GimmeMotifs documentation

Simon van Heeringen

December 7, 2011

Contents

1 Introduction

GimmeMotifs is a *de novo* motif prediction pipeline, especially suited for ChIP-seq datasets. It incorporates several existing motif prediction algorithms in an ensemble method to predict motifs and clusters these motifs using the WIC similarity scoring metric. It is freely available for download and use under the MIT license. If you find GimmeMotifs useful, please cite:

- van Heeringen SJ and Veenstra GJC, GimmeMotifs: a *de novo* motif prediction pipeline for ChIP-sequencing experiments, Bioinformatics Advance Access published November 15, 2010, doi:10.1093/bioinformatics/btq636.

This document describes how to install and use GimmeMotifs, for theoretical details, please see our publication [1].

Hopefully this document explains at least the basics of installation and usage, but it's probably far from complete. If you have any further question, please don't hesitate to contact me: `s.vanheeringen@ncmls.ru.nl`.

2 What's included?

GimmeMotifs is dependent upon the great work of others: motif prediction algorithms. GimmeMotifs includes several tools by default:

- BioProspector [2] `http://motif.stanford.edu/distributions/bioprospector/`
- GADEM [3] `http://www.niehs.nih.gov/research/resources/software/gadem/index.cfm`
- Improbizer [4] `http://users.soe.ucsc.edu/~kent/`
- MDmodule [5] (included in the MotifRegressor Package) `http://www.math.umass.edu/~conlon/mr.html`
- MEME [6] `http://meme.sdsc.edu/`
- MoAn [7] `http://moan.binf.ku.dk/`
- MotifSampler [8, 9] `http://homes.esat.kuleuven.be/~sistawww/bioi/thijs/download.html`
- trawler [10] `http://ani.embl.de/trawler/`

All these tools are installed automatically with GimmeMotifs. Due to an issue with MoAn and the Ubuntu gcc version, I was not able to compile MoAn for the Ubuntu package, which means MoAn is not included. To use these programs *no additional steps have to be taken* (except for MotifSampler see 3.3.6). **Please note: all these programs include their own license and most are free for academic or non-commercial use only! For commercial use of any of these programs, please consult the respective author!**

3 Installation

GimmeMotifs runs on Linux. Definitely not on Windows, sorry. Mac OS X should work in theory, but as I don't have the means to test this, I'm not completely sure. I have tried to make installation of GimmeMotifs as easy as possible. There are installation packages available for Fedora, Ubuntu and Debian. It's also possible to install GimmeMotifs from source, but it depends on quite some external packages. Please make sure all prerequisites are installed before installing GimmeMotifs from source.

3.1 Installation packages

3.1.1 Installation on Ubuntu or Debian

Download the latest GimmeMotifs .deb package for your distribution from `http://www.ncmls.nl/bioinfo/gimmemotifs/` (available for 32-bit and 64-bit systems). Please note: due to a different Python version, the Debian and Ubuntu .deb packages are different! Please use the correct one for your distribution. These packages have been tested on up-to-date versions of Ubuntu Oneiric Ocelot (11.10) and Debian sqeeze (6.04). Open a terminal and install GimmeMotifs as follows. Go to the directory where you downloaded GimmeMotifs, for example:

```
cd ~/Downloads
```

Start the install (substitute the package name for the .deb package you downloaded):

```
sudo dpkg -i gimmemotifs_0.65_amd64.deb
```

Likely, dpkg will complain about some missing dependencies. Install all dependencies with:

```
sudo apt-get -f install
```

Complete the GimmeMotifs installation with:

```
sudo dpkg -i gimmemotifs_0.65_amd64.deb
```

Currently, there is a bug with the versions of the Parallel Python (python-pp) and Numpy (python-numpy) in the Ubuntu and Debian repositories. Therefore the package python-pp is not installed as a dependency. This can be fixed by installing the latest version of Parallel Python from the Python Package Index:

```
sudo easy_install pp
```

Now you should have a working version of GimmeMotifs! The next steps are to install additional motif tools (optional, see section 3.3.5) and to do some configuration (required, see section 3.3). You can also directly try the quick example (section 4.1), if you're impatient (but don't forget to perform the additional steps!)

