

Stochastic Variants of EM: Monte Carlo, Quasi-Monte Carlo and More

Wolfgang Jank
Robert H. Smith School of Business
University of Maryland

KEY WORDS: Monte Carlo EM; Stochastic Approximation; Importance Sampling; Rejection Sampling; Markov Chain Monte Carlo; Stopping Rule; Global Optimization;

1. Introduction

Many statistical models involve a combination of observed and unobserved data. Examples include the linear mixed model, the generalized linear mixed model or the hierarchical model. Let y denote a vector of observed (or incomplete) data and let u be a vector of unobserved (or missing) data. The pair (y, u) is often referred to as the complete data. Let $f(y, u; \theta)$ denote the density of the complete data, where θ denotes an unknown parameter vector.

The EM (Expectation-Maximization) algorithm naturally appeals to the situation of unobserved data. One of the problems with EM is that in many contemporary models the expectation step is analytically intractable, leading to integrals that have no closed-form solution. This is especially problematic in situations where the integral is of high dimension. An increasingly popular approach to overcome this problem is to approximate the integral via simulation. This leads to a stochastic EM implementation. In this paper we review some of the challenges associated with stochastic EM implementation and comment on some recent solutions to overcome these challenges.

2. The EM Algorithm and Its Stochastic Implementation

The EM algorithm (Dempster et al., 1977) is an iterative procedure to find the maximum of likelihood functions in incomplete data problems. In each iteration, the EM algorithm performs an expectation and a maximization step. Let $\theta^{(t-1)}$ denote the current parameter value. Then, in the t th iteration of the algorithm, the E-step computes the conditional expectation of the complete data log-likelihood, conditional on the observed data and the current parameter value,

$$Q(\theta|\theta^{(t-1)}) = E \left[\log f(y, u; \theta) | y; \theta^{(t-1)} \right]. \quad (1)$$

This conditional expectation is often referred to as the Q -function since it plays a central role in the EM algorithm. The t th EM update $\theta^{(t)}$ maximizes the Q -function. That is $\theta^{(t)}$ satisfies

$$Q(\theta^{(t)}|\theta^{(t-1)}) \geq Q(\theta|\theta^{(t-1)}) \quad (2)$$

for all θ in the parameter space. This is the M-step. Given an initial value $\theta^{(0)}$, the EM algorithm produces a sequence $\{\theta^{(0)}, \theta^{(1)}, \theta^{(2)}, \dots\}$ that, under regularity conditions (Boyles, 1983; Wu, 1983), converges to $\hat{\theta}$.

What we have described so far is the basic form of the EM algorithm. EM has received widespread popularity inside statistics but also outside the field. One reason for its popularity is that it guarantees an increase in the likelihood function in every iteration. This property is remarkable since it is not shared by other optimization methods. Newton-Raphson, for instance, may well converge to a minimum and it therefore requires additional effort upon termination to verify that the stationary point is indeed a maximizer. EM is also popular because it operates on the log-scale and therefore allows significant analytical simplification, especially for models in the exponential family, and as a consequence enjoys great numerical stability.

Notice that EM is a *deterministic* method in the sense that it converges to the same stationary point if initiated repeatedly from the same starting value. What we will be concerned with in this paper are *stochastic* versions of EM. Stochastic EM versions distinguish themselves from their deterministic origin in that repeat applications from the same starting value will *not* necessarily lead to the same answer. Stochastic EM versions have become popular with the availability of more and more powerful computing. Stochastic EM versions overcome many of the computational limitations that EM encounters in complicated models. In what follows we will describe the most basic stochastic EM implementation. Our point of view is strongly influenced by the Monte Carlo EM algorithm so we are quick to point out that this may not be the only viewpoint. However, it does not matter too much which point of view one takes because, in the end, all stochastic EM versions are related with one another.

Many contemporary models render the Q-function in (1) analytically intractable. By that we mean that the conditional expectation involves an integral which has no closed form solution. Consequently, if one still wants to rely on the EM paradigm for maximization, then one has to approximate the Q-function. Early approaches to this problem involve analytical approximation or quadrature. While these approaches work well in many instances, they fail if the model becomes too complicated. For that reason our focus is on the situation when the Q-function is approximated via simulation. More precisely, we view the problem of approximating the Q-function as one of simulating an intractable integral. This leads to the concept of Monte Carlo and the Monte Carlo EM algorithm in particular.

