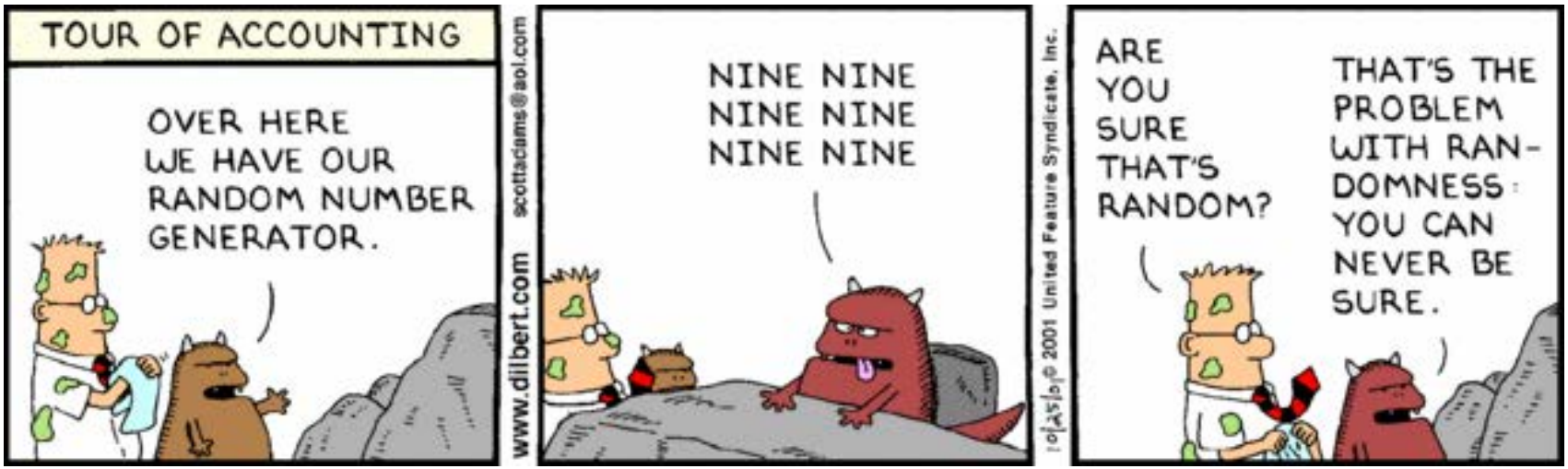

TOUR OF ACCOUNTING
OVER HERE WE HAVE OUR RANDOM NUMBER GENERATOR.
www.dilbert.com scottadams@aol.com
NINE NINE NINE NINE NINE NINE
ARE YOU SURE THAT'S RANDOM?
THAT'S THE PROBLEM WITH RANDOMNESS: YOU CAN NEVER BE SURE.
© 2001 United Feature Syndicate, Inc.

Hierarchical modelling

Erik Štrumbelj

2019

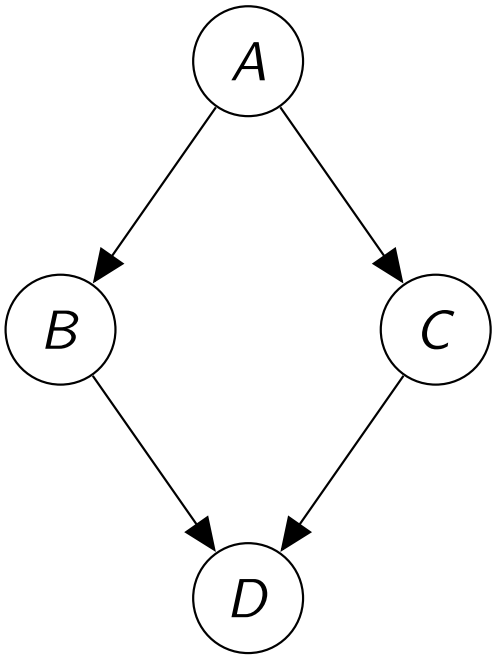
Outline

- **Plate notation** - DAGs and Bayesian networks.
- **Hierarchical modelling** - multilevel/mixed-effects modelling the Bayesian way.
- Bayesian interpretation of **regularization** and corrections for **multiple hypothesis testing**.
- Unrelated: **mixture modelling**.

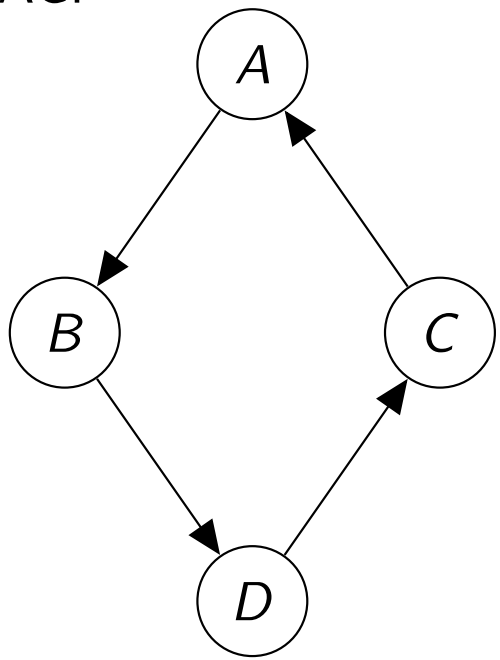
Directed Acyclic Graph (DAG)

Directed Acyclic Graph (DAG) is a directed simple graph without cycles.

DAG:



Not a DAG:



DAGs are a very useful representation of dependencies (and conditional independencies) between random variables in a Bayesian model.

Model can be represented with a DAG $\Leftrightarrow$ Model is a **Bayesian network**.

