

atSNP: affinity tests for regulatory SNP detection

Chandler Zuo*
Sunyoung Shin[†]
Sündüz Keleş[‡]

Contents

1	Introduction	1
2	Installation	1
3	Example	2
3.1	Load motif and SNP data	2
3.2	Affinity score tests	5
3.3	Additional analysis	6
4	Session Information	7

1 Introduction

This document provides an introduction to the affinity test for large sets of SNP-motif interactions using the **atSNP** package(affinity test for regulatory **SNP** detection) [?]. **atSNP** implements in-silico methods for identifying SNPs that potentially may affect binding affinity of transcription factors. Given a set of SNPs and a library of motif position weight matrices (PWMs), **atSNP** provides three main functions for analyzing SNP effects:

1. Computing the binding affinity score for each allele and each PWM;
2. Computing the p-values for allele-specific binding affinity scores;
3. Computing the p-values for affinity score changes between the two alleles for each SNP.

atSNP implements the importance sampling algorithm in [Chan et al., 2010] to compute the p-values. Compared to other bioinformatics tools, such as FIMO [Grant et al., 2011] and is-rSNP [Macintyre et al., 2010], that provides similar functionalities, **atSNP** avoids computing the p-values analytically. This reduces the execution time drastically because the probability sample space is a exponential order of the motif length. In one of our research projects, we have used **atSNP** to evaluate interactions between 26K SNPs and 2K motifs within 5 hours. We found no other existing tool can finish the analysis of such a scale.

2 Installation

We are working to make the package available through bioconductor. The developing version can be installed from the Github repository:

```
> library(devtools)
> install_github("chandlerzuo/atSNP")
```

The following dependent R packages are required:

- **data.table** is used for formatting results that are easy for users to query;
- **motifStack** is relied upon to draw sequence logo plots;

*Department of Statistics and of Biostatistics and Medical Informatics, 1300 University Avenue, Madison, WI, 53706, USA.

[†]Department of Statistics and of Biostatistics and Medical Informatics, 1300 University Avenue, Madison, WI, 53706, USA.

[‡]Departments of Statistics and of Biostatistics and Medical Informatics, 1300 University Avenue, Madison, WI, 53706, USA.

- doMC is used for parallel computation;
- Rcpp interfaces the C++ codes that implements the importance sampling algorithm;
- testthat is used for unit testing.

In addition, users also need to install the annotation package from www.bioconductor.org/packages/3.0/data/annotation/ that corresponds to the species type and genome version. Our example SNP data set in the subsequent sections corresponds to the hg19 version of human genome. To repeat the sample codes in this vignette, the `BSgenome.Hsapiens.UCSC.hg19` package is required.

