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

Andrew Ching(U of Toronto), Susumu Imai(Queen's U),
Masakazu Ishihara(U of Toronto), Neelam Jain(NIU)

June 5, 2009

Motivation

- Maximum likelihood/GMM is a common way to estimate Discrete Choice Dynamic Programming Models (DDP). However, it is hard to implement due to
  - the curse of dimensionality in the state space.
  - the complexity of the likelihood/GMM objective function makes it difficult to search for the global maximum/minimum.
- Due to the computational burdens (in particular for models w' random coefficients), there has been no work on estimating a full solution DDP model using Bayesian econometrics.
- Imai, Jain, Ching (forthcoming) (IJC) propose a new MCMC algorithm that alleviates the burden of estimating infinite horizon DDP models.
- The purpose of this paper is to discuss the practical aspects of implementing the IJC algorithm, and provide a step-by-step guide.

Roadmap

- Key features of IJC algorithm.
- A simple dynamic store choice model.
- Brief review of dynamic programming.
- IJC's new step to obtain the expected future value.
- Estimation Results
- Extensions: (i) policy experiments; (ii) continuous state space.
- Conclude.

Highlights of IJC method

- In the nested fixed point algorithm (Rust 1987), the value functions need to be solved at each trial parameter vector.
- IJC make use of the past outcomes of the estimation algorithm, $\{V^l(s, \epsilon^l; \theta^l)\}_{l=l-N}^{r-1}$, to approximate $E_\epsilon[V(s, \epsilon; \theta)]$ (take advantage of the continuity of V in θ). In contrast, the conventional method simply throws away this information.
- Applicable to general infinite horizon stationary DDP models.
- It estimates and solves the DDP model simultaneously.
  - Computational burden of IJC algorithm of each iteration is comparable to that of estimating a standard static discrete choice model.
- It can incorporate unobserved heterogeneities (random coefficients).

Model (An Example)

A simple dynamic model of store choice with reward programs.

- There are two supermarkets in a city $(j = 1, 2)$.
- Each store offers a stamp card, which can be exchanged for a gift upon completion.
- Consumers get one stamp for each visit with a purchase.
- Reward programs at the two supermarkets differ in terms of: (i) the number of stamps required for a reward $(\bar{S}_j)$, and (ii) the mean value of the reward (G_j).
- $G_{ij} \sim N(G_j, \sigma^2_{G_j})$, i indexes consumers.
- Once consumers receive a gift, they will start with a blank stamp card again in the next period.

Model (cont'd)

- $p_{ijt} \sim i.i.d.\ N(\underline{p}, \sigma_p^2)$.
- Let $s_i = (s_{i1}, s_{i2})$ be the number of stamps collected.
- Consumer i's single period utility of visiting supermarket j is:

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

- The utility of "not shopping" is normalized to zero, i.e., $U_{i0t}(s) = 0$.
- We assume that ϵ follows i.i.d. extreme value distribution.

Model (cont'd)

- The consumer i's objective is to choose a sequence of store choices to maximize the sum of the present discounted future utility.

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

where $d_{ijt} = 1$ if consumer i chooses store j in period t and $d_{ijt} = 0$ otherwise, and β is the discount factor.

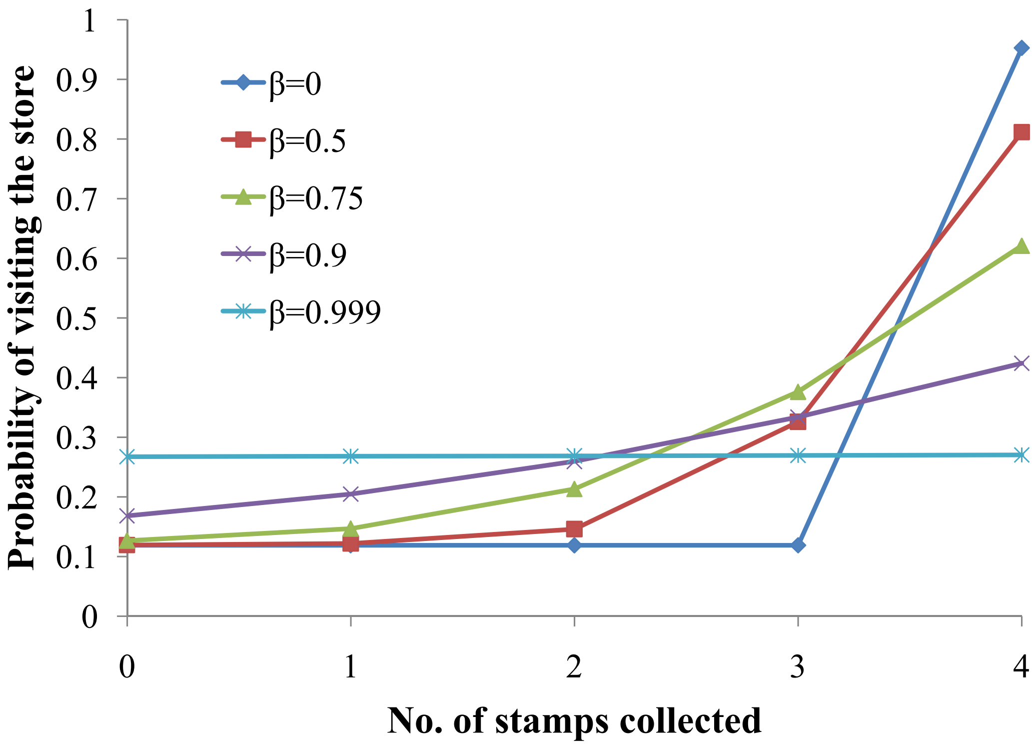
- The evolution of the state, s_{it}, is as follows.

