Manual, ver. 03/01/2008, for "Lever_1.1", and the associated programs "PhylCRM_preprocess_1.1" and "Lever_statistics_1.1" utilized in the paper:

Jason B. Warner[1,6], Anthony A. Philippakis[1,3,4,6], Savina A. Jaeger[1,6], Fangxue Sherry He[1,7], Jolinta Lin[1,5], and Martha L. Bulyk[1,2,3,4].

Systematic identification of mammalian regulatory motifs' target genes and functions.

Nature Methods (2008). Advance Online Publication, 2 Mar 2008, (10.1038/nmeth.1188).

[1]Division of Genetics, Department of Medicine; [2]Department of Pathology; [3]Harvard/MIT Division of Health Sciences and Technology (HST); Brigham and Women's Hospital and Harvard Medical School, Boston, MA 02115.

[4]Harvard University Graduate Biophysics Program, Harvard Medical School, Boston, MA 02115.

[5]Department of Biology, MIT, Cambridge, MA 02139.

[6]These authors contributed equally to this work.

[7]Current address: The Institute for Genomic Research, Rockville, MD 20850.

Please address any questions, comments, complaints, or other feedback to sjaeger@rics.bwh.harvard.edu and mlbulyk@receptor.med.harvard.edu.

I. Introduction

The suite of computational tools described in this manual provides an *in silico* framework for mapping transcription factor binding site (TFBS) motifs to their likely target genes. There are three programs in this suite:

1) PhylCRM_preprocess: Both PhylCRM and Lever require a collection of input files that parameterize various statistical properties of the utilized motifs. This program generates these parameterization files. Also, it has the ability to generate length-matched sequence sets for each of the foreground gene sets under consideration. These matched background gene sets can be used in Lever screens (see below).

2) Lever: This program takes as input a collection of TFBS motifs, a collection of sequences and a set system (i.e., a collection of subsets) for these sequences. For each subset of sequences and each motif or combination of motifs, the program evaluates whether or not the sequences in the subset (i.e., a "foreground" set of sequences) show more enrichment for the motifs under consideration than a matched "background" set of sequences. Here, enrichment is determined by comparing the PhylCRM scores of the foreground regions against the PhylCRM scores of the background regions; thus, Lever extends the capabilities of PhylCRM by screening pairings of sequence sets and motifs.

3) Lever_statistics: This program takes as input two matrices, where the first matrix indicates the membership of each sequence in the various foreground/background sets and the second matrix indicates the scores of the sequences for each of the motifs/motif combinations under consideration. It then uses these matrices to calculate which motifs/motif combinations are enriched within various motifs. Note that the computations performed by this program are also performed by Lever itself, but we have found it useful to have this as an additional, stand-alone component.

In this manual we describe how to run each of these programs, as well as the formatting of the various input and output files.

II. Compilation of programs and system requirements:

On the Bulyk Lab website (http://thebrain.bwh.harvard.edu/Lever/) we have provided pre-compiled versions of the four programs in the as well as the source code accompanied with Makefile and Readme file. The programs were compiled using what LINUX Kubuntu 32-bit processor.

If you would like to (re)compile these programs yourself, you will first need to download and install the *Gnu Scientific Library* (available from http://www.gnu.org/software/gsl/) and then follow the instructions in the Readme file.

III. PhylCRM_preprocess

As stated above, the purpose of PhylCRM preprocess is to 1) get various statistical parameters for each of the motifs under consideration, and (if the –F option described below is specified) to 2) generate length-matched background gene sets for a given input collection of foreground gene sets.

If you type "./phylcrm_preprocess_1.1" at the command line in the directory where the program is installed, you get the following output

Usage:

Required arguments:
[-T] [tree file]
[-O] [output directory]
[-S] [directory containing original sequences]
[-M] [motifs and PFM cutoffs file]

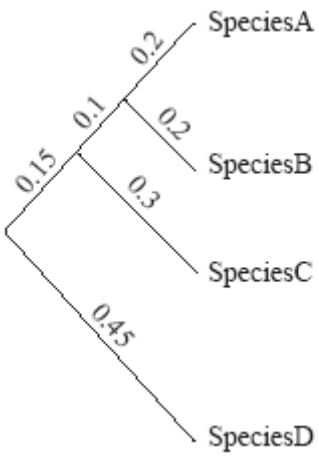
Optional arguements:
[-F] [foreground gene sets file]
[-U] [change lower to upper case|0]
[-P] [pseudo count|0.1]
[-E] [JC or HKY|HKY]
[-W] [max|500]
[-R] [inclusive or restrictive|0]
[-EP] [epsilon value for length matching|0.02]
[-DEOVERLAP]

Here, we provide a brief description of each of these input parameters. Note that the first four parameters (i.e., [-T], [-O], [-S] and [-M]) are required in order for PhylCRM_preprocess to run, while the others are optional. If the optional parameters are not set, then PhylCRM_preprocess will automatically default to the value shown in this list after the "|" value.

In this section, we first describe the function and sytax of each of these input parameters, and we then explain the various output files of PhylCRM_preprocess.

