TICO User's Guide
(Release 2.1)

Maike Tech

8th October 2006

GACCGTAGAAGCCACAAAAAATGAATGTTAATTACCTGA

CTGCAAGGACTGGATATGCTGATTCTTATTTCACCTGAATGCGCTTAT

CCTTCCCACTAAGAGCTTATGAAATCCGTTTTTACGATTTCCGCCAGC

TTTAAATAAAATTGCTGTCAATTTTACGTCTTGTCCTGCCACATTCTT

AATTAGGGAGACGTTTAGATGGGTAAAATAATTGGTATCGACCTGGGT

CAATTCTAGGAAGGTTCCTCTCCGCCCGTGCATTCAGGCTTAAAAAAG

TTACGCCGATATGATTTAAGTCGTGCCGATGAATTACTCGATAACTGG

TCACCTGAAAGAGAAATAAAAAGTGAAACATCTGGATTCTTTAGCAGT

AGCCATAAACGGCTCCCTTTTCATTGTTAGAGAGAAATGAGCACGTCT

TCGTGAGTCATCGTCGCGCGGAAATGACAGTGGGGC

Contents

Preface

The readme of TICO2.1 is still in a state of revision, therefor parts may be written unclear or contain mistakes. Unquestionable it needs to be improved. The reader may excuse the present insufficiency. We are grateful for comments and suggestions concerning both the readme and the tool.

1 Introduction

TICO is a tool for post processing of gene predictions with regard to improve the prediction of the correct translation initiation sites (TIS). Therefor TICO requires the input of a sequence in FASTA format and a set of predicted genes. With the initially predicted TIS, TICO generates a set of candidates and scores each candidate in an iterative process by means of a clustering algorithm. The initially predicted TIS will be relocated if another candidate belonging to the respective ORF has a higher positive score. A description of the algorithm can be found at the TICO web interface (http://tico.gobics.de) and in [1].

Since version **2.0** TICO has the feature of automated adaption (optional) of the smoothing parameter sigma (see section 6, p. 8). Also a bug has been eliminated causing an ArrayIndexOutOfBoundException under some circumstances (very rare). Since version **2.1**, the weights calculated by TICO can be visualized with the tool `WeightsVis`, as detailed in 9, p. 17.

There are some requirements to be fulfilled to run TICO properly. Please read the readme file carefully (at least part of it) before you try to run the program. If you did not succeed to install and run the stand-alone version of TICO, find any bugs or have any further questions, please send an email to the following address: tico-info@gobics.de.

The easiest way to install TICO is from the »complete package« version. Than you only have to have Java1.5 available on your system and to install the MATLAB® libraries (only for Windows). However, if for any reason you prefer to install the components to separated locations or if some of the components already are provided on your system, the necessary steps (under Linux) are detailed in this readme.

Currently two commandline versions of TICO for Linux and Windows are provided. The versions differ in the mail interface which is only provided by one of the implementations. This version of TICO requires Java Mail an Java Activation to be available on your system. If it is not necessary, to send the results of TICO anywhere via mail, the TICO version without mail interface would be sufficient.

The implementation of TICO consists of two parts. The user interface have been implemented in Java1.5 and requires the Java Runtime Environment 1.5 (JRE1.5). The clustering routines were developed under MATLAB®and were compiled with the MATLAB®-compiler.

The tool is developed and tested on Debian Linux (Sarge), WindowsXP and Windows2000. The jar-file `tico.jar` is platform independent (as well as the additionally needed jars), while the starter-script and the MATLAB®-classes are platform dependent. The Linux examples are given for a `bash` and the starter script is also written for a `bash`. If you use another shell some modifications may be necessary. The batch script to start the main program under Windows is written and tested for DOS version 7.0 and higher. If you use an older DOS version you should start the TICO main class manually.

Please note the license terms (see section 10, p. 19).

2 Installation – Linux

In the following a description of the necessary steps to install and run TICO under Linux is given. The »Quick Guide« gives a step-by-step instruction, the individual steps are detailed in the succeeding sections.

2.1 Quick Guide

For both versions Java 1.5 needs to be available on the system.

For the complete package:

Unpack the tarball and

start TICO with the starter script (see 3.1, p. 6)

or adjust the `PATH`, the `LD_LIBRARY_PATH` (see 2.3, p. 5) and the `CLASSPATH` (see 2.2.1, p. 5) and start the Java main class of TICO (see 3.2, p. 6).

For the individual components:

1. Make the Java components (jars) available on your system (see 2.2, p. 4).
2. Install the MCR (MATLAB® Component Runtime library archive, see 2.3, p. 5).
3. Setting the `LD_LIBRARY_PATH` (needed for the MATLAB® components, see 2.3, p. 5).
4. Adding the necessary jars to the `CLASSPATH` (see 2.2.1, p. 5).
5. Start TICO with the starter script (see 3.1, p. 6) **or** adjust the `PATH` and start the Java main class of TICO (see 3.2, p. 6).

2.2 Installation of the Java components

The Java Runtime Environment (at least version 1.5.0) should be available on your system. The Java API can be downloaded from the web interface of SUN: http://java.sun.com/. TICO additionally uses the following packages :

- Log4j (log4j.jar): Can be downloaded from the Log4j home at http://logging.apache.org/log4j/

  Java Activation and Java Mail are only needed for the version with mail interface!

