

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

Andrew 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
Northern Illinois University

May 1, 2009

*We thank Peter Rossi for encouraging us to develop this paper. We also thank the participants of 2009 UCLA Marketing Camp and 2009 SBIES conference for their helpful comments. All remaining errors are ours.

[†]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.

Abstract

This paper provides a step-by-step guide to estimating infinite horizon, discrete choice dynamic programming (DDP) models using the Bayesian Dynamic Programming algorithm developed in Imai, Jain and Ching (2009) (IJC). The IJC method combines the DDP solution algorithm with the Bayesian Markov Chain Monte Carlo algorithm into a single algorithm, which solves the DDP model and estimates its structural parameters simultaneously. The main computational advantage of this estimation algorithm is the efficient use of information obtained from the past iterations. 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 Bayesian Dynamic Programming algorithm extensively uses the computational results obtained from the past iterations to help solving the DDP model at the current iterated parameter values. Consequently, it significantly alleviates the computational burden of estimating DDP models. We carefully discuss how to implement the algorithm in practice, and use a simple dynamic store choice model to illustrate how to apply it to obtain parameter estimates. In addition, we propose a new modification of the IJC algorithm, which allows us to estimate a DDP model and conduct a policy experiment simultaneously.

Keywords: Bayesian Estimation, Dynamic Programming, Discrete Choice Models, Rewards 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) framework. This framework has been applied to study manager's decisions to replace old equipments (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., Akerberg 2003; Song and Chintagunta 2003; Yang and Ching 2009), new product introduction decisions (e.g., Hitsch 2006), 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, and reduce 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; Akerberg 2001).¹ Never-

¹Geweke and Keane (2002) 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

theless, little progress has been made in handling the complexity of the likelihood function from the DDP models. A typical approach is to use different initial values to re-estimate the model, and check which 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). However, 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 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 (2009) (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

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.

iterations remains unused in the current iteration. In contrast, the IJC algorithm extensively uses the computational results obtained from the past iterations to help solving the DDP model at the current iterated parameter values. This new method is potentially superior to prior methods because (1) it significantly reduces 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 estimating infinite horizon, discrete choice dynamic programming (DDP) models using the IJC method. We carefully discuss how to implement the algorithm in practice, and use a simple dynamic store choice model to illustrate how to apply this algorithm to obtain parameter estimates. Our goal is to reduce the costs of adopting this new method and expand the toolbox for researchers who are interested in estimating DDP models. The rest of the paper is organized as follows. In section 2, we present a dynamic store choice model, where each supermarket chain offers its own reward programs. In section 3, we present the IJC method and explain how to implement it to obtain parameter estimates of this model. We also discuss the practical aspects of using this new method. In section 4 we use the estimation results to demonstrate certain properties of the IJC algorithm. Section 5 discusses how to extend it to (i) conduct policy experiments, and (ii) allow for continuous state variables. In particular, the modification that we propose for conducting policy experiments is new – it allows us to estimate a DDP model and conduct a policy experiment simultaneously. We also comment on the choice of kernels. 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.). 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. Let $s_{ijt} \in \{0, 1, \dots, \bar{S}_j - 1\}$ 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.

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

given by

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

where α_j measures the consumer loyalty (or 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 idiosyncratic error term. We assume ϵ_{ijt} is 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.² In each period, consumers may choose not to go shopping. The single period mean utility of no shopping is normalized to zero, i.e., $U_{i0t} = \epsilon_{i0t}$.

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

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

where $d_{ijt} = 1$ if consumer i chooses chain j in period t and $d_{ijt} = 0$ 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:

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

Let θ be the vector of parameters, $p_i = (p_{i1}, p_{i2})$ and $G_i = (G_{i1}, G_{i2})$. In state s_i , the Bellman's equation for consumer i is given by

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

where the second equality follows from the extreme value assumption on ϵ . The alternative-

²For example, suppose that the gift is a vase, which some consumers value highly, while some other consumers who already have a few vases at home may not find it very useful to have an extra one.

specific value functions are written as: For $j = 1, 2$,

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

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

where the state transition from s_i to s' follows (1), and the expectation with respect to p' is defined as

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

The parameters of the model are α_j (consumer loyalty for supermarket chain j), G_j (consumers' mean value of the gift offered by chain j), σ_{G_j} (standard deviation of G_{ij}), γ (price sensitivity), β (discount factor), \bar{p} (mean price common across stores), and σ_p (standard deviation of prices).

To our knowledge, there are two papers that study reward programs in a dynamic setting: Lewis (2004), 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. The general dynamics of this model is also 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 (1) it is quite commonly used and under this distributional assumption, the choice probability and the Emax function have closed form

analytical expressions, (2) 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 simulate the choice probabilities across s for different discount factors by setting $\alpha = -2$, $\gamma = 0$, $G = 3$, 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 becomes much higher as s approaches $\bar{S} - 1$. In one extreme when $\beta = 0$, 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

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

$$W(s, \beta) \equiv \beta EV(s' = s).$$

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 as 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 $\beta \rightarrow 1$. 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 the appendix, 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 ; (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_j and G_j because α_j shifts the choice probabilities equally across s , while G_j shifts them almost equally across s .³

To illustrate the intuitions 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 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 $\beta \rightarrow 1$, $W(s = 3, \beta)$ catches up. This explains why the choice probability of visiting the chain first increases and then decreases. Finally, all the observations indicates that $R(s, \beta)$ is always higher than $W(s, \beta)$ as long as $\beta > 0$.

We now turn to discuss how to apply the IJC algorithm to estimate this model.

³Hartmann 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 frequent a customer visits a store (determined by α_j here). Lewis (2004) did not estimate the discount factor either.

3 Estimation Method

It is well-known that when using the Maximum Likelihood or Bayesian MCMC to estimate discrete choice dynamic programming models, the main computational burden is solving the value function at each set of trial parameter vector. Since both procedures, in particular Bayesian MCMC, require many iterations to achieve convergence, a typical nested fixed point algorithm will need to repeatedly apply the re-iteration procedure to solve for the value functions.⁴ As a result, the computational burden is so large that even a very simple discrete choice dynamic programming model cannot be estimated using standard Bayesian MCMC methods.

In this section, we first outline the steps required to solve the model numerically to highlight the computational burden of the nested fixed point algorithm. Then, we discuss the features of the IJC algorithm and how it reduces the computational burden in the context of the dynamic store choice model presented above.

3.1 Numerical Solution of the Model

Consider the model described in section 2. To solve the model numerically, we will carry out the following steps for each consumer i .

1. Make M draws of $\{p_j^m\}_{m=1}^M$ from the price distribution function, $N(\bar{p}, \sigma_p^2)$, for $j = 1, 2$. We

fix these draws below.

2. Start with an initial guess of the value functions, e.g., setting $V^0(s, p; G_i, \theta) = 0, \forall s, p$.

Suppose that we know V^l , where l is the number of past iterations. Step 3 & 4 discuss how to obtain V^{l+1} .

⁴It should be noted that Bayesian MCMC algorithm generally needs to be run 10,000 to 30,000 iterations to obtain enough draws for approximating the posterior distributions.

3. Substitute $\{p^m\}_{m=1}^M$ into $V^l(s, p; G_i, \theta)$, and then take the average across p^m 's to obtain a Monte Carlo approximation of $E_{p'}V^l(s', p'; G_i, \theta), \forall s'$.
4. Substitute these approximated expected future value functions into the Bellman operator and obtain $V^{l+1}(s, p; G_i, \theta), \forall s, p$.
5. Repeat step 3-4 until $E_p V^{l+1}(s, p; G_i, \theta)$ converges $\forall s$.

In general, the computational burden increases with the number of state points ($\bar{S}_1 * \bar{S}_2$) and the number of consumers (I). Also, for all s , the number of iterations required to achieve the convergence of $E_p V^{l+1}(s, p; G_i, \theta)$ increases as the discount factor β increases. The computational burden of obtaining the numerical solution of a DDP model is the main obstacle of using the nested fixed point algorithm proposed by Rust (1997). We will now turn to discuss the pseudo-MCMC algorithm proposed by IJC, and explain how it reduces the computational burden of estimating DDP models.

3.2 IJC algorithm

The IJC algorithm relies on two insights to reduce the computational burden of each MCMC iteration. 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). Second, the value functions evaluated at the past MCMC draws of parameters contain useful information about the value functions at the current draw of parameters, in particular, for those evaluated within a neighborhood of the current parameter values. However, the traditional nested fixed point algorithm hardly makes use of them. Based on these two insights, IJC propose to replace the contraction mapping

procedure of solving the value functions with a weighted average of the pseudo-value functions obtained as past outcomes of the estimation algorithm. In IJC, the weight depends on the distance between the past parameter vector draw and the current one – the shorter the distance, the higher the weight. The basic intuition is that the value function is continuous in the parameter space. Therefore, it is possible to use the past value functions to form a non-parametric estimate of the value function evaluated at the current draw of parameter vector. Such a non-parametric estimate is usually computationally much cheaper than the standard contraction mapping procedure, in particular for β close to 1. Consequently, IJC dramatically reduce the computational burden of each iteration in the Bayesian MCMC algorithm. This modified procedure differs from the standard nested fixed point algorithm in an important aspect: instead of solving the model and search for parameters alternately, it solves and estimates the model simultaneously.

