A Smooth Simulated Moment Estimator for Discrete Choice Panel Data with Arbitrary Error Structure

Daniel A. Ackerberg*

February 24, 1999

Abstract

Estimation of richly specified panel data discrete choice models has always been plagued by the inability to analytically compute choice probabilities. As basic Method of Simulated Moments (MSM) estimators have not been successfully applied to the long panel case, recent research has focused on reducing simulation error enough so that alternative methods that are not as "immune" to simulation error are usable. In particular, a number of authors have shown that use of the extremely powerful GHK probability simulator can reduce biases to acceptable levels in panel probit models. This paper argues that while the GHK simulator is a very useful tool for panel probit models, it is not anywhere near as effective for models that have decision rules that are non-linear in unobservables (resulting from, e.g. dynamic optimization models) or for models that have non-normal unobservables (e.g. resulting from less parametric unobservable specifications). These types of models are increasingly more beneficial to study given increases in the size and length of data and in the ability to solve realistic dynamic choice problems. We suggest an estimator for these models - or, more generally, for models where one cannot easily reduce simulation error. This estimator builds off McFadden's (1989) original MSM estimator - the key insight is to simulated unmanageable sums by drawing random elements of these sums. Naive simulated sum (SUM) estimators have unacceptable levels of simulation error, but by correlating simulation of sums and probabilities, we obtain an estimator that is both easy to use and very effective. In two Monte-Carlo experiments where the GHK simulator cannot be used effectively, we show that our SUM estimator dominates Simulated Maximum Likelihood (with the bias correction procedure of Lee (1995)) and the Transition Probability estimators of Berkovec and Stern (1992) and Keane (1994).

* Asst. Professor of Economics, Boston University, Boston, MA 02215 (ackerber@bu.edu).

1. Introduction

Until the 1980's, the discrete choice literature relied on particular functional forms for unobservables to generate likelihoods that were computable. The simulation literature, led by Lerman and Manski (1981), changed this by developing econometric methods in which exact choice probabilities need not be computed. These potentially complicated multivariate integrals are instead simulated. As plugging such simulated probabilities into a standard likelihood function results in inconsistent parameter estimates for a fixed number of simulation draws, one approach in this literature has been to move from likelihood based estimation into Method of Simulated Moments (MSM) frameworks (McFadden (1989) and Pakes and Pollard (1989)). Because moments have the potential to be linear in the simulation error, one can often obtain consistent estimates of θ as the number of observations increases for any *fixed* number of simulation draws. In addition, if one uses optimal instruments, MSM estimators can be asymptotically (in *both* observations and simulation draws) as efficient as Maximum Likelihood.

Unfortunately, such techniques have not been easily applied to discrete choice panel data - in particular panel data with unobservables that are correlated over time (e.g. unobserved individual heterogeneity, serial correlation, or learning). One approach, due to McFadden (1989), is to treat each possible *sequence* of choices (through time) as a choice itself. This essentially transforms the problem into a rather large multinomial choice problem - one where the number of possible (sequence) choices is equal to the number of choices in each period (J) to the Tth power. However, since J^T increases so quickly in T, actually computing the simulated moment soon becomes infeasible (in general), as one must 1) simulate probabilities of, 2) compute instruments for, and 3) sum, J^T elements. McFadden notes that computation *is* feasible in one special case - when one uses a pure frequency simulator for the probabilities. Since one draw of a pure frequency simulator places 0 probability on all but one choice sequence, the majority of the elements in the simulated moment are exactly zero and need not be computed.

Unfortunately, there are two problems with this pure-frequency simulator. The first is that the resulting moment is not continuous in the parameter vector θ. This can be very problematic in numeric optimization, particularly with lots of parameters to estimate. Second, weights need to be computed for every choice sequence that ever is "drawn" by the pure frequency simulator. This can be very time consuming, particularly because computing these weights effectively is important in attaining good efficiency relative to the Maximum Likelihood estimator. As a result of these problems, McFadden's pure-frequency approach has seen very limited (if any) use in practice.

A second approach, proposed by Keane (1990,1994) and Berkovec and Stern (1992), is to write moments in transition probabilities, i.e. the probability of the period t decision conditional on decisions prior to t. This reduces the number of probabilities that need to be computed (or simulated) from J^T to $J \cdot T$, clearly a much more feasible number. The major issue here is that one needs to simulate the period t decision *conditional* on past choices. In fact, unbiased simulators of these transition probabilities are generally computationally infeasible (for medium to large T), so Keane and Berkovec and Stern suggest more feasible, but slightly biased (for a fixed number of

simulation draws) simulators. Even so, Monte-Carlo results in Keane (1994) and Geweke, Keane and Runkle (1996) show that in multinomial, multiperiod probit models, use of the highly accurate GHK recursive simulator prevents significant biases in estimates even for small numbers (e.g. 10 or 20) of simulation draws.

On the other hand, and one of the main motivations for the estimator developed in this paper, is a point we make that for more general error structures, the GHK simulator can not always be applied effectively or easily. Examples include non-normal or less parametric heterogeneity, random coefficient models, learning models, or other dynamic optimization models. We show Monte-Carlo evidence that in these types of models - particularly in long panels, the transition probability MSM estimator can exhibit very significant biases, even for much larger numbers of simulation draws.

This paper proposes a third MSM estimator to deal with these models, i.e. for discrete choice panel data with arbitrary error structure. This estimator builds off McFadden's estimator in that we examine probabilities of choice sequences rather than transition probabilities. We call it a "simulated sum" estimator because our estimator avoids the unwieldy J^T element sums in McFadden's formulation by simulating them, i.e. by drawing N random elements of those sums. As such, unlike McFadden we are able to use smooth sequence probability simulators - as a result our moment condition is continuous in the parameter vector. Also unlike McFadden, the estimator requires computing only one small (size N) set of optimal weights and this only needs to be done once. In contrast to the feasible transition probability (TP) estimator of Keane and Berkovec and Stern, our simulated sum (SUM) estimator is consistent for any *fixed* number of simulation draws.

Moving to efficiency, we show that because the SUM estimator needs to simulate two components of the moment condition (both the choice sequence probabilities entering the sum as well as the sum itself), "naive" simulated sum estimators are impractical due to high levels of simulation error. The crucial step to obtaining a good SUM estimator is purposely *correlating* the simulation error in simulating the sum with the simulation error in computing the sequence probabilities. This turns out to be extremely easy in practice and results in an estimator whose extra variance due to simulation is asymptotically only $\frac{1}{N}$ times the variance of the exact estimator. We note that for our SUM estimator this $\frac{1}{N}$ term is in some sense a lower bound on the simulation error. This contrasts to the TP estimator, for which $\frac{1}{N}$ is essentially an upper bound on the simulation error. As such, our estimator sacrifices some efficiency. However, we exhibit Monte-Carlo results that suggest 1) this efficiency loss is not large, even in models where the TP estimator can utilize the highly accurate GHK simulator (we replicate the AR(1) plus normal random effect models of Keane (1994)), and 2) we avoid the potential biases of the TP estimator, which we show can be quite large when GHK cannot be used effectively. In some simple long panel random coefficient and dynamic optimization models, our SUM estimator shows no signs of bias and clearly dominates the TP estimator in terms of MSE. We also find that in these types of models our estimator compares very favorably to Simulated Maximum Likelihood (SML) alternatives, even when using the bias-correcting methods of Lee (1995). Importantly, our estimator is also very easy to program and use. It is essentially at the same level of difficulty as SML - only instead of simulating the probability

of one choice sequence per observation, one needs to simulate probabilities of N choice sequences.

Recent years have seen increases in both the extensiveness (e.g. size, length) of datasets and in the ability to solve complex decision models. These developments have increased the benefits to studying discrete-choice panel data models both rich in dynamics (e.g. dynamic optimization) and in heterogeneity (e.g. more heterogeneity, less restrictive parametric assumptions on heterogeneity). The caveat is that for these types of models, it will be harder and harder to easily or efficiently reduce simulation error[1]. We feel that the evidence presented here suggests our simulated sum estimator may be the estimator of choice for these types of models. It is essentially the only smooth estimator that is consistent for a fixed number of simulation draws[2], and thus the only estimator that is robust to the considerable amounts of simulation error that will typically arise in these models.

This paper is organized as follows. First, we set up the general panel discrete choice model to be studied. We then review three commonly used simulators for choice probabilities. We focus in particular on the powerful GHK simulator and the conditions under which it can (or cannot) significantly reduce simulation error. We then review prior approaches to estimation with discrete choice panel data before introducing our SUM estimator. We conclude with our Monte-Carlo study comparing the various estimators. The appendix contains internet links to the programs used for the Monte Carlo study.

2. Setup

We consider a generic discrete choice model where consumer i chooses one of J alternatives j in each period t. Consider the latent unobserved "utilities" from each alternative:

$$u_{ijt} = f_j(X_{it}, y_i^{t-1}, \epsilon_{it}; \theta)$$

and observable discrete choices y_{it}:

$$y_{it} = j \qquad \text{if } j = \arg\max_j [u_{ijt}] \tag{2.1}$$

where X_{it} is a set of observed exogenous variables and ϵ_{it} is a scalar *or* vector of econometric unobservables. As our superscripts denote past histories of a variable, we allow current util-

[1] At least in theory, one can always use importance sampling distributions to reduce simulation error to acceptable levels. The question is whether one wants to have to search out appropriate importance sampling distributions for every problem, specification, or even parameter value. The GHK simulator solves this problem for the panel probit model, as it is a very potent simulator regardless of the particular specification. Our argument is that for the more general models above, there does not exist (or does not exist yet) such a powerful and general tool. In addition, this "search out a good simulator" approach would need to define what is an "acceptable level" of simulation error for each problem. We don't feel that this is an easy task.

[2] Keane (1994) briefly suggests, but does not evaluate, an alternative to his TP estimator that *is* consistent for a finite number of simulation draws. Unfortunately, this estimator loses some of the nice "efficiency properties" of the TP estimator. We also evaluate this estimator and find strong evidence that it is, in fact, less efficient than our SUM estimator.

ity (and choice) to depend on all past choices y_i^{t-1}. We assume the ϵ_{it}'s are independent of all X_{it}'s and denote their joint distribution as $p(\epsilon_{i1}, ..., \epsilon_{iT}; \theta)$. Note that the general formulation of $f_j(X_{it}, y_i^{t-1}, \epsilon_{it}; \theta)$ encompasses both traditional ("linear") discrete choice models as well as non-linear ones (arising, for example, from an agent's dynamic programming problem). Also note that this formulation implicitly allows current utility to depend on both past utilities as well as past choices.

In this model we can write the joint probability of a choice *sequence* $(y_{i1}, ..., y_{iT})$ as:

$$\begin{aligned} p(y_{i1}, ..., y_{iT} \mid X_{i1}, ..., X_{iT}; \theta) &= \textstyle\prod_{t=1}^{T} p(y_{it} \mid y_i^{t-1}, X_{i1}, ..., X_{iT}; \theta) \\ &= \int \textstyle\prod_{t=1}^{T} I\left[y_{it} = \arg\max_j \left[f_j(X_{it}, y_i^{t-1}, \epsilon_{it}; \theta)\right]\right] p(\epsilon_{i1}, ..., \epsilon_{iT}; \theta) \end{aligned} \tag{2.2}$$

where $I[\ \cdot\ ]$ is the indicator function[3]. Note that there are J^T possible choice sequences, where J is the number of alternatives j.

An alternative derivation of the choice sequence probability involves integrating inside the product, though this integration needs to be conditional on y_i^{t-1}, i.e.

$$\begin{aligned} p(y_{i1}, ..., y_{iT} \mid X_{i1}, ..., X_{iT}; \theta) &= \textstyle\prod_{t=1}^{T} p(y_{it} \mid y_i^{t-1}, X_{i1}, ..., X_{iT}; \theta) \\ &= \textstyle\prod_{t=1}^{T} \int I\left[y_{it} = \arg\max_j \left[f_j(X_{it}, y_i^{t-1}, \epsilon_{it}; \theta)\right]\right] p(\epsilon_{it} \mid y_i^{t-1}; \theta) \end{aligned} \tag{2.3}$$

Given observed data (y^T, X^T), an obvious approach to classical inference about the parameter vector θ is Maximum Likelihood estimation using either (2.2) or (2.3). Unfortunately, exact computation of either of these quantities is infeasible except in very special cases (e.g. the multidimensional logit model, i.i.d. low dimensional probit) . In general, the integral in (2.2) is of very large dimension (typically at least $J \cdot T$) without an analytical solution. While smaller in dimension, the integrals in (2.3) are over conditional distributions ($p(\epsilon_{it} \mid y_i^{t-1}; \theta)$) which are not generally specified as primitives nor easily computable.

3. Simulators of Sequence Probabilities

Given the inability to exactly compute these choice sequence probabilities, attention has turned to how to simulate them. We briefly detail the three most commonly used methods: 1) the "pure frequency" (PF) simulator, 2) what we call the "partially analytic" (PA) simulator, and 3) the GHK recursive simulator. We end this section by discussing conditions under which the extremely powerful GHK simulator can significantly reduce simulation error.

3.1. The Pure Frequency (PF) Simulator

The PF simulator is the most straightforward way to simulate choice sequence probabilities and is discussed in McFadden (1989). One simply takes NS draws from the joint distribution of the

[3]Note that this likelihood ignores potential initial condition problems due to the lagged dependent variables.

unobservables $p(\epsilon_1, ..., \epsilon_T; \theta)$ and forms (dropping i subscripts):

$$p_s^{PF}(\theta) = \frac{1}{NS} \sum_{ns=1}^{NS} \prod_{t=1}^{T} I \left[y_t = \arg\max_j \left[f_j(X_t, y^{t-1}, \epsilon_t^{ns}; \theta) \right] \right]$$

where the scalar or vector ϵ_t^{ns} is the nsth simulation draw on ϵ_t. It is straightforward to show that $p_s^{PF}(\theta)$ is an unbiased simulator of the probability of choice sequence s, $p_s(\theta) = p(y_{i1}, ..., y_{iT} \mid X_{i1}, ..., X_{iT}; \theta)$.

