A Practitioner's Guide to Bayesian Estimation of Discrete Choice Dynamic Programming Models*

Andrew T. Ching†
Rotman School of Management
University of Toronto

Susumu Imai
Department of Economics
Queen's University

Masakazu Ishihara
Rotman School of Management
University of Toronto

Neelam Jain
Department of Economics
City University London

First draft: May 1, 2009
Revised: March 9, 2010

*We thank Martin Burda, Monica Meireles, Matthew Osborne, Peter Rossi, Andrei Strijnev, K. Sudhir and two anonymous referees for their helpful comments. We also thank the participants of the UCLA Marketing Camp, SBIES conference, Marketing Science Conference, Marketing Dynamics Conference, UTD-FORMS Conference, Canadian Economic Association Meeting, Econometric Society Meeting and Ph.D. seminars at OSU's Fisher College of Business, Yale School of Management, and University of Groningen for their useful feedback. Hyunwoo Lim provided excellent research assistance. All remaining errors are ours.

†The computer codes (in C and Matlab) for implementing the Monte Carlo exercises are available upon request. Please direct all correspondence to: Andrew Ching, Rotman School of Management, University of Toronto, 105 St George Street, Toronto, ON, CANADA M5S 3E6. Email: andrew.ching@rotman.utoronto.ca. Phone: 416-946-0728. Fax: 416-978-5433.

A Practitioner's Guide to Bayesian Estimation of Discrete Choice Dynamic Programming Models

Abstract

This paper provides a step-by-step guide to estimating infinite horizon discrete choice dynamic programming (DDP) models using a new Bayesian estimation algorithm (Imai, Jain and Ching, Econometrica 77:1865-1899, 2009) (IJC). In the conventional nested fixed point algorithm, most of the information obtained in the past iterations remains unused in the current iteration. In contrast, the IJC algorithm extensively uses the computational results obtained from the past iterations to help solve the DDP model at the current iterated parameter values. Consequently, it has the potential to significantly alleviate the computational burden of estimating DDP models. To illustrate this new estimation method, we use a simple dynamic store choice model where stores offer "frequent-buyer" type reward programs. We show that the parameters of this model, including the discount factor, are well-identified. Our Monte Carlo results demonstrate that the IJC method is able to recover the true parameter values of this model quite precisely. We also show that the IJC method could reduce the estimation time significantly when estimating DDP models with unobserved heterogeneity, especially when the discount factor is close to 1.

Keywords: Bayesian Estimation, Dynamic Programming, Discrete Choice Models, Reward Programs

JEL: C11, C35, C61, D91, M31

1 Introduction

In economics and marketing, there is a growing empirical literature which studies choice of agents in both the demand and supply side, taking into account their forward-looking behavior. A common framework to capture consumers' or firms' forward-looking behavior is the discrete choice dynamic programming (DDP) model. This framework has been applied to study a manager's decision to replace old equipment (e.g., Rust 1987), career decision choice (e.g., Keane and Wolpin 1997; Diermier, Merlo and Keane 2005), choice to commit crimes (e.g., Imai and Krishna 2004), dynamic brand choice (e.g., Erdem and Keane 1996; Gönül and Srinivasan 1996; Crawford and Shum 2005), dynamic quantity choice with stockpiling behavior (e.g., Erdem, Imai and Keane 2003; Sun 2005; Hendel and Nevo 2006), new product/technology adoption decisions (e.g., Ackerberg 2003; Song and Chintagunta 2003; Yang and Ching 2010), new product introduction decisions (e.g., Hitsch 2006), dynamic pricing decisions (e.g., Ching 2010), etc. Although the framework provides a theoretically tractable way to model forward-looking incentives, and this literature has been growing, it remains small relative to the literature that models choice using a static reduced form framework. This is mainly due to two obstacles of estimating this class of models: (i) the curse of dimensionality problem in the state space, putting a constraint on developing models that match the real world applications; (ii) the complexity of the likelihood/GMM objective function, making it difficult to search for the global maximum/minimum when using the classical approach to estimate them. Several studies have proposed different ways to approximate the dynamic programming solutions to overcome the hurdle due to the curse of dimensionality problem (e.g., Keane and Wolpin 1994; Rust 1997; Hotz and Miller 1993; Aguirregabiria and Mira 2002; Ackerberg 2009).[1] Nevertheless, little progress has been made in handling the complexity of the likelihood function resulting from DDP models. A typical approach is to use different initial values to re-estimate the model, and check which

[1]Geweke and Keane (2000) proposed to use a flexible polynomial to approximate the future component of the Bellman equation. Their approach allowed them to conduct Bayesian inference on the structural parameters of the current payoff functions and the reduced form parameters of the polynomial approximations. However, since it completely avoids solving and fully specifying the DDP model, their estimation results are not efficient and policy experiments cannot be conducted under their approach.

set of parameter estimates gives the highest likelihood value. However, without knowing the exact shape of the likelihood function, it is often difficult to confirm whether the estimated parameter vector indeed gives us the global maximum.

In the past two decades, the Bayesian Markov Chain Monte Carlo (MCMC) approach has provided a tractable way to simulate the posterior distribution of parameter vectors for complicated static discrete choice models, making the posterior mean an attractive estimator compared with classical point estimates in that setting (Albert and Chib 1993; McCulloch and Rossi 1994; Allenby and Lenk 1994; Allenby 1994; Rossi et al. 1996; Allenby and Rossi 1999). Nonetheless, researchers seldom use the Bayesian approach to estimate DDP models. The main problem is that the Bayesian MCMC approach typically requires many more iterations than the classical approach to achieve convergence. In each simulated draw of the parameter vector, the DDP model needs to be solved to calculate the likelihood function. As a result, the computational burden of solving a DDP model has essentially ruled out the Bayesian approach except for very simple models, where the model can be solved very quickly or there exists a closed form solution (e.g., Lancaster 1997).

Recently, Imai, Jain and Ching (2009a) (IJC) propose a new modified Bayesian MCMC algorithm to reduce the computational burden of estimating infinite horizon DDP models. This method combines the DDP solution algorithm with the Bayesian MCMC algorithm into a single algorithm, which solves the DDP model and estimates its structural parameters simultaneously. In the conventional nested fixed point algorithm, most of the information obtained in the past iterations remains unused in the current iteration. In contrast, the IJC algorithm extensively uses the computational results obtained from the past iterations to help solve the DDP model at the current iterated parameter values. This new method is potentially superior to prior methods because (1) it could significantly reduce the computational burden of solving for the DDP model in each iteration, and (2) it produces the posterior distribution of parameter vectors, and the corresponding solutions for the DDP model – this avoids the need to search for the global maximum of a complicated likelihood function.

The objective of this paper is to provide a step-by-step guide to use the IJC method in terms of an example relevant to marketing and industrial organization. We consider an example where consumers need to choose which store to visit, and each store offers its own frequent-buyer rewards program. We show that the parameters of this model, including the discount factor, are well-identified. Our Monte Carlo results demonstrate that the IJC method is able to recover the true parameter values of this model quite precisely. We also show that the IJC method could reduce the estimation time very significantly when estimating DDP models with unobserved heterogeneity, especially when the discount factor is close to 1.

The rest of the paper is organized as follows. In section 2, we present the dynamic store choice model with reward programs and discuss its identification. In section 3, we discuss how to implement the IJC method to estimate this model in detail. We also discuss several practical aspects of using this new method. Section 4 conducts three sets of Monte Carlo experiments to demonstrate the performance and some properties of the IJC algorithm. Section 5 discusses how to extend the algorithm to allow for continuous state variables, and compares it with other approximation approaches to estimate DDP models. Section 6 is the conclusion.

2 The Model

2.1 The Basic Framework

Suppose that there are two supermarket chains in a city ($j = 1, 2$). Each supermarket chain offers a stamp card, which can be exchanged for a gift upon completion. The stamp card for a chain is valid for all stores in the same chain. Consumers get one stamp for each visit at any store of a chain with a purchase.

Reward programs at the two supermarket chains differ in terms of (i) the number of stamps required for a gift ($\bar{S}_j$), and (ii) the mean value of the gift (G_j). Consumers get a gift in the same period that they complete the stamp card. Once consumers receive a gift, they will start with a blank stamp card again in the next period.

In each period, a consumer chooses which supermarket chain to visit. Each chain offers different prices for their products. Let p_{ijt} be the price index that consumer i faces in supermarket chain j at time t. We allow the price index to be individual specific to reflect that consumers may differ in terms of their consumption baskets (e.g., some consumers have babies and they need to shop for diapers, some consumers are vegetarian and they do not shop for meats, etc.). For simplicity, we assume that p_{ijt} is drawn from an iid normal distribution, $N(\bar{p}, \sigma_p^2)$, which is known to consumers, and they observe p_{ijt} in the period that they decide their choices.[2] Let $s_{ijt} \in \mathcal{S}_j \equiv \{0, 1, \ldots, \bar{S}_j - 1\}$ denote the number of stamps collected for chain j in period t before consumer i makes a decision. Note that s_{ijt} does not take the value $\bar{S}_j$ because of our assumption that consumers get a gift in the same period that they complete the stamp card. The state space of this dynamic model is $\mathcal{S} \equiv \mathcal{S}_1 \times \mathcal{S}_2$.

Consumer i's single period utility of visiting supermarket chain j in period t at $s_{it} = (s_{i1t}, s_{i2t})$ and $p_{it} = (p_{i1t}, p_{i2t})$ is given by

$$U_{ijt}(s_{it}, p_{it}) = \begin{cases} \alpha_j + \gamma p_{ijt} + \epsilon_{ijt} & \text{if } s_{ijt} < \bar{S}_j - 1 \\ \alpha_j + \gamma p_{ijt} + G_{ij} + \epsilon_{ijt} & \text{if } s_{ijt} = \bar{S}_j - 1 \end{cases}$$

where α_j captures the brand equity for chain j, γ is the price sensitivity, G_{ij} is consumer i's valuation of the gift for chain j, and ϵ_{ijt} is the *i.i.d.* idiosyncratic random utility term. We assume ϵ_{ijt} is unobserved to researchers and extreme-value distributed. G_{ij} is assumed to be normally distributed around G_j with the standard deviation, σ_{G_j}. We allow G_{ij} to differ across consumers to reflect that individual's valuation for a gift may vary.[3] In each period, consumers may choose not to go shopping. The single period mean utility of no shopping is normalized to zero, i.e., $U_{i0t}(s_{it}, p_{it}) = \epsilon_{i0t}$.

[2] We will discuss how to estimate an extension where p_{ijt} is serially correlated in section 5.1.

[3] Suppose that the gift is a vase. Some consumers may value it highly, but others who already have several vases at home, may not.

Consumer i's objective is to maximize the sum of the present discounted future utility:

$$\max_{\{b_{ijt}\}_{t=1}^{\infty}} E\left[\sum_{t=1}^{\infty}\beta^{t-1}\sum_{j=0}^{2} b_{ijt}U_{ijt}(s_{it}, p_{it})\right],$$

where $b_{ijt} = 1$ if consumer i chooses chain j in period t and $b_{ijt} = 0$ otherwise. β is the discount factor. The evolution of state, s_{it}, is deterministic and depends on consumers' choice. Given the state s_{ijt}, the next period state, s_{ijt+1}, is determined as follows:

$$s_{ijt+1} = \begin{cases} s_{ijt} + 1 & \text{if } s_{ijt} < \bar{S}_j - 1 \text{ and purchase at chain } j \text{ in period } t; \\ 0 & \text{if } s_{ijt} = \bar{S}_j - 1 \text{ and purchase at chain } j \text{ in period } t; \\ s_{ijt} & \text{if purchase at chain } -j \text{ or no shopping in period } t. \end{cases} \quad (1)$$

The parameters of the model are $\{\alpha_j, G_j, \sigma_{G_j}\}_{j=1}^{2}, \gamma, \beta, \bar{p}$, and σ_p. It is well-known that there is a one-to-one relationship between the solution of the dynamic optimization problem and the following functional equation (i.e., Bellman equation). Since the dynamic optimization problem is stationary, we drop the t subscript hereafter.[4] Let θ be the vector of parameters, E_ϵ denote expectation w.r.t. ϵ, and $G_i \equiv (G_{i1}, G_{i2})$. The value function,[5] which captures the expected maximum of the alternative-specific value functions, is given by: For each s_i, p_i,

$$\begin{aligned} V(s_i, p_i; G_i, \theta) &\equiv E_\epsilon \max\{V_0(s_i;\theta) + \epsilon_{i0}, V_1(s_i, p_{i1}; G_i, \theta) + \epsilon_{i1}, V_2(s_i, p_{i2}; G_i, \theta) + \epsilon_{i2}\} \\ &= \log[\exp(V_0(s_i;\theta)) + \exp(V_1(s_i, p_{i1}; G_i, \theta)) + \exp(V_2(s_i, p_{i2}; G_i, \theta))], \end{aligned} \quad (2)$$

where the second equality follows from the extreme value assumption on ϵ. Denote p' as the price vector next period. The *alternative-specific value functions* obey the Bellman equation (Bellman 1957): For $j = 1, 2$,

$$V_j(s_{ij}, s_{i-j}, p_i; G_i, \theta) = \begin{cases} \alpha_j + \gamma p_{ij} + \beta E_{p'}[V(s_{ij}+1, s_{i-j}, p'; G_i, \theta)] & \text{if } s_{ij} < \bar{S}_j - 1, \\ \alpha_j + \gamma p_{ij} + G_{ij} + \beta E_{p'}[V(0, s_{i-j}, p'; G_i, \theta)] & \text{if } s_{ij} = \bar{S}_j - 1, \end{cases} \quad (3)$$

$$V_0(s_i, p_i; \theta) = \beta E_{p'}[V(s_i, p'; G_i, \theta)], \quad (4)$$

[4]By "stationary," we mean that conditioning on the value of the state variables, the optimal decisions of the consumers do not depend on t.

[5]We should note that the literature often defines the value function as $\max_j\{V_j(s_i, p_i) + \epsilon_{ij}, j = 0, 1, 2\}$, and use the term, Emax function, to refer $E_\epsilon \max_j\{V_j(s_i, p_i) + \epsilon_{ij}, j = 0, 1, 2\}$ (e.g., Keane and Wolpin 1994). We use our defintion for the value function in equation (2) because it helps simplify the notation significantly.

where the expected future value (taking expectation with respect to p') is defined as

$$E_{p'}[V(.,p';G_i,\theta)] = \int V(.,p';G_i,\theta)dF(p').$$

Let $\Gamma_{i,\theta}$ be the Bellman operator corresponding to the value function defined by Equations (2)-(4), where the subscript "i,θ" indicates that the operator is specific to (G_i,θ). Let $\bar{U}_j(s_i,p_i;G_i,\theta) \equiv U_{ij} - \epsilon_{ij}$, and s_i' denote the vector of the number of stamps next period. Then for any arbitrary function, f,

$$(\Gamma_{i,\theta})f(s_i,p_i) = E_\epsilon \max_j\{\bar{U}_j(s_i,p_i;G_i,\theta) + \epsilon_{ij} + \beta E_{p'}[f(s_i',p')]\}. \tag{5}$$

Note that the value function, $V(s_i,p_i;G_i,\theta)$, is a fixed point of $\Gamma_{i,\theta}$, i.e., $V = \Gamma_{i,\theta}V$. Moreover, it can be shown that $\Gamma_{i,\theta}$ is a contraction mapping. This result is very useful because it implies that: (i) there is a unique fixed point of $\Gamma_{i,\theta}$, and (ii) if we start off with any arbitrary initial guess of the value function, V^0, and recursively apply the Bellman operator to it, i.e., $V^{n+1} = \Gamma_{i,\theta}V^n$, then $V^n \to V$ uniformly. This procedure, which is called *the method of successive approximation*, provides a tractable way to solve for the Bellman equation and agents' optimal decisions numerically.

It should be pointed out that the general dynamics of our model is more complicated than the one used in IJC for their Monte Carlo exercises. The model here has two endogenous state variables (s_1,s_2), while the dynamic firm entry-exit decision model used in IJC has one exogenous state variable (capital stock). However, IJC consider a normal error term, which is more general than the extreme value error term we assume here. We consider the extreme value error term because (i) the choice probability and $V(s_i,p_i;G_i,\theta)$ have closed form analytical expressions under this distributional assumption, and (ii) our analysis here would complement IJC's.

2.2 Identification

The main dynamics of the model is the intertemporal trade-off created by the reward program. Suppose that a consumer is closer to the completion of the stamp card for chain 1, but the price is lower in chain 2 today. If the consumer chooses chain 2 based on the lower price, he or she will delay the completion of the stamp card for chain 1. If the consumer takes the future into account, the delay will lower the present discounted value of the reward. Thus he/she will have an incentive to keep shopping at chain 1 even though prices at chain 2 are lower. Moreover, such an incentive should depend on the value of the discount factor.

This dynamic trade-off suggests that the variation of the empirical choice frequency of visiting the supermarket chains across states (i.e., the number of stamps collected) should allow us to pin down the discount factor. To illustrate this point, we consider a simplified version of the model where consumers are homogeneous, and there is only one supermarket chain and an outside option. We will drop the j subscript to simplify the notations in this subsection. We simulate the choice probabilities across s for different discount factors by setting $\alpha = -2$, $\gamma = 0$, $G = 3$, $\sigma_G = 0$ and $\bar{S} = 5$. Figure 1 shows how the choice probability of visiting the chain changes across states for different discount factors ($\beta = 0, 0.5, 0.75, 0.9, 0.999$). In general, we see that the choice probabilities increase with s. When β is small, the choice probabilities are relatively flat for small s, but become much higher as s approaches $\bar{S} - 1$. In the extreme case of $\beta = 0$, consumers only care about the current period utility. As a result, the choice probability of shopping at the chain is flat for $s = 0, 1, 2, 3$ and goes up only when $s = 4$ (because the current period utility of visiting the chain includes the value of the gift only when $s = 4$). As β increases, the choice probabilities become flatter as we move across s. As it approaches 1 ($\beta = 0.999$), the choice probabilities are essentially constant across states, and higher than those of $\beta = 0$ for $s = 0, 1, 2, 3$. This tendency is due to two counteracting forces: (i) the expected gain of obtaining an extra stamp today ($R(s, \beta)$); (ii) the option value of waiting ($W(s, \beta)$). We define

$$
\begin{aligned}
R(s, \beta) &\equiv \begin{cases} \beta EV(s' = s + 1) & \text{if } s < \bar{S} - 1, \\ G + \beta EV(s' = 0) & \text{if } s = \bar{S} - 1. \end{cases} \\
W(s, \beta) &\equiv \beta EV(s' = s).
\end{aligned}
$$

The choice probability of visiting a store is driven by the incentive of obtaining the gift sooner, which is measured by $R(s, \beta) - W(s, \beta)$. To understand the basic intuition, let's consider a case when β is small (say $\beta = 0.5$). When a consumer is very close to getting a reward (i.e., s is close to $\bar{S} - 1$), $R(s, \beta)$ is much higher than $W(s, \beta)$ because waiting for an extra period reduces the expected discounted value of the gift significantly. But when a consumer has very few stamps (i.e., s is close or equal to zero), both $R(s, \beta)$ and $W(s, \beta)$ are very small because the value of the gift is heavily discounted in either of them. Consequently, their difference also becomes much smaller, and so is the incentive of obtaining an extra stamp today. This explains why the increase in choice probability is relatively flat when s is small, but increases sharply when it approaches $\bar{S}$.