In the context of the reward program example without consumer heterogeneity, the outputs of the algorithm in each iteration r include $\{\theta^r, E_p \tilde{V}^r(s, p; \theta^r)\}$, where \tilde{V}^r is the pseudo-value function. To obtain these outputs, IJC make use of the past outcomes of the estimation algorithm, $H^r = \{\theta^l, E_p \tilde{V}^l(s, p; \theta^l)\}_{l=r-N}^{r-1}$ where N is the number of past expected pseudo-value functions used for approximating the expected future value. The pseudo-value functions are defined as follows. To simplify notations, we drop the i subscript for s and p . For each s ,

$$\tilde{V}^r(s, p; \theta^r) = \log(\exp(\tilde{V}_0^r(s; \theta^r)) + \exp(\tilde{V}_1^r(s, p_1; \theta^r)) + \exp(\tilde{V}_2^r(s, p_2; \theta^r))), \quad (4)$$

where for $j = 1, 2$,

$$\begin{aligned} \tilde{V}_j^r(s, p_j; \theta^r) &= \begin{cases} \alpha_j - \gamma p_j + G_j + \beta \hat{E}_p^r V(s', p'; \theta^r) & \text{if } s_j = \bar{S}_j - 1, \\ \alpha_j - \gamma p_j + \beta \hat{E}_p^r V(s', p'; \theta^r) & \text{otherwise,} \end{cases} \\ \tilde{V}_0^r(s, p_j; \theta^r) &= \beta \hat{E}_p^r V(s', p'; \theta^r). \end{aligned} \quad (5)$$

Note that \tilde{V}_0^r does not depend on p_j , but it is more convenient for our presentation later to keep

p_j as one of its arguments.

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

$$\hat{E}_p^r V(s, p; \theta^r) = \sum_{l=r-N}^{r-1} E_p \tilde{V}^l(s, p; \theta^l) \frac{K_h(\theta^r - \theta^l)}{\sum_{k=r-N}^{r-1} K_h(\theta^r - \theta^k)}, \quad (6)$$

where $K_h(\cdot)$ is a Gaussian kernel with bandwidth h . To obtain $E_p \tilde{V}^r(s, p; \theta^r)$, we simulate a set of prices, $\{p^m\}_{m=1}^M$, from the price distribution, $N(\underline{p}, \sigma_p^2)$, and evaluate $\tilde{V}^r(s, p^m; \theta^r)$ using (4) and (5), i.e.,

$$E_p \tilde{V}^r(s, p; \theta^r) = \frac{1}{M} \sum_{m=1}^M \tilde{V}^r(s, p^m; \theta^r). \quad (7)$$

IJC propose to treat the pseudo-value function as the true value function in a MCMC algorithm. They prove that under some regularity conditions, the pseudo-value functions converge to the true ones in probability uniformly, and the sequence of parameter vector draws converges to the true posterior distribution in total variation norm.⁵

It should be highlighted that the approximated expected future value function defined in (6) is the key innovation of IJC (hereafter we call it pseudo expected future value). In principles, this step is also applicable in classical estimation methods such as GMM and Maximum Likelihood.⁶ However, there are at least two advantages of implementing IJC's pseudo-value function approach in Bayesian estimation. First, the non-parametric approximation in (6) would be more efficient if the past pseudo-value functions are evaluated at θ^l 's that are randomly distributed around θ^r . This can be naturally achieved by the Bayesian MCMC algorithm. On the contrary, classical

⁵Strictly speaking, parameter vector draws obtained from the IJC algorithm is not a markov chain because the pseudo expected future value depends on the past expected pseudo-value functions, which are evaluated at $\{\theta\}_{l=r-N}^{r-2}$ in addition to θ^{r-1} . As a result, the proof of convergence is non-standard.

⁶Brown 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.

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 parameter spaces along a particular path. Consequently, we believe that the approximation step proposed by IJC should perform better under the Bayesian MCMC approach.⁷ Second, in the presence of unobserved consumer heterogeneity, it is common that the likelihood function is multi-modal even for static choice problems. In this situation, Bayesian posterior means often turn out to be better estimators of true parameter values than classical point estimates. This is because in practice, accurately simulating a posterior is usually much easier than finding the global maximum/minimum of a complex likelihood/GMM objective function.

Now we turn to discuss how to implement the IJC method to estimate the dynamic store choice model present here. We consider two versions of the model: (i) without unobserved consumer heterogeneity, (ii) with unobserved consumer heterogeneity.

3.3 Implementation of the IJC algorithm

In this subsection, we discuss how to estimate the dynamic store choice model using the IJC algorithm. The steps are similar to the standard Metropolis-Hastings algorithm, except that we use (6) to calculate the expected future value. Let $I_{buy,ijt}$ be an indicator function for purchasing, and $p_{it} = (p_{i1t}, p_{i2t})$ be the price vector for consumer i at supermarket chain j at time t . We use $(I_{buy,ijt}^d, p_{ijt}^d), T_i$ and I to denote the observed data, total number of periods observed for each consumer i and total number of consumers, respectively.

Our focus is to provide a step-by-step implementation guide and discuss the practical aspects

⁷A stochastic optimization algorithm, simulated annealing, has recently gained some attention to handle complicated objective function. This algorithm is an adaptation of the Metropolis-Hastings algorithm (Kirkpatrick et al. 1983; Černý 1985). The IJC method should also be well-suited when incorporating with simulated annealing for the classical estimation approach. 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.

of the IJC algorithm. 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.

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 $j = 1, 2$). The vector of parameters to be estimated is $\theta = (\alpha_1, \alpha_2, G_1, G_2, \gamma, \beta)$.

The IJC algorithm modifies the Metropolis-Hastings algorithm, and involves drawing a candidate parameter θ^{*r} in each iteration.

1. Suppose that we are at iteration r . We start with a history of past outcomes from the algorithm,

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

where N is the number of past iterations used for expected future value approximation; $E_p \tilde{V}^l$ is the expected pseudo-value functions w.r.t. p ; $\rho^r(\theta^{r-1})$ is the pseudo-likelihood of the data conditional on $\hat{E}_p^r V(\cdot, p; \theta^{r-1})$.

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} , $\rho^r(\theta^{*r})$,

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

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

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

Substituting this into (5) gives us \tilde{V}_j^r .

Then we determine whether or not to accept θ^{*r} based on the acceptance probability,

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

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. Computation of $E_p \tilde{V}^r(s, p; \theta^{*r})$.

(a) Make M draws of prices, $\{p^m\}_{m=1}^M$, from the price distribution.

(b) Compute $\tilde{V}_0^r(s; \theta^{*r})$, $\tilde{V}_1^r(s, p_1^m; \theta^{*r})$ and $\tilde{V}_2^r(s, p_2^m; \theta^{*r})$, using $\hat{E}_p^r V$ computed in Step 3.

(c) Given $\tilde{V}_0^r(s; \theta^{*r})$, $\tilde{V}_1^r(s, p_1^m; \theta^{*r})$ and $\tilde{V}_2^r(s, p_2^m; \theta^{*r})$, obtain the pseudo-value function, $\tilde{V}^r(s, p^m; \theta^{*r})$ (see (4)). By averaging $\tilde{V}^r(s, p^m; \theta^{*r})$ across p^m 's, we integrate out prices and obtain $E_p \tilde{V}^r(s, p; \theta^{*r})$.

5. Compute the pseudo-likelihood, $\rho^{r+1}(\theta^r)$, based on an updated set of past pseudo-value functions.⁸

6. Go to iteration $r + 1$.

Note that when implementing the IJC algorithm, we propose to store the candidate parameter vectors and their expected pseudo-value functions, $\{\theta^{*l}, E_p \tilde{V}^l(s, p; \theta^{*l})\}_{l=r-N}^{r-1}$, instead of the accepted ones, $\{\theta^l, E_p \tilde{V}^l(s, p; \theta^l)\}_{l=r-N}^{r-1}$. If we store $\{\theta^l, E_p \tilde{V}^l(s, p; \theta^l)\}_{l=r-N}^{r-1}$, 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

⁸Two points should be noted here. First, in each iteration, there is a new expected pseudo value function evaluated at θ^{*r} . Second, this step needs not be carried out in a full-solution based Bayesian MCMC algorithm.

future value, it is useful to have a set of $E_p \tilde{V}^l$'s evaluated at parameter vectors that span the parameter spaces. Since θ^{*l} 's are drawn from a candidate generating function, it is much easier for us to achieve this goal by storing $E_p \tilde{V}^l$'s at θ^{*l} 's. Moreover, storing $\{E_p \tilde{V}^r(\cdot, p; \theta^{*l})\}_{l=r-N}^{r-1}$ will impose little extra costs. For each candidate parameter vector draw, θ^{*r} , we need to evaluate the expected future payoffs, $\hat{E}_p^r V(\cdot, p; \theta^{*r})$, to form the likelihood. As we have shown above, it will only take one extra step to obtain $E_p \tilde{V}^r(\cdot, p; \theta^{*r})$.

