Appendix A - iMotifs

Motivation

[1] Short sequence motifs are an important class of models in molecular biology, used most commonly for describing transcription factor binding site specificity patterns. High-throughput methods have been recently developed for detecting regulatory factor binding sites *in vivo* and *in vitro* and consequently high-quality binding site motif data are becoming available for increasing number of organisms and regulatory factors. Development of intuitive tools for the study of sequence motifs is therefore important.

iMotifs is a graphical motif analysis environment that allows visualisation of annotated sequence motifs and scored motif hits in sequences. It also offers motif inference with the sensitive NestedMICA algorithm, as well as overrepresentation and pairwise motif matching capabilities. All of the analysis functionality is provided without the need to convert between file formats or learn different command line interfaces.

The application includes a bundled and graphically integrated version of the NestedMICA motif inference suite that has no outside dependencies. Problems associated with local deployment of software are therefore avoided.

[1] The following manuscript is published in Piipari et al. (2010b) and is a result of collaborative work between the author of this thesis (MP), Dr Thomas Down (TD) and my PhD thesis supervisor Dr Tim Hubbard. The authors' contributions are as follows: MP conceived the work, wrote the software and the manuscript. TD and TH provided feedback. All authors read the manuscript and provided feedback.

Availability

iMotifs is licensed with the GNU Lesser General Public License v2.0 (LGPL 2.0). The software and its source is available at `http://wiki.github.com/mz2/imotifs` and can be run on Mac OS X Leopard (Intel/PowerPC). I also provide a cross-platform (Linux, OS X, Windows) LGPL 2.0 licensed library `libxms` for the Perl, Ruby, R and Objective-C programming languages for input, output of XMS formatted annotated sequence motif set files.

Introduction

Until recent years, studying sequence specificity of transcription factors systematically has been limited to a relatively small number of organisms and transcription factors. High throughput protein-DNA interaction assays such as protein binding microarrays (Berger et al., 2006), bacterial one-hybrid screens (Meng et al., 2005), large ChIP-chip studies and advances in motif inference algorithms and tools has however caused an expansion of motif databases such as UNI-PROBE (Newburger and Bulyk, 2009), TRANSFAC (Matys et al., 2006) and JASPAR (Bryne et al., 2008).

Sequence motif analysis tools can be hard to deploy and use locally. Many commonly used software packages have therefore been made available as web applications (Mahony and Benos, 2007; Thomas-Chollier et al., 2008). Public servers can however be limited in the CPU time given to users which can rule out their use for large scale studies. Data exchange and usability can also be a challenge. Therefore I have created an OS X based desktop software package for sequence motif analysis that is easy to install and update. Compared to previously published desktop based *cis*-regulatory sequence analysis tools such as TOUCAN (Aerts et al., 2003) or Sockeye (Montgomery et al., 2004), iMotifs is more focused on visualisation and computation of sequence motifs, although it also supports visualising scored motif matches in sequences.