Plate notation

Simplifying repetitions with a plate:

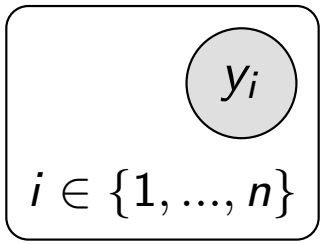
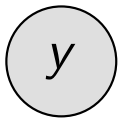


Plate notation sometimes features additional information:

Latent variable:

Deterministic node:

Data:



Constant:

μ_0

DAG-based factorization of the joint distribution

Take the joint distribution $p(\theta_1, \theta_2, \ldots, \theta_n)$ and a DAG that represents (in)dependencies between θ_i.

The joint distribution factorizes:

$$p(\theta_1, \theta_2, \ldots, \theta_n) = \prod_{i=1}^{n} p\left(\theta_i | \text{parents}(\theta_i)\right).$$

Factorizing full-conditionals

Given a DAG representation, the joint posterior of parameters θ factorizes:

$$p(\theta|y) = \frac{p(\theta, y)}{p(y)} \propto p(\theta, y) = p(V) = \prod_{v \in V} p\left(v|\text{parents}(v)\right).$$

Full-conditional for parameter θ_i:

$$p(\theta_i|V \setminus \theta_i) = \frac{p(V)}{p(V \setminus \theta_i)} \propto P(V) \propto p(\theta_i|\text{parents}(\theta_i)) \prod_{w \in \text{children}(\theta_i)} p(w|\text{parents}(w))$$

Sampling hierarchy used by software

$$p(\theta_i|y,\theta_{-i}) \propto p(\theta_i|\text{parents}(\theta_i)) \prod_{w \in \text{children}(\theta_i)} p(w|\text{parents}(w))$$

We can automatically derive and evaluate the full-conditionals.
Additionally, we have a symbolic representation of their shapes.

We can use this information to select more efficient sampling than M-H for each individual full-conditional.

Table 1. *Sampling method hierarchy used by WinBUGS. Each method is only used if no previous method in the hierarchy is appropriate*

Target distribution	Sampling method
Discrete	Inversion of cumulative distribution function (trivial)
Closed form (conjugate)	Direct sampling using standard algorithms
Log-concave	Derivative-free adaptive rejection sampling (Gilks 1992)
Restricted range	Slice sampling (Neal 1997)
Unrestricted range	Metropolis-Hastings (Metropolis *et al.* 1953, Hastings 1970)

1st generation Bayesian inference software

Input: Model description (in software's modelling language) & data.

- Deduce DAG from model description.
- Factorize full-conditionals based on DAG.
- Recognize conjugacy, etc... and select most efficient sampling for each parameter.
- Gibbs sampling.

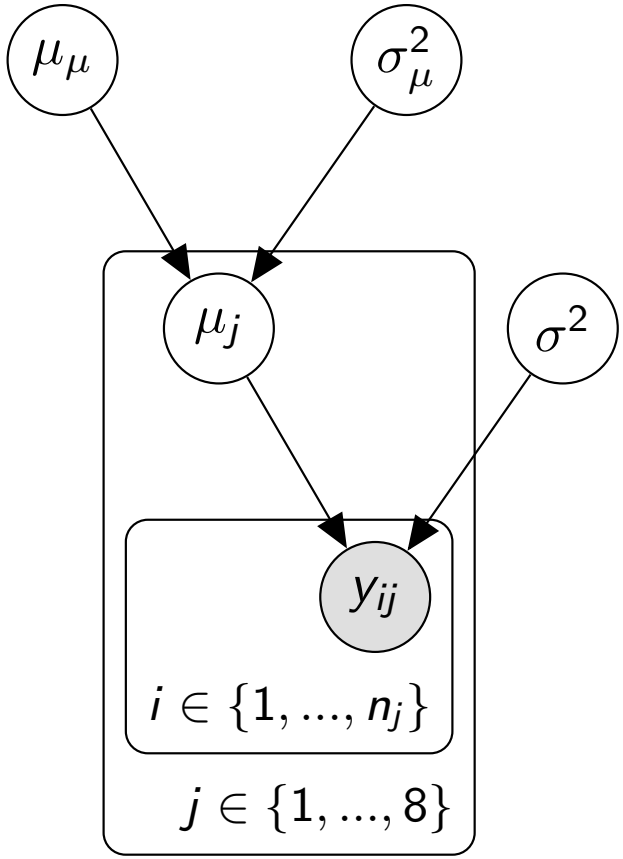
Output: MCMC samples from posterior.

Representatives: BUGS, WinBUGS, OpenBUGS, JAGS.

A hierarchical model for the means of groups

$$
\begin{aligned}
Y_{ij} &\sim N(\mu_j, \sigma^2), \\
\mu_j &\sim N(\mu_\mu, \sigma_\mu^2), \\
\sigma^2 &\sim IG(a, b), \\
\mu_\mu &\sim N(\mu_0, \sigma_0^2), \\
\sigma_\mu^2 &\sim IG(a, b).
\end{aligned}
$$

Plate notation:



JAGS example

$$
\begin{aligned}
Y_{ij} &\sim N(\mu_j, \sigma^2), \\
\mu_j &\sim N(\mu_\mu, \sigma_\mu^2), \\
\sigma^2 &\sim IG(10^{-4}, 10^{-4}), \\
\mu_\mu &\sim N(0, 10^6), \\
\sigma_\mu^2 &\sim IG(10^{-4}, 10^{-4}).
\end{aligned}
$$

JAGS:

```
model {
  tau ~ dgamma(0.001, 0.001)
  tauMu ~ dgamma(0.001, 0.001)
  muMu ~ dnorm(0,1/100)

  for (j in 1:k) {
    mu[j] ~ dnorm(muMu, tauMu)
    for (i in 1:n[j]) {
      y[j,i] ~ dnorm(mu[j], tau)
    }
  }

  sigma2 <- 1 / tau
  sigma2mu <- 1 / tauMu
}
```

Example (Estimating piglet weight for multiple litters)