The above description of the implementation of the algorithm is slightly different from IJC, who propose to draw only one unobserved error term in each iteration. In the current application, that would be similar to drawing one price shock in each iteration, p^l , and store $H^r = \{\{\theta^{*l}, \tilde{V}^l(\cdot, p^l; \theta^{*l})\}_{l=r-N}^{r-1}, \rho^r(\theta^{r-1})\}$. The main difference between these two approaches is that the original IJC propose to store \tilde{V}^l instead of $E_p \tilde{V}^l$, and the expected future value approximation is,

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

instead of equation (6). The advantage of the original approach is that there is no need to compute $E_p \tilde{V}^r$ (see equation (7)). However, we believe that the approach proposed here would allow us to achieve the same level of precision in terms of integrating out prices by using a smaller N . As a result, it would also save us some time in computing the weighted average.

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 $E_p \tilde{V}^r(s, p; \theta^r)$, 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).

We also note that in step 5, it may not be worthwhile to compute the pseudo-likelihood,

$\rho^{r+1}(\theta^r)$, in each iteration. If we accept θ^{*r} , $\rho^r(\theta^{*r})$ can be used as a proxy for $\rho^{r+1}(\theta^r)$. Note that their calculations only differ in one past expected pseudo-value function. If we reject θ^{*r} , $\theta^r = \theta^{r-1}$. So we could use $\rho^r(\theta^{r-1})$ as a proxy for $\rho^{r+1}(\theta^r)$, and only compute $\rho^{r+1}(\theta^r)$ once every several successive rejections. According to our experiences, one can obtain a fairly decent reduction in computational time when using this approach.

3.3.2 Heterogeneous Consumers

We now present the implementation of the IJC algorithm when consumers have heterogeneous valuations for the reward ($\sigma_{G_j} > 0$). The vector of parameters to be estimated is $\theta = (\theta_1, \theta_2)$, where $\theta_1 = (\alpha_1, \alpha_2, \gamma, \beta)$ and $\theta_2 = (G_1, G_2, \sigma_{G_1}, \sigma_{G_2})$. We incorporate the Bayesian approach for random-coefficient models into the estimation steps for the homogeneous case. We use a normal prior on G_j and an inverted gamma prior on σ_{G_j} .

We use the Metropolis-Hastings algorithm to draw G_{ij}^l , which is consumer i 's valuation of reward at supermarket chain j from the population distribution of G_{ij} . This draw is regarded as an individual specific parameter. As a result, conditional on G_i , the value functions do not depend on θ_2 .

Each MCMC iteration mainly consists of three blocks.

- (i) Draw $G_j^r \sim f_G(G_j | \sigma_{G_j}^{r-1}, \{G_{ij}^{r-1}\}_{i=1}^I)$ and $\sigma_{G_j}^r \sim f_\sigma(\sigma_{G_j}^r | G_j^r, \{G_{ij}^{r-1}\}_{i=1}^I)$ for $j = 1, 2$ (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 individual parameters $G_{ij}^r \sim f_i(G_{ij} | G_j^r, \sigma_{G_j}^r, \theta_1^{r-1})$ by the independent Metropolis-Hastings algorithm, using $N(G_{ij}^r, (\sigma_{G_j}^r)^2)$ as a proposal distribution for all i and $j = 1, 2$.
- (iii) Draw $\theta_1^r \sim f_{\theta_1}(\cdot | G_i^r)$ using the random-walk Metropolis-Hastings algorithm.

The details of the estimation steps are described as follows. Step 2-3 belong to block (i), step 4 belongs to block (ii) and step 5 belongs to block (iii).

1. Suppose that we are at iteration r . Start with

$$H^r = \{\{\theta^{*l}, G_{i^*}^l, E_p \tilde{V}^l(\cdot, p; G_{i^*}^l, \theta_1^{*l})\}_{l=r-N}^{r-1}, \{\rho_i^r(G_i^{r-1}, \theta_1^{r-1})\}_{i=1}^I\},$$

where I is the number of consumers; N is the number of past iterations used for the expected future value approximation; $i^* = r - I * \text{int}(\frac{r-1}{I})$ where $\text{int}(\cdot)$ is an integer function that converts any real number to an integer by discarding its value after the decimal place.

2. Draw G_j^r (population mean of G_{ij}) from the posterior density (normal) computed by $\sigma_{G_j}^{r-1}$ and $\{G_{ij}^{r-1}\}_{i=1}^I$.
3. Draw $\sigma_{G_j}^r$ (population variance of G_{ij}) from the posterior density (inverse gamma) computed by G_j^r and $\{G_{ij}^{r-1}\}_{i=1}^I$.
4. For each i , use the Metropolis-Hastings algorithm to draw G_{ij}^r .

- (a) Use the prior on G_{ij} (i.e., $N(G_j^r, (\sigma_{G_j}^r)^2)$) as the proposal distribution function to draw G_{ij}^{*r} .

- (b) Compute the pseudo-likelihood for consumer i at G_i^{*r} , i.e., $\rho_i^r(G_i^{*r}, \theta_1^{r-1})$. The pseudo-likelihood is expressed as

$$\rho_i^r(G_i^{*r}, \theta_1^{r-1}) = \prod_{t=1}^{T_i} \prod_{j=0}^2 \left(\frac{\exp(\tilde{V}_j^r(s_{it}, p_{ijt}^d; G_i^{*r}, \theta_1^{r-1}))}{\sum_{k=0}^2 \exp(\tilde{V}_k^r(s_{it}, p_{ikt}^d; G_i^{*r}, \theta_1^{r-1}))} \right)^{I_{buy,ijt}^d}.$$

To obtain \tilde{V}_j^r , we need $\hat{E}_p^r V(\cdot, p; G_i^{*r}, \theta_1^{r-1})$. Note that G_{ij}^{*r} will only affect $\hat{E}_p^r V(s, p; G_i^{*r}, \theta_1^{r-1})$ for consumer i and not for other consumers. Thus when approximating $\hat{E}_p^r V(s, p; G_i^{*r}, \theta_1^{r-1})$,

we use the weighted average of $\{E_p \tilde{V}^l(s, p; G_{i^*}^l, \theta_1^{*l})\}_{l=r-N}^{r-1}$, treating G_i as one of the parameters when computing the weights. In the case of independent kernels, we set

$$\hat{E}_p^r V(s, p; G_i^{*r}, \theta_1^{r-1}) = \sum_{l=r-N}^{r-1} E_p \tilde{V}^l(s, p; G_{i^*}^l, \theta_1^{r-1}) \frac{K_h(\theta_1^{r-1} - \theta_1^{*l}) K_h(G_i^{*r} - G_{i^*}^l)}{\sum_{k=r-N}^{r-1} K_h(\theta_1^{r-1} - \theta_1^{*k}) K_h(G_i^{*r} - G_{i^*}^k)}. \quad 9$$

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

$$\begin{aligned} \lambda &= \min \left(\frac{\pi(G_i^{*r}) \cdot \rho_i^r(G_i^{*r}, \theta_1^{r-1}) \cdot q(G_i^{*r}, G_i^{r-1})}{\pi(G_i^{r-1}) \cdot \rho_i^r(G_i^{r-1}, \theta_1^{r-1}) \cdot q(G_i^{r-1}, G_i^{*r})}, 1 \right) \\ &= \min \left(\frac{\rho_i^r(G_i^{*r}, \theta_1^{r-1})}{\rho_i^r(G_i^{r-1}, \theta_1^{r-1})}, 1 \right), \end{aligned}$$

where the second equality follows from the fact that the proposal distribution is set to the prior on G_{ij} , i.e., $q(x, y) = \pi(y)$. If accept, set $G_i^r = G_i^{*r}$; otherwise, set $G_i^r = G_i^{r-1}$.

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

5. Use the Metropolis-Hastings algorithm to draw $\theta_1^r = (\alpha_1^r, \alpha_2^r, \gamma^r, \beta^r)$ conditional on G_{ij}^r .

(a) Draw $\theta_1^{*r} = (\alpha_1^{*r}, \alpha_2^{*r}, \gamma^{*r}, \beta^{*r})$ (candidate parameter vector).