- Java Activation Framework (activation.jar): Available at
  http://java.sun.com/products/javabeans/glasgow/jaf.html
- Java Mail (mail.jar): Can be downloaded from SUN at
  http://java.sun.com/products/javamail/

Theses jars are included in the complete package version and will be included temporarily in the `CLASSPATH` during the invocation of TICO with the starter script. If you do not use this version the jars should be downloaded and installed separately, additionally they should be included in the `CLASSPATH` (see next section).

2.2.1 Setting the CLASSPATH

(only needed, if you do not use the starter script and the complete package version)

Your CLASSPATH should include the jar (Java archive) of Log4j and if you use the TICO version with mail interface, you additionally should include Java Activation Framework and Java Mail.

Example without mail interface

```
export CLASSPATH=/usr/share/java/log4j-1.3alpha-3.jar
```

Example with mail interface

```
export CLASSPATH=/usr/share/java/mail.jar:/usr/share/java/activation.jar:\
/usr/share/java/log4j-1.3alpha-3.jar
```

Note: The jar of TICO (`$TICO_HOME/tico.jar`, with `TICO_HOME` referring to the directory in which TICO is installed) should be included to the `CLASSPATH` if you do not use the starter script to run TICO. If you use the script, the `PATH` and the `CLASSPATH` will be temporarily extended (see section 3.1, p. 6).

2.3 Making the MCR available under Linux

To run a MATLAB®-compiler generated stand-alone application, the MCR (MATLAB® Component Runtime library archive) should be available on your system. If it is not yet installed please follow the instructions below. The installation only should be performed one time for all MATLAB®-compiler generated applications you want to run on your system. For Linux, the installation just means, to unpack the MCR an than set the `LD_LIBRARY_PATH`. If you use the starter skript of TICO the path will be set automatically to the MCR in the `TICO_HOME` directory.

If you do not use the starter skript, you should set the variable `LD_LIBRARY_PATH` manually. In the following the MCR directory is addressed `MCR_ROOT`. The terms `arch` and `arch2` should be set to the systems architecture, for example `arch=glnx86` and `arch2=i386` for 32 bit Intel machines or `arch=glnxa64` and `arch2=amd64` for AMD 46 bit machines, respectively.

Set the variable `LD_LIBRARY_PATH` to:

```
${MCR_ROOT}/<ver>/runtime/${arch}:\
${MCR_ROOT}/<ver>/sys/os/${arch}:${MCR_ROOT}/<ver>/bin/${arch}:\
${MCR_ROOT}/<ver>/sys/java/jre/${arch}/jre1.5.0/lib/${arch2}/native_threads:\
${MCR_ROOT}/<ver>/sys/java/jre/${arch}/jre1.5.0/lib/${arch2}/client:\
${MCR_ROOT}/<ver>/sys/java/jre/${arch}/jre1.5.0/lib/${arch2}:\
```

The term `<ver>` should be replaced by the version number (for example `v74`).

`bash` example for Intel 32 bit machines:

```
MCR_ROOT=${HOME}/MCR
export LD_LIBRARY_PATH=${MCR_ROOT}/v74/runtime/glnx86:\
${MCR_ROOT}/v74/sys/os/glnx86:\
${MCR_ROOT}/v74/bin/glnx86:\
${MCR_ROOT}/v74/sys/java/jre/glnx86/jre1.5.0/lib/i386/native_threads:\
${MCR_ROOT}/v74/sys/java/jre/glnx86/jre1.5.0/lib/i386/client:\
${MCR_ROOT}/v74/sys/java/jre/glnx86/jre1.5.0/lib/i386:
```

The LD_LIBRARY_PATH should be available on your system, thus it should be set in the shell (e. g. in the .profile or .bashrc if you want it to be avaliable permanently).

Please note the licence file for the MATLAB®-compiled application!

3 Running TICO on a Linux platform

In the following TICO_HOME refers to the directory in which TICO is installed. You have two possibilities to start TICO: You can start the program with the bash-script tico or you can start the Java main class. If you directly start the main class, you need to perform the invocation in the directory TICO_HOME or pass TICO_HOME in the commandline. Additionally it may be necessary to include TICO_HOME in the PATH of your system (if . is not included) and ${TICO_HOME}/tico.jar in your CLASSPATH.

The easier way is to start TICO with the starter script tico. The script will start the Java class and pass over the TICO_HOME directory. The PATH, the LD_LIBRARY_PATH (only in the complete package version) and the CLASSPATH will temporarily be adapted to include TICO_HOME, MCR_HOME (only in the complete package version) and the jars respectively. If you do not use a bash it may be necessary to adapt the script, at least the interpreter call.

3.1 Using the bash-script

If you call TICO with the starter script, you only need to pass over the sequence file and a file containing the predicted genes. It is necessary that the script and the configuration file are located in TICO_HOME .

```
./tico -s seq.fna -g glimmer.out
```

3.2 Running the Java main class of TICO

If you would like to run the Java main class, TICO_HOME should be available in the PATH of your system and tico.jar as well as the other jars should be included in the CLASSPATH. That is necessary for the MATLAB® components and the Java classes to be found. Furthermore it is necessary to pass TICO_HOME during invocation if you do not perform the invocation from TICO_HOME. Otherwise the configuration file will not be found.

Example:

```
java TiCo [$TICO_HOME]  -s seq.fna -co coord.out
```

If an error occurred during the invocation or while TICO is running, check the log-files for error messages (default: /var/tmp/tico.log).

4 Installation – Windows

The necessary adaption to install and run TICO properly under Windows are enumerated step-by-step in the »Quick Guide«. A detailed description of the steps to perform can be found in the referred sections.