Now let's consider the case of $\beta \to 1$. In this case, even when a consumer is close to getting a reward, waiting for an extra period would only hurt him/her very little, and consequently, $R(s, \beta) - W(s, \beta)$ becomes smaller. This explains why the choice probability at $s = 4$ decreases as β increases. On the other hand, even if consumers are far away from getting the reward, neither $R(s, \beta)$ nor $W(s, \beta)$ would be discounted much with β close to 1. As a result, $R(s, \beta) - W(s, \beta)$ remains fairly stable across s. This explains why the choice probabilities are essentially constant for all s. In Appendix A, we provide more explanations and a formal proof for this result.

The above discussion suggests that unless the choice probabilities are flat across s, the overall shape of choice probabilities across s will allow us to identify α, G, and β. Two important aspects are: (i) changes in choice probability across s identify G, and β;[6] (ii) the overall level of choice probabilities across s identifies α. If the choice probabilities are (almost) flat across s, then we could have either $G = 0$, or β is very close to 1. In practice, we expect that when β is close to 1, it could be quite hard to separately identify α and G because α shifts the choice probabilities equally across s, while G shifts them almost equally across s.

To illustrate the intuition that we discussed above further, Figure 2 shows how the choice probability of visiting the chain changes with β for any given s. In general, as we increase β, we have three

[6] β is identified because the model has an exclusion restriction (Magnac and Thesmar 2002). Note that the number of stamps, s, does not enter the current utility function other than when $s = \bar{S} - 1$.

observations: (i) when $s = 4$, the choice probability monotonically decreases; (ii) when $s < 3$, the choice probability monotonically increases; (iii) when $s = 3$, it first increases and then decreases. The discussion above has already explained observation (i). Observation (ii) indicates that $R(s, \beta) - W(s, \beta)$ always increases with β for small s. Observation (iii) shows an intermediate case: $R(s = 3, \beta)$ initially increases faster than $W(s = 3, \beta)$ as β increases; but when $\beta \to 1$, $W(s = 3, \beta)$ catches up. This explains why the choice probability of visiting the chain first increases and then decreases. Finally, $R(s, \beta)$ is always higher than $W(s, \beta)$ as long as $\beta > 0$.

We should mention that there are three papers that study reward programs in a dynamic setting similar to ours: Lewis (2004), Kopalle et al. (2007) and Hartmann and Viard (2008). In particular, Hartmann and Viard (2008) estimated a dynamic model with reward programs that is similar to the one presented here. The main differences are (1) we allow for two supermarket chains with different reward programs in terms of $(G_j, \bar{S}_j)$ while they considered one store (golf club); (2) we estimate the discount factor (i.e., β) while they fixed it according to the interest rate. Hartmann and Viard (2008) also discussed how the discount factor would affect the pattern of choice probabilities. However, they did not discuss how the patterns could separately identify β, α_j and G_j. Instead, they took the intrinsic discount factor as exogenously given (determined by the interest rate), and argued that the discounting effect would happen through the "artificial" discount factor, which depends on how frequently a customer visits a store (determined by α_j here). Neither Lewis (2004) nor Kopalle et al. (2007) estimate the discount factor.

3 Estimation Procedures

Having shown that the model is well-identified, we now discuss how to estimate this model. We describe the conventional Bayesian estimation method and the IJC method in order, and highlight their differences.

3.1 Conventional Bayesian approach (full-solution based method)

The conventional approach is essentially a nested fixed point algorithm proposed by Rust (1987). The procedure proceeds in the following two main steps: the outer loop and inner loop. We also refer readers to Figure 3 for a flowchart as they read the following two subsections.

3.1.1 The outer loop (MCMC algorithm)

The outer loop is essentially a Metropolis-Hastings (M-H) algorithm, which is a version of Markov-chain Monte Carlo (MCMC) algorithm.[7] The M-H algorithm allows us to simulate a markov-chain of $\{\theta^l\}_{l=1}^R$ which converges to the true posterior distribution. Let $b_{it} = (b_{i0t}, b_{i1t}, b_{i2t})$ be a vector of indicator functions for buying decisions, $s_{it} = (s_{i1t}, s_{i2t})$ and $p_{it} = (p_{i1t}, p_{i2t})$ be the vector of state variables and prices, respectively, for consumer i at time t. We use $(b^d_{it}, s^d_{it}, p^d_{it}), I$ and T_i to denote the observed data, total number of consumers, and total number of periods observed for each consumer i, respectively. We let $\mathbf{b^d} \equiv \{b^d_{it}, \forall i, t\}$, and $\rho(\mathbf{b^d}|\theta)$ be the likelihood of observing $\mathbf{b^d}$.

$$\rho(\mathbf{b^d}|\theta) = \prod_{i=1}^{I}\prod_{t=1}^{T_i}\prod_{j=0}^{2}\left(\frac{\exp(V_j(s^d_{it}, p^d_{ijt}; \theta))}{\sum_{k=0}^{2}\exp(V_k(s^d_{it}, p^d_{ikt}; \theta))}\right)^{b^d_{ijt}}.$$

Let $\pi(\theta)$ be the prior of θ. The posterior density of θ is proportional to $\pi(\theta) \cdot \rho(\mathbf{b^d}|\theta)$.

It is well-known that directly drawing θ from this posterior density is computationally difficult. MCMC provides a tractable way to solve this problem. In the M-H algorithm, we first draw a candidate parameter vector from a proposal density (it should be a density from which we can draw easily). Then, we decide whether or not to accept the candidate parameter vector. We will denote the candidate parameter vector in iteration r by θ^{*r} and the accepted parameter vector in iteration r by θ^r. We draw a candidate parameter vector θ^{*r} from a proposal density, $q(\theta^{r-1}, \theta^{*r})$, which is the density of θ^{*r} given θ^{r-1} (e.g., $\theta^{*r} \sim N(\theta^{r-1}, \sigma^2)$). Then we accept θ^{*r}, i.e., set $\theta^r = \theta^{*r}$ with probability,

$$\lambda = \min\left(\frac{\pi(\theta^{*r}) \cdot \rho(\mathbf{b^d}|\theta^{*r}) \cdot q(\theta^{*r}, \theta^{r-1})}{\pi(\theta^{r-1}) \cdot \rho(\mathbf{b^d}|\theta^{r-1}) \cdot q(\theta^{r-1}, \theta^{*r})}, 1\right);$$

[7] Walsh (2004) provides an excellent introduction to MCMC methods.

and we reject θ^{*r}, i.e., set $\theta^r = \theta^{r-1}$, with probability $1-\lambda$. It can be proved that the accepted draws converge to the posterior distribution (Hastings 1970).

Note that $\rho(\mathbf{b^d}|\theta^{*r})$ and $\rho(\mathbf{b^d}|\theta^{r-1})$ depend on the value function evaluated at θ^{*r} and θ^{r-1}, respectively. We now discuss how to implement *the inner loop*, which solves for the value functions numerically at these parameter vectors.

3.1.2 The inner loop (the method of successive approximation)

To solve the model described in Section 2 numerically, we take advantage of the contraction mapping property of the Bellman operator and apply the method of successive approximation as follows. To simplify notation, we drop the i subscript for s and p. For each consumer i:

1. For each $j = 1, 2$, make M independent draws of $\{\tilde{p}_j^m\}_{m=1}^M$ from the price distribution function, $N(\bar{p}, \sigma_p^2)$. We denote the draws to be $\mathcal{P}^M = \{\tilde{p}^m, m = 1, ..., M\}$, where $\tilde{p}^m = (\tilde{p}_1^m, \tilde{p}_2^m)$, and fix them below.

2. Start with an arbitrary initial guess of the value functions, e.g., set $V^0(s, p; G_i, \theta) = 0$, $\forall s \in \mathcal{S}$ and $\forall p$. Suppose that we know V^l, where l indexes the number of iterations. Steps 3 & 4 discuss how to obtain V^{l+1}.

3. For each s, substitute $\{\tilde{p}^m\}_{m=1}^M$ into $V^l(s, p; G_i, \theta)$, and then take the average across $\tilde{p}^m$'s to obtain a Monte Carlo approximation of the expected future value. Let $\bar{E}_{p'}$ denote the Monte Carlo approximation of the expectation. Then,

$$\bar{E}_{p'} V^l(s, p'; G_i, \theta) = \frac{1}{M} \sum_{m=1}^{M} V^l(s, \tilde{p}^m; G_i, \theta).$$

4. Substitute this approximated expected future value function into the Bellman operator and obtain $V^{l+1}(s, \tilde{p}^m; G_i, \theta), \forall s \in \mathcal{S}, \forall \tilde{p}^m \in \mathcal{P}^M$, that is,

$$V^{l+1}(s, \tilde{p}^m; G_i, \theta) = \log[\exp(V_0^l(s; \theta)) + \exp(V_1^l(s, \tilde{p}_1^m; G_i, \theta)) + \exp(V_2^l(s, \tilde{p}_2^m; G_i, \theta))], \quad (6)$$

where for $j = 1, 2$,

$$V_j^l(s, \tilde{p}^m; G_i, \theta) = \begin{cases} \alpha_j + \gamma \tilde{p}_j^m + \beta \bar{E}_{p'} V^l(s_j + 1, s_{-j}, p'; G_i, \theta) & \text{if } s_j < \bar{S}_j - 1, \\ \alpha_j + \gamma \tilde{p}_j^m + G_{ij} + \beta \bar{E}_{p'} V^l(0, s_{-j}, p'; G_i, \theta) & \text{if } s_j = \bar{S}_j - 1, \end{cases} \tag{7}$$

$$V_0^l(s; \theta) = \beta \bar{E}_{p'} V^l(s, p'; G_i, \theta). \tag{8}$$

5. Repeat step 3-4 until $\bar{E}_{p'} V^{l+1}(s, p'; G_i, \theta)$ converges $\forall s \in \mathcal{S}$.

In general, the computational burden increases exponentially with the number of state variables, and linearly with the number of values in each state variable. Also, the number of iterations required to achieve the convergence of $\bar{E}_{p'} V^l(s, p'; G_i, \theta)$ increases as the discount factor β increases.

We should emphasize that a M-H algorithm (or in general a MCMC algorithm) usually needs to run for 10,000 to 30,000 iterations to obtain enough draws for approximating the posterior distribution. This implies that the numerical solution of a DDP model described above also needs to be conducted for about the same number of times. This is the main reason why researchers almost never use the Bayesian approach to estimate a DDP model. It is clear that compared with maximum likelihood (ML), which typically requires a few hundred iterations to achieve convergence, the computational burden of the conventional Bayesian approach to estimating DDP models is much higher. We will now turn to discuss the IJC algorithm, and explain how it reduces the computational burden of estimating DDP models.

3.2 IJC algorithm

By now, it should be clear that the main obstacle of the conventional Bayesian approach is the computational burden of solving for the value function at a large number of parameter vectors simulated from the M-H algorithm. The main difference between the IJC and conventional Bayesian methods is in the inner loop: instead of using the contraction mapping argument to obtain the value function, IJC propose to approximate it based on the past outcomes of the algorithm. IJC relies on two insights. First, it could be quite wasteful to compute the value function exactly before the markov chain converges to the true posterior distribution. Therefore, the IJC algorithm proposes to "partially" solve for the Bellman equation for each parameter vector draw (at the minimum, only apply the Bellman operator once in each iteration). Since the value functions (alternative-specific value functions) obtained in IJC are proxies, we call them the *pseudo-value functions* (*pseudo alternative-specific value functions*). Second, because the value function is continuous in the parameter space, the pseudo-value functions evaluated at the past MCMC draws of parameters contain useful information about the value functions at the current draw of parameter vector, in particular, for those evaluated within its neighborhood. However, the traditional nested fixed point algorithm hardly makes use of its past outcomes. Based on the second insight, IJC propose to use the past pseudo-value functions to form a non-parametric estimate of the expected future value function evaluated at the current draw of parameter vector. More precisely, IJC propose to replace the contraction mapping procedure of solving the expected future value functions with a weighted average of the pseudo-value functions obtained as past outcomes of the estimation algorithm. The weights depend on the distance between the past parameter vector draws and the current one – the shorter the distance, the higher the weight. Such a non-parametric estimate is usually computationally much cheaper than the method of successive approximation described in section 3.1.2, especially for β close to 1. Consequently, IJC has the potential to significantly reduce the computational burden per iteration compared with the conventional Bayesian approach.

In the context of the reward program example without unobserved consumer heterogeneity (i.e., $G_{ij} = G_j \ \forall i$), the output of the algorithm in each iteration r is $\{\theta^r, \theta^{*r}, \tilde{V}^r(., \tilde{p}^r; \theta^{*r})\}$, where $\tilde{p}_j^r$ is a draw from $N(\bar{p}, \sigma_p^2)$ for $j = 1, 2$, and $\tilde{V}^r(., \tilde{p}^r; \theta^{*r})$ is the pseudo-value function of s, given $\tilde{p}^r$ and θ^{*r}. In addition to storing $\{\theta^l\}_{l=1}^r$, IJC propose to store $H^r = \{\theta^{*l}, \tilde{V}^l(., \tilde{p}^l; \theta^{*l})\}_{l=r-N}^{r-1}$ (i.e., the N most recent past candidate parameter vectors and the pseudo-value functions evaluated at them). We emphasize that $\tilde{V}^l(., \tilde{p}^l; \theta^{*l})$ refers to $\tilde{V}^l(s, \tilde{p}^l; \theta^{*l})$ for all $s \in \mathcal{S}$. In other words, we need to store $\tilde{V}^l$ at all s. Note that we store the past pseudo-value functions evaluated at θ^{*l} (the draw from the proposal distribution) instead of θ^l (the draw from the posterior distribution of θ) – we will explain why we take this approach later. The pseudo-value functions, $\tilde{V}^r$, and the pseudo alternative specific value functions, $\tilde{V}_j^r$, are defined as follows. First, we draw $\tilde{p}_j^r$ from $N(\bar{p}, \sigma_p^2)$, for $j = 1, 2$. Then, for each $s \in \mathcal{S}$,

$$\tilde{V}^r(s, \tilde{p}^r; \theta^{*r}) = \log[\exp(\tilde{V}_0^r(s; \theta^{*r})) + \exp(\tilde{V}_1^r(s, \tilde{p}_1^r; \theta^{*r})) + \exp(\tilde{V}_2^r(s, \tilde{p}_2^r; \theta^{*r}))], \tag{9}$$

where for $j = 1, 2$,

$$\tilde{V}_j^r(s, \tilde{p}_j^r; \theta^{*r}) = \begin{cases} \alpha_j + \gamma \tilde{p}_j^r + \beta \hat{E}_{p'}^r V(s_j + 1, s_{-j}, p'; \theta^{*r}) & \text{if } s_j < \bar{S}_j - 1, \\ \alpha_j + \gamma \tilde{p}_j^r + G_j + \beta \hat{E}_{p'}^r V(0, s_{-j}, p'; \theta^{*r}) & \text{if } s_j = \bar{S}_j - 1, \end{cases} \tag{10}$$

$$\tilde{V}_0^r(s; \theta^{*r}) = \beta \hat{E}_{p'}^r V(s, p'; \theta^{*r}). \tag{11}$$

The *pseudo* expected future value function, $\hat{E}_{p'}^r V(., p'; \theta^{*r})$, is defined as the weighted average of the past pseudo-value functions obtained from the estimation algorithm. For instance, $\hat{E}_{p'}^r V(s, p'; \theta^{*r})$ can be constructed as follows:

$$\hat{E}_{p'}^r V(s, p'; \theta^{*r}) = \sum_{l=r-N}^{r-1} \tilde{V}^l(s, \tilde{p}^l; \theta^{*l}) \frac{K_h(\theta^{*r} - \theta^{*l})}{\sum_{k=r-N}^{r-1} K_h(\theta^{*r} - \theta^{*k})}, \tag{12}$$

where $K_h(.)$ is a Gaussian kernel with bandwidth h. We make three remarks here. First, the kernel captures the idea that we assign higher weights to the past pseudo-value functions which are evaluated at parameter vectors that are closer to θ^{*r}. Second, the above approximation relies on a "moving window" of past value functions and emphasizes that we need to discard the "old" ones and use the most recent ones. This is because the pseudo-value function produced in each iteration represents an improved approximation of the true value function. By discarding the "old" pseudo-value functions, it ensures that we use the most accurate past pseudo-value functions to form a weighted average approximation for the expected future value. Third, it should be noted that the price shock is integrated out by this weighted average of the past pseudo-value functions evaluated at random draws of $\{\tilde{p}^l\}_{l=r-N}^{r-1}$.[8]

IJC prove that under some regularity conditions, $\tilde{V}^l(., \tilde{p}^l; \theta^{*l})$ and $\hat{E}_{p'}^l V(., p'; \theta^{*l})$ converge to the true ones in probability uniformly, and the sequence of parameter vector draws converges to the true

[8] Here we propose to make one draw of price vector in each iteration. However, in practice, we find it useful to draw several price vectors in each iteration and store the average of pseudo-value functions evaluated at these draws of price vectors. We will discuss this procedure in Appendix B.

posterior distribution in total variation norm. To see the intuition behind the proof, one should note that the above procedure of obtaining a pseudo-value function is similar to applying the Bellman operator once. Intuitively, as we increase the number of iterations, the algorithm will visit an aribitrarily small neighborhood of any given parameter vector infinitely often, provided that the support of the proposal distribution covers the parameter space. As a result, the "effective" number of times we apply the Bellman operator to this neighborhood would increase with the number of iterations of the MCMC algorithm. It follows from the contraction mapping property of the Bellman operator that the pseudo-value functions converge to the true value functions. When this convergence is achieved, the IJC algorithm effectively "converges" to the conventional Bayesian MCMC algorithm. Consequently, the accepted draws of parameter vectors obtained from the IJC algorithm converge to the true posterior distribution.[9]

We should emphasize that the pseudo expected future value function defined in Equation (12) is the key innovation of IJC. In principle, this step is also applicable in classical estimation methods such as GMM and maximum likelihood.[10] However, there are at least two advantages of implementing IJC's pseudo-value function approach in Bayesian estimation. First, the non-parametric approximation in Equation (12) would be more efficient if the past pseudo-value functions are evaluated at θ^{*l}'s ($l < r$), which are randomly distributed around θ^{*r}. This can be naturally achieved by the Bayesian MCMC algorithm. On the contrary, classical estimation methods typically require minimizing/maximizing an objective function. Commonly used minimization/maximization routines (e.g., BHHH, quasi-Newton methods, etc.) tend to search over the parameter space along a particular path. We therefore believe that the approximation step proposed by IJC should perform better under the Bayesian MCMC approach.[11] Second, in the presence of unobserved consumer heterogeneity, it is common that the

[9]Strictly speaking, parameter vector draws obtained from the IJC algorithm are not a markov chain because the pseudo expected future value depends on the past pseudo-value functions, which are evaluated at $\{\theta^l\}_{l=r-N}^{r-2}$ in addition to θ^{r-1}. As a result, the proof of convergence is non-standard (Imai, Jain and Ching, 2009b).

