTGGGGCTC</td>
</tr>
<tr>
<td>[2]</td>
<td>0.8435</td>
<td>CCCCCCCTGGGGCTC</td>
</tr>
<tr>
<td>[3]</td>
<td>0.8560</td>
<td>TCTGACCCAGGGAAC</td>
</tr>
<tr>
<td>[4]</td>
<td>0.8584</td>
<td>TGGCCCTAGGGACAGG</td>
</tr>
<tr>
<td>[5]</td>
<td>0.8068</td>
<td>TGGCCCTAGGGACAGG</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
5.3 searchPairBSgenome

searchPairBSgenome is designed to do the genome-wise phylogenetic footprinting. Given two BSgenome, a chain file for liftover from one genome to another, searchPairBSgenome identifies the putative transcription factor binding sites which are conserved in both genomes.

```r
library(rtracklayer)
library(JASPAR2014)
library(BSgenome.Hsapiens.UCSC.hg19)
library(BSgenome.Mmusculus.UCSC.mm10)
pfm <- getMatrixByID(JASPAR2014, ID="MA0004.1")
pwm <- toPWM(pfm)
chain <- import.chain("Downloads/hg19ToMm10.over.chain")
sitePairSet <- searchPairBSgenome(pwm, BSgenome.Hsapiens.UCSC.hg19, BSgenome.Mmusculus.UCSC.mm10, chr1="chr1", chr2="chr1", min.score="90\%", strand="+", chain=chain)
```

6 Use de novo motif discovery software

In this section, we will introduce wrapper functions for external motif discovery programs. So far, MEME is supported.

6.1 MEME

runMEME takes a DNAStringSet or a set of characters as input, and returns a MotifSet object.

```r
motifSet <- runMEME(file.path(system.file("extdata", package="TFBSTools"), "crp0.s"),
  binary="meme",
  arguments=list("-nmotifs"=3))
```

```r
## Get the sites sequences and surrounding sequences
sitesSeq(motifSet, type="all")
## Get the sites sequences only
sitesSeq(motifSet, type="none")
consensusMatrix(motifSet)
```
7 Session info

The following is the session info that generated this vignette:

```r
sessionInfo()
```

```text
## R version 3.2.0 alpha (2015-03-20 r68043)
## Platform: x86_64-unknown-linux-gnu (64-bit)
## Running under: Ubuntu 14.04.2 LTS
##
## locale:
## [1] LC_CTYPE=en_US.UTF-8   LC_NUMERIC=C   LC_TIME=en_US.UTF-8
## [4] LC_COLLATE=C           LC_MONETARY=en_US.UTF-8 LC_MESSAGES=en_US.UTF-8
## [7] LC_PAPER=en_US.UTF-8   LC_NAME=C       LC_ADDRESS=C
## [10] LC_TELEPHONE=C        LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats4 parallel stats graphics grDevices utils datasets methods
## [9] base
##
## other attached packages:
## [1] CNEr_1.3.2 JASPAR2014_1.3.1 Biostrings_2.35.11 XVector_0.7.4
## [5] IRanges_2.1.43 S4Vectors_0.5.22 BiocGenerics_0.13.8 TFBSTools_1.5.6
## [9] knitr_1.9
##
## loaded via a namespace (and not attached):
## [1] Rcpp_0.11.5 GenomicAlignments_1.3.32 GenomicRanges_1.19.47
## [4] zlibbioc_1.13.2 BiocParallel_1.1.19 BSgenome_1.35.19
## [7] TFMPvalue_0.0.5 stringr_0.6.2 highr_0.4
## [10] GenomeInfoDb_1.3.15 caTools_1.17.1 tools_3.2.0
## [13] grid_3.2.0 snow_0.3-13 seqLogo_1.33.2
## [16] DBI_0.3.1 lambda.r_1.1.7 futile.logger_1.4
## [19] gtools_3.4.1 digest_0.6.8 rtracklayer_1.27.9
## [22] formatR_1.0 codetools_0.2-11 futile.options_1.0.0
## [25] bitops_1.0-6 RCurl_1.95-4.5 DirichletMultinomial_1.9.1
## [28] evaluate_0.5.5 RSQLite_1.0.0 Rsamtools_1.19.47
## [31] XML_3.98-1.1 BiocStyle_1.5.3
```
References


