working paper 12 19

A Tractable Estimator for General Mixed Multinomial Logit Models

Jonathan James

FEDERAL RESERVE BANK OF CLEVELAND

Working Paper 12-19 September 2012

A Tractable Estimator for General Mixed Multinomial Logit Models

Jonathan James

The mixed logit is a framework for incorporating unobserved heterogeneity in discrete choice models in a general way. These models are difficult to estimate because they result in a complicated incomplete data likelihood. This paper proposes a new approach for estimating mixed logit models. The estimator is easily implemented as iteratively re-weighted least squares: the well known solution for complete data likelihood logits. The main benefit of this approach is that it requires drastically fewer evaluations of the simulated likelihood function, making it signicantly faster than conventional methods that rely on numerically approximating the gradient. The method is rooted in a generalized expectation and maximization (GEM) algorithm, so it is asymptotically consistent, efficient, and globally convergent.

Keywords: Discrete choice, mixed logit, expectation and maximization algorithm.

JEL codes: C01, C13, C25, C61.

The author thanks Kenneth Train, Peter Arcidiacono, Fabian Lange, and participants at the Cleveland Fed applied microeconomics workshop for helpful comments. Jonathan James is at the Federal Reserve Bank of Cleveland, and he can be reached at jwj8@duke.edu. Updated versions of this paper will also be posted at http://econ.duke.edu/~jwj8.

1 Introduction

Accommodating unobserved heterogeneity is critical for studying discrete response models and more importantly in using these models to construct counterfactuals. The mixed logit model is a highly flexible discrete choice framework that incorporates unobserved heterogeneity in a general way by allowing the unobserved error structure of the utility maximization problem to be arbitrarily correlated across choices and time. In theory, the mixed logit model is extremely powerful as McFadden and Train (2000) show that with a flexible enough error structure, it can approximate any random utility model with arbitrary degree of accuracy.[1]

However, unlike the standard logit framework, which has a well known simple iterative solution, the computational challenges of estimating mixed logit models are substantial, making it difficult for researchers to fully realize the potential of this flexible discrete choice framework. First, these models are estimated by maximizing an integrated likelihood function, which requires numerical simulation to evaluate the integral. This entails costly computations on a simulated data set, much larger than the original dataset. Second, the complicated likelihood function is difficult to maximize, so researchers typically rely on optimizers that compute a numerical approximation to the gradient of the likelihood. Approximating the gradient is an extremely costly activity, requiring an enormous amount of likelihood function evaluations at each iteration, typically equal to the number of parameters in the model or more.

In order to feasibly estimate these models using numerical gradient based methods, researchers search for a balance between the number of parameters in the model and the number of simulation draws used to approximate the integral. If too few simulation draws are used, Lee (1992) shows that the estimates are biased. With many simulation draws, the simulated dataset may not be storable in read/write memory, generating a tremendous amount

[1] McFadden and Train (2000) likewise show that this result is true for the probit framework. This paper will focus on the mixed logit model because it is by far the most widely used discrete choice framework. However, the methodological contribution of this paper can be applied to the mixed probit model as well.

of overhead to repeatedly read and write this dataset to the hard disk. Unfortunately the balance between model specification and the number of simulation draws is often dictated by computational tractability and not the economic model or asymptotic consistency.

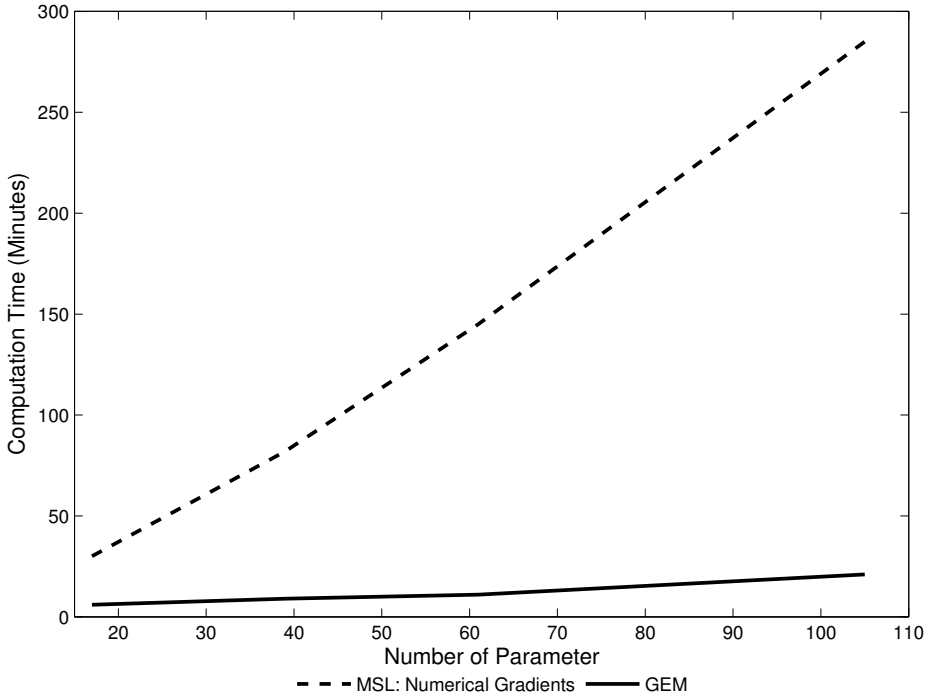
This paper outlines a new estimator for the general class of mixed multinomial logit models that yields a solution that is as simple as the procedure for estimating standard logit models. In contrast to other methods, where each iteration requires multiple simulated likelihood evaluations (increasing in the number of parameters), the premier benefit of the estimator is that each iteration of the algorithm is equivalent to only *one* evaluation of the simulated likelihood function. This makes the estimator computationally tractable for models with both many parameters and many simulation draws.

The estimator is rooted in the Expectation and Maximization (EM) algorithm, a widely used method for computing maximum likelihood estimates in the presence of unobserved data. Conventional numerical optimizers, like gradient descent or Newton-Raphson, form a sequence of linear (or quadratic) local approximations to the objective function and take steps along the these approximations towards the solution. These local approximations require the computation of the gradient of the objective function, which generates the computational challenges discussed above. In contrast, the EM algorithm at each step forms a lower bound to the objective function and iteratively maximizes this sequence of lower bounds, converging to a local maximum of the objective function.[2]

The advantages of the EM approach are that it does not require the computation of the gradient, the lower bound objective function is easy to calculate, and in many cases this lower bound is easy to maximize. Unfortunately, one such case where the lower bound is difficult to maximize is latent variable models like logit and probit. This has inhibited the widespread use of the EM algorithm for these models because while the EM algorithm avoids the computation of the gradient, it requires a costly procedure to optimize

[2]See Lange et al. (2000) and Minka (1998) for a description of the lower bound derivation of the EM algorithm and Dempster et al. (1977) for the original proof.

Figure 1: Monte Carlo Results



the lower bound function at each iteration.

The estimator takes a very simple approach to this problem. It forms an alternative lower bound to the one offered by the conventional EM algorithm, whose maximum has a simple closed form solution. Importantly, the existence of this alternative lower bound is based off of the nature of discrete choice models, not the form of the unobserved heterogeneity. This implies that this simple optimization routine can be applied to any discrete choice framework and allows for a wide degree of complexity of unobserved variables. Formally, the algorithm constitutes a Generalized Expectation and Maximization (GEM) algorithm described in Dempster et al. (1977). Since it is part of the EM family, it is not only easy to compute, but it is consistent, efficient, and globally convergent.