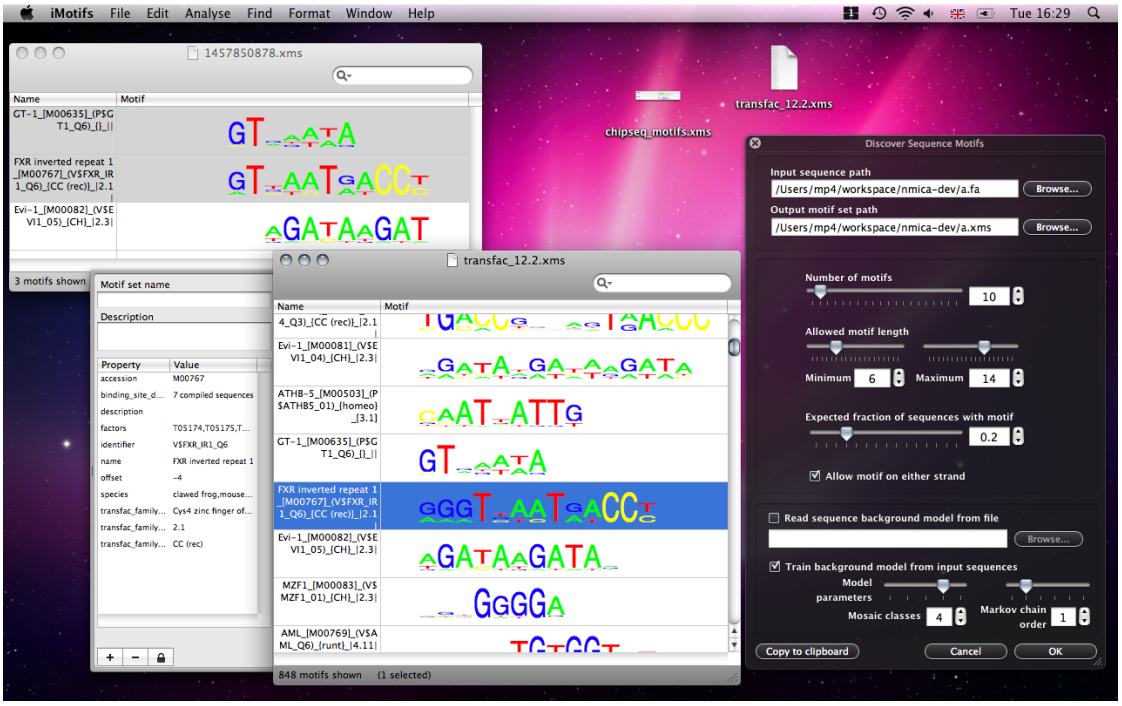

Figure A1: iMotifs can present motif sets and alignments. It integrates with the OS X desktop's previewing functionality and includes a number of analysis tools including an integrated NestedMICA motif inference tool.

Features

iMotifs is designed for visualisation and analysis of *cis*-regulatory motifs and sequences. It can be used to retrieve sequences (for example for a coregulated group of genes), infer *cis*-regulatory motifs from them and score sequences with motif models, visualise them and their scored matches, and compare them against other motifs (Fig. 1 shows the core functionality). A tutorial is included on the website for common tasks (see Availability). Motifs can be manipulated and moved between sets by dragging and dropping, and filtered using keyword searches. Summary statistics such as entropy, column count or distance from closest pair can also be shown alongside. Free form key-value pair metadata such as database identifiers, species or notes can be viewed and edited. PDF export and printing is available. Import and export of TRANSFAC formatted motif files is also possible.

iMotifs can be used to retrieve sequences from the Ensembl database (Hubbard et al., 2009). The retrieved sequences can be aligned either to transcription start sites (putative promoter sequence) or ends (e.g. for micro-RNA seed finding), and they can be filtered by gene identifiers. The retrieval tool can fetch specific sequence regions using GFF formatted annotation files, and includes specific support for ranking and retrieving regions of interest based on ChIP-seq 'peaks': MACS (Zhang et al., 2008a), FindPeaks (Fejes et al., 2008) and SWEMBL formats are supported. Sequences are optionally processed to mask repeats and translated sequence.

iMotifs supports the quick previewing and thumbnailing service native to OS X (QuickLook). Previewing is especially useful for browsing sequence motif sets stored remotely (e.g. on a remote cluster) as no manual transfer or file opening is needed. An automated software update mechanism is included.

Many common motif analysis tasks are supported. These include finding closest matching and reciprocally matching motif pairs between two motif sets with the distance metric and algorithm described in Down et al. (2007). Motif multiple alignments can be visualised and computed with a greedy gapless motif multiple alignment algorithm. Motif inference experiments can be run with the integrated NestedMICA (Down and Hubbard, 2005) tool simply by dragging FASTA for-

matted sequence files to iMotifs. Downstream analyses such as motif scanning, overrepresentation analysis, and motif hit score cutoff assignment as described in Down et al. (2007) is also possible. Analysis tasks are run in parallel without blocking the user interacting with the application.

Interoperability

Although iMotifs itself works only on computers running Mac OS X, the analysis tools developed for and included in iMotifs are cross-plaform (Java based) and depend only on libraries included with the package. Most analysis functions are implemented by stand-alone command-line programs. This makes it possible to rapidly integrate unmodified tools into iMotifs. The included analysis tools can also be run on any UNIX system without iMotifs.

