The **TFBSTools** package overview

Ge Tan

April 12, 2014

Contents

1 Introduction **2**

2 S4 classes in TFBSTools **2**
2.1 XMatrix and its subclasses . . . 2
2.2 XMatrixList and its subclasses . . . 6
2.3 SiteSet, SiteSetList, SitePairSet and SitePairSetList . . . 7
2.4 MotifSet . . . 7

3 Database interfaces for JASPAR2014 database **7**
3.1 Search JASPAR2014 database . . . 7
3.2 Store, delete and initialize JASPAR2014 database . . . 8

4 PFM, PWM and ICM methods **8**
4.1 PFM to PWM . . . 8
4.2 PFM to ICM . . . 9
4.3 Align PFM to a custom matrix or IUPAC string . . . 11
4.4 PFM permutation . . . 12
4.5 PWM similarity . . . 13

5 Scan sequence and alignments with PWM pattern **14**
5.1 searchSeq . . . 14
5.2 searchAln . . . 16
5.3 searchPairBSgenome . . . 19

6 Use external pattern generators **19**
6.1 MEME . . . 19

7 Conclusion **20**

1 Introduction

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

This *TFBSTools* package is designed to be a compuational framework for transcription factor binding site analysis. Based on the famous perl module TFBS (Lenhard and Wasserman, 2002), we extended the class definitions and enhanced implementations in an interactive environment. So far this package contains a set of integrated R 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 mutiple occurrences of the detected patterns.

TFBSTools aims to support all these functionalities in the environment R, except the external motif finding software, such as MEME (Bailey and Elkan, 1994).

2 S4 classes in TFBSTools

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

2.1 XMatrix and its subclasses

XMatrix is a virtual class, which means no concrete objects can be created directly from it. The subclass *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.

PWMatrix is used to store a position weight matrix (PWM). Compared with *PFMatrix*, it has one extra slot `pseudocounts`.

ICMatrix is used to store a information content matrix (ICM). Compared with *PWMatrix*, it has one extra slot `schneider`.

```
> library(TFBSTools)
> 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="TF0000003",
+                   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,  20L,
+                                 0L,  0L,  0L,  0L,  20L, 0L),
+                               byrow=TRUE, nrow=4,
+                               dimnames=list(c("A", "C", "G", "T")))
+         )
> ## coerced to matrix
> as.matrix(pfm)

  [,1] [,2] [,3] [,4] [,5] [,6]
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

> ## access the slots of pfm
> ID(pfm)

[1] "MA0004.1"

> name(pfm)

[1] "Arnt"

> Matrix(pfm)

  [,1] [,2] [,3] [,4] [,5] [,6]
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

> ## conversion to pwm, icm
> pwm = toPWM(pfm)
> 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:
        [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
A -0.3081223  1.884523 -4.700440 -4.700440 -4.700440 -4.700440
C  1.6394103 -4.700440  1.957772 -4.700440 -4.700440 -4.700440
G -4.7004397 -2.115477 -4.700440  1.957772 -4.700440  1.957772
T -4.7004397 -4.700440 -4.700440 -4.700440  1.957772 -4.700440

> icm = toICM(pfm)
> icm

An object of class ICMatrix
ID: MA0004.1
Name: Arnt
Matrix Class: Zipper-Type
strand: +
Pseudocounts: 0.8
Schneider correction: FALSE
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:
        [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
A 0.22700966 1.40964891 0.01697796 0.01697796 0.01697796 0.01697796
C 0.87560869 0.01468384 1.71477409 0.01697796 0.01697796 0.01697796
G 0.01080998 0.08810306 0.01697796 1.71477409 0.01697796 1.71477409
T 0.01080998 0.01468384 0.01697796 0.01697796 1.71477409 0.01697796

> ## get the reverse complment matrix with all the same information except the strand.
> reverseComplement(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:
       [,1]      [,2]      [,3]      [,4]      [,5]       [,6]
A -4.700440  1.957772 -4.700440 -4.700440 -4.700440 -4.7004397
C  1.957772 -4.700440  1.957772 -4.700440 -2.115477 -4.7004397
G -4.700440 -4.700440 -4.700440  1.957772 -4.700440  1.6394103
T -4.700440 -4.700440 -4.700440 -4.700440  1.884523 -0.3081223

>
```

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*, *PWMatrix* 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 *PWMatrix*, etc.) from scaning a nucleotide sequence with the corresponding *PWMatrix*.

