WeederTFBS - 1.2
User Manual

Giulio Pavesi
D.I.Co, University of Milan
Milan, Italy
`pavesi@dico.unimi.it`

November 14, 2005

Contents

1 Intro

WeederTFBS is a minimal, no frills implementation of the Weeder algorithm, where search parameters and statistical evaluation have been fine–tuned for the discovery of transcription factor binding sites (TFBSs) in DNA regulatory regions.

If you downloaded the program, you're probably already familiar with the biology of TFBSs. However, you might want to read, other than the NAR and Bioinformatics papers on Weeder [1, 4], also a brief survey we published on the computational discovery of TFBSs and methods available as of, well, a few months ago[2]. Also, an assessment of the performance of various tools for this task has been performed [3].

"Minimal" and "no frills" implementation means that we intentionally left out some computational tricks in order to improve memory efficiency and portability. And, most of the tricks we use in the general algorithm (Weeder was originally meant to be a general purpose motif discovery tool) are of little use in the search for short motifs like TFBS. Unfortunately, the most time consuming part is the statistical evaluation of the motifs (that is, examining all the approximate forms of each candidate oligo, see the supplementary material of [1]), that could (and will) be improved in the future versions of the program.

2 Unpacking, Compiling and Installing

The Linux/Unix release of Weeeder comes as a .tar.gz package. To unpack it, type

```
gunzip weeder1.2.tar.gz
```

followed by

```
tar xvf weeder1.2.tar
```

At this point, a directory named "Weeder1.2" should have appeared. Among other things, it contains a file named "compileall", and two directories named "FreqFiles" and "src" (containing the source code). To compile the program, just type:

```
./compileall
```

If all went well, four executable files named "weederlauncher.out", "weederTFBS.out", "locator.out" and "adviser.out" have appeared in the directory you're in. The programs are written in C, so you should have a C compiler on your computer (usually, it's gcc). If you have a C compiler other than gcc, just edit the compileall file, and replace gcc with the name of your compiler.

Instead, if you got the Windows executables, just unzip the file, and you'll be exactly at this point.

Important: to run correctly, the program needs the species–specific background frequency files, contained in the directory "FreqFiles". You are free to move the executables wherever you want, *as long as the FreqFiles directory is moved along with them, and it's kept in the same position w.r.t. the executable files.*

3 Preparing the Input

4 Running the Program

To run the program in the default settings (like the ones used in the Web interface), you can use the "weederlauncher.out" program. The syntax is the following:

```
weederlauncher.out filename organism analysis <additional parameters>
```

where:

- **filename** is the input file containing your sequences
- **organism** is a two letter code denoting the species your sequences come from. The two letter code (in capitals) corresponds to the initials of the latin name of the organism: thus, HS is Homo sapiens, MM Mus musculus, DM D.melanogaster, and so on. NOTICE: bacteria have the code preceded by "B". Thus, BEC is E.coli, BBS is Bacillus subtilis, and so on. The code–organism correspondence can also be found in the `organisms.txt` file.
- **analysis** the type of analysis you want. This parameter influences the length of the motifs sought, and the degree of approximation allowed (see [1]). Suitable values are:
  - *small*: Motifs of length 6 and 8 (quick mode of the Web interface)
  - *medium*: Motifs of length 6 8 and 10.
  - *large*: Motifs of length 6,8, 10 and 12 (normal mode of the interface)
  - *extra*: Same as large, but a higher degree of approximation is allowed for lengths 8 and 10 (thorough mode of the interface).

We allowed different types of analysis mainly because of the execution time. While for lengths 6 and 8 it is measured in seconds also for large files and long sequences, the statistical evaluation needs more time for longer lengths. The ballpark is minutes for length 10, and (unfortunately) more than 10-15 minutes, and sometimes even hours or days for length 12. So the idea is, in case of large files, to limit the analysis to short lengths, and save the "huge" jobs for later, in case the results have been interesting. In the following I'll explain how to continue a "short" analysis that has been already completed by adding longer motifs.

The three parameters above are required, exactly in that order. Plus, you can specify some additional parameters:

- `S`: consider both strands of the input sequences. Default is single–stranded analysis.
- `A`: assume that motifs appear in all the sequences of the set. Default is that motifs appear in some sequences of the set.
- `M`: a motif can appear more than once in a sequence. Default is that we expect a motif to appear once (or not to appear) in each sequence. This influences the statistical evaluation, not the results (more than a single occurrence can be reported even if you don't set this parameter).
- `Tn`: save the `n` (that should be an integer number) highest–scoring motifs of each run. Default is 10. Increasing it increases the number of putative TFBSs reported after the final post–processing. Viceversa if you reduce it (in the assessment, we used 5, since we wanted just a single motif for dataset). The more sequences you have, the longer they are, the more motifs they can contain: thus 10 is good for yeast: for human, you can use much higher values. Of course, the motifs reported are only those the program deems significant: thus if you set 100 and 70 motifs are found, 70 motifs (and not 100) are output.