To gain insight into the performance of the estimator, a number of Monte Carlo experiments are conducted. Figure 1 previews the results from one of

these experiments, which shows the effect on the computation time as the number of exogenous covariates are increased in the model. In this simple example, including additional exogenous covariates had little effect on the number of iterations it took MSL to to converge, but because each iteration required more likelihood evaluations to approximate the gradient, doubling the number of parameters effectively doubled the computation time. In contrast, because the GEM algorithm only requires one likelihood evaluation for each iteration, it experienced only a mild increase in computation time, such that the GEM estimator with more than 100 parameters was faster than the MSL estimator with 17 parameters.

An extension of the estimator to dynamic discrete choice models is in James (2012). One straightforward application is to quasi-structural models (see Bernal and Keane (2010)). In these dynamic models, observations on discrete choices are observed in conjunction with observations on a continuous outcome variable that contains information on the unobserved mixing component (e.g. wages or output). In these cases, Arcidiacono and Miller (2010) argue that consistent estimates of the model parameters, with exception to the structural choice parameters, can be found by replacing the dynamic choice probabilities with a reduced form policy function, resulting in the estimation of a static mixed logit model. The key component is that these reduced form policy functions must represent the dynamic problem well, so they likely contain many parameters. A common approach is to use McFadden (1975) universal logit (including all variables in all choice functions). This approach is infeasible with current methods because it contains too many parameters, however it can be tractably estimated with the GEM algorithm.

A primary benefit of the proposed method for estimating these complicated models is that it is implemented in a manner that is nearly identical to the very simple solution to the standard logit model. To put the simplicity of the estimator in context, it is beneficial to first review the rudimentary procedure for estimating standard logit models. This is done for the binary logit model in section 2. Section 3 briefly describes the mixed logit framework in the context of the binary choice model and discusses the current

approaches to estimating these models and their computational challenges. The estimator is derived in section 4, first for the binary mixed logit case and then extended to the multinomial mixed logit model. Section 5 discusses the results from the Monte Carlo experiments. And section 6 concludes.

2 Review of Standard Logit

In the binary logistic model with panel data, we assume the discrete outcome $d_{it} \in \{0, 1\}$ occurs when $d_{it} = \underset{j \in \{0,1\}}{\operatorname{argmax}} U_{it}(j)$ and U is a latent variable represented by,

$$\begin{aligned} U_{it}(0) &= \varepsilon_{it}(0) \\ U_{it}(1) &= X_{it}\alpha_1 + \varepsilon_{it}(1) \end{aligned}$$

Where X_{it} are observables and α_1 are parameters to be estimated. The unobserved random utility shock ε is assumed distributed i.i.d. type-I extreme value.

The logistic model is by far the most widely used approach for studying discrete response data because the choice probabilities for the observed outcomes yield a simple closed form expression. In the binary model they are defined as.

$$\begin{aligned} P_{it}(1|\alpha_1) &= \Pr(d_{it} = 1|X_{it}, \alpha_1) = \frac{\exp(X_{it}\alpha_1)}{1 + \exp(X_{it}\alpha_1)} \\ P_{it}(0|\alpha_1) &= \Pr(d_{it} = 0|X_{it}, \alpha_1) = 1 - P_{it}(1|\alpha_1) \end{aligned}$$

With these choice probabilities, the parameters α_1 are estimated by maximizing the log-likelihood function, summing over individuals and time periods,

$$\hat{\alpha}_1 = \underset{\alpha_1}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{t=1}^{T} \Big[X_{it}\alpha_1 d_{it} - \ln\big(1 + \exp(X_{it}\alpha_1)\big) \Big] \tag{1}$$

The solution to eq. (1) does not have a closed form expression so an opti-

mization routine is required. Given that the log-likelihood function is a large sum, the first and second derivatives are easy to calculate. Therefore, the most common and computationally efficient method for estimating standard logit models is Newton-Raphson. With starting values α_1^0 Newton-Raphson entails iteratively computing,

$$\alpha_1^{m+1} = \alpha_1^m - \left[\sum_{i=1}^{N} \mathbf{H}_i^m\right]^{-1} \left[\sum_{i=1}^{N} \mathbf{g}_i^m\right] \tag{2}$$

Where $\mathbf{g}_i^m$ and $\mathbf{H}_i^m$ are individual i's contribution to the gradient and the hessian of the log-likelihood function conditional on the m^{th} iteration parameter estimates. They are given by,

$$\begin{aligned} \mathbf{g}_i^m &= \sum_{t=1}^{T} X_{it}' \Big[d_{it} - P_{it}(1|\alpha_1^m) \Big] \\ \mathbf{H}_i^m &= \sum_{t=1}^{T} \Big[P_{it}(1|\alpha_1^m) \times (1 - P_{it}(1|\alpha_1^m)) \Big] X_{it}' X_{it} \end{aligned} \tag{3}$$

The Newton-Raphson procedure for the standard logit model is typically referred to as Iteratively Re-weighted Least Squares (IRLS), since it bears many similarities to the least squares solution, but includes the weights, $P_{it}(1|\alpha_1^m)$, which are re-computed at each iteration. Solving these models is straightforward since the components of the Newton-Raphson algorithm can be easily calculated.[3]

3 Binary Mixed Logit Model

The mixed logit model shares a similar structure to the standard logit model except that, in addition to the logit error, there is another element which is unobserved that needs to be integrated out of the likelihood function in

[3] Newton-Raphson continually iterates on eq. (2) until $\mathbf{g}_i^m$ gets close to zero. In general, convergence of Newton-Raphson is not guaranteed since it is not a stable algorithm. In practice the steps are often scaled to assure convergence to a local maximum and is stopped if their is little change in the parameters or objective function value.

estimation.[4] The nature of this additional unobservable is quite general, which allows mixed logit models to represent a broad class of discrete choice utility maximization problems. The estimator applies to a general mixed multinomial logit model, however, to emphasize the insights and implementation of the estimator, it is instructive to outline it in a more stylized model. We will modify the individual's latent utility by including an additional unobserved component z_i which is a vector of unobserved error terms, $z_i = \{z_i(1), \ldots, z_i(T)\}$, such that,

$$\begin{aligned} U_{it}(0) &= \varepsilon_{it}(0) \\ U_{it}(1) &= X_{it}\alpha_1 + z_i(t) + \varepsilon_{it}(1) \end{aligned}$$

In this specification, z represents the "mixing" component of the unobserved error, which is arbitrarily correlated across time, and we maintain the assumption that the remaining component of the error, ε, is i.i.d. over time. We will assume that z is distributed multivariate normal $z_i \sim \mathcal{N}(\gamma, \Delta)$ with associated probability density function $f(z|\gamma, \Delta)$. Once the main insights of the estimator are derived in this setting, a discussion of other distributional assumptions is presented in appendix A.[5]

Given observed outcomes $D_i = \{d_{i1}, \ldots, d_{iT}\}$ and $X_i = \{X_{i1}, \ldots, X_{iT}\}$ for individual $i = 1, \ldots, N$, the probability of the data conditional on z by the product of logits.

$$\Pr(D_i|X_i, z, \alpha_1) = \prod_{t=1}^{T} \left[\frac{1}{1+\exp(X_{it}\alpha_1 + z(t))}\right]^{d_{it}=0} \left[\frac{\exp(X_{it}\alpha_1 + z(t))}{1+\exp(X_{it}\alpha_1 + z(t))}\right]^{d_{it}=1}$$

To estimate this random utility model with maximum likelihood we form the incomplete data likelihood by integrating over the unobserved component, z, using the density $f(z|\gamma, \Delta)$, summing over the individual log-

[4] See Train (2009) ch 6 for a discussion on mixed logit models.