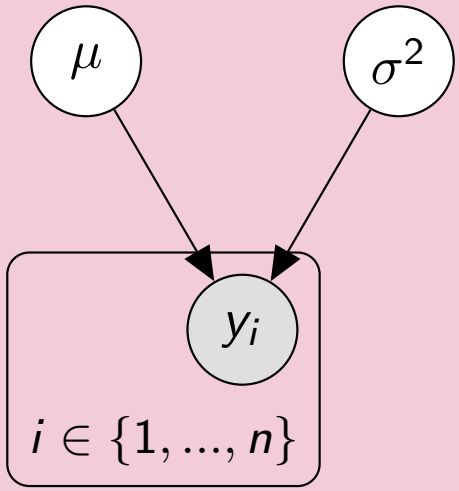
i=1	i=2	i=3	i=4	i=5	i=6	i=7	i=8
2.0	3.5	3.3	3.2	2.6	3.1	2.6	2.5
2.8	2.8	3.6	3.3	2.6	2.9	2.2	2.4
3.3	3.2	2.6	3.2	2.9	3.1	2.2	3.0
3.2	3.5	3.1	2.9	2.0	2.5	2.5	1.5
4.4	2.3	3.2	3.3	2.0		1.2	
3.6	2.4	3.3	2.5	2.1		1.2	
1.9	2.0	2.9	2.6				
3.3	1.6	3.4	2.8				
2.8		3.2					
1.1		3.2					
2.84	2.66	3.18	2.98	2.37	2.90	1.98	2.35

We are interested in estimating/comparing expected piglet weight for several litters, each litter from a different pig.

Example (Estimating piglet weight for multiple litters)

Model suggestion 1:

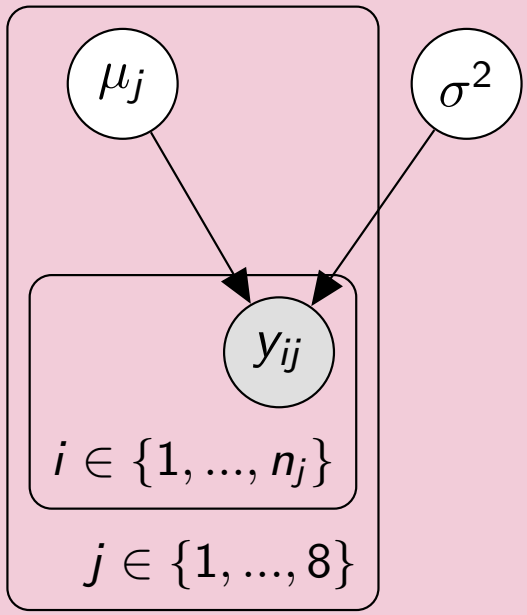
$y_i \sim N(\mu, \sigma^2)$, $\mu \sim N(\mu_0, \sigma_0^2)$, $\sigma^2 \sim IG(\alpha_0, \beta_0)$.



Example (Estimating piglet weight for multiple litters)

Model suggestion 2:

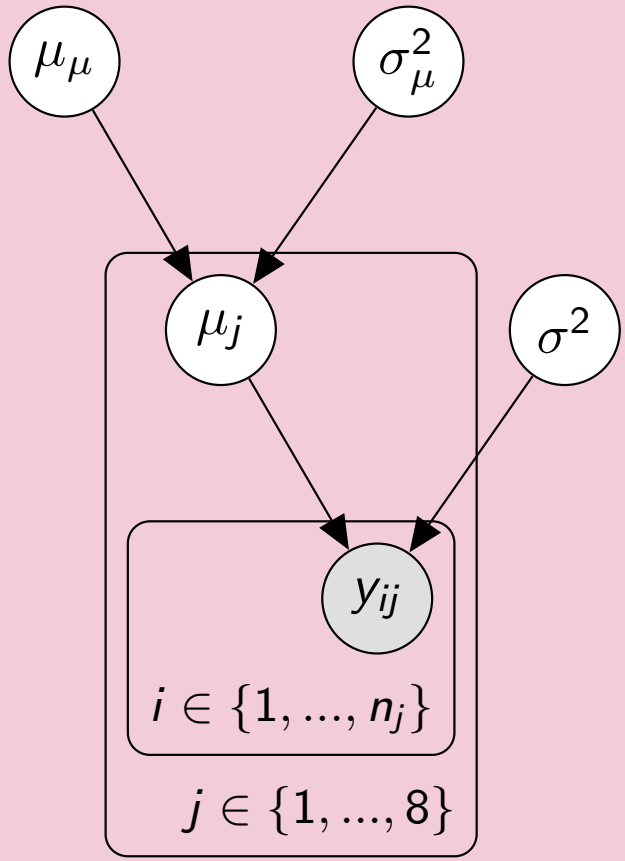
$y_{i,j} \sim N(\mu_j, \sigma^2)$, $\mu_j \sim N(\mu_0, \sigma_0^2)$, $\sigma^2 \sim IG(\alpha_0, \beta_0)$.



Example (Estimating piglet weight for multiple litters)

Model suggestion 3:

$$
\begin{aligned}
Y_{ij} &\sim N(\mu_j, \sigma^2), \\
\mu_j &\sim N(\mu_\mu, \sigma_\mu^2), \\
\sigma^2 &\sim IG(a_0, b_0), \\
\mu_\mu &\sim N(\mu_0, \sigma_0^2), \\
\sigma_\mu^2 &\sim IG(a_0, b_0).
\end{aligned}
$$



Hierarchical (multi-level) models

In practice, we often deal with data from different but similar groups and the models we use should reflect this.

