

Ascent-based MCEM

Brian Caffo

Johns Hopkins Bloomberg School of Public Health

Acknowledgments

- Wolfgang Jank, University of Maryland
- Galin Jones, University of Minnesota
- *Ascent-based MCEM*, to appear in JRSS-B

EM

- Data vector, (y, u) , comprised of observed and missing components resp.
- Complete data density $f_{Y,U}(y, u; \lambda)$
- Goal: maximize

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

- Some examples include:
 - ▶ Estimating parameters for Empirical Bayes analysis
 - ▶ Marginal maximum likelihood for random effect models

Implementing EM

- Given **previous** parameter $\lambda^{(t-1)}$ EM obtains the **current** parameter $\lambda^{(t)}$ by maximizing

$$\begin{aligned} Q(\lambda, \lambda^{(t-1)}) &= E[\log f_{Y,U}(y, u; \lambda) | y, \lambda^{(t-1)}] \\ &= \int_{\mathcal{U}} \log f(y, u; \lambda) f(u | y; \lambda^{(t-1)}) du \end{aligned}$$

with respect to λ

- Under regularity conditions $\lambda^{(t)} \rightarrow \hat{\lambda}$
- Often numerical maximization is required to maximize Q
- GEM algorithms only require

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

Reasons for implementing EM

- Simplification: $Q(\cdot, \cdot)$ is usually easier to handle than $\mathcal{L}(\lambda; y)$

Separate maximizations for different components of λ

Closed form solutions for discrete mixtures

- Numerical stability

▶ $\log \mathcal{L}$ is often of the form $\sum_i \log \int \prod_j$.

▶ Q is often of the form $\sum_i \sum_j \int \log$

- Ascent:

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

implies

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

Arguments against using EM

- Curse of numerical analysis
 - ▶ Fast algorithms are usually not convenient
 - ▶ Convenient algorithms are usually not fast
- EM is a convenient algorithm
- Linear convergence in a neighborhood of the limit

MCEM

- MCEM is useful when $Q(\lambda, \lambda^{(t-1)})$ cannot be calculated exactly
- One solution is to approximate it using Monte Carlo (Wei & Tanner JASA 1990)
- Let $U_1, \dots, U_M \sim f_{U|Y}(u|y; \lambda^{(t-1)})$

$$\tilde{Q}(\lambda, \lambda^{(t-1)}) = \frac{1}{M} \sum_{i=1}^M \log f_{Y,U}(y, U_i; \lambda)$$

- MCEM step maximizes \tilde{Q} instead of Q
- Let $\tilde{\lambda}^{(t)}$ be this maximum.
- $\tilde{\lambda}^{(t)}$ does not converge to $\hat{\lambda}$ if M is kept constant

Automated MCEM algorithms

- Attempt to minimize user input on aspects of the algorithm that can be controlled by the immense volume of Monte Carlo data
- Large sample sizes *early* in the algorithm or small Monte Carlo sample sizes *late* in the algorithm are wasteful
 - ▶ Equitable allocation of Monte Carlo resources throughout the algorithm
 - ▶ Large portion of resources devoted to the final iteration
 - Estimate posterior quantities in empirical Bayes analysis
 - Estimate inverse information for Wald confidence intervals and tests

Booth and Hobert's (JRSSB 1999) algorithm

First automated algorithm for controlling the Monte Carlo sample size within the MCEM algorithm

Algorithm

1. Estimate the MC error in $\tilde{\lambda}^{(t)}$ conditional on the previous estimate $\tilde{\lambda}^{(t-1)}$
2. Calculate an (asymptotic) confidence ellipsoid around $\tilde{\lambda}^{(t)}$
3. If this ellipsoid contains $\tilde{\lambda}^{(t-1)}$ then $\tilde{\lambda}^{(t)}$ is considered to be “swamped with Monte Carlo error” so that the sample size is increased for the next MCEM iteration