3.1.2 Installation on Fedora

Download the latest GimmeMotifs .rpm package from `http://www.ncmls.nl/bioinfo/gimmemotifs/` (available for 32-bit and 64-bit systems). This package has been tested on an up-to-date version of Fedora 16. Install GimmeMotifs as follows (substitute the package name for the .rpm package you downloaded):

```
sudo yum install --nogpgcheck gimmemotifs-0.65-1.x86_64.rpm
```

GimmeMotifs doesn't play nice with SELinux enabled on Fedora, sorry. To turn it off:

```
sudo setenforce 0
```

Now you should have a working version of GimmeMotifs. The next steps are to install additional motif tools (optional, see section 3.3.5) and to do some configuration (required, see section 3.3). You can also directly try the quick example (section 4.1), if you're impatient (but don't forget to perform the additional steps!)

3.2 Installation from source

3.2.1 Prerequisites

Before you can install GimmeMotifs you'll need:

- some Python modules and other packages
- motif prediction tools

3.2.2 Required packages (Python)

- Python 2.5, 2.6 or 2.7 (not Python 3) `http://www.python.org`
- Scipy `http://www.scipy.org/`
  SciPy is the fundamental package needed for scientific computing with Python.
- matplotlib (0.98 or higher) `http://matplotlib.sourceforge.net/`
  A python 2D plotting library. All figures and plots produced by GimmeMotifs are made using matplotlib.
- parallel python 1.6.0 `http://www.parallelpython.com/`
  A python module which provides mechanism for parallel execution of python code. This Python library is used for parallel execution of for instance the motif finding tools.
- kid `http://www.kid-templating.org/`
  A simple template language for XML based vocabularies; used to produce the HTML reports.

3.2.3 Other required packages

- gsl `http://www.gnu.org/software/gsl/`
  The GNU Scientific Library. This library might already be installed on your system, but you'll also need the development headers to compile GimmeMotifs!.

3.2.4 Additional motif prediction programs

A lot of motif prediction tools are compiled and/or installed with GimmeMotifs. The following tools have to be installed seperately:

- Weeder [11] `http://159.149.109.9/modtools/`

Please consult the respective manuals regarding installation of these tools. It's always possible to install these programs after installation of GimmeMotifs and update the configuration files to include the new tools (see section 3.3.5). However, during installation, GimmeMotifs will try to find any installed tools and add them automatically, so that's the easiest option.

3.2.5 Building from source

You can download the lastest version of GimmeMotifs at:
`http://www.ncmls.eu/bioinfo/gimmemotifs/`.

Start by unpacking the source archive

```
tar xvzf gimmemotifs-1.00.tar.gz
cd gimmemotifs-1.00
```

You can build GimmeMotifs with the following command:

```
python setup.py build
```

Run the tests to check if the basics work correctly:

```
python run_tests.py
```

If you encounter no errors, go ahead with installing GimmeMotifs (root privileges required):

```
sudo python setup.py install
```

During installation GimmeMotifs will try to locate the tools you have installed. If you have recently installed them, running an `updatedb` will be necessary. Using this option GimmeMotifs will create a configuration file, the default is:

```
/usr/share/gimmemotifs/gimmemotifs.cfg
```

This is a system-wide configuration that can be used by all users.

It is also possible to run the `setup.py install` command with the `--prefix`, `--home`, or `--install-data` options, to install in GimmeMotifs in a different location (for instance, in your own home directory). This should be fine, however, these alternative methods of installing GimmeMotifs have not been extensively tested. Please note that in this case the configuration file will be created, but every user will have to put this configuration file in his/her home directory: `~/.gimmemotifs.cfg`. The install script will also inform you of this during install. Please contact me if you run into problems with the installation. Once the installation is finished, you can try the quick example (section 4.1), or continue with the configuration in the next section.

3.3 Configuration

3.3.1 Data sources

You will need some genome fasta files for any motif-prediction if you want to run GimmeMotifs with BED files as input (which is recommended, although GimmeMotifs will also work on FASTA files). To get from a BED file to the sequence information these genomic fasta files are absolutely required. The fasta files should be organized in one directory with one file per chromosome or scaffold, with the filename being the chromosome name with an extension of `.fa`, `.fsa` or `.fasta`. No exceptions, no different layouts. A good source is the UCSC Genome Browser database [12]. For instance, the human hg18 files needed to run the examples included with GimmeMotifs can be downloaded here:

```
ftp://hgdownload.cse.ucsc.edu/goldenPath/hg18/bigZips/chromFa.zip
```

All fasta files need to be indexed before GimmeMotifs can use them, see section 3.3.2.