The Monte Carlo EM (MCEM) algorithm has been around for over 10 years (Wei and Tanner, 1990). MCEM approximates the analytically intractable expectation in (1) by the Monte Carlo average

$$\tilde{Q}(\theta|\theta^{(t-1)}) = \frac{1}{m_t} \sum_{k=1}^{m_t} \log f(y, u_k; \theta), \quad (3)$$

where u_1, \dots, u_{m_t} are simulated from the conditional distribution of the missing data, $f(u|y; \theta^{(t-1)})$. Then, by the law of large numbers, \tilde{Q} will be a reasonable approximation to Q if m_t is large enough. The MCEM proceeds in the same way as its deterministic counterpart, simply replacing Q by \tilde{Q} . We refer to this algorithm as the basic stochastic EM implementation. Other stochastic EM implementations include a stochastic-approximation version of the Q-function (Delyon et al., 1999) or versions with $m_t = 1$ for all t (Celeux and Diebolt, 1992).

3. Problems, Challenges and Some Recent Solutions

In this section we discuss problems and challenges associated with the stochastic implementation of EM. These challenges can be classified into five major categories: simulation, approximation, maximization, iteration and convergence.

3.1 Simulation

The first step in implementing a stochastic EM version involves simulating from the conditional distribution of the missing data. Three basic questions have to be answered in this step. First: What type of

simulation do we want to use? Three different simulation paradigms are available to us: i.i.d. simulation via rejection sampling, independent simulation via importance sampling, or generating dependent samples via MCMC. All of these three paradigms have their own set of advantages and disadvantages. The second question that has to be answered is: How can we use this sampler in the most efficient way? (That is, with the least amount of simulation effort.) While much of the EM literature focuses on finding the correct sampler, there exist approaches (typically outside the literature on EM) that promise a more efficient use of the simulated data using variance-reduction methods like Quasi-Monte Carlo. And the last question to be addressed is: How much simulation do we need? This question can also be asked in a slightly different way: Do we want to keep the amount of simulation constant in every EM iteration? And if not, how do we increase the simulations throughout the EM iterations in an automated way? Naturally, if we do not keep the amount of simulation constant, then we would like to increase them in a way that results in a) the least amount of simulated data at the end of the day (i.e. the shortest computing times) while guaranteeing b) sufficient accuracy of the results.

3.1.1 Type of Simulation

The challenge is to find a sampler that draws from $f(u|y; \theta)$, or at least from a distribution very close to it. We will refer to $f(u|y; \theta)$ as the target distribution. The problem is easier if u is of smaller dimension or, alternatively, breaks-down into several low-dimensional components. On the other hand, finding an appropriate sampler can be complicated if u is of very high dimension. More complex models, involving e.g. interaction terms, spatial and/or longitudinal components, often result in high-dimensional u 's.

Three basic types of simulation are available. Rejection sampling attempts to simulate i.i.d. draws exactly from $f(u|y; \theta)$. While i.i.d. draws are most preferable, this type of sampler is also the most limited one. Rejection sampling can be analytically challenging in the set-up in that it requires calculation of a certain supremum, but this can be alleviated using recent modifications (Caffo et al., 2002). Rejection sampling draws from the target distribution by thinning out samples from a candidate distribution, rejecting those that are not appropriate. Unfortunately, finding good candidate distributions gets harder and harder as the model-complexity increases and as a consequence the acceptance-rate

drops. For that reason, rejection sampling works well only for low-dimensional settings.