4.1 Quick Guide

Java 1.5 needs to be available on the system (see 2.2, p. 4).

1. Install the MCR (MATLAB® Component Runtime library archive) with `MCRInstaller.exe`
2. Start TICO with the starter script (see 5.1, p. 7) **or** adjust the `PATH`, the `CLASSPATH` (see 5.2.1, p. 8) and start the Java main class of TICO (5.2, p. 7).

5 Running TICO on a Windows platform

To run TICO under Windows you need to use the Windows commandline interpreter or DOS-box as it is also called. A GUI for TICO will be provided in the next version. In the examples the DOS prompt is denoted like `C:\>`, where `\` means that you are in the root directory of device `C:`.

If an error occurred during the invocation or while TICO is running, check the log-files for error messages (`%TEMP%\tico.log`).

5.1 Starting TICO with the batch-script

To run the starter script, you should type the path (relative or absolute) of the batch-script and pass the sequence file, the gene data file and other parameters like is shown in the example.

Note that you should set the path in double quotes, if it contains any delimiter characters like whitespace. Also note that the files should be given with the absolute path.

Example:

```
C:\> "c:\program files\tico\tico" -s "c:\data\seq.fna" -g "c:\data\glimmer.out"
```

In the example TICO is located in `C:\program files\tico\`. The batch-file will temporarily adjust the `PATH` and the `CLASSPATH` to include the program `run_clustering` and all necessary jars.

5.2 Running the Java main class of TICO under Windows

If the batch-file does not work on you Windows (maybe the DOS-version does not support all necessary commands), you have the possibility to start the Java main class directly. Therefore the paths should be adjusted manually, as described below (see section 5.2.1, p. 8). The call in principle complies with the call from the batch-script. The difference is, that you should add the Java-call. Additionally you should start the class from the TICO home directory or pass the TICO home directory as first argument.

Example:

From the TICO home directory

```
C:\program files\tico> java TiCo -s seq.fna -g glimmer.out
```

From anywhere

```
C:\> java "c:\program files\tico\TiCo" -s seq.fna -g glimmer.out
```

5.2.1 Adjusting the paths

Under Windows the `PATH` and the `CLASSPATH` can be adjusted through the graphical interface of the environment settings or temporarily through the DOS-box. To alter environment variables »permanently« under Windows 3.x, Windows9x or WindowsME you should set them in the file `autoexec.bat` which is located in the root directory on your boot device (probably `c:\autoexec.bat`). The syntax is the same as in the DOS-box (see below).

To set a path the »clicking-way«, you should find in the system settings (start menu → settings) the point *system*. One of the tabs is denoted *advanced*, there you find a button *environment variables*. A variable `PATH` is probably already defined as system variable. In oder to avoid overwriting the system variable, when you add a user-defined variable `path` you should include the system variable denoted as `%PATH%`.

Example:

Add a variable called `path` (case independent) with the value

```
%PATH%;c:\program files\tico\
```
(assuming TICO is located in the directory `c:\program files\tico\`)

and a variable `classpath` with the value

```
%CLASSPATH%;c:\program files\tico\log4j.jar; ...
c:\program files\tico\tico.jar
```

The dots in the example indicate that there is no newline.

To set a path (or any other variable) in the DOS-environment you should use the command `set` like is shown in the example below. Calling `set` without any parameters will display all environment variables. Settings, altered in the DOS-box are only available in the respective shell and do only persist as long as the shell exist. After closing the window or in a new command window, the system settings are restored.

Example:

Setting the `PATH` and the `CLASSPATH` in the DOS-box:

```
C:\> set PATH=%PATH%;c:\program files\tico\
C:\> set CLASSPATH=%CLASSPATH%;c:\program files\tico\log4j.jar; ...
                    \verb+c:\program files\tico\tico.jar
```

Note that every component of the paths should be separated with a `;` (colon).
The dots in the example indicate that there is no newline.

6 Configuration of TICO

6.1 Commandline Parameters and Configuration File

In this section the handling and configuration of TICO are described. Some parameter can only be set in the configuration file, some may also be given in the commandline. If a parameter is

given in the commandline the respective value from the configuration file will be overwritten. All commandline parameters are summarized in table 6.1.1.

6.1.1 Commandline parameters

Search Range:
Specifies the range to be searched around putative gene starts for alternative start sites. I. e. by the search range the maximum distance to a predicted TIS as derived from the input file is defined. In this range all potential start sites are considered as candidate TIS. A potential start site is defined as start codon, that shares the same reading frame of the respective gene, with no inframe stop codon between the start codon and the annotated stop.

At first the initially predicted TIS is labeled as *strong* TIS, the alternative start sites are labeled as *weak* TIS. During the iterative classification, the label strong is assigned to the candidate start with the highest PWM-Score (i. e. the value from the positional weight matrix) among the candidates of a TIS.

- up
  Parameter `-su` (commandline), `SearchUp` (config file)
  Specifies the maximal distance to a given start position for upstream (5′) alternative starts.

  Default: 250 nucleotides
  Minimum: 50 nucleotides
  Maximum: 250 nucleotides
- down
  Parameter `-sd` (commandline), `SearchDown` (config file)
  Specifies the maximal distance to a given start position for downstream (3′) alternative starts.

