# Ascent-Based Monte Carlo EM

Galin Jones
University of Minnesota

Collaborators: Brian Caffo (Johns Hopkins)
Wolfgang Jank (U. of Maryland)
Ronald Neath (U. of Minnesota)

Outline

- EM
- MCEM
- Automated MCEM
- Ascent-Based MCEM
- Examples

Notation

- $Y$ is a vector of observed data
- $U$ is a vector of "missing" data
- Joint model $f_{Y,U}(y, u; \lambda)$

The Goal

Find $\lambda$ maximizing

$$L(\lambda; y) = \int_{\mathcal{U}} f_{Y,U}(y, u; \lambda) du$$

Goal: Find $\lambda$ maximizing

$$L(\lambda; y) = \int_{\mathcal{U}} f_{Y,U}(y, u; \lambda) du$$

A Few Methods for Maximizing $L$

- Analytical Methods
- Numerical Methods
- Monte Carlo ML / Bridge Sampling
- EM / Monte Carlo EM

## The EM Algorithm

- EM is an algorithm for maximizing

$$L(\lambda; y) = \int_{\mathcal{U}} f_{Y,U}(y, u; \lambda) du$$

- $\lambda^{(t-1)} \longrightarrow \lambda^{(t)}$ by maximizing (wrt $\lambda$)

$$Q(\lambda, \lambda^{(t-1)}) = E\left[\log f_{Y,U}(y, u; \lambda) \,|\, y, \lambda^{(t-1)}\right]$$

- Ascent property:

$$Q(\lambda^{(t)}, \lambda^{(t-1)}) \geq Q(\lambda^{(t-1)}, \lambda^{(t-1)})$$

$$\Rightarrow L(\lambda^{(t)}; y) \geq L(\lambda^{(t-1)}; y)$$

- $\lambda^{(t)} \to \lambda$ as $t \to \infty$ (under regularity)

- Usually requires numerical maximization of $Q$

- Versions of EM (GEM algorithms) do not require that $Q$ be maximized, just

$$Q(\lambda^{(t)}, \lambda^{(t-1)}) \geq Q(\lambda^{(t-1)}, \lambda^{(t-1)})$$

(EM Gradient Algorithm)

Why EM?

- $Q$ may be easier to handle than $L(\lambda; y)$

  Often have separate maximizations for different components of $\lambda$.

- Numerical Stability

  $\log L$ often looks like $\sum_i \log \int \prod_j (\text{stuff})$

  but

  $Q$ often looks like $\sum_i \sum_j \int \log(\text{stuff})$

- EM is simple to program

Why not EM?

"EM is slow"

This can be true but I will give an example later accelerating deterministic EM with Monte Carlo EM

Monte Carlo EM

- More often than not

$$Q(\lambda, \lambda^{(t-1)}) = E\left[\log f_{Y,U}(y, u; \lambda) \,|\, y, \lambda^{(t-1)}\right]$$

  is analytically intractable.

- Approximate $Q$ with Monte Carlo methods. Draw $U_1, U_2, \ldots, U_M \sim f_{U|Y}(u|y; \lambda^{(t-1)})$ OR $\Phi = \{U_1, U_2, \ldots, U_M\}$ is an ergodic Markov chain with invariant density $f_{U|Y}(u|y; \lambda^{(t-1)})$

$$\begin{aligned}\tilde{Q}(\lambda, \lambda^{(t-1)}) :=& \frac{1}{M}\sum_{i=1}^{M} \log f_{Y,U}(y, U_i; \lambda) \\ \rightarrow& Q(\lambda, \lambda^{(t-1)}) \quad \text{as} \quad M \rightarrow \infty\end{aligned}$$

- MCEM maximizes $\tilde{Q}$ to obtain $\tilde{\lambda}^{(t)}$

Obvious Question: What Monte Carlo sample size should be employed? That is, $M =$?