An alternative to rejection sampling is importance sampling. In contrast to rejection sampling, importance sampling does not dispose of any of the simulations. Importance sampling draws from a suitably chosen importance distribution and accounts for the discrepancy between the target and the importance distribution by down-weighting the samples via importance weights. The main two advantages of importance sampling are that it produces independent samples and that it uses all of the simulation output. The disadvantage is that its performance depends heavily on the chosen importance sampler. In fact, the efficiency of the method can be disheartening if the sampler is chosen without care (Jank and Booth, 2002). If the target distribution is not too skewed, then decent importance samplers can be obtained via a multivariate Normal or t-distribution, shifted and scaled by the Laplace approximation to the mean and variance of $f(u|y; \theta)$ (Jank, 2004b). Importance sampling has been successfully applied to high-dimensional problems but it is generally recommended to monitor the magnitude of the importance weights to ensure numerical stability.

The third alternative for simulation from $f(u|y; \theta)$ is to use Markov Chain Monte Carlo (MCMC). MCMC produces a sequence of draws that depend on one another in the sense that at each stage of the method, the sequence either moves to a new value or remains at the current one. Thus, depending on the mixing-properties of the chain, the resulting MCMC sampler can feature strong long-range dependencies. The MCMC methodology is especially appealing if the target distribution is of non-standard form since it guarantees exact draws from $f(u|y; \theta)$, at least eventually after running it for long enough time. It is therefore conceptually also very appealing for high-dimensional simulation. The drawback of MCMC is that, as pointed out above, it produces dependent samples which makes variance-estimation hard. Moreover, it only produces draws from the target after reaching the stationary distribution, so initial simulation is typically discarded. However, determining exactly how much has to be discarded (i.e. the amount of burn-in) is not easy.

In comparing importance sampling and MCMC, the question comes up as to when should which method be used? Unfortunately there exist hardly any general recommendations or formal comparisons between the two. Quite often, the choice is ultimately influenced by personal preference. Experience also shows that both work well when dealing with high-dimensional integration. While one pro-

duces independent samples it also carries the burden of the importance weights. On the other hand, while the other one has no importance weights to worry about, it can take very long to converge to the stationary distribution and the strong dependence-structure may also make it hard to calculate appropriate standard errors. An overview of importance sampling and MCMC within the context of MCEM can be found in Caffo et al. (2005).

3.1.2 Efficiency of Simulation

The aspect of efficient simulation has, for the most part, only played a secondary role in the literature on EM. While a strong emphasis has been placed on simulating from the *correct* distribution, only little attention has been paid to whether this can also be done in an *efficient* way. By efficient simulation we mean simulation which produces estimates with very little variance. Clearly, the variance of the estimates can be reduced simply by increasing the size of the simulated data. However, this can become computationally intensive and time consuming in more complex problems. The question is whether we can produce samples from the correct (or near-correct) distribution, and also do so with the least possible amount of computation.

Variance reductions techniques attempt to make more efficient use of the simulated data. There exist a variety of variance reduction techniques such as antithetic variables, control variates or stratification. One particular set of methods that has received a lot of interest in the simulation literature is referred to as Quasi-Monte Carlo. Quasi-Monte Carlo is related to Monte Carlo in that it uses simulation to approximate an intractable integral. However, in contrast to classical Monte Carlo, Quasi-Monte Carlo does not use *random* draws. In fact, Quasi-Monte Carlo produces a sequence of *deterministic* numbers with the best-possible spread in the sampling space. This sequence is also referred to as low-discrepancy sequence. There have been many examples where Quasi-Monte Carlo significantly beats classical Monte Carlo methods by a factor of 10, 100 or even 1,000 (Lemieux and L'Ecuyer, 1998).

One drawback of Quasi-Monte Carlo is that it is deterministic in nature and, therefore, statistical methods do not apply for error estimation. Recent advances of *randomized* Quasi-Monte Carlo methodology (L'Ecuyer and Lemieux, 2002) can overcome this drawback. Randomized Quasi-Monte Carlo combines the variance-reduction benefits of Quasi-Monte Carlo with the statistical error-estimation properties of classical Monte Carlo. One way of gen-

erating randomized Quasi-Monte Carlo sequences is to initiate several parallel sequences from randomly chosen starting points (Wang and Hickernell, 2000). While Quasi-Monte Carlo has been, to date, mostly used within the context of importance sampling, there exist efforts to apply its ideas to MCMC (Owen and Tribble, 2005).