3.3.2 Indexing the genomes

All the genomes that you want to use with GimmeMotifs will need to be indexed for (relatively) fast retrieval of sequences. You can do this, once you have installed GimmeMotifs, by running the following command (as root or with sudo):

```
create_genome_index.py -f /dir/to/fasta/files/ -n genome_name
```

For instance, if I wanted to index the human genome (version hg18) on my computer, where all fasta files are located in the directory `/usr/share/genome/` I would run the following command:

```
sudo create_genome_index.py -f /usr/share/genome/hg18/ -n hg18
```

Repeat this step for every additional genome or organism that you want to use GimmeMotifs with. Please note: for Weeder, currently only hg18, hg19, mm9, sacCer2 and xenTro2 are supported as organism names (following the UCSC naming convention). This will be fixed as a configuration file in a later release.

3.3.3 Adding gene files

When using the `genomic_matched` background setting (which is the default), there needs to be a file describing genes in BED format in the `gene_dir`, which is defined in the configuration file. By default this is: `/usr/share/gimmemotifs/genes/`. The file needs to be named `<index_name>.bed`, so for instance `hg18.bed`. By default `hg18.bed`, `mm9.bed` and `xenTro2.bed` are included.

3.3.4 The easy way: `add_organism.py`

The script `add_organism.py` combines the previous two steps (indexing the fasta files, and adding a gene file), and makes sure the gene BED file is in the correct place with the correct name. This is the easiest way to add a new genome/organism for use with GimmeMotifs.

3.3.5 Adding motif prediction tools

Please note that these steps are only necessary when you have installed any of these tools after you have installed GimmeMotifs.

Weeder

After installing Weeder the following section needs to be added to the GimmeMotifs configuration file:

```
[Weeder]
bin = /usr/share/Weeder/weederTFBS.out
dir = /usr/share/Weeder/
```

All other Weeder binaries should be present in the same directory as `weederTFBS.out`. The directory specified by `dir` should contain the FreqFiles directory included with Weeder. In addition `Weeder` should be added to the line in the `params` section of the configuration file. For instance

```
tools = MDmodule,MEME,MotifSampler,trawler,Improbizer,BioProspector
```

needs to be changed to:

```
tools = MDmodule,MEME,MotifSampler,trawler,Improbizer,BioProspector,Weeder
```

3.3.6 MotifSampler configuration

If you want to use MotifSampler there is one more step that you'll have to take *after* installation of GimmeMotifs. For every organism, you'll need a MotifSampler background. Note that human (hg18), mouse (mm9) and *Xenopus* (xenTro2) background models are included, so for these organisms MotifSampler will work out of the box. For other organisms the necessary background files can be created with `CreateBackgroundModel` (which is included with GimmeMotifs or can be downloaded from the same site as MotifSampler). The background model file needs to be saved in the directory `/usr/share/gimmemotifs/MotifSampler` and it should be named `<organism_index_name>.bg`. So, for instance, if I downloaded the human epd background (`epd_homo_sapiens_499_c` this file should be saved as `/usr/share/gimmemotifs/MotifSampler/hg18.bg`.

3.3.7 Other configuration options

All of GimmeMotifs configuration is stored in `/usr/share/gimmemotifs/gimmemotifs.cfg` or `~/.gimmemotifs.cf` If the file `~/.gimmemotifs.cfg` exists in your home directory this will always have precedence over the system-wide configuration. The configuraton file is created at installation time with all defaults set, but you can always edit it afterwards. It contains two sections `main` and `params` that take care of paths, file locations, parameter settings etc. Additionally, every motif tool has it's own section. Let's have a look at the options.

```
[main]
index_dir = /usr/share/gimmemotifs/genome_index
template_dir = /usr/share/gimmemotifs/templates
seqlogo = /usr/local/bin/seqlogo
score_dir = /usr/share/gimmemotifs/score_dists
motif_databases = /usr/share/gimmemotifs/motif_databases
gene_dir = /usr/share/gimmemotifs/genes
tools_dir = /usr/share/gimmemotifs/tools
```

