METABOLIC RANDOM FIELDS AND EVOLUTION

Stochastic models for the loss and gain of reactions in a metabolic network related by a phylogeny

Contents

- Revision
- Sampling Hidden states, h
  - -----MCMC
  - -----Wolff Algorithm
- Sampling Evolutionary Parameters, λ&μ
-
- Data from KEGG: Extracting and Parsing
- Next Two Weeks
  - -----Data
  - ------General Method
- Limitations and Potential Extensions

Revision

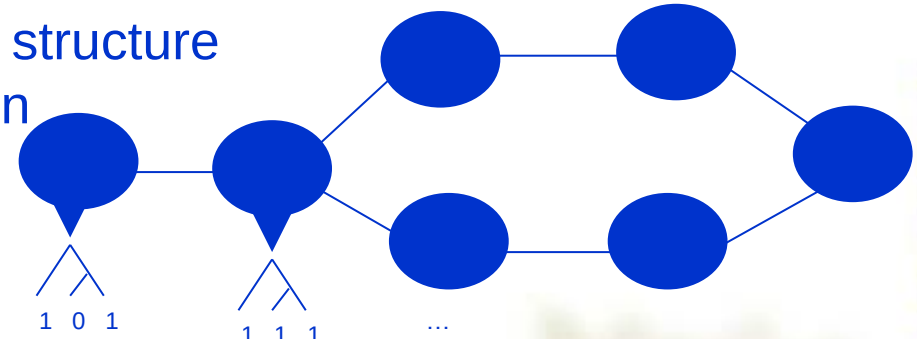
- Our Aim: to understand and model the gain and loss of reactions in metabolic networks for a known phylogeny

- Our Approach: Continuous time Markov Chain
  Felsenstein Algorithm
  Neighbour-dependent Model
  Markov Random Field

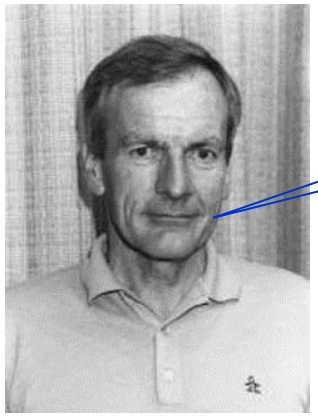
  and MORE…

Sampling Hidden States, h

- Recall that we use a Markov Random Field
  - (MRF) to model the dependency structure
  - between hidden states at reaction
  - sites



- We choose ***Potts model*** since it allows for any number of hidden states



$$p(h) = \exp\left\{\beta \sum_{i \sim j} J(h_i, h_j)\right\} / Z(\beta)$$

where βis a measure of interaction strength between neighboring hidden states

and:

$$J(h_i, h_j) = \begin{cases} 1 & if\ h_i = h_j \\ 0 & otherwise \end{cases}$$

- By Bayes' rule, the posterior probability of hidden states h given the data χ is given by:

$$p(h|\mathcal{X}) = p(\mathcal{X}|h)p(h|\beta)/p(\mathcal{X})$$
$$= \sum_i p(\mathcal{X}|h_i)p(h|\beta)/Z(\mathcal{X}|\beta)$$

- Under Potts model, we have:

$$p(h_i|h_{j\sim i}, \chi, \beta) \propto p(\chi_i|h_i)p(h_i|h_{j\sim i}) \propto p(\chi_i|h_i)\mathrm{e}^{\beta\sum_{j\sim i} J(h_i,h_j)}$$

- Based on this, we can construct an MCMC scheme to sample h, with acceptance ratio:

$$r(h_i'|h_i) = \min\left\{1, \frac{p(h_i'|\chi)q(h_i|h_i')}{p(h_i|\chi)q(h_i'|h_i)}\right\}$$

Unfortunately, such single-site proposal perform very badly in MRF!

WOLFF algorithm for the rescue

Wolff Algorithm – *simple but efficient*

1. pick a reaction i at random, and form the sets $C = \{i\}$, $\overline{C} = \emptyset$

2. for every neighbor j of i, add j to C with probability $1 - \mathrm{e}^{-\beta J(h_i, h_j)}$

3. if j is added to C, all neighbors k of j are considered for inclusion in the cluster as in step 2, where $k \notin C \cup \overline{C}$

4. after all reactions have been considered, set $h_c = h_i'$ for all reactions c in C with probability $\prod_{c \in C} \frac{p(\chi_c|h_i')}{p(\chi_c|h_i)}$, where h_i' is any state different from h_i

In essence, if a cluster, C of reactions with hidden state h_i collectively has K neighbors in state h_i and K' neighbors in state h_i', we've generated the cluster move from h_i to h_i' with probability $(1-\mathrm{e}^{-\beta})^{|C|}\mathrm{e}^{-\beta K}$ and from h_i' to h_i with probability $(1-\mathrm{e}^{-\beta})^{|C|}\mathrm{e}^{-\beta K'}$

As in the Potts model

$$\frac{p(h_i'|\chi)}{p(h_i|\chi)} = \mathrm{e}^{\beta(K'-K)} \prod_{c\in C} \frac{p(\chi|h_i')}{p(\chi|h_i)}$$

we see that the Wolff algorithm above generates exactly the same Metropolis-Hastings ratio as the single-site update, while the cluster update is much more likely to be accepted

Sampling Rate Parameters, λ&μ

λ and μ are different for each hidden state h_k, denoted as λ_{h_k} and μ_{h_k}

But we assume these λ and μ are all distributed *a priori* according to independent Gamma distributions, gamma(α, ξ) (Mithani et al. 2010)

By Bayes' rule:

$$p(\lambda, \mu | \chi, h, t) \propto p(\chi | h, \lambda, \mu) p(\lambda, \mu | h)$$

$$= \prod_i p(\chi | h_i, \lambda_{h_i}, \mu_{h_i}) p(\lambda_{h_i} | \alpha, \xi) p(\mu_{h_i} | \alpha, \xi)$$

