Overview of the *PWMEnrich* package

Robert Stojnić*

October 13, 2014

Contents

1 Introduction

The main functionality of the package is Position Weight Matrix (PWM)[1] enrichment analysis in a single sequence (e.g. enhancer of interest) or a set of sequences (e.g. set of ChIP-chip/seq peaks). Note that this is not the same as *de-novo* motif finding which discovers novel motifs, nor motif comparison which aligns motifs.

The package is built upon `Biostrings` and offers high-level functions to scan for DNA motif occurrences and compare them against a genomic background. There are multiple packages with pre-compiled genomic backgrounds such as `PWMEnrich.Dmelanogaster.background`, `PWMEnrich.Hsapiens.background` and `PWMEnrich.Mmusculus.background`. In these packages the genomic distribution is calculated for motifs from the `MotifDb` database. The `PWMEnrich` package contains all the functions used to create these packages, so you can calculate your own background distributions for your own set of motifs. In this vignette we will use the *Drosophila* package, but the other background packages are used in the same way (see Section 5 for minor human-specific differences).

*e-mail: robert.stojnic@gmail.com, Cambridge Systems Biology Institute, University of Cambridge, UK
[1] In this vignette we use "PWM", "DNA motif" and "motif" interchangeably.

1.1 Implemented algorithms

`PWMEnrich` uses the PWM scanning algorithm implemented by the package `Biostrings`. This package returns PWM scores at each position on one strand of a sequence. `PWMEnrich` extends this with a higher-level functions which automatically scans both strands for multiple motifs and sequences.

The main goal of the package is to assess the enrichment of motif hits in a sequence (or group of sequences) compared to a genomic background. The traditional way of doing this is to use a threshold for the PWM score and count the number of motif hits in the sequence(s) of interest. Since this converts the sequence into a binary bound/not-bound string, the enrichment of binding events can be assessed using a binomial formula. The `PWMEnrich` package implements this algorithm, but by default uses a lognormal threshold-free approach (Stojnic and Adryan, 2014) which is related to the score used in Clover (Frith et al., 2004).

In the lognormal threshold-free approach average affinity is calculated over the whole sequence (or set of sequences) and compared to the average affinity of length-matched sequences from the genomic background. This approach performs better or same as the best threshold approach (Stojnic and Adryan, 2014), with the added benefit of not having to choose a threshold or compare the results for multiple thresholds. We will use this threshold-free approach in all of our examples. Please consult the reference manual on how to use the fixed-threshold algorithms.

1.2 S4 class structure and accessors

As the *PWMEnrich* package builds upon the *Biostrings* package it uses the classes from this package to represent DNA sequences (`DNAString` and `DNAStringSet`). FASTA files can be loaded using functions from *Biostrings* such as `readDNAStringSet`. The package introduces a new class `PWM` to represent a PWM together with the frequency matrix and other parameters (background nucleotide frequencies and pseudo-counts). All motif scoring is performed by the *Biostrings* package which is why the *PWMEnrich* package also returns log2 scores instead of more common log base e scores.

The results of motif scanning are stored in objects of class `MotifEnrichmentResults` and `MotifEnrichmentReport`. The package also introduces a number of classes that represent different background distributions: `PWMLognBackground`, `PWMCutoffBackground`, `PWMEmpiricalBackground`, `PWMGEVBackground`. In all cases, the classes are implemented with a list-like interface, that is, individual pieces of information within the objects are accessibly using `names(obj)` and `obj$prop`.

2 Use case 1: Finding enrichment motifs in a single sequence

One of the most well-known example of combinatorial control by transcription factors in *Drosophila* is the *even skipped (eve)* stripe 2 enhancer. This well-studied enhancer has a number of annotated binding sites for TFs *Kr*, *vfl*, *bcd*, *hb* and *gt*. We will use this enhancer as an example as we already know its functional structure.

In order to predict which TFs are likely to functionally bind to the stripe 2 enhancer, we will calculate motif enrichment for a set of experimentally derived motifs from the *MotifDb* database. We will do this by comparing the average affinity of each motif in the stripe 2 enhancers to the affinity over all *D. melanogaster* promoters[2]. These background distributions are already precalculated in the `PWMEnrich.Dmelanogaster.background` package which we will simply load and use. See the last section of this vignette for using your own motifs and background sequences.

```
> library(PWMEnrich)
> library(PWMEnrich.Dmelanogaster.background)
```