- `index_dir` The location of the indeces of the genome fasta-files.
- `template_dir` The location of the KID html templates, used to generate the reports.
- `seqlogo` The seqlogo executable.
- `score_dir` To generate p-values, a pre-calculated file with mean and sd of score distributions is needed. These are located here.
- `motif_databases` For now contains only the JASPAR motifs.
- `gene_dir` Directory with bed-files containing gene locations for every indexed organism. This is needed to create the matched genomic background.
- `tools_dir` Here all tools included with GimmeMotifs are stored.

```
[params]
background = genomic_matched,random
use_strand = False
tools = MDmodule,Weeder,MotifSampler
analysis = medium
pvalue = 0.001
width = 200
fraction = 0.2
genome = hg18
lwidth = 500
cluster_threshold = 0.95
available_tools = Weeder,MDmodule,MotifSampler,gadem,meme,trawler
abs_max = 1000
```

```
enrichment = 1.5
max_time = None
```

This section specifies all the default GimmeMotifs parameters. Most of these can also be specified at the command-line when running GimmeMotifs, in which case they will override the parameters specified here.

4 Usage

4.1 Quick GimmeMotifs example

You can try GimmeMotifs with a small example dataset included in the examples directory, included with GimmeMotifs. This example does not require any additional configuration if GimmeMotifs is installed correctly.

Change to a directory where you have write permissions and run the following command (substitute the filename with the location of the file on your system):

```
gimme_motifs.py -i /usr/share/gimmemotifs/examples/TAp73alpha.fa -n p73
```

The `-n` or `--name` option defines the name of the output directory that is created. All output files are stored in this directory.

Depending on your computer you may have to wait some minutes for your results. Once GimmeMotifs is finished you can open `p73/p73_motif_report.html` in your browser.

4.2 GimmeMotifs example

This example is the same as above, except it will start from a BED file. This example does require you to have hg18 present and indexed. Change to a directory where you have write permissions and run the following command (substitute the filename with the location of the file on your system):

```
gimme_motifs.py -i /usr/share/gimmemotifs/examples/TAp73alpha.bed -n example
```

The `-n` or `--name` option defines the name of the output directory that is created. All output files are stored in this directory.

Depending on your computer you may have to wait some minutes for your results. Once GimmeMotifs is finished you can open `example/example_motif_report.html` in your browser.

4.3 Using GimmeMotifs: best practices and tips

4.3.1 GimmeMotifs is multi-threaded

GimmeMotifs runs multi-threaded and uses all the CPU's in the system. This means that all the programs will be run in parallel as much as possible. Of course some programs are still single-threaded, and will not benefit from this. Because GimmeMotifs uses all the available CPU's it does not make much sense to start multiple GimmeMotifs jobs at the same time.

4.3.2 Running time

The running time of GimmeMotifs greatly depends on which tools you use for prediction and how large the dataset is. Some of the tools might take a very long time and two of them, GADEM and MoAn, are not added to the default tools because of this reason. You can always use them for an analysis (by specifying the `-t` command-line option), but it is recommended to only do this for a small dataset (say, less than 5000 peaks). Weeder in combination with the `xl` analysis can also take a very long time, so be prepared. In general a `small` analysis will be the quickest, and a `xl` analysis will be the slowest.

While GimmeMotifs is developed specifically for ChIP-seq datasets, most motif prediction tools are not. In practice this means that it does not make much sense to predict motifs on a large amount of sequences, as this will usually not result in higher quality motifs. Therefore GimmeMotifs uses an absolute limit for the prediction set. By default 20% of the sequences are used as input for motif prediction, but with an absolute maximum. This is controlled by the `abs_max` parameter in the configuration file, which is set to 1000 by default. In general, if you have a large amount of peaks, you can also consider to run GimmeMotifs on the top sequences of your input, for instance the 5000 highest peaks.

There are two options that you can use to control the running time of GimmeMotifs. First, you can set an absolute time limit with the `max_time` option. This option (in hours) determines the maximum time used for motif prediction. If some programs take longer, the running jobs will be terminated, and the program will continue with all the motifs that have been predicted so far. The other option is kind of an emergency button: when you think that GimmeMotifs has been running long enough, you can press Ctrl+C **once, and only once!**. This will signal GimmeMotifs to terminate the running jobs and continue with the analysis. Please note that this works almost always, but still, there is a small chance that program might be in a function where the Ctrl-C option screws up, and GimmeMotifs will not be able to handle the result gracefully.