Ascent-based MCEM

- Let

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

Recall if

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

then

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

- We can't calculate ΔQ exactly, so our estimate is

$$\begin{aligned} \Delta \tilde{Q}(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) &= \tilde{Q}(\tilde{\lambda}^{(t)}, \tilde{\lambda}^{(t-1)}) - \tilde{Q}(\tilde{\lambda}^{(t-1)}, \tilde{\lambda}^{(t-1)}) \\ &= \frac{\sum_j w(U_j) \log \frac{f_{Y,U}(y, U_j; \tilde{\lambda}^{(t)})}{f_{Y,U}(y, U_j; \tilde{\lambda}^{(t-1)})}}{\sum_j w(U_j)} \end{aligned}$$

Ascent-based MCEM

- Form a lower bound estimate of ΔQ call it LB
- This requires
 - ▶ Asymptotic normality of $\Delta \tilde{Q}$
 - ▶ An asymptotic standard error for $\Delta \tilde{Q}$
- The algorithm remains at the current MCEM step until $LB > 0$
 - ▶ The algorithm repeatedly adds to the Monte Carlo sample until this is true

The algorithm

1. Draw MC sample $\{U_i\}_{i=1}^M$
2. Use this sample to estimate $\tilde{\lambda}^{(t,M)}$
3. Use this sample to estimate LB
4. If $LB > 0$, the evidence from the Monte Carlo data suggests that

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

so that we keep $\tilde{\lambda}^{(t,M)}$ and move on, $\tilde{\lambda}^{(t)} = \tilde{\lambda}^{(t,M)}$

5. Otherwise
 - a. We draw a new sample $\{U_i\}_{i=M+1}^{2M}$, append it to the previous sample
 - b. Estimate $\tilde{\lambda}^{(t,2M)}$ based on this larger sample (use $\tilde{\lambda}^{(t,M)}$ as a starting value)
 - c. Estimate LB using the larger sample
 - d. Goto 4 with $M = 2M$.

Monte Carlo Sample Sizes

- The starting Monte Carlo sample size should be chosen large enough so that the sequential algorithm need only iterate once
 - Better preservation of the coverage probability of the lower bound
 - Iterations within an MCEM step are costly
- Starting Monte Carlo sample sizes that are too large are wasteful early on in the algorithm
- **A Solution:** After a $\tilde{\lambda}^{(t)}$ has been accepted, we choose the starting sample size for the next MCEM iteration based on a power calculation to detect a change in the Q at least as large as the previous MCEM step
- Also force the starting Monte Carlo sample sizes to be non-decreasing

Benefits of Ascent-based MCEM

- Invariant to re-parameterizations
- Approximately recover the ascent property
- Our algorithm generally accepts “lucky jumps” and attempts to correct “unlucky jumps” away from $\hat{\lambda}$
- The sample size calculation and sequential testing procedure mitigates the effects of how you update the sample size
- Univariate calculations makes handling MCMC samples much easier
- Calculating UB , the upper bound on the change in the Q function is a byproduct of the algorithm and can be used as a stopping rule
- Final Monte Carlo sample sizes are large enough so that estimating the observed information is reasonable