[10]Brown and Flinn (2006) extend the implementation of this key step in estimating a dynamic model of marital status choice and investment in children using the method of simulated moments.

[11]A stochastic optimization algorithm, simulated annealing, has recently gained some attention to handle complicated

likelihood function is multi-modal even for static choice problems. In this situation, Bayesian posterior means often turn out to be better estimators of the true parameter values than classical point estimates. This is because in practice, accurately simulating a posterior is usually easier than finding the global maximum/minimum of a complex likelihood/GMM objective function.

Before we proceed any further, let us explain why we propose to store the candidate parameter vectors and their corresponding pseudo-value functions, $\{\theta^{*l}, \tilde{V}^l(., \tilde{p}^l; \theta^{*l})\}_{l=r-N}^{r-1}$, instead of the accepted ones, $\{\theta^l, \tilde{V}^l(., \tilde{p}^l; \theta^l)\}_{l=r-N}^{r-1}$. If we store $\{\theta^l, \tilde{V}^l(., \tilde{p}^l; \theta^l)\}_{l=r-N}^{r-1}$, there may be a significant portion of θ^l's that are repeated because the acceptance rate of the M-H step is usually set at around $\frac{1}{3}$. In order to conduct the non-parametric approximation for the expected future value, it is often more efficient to have a set of $\tilde{V}^l$'s evaluated at parameter vectors that span the parameter space. Since θ^{*l}'s are drawn from a candidate generating function, it is much easier for us to achieve this goal by storing $\{\theta^{*l}, \tilde{V}^l(., \tilde{p}^l; \theta^{*l})\}_{l=r-N}^{r-1}$.

3.3 Implementation of the IJC algorithm

Now we provide a step-by-step guide. The steps of implementing IJC are similar to the conventional Bayesian approach, except that we use Equation (12) to approximate the expected future value. Once the readers understand how to implement the algorithm in this simple example, they should be able to extend it to other more complicated settings. We consider two versions of the model: (i) without unobserved consumer heterogeneity, and (ii) with unobserved consumer heterogeneity. Note that the likelihood here is constructed based on pseudo alternative-specific value functions. We therefore call it *pseudo-likelihood*.

objective functions. This algorithm is an adaptation of the Metropolis-Hastings algorithm (Kirkpatrick et al. 1983; Černý 1985). The approximation step proposed by IJC should also be well-suited when researchers use simulated annealing to to maximize/minmize the objective function in classical approaches (e.g., ML and GMM). However, we should note that before a researcher starts the estimation, this method requires him/her to choose a "cooling" rate. The ideal cooling rate cannot be determined a priori. In the MCMC-based Bayesian algorithm, one does not need to deal with this nuisance parameter.

3.3.1 Homogeneous Consumers

We first present the implementation of the IJC algorithm when consumers are homogeneous in their valuations of G_j (i.e., $\sigma_{G_j} = 0$ for $j = 1, 2$). To assist readers to understand these steps, we summarize a list of notations in Table 1 and create a flowchart in Figure 4; readers may refer to them as they read through the steps below.

1. Suppose that we are at iteration r. For $r \geq N$, we store a history of past outcomes from the algorithm,

$$H^r = \{\theta^{*l}, \tilde{V}^l(., \tilde{p}^l; \theta^{*l})\}_{l=r-N}^{r-1},$$

   where N is the number of past iterations used for expected future value approximation. We will discuss how to modify the steps when $r < N$ later.

2. Draw θ^{*r} (candidate parameter vector) from a proposal distribution $q(\theta^{r-1}, \theta^{*r})$.

3. Compute the pseudo-likelihood conditional on θ^{*r}, $\rho^r(\mathbf{b^d}|\theta^{*r})$,[12]

$$\rho^r(\mathbf{b^d}|\theta^{*r}) = \prod_{i=1}^{I}\prod_{t=1}^{T_i}\prod_{j=0}^{2}\left(\frac{\exp(\tilde{V}_j^r(s_{it}^d, p_{ijt}^d; \theta^{*r}))}{\sum_{k=0}^{2}\exp(\tilde{V}_k^r(s_{it}^d, p_{ikt}^d; \theta^{*r}))}\right)^{b_{ijt}^d}.$$

   To obtain $\tilde{V}_j^r$, we need to calculate $\hat{E}_{p'}^r V(., p'; \theta^{*r})$, which is obtained using the weighted average of the past pseudo-value functions: $\{\tilde{V}^l(., \tilde{p}^l; \theta^{*l})\}_{l=r-N}^{r-1}$. The weight of each past pseudo-value function is determined by Gaussian independent kernels. For all s,

$$\hat{E}_{p'}^r V(s, p'; \theta^{*r}) = \sum_{l=r-N}^{r-1}\tilde{V}^l(s, \tilde{p}^l; \theta^{*l})\frac{K_h(\theta^{*r} - \theta^{*l})}{\sum_{k=r-N}^{r-1}K_h(\theta^{*r} - \theta^{*k})}.$$

   Substituting this into Equations (10) and (11) gives us $\tilde{V}_j^r$.

   Similarly, compute the pseudo-likelihood conditional on θ^{r-1}, $\rho^r(\mathbf{b^d}|\theta^{r-1})$. Let $\pi(\cdot)$ be the prior distribution of the parameter vector. Then we determine whether or not to accept θ^{*r} based on

[12] The superscript r for ρ^r denotes that the pseudo-likelihood is computed based on the history of the past pseudo-value functions stored up to iteration r (i.e., H^r).

the acceptance probability,

$$\min\left(\frac{\pi(\theta^{*r})\cdot\rho^r(\mathbf{b^d}|\theta^{*r})\cdot q(\theta^{*r},\theta^{r-1})}{\pi(\theta^{r-1})\cdot\rho^r(\mathbf{b^d}|\theta^{r-1})\cdot q(\theta^{r-1},\theta^{*r})},1\right).$$

Essentially, we apply the Metropolis-Hastings algorithm here by treating the pseudo-likelihood as the true likelihood. If accept, set $\theta^r=\theta^{*r}$; otherwise, set $\theta^r=\theta^{r-1}$.

4. Since we propose to store the pseudo-value functions at θ^{*r}, we need to compute $\tilde{V}^r(.,\tilde{p}^r;\theta^{*r})$.

    (a) Make one draw of $\tilde{p}_j^r$ from $N(\bar{p},\sigma_p^2)$, for $j=1,2$.

    (b) Compute $\tilde{V}_0^r(.;\theta^{*r})$, $\tilde{V}_1^r(.,\tilde{p}_1^r;\theta^{*r})$ and $\tilde{V}_2^r(.,\tilde{p}_2^r;\theta^{*r})$, using $\hat{E}_{p'}^r V(.,p';\theta^{*r})$ computed in Step 3.

    (c) Given $\tilde{V}_0^r(.;\theta^{*r})$, $\tilde{V}_1^r(.,\tilde{p}_1^r;\theta^{*r})$ and $\tilde{V}_2^r(.,\tilde{p}_2^r;\theta^{*r})$, obtain the pseudo-value function, $\tilde{V}^r(.,\tilde{p}^r;\theta^{*r})$ by using equation (9).

5. Go to iteration $r+1$.

We make three remarks here. First, when we start the algorithm (i.e., $r=1$), H^1 is empty, and we set $\hat{E}_{p'}^1 V(s,p';\theta^{*1})=0$, $\forall s\in\mathcal{S}$. Second, for $r<N$, we set $H^r=\{\theta^{*l},\tilde{V}^l(.,\tilde{p}^l;\theta^{*l})\}_{l=1}^{r-1}$, and the calculation of $\hat{E}_{p'}^r V(.,p';\theta^{*r})$ should be modified accordingly. Third, in step 3, we need to compute the pseudo-likelihood at previously accepted parameter vector, θ^{r-1}. It may first seem that this is redundant because the pseudo-likelihood at θ^{r-1} has been computed in the previous iteration, and thus we can re-use it in the current iteration. This is true in a full-solution based Bayesian MCMC algorithm, where we solve the value function exactly in each iteration. However, in the IJC algorithm, the likelihood in iteration r depends on H^r (this is why we call it pseudo-likelihood), which is updated in each iteration. Thus, in principle, we need to compute the pseudo-likelihood at θ^{r-1} using the updated set of past pseudo-value functions in iteration r.[13]

[13]In practice, however, it may not be worthwhile to compute the pseudo-likelihood at θ^{r-1} in every iteration because the set of past pseudo-value functions is updated only by one element in each iteration. Therefore, the pseudo-likelihood based on H^{r-1} could be a good approximation for the pseudo-likelihood based on H^r. We will discuss more details in Appendix B.

3.3.2 Heterogeneous Consumers

We now present the implementation of the IJC algorithm when consumers have heterogeneous valuations for the reward (i.e., $\sigma_{G_j} > 0$). If readers are not familiar with how to use the Bayesian approach to estimate static discrete choice models with unobserved heterogeneity, we suggest them to read Chapter 12 of Train (2003) or Chapter 5 of Rossi et al. (2005) first. Although we try to make the presentation self-contained, it should be easier to follow if readers have such knowledge.

Instead of just estimating the parameters that capture the distribution of the random coefficient, it is often easier to treat the random coefficient as a set of individual-specific parameters when using the Bayesian estimation approach (this is called the Hierarchical Bayes approach). We will take this approach and treat G_{ij} as individual-specific parameters in our estimation. In this case, the parameter vector can be partitioned into three parts with $\theta = (\theta_1, \theta_2, \theta_3)$, where $\theta_1 = (G_1, G_2, \sigma_{G_1}, \sigma_{G_2})$; $\theta_2 = (G_{ij}, i = 1, ..., I; j = 1, 2)$; $\theta_3 = (\alpha_1, \alpha_2, \gamma, \beta)$. In general, one can think that θ_1 captures the distribution of the individual-specific parameters, θ_2 consists of the individual-specific parameters, and θ_3 consists of parameters that are common across consumers. To simplify the discussion, we use a normal prior on G_j and an inverted gamma prior on σ_{G_j}. Based on the model specification, the prior on G_{ij} in iteration r is essentially $N(G_j^r, (\sigma_{G_j}^r)^2)$, $\forall i, j$. The prior on θ_3 can be flexible.

Each MCMC iteration mainly consists of three blocks.

(i) Draw θ_1^r, that is, for $j = 1, 2$, draw $G_j^r \sim f_G(G_j|\sigma_{G_j}^{r-1}, \theta_2^{r-1})$ and $\sigma_{G_j}^r \sim f_\sigma(\sigma_{G_j}^r|G_j^r, \theta_2^{r-1})$ (the parameters that capture the distribution of G_{ij} for the population) where f_G and f_σ are the conditional posterior distributions.

(ii) Draw θ_2^r, that is, draw individual parameters $G_{ij}^r \sim f_i(G_{ij}|\mathbf{b_i^d}, \theta_1^r, \theta_3^{r-1})$ by the Metropolis-Hastings algorithm. We use G_{ij}^{*r} to denote the candidate value for G_{ij}^r.

(iii) Draw $\theta_3^r \sim f_{\theta_3}(.|\mathbf{b^d}, \theta_2^r)$ using the Metropolis-Hastings algorithm. Note that this block is similar to the steps described in the homogeneous case.

Heterogeneity introduces an additional complication that the expected future values need to be approximated for each consumer. As before, this is achieved by taking weighted average of past pseudo-value functions based on the distance of the current parameter vector to the past parameter vector. It should be emphasized that conditional on θ_2 (i.e., G_{ij}'s), the value functions (and hence the likelihood functions) do not depend on θ_1. Consequently, when calculating the likelihoods in block (ii) and (iii), the "effective" parameter vector only consists of (θ_2, θ_3). When approximating the expected future values for each conusmer, we need to combine the common and individual-specific kernel weights to produce the weighted average of past pseudo-value functions.

We now describe the details of the estimation steps. To assist the readers to follow these steps, we summarize a list of notations in Table 2 and provide a flowchart in Figure 5; readers may refer to them as they read through the steps below. Steps 2-3 belong to block (i), step 4 belongs to block (ii) and step 5 belongs to block (iii). Note that we only describe steps 2 and 3 briefly here because they are standard. For the details of these two steps, we refer readers to Chapter 12 of Train (2003).

1. Suppose that we are at iteration r. We start with

$$H^r = \{\theta_3^{*l}, \{G_i^{*l}, \tilde{V}^l(., \tilde{p}^l; G_i^{*l}, \theta_3^{*l})\}_{i=1}^{I}\}_{l=r-N}^{r-1},$$

where I is the number of consumers; N is the number of past iterations used for the expected future value approximation.

2. For each $j = 1, 2$, draw G_j^r (population mean of G_{ij}) from the posterior density (normal) conditional on $\sigma_{G_j}^{r-1}$ and $\{G_{ij}^{r-1}\}_{i=1}^{I}$.

3. For each $j = 1, 2$, draw $\sigma_{G_j}^r$ (population variance of G_{ij}) from the posterior density (inverted gamma) conditional on G_j^r and $\{G_{ij}^{r-1}\}_{i=1}^{I}$.

4. Let $\mathbf{b_i^d} \equiv \{b_{it}^d, \forall t\}$. For each $i = 1, ..., I$, draw G_i^r from its posterior distribution conditional on $(\mathbf{b_i^d}, G_j^r, \sigma_{G_j}^r, \theta_3^{r-1})$, which is $f_i(G_i|\mathbf{b_i^d}, G_j^r, \sigma_{G_j}^r, \theta_3^{r-1}) \propto \pi(G_i|G_j^r, \sigma_{G_j}^r) \cdot \rho_i(\mathbf{b_i^d}|G_i, \theta_3^{r-1})$. Since there is no easy way to draw from this posterior, we use the M-H algorithm.

   (a) Draw G_{ij}^{*r} from the proposal distribution $q(G_{ij}^{r-1}, G_{ij}^{*r})$ (e.g., $G_{ij}^{*r} \sim N(G_{ij}^{r-1}, \sigma^2)$), where G_{ij}^{*r} is a candidate value of G_{ij}^r.

   (b) Compute the pseudo-likelihood for consumer i at G_i^{*r}, i.e., $\rho_i^r(\mathbf{b_i^d}|G_i^{*r}, \theta_3^{r-1})$. The pseudo-likelihood is expressed as

$$\rho_i^r(\mathbf{b_i^d}|G_i^{*r}, \theta_3^{r-1}) = \prod_{t=1}^{T_i}\prod_{j=0}^{2}\left(\frac{\exp(\tilde{V}_j^r(s_{it}^d, p_{ijt}^d; G_i^{*r}, \theta_3^{r-1}))}{\sum_{k=0}^{2}\exp(\tilde{V}_k^r(s_{it}^d, p_{ikt}^d; G_i^{*r}, \theta_3^{r-1}))}\right)^{b_{ijt}^d}.$$

   To obtain $\tilde{V}_j^r$, we need $\hat{E}_{p'}^r V(., p'; G_i^{*r}, \theta_3^{r-1})$, which is obtained by a weighted average of $\{\tilde{V}^l(., \tilde{p}^l; G_i^{*l}, \theta_3^{*l})\}_{l=r-N}^{r-1}$, treating G_i as one of the parameters when computing the weights. In the case of independent kernels, for all s,

$$\hat{E}_{p'}^r V(s, p'; G_i^{*r}, \theta_3^{r-1}) = \sum_{l=r-N}^{r-1}\tilde{V}^l(s, \tilde{p}^l; G_i^{*l}, \theta_3^{*l})\frac{K_h(\theta_3^{r-1} - \theta_3^{*l})K_h(G_i^{*r} - G_i^{*l})}{\sum_{k=r-N}^{r-1} K_h(\theta_3^{r-1} - \theta_3^{*k})K_h(G_i^{*r} - G_i^{*k})}.$$ [14]

   We repeat the same step and obtain the pseudo-likelihood conditional on $(G_i^{r-1}, \theta_3^{r-1})$, $\rho_i^r(\mathbf{b_i^d}|G_i^{r-1}, \theta_3^{r-1})$.

   Then, we determine whether or not to accept G_i^{*r}. The acceptance probability, λ, is given by

$$\lambda = \min\left(\frac{\pi(G_i^{*r}|G_j^r, \sigma_{G_j}^r) \cdot \rho_i^r(\mathbf{b_i^d}|G_i^{*r}, \theta_3^{r-1}) \cdot q(G_i^{*r}, G_i^{r-1})}{\pi(G_i^{r-1}|G_j^r, \sigma_{G_j}^r) \cdot \rho_i^r(\mathbf{b_i^d}|G_i^{r-1}, \theta_3^{r-1}) \cdot q(G_i^{r-1}, G_i^{*r})}, 1\right).$$

   If accept, set $G_i^r = G_i^{*r}$; otherwise, set $G_i^r = G_i^{r-1}$.[15]

   (c) Repeat (a) & (b) for all i.

[14] Note that $\{K_h(\theta_3^{r-1} - \theta_3^{*l})\}_{l=r-N}^{r-1}$ is common across consumers. Therefore, one can calculate it outside the loop that indexes consumers when programming this part.

[15] Note that if $q(.,.)$ is symmetric, the expression of the acceptance probability will be simplified to $\lambda = \min\left(\frac{\pi(G_i^{*r}|G_j^r, \sigma_{G_j}^r) \cdot \rho_i^r(\mathbf{b_i^d}|G_i^{*r}, \theta_3^{r-1})}{\pi(G_i^{r-1}|G_j^r, \sigma_{G_j}^r) \cdot \rho_i^r(\mathbf{b_i^d}|G_i^{r-1}, \theta_3^{r-1})}, 1\right)$.

5. Use the Metropolis-Hastings algorithm to draw $\theta_3^r = (\alpha_1^r, \alpha_2^r, \gamma^r, \beta^r)$ conditional on G_{ij}^r.

   (a) Draw $\theta_3^{*r} = (\alpha_1^{*r}, \alpha_2^{*r}, \gamma^{*r}, \beta^{*r})$ (candidate parameter vector).