All in all, writing for example

```
./weederlauncher.out input.fasta HS large A M S T15
```

makes the program look for motifs of length 6,8,10, and 12, in both strands of the input sequences (that come from HS - Homo sapiens), assuming that motifs appear in all the input sequence, even more than once in each sequences, and saves the best 15 results of each run.

The launcher program starts a series of runs of the weederTFBS.out program, each with a different length and approximation value. The results of the runs are saved in different files. Suppose that input.fasta is the name of your input file. The results files are:

- **input.fasta.wee** This is a text file containing the highest scoring motifs of each run, as well as the results of the final post–processing (see [1]). Basically, it's the file Web users receive by e-mail. At each run, the results are appended to the end of the file.
- **input.fasta.html** The same as above, but it's an HTML file with a better graphic layout. It's the file you can see with your browser in the Web implementation.
- **input.fasta.mix** This is a "service" file that the program uses to perform the final post-processing. It's created anew each time you use the launcher.

When the runs are completed, the "adviser" program is started. It reads the motifs saved in the ".mix" files, looks for redundant motifs (as explained in [1]), and appends its advice to the .wee and .html files.

4.1 Frequency Files

The FreqFiles directory contains the files that are used by Weeder to assess the expected frequency values of motifs. Files are species–specific, and contain for each organism expected values for sixmers and eightmers (expected frequencies for longer motifs are computed using a Markov model starting from the eightmer frequencies). The frequencies have been computed by performing an oligo count of the upstream regions of all the genes annotated for each organism (sequences come from [5]). Choosing the length of these upstream regions posed a problem. That is, while for "simple" organisms (yeasts up to Drosophila) there was no significant variation by comparing the oligo frequencies in the 1000 bp upstream and in the whole intergenic regions (Pearson correlation $> .9$), for human, mouse and so on the frequencies varied significantly. In the assessment [3] we used for all the species the frequencies of the 1000 bp upstream of the genes. However, recent results show how the frequency of some oligos in the intergenic regions is lower than in the regions immediately upstream from genes, and many of these oligos are annotated TFBS. Thus, according to these results, using the whole intergenic regions as backgroud frequencies should better highlight real TFBSs in core promoters. Unfortunately, using intergenic regions in the assessment datasets worsened the performance, but this might also depend from the fact that many of the datasets were not real sequences but "artificial" sequences built with a Markov models with planted TFBSs. All in all: for some species (human, mouse) we included also the intergenic frequency files. Just add "I" at the end of the code: HSI for human–intergenic, MMI for mouse, and so on. And, it would be interesting to compare results obtained with the two types of background frequencies.

If the organism you're working on is not in the list of organisms available, you can send me an e-mail, and I can provide the frequency files needed.

5 Reading the Output

A priori, the best (real?) motifs should be the highest scoring ones of each run. However, there are two factors to be considered: (1) the highest scoring motifs very often disagree, i.e., they are completely different, and thus it is hard to say which is the best one, and (2) unfortunately, real motifs often are not the highest scoring ones of any run.

In our experiments, we noticed an interesting feature: the "good" motifs are often *redundant*. By "redundant", we mean that: (1) if a motif is among the highest scoring ones of a run searching motifs of length m allowing e substiutions, then there are other motifs, again among the highest scoring ones of the same run, whose consensus is within e substitutions from the "good" one; (2) in the same run, there are other motifs whose consensus overlaps the "good" one; (3) in runs considering motifs of different lengths, there are among the highest scoring ones motifs whose consensus is contained within the "interesting" one (if they are shorter) or contains (if they are longer) the "interesting" one.

All in all, then, the adviser program scans the list of highest scoring motifs of each run, computing for each one the list of redundant motifs according to the criteria just explained, and reports in detail only those that have redundant mates both within the same run and in runs of different length.

So, which ones are the best? If a motif is the highest scoring one of its run and it is the one with the highest number of redundant companions, than it should be good; or, if a non-highest scoring one presents a higher number of relatives w.r.t. another one that according to the score should be better, then perhaps it is more likely to be ”good”.

6 Running the Program Without the Launcher

Runs of Weeder can of course be started without using the launcher. Required parameters are:

```
weederTFBS.out -f filename -W motifwidth -e mismatch_number -R sequences_percentage
-O organismcode
```

The additional parameters, other than filename and organism code, are the motif width (that must be even, ranging from 6 to 12), the error percentage (-e : number of mismatches allowed), and percentage of sequences (-R) that must contain the motif. Additional parameters are:

- **-S**: Process both strands of the input sequences (default is single stranded)
- **-T number**: report the ”number” highest scoring motifs of the run (default is 10). Notice the space between -T and the number
- **-M**: a motif can appear more than once in each sequence (as before, default is zero or one occurrence per sequence)
- **-V**: verbose mode.