4.3.3 Intermediate results

GimmeMotifs produces a lot of intermediate results, such as all predicted motifs, fasta-files used for validation and so on. These are deleted by default (as they can get quite large), but if you are interested in them, you can specify the `-k` option.

4.3.4 Running on FASTA files

It is also possible to run GimmeMotifs on a FASTA file as input instead of a BED file. This is detected automatically if you're inputfile correctly formatted according to FASTA specifications. In this case it is not possible to generate a genomic matched background, so only the random Markov background will be used. Please note that for best results, all the sequences should be of the same length. This is not necessary for motif prediction, but the statistics and positional preference plots will be wrong if sequences have different lengths. Also see the next section.

4.3.5 Small input sets

Keep in mind that GimmeMotifs is developed for larger datasets, where you have the luxury to use a large fraction of your input for validation. So, at least several hundred sequences would be optimal. If you want to run GimmeMotifs on a small input dataset, it might be worthwile to increase the fraction used for prediction (with the `-f` parameter.

4.4 Detailed options

- `-i` or `--inputfile`

  This is the only mandatory option. The inputfile needs to be in BED or FASTA format. BED-fomatted files need to contain at least three tab-seperated columns describing chromosome name, start and end. The fourth column is optional, if specified it will be used by MDmodule to sort the features before motif prediction. GimmeMotifs will take the center of these features, and subsequently extend those to the width specified by the `width` parameter (see below).

- `-n` or `--name`

  The name of your analysis. All outputfiles will be stored in a directory named as given by this parameter. By default this will be gimmemotifs_dd_mm_yyyy, where d,m and y are the current day, month and year respectively.

- `-a` or `--analysis`

  The size of motifs to look for: small (5-8), medium (5-12), large (6-15) or xl (6-20). The larger the motifs, the longer GimmeMotifs will run. The 'xl' can take a very long time!

- `-g` or `--genome`

  Name of the genome (index) to use. For instance, for the example in section 3.3.2 this would be `hg18`.

- `-s` or `--singlestrand`

  Only use the + strand for prediction (off by default).

- `-f` or `--fraction`

  This parameter controls the fraction of the sequences used for prediction. This 0.2 by default, so in this case a randomly chosen 20% of the sequences will be used for prediction. The remaining sequences will be used for validation (enrichment, ROC curves etc.). If you have a large set of sequences (ie. most ChIP-seq peak sets), this is fine. However, if your set is smaller, it might be worthwile to increase this prediction fraction.

- `-w` or `--width`

  This is the width of the sequences used for motif prediction. Smaller sequences will result in a faster analysis, but you are of course limited by the accuracy of your data. For the tested ChIP-seq data sets 200 performs fine.

- `-e` or `--enrichment`

  All motifs should have an absolute enrichment of at least this parameter compared to background to be called significant.

- `-p` or `--pvalue`

  All motifs should have a pvalue of at most this parameter (hypergeometric enrichment compared to background) to be called significant.

- `-b` or `--background`

  Type of background to use. By default `random` (1st order Markov model, similar dinucleotide frequencies as your sequences) and `matched_genomic` (randomly chosen from the genome with a similar distribution respective to the TSS of genes) are used.

- `-l` or `--localization_width`

  Width used in the positional preference plots.

- `-t` or `--tools`

  A comma-seperated list of all the motif prediction tools to use. By default all installed tools that are specified in the GimmeMotifs configuration file are used.

- `--max_time`

  Time limit for motif prediction in hours. Use this to control the maximum number of hours that GimmeMotifs uses for motif prediction. After this time, all jobs that are still running will be terminated, and GimmeMotifs will continue with the motifs that are predicted so far.

5 Other scripts

In addition to `gimme_motifs.py` the GimmeMotifs package contains several other tools that can perform the various substeps of GimmeMotifs, as well as other useful tools. Run them to see the options.

5.1 Input formats

Most tools in this section take a file in PWM format as input. This is actually a file with Position Specific Scoring Matrices (PSSMs) containing *frequencies*. It looks like this:

```
>motif1
0.3611 0.0769 0.4003 0.1664
0.2716 0.0283 0.5667 0.1381
0.6358 0.0016 0.3344 0.0330
0.0016 0.9859 0.0016 0.0157
0.8085 0.0063 0.0502 0.1397
>motif2
```

```
0.2276 0.0157 0.0330 0.7284
0.0031 0.0016 0.9984 0.0016
0.0377 0.3799 0.0016 0.5856
0.0816 0.7096 0.0173 0.1962
0.1350 0.4035 0.0675 0.3987
```

The frequencies are seperated by tabs, and in the order A,C,G,T.