   (b) We then compute the pseudo-likelihood conditional on θ_3^{*r} and $\{G_i^r\}_{i=1}^I$, based on the pseudo alternative-specific value functions. The pseudo-likelihood, $\rho^r(\mathbf{b^d}|\{G_i^r\}_{i=1}^I, \theta_3^{*r})$, is expressed as

$$\rho^r(\mathbf{b^d}|\{G_i^r\}_{i=1}^I, \theta_3^{*r}) = \prod_{i=1}^{I}\prod_{t=1}^{T_i}\prod_{j=0}^{2}\left(\frac{\exp(\tilde{V}_j^r(s_{it}^d, p_{ijt}^d; G_i^r, \theta_3^{*r}))}{\sum_{k=0}^{2}\exp(\tilde{V}_k^r(s_{it}^d, p_{ikt}^d; G_i^r, \theta_3^{*r}))}\right)^{b_{ijt}^d}.$$

   To obtain $\tilde{V}_j^r$, we need to calculate $\hat{E}_{p'}^r V(., p'; G_i^r, \theta_3^{*r})$, which is a weighted average of the past value functions, $\{\tilde{V}^l(., \tilde{p}^l; G_i^{*l}, \theta_3^{*l})\}_{l=r-N}^{r-1}$. In computing the weights for past pseudo-value functions, we treat G_i as a parameter. In the case of independent kernels, Equation (12) becomes

$$\hat{E}_{p'}^r V(s, p'; G_i^r, \theta_3^{*r}) = \sum_{l=r-N}^{r-1}\tilde{V}^l(s, \tilde{p}^l; G_i^{*l}, \theta_3^{*l})\frac{K_h(\theta_3^{*r} - \theta_3^{*l})K_h(G_i^r - G_i^{*l})}{\sum_{k=r-N}^{r-1}K_h(\theta_3^{*r} - \theta_3^{*k})K_h(G_i^r - G_i^{*k})}.$$ [16]

   We repeat the same step and obtain the pseudo-likelihood conditional on θ_3^{r-1} and $\{G_i^r\}_{i=1}^I$, $\rho^r(\mathbf{b^d}|\{G_i^r\}_{i=1}^I, \theta_3^{r-1})$.

   Then, we determine whether or not to accept θ^{*r}. The acceptance probability, λ, is given by

$$\lambda = \min\left(\frac{\pi(\theta_3^{*r})\cdot\rho^r(\mathbf{b^d}|\{G_i^r\}_{i=1}^I, \theta_3^{*r})\cdot q(\theta_3^{*r}, \theta_3^{r-1})}{\pi(\theta_3^{r-1})\cdot\rho^r(\mathbf{b^d}|\{G_i^r\}_{i=1}^I, \theta_3^{r-1})\cdot q(\theta_3^{r-1}, \theta_3^{*r})}, 1\right).$$

   If accept, set $\theta_3^r = \theta_3^{*r}$; otherwise, set $\theta_3^r = \theta_3^{r-1}$.

6. Computation of the pseudo-value function, $\{\tilde{V}^r(., \tilde{p}^r; G_i^{*r}, \theta_3^{*r})\}_{i=1}^I$.

   (a) Make one draw of prices, $\tilde{p}^r$, from the price distribution.

   (b) Compute the pseudo expected future value at $(G_i^{*r}, \theta_3^{*r})$. For all s,

$$\hat{E}_{p'}^r V(s, p'; G_i^{*r}, \theta_3^{*r}) = \sum_{l=r-N}^{r-1}\tilde{V}^l(s, \tilde{p}^l; G_i^{*l}, \theta_3^{*r})\frac{K_h(\theta_3^{*r} - \theta_3^{*l})K_h(G_i^{*r} - G_i^{*l})}{\sum_{k=r-N}^{r-1}K_h(\theta_3^{*r} - \theta_3^{*k})K_h(G_i^{*r} - G_i^{*k})}.$$ [17]

[16] Note that $\{K_h(\theta_3^{*r} - \theta_3^{*l})\}_{l=r-N}^{r-1}$ is common across consumers. Therefore, one can compute it outside the loop that indexes consumers to save computational time.

[17] Note that both the common and individual-specific parts of the weights have been already computed seprately in Steps 4 and 5, and can thus be re-used here.

We should emphasize that there is a subtle difference between this step and step 5(b) above, which evaluates $\hat{E}^r_{p'}V$ at G^r_i instead of G^{*r}_i.

(c) Compute $\tilde{V}^r_0(.;\theta^{*r}_3)$, $\tilde{V}^r_1(.,\tilde{p}^r_1;G^{*r}_i,\theta^{*r}_3)$ and $\tilde{V}^r_2(.,\tilde{p}^r_2;G^{*r}_i,\theta^{*r}_3)$, using the pseudo expected future values computed in (b).

(d) Given $\tilde{V}^r_0(.;\theta^{*r}_3)$, $\tilde{V}^r_1(.,\tilde{p}^r_1;G^{*r}_i,\theta^{*r}_3)$ and $\tilde{V}^r_2(.,\tilde{p}^r_2;G^{*r}_i,\theta^{*r}_3)$, obtain the pseudo-value function, $\tilde{V}^r(.,\tilde{p}^r;G^{*r}_i,\theta^{*r}_3) = \log[\exp(\tilde{V}^r_0(.;\theta^{*r}_3)) + \exp(\tilde{V}^r_1(.,\tilde{p}^r_1;G^{*r}_i,\theta^{*r}_3)) + \exp(\tilde{V}^r_2(.,\tilde{p}^r_2;G^{*r}_i,\theta^{*r}_3))]$.

(e) Repeat (b)-(d) for all i.

7. Go to iteration $r+1$.

We should make five remarks regarding the above procedure. First, in step 5(b), one can re-use the individual pseudo-likelihoods computed in step 4(b) to form the joint pseudo-likelihood conditional on θ^{r-1}_3 and $\{G^r_i\}^I_{i=1}$. Second, note that when implementing step 5, it could be more efficient to separate the parameters by blocks if the acceptance rate is low. The trade-off is that when implementing this step by blocks, we also increase the number of expected future value approximation calculations and likelihood evaluations. Third, note that the kernel weights based on θ_3 are common across all consumers, and can thus be pre-computed prior to step 4. This can save some computational time.

Fourth, we should also point out that the above procedure is much more memory and computationally demanding compared with the homogeneous case, because now we need to store and compute $I \times N$ instead of N past pseudo-value functions. If computer memory is a constraint faced by a researcher, an alternative procedure is to randomly pick one consumer's pseudo-value function to store for each iteration l. When approximating the expected future value for each consumer, one can then treat all the past pseudo-value functions stored as a common pool to form the kernel approximation. In Appendix C, we explain how to implement this approach.

Fifth, it is worth emphasizing that when estimating a model with unobserved heterogeneity, a key difference from the homogeneous model is that one needs to draw individual-specific parameters. When the number of individuals is large, this part of the MCMC algorithm can be slow even for static choice models (e.g., Rossi et al. 2005). When estimating a DDP model, this part of the MCMC algorithm is even more computationally intensive than estimating a static choice model, because the expected future value functions now need to be computed individual-by-individual. This feature suggests that the IJC method may be particularly useful in estimating models with unobserved heterogeneity. Suppose that we can save one second by using the IJC method to approximate one expected future value function in the inner loop. If there are 1000 individuals in the sample, we can then approximately save 1000 seconds per MCMC outer loop iteration. In our Monte Carlo exercises, we will illustrate the potential time savings that one can achieve by using the IJC method.

3.4 Choice of kernel's bandwidth and N

The IJC method relies on classical non-parametric methods to approximate the expected future values using the past pseudo-value functions generated by the algorithm. One practical problem of nonparametric regression analysis is that the data becomes increasingly sparse as the dimensionality of the explanatory variables increases (note that the number of parameters of the DDC model in IJC corresponds to the number of explanatory variables in the traditional non-parameteric regression). For instance, ten points that are uniformly distributed in the unit cube are more scattered than ten points distributed uniformly in the unit interval. Thus the number of observations available to provide information about the local behavior of a function becomes small with large dimension. The curse of dimensionality of this non-parametric technique (in terms of number of parameters) could be something that we need to worry about.[18]

However, in implementing the IJC algorithm, the nature of this problem is different from the standard non-parametric estimation. Unlike a standard estimation problem where an econometrician cannot control the sample size of the data set, we can control the sample size for our nonparametric

[18] This curse of dimensionality problem is different from that of solving for a dynamic programming model, where it refers to the size of the state space increasing exponentially with the number of state variables and linearly with the number of values for each state variable.

regressions by storing/using more past pseudo-value functions (by increasing N). In practice, it is possible that N may need to increase with the number of parameters in the model to ensure that the IJC algorithm performs well. As a result, it would also take more time to compute one iteration if the model becomes more complicated.

The discussion above suggests that the convergence rate is typically inversely related to the number of dimensions. But the situation that we face now is more subtle for two reasons. First, it is likely that the convergence rate is model specific, as the shape of the likelihood function is also model specific. Second, it should also depend on the data sample size. In general, when a model is well-identified and the data has sufficient variation, the posterior variance of the parameters decreases with the sample size. This suggests that when the MCMC converges, the simulated parameter values would move within a small neighborhood of the posterior means. This implies that the set of past expected pseudo-value functions would be evaluated at parameter vectors that are concentrated in a small neighborhood in the parameter space. We expect that this should alleviate the curse of dimensionality problem.

It is worth discussing the impact of N on the estimation results. If we increase N, more older past pseudo-value functions will be used in the approximation. This may result in slow improvements in the approximation, and may slow down the MCMC convergence rate. If we decrease N, more recent and accurate pseudo-value functions will be used in the approximation. However, by decreasing N, the variance of the pseudo expected future values may increase. This may result in a higher standard deviation of the posterior distribution for some parameters. One way of mitigating this trade-off is to set N to be small at the beginning of the IJC algorithm and let N increase during the MCMC iterations. In this way, we can achieve a faster convergence and more stable posterior distributions at the same time. Another way to address this issue is to weight the past N expected pseudo-value functions differently so that the more recent expected pseudo-value functions receive higher weights (because they should be more accurate approximations). In one Monte Carlo experiment conducted in the next section, we show some evidence about the impact of N on the estimation results.

An obvious question that would likely come to researchers' mind is: How do we choose N and the bandwidth (h)? We believe that any suggested guidelines should ensure that the pseudo-value function gives us a good proxy for the true value function. We suggest that researchers check the distance between the pseudo-value function and the exact value function during the estimation, and adjust N and h within the iterative process. For instance, researchers can store a large set of past pseudo-value functions (i.e., large N), and use the most recent $N' < N$ of them to do the approximation. This has the advantage that researchers can immediately increase N' if they discover that the approximation is not good enough. Researchers can start the algorithm with a small N' (say $N' = 100$), and an arbitrary bandwidth (say 0.01). Every 1,000 iterations, they can compute the means of the MCMC draws, $\bar{\bar{\theta}}$, and solve for the exact value function at $\bar{\bar{\theta}}$ numerically. Then they can compare the distance between the pseudo-value function and the exact value function at $\bar{\bar{\theta}}$. If the distance is larger than what the researcher would accept, N' should be increased. Researchers can then use this new larger set of past pseudo-value functions to compute summary statistics and apply a standard optimal bandwidth formula, e.g., Silverman's rule of thumb (Silverman 1986, p.48), to set h. Of course, the cost of storing a large number of past pseudo-value function is that it requires more memory. But thanks to the advance of computational power, the cost of memory is decreasing rapidly over time these days. Hence, we expect that memory would become less of a constraint in the near future. This suggestion would require us to solve for the DDP model exactly once every 1,000 iterations. For complicated DDP models with random coefficients, this could still be computationally costly. But even in this case, one could simply compare the pseudo-value function and the exact value function at a small number of simulated heterogeneous parameter vectors, say 5. This would be equivalent to solving 5 homogeneous DDP models numerically and should be feasible even for complicated DDP models.

4 Estimation Results

To illustrate the performance of the IJC algorithm and investigate some of its properties, we conduct three sets of Monte Carlo experiments. For each experiment, the simulated sample size is 1,000 consumers and 100 periods. We use the Gaussian kernel to weigh the past pseudo-value functions when approximating the expected future values. The total number of MCMC iterations is 10,000, and we report the posterior distributions of parameters based on the 5,001-10,000 iterations. For all experiments, the following parameters are fixed and not estimated: $\bar{S}_1, \bar{S}_2, \bar{p} = 1.0$, and $\sigma_p = 0.3$. In the first set of experiments, we check whether IJC is able to recover the true parameter values of the model presented here.

4.1 Ability of Recovering True Parameter Values

We first estimate a version of the model *without unobserved heterogeneity.* When simulating the data, we set $\bar{S}_1 = 2$, $\bar{S}_2 = 4$, $\sigma_{G_1} = \sigma_{G_2} = \alpha_1 = \alpha_2 = 0$, $G_1 = 1.0$, $G_2 = 5.0$, $\gamma = -1.0$, and $\beta = 0.6$ or 0.8. Our goal is to estimate $\alpha_1, \alpha_2, G_1, G_2, \gamma$, and β, treating other parameters as known. To ensure that $\beta < 1$ during the estimation, we transform it as $\beta = \frac{1}{1+exp(\phi)}$ and estimate ϕ instead. For all parameters, we use the flat prior (i.e., $\pi(\theta) = 1, \forall\theta$). In addition, we use a normal random-walk proposal function (i.e., $\theta^{*r} \sim N(\theta^{r-1}, \sigma^2)$). This implies that the acceptance probability stated in step 3 of section 3.3.1 becomes $\min\left(\frac{\rho^r(\theta^{*r})}{\rho^r(\theta^{r-1})}, 1\right)$ because the proposal function is symmetric. Table 3 summarizes the estimation results, and Figure 6 plots the MCMC draws of parameters for the case of $\beta = 0.8$. The posterior means and standard deviations show that the IJC algorithm is able to recover the true parameter values well. Moreover, it appears that the MCMC draws converge after 2,000 iterations.

Now we estimate a version of the model *with unobserved heterogeneity.* For simplicity, we only allow for consumer heterogeneity in G_2 (i.e., we fix $\sigma_{G_1} = 0$). The data is simulated based on the following parameter values: $\bar{S}_1 = 2$, $\bar{S}_2 = 4$, $\alpha_1 = \alpha_2 = 0.0$, $G_1 = 1.0$, $G_2 = 5.0$, $\sigma_{G_1} = 0.0$, $\sigma_{G_2} = 1.0$, $\gamma = -1.0$, and $\beta = 0.6$ or 0.8. As before, we transform β by the logit formula, i.e., $\beta = \frac{1}{1+exp(\phi)}$. Our goal is to estimate $\alpha_1, \alpha_2, G_1, G_2, \sigma_{G_2}, \gamma$, and β, treating other parameters as known. For α_1, α_2, G_1, γ, and ϕ, we use flat prior. For G_2, we use a diffuse normal prior (i.e., setting the standard deviation of the prior to ∞). For σ_{G_2}, we use a diffuse inverted gamma prior, $\text{IG}(\nu_0, s_0)$ (i.e., setting $s_0 = 1$, $\nu_0 \to 1$). In step 4(b) of section 3.3.2, we use $N(G_j^r, (\sigma_{G_j}^r)^2)$ as the proposal distribution for generating G_{ij}^{*r}. Given this proposal distribution, the probability of accepting G_{ij}^{*r} then becomes $\lambda = \min\left(\frac{\rho_i^r(\mathbf{b_i^d}|G_i^{*r}, \theta_3^{r-1})}{\rho_i^r(\mathbf{b_i^d}|G_i^{r-1}, \theta_3^{r-1})}, 1\right)$. In each iteration, we also implement the procedure in Appendix C by randomly selecting one consumer's pseudo-value function to compute and store, and use a common pool of past pseudo-value functions to approximate the expected future values for all consumers. Table 4 shows the estimation results, and Figure 7 plots the simulated draws of parameters for $\beta = 0.8$. Again, the IJC algorithm is able to recover the true parameter values well. The MCMC draws appear to converge after 2,000 iterations for most of the parameters except G_1, which takes about 3,000 iterations to achieve convergence.

4.2 Potential Reduction in Computation Time

In the second set of experiments, we compare the computational time between the IJC and conventional Bayesian approaches. To learn more about the potential gain of IJC in terms of computational time, we compute the time per iteration and compare the IJC algorithm with the full-solution based Bayesian MCMC algorithm for both homogeneous model and heterogeneous model. In the full-solution based Bayesian algorithm, we set $M = 100$ for $\{\tilde{p}^m\}_{m=1}^M$, that is, we use 100 simulated draws of prices to integrate out the future price. For each model, we study three cases: $\beta = 0.6, 0.8$ and 0.98. Table 5 summarizes the results based on the average computation time based on 1,000 iterations. The estimation is done based on a C program compiled by the gcc compiler, and run in a linux workstation with Intel Core 2 Duo E4400 2GHz processor.

In the homogeneous model, the computation for the full-solution based Bayesian is faster for $\beta = 0.6$ and 0.8. This is because: (i) when β is small, solving for a contraction mapping to get the exact value function is not that costly compared with computing the weighted average of 1,000 past pseudo-value functions; (ii) full-solution based Bayesian approach does not need to compute the pseudo-likelihood conditional on previously accepted parameter vector (step 3 in the homogeneous case, and step 4(b) in the heterogeneous case).[19] However, when $\beta = 0.98$, the IJC algorithm is 40% faster than the full-solution algorithm. This is because in the full-solution based Bayesian algorithm, the number of iterations required for convergence in a contraction mapping increases with β (i.e., the modulus), and hence the computation time of the inner loop will generally increase with β. However, the computation time of the inner loop will not be influenced by the value of β in the IJC algorithm.

In the heterogeneous model, we can see the advantage of the IJC algorithm much clearer. When $\beta = 0.6$, the IJC algorithm is 50% faster than the full-solution based Bayesian algorithm; when $\beta = 0.8$, it is about 200% faster; when $\beta = 0.98$, it is about 3,000% faster. In particular, it is clear that average computational time per iteration basically remains unchanged in the IJC algorithm. For the full solution based method, the computational time per iteration increases exponentially in β because, roughly speaking, we need to solve for the DDP model for each individual. If there are 1,000 individuals, the computational time is approximately equal to *time for solving the value function once* multiplying by one thousand. For the heterogeneous model, with $\beta = 0.98$, it would take about 70 days ($\simeq 613 \times 10,000$ seconds) to run the full-solution based Bayesian MCMC algorithm for 10,000 iterations.[20] Using the IJC algorithm, it would take less than 2.5 days ($\simeq 18.4 \times 10,000$ seconds) to obtain 10,000 iterations.