Input Parameters

1) [-T][tree file] This is the directory and file name of where the tree that will be utilized in this PhylCRM run is stored (e.g. "/home/trees/sample.txt"). We note that there are no restrictions on the naming of this tree. The input phylogenetic tree must be in *Newick format*, which makes use of nested parentheses. For example, the phylogenetic tree



is represented by the tree

(((SpeciesA:0.2,SpeciesB:0.2):0.1,SpeciesC:0.3):0.15,SpeciesD:0.45);

The following link from the web-page of Felsenstein contains more details on Newick-formatting

http://evolution.genetics.washington.edu/phylip/newicktree.html

Here, the names given to the species and written inside the parentheses can be any string, but these names must correspond to the names given to the files containing the sequences. Specifically, if one was using the phylogenetic tree shown above, then the sequence files would need to be named "sequence_SpeciesA.txt," "sequence_SpeciesB.txt," "sequence_SpeciesC.txt," and "sequence_SpeciesD.txt" (see the **"Sequence Formatting**" section for more details on the input sequence files).

Note that if you wanted to consider only one genome, then the input tree file should simply state the name of the base genome and not include any parentheses. For example, using the example above would give

SpeciesA

Note that for all programs in this suite that require an input phylogenetic tree, it is CRUCIAL *that the FIRST species listed in the input string be the base genome (i.e., the species within which it is desired to find cis regulatory modules (CRMs))*. Also, note that we will use the term "base genome" throughout this manual to denote the genome being searched for CRMs, and we will refer to all other genomes as "alignment genomes."

2) [-O][output directory] This parameter states the directory where all of the output files that are generated by PhylCRM_preprocess should go. A description of these output files is provided below.

3) [-S][directory containing original sequences] PhylCRM_preprocess (as well as PhylCRM and Lever) takes as input a collection of sequence files that should be in one-to-one correspondence with the species in the input phylogenetic tree. Each of these files

must begin with "sequence_", end in ".txt", and be exactly identical to the species list in the phylogenetic tree between these identifiers. Thus, as stated in the description of the –T parameter in 1), the input sequence files should be named "sequence_SpeciesA.txt," "sequence_SpeciesB.txt," "sequence_SpeciesC.txt," and "sequence_SpeciesD.txt". Each of these files should be located in the same directory, and this input parameter lets the program know where they are located.

The input sequence files should be in FASTA format, where the first line of each sequence record begins with a ">" followed by any desired identification about the sequence. The next line contains the sequence to be scanned for matches to the input motifs. For example, an input sequence file might look like

```
>Sequence_1         chr22 start=130001      stop=130101 strand=+
ACATACAGTTGTTCACTGATACATTACCATTAACCTTAAAATCTGTCATCTCTCTATGTTTTGGGTC
TCTGTACTATACCTTAGTACCTATAGGTATACT
>Sequence_2   chr3  start=15022 stop=15122  strand=-
CATTCATGATCTAGTTGGGGCGCGCGCatatcttgctgactgactgggtttagtgatgtctagtatt
tcccttagtatctagggctatcttggtctcGGG
```

. . .

There are a few important conventions to be aware of
Convention #1) We use lower-case letters in the *base genome* to indicate sequences that should be avoided when searching for conserved clusters of TF binding sites (for example exons or repetitive regions). If you want to include the lower-case sequences in scanning, then you can do this with the [-U] input parameter (see below).

Convention #2) There should be no breaks (i.e., newlines, tabs, spaces, etc) between sequence records.

Convention #3) As stated above, there should be one sequence file for each species listed in the input phylogenetic tree. The sequence records within these files need to be in one-to-one correspondence across all of the input sequence files. First, this means that the order of the sequence records needs to be the same across each of the input files. Second, the identification line (i.e., the line that begins with a ">") needs to be exactly identical across the sequence files for each corresponding sequence record. Finally, the lengths of corresponding sequences needs to be identical across the input files.

Convention #4) We have established a format for storing multiple alignments. In particular, there are two conventions to be aware of regarding the alignment genomes. First, if there is no orthologous sequence in an alignment genome, then a run of "r" letters is put at that locus. The following example shows what the sequence files might contain if there was no sequence in SpeciesB at an orthologous locus for the four example species.

```
SpeciesA: …GTCACGTG…
SpeciesB: …rrrrrrrr…
SpeciesC: …GTGACGTG…
SpeciesD: …GTGAGGTG…
```

Again, it is crucial that the lengths of all corresponding sequences in the base and alignment genomes be identical.

Second, let's assume that Species_A is the base genome (i.e., the genome being searched for CRMs), and let's assume that there was an insertion at this locus in Species_A and, say, Species_B.

```
SpeciesA: …TT---GCA…
SpeciesB: …TT---GCA…
SpeciesC: …ATGACGCC…
SpeciesD: …TTGTCGTA…
```

If you would like to remove such indels in the base genome so that motif matches can be observed that might involve, say, the sequence "TTGCA", then we have established a simple convention that allows conservation to be consistently evaluated. To denote that any such sequence cannot be a conserved motif match in Species_C and Species_D, we use lower case letters to denote the positions one to the left and one to the right of the observed indels. Thus, the example above becomes

```
SpeciesA: …TTGCA…
SpeciesB: …TTGCA…
SpeciesC: …AtgCC…
SpeciesD: …TtgTA…
```

Here, if PhylCRM_preprocess (or PhylCRM or Lever) observes a lower case letter at a position other than the first or last position of the motif in one of the alignment genomes, then it knows that this motif match in the alignment genome cannot be conserved.

Convention #5) It is important that the sequence file that is input to PhylCRM_preprocess be very large (more than 2 Mb of sequence per genome), as it is using these to estimate various aspects of the motif distributions. Later on, when you are running PhylCRM or Lever, if you want to consider a smaller set of sequences, that is fine—you can shrink it down.

