eCAMBer user's manual

Release 1.4

Michal Wozniak

20 Dec 2013

Contents

1 About the software **1**
- 1.1 Background . . . 1
- 1.2 Availability . . . 1
- 1.3 About the authors . . . 2
- 1.4 Installation . . . 2
- 1.5 Design of eCAMBer . . . 2
- 1.6 Dataset files . . . 4

2 Usage manual **5**
- 2.1 Parameters . . . 5
- 2.2 Preparetion of input annotations . . . 6
  - Download of input annotations . . . 6
  - Prodigal input annotations . . . 6
- 2.3 Phase 1: The closure procedure . . . 6
- 2.4 Phase 2: refinement, TIS voting and clean up procedures . . . 7
  - Step 0: The sequence consolidation graph and multigene clusters . . . 7
  - Step 1: Refinement procedure . . . 7
  - Step 2: TIS voting procedure . . . 7
  - Step 3: clean up procedure . . . 7
- 2.5 Generating output . . . 8

3 Examples **8**
- 3.1 2 strains of M. tuberculosis (included in eCAMBer package) . . . 8
- 3.2 65 strains of M. tuberculosis (from PATRIC) . . . 8
- 3.3 20 strains of E. coli (from ColiScope) . . . 9

1 About the software

1.1 Background

˜Due to the fast progress in sequencing high-throughput technologies, the number of bacterial sequences grows rapidly (*Loman et al., Nature Reviews Microbiology, 2012*). This enables new interesting comparative genome analysis. On the other hand, while there are many annotation tools dedicated to a single genome, there are relatively few tools supporting comparative annotation and analysis of multiple bacterial genomes (*Poptsova et al., 2010*). Moreover, we

observe a lot of inconsistencies in the genome structure annotations among bacterial strains (*John Dunbar et al. BMC Genomics, 12(1):125, 2011*). This inconsistency is a frustrating impedance to effective comparative genomic analysis of bacterial strains in promising applications such as gaining insights into bacterial drug resistance.

We have developed *eCAMBer*, a tool efficiently supporting comparative analysis of multiple bacterial strains within the same species (paper submitted to Bioinformatics 2013). In this work we achieve three major goals. First, eCAMBer is a highly optimized revision of our earlier tool, CAMBer (*Wozniak M, Wong L, Tiuryn J. BMC Genomics 2011*), scaling it up for significantly larger datasets comprising hundreds of bacterial strains. Second, eCAMBer is capable of identifying and resolving annotation inconsistencies. Finally, eCAMBer improves the overall quality of annotations.

eCAMBer works in two phases. First, it unifies annotations of closely related species by homology transfer in the *closure procedure*. Second, eCAMBer identifies and tries to resolve annotation inconsistencies in the three procedures: (i) *the refinement procedure* for splitting homologous gene families into orthologous gene clusters; (ii) *refinement procedure* for selecting the most reliable TIS; (iii) *TIS voting procedure* for removal of gene clusters that are likely to be annotation errors propagated during the *closure procedure*.

Details of the methodology are described in the eCAMBer paper: (submitted to Bioinformatics, 2013).

1.2 Availability

This software is an open source application (GPL 3 license). Sources are available at the project website:

`http://bioputer.mimuw.edu.pl/ecamber`

Please don't hesitate to contact us with any comments and suggestion or if you are interested in co-developing this software.

1.3 About the authors

This software was implemented by Michal Wozniak. All authors contributed to design of the method and analysis of results. Project idea and guidance came from prof. Limsoon Wong (National University of Singapore) and prof. Jerzy Tiuryn (University of Warsaw).

Corresponding author: Michal Wozniak (`m.wozniak@mimuw.edu.pl`)

1.4 Installation

This software is written in Python, thus Python 2 or 3 is required to run eCAMBer. The software does not need a classical installation, you only need to download and extract the zip package (versions for Windows, Linux and Mac OS are available) from the project website:

`http://bioputer.mimuw.edu.pl/ecamber`

1.5 Design of eCAMBer

The software is designed as a set of executable scripts written in Python, which can be run via the console script `ecamber.py`.

Figure 1 presents the hierarchy of folders in eCAMBer open in Eclipse.

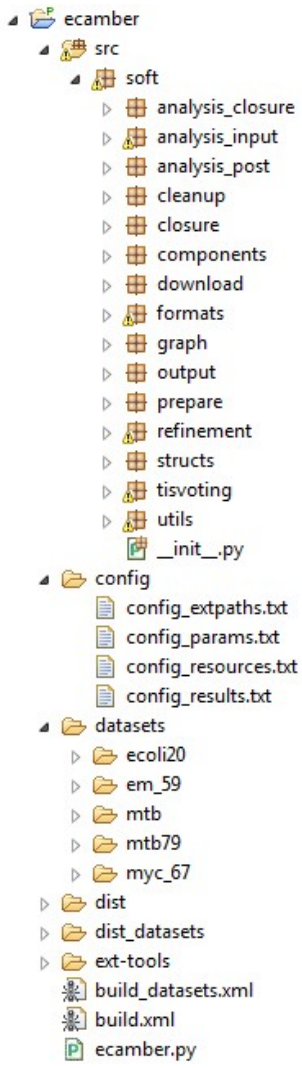

Figure 1: Hierarchy of folders in eCAMBer open in Eclipse.