(b) We then compute the pseudo-likelihood conditional on $(\alpha_1^{*r}, \alpha_2^{*r}, \gamma^{*r}, \beta^{*r})$ and $\{G_i^r\}_{i=1}^I$, based on the pseudo expected future values. The pseudo-likelihood, $\rho^r(\{G_i^r\}_{i=1}^I, \theta_1^{*r})$, is expressed as

$$\rho^r(\{G_i^r\}_{i=1}^I, \theta_1^{*r}) = \prod_{i=1}^I \prod_{t=1}^{T_i} \prod_{j=0}^2 \left(\frac{\exp(\tilde{V}_j^r(s_{it}, p_{ijt}^d; G_i^r, \theta_1^{*r}))}{\sum_{k=0}^2 \exp(\tilde{V}_k^r(s_{it}, p_{ikt}^d; G_i^r, \theta_1^{*r}))} \right)^{I_{buy,ijt}^d}.$$

To obtain \tilde{V}_j^r , we need to calculate $\hat{E}_p^r V(s, p; G_i^r, \theta_1^{*r})$, which is a weighted average of the past value functions, $\{E_p \tilde{V}^l(s, p; G_{i^*}^l, \theta_1^{*l})\}_{l=r-N}^{r-1}$. In computing the weights for

⁹Note that $\{K_h(\theta_1^{r-1} - \theta_1^{*l})\}_{l=r-N}^{r-1}$ is common across consumers. Therefore, one can calculate it outside the loop that indexes consumers when programming this part.

past value functions, we treat G_i as a parameter. Note that in the case of independent kernels, (6) becomes

$$\hat{E}_p^r V(s, p; G_i^r, \theta_1^{*r}) = \sum_{l=r-N}^{r-1} E_p \tilde{V}^l(s, p; G_{i^*}^l, \theta_1^{*l}) \frac{K_h(\theta_1^{*r} - \theta_1^{*l}) K_h(G_i^r - G_{i^*}^l)}{\sum_{k=r-N}^{r-1} K_h(\theta_1^{*r} - \theta_1^{*k}) K_h(G_i^r - G_{i^*}^k)}.^{10}$$

We then determine whether or not to accept α_1^{*r} , α_2^{*r} , γ^{*r} , and β^{*r} . The acceptance probability is similar to that in step 3 of section 3.3.1.

6. Computation of the expected pseudo-value function, $E_p \tilde{V}^r(s, p; G_{i^*}^r, \theta_1^{*r})$.

(a) Make M draws of prices, $\{p^m\}_{m=1}^M$, from the price distribution.

(b) Compute $\tilde{V}_0^r(s; \theta_1^{*r})$, $\tilde{V}_1^r(s, p_1^m; G_{i^*}^r, \theta_1^{*r})$ and $\tilde{V}_2^r(s, p_2^m; G_{i^*}^r, \theta_1^{*r})$, using the pseudo expected future values computed in Step 5(b).

(c) Given $\tilde{V}_0^r(s; \theta_1^{*r})$, $\tilde{V}_1^r(s, p_1^m; G_{i^*}^r, \theta_1^{*r})$ and $\tilde{V}_2^r(s, p_2^m; G_{i^*}^r, \theta_1^{*r})$, obtain the pseudo-value function, $\tilde{V}^r(s, p^m; G_{i^*}^r, \theta_1^{*r})$. By averaging $\tilde{V}^r(s, p^m; G_{i^*}^r, \theta_1^{*r})$ across p^m 's, we integrate out prices and obtain $E_p \tilde{V}^r(s, p; G_{i^*}^r, \theta_1^{*r})$.

7. Compute the pseudo-likelihood, $\rho_i^{r+1}(G_i^r, \theta_1^r) \forall i$, based on an updated set of past expected pseudo-value functions.

8. Go to iteration $r + 1$.

In step 1 of the procedure described above, we pick one consumer in each iteration and store his/her pseudo-value function. Then, we use this pooled set of past expected pseudo-value functions across consumers to approximate the expected future values for all consumers.

This is slightly different from IJC, who originally propose to store an individual-specific set

¹⁰Note that $\{K_h(\theta_1^{*r} - \theta_1^{*l})\}_{l=r-N}^{r-1}$ is common across consumers. Therefore, one can compute it outside the loop that indexes consumers to save computational time.

of past pseudo-value functions for each consumer. That is, in each iteration we store $H^r = \{\theta^{*l}, \{G_i^l, E_p \tilde{V}^l(\cdot, p; G_i^l, \theta^{*l})\}_{i=1}^I\}_{l=r-N}^{r-1}$, and use $\{E_p \tilde{V}^l(\cdot, p; G_i^l, \theta^{*l})\}_{l=r-N}^{r-1}$ to approximate consumer i 's expected future values. One advantage of this approach is that the past pseudo-value functions used in the expected future value approximation are more relevant to each consumer i , because they are evaluated at G_i^l 's, which represent the posterior distribution of consumer i 's value for the gift, and should be closer to G_i^{*r} . Note that this is not the case when we pool past pseudo-value functions across consumers because different consumers may have very different values of G_i . This suggests that if we store past pseudo-value functions individually, we may be able to reduce N in order to achieve the same level of precision for the expected future value approximation. This in turn should reduce the computation time. But one drawback is that we need much more memory to store past pseudo-value functions individually, although this may not be an important concern in the near future, given that the price of computer memory has been decreasing rapidly over time.

Note that we only describe steps 2 and 3 briefly here. For the details of these two steps, we refer readers to Chapter 12 of Train (2003), or Rossi et al. (2005). When implementing step 5, it could be more efficient to separate them 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.

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 (x) increases – in IJC, the number of parameters corresponds to the number of explanatory variables in the traditional non-parametric 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 an arbitrary regression 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.¹¹ The root of this problem is due to the bias-variance trade-off. In general, when the kernel bandwidth is small, the effective number of sample points available around x that influence the prediction would also be small, making the prediction highly sensitive to that particular sample, i.e., yielding to high variance. When the kernel bandwidth is large, the prediction becomes overly smooth, i.e., yielding to high bias.

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 regressions by storing/using more past pseudo-value functions by increasing N . This is similar to the advantage of using the standard MCMC method to draw from the posterior distribution, where the econometrician can control the number of iterations that requires to obtain convergence. In practice, we expect that N may need to increase with the number of parameters in the model to ensure convergence. As a result, it would also take more time to compute one iteration if the model becomes more complicated. One way to alleviate this issue

¹¹Note that 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 the number of values for each state variable.

is to estimate a few parameters at a time. For complicated DDP models with many parameters, this is a common practice adopted by empirical researchers.

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 estimating a well-identified model and the data have sufficient variations, 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 expected 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 past expected 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, $\bar{\theta}$, and solve for the exact value function at $\bar{\theta}$ numerically. Then they can compare the distance between the pseudo-value function and the exact value function at $\bar{\theta}$. If the distance is larger than what the researcher would accept, increase N' . Then use N' past pseudo-value functions to compute summary statistics and use 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 again 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 how to implement the IJC algorithm and investigate some of its properties, we conduct three 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 experiment, we are interested in estimating 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, flat prior is used. In addition, we use a random-walk proposal function. Table 1 summarizes the estimation results, and Figure 3 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.

In the second experiment, 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: $\alpha_1 = \alpha_2 = 0.0$, $G_1 = 1.0$, $G_2 = 5.0$,

$\sigma_{G_1} = 0.0$, $\sigma_{G_2} = 1.0$, γ , and $\beta = 0.6$ or 0.8 . Again, 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 $\alpha_1, \alpha_2, G_1, \gamma$, 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, $IG(\nu_0, s_0)$ (i.e., setting $s_0 = 1, \nu_0 \rightarrow 1$). Table 2 shows the estimation results, and Figure 4 plots the simulated draws of parameters for $\beta = 0.8$. The IJC algorithm again 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.

Potential Reduction in Computation Time

To learn more about the potential gain of IJC in terms of computational time, we compute the time per iteration and compare IJC's Bayesian MCMC algorithm with the full-solution based Bayesian MCMC algorithm for both homogeneous model and heterogeneous model. In the full-solution based Bayesian algorithm, 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 3 summarizes the results based on the average computation time based on 1,000 iterations. The estimation is done based on a C program running in a linux workstation with Intel Core 2 Duo E4400 2GHz processor. Note that in the full-solution based Bayesian algorithm, the computation time will increase with β . This is because the number of iterations required for convergence in a contraction mapping increases with β (i.e., the modulus). However, the computation time will not be influenced by the value of β in the IJC algorithm.

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 perform step 5 in the homogeneous case, and step 7 in the heterogeneous case.¹² However, when $\beta = 0.98$, IJC algorithm is 40% faster than the full-solution 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 will then be roughly (time per contraction mapping) X 1,000. 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.¹³ Using the IJC's Bayesian MCMC algorithm, it would take less than 2.5 days ($\simeq 18.4 \times 10,000$ seconds) to obtain 10,000 iterations.

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 Monte Carlo experiment, using the homogeneous model with $\beta = 0.98$, we investigate how changes in N influence the speed of convergence and the posterior 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 4 shows the

¹²In this exercise, we perform step 5 in the homogeneous case and step 7 in the heterogeneous case every time a candidate parameter vector is rejected.

¹³Depending on the convergence rate, the number of iterations required for Bayesian estimation could be higher than 10,000.

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 5, and G_1 and G_2 in Figure 6. 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.

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 .

We now turn to discuss how to extend the IJC algorithm to (i) conduct policy experiments, and (ii) allow for continuous state space. We will also comment on the choice of kernels.