I feel that the use of a standard format for exchanging sequence motif data is beneficial for the research community, given the literally hundreds of motif inference tools and databases that are available (reviewed in Das and Dai (2007)). To encourage the take up of a standard file format for motifs, I provide a programming interface for the input and output of the annotated motif file format XMS for the Perl, Ruby, R and Objective-C languages. The Perl and R libraries can also be used to visualise sequence logos.

Conclusions

I have created an integrated desktop application for short sequence motif analysis. It incorporates visualisation, inference, alignment and comparison tools. The application widens the user base of sequence motif analysis tools and can improve the productivity of researchers working with sequence motif data. I aim to integrate with more sequence motif analysis tools and web services and to develop further the already included basic protein motif visualisation and inference support.

I also encourage the introduction of a standard format for exchange of sequence motif data by providing conversion utilities and an API for input and

output of XMS motif set files for a number of common bioinformatics programming languages.

Appendix B - The motif inference tutorial

Introduction

[1]The tutorial below is aimed to introduce a researcher new to regulatory genomics to taking use of the NestedMICA and NMICA-extra motif inferecene tools to identify and analyze sequence motifs from noncoding genomic sequence. We demonstrate uses of the NMICA-extra package with a short sequence analysis project where NestedMICA is first used to recreate the STAT1 transcription factor binding motif from Robertson et al. (2007).

The first step is retrieving input genomic sequences corresponding to the ChIP-seq peak regions. To ease the retrieval and importantly preprocessing of input sequence (repeat masking and exclusion of translated sequences), NestedMICA has been enhanced with a number of tools for retrieving sequence from the Ensembl database (Flicek et al. 2008): `nmensemblseq`, `nmensemblfeat` and `nmensemblpeakseq`.

1. `nmensemblseq`: retrieves sequences around transcription start sites or 3 UTRs or introns.

[1]The following manuscript is a result of collaboration between the author of this thesis (MP), Dr. Thomas Down (TD), and MP's thesis supervisor Dr. Tim Hubbard (TH). The work is published in (Piipari et al., 2011). Authors' contributions are as follows: MP wrote the manuscript, all authors read it and provided feedback.

2. `nmensemblfeat`: retrieves specific sequence regions using GFF formatted annotation files as input.

3. `nmensemblpeakseq`: retrieves sequence regions close to ChIP-seq peaks

   - MACS (Zhang et al., 2008b)
   - FindPeaks (Fejes et al., 2008)
   - SWEMBL (`http://www.ebi.ac.uk/~swilder/SWEMBL/`)

Two more generic sequence feature formats are also supported:

1. BED (`https://cgwb.nci.nih.gov/goldenPath/help/customTrack.html`)
2. GFF (`http://www.sanger.ac.uk/resources/software/gff/spec.html`)

We will use `nmensemblpeakseq` to retrieve sequence windows corresponding to 50 base long sequence windows around ranked ChIP-sequencing peak maximum positions of the 500 top-ranking peaks.

```
nmensemblpeakseq -database homo_sapiens_core_52_36n \
-host ensembldb.ensembl.org \
-user anonymous -port 5306 \
-inputFormat peaks \
-peaks STAT1_IFNGstim_hg18_xset200_dupsN_ht10.sub.peaks \
-maxCount 500 \
-aroundPeak 50 \
-minLength 50 \
-minNonN 80 \
-repeatMask \
-excludeTranslations \
-chunkLength 100 > stat1-stimulated-50bp-around-max.fasta
```

The regions included in the dataset have been mapped to the NCBI36 human genome assembly (Ensembl release 52). We therefore request sequences relative to the same release of the Ensembl database (`homo_sapiens_core_52_36n`). The reason for choosing the database, hostname and port combination above is that at the time of writing the publicly available Ensembl instance that serves the Ensembl release 52 is the port 5306 on `ensembldb.ensembl.org`.

