Description of the package MMG (version 1.4.0)

J. Noirel, G. Sanguinetti, and P. C. Wright
Chemical and Process Engineering — Computer Science
University of Sheffield, United Kingdom

November 18, 2008

Contents

1 Introduction

The `MMG` package should soon be part of the BioConductor[1] project.

Abstract of the orginal paper A fundamental task in systems biology is the identification of groups of genes that are involved in the cellular response to particular signals. At its simplest level, this often reduces to identifying biological quantities (mRNA abundance, enzyme concentrations, etc.) which are differentially expressed in two different conditions. Popular approaches involve using t-test statistics, based on modelling the data as arising from a mixture distribution. A common assumption of these approaches is that the data are independent and identically distributed; however, biological quantities are usually related through a complex (weighted) network of interactions, and often the more pertinent question is which subnetworks are differentially expressed, rather than which genes. Furthermore, in many interesting cases (such as high-throughput proteomics and metabolomics), only very partial observations are available, resulting in the need for efficient imputation techniques.

We introduce Mixture Model on Graphs (MMG), a novel probabilistic model to identify differentially expressed submodules of biological networks and pathways. The method can

[1] `http://www.bioconductor.org/`

easily incorporate information about weights in the network, is robust against missing data and can be easily generalized to directed networks. We propose an efficient sampling strategy to infer posterior probabilities of differential expression, as well as posterior probabilities over the model parameters. We assess our method on artificial data demonstrating significant improvements over standard mixture model clustering. Analysis of our model results on quantitative high-throughput proteomic data leads to the identification of biologically significant subnetworks, as well as the prediction of the expression level of a number of enzymes, some of which are then verified experimentally.

MMG relies on a Gibbs sampler to evaluate the posterior probability of certain genes to be down-regulated, up-regulated or merely unchanged depending on their location in the metabolic network and proteomic measurements. (Other networks and other sources of relative measurement may be considered.) This package furthermore implements a function to retrieve subnetworks that seem to behave consistently (which are overall down-regulated or up-regulated, for instance).

Note: If you use this package please cite

Sanguinetti, G. *et al.*, "MMG: a probabilistic tool to identify submodules of metabolic pathways", *Bioinformatics*, **8**, pp. 1078-1084, 2008.

2 Quick start

There are three important functions:

MMG.compute Runs the Gibbs sampler.

MMG.cut.graph Identify parts of the network that behave consistently.

MMG.make.dot Produces a DOT file in order to visualise the result of `MMG.cut.grahp`

The inputs are as follows:

- `MMG.compute` uses an input file that described the network of interest and the measurements that are available. The network may be directed. The kth line must resemble:

  $$k \quad m_k \quad n_{k1}\ w_{k1} \quad n_{k2}\ w_{k2} \quad \dots \quad n_{kN}\ w_{kN}$$

  The interpretation of the line is the following:

  - k is provided for legibility's sake (it could be any number),
  - m_k is the relative measurement ($\log_2$-ratio, or `NA` when not available),
  - there are N neighbours within the network, n_{ki} $(1 \leq i \leq N)$,
  - the weight of the connexion $i \to k$ is w_{ki} $(1 \leq i \leq N)$ and must be positive.

2.1 Calling the package

```
> library("MMG")
```

2.2 Running the Gibbs sampler

The following file represents a hexagonal network:

```
1  NA   6 1  2 1
2  NA   1 1  3 1
3  -1.0 2 1  4 1
4  NA   3 1  5 1
5  NA   4 1  6 1
6  +1.0 5 1  1 1
```

Two nodes have measurements, one up, the other down. The file containing the data is `hexag.dat`. We run `MMG.compute` on this file:

```
> r <- MMG.compute(file.name = "hexag.dat", steps = 1e+05, alpha = 0.1)

MMG Gibbs sampler (1e+05 steps)
Lambda down:     0.7226478; SD = 0.7325744
Lambda up:       0.7186159; SD = 0.7390824
Shannon entropy: 0.6457743; SD = 0.3960363

  The decimal point is 1 digit(s) to the left of the |

  0 | 34
  2 |
  4 |
  6 |
  8 | 0001
```

2.3 Cutting the graph

There are different methods to cut the graph. See documentation for a complete description. `MMG.cut.graph` operates on `r` as returned by `MMG.compute`:

```
> s <- MMG.cut.graph(r, select = "UP")

COMPONENT 1
  * Node   1 NA [0.305 0.100 0.594]
  * Node   5 NA [0.308 0.102 0.590]
  * Node   6 1.000 [0.000 0.030 0.970]
```

It prints out the components that looks up-regulated. Given the simple situation envisaged, only the node 6 ($\log_2$-ratio equal to +1.0) and its neighbours have a substantial probability of being up-regulated (97%, 59%, and 59%).