5 Extensions

5.1 Conducting Policy Experiments

The output of the IJC algorithm is the posterior distribution for the parameters of the model, along with a set of value functions (and expected future value functions) estimates associated with each parameter vector. One natural question to ask is: What if we are interested in conducting a policy experiment that involves changing a policy parameter by a large amount of percentage (e.g., increase the cost of entry by 100% percentage), such that the new parameter vectors do not belong to the support of the posterior distribution or belong to the tail of the

posterior distribution? If one uses the expected pseudo-value functions stored in the IJC algorithm to approximate the expected future values at the new policy parameter vectors, it would likely yield poor results because the simulated draws of the parameter vectors from the algorithm are very far away from the ones that we are interested in the policy experiment. Recomputing the value functions at the set of new policy parameter vectors, taking the uncertainty about the parameter values into account, appears to be a formidable task in terms of computational burden.¹⁴ In fact, this criticism applies even if one uses full-solution based Bayesian MCMC algorithm.

Here we propose a *new* modification of the IJC algorithm, which allows us to use its outputs to generate a set of value functions and choice probabilities that are evaluated at a set of simulated new policy parameter vectors. Suppose that the researcher is interested in the effect of changing the mean value of the gift from G_j , to \acute{G}_j , where $\acute{G}_j = (1 + t)G_j$, $\forall j$. The modified procedure needs to store the following additional information: $\{\acute{G}_{i^*}^l, E_p \tilde{V}^l(\cdot, p; \acute{G}_{i^*}^l, \theta_1^{*l})\}_{l=L-N+1}^L$, where L is the total number of MCMC iterations. This requires us to slightly modify the solution/estimation algorithm for the model with unobserved heterogeneity discussed in section 3.3.2. We first add the following step after step 5(b). We call it Step 5(c).

Step 5(c). Approximate the expected future values at $(\acute{G}_{i^*}^r, \theta_1^{*r})$ as follows:

$$\hat{E}_p^r V(s, p; \acute{G}_{i^*}^r, \theta_1^{*r}) = \sum_{l=r-N}^{r-1} E_p \tilde{V}^l(s, p; \acute{G}_{i^*}^l, \theta_1^{*l}) \frac{K_h(\theta_1^{*r} - \theta_1^{*l}) K_h(\acute{G}_{i^*}^r - \acute{G}_{i^*}^l)}{\sum_{k=r-N}^{r-1} K_h(\theta_1^{*r} - \theta_1^{*k}) K_h(\acute{G}_{i^*}^r - \acute{G}_{i^*}^k)}.$$

Then, we add step 6(d), which uses this pseudo expected future value to obtain $E_p \tilde{V}^r(s, p; \acute{G}_{i^*}^r, \theta_1^{*r})$.

Note that step 6(d) is similar to step 6(c).

¹⁴If a researcher takes posterior means as point estimates for the true parameter values, and is only interested in the effect of the policy experiment by changing the point estimate of a policy parameter (e.g., Osborne 2008), the computational burden of this task would be much lower. This is because he/she only needs to solve the value function once at the new parameter vector. Nevertheless, this approach does not tell us the standard deviation of the effects of the policy experiment, which would be important to know if some parameters of the model are not estimated very precisely (i.e., with relatively large posterior standard deviation).

Once the modified IJC algorithm converges, we will have $\{\{G_i^l\}_{i=1}^I, \theta_1^l\}_{l=1}^L$ as well as $\{\acute{G}_{i^*}^l, \theta_1^{*l}, E_p \tilde{V}^l(\cdot, p; \acute{G}_{i^*}^l, \theta_1^{*l})\}_{l=L-N+1}^L$ as the outputs. To conduct the policy experiment, say for consumer i : (a) use the most recent D draws of $\{G_i^r, \theta_1^r\}_{r=L-D+1}^L$ to set the simulated policy parameter vectors as follows: $\acute{G}_i^r = (1+t)G_i^r$; (b) use $\{E_p \tilde{V}^l(\cdot, p; \acute{G}_{i^*}^l, \theta_1^{*l})\}_{l=L-N+1}^L$ to form pseudo expected future value at $\{\acute{G}_i^r, \theta_1^r\}_{r=L-D+1}^L$, and then obtain their pseudo value function $\{\tilde{V}_j^r(\cdot, \cdot; \acute{G}_i^r, \theta_1^r)\}_{r=L-D+1}^L$ and the corresponding choice probabilities $\{Prob^r(j|\acute{G}_i^r, \theta_1^r)\}_{r=L-D+1}^L$; (c) use $\{Prob^r(j|\acute{G}_i^r, \theta_1^r)\}_{r=L-D+1}^L$ to compute the posterior mean and standard deviation of the policy experiment outcomes (or other statistics of interests).

As shown above, this procedure is relatively straightforward to implement, and requires very little extra programming efforts. Moreover, it only increases the computational burden of each iteration slightly. For instance, the average computation times per iteration for the heterogeneous model are only 0.9s and 1.74s for $N = 500$ and 1000, respectively. Finally, we note that there is a limitation of this modified procedure: we need to know the magnitude of the change in the policy parameter before seeing the estimation results. Sometimes researchers may not be able to determine this until they obtain the parameter estimates.

5.2 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 equation (5) 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, $p^r = (p_1^r, p_2^r)$, 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 p^r , $\tilde{V}^r(s, p^r; \theta^r)$ for all s . Thus, H^r in step 1 of section 3.3.1 needs to be changed to

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

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

$$\hat{E}_{p'}^r[V(s', p'; \theta^r)|p] = \sum_{l=r-N}^{r-1} \tilde{V}^l(s', p^l; \theta^{*l}) \frac{K_h(\theta^r - \theta^{*l})f(p^l|p; \theta_p)}{\sum_{k=r-N}^{r-1} K_h(\theta^r - \theta^{*k})f(p^k|p; \theta_p)}. \quad (8)$$

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 .

To use this approach, the procedure for obtaining the pseudo-value function in step 4 of section 3.3.1 will need to be modified slightly. We store $\tilde{V}^r(s, p^r; \theta^{*r})$ instead of $E_p \tilde{V}^r(s, p; \theta^{*r})$. Specifically, the pseudo-value function at p^r (and θ^{*r}) is computed as follows. For each s ,

$$\tilde{V}^r(s, p^r; \theta^{*r}) = \log(\exp(\tilde{V}_0^r(s; \theta^{*r})) + \exp(\tilde{V}_1^r(s, p_1^r; \theta^{*r})) + \exp(\tilde{V}_2^r(s, p_2^r; \theta^{*r}))),$$

where

$$\begin{aligned}\tilde{V}_j^r(s, p_j^r; \theta^{*r}) &= \begin{cases} \alpha_j - \gamma p_j^r + G_j + \beta \hat{E}_{p'}^r[V(s', p'; \theta^{*r})|p^r] & \text{if } s_j = \bar{S}_j - 1, \\ \alpha_j - \gamma p_j^r + \beta \hat{E}_{p'}^r[V(s', p'; \theta^{*r})|p^r] & \text{otherwise,} \end{cases} \\ \tilde{V}_0^r(s; \theta^{*r}) &= \beta \hat{E}_{p'}^r[V(s', p'; \theta^{*r})|p^r].\end{aligned}$$

The pseudo expected future values above are computed using equation (8).

We should point out that if ones 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 $M * \bar{S}_1 * \bar{S}_2$, while the IJC's random grid approach will keep it at $\bar{S}_1 * \bar{S}_2$. 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 pseudo-value functions. This allows us to compute the pseudo-value function at only one grid point, p^r , in each iteration.

5.3 Choice of Kernels

It should 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 (2008) extends IJC's method by using the "nearest neighbors" kernel instead of Gaussian kernel, and allowing the error terms to be serially correlated. One advantage of using the "nearest neighbors" kernel is that any past value functions that are evaluated at parameter vectors that are far away from the current one will not be used in the approximation. But one drawback is that it may need to store many more past value functions, and the computer program needs to check whether each past parameter vector belongs to the defined neighborhood or not – this could be quite computationally burdensome.

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.

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. Ching et al. (2009) have also successfully estimated a learning and forgetting model where consumers are forward-looking.

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. 2001. A New Use of Importance Sampling to Reduce Computational Burden in Simulation Estimation. Working paper, Department of Economics, UCLA.
- Ackerberg, Daniel A. 2003. Advertising, Learning, and Consumer Choice in Experience Good Markets: An Empirical Examination. *International Economic Review* **44**(3) 1007–1040.
- 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.
- 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, 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. 2002. 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, Füsün, 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., 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.
- 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. 2009. Bayesian Estimation of Dynamic Discrete Choice Models. Forthcoming in *Econometrica*. Available at SSRN: <http://ssrn.com/abstract=1118130>.
- 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.
- 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 Retention. *Journal of Marketing Research* **41**(3) 281–292.