4) [-M][Motifs and PFM cutoffs file] This file contains the collection of input motifs, as represented by position frequency matrices (PFMs). Here are three examples of what acceptable input matrices might look like:

```
Sample_motif1       log     10
24      17      0       39      0       3       0       0
2       4       39      0       5       13      0       1
12      13      0       0       34      14      0       38
1       5       0       0       0       9       39      0

Sample_motif2       sd      2
0       0       100     25      0       0       100
0       0       0       25      0       100     0
0       100     0       25      0       0       0
100     0       0       25      100     0       0
```

```
Sample_motif3
0       0       0       0       0       0
100     0       0       0       0       0
0       0       0       0       100     0
0       100     100     100     0       100
```

The first line of each is tab-separated, where the first element is a motif identifier that can be any string, the second element (if it exists) is either "sd" or "log", and the third element (if it exists) is a number. For each motif, PhylCRM_preprocess (as well as PhylCRM and Lever) needs to know a cutoff value to use in order to call a given locus a "match" to the given motif. Here, let *P(i,j)* be the frequency with which letter *i* occurs in column *j* of the position frequency matrix, and let *Q(i)* be the genomic frequency of this letter. In deciding whether or not a given locus should be considered a "match" to a motif at a given locus, PhylCRM first computes the standard likelihood ratio score (which we denote here as *S*)

$$S = \sum_{i \in \{A,C,G,T\}} \sum_{j=1}^{w} \log_2 \left(\frac{P(i,j)}{Q(i)} \right)$$

If the second entry in the tab-separated list is "log," then the next number indicates a cutoff value for *S* (for example, any locus where *S*>10 would be a match to Sample_motif1 in the example above).

Alternatively, if the second entry of the tab-separated list is "sd," then PhylCRM_preprocess computes the expected value and variance of *S* under the assumption that the locus is a motif. These are given by the values

$$\mu = \sum_{j=1}^{w} \sum_{i \in \{A,C,G,T\}} P(i,j) \log_2 \left(\frac{P(i,j)}{Q(i)} \right)$$

$$\sigma^2 = \sum_{j=1}^{w} \left(\left(\sum_{i \in \{A,C,G,T\}} P(i,j) \left(\log_2 \left(\frac{P(i,j)}{Q(i)} \right) \right)^2 \right) - \left(\sum_{i \in \{A,C,G,T\}} P(i,j) \log_2 \left(\frac{P(i,j)}{Q(i)} \right) \right)^2 \right)$$

In this case, PhylCRM_preprocess will consider the locus a match to the motif if $S > (\mu - \alpha * \sigma)$ where α is the third value in the tab-separated list ("2" in Sample_motif2).

Finally, if no values are specified in the second and/or third elements of the tab-separated list, or if the second value is not "log" or "sd" then PhylCRM_preprocess will use a **default setting of "sd" and "2"** (i.e., the same as what is shown in Sample_motif2).

5) [-F] [foreground gene sets file] In performing a Lever screen, one must input a set-system that is comprised of (possibly many) subsets of a given collection of foreground genes. For example, the initial foreground collection of genes might be the set of all genes that are up-regulated in some cell type, and the subsets might be the intersections of these genes with genes in various Gene Ontology categories that are over-represented among the set of all up-regulated genes (as was done in the manuscript of *Warner et al* where the PhylCRM and Lever tools were initially described). For each of these gene subsets, it is also necessary that a background gene set be specified, so that the degree of

enrichment for each motif/motif combination in the foreground subset, as compared to the background subset, can be determined. In cases where the considered sequences are of variable lengths, it is often the case that foreground subsets will on average have longer/shorter lengths than expected by chance. Therefore, it is necessary that the background set of genes be length-matched with respect to the foreground gene set.

This parameter of the program indicates where the set system on the foreground set of genes is located. If it is specified, then PhylCRM will automatically build a length-matched background set of genes for each input foreground set, and it will output a new file that contains the information on the background set corresponding to each foreground subset (see below for a description of this output file). The format of the input collection of foreground subsets is as follows.

```
SET  sample_geneset_1
>sequence1    chr=1   start=54312    stop=65371    strand=+
>sequence7    chr=13 start=10001    stop=17391    strand=-
>sequence3    chr=21 start=12020    stop=15533    strand=+
…
SET sample_geneset_2
>sequence14   chr=5   start=79179    stop=83123    strand=-
>sequence10   chr=21 start=84843    stop=88943    strand=-
>sequence6    chr=2   start=102      stop=1953     strand=+
…
```

Here, the use of the word "SET" tells PhylCRM_preprocess that this is the start of a new foreground gene set. The letters that follow the word "SET" on the same line can be any string that you want in order to identify the set, but the first three characters of that line should be "SET". The other lines tell PhylCRM_preprocess what sequences are in that particular set. These lines must correspond EXACTLY to the identification lines in the FASTA-formatted input sequence file given in parameter [-S] above—it is very important that these agree exactly.

It should be noted that PhylCRM_preprocess will automatically remove any small gene sets (less than 10 genes). Also, if you are running PhylCRM_preprocess with this option, it is important that you have a large number of candidate background genes in your original input sequence files, where "candidate background" means sequences that do not appear in any foreground set. The basic way that length-matching works is that one begins with a large number of candidate background genes and, for each foreground set, PhylCRM_preprocess chooses some subset of these candidate background genes as a true background set. In order to ensure that the length-matching is done accurately, it is important that the initial list of candidates be as large as possible.