[5] Alternatively, the mixing component could be represented by a random coefficient model, such that $z_i(t) = Z_{it}\beta_i$, where Z_{it} are observed regressors and β_i are unobserved individual specific random coefficients with distribution $f(\beta|\Gamma)$.

likelihood functions. The maximum likelihood estimator solves,

$$\hat{\Psi}^{MLE} = \underset{\Psi \in \{\alpha_1, \gamma, \Delta\}}{\operatorname{argmax}} \sum_{i=1}^{N} \ln \left(\int_z \Pr(D_i | X_i, z, \alpha_1) f(z | \gamma, \Delta) \, dz \right) \tag{4}$$

Where Ψ represents the collective set of parameters.

The integration does not have a closed form solution, so it must be evaluated using numerical simulation. This constitutes a maximum simulated likelihood (MSL) estimator, which is the most widely used approach for estimating these models. Assuming straight forward Monte Carlo integration, for each person draw R values of z randomly from $\mathcal{N}(\gamma, \Delta)$, labeled z_{ir}, then solve,

$$\hat{\Psi}^{MSL} = \underset{\Psi \in \{\alpha_1, \gamma, \Delta\}}{\operatorname{argmax}} \sum_{i=1}^{N} \ln \left(\frac{1}{R} \sum_{r=1}^{R} \Pr(D_i | X_i, z_{ir}, \alpha_1) \right) \tag{5}$$
$$\text{where } z_{ir} \sim \mathcal{N}(\gamma, \Delta)$$

The mixed logit objective function is much more complicated than the standard logit objective function in eq. (1). The primary issue is that integration is located inside of the log operator, which prevents the log operator from linearizing the products in $\Pr(D_i|\cdot)$. Because of this, analytical expressions for the first and second derivative are very difficult to derive, so in practice estimation of these models often relies on Quasi-Newton methods that form an approximation to the gradient and the hessian. These approximations are calculated by preforming many evaluations of the likelihood function at different points in the state space, typically requiring one likelihood evaluation for each parameter in the model to approximate the gradient in all directions.

Unfortunately, the gradient approximation is necessary at each iteration of the algorithm, resulting in an enormous number of likelihood evaluations. Evaluating the likelihood in eq. (5) may consist of a number of simple computations, however, even these simple computations can become sluggish when the data is multiplied by a factor of R. If the original dataset is

large to begin with, one may quickly approach memory capacity, requiring the extremely expensive task of reading and writing data to the hard disk or looping over individuals and recomputing the simulated values at each evaluation of the likelihood. Approximating the gradient requires so many evaluations of the simulated likelihood function that the costs of simulation rapidly compound. As the number of parameters increases, so does the number of likelihood evaluations required to approximate the gradient, which quickly becomes intractable with more than a few parameters.

4 The Estimator

This section describes the estimator for the binary mixed logit model and then extends it to the multinomial choice framework. The estimator is rooted in a Generalized Expectation and Maximization (GEM) algorithm, so we begin by describing the EM algorithm in the mixed logit framework and then move to the specifics of the GEM estimator.

4.1 Review of the EM Algorithm

The Expectation and Maximization (EM) algorithm formalized in Dempster et al. (1977) is a popular method for computing maximum likelihood estimates in the presence of unobserved data.[6] The EM algorithm entails repeatedly maximizing, in many cases, a simpler augmented data likelihood, which as shown below bears a close resemblance to the complete data likelihood.

Instead of directly maximizing this very complicated likelihood function in eq. (4), each step of the EM algorithm forms a lower bound to this function and maximizes this simpler function.[7] This lower bound has two special properties. First it is simple to compute, and second it touches the likelihood function at any arbitrary value, Ψ^m. This implies that the parameters that maximize the lower bound function necessarily increase the

[6] For a review of the EM algorithm see Train (2009) and McLachlan and Krishnan (1997).

[7] This derivation of the EM algorithm borrows from that outlined in Minka (1998).

likelihood function. Using this as the next point in forming the lower bound and repeating this process, the EM algorithm converges to a local maximum of the likelihood.

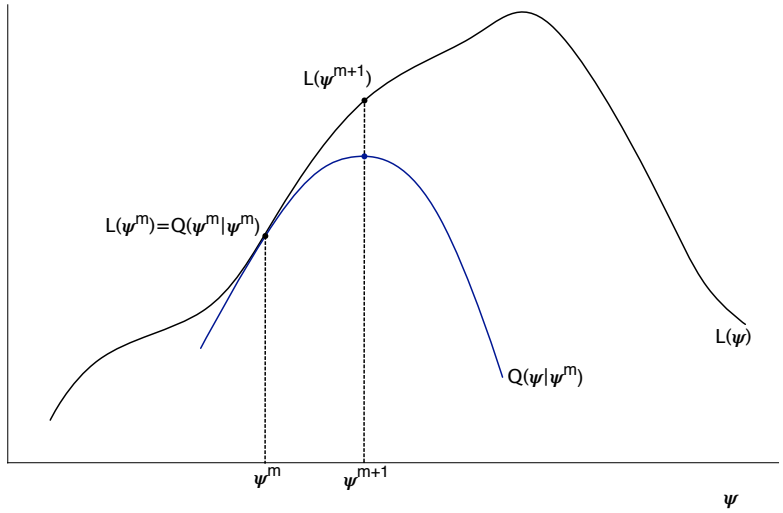
Let $h_i^*(z)$ indicate an arbitrary density function (i.e. $\int_z h_i^*(z)dz = 1$ and $h_i^*(z) \geq 0$). The EM algorithm is based on the following principle.

$$\ln\left(\int_z \frac{\Pr(D_i|X_i, z, \alpha_1) f(z|\gamma, \Delta) h_i^*(z)}{h_i^*(z)}\, dz\right) \geq$$
$$\ln\left(\prod_z \left(\frac{\Pr(D_i|X_i, z, \alpha_1) f(z|\gamma, \Delta)}{h_i^*(z)}\right)^{h_i^*(z)}\right) \tag{6}$$

Where $\prod_z$ represents the continuous product, analogous to the integral over an infinite support. Equation (6) follows from Jensen's inequality which states that the arithmetic mean (the inside of the log on the left hand-side) is never smaller than the geometric mean (the inside of the log on the right-hand side). We will re-write this statement more compactly as $L(\Psi) \geq Q(\Psi)$.

Equation (6) holds for any density function $h_i^*(z)$. However, the EM algorithm chooses a very particular density function. Namely, for any arbitrary value of parameters, Ψ^m, $h_i(z|\Psi^m)$ is chosen so that eq. (6) holds with equality at Ψ^m (i.e. $L(\Psi^m) = Q(\Psi^m)$, the lower bound touches the likelihood at Ψ^m). Using $h_i(z|\Psi^m)$, the lower bound function can be expressed as $Q(\Psi|\Psi^m)$. Given these conditions, any value of Ψ such that $Q(\Psi|\Psi^m) \geq Q(\Psi^m|\Psi^m)$ implies that $L(\Psi) \geq L(\Psi^m)$, with equality at $\Psi = \Psi^m$. The E-step of the algorithm is to form this lower bound function, and the M-Step is to find new parameters $\Psi^{m+1} = \underset{\Psi}{\operatorname{argmax}}\, Q(\Psi|\Psi^m)$. Because the bound improves at each iteration, the sequence of parameter estimates are guaranteed to converge to a local maximum of $L(\Psi)$. This bounding approach is sketched in figure 2.

