TFM-Explorer
user manual

Laurie Tonon

January 27, 2010

Contents

1 Introduction **3**
1.1 What is TFM-Explorer? 3
1.2 Versions 3
1.3 Licence 3
1.4 Publications 3
1.5 Aim of this document 4

2 Installation **5**
2.1 Pre-required 5
2.1.1 Linux platforms 5
2.1.2 Windows platforms 5
2.1.3 MacOSX platforms 6
2.2 Downloads 6
2.3 Optional compilation 6

3 Use **8**
3.1 Principle of TFM-Explorer 8
3.2 Available files 9
3.3 Command lines 10
3.3.1 fasta2dump 10
3.3.2 matrix2dump 11
3.3.3 tfmebg 12
3.3.4 tfme 13

4 FAQ **16**

5 Contact **17**

1 Introduction

1.1 What is TFM-Explorer?

TFM-Explorer is a program for analysing regulatory regions of eukaryotic genomes. It takes a set of coregulated gene sequences, and search for locally overrepresented transcription factor biding sites.
The algorithm proceeds in two steps:

- scans sequences for detecting all potential transcription factor binding sites, using position frequency matrices
- extracts significant clusters (region of the input sequences associated with a factor) by calculated a score function

The results of TFM-Explorer can be used to identify new regulatory mecanisms by searching for putative transcription factor biding sites and cis-regulatory modules.

1.2 Versions

version	release date	description
1.1	August 2006	The first version of TFM-Explorer, that allows to analyse sequences with weight matrices from JASPAR and TRANSFAC and output the results on the screen.
2.0	January 2010	The current version of TFM-Explorer, that allows to use personal weight matrices, automatically calculates pairwise correlations and generate results files in addition to a screen output. The required tools and dependencies are reduced and the user inputs are safer.

A web version of TFM-Explorer is also available at `http://bioinfo.lifl.fr/TFME/`. It has limited options for the available organisms and the transcription factor biding profiles, but it offers useful tools to analyse further the results.

1.3 Licence

1.4 Publications

Predicting transcription factor binding sites using local over-representation and comparative genomics
Matthieu Defrance, Helene Touzet
BMC Bioinformatics 2006

1.5 Aim of this document

The aim of this document is to provide all the information to a user to be able to fully use TFM-Explorer and to understand its results. The document will detailled the content of TFM-Explorer, how to install it and how to use it. Detailed about how to read the results will also be given. A set of frequently ask questions is available in this document, and a contact email adress can be used in case of unexpected bugs or additional information.

2 Installation

TFM-Explorer was developped in a Unix environment, and is a command line tool. Depending on the platform you are working on, you may or may not need additional tools to run it.

2.1 Pre-required

2.1.1 Linux platforms

TFM-Explorer was written in Python, so to use it on Linux you need to have a version of Python above 2.3 installed.
Python is usually already installed on Linux distributions, so you just need to check the version.

TFM-Explorer also uses the following Python librairies:

- Numpy
- Pyrex: version $\geq$ 0.9.6.3

2.1.2 Windows platforms

To use TFM-Explorer on Windows you have to choices: use a source version or a compiled one.

If you use the source version, you need to have Python installed, with a version older than 2.3. You also need to install the following libraries:

- Numpy
- Pyrex: version $\geq$ 0.9.6.3

To install Python the best option is ActivePython
(`https://www.activestate.com/activepython/downloads/`).
For the librairies : `http://sourceforge.net/projects/numpy/files/` and `http://www.cosc.canterbury.ac.nz/greg.ewing/python/Pyrex/`.

If you use a compiled version there is no need to have Python and the libraries installed, as it is included in the package. To use this version, the commands are the same as for the source version, except that the '.py' files are replaced by '.exe'.

Note: on Windows, the command to run a Python script is different than on UNIX. The command 'python' can be skipped, or replaced by the path toward the Python executable file. For example, 'c:\Python26\python.exe' if you installed Python 2.6 on the C drive.