4.3 The Role of N

As discussed above, one issue in using the IJC algorithm is how to choose N, the number of the past pseudo-value functions. In the third set of Monte Carlo experiments, using the homogeneous model with $\beta = 0.98$, we investigate how changes in N influence the speed of convergence and the posterior

[19]In this exercise, we computed the pseudo-likelihood conditional on previously accepted parameter vector every time a candidate parameter vector was rejected.

[20]Depending on the convergence rate, the number of iterations required for Bayesian estimation could be higher than 10,000.

distributions. We simulate the data using the following set of parameter values: $\bar{S}_1 = 5$, $\bar{S}_2 = 10$, $\alpha_1 = \alpha_2 = 0$, $G_1 = 1$, $G_2 = 10$, $\gamma = -1$, $\bar{p} = 1.0$, and $\sigma_p = 0.3$. Our goal is to compare the performance of the IJC algorithm using $N = 100$ and $1,000$. Table 6 shows the posterior distributions of the parameters. The results show that the posterior means are very similar for both cases. But the standard deviations for G_1 and G_2 are smaller for $N = 1,000$. This is consistent with our arguments earlier in section 3.4 – when using more pseudo-value functions to do the approximation, the variance of the approximation should become smaller. To see how the speed of convergence changes with N, we plot the MCMC samplers for α_1 and α_2 in Figure 8, and G_1 and G_2 in Figure 9. It can be seen that when $N = 100$, the speed of convergence is faster, but the paths also fluctuate more. Again, this is consistent with our discussion in section 3.4.

Finally, note that when $\beta = 0.98$, the true parameter values are recovered less precisely, in particular, α_j and G_j. This is due to the identification problem that we discussed earlier: When β is close to 1, changing G_j would simply shift the choice probabilities almost equally across s, similar to changing α_j.

5 Extension and Discussions

We now discuss how to extend the IJC algorithm to allow for continuous state space. We will also compare the IJC algorithm with two other estimation approaches by Ackerberg (2009) and Keane and Wolpin (1994), which also rely on approximating the likelihood.

5.1 Continuous State Space

The state space of the dynamic store choice model described earlier is the number of stamps collected for each supermarket chain, which takes a finite number of values. In many marketing and economics applications, however, we have to deal with continuous state variables such as prices, advertising expenditures, capital stocks, etc. IJC also describe how to extend the algorithm to allow for continuous state variables, by combining it with the random grid approximation proposed by Rust (1997). To illustrate how it works, we consider the homogeneous model here.

Consider a modified version of the dynamic store choice model without unobserved consumer heterogeneity. Suppose that prices set by the two supermarket chains follow a first-order Markov process (instead of an iid process across time): $f(p'|p;\theta_p)$, where θ_p is the vector of parameters for the price process. In this setting, the expected value functions in Equations (10) and (11) are conditional on current prices, $E_{p'}[V(s',p';\theta)|p]$. Clearly, prices are part of the state spaces and they are continuous. To handle this situation, for each iteration r, we can make one draw of prices, $\tilde{p}^r = (\tilde{p}_1^r, \tilde{p}_2^r)$, from a distribution. For example, we can define this distribution as uniform on $[\underline{p}, \bar{p}]^2$ where $\underline{p}$ and $\bar{p}$ are the lowest and highest observed prices, respectively. Then, we compute the pseudo-value function at $\tilde{p}^r$, $\tilde{V}^r(s,\tilde{p}^r;\theta^{*r})$ for all s. Thus, H^r in step 1 of section 3.3.1 needs to be changed to

$$H^r = \{\theta^{*l}, \tilde{p}^l, \tilde{V}^l(.,\tilde{p}^l;\theta^{*l})\}_{l=r-N}^{r-1}.$$

The expected value function given s', p, and θ^{*r} is then approximated as follows.

$$\hat{E}^r_{p'}[V(s',p';\theta^{*r})|p] = \sum_{l=r-N}^{r-1} \tilde{V}^l(s',\tilde{p}^l;\theta^{*l}) \frac{K_h(\theta^{*r}-\theta^{*l}) f(\tilde{p}^l|p;\theta_p)}{\sum_{k=r-N}^{r-1} K_h(\theta^{*r}-\theta^{*k}) f(\tilde{p}^k|p;\theta_p)}. \tag{13}$$

Compared with Equation (12), the main difference is that in Equation (13), the transition density of prices is also used in creating the weights.

Unlike Rust's random grid approximation which fixes the number of grid points throughout the estimation, the random grid points here change at each MCMC iteration. Most importantly, one can easily adjust the precision of the approximation in this approach because the total number of random grid points can be made arbitrarily large by increasing N. We should also point out that if researchers simply apply the conventional Rust's random grid approximation with M fixed grid points in the IJC algorithm, they need to compute the pseudo-value functions at M grid points in each iteration. As a result, the effective size of the state space will become $M \times \bar{S}_1 \times \bar{S}_2$, while the IJC's random grid approach will keep it at $\bar{S}_1 \times \bar{S}_2$. The main advantage of the IJC's random grid algorithm comes from the fact that the integration of the continuous state variables (prices in this case) is already incorporated when we compute the weighted average of the past pseduo-value functions. This allows us to compute the pseudo-value function at only one grid point, $\tilde{p}^r$, in each iteration.

5.2 Comparison with other approximation approaches

5.2.1 Ackerberg (2009)

Recently, Ackerberg (2009) proposed an alternative estimation approach that makes use of importance sampling and change of variables techniques. To use this approach, one would simulate a set of parameter vectors, $\{\tilde{\theta}^m\}_{m=1}^M$, solve the model and obtain the sub-likelihood at each $\tilde{\theta}^m$ upfront. The likelihood is a weighted average of these sub-likelihoods, where the weights are partly determined by the importance sampling density chosen. When searching over the parameter space to maximize the likelihood, one only needs to change the weights associated with each sub-likelihood, and does not have to solve the model at different simulated parameter vectors again in each outer loop iteration. As a result, Ackerberg's approach also has the potential to significantly reduce the computational burden of estimating DDP models. This method has been applied to several DDP problems, e.g., Hartmann (2006), Pantano (2008).

Ackerberg's approach is more general than IJC in the sense that it can be applied to a larger class of DDP models (including finite horizon models with deterministic transition of state variables), or other structural models which do not rely on using the contraction mapping arguments to solve the model. However, often times, to apply his method, one needs to allow all parameters of the model to have continuous random effects, and this could lead to identification problems for some of the parameters. Moreover, a poor choice of importance sampling density function could lead to very large simulation errors in this approach. Ackerberg (2009) discusses some practical ways to address this issue.

The IJC approach is most useful when applying to stationary DDP models. In principle, the IJC method should also be applicable to any structural models which can be solved using contraction mapping arguments. For instance, Berry et al. (1995) propose to use the method of successive approximation to back out the unobserved product characteristics in their nested fixed point GMM approach when estimating static demand models using product level data; some game theoretical models (e.g., bargaining models) which consist of a unique equilibrium, the method of successive approximation is also a common way to numerically solve for the equilibrium. The IJC method can be potentially applicable in these situations as well.

It should also be noted that the IJC method can be used in both Bayesian and classical estimation, while Ackerberg's approach is less suitable for Bayesian approach. This is because in the Bayesian approach, one does not have to marginalize the unobserved heterogeneity. Rather, it is more straightforward to treat the random coefficient as a set of individual-specific parameters in the Hierarchical Bayes approach. For models with complicated likelihoods (i.e., with multiple modes), the MCMC based IJC approach might lead to more stable estimation results because, in practice, simulating draws from posterior distributions using MCMC appears to be easier than searching for the global maximum in such situations.

5.2.2 Keane and Wolpin (1994)

It should be highlighted that the IJC method is not particularly suited for finite horizon DDP models. When solving a finite horizon DDP model, the typical approach is to simply use backward induction and start solving the model from the terminal period, T. Unlike the infinite horizon DDP models, the computational burden of solving finite horizon DDP models does not depend on the discount factor, β. Instead, it solely depends the size of the state space. One of the most commonly used methods to reduce the computational burden for this type of models is by Keane and Wolpin (1994). In their approach, the expected future value functions are evaluated at a subset of the state points and some methods of interpolation are used to evaluate the expected future value functions at other values of the state space. Keane and Wolpin (1994) provide Monte Carlo evidence that suggests their approximation would converge to the true solution as the subset of the state points that are chosen increases. This method has proven to be very effective in reducing the computational time for finite horizon DDP models with a large state space (e.g., Keane and Wolpin 1997; Erdem and Keane 1996; Ackerberg 2003). Researchers who face this type of DDP problems could consider applying Keane and Wolpin (1994).

6 Conclusion

In this paper, we discuss how to implement the IJC method using a dynamic store choice model. For illustration purpose, the specification of the model is relatively simple. We believe that this new method is quite promising in estimating DDP models. Osborne (2008) has successfully applied this method to estimate a much more detailed consumer learning model. The IJC method allows him to incorporate more general unobserved consumer heterogeneity than the previous literature, and draw inference on the relative importance of switching costs, consumer learning and consumer heterogeneity in explaining customers' persistent purchase behavior observed in scanner panel data. Brown and Flinn (2006) applies the IJC idea and use GMM to estimate a model that investigates how marital states affect parents' incentive to invest in their children. Ching et al. (2009) have also successfully estimated a learning and forgetting model where consumers are forward-looking.

It should also be noted that there are many kernels that one could use in forming a non-parametric approximation for the expected future values. IJC discuss their method in terms of the Gaussian kernel. Norets (2009) extends IJC's method by using the "nearest neighbors" kernel instead of Gaussian kernel, and allowing the error terms to be serially correlated. At this point, the relative performances of different kernels in this setting are still largely unknown. It is possible that for models with certain features, the Gaussian kernel performs better than other kernels in approximating the pseudo-value function, while other kernels may outperform the Gaussian kernel for models with other features. More research is needed to document the pros and cons of different kernels, and provide guidance in the choice of kernel when implementing the IJC method.

Bayesian inference has allowed researchers and practitioners to develop more realistic static choice models in the last two decades. It is our hope that the new method presented here and its extensions would allow us to take another step to develop more realistic DDP models and ease the burden of estimating them in the near future.

References

Ackerberg, Daniel A. 2003. Advertising, Learning, and Consumer Choice in Experience Good Markets: An Empirical Examination. *International Economic Review* **44**(3) 1007–1040.

Ackerberg, Daniel A. 2009. A New Use of Importance Sampling to Reduce Computational Burden in Simulation Estimation. *Quantitative Marketing and Economics* **7**(4) 343–376.

Aguirregabiria, Victor, Pedro Mira. 2002. Swapping the Nested Fixed Point Algorithm: A Class of Estimators for Discrete Markov Decision Models. *Econometrica* **70**(4) 1519–1543.

Albert, James H., Siddhartha Chib. 1993. Bayesian Analysis of Binary and Polychotomous Response Data. *Journal of the American Statistical Association* **88** 669–679.

Allenby, Greg M. 1994. An Introduction to Hierarchical Bayesian Modeling. Tutorial Notes, Advanced Research Techniques Forum, American Marketing Association.

Allenby, Greg M., Peter J. Lenk. 1994. Modeling Household Purchase Behavior with Logistic Normal Regression. *Journal of the American Statistical Association* **89** 1218–1231.

Bellman, Richard. 1957. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA.

Berry, Steven T., James Levinshohn, Ariel Pakes. 1995. Automobile Prices in Market Equilibrium. *Econometrica* **63**(4) 841–890.

Brown, Meta, Christopher J. Flinn. 2006. Investment in Child Quality Over Marital States. Working paper, Department of Economics, New York University.

Černý, V. 1985. Thermodynamical Approach to the Travelling Salesman Problem: An Efficient Simulation Algorithm. *Journal of Optimization Theory and Applications* **45**(1) 41–51.

Ching, Andrew T. 2010. A Dynamic Oligopoly Structural Model for the Prescription Drug Market after Patent Expiration. *International Economic Review* **54**(4) 1175–1207.

Ching, Andrew T., Susumu Imai, Masakazu Ishihara, Neelam Jain. 2009. A Dynamic Model of Consumer Learning and Forgetting. Work-in-progress, Rotman School of Management, University of Toronto.

Crawford, Gregory S., Matthew Shum. 2005. Uncertainty and Learning in Pharmaceutical Demand. *Econometrica* **73**(4) 1137–1174.

Diermeier, Daniel, Michael P. Keane, Antonio M. Merlo. 2005. A Political Economy Model of Congressional Careers. *American Economic Review* **95** 347–373.

Erdem, Tülin, Susumu Imai, Michael P. Keane. 2003. Brand and Quality Choice Dynamics under Price Uncertainty. *Quantitative Marketing and Economics* **1**(1) 5–64.

Erdem, Tülin, Michael P. Keane. 1996. Decision Making under Uncertainty: Capturing Dynamic Brand Choice Processes in Turbulent Consumer Goods Markets. *Marketing Science* **15**(1) 1–20.

Geweke, John F., Michael P. Keane. 2000. Bayesian Inference for Dynamic Discrete Choice Models Without the Need for Dynamic Programming. Mariano, Schuermann, Weeks, eds., *Simulation Based Inference and Econometrics: Methods and Applications*. Cambridge University Press, Cambridge, UK.

Gönül, Füsun, Kannan Srinivasan. 1996. Estimating the Impact of Consumer Expectations of Coupons on Purchase Behavior: A Dynamic Structural Model. *Marketing Science* **15**(3) 262–279.

Hartmann, Wesley R. 2006. Intertemporal Effects of Consumption and their implications for Demand Elasticity Estimates. *Quantitative Marketing and Economics* **4**(4) 325–349.

Hartmann, Wesley R., V. Brian Viard. 2008. Do Frequency Reward Programs Create Switching Costs? A Dynamic Structural Analysis of Demand in a Reward Program. *Quantitative Marketing and Economics* **6**(2) 109–137.

Hastings, W.K. 1970. Monte Carlo sampling methods using Markov Chains and their applications. *Biometrica* **57** 97–105.

Hendel, Igal, Aviv Nevo. 2006. Measuring the Implications of Sales and Consumer Inventory Behavior. *Econometrica* **74**(6) 1637–1673.

Hitsch, Günter. 2006. An Empirical Model of Optimal Dynamic Product Launch and Exit Under Demand Uncertainty. *Marketing Science* **25**(1) 25–50.

Hotz, Joseph V., Robert Miller. 1993. Conditional Choice Probabilities and the Estimation of Dynamic Models. *Review of Economic Studies* **60**(3) 497–529.

Imai, Susumu, Neelam Jain, Andrew Ching. 2009a. Bayesian Estimation of Dynamic Discrete Choice Models. *Econometrica* **77**(6) 1865–1899.

Imai, Susumu, Neelam Jain, Andrew Ching. 2009b. Supplement to 'Bayesian Estimation of Dynamic Discrete Choice Models'. *Econometrica Supplementary Material*, 77, http://www.econometricsociety.org/ecta/Supmat/5658_proofs.pdf.

Imai, Susumu, Kala Krishna. 2004. Employment, Deterrence and Crime in a Dynamic Model. *International Economic Review* **45**(3) 845–872.

Keane, Michael P., Kenneth I. Wolpin. 1994. The Solution and Estimation of Discrete Choice Dynamic Programming Models by Simulation and Interpolation: Monte Carlo Evidence. *Review of Economics and Statistics* **74**(4) 648–672.

Keane, Michael P., Kenneth I. Wolpin. 1997. The Career Decisions of Young Men. *Journal of Political Economy* **105** 473–521.

Kirkpatrick, S., C.D. Gelatt, M.P. Vecchi. 1983. Optimization by Simulated Annealing. *Science* **220** 671–680.

Kopalle, Praveen K., Scott A. Neslin, Baohong Sun, Yacheng Sun, Vanitha Swaminathan. 2007. A Dynamic Structural Model of the Impact of Loyalty Programs on Customer Behavior. Working paper, Tuck School of Business, Dartmouth College.

Lancaster, Tony. 1997. Exact Structural Inference in Optimal Job Search Models. *Journal of Business and Economic Statistics* **15**(2) 165–179.

Lewis, Michael. 2004. The Influence of Loyalty Programs and Short-Term Promotions on Customer Retension. *Journal of Marketing Research* **41**(3) 281–292.

Magnac, Thierry, David Thesmar. 2002. Identifying Dynamic Discrete Decision Processes. *Econometrica* **70**(2) 801–816.

McCulloch, Robert, Peter E. Rossi. 1994. An Exact Likelihood Analysis of the Multinomial Probit Model. *Journal of Econometrics* **64** 207–240.

Norets, Andriy. 2009. Inference in Dynamic Discrete Choice Models with Serially Correlated Unobserved State Variables. *Econometrica* **77**(5) 1665–1682.

Osborne, Matthew. 2008. Consumer Learning, Switching Costs, and Heterogeneity: A Structural Examination. Working paper, U.S. Department of Justice.

Pantano, Juan. 2008. *Essays in Applied Microeconomics*. Ph.D. dissertation, UCLA.

Rossi, Peter E., Greg M. Allenby. 1999. Marketing Models of Consumer Heterogeneity. *Journal of Econometrics* **89** 57–78.

Rossi, Peter E., Greg M. Allenby, Robert McCulloch. 2005. *Bayesian Statistics and Marketing*. John Wiley and Sons Ltd, Chichester, UK.

Rossi, Peter E., Robert McCulloch, Greg M. Allenby. 1996. The Value of Purchase History Data in Target Marketing. *Marketing Science* **15** 321–340.

Rust, John. 1987. Optimal Replacement of GMC Bus Engines: An Empirical Model of Harold Zurcher. *Econometrica* **55**(5) 999–1033.

Rust, John. 1997. Using Randomization to Break the Curse of Dimensionality. *Econometrica* **65**(3) 487–516.

Silverman, Bernard W. 1986. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London, UK.

Song, Inseong, Pradeep K. Chintagunta. 2003. A Micromodel of New Product Adoption with Heterogeneous and Forward Looking Consumers: Application to the Digital Camera Category. *Quantitative Marketing and Economics* **1**(4) 371–407.

Sun, Baohong. 2005. Promotion Effect on Endogenous Consumption. *Marketing Science* **24**(3) 430–443.

Train, Kenneth E. 2003. *Discrete Choice Methods with Simulation*. Cambridge University Press, Cambridge, UK. Available at http://elsa.berkeley.edu/books/choice2.html.

Walsh, Bruce. 2004. Markov Chain Monte Carlo and Gibbs Sampling. Lecture Notes for EEB 581, University of Arizona. http://nitro.biosci.arizona.edu/courses/EEB581-2004/handouts/Gibbs.pdf.

Yang, Botao, Andrew Ching. 2010. Dynamics of Consumer Adoption of Financial Innovation: The Case of ATM Cards. Working paper, Rotman School of Management, University of Toronto. Available at SSRN: http://ssrn.com/abstract=1434722.