$\tilde{\lambda}^{(t)} \not\to \lambda$ if $M$ is constant across MCEM steps.

MCEM does not automatically inherit the ascent property.

Large Monte Carlo sample sizes early on in the algorithm are wasteful.

Small Monte Carlo sample sizes late in the algorithm are wasteful.

Automated MCEM: Automates the choice of $M$ across MCEM steps – obtain efficient use of Monte Carlo resources and user time

Booth and Hobert (*JRSSB* 1999)

First automated MCEM algorithm.

Algorithm:

1. Estimate the (multivariate) Monte Carlo error in $\tilde{\lambda}^{(t)}$ conditional on $\tilde{\lambda}^{(t-1)}$

2. Calculate an (asymptotic) confidence ellipsoid around $\tilde{\lambda}^{(t)}$

3. If the ellipsoid contains $\tilde{\lambda}^{(t-1)}$ then $\tilde{\lambda}^{(t)}$ is considered to be swamped with Monte Carlo error and the sample size is increased at the next iteration

Comments:

1. How should the Monte Carlo sample size be updated?

2. Booth and Hobert saw problems with estimating the multivariate Monte Carlo error when using MCMC.

3. Stopping rule?

4. Reparameterization leads to a different algorithm.

5. Does not inherit the ascent property.

6. Requires additional simulation to obtain a stable estimate of the information.

Toy Example

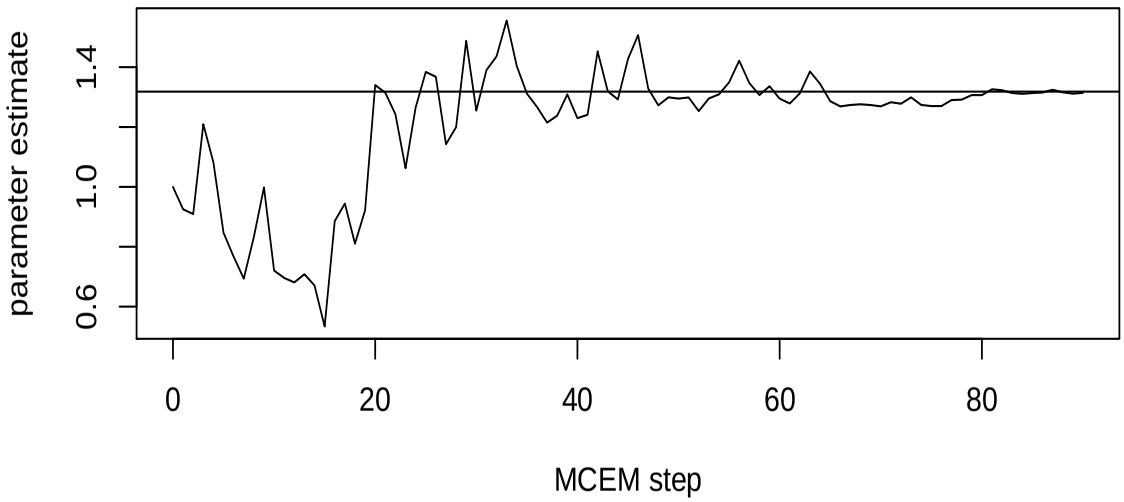
Model:

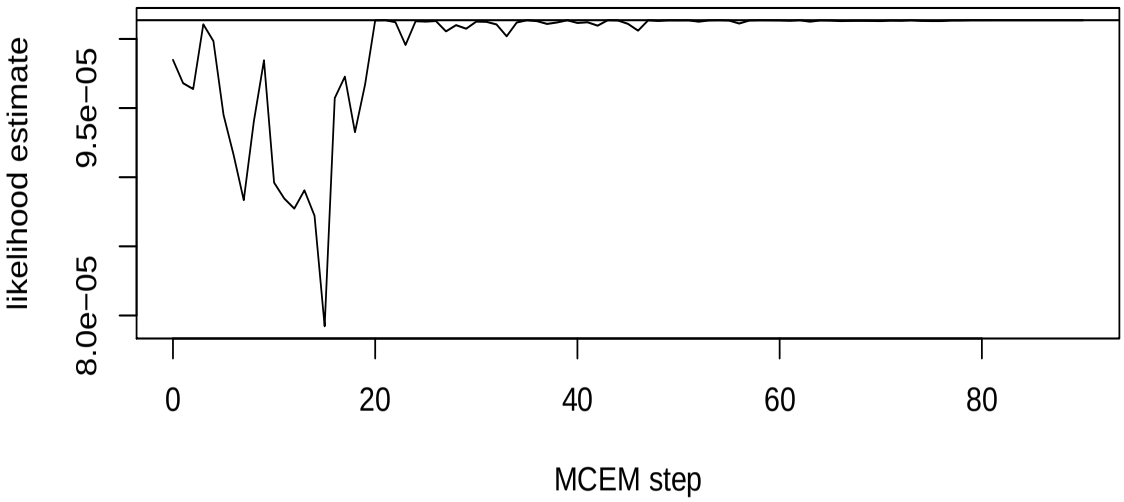
$$Y_i|u_i \sim \mathrm{N}\ (u_i, 1)$$
$$u_i \sim\ \mathrm{N}(0, \lambda)$$

Simulated data (with $\lambda = 1$):

$$0.3365, -2.6339, 0.9080, 1.8898, -0.3811$$

MLE: $\lambda = 1.3183$


parameter estimate
0.6
1.0
1.4
0
20
40
60
80
MCEM step


likelihood estimate
8.0e−05
9.5e−05
0
20
40
60
80
MCEM step

Ascent-Based MCEM

Basic Idea: Make sure the ascent property holds with high probability before moving to the next EM step.

Let

$$\Delta Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) := Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) - Q(\tilde{\lambda}^{(t-1)}, \tilde{\lambda}^{(t-1)})$$

Remember that if

$$\Delta Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) > 0$$

then

$$L(\tilde{\lambda}^{(t)}) \geq L(\tilde{\lambda}^{(t-1)})$$

But we can't calculate $Q$ so we use

$$\Delta \tilde{Q}(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)})$$

Want $\Delta Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) > 0$ (Ascent!)

We show that as $M \to \infty$

$$\Delta\tilde{Q}(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) - \Delta Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) \to \mathrm{N}(0, \sigma^2)$$

$\sigma^2$ depends on the sampling mechanism!

Given an easily computed and consistent estimate, $\hat{\sigma}^2$, of $\sigma^2$ it is easy to form an asymptotic lower bound for $\Delta Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)})$

$$\mathrm{LB} = \Delta\tilde{Q}(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) - z_\alpha \mathrm{ASE}$$

$$\mathrm{LB} < \Delta Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) \text{ w.p. } 1 - \alpha$$

$$\Delta\tilde{Q}(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) - \Delta Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) \rightarrow \text{N}(0, \sigma^2)$$

Stopping Rules:

We want to stop when the marginal likelihood stabilizes. Set

$$\text{UB} = \Delta\tilde{Q}(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) + z_\gamma \text{ASE}$$

And

$$\text{UB} > \Delta Q(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) \text{ w.p. } 1 - \gamma$$

A convenient stopping rule is to wait until UB is below some prespecified constant.

The Algorithm: $\tilde{\lambda}^{(t-1)} \rightarrow \tilde{\lambda}^{(t)}$

1. Obtain $\{U_i\}_{i=1}^{M}$
2. Estimate $\tilde{\lambda}^{(t,M)}$ and LB
3. LB $> 0 \Rightarrow L(\tilde{\lambda}^{(t,M)}) \geq L(\tilde{\lambda}^{(t-1)})$
   $\Rightarrow$ set $\tilde{\lambda}^{(t)} = \tilde{\lambda}^{(t,M)}$
4. Otherwise
   a. Draw $\{U_i\}_{i=M}^{2M}$
   b. Estimate $\tilde{\lambda}^{(t,2M)}$ and LB
   c. Go to 3

Continue until UB $< \epsilon$.