2.1.3 MacOSX platforms

Same as for Linux platforms.

2.2 Downloads

TFM-Explorer is available to download at the following adress:
`http://bioinfo.lifl.fr/TFME/downloads.php`.
On this page you can donwload the TFM-Explorer version adapted to your system.
Each download link provides you with a compressed file.
Extract the content wherever you want, it will give a unique folder called **TFM_Explorer**.

```
        tar zxvf TFM-Explorer_2.0.tar.gz
```

Run the *configure.py* script to check your dependences:

```
configure.py
```

Install the required librairies if needed.
That is all, there is no other installation step.

<u>Windows version</u>: you only have to extract the folder:

```
        unzip TFM_Explorer_2.0.zip
```

No matter which version you are using, you will need to use a command line prompt and to place yourself inside this folder to be able to use TFM-Explorer.

```
        cd TFM_Explorer
```

2.3 Optional compilation

One library of TFM-Explorer is written in C to make it faster: *cPataud.c*.
Binaries of this library for the main platform distributions are distributed with TFM-Explorer,but you may need to recompile it in order to make it compatible with your system.

To do that, a script is available in the folder **TFME**: *setup.py*

To use it, place yourself into the TFME folder:

```
        cd TFME
```

then type the following command:

```
 setup.py build_ext --inplace
```

This will create, inside the folder **Pataud**, a file named *cPataud.so* (or *cPataud.pyd* if you are on Windows). You need to put this file in one of the folder corresponding to you platform: Linux, Windows or Mac (replace the file that is already inside).

Note that if you are using a compiled version of TFM-Explorer (if you do not have Python installed on your machine), you do not need to recompile the library as everything is inside the binary file.

3 Use

3.1 Principle of TFM-Explorer

TFM-Explorer is a software that search for locally overrepresented transcription factor biding sites in a set of corregulated genes. It proceeds in three main steps.

First, it uses transcription factor biding profiles to analyse gene sequences and detect all potential transcription factor biding sites.

In a second step, TFM-Explorer calculates a score function by estimating a score for each position of the input sequences, using an adapted background model. The background model represents, for each position of a sequence, the probability to randomly have a hit for a precise transcription factor. The score function is calculated by estimating a score value for each position of a group of sequences. Then, windows with a high score are extracted.

In the last step, the significance of the extracted windows is evaluated. For each window a P-value is calculated and the windows are classified by increasing P-value.

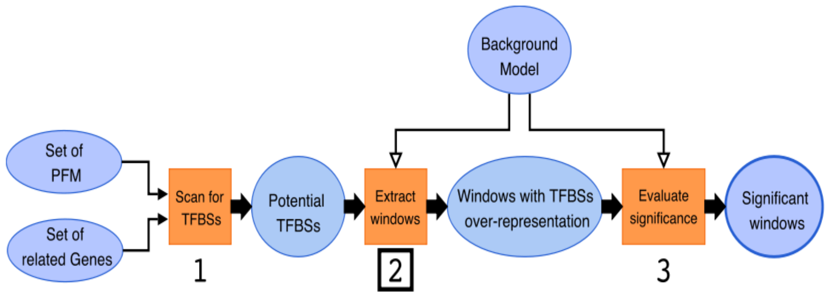

Figure 1: General schema of TFM-Explorer

To conclude, in order to use TFM-Explorer you need to:

1. have the complete sequences of the genes you want to scan for
2. have the different transcription factor biding profiles you want to use
3. build the background models adapted to the sequences
4. run TFM-Explorer

3.2 Available files

The TFM-Explorer folder contains all the files that are needed to run it.
Four executable files are available:

- *fasta2dump*: convert sequence files into a binary file readable by TFM-Explorer
- *matrix2dump*: convert weight matrices files into a binary file readable by TFM-Explorer
- *tfmebg*: build background models from sequences and matrices files
- *tfme*: run TFM-Explorer itself

All the libraries used by these files are contained in the folder TFME (or in the TFM_Explorer folder for Windows version) and should not be modified.