An important characteristic of the pure frequency simulator is that it can place probability 0 on sequences (for example, when $NS = 1$, $p_s^{PF}(\theta) = 1$ with probability $p_s(\theta)$ and $p_s^{PF}(\theta) = 0$ with probability 1-$p_s(\theta)$). In some applications (e.g. simulated maximum likelihood) this characteristic can be problematic, but in the long panel case it can actually be helpful, as we will see below. Another characteristic of the pure frequency simulator is that it is typically not continuous in the parameter vector θ and has flats ($\frac{\partial p_s^{PF}(\theta)}{\partial \theta} = 0$). The reason is that when θ changes, the indicator functions (and thus the product of indicator functions) may change discretely or not change at all.

Note that the feasibility of the PF simulator is not dependent on the functional form of $f_j(X_t, y^{t-1}, \epsilon_t^{ns}; \theta)$. It is also not particularly dependent on the distribution of ϵ. One solely needs to be able to take (pseudo) random draws from the joint distribution $p(\epsilon_1, ..., \epsilon_T; \theta)$.

3.2. The Partially Analytic (PA) Simulator

The PA simulator works off the fact that in most economic applications, one typically wants to include some source of iid (e.g. over time) unobservables. This is typically necessary in order to "explain" the data, i.e. so we do not observe probability zero events. These iid unobservables often result in one-dimensional integrals that are analytically computable *conditional* on the other unobservables in the model.

More formally, divide the unobservables ϵ into two components, μ and η. The distinction is that the integral in the choice sequence probability is analytically computable *conditional* on μ, i.e.

$$p_s(\theta) = \int p_s(\mu; \theta) f(\mu)$$

In other words, $p_s(\mu; \theta)$ - the probability of choice sequence s *given* the "non-analytic" unobservables μ, can be analytically computed (or feasibly computed to a very high degree of numerical accuracy, e.g. univariate normal integrals).

Many models fit into this formulation. Typically η contains the i.i.d. error and the persistent (e.g. over time) errors are in μ. One example is a multinomial logit model with random effects and/or random coefficients. In this case η includes the iid logit errors and μ includes the random coefficients/effects[4]. Conditional on μ, choice probabilities are the standard logit probabilities and

[4]Note that we have written the distribution of μ as not depending on θ. This is typically WLOG as any transformation of the random variable (depending on θ) can take place in the $p_s(\mu; \theta)$ function. A simple of example of this is where μ is a normal random effect and θ contains the standard deviation of this random effect. Keeping $f(\mu)$ as the N(0,1) density, we can adjust this standard deviation by multiplying μ by θ in the $p_s(\mu; \theta)$ function.

choice sequence probabilities are products of these standard logit probabilities. Importantly, Stern (1992) showed that one can also use the PA simulator in most probit models, even in those without an explicit i.i.d. term. In the binary AR(1) case, for example, one can "pull out" of the AR(1) process an iid process that is analytically integrable[5].

Because the integrals over η are done analytically, the PA simulator typically has less simulation error than the corresponding PF simulator (for a given number of draws). Computing the PA simulator is almost as simple as the PF simulator - one simply takes NS draws from the joint distribution of μ, $f(\mu)$, and forms:

$$p_s^{PA}(\theta) = \frac{1}{NS}\sum_{ns=1}^{NS} p_s(\mu_{ns};\theta)$$

An important characteristic of $p_s^{PA}(\theta)$ is that, unlike $p_s^{PF}(\theta)$, it is usually smooth in the parameter vector θ and that the simulated sequence probabilities are typically strictly between 0 and 1[6].

As is the PF simulator, the PA simulator is fairly robust to unobservables entering nonlinearly into the decision rule. In this case it is inconsequential how μ enters the decision rule, e.g. it doesn't matter whether we have $u_{jt} = f_j(X_t, y^{t-1};\theta)+\mu_{jt}+\eta_{jt}$ or $u_{jt} = f_j(X_t, y^{t-1}, \mu_{jt};\theta)+\eta_{jt}$. This makes the simulator straightforward to apply to discrete choice problems with dynamic optimization on the part of agents. In these problems, persistent unobservables that are linear in the static utility function typically enter the decision rule non-linearly (see below). Also, like the PF simulator, the PA simulator is not particularly dependent on distributional assumptions on the unobservables μ . One simply needs to be able to draw from the joint distribution of μ.

3.3. The Geweke-Hajivassiliou-Keane (GHK) Simulator

The last sequence probability simulator we discuss is the extremely powerful GHK simulator developed by Geweke (1989), Hajivassiliou and McFadden (1990), and Keane (1990). The GHK simulator is in the class of importance sampling simulators (see, e.g. McFadden (1989), Ross (1990)) - these are simulators where one draws from some distribution other than $p(\epsilon_1, ..., \epsilon_T;\theta)$ and then reweights to obtain an unbiased simulator. Importance sampling can reduce simulation error by oversampling parts of the error distribution that are most informative.

[5]For example, consider a normal AR(1) process ϵ_t with persistance ρ. The variance matrix Σ_ϵ of this (normalized) process has diagonal elements 1 and off diagonal elements ρ^{t-s}. Stern notes that we can write $\epsilon_t = \epsilon_t^A + \epsilon_t^{NA}$, where ϵ_t^A is an i.i.d. process and ϵ_t^{NA} is a serially correlated process. As a result, for, e.g. a binary probit model, choice probabilties are analytic conditional on ϵ_t^{NA}. The intuition here is that one is dividing the variance matrix into Σ_ϵ $=\Sigma_\epsilon^A + \Sigma_\epsilon^{NA}$ where Σ_ϵ^A is diagonal and Σ_ϵ^{NA} is not. To make "as much" variation as possible analytically integrable, one typically wants to make the diagonal of Σ_ϵ^A as big as possible while keeping Σ_ϵ^{NA} a proper variance matrix. Stern (1992) shows that an effective way of doing this is to make the diagonal of Σ_ϵ^{NA} equal to the smallest eigenvalue of Σ_ϵ. Stern also examines this technique for the multinomial probit model.

[6]This is typically the case since the i.i.d. unobservables typically have infinite support (e.g. logit, normal errors) and thus put strictly positive probability on any choice sequence.

For ease of discussion we specialize to the binary choice case, noting that we can now write:

$$p_s(\theta) = p(y_1, ..., y_T \mid X_1, ..., X_T; \theta) \quad = \int \prod_{t=1}^{T} I\left[(2y_t - 1) f(X_t, y^{t-1}, \epsilon_t; \theta) > 0\right] p(\epsilon_1, ..., \epsilon_T; \theta)$$

where the model is $y_t = 1$ iff $f(X_t, y^{t-1}, \epsilon_t; \theta) > 0$. For the multinomial case one can consult the literature above or Borsch-Supan and Hajivassiliou (1992), Keane (1994), Hajivassiliou (1996). The GHK simulator can be written as follows:

$$\begin{aligned} p_s^{GHK}(\theta) &= \tfrac{1}{NS} \textstyle\sum_{ns=1}^{NS} \int \prod_{t=1}^{T} I\left[(2y_t - 1) f(X_t, y^{t-1}, \epsilon_t; \theta) > 0\right] \\ &\qquad p(\epsilon_1; \theta) p(\epsilon_2 \mid \epsilon_{1ns}; \theta) p(\epsilon_3 \mid \epsilon_{2ns}, \epsilon_{1ns}; \theta) ..... p(\epsilon_T \mid \epsilon_{ns}^{T-1}; \theta) \qquad (3.1) \\ &= \tfrac{1}{NS} \textstyle\sum_{ns=1}^{NS} \prod_{t=1}^{T} \int I\left[(2y_t - 1) f(X_t, y^{t-1}, \epsilon_t; \theta) > 0\right] p(\epsilon_t \mid \epsilon_{ns}^{t-1}; \theta) \end{aligned}$$

where the ϵ_{tns} are recursively generated random draws from the distribution of ϵ_t *conditional* on *both* the prior draws ϵ_{ns}^{t-1} *and* the condition that $I\left[(2y_t - 1) f(X_t, y^{t-1}, \epsilon_{tns}; \theta) > 0\right] = 1$. Note that in this general formulation it has not been specified whether ϵ_t is a scalar unobservable or a vector of unobservables. It is fairly straightforward to show that $p_s^{GHK}(\theta)$ is an unbiased simulator of $p_s(\theta)$[7].

Examining the above suggests that we need to do two things to compute $p_s^{GHK}(\theta)$:

1) draw the conditional, truncated, random variables ϵ_{tns} according to the above process.

2) evaluate the integrals over ϵ_t in (3.1).

The fact that normal distributions are closed under addition and conditioning makes this process extremely easy for the panel probit model. For example, consider a probit model with a random effect, an AR(1) process, and an i.i.d. error term, - all normals and entering additively in the latent variable function, i.e.

$$u_t = f(X_t, y^{t-1}; \theta) + \epsilon^A + \epsilon_t^B + \epsilon_t^C$$

[7] Let $f()$ denote the joint distribution of $(\epsilon_{ns1}, ..., \epsilon_{nsT})$ under the recursive drawing scheme. Then

$$\begin{aligned} Ep_s^{GHK} &= E_{ns} \tfrac{1}{NS} \textstyle\sum_{ns=1}^{NS} \prod_{t=1}^{T} \int I(\epsilon_t) p(\epsilon_t \mid \epsilon_{ns}^{t-1}) \\ &= \int \left[\prod_{t=1}^{T} \int I(\epsilon_t) p(\epsilon_t \mid \epsilon_{ns}^{t-1}; \theta) d\epsilon_t\right] f(\epsilon_{ns1}, ..., \epsilon_{nsT}) \\ &= \int \left[\prod_{t=1}^{T} \Pr(y_t \mid \epsilon_{ns}^{t-1})\right] \prod_{t=1}^{T} f(\epsilon_{nst} \mid \epsilon_{ns}^{t-1}) \\ &= \int_{I_1, ..., I_T} \left[\prod_{t=1}^{T} \Pr(y_t \mid \epsilon_{ns}^{t-1})\right] \prod_{t=1}^{T} \tfrac{p(\epsilon_{nst} \mid \epsilon_{ns}^{t-1})}{\Pr(y_t \mid \epsilon_{ns}^{t-1})} \\ &= \int_{I_1, ..., I_T} \prod_{t=1}^{T} p(\epsilon_{nst} \mid \epsilon_{ns}^{t-1}) = \int_{I_1, ..., I_T} p(\epsilon) = p_s \end{aligned}$$

Note that $\int_{I_1, ..., I_T}$ indicates the integral over the region such that $I_t = 1$ for $t = 1, ..., T$ (and note that $f() = 0$ outside of this region). We note for later that unfortunately the individual elements in the product in (3.1) are *not* (generally) unbiased (nor consistent) estimates of transition probabilities, i.e. $\Pr(y_t \mid y^{t-1})$. Lastly, note that GHK corresponds with an importance sampling density of

$$h(\epsilon_1, ..., \epsilon_T) = \frac{\prod p(\epsilon_t \mid \epsilon^{t-1})}{\prod \Pr(y_t \mid \epsilon^{t-1})}$$

defined over the above region.

Define $\epsilon_t = \epsilon^A + \epsilon_t^B + \epsilon_t^C$ so the ϵ_t's in (3.1) are scalar. Then

$$\epsilon = \begin{pmatrix} \epsilon_1 \\ . \\ . \\ \epsilon_T \end{pmatrix} \sim N(0, \Sigma^A + \Sigma^B + \Sigma^C)$$

where Σ^A is a diagonal matrix with diagonal elements equal to the variance of the random effect, Σ^B is the AR(1) covariance matrix, and Σ^C is a diagonal matrix with diagonal elements equal to the variance of the iid error[8].

Since ϵ is a T-variate normal, the conditional distributions $p(\epsilon_t \mid \epsilon^{t-1}; \theta)$ are also normal. The Cholesky decomposition of $\Sigma = \Sigma^A + \Sigma^B + \Sigma^C$ defines the mean and variance of these conditional distributions. As a result, steps 1) and 2) above are extremely simple. For step 2), the integrals in (3.1) are all over *univariate* normals and thus equal to simple transformations of the standard normal CDF. Drawing from the univariate truncated normal distributions for step 1) is also trivial using uniform random draws and the inverse normal CDF. In the binary choice case, the bulk of the computational work in evaluating (3.1) is computing $NS * T$ univariate normal CDFs (to compute the integrals) and $NS*T$ inverse CDFs (to draw from the truncated normal distributions). Note that this will tend to be, at most, twice as slow as the PA simulator, for which the bulk of computational work would be in computing $NS * T$ univariate normal CDFs[9].

There is a great deal of evidence (e.g. Borsch-Supan and Hajivassiliou (1992), Hajivassiliou, McFadden, and Ruud (1996)) that the GHK simulator is extremely efficient for simulating sequence probabilities in probit models. The first two columns of Table 1 present some additional evidence comparing the GHK simulator to a simple PA simulator where ϵ^A and ϵ_t^B are drawn from their marginal distributions and ϵ_t^C is integrated over analytically[10]. The number of simulation draws is set such that computation time is equalized across the two simulators. The cells report the simulation variance[11] in simulating the log-likelihood of a sample of size 500[12]. For alternative

[8]For estimation there needs to be a multiplicative normalization somewhere. Note also that the iid error is seperately identified from the AR(1). The iid innovation in the AR(1) process persists through time, but the iid error does not.

[9]In arriving at this doubled computational time we do not include the time required to draw the "seed" standard normals and standard uniforms for use in simulation (which itself requires inverse CDFs). The reason is that in practice, these draws need to be held constant over function evaluations, and as such, it is far more efficient to draw them initially and store them for continued use.

[10]Stern's PA simulator, where one additionally pulls out the iid "component" of ϵ_t^B, does slightly better. The reason we focus on the simple PA simulator is that in the more complicated models below where the GHK simulator is not easily applicable, Stern's PA simulator is also not easily applicable.

[11]We report simulation variance rather than simulation standard error because this gives us a better idea of the time-wise efficiency of GHK. Simulation variance should be approximately proportional to NS, so a simulator that has half the simulation variance for equal computational time should also take half the time for the same simulation variance (or simulation error).

[12]We examine at simulation error in the log-likelihood function rather than simultion error in probabilities because in some experiments this appeared to correspond better to error in estimated parameters than does error in probabilities. This makes a difference for the comparisons because the GHK simulator does relatively better with small probabilities than the other simulators. The general structure of the model is very similar to those in Keane (1994). This model

parameter values, GHK can be 5-60 times more efficient than the PA simulator time-wise (the relative variance column gives the approximate time-wise efficiency of GHK, e.g. for the baseline model, GHK is 13 times quicker than the PA at achieving a given simulation variance)[13]. Note that the relative effectiveness of the GHK simulator increases as the length of the panel increases.