Comments:

- Large Monte Carlo sample sizes early on in the algorithm are wasteful.
- Small Monte Carlo sample sizes late in the algorithm are wasteful.
- Inflated Type I error rate due to sequential decisions.

Solution:

Use a power calculation. That is, choose the starting sample size for the next MCEM iteration so that we are likely to detect a change in the $Q$-function at least as large as that detected in the previous step.

Toy Example

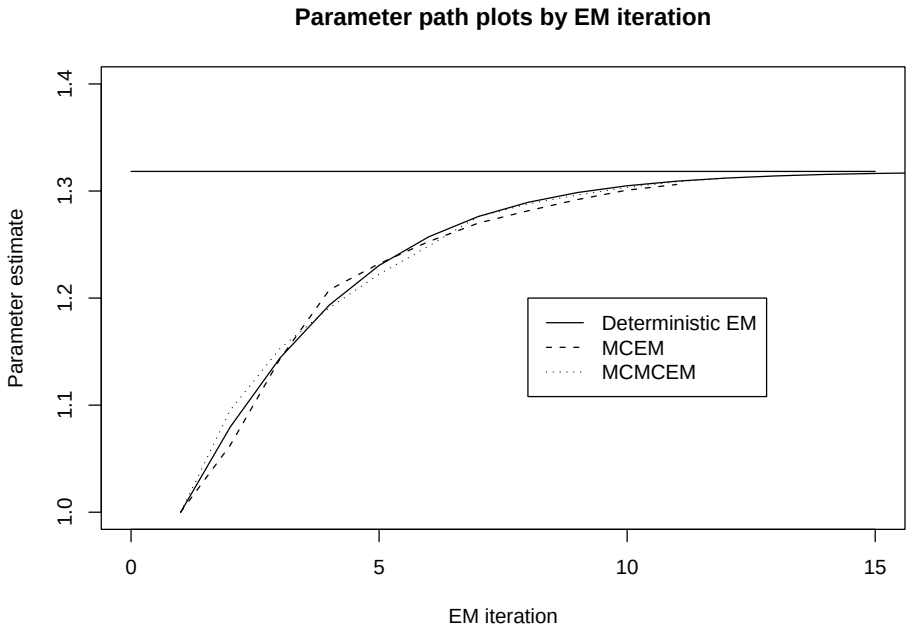
Model:

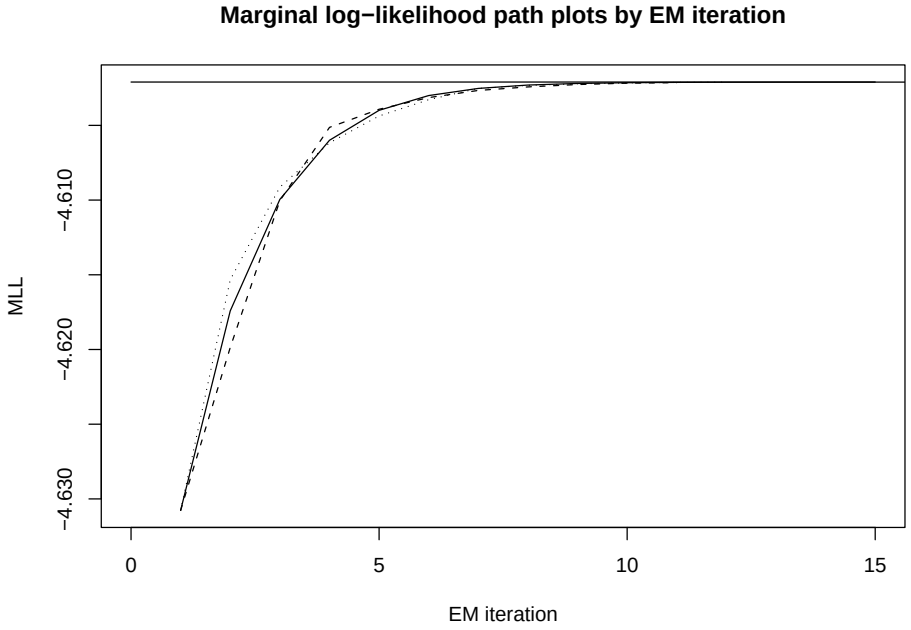
$$Y_i|u_i \sim \mathrm{N}\ (u_i, 1)$$
$$u_i \sim\ \mathrm{N}(0, \lambda)$$