Hence, we use MCMC to sample λ and μ: a symmetric normal random walk on $\log \lambda$ and $\log \mu$ with Metropolis Hastings ratios:

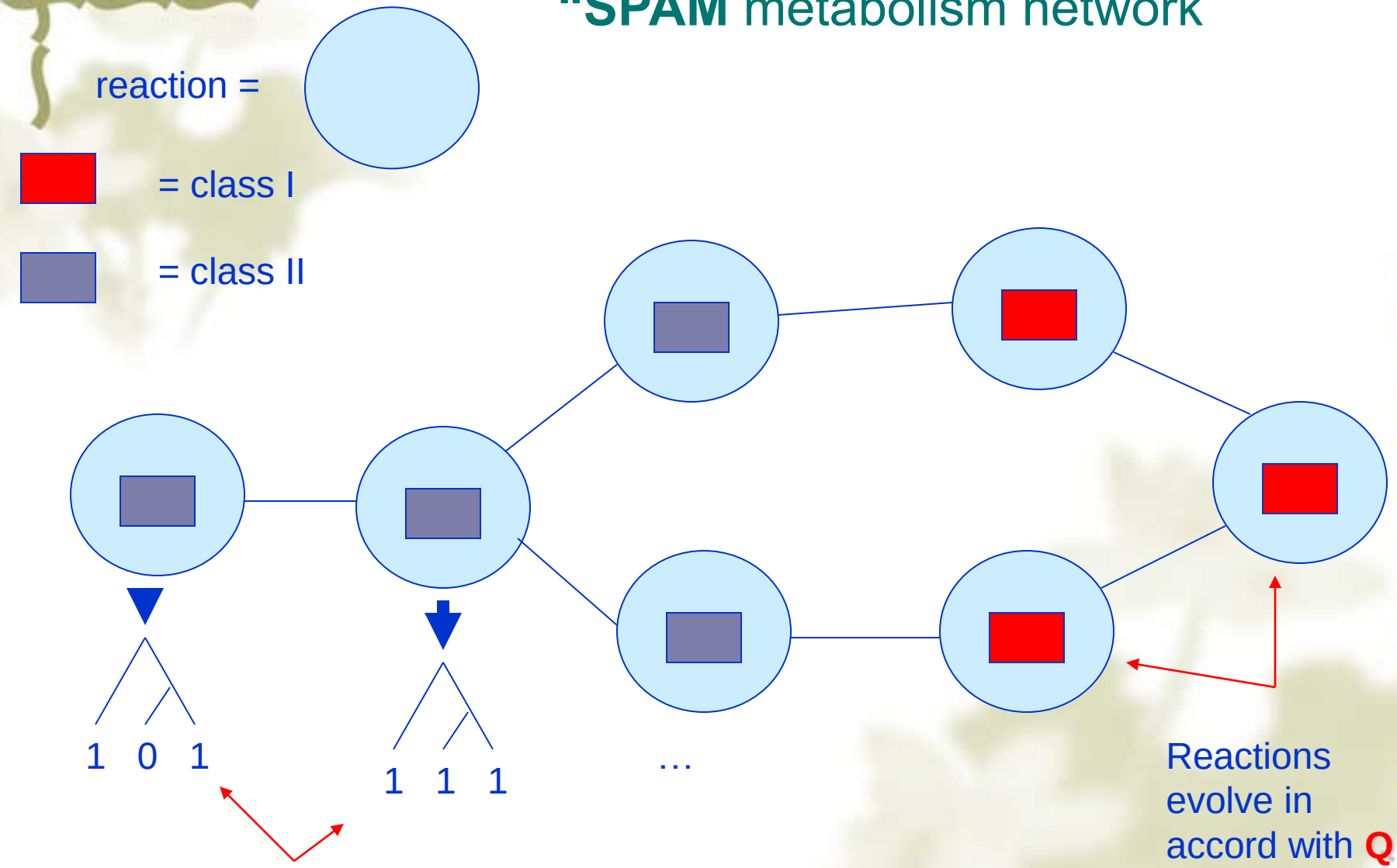
$$r(\lambda'_{h^{(k)}}|\lambda_{h^{(k)}}) = \min\{1, \frac{f(\chi|\lambda'_{h^{(k)}}, \mu_{h^{(k)}})p(\lambda'_{h^{(k)}}|\alpha, \xi)}{f(\chi|\lambda_{h^{(k)}}, \mu_{h^{(k)}})p(\lambda_{h^{(k)}}|\alpha, \xi)}\}$$

$$r(\mu'_{h^{(k)}}|\mu_{h^{(k)}}) = \min\{1, \frac{f(\chi|\mu'_{h^{(k)}}, \lambda_{h^{(k)}})p(\mu'_{h^{(k)}}|\alpha, \xi)}{f(\chi|\mu_{h^{(k)}}, \lambda_{h^{(k)}})p(\mu_{h^{(k)}}|\alpha, \xi)}\}$$