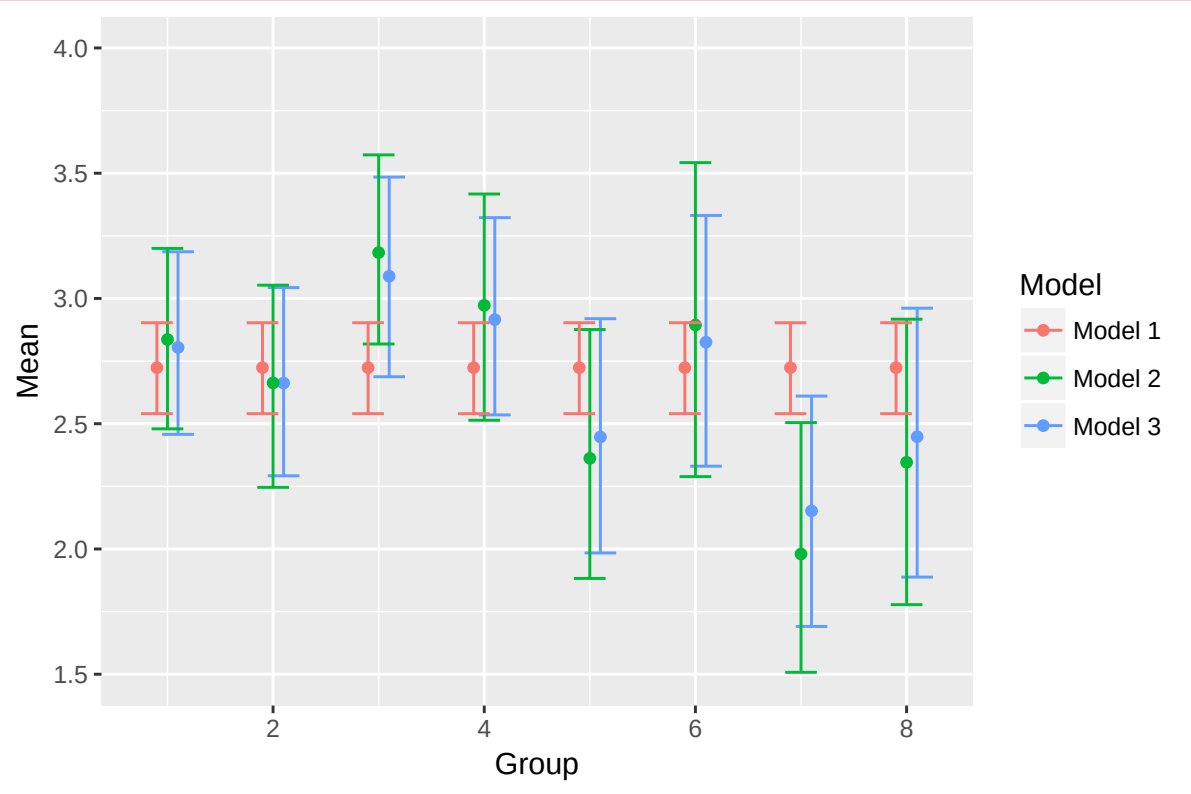
The Bayesian approach to this is to put priors on priors (hyper-priors), resulting in a hierarchical (multi-level) structure of the model.

A hierarchical structure also lets us use more parameters (a more complex model) without overfitting!

Hierarchical models are the quintessential Bayesian approach:

- Group comparison.
- Correction for testing multiple hypotheses.
- Regularizaton.
- ...

Example (Estimating piglet weight for multiple litters)