Jank (2004b) proposes an automated MCEM algorithm based on randomized Quasi-Monte Carlo methods. The method uses Quasi-Monte Carlo to simulate from $f(u|y; \theta)$ based on Laplace importance sampling. It also uses the ideas of randomized Quasi-Monte Carlo to measure the error of the integral-estimate in every iteration of the algorithm. The resulting Quasi-Monte Carlo EM (QMCEM) algorithm is embedded within the framework of the automated MCEM formulation proposed by Booth and Hobert (1999).

3.1.3 Amount of Simulation

There exist two basic philosophies when it comes to choosing the simulation size for stochastic EM implementations. One philosophy picks a value for the sample size and holds this value fixed throughout all iterations. This philosophy is associated with *stochastic approximation* versions of EM (Delyon et al., 1999) and we will get back to it later. The other philosophy increases the sample size steadily throughout all iterations. A complicating factor with the latter approach is that the sample size has to be determined anew in every iteration if the approach is supposed to result in efficient use of the simulations.

When approximating Q by \tilde{Q} , the Monte Carlo sample size m_t has to be increased successively as the algorithm moves along. In fact, Booth et al. (2001) argue that MCEM will never converge if m_t is held fixed across iterations because of a persevering Monte Carlo error (see also Chan and Ledolter, 1995). While earlier versions of the method choose the Monte Carlo sample sizes in a deterministic fashion before the start of the algorithm (e.g. McCulloch, 1997), the same deterministic allocation of Monte Carlo resources that works well in one problem may result in a very inefficient (or inaccurate) algorithm in another problem. Thus, data-dependent (and user-independent) sample size rules are necessary in order to implement MCEM in an automated way. Automated MCEM implementations have been proposed by several researchers (Booth and Hobert, 1999; Levine and Casella, 2001; Levine and Fan, 2004; Caffo et al., 2005).

Booth and Hobert (1999) are first to propose

an automated implementation of MCEM. Using a Taylor-series argument, they derive approximate confidence bounds for the MCEM parameter estimate under independent sampling schemes. Then, when the next update falls within this confidence bound, it is said to be swamped with Monte Carlo error and consequently a larger sample size is needed to obtain a more accurate estimate of the Q -function. Levine and Casella (2001) and Levine and Fan (2004) build upon Booth and Hobert's method for MCMC sampling. Caffo et al. (2005) propose a new approach based on the difference in the Q -functions. Their approach has several advantages compared to the earlier implementations. First, it operates on a univariate quantity and therefore makes it easier to incorporate more complicated sampling schemes like MCMC or Quasi-Monte Carlo under one umbrella. Second, it assures that EM's famous likelihood-ascent property holds in every iteration, at least with high probability, and thus counterproductive use of the simulation is rare. It also results in more stable variance-covariance estimates since the final-iteration sample size is typically larger than in previous approaches. And finally, by using a one-sided confidence bound approach on the difference in the Q -functions, their method encourages parameter updates with a larger likelihood-increase than the deterministic EM algorithm and thus can result in a defacto acceleration of the method.

One question remains: If we don't want to increase the sample size in every iteration of MCEM, what alternatives do we have? One potential solution is to hold the sample size fixed and simply average over the MCEM output. This makes somewhat sense, because once the method reaches the stationary point, it fluctuates randomly about it with constant noise. Clearly, averaging is a very straightforward approach and it is likely to be very popular with many researchers who do not want to invest much effort into learning an automated implementation. However, what are the dangers in simply averaging over the MCEM output? If we start averaging too early, then our estimate is likely to be biased since we average over early updates which are far from the solution. Bias can also happen if we start averaging too late and, additionally, the estimate will then also be very noisy. But determining when to start averaging (and when to stop) is a problem that is at least equally challenging as finding the right amount of simulation.

3.2 Approximation

After generating draws from the conditional distribution of the missing data, one approximates the Q-functions via the empirical average \tilde{Q} in (3). This is in principle straightforward to do and does not pose any major problems. However, there are modifications of the basic approach that promise several advantages. One of these modifications is to re-use the samples via importance re-weighting; another approach is to employ a stochastic approximation scheme.