Table 1: List of notations for homogeneous model in Section 3.3.1

Notation	Description
θ^{*r}	Candidate parameter vector in iteration r
θ^{r}	Accepted parameter vector in iteration r
$\tilde{p}^{r}$	$=(\tilde{p}_1^r, \tilde{p}_2^r)$; a draw of price vector in iteration r such that $\tilde{p}_j^r \sim N(\bar{p}, \sigma_p^2)$
$\tilde{V}_j^r(s, \tilde{p}^r; \theta^{*r})$	Pseudo alternative-specific value function for alternative j in iteration r conditional on $(s, \tilde{p}^r; \theta^{*r})$
$\tilde{V}^r(s, \tilde{p}^r; \theta^{*r})$	$=E_\epsilon \max_j\{\tilde{V}_j^r(s, \tilde{p}^r; \theta^{*r}) + \epsilon_{ij}\}$; pseudo-value function in iteration r conditional on $(s, \tilde{p}^r; \theta^{*r})$
H^r	$=\{\theta^{*l}, \tilde{V}^l(., \tilde{p}^l; \theta^{*l})\}_{l=r-N}^{r-1}$; set of past pseudo-value functions used for approximating the expected future value in iteration r
$\hat{E}_{p'}^r V(s, p'; \theta^{*r})$	$= \sum_{l=r-N}^{r-1} \tilde{V}^l(s, \tilde{p}^l; \theta^{*l}) \frac{K_h(\theta^{*r}-\theta^{*l})}{\sum_{k=r-N}^{r-1} K_h(\theta^{*r}-\theta^{*k})}$; pseudo expected future value in iteration r conditional on $(s; \theta^{*r})$
$\mathbf{b^d}$	$=\{b_{ijt}^d \ \forall i, j, t\}$; a vector of observed buying decisions
$\rho^r(\mathbf{b^d} \| \theta^{*r})$	Pseudo-likelihood conditional on H^r and θ^{*r}

Table 2: List of notations for heterogeneous model in Section 3.3.2

Notation	Description
G_j^r	Draw of G_j (population mean) in iteration r
$\sigma_{G_j}^r$	Draw of σ_{G_j} (population standard deviation) in iteration r
G_i^{*r}	$= (G_{i1}^{*r}, G_{i2}^{*r})$; Candidate parameter value specific to consumer i in iteration r
G_i^r	$= (G_{i1}^r, G_{i2}^r)$; Accepted parameter value specific to consumer i in iteration r
θ_3^{*r}	Candidate parameter vector common across consumers in iteration r
θ_3^r	Accepted parameter vector common across consumers in iteration r
$\tilde{p}^r$	$=(\tilde{p}_1^r, \tilde{p}_2^r)$; a draw of price vector in iteration r such that $\tilde{p}_j^r \sim N(\bar{p}, \sigma_p^2)$
$\tilde{V}_j^r(s, \tilde{p}^r; G_i^{*r}, \theta_3^{*r})$	Consumer i's pseudo alternative-specific value function for alternative j
	in iteration r conditional on $(s, \tilde{p}^r; G_i^{*r}, \theta_3^{*r})$
$\tilde{V}^r(s, \tilde{p}^r; G_i^{*r}, \theta_3^{*r})$	$=E_\epsilon \max_j\{\tilde{V}_j^r(s, \tilde{p}^r; G_i^{*r}, \theta_3^{*r}) + \epsilon_{ij}\}$; consumer i's pseudo-value function
	in iteration r conditional on $(s, \tilde{p}^r; G_i^{*r}, \theta_3^{*r})$
H^r	$=\{\theta^{*l}, \{G_i^{*r}, \tilde{V}^l(., \tilde{p}^l; G_i^{*r}, \theta_3^{*l})\}_{i=1}^I\}_{l=r-N}^{r-1}$; set of past pseudo-value functions
	used for approximating the expected future value in iteration r
$\hat{E}_{p'}^r V(s, p'; G_i^{*r}, \theta_3^{*r})$	$= \sum_{l=r-N}^{r-1} \tilde{V}^l(s, \tilde{p}^l; G_i^{*r}, \theta_3^{*l}) \frac{K_h(\theta_3^{*r}-\theta_3^{*l})K_h(G_i^{*r}-G_i^{*l})}{\sum_{k=r-N}^{r-1} K_h(\theta_3^{*r}-\theta_3^{*k})K_h(G_i^{*r}-G_i^{*k})}$; pseudo-expected
	future value for consumer i in iteration r conditional on $(s; G_i^{*r}, \theta_3^{*r})$
$\mathbf{b_i^d}$	$=\{b_{ijt}^d \ \forall j, t\}$; a vector of observed buying decisions for consumer i
$\rho_i^r(\mathbf{b_i^d} \mid G_i^{*r}, \theta_3^{*r})$	Pseudo-likelihood for consumer i conditional on H^r and $(G_i^{*r}, \theta_3^{*r})$
$\mathbf{b^d}$	$=\{b_{ijt}^d \ \forall i, j, t\}$; a vector of observed buying decisions
$\rho^r(\mathbf{b^d} \mid \{G_i^{*r}\}_{i=1}^I, \theta_3^{*r})$	Joint pseudo-likelihood conditional on H^r and $(\{G_i^{*r}\}_{i=1}^I, \theta_3^{*r})$

Table 3: Estimation Results: Homogeneous Model

		β = 0.6		β = 0.8	
parameter	TRUE	mean	sd	mean	sd
α_1 (intercept for store 1)	0.0	-0.001	0.019	-0.030	0.022
α_2 (intercept for store 2)	0.0	-0.002	0.019	-0.018	0.028
G_1 (reward for store 1)	1.0	0.998	0.017	1.052	0.021
G_2 (reward for store 2)	5.0	5.032	0.048	5.088	0.085
γ (price coefficient)	-1.0	-0.999	0.016	-0.996	0.019
β (discount factor)	0.6/0.8	0.601	0.008	0.800	0.010

Notes
Sample size: 1,000 consumers for 100 periods.
Fixed parameters: $\bar{S}_1 = 2$, $\bar{S}_2 = 4$, $\bar{p} = 1.0$, $\sigma_p = 0.3$, $\sigma_{G_j} = 0$ for $j = 1, 2$.
Tuning parameters: $N = 1,000$ (number of past pseudo-value functions used for expected future value approximations), $h = 0.01$ (bandwidth).

Table 4: Estimation Results: Heterogeneous Model

		β = 0.6		β = 0.8	
parameter	TRUE	mean	sd	mean	sd
α_1 (intercept for store 1)	0.0	-0.005	0.019	-0.022	0.022
α_2 (intercept for store 2)	0.0	0.010	0.021	0.005	0.037
G_1 (reward for store 1)	1.0	1.017	0.017	1.010	0.019
G_2 (reward for store 2)	5.0	5.066	0.065	4.945	0.130
σ_{G2} (sd of G_2)	1.0	1.034	0.046	1.029	0.040
γ (price coefficient)	-1.0	-1.004	0.016	-0.985	0.019
β (discount factor)	0.6/0.8	0.595	0.005	0.798	0.006

Notes
Sample size: 1,000 consumers for 100 periods.
Fixed parameters: $\bar{S}_1 = 2$, $\bar{S}_2 = 4$, $\bar{p} = 1.0$, $\sigma_p = 0.3$, $\sigma_{G_1} = 0$.
Tuning parameters: $N = 1,000$ (number of past pseudo-value functions used for expected future value approximations), $h = 0.01$ (bandwidth).

Table 5: Computation Time Per MCMC Iteration (in seconds)

	Homogeneous Model			Heterogeneous Model		
algorithm	$\beta = 0.6$	$\beta = 0.8$	$\beta = 0.98$	$\beta = 0.6$	$\beta = 0.8$	$\beta = 0.98$
Full solution based Bayesian	0.782	0.807	1.410	31.526	65.380	613.26
IJC with N=1000	1.071	1.049	1.006	19.300	19.599	18.387

Notes
Sample size: 1,000 consumers for 100 periods.
Number of state points: 8 ($\bar{S}_1 = 2$, $\bar{S}_2 = 4$).
Parameters:

- Homogeneous model:$(\alpha_1, \alpha_2, G_1, G_2, \gamma, \beta)$. We drew each parameter separately using the Metropolis-Hastings within Gibbs.
- Heterogeneous model: $(\alpha_1, \alpha_2, G_1, G_2, \sigma_{G_2}, \gamma, \beta)$. We drew each parameter except for G_2 and σ_{G_2} separately using the Metropolis-Hastings within Gibbs.

Table 6: The Impact of N

		N=100		N=1000	
parameter	TRUE	mean	sd	mean	sd
α_1 (intercept for store 1)	0.0	-0.049	0.020	-0.061	0.020
α_2 (intercept for store 2)	0.0	0.032	0.019	0.022	0.019
G_1 (reward for store 1)	1.0	1.234	0.034	1.246	0.021
G_2 (reward for store 2)	10.0	9.740	0.063	9.751	0.028
γ (price coefficient)	-1.0	-1.000	0.018	-0.991	0.018

Notes
Sample size: 1,000 consumers for 100 periods.
Fixed parameters: $\bar{S}_1 = 5$, $\bar{S}_2 = 10$, $\bar{p} = 1.0$, $\sigma_p = 0.3$, $\sigma_{G_j} = 0$ for $j = 1, 2$, $\beta = 0.98$.
Tuning parameters: $h = 0.01$ (bandwidth).

Figure 1: Choice probabilities across states for different discount factors

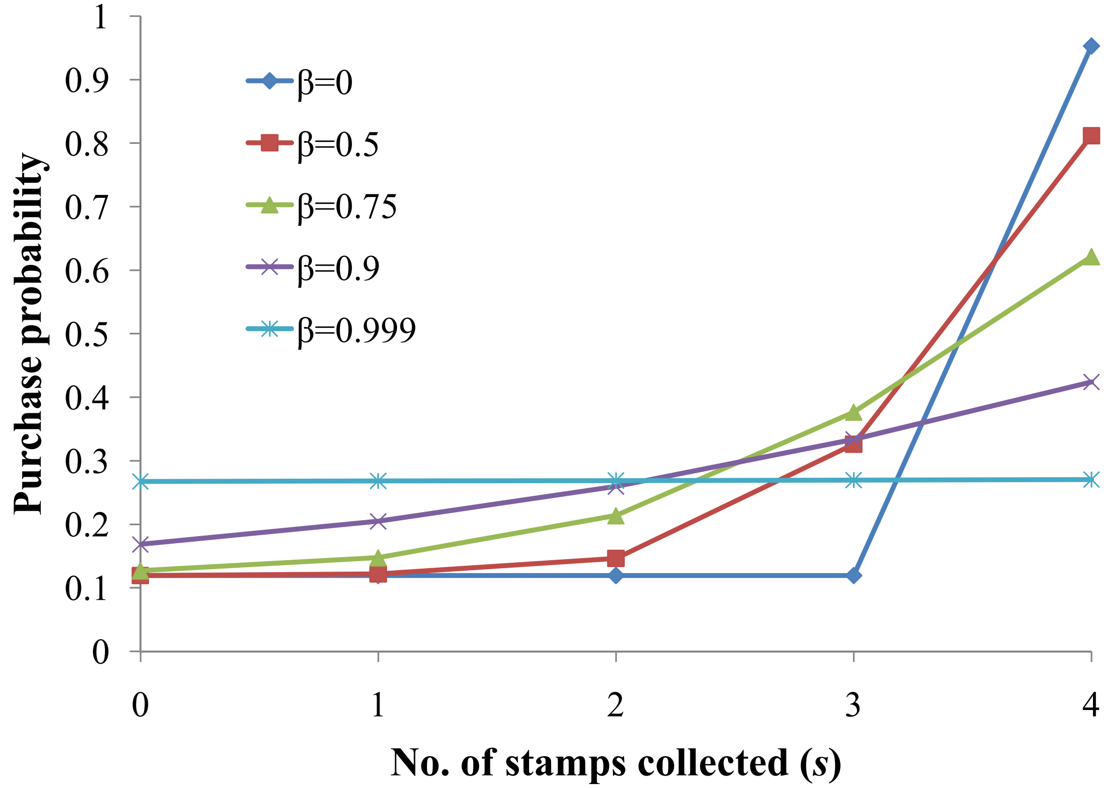


Figure 2: Choice probabilities for different discount factors across states

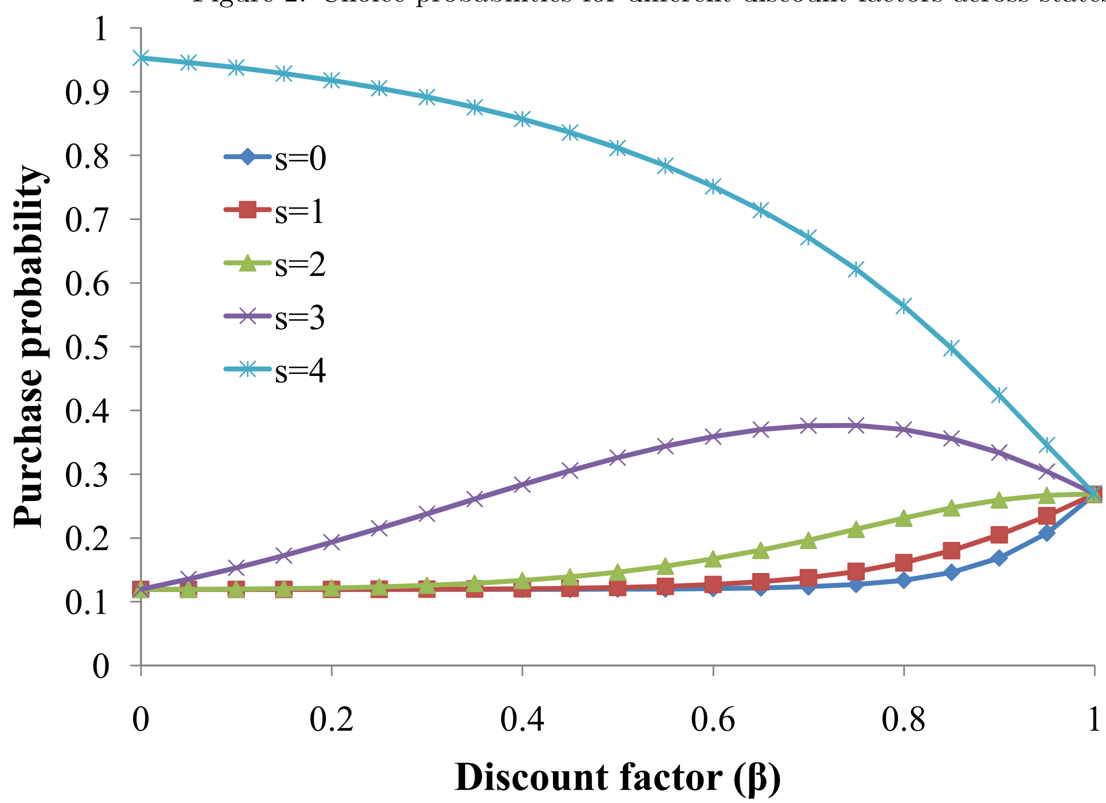


Figure 3: Flowchart for the conventional Bayesian approach (homogeneous model)

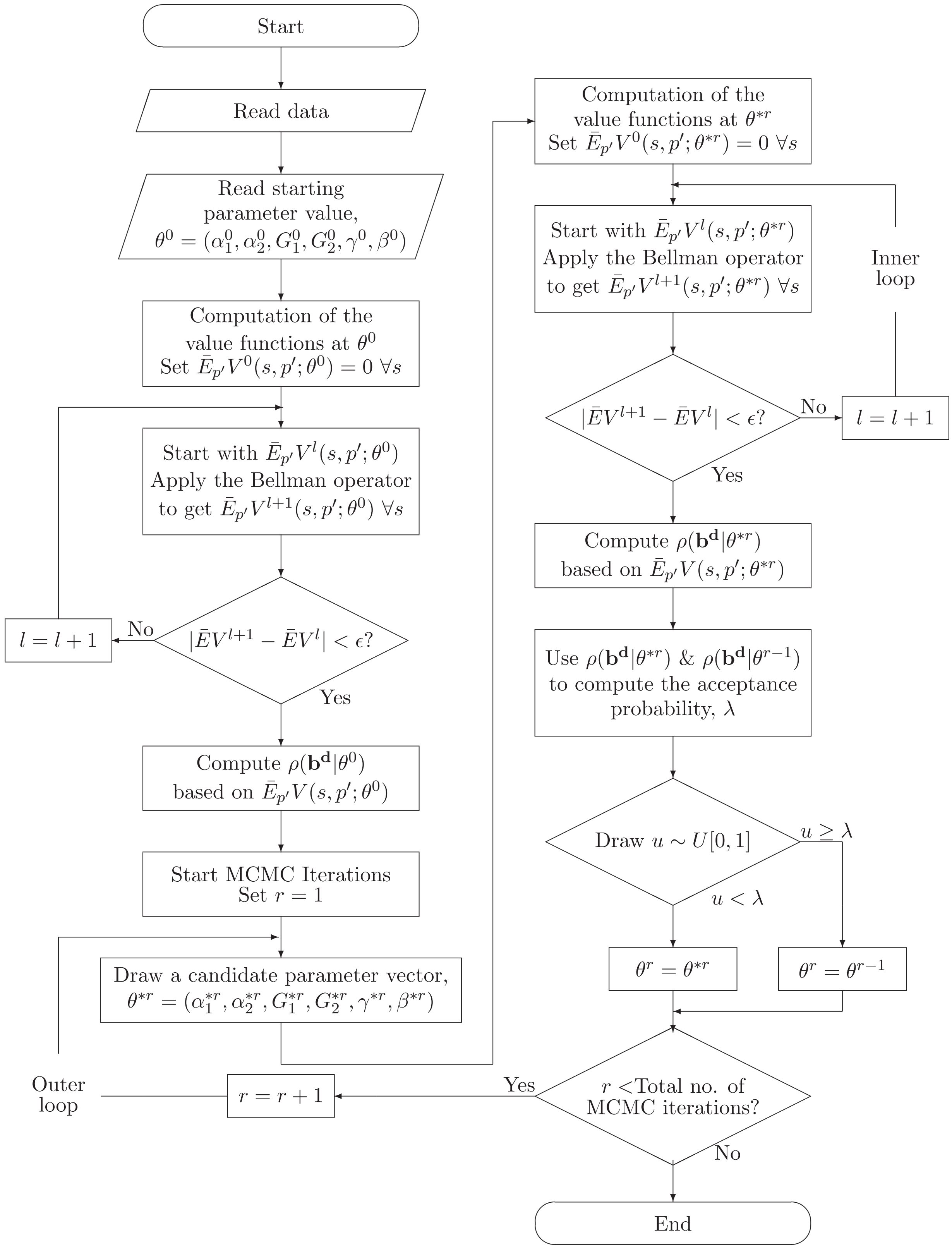


Figure 4: Flowchart for the IJC algorithm (homogeneous model)