$$s_{ijt+1} = \begin{cases} s_{ijt} + 1 & \text{if } d_{ijt} = 1 \text{ and } s_{ijt} < \bar{S}_j - 1 \\ 0 & \text{if } d_{ijt} = 1 \text{ and } s_{ijt} = \bar{S}_j - 1 \\ s_{ijt} & \text{if } d_{ijt} = 0. \end{cases}$$

Implications of the Model

- The main dynamics of the model is the intertemporal trade-off created by the reward program.
- Suppose that a consumer is close to completion of the stamp card for store 1, and store 2 offers a lower price today.
- If the consumer chooses store 2 today, he/she will get a better deal today but will delay the completion of the stamp card for store 1.
- If β is less than one, the delay will lower the present discounted value of the gift.
- The incentive not to delay the completion will increase as he/she gets closer to the completion of the stamp card.
- Hartmann and Viard (2008) use a similar model to study the switching costs created by reward programs.

Choice Probabilities



- Only one store and the outside option.
- Other parameters are set as $\bar{S}_1 = 5$, $\alpha_1 = -2.0$, $G_1 = 3.0$, $\sigma_{G_1} = 0$, and $\gamma = -1.0$.

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

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

$$
\begin{aligned}
V_j(s_i; p_{ij}, G_i, \theta) &= \begin{cases} \alpha_j - \gamma p_{ij} + G_{ij} + \beta E_{p'} V(s'; p', G_i, \theta) & \text{if } s_{ij} = \bar{S}_j - 1, \\ \alpha_j - \gamma p_{ij} + \beta E_{p'} V(s'; p', G_i, \theta) & \text{otherwise,} \end{cases} \\
V_0(s_i; G_i, \theta) &= \beta E_{p'} V(s'; p', G_i, \theta)
\end{aligned}
$$

The expectation with respect to p' is defined by

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

Numerical Solution of the Model

- We use contraction mapping to solve for V. Start with an initial guess, V^0. Then use Bellman operator to update V^n recursively until it converges (Refer this to standard DP step).

$$\bar{E}_p[V^{n-1}(s';p',G_i,\theta)] = \frac{1}{M}\sum_{m=1}^{M} V^{n-1}(s';p^m,G_i,\theta)$$

$$V^n(s_i;p_i,G_i,\theta) = E\max_j\{U_{ij}(s_i;p_i,\theta) + \beta\bar{E}_{p'}[V^{n-1}(s';p',G_i,\theta)]\},$$

where $\{p^m\}_{m=1}^M$ are drawn from the price distribution.