  Default: 250 nucleotides
  Minimum: 50 nucleotides
  Maximum: 500 nucleotides

Extract Range:
Specifies the range to be extracted around each candidate start site. The resulting sequence window is used for the unsupervised learning. It is assumed to contain the characteristics of respective start site, e. g. the ribosome binding site.

- up
  Parameter `-exu` (commandline), `ExtractUp` (config file)
  Specifies the number of nucleotides to be be extracted upstream (5′) a given start position.

  Default: 30 nucleotides
  Minimum: 10 nucleotides
  Maximum: 100 nucleotides
- down
  Parameter `-exd` (commandline), `ExtractDown` (config file)

Specifies th number of nucleotides to be extracted downstream (3′) a given start position (inclusive start).

Default: 30 nucleotides
Minimum: 10 nucleotides
Maximum: 100 nucleotides

Sigma:
The standard deviation parameter sigma of the Gaussian density specifies the smoothing [2] of the positional probabilities of the second order Markov Models. A high value for sigma means the positional probabilities are highly smoothed. The parameter doesn't imply any assumptions on trinucleotide positions in the sequence, but adapts the estimation to a varying number of genes under consideration. The default value 0.5 works well with approximately 4000 genes. For a set with a smaller number of genes it may be useful to chose a higher value for sigma to prevent vanishing probabilities.

Parameter `-sig` (commandline), `Sigma` (config file)

Range: 0.1 - 2.0

Default: 0.5

ROC-flag (since release 2.0)
The ROC (*Receiver Operating Characteristics* curve) can be used to optimize the smoothing parameter sigma in an automated way. In the default configuration the flag is set to 1, i. e. true, so the sigma is automatically adapted. The final sigma value is given in the last line of the file `tis.res` in the output directory and in the log-file. If the flag is set to 0, the initial sigma value as set in the configuration file or given as commandline parameter is used as smoothing parameter. The ROC-flag can be either set in the configuration file or as commandline parameter.

Parameter `-roc` (commandline), `ROC` (config file)

Range: 1 (true) or 0 (false)

Default: 1

Minimum gene length:
Specifies the minimum length gene after may have after reannotation of the TIS (denoted in bp). If the distance of a potential candidate TIS falls below the minimum length it is omitted from the list of candidates.

Parameter `-minlength` (commandline), `MinLength` (config file)

Default: 60 bp

Output directory:
If an output path is given in the commandline, all files generated by TICO (see 7.2, p. 14) are written directly to this directory. If no path is given, the results are written to a new created folder in the default path, which is set in the configuration file. The preset default path is `/var/tmp/` (under Windows the default path is: `c:\var\tmp\`) . The name of the output directory will be generated random numbers like `9104113053744858521`.

Parameter `-io` (commandline), `OutputPath` (config file)

Summary of commandline parameters

Required Parameters			
`-s seq-file`	the sequence in FASTA format (see section 7.1.3, p. 13)		
and one of the following files	containing the initial prediction		
`-g glimmer-file`	GLIMMER predictions (see section 7.1.1, p. 13)		
`-co simple-coord-file`	predictions in simple-coord format (see section 7.1.2, p. 13)		
Optional Parameters			
`-io dir`	output directory		
		default range	default value
`-su number`	upstream search window	1–500	250
`-sd number`	downstream search window	1–500	250
`-exu number`	upstream extracted window	1–100	30
`-exd number`	downstream extracted window	1–100	30
`-minlength number`	minimum gene length	10–...	30
`-sig number`	smoothing parameter	0.1–2.0	0.5
`-roc 1|0`	ROC-flag	1 or 0	1
TICO with mail interface			
`-u user@domain.xy`	Email address		
`-glimmer file-name`	GLIMMER output will be attached (see section 7.2.1), p. 14		
`-coord file-name`	output in simple coord format will be attached (see section 7.2.3, p. 16)		
`-gff file-name`	output in GFF will be attached (see section 7.2.2, p. 15)		

6.1.2 Configuration file `.tico`

The configuration file provides the possibility to configure the external interfaces and to set default values, to shorten the commandline call. In the package an example `.tico` is given, with comments on the parameters. To leave a parameter empty modifies the behavior of TICO in some cases. In the case of the output files, the result in respective format is not written in consequence of a deleted value. If a key is missing, an entry of level `WARN` is written to the log file.

If you would like use the mailer interface, your outgoing mail server should be entered (key `mail.smtp.host`). You also should store the email address of the sender in the parameter `fromAdr`. The recipients address may also be set in the configuration file (`DefaultRecipient`) or can be given in the commandline with the parameter `-u`.

Change the Nucleotide- or Codon-Table: You can change the nucleotide symbols to be considered by TICO as well as the sets of start and stop codons. Therefor you should add/adjust the respective table files to the TICO home directory. Example files are given with in the TICO tarball (`${TICO_HOME}/nucleotides` and `${TICO_HOME}/codons`). The name of the files which should be read by TICO should be given in the configuration file with the key `Nucleotides` or `CodonTable` respectively.

By default (without including a codon table) `ATG`, `GTG` and `TTG` are considered as potential start codons and `TAG`, `TAA` and `TGA` as stops. To change the set of start or the stop codons, first of all you should set a codon file in the config of TICO (by default `${TICO_HOME}/.tico`). Then you should include/exclude codons as given in the example:

```
STOPCODON=TAG
STOPCODON=TGA
#STOPCODON=TAA   #this stop codon is excluded
STARTCODON=ATG
```

You can either delete the respective line or just comment the line out.