[2]For more information see (Stojnic and Adryan, 2014)

```
> # load the pre-compiled lognormal background
> data(PWMLogn.dm3.MotifDb.Dmel)
> # load the stripe2 sequences from a FASTA file for motif enrichment
> sequence = readDNAStringSet(system.file(package="PWMEnrich",
+   dir="extdata", file="stripe2.fa"))
> sequence

  A DNAStringSet instance of length 1
    width seq                                                names
[1]   484 GGTTACCCGGTACTGCATAACAA...AATGATGTCGAAGGGATTAGGGG  eve_stripe2

> # perform motif enrichment!
> res = motifEnrichment(sequence, PWMLogn.dm3.MotifDb.Dmel)

Calculating motif enrichment scores ...

> report = sequenceReport(res, 1)
> report

An object of class 'MotifEnrichmentReport':
    rank target                                                    id
1      1     oc   Dmelanogaster-FlyFactorSurvey-Oc_SOLEXA_FBgn0004102
2      2    bcd  Dmelanogaster-FlyFactorSurvey-bcd_FlyReg_FBgn0000166
3      3   Ptx1 Dmelanogaster-FlyFactorSurvey-Ptx1_SOLEXA_FBgn0020912
4      4    bcd    Dmelanogaster-FlyFactorSurvey-Bcd_Cell_FBgn0000166
5      5    bcd  Dmelanogaster-FlyFactorSurvey-Bcd_SOLEXA_FBgn0000166
6      6    Gsc  Dmelanogaster-FlyFactorSurvey-Gsc_SOLEXA_FBgn0010323
7      7    Gsc    Dmelanogaster-FlyFactorSurvey-Gsc_Cell_FBgn0010323
8      8   Ptx1               Dmelanogaster-JASPAR_CORE-Ptx1-MA0201.1
9      9      D       Dmelanogaster-FlyFactorSurvey-D_NAR_FBgn0000411
10    10    Gsc                Dmelanogaster-JASPAR_CORE-Gsc-MA0190.1
...  ...    ...                                                   ...
709  709    vis  Dmelanogaster-FlyFactorSurvey-Vis_SOLEXA_FBgn0033748
             raw.score              p.value
1     12.0647987758141 0.000376592081390237
2     5.63411908732576 0.000412409209523563
3     21.2538368223138 0.000649473662007989
4     16.8158641518872 0.000748084265069388
5     6.52627803922005  0.00163314432973656
6     6.61030691892303  0.00164152477278935
7     8.57034891276624  0.00202747679807863
8     12.5061755821191  0.00230701519060613
9     23.1334053023326  0.00267959880398655
10    6.40551327159533  0.00281963918165213
...                ...                  ...
709 0.0136301331722268    0.999904947676631

> # plot the top 30 most enriched motifs
> plot(report[1:30], fontsize=7, id.fontsize=5)
```

Rank	Target	PWM	Motif ID	Raw score	P-value
1	oc		Dmelanogaster-FlyFactorSurvey-Oc_SOLEXA_FBgn0004102	12.1	0.000377
2	bcd		Dmelanogaster-FlyFactorSurvey-bcd_FlyReg_FBgn0000166	5.63	0.000412
3	Ptx1		Dmelanogaster-FlyFactorSurvey-Ptx1_SOLEXA_FBgn0020912	21.3	0.000649
4	bcd		Dmelanogaster-FlyFactorSurvey-Bcd_Cell_FBgn0000166	16.8	0.000748
5	bcd		Dmelanogaster-FlyFactorSurvey-Bcd_SOLEXA_FBgn0000166	6.53	0.00163
6	Gsc		Dmelanogaster-FlyFactorSurvey-Gsc_SOLEXA_FBgn0010323	6.61	0.00164
7	Gsc		Dmelanogaster-FlyFactorSurvey-Gsc_Cell_FBgn0010323	8.57	0.00203
8	Ptx1		Dmelanogaster-JASPAR_CORE-Ptx1-MA0201.1	12.5	0.00231
9	D		Dmelanogaster-FlyFactorSurvey-D_NAR_FBgn0000411	23.1	0.00268
10	Gsc		Dmelanogaster-JASPAR_CORE-Gsc-MA0190.1	6.41	0.00282
11	bcd		Dmelanogaster-FlyFactorSurvey-bcd_NAR_FBgn0000166	3.49	0.00314
12	oc		Dmelanogaster-JASPAR_CORE-oc-MA0234.1	7.05	0.00317
13	bcd		Dmelanogaster-JASPAR_CORE-bcd-MA0212.1	7.33	0.00333
14	CG12768		Dmelanogaster-FlyFactorSurvey-CG12768_SANGER_5_FBgn0037206	18.3	0.00461
15	oc		Dmelanogaster-FlyFactorSurvey-Oc_Cell_FBgn0004102	6.97	0.00512
16	gt		Dmelanogaster-FlyFactorSurvey-gt_FlyReg_FBgn0001150	8.35	0.00568
17	CG3407		Dmelanogaster-FlyFactorSurvey-CG3407_SANGER_2.5_FBgn0031573	8.66	0.00583
18	D		Dmelanogaster-JASPAR_CORE-D-MA00445.1	20	0.00709
19	CAD		Dmelanogaster-JASPAR_2014-CAD-MA0216.2	25.8	0.00925
20	Her		Dmelanogaster-FlyFactorSurvey-Her_SANGER_5_FBgn0030899	5.21	0.0144
21	CG3407		Dmelanogaster-FlyFactorSurvey-CG3407_SOLEXA_2.5_FBgn0031573	10.3	0.0155
22	lola		Dmelanogaster-FlyFactorSurvey-lola.PA_SANGER_5_FBgn0005630	3.56	0.0195
23	kni		Dmelanogaster-FlyFactorSurvey-kni_SANGER_5_FBgn0001320	8.99	0.0201
24	E(spl)mgamma-HLH		Dmelanogaster-FlyFactorSurvey-HLHmgamma_SANGER_5_2_FBgn0002735	4.28	0.0216
25	zen		Dmelanogaster-JASPAR_CORE-zen-MA0256.1	4.25	0.0229
26	ttk		Dmelanogaster-FlyFactorSurvey-ttk.PF_SANGER_5_FBgn0003870	5.22	0.0238
27	eg		Dmelanogaster-FlyFactorSurvey-eg_SANGER_5_FBgn0000560	7.66	0.0243
28	Kr		Dmelanogaster-JASPAR_CORE-Kr-MA0452.1	4.34	0.0354
29	Kr		Dmelanogaster-FlyFactorSurvey-Kr_FlyReg_FBgn0001325	3.09	0.0371
30	Abd-B		Dmelanogaster-FlyFactorSurvey-Abd.B_FlyReg_FBgn0000015	3.9	0.0379