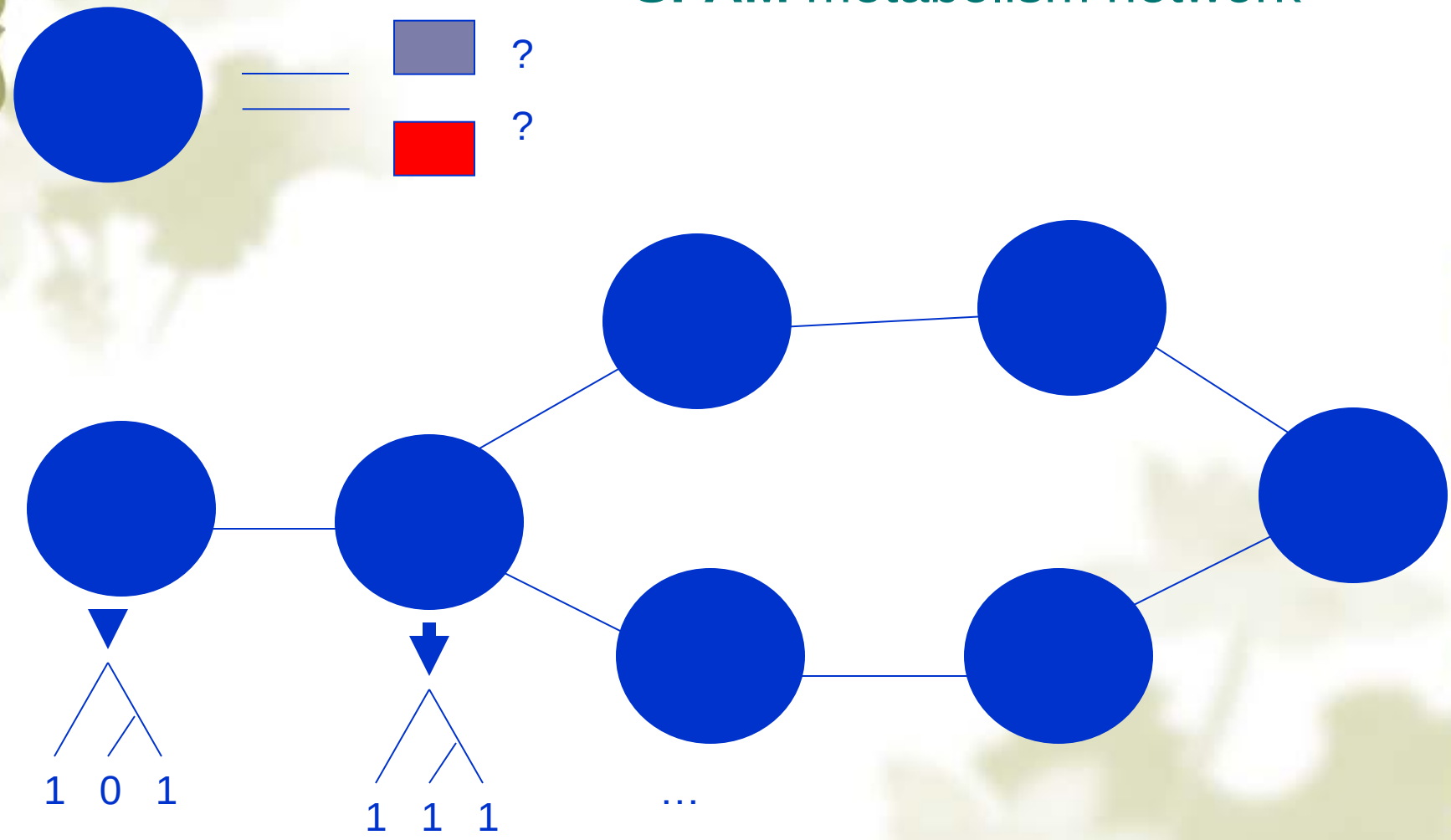
where f is the product of the likelihood of reactions in state $h^{(k)}$ and can be calculated with the Felsenstein algorithm

We have looped these sampling algorithms together and tested it on simulated datasets.

“**SPAM** metabolism network”

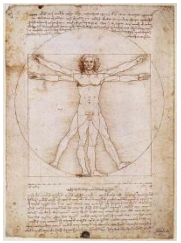


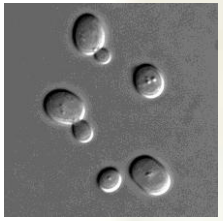
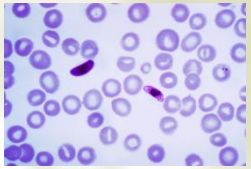
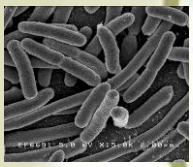
SPAM metabolism network