Notice that by starting a run in this way, results are appended to the .mix .wee and .html result files. Thus, you can perform a series of runs by yourself, and at the end nevertheless run the ”adviser” program. Just type `./adviser.out filename`, where filename is the original input file of your analysis. If you performed double–stranded analyses, you can run the adviser to take this into account, and compare also the reverse complement of the motifs found. Command line in this case is `./adviser.out filename S` (add an ”S” at the end of the line). In this way, you can expand an analysis by performing the additional runs. Suppose you completed a medium job (`./weederlaucher.out filename HS medium`), and now you want to add motifs of length 12. You can type

```
weederTFBS.out -f filename -W 12 -e 30 -R 50 -O HS
```

The results of this additional run will be appended to the output and .mix files: if at the end you run the adviser, then you'll get the additional advice (appended at the end of the .wee and .html files) considering also motifs of length 12. Notice instead that every time you run the launcher the .mix file is cleared: thus the new results are processed by themselves. Also, you can save separately results of different experiments (the .mix files), merge them into a single .mix file, and run the adviser on the latter. Finally, if you get impatient, you can run the adviser while Weeder has still to complete an analysis (e.g., it is working on length 12 but you already have the results on 6,8,and 10), and see whether something interesting came up in the results.

The launcher program starts a series of run with the following parameters:

- **small**: length 6 with 1 mutation (-W 6 -e 1), length 8 with 2 mutations (-W 8 -e 2).
- **medium**: like small, plus length 10 with 3 mutations (-W 10 -e 3).
- **large**: like medium, plus length 12 with 4 mutations (-W 12 -e 4).
- **extra**: length 6 with 1 mutation (-W 6 -e 1), length 8 with 3 mutations (-W 8 -e 3), length 10 with 4 mutations (-W 10 -e 4), length 12 with 4 mutations (-W 12 -e 4).

7 I Got My Output. And Now?

Well, the first thing to do is to check whether Weeder came up with known binding sites. You probably already know how to do this, but in case you don't you can connect to the TRANSFAC database (or SCPD for yeast). Then, you can go to the "site" search and look whether the motif appears in the "sequence" field of some site (if the motif is long, you might also check for parts of it). Also, try some of the occurrences reported by Weeder: often the consensus does not appear, but some of its approximate occurrences do (and perhaps the most interesting ones are those common to the motifs reported as "redundant"). It takes a bit of copying and pasting, but it's quite automatic. Also, TFBSs are annotated for some genes: thus you can also go to the entries corresponding to your genes, and check whether Weeder reported in their sequence something that was already annotated. If you find something, you have an hint on a possible common TF regulating your sequences.

8 Using the Locator

The program called locator.out included in the Weeder package is a utility designed for locating in a set of sequences instances of a motif described with a consensus. It might be handy for studying more in depth a motif not considered to be "interesting" by Weeder, or to re-process the occurrences reported by the adviser. In fact, the default threshold of 85% for reporting the best occurrences of a motif sometimes has the effect of reporting too few or too many occurrences. The strategy used is the same employed in the adviser program to locate occurrences of interesting motifs. The program takes as input a set of sequences, a motif, and threshold values for substitutions and score percentage.

The syntax is the following:

```
./locator.out inputfile consensus substitutions threshold
```

Parameters must appear in the order they are listed above. "inputfile" is the sequence file (in FASTA format); "consensus" is the motif descriptor, that can be of any length; substitutions is the maximum number of substitutions in motif occurrences and, finally, threshold is a number between 1 and 100 indicating the percentage threshold for the occurrences to be reported. To perform a search on double strands, add S at the end of the command line. As in the post - processing performed by the advisor first all the occurrences of the consensus with at most the specified number of substitutions are collected; then, a frequency matrix is built; next, the sequences are scanned with the matrix, and all the occurrences with percentage value greater than the specified threshold are reported. The "All Occs" and "Best Occs" matrices are also reported. The output is written in two files, called inputfile.consensus.wee and inputfile.consensus.html, for text and html-formatted output, respectively.

9 What's New

There are a couple of minor changes w.r.t. the original NAR paper:

- Changed the output layout, now more "MEME-like".
- Added the *locator* program. See the corresponding section for instructions.
- Fixed a bug that sometimes computed higher background frequencies for palindromes and quasi–palindromes in the "double strand" mode.
- Changed the computation of background frequencies for motifs longer than 8 nts. Instead of proceeding left–to–right we now move from the core to the edges. For example, let $p = p_1 \dots p_{10}$. We have

  $$Exp(p) = \Pr(p_1|p_2 \dots p_8) \times Exp(p_2 \dots p_9) \times \Pr(p_{10}|p_3 \dots p_9)$$

  Analogously for motifs of length 12.