6) [-U] [change lower to upper case|0] As stated in the description of the [-S] parameter above, PhylCRM_preprocess (as well as PhylCRM and Lever) ignores lower case letters in the input sequence. If you want to include these lower-case letter, then set this

parameter to "1". Otherwise keep it as "0" (this is the default that will be used if no argument is passed).

7) [-P] [pseudo-count|0.1] Values of "0" in the input PFM cause problems in computing whether or not a given locus is a match to a PFM. A common practice to avoid this is to include a pseudo-count to each element in the PFM, and this parameter specifies what size of pseudo-count to use. If no value is specified, then a default value of 0.1 is used.

8) [-E] [JC or HKY|HKY] In calculating the Halpern-Bruno score, a model of nucleotide change under the neutral model (i.e., no selective pressure) is needed. Two common models are the "Jukes-Canotor" (JC) and the "Hasegawa-Kishino-Yano" (HKY) models. The HKY model models the nucleotide frequencies as well as the transition vs. transversion probabilities, while the JC model does not. If input, this parameter must be either "JC" or "HKY". If no value is passed, then the program automatically uses HKY.

9) [-W][max window size|500] This value specifies the maximum window size to use in scoring windows of sequence as candidate CRMs in PhylCRM preprocess. The input value must be an integer. If no value is passed, then a default of 500 is used.

10) [-R][inclusive or restrctive|0] As stated in the Supplementary Methods of the manuscript of *Warner et al*, PhylCRM_preprocess, PhylCRM, and Lever have two modes of evaluating conservation—where observing a lack of conservation decreases the likelihood that a given motif match is functional (restrictive), and one where it does not decrease this likelihood (inclusive). Passing a value of "1" in this parameter uses the restrictive option, and a value of "0" uses the inclusive option ("0" is the default setting if no value is passed).

11) [-EP][epsilon value for length matching| 0.02] This value only applies if one is using PhylCRM_preprocess to generate a length-matched background sequence set (*i.e.,* if you are using the "–F" parameter), and it specifies the amount of length bias to tolerate in selecting the length-matched background. In order to generate a length-matched background, PhylCRM_preprocess works to ensure that the area under the receiver-operating curve (AUC) does not deviate much from the expected value of 0.5 under the null hypothesis of no distributional difference in lengths when comparing the foreground and background regions. Let α be the input value given at this parameter, and let θ be the observed AUC between foreground and background regions when ranking them by lengths. Then PhylCRM_preprocess will ensure that

$$|\theta - 0.5| < \alpha$$

If no value is specified at this parameter, then a default value of 0.02 is used.

13) [-DEOVERLAP] As described in the Supplementary Methods of *Warner et al*, the PhylCRM scoring scheme has the option of removing overlapping positions between distinct motifs in order to avoid double-counting. This option specifies whether or not to perform this de-overlapping step in PhylCRM_preprocess. If "-DEOVERLAP" is typed at the command line, then de-overlapping will be performed; otherwise it will not be performed.

Note that the output motif statistics (see below) might be slightly different depending on whether or not this de-overlapping step is performed. Therefore, whichever way it is desired to run PhylCRM (i.e., with or without this overlapping step), PhylCRM_preprocess should be run in the same way. Also, note that (because of computational considerations) the Lever program does not perform de-overlapping. Therefore if you are running PhylCRM_preprocess in order to generate motif statistics files to later use with Lever, then you SHOULD NOT use the –DEOVERLAP option.

Output files from PhylCRM_preprocess

There are (up to) four basic types of output files from PhylCRM_preprocess. All of them will be located in the directory that you specified as output in the [-O] parameter described previously in this section.

1) Motif statistics files. These files will all be named "motif_parametersX_Y.txt" where the number X corresponds to the number of the motif in the input file [-M], and it takes values in the range 1, 2, 3, …. Thus, there will be one of these files for each motif in the input motifs file. Similarly, Y will be the cutoff value used for declaring a motif match ON THE LOG SCALE. Thus, a typical name for this output file might be something like "motif_parameters3_6.29.txt". If you open up this file, it should look like

```
500
0.000000
0.000000
0.974966    0.974966    0.512983    0.138900
0.972006    0.972006    0.920219    0.517723
0.969057    0.969057    1.252221    1.094035
0.966118    0.966118    1.528653    1.804756
0.963188    0.963188    1.762757    2.590461
…           …           …           …
```

where there are three numbers that are each on their own line, followed by four columns of numbers. The first number (here 500) indicates the maximum window size used in PhylCRM_preprocess. The second two numbers (both 0.000000 here) indicate two statistical parameters that will be used by the PhylCRM scoring scheme. The subsequent four columns of numbers contain information on the parameterizations of the window scores for each possible window size up to the maximum window size (thus, in this example there should be 500 rows for each of these four columns).

Together, all of these numbers will be used by PhylCRM and/or Lever to construct distributions of window scores under the null hypothesis of no enrichment. Note that when using these files in PhylCRM or Lever, the max window size can be less than the value input to PhylCRM_preprocess. Thus, if a max window size of 500 was used in PhylCRM_preprocess, then a max window size of 300 (but not 600) could be used by PhylCRM or Lever.