Simulated data (with $\lambda = 1$):

$$0.3365, -2.6339, 0.9080, 1.8898, -0.3811$$

MLE: $\lambda = 1.3183$


Parameter path plots by EM iteration
Parameter estimate
1.0
1.1
1.2
1.3
1.4
0
5
10
15
EM iteration
Deterministic EM
MCEM
MCMCEM


Marginal log−likelihood path plots by EM iteration
MLL
−4.630
−4.620
−4.610
0
5
10
15
EM iteration

Comments:

Ascent-Based MCEM

- is invariant to reparameterizations
- makes the choice of stopping rules less critical
- uses univariate calculations–this makes a lot of the computations easier and allows us to handle MCMC sampling mechanisms within the same framework
- recovers the ascent property and hence minimizes counterproductive use of simulated data

More Comments:

As our algorithm nears convergence it often requires large Monte Carlo samples.

*Complaint:* This algorithm seems to take longer than other MCEM algorithms.

*Response:* The other algorithms are wasting your time by

- Accepting parameter estimates that decrease the likelihood.
- Using Monte Carlo Sample sizes that are too small.
- Prematurely claiming convergence. That is, stopping rules are critical!
- Requiring auxiliary simulation to estimate the information matrix.

A Hybrid Algorithm:

Let $Y_i$ be a vector of $J$ responses for $i = 1, \ldots n$ and let $X$ be a $J \times p$ matrix of covariates.

$$\begin{aligned} Y_i | u_{1i}, u_{2i} &\overset{\text{ind}}{\sim} \text{N}(X u_{1i}, I u_{2i}^{-1}) \\ U_{1i} | u_{2i} &\overset{\text{ind}}{\sim} \text{N}(0, \lambda_1 u_{2i}^{-1}) \\ U_{2i} &\overset{\text{ind}}{\sim} \text{Gamma}(\lambda_2, \lambda_3) \end{aligned}$$

We want to use MCEM to estimate
$\lambda = (\lambda_1, \lambda_2, \lambda_3)^T$.

The $Q$-function is a sum of expectations over the index $i$.

We sum over only a random sample.

We use MCEM until it gets too complicated then we switch to deterministic EM.

Result: the hybrid algorithm is much faster than deterministic EM.