Figure 2: Sketch of EM Algorithm



Conditional on Ψ^m, the density function is defined as,

$$\begin{aligned} h_i(z|D_i, X_i, \Psi^m) &= \Pr(z = z_i|D_i, X_i, \Psi^m) \\ &= \frac{\Pr(D_i|X_i, z, \alpha_1^m) f(z|\gamma^m, \Delta^m)}{\int_{z'} \Pr(D_i|X_i, z', \alpha_1^m) f(z'|\gamma^m, \Delta^m)\, dz'} \end{aligned}$$

This function is described as the density of the individual's unobserved information, z, conditional on the individual's observed data and a given value of the parameters. This density does not have an analytical form, so for each individual is is approximated by R points labeled z_{ir} with associated weights w_{ir}^m, where m highlights that the weights are a function of parameters Ψ^m. Because these densities are approximated, the EM applied to the mixed logit framework is more formally a Simulated EM (SEM) algorithm, which preserves all of the properties of the EM algorithm with the identical condition to MSL that the number of simulation draws rise at a rate faster than the sample size.[8]

To compute the weights, for each individual $i = 1, \ldots, N$, draw R values of z randomly from $f(z|\gamma^m, \Delta^m)$ labeled z_{ir} and then compute the choice

[8] McLachlan and Krishnan (1997) provide a survey of Simulated EM algorithms.

probability,

$$P_{irt}^m(1) = \frac{\exp(X_{it}\alpha_1^m + z_{ir}(t))}{1 + \exp(X_{it}\alpha_1^m + z_{ir}(t))}$$

The weights are then calculated using Bayes' Rule.

$$w_{ir}^m = \frac{\prod_{t=1}^T (1 - P_{irt}^m(1))^{(d_{it}=0)} P_{irt}^m(1)^{(d_{it}=1)}}{\sum_{r'=1}^R \left(\prod_{t=1}^T (1 - P_{ir't}^m(1))^{(d_{it}=0)} P_{ir't}^m(1)^{(d_{it}=1)} \right)} \tag{7}$$

Using these approximations to $h_i(z|\Psi^m)$, the lower bound of the objective function (or augmented data likelihood) for the EM step is,

$$\begin{aligned} Q(\Psi|\Psi^m) &= \sum_{i=1}^N \ln \left(\prod_{r=1}^R \left(\frac{\Pr(D_i|X_i, z_{ir}, \alpha_1) f(z_{ir}|\gamma, \Delta)}{w_{ir}^m} \right)^{w_{ir}^m} \right. \\ &= \sum_{i=1}^N \sum_{r=1}^R w_{ir}^m \ln \Big[\Pr(D_i|X_i, z_{ir}, \alpha_1) f(z_{ir}|\gamma, \Delta) \Big] \\ &\quad - \sum_{i=1}^N \sum_{r=1}^R w_{ir}^m \ln \Big[w_{ir}^m \Big] \end{aligned} \tag{8}$$

As described, the main computational challenge of the mixed logit is the presence of the very large integral inside of the log function in eq. (4). Jensen's inequality allows us to replace this integral with a sequence of products, which the log operator transforms into the sum of a sequence of logs. This linearizes the objective function in a very desirable way, making this function easier to maximize.

Equation (8) is a simpler function to maximize for two reasons. First, the parameters α_1 are independently maximized from γ and Δ because the log operator makes these values additively separable in the objective function. This feature of the EM algorithm is discussed in Arcidiacono and Jones (2003). Therefore maximizing the objective function involves maximizing

two simpler functions,

$$\alpha_1^{m+1} = \underset{\alpha_1}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{r=1}^{R} w_{ir}^m \ln \Big[\Pr(D_i | X_i, z_{ir}, \alpha_1) \Big] \tag{9}$$

$$\gamma^{m+1}, \Delta^{m+1} = \underset{\gamma,\Delta}{\operatorname{argmax}} \sum_{i=1}^{N} \sum_{r=1}^{R} w_{ir}^m \ln \Big[f(z_{ir} | \gamma, \Delta) \Big] \tag{10}$$

Second, eq. (9)-(10) represent expected complete data likelihoods where the values z_{ir} and weights w_{ir}^m are treated as data. Often the maximum likelihood solution when all of the data is observed has a closed form expression. For example, because of the normality assumption on z, the maximum likelihood estimates of the mean, γ, and covariance, Δ, given observations on z, are simply the mean and covariance of the sample. Similarly, eq. (10) is equivalent to the likelihood of weighted observations on z and has a simple closed form solution.[9]

$$\gamma^{m+1} = \frac{1}{N} \sum_{i=1}^{N} \sum_{r=1}^{R} w_{ir}^m z_{ir}$$

$$\Delta^{m+1} = \frac{1}{N} \sum_{i=1}^{N} \sum_{r=1}^{R} w_{ir}^m z_{ir} z_{ir}' - \gamma^{m+1} \gamma^{m+1'}$$

The fact that the EM algorithm typically produces analytical expressions for the distributional parameters in mixed logit models is well established (see Train (2008)).[10] The drawback of the EM algorithm for estimating these

[9] The closed form solution is not driven by the normality assumption, but by the fact that the maximum likelihood estimates in the full information case are related to the sample moments in the data. This is true for other distributions as well, for example the exponential distribution. Appendix A discusses estimation of these parameters when a closed form solution is not apparent.

[10] Using this result, Train (2007) proposes an EM algorithm for estimating random coefficient models that contain only unobserved distributional parameters and no fixed parameters in the logit (i.e. no α terms). Estimation in this case requires iteratively maximizing an equation similar to eq (10). When this maximization has a closed form, Train (2007) shows computational gains of the EM algorithm over MSL with numerical gradients on the order of 25 times.

models is that no closed form exists for maximizing the objective function in eq. (9) for the parameters α_1. This objective function is conceptually simple to maximize as it represents a weighted standard logit model, which, as discussed in section 2, can be solved with an iterative Newton-Raphson algorithm. However, performing this expensive inner iterative routine within the outer routine of the EM algorithm substantially reduces the attractiveness of this approach for estimating mixed logit models.

4.2 The GEM Algorithm

The main contribution of this paper is to derive a simple method for implementing the EM algorithm that circumvents this inner routine. The method is based off of an important result in Bohning and Lindsay (1988) which says that for a twice differentiable, concave function $l(\theta)$, if there exists a symmetric, negative definite matrix B such that $B \leq \bigtriangledown^2 l(\theta_0)$ for all values of θ_0, then for any value θ, the following is true.

$$l(\theta) \geq l(\theta_0) + (\theta - \theta_0)' \bigtriangledown l(\theta_0) + (\theta - \theta_0)' B (\theta - \theta_0)/2 \tag{11}$$

Similar to the bounding approach of the EM algorithm, the results in Bohning and Lindsay (1988) show that finding θ_1 that maximizes the right hand side of eq. (11) implies that $l(\theta_1) \geq l(\theta_0)$ with equality only at $\theta_1 = \theta_0$. Repeating this process, the parameters converge monotonically to a local maximum of the function $l(\theta)$

This result relates to the EM algorithm for the mixed logit model in an important way. Turning to the EM objective function for the logit parameters in eq. (9) and incorporating the expression for $\Pr(D_i|\cdot)$, gives,

$$Q(\alpha_1|\Psi^m) = \sum_{i=1}^{N} \sum_{r=1}^{R} w_{ir}^m \sum_{t=1}^{T} \left[\Big(X_{it}\alpha_1 + z_{ir}(t) \Big) d_{it} - \ln \left(1 + \exp(X_{it}\alpha_1 + z_{ir}(t)) \right) \right] \tag{12}$$

This objective function is functionally equivalent to the standard logit in eq. (1) except with the addition of the weights, w_{ir}^m.