3 Example

3.1 Load motif and SNP data

`atSNP` provides a default motif library downloaded from compbio.mit.edu/encode-motifs/motifs.txt. This library contains 2065 known and discovered motifs from ENCODE TF ChIP-seq data sets. The following commands allow you to load this motif library:

```
> library(atSNP)
> data(encode_motif)
> length(motif_encode)
[1] 2065
> motif_encode[seq(3)]
$SIX5_disc1
      [,1]      [,2]      [,3]      [,4]
[1,] 8.51100e-03 4.2550e-03 0.987234 1.00000e-10
[2,] 9.02127e-01 1.2766e-02 0.038298 4.68090e-02
[3,] 4.55319e-01 7.2340e-02 0.344681 1.27660e-01
[4,] 2.51064e-01 8.5106e-02 0.085106 5.78724e-01
[5,] 1.00000e-10 4.6809e-02 0.012766 9.40425e-01
[6,] 1.00000e-10 1.0000e-10 1.000000 1.00000e-10
[7,] 3.82980e-02 2.1277e-02 0.029787 9.10638e-01
[8,] 9.44681e-01 4.2550e-03 0.051064 1.00000e-10
[9,] 1.00000e-10 1.0000e-10 1.000000 1.00000e-10
[10,] 1.00000e-10 1.0000e-10 0.012766 9.87234e-01

$MYC_disc1
      [,1]      [,2]      [,3]      [,4]
[1,] 1.73516e-01 1.05023e-01 7.21461e-01 1.00000e-10
[2,] 1.00000e-10 1.00000e-10 1.00000e-10 1.00000e+00
[3,] 1.00000e-10 1.00000e+00 1.00000e-10 1.00000e-10
[4,] 1.00000e+00 1.00000e-10 1.00000e-10 1.00000e-10
[5,] 1.00000e-10 9.58904e-01 1.00000e-10 4.10960e-02
[6,] 5.93610e-02 1.00000e-10 9.40639e-01 1.00000e-10
[7,] 1.00000e-10 1.00000e-10 1.00000e-10 1.00000e+00
[8,] 1.00000e-10 1.00000e-10 1.00000e+00 1.00000e-10
[9,] 1.00000e+00 1.00000e-10 1.00000e-10 1.00000e-10
[10,] 1.00000e-10 7.26028e-01 1.14155e-01 1.59817e-01

$SRF_disc1
      [,1] [,2] [,3]      [,4]
[1,] 1.00000e-10 1e+00 1e-10 1.00000e-10
[2,] 1.00000e-10 1e+00 1e-10 1.00000e-10
[3,] 4.95495e-01 1e-10 1e-10 5.04505e-01
[4,] 2.61261e-01 1e-10 1e-10 7.38739e-01
[5,] 1.00000e+00 1e-10 1e-10 1.00000e-10
[6,] 1.00000e-10 1e-10 1e-10 1.00000e+00
[7,] 7.29730e-01 1e-10 1e-10 2.70270e-01
[8,] 5.04505e-01 1e-10 1e-10 4.95495e-01
[9,] 1.00000e-10 1e-10 1e+00 1.00000e-10
[10,] 1.00000e-10 1e-10 1e+00 1.00000e-10
```

The data object 'encode_motif' also contains a character vector 'motif_info' that contains detailed information for each motif:

```
> length(motif_info)
[1] 2065
> head(motif_info)

                                SIX5_disc1
"SIX5_GM12878_encode-Myers_seq_hsa_r1:MEME#1#Intergenic"
                                MYC_disc1
"USF2_K562_encode-Snyder_seq_hsa_r1:MDscan#1#Intergenic"
                                SRF_disc1
"SRF_H1-hESC_encode-Myers_seq_hsa_r1:MDscan#2#Intergenic"
                                AP1_disc1
"JUND_K562_encode-Snyder_seq_hsa_r1:MEME#1#Intergenic"
                                SIX5_disc2
"SIX5_H1-hESC_encode-Myers_seq_hsa_r1:MDscan#1#Intergenic"
                                NFY_disc1
"NFYA_K562_encode-Snyder_seq_hsa_r1:MEME#2#Intergenic"
```

Users can also provide a list of PWMs as the motif library via the 'LoadMotifLibrary' function. In this function, 'tag' specifies the string that marks the start of each block of PWM; 'skiprows' is the number of description lines before the PWM; 'skipcols' is the number of columns to be skipped in the PWM matrix; 'transpose' is TRUE if the PWM has 4 rows representing A, C, G, T or FALSE if otherwise; 'field' is the position of the motif name within the description line; 'sep' is a vector of separators in the PWM; 'pseudocount' is the number added to the raw matrices, recommended to be 1 if the matrices are in fact position frequency matrices. These arguments provide the flexibility of loading a number of varying formatted files. The PWMs are returned as a list object. This function flexibly adapts to a variety of different formats. Some examples using online accessible files from other research groups are shown below.

```
> pwms <- LoadMotifLibrary(
+ "http://meme.nbcr.net/meme/examples/sample-dna-motif.meme-io")
> pwms <- LoadMotifLibrary(
+ "http://compbio.mit.edu/encode-motifs/motifs.txt",
+ tag = ">", transpose = FALSE, field = 1,
+ sep = c("\t", " ", ">"), skipcols = 1,
+ skiprows = 1, pseudocount = 0)
> pwms <- LoadMotifLibrary(
+ "http://johnsonlab.ucsf.edu/mochi_files/JASPAR_motifs_H_sapiens.txt",
+ tag = "/NAME", skiprows = 1, skipcols = 0, transpose = FALSE,
+ field = 2)
> pwms <- LoadMotifLibrary(
+ "http://jaspar.genereg.net/html/DOWNLOAD/ARCHIVE/JASPAR2010/all_data/matrix_only/matrix.txt",
+ tag = ">", skiprows = 1, skipcols = 1, transpose = TRUE,
+ field = 1, sep = c("\t", " ", "\\[", "\\]", ">"),
+ pseudocount = 1)
> pwms <- LoadMotifLibrary(
+ "http://jaspar.genereg.net/html/DOWNLOAD/JASPAR_CORE/pfm/nonredundant/pfm_vertibrates.txt",
+ tag = ">", skiprows = 1, skipcols = 0, transpose = TRUE, field = 1,
+ sep = c(">", "\t", " "), pseudocount = 1)
> pwms <- LoadMotifLibrary(
+ "http://gibbs.biomed.ucf.edu/PreDREM/download/nonredundantmotif.transfac",
+ tag = "DE", skiprows = 1, skipcols = 1,
+ transpose = FALSE, field = 2, sep = "\t")
```