Quintana et al. (1999) propose to use some of the simulated data repeatedly throughout the EM iterations by strategically re-weighting them. While this approach seems very reasonable it has not received much wide-spread popularity. One reason for this could be that it is hard to come up with automated sample size rules for a re-weighting scheme.

Another variant of the basic approach is to use a stochastic approximation version of EM (Delyon et al., 1999). That is, rather than steadily increasing the Monte Carlo sample size throughout the algorithm, there exist versions that converge with a constant (and typically small) value of m_t . Let γ_t be a sequence of positive step sizes such that $\sum \gamma_t = \infty$ and $\sum \gamma_t^2 < \infty$ and define

$$\tilde{P}^{(t)}(\theta) = (1 - \gamma_t)\tilde{P}^{(t-1)}(\theta) + \gamma_t\tilde{Q}(\theta|\theta^{(t-1)}). \quad (4)$$

Notice that the recursion in (4) is similar to the stochastic approximation method of Robbins and Monro (1951). For that reason it is referred to as the Stochastic Approximation EM (SAEM) algorithm. One of the noteworthy features of this algorithm is that it converges with constant value of m_t (see Delyon et al., 1999). It is also conceptually very appealing since, by the recursion in (4), it makes use of all the simulated data.

Another appeal of the method is that, at least in principle, the only decision that has to be made is the choice of the step sizes γ_t . This is a one-time decision which is usually done *before* starting the algorithm. Polyak and Juditsky (1992) show that for step sizes $\gamma_t \propto (1/t)^\alpha$, $1/2 < \alpha \leq 1$, the method converges at an optimum rate (if used in conjunction with offline averaging). Thus, at first glance, the method appears to be easier to implement than the Monte Carlo EM algorithm who requires a decision about the new value of m_t *in every iteration*. However, as it is often the case, no lunch is free. Indeed, while large step sizes (i.e. $\alpha \approx 1/2$) quickly bring the method in the neighborhood of the solution, they inflate the Monte Carlo error. On the other hand, while small step sizes (i.e. $\alpha \approx 1$) result

in a fast reduction of the Monte Carlo error, they slow down the rate of convergence of the method.

Jank (2004a) connects the problem of finding the right SAEM step size with EM's missing information principle. It is known that the convergence rate of EM depends on the fraction of missing-to-complete information (Meng, 1994). In particular, if the fraction of missing information is large (and thus EM's convergence rate is already very slow), then it appears unwise to choose small step sizes and thereby even further slowing down SAEM. On the other hand, if EM converges fast then a large step size introduces an unnecessary amount of extra noise which should be avoided. Jank (2004a) estimates EM's rate of convergence from the data and uses this estimate to choose a step size that balances the improvements in bias and variance of SAEM.

3.3 Maximization

The challenges in the M-step are in principle the same as those for the deterministic EM algorithm. If the M-step has no closed form solution (and it typically has not) then one has to resort to numerical methods to maximize the Q-function. The most common approach is to use a version of Newton-Raphson but alternative approaches also exist. The Newton-Raphson procedure has several nice features: It converges at a quadratic rate and it is often straightforward to implement. One drawback of Newton-Raphson (as with many other optimization routines) is that it requires relatively good starting values. Satisfactory starting values can often be found using a coarse grid search.

Another drawback of Newton-Raphson is that it requires evaluation of the Hessian matrix in every iteration. In situations when the Hessian is computationally too involved or numerically unstable, quasi-Newton methods can be used instead. The methods of Davidson Fletcher-Powell (DFP) and Broyden-Fletcher-Goldfarb-Shanno (BFGS) are quasi-Newton procedures that only rely on the gradient of the objective function and are implemented in many software packages.

There exist further modifications of EM that are particularly aimed at simplifying or even accelerating its M-step. See for example the Newton-Raphson type modifications of Jamshidian and Jennrich (1993), Jamshidian and Jennrich (1997) or Lange (1995). Meng and Rubin (1993) on the other hand propose to break-up a complicated M-step into smaller, more tractable conditional M-steps (see also Liu and Rubin, 1994).