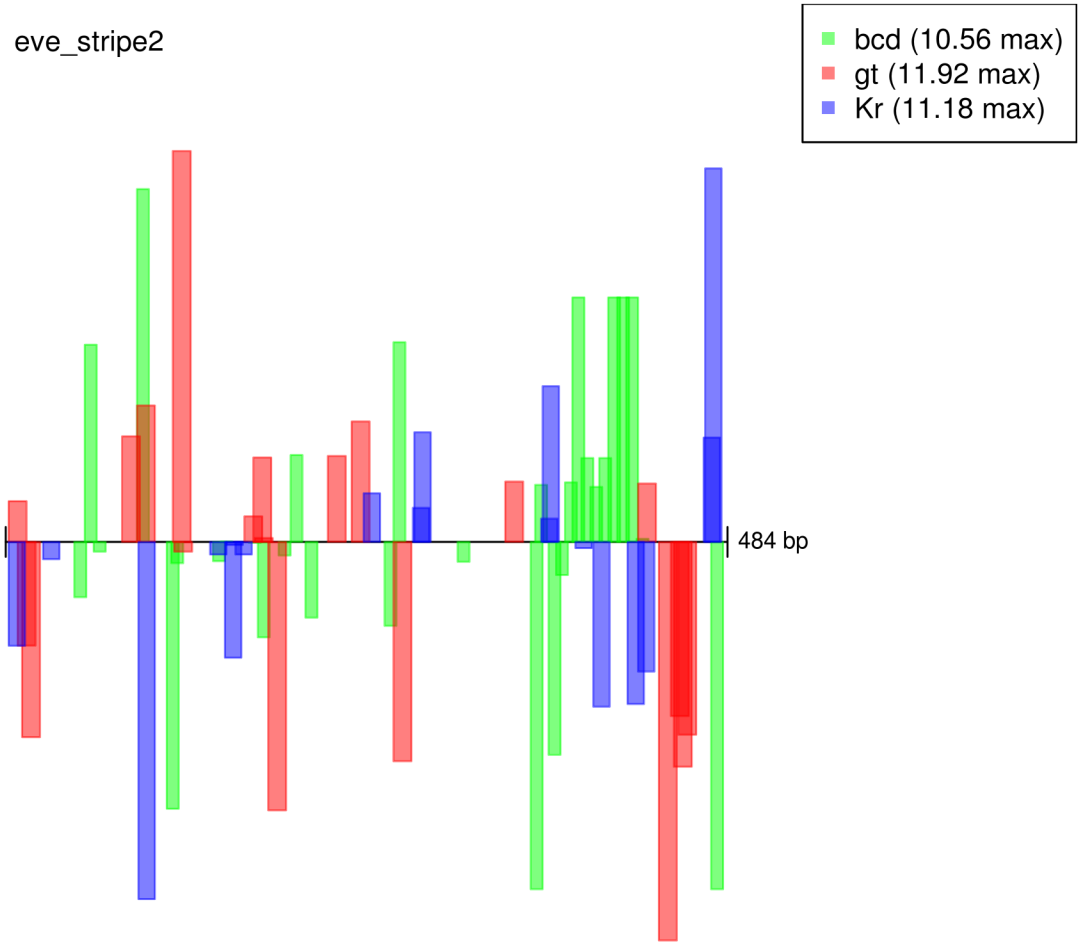
The main function we used is `motifEnrichment` which took our sequence and calculated motif enrichment using the lognormal affinity background distribution (fitted on a set of 10031 *D. melanogaster* 2kb promoters). This function returns a set of scores and P-values for our sequence. We then used the `sequenceReport` function that create a ranked list of motifs, which we then plot using `plot`. The first column is the rank, the second shows the target name, which is either a gene name, an isoform name (such as ttk-PF), or a dimer name (such as tgo_sim not present in this list). The next column in the plot is the PWM logo, and after that the motif ID. This ID comes from the `MotifDb` package and can be used to look up further information about the motif (such as the motif source). The next-to-last column is the raw affinity score, and the last column is the P-value of motif enrichment.

As we can see, the top of the list is dominated by motifs similar to bcd. By further examining the list, we find we recovered the Kr, bcd and gt motifs, but not the vfl and hb motifs. These two TFs (vfl and hb) have the smallest number of annotated binding sites out of the five TFs in the stripe 2 enhancer. As a result, this affinity is not large enough to be picked up by motif enrichment. However, the other three motifs were picked up. We find this to be the typical case for many enhancers.

