BMGE User Guide

Block **M**apping and **G**athering using **E**ntropy

[Version 1.1] *March 2011*

by Alexis Criscuolo & Simonetta Gribaldo

```
ftp://ftp.pasteur.fr/pub/GenSoft/projects/BMGE/
http://mobyle.pasteur.fr/cgi-bin/portal.py
```

Criscuolo A, Gribaldo S (2010) BMGE (Block Mapping and Gathering with Entropy): selection of phylogenetic informative regions from multiple sequence alignments. *BMC Evolutionary Biology* 10:210.

BMGE (Block Mapping and Gathering with Entropy) is a program that selects regions in a multiple sequence alignment that are suited for phylogenetic inference. BMGE selects characters that are biologically relevant, thanks to the use of standard similarity matrices such as PAM or BLOSUM. Moreover, BMGE provides other character- or sequence-removal operations, such stationary-based character trimming (that provides a subset of compositionally homogeneous characters) or removal of sequences containing a too large proportion of gaps. Finally, BMGE can simply be used to perform standard conversion operations among DNA-, codon-, RY- and amino acid-coding sequences.

Installation and Execution

BMGE runs on all operating systems that run Java 1.6 or more. Sun Java is freely available and presents very good performances. If Sun Java 1.6 (or higher) is not installed on your computer, select the last update of JDK/JRE-6 (or higher) at the following URL:

`http://java.sun.com/products/archive/`

and follow the 'Installation Instructions' to install the JRE (Java Runtime Environment) on your computer. This allows BMGE to be run. You can also install the JDK (Java Development Kit) that allows the BMGE source code to be compiled and run.

On computers with Sun Java 1.6 (or higher) installed, you just have to open a terminal, move to the directory `BMGE/` containing the executable jar file `BMGE.jar`, and launch it with the following command-line:

```
java -jar BMGE.jar [options]
```

For example, you can get help by using the following command-line:

```
java -jar BMGE.jar -?
```

When the multiple sequence alignment to be trimmed is very large, it could happen that the virtual machine `java` do not allocate enough memory and stops with an error message starting like

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

In this case, the `java` option `-Xmx` can be used to increase the maximum size of the memory allocation pool in MB; for example, for a maximum size of 500MB:

```
java -Xmx500M -jar BMGE.jar [options]
```

It is not impossible (albeit uncommon) that the executable jar file `BMGE.jar` do not work. In this case, you must compile the BMGE source code. To do so on Linux System, you must have the Sun JDK 1.6 (or higher) installed (see above); then move to the `BMGE/src/` directory, and launch the jar builder:

```
chmod a+x JarMaker.sh
./JarMaker.sh
```

to create a new executable jar file `BMGE.jar`.

Quick Start

All BMGE input files are multiple sequence alignment files (named here `msa`). To perform an entropy-based trimming (with default options) on a multiple alignment of amino acid sequences in FASTA format, use the following command-line:

```
java -jar BMGE.jar -i msa.faa -t AA -o tmsa.phy
```

This will create the file `tmsa.phy` containing the trimmed multiple sequence alignment (`tmsa`) in PHYLIP sequential format. You can also use different `-o` options to request output in alternative formats; for example, the following command-lines:

```
java -jar BMGE.jar -i msa.faa -t AA -of tmsa.faa
java -jar BMGE.jar -i msa.faa -t AA -on tmsa.nex
java -jar BMGE.jar -i msa.faa -t AA -oh msa.html
```

will create the files `tmsa.faa` in FASTA format, `tmsa.nex` in NEXUS format and `msa.html` in HTML format, respectively. There is no limitation to the number of different `-o` options; for example:

```
java -jar BMGE.jar -i msa.faa -t AA -o tmsa.phy -of tmsa.faa -oh msa.html
```

If you wish to trim a multiple alignment of DNA sequences, you must modify the `-t` option:

```
java -jar BMGE.jar -i msa.fna -t DNA -o tmsa.phy
```

It is also possible to regard nucleotide sequences as codon ones with the `-t` option:

```
java -jar BMGE.jar -i msa.fco -t CODON -o tmsa.phy
```

For more practical command-line, go to the Current Usage section (see below).

File Formats and Sequence Coding

Input File Formats `-i` *`infile`*

BMGE uses FASTA or PHYLIP sequential format for input. These are plain text files from any operating systems (Unix, Windows, Mac). There is no limit on the length of the alignment. There is also no limit on the length of the label of a sequence (i.e. its FASTA annotation line), although a too long label (e.g. more than 100 letters) will be truncated if the output format is PHYLIP sequential.
There is a unique option to set the input file (i.e. `-i`). By default, BMGE looks at the first character of the first non-empty line inside the input file. If this character is '`>`', then BMGE reads the input file as a FASTA formatted file; otherwise, BMGE reads it as a PHYLIP sequential formatted file.