In order to make it easier to read sequences and weight matrices, they need to be converted into a binary format that is readable by TFM-Explorer. That is the aim of the programs *fasta2dump* and *matrix2dump*.These two files read an input text file in a specific format and output a binary file that will be used in the following steps.

The program *tfmebg* takes a sequences file and a matrices file, both in binary format from the programs described above, and builds a background model, which is another binary file that is used by TFM-Explorer.

Finally, the program *tfme* runs the TFM-Explorer algorithm itself, using one or several background models created by the previous program, as well as binary matrices files and optionaly binary sequences files.

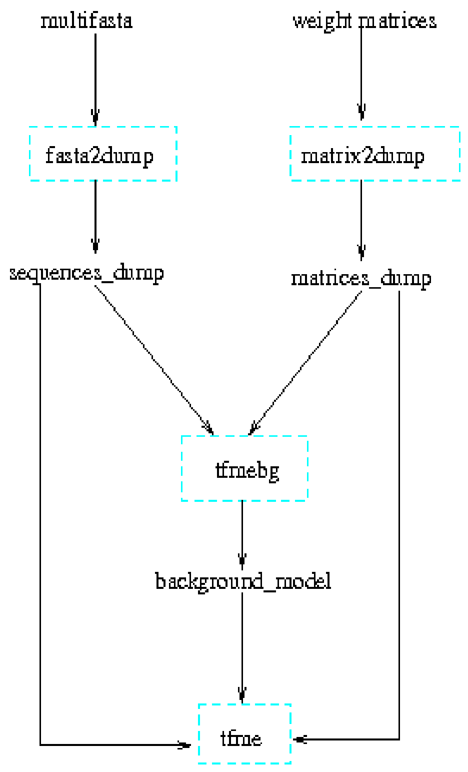

Figure 2: Architecture of executable files

3.3 Command lines

In this section we will describe how to use each file of TFM-Explorer.
The first thing you can do is to run the example.

```
cd Example
make new
```

Or if you are on Windows:

```
cd Example
```

and launch *example.bat*
This will run the four programs of TFM-Explorer on a small example to check everything works well.
It is at this step that you may realize that you need to recompile *cPataud* (see Optional compilation).

3.3.1 fasta2dump

The aim of this script is to transform a mulifasta of dna sequences into a binary file used as a database of sequences. This file will then be used to build a background model or to retrieve sequences in TFM-Explorer.

Use
The command line to run *fasta2dump* is:

```
fasta2dump.py --input=filename.fasta --output=dumpfilename.seq.db
--organism=organism --location=x:x
```

Where **input** is a path toward a multifasta file, **output** a path toward a file to create, **organism** is the organism of the sequences in the multifasta and **location** is the zone of the gene where the sequences come from (start and end position regarding to the Transcription start site (TSS)). This location is used to find the TSS and to be able to extract smaller sequences from the database later.

Formats
The input of this script is a multifasta with dna sequences. The format is the one of a standard fasta file. The headers must contain an identifier for each sequence, and the organism, separated by a vertical bar (|). The organism must be the same as the one put in the command line.

```
>sequenceID | organism
ACGTGCACGTAACGTCA...
```

All the sequences must have the same length. If it is not the case, two options are taken:

- If a sequence is longer than the entered location, it is cut in order to have the same length as others
- If a sequence is shorter than the entered location, its length is taken as a rule for others. All the sequences are cut to fit this length, and the location is changed to: (-smallerdna:0).

Results
The result of this script is a binary file containing a database of sequences and usable in the next steps of TFM-Explorer. The evolution of the sequences extraction is displayed on the screen while the script is run. A message will appear in case of error, and if the script run without problems.

Note: on MacOS it can happen that a '.db' is added at the end of the filename you chose for the binary file. This is a normal behaviour and is not harmful for TFM-Explorer.