Similarly, *SiteSetList* stores a set of *SiteSet* objects. To store the results from scanning a pairwise alignment, *SitePairSet* is created.

2.4 MotifSet

This *MotifSet* class is used to store the generated motifs from motifs discovery software, such as MEME.

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 Position Weight Matrices (PWMs or PSSMs), used for scanning genomic sequences. JASPAR is the only database with this scope where the data can be used with no restrictions (open-source).

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 criterias, please see the help page for (getMatrixSet).

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

```
PFMatrixList of length 1
names(1): MA0002.1
```

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

PFMatrixList of length 111
names(111): MA0004.1 MA0006.1 MA0008.1 ... MA0588.1 MA0589.1 MA0590.1
```

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.

```
> 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 conversion from PFM.

4.1 PFM to PWM

The method `toPWM` can convert PFM to PWM (Wasserman and Sandelin, 2004). Optional parameters include `type`, `pseudocounts`, `bg`. The implementation in this package is a bit different from *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 (Nishida *et al.*, 2009). 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.

```
> 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:
        [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
A -0.3081223  1.884523 -4.700440 -4.700440 -4.700440 -4.700440
C  1.6394103 -4.700440  1.957772 -4.700440 -4.700440 -4.700440
G -4.7004397 -2.115477 -4.700440  1.957772 -4.700440  1.957772
T -4.7004397 -4.700440 -4.700440 -4.700440  1.957772 -4.700440
```

4.2 PFM to ICM

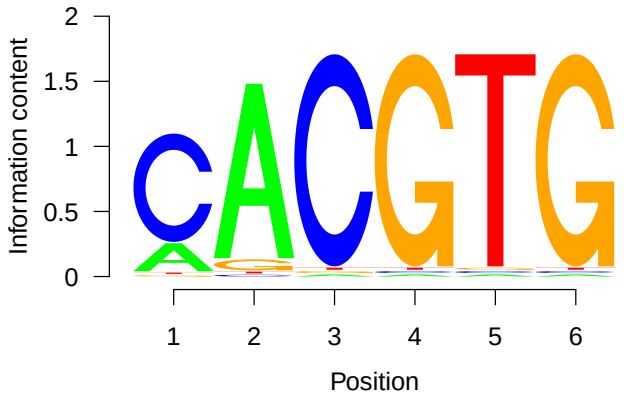
The method `toICM` can convert PFM to ICM (Schneider *et al.*, 1986). Besides the similar `pseudocounts`, `bg`, you can also choose to do the `schneider` correction.

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

$TFBSshape_ID
[1] "11"

$TFencyclopedia_ID
[1] "580"

Background:
   A    C    G    T
0.25 0.25 0.25 0.25
Matrix:
         [,1]       [,2]       [,3]       [,4]       [,5]       [,6]
A 0.203985791 1.30439694 0.01588159 0.01588159 0.01588159 0.01588159
C 0.786802337 0.01358747 1.60404024 0.01588159 0.01588159 0.01588159
G 0.009713609 0.08152481 0.01588159 1.60404024 0.01588159 1.60404024
```

```
T 0.009713609 0.01358747 0.01588159 0.01588159 1.60404024 0.01588159
```

To plot the sequence logo, we use the package *seqlogo*.

```
> seqLogo(icm)
```



4.3 Align PFM to a custom matrix or IUPAC string

In some cases, it is beneficial to assess similarity to input data (as with using BLAST for sequence data comparison when using Genbank).

```
> ## one to one comparison
> data(MA0003.2)
> data(MA0004.1)
> pfmSubject = MA0003.2
> pfmQuery =  MA0004.1
> PFMSimilarity(pfmSubject, pfmQuery)

    score  relScore
 8.103627 27.012091

> ## one to several comparsion
> PFMSimilarity(pfmList, pfmQuery)

$pfm1
   score relScore
      12      100

$pfm2
   score relScore
      12      100
```

```
> ## align IUPAC string
> IUPACString = "ACGTMRWSYKVHDBN"
> PFMSimilarity(pfmList, IUPACString)

$pfm1
   score relScore
 8.81500 29.38333

$pfm2
   score relScore
 8.81500 29.38333
```

4.4 PFM permutation

In this section, we will demonstrate the matrix permutation. This method simply shuffles the columns in matrices. This can either be done by just shuffling columns within each selected matrix, or by shuffling columns almong all selected matrices.