3.4 Iteration

In this section we discuss for how many iterations one should run the method. The right amount of iterations is closely connected with the choice of the stopping rule. Finding appropriate stopping rules for stochastic EM versions is challenging. The deterministic EM algorithm is typically terminated if the relative change in two successive parameter estimates is small, smaller than some pre-defined threshold. The same stopping rule is not very useful for its stochastic counterparts. The reason for this is that any deterministic stopping rule can be satisfied by a stochastic method simply because of random chance, and not because convergence has occurred. Recognizing this, Booth and Hobert (1999) recommend to apply a deterministic rule for several successive times, thereby reducing the chances of a premature stop. In the following we review alternative approaches for stopping a stochastic EM algorithm.

Caffo et al. (2005) suggest to terminate MCEM when the difference in likelihood functions becomes small. However, rather than directly estimating the likelihood-differences, they appeal to EM's likelihood-ascent property and instead operate on the differences in the Q-functions. This allows for an efficient implementation at no extra simulation expense. Other approaches are possible. Gu and Zhu (2001) for example propose to monitor the gradient of the likelihood (see also Gu and Li, 1998).

Choosing the right stopping rule can be extra hard for SAEM, especially when an already slowly converging EM algorithm is coupled with a small step size. Standard stopping rules do not take into account the effect of the step size or EM's convergence rate. Jank (2004a) proposes a new way of monitoring SAEM. The approach is based on EM's likelihood-ascent property and measures the long-range improvements in the parameter updates over a flexible time window. It also provides a heuristic to gauge whether there still exists a significant trend in the long-range improvements based on the ideas of permutation tests.

And lastly a comment on the relationship between stopping rules and automated sample size rules: Automated sample size rules generally make it easier to find reasonable stopping rules for MCEM because the resulting algorithm mimics more closely a deterministic algorithm for which reasonable stopping rules are already well-established. On the other hand, consider the "quick-and-dirty" MCEM implementation via averaging of the parameter updates discussed earlier. The resulting averages will generally still show a good amount of variability (espe-

cially if we use a moving-average approach with a fixed time window). Consequently we cannot rely on deterministic stopping rules based on the change in the average parameter updates. For the same reason it is also harder to implement likelihood-based rules. Clearly, while averaging is, at least at first glance, a seemingly convenient approach and easier to implement than automated sample size rules, it does come with a whole additional package of complicating factors.

3.5 Convergence

The EM algorithm converges to the maximum of the likelihood function (Boyles, 1983; Wu, 1983); that is, at least to a local maximum. Stochastic versions of EM typically mimic that behavior under very mild regularity conditions (Delyon et al., 1999; Fort and Moulines, 2003). It is important to point out though that this convergence only occurs if the Monte Carlo sample size is increased successively which again underlines the importance of automated sample size rules. (Increasing m_t is of course only necessary for MCEM; SAEM converges, as discussed earlier, with a fixed m_t .)

While stochastic EM versions typically converge to a *local* optimum, there is no guarantee that this value is also the *global* optimum. The EM algorithm is a greedy method in the sense that it is attracted to the solution closest to its starting value. This can be a problem when several sub-optimal solutions exist. The mixture model, for example, is well-known to feature many local maxima, especially when the number of mixture-components is large. This puts the additional burden on the researcher that any solution found by EM may not be the best solution. One ad-hoc approach to alleviate this problem is to initialize EM from a variety of different starting values, but this approach can be burdensome if the parameter-space is of high dimension.

While the EM algorithm is a local optimization method by nature, there is a growing literature on methods that promise convergence to a global solution. Different global optimization paradigms exist. One very popular approach is the concept of *evolutionary computation*, and the *genetic algorithm* (GA) in particular. Evolutionary computation is associated with the groundbreaking work of Holland (1975). Evolutionary algorithms find their inspiration from natural selection and survival of the fittest in the biological world. These algorithms weed out poor solutions, and combine good solutions with other good solutions to create new generations of even better solutions. Although highly heuristic in

nature, formal proofs of convergence to a global optimum exist (see Holland, 1975).