6.2 Configure the Logging

The logging can be adapted with the configuration file `.log4j` which is located in the TICO home directory. The handling of the logging is commented in the configuration file and is documented in detail in the Log4j documentation which is available at Log4j home (see 11, p. 20).

Examples for the default the logging format:

```
[2005.03.04-17:06:22,817] ERROR - GlimmerOutputParser: Not a valid \
Glimmer output file! Line: 3
```

```
[2005.05.07-14:46:54,522] ERROR - TiCo: Error while running run_clustering.\
The external interface did not work properly, IO error reading results\
no results were written! Please consult the readme file.
```

By default TICO logs to the file `/var/tmp/tico.log`. If you would like to log to another path, you should modify the line:

```
log4j.appender.R.File=/var/tmp/tico.log
```

in the log4j configuration file.

If you want the logging to be directed to standardout, the line

```
#log4j.rootLogger=INFO, stdout, R
```

should be commented in and the line

```
log4j.rootLogger=INFO, R
```

should be commented out.

As you see, the log level is set to `INFO` by default. This is a verbose level, TICO informs you, which paths are set and which steps are performed. The log level can be changed by replacing the keyword `INFO` with `WARN`, `ERROR` or `FATAL`. But note that even an event reported in the level `WARN` may cause the tool to produce empty results.

7 Input and Output formats

In this section the input and output formats are described, which are provided by TICO at present. Additional input and output formats (e. g. GenBank format) are in preparation and will be integrated in the next release.

7.1 Input formats

For the moment only GLIMMER [3] format and our own format (called »simple coord«) are provided for the post processing of gene predictions.

7.1.1 GLIMMER predictions

The input file may contain the whole GLIMMER output, but needed is only the section with the putative genes. TICO searches the file for the line *Putative Genes:* and reads all predicted ORFs from there on.

Note: In GLIMMER output the genes are denoted exclusive stop!

Example for an input file in GLIMMER format:

```
Putative Genes:
    2      337     2796  [+1 L=2460]
    3     2801     3730  [+2 L= 930]
    5     3734     5017  [+2 L=1284]
    6     5088     5234  [+3 L= 147]  [Vote]
    8     5720     5313  [-3 L= 408]  [DelayedBy #10 L=21]
   10     6459     5686  [-1 L= 774]
   12     7959     6532  [-1 L=1428]
   14     8175     9188  [+3 L=1014]
   15     9303     9890  [+3 L= 588]
   17    10494     9931  [-1 L= 564]
   19    11356    10646  [-2 L= 711]
```

GLIMMER files should be passed with the parameter `-g`.

7.1.2 »Simple coord« format

Simple coord format is as the name indicates a simple format containing only an id, the coordinates of the gene (inclusive stop codon) and the strand, denoted in the following format.

```
>id_pos1_pos2_strand[_score]
```

Example for the simple coord format:

```
>2_337_2799_+
>3_2801_3733_+
>5_3734_5020_+
>6_5088_5237_+
>8_5310_5720_-
>10_5683_6459_-
>12_6529_7959_-
>14_8175_9191_+
>15_9303_9893_+
>17_9928_10494_-
>19_10643_11356_-
```

Simple coord files should be passed with the parameter `-co`. Note that either a file in simple coord format or in GLIMMER format may be given.

7.1.3 Sequence format

The genome sequence should be given in FASTA format as shown below. The first line may contain details for identification of the organism, but may also be omitted. As symbols for nucleotides both upper- and lowercase characters are accepted. To be processed the sequence should only contain valid nucleotide symbols according to the IUPAC-standard (table of valid symbols is available on the TICO website.). For the training only the symbols A, C, G and T (upper and lower case) are considered. All other IUPAC symbols will be ignored.

Example for the FASTA format:

```
>gi|6626251|gb|U00096.1|U00096 Escherichia coli K-12
AGCTTTTCATTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTAAAAAAAGAGTGTCTGATAGCAGC
TTCTGAACTGGTTACCTGCCGTGAGTAAATTAAAATTTTATTGACTTAGGTCACTAAATACTTTAACCAA
TATAGGCATAGCGCACAGACAGATAAAAATTACAGAGTACACAACATCCATGAAACGCATTAGCACCACC
ATTACCACCACCATCACCATTACCACAGGTAACGGTGCGGGCTGACGCGTACAGGAAACACAGAAAAAAG
CCCGCACCTGACAGTGCGGGCTTTTTTTTTCGACCAAAGGTAACGAGGTAACAACCATGCGAGTGTTGAA
GTTCGGCGGTACATCAGTGGCAAATGCAGAACGTTTTCTGCGTGTTGCCGATATTCTGGAAAGCAATGCC
AGGCAGGGGCAGGTGGCCACCGTCCTCTCTGCCCCCGCCAAAATCACCAACCACCTGGTGGCGATGATTG
```

The FASTA file should be given with the parameter `-s`.

7.2 Output formats

At the moment TICO provides three kinds of output format: An adapted GLIMMER format, general feature format (GFF) and the simple coord format. The names of the output files are set in the configuration file of TICO. If the name is omitted in the configuration file, respective output will not be written.