2) A motif locations file. This is a single file named "motif_locations.txt" that specifies the names and directories of all of the motif files generated by PhylCRM_preprocess. It might look something like

/home/myOutput/motif_parameters1_6.33.txt
/home/myOutput/motif_parameters2_12.01.txt
/home/myOutput/motif_parameters3_8.42.txt
/home/myOutput/motif_parameters4_3.75.txt

This file will be passed to PhylCRM and/or Lever (see below). Note that it is possible to edit this file if you want to change the set of motifs considered. For example, let's say you run PhylCRM_preprocess on four motifs and generate the above motif_locations file, but in a later run you want to ignore motif 3. Then you could simply use a file that contained

/home/myOutput/motif_parameters1_6.33.txt
/home/myOutput/motif_parameters2_12.01.txt
/home/myOutput/motif_parameters4_3.75.txt

Note that if you start making your own motif_locations.txt based on motif parameters files from old PhylCRM_preprocess runs, that's fine. The motif parameters files don't need to be in the same directory as your newly-created motif_locations file (although you do need to correctly specify the directories and file names in the motif_locations file).

3) Sequence files. If you run PhylCRM_preprocess with the –F parameter, it will generate a background gene set for each input foreground gene set. The original sequence files passed to PhylCRM_preprocess with the –S parameter contain many sequences, and not necessarily all of these sequences will end up in some foreground or background set. Therefore, PhylCRM_preprocess generates a new set of sequence files (also named, for example, "sequence_SpeciesA.txt," "sequence_SpeciesB.txt," "sequence_SpeciesC.txt," sequence_SpeciesD.txt") where ANY UNUSED SEQUENCES FROM THE ORIGINAL SEQUENCE FILES HAVE BEEN REMOVED. Note that these files will be written to whatever directory you specified as the output directory with the "-O" argument.

Note that IT IS VERY IMPORTANT THAT YOU USE THESE SEQUENCE FILES, INSTEAD OF THE ORIGINAL SEQUENCE FILES, AS INPUT TO PHYLCRM OR LEVER. PhylCRM_preprocess outputs a set systems file that contains information on which sequences are in each foreground/background set (see the next item below), and the indexes used in that file refer to these newly created sequence files rather than the original ones. Thus, you need to make sure that you are using these new sequence files in all subsequent steps.

4) A set systems file. If you run PhylCRM_preprocess with the –F parameter, then this file will appear in the output directory that contains information on which sequences are in each of the foreground and corresponding background sets. This file looks like

```
SET  sample_geneset_1
73
        14
        53
        72
        81
        96
        233
        234
        …
SET sample_geneset_2
57
        7
        19
        53
        72
        99
        133
        187
        212
        …
```

Each instance of the word SET indicates a new foreground gene set and its corresponding background gene set. The number in the line directly below the line containing the word SET indicates the number of *foreground* genes in the set. Thus, in this example, sample_geneset_1 has 73 foreground genes and sample_geneset_2 has 57 foreground genes. Next, the tab-indented numbers that follow in each line indicate the 0-based indexes of the genes in the sequence files that are part of one of these foreground or background gene sets, where the foreground genes are listed first, followed by the background genes. Thus, in sample_geneset_1 the 7th, 19th, 53rd, etc genes in the corresponding "sequence_..." files comprise the foreground regions for that set, and all numbers after the first 73 indicate the indexes of the background genes that should be used.

5) A counts file. This file is named "counts1" and contains the mononucleotide frequencies of all of the utilized nucleotides. The output looks like

```
A       0.2582393484
C       0.2442982456
G       0.2476503759
T       0.2498120301
```

IV. Lever

The purpose of the Lever program is to perform an *in silico* screen that systematically evaluates the degree of enrichment for various motifs/motif combinations within various input foreground gene sets.

If you type "./lever_1.1" at the command line in the directory where the program is installed, you get the following output

```
Usage:
    Required arguments:
    [-T] [tree file]
    [-O] [output directory]
    [-N] [output file name]
    [-S] [directory sequences]
    [-M] [motifs and PFM cutoffs file]
    [-BG] [file giving locations of gamma parameters]
    [-ID] [set systems file]

    Optional arguments:
    [-OR] [0 or 1|1]
    [-AND] [0 or 1|0]
    [-WOR] [0 or 1|0]
    [-WAND] [0 or 1|0]
    [-COMP] [0 or 1|0]
    [-CO] [number of motifs in a combination]
    [-U] [change lower to upper case|0]
    [-P] [pseudo count|0.1]
    [-E] [JC or HKY|HKY]
    [-W] [max|500][min|100]
    [-R] [inclusive or restrctive|0]
    [-MH] [FWER or FDR|FDR]
    [-nP] [number of permutations|1000]
    [-matrixoutput]
    [-statistics]
    [-LP] [length penalization]
```

Here, we explain each of these input parameters, and the syntax for using them. We then explain the outputs of Lever.

Input parameters

1) [-T][tree file] This is exactly the same as the [-T] file described in PhylCRM_preprocess above. Note that you should make sure to use the same phylogenetic tree as what was used by PhylCRM_preprocess to make the corresponding motif parameterization files.

2) [-O][output directory] As with the –O command described for PhylCRM_preprocess above, this parameter specifies where the output should go.

3) [-N][output file name] This parameter specifies what the name of the output file for Lever should be. Let's say you specify this parameter to be "-N lever_output.txt." Then Lever will generate an output file named "lever_output.txt."

4) [-S][sequence directory] As with the –S parameter for PhylCRM_preprocess above, this parameter describes where the input sequence files reside. Again, this directory should contain one sequence file for each genome being considered.

5) [-M][motifs and PFMs cutoffs file] As with the –M file described above for PhylCRM_preprocess, this file will contain all of the input motifs as well as information on what cutoff to use in considering a motif a match.