- McCulloch, Robert, Peter E. Rossi. 1994. An Exact Likelihood Analysis of the Multinomial Probit Model. *Journal of Econometrics* **64** 207–240.
- Norets, Andriy. 2008. Inference in Dynamic Discrete Choice Models with Serially Correlated Unobserved State Variables. Mimeo, Department of Economics, Princeton University.
- Osborne, Matthew. 2008. Consumer Learning, Switching Costs, and Heterogeneity: A Structural Examination. Working paper, U.S. Department of Justice.
- 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.

Yang, Botao, Andrew Ching. 2009. Dynamics of Consumer Adoption Decisions of Financial Innovation: The Case of ATM Cards in Italy. Working paper, Rotman School of Management, University of Toronto.

Table 1: Estimation Results: Homogeneous Model

parameter	TRUE	$\beta = 0.6$		$\beta = 0.8$	
		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 $j = 1, 2$.

Turning parameters: $N = 1,000$ (number of past pseudo-value functions used for expected future value approximations), $h = 0.01$ (bandwidth).

Table 2: Estimation Results: Heterogeneous Model

parameter	TRUE	$\beta = 0.6$		$\beta = 0.8$	
		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_{G_2} (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: $\bar{S}_1 = 2$, $\bar{S}_2 = 4$, $\bar{p} = 1.0$, $\sigma_p = 0.3$, $\sigma_{G_1} = 0$.

Turning parameters: $N = 1,000$ (number of past pseudo-value functions used for expected future value approximations), $h = 0.01$ (bandwidth).

Table 3: Computation Time Per MCMC Iteration (in seconds)

algorithm	Homogeneous Model			Heterogeneous Model		
	$\beta = 0.6$	$\beta = 0.8$	$\beta = 0.98$	$\beta = 0.6$	$\beta = 0.8$	$\beta = 0.98$
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 4: The Impact of N

parameter	TRUE	N=100		N=1000	
		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 $j = 1, 2$, $\beta = 0.98$.

Turning parameters: $h = 0.01$ (bandwidth).

Figure 1: Choice probabilities across states for different discount factors

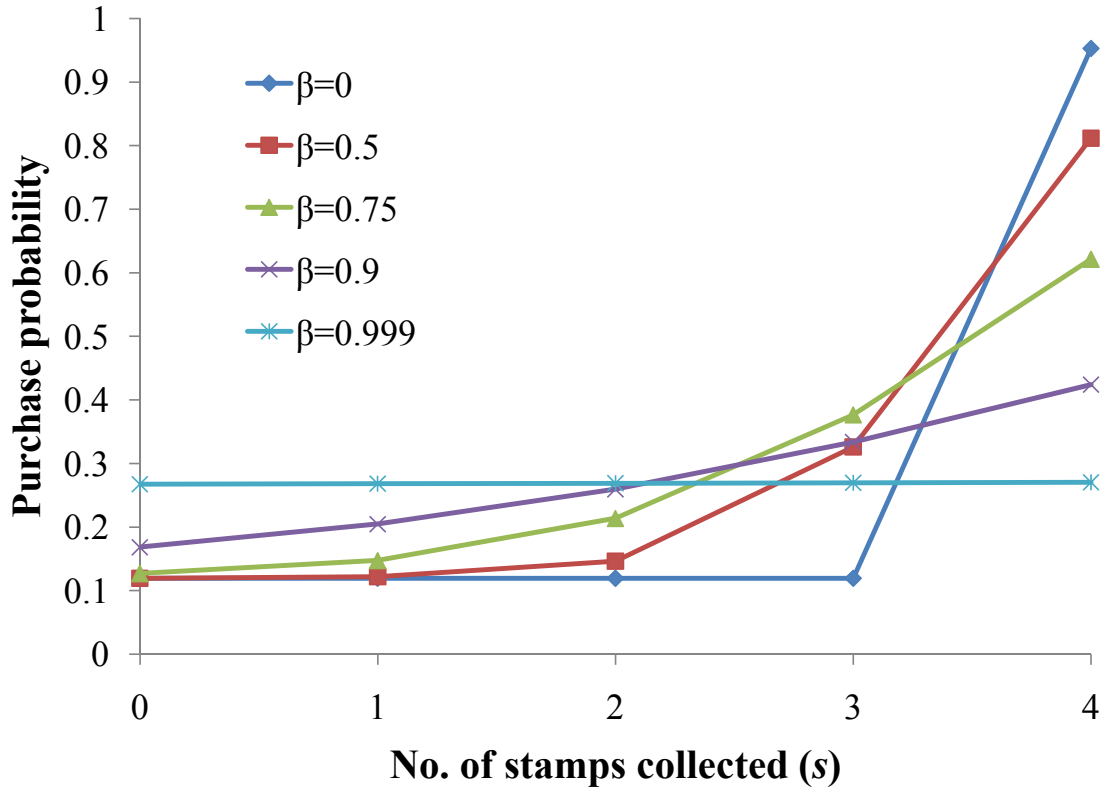


Figure 2: Choice probabilities for different discount factors across states

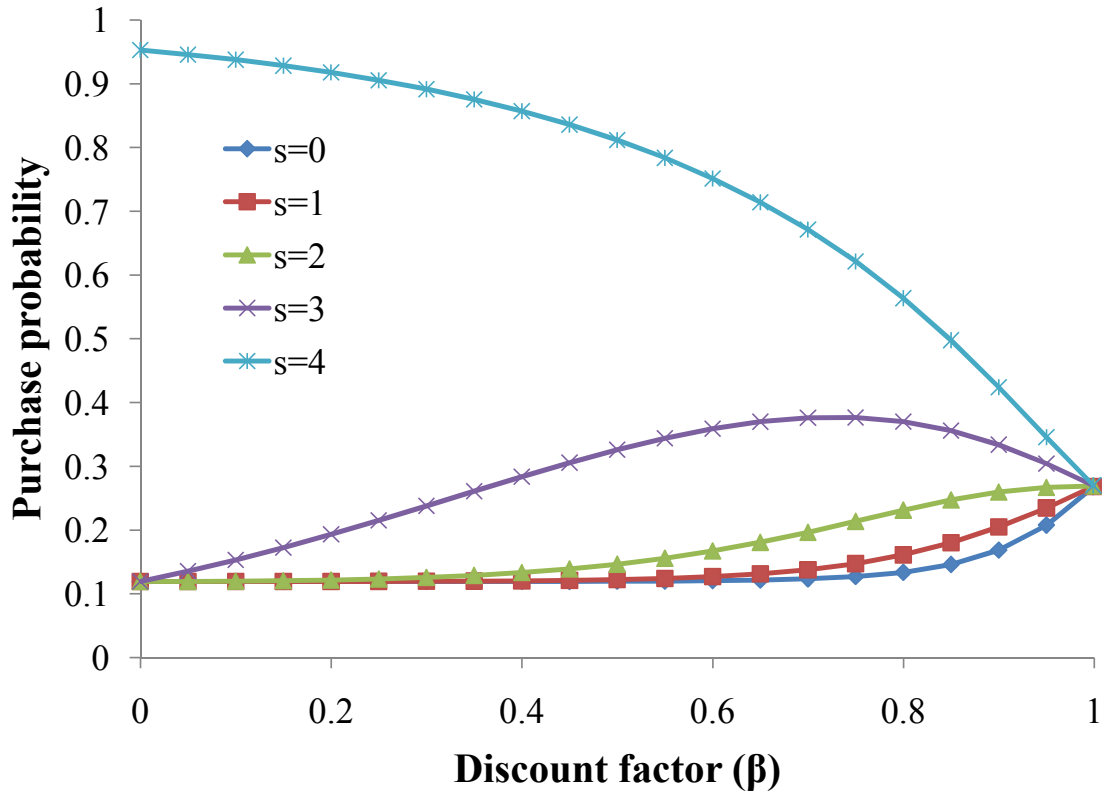
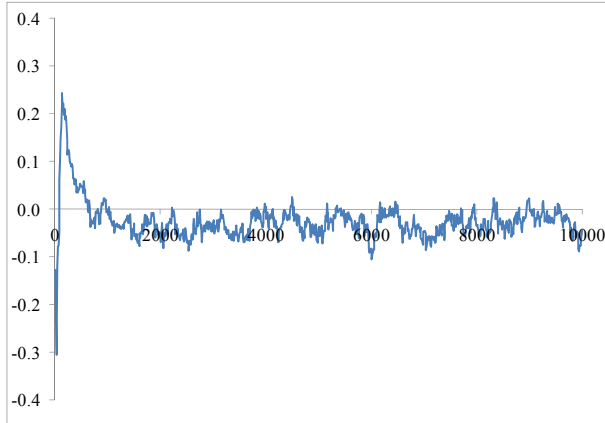
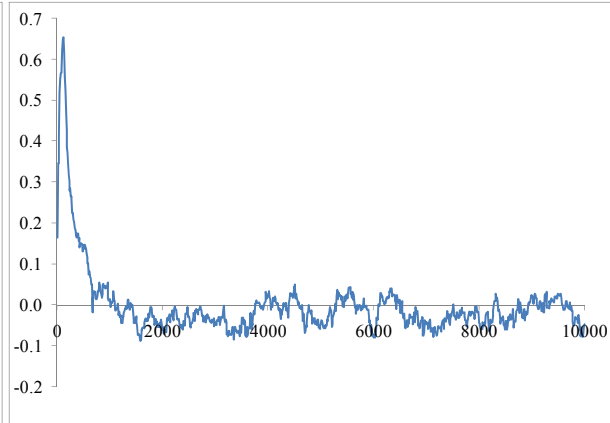