Sequence background model estimation

Before motif inference from the retrieved sequences, it is advisable to estimate a NestedMICA sequence background model as a separate step. This can be done with the command `nmmakebg`, which requires two input parameters: Markov chain order and the number of mosaic classes. The Markov chain parameter is usually set to 1st order because some of the DNA motif specific downstream analysis tools require this. The class count parameter that yields best performance tends to be 4 (Down and Hubbard, 2005), but it is best to evaluate different mosaic class parameters before the potentially long-running motif inference analysis. Background models can be evaluated using the command nmevaluatebg

```
nmevaluatebg -order 1 \
-minClasses 1 -maxClasses 8 \
-seqs stat1-stimulated-500bp-around-max.fasta \
-testSeqs stat1-stimulated.fasta \
> min1classes-max8classes-eval-bg.eval
```

The output of `nmevaluatebg` can be used to find the mosaic order parameters at which the background model performance, as measured by sequence likelihood given the background model, shows little increase or drops. These parameter values are then taken as the optimal ones. The easiest way to interpret the results is to plot them using R with the `nmica` R package (http://github.com/mz2/r-utilities).

```
>library(nmica)
>eval.results <-
        read.nmevaluatebg(
                min1classes-max8classes-eval-bg.eval )
>plot(eval.results$classes ~ eval.results$likelihood)
```

This evaluation (Figure A2) suggest a suitable order parameter as 4. We can now commence with the background model estimation:

```
nmmakebg -classes 4 -order 1 \
-seqs stat1-stimulated-500bp-around-max.fasta \
-out seqs-4classes-1storder.bg
```

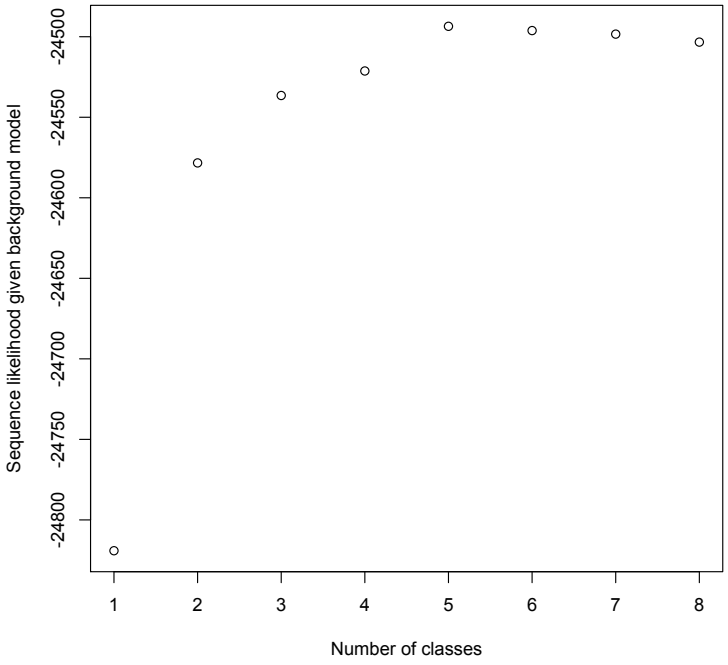

Figure A2: Output of the nmevaluatebg command plotted in R.

Motif inference

After retrieving the input sequences and determining class and order parameters with `nmmakebg`, we can now run the NestedMICA motif inference with the command `nminfer`.

```
nminfer -seqs stat1-stimulated.fasta \
-numMotifs 1 \
-backgroundModel stat1-stimulated-4classes.bg \
-minLength 6 -maxLength 14 \
-minSeqLength 50 \
-maxCycles 1000000 \
-revComp \
-expectedUsageFraction 0.70 \
-checkpoint stat1-stimulated-checkpoint \
-sampleFile stat1-stimulated-sample \
-sampleInterval 10000 \
-checkpointInterval 10000 \
-logInterval 100 \
-distributed -port 5001 -threads 4 \
-out motifs.xms > nminfer.log 2> nminfer.err
```

Note that the above command line instructs periodic output of checkpoint files that can be used to restart the computation, as well as sample motif set files (preliminary motif set solutions that can visualised whilst the computation is still running). The above `nminfer` command line also demonstrates distributed computing with NestedMICA: the `-distributed` and `-port 5001` instruct `nminfer` to act as a server that responds at port 5001 to distribute its work load to separate worker nodes (each of which would typically correspond to one computer in a computational cluster). Worker nodes that connect to a server can be created with the command `nmworker`.

```
nmworker -server nmica_server_hostname -port 5001 -threads 4
```

The actual host name given above depends on the host name of the computer where `nminfer` was set to run.