The `components` field has as many entries as there are nodes in the network.

```
> s$components

[1] 1 0 0 0 1 1
```

A value of 0 means that a node could not be identified as being up or down, a positive value m indicates that the node belongs to the mth connected component. In our example, only one component is identified and therefore only the value 1 occurs for node 1, 5, and 6.

2.4 Producing the DOT file

The value `r` returned by `MMG.compute` may be used to produce DOT files (see the result in Figure 1).

```
> MMG.make.dot(r, file.name = "test1.dot", selection = c(1, 5,
+     6))

NULL

> MMG.make.dot(r, file.name = "test2.dot", selection = 1:6)

NULL
```

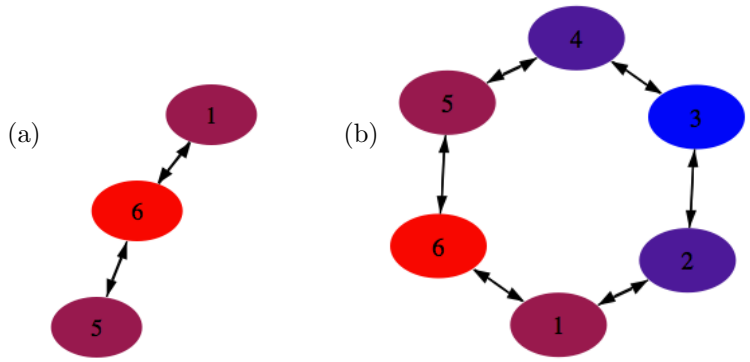

Figure 1: Result of the calls `MMG.make.dot`. (a) The up-regulated selection; (b) the entire network. The probabilities (p_-, p_0, p_+) are trivially mapped onto the RGB colour scheme. Therefore, red indicates likely to be up-regulated nodes and blue likely to be down-regulated nodes.

2.5 Altering the data

One can also, alter the data using an additional file. This only affects the values m_k.

```
> r <- MMG.compute(file.name = "hexag.dat", data = "new.dat", steps = 1e+05,
+     alpha = 0.1)

MMG Gibbs sampler (1e+05 steps)
Lambda down:     0.7208449; SD = 0.7552587
Lambda up:       0.7292316; SD = 0.7349576
Shannon entropy: 0.6410124; SD = 0.39674

  The decimal point is 1 digit(s) to the left of the |

  0 | 33
  2 |
  4 |
  6 |
  8 | 0000
```

```
> s <- MMG.cut.graph(r, select = "UP")

COMPONENT 1
  * Node   2 NA [0.305 0.098 0.597]
  * Node   3 -1.000 [0.000 0.028 0.972]
  * Node   4 NA [0.302 0.098 0.600]

> s$components

[1] 0 1 1 1 0 0
```

If manual modifications are needed, the data point m_k is stored in `r$dat[`k`, 2]`