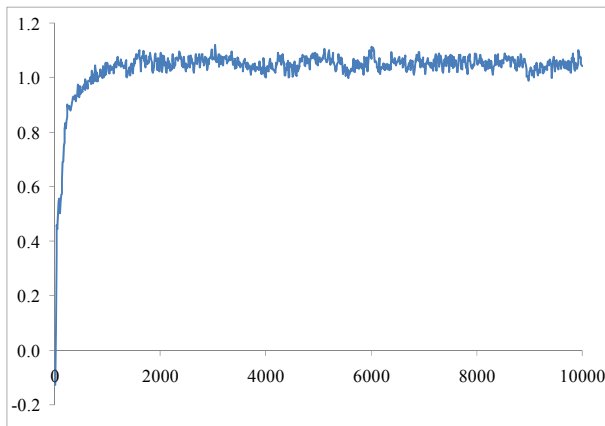
Figure 3: MCMC plots: Homogeneous Model with $\beta = 0.8$



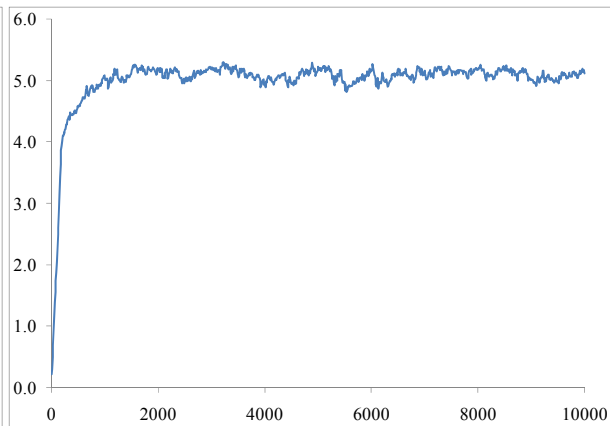
α_1 (true value = 0.0)



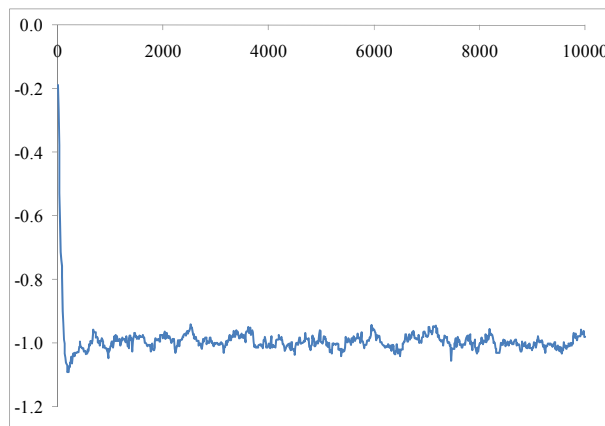
α_2 (true value = 0.0)



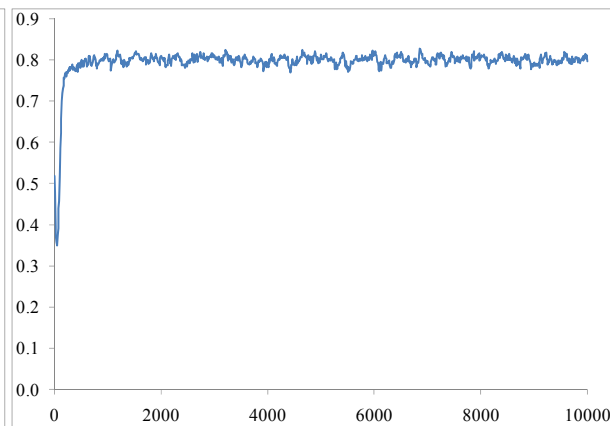
G_1 (true value = 1.0)



G_2 (true value = 5.0)

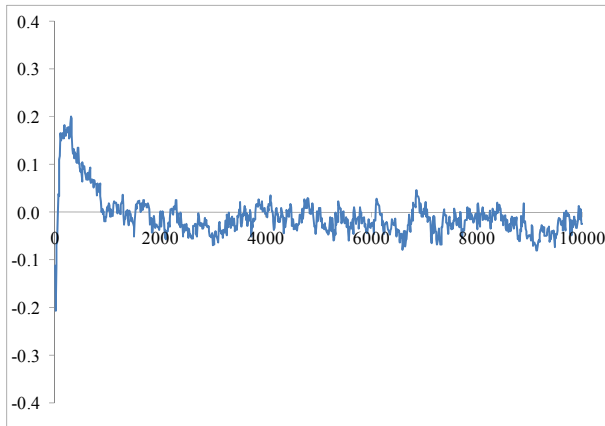


γ (true value = -1.0)

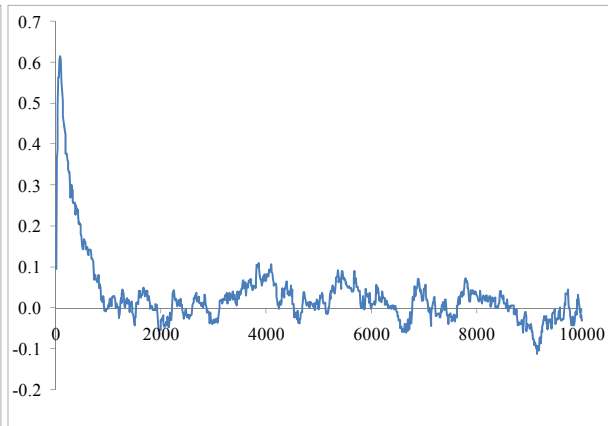


β (true value = 0.8)

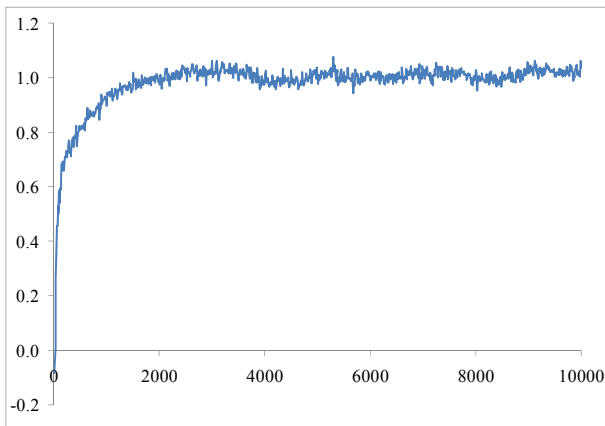
Figure 4: MCMC plots: Heterogeneous Model with $\beta = 0.8$



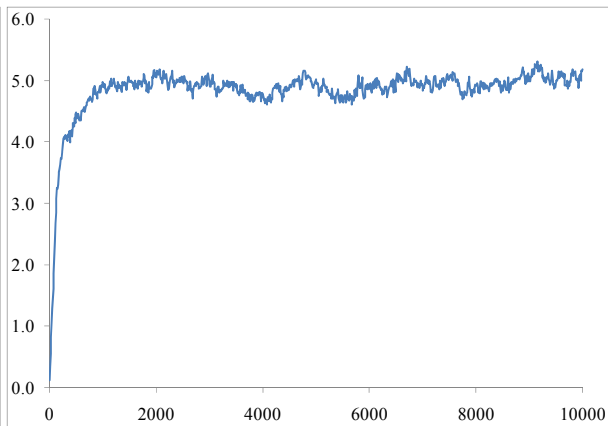
α_1 (true value = 0.0)



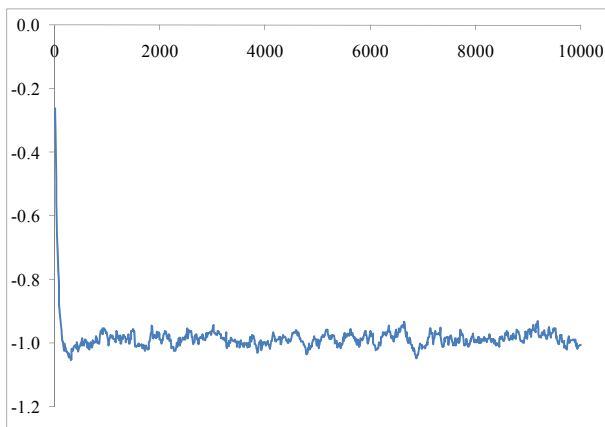
α_2 (true value = 0.0)