The hierarchy of folders in eCAMBer (including the dataset folders):

- `ecamber/` — the main eCAMBer folder with the executable files
  - `ecamber.py` — the main console script to run eCAMBer
  - `ext-tools/` — BLAST+ or BLASTALL executable files
  - `config/` — configuration files:
    - `config_extpaths.txt` — BLAST paths configuration file
    - `config_resources.txt` — configuration of the paths for datasets
    - `config_params.txt` — configuration of the eCAMBer parameters
    - `config_results.txt` — configuration of the paths with experiment results
  - `datasets/` — input datasets
    - `dataset1/` — the folder with `dataset1` input files (for example `dataset1=ecoli20`)
    - `dataset2/` — the folder with `dataset2` input files (for example `dataset2=mtb`)

- src/
  - soft — the source code of eCAMBer
  - soft/analysis_closure — scripts for analysing output of the closure procedure
  - soft/analysis_input — scripts for analysing dataset input
  - soft/analysis_post — scripts for analysing results of post-processing procedures
  - soft/download — scripts for downloading the bacterial genomes and annotations from PATRIC
  - soft/formats — scripts for formatting the downloaded data into eCAMBer input format
  - soft/prepare — scripts for preparing input for the 1st iteration of the closure procedure
  - soft/closure — scripts for computing the closure procedure
  - soft/graph — scripts for constructing the sequence consolidation graph
  - soft/components — scripts for computing the homologous gene clusters
  - soft/refinement — scripts for computing the refinement procedure
  - soft/tisvoting — scripts for computing the TIS voting procedure
  - soft/cleanup — scripts for computing the clean up procedure
  - soft/output — contains a script to generate input data for CAMBerVis
  - soft/structs — structures used in eCAMBer
  - soft/utils — implementation of common methods used in eCAMBer

1.6 Dataset files

Input files for eCAMBer consist of genome sequences and annotations for a set of bacterial strains. There is also file 'strains.txt' required to specify the list of strains. The structure of input files for each dataset:

- anns_input — contains a set of annotation files downloaded from PATRIC
- anns_parsed — contains a set of annotation files downloaded from PATRIC and parsed as eCAMBer input
- genomes — contains a set of FASTA files
- strains.txt — list of strains

Each line of an annotation file should be in the following tab-separated format:

```
gene_id start end strand gene_name contig_id
```

The last two pieces of information (gene_name and contig_id) are optional. Below we present an an example of the first 7 lines from the annotation file for the E. coli strain K-12 MG1655.

```
ECK0001   190   255    +   thrL
ECK0002   337   2799   +   thrA
ECK0003   2801  3733   +   thrB
ECK0004   3734  5020   +   thrC
ECK0005   5234  5530   +   yaaX
ECK0006   6459  5683   -   yaaA
ECK0007   7959  6529   -   yaaJ
```

The partial results of eCAMBer computations for each dataset are organized in the following structure:

- blast — BLAST results
- cambervis — eCAMBer output formatted as CAMBerVis input

- `output` — eCAMBer annotation output
- `genomes_dbs` — genomes formetted as BLAST databases
- `genomes_tmp` — FASTA genomes downloaded from PATRIC
- `results/exp-refseq-20-80-1e-10-YES-Q/` — partial results for a combination of parameters
  - `gene_names` — gene names
  - `excluded` — gene annotations excluded from the analysis
  - `analysis` — results of automatic analysis of results
  - `blast-tmp` — temporal BLAST results
  - `phase1` — results for the closure procedure (phase 1)
  - `phase2` — results for the phase 2

Each line of the files with the result annotations has the following tab-delimited format:

`mg_cluster_id mg_id mg_lengths`

Here, `mg_cluster` denotes the multigene cluster id, `mg_id` denotes the multigene id (`end strand contig_-id`), `mg_lengths` denote different lengths of different putative genes, sharing the same end, but corresponding to different putative TISs. Additionally, a star, $\star$ in front of the number representing the Orf length, denotes that its corresponding TIS was originally annotated. Similarly, a hash #, in front of the number representing the Orf length denotes, that the TIS was selected during the TIS voting procedure.