3 Use case 2: Examining the binding sites

We continue with our example of the eve stripe 2 enhancer from the previous section. We now want to visualise the binding sites for Kr, bcd and gt.

```
> # extract the 3 PWMs for the TFs we are interested in
> ids = c("Dmelanogaster-FlyFactorSurvey-bcd_FlyReg_FBgn0000166",
+         "Dmelanogaster-FlyFactorSurvey-gt_FlyReg_FBgn0001150",
+         "Dmelanogaster-JASPAR_CORE-Kr-MA0452.1")
> sel.pwms = PWMLogn.dm3.MotifDb.Dmel$pwms[ids]
> names(sel.pwms) = c("bcd", "gt", "Kr")
> # scan and get the raw scores
> scores = motifScores(sequence, sel.pwms, raw.scores=TRUE)
> # raw scores for the first (and only) input sequence
> dim(scores[[1]])

[1] 968   3

> head(scores[[1]])

              bcd           gt           Kr
[1,] 7.484914e-05 4.213929e-05 1.141957e-07
[2,] 9.180413e-02 4.275114e-04 1.162378e-03
[3,] 1.020698e-02 2.326263e+00 1.480311e-02
[4,] 2.206202e-07 4.600757e-07 2.085725e-07
[5,] 7.044890e-06 7.690586e-07 1.638103e-06
[6,] 4.913950e-04 7.229475e-08 4.625971e-07

> # score starting at position 1 of forward strand
> scores[[1]][1, "bcd"]

         bcd
7.484914e-05

> # score for the reverse complement of the motif, starting at the same position
> scores[[1]][485, "bcd"]

         bcd
2.055192e-06

> # plot
> plotMotifScores(scores, cols=c("green", "red", "blue"))
```



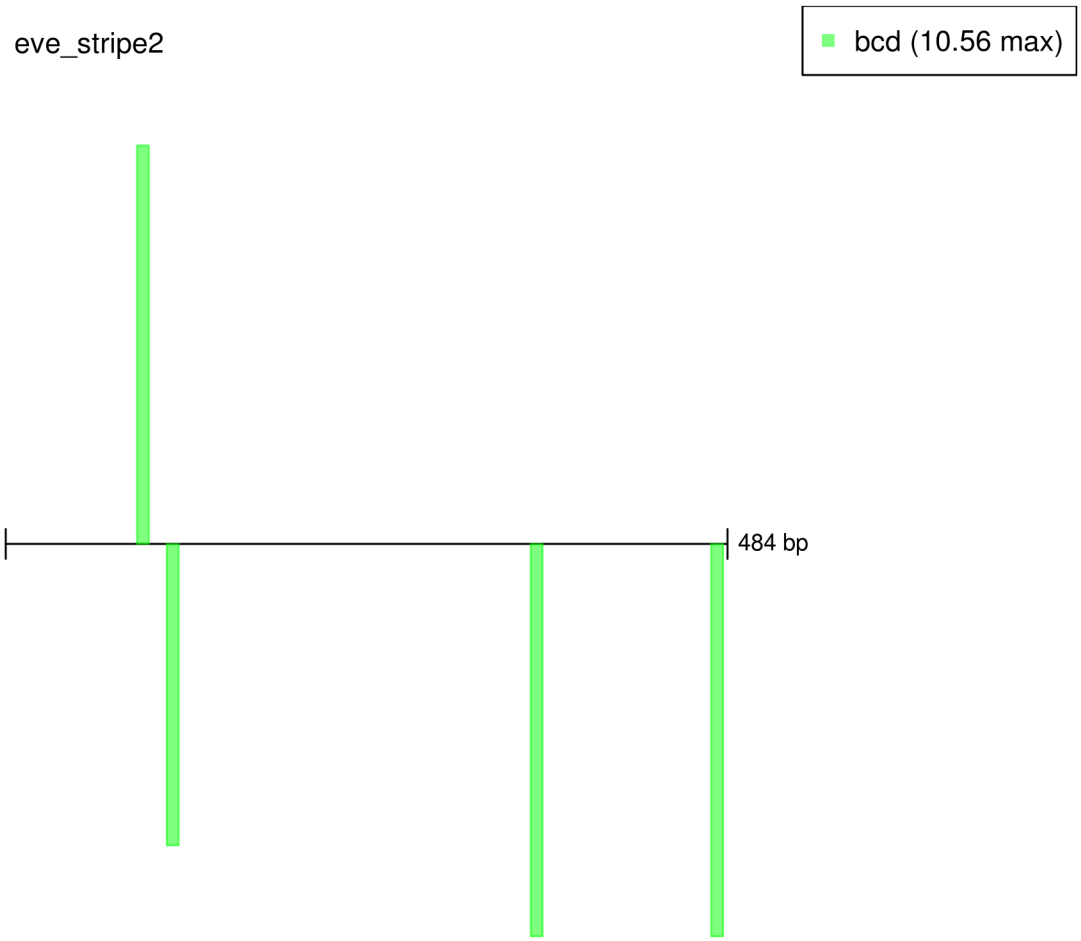
Here we used the `motifScores` function to obtain the raw scores at each position in the sequence. The result of this function is a list of matrices, each element of the list corresponding to an input sequence. In this case we had only one input sequence, and as a result we get a list of length 1. The matrix of scores is a 968 x 3 matrix, where the rows correspond to the two strands (2 x 484) and the columns correspond to motifs. It is important to remember that the scores are in real and not log space. In other words, a conventional PWM log2 score of 3 is represented as number 8 (2^3).

