Manual, ver. 03/01/2008, for "PhylCRM_1.1" and the associated program "PhylCRM_preprocess_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

PhylCRM is a computational tool that takes as input a collection of TFBS motifs and a set of input sequences. It then searches for sequence windows that are enriched for evolutionarily conserved clusters of the input motifs. PhylCRM requires a collection of input files that parameterize various statistical properties of the utilized motifs. These parameters are calculated in a separate program PhylCRM_preprocess that have to be executed before PhylCRM. Also, PhylCRM_preprocess 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 Lever manual).

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/PhylCRM/) we have provided pre-compiled versions of the two programs in the PhylCRM suite 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]

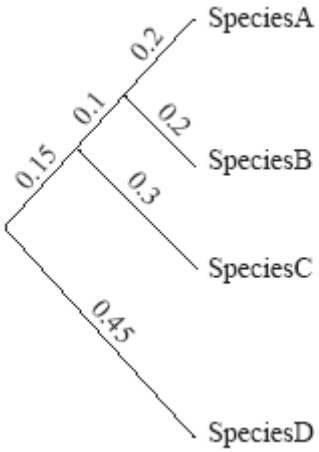
[-W] [max|500]
[-R] [inclusive or restrictive|0]
[-EP] [epsilon value for length matching|0.02]
[-DEOVERLAP]

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. PhylCRM

As stated above, the purpose of PhylCRM is to scan an input collection of sequences with an input collection of transcription factor binding site motifs in order to identify windows of sequence that are enriched for evolutionarily conserved clusters of these motifs.

If you type "./phylcrm_1.1" at the command line in the directory where the program is installed, you get the following output

```
Required arguments:
[-T] [tree file]
[-O] [output directory]
[-N] [output file name]
[-S] [sequence directory]
[-M] [motifs and PFM cutoffs file]
[-BG] [file giving locations of motif parameters]

Optional arguements:
[-OR or -AND or -WOR or -WAND or -COMP | -OR]
[-U] [0 or 1|0]
[-P] [pseudo count|0.1]
[-E] [JC or HKY|HKY]
[-W] [max|500][min|100]
[-R] [inclusive or restrictive |0]
[-C] [window score cutoff|4]
[-l] [long output]
[-s] [spike plot]
[-DEOVERLAP]
[-COUNTS] [apriori given frequency of A C G T file]
```

Here, we explain how each of these input parameters functions, as well as the outputs of PhylCRM.

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 PhylCRM should be. Let's say you specify this parameter to be "-N phylcrm_output.txt." Here, PhylCRM will generate up to three output files that are named "b_phylcrm_output.txt," "l_phylcrm_output.txt" and "s_PhylCRM_output.txt." Each of these will be described in the "**Output files of PhylCRM**" section below.

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. The default cutoff for a PFM is 2 sd.

6) [-BG][file giving locations of motif parameters] This file contains the locations (i.e., directories and file names) of all of the motif parameters files used in the current PhylCRM 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.

7) [-OR or -AND or -WOR or -WAND or -COMP | -OR] This argument contains what type of Fuzzy Boolean combination of motifs to consider. These are described in the supplement to *Warner et al*, where "OR" and "AND" refer to the Fuzzy Boolean OR and AND combinations described there, –WOR and –WAND refer to weighted OR and AND combinations of these motifs, –COMP refers to more complex, compound Boolean combinations (e.g., "A AND B AND NOT C").

There are a few things to be aware of. First, -WOR and –WAND can only be used with combinations involving 2 or three motifs, and it is necessary that numbers indicating the desired weightings follow them, where the weightings are numbers between X and Y. Thus, if one wants to consider an OR combi LOOK UP how to specify the BOOLEAN combinations

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

9) [-P] [pseudo count|0.1] This is exactly the same as the –P parameter described above for PhylCRM_preprocess. 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. 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 very similar to the –W parameter that was used by PhylCRM_preprocess in order to specify the maximum window size to consider. Here, however, both a maximum and a minimum window size must be specified. For example "–W 700 150" specifies a maximum window size of 700 and a minimum of 150 in looking for candidate CRMs. If no value is specified, then a maximum size of 500 is used and a minimum window size of 100 is used. Note again that the maximum window size used must be less than what was used in PhylCRM_preprocess.