Formally handling MCMCEM

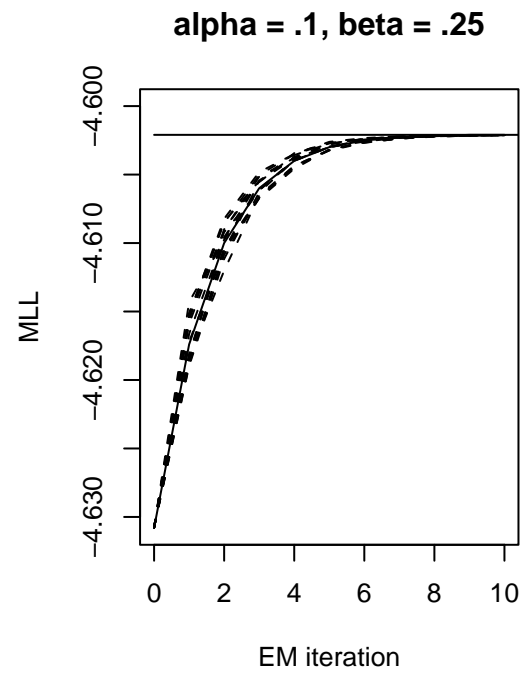
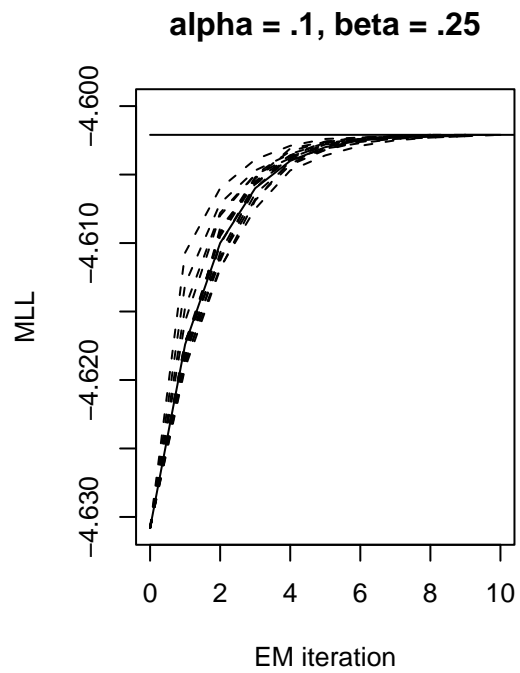
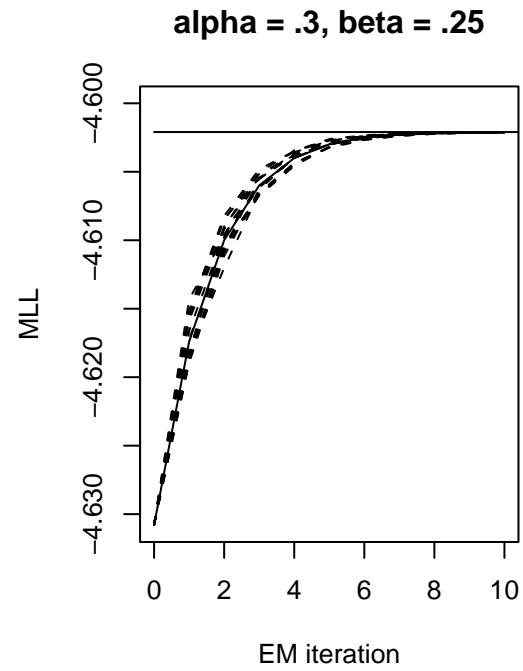
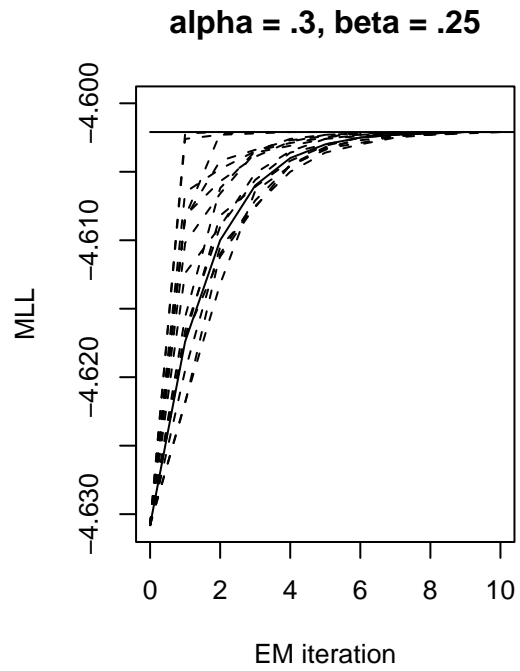
- Since the change in the Q functions, ΔQ is a univariate quantity handling MCMC is easier
- We suggest using regenerative simulation to estimate the Monte Carlo variability in ΔQ
 - Identify points where the Markov chain probabilistically restarts itself
 - Breaks the chain into i.i.d. subchains
 - CLT
 - Consistent estimates of variance of ΔQ
- Prove that ΔQ evaluated at $\tilde{\lambda}^{(t)}$ is still asymptotically normal