An example:

```
8204 255 + K-12_MG1655 ECK0001 #⋆66:ATG
1277 2799 + K-12_MG1655 ECK0002 #⋆2463:ATG
1153 3733 + K-12_MG1655 ECK0003 #⋆933:ATG
207 5020 + K-12_MG1655 ECK0004 #⋆1287:ATG
8028 5530 + K-12_MG1655 ECK0005 288:ATG #⋆297:GTG
2276 5683 - K-12_MG1655 ECK0006 #⋆777:ATG
3635 6529 - K-12_MG1655 ECK0007 #⋆1431:ATG
(...)
5850 1702700 + K-12_MG1655 ECK1620 ⋆126:ATG #201:TTG
```

2 Usage manual

2.1 Parameters

There are two ways to specify parameters in eCAMBer:

- -<key> <value>
- <key>=<value>

The two following commands (to download annotations from PATRIC) are equivalent:

- `>python ecamber.py -a d -w 4`
- `>python ecamber.py a=d w=4`

The following table presents description of eCAMBer parameters:

Parameter	Default value	Example values (comma separated)	Description
a	–	d,pr,f,ph1,ph2,out	action
w	1	1,2,3,...	# of threads (workets used)
d	–	mtu2,ecoli20	dataset
as	patric	patric,refseq,prodigal	annotation source
ep	exp	any string	prefix of the results folder
m	100	1...100	maximal number of iterations
be	10^{10}	0.0 ... 1.0	BLAST e-value running parameter
te	100	0.0 ... 1.0	BLAST e-value threshold parameter
lcm	Y	Y,N	BLAST Low Complexity Masking
hssp	Y	Y,N	HSSP correction of PID
pid	80	30...100	PID threshold for acceptable hits
step	–	9,1,2,3	iteration of phase 2
tvc	C	1 ... n	(C)onservative; (R)elaxed
tvr	0.8	0.0 ... 1.0	conservation threshold
cur	0.3334	0.0 ... 1.0	annotation ratio
cup	0.3334	0.0 ... 1.0	conservation ratio
cul	150	30 ... 500	the median multigene length
cuv	0.5	0.0 ... 1.0	the overlapping ratio

2.2 Preparetion of input annotations

The input files for a dataset can be prepared manually according to the input format described above. However, eCAMBer supports two convinient ways to prepare input annotations: (1) automatic download of annotations for the PATRIC database; and (2) usage of Prodigal to generate the input annotations.

Download of input annotations

eCAMBer supports downloading of the input annotations from the PATRIC database (both PATRIC and RefSeq annotations can be downloaded), available under this ftp link:

`http://brcdownloads.vbi.vt.edu/patric2/`

In order to download the input dataset from PATRIC run:

```
>python ecamber.py -a d -w 4
```

Next, to format the downloaded data into eCAMBer input format:

```
>python ecamber.py -a f -d dir -w 4
```

Here, `dir` denotes the name of the folder to which the dataset files were downloaded (chosen during the previous phase).

Prodigal input annotations

Alternatively, input annotations can be prepared using Prodigal:

```
>python ecamber.py -a pr -d dir -w 4
```

2.3 Phase 1: The closure procedure

In order to run the closure procedure (phase 1), on Prodigal annotations, invoke that command:

```
>python ecamber.py -a ph1 -d dir -w 4 -as prodigal
```

You can also execute the closure procedure step by step by running the two commands below. The first command prepares the input data for the 1st iteration of the closure procedure:

```
>python ecamber.py -a p -d dir -w 4 -as prodigal
```

Then, the following command runs the closure procedure:

```
>python ecamber.py -a c -d dir -w 4 -as prodigal -m 100
```

Here, the default `-as prodigal` parameter denotes the source of annotations, acceptable values comprise:

- `patric` — PATRIC annotations, read from the folder `anns_parsed/patric/`
- `refseq` — RefSeq annotations, read from the folder `anns_parsed/refseq/`
- `prodigal` — Prodigal annotations, read from the folder `anns_parsed/prodigal/`

2.4 Phase 2: refinement, TIS voting and clean up procedures

In order to run the closure procedure (phase 1), invoke that command:

```
>python ecamber.py -a ph2 -d dir -w 4 -as prodigal
```

You can also perform the procedure step by step by running subsequenctly the commands listed below.

Step 0: The sequence consolidation graph and multigene clusters

In this step eCAMBer constructs the sequence consolidation graph. and identifies the homologous clusters as connected components of the multigene consolidation graph. To perform this step, run:

```
>python ecamber.py -a cc -d dir -w 4 -as prodigal
```

Step 1: Refinement procedure

In this procedure eCAMBer splits multigene clusters with more than one multigene per strain. To perform this step, run:

```
>python ecamber.py -a rf -d dir -w 4 -as prodigal -step 0
```

The parameter `COMP_IT` denotes the current iteration within the second phase of eCAMBer. It should be incremented by one with every run of one of the three procedures: refinement, TIS voting, or clean up.