Motif overrepresentation

When interpreting the output of NestedMICA, it is important to note that the algorithm does not rank its output motifs relative to each other or predict hit positions for them. A common way of assessing computationally inferred motifs is through a motif overrepresentation analysis. By overrepresentation analysis we mean a statistical exercise where sequences with the motif (the positive set) are discriminated from those assumed to be devoid of it (the negative set). The approach taken in NMICA-extra for computing the degree of overrepresentation in a set of sequences is the ROC-AUC (Receiver-Operator Characteristic Area Under the Curve) statistic, computed with the tool `nmrocauc`. In short, sequences are labelled as positive or negative and the maximum motif bit score is used to predict if any given sequence is part of the positive or the negative sequence set – the maximum motif hit score is used to classify the sequences. The AUC statistic that is reported by this analysis is a measure of how often a randomly chosen positive sequence is ranked above a randomly chosen negative sequence. It therefore provides a measure of separation of maximum motif hit score distribution of the positive examples from the negative examples. To estimate the null distribution of scores with the length distribution and sequence composition used, the negative sequences are shuffled and the randomly generated sequences are then scored according to the same criterion. The shuffling conducted as part of this method accounts for the fact that the maximum hit score distributions of sequences can vary based on nucleotide composition.

```
#Retrieve 1000 random core promoter sequences:
#900bp upstream of TSS and up to 100bp downstream
#Exclude any repeats and translated sequence
nmensemblseq \
-sampleRandomGenes 1000 \
-fivePrimeUTR 900 100 \
-proteinCoding \
-repeatMask \
-excludeTranslations \
-database homo_sapiens_core_52_36n \
```

```
-host ensembldb.ensembl.org \
-port 5306 \
-user anonymous > \
1000-random-human-promoters_900bp-upstream-100bp5utr.fasta

#Sample 1000 random sequences of length 50
#The sequence window length
#is the same as that of the peak sequence windows
nmrandomseq \
-count 500 \
-length 50 \
-seqs 500-random-human-promoters_900bp-upstream-100bp5utr.fasta > \
100bp-windows-from-random-human-promoters.fasta

nmrocauc \
-positiveSeqs stat1_chip_peaks.fasta \
-negativeSeqs \
50bp-windows-from-random-human-promoters.fasta \
-motifs stat1_human.xms
#Output:
#motif2   0.992880          0.00000
```

The above analysis shows that the discovered motif is strongly over-represented in the ChIP-sequencing peaks when compared to random noncoding sequence regions of the same genome (the empirical p-value, which is the second value in the `nmrocauc` output, is below 10^{-5}.

The STAT transcription factors are a well studied family of transcription factors and DNA binding motif have therefore been deposited to publicly available databases such as TRANSFAC (Matys et al., 2006). This makes it possible to validate the sequence motif we have inferred from the ChIP-seq data with NestedMICA by searching it against motif databases with the reciprocal matching procedure described above. Reciprocal matching of motifs is implemented in the tool nmshuffle that is distributed as part of NestedMICA.

```
nmshuffle -bootstraps 100000 \
transfac_12.2.xms stat1-human.xms
#Output:
#motif0   STAT5A_[M00457]   0.531520        0        0.00000
```

A statistically significant match is identified for the NestedMICA STAT1 motif in the TRANSFAC database (the empirical p-value which is the last column in the nmshuffle output above, is below 10^{-5}). An inspection of the closest matching motifs makes it clear that NestedMICA infers a very similar binding specificity pattern for STAT1 as has been previously reported for members of the STAT family transcription factors (Figure A3).