The output directory of TICO in the default configuration will contain the files:

```
tico_results.coord
tico_results.gff
tico_results.gl
tico.weights
tis.altseq
tis.ref2line
tis.res
tis.seq
```

The files `tico_results.coord`, `tico_results.gff` and `tico_results.gl` contain the prediction of TICO as described in the following sections. The files `tis.seq` and `tis.altseq` contain the sequence windows around the candidate TIS. `tis.seq` contains the candidates that were given to TICO during the invocation i. e. the candidates initially labeled as *strong* TIS. `tis.altseq` contains the additional candidates generated by TICO, i. e. the candidates initially labeled as *weak* TIS. In the file `tis.ref2line` the mapping of the initially labeled *weak* candidates to the respective *strong* candidate are denoted. That is to say, this file contains for each sequence from `tis.altseq` the line number of the respective sequence in `tis.seq`. If for example the first three candidates in `tis.altseq` are denoted with the number `1`, these three plus the first from `tis.seq` are the TIS candidates for the first gene.

In the file `tico.weights` the weights calculated by TICO during the clustering are written, the file `tis.res` contains the results of the clustering algorithm. Each line contains the label (`+` or `-`), the line number and the PWM score calculated for the candidate TIS.

7.2.1 Adapted GLIMMER format

The output is denoted in a GLIMMER-like format. That means, it contains all predictions from the input file in the same format like GLIMMER with two additional columns from the TICO prediction. In the first column after the GLIMMER output the PWM score is given, in the second the shift of the start during reannotation. Additionally, genes with a negative score are labeled with a hash mark (#) at the end of line. See also section 7.1.1, p. 13.

```
<id> <start> <stop> [comments] <PWM score> <shift>
```

The shift is given in respect of the strand of the gene. A positive value means the reannotated start is located upstream of the original start, a negative value indicates a downstream shift. If the value of the shift is 0, the start is not changed from the original prediction.

Example output:

```
Putative genes:
    2      337     2796  [+1 L=2460]  5.347931 0
    3     2801     3730  [+2 L= 930]  11.448764 0
    5     3734     5017  [+2 L=1284]  6.230648 0
    6     5088     5234  [+3 L= 147] [Vote]  3.815619 0
    8     5741     5313  [-3 L= 408] [DelayedBy #10 L=21]  -0.111382 -21 #
   10     6459     5686  [-1 L= 774]  0.234908 0
   12     7959     6532  [-1 L=1428]  19.753130 0
   14     8238     9188  [+3 L=1014]  19.169035 63
   15     9306     9890  [+3 L= 588]  19.613488 3
   17    10494     9931  [-1 L= 564]  4.670315 0
   19    11356    10646  [-2 L= 711]  13.285624 0
```

The GLIMMER-like output is by default written to `tico_results.gl`.

7.2.2 General Feature Format (GFF)

TICO provides output in general feature format (GFF) [4]. The output is given according to the specifications at the Sanger Institute (http://www.sanger.ac.uk/Software/formats/GFF/GFF_Spec.shtml):

```
<seqname> <source> <feature> <start> <end> <score> <strand> <frame> [attributes] [comments]
```

By default, the GFF output is written to a file named `tico_results.gff`.

Example output:

```
##gff-version 2
##Type DNA
Escherichia_coli  glimmer/tico  CDS      337   2799  5.347931   +
Escherichia_coli  glimmer/tico  CDS      2801  3733  11.448764  +
Escherichia_coli  glimmer/tico  CDS      3734  5020  6.230648   +
Escherichia_coli  glimmer/tico  CDS      5088  5237  3.815619   +
Escherichia_coli  glimmer/tico  REANNCDS 5310  5741  -0.111382  -  shift -21 ; note "weak tis"
Escherichia_coli  glimmer/tico  CDS      5310  5720  -0.111382  -  note "weak tis"
Escherichia_coli  glimmer/tico  CDS      5683  6459  0.234908   -
Escherichia_coli  glimmer/tico  CDS      6529  7959  19.75313   -
Escherichia_coli  glimmer/tico  REANNCDS 8238  9191  19.169035  +  shift 63 ;
Escherichia_coli  glimmer/tico  CDS      8175  9191  19.169035  +
Escherichia_coli  glimmer/tico  REANNCDS 9306  9893  19.613488  +  shift 3 ;
Escherichia_coli  glimmer/tico  CDS      9303  9893  19.613488  +
Escherichia_coli  glimmer/tico  CDS      9928  10494 4.670315   -
Escherichia_coli  glimmer/tico  CDS      10643 11356 13.285624  -
```

The GFF-output can be visualized with the program ARTEMIS [4]. The originally predicted genes are denoted with the tag `CDS`. By default the entries with the tag `CDS` are displayed in light blue. The relocated TIS are denoted with the tag `REANNCDS`. To visualize this entries, the tag should be added to the configuration file (by default `ARTEMIS_HOME/etc/options`) of ARTEMIS, for example by adding the line `colour_o_REANNCDS = 1`. Through this adaption the relocated TIS will appear grey.

7.2.3 »Simple coord« format

By default, output in simple coord format is written to a file named `tico_results.coord`. The format is denoted as shown below. See also section 7.1.2, p. 13.

Example output:

```
>2_337_2799_+_5.347931
>3_2801_3733_+_11.448764
>5_3734_5020_+_6.230648
>6_5088_5237_+_3.815619
>8_5310_5741_-_-0.111382#
>10_5683_6459_-_0.234908
>12_6529_7959_-_19.753130
>14_8238_9191_+_19.169035
>15_9306_9893_+_19.613488
>17_9928_10494_-_4.670315
>19_10643_11356_-_13.285624
```

8 Troubleshooting

In this section some difficulties are summarized that may occur during installation and invocation of TICO. This chapter will be extended when new sources of error are reported by users of TICO. So do not hesitate to give us feedback, if you had problems running the tool.