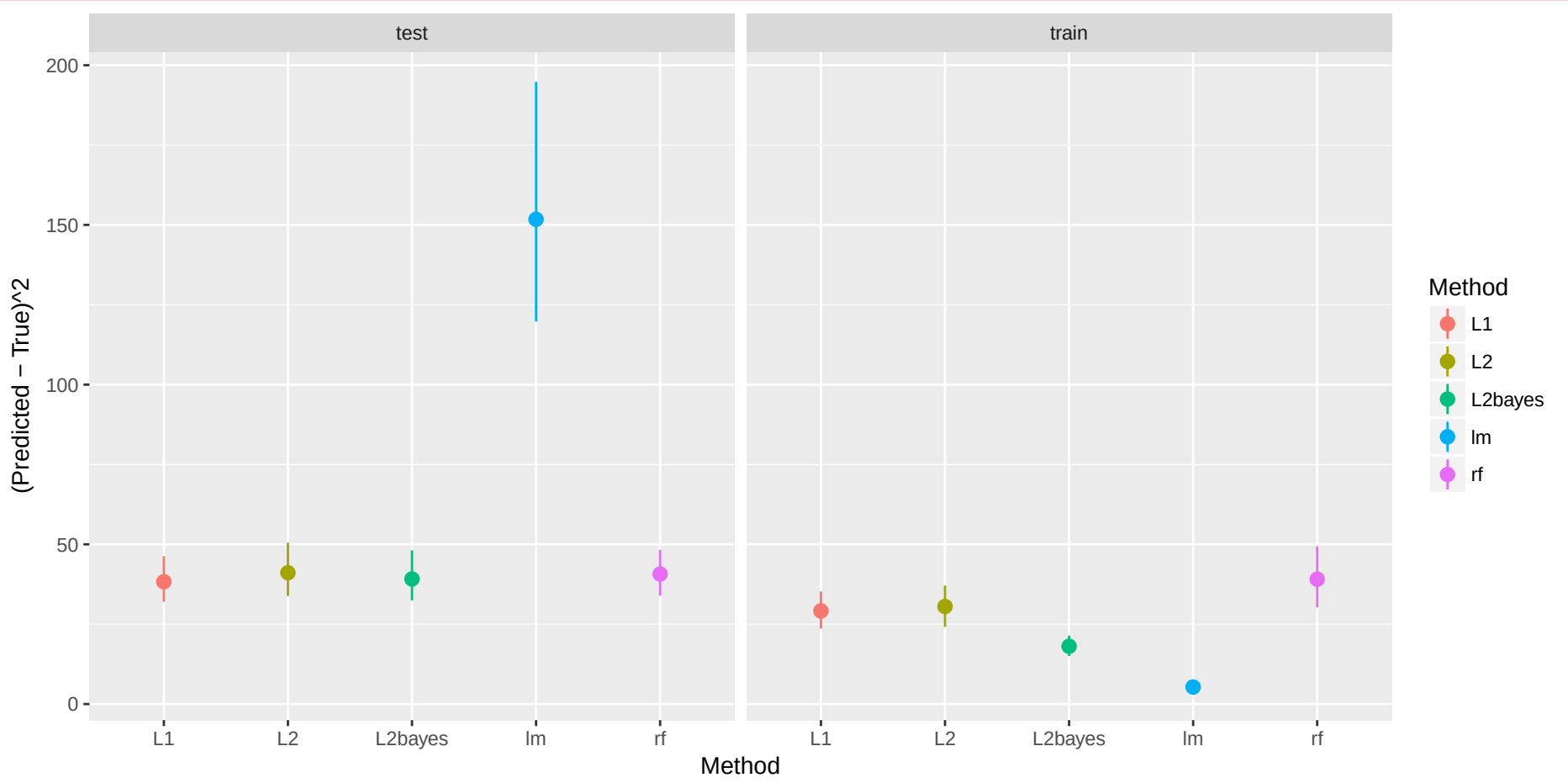


Estimate shrinkage: Same principles alleviate issues with multiple comparisons.

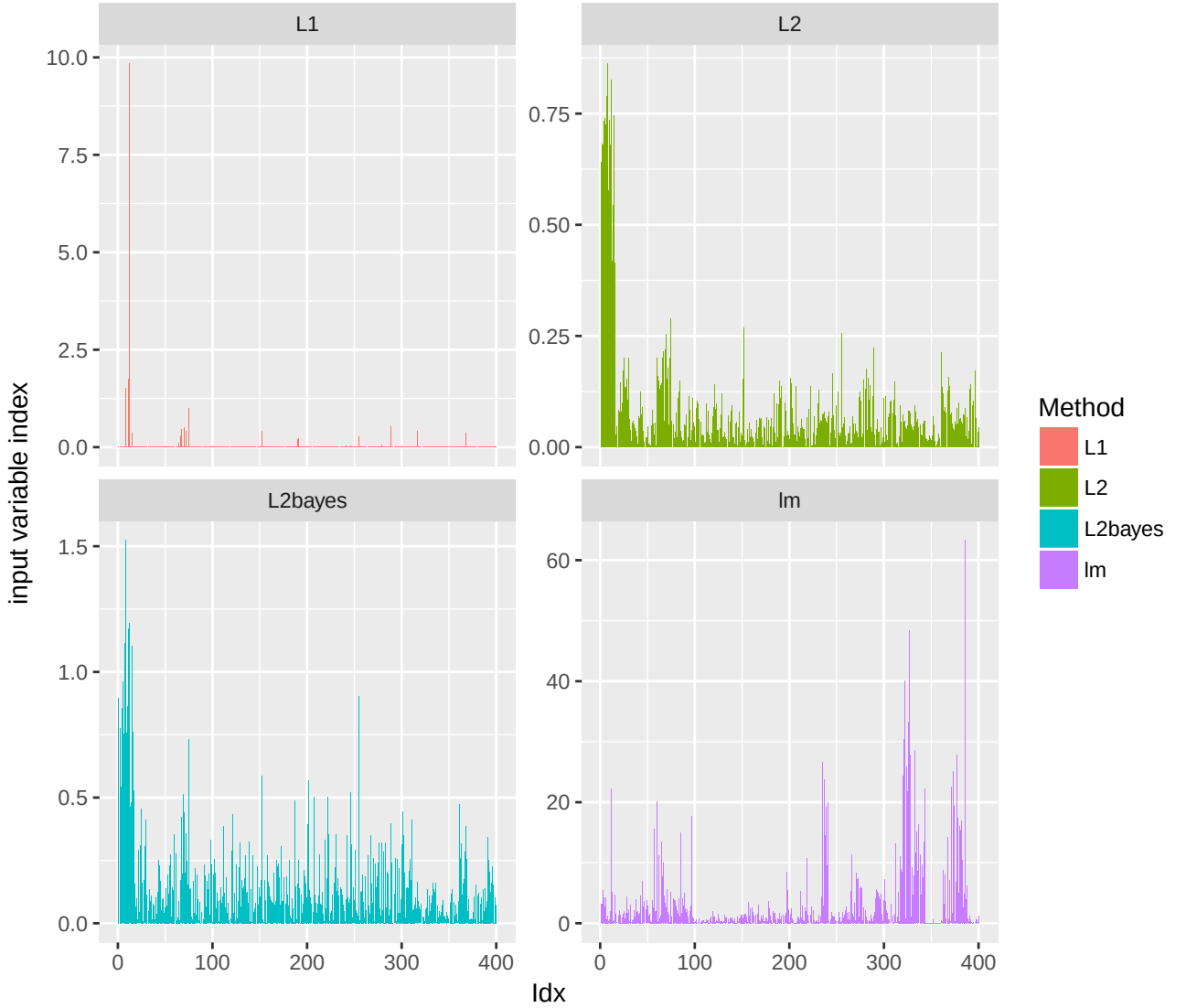
Example (Ozone concentration forecasting)

Task: One-day-ahead forecasting of O_3 concentration from 100s of meteorological, pollution-related, and other input variables.