The scores for the two strands are concatenated one after the other. Therefore, row 1 has the scores for the motif starting at position 1, and row 485 has the score at the same position, but with the reverse complement of the motif (i.e. motif score on the reverse strand). Note that there will be some NA values at the end of the sequence (e.g. position 484) because we do not support partial motif matches.

Finally we use the `plotMotifScores` function to plot the log2 scores over the sequence. We colour-code the motifs with green, red and blue. The motif hits are shown as rectangles with the base being the length of the motif, and the hight being the log2 score of the motif hit. By default we show all motif hits with log2 scores larger then 0. The forward strand hits are shown on the top, and the reverse strand hits are shown on the bottom.

We next might be interested in finding the P-value for individual motif hits so we can get an idea which sites are the most important. To do this we need to calculate the empirical PWM score distribution for single sites. We did not provide these values precalculated because they take up a very large amount of memory. To calculate it based on a set of promoter, we will need the *D. melanogaster* genome sequence. Because the objects are so large, in this example we will determine the P-value only for the hits of the bcd motif, using only a small subset of promoters (controlled by the parameter `quick=TRUE`).

```
> library(BSgenome.Dmelanogaster.UCSC.dm3)
> # empirical distribution for the bcd motif
> bcd.ecdf = motifEcdf(sel.pwms$bcd, Dmelanogaster, quick=TRUE)[[1]]
> # find the score that is equivalent to the P-value of 1e-3
> threshold.1e3 = log2(quantile(bcd.ecdf, 1 - 1e-3))
> threshold.1e3

   99.9%
7.334504

> # replot only the bcd motif hits with the P-value cutoff of 1e-3 (0.001)
> plotMotifScores(scores, cols="green", sel.motifs="bcd", cutoff=threshold.1e3)
> # P-value at each position
> pvals = 1 - bcd.ecdf(scores[[1]][,"bcd"])
> # position where the P-value is smaller that 1e-3
> which(pvals < 1e-3)

[1]  89 593 837 958
```



Here we have used the `motifEcdf` function to create an empirical cumulative distribution function (ECDF) for the bcd motif score on Drosophila promoters. This function returns an `ecdf` object which is part of base R. We can then use the quantile function to find which scores correspond to a P-value of 0.001, or we can use it to convert all the scores into P-values (not shown above). To plot the individual motif hits with P-values smaller than 0.001 we again use the `plotMotifScores` function, but now we apply the threshold so that only those motif hits above the threshold are drawn.

In the last line we find out the positions of those motif hits where the P-value is smaller then 1e-3. Note that the values larger than the sequence length (484) indicate the reverse strand. Therefore, we find the four strong motif hits at positions 90 on the forward strand and 110, 354 and 475 on the reverse strand.

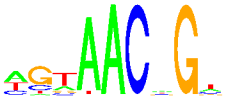
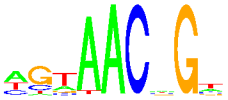
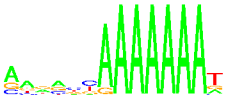
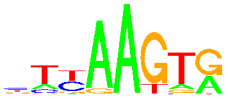
Note that `plotMotifScores` can also plot multiple sequences on a single plot, and that the `cutoff` parameter can contain a vector of values if we wish to apply different cutoff to different motifs.

4 Use case 3: Finding enriched motifs in multiple sequences

So far we have only looked at motif enrichment in a single sequence, which was able to recover some but not all of the truly functional motifs. The power of the motif enrichment approach can be significantly boosted by performing it jointly on multiple sequences.

For this example we are going to use the top 20 ChIP-chip peaks for transcription factor Tinman in *Drosophila* (Jin et al., 2013). We are going to scan these 20 ChIP-chip peaks with all the motifs and then compare their enrichment to genomic background. Running on the whole set of peaks (i.e. thousands) is also possible but can take a long time (i.e. tens of minutes). The speed can be improved by using multiple CPU cores (see next section).

```
> library(PWMEnrich.Dmelanogaster.background)
> # load the pre-compiled lognormal background
> data(PWMLogn.dm3.MotifDb.Dmel)
> sequences = readDNAStringSet(system.file(package="PWMEnrich",
+   dir="extdata", file="tinman-early-top20.fa"))
> res = motifEnrichment(sequences, PWMLogn.dm3.MotifDb.Dmel)

Calculating motif enrichment scores ...

> report = groupReport(res)
> report

An object of class 'MotifEnrichmentReport':
    rank  target                                                        id
1      1     vnd        Dmelanogaster-FlyFactorSurvey-Vnd_SOLEXA_FBgn0003986
2      2     tin                      Dmelanogaster-JASPAR_CORE-tin-MA0247.1
3      3     tin                      Dmelanogaster-JASPAR_2014-tin-MA0247.2
4      4 CG16778  Dmelanogaster-FlyFactorSurvey-CG16778_SANGER_5_FBgn0003715
5      5     vnd                      Dmelanogaster-JASPAR_CORE-vnd-MA0253.1
6    6.5     ovo                      Dmelanogaster-JASPAR_CORE-ovo-MA0126.1
7    6.5     prd                      Dmelanogaster-JASPAR_CORE-prd-MA0239.1
8      8   Lin29 Dmelanogaster-FlyFactorSurvey-CG2052_SANGER_2.5_FBgn0039905
9      9     tin        Dmelanogaster-FlyFactorSurvey-tin_FlyReg_FBgn0004110
10    10     tin        Dmelanogaster-FlyFactorSurvey-Tin_SOLEXA_FBgn0004110
...  ...     ...                                                         ...
709  709     pnr                      Dmelanogaster-JASPAR_2014-pnr-MA0536.1
            raw.score              p.value top.motif.prop
1      1.331642560015 2.99289122316941e-05           0.45
2    2.30913191829455 5.00378175587895e-05            0.4
3    4.15348380046138 0.000716366551558128           0.35
4    2.44073158041219  0.00163387746548455            0.3
5    1.99855914798949  0.00166810428702504            0.3
6    1.04282788163688   0.0019029014551596           0.15
7    1.04282788163688   0.0019029014551596           0.15
8    8.93615013353702  0.00191384687276172            0.2
9    4.56378298273599    0.002270048621663           0.25
10   1.22390594864211  0.00237440092243837            0.2
...               ...                 ...            ...
709 0.365937057198077    0.999999999367786              0

> plot(report[1:10], fontsize=7, id.fontsize=5)
```