Key features of IJC method

- MCMC algorithm that bases on a proxy of the value function.
- Consider a model without unobserved consumer heterogeneity.
- The outputs of the algorithm in iteration r include $\{\theta^r, E_p\tilde{V}^r(s,p;\theta^r)\}$. Store $H^r = \{\theta^l, E_pV^l(s,p;\theta^l)\}_{l=r-N}^{r-1}$.
- Use Metropolis-Hastings within Gibbs to obtain θ^r.
- Given a candidate draw θ^{*r}, use $\{E_p\tilde{V}^l(s,p;\theta^l)\}_{l=r-N}^{r-1}$ to obtain an estimate of $\hat{E}_{p'}[V(s',p';\theta^{*r})]$.
- Make M draws of prices, $p^m = (p_1^m, p_2^m)$, from the price distribution. Obtain $\tilde{V}^r(s,p^m;\theta^{*r})$ by applying the Bellman operator only once based on $\hat{E}_{p'}[V(s',p';\theta^{r*})]$. Then,

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

IJC's DP step

IJC propose to replace the standard DP step with the following step:

- Let $H^r = \{\theta^{*l}, E_p\tilde{V}^l(., p; \theta^{*l})\}_{l=r-N}^{r-1}$ be the history of simulated draws up to the previous iteration $r-1$.
- For each s,

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

  where $K_h()$ is a kernel with bandwidth $h > 0$.
- Simulate $\{p^m\}_{m=1}^{M}$, and compute the pseudo-value function

$$\tilde{V}^r(s, p^m; \theta^{*r}) = \max_j\{U_j(s, p^m; \theta^{*r}) + \beta\hat{E}_{p'}[V(s', p'; \theta^{*r})]\},$$

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

The Pseudo-Value Function

- The pseudo-value function is defined as follows.

$$
\begin{aligned}
&\tilde{V}^r(s, p^m; \theta^{*r}) \\
&\quad \equiv \quad E_\epsilon \max\{\tilde{V}_0(s; \theta^{*r}) + \epsilon_0, \tilde{V}_1(s; p_1^m, \theta^{*r}) + \epsilon_1, \tilde{V}_2(s; p_2^m, \theta^{*r}) + \epsilon_2\} \\
&\quad = \quad \log(\exp(\tilde{V}_0^r(s; \theta^{*r})) + \exp(\tilde{V}_1^r(s, p_1^m; \theta^{*r})) + \exp(\tilde{V}_2^r(s, p_2^m; \theta^{*r})))
\end{aligned}
$$

where

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

Remark 1

When S is discrete, and N increases at an appropriate rate as l increases, IJC show that:

- Theorem 1: The sequence of approximated value function V^l converges to V in probability uniformly over (S, ϵ, Θ).
- Theorem 2: θ^l converges to the true posterior distribution in total variation norm.
- Note that in practice, we need to fix N.

Remark 2

- The approximation step of the Emax functions can also be applied in the classical methods (Brown and Flinn 2006).
- However, applying it in the Bayesian estimation has at least two advantages:
  - Past value functions are evaluated at parameter vectors, which are randomly distributed around the current parameter value – good for non-parametric approximation. This is typically not the case in the classical methods.
  - Accurately simulating a posterior distribution is usually easier than finding the global maximum/minimum of a complex likelihood/GMM objective function.

How to choose N and h

- The rate of convergence is usually faster for smaller N, because the set of past pseudo-value functions used for approximation are more recent and accurate.
- However, smaller N also leads to the weighted average of the pseudo-value functions more sensitive to the change in the set. As a result, we may get larger posterior standard deviation.
- One trick we can use is to set N to be small at the beginning and increase it as the number of iterations increases.

How to choose N and h (cont'd)

- A general strategy to choose N and h (bandwidth): check how good the approximation for the value function is during the estimation.
    - For instance, in every 1,000 iterations, compute the means of the MCMC draws, $\bar{\tilde{\theta}}$, and compare the distance between the pseudo-value function and the exact value function at $\bar{\tilde{\theta}}$.
    - If memory is not a constraint, store a large N past pseudo-value functions and use the most recent $N' < N$ to do the Emax approximation.
    - One can increase N' as soon as the researcher discovers that the approximation is not good enough.
    - Then set a new h based on standard optimal bandwidth formula, e.g., Silverman's rule of thumb.

Estimation 1: Impact of N

- We simulate samples based on the solution of the dynamic model. We then estimate the parameters of the model by using different N's.
- True parameters of the simulation
  - Store intercepts: $\alpha_1 = \alpha_2 = 0$.
  - Population mean of rewards: $G_1 = 1.0$, $G_2 = 10.0$.
  - Population variance of rewards: $\sigma_{G_1} = \sigma_{G_2} = 0$ (no heterogeneity).
  - Number of stamps for rewards: $\bar{S}_1 = 5$, $\bar{S}_2 = 10$.
  - Price coefficient: $\gamma = -1.0$.
  - Discount factor: $\beta = 0.98$.
- We use gaussian kernel with bandwidth=0.02, and estimate α_1, α_2, G_1, G_2, γ.