Input Sequence Coding `-t [AA,DNA,CODON]`

One must always set the input sequence coding with option `-t`:

- amino acid sequences `-t AA`
- nucleotide sequences `-t DNA`
- codon sequences `-t CODON`

Both standard single-letter amino acid and nucleotide alphabets are used by BMGE (e.g. IUPAC-IUB 1970, 1972; see Table below). When using amino acid sequences, degenerated character states `B` and `Z` are understood by BMGE; similarly, degenerated nucleotide characters are also understood (see Table below). The character state `X` is understood to be any of the 4 or 20 character states when using as input nucleotide or amino acid sequences, respectively. Dashes (i.e. '`-`') are understood as gaps, whereas dots (i.e. '`.`'), as any other single letter that are not inside standard alphabets, are considered as unknown character state (i.e. '`?`').
Nucleotide sequences can be set as codon ones. In this case, each successive nucleotide character triplet is considered as one codon character. This particular `-t` option allows converting an alignment of codon sequences into its corresponding amino acid sequence (following the universal genetic code; see below).

Output File Formats `-o` *`outfile`* `-c` *`outfile`*

BMGE can output the (trimmed) multiple sequence alignment into several formats:

- selected characters in PHYLIP sequential format `-op`
- selected characters in FASTA format `-of`
- selected characters in NEXUS format `-on`
- alignment and information in HTML format `-oh`

If input sequences are in FASTA format with NCBI-formatted annotation lines, e.g.

```
>field1|field2|field3|field4| field5 [field6]
```

then options `-opp` and `-onn` allow outputting in PHYLIP and NEXUS format, respectively, but with sequence named by `field6` only (which is generally the taxon name). Moreover, the options `-oppp` and `-onnn` allow naming sequences by `field6______field4` ; knowing that `field4` is generally an accession number, these options lead to PHYLIP or NEXUS files where each sequence is labelled as a taxon name and an accession number. If these output options (i.e. `-opp`, `-oppp`, `-onn`, `-onnn`) are selected with non NCBI-formatted sequences, then BMGE uses the whole annotation line as sequence name (which corresponds to the `-op` and `-on` options).
By default (i.e. `-o`), BMGE uses the output option `-oppp`.
The HTML output format (`-oh`) writes the initial alignment with a graphical representation of the smoothed entropy-like score and the gap rates across characters. HTML output file also contains the

indexes of the selected and removed characters.
The non-selected (i.e. removed) characters can be outputted in a similar way by replacing `-o` by `-c` in the previous options (i.e. `-c`, `-cp`, `-cpp`, `-cppp`, `-cf`, `-cn`, `-cnn`, `-cnnn`). The output option `-ch` is not allowed.
There is no limitation to the number of different output options; for example:

```
java -jar BMGE.jar -i msa.faa -t AA -of tmsa.faa -cf cmsa.faa -oh msa.html
```

Output Sequence Recoding `-o` *`outfile`* `-c` *`outfile`*

BMGE is able to perform sequence coding conversions:

• amino acid-coding	`-oaa`
• nucleotide-coding	`-odna`
• codon-coding	`-oco`
• RY-coding (Phillips et al. 2004)	`-ory`

Only two conversions are impossible depending on the initial sequences: from nucleotide (`-t DNA`) to amino acid (`-oaa`) sequences or to codon (`-oco`) sequences. The other conversions performed by BMGE (and the corresponding option) are listed in the following table:

		input coding		
		amino acid	nucleotide	codon
		`-t AA`	`-t DNA`	`-t CODON`
output coding	amino acid	`-o / -oaa`	.	`-oaa`
	nucleotide	`-oco`	`-o / -odna`	`-o / -oco`
	codon	`-oco`	.	`-o / -oco`
	RY	`-ory`	`-ory`	`-ory`

Any of the three codon position(s) can be selected with the output option `-oco` (or `-ory` when input is amino acid or codon sequences) by setting the position index(es) just after the options `-oco` or `-ory`. There is 7 different combinations: `-oco1`, `-oco2`, `-oco3`, `-oco12`, `-oco13`, `-oco23`, and `-oco123` (the last `-oco123` option is identical to the default `-oco` option). Same with `-ory`. For example, the following command-line:

```
java -jar BMGE.jar -i msa.fco -t CODON -o tmsa1.phy -ory12 tmsa2.phy
```

allows the codon-based alignment `msa.fco` to be trimmed with default options, and leads to two output files:

- `tmsa1.phy`, the codon characters selected by BMGE;
- `tmsa2.phy`, the RY-coding of the two first positions (`12`) of each codon.