```
> #library(JASPAR2014)
> #pfmSubject = getMatrixByID(JASPAR2014, ID="MA0043")
> #pfmQuery = getMatrixByID(JASPAR2014, ID="MA0048")
> #opts = list()
> #opts[["class"]] = "Ig-fold"
> #pfmList = getMatrixSet(JASPAR2014, opts)
> 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:
  [,1] [,2] [,3] [,4] [,5] [,6]
A    0   19    0    0    4    0
C    0    0   20    0   16    0
G   20    1    0   20    0    0
T    0    0    0    0    0   20

> permuteMatrix(pfmList, type="intra")

PFMatrixList of length 2
names(2): pfm1 pfm2

> permuteMatrix(pfmList, type="inter")

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

4.5 PWM similarity

To measure the similarity of two PWM matrix in three measurements: "normalised Euclidean distance", "Pearson correlation" and "Kullback Leibler divergence" (Linhart *et al.*, 2008). 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

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

This distance is between 0 (perfect identity) and 1 (complete dis-similarity). The pearson correlation coefficient is computed in

$$r(P^1, P^2) = \frac{1}{l} \cdot \sum_{i=1}^{l} \frac{\sum_{b \in \{A,C,G,T\}} (P^1_{i,b} - 0.25)(P^2_{i,b} - 0.25)}{\sqrt{\sum_{b \in \{A,C,G,T\}} (P^1_{i,b} - 0.25)^2 \cdot \sum_{b \in \{A,C,G,T\}} (P^2_{i,b} - 0.25)^2}}. \tag{2}$$

The Kullback-Leibler divergence is computed in

$$KL(P^1, P^2) = \frac{1}{2l} \cdot \sum_{i=1}^{l} \sum_{b \in \{A,C,G,T\}} (P^1_{i,b} \log \frac{P^1_{i,b}}{P^2_{i,b}} + P^2_{i,b} \log \frac{P^2_{i,b}}{P^1_{i,b}}). \tag{3}$$

```
>   data(MA0003.2)
>   data(MA0004.1)
>   pwm1 = toPWM(MA0003.2, type="prob")
>   pwm2 = toPWM(MA0004.1, type="prob")
>   PWMSimilarity(pwm1, pwm2, method="Euclidean")

[1] 0.5134956

>   PWMSimilarity(pwm1, pwm2, method="Pearson")

[1] 0.2828507

>   PWMSimilarity(pwm1, pwm2, method="KL")

[1] 1.000865
```