6) [-BG][file giving locations of motif parameters] As with the [-BG] parameter described above for PhylCRM, this file corresponds contains the locations (i.e., directories and file names) of all of the motif parameters files used in the current Lever run. This will be the motif_locations.txt file described in the section above on the outputs of PhylCRM_preprocess, or a modified version of it. Note that, due to computational constraints, Lever does not consider the "deoverlapping" step described in the section on PhylCRM_preprocess. Thus, in using PhylCRM_preprocess to generate the motif parameters for a Lever screen, it is important that you *do not* use the "-DEOVERLAP" parameter.

7) [-ID] [set systems file] This is an input file that contains information on all of the foreground gene sets (and their corresponding background gene sets) to use in the current Lever screen. The format of this file should be exactly the same as what was described above for the set systems output of PhylCRM_preprocess (i.e., output 4) described in the section on PhylCRM_preprocess outputs).

8) [-OR] [0 or 1|1]
[-AND] [0 or 1|0]
[-WOR] [0 or 1|0]
[-WAND] [0 or 1|0]
[-COMP] [0 or 1|0]
These parameters indicate which Boolean combinations of the motifs to consider in the current Lever screen. The terms "OR," "AND," "WOR," "WAND," and "COMP" refer to the Boolean and weighted Boolean motif combinations described for input parameter 7) in the description of PhylCRM. Note, however, that here the Boolean combination choices are not mutually exclusive, as they were in PhylCRM. Thus, if you type "-OR 1 –AND 1 –WOR 1 –WAND 1 –COMP 1", you will consider OR, AND, weighted OR, weighted AND, and Compound Boolean combinations of the input motifs (a full list of compound Boolean Combinations considered by PhylCRM/Lever is given in the Supplementary Methods of *Warner et al*). Similarly, "-OR 1 –AND 1 –WOR 0 –WAND 0 –COMP 0" would consider only OR and AND combinations. Note that if no values are specified, then only the OR combinations of the input motifs will be considered.

9) [-CO] [number of motifs in a combination] Lever can take large numbers of motifs (i.e. >100) as input. Practically, it might not make sense to consider all combinations of the input motifs (for example considering all combinations of 100 motifs would involve 2^{100} combination). This parameter is used to specify the maximum number of motifs that can be in a combination. For example if you use "-CO 2," and you have 100 input motifs, then you will screen with all 100 motifs individually, plus all 4950 (=100*99/2) pair-wise combinations.

Note that, because of computational considerations no more than 3-way combinations of motifs can be considered with the WOR and WAND combinations, and no more than 4-way combinations can be used with the COMP parameter. Thus, if you use "–CO 2" only weighted and compound combinations involving 2 motifs will be considered. Similarly, if you use "-CO 5" then only weighted combinations involving 3 or fewer motifs and Compound combinations involving 4 or fewer motifs will be considered (although OR and AND combinations involving up to 5 motifs will still be considered).

10) [-U] [change lower to upper case|0] This is exactly the same as the –U command for PhylCRM_preprocess and PhylCRM. It specifies whether or not to change the lower case letters to upper case.

11) [-P] [pseudo count|0.1] This is exactly the same as the –P parameter described above for PhylCRM_preprocess and PhylCRM. Again, it is important that the same value be used here as what was used in PhylCRM_preprocess in order to generate the motif parameterizations.

10) [-E] [JC or HKY|HKY] This is exactly the same as the –E parameter described above for PhylCRM_preprocess and PhylCRM. Again, it is important that the same value be used here as what was used in PhylCRM_preprocess in order to generate the motif parameterizations.

11) [-W] [max|500][min|100] This is exactly the same as the –W parameter described above for PhylCRM.

12) [-R][inclusive or restrctive|0] This is exactly the same as the –R parameter described above for PhylCRM_preprocess and PhylCRM. Again, it is important that the same value be used here as what was used in PhylCRM_preprocess in order to generate the motif parameterizations.

13) [-MH] [FWER or FDR|FDR] As stated above, Lever screens many gene sets for enrichment for many different motifs. A crucial aspect of this screening procedure is correcting for the many hypotheses being tested. Two common means for multiple hypothesis testing are "family-wise error rate (FWER)" and "false discovery rate (FDR)". This parameter specifies which of these two options to use, and the input syntax should be either "-MH FWER" or "-MH FDR". If no input value is specified, then "FDR" is used.

14) [-nP] [number of permutations|1000] In order to calculate statistical significance while controlling for the many (non-independent) hypotheses being tested, Lever performs a permutation test. This parameter specifies how many permutations to perform in order to calculate significance. If no value is specified, then a default of "1000" is used.

15) [-THRESHOLD] This is exactly the same as the –THRESHOLD parameter described above for PhylCRM_preprocess and PhylCRM.

16) [-matrixoutput] The Lever algorithm generates a pair of matrices which are referred to as "*Y*" and "*X*" in the Methods section of the main text of *Warner et al*. The rows of *Y* correspond to the input sequences (both foreground and background) and the columns correspond to the gene sets under consideration. Each element of the *Y* matrix specifies whether the corresponding gene is a member of the foreground or background set for the corresponding gene set. Similarly, the rows of *X* correspond to the input sequences (as for the *Y* matrix), and the columns correspond to motifs/combinations of motifs. Each element of the *X* matrix gives the score of the maximum-scoring window of sequence when scanning the corresponding gene with the corresponding combination of motifs with PhylCRM.