5.2 Descriptions

closest_motif_match.py

Taking an input file with motifs, find the best matching file in another file of motifs (according to the WIC metric).

create_genome_index.py

Creates an index to use with GimmeMotifs. See section 3.3 for details.

generate_background_sequences.py

Generate random sequences according to one of two methods: random or matched_genomic. With the argument `type` set to `random`, and an input file in FASTA format, this script will generate sequences with the same dinucleotide distribution as the input sequences according to a 1st order Markov model trained on the input sequences. The `-n` options is set to 10 by default. The length distribution of the sequences in the output file will be similar as the inputfile. The Markov model can be changed with option `-m`. If the `type` is specified as `matched_genomic` the inputfile needs to be in BED format, and the script will select genomic regions with a similar distribution relative to the transcription start of genes as the input file. Make sure to select the correct genome. The length of the sequences in the output file will be set to the median of the features in the input file.

motif_cluster.py

Cluster a set of motifs with the WIC metric.

motif_localization_plots.py

Create the positional preference plots for all the motifs in the input PWM file. This will give best results if all the sequences in the FASTA-formatted inputfile have the same length. Keep in mind that this only makes sense if the sequences are centered around a similar feature (transcription start site, highest point in a peak, etc.). The default threshold for motif scanning is 0.95, see `pwmscan.py` for more details.

motif_roc.py

Given a sample (positives, peaks) and a background file (random sequences, random promoters or similar), creates a ROC plot for all the motifs in an input PWM file. All the motifs will be plotted in the same graph, you can select one or more specific motifs to plot with the `-i` option.

motif_roc_metrics.py

Given a sample (positives, peaks) and a background file (random sequences, random promoters or similar), calculated several statistics for all the motifs in an input PWM file. You can select one or more specific motifs with the `-i` option. The ROC area under curve (ROC_AUC), Mean Normalized Conditional Probability (MNCP) and maximum f-measure are reported for each motif.

pwm2logo.py

Convert the motifs in a PWM file to a logo using weblogo.

pwmscan.py

Scan a set of sequences with a set of motifs, and give the resulting matches in GFF or BED format. The threshold is based on the maximum and minimum possible score for each motif. So, 0.95 means that the score of a motif should be at least 95% of the (maximum score - minimum score). This should probably not be set much lower than 0.8, and should be generally at least 0.9 for good specificity. Keep in mind that the optimal threshold might be different for each motif!

track2fasta.py

Convert a set of BED formatted sequences to a FASTA file. The genome needs to be indexed for GimmeMotifs using `create_genome_index.py`.

6 Acknowledgements

We are grateful to Waseem Akhtar, Robert Akkers, Max Koeppel, Evelyn Kouwenhoven, Marion Lohrum, Leonie Smeenk and Jo Zhou for providing data and feedback during GimmeMotifs development. Also we would like to thank Stefanie Bartels, Adalberto Costessi, Joost Martens and Nagesha Rao for testing and helpful discussion. Of course GimmeMotifs by itself wouldn't be able to do anything, if there wasn't such a number of excellent tools available. Therefore, a big thanks to all the authors of the motif prediction programs for making their software publicly available and allowing me to distribute them with GimmeMotifs! In addition, I would like to thank Wolfgang Lugmayr and Aaron Statham for various bugfixes.

References

[1] Simon J van Heeringen and Gert Jan C Veenstra. GimmeMotifs: a de novo motif prediction pipeline for ChIP-sequencing experiments. *Bioinformatics (Oxford, England)*, November 2010.

ISSN 1367-4811. doi: 10.1093/bioinformatics/btq636. URL http://www.ncbi.nlm.nih.gov/pubmed/21081511. PMID: 21081511.

[2] X Liu, D L Brutlag, and J S Liu. BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 127–138, 2001. ISSN 1793-5091. URL http://www.ncbi.nlm.nih.gov.proxy.ubn.ru.nl:8080/pubmed/11262934. PMID: 11262934.

