INSTALLATION AND USER GUIDE FOR BIOMENET

1. Contents of the download

Directory	**Contents**
`BiomeNet_R_Package`	Contains code for building BiomeNet.
`kegg_files`	KEGG files needed for conversion of EC abundance data to reaction information.
`metabolic_nets`	Directory containing reaction information used as input for BiomeNet.
`results`	Directory where R binary images from BiomeNet runs are placed.
`Mammal_metabolic_nets`	Directory containing the metabolic.net files for the mammal data used as input for BiomeNet.
`Qin_metabolic_nets`	Directory containing the metabolic.net files for the Qin data used as input for BiomeNet.

Files	**Function**
`write_network.py`	Example python script for converting EC abundance count into reaction information (`metabolic.net` files) which are used as input to BiomeNet.
`readnetworks.R`	R script that reads `metabolic.net` files and creates, processes and stores the networks.
`K3_L100.R`	This R script provides an example of how to train the model using K=3 metabotypes and L=100 subnetworks.
`Analysis.R`	Example R script provided to create plots, and examine subnetworks and metabotypes.
`InferenceFunctions.R`	R script used by `Analysis.R` to draw samples from the model and get parameter estimates.

2. Installing BiomeNet R package

2.1 Prerequisites:

Boost C++ Libraries
GNU scientific library (GSL)

2.2 Installation instructions:

Before building BiomeNet you need to edit the `Makevars` file which is found in the `src` directory in the `BiomeNet_R_Package` directory. The `Makevars` file must indicate the location of the boost header files *on your system*.

Specifically, you must edit the fourth line of `Makevars` file to point to the path where your boost header files are located. Something like `/opt/local/include/`.

To Build BiomeNet you must be in the main directory (`BiomeNet`) which contains the directory `BiomeNet_R_Package`.

To build and install, type the following at the command line:

```
R CMD build BiomeNet_R_Package
R CMD INSTALL BiomeNet_1.0.1.tar.gz
```

Note: The above commands are case sensitive.

3. Preprocessing data

BiomeNet requires reaction information (`metabolic.net`) files. These files are formatted such that each line contains a substrate compound number, a reaction number, and a product compound number (e.g., `C04507,R08857,C17959`). This information can be obtained from EC information using the provided python script "`write_network.py`". This script takes EC abundance information from each sample and converts it to the necessary metabolic.net file. The python script requires an EC abundance file for each sample. The name of this file is used as the identifier for the sample and should follow the format "`[`*`SampleName`*`].EC.abundance`". The script assumes that these files are located in the `DATA` directory. The format is very simple. Each line represents an EC number, and its count in the sample is delimited by a colon (:).

For example:

```
EC 1.1.1.1:189
EC 1.1.1.2:5
EC 1.1.1.3:47
```

(Note that there is a space between EC and 1.1.1.1, and it is required).

We have included EC abundance files from the mammal dataset (Muegge et al. 2011) in the directory `DATA` for use as an example. Users who wish to skip the pre-processing step, can obtain the `metabolic.net` files (one for each sample) that we also provide for these data so they can immediately run BiomeNet.

To run your own data, replace the mammal files with your own files. **NOTE: the python script will process all files in the `DATA` directory having the *`.EC.abundance`* extension to produce metabolic.net files, so please be careful to ensure that you have exactly the data you want within the `DATA` directory.**

The python script will create the metabolic.net files for each sample in the `DATA` directory and place them all in the `metabolic_nets` directory. These files will follow the naming convention of "`SampleName.metabolic.net`".

4. Running BiomeNet

Step 1 Creating the network

Once you have metabolic.net files, you will need to create a network which is done by using the R script `readnetworks.R`. To do this, type the following:

```
R CMD BATCH readnetworks.R
```

This script will look in the `metabolic_nets` directory for the required files and use all files contained within that folder. It will read, process and store the data as an R binary image in the `results` directory as `network.RData`. This script took 35 minutes to run on a MacBook Air (1.7 GHz Intel Core i7, and 8 GB RAM) on the mammal data provided.

Step 2 Running the model

With the networks read, processed and stored as an R binary image, the model can be run. We provide a sample script to run a model for K=3 metabosytems and L=100 subnetworks. The provided script called `K3_L100.R` uses the network created by the previous script and runs the model to get the parameter estimates for phi, theta, etc in the model. To do this, type the following:

```
R CMD BATCH K3_L100.R
```

In this example, the output is saved as an R binary image in the `results` directory as `K3_L100.RData`.