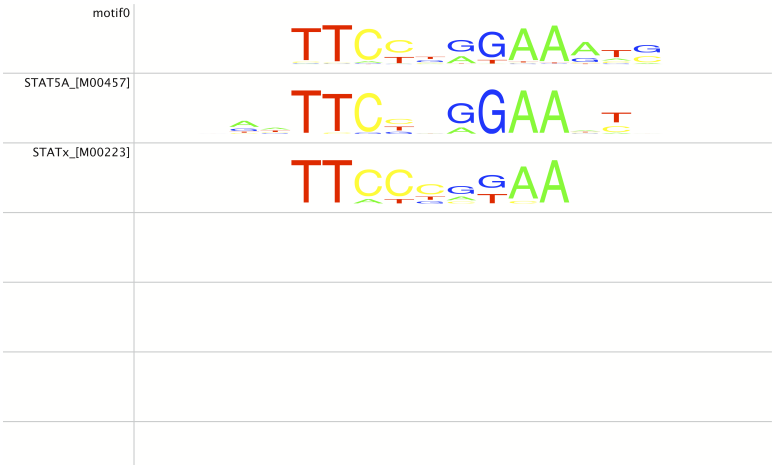

Figure A3: The predicted motif alongside known STAT motifs from the TRANSFAC database.

Appendix C - Motif inference algorithm assessment parameters

The parameters given for each of the motif inference methods tested in Chapter 5 are given below.

NestedMICA

The NestedMICA algorithm was run with the following parameters:

```
nminfer -numMotifs 200 \
-minLength 6 -maxLength 14 \
-expectedUsageFraction 0.2 \
-backgroundModel sc_4classes_1order.bg \
-seqs orthologs-sc-1000.fa
```

Sequence background model parameters were evaluated with `nmevaluatebg` using a randomly chosen half of the input sequence for model learning (`-trainSeqs`) and the remaining half for model evaluation (`-testSeqs`). As suggested in the NestedMICA manual, the Markov chain order was kept constant at 1 (`-order 1`) and the mosaic class parameter was varied between 1 and 8 (`-minClasses 1 -maxClasses 8`). The sequence likelihood values achieved with each of these parameter settings are shown in Figure A4.

Mosaic class count 4 was chosen based on the above evaluation because it presents an acceptable compromise between a descriptiveness and complexity of

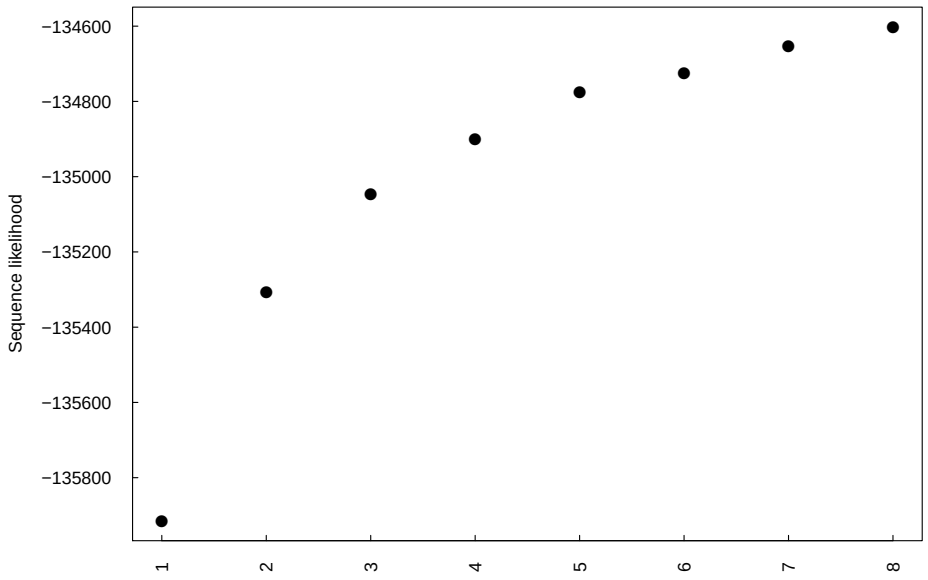

Figure A4: Evaluation of sequence background model class counts at Markov chain order 1.

the model; Increasing the class count beyond 4 results in diminishing gains in the sequence likelihood. The runtime of the application also increases.

Weeder

The weeder algorithm (Pavesi et al., 2001) was run with the `weederlauncher.out` driver script distributed with the program. The 'large' settings were used to search for motifs between 6 and 12 nucleotides long, and motifs were allowed to be present on either strand:

```
weederlauncher.out orthologs-sc-1000.fa SC large S M T200
```

For all downstream analyses, the motif output by the program were trimmed to the top 200 reported motifs.