Since eq. (12) is difficult to maximize, the estimator will instead use the results of Bohning and Lindsay (1988) and bound the complicated function with a simpler function. Letting $\widetilde{Q}(\alpha_1|\Psi^m) \leq Q(\alpha_1|\Psi^m)$ denote the bound such that $\widetilde{Q}(\alpha_1^m|\Psi^m) = Q(\alpha_1^m|\Psi^m) = L(\Psi^m)$.[11] Then by construction, choosing $\alpha_1^{m+1} = \underset{\alpha_1}{\operatorname{argmax}}\, \widetilde{Q}(\alpha_1^m|\Psi^m)$ implies that $Q(\alpha_1^{m+1}|\Psi^m) \geq Q(\alpha_1^m|\Psi^m)$ and more importantly $L(\alpha_1^{m+1}, \gamma^m, \Delta^m) \geq L(\alpha^m, \gamma^m, \Delta^m)$. This approach maintains the necessary condition of the EM procedure, that the objective function is maximized at each iteration. This constitutes a generalized EM algorithm defined in Dempster et al. (1977) since the EM objective is merely improved at each iteration rather than maximized.[12] We can illustrate this idea by adding this new bound to the augmented data likelihood in figure 3.

Forming this second bound requires that the hessian of eq. (12) is bounded from below. Bohning and Lindsay (1988) indeed show that such a lower bound exists for discrete choice models. The hessian of eq. (12) for individual i is similar to the formula in the standard logit model in eq. (3) with the exception of the additional weights.

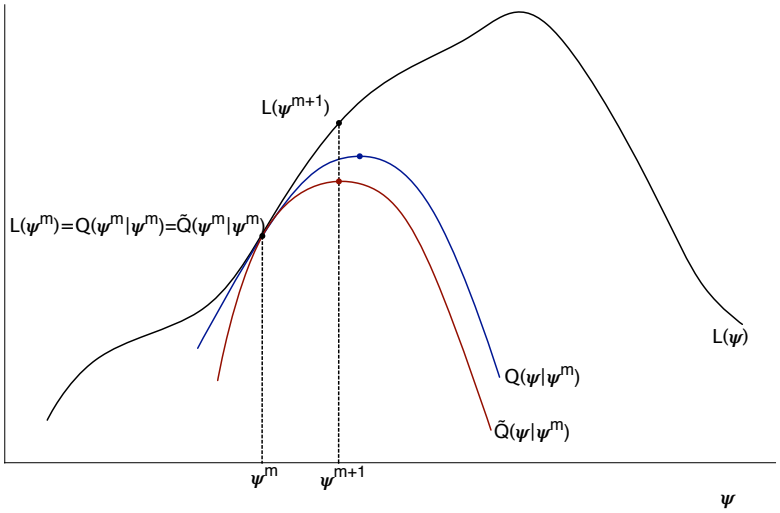
$$\mathbf{H}_i^m = \sum_{r=1}^{R} w_{ir}^m \sum_{t=1}^{T} \Big[P_{irt}^m(1) \times (1 - P_{irt}^m(1)) \Big] X_{it}' X_{it}$$

Since the probabilities $P_{irt}^m(1) \in (0, 1)$, then $P_{irt}^m(1) \times (1 - P_{irt}^m(1)) \geq -(1/4)$ for all values of X_{it}, α_1, and z_{ir}. Additionally, the weights w_{ir}^m represent a density, which must sum to one. Therefore it is easy to show that the

[11] These equalities are not completely correct as eq. (12) has dropped the constant and other parts of the function in eq. (8). More formally the necessary condition required to increase the likelihood is that $Q(\Psi|\Psi^m) - L(\Psi)$ is smallest at $\Psi = \Psi^m$, which still holds. Lange et al. (2000).

[12] Another possibility is to follow the approach of Rai and Matthews (1993), which proposes conducting a single cycle of a scaled Newton-Raphson step on the objective function in eq. (12) to update the parameters, adjusting the scale parameter to guarantee that the likelihood increases. The scaling is necessary to maintain monotonicity since Newton-Raphson is an unstable algorithm and a single cycle does not guarantee that the augmented likelihood function will be improved as pointed out in Meng and Rubin (1993). This approach is likely inferior to the one outlined here because it requires the computation of the hessian matrix and the optimal scale factor. The proposed methods requires neither.

Figure 3: Sketch of GEM Algorithm



hessian is bounded from below by, $\mathbf{B}_i$ defined as,

$$\mathbf{H}_i^m \geq -(1/4)\sum_{t=1}^{T} X_{it}' X_{it} = \mathbf{B}_i \tag{13}$$

We can define the lower bound $\widetilde{Q}(\alpha_1|\Psi^m)$ as,

$$\widetilde{Q}(\alpha_1|\Psi^m) = Q(\alpha_1^m|\Psi^m) + (\alpha_1 - \alpha_1^m)' \left[\sum_{i=1}^{N} \mathbf{g}_i^m\right] + (\alpha_1 - \alpha_1^m)' \left[\sum_{i=1}^{N} \mathbf{B}_i^m\right] (\alpha_1 - \alpha_1^m)/2 \tag{14}$$

Where $\mathbf{g}_i^m$ is individual i's contribution to the gradient of eq. (12) defined as

$$\mathbf{g}_i^m = \sum_{r=1}^{R} w_{ir}^m \sum_{t=1}^{T} X_{it}' \left(d_{it} - P_{irt}^m(1)\right)) \tag{15}$$

The maximum of equation (14) gives a simple closed form expression by

taking the derivative with respect to α_1 and setting it equal to zero.

$$\begin{aligned}\alpha_1^{m+1} =& \underset{\alpha_1}{\operatorname{argmax}}\, \widetilde{Q}(\alpha_1|\Psi^m) \\ =& \alpha_1^m - \left[\sum_{i=1}^{N} \mathbf{B}_i^m\right]^{-1} \left[\sum_{i=1}^{N} \mathbf{g}_i^m\right] \end{aligned} \tag{16}$$

So far the description of the estimator has largely focused on it's convergence properties. Since it is based on the EM algorithm it is globally convergent, consistent, and efficient. There has been little discussion regarding whether it is easy to implement.

In simple terms, each iteration of the GEM algorithm requires the computation of the weights, w_{ir}^m in eq. (7), the gradients $\mathbf{g}_i^m$ in eq. (15), and the parameter updates α_1^{m+1} in eq. (16). This procedure is very similar to the Iteratively Re-weighted Least Squares procedure for the standard logit outlined in eq. (2). In the standard logit model, the least squares weights, the choice probabilities, are re-computed at each iteration. Likewise these choice probabilities are also used in eq. (15). The main difference between these two methods is that the mixed logit also requires the computation of the weights, w_{ir}^m. Conveniently, these weights are functions of the choice probabilities, which are already used in the calculation of the gradient. Therefore, the GEM algorithm requires no more additional computations than the standard logit model, except that these computations are done on a data set that is R times the original data.

In fact, in some ways, implementing the estimator is simpler than Newton-Raphson because it does not require the computation and inverse of the hessian at each iteration. By construction, the lower bound of the hessian $\mathbf{B}_i$ described in eq. (13) is independent of both the parameters and the weights, so it can be computed and inverted outside of the algorithm all together.

Finally, how does one iteration of the GEM algorithm compare to one iteration of MSL? Importantly, each iteration of the GEM algorithm is approximately equal to only one evaluation of the simulated likelihood function. This is seen by looking at the denominator of eq. (7), which is identical

to value inside of the log of eq. (5). The source of intractability of these models was that conventional methods required at each iteration an extensive number of simulated likelihood evaluations, either to approximate the gradient or as an iterative routine inside of the traditional EM algorithm. The estimator completely circumvents this issue.

4.3 GEM for Multinomial Mixed Logit

Extending the GEM algorithm from the binary case to the multinomial logit is straightforward. This requires computing additional choice probabilities and redefining the lower bound of the EM objective function. Modify the choice set to $J+1$ and set $j=0$ as the normalizing choice. We will redefine z_i to be a $J \times T$ dimensional vector of unobserved values which are arbitrarily correlated across choices and time, maintaining the notation that $f(z|\gamma, \Delta)$. Utility for $j \in \{1, \ldots, J\}$ is described as, $U_{it}(j) = X_{it}\alpha_j + z(j,t) + \varepsilon_{it}(j)$.