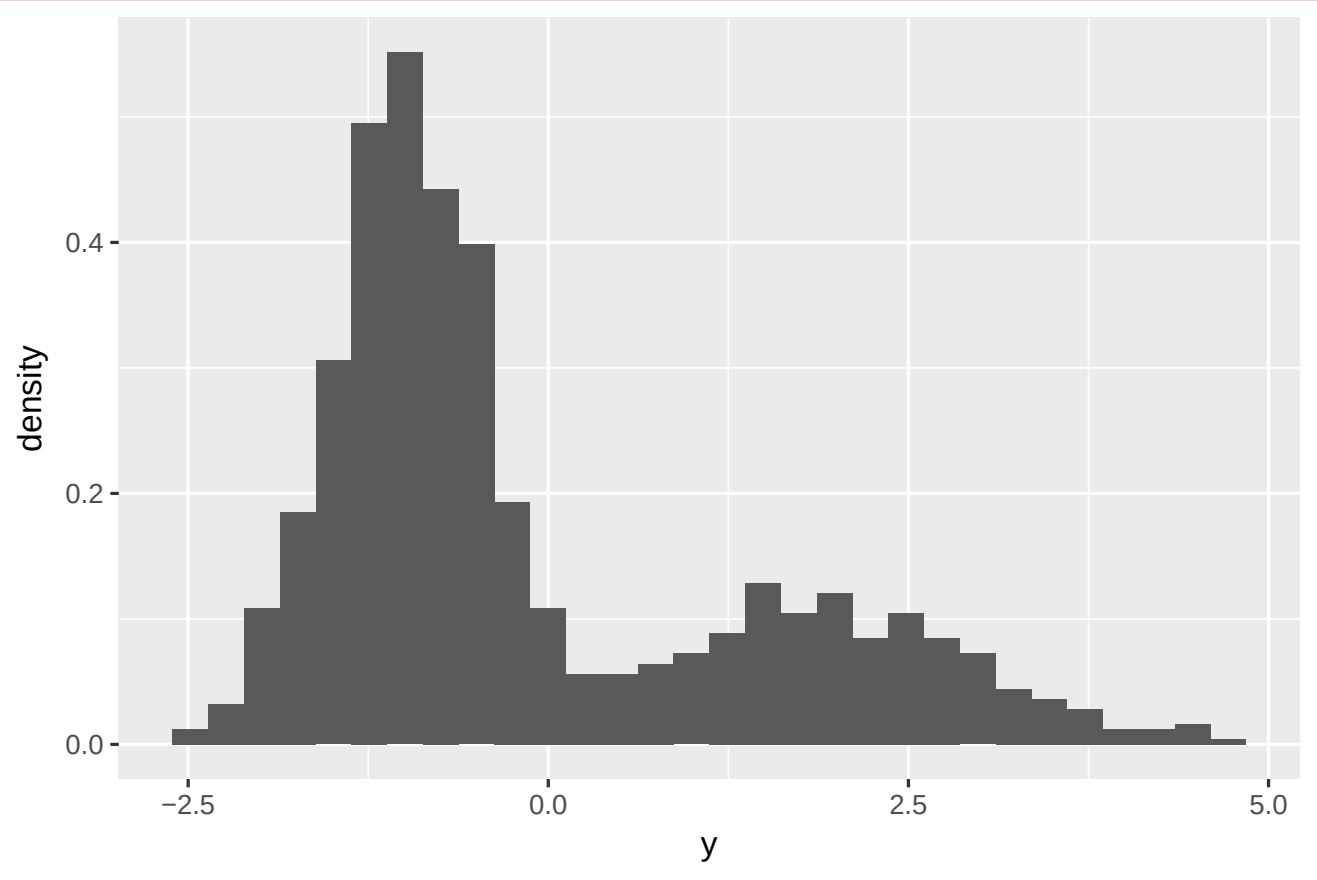
Example (Ozone concentration forecasting - Prediction quality)



Example (Ozone concentration forecasting - Coefficients)



Example (Fitting multimodal densities)