The intuition behind GHK's effectiveness is that it is a very potent form of importance sampling, i.e. it oversamples parts of the error distribution that are most informative. Importantly, this importance sampling is continuously refined as one works forward through time in the iterative process. Draws of ϵ_{tns} come from an importance sampling distribution that implicitly depends on *all* past choices. It *is not* surprising that there *exists* such a good importance sampling simulator for panel probit sequence probabilities[14]. What *is* surprising about the GHK simulator is the combination of its ease and its effectiveness. It is essentially a "cookbook" procedure that works extremely well for any simple probit model with arbitrary correlation structure.

3.4. Limitations of the GHK Simulator

Unfortunately, for more general error structures the GHK simulator can lose a considerable amount of its effectiveness and ease. We focus on two cases - the first are situations where there is non-normality in the unobservables. The second is when the decision rule is non-linear in the unobservables. Such non-linearities can arise naturally through dynamic optimization on the part of agents. The essential problem in both cases is that one can (generally) no longer "add" multiple unobservables (e.g. ϵ^A, ϵ^B_t, and ϵ^C_t from above) in period t into a single scalar unobservable ϵ_t for which the density $p(\epsilon_t \mid \epsilon^{t-1})$ can be "analytically" (i.e. quickly and accurately) integrated over and drawn from.

Again, it is important to note that even in these more general cases, there exist good importance sampling distributions. The problem is finding and implementing them. Our argument is that the already "found" GHK importance sampling scheme (and slight derivatives of it) lose their effectiveness on these problems. As a result, unless one wants to search out idiosyncratic importance sampling densities for idiosyncratic specifications, there is large value in finding estimation procedures that are robust to significant amounts of simulation error.

---

has the following setup: $T = 8$, the single covariate X_{it} has mean 6, within (individual) variance 2, and across individual variance 3. The constant term equals -0.9 and the coefficient on X is .25. Variances of the 3 unobservables and ρ are reported on the table.

[13] All simulations were done in highly optimized C code. Notethat in timing, we did not include the time required to draw the "seed" standard normals and standard uniforms for use in simulation. The reason is that in practice, these draws need to be held constant over function evaluations, and as such, it is far more efficient to draw them initially and store them for continued use. As such, the bulk of computation time for the PA simulator is computing the normal CDF (this needs to be done $NS * T$ times), for GHK in computing the normal CDF and inverse normal CDF (each needs to be done $NS * T$ times). As a result, the PA simulator was about 1.7 times as fast per simulation draw. We used lookup tables and interpolation to compute both normal CDFs and inverse CDFs. These were faster (about 2X) than any other methods we have seen, and seemingly fairly accurate (in estimation, their use had virtually 0 effect on estimated parameters). Use of slower CDF/PDF methods relatively helped (very slightly) the PA method speedwise.

[14] In fact, in theory there is an optimal importance sampling distribution such that simulation error equals zero (although this optimal distribution requires knowledge of the integral itself).

3.4.1. Non-Normal Unobservables

First consider the simple example:

$$u_t = f(X_t, d^{t-1}; \theta) + \epsilon^A + \epsilon_t^C$$

where ϵ_t^C is still an i.i.d normal unobservable but now the random effect ϵ^A has some non-normal distribution (e.g. a mixture of normals). As a result, steps 1 and 2 above will typically not be straightforward. In particular, it is unlikely that we will be able to easily either draw from or integrate over the conditional distributions of the summed unobservables $p(\epsilon_t \mid \epsilon^{t-1})$ (where $\epsilon_t = \epsilon^A + \epsilon_t^C$). Thus, the "scalar" (referring to the dimension of ϵ_t) GHK process described above will not be easily applicable[15].

In this situation there are two options to apply the spirit of GHK to this problem. The first is to continue to follow equation (3.1) now treating ϵ_t as the *vector* $(\epsilon^A, \epsilon_t^C)$ rather than the scalar sum $\epsilon^A + \epsilon_t^C$. We call this "vector" GHK. In this case, we do know $p(\epsilon_t \mid \epsilon^{t-1})$ (and it is fairly simple - the first element of ϵ_t equals the first element of ϵ_{t-1} with probability 1, the second element is an i.i.d normal). Thus for $t > 1$, the integration (which are essentially one-dimensional normals since the first element ϵ^A is degenerate) and drawing (in fact one doesn't even need to draw because the first element is degenerate and the second element is iid and thus inconsequential for the future) is straightforward. The only minor complication is for $t = 1$. To follow (3.1) exactly, the first integral (which is 2-dimensional) would probably need to be simulated and one might need to use acceptance/rejection methods to draw $(\epsilon^A, \epsilon_1^C)$ from the appropriate truncated distribution[16].

Perhaps more important than these complications is the fact that this "vector" GHK method will typically have significantly worse importance sampling properties than a "scalar" GHK method that utilizes the distribution of the sum $\epsilon^A + \epsilon_t^C$. In the vector method, the random effect ϵ^A is drawn from a truncated distribution depending on the $t = 1$ choice but is then held fixed through time. Thus, although ϵ^A is still being importance sampled, this importance sampling is only based on the period 1 choice. This contrasts to the scalar GHK case where at every t, ϵ_t (and thus ϵ^A) is implicitly importance sampled depending on *all* past choices. We show below that this distinction has very large implications on the precision of simulated choice sequence probabilities.

Note that if one adds back in a non-normal serially correlated unobservable (ϵ_t^B) to the above matters get worse as one needs to evaluate (probably by simulation) multiple integrals in each time period as well as drawing from multidimensional truncated distributions[17]. This is extremely costly because one is essentially doing simulations within simulation, i.e. one would need to simulate T two dimensional integrals for *every* GHK draw.

A second approach is to break up ϵ^A, ϵ_t^B, and ϵ_t^C individually in the sequential conditioning.

[15] Note that one could draw from or integrate over these conditional distributions using simulation, but this would involve embedding simulation procedures within simulation procedures, something that would pretty clearly not be time-efficient.

[16] There is an easier way to deal with $T = 1$ which involves sequentially drawing ϵ^A and ϵ_1^C. See below for related discussion.

[17] If ϵ_t^B were a serially correlated normal one could work with the sum $\epsilon_t^B + \epsilon_t^C$ to simplify things.

This eliminates the need to either simulate multiple integrals or draw from multivariate truncated distributions within each GHK draw. Unfortunately, since the three unobservables in each period combine to satisfy only one inequality, we can only draw one of the three unobservables in each period from the truncated importance sampling distribution[18]. For example, we could draw ϵ^A_{ns} and $\epsilon^C_{1ns}, ..., \epsilon^C_{Tns}$ from their marginal distributions and then recursively draw ϵ^B_{nst}'s from the truncated conditional distributions such that the observed choices are realized. Formally we can write this simulator as

$$p^{GHK}_s(\theta) = \frac{1}{NS}\sum_{ns=1}^{NS}\int\prod_{t=1}^{T} I\left[(2y_t-1)f(X_t, y^{t-1}, \epsilon^A_{ns}, \epsilon^B_t, \epsilon^C_{tns};\theta) > 0\right] \\ p(\epsilon^B_1;\theta)p(\epsilon^B_2 \mid \epsilon^B_{1ns};\theta)p(\epsilon^B_3 \mid \epsilon^B_{2ns}, \epsilon^B_{1ns};\theta).....p(\epsilon^B_T \mid \epsilon^{B,T-1}_{ns};\theta)$$

We call this "partial GHK" as some of the unobservables ($\epsilon^A_{ns}, \epsilon^C_{1ns}, ..., \epsilon^C_{Tns}$) are being simulated from their marginal distributions while conditional on these unobservables, the remaining unobservables (the ϵ^B_{nst}'s) are being importance sampled according to truncated distribution. There are other alternatives based on which unobservables get importance sampled. For example, one could draw ϵ^B_{1ns} and $\epsilon^C_{1ns}, ..., \epsilon^C_{Tns}$ from their marginal distributions, and then ϵ^A_{ns} and $\epsilon^B_{2ns}, ..., \epsilon^B_{Tns}$ from truncated conditional distributions. A third alternative would be to draw ϵ^A_{ns} and $\epsilon^B_{1ns}, ..., \epsilon^B_{Tns}$ from their marginal distributions and the ϵ^C_{nst}'s from truncated conditional distributions. Note that this last alternative corresponds exactly to our simple PA simulator[19].

The last two columns of Table 1 compare these alternative GHK methods to the full GHK simulator. As with the simple PA simulator, there is a very large degradation in simulation efficiency relative to full GHK. The reasoning behind this degradation is the same as that in the discussion of the "vector GHK" above - i.e. ϵ^A is either not being importance sampled or is being importance sampled based on only the period 1 decision. Which performs best out of the 2 partial GHK simulators and the PA simulator (which is also the last partial GHK simulator) depends on parameter values. What this evidence suggests is that these partial GHK simulators are not good substitutes for full GHK. Thus, for models where full GHK cannot be used, we may have to live with large amounts of simulation error.

The previous arguments also apply to random coefficients models, e.g.

$$u_t = \beta_i X_t + \epsilon_t$$

with random coefficients β_i. Again the crucial distinction is what one is willing to assume about the unobservables. If the random coefficients are assumed normally distributed (as well as the ϵ_t), one could do full GHK (Although there is a caveat that the variance matrix differs over individuals (due to differences in X's) and this would therefore require separately computing the variance covariance

[18] Note the difference between this and the multinomial probit model, where there are also multiple observables in each period. The key there is to write the inequalities as a sequence of recursive indicator functions (this cannot be done with any effectiveness in our case).

[19] In this alternative one doesn't actually have to draw the ϵ^C_{nst}'s since they are i.i.d. and have no impact on the future.

matrix and Cholesky decompositions for each observation - this can be time consuming for large T.) More importantly for our purposes, if one is not willing to assume that the β_i are normal, ability to capitalize on full GHK will again be limited.

Normals may be the obvious first choice for heterogeneity - whether it is random effects, random coefficients, or serially correlated processes. However, given the size and extent of today's panel data sets, assuming normality is clearly "overparameterized" in the sense of Heckman and Singer (1987). The point of the above is to argue that as we move towards less parametric specifications, we can no longer rely on the GHK simulator to significantly reduce simulation error.

3.4.2. Non-Linear Models

Another set of models where the GHK simulator cannot be applied effectively are models where the latent variable is a non-linear function of the unobservables. An often cause of such non-linearities is optimal dynamic behavior on the part of agents. Consider, for example, a simple model where an agents single period utility (or profit) is linear in the unobservables, i.e.

$$U_t = X_t\beta + g(d^{t-1}) + \epsilon^A + \epsilon_t^B + \epsilon_t^C$$

where ϵ^A, ϵ_t^B, and ϵ_t^C are again respectively a random effect, AR(1) process, and iid process, all normal. Because past choices affect current utility, one might assume a forward looking agent who maximizes the expected discounted value of future utilities $\sum_{t=1}^{\infty}\delta^t U_t$. In general, this sort of a model will result in a decision rule (i.e. latent variable function) that is non-linear in the persistent unobservables, i.e.

$$d_t = 1 \quad iff \quad u_t = f(X_t, d^{t-1}, \epsilon^A, \epsilon_t^B; \theta) + \epsilon_t^C > 0$$

Note that $f(\ )$ is a function of ϵ^A and ϵ_t^B individually, not just the sum. In the static optimization problem, it doesn't matter whether a high $\epsilon^A + \epsilon_t^B$ was caused by a high ϵ^A or a high ϵ_t^B. In the dynamic case it does because of their different implications for the future. ϵ^A persists completely into the future, while ϵ_t^2 will "depreciate". Note that the iid error ϵ_t^C will typically enter the decision rule linearly since it has no effect on the future[20].

In this case it is the non-linearities in the decision rule that prevents us from combining (adding) the unobservables into a single scalar. Again, a vector GHK procedure with $\epsilon_t = (\epsilon^1, \epsilon_t^2, \epsilon_t^3)$ is conceivable but involves integrating over and drawing from two dimensional distributions (for every T within every simulation draw NS), which would be prohibitively expensive to do[21]. Similarly,

[20] This assumes that the unobservables ϵ^A, ϵ_t^B, and ϵ_t^C are individually (seperately) observed by the optimizing agent. This makes sense, e.g., if ϵ^A is the individuals known taste for a particular alternative. Things would be different if the agent only observed the sum $\epsilon_t = \epsilon^A + \epsilon_t^B + \epsilon_t^C$. While this assumption might make integration easier (in this case there really is only one unobservable in each period), it makes dynamic programming much harder as the program loses its simple Markov nature and becomes more of a learning problem (the state space at t would need to include information on all past ϵ_t's because, e.g. they all provide information on ϵ^A).

[21] Even if one could do this, the importance sampling would likely be much less successful, as again, e.g. ϵ^A would be importance sampled based only on the period 1 decision. Note that in a simpler model without the iid error ϵ_t^C, the integrals are back to one dimensional. Thus, if $f(.)$ is monotonic in ϵ_t^B (which is not necessarily the case),

partial GHK is possible, but would generally suffer from the same problems as those addressed in Table 1 - i.e. it will tend to be a far less effective importance sampling distribution.

4. Review of Existing MSM Estimators for Panel Data

We now review more thoroughly two extant MSM methods of estimating panel data discrete choice models. As MSM estimators can be more robust to simulation error than ML estimators, this is a natural place to turn for the non-normal and non-linear models described above where it will generally be hard to reduce simulation error.

The first method, due to McFadden (1989), follows his simulation methods for the single period discrete choice model by simply treating sequences of choices as choices. Let s index the J^T possible choice sequences (i.e. possible $(y_{i1}, ..., y_{iT})$), let $p_s(\theta)$ denote the probability of choice sequence s (given by (2.2)), and let d_s be an indicator variable equal to 1 for the observed choice sequence for individual i. McFadden considers the moment (for observation i - we drop i subscripts and explanatory variables from now on) :

$$g(\theta) = \sum_{s=1}^{J^T} w_s \left(d_s - p_s(\theta)\right) \tag{4.1}$$

where w_s are exogenous weight vectors of dimension greater than that of θ. As $d_s = 1$ with probability $p_s(\theta_0)$ (θ_0 is the true parameter vector), the expectation of $g(\theta)$ is equal to 0 at θ_0. Thus, the estimator θ^* that sets the sample average of the $g(\theta)$'s as close as possible to zero is a consistent estimator of θ_0 for any exogenous w's. If $w_s = \frac{\partial \ln(p_s(\theta))}{\partial \theta}$, then (4.2) reduces to the Maximum Likelihood first order condition and θ^* is efficient.