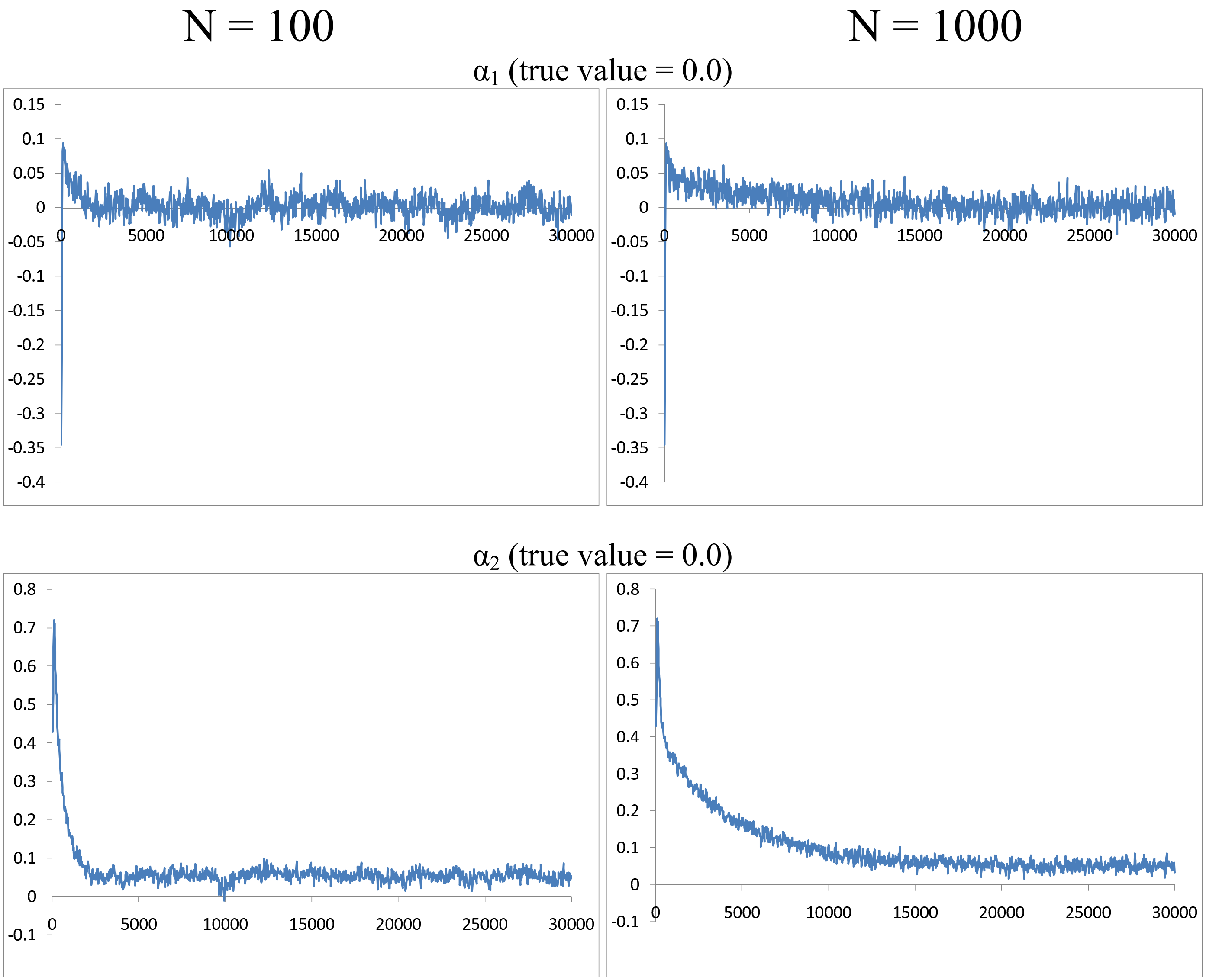
Estimation 1: Impact of N (cont'd)