The data set for the SNP information must be a table including five columns:

- chr: the chromosome ID;
- snp: the genome coordinate of the SNP;
- snpid: the string for the SNP name;
- a1, a2: nucleotides for the two alleles at the SNP position.

This data set can be loaded using the 'LoadSNPData' function. The 'genome.lib' argument specifies the annotation package name corresponding to the SNP data set, with the default as 'BSgenome.Hsapiens.UCSC.hg19'. Each side of the SNP is extended by a number of base pairs specified by the 'half.window.size' argument. 'LoadSNPData' extracts the genome sequence within such windows around each SNP using the 'genome.lib' package. An example is the following:

If 'default.par = FALSE', 'LoadSNPData' simultaneously estimates the parameters for the first order Markov model in the reference genome using the nucleotides within the SNP windows. Otherwise, it loads a set of parameter values pre-fitted from sequences around all the SNPs in the NHGRI GWAS catalog ([Hindorff et al.,]). We recommend setting 'default.par = TRUE' when we have fewer than 1000 SNPs. 'LoadSNPData' returns a list object with five fields:

- \$sequence_matrix: a matrix with $(2 \times \text{half.window.size} + 1)$, with each column corresponding to one SNP. The entries 1-4 represent the A, C, G, T nucleotides;
- \$ref_base: a vector coding the reference allele nucleotides for all SNPs;
- \$snp_base: a vector coding the SNP allele nucleotides for all SNPs;
- \$prior: the stationary distribution parameters for the Markov model;
- \$transition: the transition matrix for the first order Markov model.

A toy sample data set including a preloaded motif library and a SNP set is included in the package:

```
> data(example)
> names(motif_library)
[1] "SIX5_disc1" "MYC_disc1" "SRF_disc1" "AP1_disc1" "SIX5_disc2"
[6] "ALX3_2"     "AFP_1"
> str(snpInfo)
List of 5
 $ sequence_matrix: int [1:61, 1:1997] 4 3 1 4 3 2 2 1 3 3 ...
 ..- attr(*, "dimnames")=List of 2
 .. ..$ : NULL
 .. ..$ : chr [1:1997] "rs10910078" "rs4486391" "rs3748816" "rs2843401" ...
 $ ref_base      : int [1:1997] 4 1 1 4 4 4 4 1 1 4 ...
 $ snp_base      : int [1:1997] 2 4 3 2 2 2 2 2 3 2 ...
 $ transition    : num [1:4, 1:4] 0.317 0.354 0.292 0.214 0.177 ...
 ..- attr(*, "dimnames")=List of 2
 .. ..$ : chr [1:4] "A" "C" "G" "T"
 .. ..$ : chr [1:4] "A" "C" "G" "T"
 $ prior        : Named num [1:4] 0.289 0.207 0.207 0.297
 ..- attr(*, "names")= chr [1:4] "A" "C" "G" "T"
> ## to look at the motif information
> data(encode_motif)
> motif_info[names(motif_library)]
                                SIX5_disc1
"SIX5_GM12878_encode-Myers_seq_hsa_r1:MEME#1#Intergenic"
                                MYC_disc1
"USF2_K562_encode-Snyder_seq_hsa_r1:MDscan#1#Intergenic"
                                SRF_disc1
"SRF_H1-hESC_encode-Myers_seq_hsa_r1:MDscan#2#Intergenic"
                                AP1_disc1
"JUND_K562_encode-Snyder_seq_hsa_r1:MEME#1#Intergenic"
                                SIX5_disc2
"SIX5_H1-hESC_encode-Myers_seq_hsa_r1:MDscan#1#Intergenic"
                                ALX3_2
                                "ALX3_jolma_DBD_M449"
                                AFP_1
                                "AFP1_transfac_M00616"
```