All these output coding options are compatibles with the previous output format options (`-o` and `-c`); for example, if one has a file `msa.faa` containing an alignment of codon sequences in FASTA format, then the command-line

```
java -jar BMGE.jar -i msa.fco -t CODON -o tmsa.phy -oaaf tmsa.faa -onnry tmsa.nex
```

leads to three output files:

- `tmsa.phy`, the selected codon characters in PHYLIP sequential format (`-o`);
- `tmsa.faa`, the selected codon characters converted into amino acids (`aa`) in FASTA format (`f`);
- `tmsa.nex`, the RY-coding (`ry`) of the selected codon characters in NEXUS format with only species names as sequence names (`nn`).

Character state coding used by BMGE

Nucleotide	1-letter symbol
Adenosine	A
Guanine	G
Cytosine	C
Thymine	T

Degenerated nucleotide	1-letter code	Meaning
Methyl	M	A or C
Purine	R	A or G
Weak (3H bonds)	W	A or T
Strong (3H bonds)	S	C or G
Pyrimidine	Y	C or T
Keto	K	G or T
	B	not A
	D	not C
	H	not G
	V	not T
Any	N or X	one of the 4

Degenerated amino acid	1-letter code	Meaning	Degenerated codon
Aspartate	B	N or D	RAY
Glutamate	Z	Q or E	SAR
Any	X	one of the 20	XXX

Amino acid	1-letter code	Degenerated codon
Alanine	A	GCX
Arginine	R	MGX
Asparagine	N	AAY
Aspartic acid	D	GAY
Cysteine	C	TGY
Glutamine	Q	CAR
Glutamic acid	E	GAR
Glycine	G	GGX
Histidine	H	CAY
Isoleucine	I	ATH
Leucine	L	YTX
Lysine	K	AAR
Methionine	M	ATG
Phenylalanine	F	TTY
Proline	P	CCX
Serine	S	WSX
Threonine	T	ACX
Tryptophan	W	TGG
Thyrosine	Y	TAY
Valine	V	GTX

BMGE Command Line Options

Similarity Matrices `-m BLOSUM`*n* `-m DNAPAM`*n*`:`*r* `-m ID`

For each character, BMGE computes a score mainly determined by the entropy induced by the respective proportion of each residue (Criscuolo and Gribaldo 2010). To estimate realistic scores that take into account biologically relevant substitution processes, BMGE weights the entropy estimation with substitution matrices.
With amino acid input sequences (`-t AA`), BMGE uses by default the popular BLOSUM62 matrix (Eddy 2004). However, one can use another BLOSUM matrix with the option `-m`; for example:

```
java -jar BMGE.jar -i msa.faa -t AA -m BLOSUM50 -o tmsa.phy
```

The option `-m` can be used with the 15 estimated BLOSUM matrices, i.e. `BLOSUM30`, `BLOSUM35`, `BLOSUM40`, `BLOSUM45`, `BLOSUM50`, `BLOSUM55`, `BLOSUM60`, `BLOSUM62`, `BLOSUM65`, `BLOSUM70`, `BLOSUM75`, `BLOSUM80`, `BLOSUM85`, `BLOSUM90`, `BLOSUM95` (Henikoff and Henikoff 1992). The trimming is progressively more stringent as the BLOSUM index increases (e.g. `BLOSUM95`); reciprocally, the trimming is progressively more relaxed as the BLOSUM index is lower (e.g. `BLOSUM30`). In practice, it is recommended to use BLOSUM95 with closely related sequences, and BLOSUM30 with distantly related sequences.
For nucleotide input sequences (`-t DNA`), BMGE uses PAM matrices with a fixed transition/transition ratio (States et al. 1991). BMGE can be used with all possible PAM matrices, from the most stringent (i.e. `-m DNAPAM1`) to highly relaxed ones (e.g. `-m DNAPAM500`). There is no limitation for the PAM matrices (the PAM indexes must be positive integers), but incoherent entropy scores are expected with too high PAM indexes (e.g. `-m DNAPAM123456`). It is also possible to indicate a transition/transversion ratio to better define the PAM matrices. For example, if one wishes to estimate entropy-like scores with a (relaxed) PAM-250 matrix and a transition/transversion ratio of 4, then one uses the following command-line:

```
java -jar BMGE.jar -i msa.fna -t DNA -m DNAPAM250:4 -o tmsa.phy
```