Parameter estimates

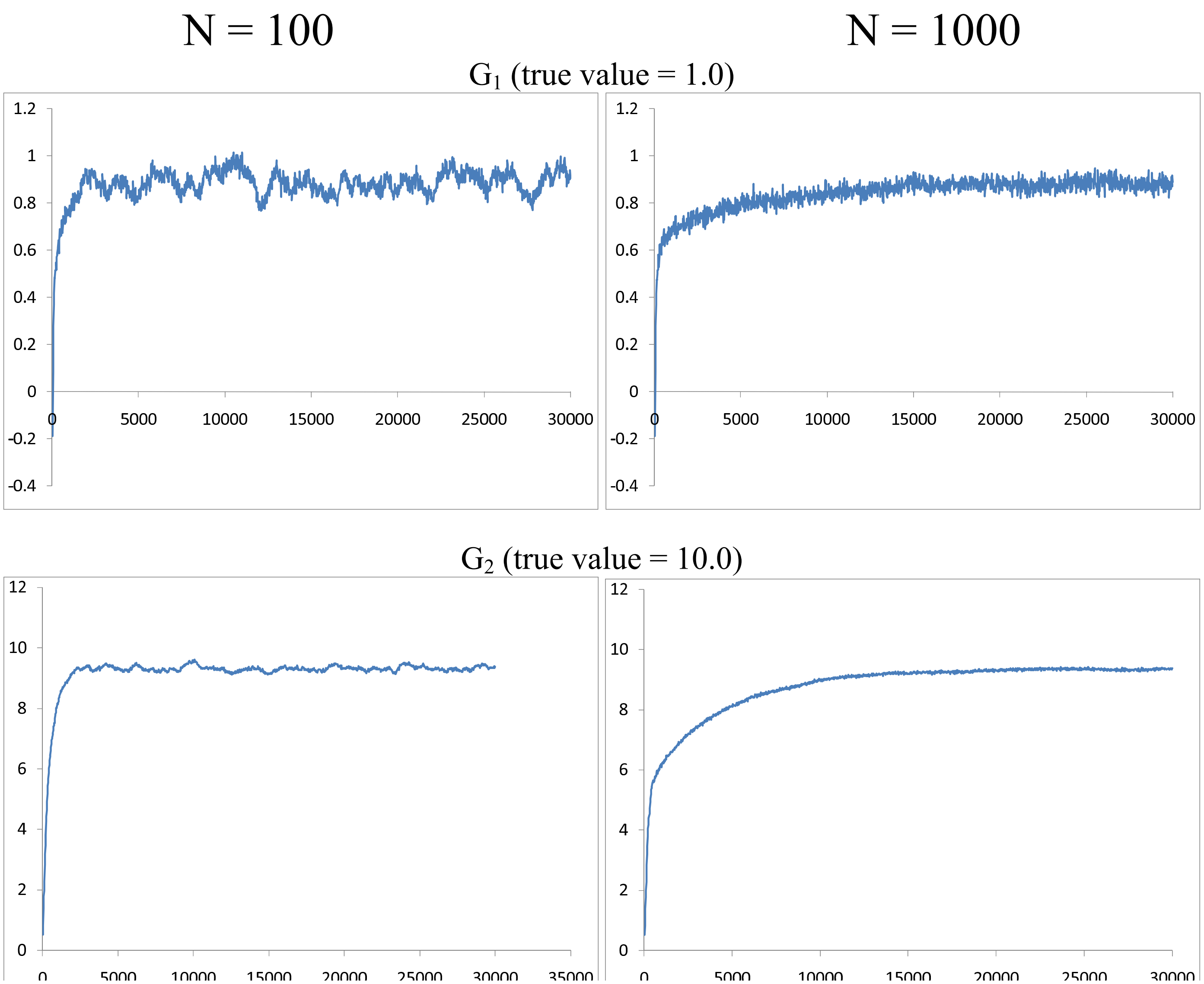
		N=100		N=1000	
parameter	TRUE	mean	sd	mean	sd
α_1 (intercept for store 1)	0.0	-0.049	0.020	-0.061	0.020
α_2 (intercept for store 2)	0.0	0.032	0.019	0.022	0.019
G_1 (reward for store 1)	1.0	1.234	0.034	1.246	0.021
G_2 (reward for store 2)	10.0	9.740	0.063	9.751	0.028
γ (price coefficient)	-1.0	-1.000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

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

Estimation 1: Impact of N (cont'd)



Estimation 1: Impact of N (cont'd)



Estimation 2

- We simulate samples based on the solution of the dynamic model with and without unobserved consumer heterogeneity.
- True parameters of the simulation
  - Store intercepts: $\alpha_1 = \alpha_2 = 0$.
  - Population mean of rewards: $G_1 = 1.0$, $G_2 = 5.0$.
  - Population variance of rewards: $\sigma_{G_1} = 0$, and $\sigma_{G_2} = 0$ (no heterogeneity) or 1.0 (heterogeneity).
  - Number of stamps for rewards: $\bar{S}_1 = 2$, $\bar{S}_2 = 4$.
  - Price coefficient: $\gamma = -1.0$.
  - Discount factor: $\beta = 0.6$ or 0.8.