Recap

- Goal of ascent-based MCEM is to approximately recover the ascent property of deterministic EM algorithms
- Monte Carlo sample sizes are controlled via a lower bound estimate on the change in the Q function
- Samples are repeatedly appended to until the lower bound is positive

A toy example

- Data: $Y = (.336, -2.634, .908, 1.890, -.381)$
- Model $Y_i | u_i \sim \mathbf{N}(u_i, 1)$
- $U_i \sim \mathbf{N}(0, \lambda)$



Accelerating EM for high throughput data

- As Ascent-based MCEM often appears to follow a path closer to the limit than deterministic EM
- Start an algorithm with Ascent-based MCEM and complete it with deterministic EM

- Consider a model where

$$Y_i | u_{1i}, u_{2i} \sim \text{Normal}(x u_{1i}, I u_{2i}^{-1})$$

$$U_{1i} | u_{2i} \sim \text{Normal}(0, \lambda_1 u_{2i}^{-1})$$

$$U_{2i} \sim \text{Gamma}(\lambda_2, \lambda_3)$$

where $i = 1, \dots, n$ for very large n

- EM is convenient allows a closed form for the matrix λ_1
- Originally this model was applied to genomics data see Caffo et al. (2004) and Liu et al. (2003)

EM acceleration

- The Q function can be written as

$$Q(\lambda, \lambda^{(t-1)}) = \frac{1}{n} \sum E[\log f_{Y,U}(y_i, U_{1i}, U_{2i}; \lambda) | y_i, \lambda^{(t-1)}]$$

- This is equivalent to

$$Q(\lambda, \lambda^{(t-1)}) = E[\log f_{Y,U}(y_{U_3}, U_{1U_3}, U_{2U_3}, U_3; \lambda) | y, \lambda^{(t-1)}]$$