AlignACE

The parameters used for running AlignACE (Roth et al., 1998) are described below:

```
AlignACE -numCols 10 -gcback 0.38 -i orthologs-sc-1000.fa
```

The sequence background model used by AlignACE is a 0^{th} order Markov chain, simply parameterised by the overall GC content of the yeast genome (Goffeau et al., 1996). The motif length (number of columns) was set to 10. Length of 10 was chosen because it is the median motif length in the JASPAR motif database which the predicted motifs are primarily compared with.

MEME

MEME version 4.3.0 (Bailey et al., 2006) was run with the following parameters:

```
meme.bin orthologs-sc-1000.fa \
-dna -mod anr \
-nmotifs 100 -minw 6 -maxw 14 \
-bfile ~/meme_4.3.0/tests/common/yeast.nc.6.freq
```

The motifs were constrained to lengths between 6 and 14, similarly as done with NestedMICA. The background model used was the 6^{th} order Markov chain background model trained from *S. cerevisiae* intergenic sequences which is supplied with MEME 4.3.0 (motif finding with a 3^{rd} order background was also attempted). The sequence-motif model used was the "any number of repeats" model (`-mod anr`). Number of motifs was set to 100 – it was the largest number of motifs that MEME allows.

MotifSampler

MotifSampler (Thijs et al., 2001) was run with the following parameters:

```
MotifSampler -f orthologs-sc-1000.fa \
```

```
-b orthologs-sc-1000.motifsamplerbg \
-r 50 -s 1 -M 1 -n 50 -w 10 \
-o orthologs-sc-1000.motifsamplerout \
-m orthologs-sc-1000.motifs
```

The motif count parameter 50 (`-n 50`) was used because the program did not report motifs when large numbers of motifs were requested. The motif width 10 was chosen as it was the maximum allowed by the program, and the median motif length in the JASPAR database. Before the motif inference program was run, a 2^{nd} order background model was trained from the input sequences using the `CreateBackgroundModel` tool supplied with `MotifSampler`, with the following parameters:

```
CreateBackgroundModel \
-f ../orthologs-sc-1000.fa \
-b orthologs-sc-1000.motifsamplerbg \
-o 2 -n SC
```

YMF

YMF (Sinha and Tompa, 2003b) was run with the following parameters::

```
./stats stats.config 200 8 \
ymftables/yeast -sort orthologs-sc-1000.fa
```

Two hundred 8-mers were inferred, using the yeast background nucleotide frequencies from the table supplied with the program (`../ymftables/yeast`). The output of YMF was post-processed another program, FindExplanators (Blanchette and Sinha, 2001), which removes redundancy amongst the consensus strings, outputting supposedly independent motifs.

```
find_explanators \
ymftables/yeast_powersGeneralized.3.bin \
orthologs-sc-1000.fa stats/results 5
```

FindExplanators reported a single motif AAARNRAAA regardless of the final explanator motif count parameter, which was varied. An inspection of the YMF results, which were given to FindExplanators as input, shows that the YMF output indeed only contains consensus strings that closely fit either AAARNRAAA or its reverse complement TTTYNYTTT. An excerpt with the first ten motifs from the set of 200 are given below.

```
2 AAARNRAAA 1529 48.93 345.6754 584.8017
3 TTTYNTTTY 1582 48.37 365.7588 632.3148
4 AAAANRAAA 1223 48.17 242.6953 414.1873
5 AAAANAAAA 994 47.53 167.9777 302.0721
6 AAARNAAAA 1202 47.46 239.8017 411.0152
7 ARAANRAAA 1478 47.30 354.0071 564.6885
8 TTTTNTTTY 1258 47.03 253.1523 456.4886
9 TTYTNTTTY 1514 47.00 360.8600 602.0605
10 AAAANRRAA 1493 46.94 351.1163 591.8228
```

As one can see, motifs output by YMF with these parameters are a largely redundant set. I chose to still analyse these motifs alongside the other predictions further, to see how a highly redundant motif set would perform in my assessment.