The beauty of this estimator is that the p_s's need not be exactly computed for consistency in the sample size. If one replaces the p_s with unbiased simulations of these values, $\widehat{p}_s(\theta) = p_s(\theta) + \nu_s$ (where ν_s is mean 0 simulation error), we obtain:

$$g_{MSM}(\theta) = \sum_{s=1}^{J^T} w_s \left(d_s - \widehat{p}_s(\theta)\right) = \sum_{s=1}^{J^T} w_s \left(d_s - p_s(\theta) - \nu_s\right) \tag{4.2}$$

Since $g_{MSM}(\theta)$ is linear in the simulation error ν_j, this error averages out over observations. Thus, regardless of the (finite) variance of the simulation error (i.e. the number of simulation draws), θ_{MSM} is consistent as the number of observations increases (for proofs see McFadden (1989) and Pakes and Pollard (1989)).

Unfortunately, there are number of problems with McFadden's panel data estimator. First,

one could do the integral "analytically". However, this would require inverting $f(.)$, which could potentially be time consuming given that in the typical dynamic programming problem, one will only have numeric approximations to $f(.)$. In addition, this reduction to a one dimensional integral would quickly disappear with either the addition of any other time varying unobservables or moving to a multinomial model (In a multinomial *dynamic* choice problem, one will generally not be able to write the problem do as a sequence of recursive inequalities (think this through again....)).

as T increases, it quickly becomes infeasible to calculate (or in general simulate) probabilities of and weights for all J^T choice sequences. McFadden handily avoids these computational problems by utilizing a pure frequency simulator of the p_s's. Since a draw from a pure frequency simulator places probability zero on all choice sequences except for one, most of the moments corresponding to the J^T elements in the "choice sequence set" are equal to zero. In fact, the number of non-zero elements in the sum is at most $NS + 1$, where NS is the number of pure frequency simulation draws (the 1 corresponds to the observed choice sequence where $d_s = 1$).

While this method provides consistent parameter estimates for a finite number of simulation draws, there are a couple of problems. First the pure frequency simulator provides relatively high variance simulation error. Typically one can obtain much lower simulation variance using the partially analytic or GHK methods described above. None of these methods can be used in this formulation as they place positive probability on each of the J^T choice sequences and thus result in an incomputable sum. On the other hand, even the variance using a pure frequency simulator is very manageable. First note that the variance of the true $g(\theta)$ (the variance due to the data generating process, i.e. the realization of d_s) is:

$$Var\left(g(\theta)\right) = Var\left(\sum_{s=1}^{J^T} w_s\left(d_s - p_s(\theta)\right)\right) = Var\left(\sum_{s=1}^{J^T} w_s d_s\right) = \Phi$$

Now, the variance of McFadden's MSM pure frequency moment, $g_{PF}(\theta)$ is:

$$\begin{aligned} Var(g_{PF}(\theta)) &= Var\left(\textstyle\sum_{s=1}^{J^T} w_s d_s\right) + Var\left(\textstyle\sum_{s=1}^{J^T} w_s \widehat{p}_s(\theta)\right) \\ &= \Phi + Var\left(\tfrac{1}{NS}\textstyle\sum_{n=1}^{NS}\sum_{s=1}^{J^T} w_s \widehat{p}_s^n(\theta)\right) \\ &= \Phi + \tfrac{1}{NS} Var\left(\textstyle\sum_{s=1}^{J^T} w_s \widehat{p}_s^n(\theta)\right) \\ &= (1 + \tfrac{1}{NS})\Phi \qquad \text{at } \theta_0 \end{aligned}$$

where $\widehat{p}_s^n(\theta)$ is equal to 1 for the sequence generated by the nth simulation draw. Note that the last line follows because at θ_0, $\widehat{p}_s^n(\theta_0)$ has a distribution identical to d_s. As the variance of the GMM estimator θ_{GMM} is proportional to $Var\left(g(\theta)\right) = \Phi$, this implies that the asymptotic (as $\theta_{PF} \to \theta_0$) variance of θ_{PF} is only $(1 + \frac{1}{NS})$ times the variance of θ_{GMM}. Even though smooth simulators will typically do better than $(1 + \frac{1}{NS})\Phi$, they obviously can't do better than Φ - so even for small NS (e.g. 10) the efficiency loss from using a PF simulator relative to smooth simulators will be small.

Perhaps more problematic with McFadden's solution is that the pure frequency simulator results in an objective function that is not continuous in θ. This follows directly from the fact that the PF sequence probability simulator $\widehat{p}_s^n(\theta)$ changes discretely (or not at all) in θ. The resulting jumps and flats[22] in $g_{PF}(\theta)$ can be a huge problem to finding (correctly) global extremum of the objective function. A second issue regarding this discreteness is that it results in the non-zero elements of the sum in (4.2) changing over θ. Thus one needs to compute w_s for all sequences that are ever

[22]One can potentially eliminate the flats by allowing the weights to change with θ, although this would be time consuming if one wants to approximate optimal weights. This also would not eliminate the jumps.

encountered in an optimization procedure (i.e. for all θ ever searched over). If one is attempting to approximate optimal w_s's this is particularly time consuming because the score is hard to simulate precisely, particularly with the low sequence probabilities which will occur as T increases.

Keane (1990,1994) and Berkovec and Stern (1991) suggest an alternative method for dealing with discrete choice panel data. We call this the transition probability (TP) estimator as they form moments in transition probabilities, i.e. the expectations (probabilities) of choices in period t conditional on observed choices in periods 1,....,$t-1$. Formally, with the dummy variable $d_{jt}=1$ iff $y_t=j$ (and $d_t=\begin{pmatrix} d_{1t} \\ . \\ . \\ d_{Jt} \end{pmatrix}$), they examine the moment:

$$g(\theta)=\sum_{t=1}^{T}\sum_{j=1}^{J} w_{jt}\left(d_{jt}-\Pr(d_{jt}=1 \mid d_1,.....,d_{t-1};\theta)\right)$$

where $\Pr(d_{jt}=1 \mid d_1,.....,d_{t-1};\theta)$ is the probability that j is chosen at t *conditional* on past choices. This moment clearly has expectation 0 at θ_0, and one can again show that given an optimal weight matrix, this moment is equal to the derivative of the likelihood function (in particular the formulation in (2.3)). Note that in theory this formulation is more parsimonious than McFadden's in that only $T*J$ conditional probabilities need to be calculated, rather than J^T choice sequence probabilities.

The main issue here is that to simulate these conditional probabilities, one needs to integrate out over the distribution of period t unobservables *conditional* on the history of choices (and exogenous variables). With any persistence in errors through time (e.g. individual heterogeneity, serial correlation), this conditional distribution *does* depend on this history and is not analytically known. These authors tackle this problem by simulating these transition probabilities. The crudest version of this approach is to take draws from the joint distribution of unobservables in periods 1,....,t. These unobservables generate a sequence of choices for the periods 1,....,$t-1$. If this sequence equals the observed choice sequence for 1,....,$t-1$, this draw is called an accepted conditioning sequence. Repeatedly doing this, the period t unobservables from the accepted conditioning sequences are draws from the correct conditional distribution and can be used to form unbiased simulation estimates of the conditional probabilities of choices in period t. The problem with this crude method is that even for medium T, it can take a large number of attempts just to get 1 accepted conditioning sequence, let alone enough to get a precise simulation estimate. This renders this acceptance/rejection method computationally impractical except for very small T.

Both authors suggest a second method which is much more computationally efficient. First note that

$$\Pr(d_{jt}=1 \mid d_1,.....,d_{t-1};\theta)=\frac{\Pr(d_1,.....,d_{t-1},d_{jt}=1;\theta)}{\Pr(d_1,.....,d_{t-1};\theta)}=\frac{p_{s^{t-1},j}(\theta)}{p_{s^{t-1}}(\theta)} \tag{4.3}$$

where $p_{s^{t-1}}(\theta)$ is the probability of the observed choice sequence through time $t-1$, and $p_{s^{t-1},j}(\theta)$

is the joint probability of the observed choice sequence through time $t-1$ *and* $d_{jt}=1$. Both $p_{s^{t-1}}(\theta)$ and $p_{s^{t-1},j}(\theta)$ are choice sequence probabilities that can be simulated using the methods described in section 2. The difference between the Keane approach and the Berkovec and Stern approach is that Berkovec and Stern use a PA simulator to simulate the numerator and denominator of (4.3) while Keane uses the GHK simulator[23]. Keane's moment condition, for example, is:

$$g^{TP}(\theta)=\sum_{t=1}^{T}\sum_{j=1}^{J}w_{jt}\left(d_{jt}-\frac{\widehat{p}^{GHK}_{s^{t-1},j}(\theta)}{\widehat{p}^{GHK}_{s^{t-1}}(\theta)}\right)$$

It is important to note that because both these methods must simulate the *denominator* of this transition probability, neither provides unbiased simulations of either the transition probability or the conditional moment. As such, neither estimator is consistent for a fixed number of simulation draws. However, Keane shows that the TP estimator is consistent if $\frac{NS}{\sqrt{N}}\rightarrow\infty$ as $N\rightarrow\infty$. In addition, the extra variance induced by simulation should be less than $(1+\frac{1}{NS})\Phi$ (asymptotically) since these smooth simulators of transition probabilities should exhibit less variance than those of the unbiased acceptance/rejection method described above.

Keane exhibits Monte-Carlo evidence[24] that even for small NS (e.g. 10), the small sample bias of the TP estimator is negligible in some 8-period binomial probit models with AR(1) plus random effect unobservables . Geweke, Keane, and Runkle (1997) present some additional Monte-Carlo evidence verifying this limited bias on longer panel multinomial probit models. Importantly, both these Monte-Carlo experiments study pure probit models, i.e. models where all the unobservables are normal and enter the decision rule additively. As a result the GHK simulator is fairly easily applicable and very effective at simulating probabilities and transition probabilities. The goal of this paper is to study the models described at the end of section 2 in which GHK is not as easy or effective, i.e. dynamic programming models or models with less parametric heterogeneity. Unfortunately, we find evidence that the TP estimator does not perform nearly as well in these models. It appears that the TP estimator can exhibit significant biases, even when many more simulation draws are used.

Keane (1994) also mentions (but does not test) an alternative moment condition that does result in consistent estimates for fixed NS. He gets this by multiplying the moment by $\widehat{p}_{s^{t-1}}(\theta)$, i.e.

$$g^{CTP}(\theta)=\sum_{t=1}^{T}\sum_{j=1}^{J}w'_{jt}\left(d_{jt}\widehat{p}_{s^{t-1}}(\theta)-\widehat{p}_{s^{t-1},j}(\theta)\right)$$

where the optimal weights are now $w'_{jt}=\frac{w_{jt}}{\widehat{p}_{s^{t-1}}(\theta)}$. As long as different random draws are used to compute both the elements of w'_{jt} than are used to compute $d_{jt}\widehat{p}_{s^{t-1}}(\theta)-\widehat{p}_{s^{t-1},j}(\theta)$ this moment has expectation 0 at θ_0 for fixed NS. As such, the estimator should be consistent for fixed NS.

[23] As Keane develops his estimator as an importance sampling version of the acceptance-rejection method described above, it is not completely obvious that his simulated transition probabilities are ratios of GHK sequence probabilities.

[24] The TP estimator has been applied to actual data in, e.g. Berkovec and Stern (1991), Elrod and Keane, Stern (1994), Roberts & (1997).

While this estimator is also asymptotically efficient (in NS and N), this simulation estimator loses the "replication of the data generating process" characteristic of the original TP estimator and it is not clear how much variance simulation error adds to estimates. In Monte-Carlo results below, we exhibit evidence that while this estimator does appear to have virtually no small-sample bias, it works very poorly (efficiency-wise - in comparison to our proposed estimator) for random coefficient/dynamic programming models in which the GHK simulator cannot be used effectively.

Before proceeding with presentation of our estimator we briefly discuss other methods for estimating discrete-choice panel data models. Simulated Maximum Likelihood has in some sense been revived with the development of the GHK simulator. Borsch-Supan and Hajivassiliou (1992) show that use of GHK can reduce biases to reasonable levels in some small scale problems. In addition, post-estimation bias correction procedures have been suggested by Lee (1995). In our Monte-Carlo models that cannot fully utilize GHK, we find very significant biases in the SML estimator. We also find that while these bias-correction procedures do help a bit, they do not eliminate the biases in our longer panel models. Hajivassiliou and McFadden (1998) introduce the Method of Simulated Scores (MSS) for application to very general latent variable models (i.e. pure discrete choice models, truncated and censored models, etc.). This MSS methodology encompasses the SML estimator, but also includes two other alternatives for panel discrete choice models. The first of these other alternatives uses acceptance rejection techniques to simulate the score without bias (and result in estimates consistent for fixed NS), but is hard for longer panels and it is discontinuous in θ. The second method uses Gibbs sampling to simulate the score (which is unbiased for an infinite number of Gibbs resamplings), but depends on normality-type assumptions. Pure Bayesian methods have also been taken to panel discrete choice data in McCulloch and Rossi (1994) and Geweke, Keane, and Runkle (1998). These appear to work well, but are also tied down fairly heavily to assumptions of normality and linearity[25].

5. Our Simulated Sum Estimator

5.1. Naive Simulated Sum Estimators

Starting with McFadden's original moment:

$$g_{MSM}(\theta) = \sum_{s=1}^{J^T} w_s \left(d_s - \widehat{p}_s(\theta)\right) \tag{5.1}$$

recall that the problem with using smooth simulators for the $\widehat{p}_s(\theta)$'s is that unless T is very small, this sum will have too many elements to compute. Our basic idea is that one can simulate this unmanageable sum by summing random elements of the sum. In other words, we want to randomly draw N choice sequences, sum them, and multiply by an appropriate factor. Clearly it is very easy

[25]McCulloch and Rossi note that their Gibbs sampling routine can work with a random effect composed of a mixture of normals. On the other hand, actual linearity of $f(X,\theta)$ in X and θ seems important for the feasibility (speedwise) of their methodology.

to draw such random sequences. For example, in a binary choice model ($J = 2$), one can simply draw T-vectors where each element is equal to 1 with probability .5[26]. This results in the following simulated sum moment:

$$g_{SUM1}(\theta) = \frac{J^T}{N} \sum_{n=1}^{N} w_n \left(d_n - \widehat{p}_n(\theta)\right)$$

where n indexes N randomly chosen choice sequences. Since $g_{SUM1}(\theta)$ is an unbiased simulator of $g_{MSM}(\theta)$, i.e.

$$E_n g_{SUM1}(\theta) = J^T E_n w_n \left(d_n - \widehat{p}_n(\theta)\right) = J^T \sum_{s=1}^{J^T} w_s \left(d_s - \widehat{p}_s(\theta)\right) \frac{1}{J^T} = g_{MSM}(\theta)$$

it is also an unbiased simulator of $g(\theta)$, and thus has expectation 0 at θ_0. $g_{SUM1}(\theta)$ can therefore be used as a moment to estimate θ consistently for *any finite* number (N) of simulated sequences and *finite* number of simulation draws used to compute $\widehat{p}_n(\theta)$[27].