The lower bound of the hessian for the multinominal logit equivalent to eq. (13) is,[13]

$$\mathbf{B}_i = -(1/2)(eye(J) - ones(J)/(J+1)) \otimes \left(\sum_{t=1}^{T} X_{it}' X_{it} \right) \tag{17}$$

Where $eye(\cdot)$ is the identity matrix and $ones(\cdot)$ is a matrix of ones.

The algorithm entails: First computing, $B^{-1} = \left[\sum_{i=1}^{N} \mathbf{B}_i \right]^{-1}$ from eq. (17). Then given current iteration estimates, α^m, γ^m, and Δ^m,

1. Draw R values of z randomly from $f(z|\gamma^m, \Delta^m)$ and label them z_{ir} for each individual $i = 1, \ldots, N$.
2. Compute choice probabilities $P_{irt}^m(0:J)$, for each draw r

$$P_{irt}^m(0) = \frac{1}{1 + \sum_{j'=1}^{J} \exp(X_{it}\alpha_{j'}^m + z_{ir}(j',t))}$$

[13] The proof is in Bohning (1992). This expression is correct when all characteristics enter all choice equations and must be modified when this is not the case.

$$P_{irt}^m(j) = \frac{\exp(X_{it}\alpha_j^m + z_{ir}(j,t))}{1 + \sum_{j'=1}^J \exp(X_{it}\alpha_{j'}^m + z_{ir}(j',t))}$$

3. Compute

$$w_{ir}^m = \frac{\prod_{t=1}^T \prod_{j=0}^J P_{irt}^m(j)^{(d_{it}=j)}}{\sum_{r'=1}^R \left(\prod_{t=1}^T \prod_{j=0}^J P_{ir't}^m(j)^{(d_{it}=j)}\right)}$$

4. Compute gradient contribution:[14]

$$\mathbf{g}_i^m = \sum_{r=1}^R w_{ir}^m \sum_{t=1}^T X_{it}' \left(y_{it} - P_{irt}^m(1:J)\right)$$

5. Update parameters,

$$\alpha^{m+1} = \alpha^m - B^{-1}\left[\sum_{i=1}^N \mathbf{g}_i^m\right]$$

6. Update $\gamma^{m+1} = \frac{1}{N}\sum_{i=1}^N \sum_{r=1}^R w_{ir}^m z_{ir}$

7. Update $\Delta^{m+1} = \frac{1}{N}\sum_{i=1}^N \sum_{r=1}^R w_{ir}^m z_{ir} z_{ir}' - \gamma^{m+1}\gamma^{m+1'}$

When the parameters reach some converge criteria, i.e. $||\Psi^{m+1} - \Psi^m||_\infty < \epsilon_1$ or $||(\Psi^{m+1} - \Psi^m)/\Psi^m||_\infty < \epsilon_1$, then the algorithm stops.[15] Standard errors for EM algorithms are discussed in Ruud (1991) and outlined in appendix B.

[14] This calculation of the gradient produces a matrix, which needs to be vectorized in the modified newton step. y_{it} is a J dimensional vector with $y_{it}(d_{it}) = 1$ and zero everywhere else. $P_{irt}^m(1:J)$ represents a $1 \times J$ vector of choice probabilities. Notice $P_{irt}^m(0)$ is excluded.

[15] If the random draws are fixed, then the GEM algorithm will converge deterministically, and one can use traditional convergence criteria like those stated above (see Nielsen (2000)). Other methods for simulated EM algorithms calculate new random draws at each iteration, and the sequence of parameter estimates form a Markov chain converging to a stable distribution. Further discussion on the differences in these approaches is in Nielsen (2000). Because detecting convergence for the algorithm is more straightforward with fixed random draws, this is the approach focused on in this paper.

5 Monte Carlo Example

This section explores the computational efficiency of the GEM algorithm through a number of Monte Carlo experiments. The first experiment holds the model complexity (the number of choices and the number of unobservables) fixed and increases the number of exogenous covariates in the utility function. The second experiment allows the model complexity to increase with the number of parameters and shows the effect on the computation time as the number *unobserved* parameters increases from one to five. In both experiments, the GEM algorithm results in substantial computational savings over MSL with numerical gradients, but more importantly, the savings are exponentially increasing with more complex models.

Data for the Monte Carlo experiments are created from the same underlying model but differ in the way the number of parameters are increased. In each period, $t = 1, \ldots, T$ individuals make a discrete choice from $j \in \{0, 1, \cdots, J\}$. Choice specific utility is defined by a vector of exogenous observables X_{it} and an observed endogenous vector H_{it}, defined as the history of previous choices. i.e.

$$H_{it} = \left[\sum_{t'=1}^{t-1} I \cdot (d_{it'} = 1), \cdots, \sum_{t'=1}^{t-1} I \cdot (d_{it'} = J) \right]$$

For $t > 0$. Where d_{it} denotes the choice in period t, and $H_{i1} = 0$.

In addition to these factors, choices are also driven by an unobserved taste vector z_i, where $z_i(j)$ is individual i's persistent unobserved taste for choice j. Choice specific utility is defined as,

$$U_{ijt} = X_{it}\alpha_{1j} + H_{it}\alpha_{2j} + z_i(j) + \varepsilon_{ijt} \tag{18}$$

H is described as endogenous because it is correlated with unobserved taste z. Failing to account for z will bias all of the parameter estimates, in particular, α_{2j}.

5.1 MC1: Increasing the Number of Observed Covariates

In the first set of experiments the number of choices is set to $J = 2$, which fixes the size of the unobserved components, z, and the endogenous vector H. More parameters are introduced into the model by increasing the dimension of X_{it}, the exogenous regressors. The number of variables are increased from 4 to 48 implying a range of total number of parameters from 17 to 105. Data is generated and estimated for 100 different samples with $N = 1,000$ and $T = 10$. Convergence for the GEM is set to a tolerance of $||\Psi^{m+1} - \Psi^m||_\infty < 1e-6$ and discussed further in appendix C.

First, table 1 compares a selection of the parameter estimates between MSL with numerical gradients and the GEM algorithm for the smallest model (17 parameters total). These include the estimates for the distributional parameters, γ and Δ, and those for the endogenous covariates, α_{2j}. Both MSL and the GEM algorithm do very well in estimating the parameters, so the focus of this section will be on the computational differences.

Table 2 compares the key statistics which describe the computational complexity for each estimator. These include the number of iterations, the number of simulated likelihood evaluations, and the total computation time. First, MSL requires many fewer iterations than the GEM algorithm. This is not surprising given that MSL uses a quasi-Newtown method with superlinear rates of convergence, while EM algorithms exhibit only linear rates of convergence. However, the main feature of the GEM algorithm is that one iteration requires only one simulated likelihood evaluation. This advantage made the GEM algorithm significantly faster than MSL in all cases and increasingly so as the parameters increased. These computation times are plotted in figure 1 in the introduction.