where U_3 is a random index

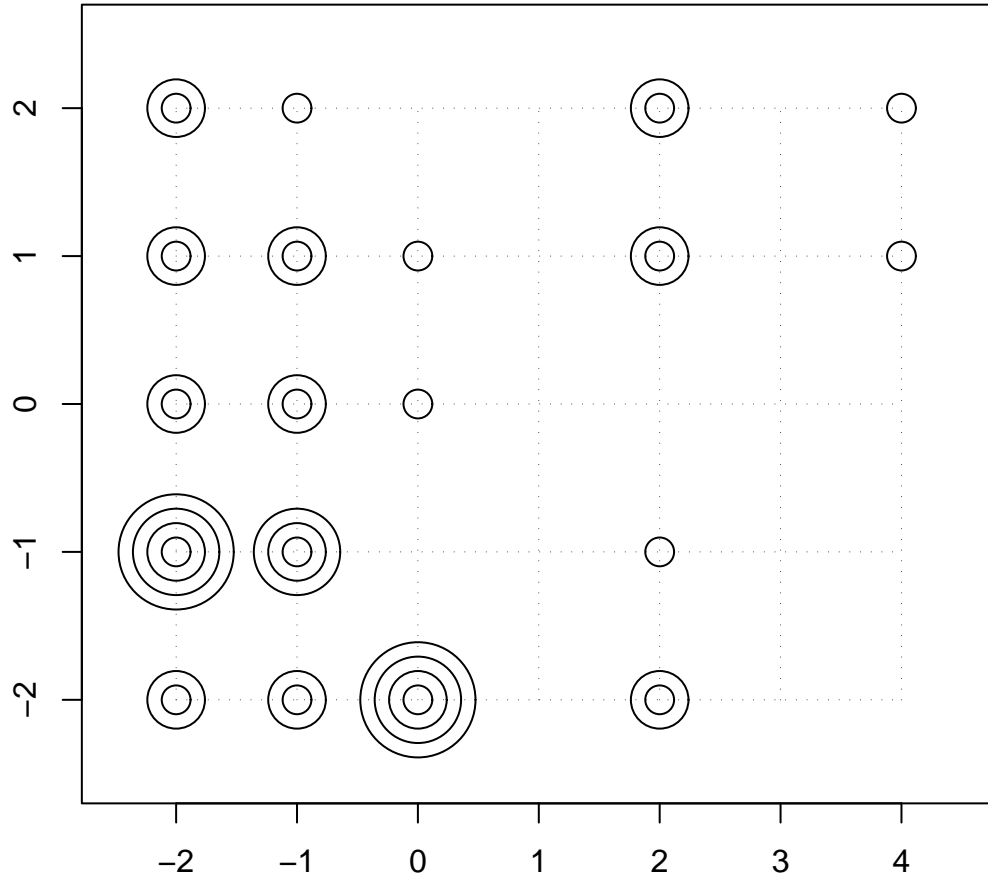
- Ascent-based MCEM on this parameterization randomly samples from a subset of the indices (with replacement)
- Switch to deterministic EM when the Monte Carlo sample size gets large
- This strategy can be used whenever the Q function decomposes into a large number of exchangeable components

Results

- Simulated data with $\lambda_1, \lambda_2, \lambda_3$ obtained from a microarray experiment
- Used the same code for the deterministic portion of the EM and hybrid algorithms
- Compared wall clock time

n	Deterministic		Hybrid			
	EM	min	25	50	75	max
20K	6:17	1:29	3:24	3:47	4:13	4:56
60K	18:60	3:25	8:55	9:44	11:00	13:23
100K	31:46	5:35	13:21	15:27	16:42	21:20

- Improvements will be less if better starting values are used!



Spatial data example

- Nonlinear models with random effects for spatial data are particularly difficult to fit
- y_i be the count at location i
- Let x_i be the corresponding grid point
- We assume that

$$y_i | U_i \sim \text{Poisson}(\mu_i)$$

$$\log \mu_i = \lambda_1 + u_i$$

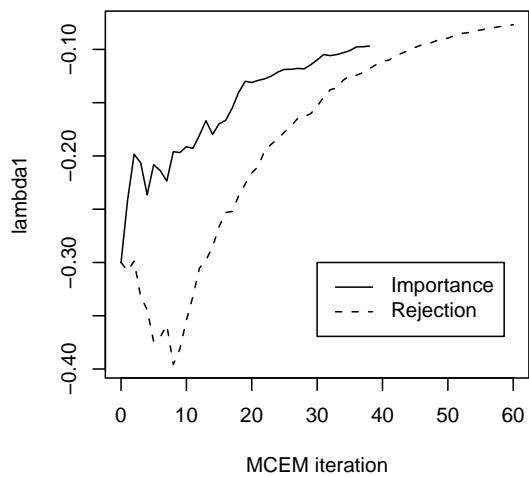
$$u = (u_1, \dots, u_n)^t \sim \text{Normal}(0, \Sigma)$$