As is McFadden's pure frequency simulator, $g_{SUM1}(\theta)$ is feasible to compute, since it only involves summing N components, not J^T. On the other hand, if one keeps the random sequences the same over different θ and if one uses a smooth simulator $\widehat{p}_n(\theta)$, $g_{SUM1}(\theta)$ is *continuous* in θ, unlike McFadden's estimator. In addition, one only needs to compute N w_s's, and this only needs to be done once because (unlike McFadden) the sequences don't change over alternate θ. Thus $g_{SUM1}(\theta)$ solves the two computational major problems of the McFadden pure frequency simulator.

Unfortunately, for reasonable N, the estimator θ_{SUM1} that minimizes the sample average of $g_{SUM1}(\theta)$ will have unacceptable levels of simulation error. Essentially we are utilizing N randomly chosen moments out of J^T. As this choice is without regard to which are the most informative of these J^T moments, intuitively one would expect us to obtain $\frac{N}{J^T}$ of the information in the true moment. Note that the above procedure is essentially equivalent to arbitrarily choosing N of the sequence moments to consider, i.e. there is not even a reason to randomly choose them (this will not the case below). One can show (see Appendix 1) that even ignoring the variance due to the simulation of $\widehat{p}_n(\theta)$, the variance of $g_{SUM1}(\theta)$ is on the order of $(1+\frac{J^T}{N})\Phi$, which compared to the $(1+\frac{1}{N})\Phi$ of the alternative estimators is far too large for practical use[28].

We therefore consider the alternative moment,

$$g_{SUM2}(\theta) = \sum_{s=1}^{J^T} w_s d_s - \frac{J^T}{N} \sum_{n=1}^{N} w_n \widehat{p}_n(\theta)$$

which simply notes that $\sum_{s=1}^{J^T} w_s d_s$ has only one non-zero element (that corresponding to the

[26] This formulation places equal weight on the J^T possible sequences. Later we will draw sequences according to different weighting schemes.

[27] Consistency and asymptotic normality follow directly (given their regularity assumptions) from the proofs in McFadden and Ruud (1994). Note that conditional on the w_s, the simulation error, i.e. the difference between $g_{SUM1}(\theta)$ and $g(\theta)$, is bounded by $2 \cdot J^T \max_s \{w_s\}$ (element by element) and thus stochastically bounded.

[28] Again, recall that the variance of an MSM estimator of θ is proportional to the variance of $g_{MSM}(\theta)$.

observed choice sequence) and thus can be computed exactly. $g_{SUM2}(\theta)$ thus simulates only the second sum, that of the weights times the probabilities. As the simulated sum is an unbiased simulator of the true sum, i.e.

$$E_n\left[\frac{J^T}{N}\sum_{n=1}^{N} w_n\widehat{p}_n(\theta)\right] = \sum_{s=1}^{J^T} w_s\widehat{p}_s(\theta)$$

$g_{SUM2}(\theta)$ also has expectation 0 at θ_0[29] and can be used to consistently estimate θ.

We now consider how the variance of $g_{SUM2}(\theta)$ compares to that of $g_{MSM}(\theta)$ and $g(\theta)$. First we compute this variance *ignoring* simulation error in the probabilities, i.e. assume $\widehat{p}_s(\theta) = p_s(\theta)$. We return to this extra source of variance later. Note that the w_s's are vectors - we adopt the convention that $w_s^2 = w_s w_s^{'}$ (same for all other vectors).

$$\begin{aligned}
Var(g_{SUM2}(\theta)) &= Var\left(\sum_{s=1}^{J^T} w_s d_s\right) + Var\left(\frac{J^T}{N}\sum_{n=1}^{N} w_n p_n(\theta)\right) \\
&= \Phi + \frac{J^{2T}}{N^2} N Var(w_n p_n(\theta)) \\
&= \Phi + \frac{J^{2T}}{N}\left[\frac{1}{J^T}\sum_s w_s^2 p_s(\theta)^2 - \left(\frac{1}{J^T}\sum_s w_s p_s(\theta)\right)^2\right] \\
&= \Phi + \frac{1}{N}\left[\sum_s w_s^2 p_s(\theta)^2 J^T - \left(\sum_s w_s p_s(\theta)\right)^2\right]
\end{aligned}$$

Note that if $p_s(\theta)^2 J^T = p_s(\theta)$, this variance is asymptotically $(1+\frac{1}{N})\Phi$[30]. This would be the case if all choice sequences have equal probability of occurring (in the data generating process), i.e. $p_s(\theta) = \frac{1}{J^T}$. With unequal probabilities this variance is higher.

Examination of the last derivation suggests that one might reduce this variance by importance sampling sequences, i.e. choosing random sequences with probabilities other than $\frac{1}{J^T}$. One natural and very feasible importance sampling density is to select sequences according to the probabilities of those sequences occurring in the data, i.e. $p_s(\theta)$. This is extremely easy to do (at a particular θ) - simply take a draw from the joint distribution of unobservables and see what choice sequence

[29] Note that the elements of this sum have different expectations (unlike $g_{SUM1}(\theta)$ where the expectation of each element is zero). Thus, with $g_{SUM2}(\theta)$ there **is** a need for (pseudo) randomization, i.e. we cannot arbitrarily pick sequences to sum.

[30] Since

$$\begin{aligned}
\sum_{s=1}^{J^T} w_s^2 p_s(\theta_0) - \left[\sum_{s=1}^{J^T} w_s p_s(\theta_0)\right]^2 &= E\left[\left(\sum_{s=1}^{J^T} w_s d_s\right)^2\right] - E\left[\left(\sum_{s=1}^{J^T} w_s d_s\right)\right]^2 \\
&= Var\left(\sum_{s=1}^{J^T} w_s d_s\right)
\end{aligned}$$

these unobservables generate at θ. As the importance sampled draws need to be weighted by their inverse probabilities (assume that we can compute these probabilities for the moment) we obtain the moment

$$g_{SUM3}(\theta) = \sum_{s=1}^{J^T} w_s d_s - \frac{1}{N}\sum_{n=1}^{N} \frac{w_n p_n(\theta)}{p_n(\theta)} = \sum_{s=1}^{J^T} w_s d_s - \frac{1}{N}\sum_{n=1}^{N} w_n \tag{5.2}$$

where

$$[p_n(\theta) = p_s(\theta), w_n = w_s] \text{ with probability } p_s(\theta)$$

In this case

$$\begin{aligned} Var(g_{SUM3}(\theta)) &= \Phi + Var\left(\frac{1}{N}\sum_{n=1}^{N} w_n\right) \\ &= \Phi + \frac{1}{N} Var(w_n) \\ &= \Phi + \frac{1}{N}\left[\sum_s w_s^2 p_s(\theta) - \left(\sum_s w_s p_s(\theta)\right)^2\right] \\ &= (1+\frac{1}{N})\Phi \qquad \text{at } \theta_0 \end{aligned}$$

so in theory this estimator achieves the McFadden pure frequency variance of $(1+\frac{1}{N})\Phi$ regardless of what the $p_s(\theta)$'s are[31,32].

Unfortunately, there are three caveats regarding $g_{SUM3}(\theta)$. First, we need to hold the simulated sequences constant over alternative values of θ so that 1) our moment condition is continuous in θ, and so 2) we can avoid computing new w_n's at different θ. Thus, in practice, we need to draw the simulated sequences at some initial value of θ, say θ'. This results in

$$g_{SUM3}(\theta) = \sum_{s=1}^{J^T} w_s d_s - \frac{1}{N}\sum_{n=1}^{N} \frac{w_n p_n(\theta)}{p_n(\theta')}$$

so the p's in the numerator and denominator do not exactly cancel out unless $\theta = \theta'$. As such, this $(1+\frac{1}{N})\Phi$ result is dependent on the initial values at which sequences are drawn[33]. Second, if we cannot compute $p_n(\theta')$ exactly, simulation error in the denominator will prevent consistency for fixed number of simulation draws. We solve this issue later. Lastly, we have ignored the extra variance in $g_{SUM3}(\theta)$ due to simulating the $p_n(\theta)$'s. We consider this source of variance next.

To evaluate this additional variance, we consider a generic partially analytic simulator of sequence probabilities. As in section 2, we divide the unobservables $(\epsilon_1, ..., \epsilon_T)$ into two components,

[31] This should not be surprising because g_{SUM3} essentially *is* g_{PF} (at the θ at which sequences are drawn). As we shall see below, the difference is away from this θ - while g_{PF} changes the sequences, we hold the sequences constant and change the p's.

[32] There may be better ways to importance sample these sequences (in particular based on the weights w_s), but we feel that these would be too time-intensive to make for an efficient procedure. Simulating based on just p is both easy and very quick.

[33] As such, we pay particular attention to starting θ' in our Monte-Carlo work

μ and η, the difference between the two being that the integral in the choice sequence probability is analytically computable *conditional* on μ, i.e.

$$p_s(\theta) = p(y_{i1}, ..., y_{iT} \mid X_{i1}, ..., X_{iT}; \theta) = \int p_s(\mu; \theta) f(\mu)$$

Note that this formulation is generic enough to include not only the common PA simulators mentioned in section 2, but also the PF simulator (when μ contains all the unobservables) and the GHK simulator (let μ be the vector of uniform random variables used in the recursive simulation - conditional on μ one can compute analytically the probability of each choice sequence). We can now write our simulated probabilities as just a function of the random draws μ, i.e.

$$\widehat{p}_s(\mu_{NS}; \theta) = p_s(\theta) + \delta_s(\mu_{NS}; \theta)$$

where μ_{NS} contains NS draws of μ and where $\delta(\mu_{NS}; \theta)$ is a simulation error term that has expectation 0 given the use of any of the unbiased probability simulators mentioned above.

Now suppose that we still draw simulated sequences with the probabilities $p_j(\theta)$ and that we can somehow exactly compute the appropriate weight $p_j(\theta)$ to put in the denominator. Note that we are again ignoring the fact that we need to draw simulated sequences according to some initial $\theta = \theta'$. We now have:

$$g_{SUM4}(\theta) = \sum_{s=1}^{J^T} w_s d_s - \frac{1}{N} \sum_{n=1}^{N} \frac{w_n \widehat{p}_n(\mu_{NS}; \theta)}{p_n(\theta)}$$

There are three sources of variance in $g_{SUM4}(\theta)$ - that due to d_s, μ_{NS}, and the simulated sequences n. Appendix 2 shows that:

$$\begin{aligned} Var(g_{SUM4}(\theta)) &= \Phi + Var\left(\frac{1}{N} \sum_{n=1}^{N} \frac{w_n \widehat{p}_n(\mu_{NS}; \theta)}{p_n(\theta)}\right) \\ &= \Phi + Var_\mu\left(\sum_s w_s \widehat{p}_s(\mu_{NS}; \theta)\right) + \frac{1}{N}\left[\sum_s w_s^2 p_s(\theta) - \left(\sum_s w_s p_s(\theta)\right)^2\right] \\ &\quad + \frac{1}{N}\left[\sum_s \frac{w_s^2 E\delta_s(\mu_{NS}; \theta)^2}{p_s(\theta)}\right] - \frac{1}{N} E\left[\left(\sum_s w_s \delta_s(\mu_{NS}; \theta)\right)^2\right] \end{aligned}$$

There are five terms in this expression. It is useful to evaluate these terms by hypothetically considering a pure frequency simulator (e.g. all the unobservables are in μ). In this case the second term is asymptotically $\frac{1}{NS}\Phi$. The third term is asymptotically $\frac{1}{N}\Phi$. Essentially these two terms respectively would correspond to the extra variance induced by either only simulating the probabilities or only simulating the sum. The last two terms arise from the interaction between the two simulation processes and are problematic. The last term is small so it doesn't help us. On the other hand, the fourth term is big. Note that it depends on variance of the simulation error.

In the pure frequency case

$$E\delta_s(\mu_{NS};\theta)^2 = \frac{p_s(\theta)(1-p_s(\theta))}{NS}$$

so that this term

$$\frac{1}{N}\left[\sum_s \frac{w_s^2 E\delta_s(\mu_{NS};\theta)^2}{p_s(\theta)}\right] = \frac{1}{N\cdot NS}\left[\sum_s w_s^2(1-p_s(\theta))\right] \approx \frac{1}{N\cdot NS}\left[\sum_s w_s^2\right]$$

which unfortunately is at least order $\frac{J^T}{N}\Phi$.

What is going on here is that the two simulation errors are interacting with each other. As a simple example, suppose both $N = 1$ and $NS = 1$ with a pure frequency simulator (and for simplicity assume that $p_s = \frac{1}{J^T} \ \ \forall s$). We are therefore simulating $\sum_{s=1}^{J^T} w_s p_s$ with $J^T w_n \widehat{p}_n$. With the PF simulator, $\widehat{p}_n = 1$ with probability $\frac{1}{J^T}$ and 0 otherwise. Therefore $J^T w_n \widehat{p}_n$ will $= 0$ most of the time (with probability $1 - \frac{1}{J^T}$) and $= J^T w_n$ otherwise. With smooth simulators such as PA or GHK, these simulation variances decrease, but some brief experiments suggested that this decrease is not to acceptable levels.

5.2. Our Preferred Simulated Sum Estimator

The key to obtaining a simulated sum estimator with reasonable simulation error is to purposely correlate the simulation error in the simulated sequences with that of the simulated sequence probabilities. Intuitively, in the above pure frequency example, we would want to make sure that the sequence that gets positive probability is also the drawn simulated sequence. In the case of smooth simulators, the way we do this is by using the *exact same* random draws μ_{NS} (recall that μ_{NS} is a set of NS random draws on the "non-analytic" unobservables μ) to *both* draw the simulated sequences *and* to compute simulated probabilities.