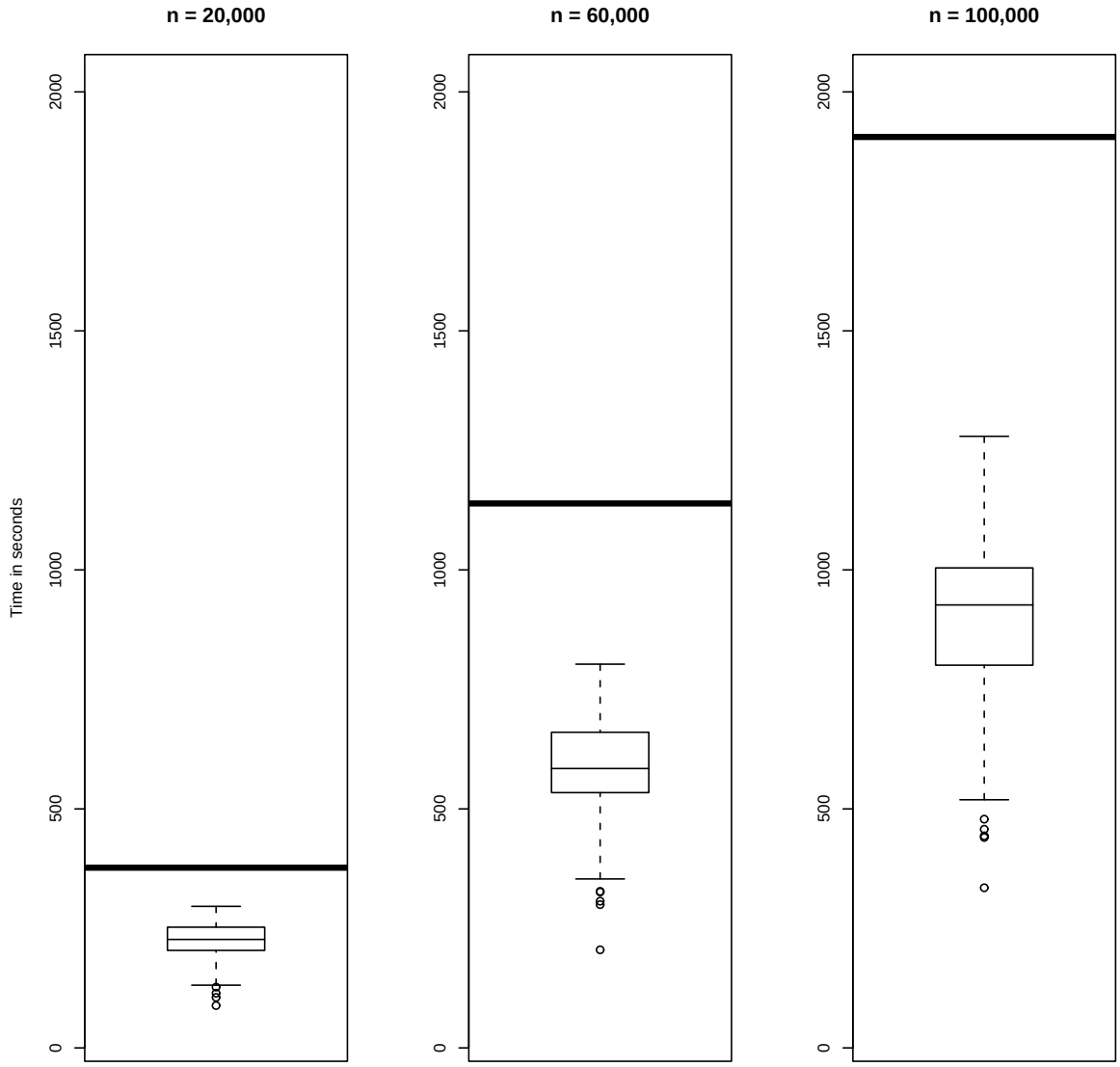



Figure 1: Run time in seconds for 100 applications of MCEM for microarray example for three simulated data sets with different values of $n$. Solid black lines are the run times for the deterministic EM algorithm.