12) [-R][inclusive or restrctive|0] This is exactly the same as the –R parameter described above for PhylCRM_preprocess. 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) [-C] [window score cutoff|4] If you have specified that you wish to have PhylCRM give the "long form" of the output by including the parameter "-l" (see below), then all hits above a given cutoff score will be shown in the long output. This parameter specifies the window cutoff score to use. If no number is specified, then it uses a cutoff score of 4.0.

Note that the choice of a suitable cutoff will change according to the number of motifs considered, as well as the Boolean combination of motifs considered. In general, if you are using the –OR combination of motifs, then window scores tend to rise as the number of considered motifs rises. Conversely, if you are using the –AND combination of motifs, the window scores tend to be smaller than their –OR counterparts, and the window scores tend to get smaller as the number of considered motifs rises (the rationale for this is explained in the Supplementary Methods of *Warner et al*). Thus, it might take some familiarity with the program in order to choose a suitable cutoff.

14) [-l] [long output] This parameter specifies whether or not to output the "long form" of PhylCRM output (see "**Output files of PhylCRM**" for a description of this output).

15) [-s] [spike plot] This parameter specifies whether or not to output the "spike plot" form of the output for PhylCRM that can be used to visualize the Halpern-Bruno scores of the motif matches (see ""**Output files of PhylCRM**" for a description of this output).

16) [-DECORR] This is exactly the same as the –DECORR parameter described above for PhylCRM_preprocess. 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.

17) [-COUNTS] [frequency of A C G T file] As part of its scoring scheme, PhylCRM accounts for the mononucleotide frequencies of letters. If the sequences you are scanning with PhylCRM are not very large or have a very biased sequence composition, then this parameter allows you to input a different set of mononucleotide frequencies. For example, if you used a large set of sequences to generate the motif parameters with PhylCRM_preprocess, and you then want to use these to scan a smaller set of regions that you are especially interested in, then you might want to input a counts file to insure that the nucleotide counts that PhylCRM is using to score windows are the same as what was used by PhylCRM_preprocess to generate the motif parameters file. The form of this counts file is

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

Output files of PhylCRM

PhylCRM outputs three basic files.

1) A brief output file. If the –N parameter you used was, say, "-N phylcrm_output.txt," then this file is named "b_phylcrm_output.txt". It contains only the most basic information on the scores of the input sequences, giving the score, window size, and location of the highest scoring PhylCRM hit for each window of sequence. It looks like

```
>sequence1  chr=1   start=54312   stop=65371   strand=+     512   77    3.43
>sequence7  chr=13  start=10001   stop=17391   strand=-     812   133   4.12
>sequence3  chr=21  start=12020   stop=15533   strand=+     101   255   3.00
```

The last entry in the tab-delimited list is the PhylCRM score of the highest scoring window for that sequence. The second-to-last entry is the size of the highest scoring window of sequence. The third-to-last entry is the location (1-based) relative to the start of the input sequence where the highest scoring window occurs. All entries that precede the third-to-last entry should be exactly the same as the identification line in the input sequence file (i.e., the sequence that begins with ">").

2) A long output file. If the –N parameter you used was, say, "-N phylcrm_output.txt," then this file is named "l_phylcrm_output.txt". Note that this output file will only be generated if you use the "-l" parameter at the command line (see 14 above). It contains more detailed information on *all* of the windows of sequence above some input cutoff score (parameter "-C" listed above in this section). The output file contains records that look like the following

```
>sequence20          chr=10 start=57312    stop=65371    strand=+
      region location: (21937, 21980)
      max window in region: position = 21937, size = 44, score = 6.15810
      motif 1: 1      (21937)       (1.44441)
      motif 2: 1      (21971)       (2.15328)
```

region location: (31415, 31484)
max window in region: position = 31415, size = 70, score = 5.97709
motif 1: 3 (31432, 31475, 31477) (1.30801, 1.65516, 1.28941)
motif 2: 1 (31415) (1.20404)

where there is one such record for each input sequence containing at least one window of sequence scoring above the input threshold. Here, the first line of the record (the line that beigns with ">") is exactly the same as the identification line in the input sequence files.