In particular, first draw the set μ_{NS}. Note that this set of draws implies a distribution over possible sequences, $\{\widehat{p}_s(\mu_{NS};\theta')\}_{s=1}^{J^T}$. Now importance sample (draw) N sequences according to this exact distribution. This is very simple to do: First, one can randomly select a particular draw μ_n from the set of draws μ_{NS}[34] - Secondly, one can take a draw from the distribution of the "analytic" unobservables, η_n. The combined (μ_n, η_n) (at θ') will generate a such a choice sequence.

Importantly, since we are now importance sampling sequences from a *known* distribution we have simultaneously solved one of the above caveats. We now know *exactly* the appropriate weight to give each importance sampling draw, i.e. $\frac{1}{\widehat{p}_s(\mu_{NS};\theta)}$. This means that there is no denominator bias in using these importance sampling weights, and that $g_{SUM}(\theta)$ as defined by:

$$g_{SUM}(\theta) \ = \ \sum_{s=1}^{J^T} w_s d_s - \frac{1}{N}\sum_{n=1}^{N} \frac{w_n \widehat{p}_n(\mu_{NS};\theta)}{\widehat{p}_n(\mu_{NS};\theta')} \tag{5.3}$$

[34]One can actually do slightly better by selecting these draws deterministically, see below.

$$= \sum_{s=1}^{J^T} w_s d_s - \frac{1}{N}\sum_{n=1}^{N} w_n \qquad \text{at } \theta = \theta'$$

and is an unbiased simulator of the exact moment $g_{GMM}(\theta)$ for any fixed N and NS[35]. Thus, $\widehat{\theta}_{SUM}$ that minimizes $\|\sum_i g_{SUMi}(\theta)\|$ should be consistent as I goes to infinity for fixed N and NS.

From the second line of (5.3), note that at $\theta = \theta'$ the variance of $g_{SUM}(\theta)$ is equivalent to the variance of McFadden's pure frequency MSM estimator, i.e. $(1 + \frac{1}{N})\Phi$. In fact, at $\theta = \theta'$, the estimators are identical. The estimators differ away from θ'. As θ moves away from θ', McFadden's estimator changes sequences (which is what creates discontinuities and requires computing instruments for many sequences). In contrast, our SUM estimator holds the sequences constant but uses smooth simulators to change the sequence weights $\frac{\widehat{p}_n(\mu_{NS};\theta)}{\widehat{p}_n(\mu_{NS};\theta')}$. This is why we interpret our estimator as a smoothed version of McFadden's estimator. Although the $(1 + \frac{1}{N})\Phi$ variance result disappears away from $\theta = \theta'$ [36], there is hope that either reasonable starting values or iterating estimation can come close to this ideal . We pay particular attention to initial θ' in our Monte Carlo study.

5.3. Operation of our Simulated Sum Estimator

Before proceeding to the Monte-Carlo results, we briefly summarize the operationalization of our SUM estimator. We begin with the case where $NS = N$.

1) Draw NS sets of unobservables $((\mu_1, \eta_1), ..., (\mu_{ns}, \eta_{ns}))$. Store the choice sequences $(s_1, ..., s_{ns})$ generated by these unobservables at some initial θ' and the simulated probabilities of these sequences conditional on the same $\mu_{NS} = (\mu_1, ..., \mu_{ns})$, i.e. $\widehat{p}_n(\mu_{NS};\theta')$[37].

2) Using different random draws (NSW), simulate and store optimal weights w_s for both the observed choice sequence and the NS simulated sequences.

3) To evaluate the moment condition for a particular θ, compute $\widehat{p}_n(\mu_{NS};\theta)$ for each of the N sequences and compute (5.3).

---

[35] Formally,

$$\begin{aligned} E_{\mu,n}\left[\tfrac{1}{N}\textstyle\sum_{n=1}^{N} \frac{w_n \widehat{p}_n(\mu_{NS};\theta)}{\widehat{p}_n(\mu_{NS};\theta')}\right] &= E_\mu E_{n|\mu}\left[\tfrac{1}{N}\textstyle\sum_{n=1}^{N} \frac{w_n \widehat{p}_n(\mu_{NS};\theta)}{\widehat{p}_n(\mu_{NS};\theta')}\right] \\ &= E_\mu E_{n|\mu}\left[\frac{w_n \widehat{p}_n(\mu_{NS};\theta)}{\widehat{p}_n(\mu_{NS};\theta')}\right] \\ &= E_\mu\left[\textstyle\sum_s \frac{w_s p_s(\mu_{NS};\theta)}{p_s(\mu_{NS};\theta')}\widehat{p}_s(\mu_{NS};\theta')\right] \\ &= E_\mu\left[\textstyle\sum_s w_s \widehat{p}_s(\mu_{NS};\theta)\right] \\ &= \textstyle\sum_s w_s p_s(\theta) \end{aligned}$$

and the simulated moment is unbiased.

[36] For two reasons - unlike McFadden's estimator for only one.

[37] Note that this procedure does not literally draw randomly from the set μ_{NS}. We instead are implicitly drawing randomly *without* replacement. This is not only easier than actually drawing randomly, but also performs slightly better for obvious reasons.

Note that besides the straightforward drawing of the simulated sequences and the weights, the SUM estimator is essentially equivalent in computational complexity to SML. One simply needs to simulate probabilities of N sequences rather than 1 observed sequence. For cases when $NS > N$, one can just use the first N μ_{ns}'s to draw simulated sequences[38]. If $N > NS$, one can cycle though the μ_{ns}'s multiple times to draw the simulated sequences (e.g. if $N = 2 * NS$, one can use each μ_{ns} to draw two simulated sequences.)

6. Monte-Carlo Results

This section contains three sets of Monte-Carlo results. The first are a partial replication of the short ($T = 8$) panel probit models of Keane (1994) in which GHK was used to simulate transition probabilities. We find that while our SUM estimator using a PA simulator sacrifices some efficiency relative to the SML and TP estimators using GHK, it performs reasonably well. The second and third experiments are intended to emulate the non-normal and non-linear cases above where GHK is not effective. We find that in longer panels ($T = 24$) our SUM estimator dominates the SML, TP, and Keane's (1994) Consistent Transition Probability (CTP) estimators in terms of mean squared error. We lastly note that a natural way to compare estimators with different levels of efficiency and bias is with root MSE. However, if one has any interest in hypothesis testing (rather than just point estimation) this metric undervalues consistent estimates. In other words, even if our SUM estimator had done worse in MSE than the inconsistent and biased SML and TP estimators, we would value it for its consistency and apparent lack of bias.

6.1. Computational Issues

We first describe some of the issues with the operationalization of each of the estimators. All are programmed in equally highly optimized C code[39] (see appendix for programs). The SML estimates are fairly straightforward - we simply need to set the number of simulation draws used in computing the sequence probabilities. Starting parameters are irrelevant except for the starting point of the search procedure. After estimation, we use the bias correction procedure detailed in Lee (1995).

For the three MSM estimators, there are two additional issues. First, starting parameters are important. In all three MSM estimators, one needs starting parameters for approximating the optimal weight matrices. For the SUM estimator, starting parameters are also important for the generation of simulated sequences. Unless otherwise noted, we use the SML estimates as the starting parameters. The median times reported for the MSM estimators **do not include** the time used to compute the initial SML estimate, so this needs to be added in for comparison purposes.

Second, to gain efficiency it is helpful to use lots of draws for computation of the weight matrices. In the TP estimator, we typically chose NSW, the number of simulation draws for the weight matrix

[38] Since the μ_{ns}'s are in no particular order, this is equivalent to random sampling without replacement.

[39] Experiments 1 and 2 were run on an UltraSparc 143 Mhz chip. Experiment 3 was run on a PII - 400 Mhz chip.

to be 10 times NS, the number of simulation draws used in simulating the transition probabilities[40].

For the CTP estimator, there is both a numerator $(\frac{\partial\left(\frac{p_{s^{t-1},j}(\theta)}{p_{s^{t-1}}(\theta)}\right)}{\partial\theta})$ and denominator $(p_{s^{t-1}}(\theta))$ of the weights - these both need to be simulated. In some brief experimentation, we found that 1) it was more efficient to use separate simulation draws for the numerator and denominator, and 2) it was helpful to significantly boost the number of simulation draws used for the denominator ($NSW2$) relative to those used for the numerator (NSW).

For our SUM estimator, there are also three choices - NS, NSW, and N - the number of simulated sequences. Conforming to our result that around $\theta = \theta'$ the variance depends on N and not NS, we typically set $NS = 1$ and chose N to satisfy a given computational time. In some experimentation, we did find that one could often do better by decreasing N and increasing NS. We typically chose NSW to be 10 times N[41].

We note that these relationships (e.g. for the TP estimator the choice between NS and NSW; for the CTP estimator the choice between NS, NSW, and $NSW2$; and for the SUM estimator the choice between NS, N, and NSW; all to satisfy a given computation time) were not optimized in any formal way. Doing so would be unfairly biasing conclusions towards the MSM estimators, as an advantage of SML is its lack of multidimensional "control" parameters that need to be decided upon[42].

6.2. Experiment 1

We first briefly replicate one of the Monte-Carlo models in Keane (1994). In these models we allow SML and the TP estimators to use the GHK simulator. The point is to see how well our SUM estimator (using only a PA simulator) fares. Our hypothesis is that we will lose some efficiency relative to SML and TP using GHK.

This model has the following setup: $T = 8$, the single covariate X_{it} has mean 6, within (individual) variance 2, and across individual variance 3. The normal error structure is AR(1) plus random effects. In the model which we replicate the persistence of the AR(1) process is $\rho = .6$ and its marginal variance is .8. The variance of the random effect is .2.

Table 2 presents results from three estimation approaches on 1000 simulated datasets. The first is Simulated Maximum Likelihood (SML) using 10 draws of the GHK simulator. The second is the TP estimator also using 10 GHK draws. The third is our SUM estimator with $N = 20$ and $NS = 1$, i.e. we draw 20 simulated sequences and use 1 simulation draw of the PA simulator to compute

[40] We used one sided approximations to the derivatives in the weight matrix. We didn't find much difference between approximating $\frac{\partial \ln p}{\partial \theta}$ or approximating $\frac{\partial p}{\partial \theta}$ and dividing by p.

[41] Another issue with the SUM estimator is the possbility of a shortcut in computing the probabilities of multiple sequences given the same simulation draws on μ's. In the normal, linear, case (where GHK is applicable) with no lagged dependent variables, one only needs to compute T CDF's, not $N * T$, as one would need more generally. We only use this shortcut once in our experiments, in Experiment 1 where we also allow use of full GHK for the SML and TP estimators. The idea is that generally, when GHK is available, the shortcut will also be - when GHK is not available (due to non-linearities or non-normality), the shortcut will also generally not be.

[42] There is an additional control variable in the MSM estimators which is the time allocated to the SML estimation of intial parameters.

the probability of each sequence (for all estimators we use different draws across observations). Timewise, SML took about 10 seconds, while the TP and SUM estimators took approximately 20 seconds to estimate (including computation of the weight matrix (at the SML estimates)). There are two reasons why the SML was considerably faster than the TP estimator for the same number of simulation draws. First, there is no weight matrix to compute. Second, the objective function tended to be better behaved, lessening the number of function evaluations necessary for optimization[43].

The SML and TP results in table 2 are essentially identical to the results in Keane. While the SML estimator exhibits fairly large biases, the TP estimator (as well as the SUM estimator) exhibits almost none. On the other hand, increasing the SML draws eliminated these biases fairly quickly (as did applying Lee's bias correction procedure). We are more concerned with the relative efficiency of our SUM estimator. As evidenced by the table, standard deviations of the distribution of the estimated coefficients are clearly higher than those of the other 2 - usually by around 20% . As we expected, this suggests that while our SUM estimator performs reasonably, it clearly gives up some efficiency to the GHK based procedures when full GHK is available.[44]

6.3. Experiment 2

Our second experiment expands the Keane model in two ways. First, we triple the length of the panel to 24 periods. This significantly lowers sequence probabilities and increases simulation error for all simulators. Second, we do not allow use of the GHK simulator. Note that even though we keep (for lack of anything obvious to switch to) the linear, normally distributed framework where GHK actually can be applied , our intention is to emulate either a non-linear or non-normal environment where GHK would not be effective. We therefore use the PA simulator to simulate all sequence probabilities (and derivatives).

Table 3a presents SML estimates with bias correction. Due to the longer panel and the fact that we can no longer use GHK, we boost our number of simulation draws to 100. As can be seen, there are very considerable biases in all coefficients except for β_x. The next three parts of the table analyze the 3 MSM estimators. The TP estimator also has very significant biases, but slightly less than those of the SML estimator. Interestingly, many of them are the opposite direction of the biases of the SML estimator. As expected, the CTP estimates exhibit very little bias. However, there is a huge efficiency loss. Standard deviations are 5-8 times larger than those of the SML estimator.

Lastly, our SUM estimates in Table 3d also exhibit essentially no bias. However, it is far more

[43] We used the Nelder-Mead Simplex routine for optimization.

[44] A couple of other notes. We also tried our SUM estimator starting at arbitrary parameters (e.g. .1,.1,.1,.1) - then iterating the estimation process 2 or 3 times. This led to similar standard errors, although the TP estimator degraded less (in response to bad starting values) for the initial iteration. We also increased T and obtained similar results - if anything the TP estimator performed relatively better. On the other hand, the fact that we programmed these routines in C has implications on these comparisons. Compiled languages such as Gauss and Matlab are inefficient at loops, which are a neccesity for the recursive GHK simulator. Since the PA simulator can easily programmed without loops, our SUM estimator might perform a bit better for someone constrained to these languages.

efficient than the CTP estimator, with standard deviations 2-6 times smaller. Although the S.D.'s of the SUM estimates tend to be about twice those of SML, the lack of bias makes up for it in terms of root-MSE, particularly in the parameters of the error process. Again, we feel that root-MSE undervalues consistency because of the benefits of hypothesis testing.

Given that the MSM estimators incur extra time (~455 sec) for the computation of the SML estimates as starting parameters, we boost the SML simulation draws to 300 in Table 4. This model now takes approximately 20 minutes per run. Note that the biases decrease, but not by that much. ρ is underestimated by 30%, the variance of the random effect is overestimated by 25%, and the first off diagonal element of the variance matrix is underestimated by 20%. Root-MSEs for the error parameters still don't come close to those of the SUM estimates in Table 3d, even though total estimation time is now considerably longer (1177sec > (365sec + 455sec)).