Once these *Y* and *X* matrices have been generated, Lever uses them in a permutation test in order to calculate the enrichment of each motif/motif combination in each gene set of interest. If one types "-matrixoutput" then these matrices will be output to the disk, otherwise they will not (see below for a description of what the output matrices will look like). If these matrices are output, then they can be used in subsequent Lever_statistics (see below) runs.

17) [-statistics] This parameter indicates whether or not to perform the permutation test within Lever, or to stop after the generation of the *Y* and *X* matrices. If "-statistics" is not typed at the command line, then Lever will not perform the permutation test (if you use this option, then you should make sure to also type "-matrixoutput", or else there will be no output at all). If you type "-statistics" then the permutation test will be performed using the method of multiple hypothesis correction specified by the "-MH" parameter and the number of permutations specified by the "-nP" parameter.

18) [-LP] [length penalization] In the case where the input sequences are of variable lengths, there can be substantial correlation between the score of the maximum scoring window of sequence and the overall length of the sequence. If one uses this option by typing "-LP" at the command line, then a regression of scores against sequence lengths is performed. Thus, this option helps to remove some of the dependence of the scores on the lengths of the sequence, which we have observed to improve the observed signals.

Output files of Lever

Lever outputs three basic types of files.

1) Output statistics from the permutation test. Let's say you use "–N lever_output.txt", then a file named "lever_output.txt" will be created in your output directory. This will be

non-empty only if you run Lever with the "-statistics 1" option, and it contains a summary of the results from the permutation test.

OR COMBINATIONS OUTPUT:

SET=0	NUM TFs=1	COMBO INDEX=(0)	pval=1.00	stat=0.40	mean=0.49	sd=0.04	error=(0.35,0.45)
SET=0	NUM TFs=1	COMBO INDEX=(1)	pval=0.11	stat=0.58	mean=0.50	sd=0.05	error=(0.53,0.63)
SET=0	NUM TFs=1	COMBO INDEX=(2)	pval=0.12	stat=0.57	mean=0.49	sd=0.04	error=(0.52,0.65)
SET=0	NUM TFs=1	COMBO INDEX=(3)	pval=0.21	stat=0.55	mean=0.49	sd=0.04	error=(0.49,0.68)
SET=0	NUM TFs=2	COMBO INDEX=(0,1)	pval=0.34	stat=0.52	mean=0.49	sd=0.04	error=(0.47,0.58)
SET=0	NUM TFs=2	COMBO INDEX=(0,2)	pval=0.18	stat=0.55	mean=0.49	sd=0.04	error=(0.51,0.61)
SET=0	NUM TFs=2	COMBO INDEX=(0,3)	pval=0.98	stat=0.44	mean=0.49	sd=0.04	error=(0.38, 0.48)
SET=0	NUM TFs=2	COMBO INDEX=(1,2)	pval=0.00	stat=0.67	mean=0.49	sd=0.04	error=(0.64,0.71)
SET=0	NUM TFs=2	COMBO INDEX=(1,3)	pval=0.01	stat=0.65	mean=0.50	sd=0.05	error=(0.605,0.69)
SET=0	NUM TFs=2	COMBO INDEX=(2,3)	pval=0.11	stat=0.57	mean=0.49	sd=0.04	error=(0.51,0.62)
SET=0	NUM TFs=3	COMBO INDEX=(0,1,2)	pval=0.01	stat=0.63	mean=0.49	sd=0.04	error=(0.59,0.67)
SET=0	NUM TFs=3	COMBO INDEX=(0,1,3)	pval=0.08	stat=0.58	mean=0.49	sd=0.04	error=(0.53,0.63)
SET=0	NUM TFs=3	COMBO INDEX=(0,2,3)	pval=0.26	stat=0.54	mean=0.49	sd=0.04	error=(0.49,0.59)
SET=0	NUM TFs=3	COMBO INDEX=(1,2,3)	pval=0.02	stat=0.65	mean=0.49	sd=0.04	error=(0.62,0.69)
SET=0	NUM TFs=4	COMBO INDEX=(0,1,2,3)	pval=0.01	stat=0.64	mean=0.49	sd=0.04	error=(0.59,0.69)

AND COMBINATIONS OUTPUT:

SET=0	NUM TFs=1	COMBO INDEX=(0)	pval=1.00	stat=0.40	mean=0.49	sd=0.04	error=(0.35,0.45)
SET=0	NUM TFs=1	COMBO INDEX=(1)	pval=0.11	stat=0.58	mean=0.50	sd=0.05	error=(0.53,0.63)
SET=0	NUM TFs=1	COMBO INDEX=(2)	pval=0.12	stat=0.57	mean=0.49	sd=0.04	error=(0.52,0.61)
SET=0	NUM TFs=1	COMBO INDEX=(3)	pval=0.21	stat=0.55	mean=0.49	sd=0.04	error=(0.49,0.60)
SET=0	NUM TFs=2	COMBO INDEX=(0,1)	pval=0.31	stat=0.53	mean=0.49	sd=0.04	error=(0.48,0.59)
SET=0	NUM TFs=2	COMBO INDEX=(0,2)	pval=0.32	stat=0.53	mean=0.49	sd=0.04	error=(0.49,0.57)
SET=0	NUM TFs=2	COMBO INDEX=(0,3)	pval=0.97	stat=0.44	mean=0.50	sd=0.04	error=(0.39,0.49)
SET=0	NUM TFs=2	COMBO INDEX=(1,2)	pval=0.01	stat=0.63	mean=0.49	sd=0.04	error=(0.59,0.67)
SET=0	NUM TFs=2	COMBO INDEX=(1,3)	pval=0.02	stat=0.63	mean=0.50	sd=0.04	error=(0.58,0.68)
SET=0	NUM TFs=2	COMBO INDEX=(2,3)	pval=0.20	stat=0.56	mean=0.49	sd=0.05	error=(0.51,0.62)
SET=0	NUM TFs=3	COMBO INDEX=(0,1,2)	pval=0.10	stat=0.59	mean=0.49	sd=0.05	error=(0.54,0.63)
SET=0	NUM TFs=3	COMBO INDEX=(0,1,3)	pval=0.10	stat=0.59	mean=0.50	sd=0.04	error=(0.53,0.63)
SET=0	NUM TFs=3	COMBO INDEX=(0,2,3)	pval=0.28	stat=0.53	mean=0.49	sd=0.04	error=(0.48,0.58)
SET=0	NUM TFs=3	COMBO INDEX=(1,2,3)	pval=0.02	stat=0.62	mean=0.50	sd=0.04	error=(0.57,0.67)
SET=0	NUM TFs=4	COMBO INDEX=(0,1,2,3)	pval=0.01	stat=0.65	mean=0.50	sd=0.05	error=(0.59,0.70)