3.2 Affinity score tests

The binding affinity scores for all pairs of SNP and PWM can be computed by the 'ComputeMotifScore' function. It returns a list of two fields: 'snp.tbl' is a data.table containing the nucleotide sequences for each SNP; 'motif.scores' is a data.table containing the binding affinity scores for each SNP-motif pair.

```
> motif_score <- ComputeMotifScore(motif_library, snpInfo, ncores = 2)
> motif_score$snp.tbl
```

	snpid	ref_seq
1:	rs10910078	TGATGCCAGGTGGTCAGTGGGTTTTTGCCATCCGCCAGGAGCTTCACTGGGCCTCCCGTTG
2:	rs4486391	ATGGAGAATTCCACAGCTGATTGGAACCTAAACGAGAGAACCAAATGGACATCCCAGGGCT
3:	rs3748816	TTGGAGTACTCCTCGTCCAGGCGCCTGTTTCATCTCCTCCAGGATGTAGTCAGGGTGCCCGA
4:	rs2843401	TCCTCCACCATTGTGCCAAACAGCGCCTGGTGGGGCCACCCGATCATCCCACGGGCCCCCA
5:	rs2843402	CACCTTCTGGGCTGCAGGACTTCCTGCCCTTTAGGAAAGGGAGGCAGCCCTTCTTCTCTCC

1993:	rs3003207	CACCAAAACAGCTATGTCAATTTAGAAATGCAAATTTGGACCCTAGAGTTTGATTATCAGCA
1994:	rs1173830	ATATAATCTATTCTTTCTTTCCCTTTCTTCTCCAGAAACAGCTCAAATTATGAAATAACT
1995:	rs654873	GCTTCAGTAAATTAACATTGATGAGTACCCTTATGGAATCTATCATGCATGTTCCAATTT
1996:	rs1768071	GGTGTGAGGGAATGCCAATCTCGCTGGTTCCGGTTTATTAGCTCGGGGAATCTCGTTAGAT
1997:	rs1538961	CTACCAGGTGCCAGTTGAGGATAGTCTAAACTACATTCTCAGCTTGAAATATTTTGGACCA

```

                                snp_seq
1: TGATGCCAGGTGGTCAGTGGGTTTTTGCCACCCGCCAGGAGCTTCACTGGGCCTCCCGTTG
2: ATGGAGAATTCCACAGCTGATTGGAACCTATACGAGAGAACCAAATGGACATCCCAGGGCT
3: TTGGAGTACTCCTCGTCCAGGCGCCTGTTCTGCTCTCCTCCAGGATGTAGTCAGGGTGCCCGA
4: TCCTCCACCATTGTGCCAAACAGCGCCTGGCGGGGCCACCCGATCATCCCACGGGCCCCCA
5: CACCTTCTGGGCTGCAGGACTTCCTGCCCTCTAGGAAAGGGAGGCAGCCCTTCTTCTCTCC
---
```

	snpid	ref_seq
1993:	rs3003207	CACCAAAACAGCTATGTCAATTTAGAAATGCAAATTTGGACCCTAGAGTTTGATTATCAGCA
1994:	rs1173830	ATATAATCTATTCTTTCTTTCCCTTTCTTCTCCAGAAACAGCTCAAATTATGAAATAACT
1995:	rs654873	GCTTCAGTAAATTAACATTGATGAGTACCCTTATGGAATCTATCATGCATGTTCCAATTT
1996:	rs1768071	GGTGTGAGGGAATGCCAATCTCGCTGGTTCCGGTTTATTAGCTCGGGGAATCTCGTTAGAT
1997:	rs1538961	CTACCAGGTGCCAGTTGAGGATAGTCTAAACTACATTCTCAGCTTGAAATATTTTGGACCA

```

                                ref_seq_rev
1: CAACGGGAGGCCAGTGAAGCTCCTGGCGGATGGCAAAAACCCACTGACCACCTGGCATCA
2: AGCCCTGGGATGTCCATTTGGTTCTCTCGTTTAGGTTCCAATCAGCTGTGGAATTCTCCAT
3: TCGGGCACCCCTGACTACATCCTGGAGGAGATGAACAGGCGCCTGGACGAGGAGTACTCCAA
4: TGGGGGCCCGTGGGATGATCGGGTGGCCCCACCAGGCGCTGTTTGGCACAATGGTGGAGGA
5: GGAGGAAGAAAGGGCTGCCTCCCTTTCTTAAAGGGCAGGAAGTCTGCAGCCCAGAAGGTG
---
```