5 Scan sequence and alignments with PWM pattern

5.1 searchSeq

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

```
> library(Biostrings)
> data(MA0003.2)
> data(MA0004.1)
> pwmList = PWMatrixList(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 gff style output
> head(writeGFF3(siteset))

  seqname source feature start end     score strand frame
1    seq1   TFBS    TFBS     8  13 -1.888154      +     .
2    seq1   TFBS    TFBS    21  26 -1.888154      +     .
3    seq1   TFBS    TFBS    29  34 -3.908935      +     .
4    seq1   TFBS    TFBS     8  13 -1.961403      -     .
5    seq1   TFBS    TFBS    10  15 -3.908935      -     .
6    seq1   TFBS    TFBS    21  26 -1.961403      -     .
                                attributes
1 TF=Arnt;class=Zipper-Type;sequence=CTCTTG
2 TF=Arnt;class=Zipper-Type;sequence=CTCTTG
3 TF=Arnt;class=Zipper-Type;sequence=AAAATG
4 TF=Arnt;class=Zipper-Type;sequence=CAAGAG
5 TF=Arnt;class=Zipper-Type;sequence=AACAAG
6 TF=Arnt;class=Zipper-Type;sequence=CAAGAG

> head(writeGFF3(sitesetList))

           seqname source feature start end      score strand frame
MA0003.2      seq1   TFBS    TFBS    18  32 -16.437682      -     .
MA0004.1.1    seq1   TFBS    TFBS     8  13  -1.888154      +     .
MA0004.1.2    seq1   TFBS    TFBS    21  26  -1.888154      +     .
MA0004.1.3    seq1   TFBS    TFBS    29  34  -3.908935      +     .
MA0004.1.4    seq1   TFBS    TFBS     8  13  -1.961403      -     .
MA0004.1.5    seq1   TFBS    TFBS    10  15  -3.908935      -     .
                                                    attributes
MA0003.2   TF=TFAP2A;class=Zipper-Type;sequence=TTTTGTCAAGAGACT
MA0004.1.1            TF=Arnt;class=Zipper-Type;sequence=CTCTTG
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

> head(writeGFF2(siteset))

  seqname source feature start end     score strand frame
1    seq1   TFBS    TFBS     8  13 -1.888154      +     .
2    seq1   TFBS    TFBS    21  26 -1.888154      +     .
3    seq1   TFBS    TFBS    29  34 -3.908935      +     .
4    seq1   TFBS    TFBS     8  13 -1.961403      -     .
```

```
5    seq1   TFBS    TFBS    10  15 -3.908935      -     .
6    seq1   TFBS    TFBS    21  26 -1.961403      -     .
                                        attributes
1 TF "Arnt"; class "Zipper-Type"; sequence "CTCTTG"
2 TF "Arnt"; class "Zipper-Type"; sequence "CTCTTG"
3 TF "Arnt"; class "Zipper-Type"; sequence "AAAATG"
4 TF "Arnt"; class "Zipper-Type"; sequence "CAAGAG"
5 TF "Arnt"; class "Zipper-Type"; sequence "AACAAG"
6 TF "Arnt"; class "Zipper-Type"; sequence "CAAGAG"

> ## get the relative score
> relScore(siteset)

[1] 0.6652185 0.6652185 0.6141340 0.6633668 0.6141340 0.6633668

> relScore(sitesetList)

$MA0003.2
[1] 0.6196884

$MA0004.1
[1] 0.6652185 0.6652185 0.6141340 0.6633668 0.6141340 0.6633668
```