Network hidden: cannot see underlying state, but we know the phylogenies

Data



Data

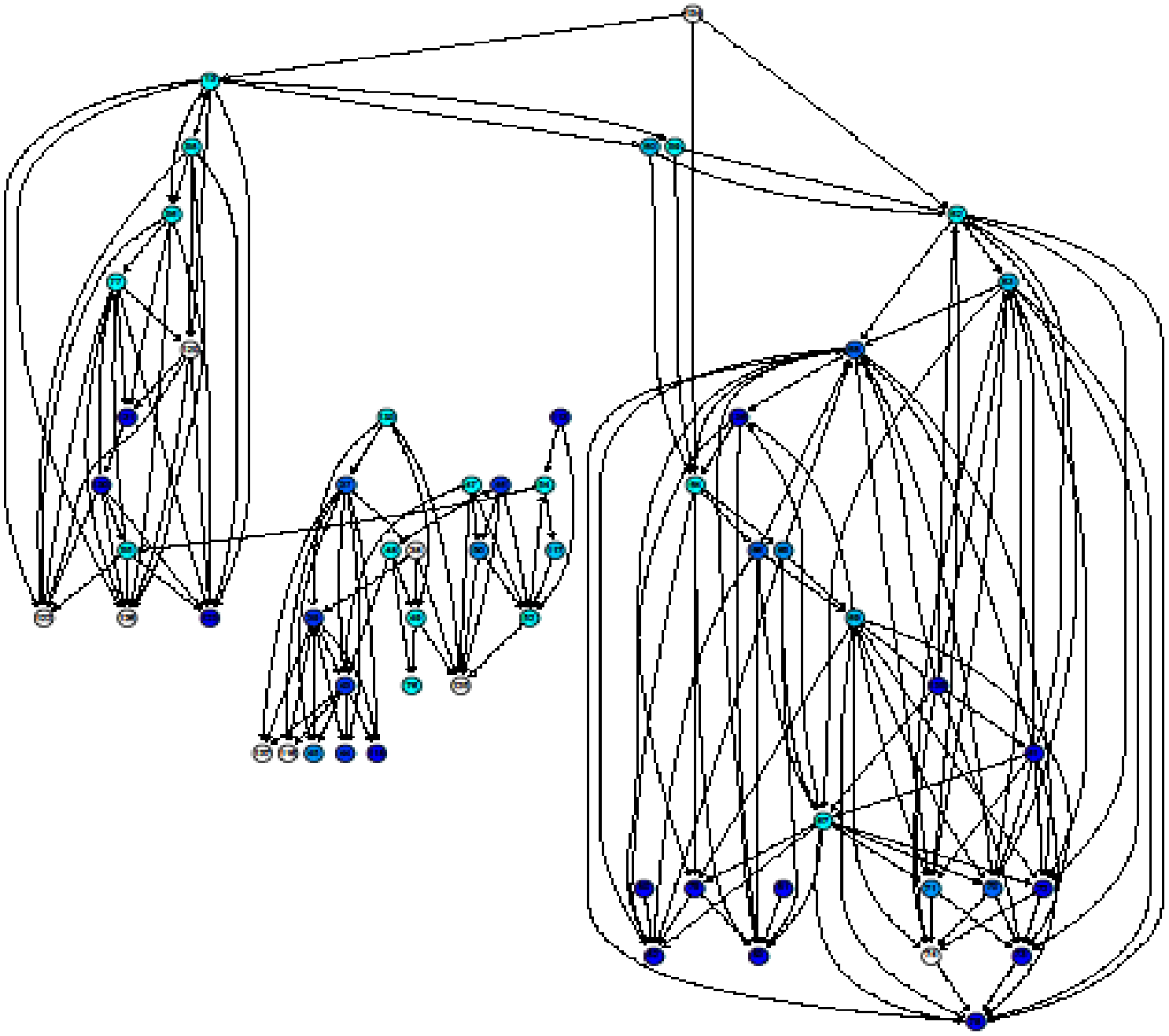
- rn: 147
- hsa: 87
- dme: 86
- ath: 95
- sce: 72
- pfa: 57
- eco: 89

- rn00010 (glycolysis)
- rn00020 (TCA cycle)
- rn00030 (pentose phosphate)
- rn00040 (pentose/glucuronate interconversions)
- rn00051 (fructose/mannose)
- rn00052 (galactose)
- rn00053 (ascorbate/aldarate)
- rn00061 (fatty acid biosynthesis)

Data

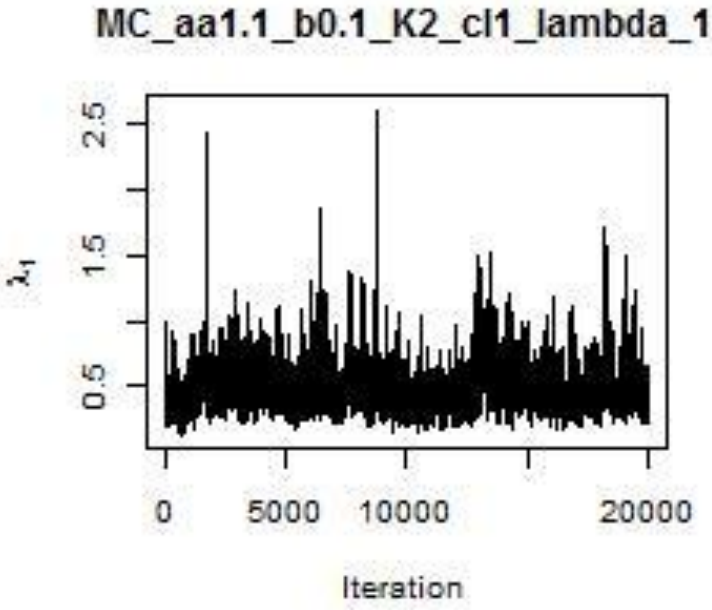
- Load maps into R using *KEGGgraph*
- Parse reactions
- Save as edgelists and nodelists (load into simulator later)