	snpid	ref_seq
1993:	rs3003207	TGCTGATAATCAAACCTAGGGTCCAATTTGCATTTCTAAAATGACATAGCTGTTTTGGTG
1994:	rs1173830	AGTTATTTTCATAATTTGAGCTGTTTCTGGAGAAGGAAAGGGAAAGAAAGAATAGATTATAT
1995:	rs654873	AAATTGGAACATGCATGATAGATTCCATAAGGGTACTCATCAATGTTAATTTTACTGAAGC
1996:	rs1768071	ATCTAACGAGATTCCCCGAGCTAATAAACCGGAACCAGCGAGATTGGCATTCCCTGACACC
1997:	rs1538961	TGGTCAAAAATATTTCAAGCTGAGAATGTAGTTTAGACTATCCTCAACTGGCACCTGGTAG

```

                                snp_seq_rev
1: CAACGGGAGGCCAGTGAAGCTCCTGGCGGATGGCAAAAACCCACTGACCACCTGGCATCA
2: AGCCCTGGGATGTCCATTTGGTTCTCTCGTATAGGTTCCAATCAGCTGTGGAATTCTCCAT
3: TCGGGCACCCCTGACTACATCCTGGAGGAGACGAACAGGCGCCTGGACGAGGAGTACTCCAA
4: TGGGGGCCCGTGGGATGATCGGGTGGCCCCGCCAGGCGCTGTTTGGCACAATGGTGGAGGA
5: GGAGGAAGAAAGGGCTGCCTCCCTTTCTTAAAGGGCAGGAAGTCTGCAGCCCAGAAGGTG
---
```

	snpid	motif	log_lik_ref	log_lik_snp	log_lik_ratio
1:	rs10910078	AFP_1	-117.69581	-116.93812	-0.7576857
2:	rs4486391	AFP_1	-73.54122	-95.18078	21.6395566

3:	rs3748816	AFP_1	-73.94669	-96.74940	22.8027074
4:	rs2843401	AFP_1	-139.33536	-141.41480	2.0794415
5:	rs2843402	AFP_1	-95.07542	-95.40392	0.3285041

13975:	rs3003207	SRF_disc1	-72.43032	-73.11450	0.6841775
13976:	rs1173830	SRF_disc1	-93.09039	-93.09039	0.0000000
13977:	rs654873	SRF_disc1	-93.09039	-70.39764	-22.6927498
13978:	rs1768071	SRF_disc1	-116.44934	-115.83145	-0.6178913
13979:	rs1538961	SRF_disc1	-96.14035	-73.11450	-23.0258509

The affinity scores for the reference and the SNP alleles are represented by the 'log_lik_ref' and 'log_lik_snp' columns in 'motif.scores'. The affinity score change is included in the 'log_lik_ratio' column. These three affinity scores are tested in the subsequent steps. 'motif.scores' also include other columns for the position of the best matching subsequence on each allele. For a complete description on all these columns, users can look up the help documentation.

After we have computed the binding affinity scores, they can be tested using the 'ComputePValues' function. The result is a data.table extending the affinity score table by six columns:

- 'pval_ref': p-value for the reference allele affinity score;
- 'pval_snp': p-value for the SNP allele affinity score;
- 'pval_cond_ref' and 'pval_cond_snp': conditional p-values for the affinity scores of the reference and SNP alleles;
- 'pval_diff': p-value for the affinity score change between the two alleles;
- 'pval_rank': p-value for the rank test between the two alleles.