Results: Homogeneous Model

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

Sample size: 100,000 (1,000 consumers for 100 periods)
Fixed parameters: $\bar{S}_1 = 2$, $\bar{S}_2 = 4$, $\bar{p} = 1.0$, $\sigma_p = 0.3$, $\sigma_{G_j} = 0$ for $j = 1, 2$
Tuning parameters: $N = 1,000$ (no. of past pseudo-value functions used for Emax function approximation), $h = 0.01$ (bandwidth).

Results: Heterogeneous Model

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

Sample size: 100,000 (1,000 consumers for 100 periods)
Fixed parameters: $\bar{S}_1 = 2$, $\bar{S}_2 = 4$, $\bar{p} = 1.0$, $\sigma_p = 0.3$, $\sigma_{G_1} = 0$
Tuning parameters: $N = 1,000$ (no. of past pseudo-value functions used for Emax function approximation), $h = 0.01$ (bandwidth).

Comparison of Computation Time

Average seconds per iteration

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

- For the IJC algorithm, every time a candidate parameter value is rejected, we re-computed the likelihood function at previously accepted parameter value, which will be used to evaluate the likelihood ratio in the Metropolis-Hastings step.
- For the full solution based Bayesian algorithm, we set the number of draws for the integration of prices to be 100.
- Note that the average time per iteration for the full-solution based MCMC increases exponentially with β when estimating a model with unobserved heterogeneity, while it remains unchanged for IJC.
- For $\beta = 0.98$ and no. of iterations = 10,000, full-solution based MCMC would run for 70 days, while IJC would run for 2.5 days.

Policy Experiments

- Policy experiments may seem hard to do from the Bayesian perspective.
- Suppose we are interested in the effect of changing θ_i to $\acute{\theta}_i = (1+t)\theta_i$.
- Modify the IJC algorithm as follows:
  - In each iteration r, set the draw of the policy parameter vector as $\acute{\theta}^r_{-i} = \theta^r_{-i}$ and $\acute{\theta}^r_i = (1+t)\theta^r_i$.
  - Need to store $\{\acute{\theta}^{*l}, E_p\tilde{V}^l(.,p;\acute{\theta}^{*l})\}_{l=r-N}^{r-1}$ in addition to what we store before.
  - After convergence, use $\{\theta^r\}_{r=L-D-1}^{L}$ to set the simulated policy parameters: $\acute{\theta}^r_i = (1+t)\theta^r_i$. Then use $\{\acute{\theta}^{*l}, E_p\tilde{V}^l(.,p;\acute{\theta}^{*l})\}_{l=r-N}^{r-1}$ to construct the corresponding pseudo-value functions and choice probabilities for the simulated policy experiment outcomes.
  - Extra computation time per iteration for the heterogeneous model: 0.9s for $N = 500$, and 1.74s for $N = 1000$.

Continuous State Space

- What if we allow prices to be serially correlated? (e.g., $f(p'|p;\theta_p)$)
- Modify the IJC algorithm above and combine it with the random grid approximation by Rust (1997).
  - In each iteration r, make a draw of prices, $p^r = (p_1^r, p_2^r)$, from $U[\underline{p}, \bar{p}]^2$, and compute the pseudo-value function at p^r.
  - Outputs of the algorithm become (in the homogeneous model)

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

  - The expected value function at $(s, p; \theta^r)$ is then approximated as

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

Final Remarks

- Curse of Dimensionality of parameter space – in this case, the problem can be mitigated because we can control N.
- If the size of data set is large, and the model is well-identified, MCMC draws would concentrate in a small neighborhood of the posterior means – reduce the need to use a large N.
- There are many kernels that one could use. Norets (2008) extends IJC by using "nearest" neighbors, and consider serially correlated error terms.
- One can also weigh the past pseudo-value functions depending how recent they are.

Conclusion

- This paper uses an example of dynamic store choice to illustrate how to implement the IJC algorithm.
- IJC is particularly beneficial in estimating a model with unobserved heterogeneity.
- Osborne (2008) applies the IJC method to estimate a consumer learning model.
- Ching et al. (2009) estimate a dynamic model of learning and forgetting.
- Our C programs which implement the IJC method are available upon request.
- The paper is available at SSRN: http://ssrn.com/abstract=1398444