Step 2: TIS voting procedure

In the TIS voting procedure eCAMBer, for each multigene, selects one TIS (originally annotated or predicted), that is most often annotated among the strains. To perform this step, run:

```
>python ecamber.py -a tv -d dir -w 4 -as prodigal -step 1
```

Step 3: clean up procedure

In the clean up procedure, eCAMBer, removes multigene clusters, that are likely to be annotation errors propagated during the closure procedure. To perform this step, run:

```
>python ecamber.py -a cu -d dir -w 4 -as prodigal -step 2
```

2.5 Generating output

The following command generates output annotations and input for CAMBerVis:

```
>python ecamber.py -a out -d dir -w 4 -as patric -step 3
```

3 Examples

3.1 2 strains of M. tuberculosis (included in eCAMBer package)

We prepared one small example dataset incorporated into the eCAMBer package. The input files for the dataset are in the folder `ecamber/datasets/mtu2`. To perform eCAMBer just execute subsequently the following commands. Formatting the input:

```
>python ecamber.py -a f -w 4 -d mtu2
```

The closure procedure (phase 1) of eCAMBer.

```
>python ecamber.py -a ph1 -w 4 -d mtu2 -as patric
```

Phase 1 of eCAMBer (computing of gene clusters and refining annotations):

```
>python ecamber.py -a ph2 -w 4 -d mtu2 -as patric
```

Saving the output annotations to the folder `ecamber/datasets/output` and input for CAMBerVis to the folder `ecamber/datasets/cambervis`:

```
>python ecamber.py -a out -w 4 -d mtu2 -as patric
```

3.2 65 strains of M. tuberculosis (from PATRIC)

Here we present a step-by-step example to apply eCAMBer to analize a dataset of 67 strains of *M. tubeculosis* strains, with genomes available in the PATRIC database.

```
>python ecamber.py -a d -w 4
```

Then, eCAMBer will connect to PATRIC and will provide you options to choose the database release:

```
Platform: Linux
PATRIC: http://brcdownloads.vbi.vt.edu/patric2/
download_patric_genomes.py: downloads genome sequences and annotations from the PATRIC database.
0: genomes/      date: 16-Mar-2013 22:19 [current release]
1: genomes.Feb2012/      date: 05-Mar-2012 01:34
2: genomes.Jul2011/      date: 27-Jul-2011 10:03
3: genomes.Jun2011/      date: 08-Jun-2011 06:16
4: genomes.May2012/      date: 24-Jul-2012 21:19
5: genomes.Nov2012/      date: 24-Feb-2013 03:43
6: genomes.Oct2011/      date: 02-Dec-2011 11:07
7: genomes.Sept2012/     date: 05-Oct-2012 06:50
Select version's index: (default: 0 -current release):
```

In this example we select the database from 24-Feb-2013 (number 5).

Next, eCAMBer asks to select the bacterial species. In this example we choose the default option — *M. tuberculosis*. A list of all available *M. tuberculosis* strains will be displayed. We confirm download by 'd'. A dataset folder with default name 'mtu65' will be created at: `ecamber/datasets/mtu65`.

Next, we format the downloaded dataset as eCAMBer input:

```
>python ecamber.py -a f -w 4 -d mtu65
```

Next, we compute the closure procedure (phase 1) using annotations from PATRIC. Use can also use annotations from RefSeq by specyfying ANN_SOURCE=R:

```
>python ecamber.py -a ph1 -w 4 -d mtu65 -as patric
```

Next, run the computing and cleaning of the gene clusters (phase 2):

```
>python ecamber.py -a ph2 -w 4 -d mtu65 -as patric
```

Run this command to generate the input for CAMBerVis:

```
>python ecamber.py -a out -w 4 -d mtu65 -as patric
```

Copy the created folder `ecamber/mtu65/cambervis` into the folder `camber-vis/examples`. You may also change its name, for example: `camber-vis/examples/mtu65`.

3.3 20 strains of E. coli (from ColiScope)

We download the dataset (with annotations formatted as eCAMBer input) from the eCAMBer project webpage. We unzip the packege and put here: `ecamber/datasets/ecoli20`.

Next, run subsequently the following commands:

```
>python ecamber.py -a f -w 4 -d ecoli20 -as coliscope
```

```
>python ecamber.py -a ph1 -w 4 -d ecoli20 -as coliscope
```

```
>python ecamber.py -a ph2 -w 4 -d ecoli20 -as coliscope
```

```
>python ecamber.py -a out -w 4 -d ecoli20 -as coliscope
```

The last commnad generates input for CAMBerVis. Copy the created folder `ecamber/datasets/ecoli20/cambervis` into the folder `camber-vis/examples`. You may also change its name, for example: `camber-vis/examples/ecoli20`.