The last three parts of Table 4 increase the number of simulation draws for the MSM estimators. Importantly, these use as starting values the SML estimates from Table 3, not Table 4, so we again need to add 455sec to computation times. All SD's decrease and the biases of the TP estimator go down a bit, but the SUM estimator still clearly dominates in terms of MSE.

6.4. Experiment 3

Our third set of experiments examines a random coefficients model. We keep $T = 24$, and to focus on the effects of the random coefficients we also set $\rho = 0$, resulting in the model:

$$u_{it} = \beta_i x_{it} + \epsilon_i^1 + \epsilon_{it}^2$$

where x_{it} is distributed identically to the above, ϵ_i^1 is a normal random effect with variance 0.25. ϵ_{it}^2 is now completely i.i.d. error. Rather than add explanatory variables to introduce more random coefficients/effects, a simple way to increase the dimensionality of the integrals is to allow different random coefficients in different sets of time periods. We simply divide the time frame into 6 sets of 4 periods and assume a different β_{ir} for each set r of periods (distributed independently with mean β_x =0.25 and standard deviation $\sigma_r = 0.25$). This is also interesting as we can see if there are any differential biases according to time period. It also may in some sense imitate learning models where consumers are hit with particular shocks after particular time periods (e.g. learning shocks after consumption experiences). Again, note that full GHK actually could be applied to this literal model[45] but the goal is to emulate models with β_i's either non-normal or entering non-linearly due to dynamics. Thus, we again restrict attention to the PA simulator where the integrals over $(\beta_{i1}, ..., \beta_{i6}, \epsilon_i^1)$ are directly simulated and the integrals over the ϵ_{it}^2 are analytically computed.

Results are presented in Table 5. Bias-corrected SML estimates with 75 draws exhibit large biases in all the parameters. The TP estimator also exhibits large biases in all the parameters. Interestingly, the biases on the random coefficient standard deviation increase as one goes from σ_1 to σ_6. This results from the fact that the simulated transition probabilities have more bias in them

[45] Although it would require different Cholesky decompositions for different individuals.

as time increases (since the component probabilities are smaller and harder to simulate). This differential bias might be a matter of large concern, e.g. if one is estimating the trying to estimate the tail of a learning process (which only occurs far ahead in time).

The CTP estimates again appear relatively unbiased, but sacrifice efficiency to the SML and TP estimators (though the sacrifice is less than that in Experiment 2). Lastly our SUM estimates also appear unbiased and dominate the CTP estimates by 1.5 to 2.5 times in terms of efficiency. The root-MSE comparison to SML and TP again falls in favor of the SUM estimator.

Lastly, we again boosted the SML simulation draws. Results are given in Table 6a. Biases decrease, but are still very significant. Comparing the SML results in 6a (total time = 2786 seconds) vs. the SUM results in Table 5d (total time = 735+822=1557 sec), we see that the SUM estimates have considerably less MSE in almost half the time.

7. Conclusions

8. Appendices

8.1. Appendix 1

The minor (but annoying) complication here is that the elements in the sum of the true moment are correlated (if $d_j = 1$, the other d's must $= 0$). Formally, letting $w_j(d_j - p_j) = x_j$, and noting that there are two sources of variance: the variance due to the realizations of x_j and variance due to the random draws n (again, in practice there is a third source of variance - that due to simulated p_j's which we ignore at the moment):

$$\begin{aligned}
Var(g_{SUM1}(\theta)) &= Var(\frac{J^T}{N}\sum_n x_n) \\
&= \frac{J^{2T}}{N^2}\left[Var_x E_{n|x}\left[\sum_n x_n\right] + E_x Var_{n|x}\left[\sum_n x_n\right]\right] \\
&= \frac{J^{2T}}{N^2}\left[Var_x\left[NE_{n|x}x_n\right] + E_x\left[NVar_{n|x}x_n\right]\right] \\
&= \frac{J^{2T}}{N^2}\left[Var_x\left[N\frac{1}{J^T}\sum_j x_j\right] + E_x\left[NVar_{n|x}x_n\right]\right] \\
&= \Phi + \frac{J^{2T}}{N}E_x\left[E_{n|x}[x_n^2] - \left[E_{n|x}x_n\right]^2\right] \\
&= \Phi + \frac{J^{2T}}{N}E_x\left[\frac{1}{J^T}\sum_j x_j^2 - \left[\frac{1}{J^T}\sum_j x_j\right]^2\right] \\
&= \Phi + \frac{J^T}{N}\sum_j E_x x_j^2 - \frac{1}{N}Var(\sum_j x_j) \\
&= \left(1 - \frac{1}{N}\right)\Phi + \frac{J^T}{N}\sum_j E_x x_j^2 \\
&\approx \left(1 + \frac{J^T - 1}{N}\right)\Phi
\end{aligned}$$

Line 2 follows from the Law of Iterated Variance. Line 3 because the randomly drawn sequences are independent. Line 5 as $\Phi = Var(\sum_j x_j)$. Line 7 because $E[x_j] = E\left[\sum_j x_j\right] = 0$. The last line is approximate since the variance of the sum (Φ) is not equal to the sum of the variances due to correlation but this correlation is of order (The example here is with $J = 2, T = 1$ - clearly one only needs to choose one of the moment conditions)

8.2. Appendix 2

$$
\begin{aligned}
Var(g_{SUM4}(\theta)) &= \Phi + Var\left(\frac{1}{N}\sum_{n=1}^{N}\frac{w_n\widehat{p}_n(\mu_{NS};\theta)}{p_n(\theta)}\right) \\
&= \Phi + Var_\mu E_{n|\mu}\left(\frac{1}{N}\sum_{n=1}^{N}\frac{w_n\widehat{p}_n(\mu_{NS};\theta)}{p_n(\theta)}\right) + E_\mu Var_{n|\mu_{NS}}\left(\frac{1}{N}\sum_{n=1}^{N}\frac{w_n\widehat{p}_n(\mu_{NS};\theta)}{p_n(\theta)}\right) \\
&= \Phi + Var_\mu\left(\sum_s \frac{w_s\widehat{p}_s(\mu_{NS};\theta)}{p_s(\theta)}p_s(\theta)\right) + \frac{1}{N}E_\mu Var_{n|\mu_{NS}}\left(\frac{w_n\widehat{p}_n(\mu_{NS};\theta)}{p_n(\theta)}\right) \\
&= \Phi + Var_{\mu_{NS}}\left(\sum_s w_s\widehat{p}_s(\mu;\theta)\right) + \frac{1}{N}E_\mu E_{n|\mu}\left[\left(\frac{w_n\widehat{p}_n(\mu;\theta)}{p_n(\theta)}\right)^2 - E_{n|\mu}\left(\frac{w_n\widehat{p}_n(\mu;\theta)}{p_n(\theta)}\right)^2\right] \\
&= \Phi + Var_\mu\left(\sum_s w_s\widehat{p}_s(\mu;\theta)\right) + \frac{1}{N}E_\mu\left[\sum_s\left(\frac{w_s\widehat{p}_s(\mu;\theta)}{p_s(\theta)}\right)^2 p_s(\theta) - \left(\sum_s w_s\widehat{p}_s(\mu;\theta)\right)^2\right] \\
&= \Phi + Var_\mu\left(\sum_s w_s\widehat{p}_s(\mu;\theta)\right) \\
&\qquad + \frac{1}{N}E_\mu\left[\sum_s\frac{w_s^2\widehat{p}_s(\mu;\theta)^2}{p_s(\theta)} - \sum_s w_s^2\widehat{p}_s(\mu;\theta)^2 - \sum_s\sum_{t\neq s} w_s w_t\widehat{p}_s(\mu;\theta)\widehat{p}_t(\mu;\theta)\right] \\
&= \Phi + Var_\mu\left(\sum_s w_s\widehat{p}_s(\mu;\theta)\right) \\
&\qquad + \frac{1}{N}E_\mu\left[\sum_s\frac{w_s^2\widehat{p}_s(\mu;\theta)^2}{p_s(\theta)} - \sum_s w_s^2\widehat{p}_s(\mu;\theta)^2 - \sum_s\sum_{t\neq s} w_s w_t\widehat{p}_s(\mu;\theta)\widehat{p}_t(\mu;\theta)\right] \\
&= \Phi + Var_\mu\left(\sum_s w_s\widehat{p}_s(\mu;\theta)\right) \\
&\qquad + \frac{1}{N}\left[\sum_s\frac{w_s^2 p_s(\theta)^2}{p_s(\theta)} - \sum_s w_s^2 p_s(\theta)^2 - \sum_s\sum_{t\neq s} w_s w_t p_s(\theta)p_t(\theta)\right] \\
&\quad + \frac{1}{N}\left[\sum_s\frac{w_s^2 E\delta_s(\mu;\theta)^2}{p_s(\theta)} - \sum_s w_s^2 E\delta_s(\mu;\theta)^2 - \sum_s\sum_{t\neq s} w_s w_t E\delta_s(\mu;\theta)\delta_s(\mu;\theta)\right] \\
&= \Phi + Var_\mu\left(\sum_s w_s\widehat{p}_s(\mu;\theta)\right) + \frac{1}{N}\left[\sum_s w_s^2 p_s(\theta) - \left(\sum_s w_s p_s(\theta)\right)^2\right] \\
&\quad + \frac{1}{N}\left[\sum_s\frac{w_s^2 E\delta_s(\mu;\theta)^2}{p_s(\theta)} - E\left[\left(\sum_s w_s\delta_s(\mu;\theta)\right)^2\right]\right]
\end{aligned}
$$

Table 1 - Comparison of Various Sequence Probability Simulators

	Baseline Parameters -T=8, I=500,ρ=0.6,σ^2_{RE}=.33,σ^2_{AR1}=.33,σ^2_{IID}=.33						
Perturbation	Full GHK	PA		Partial GHK 1		Partial GHK 2	
	Var	Var	Rel. Var.	Var	Rel Var	Var	Rel Var
Base Parameters	4.85	63.98	13.17	162.83	33.52	168.46	34.68
ρ=0.9	6.86	32.79	4.77	566.93	82.56	617.11	89.87
ρ=0.3	3.32	78.05	23.47	119.82	36.04	122.12	36.73
$\sigma^2_{RE=.10}$, $\sigma^2_{AR1=.45}$, $\sigma^2_{IID=.45}$	2.30	44.57	19.36	135.63	58.91	168.41	73.15
$\sigma^2_{RE=.45}$, $\sigma^2_{AR1=.10}$, $\sigma^2_{IID=.45}$	4.81	24.99	5.18	898.44	186.48	981.17	203.65
$\sigma^2_{RE=.45}$, $\sigma^2_{AR1=.45}$, $\sigma^2_{IID=.10}$	8.33	226.80	27.22	50.09	6.01	59.09	7.09
$\sigma^2_{RE=.80}$, $\sigma^2_{AR1=.10}$, $\sigma^2_{IID=.10}$	9.86	64.84	6.57	194.45	19.71	161.62	16.38
$\sigma^2_{RE=.10}$, $\sigma^2_{AR1=.80}$, $\sigma^2_{IID=.10}$	6.27	403.02	64.22	22.52	3.58	41.15	6.55
$\sigma^2_{RE=.10}$, $\sigma^2_{AR1=.10}$, $\sigma^2_{IID=.80}$	0.66	7.08	10.71	1478.07	2236.16	1531.45	2316.91
T=20	13.59	339.78	24.98	1281.61	94.25	1258.29	92.53
T=50	45.85	2503.84	54.60	5913.31	128.96	5979.05	130.40
T=50, ρ=0.9	66.97	1624.36	24.25	34006.48	507.72	32070.26	478.81
T=100	98.80	7802.21	78.96	16814.77	170.17	17544.78	177.56
T=100, ρ=0.9	219.05	6861.51	31.32	95850.80	437.57	112838.10	515.11

Table 2 - Replication of Experiment in Keane

Table 2a- Simulated Maximum Likelihood					
I=500, T=8, NS=10, Median Time = 10 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.9018	0.0878	0.878	-0.68
β_x	0.25	0.2500	0.0140	0.0140	0.20
σ^2_{RE}	0.20	0.2175	0.0567	0.0594	9.77
ρ	0.60	0.5400	0.0427	0.0736	-44.29
Table 2b - Transition Probability					
I=500, T=8, NS=10, NSW=50, Median Time = 20 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.8946	0.0877	0.0878	1.92
β_x	0.25	0.2492	0.0140	0.0140	-1.64
σ^2_{RE}	0.20	0.1935	0.0660	0.0663	-3.06
ρ	0.60	0.6047	0.0437	0.0440	3.45
Table 2c - Simulated Sum					
I=500, T=8, N=20, NS=1, NSW=200, Median Time = 22 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.8997	0.1077	0.1076	0.09
β_x	0.25	0.2499	0.0173	0.0173	-0.07
σ^2_{RE}	0.20	0.1897	0.0849	0.0855	-3.81
ρ	0.60	0.6013	0.0527	0.0527	0.82

Notes: For all models, 1000 replications were used. SML estimates used for starting parameters for the TP and SUM models.

Table 3 - Basic Model

Table 3a- SML with Bias Correction					
I=1000, T=24, NS=100, Median Time = 455 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.9577	0.0417	0.0711	-13.8431
β_x	0.25	0.2491	0.0065	0.0065	-1.2419
σ^2_{RE}	0.20	0.2841	0.0216	0.0868	38.8296
ρ	0.60	0.2722	0.0168	0.3281	-194.7793
$E[\epsilon_t \epsilon_{t-1}]$	0.68	0.4791	0.0154	0.2014	-129.9734

Table 3b - Transition Probability					
I=1000, T=24, NS=50, NSW=500, Median Time = 436 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.8419	0.0469	0.0744	12.3604
β_x	0.25	0.2356	0.0064	0.0156	-22.3786
σ^2_{RE}	0.20	0.1222	0.0464	0.0904	-16.7243
ρ	0.60	0.7455	0.0260	0.1478	55.8107
$E[\epsilon_t \epsilon_{t-1}]$	0.68	0.7776	0.0151	0.0988	64.4991

Table 3c - Keane's Consistent Transition Probability					
I=1000, T=24, NS=50, NSW=500, NSW2=1000, Median Time = 530 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.9718	0.3394	0.3453	-2.1152
β_x	0.25	0.2593	0.0470	0.0477	1.9860
σ^2_{RE}	0.20	0.2057	0.1033	0.1030	0.5583
ρ	0.60	0.5612	0.1140	0.1199	-3.3941
$E[\epsilon_t \epsilon_{t-1}]$	0.68	0.6552	0.0824	0.0857	-3.0059

Table 3d - Simulated Sum					
I=1000, T=24, N=20, NS=1, NSW=200, Median Time = 365 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.9041	0.0761	0.0759	-0.5401
β_x	0.25	0.2487	0.0108	0.0109	-1.1843
σ^2_{RE}	0.20	0.1945	0.0422	0.0423	-1.2954
ρ	0.60	0.6110	0.0388	0.0401	2.8461
$E[\epsilon_t \epsilon_{t-1}]$	0.68	0.6877	0.0250	0.0261	3.0782

Notes: For all models, 100 replications were used. SML estimates used for starting parameters for the TP, KTP, and SSUM models.