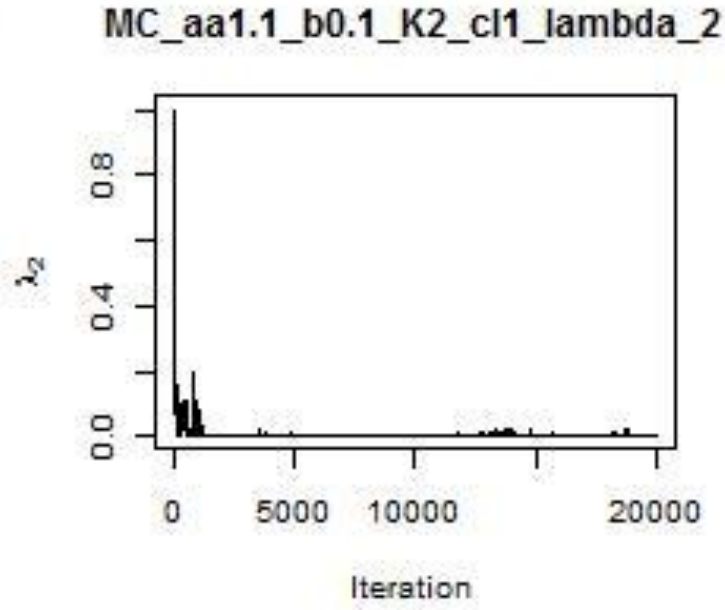
Data

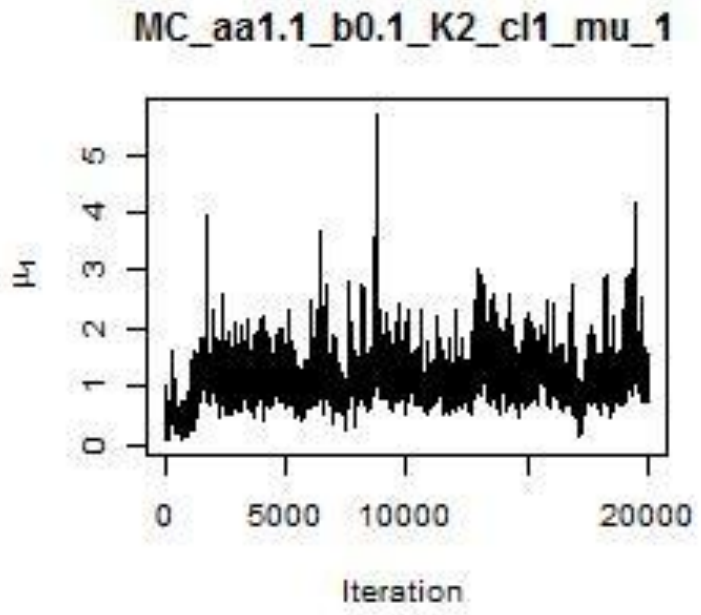


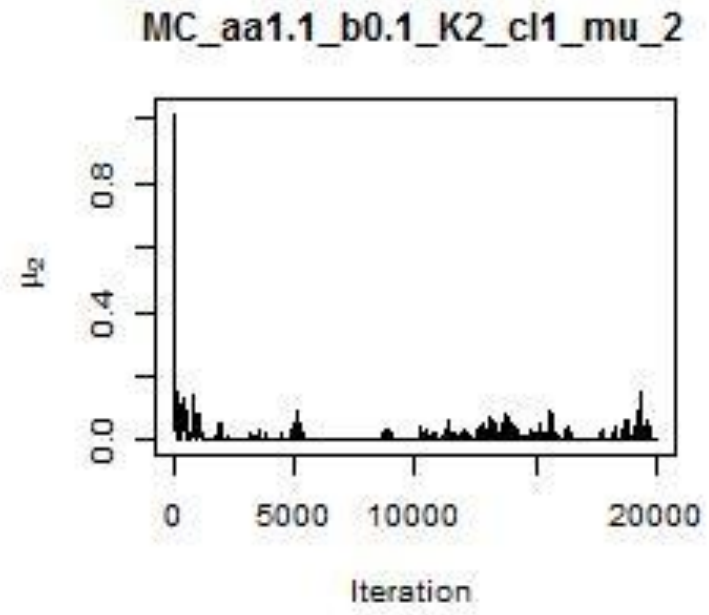
Results

- For now 1 network with 6 species phylogeny.
- Branch lengths of phylogenetic tree are randomly generated.
- Each reaction takes two possible annotations or states. These correspond to the rates of addition or deletion of reactions in the network.
- Run MCMC on lambda mu and h (the hidden state).