If you start the program and some error occur, for example you do not get the results you expected, the first thing to do is to **check the log-file** of TICO and, if there is, output in the commandline. The log-file is located by default in `/var/tmp/tico.log` (Windows: `c:\var\tmp\tico.log`). There you find information of the settings (for example paths and arguments) and the steps performed by TICO (log-level `INFO`). Errors are denoted with the log-level `WARN`, `ERROR` or `FATAL`.

8.1 Java Memory Error

If you want to predict the TIS for a large genome, the Java heap space may overflow. The following exception is displayed in the commandline:

```
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
```

You should add the parameter `-Xmx` to the Java call (in the starter script or in your commandline call, if you start the TICO main class directly).

Example:

```
java -Xmx512m TiCo -s seq.fna -g geneData
```

In the example the java heap space is set to 512MB. The heap should not exceed 1/4 of your physical memory, maximum is 1GB.

8.2 Java virtual machine errors

The MATLAB® libraries use the Java virtual machine. Especially on 64 bit machines this may cause some incompatibilities an errors. You will get messages like:

```
Error occurred during initialization of VM
Unable to load native library: ...
```

You should check the java you have installed on the system. In some cases the error may be solved if you set the MATLAB® variable `MATLAB_JAVA` to your system java.

8.3 Running the MATLAB® Compiler generated program

If you got an error message like:

```
Error while running run_clustering.
The external interface did not work properly ...
```

Different sources of error are possible.

1. You did not `unzip`/install the MCR-components.
2. The MATLAB® path (`LD_LIBRARY_PATH`) is not set correctly (in the starter script or in your environment).
3. The directory containing the script `run_clustering` is not included in the `PATH` or is not accessible.
4. The directory `run_clustering_mcr` of the older version has not been removed during installation of a newer version. `run_clustering_mcr` is created during the first invocation of the MATLAB® components. That causes problems, when the MATLAB® components are changed, for example if you installed a newer version to the TICO-home directory.
5. The output directory is not readable or not writable.

8.4 For TICO with Mail-interface

1. If no mail is send: The mail-server may be set incorrectly in the configuration file.
2. If no output is included to the mail: Check the commandline parameters, the files that shall be attached to the mail should be given as parameter with a name they should be given (`-coord filename`, `-glimmer filename`, `-gff filename`, see 6.1.1, p. 9).

9 Visualization of the Weights (since TICO2.1)

Since version 2.1 the weights matrix calculated by TICO is written to the file `tico.weights` in the output directory. The matrix can be visualized with the MATLAB® compiler generated tool `WeightsVis` which is provided in TICO2.1 complete package. The tool also is provided separately as a tarball which should be unpacked in the TICO-home directory or as independent »complete package« version.

The visualization of the positional weights (see figure) is realized in the form of a colored scheme. The colors represent the level of the weights of a trinucleotide at the respective position. High positive weights produce deep red areas in the plot, high negative weights produce deep blue ones, intermediate weights are represented by orange, yellow and green areas. A color scale displays the colors with the associated weight values.

The positions correspond to those in the *extract window* (see 6.1.1 p. 9), which are aligned to the position of the candidate TIS denoted at position 0. The positions with negative value indicate the upstream region, the positions with positive value indicate the downstream region, respectively. Note that the last two positions of the sequences are not considered in the evaluation for

they do not represent a trimer. So if for example the default values are used for up- and downstream extraction, the example sequences (output in `tis.seq` and `tis.altseq`) have the length 60, with the TIS candidates at position 31 (giving the first position the index 1). For the evaluation the trimer occurrences are counted for position 1-58. Additionally, to exclude boundary effects, the weights of the first and the last three positions are neglect. The latter positions are part of the weights matrix but are excluded from the visualization. So the visualization of a weights matrix calculated for the settings described above would show the position -27..0..24.

`WeightsVis` can be started from a script, adapting the necessary paths and setting some default parameters or the program may be called directly.

During invocation a flag with value `0` or `1` may be given to the script. The default value of the flag is `0`. If the flag has value `1` the trinucleotide with maximum positive score is marked in the visualized matrix. Additionally the title and the labels of the axis may be altered in the starter script or during the direct invocation of the program.

Syntax for invocation with the starter script:

```
vis weights-file [max-flag]
```

Example for invocation with the `bash`-script:

```
vis /var/tmp/5793472968765/tico.weights
```

Example for invocation with the `batch`-script:

```
vis "c:\my documents\758943764892\tico.weights"
```

Note for invocation with the batch-script: The `maxFlag` cannot be given in the commandline, but may be altered in the batch script (`vis.bat`). **Syntax for invocation without the starter script:**

```
WeightsVis oligos figure-title X-Label Y-Label weights-file max-flag
```

Example for invocation without the starter script:

```
WeightsVis ./oligos 'TICO-Weights' 'Position' 'Trinucleotide' ../123/tico.weights 0
```

Example for invocation with the `batch`-script:

```
vis.bat c:\var\tmp\5793472968765\tico.weights
```