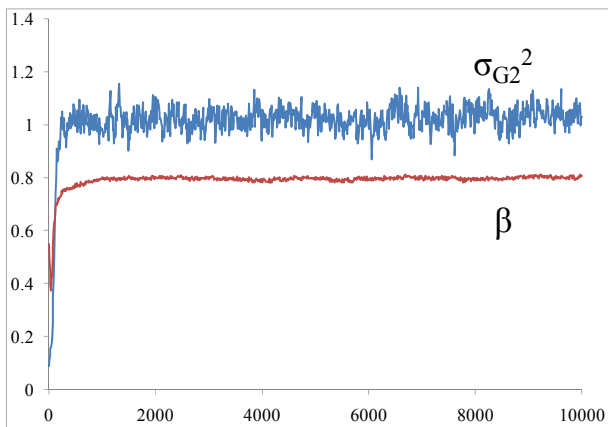
G_1 (true value = 1.0)



G_2 (true value = 5.0)



γ (true value = -1.0)



β (true value = 0.8)

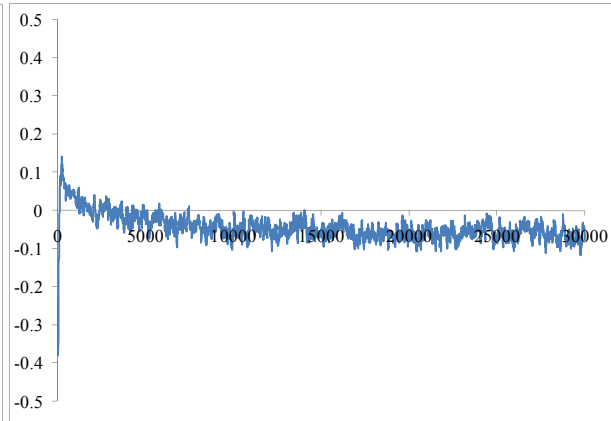
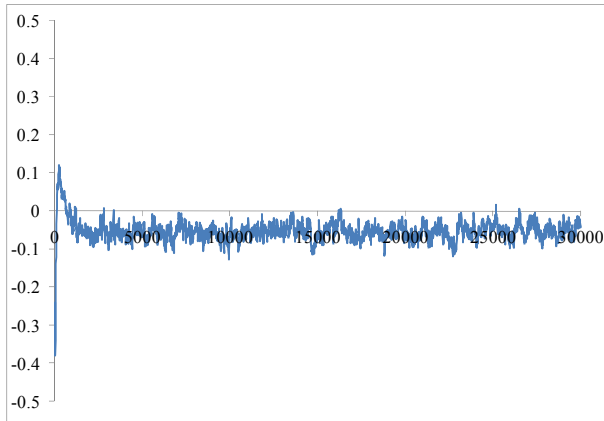
$\sigma_{G_2}^2$ (true value = 1.0)

Figure 5: MCMC plots: Impact of N on α_1 and α_2 when $\beta = 0.98$

$N = 100$

$N = 1000$

α_1 (true value = 0.0)



α_2 (true value = 0.0)

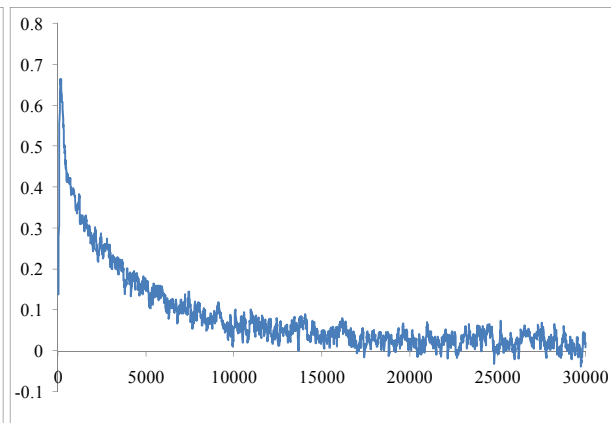
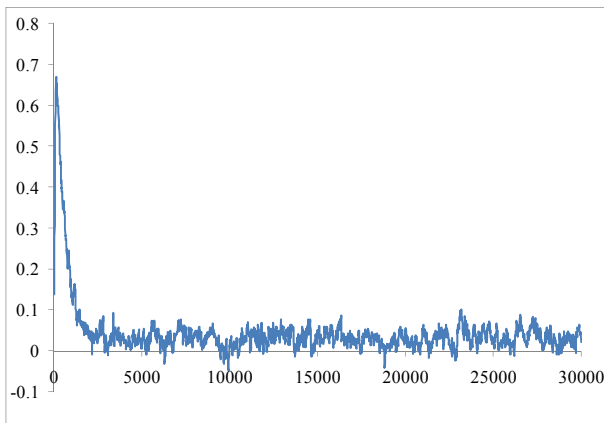
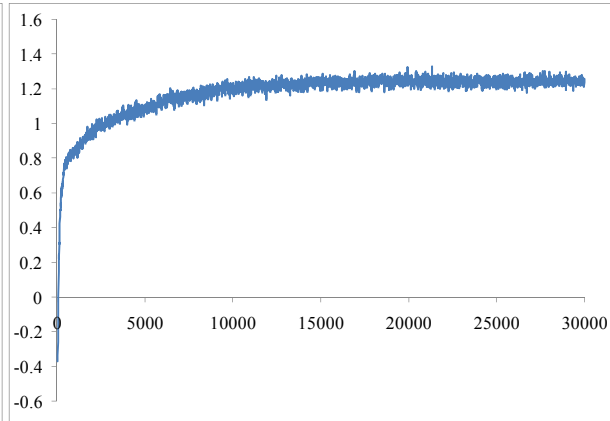
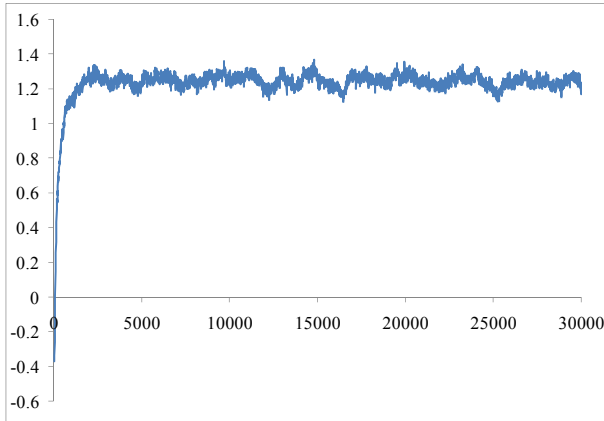


Figure 6: MCMC plots: Impact of N on G_1 and G_2 when $\beta = 0.98$

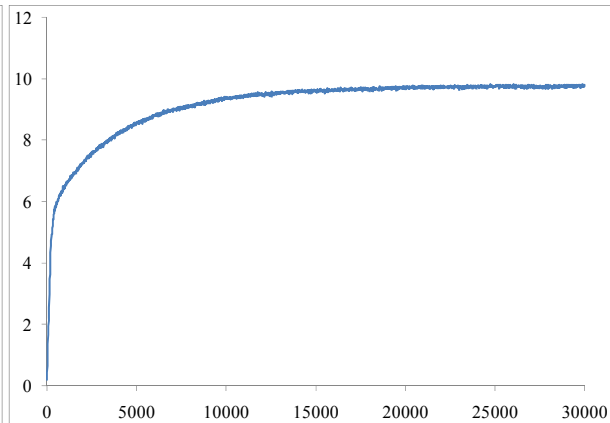
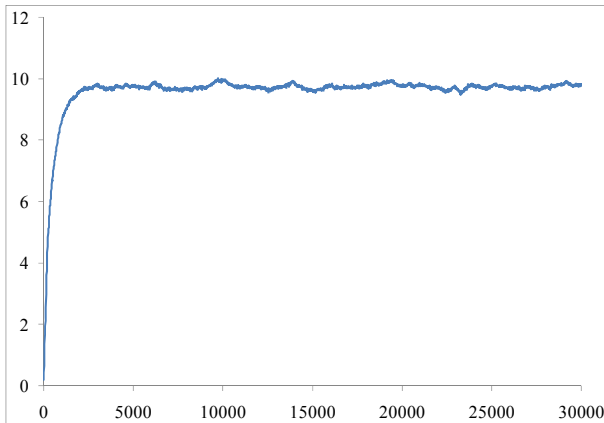
$N = 100$

$N = 1000$

G_1 (true value = 1.0)



G_2 (true value = 10.0)



Appendix

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

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

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

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

Thus

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

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 $\beta \rightarrow 1$, we have

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

Thus, when β approaches one, we observe a flat choice probability across s .

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 $\beta \rightarrow 1$, $(R(s, \beta) - W(s, \beta))$ 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

$$V_1(s = 0) = \alpha + \beta EV(s = 1)$$

$$V_0(s = 0) = \beta EV(s = 0).$$

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

$$V_1(s = 1) = \alpha + G + \beta EV(s = 0)$$

$$V_0(s = 1) = \beta EV(s = 1).$$

Note first that

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

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

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

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

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