3.3.2 matrix2dump

The aim of this script is to transform position frequency matrices into a binary file used as a database of weight matrices. This file will be used to build a background model or to run the TFM-Explorer algorithm.

Use
The command line to run *matrix2dump* is:

```
matrix2dump.py --input=filename --output=filename.mat.db --type=TYPE
```

type can be one of the three possible input format matrix: jaspar, transfac or perso.
TFM-Explorer can read position frequency matrices from the Jaspar database (`http://jaspar.cgb.ki.se/`), the public Transfac database (`http://www.gene-regulation.com/pub/databases.html`), or in a personal format described later.

Formats
The input matrices can be in three possible formats, depending the type of matrices entered.

For the Jaspar database, the input will be the directory where the pfm files are, with their annotation file.

For the Transfac database, the input will be the .dat file with the matrices.

For personal matrices, the input will be a fasta-like file, with for each matrix a header with the identifier, the transcription factor name, the class and the species. The matrix itself must be in Jaspar format (ie: one base per line).
Example:

```
>ID,name,class,species
 4 19  0  0  1  0
16  0 20  0  0  0
 0  1  0 20  0 20
 0  0  0 10  0  5
```

All matrices must contain position frequency and not weights. They will be converted into position-weight matrices later.

Results
The result of this script is a binary file containing a database of matrices and usable in the next steps of TFM-Explorer. The evolution of the matrices extraction is displayed on the screen while the script is run. A message will appear in case of error, and if the script run without problems.

3.3.3 tfmebg

The aim of this script is to build a background model for TFM-Explorer using sequences and matrices(see Principle of TFM-Explorer). The background model will then be used in the TFM-Explorer algorithm. For running TFM-Explorer you need to have a background model for the matrices your are using. Ideally, a background model is made using a large number of representative sequences for an organism.

Use
The command line to run *tfmebg* is:

```
tfmebg.py --matrices=file.mat.db --sequences=file.seq.db
--output=file.bg [--prioriprob=x]
```

Where **matrices** and **sequences** are binary files produced by the two previous scripts, respectively *fasta2dump.py* and *matrix2dump.py*. **prioriprob** is an optionnal option to change the probability as a threshold of acceptance for a matrix.

Note: depending on how many sequences and matrices you are using, building a background model may be very long (it can take several hours).

Formats
The two input files must be binary files created by the other TFM-Explorer scripts.

Results
The result of this script is a binary file containing the background model and usable in the next step of TFM-Explorer. The evolution of the building is displayed on the screen while the script is run, with an estimation of the remaining time. A message will appear in case of error, and if the script run without problems.

3.3.4 tfme

The aim of this script is to run the TFM-Explorer algorithm itself. It will scan dna sequences with position frequency matrices and extract overrepresented windows.

Use
The command line to run *tfme* is:

```
tfme.py --input=filename  [--fasta] --background=file.bg
--matricesDB=file.mat.db --output= ../resultsTFM/ --location=x:x
[--sequencesDB=file.seq.db]  [--top=x or --pvalmax=x] [--ratio=x] [-v 3]
```

where:

- input: a file with the dna sequences to scan. Either a list of sequence identifers to retrieve in the *sequencesDB* database, either a multi fasta. If the input is a fasta, the option *–fasta* must appear. If it is a list of identifers, a sequences database must be given with the option *sequencesDB*.
- background: a background model built with the *tfmebg.py* program. The sequences used to build it must be longer or equal to the one used as input.
- matricesDB: a database of position frequency matrices created with the *matrix2dump.py* program and used to scan the sequences. Must be the same as the one used to create the background model.
- output: a directory where to put the result files. If nothing specified, the files are put in the current directory.
- location: the scanned zone, ie the position of the sequences (default: -2000:200, must be between -10000 and 5000). If the input is a list of identifers, the sequences are extracted from the database according to this location. If the input is a fasta file, the same rules as for *fasta2dump.py* are applied (see 3.3.1) .
- sequencesDB : a database of sequences created by *matrix2dump.py*. Used to retrieve dna sequences from identifiers.
- top: The number of result clusters to output (max=50, default=20).
- pvalmax: alternative to 'top', maximum p-value for the outputted clusters. If both 'top' and 'pvalmax' are specified, 'top' is used.
- ratio: optional option, used to count against background model, level above which a motive is considered overrepresented (default= 2.5, must be between 1.0 and 3.0).
- v: verbosity level.