Rank	Target	PWM	Motif ID	Raw score	P-value	In top motifs
1	vnd		Dmelanogaster-FlyFactorSurvey-Vnd_SOLEXA_FBgn0003986	1.33	2.99e-05	45 %
2	tin		Dmelanogaster-JASPAR_CORE-tin-MA0247.1	2.31	5e-05	40 %
3	tin		Dmelanogaster-JASPAR_2014-tin-MA0247.2	4.15	0.000716	35 %
4	CG16778		Dmelanogaster-FlyFactorSurvey-CG16778_SANGER_5_FBgn0003715	2.44	0.00163	30 %
5	vnd		Dmelanogaster-JASPAR_CORE-vnd-MA0253.1	2	0.00167	30 %
6.5	ovo		Dmelanogaster-JASPAR_CORE-ovo-MA0126.1	1.04	0.0019	15 %
6.5	prd		Dmelanogaster-JASPAR_CORE-prd-MA0239.1	1.04	0.0019	15 %
8	Lin29		Dmelanogaster-FlyFactorSurvey-CG2052_SANGER_2.5_FBgn0039905	8.94	0.00191	20 %
9	tin		Dmelanogaster-FlyFactorSurvey-tin_FlyReg_FBgn0004110	4.56	0.00227	25 %
10	tin		Dmelanogaster-FlyFactorSurvey-Tin_SOLEXA_FBgn0004110	1.22	0.00237	20 %

As in Use case 1, the main function is `motifEnrichment` which took our sequences and calculated motif enrichment using the lognormal affinity background distribution (fitted on a set of 10031 *D. melanogaster* 2kb promoters). We then applied the `groupReport` function to calculate the enrichment over the whole group of sequences. This produced a ranked list of motifs according to the estimated P-values. Then we used `plot` to plot the top 10 enriched motifs.

The first three motifs are very similar and correspond to the tinman, which is the transcription factor for which the ChIP-chip experiment was performed. The first five columns are the same as before (see Use case 1). The sixth column gives the estimate P-value. The last column indicates the breadth of enrichment using a 5% ranking threshold. This column helps to differentiate cases where the motif enrichment is strongly focused to a small subset of sequences (in which case breadth is small), versus being more widespread but weaker (in which case breadth is bigger). We can also sort by this column:

```
> report.top = groupReport(res, by.top.motifs=TRUE)
> report.top

An object of class 'MotifEnrichmentReport':
    rank  target                                                    id
```

```
1      1     vnd       Dmelanogaster-FlyFactorSurvey-Vnd_SOLEXA_FBgn0003986
2      2     tin                     Dmelanogaster-JASPAR_CORE-tin-MA0247.1
3      3     tin                     Dmelanogaster-JASPAR_2014-tin-MA0247.2
4    4.5 CG16778 Dmelanogaster-FlyFactorSurvey-CG16778_SANGER_5_FBgn0003715
5    4.5     vnd                     Dmelanogaster-JASPAR_CORE-vnd-MA0253.1
6     10    Aef1    Dmelanogaster-FlyFactorSurvey-Aef1_SANGER_5_FBgn0005694
7     10     fru     Dmelanogaster-FlyFactorSurvey-fru_SOLEXA_5_FBgn0004652
8     10     ken     Dmelanogaster-FlyFactorSurvey-ken_SOLEXA_5_FBgn0011236
9     10     klu     Dmelanogaster-FlyFactorSurvey-klu_SOLEXA_5_FBgn0013469
10    10    lola Dmelanogaster-FlyFactorSurvey-lola.PC_SANGER_5_FBgn0005630
...  ...     ...                                                        ...
709  539     ttk                     Dmelanogaster-JASPAR_CORE-ttk-MA0460.1
            raw.score              p.value top.motif.prop
1      1.331642560015 2.99289122316941e-05           0.45
2    2.30913191829455 5.00378175587895e-05            0.4
3    4.15348380046138 0.000716366551558128           0.35
4    2.44073158041219  0.00163387746548455            0.3
5    1.99855914798949  0.00166810428702504            0.3
6    31.3169137912424   0.0863152680572984           0.25
7    1.43439128314498   0.0476671377392103           0.25
8    2.34041244967485   0.0479090721465239           0.25
9    14.9381391375605   0.0492071601987718           0.25
10    1.9130968547286   0.0891502351674363           0.25
...               ...                  ...            ...
709 0.418754284960579    0.992543321683472              0
```

This ranks motifs by breadth of enrichment, which is calculated by comparing enrichment *between motifs* in individual sequences. This measure only makes sense when applied to a large number of sequence and when scanning with a large number of motifs (>20).