This output shows a Lever screen on a single gene set using four TFs and the Boolean combinations OR and AND. Here the first column indexes the gene sets that are considered (0-based), and the second column indexes the number of TFs in the combination under consideration. The third column indexes which TFs are in that combination. The fourth column gives the statistical significance of the enrichment after correction for multiple hypothesis testing (either FDR or FWER), and the fifth column gives the value of the AUC that measures the degree of enrichment for the TFs under consideration in the given gene set under consideration (see *Warner et al*). The columns with the "mean" and "sd" indicate the mean and standard deviation, respectively, of the AUC under the permutation test. Finally, the "error" column gives a bootstrap-based estimate of the confidence intervals (1 standard deviation) for the AUC.

Note that if the –WOR, -WAND, or –COMP options had been used, then additional sets of outputs would have appeared for these Boolean output types.

2) A file that indicates which genes are in the various gene sets of interest. This is in the output directory and named "indicator_matrix.txt", and it is the *Y* matrix described above. This file is in a matrix form (tab-separated, new-line delimited), and should have as many rows as there are genes in the input sequence file, and the number of columns should be 2

greater than the number of input gene sets. Here, the first column indexes the number of genes under consideration, and the second column is a "1" if the corresponding gene is in *some* foreground set, and a "0" otherwise. The remaining columns correspond to gene sets, and each element is a "1" if the corresponding gene is in the foreground or background of that gene set, and a "0" otherwise.

3) A file that indicates the scores of all of the genes for each of the motif combinations under consideration. These will be called "ORmatrix.txt", "ANDmatrix.txt", "WORmatrix.txt", "WANDmatrix.txt" and "COMPmatrix.txt" and (depending on which Boolean combinations of motifs you consider) they will appear in your stated output directory. The rows of these matrices correspond to the genes in the input sequence files, and the columns correspond to the various Boolean combinations of motifs that are being considered (thus, this file is the *X* matrix described above).

Note that if you want to pass these matrices on to Lever_statistics as one big matrix, you will need to merge them while keeping corresponding rows together.

Lever_statistics

As stated in the previous sections, Lever performs two basic operations: 1) It first builds the *Y* and *X* matrices that indicate set membership of genes and their scores across various Boolean combinations, respectively. 2) It performs a permutation test in order to calculate the statistical significance of each motif combination within each gene set. The program Lever_statistics allows step 2) to be performed in isolation. It takes as input a pair of *Y* and *X* matrices, and it outputs the statistical significance of genes within gene sets.

If you type "./lever_statistics_1.1" in the directory where the program is stored, the following output appears.

```
Usage:
    Required arguments:
    [-O] [output directory]
    [-N] [output file name]
    [-Y][Indicator Matrix]
    [-X] [Score Matrix]

    Optional arguements:
    [-MH] [FWER or FDR|FDR]
    [-nP] [number of permutations|1000]
```

Here, we explain each of these input parameters, and the syntax for using them. We then explain the outputs of Lever_statistics.

Input parameters

1) [-O] [output directory] This parameter indicates where the directory where all output files will be located.

2) [-N] [output file name] This parameter indicates what to name the output file given by Lever_statistics.

3) [-Y] [Indicator Matrix] This parameter is the indicator matrix described as output 2 of Lever in the preceding section. It contains information on which genes are elements of which gene sets.

4) [-X] [Score Matrix] This parameter is the score matrix described as output 3 of Lever in the preceding section. It contains information on which genes are elements of which gene sets. Note that if it is desired to evaluate various types of Boolean motif combinations (OR, AND, WOR, WAND, COMP) while correcting for multiple hypothesis testing across all of them, then these matrices will need to be merged together into a single matrix.

5) [-MH] [FWER or FDR|FDR] This is exactly the same as the –MH parameter described above for Lever.

6) [-nP] [number of permutations|1000] This is exactly the same as the –MH parameter described above for Lever.

Files Output by Lever_statistics

Lever_statistics outputs a single file that very closely resembles output 1) of Lever described above. However, information on which columns correspond to which Boolean combinations of motifs has been lost. Thus, the output given by Lever_statistics does not contain a column that indicates the number of TFs in the combination. Also, the column with the "COMBO INDEX=" parameter now simply has an integer following it that indexes the corresponding column in the input *X* matrix.