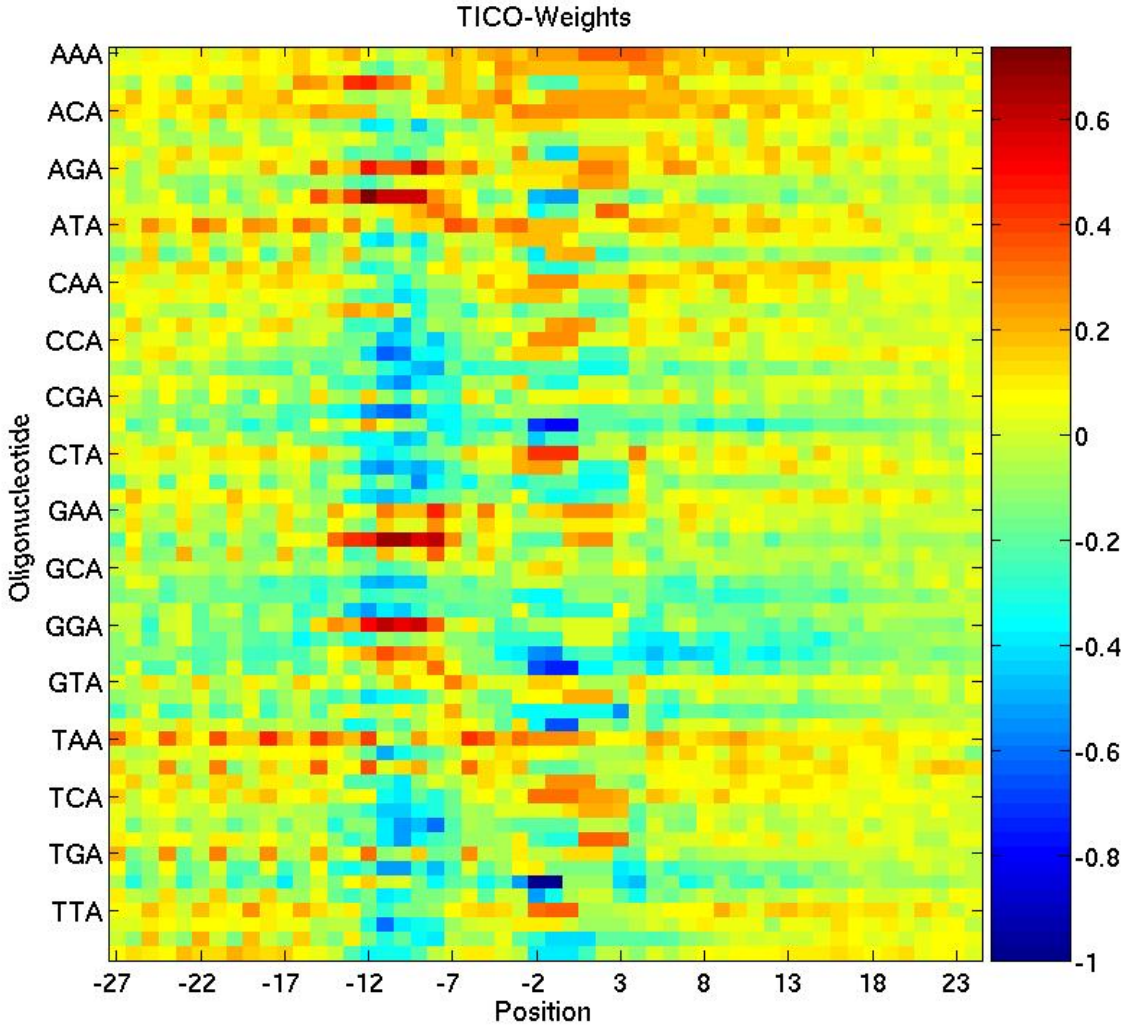

Figure 1: Visualization of the weights matrix calculated for *E. coli* using the default settings.

10 License terms

The complete package is free for academic use. The MATLAB®-compiler generated files are distributed under the MathWorks, Inc Software License, which is included in the package.

The Java classes of the user interface will be OpenSource, when the code is cleaned and completely documented.

11 Links

TICO	
	TICO home – http://tico.gobics.de/
	Göttingen Bioinformatics – http://gobics.de/
	University of Göttingen – http://www.uni-goettingen.de/
Java	
	Sun Java Home – http://java.sun.com/
	Java 1.5 Documentation - http://java.sun.com/j2se/1.5.0/docs/api
	Java Activation – http://java.sun.com/products/javabeans/glasgow/jaf.html
	Java Mail – http://java.sun.com/products/javamail/
	Log4J Home – http://logging.apache.org/log4j/
Others	
	MathWorks Home – http://www.mathworks.com/
	The Sanger Institute – http://www.sanger.ac.uk/
	GFF specification – http://www.sanger.ac.uk/Software/formats/GFF/GFF_Spec.shtml
	GLIMMER Home – http://www.tigr.org/software/glimmer/

References

[1] M. Tech, N. Pfeifer, B. Morgenstern, and P. Meinicke. Tico: A tool for improving predictions of prokaryotic translation initiation sites. *Bioinformatics*, 2005.

[2] P. Meinicke, M. Tech, B. Morgenstern, and R. Merkl. Oligo kernels for datamining on biological sequences: A case study on prokaryotic translation initiation sites. *BMC Bioinformatics*, 5(169), 2004.

[3] A. L. Delcher, D. Harmon, S. Kasif, O. White, and S. L. Salzberg. Improved microbial gene identification with GLIMMER. *Nucleic Acids Res*, 27(23):4636–4641, 1999.

[4] K. Rutherford, J. Parkhill, J. Crook, T. Horsnell, P. Rice, M.-A. Rajandream, and B. Barrell. Artemis: sequence visualisation and annotation. *Bioinformatics*, 16(10):944–945, 2000.