Jank (2005) proposes a new implementation of the EM algorithm that incorporates the ideas of global optimization. This implementation is based on the Monte Carlo EM algorithm. The Monte Carlo EM algorithm is the natural platform for such an implementation since it is a stochastic method. MCEM is stochastic since it is based on randomly chosen samples and thus two runs from the same starting value may not lead to the same solution. It thus shares the advantages of other stochastic variants of EM. In particular, it has been shown that stochastic implementations of EM have the ability to overcome local, sub-optimal solutions (e.g. Celeux and Diebolt, 1992; Delyon et al., 1999). The basic idea is to inflict random noise into the deterministic updating scheme of EM in the hope that this noise will “push” the method away from a local trap and hence lead to a better solution. MCEM thus has the potential to overcome local traps. However, it is also clear that its ability to overcome these traps is entirely due to chance, since there are no built-in features that systematically steer-free of local solutions. Jank (2005) proposes a new version of MCEM with features borrowed from the genetic algorithm that can overcome local solutions more systematically. (See also Tu et al. (2005) for a related approach.)

4. Conclusion

In this paper we review the basic challenges when implementing stochastic variants of the EM algorithm. We break these challenges into five basic components: simulation, approximation, maximization, iteration and convergence. The simulation step impacts the type, efficiency and amount of simulation. In the approximation step, the Q-function is estimated via an empirical average but modifications of this approach also exist. The maximization step maximizes (and sometimes also simplifies maximization of) the estimated Q-function. The iteration step decides how long the algorithm is to be run and what stopping rule to use. Finally, while convergence to a local solution is typically guaranteed, more work is necessary in order find the global solution using the EM paradigm.

References

Booth, J. G. and Hobert, J. P. “Maximizing Generalized Linear Mixed Model Likelihoods with an

Automated Monte Carlo EM Algorithm.” *Journal of the Royal Statistical Society B*, 61:265–285 (1999).

Booth, J. G., Hobert, J. P., and Jank, W. S. “A survey of Monte Carlo algorithms for maximizing the likelihood of a two-stage hierarchical model.” *Statistical Modelling*, 1:333–349 (2001).

Boyles, R. A. “On the convergence of the EM algorithm.” *Journal of the Royal Statistical Society B*, 45:47–50 (1983).

Caffo, B. S., Booth, J. G., and Davison, A. C. “Empirical Sup Rejection Sampling.” *Biometrika*, 89:745–754 (2002).

Caffo, B. S., Jank, W. S., and Jones, G. L. “Ascent-Based Monte Carlo EM.” *Journal of the Royal Statistical Society, Series B*, 67:235–252 (2005).

Celeux, G. and Diebolt, J. “A stochastic approximation type EM algorithm for the mixture problem.” *Stochastics and Stochastics Reports*, 41:127–146 (1992).

Chan, K. S. and Ledolter, J. “Monte Carlo EM Estimation for Time Series Models Involving Counts.” *Journal of the American Statistical Association*, 90:242–252 (1995).

Delyon, B., Lavielle, M., and Moulines, E. “Convergence of a Stochastic Approximation version of the EM Algorithm.” *The Annals of Statistics*, 27:94–128 (1999).

Dempster, A. P., Laird, N. M., and Rubin, D. B. “Maximum Likelihood From Incomplete Data Via the EM Algorithm.” *Journal of the Royal Statistical Society B*, 39:1–22 (1977).

Fort, G. and Moulines, E. “Convergence of the Monte Carlo expectation maximization for curved exponential families.” *The Annals of Statistics*, 31:1220–1259 (2003).

Gu, M. G. and Li, S. “A Stochastic Approximation Algorithm for Maximum Likelihood Estimation with Incomplete Data.” *Canadian Journal of Statistics*, 26:567–582 (1998).

Gu, M. G. and Zhu, H.-T. “Maximum likelihood estimation for spatial models by Markov chain Monte Carlo stochastic approximation.” *Journal of the Royal Statistical Society B*, 63:339–355 (2001).