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.

```
> 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",
+                         min.score="50%", cutoff=0.5,
+                         strand="*", type="any")
> ## generate gff style output
> head(writeGFF3(sitePairSet))

  seqname source feature start end      score strand frame
1    seq1   TFBS    TFBS     6  20  -9.515444      +     .
2    seq1   TFBS    TFBS     7  21 -13.348617      +     .
3    seq1   TFBS    TFBS     8  22 -13.182322      +     .
4    seq1   TFBS    TFBS     9  23  -3.729917      +     .
5    seq1   TFBS    TFBS    10  24  -7.677850      +     .
```

```
6    seq1   TFBS    TFBS    14  28 -20.774619      +     .
                                              attributes
1 TF=TFAP2A;class=Zipper-Type;sequence=ACCAGCTCCCTGGCG
2 TF=TFAP2A;class=Zipper-Type;sequence=CCAGCTCCCTGGCGG
3 TF=TFAP2A;class=Zipper-Type;sequence=CAGCTCCCTGGCGGT
4 TF=TFAP2A;class=Zipper-Type;sequence=AGCTCCCTGGCGGTA
5 TF=TFAP2A;class=Zipper-Type;sequence=GCTCCCTGGCGGTAA
6 TF=TFAP2A;class=Zipper-Type;sequence=CCTGGCGGTAAGTTG
```

```
> head(writeGFF2(sitePairSet))
```

```
  seqname source feature start end      score strand frame
1    seq1   TFBS    TFBS     6  20  -9.515444      +     .
2    seq1   TFBS    TFBS     7  21 -13.348617      +     .
3    seq1   TFBS    TFBS     8  22 -13.182322      +     .
4    seq1   TFBS    TFBS     9  23  -3.729917      +     .
5    seq1   TFBS    TFBS    10  24  -7.677850      +     .
6    seq1   TFBS    TFBS    14  28 -20.774619      +     .
                                                 attributes
1 TF "TFAP2A"; class "Zipper-Type"; sequence "ACCAGCTCCCTGGCG"
2 TF "TFAP2A"; class "Zipper-Type"; sequence "CCAGCTCCCTGGCGG"
3 TF "TFAP2A"; class "Zipper-Type"; sequence "CAGCTCCCTGGCGGT"
4 TF "TFAP2A"; class "Zipper-Type"; sequence "AGCTCCCTGGCGGTA"
5 TF "TFAP2A"; class "Zipper-Type"; sequence "GCTCCCTGGCGGTAA"
6 TF "TFAP2A"; class "Zipper-Type"; sequence "CCTGGCGGTAAGTTG"
```

```
> ## search the Axt alignment
> library(CNEr)
> axtFilesHg19DanRer7 <- file.path(system.file("extdata", package="CNEr"),
+                                  "hg19.danRer7.net.axt")
> axtHg19DanRer7 <- readAxt(axtFilesHg19DanRer7)
```

```
The number of axt files 1
The number of axt alignments is 133
```

```
> sitePairSet <-  searchAln(pwm, axtHg19DanRer7, min.score="80%",
+                           windowSize=51L, cutoff=0.7, strand="*",
+                           type="any", conservation=NULL, mc.cores=2)
> GRangesTFBS <- toGRangesList(sitePairSet, axtHg19DanRer7)
> GRangesTFBS$targetTFBS
```

```
GRanges with 33 ranges and 5 metadata columns:
       seqnames              ranges strand   |   matrix.ID
          <Rle>           <IRanges>  <Rle>   | <character>
   [1]    chr11 [31128026, 31128040]      +   |    MA0003.2
   [2]    chr11 [31128028, 31128042]      +   |    MA0003.2
   [3]    chr11 [31437161, 31437175]      +   |    MA0003.2
```

```
  [4]    chr11 [31437163, 31437177]      +   |    MA0003.2
  [5]    chr11 [31499628, 31499642]      +   |    MA0003.2
  ...      ...                   ...     ... ...         ...
 [29]    chr11 [32198154, 32198168]      +   |    MA0003.2
 [30]    chr11 [32221916, 32221930]      +   |    MA0003.2
 [31]    chr11 [32221918, 32221932]      +   |    MA0003.2
 [32]    chr11 [32456353, 32456367]      +   |    MA0003.2
 [33]    chr11 [32456355, 32456369]      +   |    MA0003.2
     matrix.strand abs.score rel.score        sitesSeq
       <character> <numeric> <numeric>  <DNAStringSet>
  [1]             +  2.089093 0.8130353 GGCCCCCCGTGGCGT
  [2]             -  1.404957 0.8058956 CCCCCCGTGGCGTTT
  [3]             +  7.785995 0.8724886 ACTGGCCTAGGGACA
  [4]             -  7.796633 0.8725997 TGGCCTAGGGACAGG
  [5]             +  2.239032 0.8146001 TCTGACCCAGAGGCA
  ...           ...       ...       ...             ...
 [29]             +  2.511814 0.8174469 ACCGTCTTTAGGGGA
 [30]             +  4.672261 0.8399935 TCTACCCTCGAGCAC
 [31]             - 10.609562 0.9019556 TACCCTCGAGCACTG
 [32]             +  4.204543 0.8351123 AAGGGCCCGTAGCGA
 [33]             -  1.176965 0.8035163 GGGCCCGTAGCGACA
 ---
 seqlengths:
  chr11 chr25  chr7
     NA    NA    NA
```

```
> GRangesTFBS$queryTFBS
```

```
GRanges with 33 ranges and 5 metadata columns:
       seqnames              ranges strand   |   matrix.ID
          <Rle>           <IRanges>  <Rle>   | <character>
   [1]    chr25 [15057430, 15057444]      +   |    MA0003.2
   [2]    chr25 [15057432, 15057446]      +   |    MA0003.2
   [3]    chr25 [15149515, 15149529]      +   |    MA0003.2
   [4]    chr25 [15149517, 15149531]      +   |    MA0003.2
   [5]    chr25 [15163682, 15163696]      +   |    MA0003.2
   ...      ...                 ...     ... ...         ...
  [29]    chr25 [15359169, 15359183]      +   |    MA0003.2
  [30]    chr25 [15472022, 15472036]      +   |    MA0003.2
  [31]    chr25 [15472024, 15472038]      +   |    MA0003.2
  [32]    chr25 [15570786, 15570800]      +   |    MA0003.2
  [33]    chr25 [15570789, 15570803]      +   |    MA0003.2
      matrix.strand abs.score rel.score        sitesSeq
        <character> <numeric> <numeric>  <DNAStringSet>
   [1]             +  2.656516 0.8189570 GGCCCCCTGCGGCTC
   [2]             -  5.005684 0.8434731 CCCCCTGCGGCTCTC
```

```
 [3]             +  5.686105 0.8505740 TCTCGCCTAAGGAAA
 [4]             -  6.434853 0.8583880 TCGCCTAAGGAAAAA
 [5]             +  1.492573 0.8068100 TGCCACCCTTGGGCA
 ...           ...       ...       ...             ...
[29]             +  3.134736 0.8239477 ACCATCTTTAGGGGA
[30]             +  1.340569 0.8052237 TCTCCCCTCCAGCTC
[31]             -  5.674186 0.8504496 TCCCCTCCAGCTCTG
[32]             +  1.604621 0.8079793 CCGTACCTGCAGGCC
[33]             -  5.261771 0.8461456 TACCTGCAGGCCCCT
---
seqlengths:
 chr11 chr25  chr7
    NA    NA    NA
```