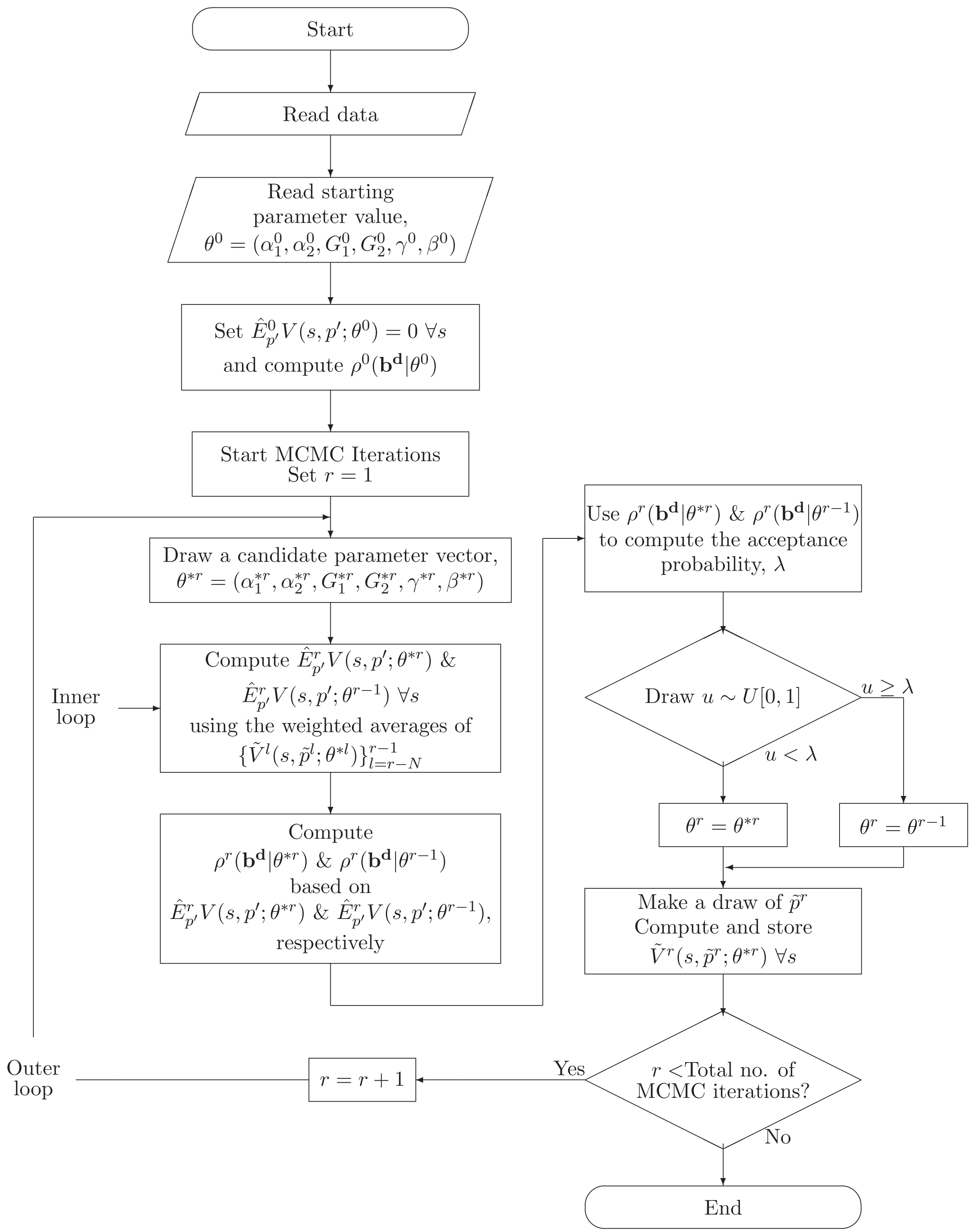


Figure 5: Flowchart for the IJC algorithm (heterogeneous model)

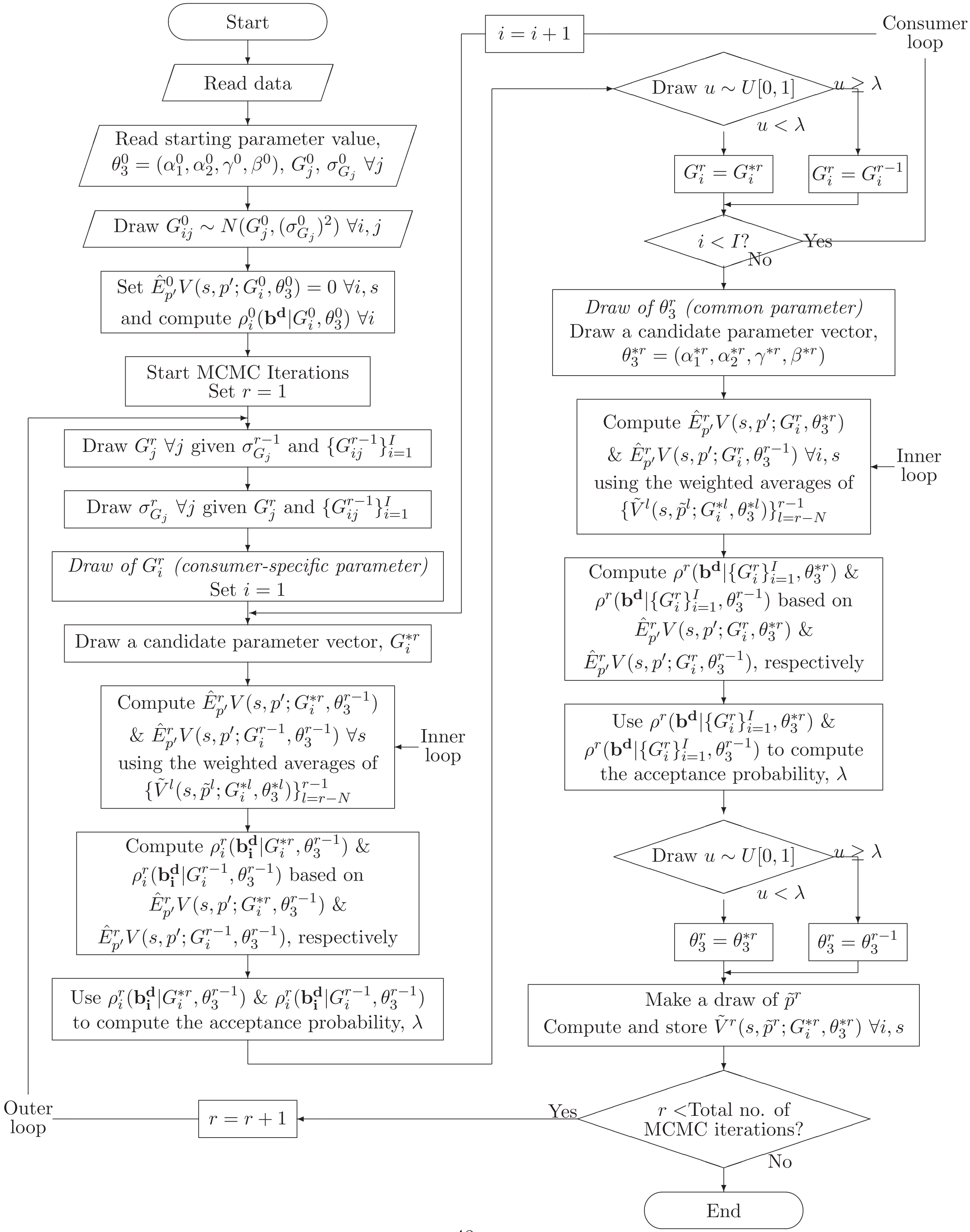
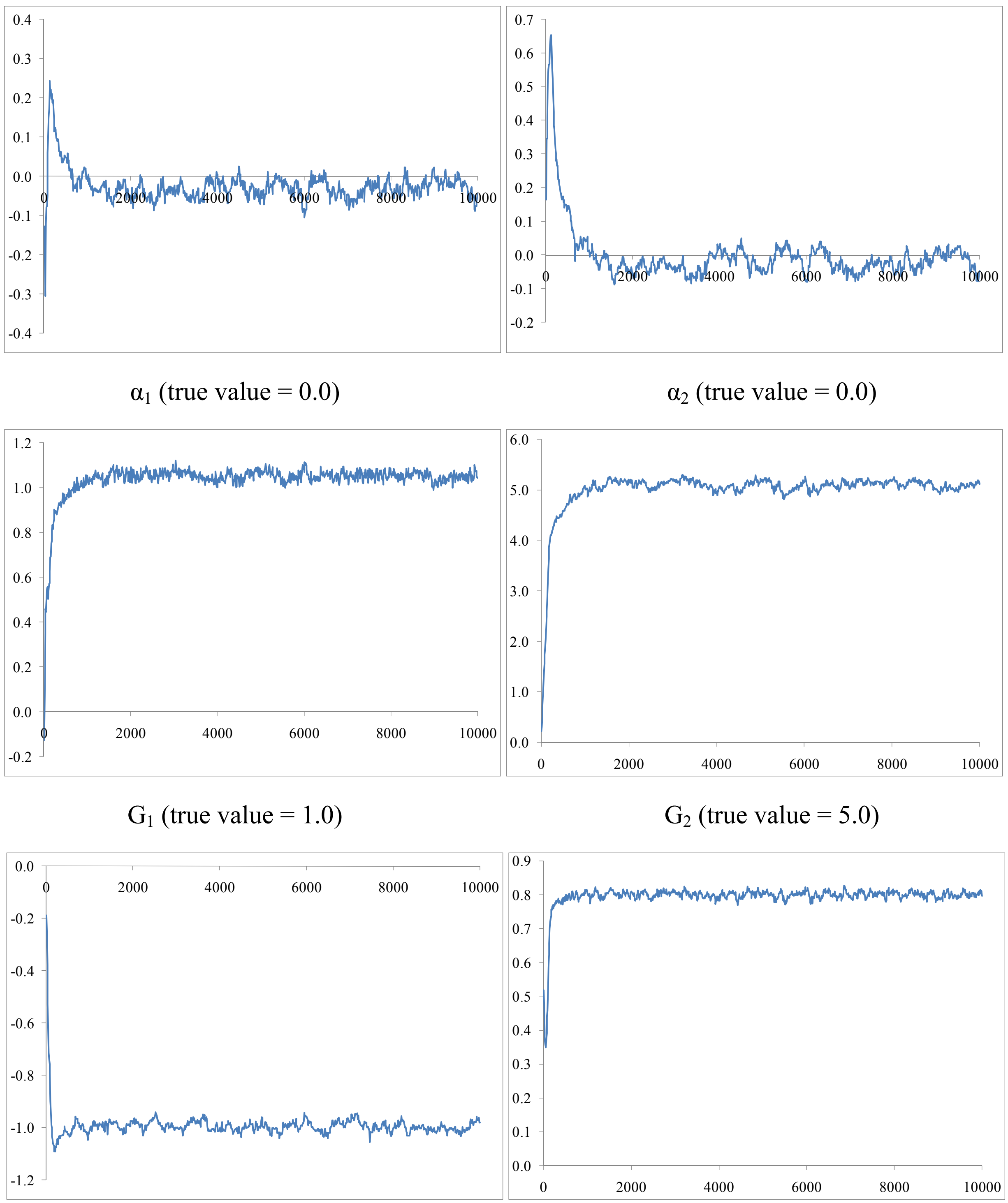


Figure 6: MCMC plots: Homogeneous Model with $\beta = 0.8$



α_1 (true value = 0.0)

α_2 (true value = 0.0)

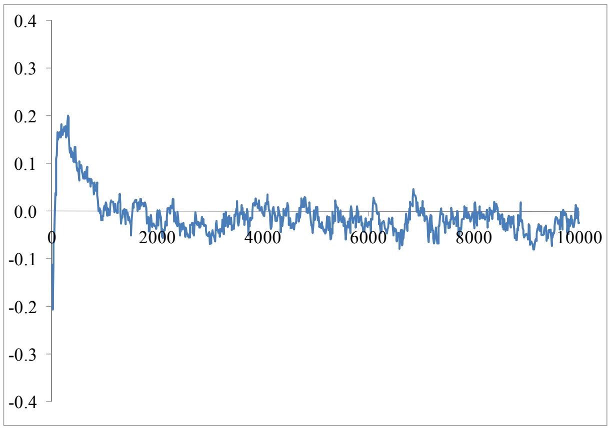
G_1 (true value = 1.0)

G_2 (true value = 5.0)

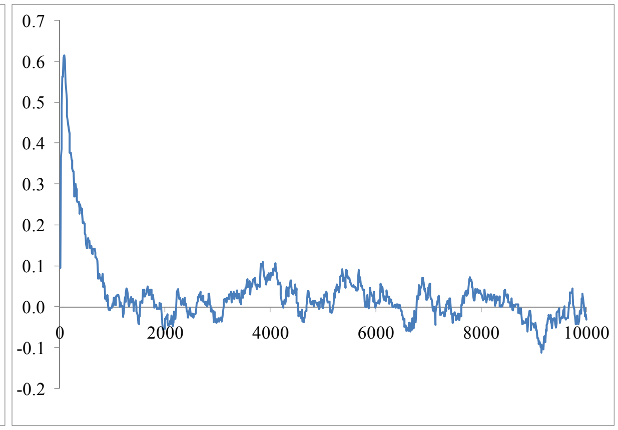
γ (true value = -1.0)

β (true value = 0.8)

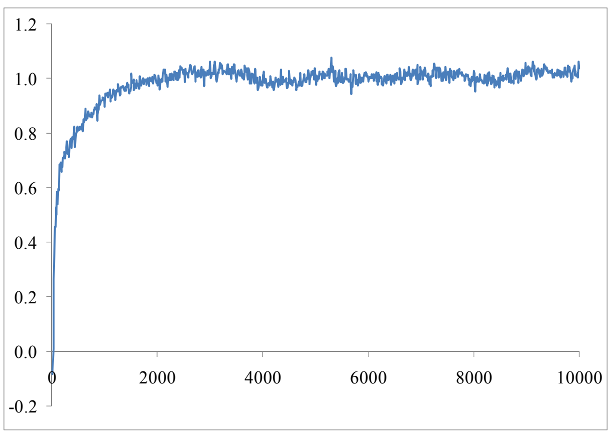
Figure 7: MCMC plots: Heterogeneous Model with $\beta = 0.8$



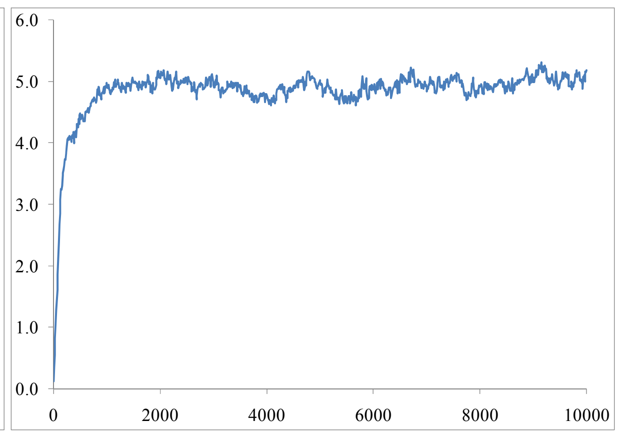
α_1 (true value = 0.0)



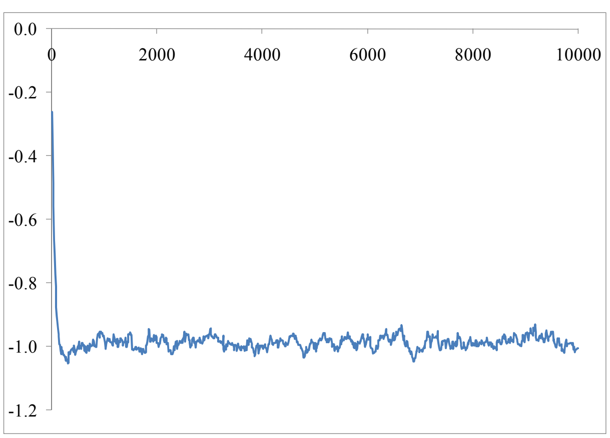
α_2 (true value = 0.0)



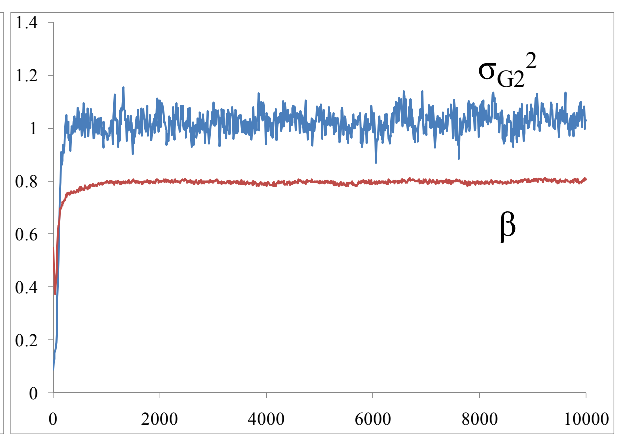
G_1 (true value = 1.0)



G_2 (true value = 5.0)



γ (true value = -1.0)



β (true value = 0.8)

σ_{G2}^2 (true value = 1.0)