3 Real example

Here is a real example that uses the data from the article

> Ow, S. Y. *et al.*, "Quantitative overview of N_2 fixation in *Nostoc punctiforme* ATCC 29133 through cellular enrichments and iTRAQ shotgun proteomics", submitted to *J Prot Res*, 2008.

```
> library("MMG")
> r <- MMG.compute(file.name = "R_net.dat", steps = 50000, burn.in = 1000,
+     sigma = 0.3, alpha = 1)

MMG Gibbs sampler (50000 steps)
Lambda down:     3.849416; SD = 0.8271672
Lambda up:       1.362223; SD = 0.1788497
Shannon entropy: 0.997166; SD = 0.2392688

  The decimal point is 1 digit(s) to the left of the |

   0 | 000011245
   1 | 46
   2 | 01258
   3 | 066
   4 | 7
   5 | 4579
   6 | 022234466666677777777777888888888999999999999999
   7 |
   8 |
   9 |
  10 | 667778888899999999999999999999999999999999999999999999999999999999
  11 | 000000000000000000000000000000000000000000000000000000000000000000000+202

> n <- r$dat$n.nodes
> n

[1] 422

> s <- MMG.cut.graph(r, descriptions = "R_descr.dat", method = "THRESHOLD",
+     threshold = 0.25, select = "UP")
```

```
COMPONENT 1
  * Node  11 1.309 [0.000 0.000 1.000] fructose-bisphosphate aldolase
  * Node  13 0.718 [0.000 0.090 0.910] fructose-1,6-bisphosphatase
  * Node  14 0.805 [0.000 0.053 0.947] glucose-6-phosphate isomerase
  * Node  15 0.803 [0.000 0.057 0.943] sugar kinase; hypothetical protein; glucokinase
  * Node  22 0.865 [0.000 0.039 0.961] isocitrate dehydrogenase
  * Node  35 1.157 [0.000 0.002 0.998] 6-phosphogluconate dehydrogenase
  * Node  37 1.434 [0.000 0.000 1.000] glucose-6-phosphate 1-dehydrogenase
  * Node  98 0.481 [0.000 0.307 0.693] argininosuccinate synthase
  * Node  99 0.783 [0.000 0.082 0.918] argininosuccinate lyase
  * Node 151 0.546 [0.000 0.242 0.758] aspartate aminotransferase
  * Node 155 0.842 [0.000 0.051 0.949] glutamate--ammonia ligase
  * Node 161 0.386 [0.000 0.325 0.675] glutathione reductase
  * Node 196 0.801 [0.000 0.069 0.931] cysteinyl-trna synthetase
  * Node 329 0.540 [0.000 0.259 0.741] methionine sulfoxide reductase a; ribulose 1,5-bisphosphat
  * Node 417 2.602 [0.000 0.000 1.000] nitrogenase molybdenum-iron protein beta chain; nitrogenas

COMPONENT 2
  * Node  46 0.650 [0.000 0.178 0.822] phosphoglucomutase/phosphomannomutase; mannose-1-phosphate

COMPONENT 3
  * Node  47 0.921 [0.000 0.031 0.969] dtdp-glucose dehydratase

COMPONENT 4
  * Node  63 1.280 [0.000 0.001 0.999] hypothetical protein

COMPONENT 5
  * Node 122 1.055 [0.000 0.009 0.991] adenine phosphoribosyltransferase

COMPONENT 6
  * Node 133 1.281 [0.000 0.001 0.999] phosphoribosylformylglycinamidine synthase subunit ii; pho

COMPONENT 7
  * Node 179 0.748 [0.000 0.114 0.886] seryl-trna synthetase

COMPONENT 8
  * Node 203 0.599 [0.000 0.232 0.768] 3-isopropylmalate dehydrogenase

COMPONENT 9
  * Node 209 0.452 [0.000 0.370 0.630] valine--pyruvate transaminase

COMPONENT 10
  * Node 228 1.084 [0.000 0.006 0.994] proline iminopeptidase

COMPONENT 11
  * Node 230 0.481 [0.000 0.345 0.655] histidyl-trna synthetase

COMPONENT 12
  * Node 280 1.291 [0.000 0.001 0.999] hypothetical protein
```

```
COMPONENT 13
  * Node 404 0.737 [0.000 0.116 0.884] ferrochelatase

COMPONENT 14
  * Node 420 0.528 [0.000 0.306 0.694] phosphoadenosine phosphosulfate reductase
```

```
> l <- (1:n)[s$components != 0]
> l

 [1]  11  13  14  15  22  35  37  46  47  63  98  99 122 133 151 155 161 179 196
[20] 203 209 228 230 280 329 404 417 420
```

```
> MMG.make.dot(r, file.name = "nostoc.dot", selection = l, type = "UNDIRECTED",
+     rem.loops = TRUE)

NULL
```

The result is presented in Figure 2

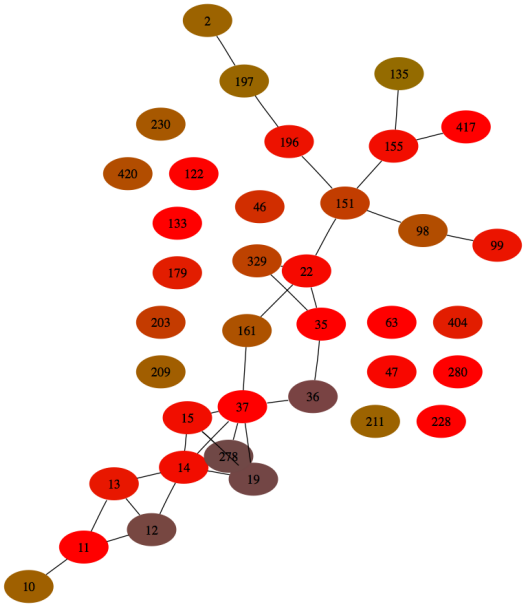

Figure 2: Up-regulated pathways in *Nostoc punctiforme*'s metabolic network in nitrogen-fixing conditions.