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.

```
> 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 external pattern generators

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.

```
> motifSet = runMEME(file.path(system.file("extdata",
+                                          package="TFBSTools"), "crp0.s"),
+                    binary="meme",
```

```
+                    arguments=list("-nmotifs"=3)
+                   )
> sitesSeq(motifSet, type="all")
> sitesSeq(motifSet, type="none")
> consensusMatrix(motifSet)
```

7 Conclusion

The following is the session info that generated this vignette:

```
>   sessionInfo()

R version 3.1.0 RC (2014-04-02 r65358)
Platform: i386-w64-mingw32/i386 (32-bit)

locale:
[1] LC_COLLATE=C
[2] LC_CTYPE=English_United States.1252
[3] LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.1252

attached base packages:
[1] parallel  stats     graphics  grDevices utils     datasets
[7] methods   base

other attached packages:
[1] CNEr_1.0.0          JASPAR2014_0.99.5   Biostrings_2.32.0
[4] XVector_0.4.0       IRanges_1.21.45     BiocGenerics_0.10.0
[7] TFBSTools_1.2.0

loaded via a namespace (and not attached):
 [1] BBmisc_1.5                BSgenome_1.32.0
 [3] BatchJobs_1.2             BiocParallel_0.6.0
 [5] DBI_0.2-7                 DirichletMultinomial_1.6.0
 [7] GenomeInfoDb_1.0.0        GenomicAlignments_1.0.0
 [9] GenomicRanges_1.16.0      RCurl_1.95-4.1
[11] RSQLite_0.11.4            Rcpp_0.11.1
[13] Rsamtools_1.16.0          XML_3.98-1.1
[15] bitops_1.0-6              brew_1.0-6
[17] caTools_1.16              codetools_0.2-8
[19] digest_0.6.4              fail_1.2
[21] foreach_1.4.2             grid_3.1.0
[23] gtools_3.3.1              iterators_1.0.7
[25] plyr_1.8.1                rtracklayer_1.24.0
[27] sendmailR_1.1-2           seqLogo_1.30.0
```

```
[29] stats4_3.1.0              stringr_0.6.2
[31] tools_3.1.0               zlibbioc_1.10.0
```

References

Bailey TL, Elkan C (1994). "Fitting a mixture model by expectation maximization to discover motifs in biopolymers." *Proc Int Conf Intell Syst Mol Biol*, **2**, 28–36. ISSN 1553-0833. PMID: 7584402.

Lenhard B, Wasserman WW (2002). "TFBS: Computational framework for transcription factor binding site analysis." *Bioinformatics*, **18**(8), 1135–1136. ISSN 1367-4803. PMID: 12176838.

Linhart C, Halperin Y, Shamir R (2008). "Transcription factor and microRNA motif discovery: the Amadeus platform and a compendium of metazoan target sets." *Genome Res.*, **18**(7), 1180–1189. ISSN 1088-9051. doi:10.1101/gr.076117.108. PMID: 18411406.

Nishida K, Frith MC, Nakai K (2009). "Pseudocounts for transcription factor binding sites." *Nucleic Acids Res.*, **37**(3), 939–944. ISSN 1362-4962. doi:10.1093/nar/gkn1019. PMID: 19106141.

Schneider TD, Stormo GD, Gold L, Ehrenfeucht A (1986). "Information content of binding sites on nucleotide sequences." *J. Mol. Biol.*, **188**(3), 415–431. ISSN 0022-2836. PMID: 3525846.

Wasserman WW, Sandelin A (2004). "Applied bioinformatics for the identification of regulatory elements." *Nature Publishing Group*, **5**(4), 276–287.