Table 4 - Increased Simulation Draws

Table 4a - SML with Bias Correction					
I=1000, T=24, NS=300, Median Time = 1177 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.9505	0.0356	0.0618	-6.7944
β_x	0.25	0.2498	0.0045	0.0044	-0.1468
σ^2_{RE}	0.20	0.2572	0.0206	0.0607	13.3025
ρ	0.60	0.4095	0.0167	0.1911	-54.6552
$E[\epsilon_t \epsilon_{t-1}]$	0.68	0.5615	0.0133	0.1191	-42.5900

Table 4b - Transition Probability - Increased Draws					
I=1000, T=24, NS=100, NSW=1000, Median Time = 812 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.8740	0.0352	0.0433	4.0312
β_x	0.25	0.2425	0.0054	0.0091	-7.5572
σ^2_{RE}	0.20	0.1710	0.0242	0.0374	-6.5454
ρ	0.60	0.6922	0.0185	0.0940	27.2528
$E[\epsilon_t \epsilon_{t-1}]$	0.68	0.7451	0.0125	0.0662	28.3677

Table 4c - Consistent Transition Probability - Increased Draws					
I=1000, T=24, NS=100, NSW=1000, NSW2=1000, Median Time = 999 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.9578	0.3308	0.3304	-0.9568
β_x	0.25	0.2609	0.0417	0.0424	1.4417
σ^2_{RE}	0.20	0.2009	0.0878	0.0863	0.0573
ρ	0.60	0.5922	0.0651	0.0644	-0.6501
$E[\epsilon_t \epsilon_{t-1}]$	0.68	0.6764	0.0479	0.0472	-0.4038

Table 4d - Simulated Sum - Increased Draws					
I=1000, T=24, N=35, NS=1, NSW=350, Median Time = 864 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.90	-0.9096	0.0528	0.0528	-0.9990
β_x	0.25	0.2497	0.0088	0.0086	-0.1605
σ^2_{RE}	0.20	0.2027	0.0412	0.0406	0.3587
ρ	0.60	0.6077	0.0305	0.0310	1.3897
$E[\epsilon_t \epsilon_{t-1}]$	0.68	0.6880	0.0188	0.0202	2.3340

Notes: For all models, 30 replications were run. SML estimates from Table 3 (with 100 simulation draws) were used for starting parameters for the TP, KTP, and SSUM models.

Table 5- Random Coefficient Models

Table 5a - SML with Bias Correction					
I=1000, T=24, NS=75, Median Time = 735 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.9	-0.7773	0.0501	0.1324	24.4458
β_x	0.25	0.2003	0.0096	0.0505	-51.4719
${\sigma_{RE}}^2$	0.25	0.1832	0.0313	0.0737	-21.3011
σ_1	0.25	0.1944	0.0181	0.0584	-30.5868
σ_2	0.25	0.1951	0.0162	0.0571	-33.706
σ_3	0.25	0.1911	0.0167	0.0611	-35.1183
σ_4	0.25	0.1942	0.0168	0.0581	-33.0445
σ_5	0.25	0.1918	0.0181	0.0608	-32.0196
σ_6	0.25	0.1954	0.0152	0.0565	-35.6487
Table 5b - Transition Probability - Start Values - SML w/ 75 draws					
I=1000, T=24, NS=50, NSW=500, Med Time = 993 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.9	-0.9685	0.0687	0.0968	-9.9642
β_x	0.25	0.2726	0.0176	0.0287	12.823
${\sigma_{RE}}^2$	0.25	0.3539	0.0535	0.1167	19.4126
σ_1	0.25	0.2434	0.0207	0.0216	-3.1649
σ_2	0.25	0.2539	0.0185	0.0188	2.1106
σ_3	0.25	0.2699	0.0229	0.0303	8.7035
σ_4	0.25	0.3104	0.0273	0.0662	22.0912
σ_5	0.25	0.3748	0.0312	0.1286	39.9665
σ_6	0.25	0.4569	0.0448	0.2116	46.1823
Table 5c - Keane's Consistent TP- Start Values - SML w/ 75 draws					
I=1000, T=24, NS=50, NSW=500, NSW2=1000, Med Time = 991 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.9	-0.8939	0.0974	0.0971	0.6169
β_x	0.25	0.247	0.0278	0.0278	-1.0679
${\sigma_{RE}}^2$	0.25	0.2521	0.0861	0.0857	0.2457
σ_1	0.25	0.2537	0.0304	0.0304	1.2261
σ_2	0.25	0.2513	0.0361	0.0359	0.3647
σ_3	0.25	0.2527	0.032	0.0319	0.8451
σ_4	0.25	0.2542	0.0336	0.0337	1.2581
σ_5	0.25	0.2533	0.0368	0.0368	0.9136
σ_6	0.25	0.2628	0.0571	0.0583	2.2499
Table 5d - Simulated Sum - Start Values - SML w/ 75 draws					
I=1000, T=24, N=20, NS=1, NSW=200, Med. Time = 822 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.9	-0.8911	0.0688	0.0690	1.2833
β_x	0.25	0.2485	0.0152	0.0152	-0.9209
${\sigma_{RE}}^2$	0.25	0.2516	0.0495	0.0493	0.3235
σ_1	0.25	0.2480	0.0198	0.0198	-0.9973
σ_2	0.25	0.2506	0.0197	0.0196	0.3168
σ_3	0.25	0.2523	0.0221	0.0221	1.0567
σ_4	0.25	0.2522	0.0222	0.0222	1.0185
σ_5	0.25	0.2509	0.0197	0.0197	0.4916
σ_6	0.25	0.2497	0.0194	0.0193	-0.1037

Notes: For all models, 100 replications were run.

Table 6- R. C. Model - Increased Draws

Table 6a - SML with Bias Correction					
I=1000, T=24, NS=300, Median Time = 2786 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.9	-0.8647	0.0572	0.0663	3.32
β_x	0.25	0.2275	0.0130	0.0258	-9.28
${\sigma_{RE}}^2$	0.25	0.2204	0.0435	0.0519	-3.66
σ_1	0.25	0.2350	0.0192	0.0241	-4.18
σ_2	0.25	0.2299	0.0153	0.0250	-7.03
σ_3	0.25	0.2335	0.0181	0.0242	-4.86
σ_4	0.25	0.2362	0.0188	0.0230	-3.95
σ_5	0.25	0.2324	0.0185	0.0253	-5.09
σ_6	0.25	0.2353	0.0115	0.0184	-6.83
Table 6b - Transition Probability - Starting Values - SML w/75 draws					
I=1000, T=24, NS=100, NSW=1000, Median Time = 1758 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.9	-0.9316	0.0599	0.0669	-2.89
β_x	0.25	0.2595	0.0131	0.0161	3.96
${\sigma_{RE}}^2$	0.25	0.3050	0.0405	0.0679	-0.14
σ_1	0.25	0.2495	0.0158	0.0156	7.43
σ_2	0.25	0.2475	0.0157	0.0157	-0.86
σ_3	0.25	0.2591	0.0160	0.0182	3.12
σ_4	0.25	0.2785	0.0154	0.0322	10.13
σ_5	0.25	0.3110	0.0194	0.0639	17.16
σ_6	0.25	0.3581	0.0200	0.1099	29.49
Table 6c - Keane's Consistent TP - Starting Values - SML w/ 75 draws					
I=1000, T=24, NS=1000, NSW=1000, NSW2=1000, Median Time = 1815 sec					
Parameter	True Value	Mean Estimate	SD of Estimate	MSE	T-Stat Bias
Constant	-0.9	-0.8907	0.0819	0.0810	0.61
β_x	0.25	0.2464	0.0196	0.0196	-0.99
${\sigma_{RE}}^2$	0.25	0.2505	0.0550	0.0540	0.05
σ_1	0.25	0.2538	0.0230	0.0229	0.90
σ_2	0.25	0.2432	0.0177	0.0186	-2.07
σ_3	0.25	0.2486	0.0260	0.0256	-0.28
σ_4	0.25	0.2473	0.0200	0.0198	-0.73
σ_5	0.25	0.2520	0.0274	0.0270	0.40
σ_6	0.25	0.2532	0.0345	0.0341	0.50

Table 6d - Simulated Sum - Starting Values - SML w/ 75 draws					
I=1000, T=24, N=35, NS=1, NSW=350, Median Time = 1578 sec					
Constant	-0.9	-0.9002	0.0640	0.0630	-0.01
β_x	0.25	0.2510	0.0128	0.0126	0.45
${\sigma_{RE}}^2$	0.25	0.2433	0.0426	0.0424	-0.85
σ_1	0.25	0.2547	0.0180	0.0183	1.43
σ_2	0.25	0.2492	0.0202	0.0199	-0.19
σ_3	0.25	0.2536	0.0214	0.0213	0.92
σ_4	0.25	0.2545	0.0184	0.0186	1.36
σ_5	0.25	0.2531	0.0146	0.0147	1.17
σ_6	0.25	0.2497	0.0160	0.0158	-0.09

Notes: For all models, 30 replications were run.

References

[1] Berkovec, James; Stern, Steven. 1991. "Job Exit Behavior of Older Men", *Econometrica*, 59(1), January 1991, pages 189-210.

[2] Bőrsch Supan, A., and Hajivassiliou, V. 1993. "Smooth Unbiased Multivariate Probability Simulators for Maximum Likelihood Estimation of Limited Dependent Variable Models", *Journal of Econometrics*, 58(3), 347-368.

[3] Elrod and Keane. 1995, "A Factor-Analytic Probit Model for Representing the Market Structure in Panel Data", *Journal of Marketing Research*, Feb. 1995, Vol. XXXII, 1-16.

[4] Geweke, J. 1989, "Efficient Simulation from the Multivariate Normal Distribution Subject to Linear Inequality Constraints and the Evaluation of Constraint Probabilities"

[5] Geweke, John F.; Keane, Michael P.; Runkle, David E. 1997, "Statistical Inference in the Multinomial Multiperiod Probit Model", *Journal of Econometrics*, 80(1), pages 125-65.

[6] Hajivassiliou, V. 1993, "Simulation of multivariate normal rectangle probabilities and their derivatives: the effects of vectorization", *International Journal of Supercomputer Applications*, Fall, 231-253.

[7] Hajivassiliou, V. 1994, "A Simulation Estimation Analysis of External Repayments Problems of Developing Countries", *Journal of Applied Econometrics*, 9(2), 109-132.

[8] Hajivassiliou, V. 1996. "A Monte Carlo Comparison of Leading Simulation Estimators for LDV Models", Mimeo, Department of Economics, London School of Economics.

[9] Hajivassiliou, V. 1997, "Simulation-Based Inference and Diagnostic Tests: Some Practical Issues", Cambridge University Press

[10] Hajivassiliou, V. and Ruud, P. 1994, "Classical Estimation Methods Using Simulation" Pages 2383-2441 of: Engle, R., and McFadden, D. (eds), *Handbook of Econometrics*, Vol. 4. North Holland.

[11] Hajivassiliou, Vassilis A.; McFadden, Daniel L. 1998, "The Method of Simulated Scores for the Estimation of LDV Models", *Econometric*, 66(4), July 1998, pages 863-96.

[12] Hajivassiliou, V., McFadden, D., and Ruud, P. 1996, "Simulation of Multivariate Normal Rectangle Probabilities and Their Derivatives: Theoretical and Computational Results", *Journal of Econometrics*, 72(1&2), 85-134.

[13] Keane, M. 1994. "A Computationally Efficient Practical Simulation Estimator for Panel Data", *Econometrica*, 62(1), 95-116.

[14] Keane, Michael P.; Wolpin, Kenneth I. 1994, "The Solution and Estimation of Discrete Choice Dynamic Programming Models by Simulation and Interpolation", *Review of Economics and Statistics*, 76(4), November 1994, pages 648-72.

[15] Lee, Lung Fei. 1995, "Asymptotic Bias in Simulated Maximum Likelihood Estimation of Discrete Choice Models", *Econometric Theory*, 11(3), August 1995, pages 437-83.

[16] Lee, Lung Fei. 1998, "Simulated Maximum Likelihood Estimation of Dynamic Discrete Choice Statistical Models: Some Monte Carlo Results", *Journal of Econometrics* 82(1), January 1998, pages 1-35.

[17] Lerman, S. and Manski, C. 1981. "On the Use of Simulated Frequencies to Approximate Choice Probabilities", Pages 305-319 of: Manski, C., and McFadden, D. (eds), *Structural Analysis of Discrete Data with Econometric Applications. MIT Press.*

[18] McCulloch, R., and Rossi, P. 1994, "An Exact Likelihood Analysis of the Multinomial Probit Model", *Journal of Econometrics*, 64.

[19] McFadden, D. 1989, "A Method of Simulated Moments for Estimation of Discrete Response Models without Numerical Integration", *Econometrica*, 57(5), 995-1026.

[20] McFadden, Daniel; Ruud, Paul A. 1994, "Estimation by Simulation", *Review of Economics and Statistics*, 76(4), November 1994, pages 591-608.

[21] Pakes, A., and Pollard, D. 1989, "Simulation and the Asymptotics of Optimization Estimators", *Econometrica*, 57, 1027-1057.

[22] Stern, S. 1992, "A Method for Smoothing Simulated Moments of Discrete Probabilities in Mutinomial Probit Models", *Econometrica*, 60, 943-952.

[23] Stern, Steven 1994,"Two Dynamic Discrete Choice Estimation Problems and Simulation Method Solution", *Review of Economics and Statistics*, 76(4), November 1994, pages 695-702.