The object returned by `motifEnrichment` has more information in it, as can be seen below:

```
> res

An object of class 'MotifEnrichmentResults':
* created with 'affinity' scoring function with 'logn' background correction
* on a set of 20 sequence(s) and 709 PWMs
Result sets for the group: $group.nobg, $group.bg, $group.norm
Result sets for individual sequences: $sequence.nobg, $sequence.bg, $sequence.norm
Report methods: groupReport(), sequenceReport()

> # raw scores
> res$sequence.nobg[1:5, 1:2]

                 Dmelanogaster-FlyFactorSurvey-ab_SANGER_10_FBgn0259750
tinman-early_885                                              0.32258956
tinman-early_2150                                             0.13578745
tinman-early_280                                              0.01968489
tinman-early_1353                                             0.10983606
tinman-early_1624                                             2.31211300
                 Dmelanogaster-FlyFactorSurvey-ab_SOLEXA_5_FBgn0259750
tinman-early_885                                              0.05464342
tinman-early_2150                                             0.89276819
tinman-early_280                                              0.02827202
tinman-early_1353                                             0.26224401
tinman-early_1624                                             1.35872007
```

```
> # P-values
> res$sequence.bg[1:5, 1:2]

                  Dmelanogaster-FlyFactorSurvey-ab_SANGER_10_FBgn0259750
tinman-early_885                                                0.4101435
tinman-early_2150                                               0.6710786
tinman-early_280                                                0.9072467
tinman-early_1353                                               0.6338600
tinman-early_1624                                               0.1043003
                  Dmelanogaster-FlyFactorSurvey-ab_SOLEXA_5_FBgn0259750
tinman-early_885                                                0.7668959
tinman-early_2150                                               0.3027045
tinman-early_280                                                0.8664515
tinman-early_1353                                               0.4849721
tinman-early_1624                                               0.2067721
```

In these two matrices the rows correspond to the different input sequences and the columns correspond to motifs. The first matrix (sequence.nobg) contains the raw affinity scores, while the second (sequence.bg) contains the corresponding P-values. If you are using a fixed threshold background (e.g. scanning with `PWMPvalueCutoff1e3.dm3.MotifDb.Dmel`) the first matrix will contain the number of motif hits, and the second the corresponding Z-scores.

5 Using PWMEnrich on human sequences

Starting from PWMEnrich version 4.0 (October 2014) a new algorithm is used to better fit the background distributions in human sequences. From the usage perspective the only major difference is that the P-value of groups of sequences (i.e. groupReport()) is replaced with an average of log(P-values) of individual sequences. To get most complete results we recommend examining motif enrichment based on this score and breadth (last column of groupReport() output - see above).

When compiling a new background for human sequences make sure to set parameter `algorithm="human"` in `makeBackground()`.

6 Speeding up execution

6.1 Parallel execution

Motif scanning is the most time consuming operation. Because of this, the package has a support for parallel motif scanning using the *parallel* core package. Note that parallel execution is currently not supported on Windows. To turn on parallel scanning, simply register a number of cores available to the package:

```
> registerCoresPWMEnrich(4)
```

After this command is executed, all further calls to *PWMEnrich* functions are going to be run in parallel using 4 cores (if possible). To turn off parallel execution call the function with parameter NULL:

```
> registerCoresPWMEnrich(NULL)
```

6.2 Large memory backend

Motif scanning can be further speeded up by using large amount of memory. If you have an access to a machine with a lot of RAM, you can switch to the ”big memory” backend:

```
> useBigMemoryPWMEnrich(TRUE)
```

From this point on, all motif scanning will be done using the optimised big memory backend. The memory requirement depends on the number of sequences scanned, and might require tens of GB of RAM. To turn it off:

```
> useBigMemoryPWMEnrich(FALSE)
```