Formats
If the input is a multifasta file, the format of the sequences must be the same as for all TFM-Explorer scripts (see 3.3.1)

The location must be given in the format '(lowerBoundary : upperBoundary)' regarding to the Transcription Start Site.
example: (-2000:200)

Results
The results of TFM-Explorer are clusters. A cluster is a zone of dna sequences associated with an overrepresented transcription factor. It is a zone common to all the input sequences, but not all the sequences have a hit for this transcription factor. The overrepresentation is calculated on all the sequences.

Screen output

While the script is executed, its evolution is displayed on the screen. When it is over the results are displayed, as a list of top ranked clusters with their matrix, transcription factor, location, p-value and other information. If the verbosity is set at 2 or above, a detailed description of each cluster is also given, with the hits of the transcription factor.

Result files

In addition to the screen output, three result files are created. These files are not designed to be read by humans but by other programs. They can be used as inputs for another software with the purpose of making additional treatment.
There are three result files:

- windows.xml: an xml file containing the information about the clusters displayed on the screen.
- hits.csv: a csv file with all the hits found by TFM-Explorer.
- correlation.csv: a csv file containing all the pairwise correlations between the clusters.

These file are put in a folder named with a unique identifer.

Using the four scripts described here, one can use TFM-Explorer to obtain precise predictions about transcription factor binding sites on a set of corregulated genes.

4 FAQ

I got an error while compiling cPataud:

Check if the files *cPataud.c* or *cPataud.so* do not already exist in *TFME/Pataud*. If yes, delete them and run again the command to compile, the files will be created again. It can happen that exisiting files prevent the compilation to run correctly. Deleting them should solve the problem.

I got the "unable to find vcvarsall.bat" error while compiling cPataud (Windows)

This probably means that Pyrex does not find your C compiler. Try to specify it in the command by adding ' –compiler=mingw32' (if you use another C compiler replace *mingw32*).
The command line to compile cPataud now should be:

```
setup.py build_ext --compiler=mingw32 --inplace
```

Why tfmebg is so long?

The time to build a background model is long because all the input sequences are scanned with each of the input matrices, a threshold is calculated for each sequence to determine when a hit is valid, and then all the sequences are scanned again to build the background model itself, with the probability on each position to have a hit for each matrix. This is extremly long but compulsory for the well behaviour of TFM-Explorer.

Why don't you make some background model available?

Because the binary format used to store the background models depends on your Python distribution. You may not be able to read already built background models.

Why two versions: a command line and a web version?

The web version of TFM-Explorer offers great tools to vizualise and analyse the resulted clusters. But on the other hand, it is limited to the organisms and the matrices we chose. The command line version allows you to keep your data on your machine and to use other organisms/assemblies or matrices. This could be useful if you are working on specific data.

Why binary files to save matrices and sequences?

TFM-Explorer deals with a lot of data that need to be easily accessed. The choice of binary files is a security to prevent for modifying the sequences or matrices databases, as they contain computer language code that is not easy to read for a human. As a result, every file created by a TFM-Explorer script can be read by another one.

How to use the compiled version on Windows?

This version already contains Python and its libraries inside. To use it the command lines are exactely the same as describe in this document, except that you do not need to specify the path to the Python executable file. For example to run the *fasta2dump* script:

```
fasta2dump.exe --input=filename.fasta --output=dumpfilename.seq.db
--organism=organism --location=x:x
```

The behaviour of the scripts is exactely the same as for the source version.

5 Contact

For information, questions or bug reports, please contact the authors at: `tfm@lifl.fr`