We recommend using 'pval_ref', 'pval_snp' for assessing the significance of allele specific affinity, and using 'pval_rank' for assessing the significance of the SNP effect on the affinity change.

```
> motif.scores <- ComputePValues(motif.lib = motif_library,
+                               snp.info = snpInfo,
+                               motif.scores = motif_scores$motif.scores,
+                               ncores = 7)
> motif.scores[, list(snpid, motif, pval_ref, pval_snp, pval_rank, pval_diff)]
```

	snpid	motif	pval_ref	pval_snp	pval_rank	pval_diff
1:	rs10910078	AFP_1	0.9776495	0.96265483	0.87905728	0.77254685
2:	rs4486391	AFP_1	0.3767000	0.76858447	0.30101164	0.25680145
3:	rs3748816	AFP_1	0.4267265	0.90817363	0.28624944	0.08460265
4:	rs2843401	AFP_1	0.9911095	0.99822551	0.89524475	0.58550863
5:	rs2843402	AFP_1	0.7636040	0.80776942	0.82713921	0.86817355

13975:	rs3003207	SRF_disc1	0.2372302	0.24653492	0.76685914	0.66522050
13976:	rs1173830	SRF_disc1	0.2673992	0.26739919	1.00000000	1.00000000
13977:	rs654873	SRF_disc1	0.2618610	0.05817295	0.04413835	0.25026857
13978:	rs1768071	SRF_disc1	0.7770457	0.69299844	0.67095930	0.68555718
13979:	rs1538961	SRF_disc1	0.6604981	0.24653492	0.17069042	0.03966129

3.3 Additional analysis

atSNP provides additional functions to extract the matched nucleotide subsequences that match to the motifs. Function 'MatchSubsequence' adds the subsequence matches to the affinity score table by using the motif library and the SNP set. The subsequences matching to the motif in the two alleles are returned in the 'ref_match_seq' and 'snp_match_seq' columns. The 'IUPAC' column returns the IUPAC letters of the motifs. Notice that if you have a large number of SNPs and motifs, the returned table can be very large.

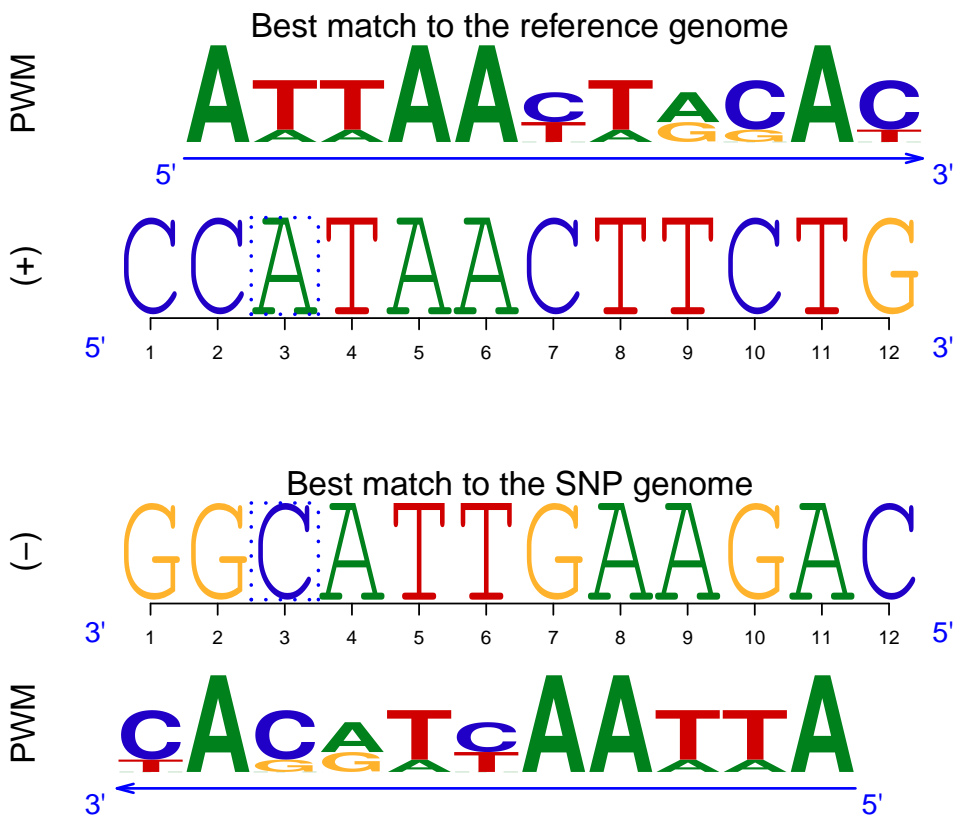
```
> match_result <- MatchSubsequence(snp.tbl = motif_scores$snp.tbl,
+                                 motif.scores = motif.scores,
+                                 motif.lib = motif_library,
+                                 snpids = c("rs10910078", "rs4486391"),
+                                 motifs = names(motif_library)[1:2],
+                                 ncores = 2)
> match_result[, list(snpid, motif, IUPAC, ref_match_seq, snp_match_seq)]
```