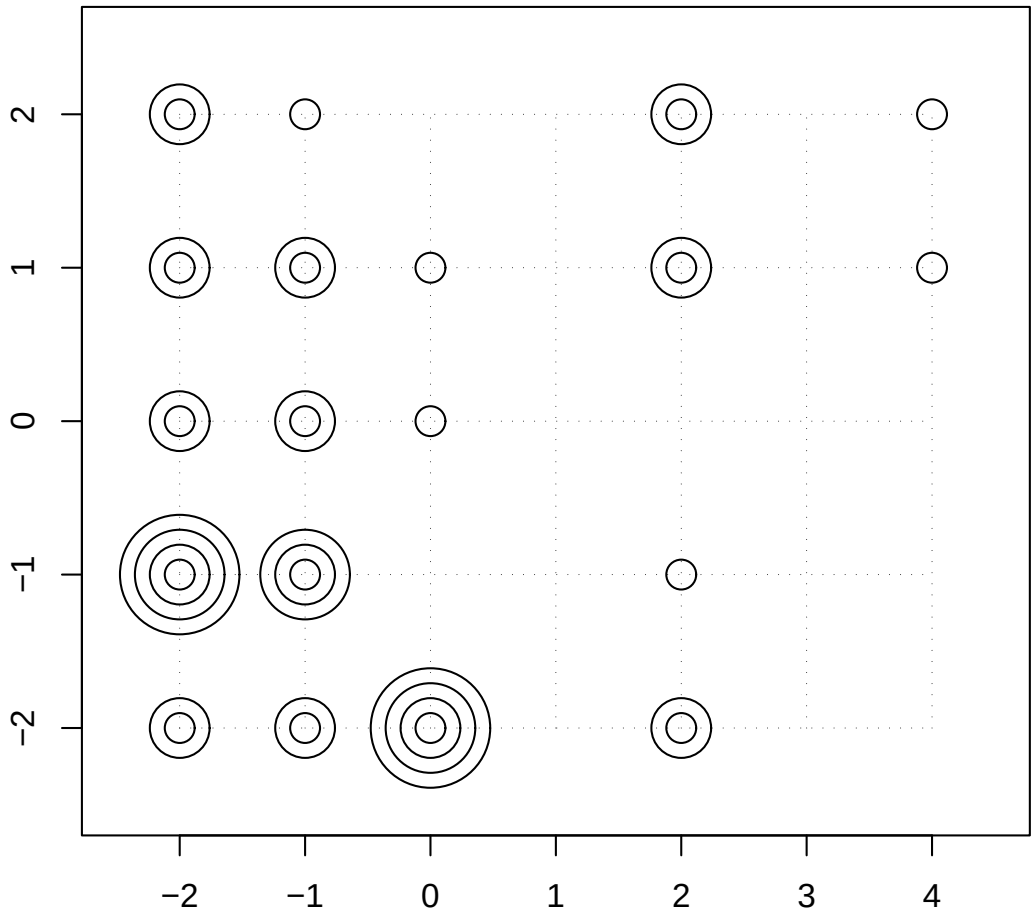



Figure 2: Car thefts and larcenies by intersection. Individual circles represent one theft or larceny at that intersection. Source: http://www.ci.baltimore.md.us/government/police/.

- $x_i$ is a bivariate data point representing the 35 intersections.
- $Y_i$ is the number of auto thefts and larcenies at intersection $x_i$.
- $U = (U_1, \ldots, U_n)^T$ is a vector of random effects.

Model:

$$\begin{aligned} Y_i|u_i &\sim \text{Poisson}(\mu_i) \\ \log(\mu_i) &= \lambda_1 + u_i \\ U &\sim \text{MVN}(0, \Sigma) \\ \sigma_{ij} &= \lambda_2 \exp\{-\lambda_3 \|x_i - x_j\|\} \end{aligned}$$

We want to find the mle of $\lambda = (\lambda_1, \lambda_2, \lambda_3)^T$.

Importance Sampling: $\tilde{\lambda}^{(44)} = (-.087, .776, 4.00)^T$
ESUP: $\tilde{\lambda}^{(56)} = (-.082, .775, 3.99)^T$