$$\Sigma_{ij} = \lambda_2 \exp(-\lambda_3 \|x_i - x_j\|)$$

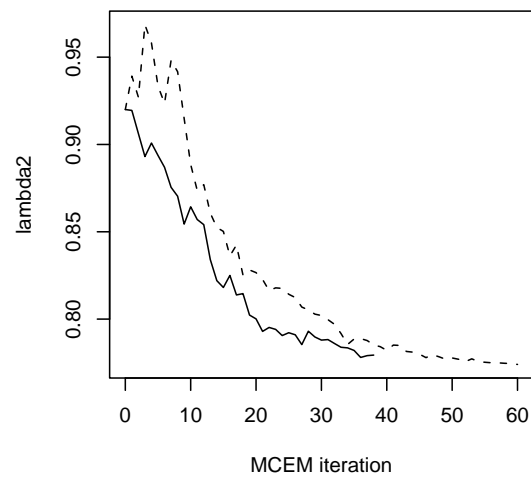
Sampling

- Sampling from the distribution of $U|y$ to perform MCEM is non trivial
- We adopt two schemes
 - ▶ Importance sampling
 - ▶ ESUP accept/reject sampling
- Candidates based on the marginal distribution of U perform very poorly
- In both case the candidate distribution function is a shifted and scaled multivariate t distribution
- Shift and scale by the Laplace approximation to the mean and variance of $U|y$.

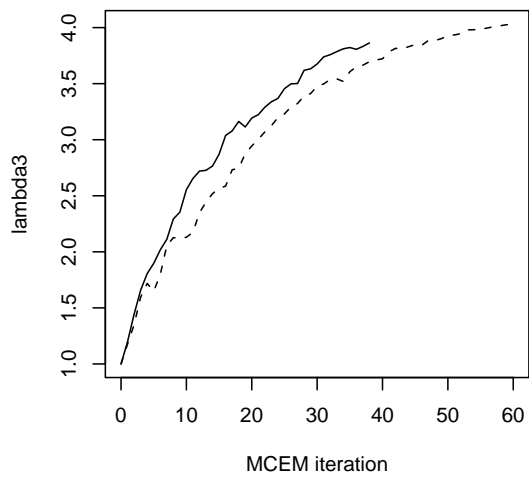
Convergence of lambda1 parameter



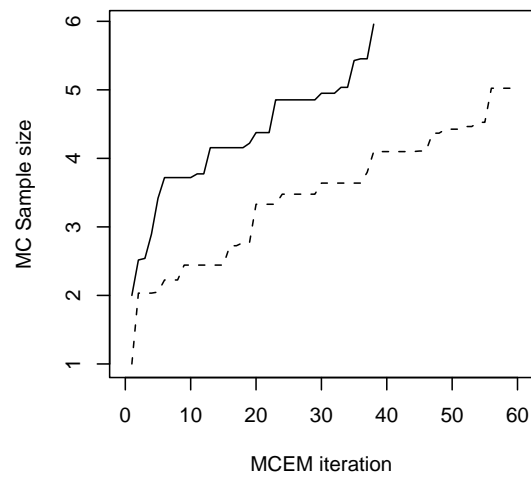
Convergence of lambda2 parameter



Convergence of lambda3 parameter



Log10 of Ending Monte Carlo sample sizes



Discussion

- Controlling the Monte Carlo sample size by the change in the Q function
 - ▶ Computationally easier
 - ▶ Allows MCEM and MCMCEM to be handled within a common framework
 - ▶ Invariant to reparameterizations
 - ▶ Attempts to recover a fundamental property of EM
- Appending to force the Ascent property
 - ▶ Knowingly trades computing time for stability
 - ▶ Causes parameter estimates to follow smooth paths to their limit
 - ▶ Allocates the bulk of computational resources in the final iterations

Discussion

- In a canonical logit-normal example
 - ▶ In repeated applications, on average ABMCEM devoted 55% of the total simulation effort to the final iteration
 - ▶ Though we do not endorse such an approach, this property implied that deterministic stopping rules were more reasonable with ABMCEM than with other MCEM algorithms
 - ▶ Faster algorithms required run times and total simulation efforts equivalent to ABMCEM when considering estimating the observed information matrix in addition to the slope parameters
- No proof of convergence for any automated MCEM algorithm