10 Frequently Asked Questions

- **Can I use the program for UTRs or introns?**. A priori, yes. My suggestion is to use upstream frequencies for UTR (whose definition is very often quite blurred) and intergenic for introns.
- **Can I use the program for the analysis of the same gene from different organisms?** Yes. As a matter of fact, TFBSs are often more conserved in orthologous sequences than in the same organism. Just choose

the frequencies of your ”reference” species, and set motifs appearing in all sequences. Advice: report many motifs (even 100!) in each run, since there will be many conserved and overlapping motifs. However, we are addressing this type of analysis right now, and we are preparing a special version of Weeder for this case, that takes into account some additional considerations.

- **In the Nature Biotechnology assessment [3], the performance of the various tools on human have been quite disappointing: is there any chance to get some useful result?** One reason for the worse performance of motif discovery tools on human and mouse could be that, while in yeast a few TFBSs are responsible for the transcription of a gene, in metazoa the regulation is more complex, i.e. regulatory regions contain many more TFBSs. In the assessment we were asked to report just one motif per dataset: and often, in my results, the answer was in the list, but not in the best spot. To give an example, I got on several man and mouse datasets SP1 BSs as best answer, in one I had a perfect TATA box (even located 20bps upstream from the gene!). Unfortunately, in these cases there were other binding sites that corresponded to the ”solution”. Finally, we did not use the double strand option (and, after a bug in the code has been fixed, it surely can somewhat improve the performance of Weeder).

  Thus, the idea is to report more than 10 motifs (this parameter was set to have one or two motifs), and to look also to non highest scoring motifs. Another idea is to select from the sequences the most conserved parts w.r.t. orthologous genes, and mask out from the sequences the non–conserved parts. This is quite a tedious work, but reduces false positives. Also, you can cross-check the results with an analysis on orthologous sequences performed with Weeder.

- **I want to explore a motif by myself, or one that wasn’t reported as interesting by Weeder**. You can use the pattern matching facilities at RSAT tools (`http://rsat.ulb.ac.be/rsat/` [5]). Quite easy to use, they allow you to perform exactly the same type of analysis the adviser performs on interesting motifs (including building a frequency matrix), plus it has a nice graphical output. Moreover, the occurrences reported by the adviser are collected by using the ”All Occs” matrix. With these tools, you may perform an additional scan of the sequences using the ”Best Occs” matrix, build a new one by using the occurrences collected, and so on.

- **Is there any chance to find TFBS in enhancers located far away from the genes?**. Unfortunately, given the statistical evaluation we perform, running the program on, say, the 20Kbp upstream of your genes does not work. You’ll probably get only false positives. A partial solution is to run the program on core promoters separately. Then, for far away regions, to filter them by aligning with orthologous sequences, until from the 20kbp you get about 1-2 K of conserved sequence and mask out the rest. We anyway will address this problem in the next version of Weeder.

- **Will FC Inter Milan even win the Italian Soccer League?** Sure. Next year.

11 All in All

Well, this is basically it. If you have any bug report, question concering the program, its usage, or the best way to use it to perform a given analysis, or for adding new organisms, feel free to contact me (`pavesi@dico.unimi.it`). Good luck with your research, and in case we've been useful, don't forget to cite us!

References

[1] Pavesi G, Mereghetti P, Mauri G, Pesole G. Weeder Web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. *Nucleic Acids Res.* 2004 Jul 1;32(Web Server issue):W199-203.

[2] Pavesi G, Mauri G, Pesole G. In silico representation and discovery of transcription factor binding sites. *Brief Bioinform.* 2004 Sep;5(3):217-36.

[3] Tompa M, Li N, Bailey TL, Church GM, De Moor B, Eskin E, Favorov AV, Frith MC, Fu Y, Kent WJ, Makeev VJ, Mironov AA, Noble WS, Pavesi G, Pesole G, Regnier M, Simonis N, Sinha S, Thijs G, van Helden J, Vandenbogaert M, Weng Z, Workman C, Ye C, Zhu Z. Assessing computational tools for the discovery of transcription factor binding sites. *Nat Biotechnol.* 2005 Jan;23(1):137-44.

[4] Pavesi G, Mauri G, Pesole G. An algorithm for finding signals of unknown length in DNA sequences. *Bioinformatics* 2001;17 Suppl 1:S207-14.

[5] van Helden J. Regulatory sequence analysis tools. *Nucleic Acids Res.* 2003 Jul 1;31(13):3593-6.