5.2 MC2: Increasing the Model Complexity

The results in MC1 are informative because by holding fixed the choice set, it abstracts away from other factors influencing the computation time to isolate the effect of including more parameters. The experiments here depart from this approach to study the performance of the estimator as the model

Table 1: Comparison of Estimates (selected parameters) [a]

	True Values	MSL	GEM
$\gamma(1)$	0.5000	0.5159	0.5183
	(0.0000)	(0.0902)	(0.0916)
$\gamma(2)$	0.5000	0.5124	0.5144
	(0.0000)	(0.1014)	(0.1025)
$\Delta(1,1)$	2.0000	2.0145	2.0003
	(0.0000)	(0.2349)	(0.2231)
$\Delta(2,1)$	1.1547	1.1678	1.1710
	(0.0000)	(0.1628)	(0.1629)
$\Delta(2,2)$	2.0000	1.9993	1.9949
	(0.0000)	(0.1797)	(0.1817)
$\alpha_{21}(1)$	0.0000	-0.0034	-0.0023
	(0.0000)	(0.0240)	(0.0236)
$\alpha_{21}(2)$	0.0000	-0.0003	-0.0020
	(0.0000)	(0.0337)	(0.0340)
$\alpha_{22}(1)$	0.0000	-0.0038	-0.0048
	(0.0000)	(0.0304)	(0.0310)
$\alpha_{22}(2)$	0.0000	0.0006	0.0010
	(0.0000)	(0.0233)	(0.0233)

[a] Numbers averaged over 100 simulations with $N = 1,000$ and $T = 10$. Integration for both methods uses 2,000 random draws. All models use the same starting values, those generated by the standard logit

Table 2: Computational Comparison: MC1 [a]

Number of Parameter	MSL			GEM	
	No. Iters	No. Fun. Evals.	Comp. Time (min.)[b]	No. Iters/Fun. Evals.	Comp. Time (min.)
17	63	1253	30	157	6
39	79	3492	82	237	9
61	90	6166	145	311	11
83	97	9067	215	430	16
105	103	12008	285	570	21

[a] Numbers averaged over 100 simulations with $N = 1,000$ and $T = 3$. Integration for both methods uses 2,000 random draws. All models use the same starting values, those generated by the standard logit

[b] The replications where conducted on a shared computing cluster. To benchmark the computation time, each experiment was run once on an isolated processor to calculate the average time per function evaluation and then multiplied by the average number of function evaluations for the 100 replications.

complexity increases simultaneously with the number of parameters. This is accomplished by changing the size of the choice set. As the choice set increases, the likelihood becomes more difficult to calculate because more choice probabilities need to be computed. In addition, increasing J increases the dimension of both the unobserved taste vector z and endogenous observables H. Collectively for $J = 1, 2, \ldots, 5$, the number of parameters range from 5 to 95.[16]

Increasing the number of parameters as well as the model complexity substantially increases the computation time of both estimators. These results are shown in table 3. For MSL, not only are the number of simulated likelihood evaluations increasing with the number of parameters but so are the number of iterations, causing the computation time to explode. Figure 4 plots the computation time between the two estimators. While the computation time increases with the GEM algorithm, the effects are much more mild compared to MSL.

[16] The dimension of X is set to $2 \times J$.

Table 3: Computational Comparison: MC2 [a]

Number of Parameter	MSL			GEM	
	No. Iters	No. Fun. Evals.	Comp. Time (min.)[b]	No. Iters/Fun. Evals.	Comp. Time (min.)
5	25	172	2	56	1
17	63	1253	30	157	6
36	91	3796	122	220	12
62	119	8597	355	591	41
95	138	15188	765	1591	138

[a] Numbers averaged over 100 simulations with $N = 1,000$ and $T = 10$. Integration for both methods uses 2,000 random draws. All models use the same starting values, those generated by the standard logit

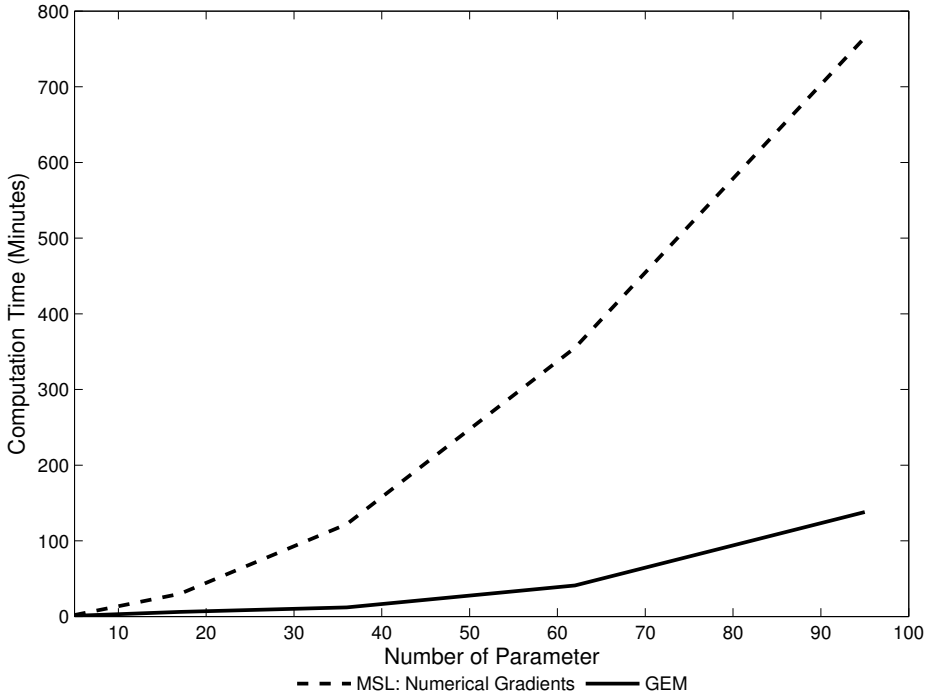
[b] The replications where conducted on a shared computing cluster. To benchmark the computation time, each experiment was run once on an isolated processor to calculate the average time per function evaluation and then multiplied by the average number of function evaluations for the 100 replications.

These Monte Carlo experiments showcase the substantial computational gains that can be realized by the GEM algorithm. Most importantly, these gains are strongest where current methods become increasingly intractable, potentially making accessible entire classes of models which have been avoided for computational reasons.

6 Conclusion

This paper outlines a simple routine for estimating complex mixed logit models. The estimator is based off of a Generalized Expectation and Maximization algorithm. The primary benefit of the estimator is that it requires drastically fewer evaluations of the likelihood function than conventional methods, making the estimator computationally tractable for models with complicated error structures and many parameters. This paper applies the estimator to static discrete choice models. These insights are extended to dynamic discrete choice models in James (2012).

Figure 4: Computation Time MC2



References

P. Arcidiacono and J. B. Jones. Finite mixture distributions, sequential likelihood and the em algorithm. *Econometrica*, 71(3):933–946, May 2003.

P. Arcidiacono and R. A. Miller. Ccp estimation of dynamic discrete choice models. Duke University, July 2010.

R. Bernal and M. Keane. Quasi-structural estimation of a model of childcare choices and child cognitive ability production. *Journal of Econometrics*, 156:164–189, 2010.

D. Bohning. Multinomial logistic regression algorithm. *Annals of the Institute of Statistical Mathematics*, 44(1):197–200, 1992.

D. Bohning and B. G. Lindsay. Montonicity of quadratic-approximation algorithms. *Annals of the Institute of Statistical Mathematics*, 40(4):641–663, 1988.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

J. James. A gem estimator for dynamic discrete choice models. Federal Reserve Bank of Cleveland, 2012.

M. Kuroda and M. Sakakihara. Accelerating the convergence of the em algorithm using the vector ε algorithm. *Computational Statistics & Data Analysis*, 51(3):1549–1561, 2006.

K. Lange, D. R. Hunter, and I. Yang. Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, 9 (1):1–20, Mar. 2000.

L.-F. Lee. On efficiency of methods of simulated moments and maximum simulated likelihood estimation of discrete reponse models. *Econometric Theory*, 8(4):518–552, Dec. 1992.

D. McFadden. On independence, structure, and simultaneity in transportation demand analysis. working paper no 7511, Urban Travel Demand Forcasting Project, Institute of Transportation and Traffic Engineering, University of California, Berkeley, 1975.