For a (very stringent) PAM-5 matrix with a transition/transversion ratio of 1, the following command lines are equivalent:

```
java -jar BMGE.jar -i msa.fna -t DNA -m DNAPAM5:1 -o tmsa.phy
java -jar BMGE.jar -i msa.fna -t DNA -m DNAPAM5 -o tmsa.phy
```

By default with nucleotide sequences, BMGE uses the PAM-100 matrix with a transition/transversion ratio of 2 (i.e. `-m DNAPAM100:2`).
If input sequences are set as codons (`-t CODON`), BMGE performs a conversion into amino acid sequences (following the universal genetic code) and uses BLOSUM matrices to estimate the entropy-like score for each codon character. So, with option `-t` set as `CODON`, one can set the option `-m` only with BLOSUM matrices. However, it is possible to use PAM matrices by setting the option `-t` as `DNA`, but the trimmed alignment will be no longer a codon one.
Finally, it is also possible to use the identity matrix with option `-m ID` (or `-m PAM0`) with any sequence types (`-t AA`, `DNA` or `CODON`).

Sliding Windows Size `-w` *`odd_integer`*

All entropy-score estimated for each character are smoothed by BMGE by using a sliding window. The size of the sliding window can be modified with the option `-w` (which is set as `-w 3` by default). It is not recommended to increase this size (which must be odd). However, when set to 1, no averaging operations will be performed by BMGE. This can be useful to select precise characters instead of regions (see below).

Entropy Score Cut-off `-h` *`max`* `-h` *`min`*`:`*`max`*

Following the smoothing operation of the entropy-like score values across characters, BMGE selects characters associated with a score value below a fixed threshold. This cut-off is set to 0.5 by default, but it can be modified with the option `-h`. For example, a stringent trimming on amino acid sequence alignment is performed with the following command line:

```
java -jar BMGE.jar -i msa.faa -t AA -m BLOSUM90 -h 0.4 -o tmsa.phy
```

Indeed, with the BLOSUM90 matrix, BMGE estimates stringent entropy-like scores, but it only selects the characters with a score smaller than 0.4. However, it is strongly recommended to use the option `-m` rather than the option `-h` to obtain biologically-relevant stringent or relaxed character trimming.
The option `-h` can also be used to set a min-threshold. For example, to select characters with a score drawn between 0.2 and 0.4, the following command line can be used:

```
java -jar BMGE.jar -i msa.faa -t AA -m BLOSUM95 -h 0.2:0.4 -o tmsa.phy
```

This option can be useful to remove all constant characters from an alignment, with the following command line:

```
java -jar BMGE.jar -i msa.faa -t AA -w 1 -h 1E-5:1 -o tmsa.phy
```

Indeed, knowing that constant characters have an entropy-like score of 0, this allows selecting all characters with a score greater of equal than 0.00001 (and thus removing all constant characters).

Gap Rate Cut-off `-g` *`col_rate`* `-g` *`row_rate`*`:`*`col_rate`*

BMGE allows characters containing too many gaps to be removed with the option `-g`. By default, BMGE removes all characters with a gap frequency greater than 0.2. For example, to perform default trimming operations and the removal of all characters with more than 5% gaps, use the following command line:

```
java -jar BMGE.jar -i msa.faa -t AA -g 0.05 -o tmsa.phy
```

The option `-g` can also be used to remove sequence(s) from the alignment that contain(s) important proportion of gaps. For example, use the following command line

```
java -jar BMGE.jar -i msa.faa -t AA -g 0.7:0.3 -o tmsa.phy
```

to (i) perform default trimming operations, (ii) remove all characters with more than 30% gaps, and (iii) remove sequence(s) with more than 70% gaps. If at least one sequence is removed, then the whole character-trimming process is re-launched on the initial alignment without the removed sequence(s).

Minimum Block Size `-b` *`integer`*

By default, BMGE only selects regions of size greater than or equal to 5 characters. Use the `-b` option to modify this minimum block size parameter. For example, the following command line

```
java -jar BMGE.jar -i msa.fna -t DNA -m DNAPAM1 -b 20 -o tmsa.phy
```

allows selecting very conserved regions (`DNAPAM1`) of at least 20 nucleotide-long.

Stationary-based Trimming `-s [NO,YES,FAST]`

By setting the option `-s` to `YES` (`NO` by default), BMGE performs another character trimming until the remaining characters are compositionally homogeneous, as assessed by Stuart's (1955) test of marginal homogeneity between each pair of sequences. If an HTML file is created (`-oh`), then all the Stuart's (1955) *p*-values estimated before and after the stationary-based trimming will be written.
It should be stressed that the stationary-based trimming is biased with short alignments (e.g. less than 1,000 character length); consequently, it is more efficient on a supermatrix of characters. Unfortunately, the running time is quite long (e.g. several hours for more than 10,000 amino acid characters). However, by setting the option `-s` to `FAST`, BMGE performs a faster stationary-based character trimming.