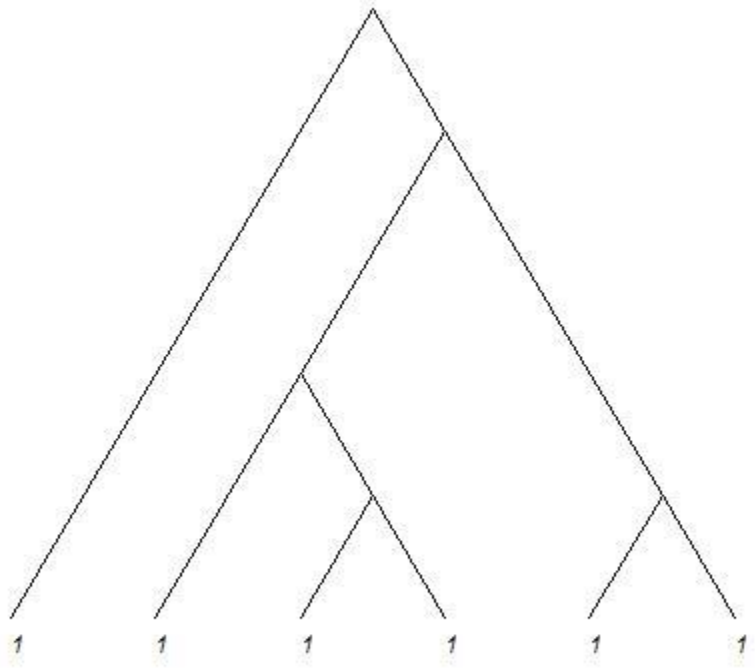

MC_aa1.1_b0.1_K2_cl1_lambda_1
λ1
0.5
1.5
2.5
0
5000
10000
20000
Iteration

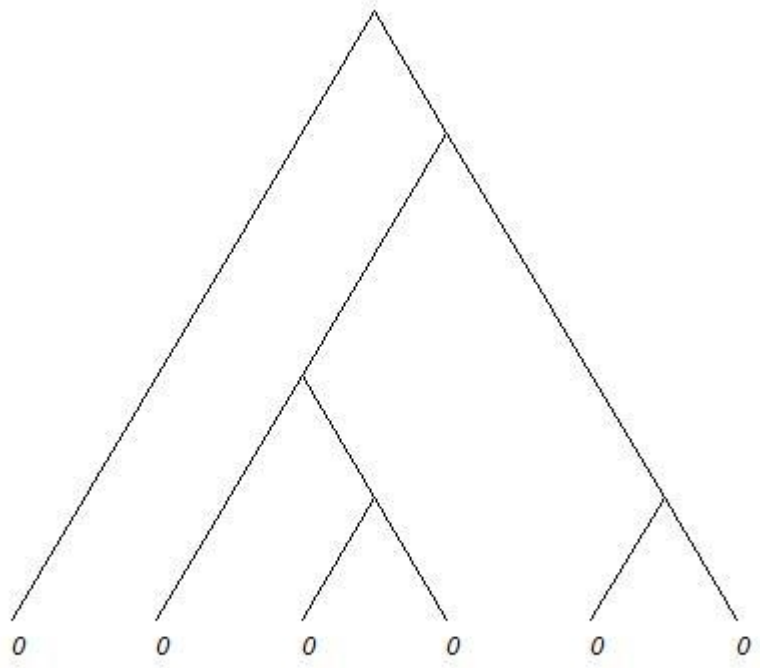

MC_aa1.1_b0.1_K2_cl1_lambda_2
λ2
0.0
0.4
0.8
0
5000
10000
20000
Iteration

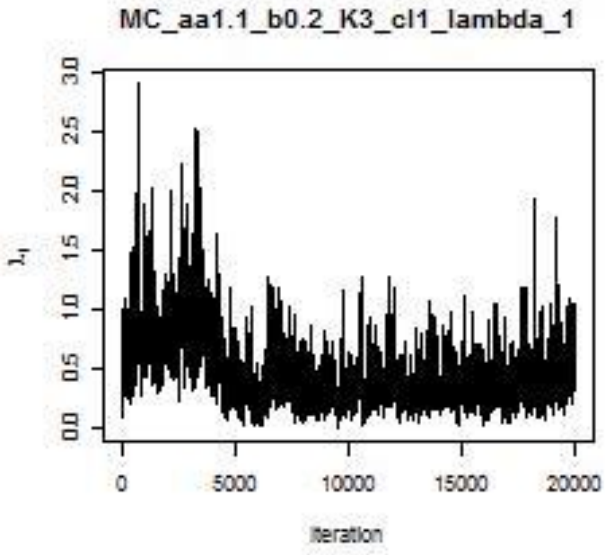

MC_aa1.1_b0.1_K2_cl1_mu_1
μ1
0
1
2
3
4
5
0
5000
10000
20000
Iteration

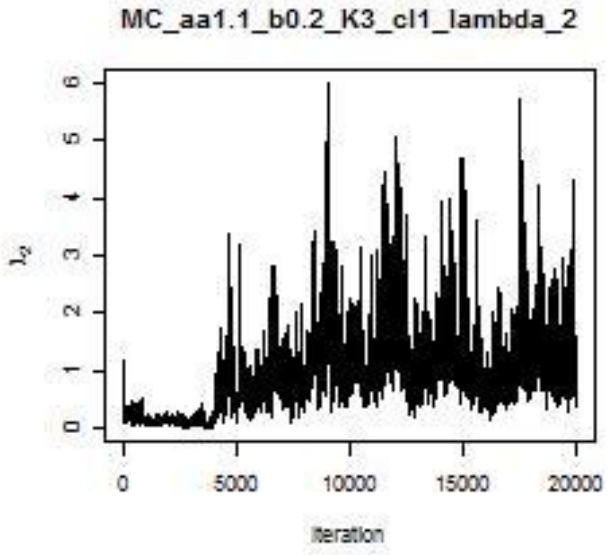

MC_aa1.1_b0.1_K2_cl1_mu_2
μ2
0.0
0.4
0.8
0
5000