D. McFadden and K. Train. Mixed mnl models of discrete response. *Journal of Applied Econometrics*, 15:447–470, 2000.

G. J. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. Wiley, New York, 1997.

X.-L. Meng and D. B. Rubin. Maximum likelihood estimation via the ecm algorithm: A general framework. *Biometrika*, 80(2):267–278, Jun. 1993.

T. P. Minka. Expectation-maximization as lower-bound maximization. http://research.microsoft.com/en-us/um/people/minka/papers/, Novermber 1998.

S. F. Nielsen. On the simulated em algorithm. *Journal of Econometrics*, 96: 267–292, 2000.

S. N. Rai and D. E. Matthews. Improving the em algorithm. *Biometrics*, 49(2):587–591, Jun. 1993.

P. A. Ruud. Extensions of estimation methods using the em algorithm. *Journal of Econometrics*, 49:305–341, 1991.

K. Train. A recursive estimator for random coefficient models. *Working Paper*, 2007.

K. Train. Em algorithms for nonparametric estimation of mixing distributions. *Journal of Choice Modelling*, 1(1):40–69, 2008.

K. Train. *Discrete Choice Methods with Simulation*. Cambridge University Press, 2009.

A Solutions with Non-Normal Distribution

The derivation of the GEM algorithm assumed that the mixing distribution was multivariate normal. This lead to a simple closed form expression for the solution to the objective function. While normality may be an appropriate assumption in many applications, there may be instances where other distributions are desired, or a closed form solution is not apparent.

First, a number of important distributions are related to normal distributions and as shown by Train (2007) can be seamlessly incorporated into the estimator. He discuses a number of distributions, for example log-normal and censored normal. To accommodate the log-normal the only modification of the algorithm is in step two where the choice probabilities are instead calculated as,

$$P_{irt}^{m}(j) = \frac{\exp(X_{it}\alpha_{j}^{m} + \ln(z_{ir}(j,t)))}{\sum_{j'=0}^{J}\exp(X_{it}\alpha_{j'}^{m} + \ln(z_{ir}(j,t)))}$$

And z_{ir} is still drawn from the normal distribution, but transformed in the utility function.

Second, similar to the normal, any distribution which has a closed form in the complete data case will also have a closed form solution in the weighted EM objective function, for example the exponential distribution.

Finally, if the maximization over the distributional parameters does not admit a closed form or a closed form is not apparent, then the objective function must be numerically optimized. In some cases, a numerical optimizer can be implemented in a very efficient manner, producing a negligible increase in computation time. In particular, the weights of the augmented data likelihood form an expectation. If this expectation can be passed through the likelihood, then the dimension of the problem can be reduced from $N \times R$ back to N. Numerically optimizing this much smaller dataset is much faster and in addition, since we use the previous iterations parameter estimates as starting values, we begin very close to the solution, where most numerical optimizers have super-linear rates of convergence. Outlining this procedure in the case of the multivariate normal, the augmented likelihood function

can be modified in the following way.

$$
\begin{aligned}
&\underset{\Gamma,\Delta}{\operatorname{argmax}} \sum_{i=1}^{N}\sum_{r=1}^{R} w_{ir}^{m} \ln\Big[f(z_{ir}|\Gamma,\Delta)\Big] \\
&= \underset{\Gamma,\Delta}{\operatorname{argmax}} \sum_{i=1}^{N}\sum_{r=1}^{R} w_{ir}^{m}\Big[-\frac{1}{2}\ln(|\Delta|) - \frac{1}{2}(z_{ir}-\Gamma)'\Delta^{-1}(z_{ir}-\Gamma)\Big] \\
&= \underset{\Gamma,\Delta}{\operatorname{argmax}} -\frac{N}{2}\ln(|\Delta|) - \sum_{i=1}^{N}\frac{1}{2}\Big[\operatorname{tr}\left((\sum_{r=1}^{R} w_{ir}z_{ir}z_{ir}')\Delta^{-1}\right) \\
&\qquad\qquad - 2\operatorname{tr}\left((\sum_{r=1}^{R} w_{ir}z_{ir})\Gamma\Delta^{-1}\right) + \operatorname{tr}(\Gamma\Gamma'\Delta^{-1})\Big]
\end{aligned}
$$

Which takes advantage of the exchangeability of expectations in the trace operator. The elements within the trace can be computed outside of the maximization, reducing the data from $N \times R$ back to N, avoiding maximizing over the entire simulated data set.

B Standard Errors

Train (2007) describes the computation of standard errors in simulated EM algorithms using the information matrix, which is constructed from the simulated scores at the solution Ψ^*. With K parameters let S_i be a $1 \times K$ vector of simulated scores for individual i, such that $S_i = [S_i(\alpha^*)\ S_i(\gamma^*)\ S_i(\Delta^*)]$. Then the covariance of the parameter estimates is described by $V = (S'S)^{-1}$ where S is an $N \times K$ matrix whose i^{th} row contains S_i.

$S_i(\alpha)$ is already computed in step 4 of the algorithm such that $S_i(\alpha) = (g_i^{m^*})'$. Finally,

$$S_i(\gamma) = \sum_{r=1}^{R} w_{ir}^{m^*}(\Delta^*)^{-1}(z_{ir}-\gamma^*)$$

$$S_i(\Delta) = -.5\sum_{r=1}^{R} w_{ir}^{m^*}\left((\Delta^*)^{-1} - (\Delta^*)^{-1}(z_{ir}-\gamma^*)(z_{ir}-\gamma^*)'(\Delta^*)^{-1}\right)$$

C A Simple Accelerator

The GEM algorithm produces a sequence of estimates $\Psi^1, \Psi^2, \ldots, \Psi^m$ which converge to a value Ψ^*. The algorithm is typically terminated when the

difference or percentage change between successive parameter estimates is less than some tolerance. Kuroda and Sakakihara (2006) propose a novel convergence criteria for the EM algorithm which exploits a vector epsilon algorithm. Given a linearly converging vector sequence, the vector epsilon algorithm generates an alternative sequence that converges to the same stationary value. Using the sequence of EM parameter estimates, Ψ^m, the vector epsilon sequence, ψ^m is defined as,

$$\psi^{m+1} = \Psi^m + \left[\left[\Psi^{m-1} - \Psi^m\right]^{-1} + \left[\Psi^{m+1} - \Psi^m\right]^{-1}\right]^{-1} \tag{19}$$

Where $x^{-1} = \dfrac{x}{||x||^2}$ with the denominator representing the dot product of vector x. Kuroda and Sakakihara (2006) show that this sequence $\psi^1, \psi^2, \ldots, \psi^m$ converges to the same stationary value as the EM sequence, Ψ^*, but uses the magnitude and direction of the EM sequence to extrapolate to the convergence point, requiring significantly fewer iterations. Implementing this requires no modification to the EM code, and it is important to highlight that the sequence ψ^m never enters any of the parts of the algorithm. It is a sequence which is computed in tandem with the algorithm and provides a stopping criteria in the same manner one would use the EM sequence to form a stopping criteria, i.e. once $||\psi^{m+1} - \psi^m||_\infty < \epsilon_1$, then $\psi^{m+1} \approx \Psi^*$.[17]

[17]Kuroda and Sakakihara (2006) originally describe their estimator as an EM accelerator. However, since it requires no modification to the EM code it is perhaps better characterized as a stopping criteria. A number of accelerators have been proposed for the EM algorithm. McLachlan and Krishnan (1997) provides a good introduction to some of these methods. Even though the GEM algorithm overcomes the main source of computational tractability for the mixed logit model, it is worthwhile to consider incorporating these accelerators to improve the speed of these methods even further. Caution should be used with some accelerators as computational speed is often achieved at the expense numerical stability or significant additional coding. In addition, all EM accelerators do not apply to accelerating simulated EM algorithms.