This script can be modified to examine different numbers of metabosystems (K) and subnetworks (L). The variables that can be changed for this purpose are mostly self-explanatory, but are shown below for completeness.

`K=3`	number of metabosystems
`L=100`	number of subnetworks
`max.iter = 500`	number of Gibbs samples
`iter.lag = 20`	how often you save a sample
`burnin = 100`	number of runs at beginning to exclude

Note that this can take some time to run depending on the number of reactions, samples and abundance counts of reactions. For example, the mammal dataset took 30 hours to run on a MacBook Air (1.7 GHz Intel Core i7, and 8 GB RAM).

The effect of different K and L values can be explored by changing the supplied R script as described above and re-running *Step 2*; furthermore, the same R binary generated in *Step 1* (`network.RData`) can be re-used for each case, thereby avoiding the need to rerun *Step 1*.

Note: Any inference from a finite number of MCMC samples can only approximate the target distribution. The number of steps required for convergence with an acceptable amount of error could differ among datasets. Users are advised to assess trends within a chain to monitor stationarity. Running multiple chains is a way to check for convergence to a stable distribution. Users should run chains longer, colleting more Gibbs samples, when there is concern about convergence.

5. Analysis

We provide an R script `Analysis.R` that reproduces the plots in the manuscript. In addition, the script creates a file for each subnetwork that contains posterior probabilities of membership to that subnetwork for each reaction. This script relies on two R packages (*plotrix*, and *shape*) and functions implemented in `InferenceFunctions.R` script (provided). The user must install the R packages *plotrix* and *shape* before running the analysis script.

Note: Coloring of samples in the simplex plot uses the numbering system for samples derived from the ordering of samples in R (note that R is case sensitive, with capitals coming before lower case letters; thus the order will be case sensitive). The coloring scheme in the provided R script `Analysis.R` is for the mammal data provided. For your own data we recommend you load your results (e.g., `K3_L100.RData`) into R and type **sample.names** to obtain a list of sample labels in which the order of this list is the order that R assumes. By using this system, you can edit the script `Analysis.R` to assign unique colors for your samples.

To run the analysis script type:

```
R CMD BATCH Analysis.R
```

General guidelines for interpreting subnetworks and metabosystems: It is common for different runs of any clustering algorithm under a mixture models to produce different labels for the same cluster of data. Because the analysis is unsupervised, labels are assigned arbitrarily. For example, different runs of BiomeNet can potentially use different ID numbers to label metabosystem having a common biological underpinning. This also applies to the numbering of subnetworks in different runs. Furthermore, when there are potentially large numbers of subnetworks making only a trivial contribution to a given metabosystem (*e.g.*, expected when L=100) the mixing probabilities of some subsystems can vary a little between runs. Hence, we provide below some general guidelines for interpreting the output of BiomeNet when it has been run multiple times.

- Users are encouraged to run the inference algorithm multiple times to assess and identify stable signal within the results. If results are not stable, users should try extending the run to obtain more MCMC samples.
- When the inferred metabosystems have characteristic reaction profiles, they will be stable across analyses and the relative positions of the samples in the simplex plots will be the same even though the numbering system differs. In this case the simplex plots can be rotated to coordinate the metabosystem IDs among different runs. This is described and illustrated for both example datasets in supplementary TextS6.
- Subnetworks can be numbered differently in different runs. For example, discriminatory subnetwork 49 in one run maybe labeled subnetwork 77 in another run.
- The colors assigned to subnetworks in the ribbon plots are intended to aid visual inspection; like the ID numbers, the colors assigned to subnetworks in the ribbon plots can differ across runs.
- When reaction profiles are stable across analyses, the subnetwork IDs can be coordinated across runs according to similarity in the mixing probabilities of their component reactions.
- As some subnetworks can be superfluous (*e.g.*, when L=100) there can be differences among the mixing probabilities of such subnetworks among runs. Such subnetworks will have trivial mixing probabilities and un-stable reaction profiles.
- Occasionally, the mixing probabilities of some non-trivial subnetworks will vary among runs. This is most often associated with a single subnetwork in one run being split into two subnetworks in another run. This is evident when the reaction composition of the separate subnetworks is the same as the reaction composition of a single subnetwork in another run. When this occurs, users

should explore running the MCMC longer, decreasing the value of the Dirichlet concentration parameter (α), or both.

- Highly discriminatory subnetworks should have very stable mixing probabilities and reaction profiles.