SOMBRERO

SOMBRERO (Mahony et al., 2005b) was run with the following parameters:

```
SOMBRERO -t orthologs-sc-1000.fa \
-b /nfs/users/nfs_m/mp4/sombrero/yeast.back \
-lm 6 14 \
-time 200 \
-out results.sombrero
```

The 2^{nd} order sequence background model of the yeast genome was downloaded from http://bioinf.nuigalway.ie/sombrero/binaries/backgrounds.zip. The training iteration count was set to 1000 (ten times larger value than the default, to reflect the large nature of the problem). The minimum and maximum motif

lengths were set to 6 and 14 respectively. The program output was cut to 200 motifs by ranking motifs by the z-score which SOMBRERO reports.

Oligoanalysis

Oligo-analysis (Thomas-Chollier et al., 2008) was run with the web form included in the RSA Tools web server at `http://rsat.ulb.ac.be/rsat/oligo-analysis_form.cgi`, with the parameters shown in Figure A5, to discover a total of 50 over-represented sequence words.

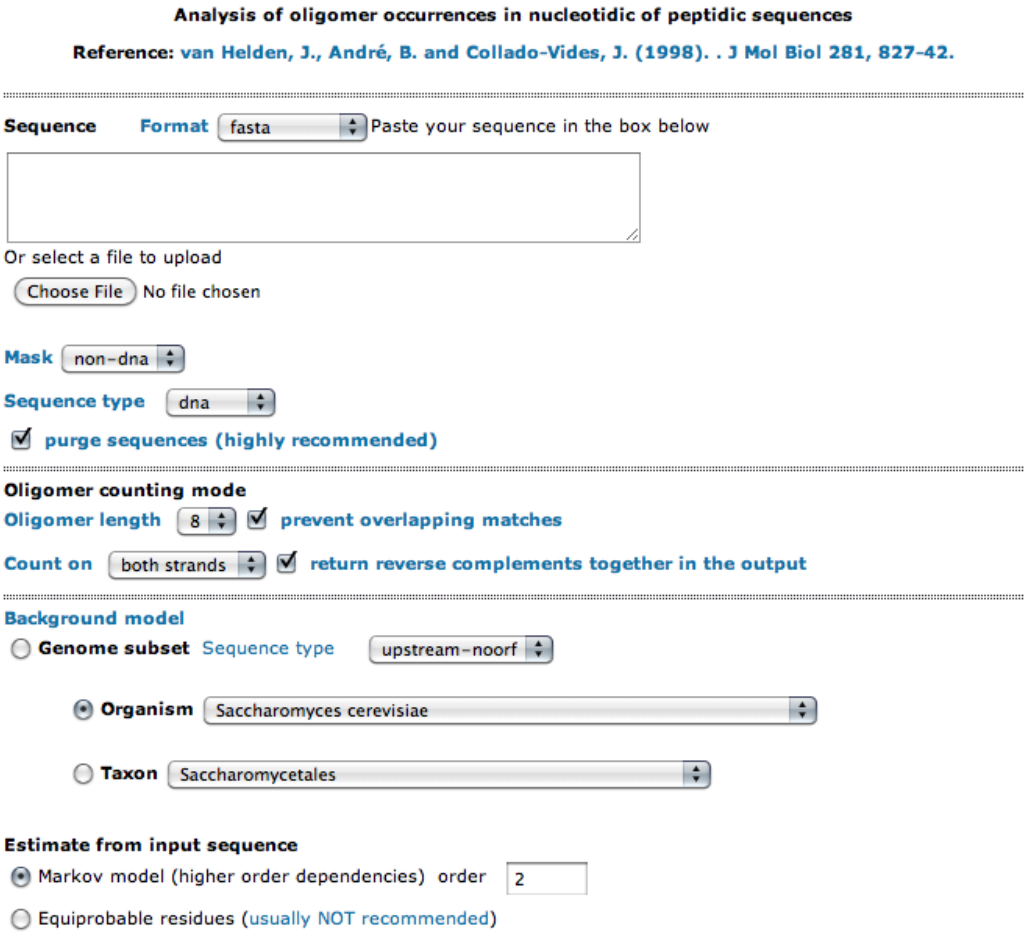

Figure A5: Parameter choices used with Oligo-analysis.