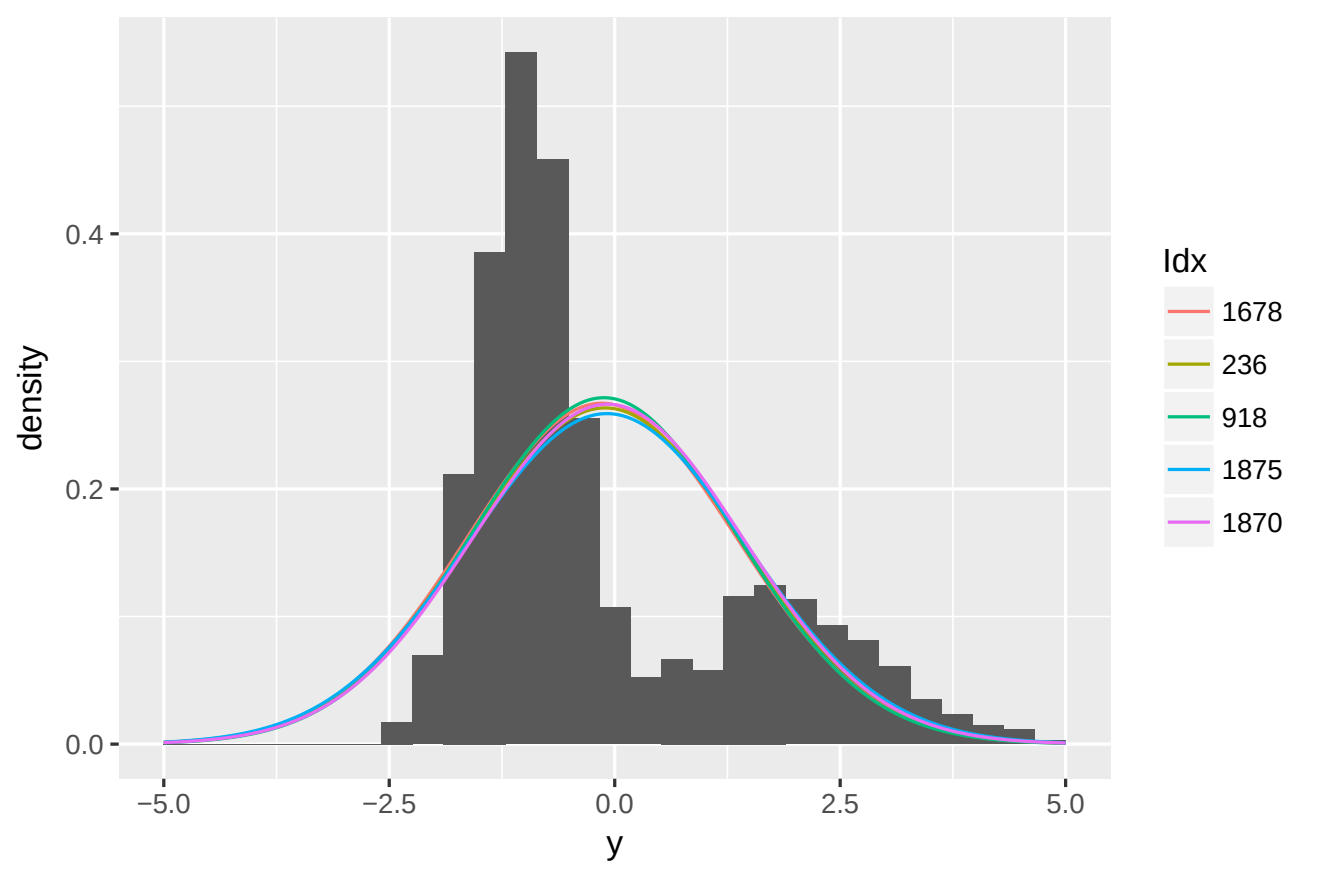


Example (A k-component finite Normal mixture model)

```
data {
  int<lower=1> K; // number of mixture components
  int<lower=1> N;
  real y[N];
}

parameters {
  simplex[K] theta;       // mixing proportions
  real mu[K];             // locations of mixture components
  real<lower=0> sigma[K]; // scales of mixture components
}

model {
  real ps[K];              // temp for log component densities
  sigma ~ cauchy(0,2.5);
  mu ~ normal(0,10);

  for (n in 1:N) {
    for (k in 1:K) {
      ps[k] = log(theta[k]) + normal_lpdf(y[n] | mu[k], sigma[k]);
    }
    target += log_sum_exp(ps);
  }
}

generated quantities {
// only goal is for the model to output the log-likelihood
  real ps[K];
  real log_lik[N];
  for (n in 1:N) {
    for (k in 1:K) {
         ps[k] = log(theta[k]) + normal_lpdf(y[n] | mu[k], sigma[k]);
    }

    log_lik[n] = log_sum_exp(ps);
  }
}
```

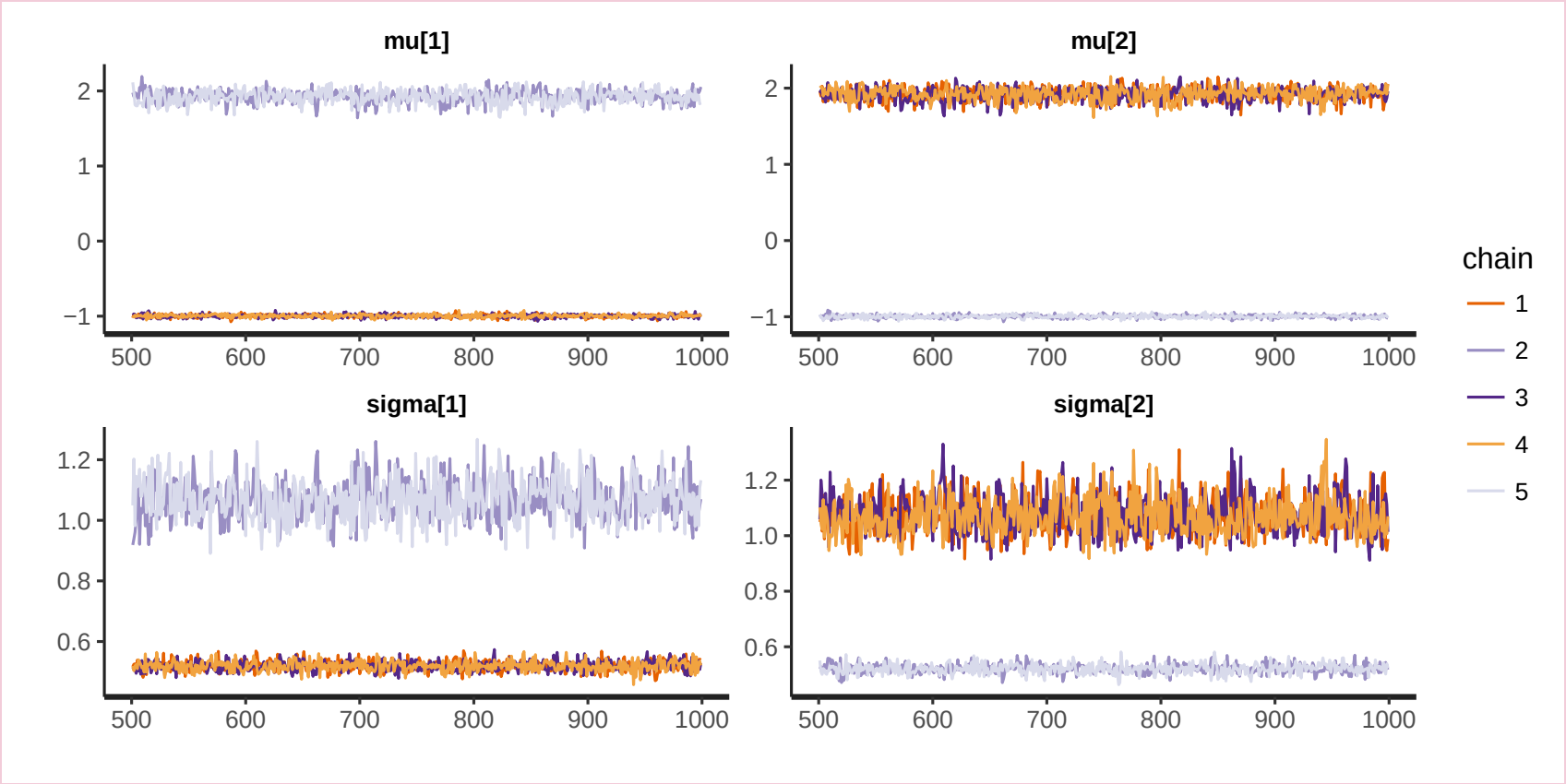
Example (K=1)