7 Customisation

7.1 Using a custom set of PWMs

Background motif distributions for a custom set of PWMs can be easily calculated for all model organisms. We will illustrate this by creating a new lognormal background for two *de-novo* motifs in Drosophila. To load in the motifs the package provides functions to read standard JASPAR and TRANSFAC formats.

```
> library(PWMEnrich.Dmelanogaster.background)
> motifs.denovo = readMotifs(system.file(package="PWMEnrich",
+   dir="extdata", file="example.transfac"), remove.acc=TRUE)
> motifs.denovo

$tin_like_motif
  [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14]
A   12    5    2    1    0   36   37    0    0     0     5     4     8    10
C   10    7   24    0   36    0    0    1    0     0     6    19     8     4
G   10   13    6    0    0    1    0   36    0    36    22     7     6     8
T    5   12    5   36    1    0    0    0   37     1     4     7    15    15

$gata_like_motif
  [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
A   17   17   13   42    0   42    0   42    0    21    12
C    7   12   19    0    0    0    0    0   42     5    16
G    6    6    7    0   42    0    0    0    0     8     5
T   12    7    3    0    0    0   42    0    0     8     9

> # convert count matrices into PWMs
> genomic.acgt = getBackgroundFrequencies("dm3")
> pwms.denovo = toPWM(motifs.denovo, prior=genomic.acgt)
> bg.denovo = makeBackground(pwms.denovo, organism="dm3", type="logn", quick=TRUE)

NOTE: Using the 'default' algorithm to infer background parameters,
      appropriate for all organisms except human.

> # use new motifs for motif enrichment
> res.denovo = motifEnrichment(sequences[1:5], bg.denovo)

Calculating motif enrichment scores ...

> groupReport(res.denovo)

An object of class 'MotifEnrichmentReport':
  rank          target              id raw.score      p.value top.motif.prop
1    1  tin_like_motif  tin_like_motif  9.465309 1.162377e-06              0
2    2 gata_like_motif gata_like_motif  2.544327 1.586457e-03              0
```

We load in the count matrices and then convert them into PWMs using the genomic distributions of the A, C, G, T nucleotides. Next we use these PWMs to calculate the properties of the affinity distribution on the set of *D. melanogaster* promoters. In this example we used `quick=TRUE` for illustrative purposes. This fits the parameters quickly on a reduced set of 100 promoters. We strongly discourage the users to use this parameter in their research, and instead only use it to obtain rough estimates and for testing. The resulting object `bg.denovo` can be used same as before to perform motif enrichment.

The background object `bg.denovo` contains the two PWMs and their background distribution parameters. All of these can be accessed with the $ operator.

```
> bg.denovo

An object of class 'PWMLognBackground'
Background source: D. melanogaster (dm3) 100 unique 2kb promoters
Fitted on a mean sequence length of 238 for a set of 2 PWMs
Lognormal parameters: $bg.mean, $bg.sd
PWMS: $pwms

> bg.denovo$bg.mean

 tin_like_motif gata_like_motif
      0.7538848       0.6441579
```

7.2 Using a custom set of background sequences

Low-level functions are available for constructing custom backgrounds. We start with the two denovo motifs from previous section and fit the background to first 20 *D. melanogaster* promoters.

```
> library(PWMEnrich.Dmelanogaster.background)
> data(dm3.upstream2000)
> # make a lognormal background for the two motifs using only first 20 promoters
> bg.seq = dm3.upstream2000[1:20]
> # the sequences are split into 100bp chunks and fitted
> bg.custom = makeBackground(pwms.denovo, bg.seq=bg.seq, type="logn", bg.len=100,
+         bg.source="20 promoters split into 100bp chunks")

NOTE: Using the 'default' algorithm to infer background parameters,
      appropriate for all organisms except human.

> bg.custom

An object of class 'PWMLognBackground'
Background source: 20 promoters split into 100bp chunks
Fitted on a mean sequence length of 88 for a set of 2 PWMs
Lognormal parameters: $bg.mean, $bg.sd
PWMS: $pwms
```

The resulting `bg.custom` object can be used as before for motif enrichment with the `motifEnrichment` function (as described before).

8 Session information

- R version 3.1.1 Patched (2014-09-25 r66681), `x86_64-unknown-linux-gnu`

- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C, LC_MONETARY=en_US.UTF-8, LC_MESSAGES=en_US.UTF-8, LC_PAPER=en_US.UTF-8, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, stats, stats4, utils
- Other packages: BSgenome 1.34.0, BSgenome.Dmelanogaster.UCSC.dm3 1.4.0, BiocGenerics 0.12.0, Biostrings 2.34.0, GenomeInfoDb 1.2.0, GenomicRanges 1.18.0, IRanges 2.0.0, PWMEnrich 4.2.0, PWMEnrich.Dmelanogaster.background 4.0.2, S4Vectors 0.4.0, XVector 0.6.0, rtracklayer 1.26.0
- Loaded via a namespace (and not attached): BBmisc 1.7, BatchJobs 1.4, BiocParallel 1.0.0, DBI 0.3.1, GenomicAlignments 1.2.0, RCurl 1.95-4.3, RSQLite 0.11.4, Rsamtools 1.18.0, XML 3.98-1.1, base64enc 0.1-2, bitops 1.0-6, brew 1.0-6, checkmate 1.4, codetools 0.2-9, digest 0.6.4, evd 2.3-0, fail 1.2, foreach 1.4.2, gdata 2.13.3, gtools 3.4.1, iterators 1.0.7, sendmailR 1.2-1, seqLogo 1.32.0, stringr 0.6.2, tools 3.1.1, zlibbioc 1.12.0

References

Frith, M. C., Fu, Y., Yu, L., Chen, J., Hansen, U., and Weng, Z. (2004). Detection of functional DNA motifs via statistical over-representation. *Nucl. Acids Res.*, 32(4):1372–1381.

Jin, H., Stojnic, R., Adryan, B., Ozdemir, A., Stathopoulos, A., and Frasch, M. (2013). Genome-wide screens for in vivo Tinman binding sites identify cardiac enhancers with diverse functional architectures. *PLoS Genet*, 9:e1003195.

Stojnic, R. and Adryan, B. (2014). Affinity based DNA motif enrichment analysis with R/Bioconductor package PWMEnrich. *in preparation*.