- Holland, J. H. *Adaptation in natural and artificial systems*. The University of Michigan Press, Ann Arbor, MI (1975).
- Jamshidian, M. and Jennrich, R. I. “Conjugate Gradient Acceleration of the EM Algorithm.” *Journal of the American Statistical Association*, 88:221–228 (1993).
- . “Acceleration of the EM algorithm by using Quasi-Newton methods.” *Journal of the Royal Statistical Society B*, 59:569–587 (1997).
- Jank, W. and Booth, J. G. “Efficiency of Monte Carlo EM and Simulated Maximum Likelihood in two-stage hierarchical models.” *Journal of Computational and Graphical Statistics*, in print (2002).
- Jank, W. S. “Implementing and Diagnosing the Stochastic Approximation EM Algorithm.” Technical report, University of Maryland (2004a).
- . “Quasi-Monte Carlo Sampling to Improve the Efficiency of Monte Carlo EM.” *Computational Statistics and Data Analysis*, 48:685–701 (2004b).
- . “Ascent EM for Fast and Global Model-Based Clustering: An Application to Curve-Clustering of Online Auctions.” Technical report, University of Maryland (2005).
- Lange, K. “A Gradient Algorithm Locally Equivalent to the EM Algorithm.” *Journal of the Royal Statistical Society B*, 57:425–437 (1995).
- L’Ecuyer, P. and Lemieux, C. “Recent Advances in Randomized Quasi-Monte Carlo Methods.” In Dror, M., L’Ecuyer, P., and Szidarovszki, F. (eds.), *Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications*, 419–474. Kluwer Academic Publishers (2002).
- Lemieux, C. and L’Ecuyer, P. “Efficiency Improvement by Lattice Rules for Pricing Asian Options.” In *Proceedings of the 1998 Winter Simulation Conference*, 579–586. IEEE Press (1998).
- Levine, R. and Casella, G. “Implementations of the Monte Carlo EM algorithm.” *Journal of Computational and Graphical Statistics*, 10:422–439 (2001).
- Levine, R. and Fan, J. “An Automated (Markov Chain) Monte Carlo EM Algorithm.” *Journal of Statistical Computation and Simulation*, 74:349–359 (2004).
- Liu, C. and Rubin, D. B. “The ECME Algorithm: A Simple Extension of EM and ECM With Faster Monotone Convergence.” *Biometrika*, 81:633–648 (1994).
- McCulloch, C. E. “Maximum Likelihood Algorithms for Generalized Linear Mixed Models.” *Journal of the American Statistical Association*, 92:162–170 (1997).
- Meng, X.-L. “On the Rate of Convergence of the ECM Algorithm.” *The Annals of Statistics*, 22:326–339 (1994).
- Meng, X.-L. and Rubin, D. B. “Maximum Likelihood Estimation Via the ECM Algorithm: A General Framework.” *Biometrika*, 80:267–278 (1993).
- Owen, A. and Tribble, S. “A quasi-Monte Carlo Metropolis algorithm.” *Proceedings of the National Academy of Sciences (to appear)* (2005).
- Polyak, B. T. and Juditsky, A. B. “Acceleration of stochastic approximation by averaging.” *SIAM Journal of Control and Optimization*, 30:838–855 (1992).
- Quintana, F., Liu, J., and delPino, G. “Monte Carlo EM with Importance Reweighting and its Applications in Random Effects Models.” *Computational Statistics and Data Analysis*, 29:429–444 (1999).
- Robbins, H. and Monro, S. “A Stochastic Approximation Method.” *The Annals of Mathematical Statistics*, 22:400–407 (1951).
- Tu, Y., Ball, M., and Jank, W. S. “Estimating Flight Departure Delay Distributions A Statistical Approach with Long-Term Trend and Short-Term Pattern.” Technical report, University of Maryland (2005).
- Wang, X. and Hickernell, F. J. “Randomized Halton Sequences.” *Mathematical and Computer Modelling*, 32:887–899 (2000).
- Wei, G. C. G. and Tanner, M. A. “A Monte Carlo Implementation of the EM Algorithm and the Poor Man’s Data Augmentation Algorithms.” *Journal of the American Statistical Association*, 85:699–704 (1990).
- Wu, C. F. J. “On the Convergence Properties of the EM Algorithm.” *The Annals of Statistics*, 11:95–103 (1983).