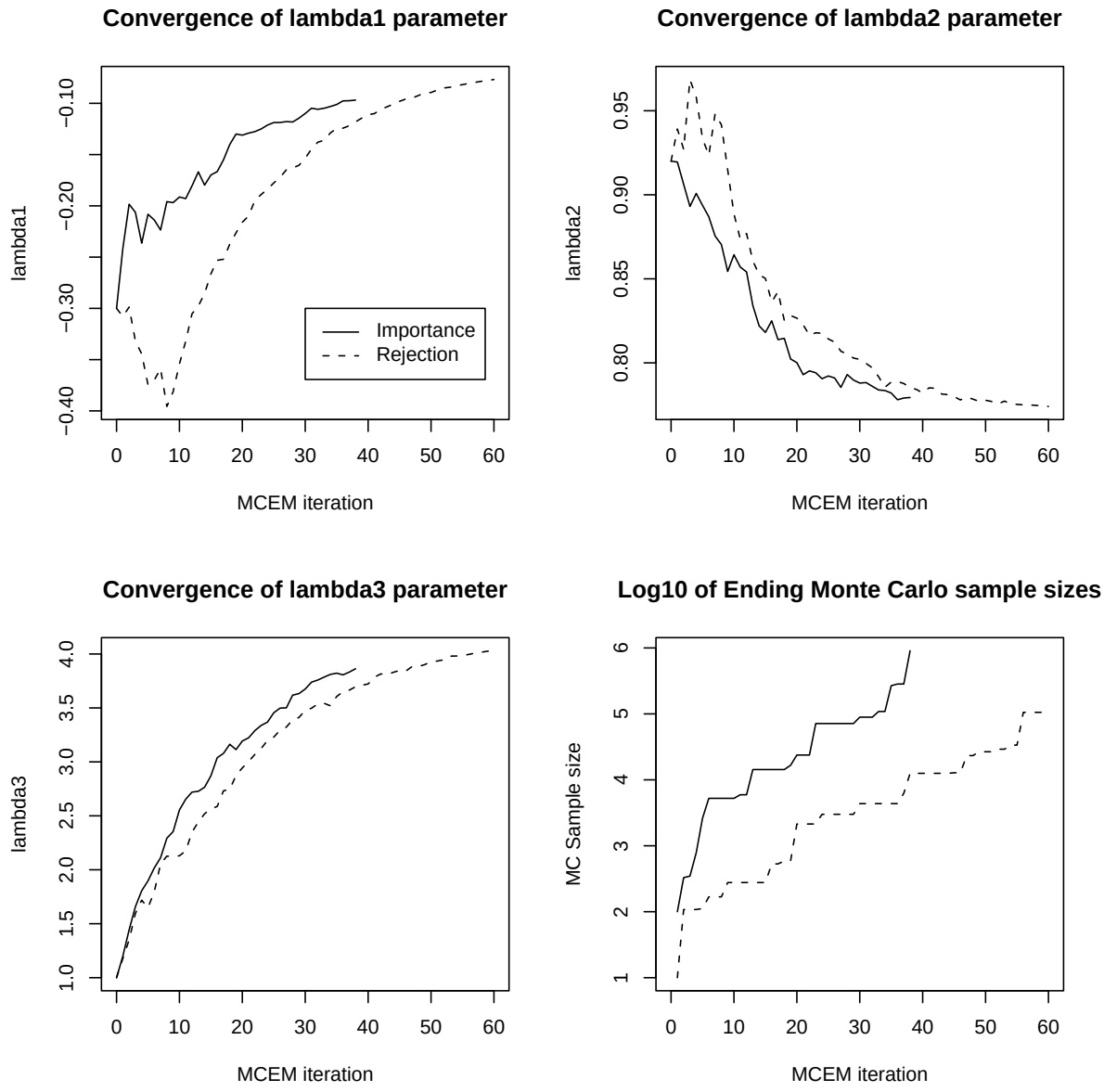



Figure 3: Convergence of parameter estimates for MCEM applied to a spatial application of generalized linear mixed models.

Final Comments: Ascent-based MCEM

- allows independent sampling and MCMC within a unified framework
- recovers the ascent property with a specified probability
- facilitates a stopping rule based on the change in the likelihood
- is invariant to reparameterizations
- appears to inherit a lot of the stability associated with EM
- puts most of the simulation effort into the last step which yields a stable estimate of the information