	snpid	motif	IUPAC	ref_match_seq	snp_match_seq
1:	rs10910078	MYC_disc1	GTCACGTGAC	GGCGGATGGC	GCCACCCGCC
2:	rs10910078	SIX5_disc1	GARWTGTAGT	CTGGCGGATG	GGCGGGTGGC
3:	rs4486391	MYC_disc1	GTCACGTGAC	TAAACGAGAG	CTCGTATAGG
4:	rs4486391	SIX5_disc1	GARWTGTAGT	CTCGTTTAGG	CTCGTATAGG

We can also visualize how each motif is matched to each allele using the 'plotMotifMatch' function:

```
> plotMotifMatch(snp.tbl = motif_scores$snp.tbl,
+               motif.scores = motif_scores$motif.scores,
+               snpid = motif_scores$snp.tbl$snpid[50],
+               motif.lib = motif_library,
+               motif = motif_scores$motif.scores$motif[1])
```

AFP_1 Motif Scan for rs301789



4 Session Information

R version 3.1.1 (2014-07-10)

Platform: x86_64-redhat-linux-gnu (64-bit)

locale:

[1] LC_CTYPE=zh_TW.UTF-8	LC_NUMERIC=C
[3] LC_TIME=en_US.UTF-8	LC_COLLATE=en_US.UTF-8
[5] LC_MONETARY=en_US.UTF-8	LC_MESSAGES=en_US.UTF-8
[7] LC_PAPER=en_US.UTF-8	LC_NAME=C
[9] LC_ADDRESS=C	LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8	LC_IDENTIFICATION=C

```

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

other attached packages:
[1] atSNP_1.0          motifStack_1.6.5    ade4_1.6-2          MotIV_1.18.0
[5] BiocGenerics_0.8.0 grImport_0.9-0      XML_3.98-1.1        testthat_0.9.1
[9] doMC_1.3.3         iterators_1.0.7     foreach_1.4.2       data.table_1.9.4
[13] Rcpp_0.11.3

loaded via a namespace (and not attached):
[1] Biostrings_2.30.1  BSgenome_1.30.0     chron_2.3-45
[4] codetools_0.2-8    compiler_3.1.1       GenomicRanges_1.14.4
[7] IRanges_1.20.7     lattice_0.20-29      plyr_1.8.1
[10] reshape2_1.4.1     rGADEM_2.10.0        seqLogo_1.28.0
[13] stats4_3.1.1       stringr_0.6.2        tools_3.1.1
[16] XVector_0.2.0

```

References

- [Chan et al., 2010] Chan, H. P., Zhang, N. R., and Chen, L. H. (2010). Importance sampling of word patterns in DNA and protein sequences. *Journal of Computational Biology*, 17(12):1697–1709.
- [Grant et al., 2011] Grant, C. E., Bailey, T. L., and Nobel, W. S. (2011). FIMO: Scanning for occurrences of a given motif. *Bioinformatics*, 7:1017.
- [Hindorff et al.,] Hindorff, L., J, J. M., Morales, J., Junkins, H., Hall, P., Klemm, A., and Manolio, T. A catalog of published genome-wide association studies.
- [Macintyre et al., 2010] Macintyre, G., Bailey, J., Haviv, I., and Kowalczyk, A. (2010). is-rSNP: a novel technique for in silico regulatory SNP detection. *Bioinformatics*, 26(18):524–530.