[3] Leping Li. GADEM: a genetic algorithm guided formation of spaced dyads coupled with an EM algorithm for motif discovery. *Journal of computational biology : a journal of computational molecular cell biology*, 16(2):317–329, February 2009. ISSN 1066-5277. doi: 10.1089/cmb.2008.16TT. PMID: 19193149 PMCID: 2756050.

[4] Wanyuan Ao, Jeb Gaudet, W. James Kent, Srikanth Muttumu, and Susan E. Mango. Environmentally induced foregut remodeling by PHA-4/FoxA and DAF-12/NHR. *Science*, 305(5691): 1743–1746, September 2004. doi: 10.1126/science.1102216. URL http://www.sciencemag.org.proxy.ubn.ru.nl:8080/cgi/content/abstract/305/5691/1743.

[5] X. Shirley Liu, Douglas L. Brutlag, and Jun S. Liu. An algorithm for finding protein-DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments. *Nat Biotech*, 20(8):835–839, 2002. ISSN 1087-0156. doi: 10.1038/nbt717. URL http://dx.doi.org/10.1038/nbt717.

[6] Timothy L. Bailey, Mikael Boden, Fabian A. Buske, Martin Frith, Charles E. Grant, Luca Clementi, Jingyuan Ren, Wilfred W. Li, and William S. Noble. MEME SUITE: tools for motif discovery and searching. *Nucl. Acids Res.*, 37(suppl_2):W202–208, July 2009. doi: 10.1093/nar/gkp335. URL http://nar.oxfordjournals.org/cgi/content/abstract/37/suppl_2/W202.

[7] Eivind Valen, Albin Sandelin, Ole Winther, and Anders Krogh. Discovery of regulatory elements is improved by a discriminatory approach. *PLoS Comput Biol*, 5(11):e1000562, November 2009. doi: 10.1371/journal.pcbi.1000562. URL http://dx.doi.org/10.1371/journal.pcbi.1000562.

[8] G Thijs, M Lescot, K Marchal, S Rombauts, B De Moor, P Rouz, and Y Moreau. A higher-order background model improves the detection of promoter regulatory elements by gibbs sampling. *Bioinformatics (Oxford, England)*, 17(12):1113–1122, December 2001. ISSN 1367-4803. URL http://www.ncbi.nlm.nih.gov/pubmed/11751219. PMID: 11751219.

[9] Gert Thijs, Kathleen Marchal, Magali Lescot, Stephane Rombauts, Bart De Moor, Pierre Rouz, and Yves Moreau. A gibbs sampling method to detect overrepresented motifs in the upstream regions of coexpressed genes. *Journal of Computational Biology: A Journal of Computational Molecular Cell Biology*, 9(2):447–464, 2002. ISSN 1066-5277. doi: 10.1089/10665270252935566. URL http://www.ncbi.nlm.nih.gov.proxy.ubn.ru.nl:8080/pubmed/12015892. PMID: 12015892.

[10] Laurence Ettwiller, Benedict Paten, Mirana Ramialison, Ewan Birney, and Joachim Wittbrodt. Trawler: de novo regulatory motif discovery pipeline for chromatin immunoprecipitation. *Nat*

Meth, 4(7):563–565, July 2007. ISSN 1548-7091. doi: 10.1038/nmeth1061. URL `http://dx.doi.org/10.1038/nmeth1061`.

[11] Giulio Pavesi, Paolo Mereghetti, Giancarlo Mauri, and Graziano Pesole. Weeder web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. *Nucl. Acids Res.*, 32(suppl_2):W199–203, July 2004. doi: 10.1093/nar/gkh465. URL `http://nar.oxfordjournals.org.proxy.ubn.ru.nl:8080/cgi/content/abstract/32/suppl_2/W199`.

[12] Brooke Rhead, Donna Karolchik, Robert M. Kuhn, Angie S. Hinrichs, Ann S. Zweig, Pauline A. Fujita, Mark Diekhans, Kayla E. Smith, Kate R. Rosenbloom, Brian J. Raney, Andy Pohl, Michael Pheasant, Laurence R. Meyer, Katrina Learned, Fan Hsu, Jennifer Hillman-Jackson, Rachel A. Harte, Belinda Giardine, Timothy R. Dreszer, Hiram Clawson, Galt P. Barber, David Haussler, and W. James Kent. The UCSC genome browser database: update 2010. *Nucl. Acids Res.*, 38(suppl_1):D613–619, January 2010. doi: 10.1093/nar/gkp939.