Figure 8: MCMC plots: Impact of N on α_1 and α_2 when $\beta = 0.98$

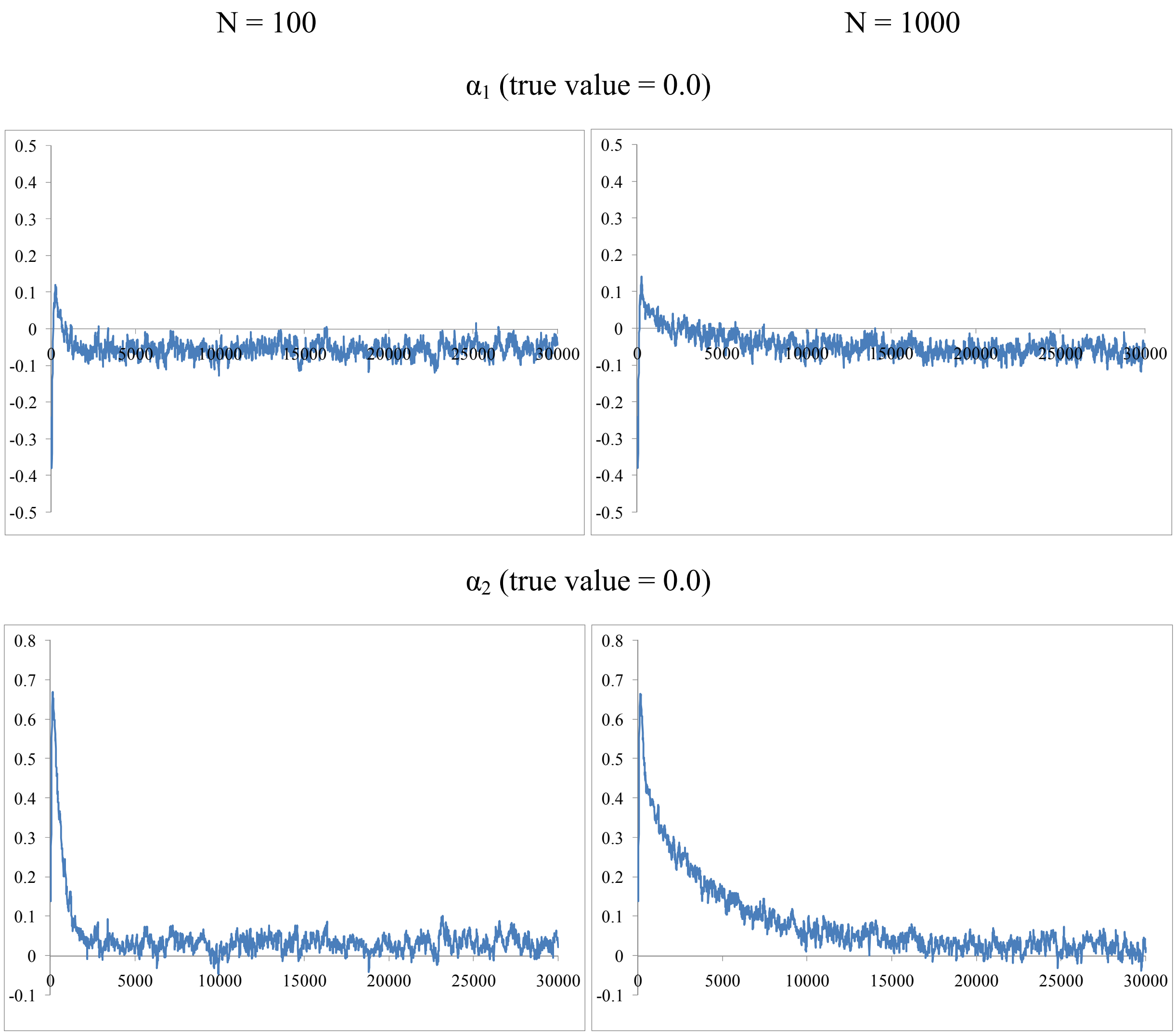
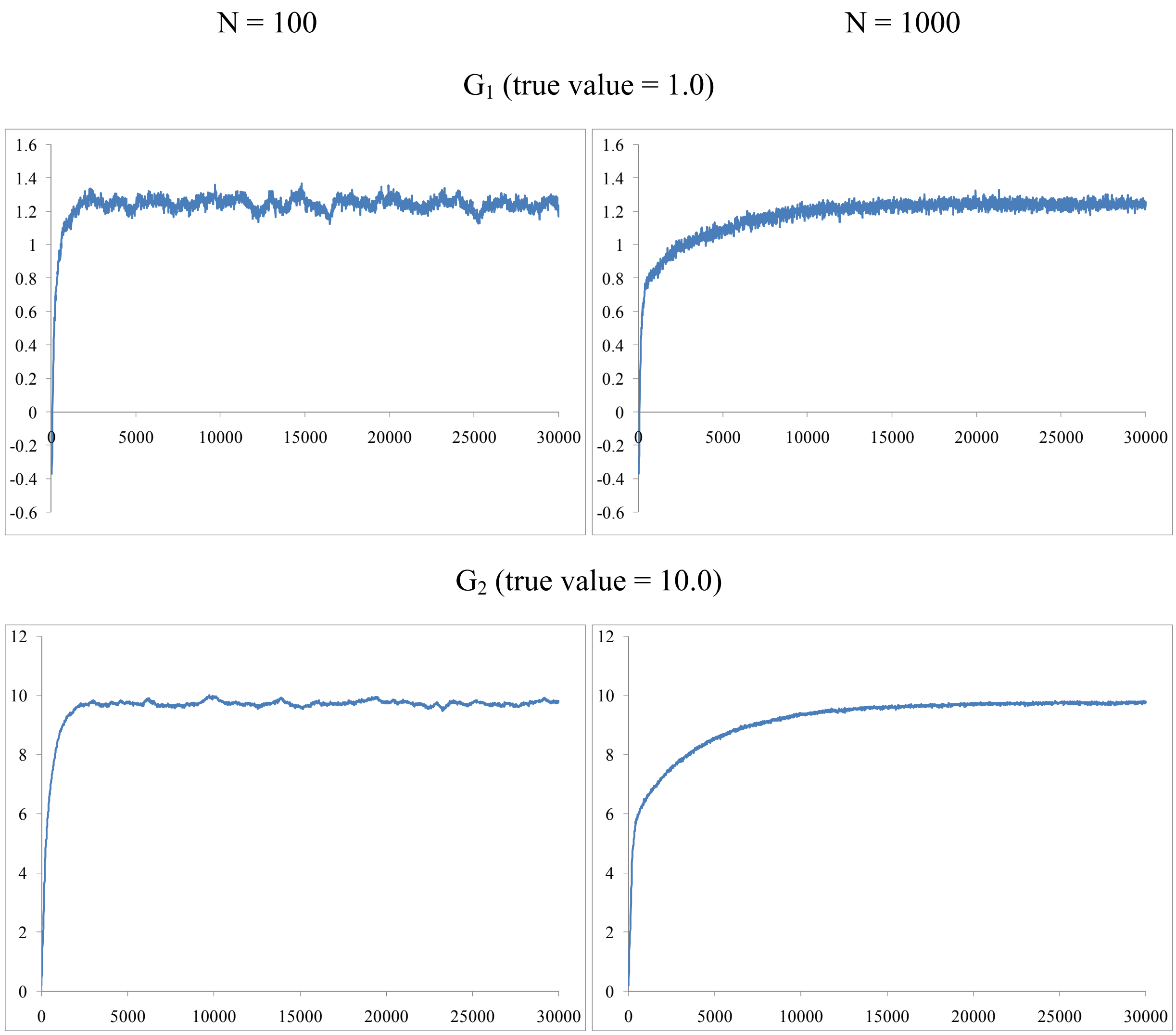


Figure 9: MCMC plots: Impact of N on G_1 and G_2 when $\beta = 0.98$



Appendix A

In this appendix, we provide some theoretical results about the behavior of $R(s,\beta)-W(s,\beta)$ using the simple dynamic store choice model with only one store presented in section 2.2.

Result 1

To illustrate how $R(s,\beta)-W(s,\beta)$ changes across s, we first consider consumers' purchase decision today under the assumption that they will visit the store in every period from tomorrow on. Under this assumption, the incentive to earn an extra stamp today, $R(s,\beta)-W(s,\beta)$, will have a simple and intuitive expression.

We consider the present discounted value of future utilities that a consumer in state s will obtain *if he/she visits the store in every period from the next period on.* Suppose that $s=0$. If he/she visits the store today, then the present discounted value, $R(s=0,\beta)$, under the assumption will be

$$R(s=0,\beta) = \alpha+\beta\alpha+\beta^2\alpha+\beta^3\alpha+\beta^4(\alpha+G)+\beta^5\alpha+\beta^6\alpha+\beta^7\alpha+\beta^8\alpha+\beta^9(\alpha+G)+\cdots.$$

On the other hand, if he/she does not visit the store today, then the present discounted value, $W(s=0,\beta)$, will be

$$W(s=0,\beta)=0+\beta\alpha+\beta^2\alpha+\beta^3\alpha+\beta^4\alpha+\beta^5(\alpha+G)+\beta^6\alpha+\beta^7\alpha+\beta^8\alpha+\beta^9\alpha+\beta^{10}(\alpha+G)+\cdots.$$

Thus

$$\begin{aligned}
R(s=0,\beta)-W(s=0,\beta) &= \alpha+\beta^4(1-\beta)G+\beta^9(1-\beta)G+\cdots \\
&= \alpha+\beta^4(1-\beta)(1+\beta^5+\beta^{10}+\cdots)G \\
&= \alpha+\frac{\beta^4(1-\beta)}{(1-\beta)(1+\beta+\beta^2+\beta^3+\beta^4)}G \\
&= \alpha+\frac{\beta^4}{1+\beta+\beta^2+\beta^3+\beta^4}G
\end{aligned}$$

In general, it is easy to verify that for any s,

$$R(s,\beta)-W(s,\beta)=\alpha+\frac{\beta^{\bar{S}-1-s}}{\sum_{k=0}^{\bar{S}-1}\beta^k}G.$$

This equation implies that when $\beta < 1$, (i) $R(s,\beta) - W(s,\beta)$ increases with s; (ii) $\frac{\partial(R(s,\beta)-W(s,\beta))}{\partial s}$ increases with s. It follows from (i) that the choice probability increases with s. It follows from (ii) that the increase in choice probability is relatively flat when s is small, but increases sharply when s approaches $\bar{S}$. The expression above also demonstrates why the shape of the choice probabilities across s would identify β and G. As long as $G > 0$ and $\beta > 0$, it is not possible to find two different combinations of (β, G) that gives the same value of $R(s,\beta) - W(s,\beta)$, $\forall s$. Finally, the expression above implies that as $\beta \to 1$, we have

$$R(s,\beta) - W(s,\beta) \to \alpha + \frac{G}{\bar{S}},\ \forall s.$$

Thus, when β approaches one, we observe a flat choice probability across s. Also, it is clear that it becomes hard to separately identify α and G as β approaches one.

Result 2

Now we provide a formal proof for the convergence of $R(s,\beta) - W(s,\beta)$ under the extreme value distribution assumption on the unobserved state variable.

Proposition 1 *Assume that the unobserved state variable, ϵ, follows the extreme value distribution, and $\bar{S} = 2$. Then, as $\beta \to 1$, $(R(s,\beta) - W(s,\beta))$ converges to $\alpha + \frac{G}{2}$, $\forall s$.*

Proof. For $s = 0$, the Bellman equation is given by

$$EV(s=0) = E\max\{V_0(s=0) + \epsilon_0, V_1(s=0) + \epsilon_1\}$$

where

$$\begin{aligned} V_1(s=0) &= \alpha + \beta EV(s=1) \\ V_0(s=0) &= \beta EV(s=0). \end{aligned}$$

For $s = 1$, the Bellman equation is given by

$$EV(s=1) = E\max\{V_0(s=1) + \epsilon_0, V_1(s=1) + \epsilon_1\}$$

where

$$\begin{aligned} V_1(s=1) &= \alpha + G + \beta EV(s=0) \\ V_0(s=1) &= \beta EV(s=1). \end{aligned}$$

Note first that

$$\begin{aligned} R(s=0,\beta) - W(s=0,\beta) &= V_1(s=0) - V_0(s=0) \\ &= \alpha + \beta(EV(s=1) - EV(s=0)). \\ R(s=1,\beta) - W(s=1,\beta) &= V_1(s=1) - V_0(s=1) \\ &= \alpha + G - \beta(EV(s=1) - EV(s=0)). \end{aligned}$$

Define $\Delta \equiv EV(s=1) - EV(s=0)$. If we assume that ϵ follows the extreme value distribution, then we have

$$\begin{aligned} & \Delta = \ln(\exp(V_0(s=1)) + \exp(V_1(s=1))) - \ln(\exp(V_0(s=0)) + \exp(V_1(s=0))) \\ \Leftrightarrow \quad & \Delta = \ln(1 + \exp(V_1(s=1) - V_0(s=1))) + V_0(s=1) \\ & \qquad - \ln(1 + \exp(V_1(s=0) - V_0(s=0))) - V_0(s=0) \\ \Leftrightarrow \quad & \Delta = \ln(1 + \exp(\alpha + G - \beta\Delta)) + \ln(1 + \exp(\alpha + \beta\Delta)) + \beta\Delta \\ \Leftrightarrow \quad & \exp((1-\beta)\Delta) = \frac{1 + \exp(\alpha + G - \beta\Delta)}{1 + \exp(\alpha + \beta\Delta)} \end{aligned}$$

Now note that when $\beta \to 1$, the LHS will approach one. Thus, we have $G - 2\beta\Delta \to 0$, or $\Delta \to \frac{G}{2}$. Therefore, as $\beta \to 1$, $R(s,\beta) - W(s,\beta)$ converges to $\alpha + \frac{G}{2}$ for all s. □

Appendix B

In this appendix, we discuss some techniques that one can use in practice to reduce the computational burden further. While we will use the model without unobserved heterogeneity for illustration purpose, the same ideas apply to the model with unobserved heterogeneity.

Integration of iid price shocks

In the base model specification of the store choice model with reward programs, we assume that prices are iid normal random variable. When implementing the IJC algorithm, we propose to make one draw of price vector, $\tilde{p}^r$, and store $\tilde{V}^r(s, \tilde{p}^r; \theta^{*r})$ in each iteration. Alternatively, we may draw a number of price vector in each iteration, $\{\tilde{p}^m\}_{m=1}^M$, evaluate $\bar{E}_{p'}\tilde{V}^r(s, p'; \theta^r)$ using

$$\bar{E}_{p'}\tilde{V}^r(s, p'; \theta^{*r}) = \frac{1}{M}\sum_{m=1}^{M}\tilde{V}^r(s, \tilde{p}^m; \theta^{*r}), \tag{14}$$

and store $\bar{E}_{p'}\tilde{V}^r(s, p'; \theta^{*r})$ instead of $\tilde{V}^r(s, \tilde{p}^r; \theta^{*r})$. The expected future value can then be approximated as follows (correspond to step 3 in Section 3.3.1).

$$\hat{E}^r_{p'}V(s, p'; \theta^{*r}) = \sum_{l=r-N}^{r-1}\bar{E}_{p'}\tilde{V}^l(s, p'; \theta^{*l})\frac{K_h(\theta^{*r} - \theta^{*l})}{\sum_{k=r-N}^{r-1}K_h(\theta^{*r} - \theta^{*k})}.$$

In this alternative approach, we integrate out price first, before using the kernel regression to obtain the pseudo expected future value function $\hat{E}^r_{p'}V(s, p'; \theta^{*r})$. So this should allow us to achieve the same level of precision by using a smaller N. One potential advantage is that it saves us some memory when computing the weighted average. The additional cost is that we need to compute $\bar{E}_{p'}\tilde{V}^r$ in each MCMC iteration. In terms of computational time, we find that these two approaches are roughly the same in our example.

We should also note that in the present example where we assume prices are observed, one can use the observed prices as random realizations in computing $\bar{E}_{p'}\tilde{V}^r(s, p'; \theta^{*r})$, provided that there are a sufficient number of observations for each s. The advantage of using this approach is that the pseudo-value functions of the observed prices, $\tilde{V}_j^r(s, p^d; \theta^{*r})$, are by-products of the likelihood function computation. So we can skip step 4(a) and (b) in section 3.3.1.

Computation of $\rho^r(\theta^{r-1})$

In Section 3.3.1, we propose to compute the pseudo-likelihood at previously accepted parameter vector, $\rho^r(\theta^{r-1})$, in each iteration. This is mainly because in IJC, the set of past pseudo-value functions is updated in each iteration, and thus the pseudo-likelihood computed in the previous iteration, $\rho^{r-1}(\theta^{r-1})$, is different from $\rho^r(\theta^{r-1})$. However, in practice, the computation of pseudo-likelihood is the most time-consuming part in the algorithm. Moreover, the set of past pseudo-value functions is updated only by one element in each iteration. Thus, we propose the following procedure, which avoids computing $\rho^r(\theta^{r-1})$ in every iteration.

Suppose that we are in Step 3 of iteration r (Section 3.3.1). If we have accepted the candidate parameter value in iteration $r-1$ (i.e., $\theta^{r-1} = \theta^{*(r-1)}$), then use $\rho^{r-1}(\theta^{*(r-1)})$ as a proxy for $\rho^r(\theta^{r-1})$. Note that the calculations of $\rho^r(\theta^{r-1})$ and $\rho^{r-1}(\theta^{*(r-1)})$ only differ in one past pseudo-value function, and $\rho^{r-1}(\theta^{*(r-1)})$ has already been computed in iteration $r-1$. If we have rejected the candidate parameter vector (i.e., $\theta^{r-1} = \theta^{r-2}$), then we could use $\rho^{r-1}(\theta^{r-2})$ as a proxy for $\rho^r(\theta^{r-1})$, and only compute $\rho^r(\theta^{r-1})$ once every several successive rejections. This avoids using the pseudo-likelihood that is based on an old set of past pseudo-value functions as a proxy for $\rho^r(\theta^{r-1})$. According to our experience, one can obtain a fairly decent reduction in computational time when using this approach.

Appendix C

In this appendix, we explain an alternative way to implement IJC when estimating the model with unobserved heterogeneity. The main goal of this alternative approach is to reduce the memory requirement and computational burden further. Instead of storing $\{\theta_3^{*l}, \{G_i^{*l}, \tilde{V}^l(., p^l; G_i^{*l}, \theta_3^{*l})\}_{i=1}^{I}\}_{l=r-N}^{r-1}$, one can store $\{\theta_3^{*l}, G_{i'}^{*l}, \tilde{V}^l(., p^l; G_{i'}^{*l}, \theta_3^{*l})\}_{l=r-N}^{r-1}$, where $i' = r - I * int(\frac{r-1}{I})$; $int(.)$ is an integer function that converts any real number to an integer by discarding its value after the decimal place. i' is simply one way to "randomly" select a consumer's pseduo-value function to be stored in each iteration. When approximating the expected future value in, say step 4(b) in section 3.3.2, we can then set

$$\hat{E}_{p'}^r V(s, p'; G_i^{*r}, \theta_3^{r-1}) = \sum_{l=r-N}^{r-1} \tilde{V}^l(s, p^l; G_{i'}^{*l}, \theta_3^{*l}) \frac{K_h(\theta_3^{r-1} - \theta_3^{*l}) K_h(G_i^{*r} - G_{i'}^{*l})}{\sum_{k=r-N}^{r-1} K_h(\theta_3^{r-1} - \theta_3^{*k}) K_h(G_i^{*r} - G_{i'}^{*k})}.$$

Note that we are using the same set of past psuedo-value functions for all consumers here. If there is a large number of consumers in the sample, this approach, which is also independently adopted by Osborne (2008), can dramatically reduce the memory requirement and computational burden for implementing IJC.

This approach works because $G_{i'}^{*l}$ is a random realization from a distribution that covers the support of the parameter space. This is one important requirement that ensures the pseudo-value functions converge to the true ones in the proof of IJC.