A few samples from the posterior:



Example (K=2)

Traceplot (5 chains):



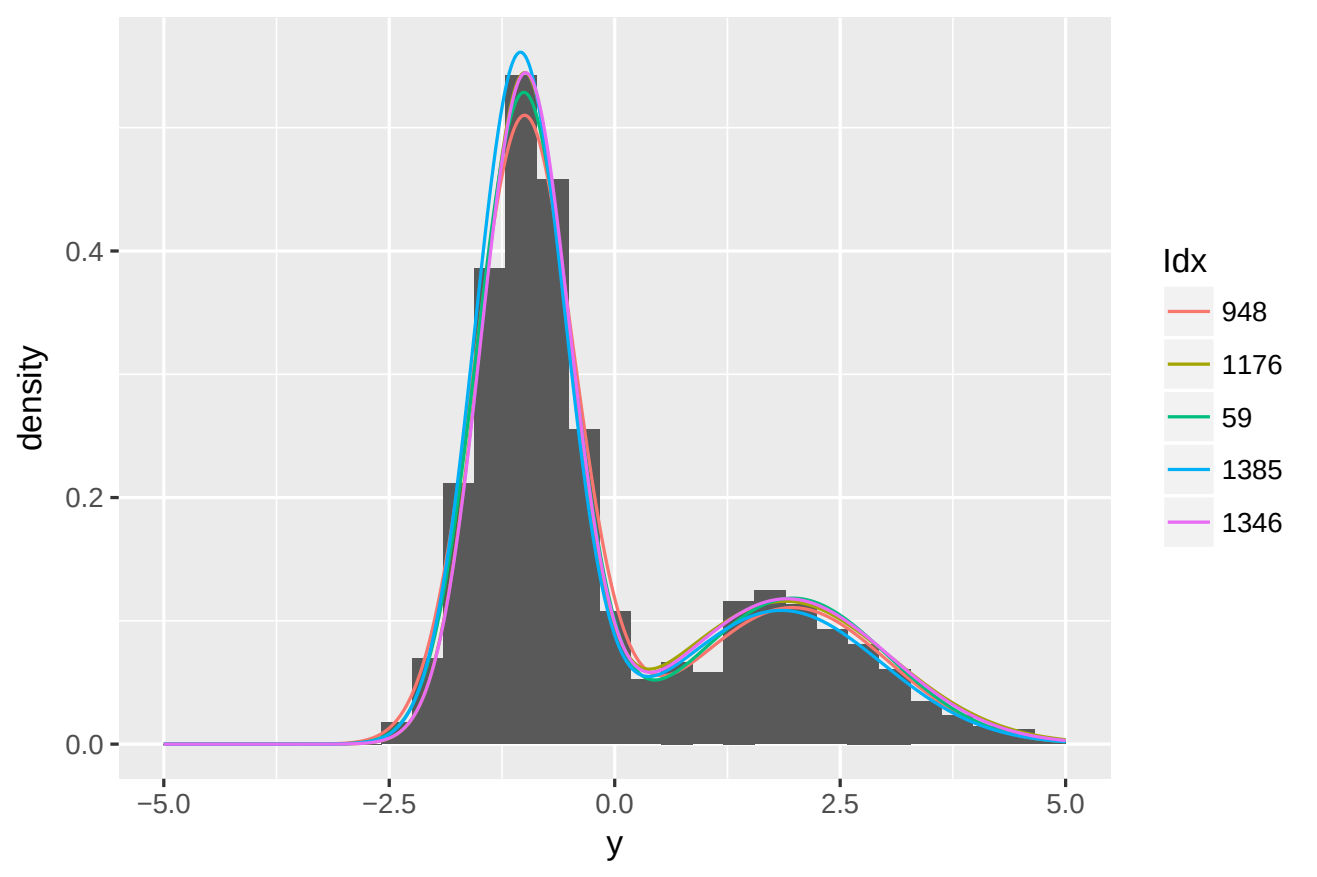
Example (K=2)

Summary (5 chains):

	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
theta[1]	0.54	0.12	0.19	0.28	0.31	0.68	0.70	0.73	3	12.13
theta[2]	0.46	0.12	0.19	0.27	0.30	0.32	0.69	0.72	3	12.13
mu[1]	0.17	0.91	1.43	-1.04	-1.00	-0.98	1.90	2.05	3	26.04
mu[2]	0.75	0.90	1.43	-1.03	-0.99	1.83	1.94	2.06	3	21.62
sigma[1]	0.74	0.17	0.27	0.49	0.52	0.54	1.04	1.18	3	6.44
sigma[2]	0.85	0.17	0.28	0.49	0.53	1.01	1.08	1.20	3	5.41
ps[1]	-4.60	2.04	3.27	-9.95	-8.18	-2.05	-1.96	-1.84	3	5.82
ps[2]	-5.86	2.02	3.25	-10.05	-8.65	-7.51	-1.99	-1.86	3	4.80

Example (K=2)

A few samples from the posterior:



Example (Model quality)

MSE:

	Estimate	SE
model1	2.32	0.10
model2	3.09	0.18

Example (Model quality)

```
MSE:
         Estimate    SE
model1       2.32  0.10
model2       3.09  0.18

looic (approximate LOOCV log score):
         Estimate    SE
model1     3684.1  45.3
model2     3049.1  57.2
```