Current Usage

Alignment file format conversion

Converting a FASTA format into PHYLIP sequential format:

```
java -jar BMGE.jar -i msa.faa -t AA -h 1 -g 1 -o msa.phy
```

Converting a FASTA or PHYLIP format into NEXUS format:

```
java -jar BMGE.jar -i msa.faa -t AA -h 1 -g 1 -on msa.nex
```

Sequence coding conversion

Converting codon sequences into their amino acid ones in FASTA format:

```
java -jar BMGE.jar -i msa.fco -t CODON -h 1 -g 1 -ofaa msa.faa
```

Obtaining a RY-coding from DNA sequences:

```
java -jar BMGE.jar -i msa.fna -t DNA -h 1 -g 1 -ory msa.phy
```

Gap removing (without entropy-based trimming)

Removing characters with more than 20% gaps:

```
java -jar BMGE.jar -i msa.faa -t AA -h 1 -w 1 -o tmsa.phy
```

Removing characters with more than 10% gaps:

```
java -jar BMGE.jar -i msa.faa -t AA -h 1 -w 1 -g 0.1 -o tmsa.phy
```

Removing sequences with more than 60% gaps:

```
java -jar BMGE.jar -i msa.faa -t AA -h 1 -g 0.6:1 -o tmsa.phy
```

Alignment trimming

Removing constant DNA characters:

```
java -jar BMGE.jar -i msa.fna -t DNA -m ID -h 0.0005:1 -o tmsa.phy
```

Stringent trimming on amino acid sequence alignments:

```
java -jar BMGE.jar -i msa.faa -t AA -m BLOSUM95 -h 0.4 -o tmsa.phy
```

Relaxed trimming on (distantly-related) amino acid sequence alignments:

```
java -jar BMGE.jar -i msa.faa -t AA -m BLOSUM30 -o tmsa.phy
```

Stringent trimming on codon-based alignments:

```
java -jar BMGE.jar -i msa.fco -t CODON -m BLOSUM95 -h 0.4 -o tmsa.phy
```

Stringent trimming on codon-based alignments considered at the nucleotide level:

```
java -jar BMGE.jar -i msa.fco -t DNA -m DNAPAM50 -h 0.4 -o tmsa.phy
```

Stationary-based trimming on DNA sequence alignments:

```
java -jar BMGE.jar -i msa.fna -t DNA -h 1 -g 1 -s YES -o tmsa.phy
```

Previous Versions & Fixed Bugs

[Version 1.1]
The initial stationary-based character trimming was replaced by two new methods: by setting the option `-s` to `YES`, BMGE removes one character during each iteration; by setting the option `-s` to `FAST`, BMGE removes more than one character during each iteration. The first option (`-s YES`) is slow but allows building larger compositionally homogeneous character subset than the method implemented in Version 1.0. The second option (`-s FAST`) is slightly cruder, then fast, but shows quite similar results than the first version (`-s YES`).
When using option `-t CODON`, the FASTA output sequences (`-of`) were truncated to the length the amino acid sequences would be. This is fixed in Version 1.1.

References

Criscuolo A, Gribaldo S (2010) BMGE (Block Mapping and Gathering with Entropy): selection of phylogenetic informative regions from multiple sequence alignments. *BMC Evol Biol* 10:210.

Eddy SR (2004) Where did the BLOSUM62 alignment score matrix come from? *Nat Biotechnol* 22:1035-1036.

Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci* 89:10915-10919.

IUPAC-IUB [International Union of Pure and Applied Chemistery and International Union of Biochemistery] (1970) Commission on Biochemical Nomenclature: Abbreviations and symbols for nucleic acids, polynucleotides and their constituents. *Biochem J* 120:449-454.

IUPAC-IUB [International Union of Pure and Applied Chemistery and International Union of Biochemistery] (1972) Commission on Biochemical Nomenclature: A one-letter notation for amino acid sequences (definitive rules). *Pure Appl Chem* 31:639-645.

Phillips MJ, Delsuc F, Penny D (2004) Genome-scale phylogeny and the detection of systematic biases. *Mol Biol Evol* 21:1455-1458.

States DJ, Gish W, Altschul SF (1991) Improved sensitivity of nucleic acid database searches using application-specific scoring matrices. *Methods: A Companion to Methods Enzymol* 3:66-70.

Stuart A (1955) A test for homogeneity of the marginal distributions in a two-way classification. *Biometrika* 42:412-416.