Next, let's say that you used as an input cutoff "-C 4.0". If there is one window of sequence that scores above this cutoff, then there are likely to be many such windows that are largely overlapping. For example, let's say that the maximum scoring window occurs between positions (13445,13501) and has a score of 5.5. Then the windows (13444, 13501) and (13445, 13502) would score only slightly worse (each being 1 bp bigger than the maximum scoring window), and thus both would also likely score above the cutoff of 4.0. It is desired to output all windows that score above the input cutoff, but outputting 100's of largely overlapping windows would clearly be a nuisance. To resolve this, PhylCRM first identifies all windows that score above some input cutoff score, and it then merges all overlapping windows. We shall henceforth refer to these merged sets of windows as "regions". Therefore the line

region location: (21937, 21980)

states that there was a set of windows scoring above the input cutoff that were merged together into a region. This region begins at position "21937" and ends at position "21980" *relative to the start of the corresponding sequence* (i.e., this coordinate is 1-based and refers to the position relative to the start of the input sequence, not the actual genomic coordinate).

The next line

max window in region: position = 31415, size = 70, score = 5.97709

indicates where the maximum-scoring window within the corresponding region is located. In this example, the maximum scoring window begins at position 31415, is 70 bp in length, and has a score of 5.97709.

Finally, the lines

motif 1: 1 (21937) (1.44441)
motif 2: 1 (21971) (2.15328)

contain information on where the motif matches can be found within the corresponding windows. There is one line for each input motif (so in this case we were considering two input motifs). The number directly after the colon indicates how many motif matches can be found in the region (so in this example there was one match to each motif). The following set of numbers that are encased by parentheses indicate where these motif matches are located (again 1-based and relative to the start of the input sequence). Finally, the last set of numbers in parentheses gives the Halpern-Bruno score that indicates the degree of conservation, *divided by the width of the motif* (the motivation for

dividing by the width of the motif is explained in the Supplementary Methods of *Warner et al*, as we use it to avoid double-counting overlapping motif matches).

3) A file that can be used for graphically visualizing the corresponding "spike plots." If the –N parameter you used was, say, "-N phylcrm_output.txt," then this file is named "s_phylcrm_output.txt". It will only be generated if you use the "-s" parameter at the command line (see parameter 15 above). This file contains columns of numbers, where there is one column per motif, and one row of numbers for each position within each region scoring above the input cutoff. The output of the file looks like

```
>sequence20          chr=10 start=57312     stop=65371     strand=+
      region location: (21937, 21980)
      max window in region: position = 21937, size = 44, score = 6.15810
      21937  1.44441          0.00000
      21938  1.44441          0.00000
      21939  1.44441          0.00000
      21940  1.44441          0.00000
      21941  1.44441          0.00000
      21942  1.44441          0.00000
      21943  1.44441          0.00000
      21944  1.44441          0.00000
      21945  0.00000          0.00000
      21946  0.00000          0.00000
      21947  0.00000          0.00000
      …
```

Note that this region corresponds to the first region shown in the previous example for the long output. The first three lines are exactly the same as what was shown for the –l output files. The following three columns of numbers indicate the position within the region (first column), the Halpern-Bruno score divided by the width of the motif for motif 1 at each position in the region (second column), and the Halpern-Bruno score divided by the width of the motif for motif 2 at each position in the region (third column).

These output files are used for generating the kinds of "spike plots" shown in Supplementary Figures 1 and 3 of our Warner *et al.* paper, and the output format is designed to be easily pasted into programs like Excel. Note that these output files can be *very* large, so you might want to only use the –s parameter if you are considering a relatively small number of input files.