10000
20000
Iteration

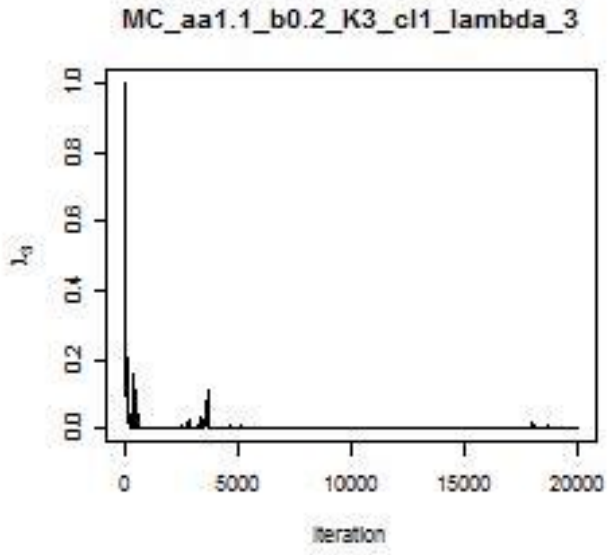
Conserved Reactions

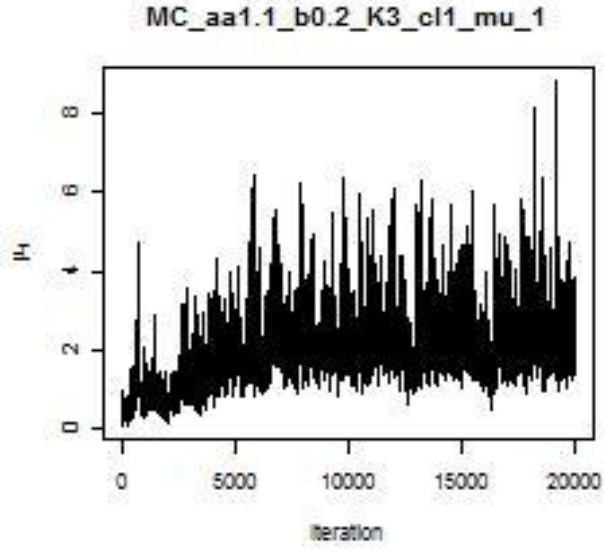


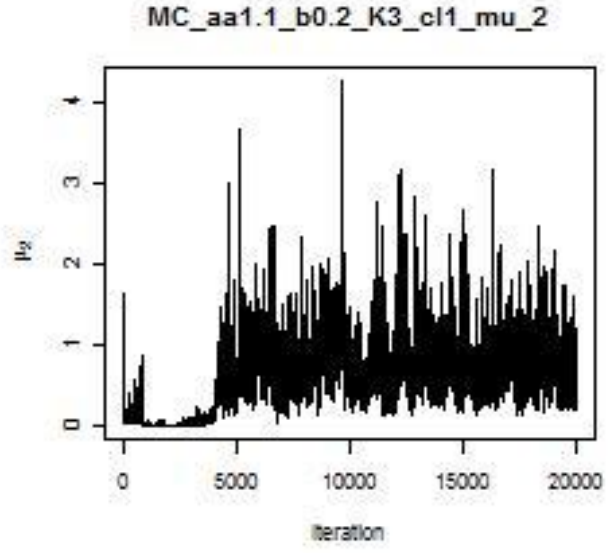


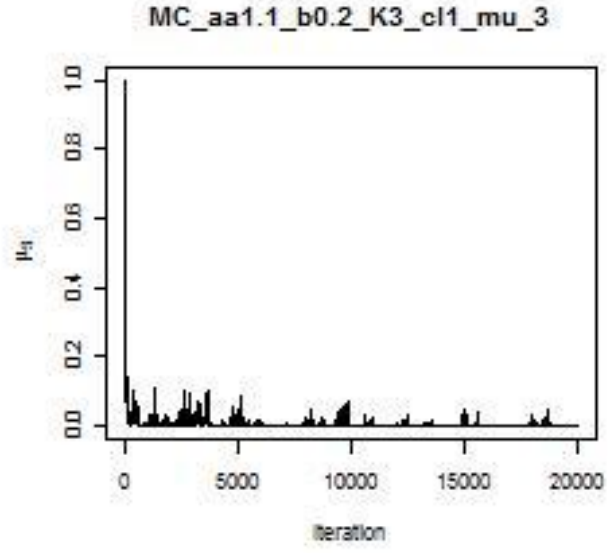

MC_aa1.1_b0.2_K3_cl1_lambda_1
λ1
0.0 0.5 1.0 1.5 2.0 2.5 3.0
0 5000 10000 15000 20000
Iteration

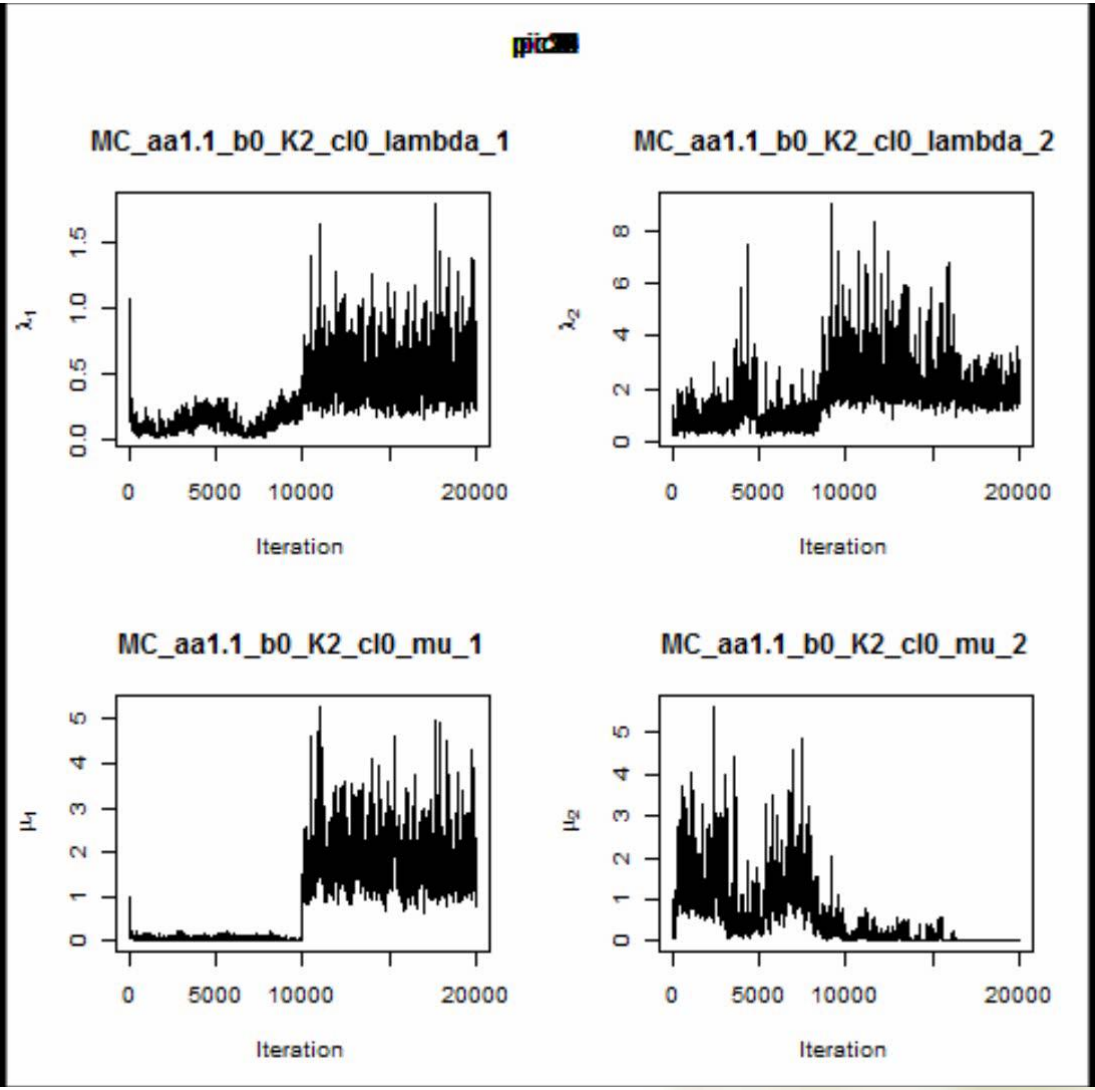

MC_aa1.1_b0.2_K3_cl1_lambda_2
λ2
0 1 2 3 4 5 6
0 5000 10000 15000 20000
Iteration

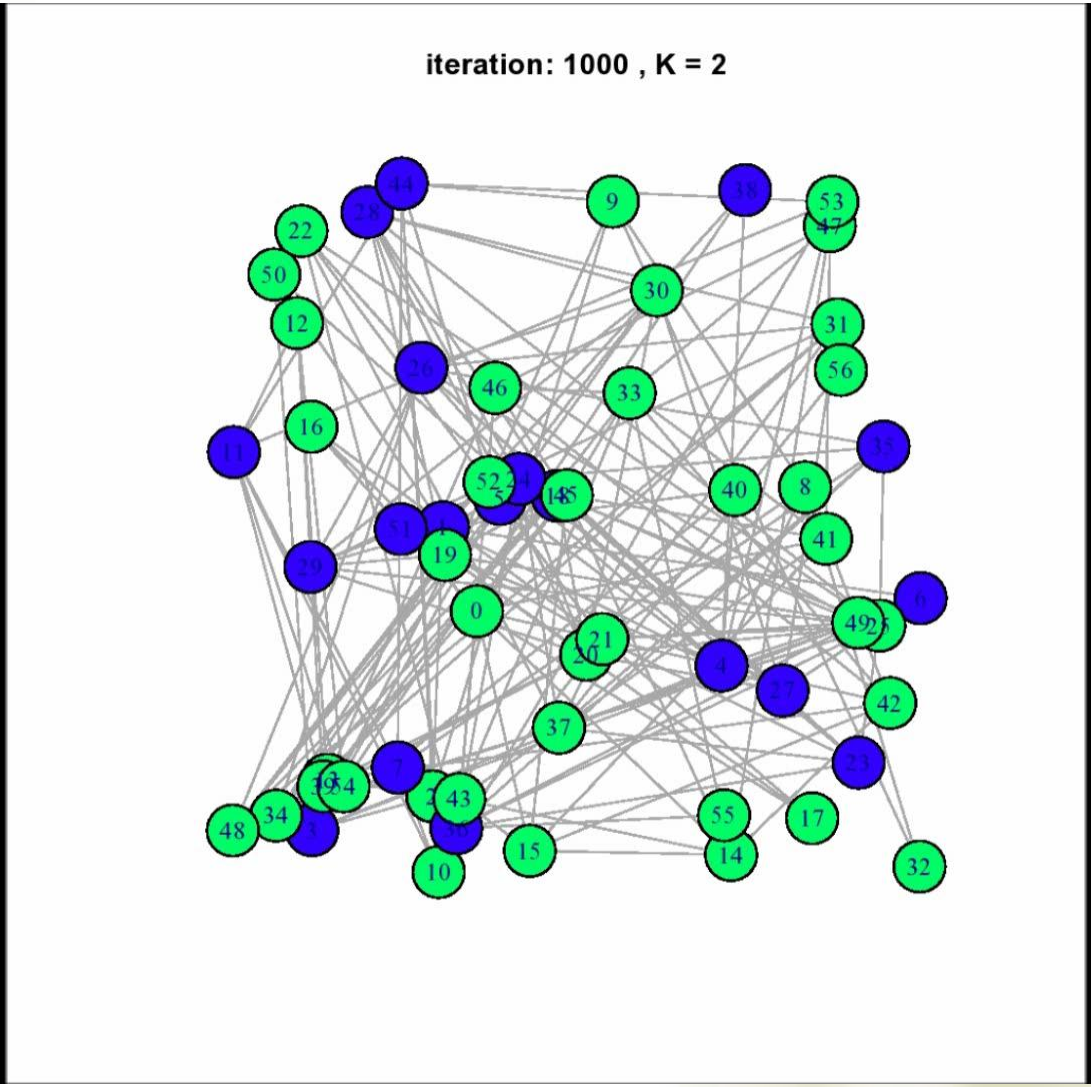

MC_aa1.1_b0.2_K3_cl1_lambda_3
λ3
0.0 0.2 0.4 0.6 0.8 1.0
0 5000 10000 15000 20000
Iteration


MC_aa1.1_b0.2_K3_cl1_mu_1
μ1
0 2 4 6 8
0 5000 10000 15000 20000
Iteration


MC_aa1.1_b0.2_K3_cl1_mu_2
μ2
0 1 2 3 4
0 5000 10000 15000 20000
Iteration


MC_aa1.1_b0.2_K3_cl1_mu_3
μ3
0.0 0.2 0.4 0.6 0.8 1.0
0 5000 10000 15000 20000
Iteration


MC_aa1.1_b0_K2_cl0_lambda_1
λ1
0.0
0.5
1.0
1.5
0
5000
10000
20000
Iteration
MC_aa1.1_b0_K2_cl0_lambda_2
λ2
0
2
4
6
8
0
5000
10000
20000
Iteration
MC_aa1.1_b0_K2_cl0_mu_1
μ1
0
1
2
3
4
5
0
5000
10000
20000
Iteration
MC_aa1.1_b0_K2_cl0_mu_2
μ2
0
1
2
3
4
5
0
5000
10000
20000
Iteration


iteration: 1000 , K = 2
44
9
38
53
22
50
30
12
31
26
46
33
56
16
11
52
40
8
41
19
29
0
49
21
42
37
43
48
34
55
17
15
14
32
10

Why Such Variability??

- Variation of interaction strength between neighbouring reactions.
- Variation in jump sizes.
- Cluster flips or single state flips.
- Number or possible hidden states.
- Prior for addition / deletion probabilities.
- Parameterisation of these?

What's Next

- Account for popularity tendency.
- Testing for convergence.
- Different species? more species?
- Branch lengths?
- Enough nodes? (could combine networks)
- Write the report


THE
END