Bayesian Estimation of Dynamic Discrete Choice Models*

Susumu Imai
Pennsylvania State University

Neelam Jain
Northern Illinois University

and

Andrew Ching
Ohio State University

April 6, 2003

*Very preliminary. Comments welcome. Please direct all correspondence to Susumu Imai, Concordia University, Department of Economics, 1455 de Maisonneuve Blvd. W., Montreal, Quebec H3G 1M8, Canada, Email: simai@alcor.concordia.ca; or Neelam Jain, e-mail: jain@rice.edu.
1 Introduction

Structural estimation of Dynamic Discrete Choice (DDC) models has become increasingly popular in both empirical economics and marketing. Examples include Keane and Wolpin (1997), Erdem and Keane (1995). Recently, it has also been applied in analyzing criminal behavior, as in Imai and Krishna (2001). Structural estimation is appealing for at least two reasons. First, it captures the dynamic forward-looking behavior of individuals, which is very important in understanding agents’ behaviors in various settings. For example, in labor market, individuals carefully consider future prospects when they switch occupations. Secondly, since the estimation is based on explicit solution of a structural model, it avoids the Lucas Critique. Hence, after the estimation, policy experiments can be relatively straightforwardly conducted by simply changing the estimated value of “policy” parameters and simulating the model to assess the change. However, one major obstacle in adopting the structural estimation method has been its computational burden. There are mainly two reasons why this estimation method is computationally demanding. First, in order to solve a dynamic model, we need to compute the Bellman equation repeatedly until the calculated expected value function (Emax function) converges. This Emax function is used to form the likelihood. That is, given a parameter value, Bellman equation has to be computed many times until convergence. Secondly, in solving the Dynamic Programming Problem, the Bellman equation has to be solved at each possible point in the state space. The possible number of points in the state space increases exponentially with the increase in the dimensionality of the state space. This is commonly referred to as the “Curse of Dimensionality”, and makes the estimation of the dynamic model even in relatively simple setting infeasible. Examples of the model where the Curse of Dimensionality is severe include the dynamic oligopoly models, where the dimensionality of the state space increases with the number of firms.

In this paper, we propose an estimator that helps overcome the two computational difficulties of structural estimation. Our estimation algorithm combines a stochastic approximation algorithm introduced by Pakes and McGuire (2001, henceforth PM) and a Bayesian Markov Chain Monte Carlo (MCMC) algorithm. The PM algorithm is a Markov Chain simulation based algorithm, and is not subject to the “Curse of Dimensionality”. A Bayesian MCMC algorithm is also a Markov Chain based simulation algorithm. Our estimation algorithm (we call it the Bayesian Dynamic Programming (DP) Algorithm) takes advantage of their similarities by merging these two Markov Chains into one. The key innovation in our algorithm is that we need to solve the Bellman equation only once and only at a single point in the state space between each estimation step. Since a single Bellman equation is as computationally demanding as computing a static model, the computational burden of estimating a DP model is in order of magnitude comparable to that of estimating a static model. Furthermore, since we move the parameters following the MCMC algorithm after each stochastic Bellman step, we are “estimating” the model and solving for the DP problem at the same time. This is in contrast to conventional estimation methods that “estimate” the model only after solving the DP problem. In that sense, our estimation method is related to the algorithm advocated by Aguirreagabiria and Mira (2001), where they propose to solve the DP problems “roughly” at the initial stage of the Maximum Likelihood routine and increase the precision of the DP solution with the iteration of the Maximum Likelihood routine. Notice that
they still compute the solution of the DP problem, whether exact or inexact, during each estimation step. In our algorithm, we only need to solve the Bellman equation once between each estimation step.

Specifically, we start with some initial continuation value or the emax function and draw the current state vector, including the parameter vector, from the initial distribution. We then solve the optimization problem at this state vector, that is, we solve the optimal policies and calculate the value function. We then use Bayesian MCMC to update the parameter vector and draw the remaining state variables from the distribution determined by the optimal policies. We update the emax function for the current state by averaging with those past iterations in which the parameter vector is ‘close’ and the other state variables are exactly the same. As the iterations grow, the approximation becomes more precise at states that are visited more frequently. We conjecture that our estimation method overcomes the “Curse of Dimensionality” due to the fact that we do not require calculating optimal policies and the emax function at every possible point in the state space. Since we solve the model and estimate it at the same time, our state space includes the parameter vector as well. Thus, the model is solved accurately at those points in the state space that are recurrent. As PM argue, the set of recurrent states need not grow at all in the number of state variables and thus, our estimation algorithm may overcome the curse of dimensionality. Further, although the degree of precision is low in early iterations, the precision grows as iterations do since we make use of the past iterations. We conjecture that the larger the state space associated with the underlying economic problem, the greater is the advantage in using our Bayesian DP estimation algorithm since the precision of our estimates does not necessarily depend on the dimension of the state space.

Application of Bayesian methods to estimate the structural DDC models has been particularly difficult. The main reason is that the solution of the DP problem, i.e. the repeated computation of the Bellman equation is computationally so demanding that the MCMC, which typically involves far more iterations than the standard Maximum Likelihood routine, becomes infeasible. One of the few examples of Bayesian estimation is Lancaster (1997). He successfully estimates the equilibrium search model where the Bellman equation can be transformed into an equation where all the information on optimal choice of the individual can be summarized in the reservation wage, and hence, there is no need for solving the value function. Another example is Geweke and Keane (1995) who estimate the DDC model without solving the DP problem. In contrast, our paper accomplishes Bayesian estimation based on full solution of the general DP problem. As discussed earlier, we do so by exploiting the Markov properties of the MCMC algorithm as well as of the PM stochastic algorithm. Our estimation method not only makes Bayesian application to DDC models computationally feasible, but even possibly superior to some conventional (non-Bayesian) methods, by reducing the computational burden of estimating a dynamic model to that of estimating a static

---

1 Rust (1997) provides another way to overcome the curse of dimensionality, namely the random grid method. We too use random grids, however, our random grid changes, actually increases, over the iterations. Indeed, we conjecture that this property of our algorithm allows us to overcome the curse of dimensionality in settings where a fixed random grid based method may not, e.g. in cases of deterministic transitions.

2 Although we use the PM solution algorithm (which is stochastic) in this paper, our estimation algorithm can be used with other solution methods as well, such as the random grid method proposed in Rust (1997). The key aspect of our Bayesian DP estimation algorithm is to solve the model and estimate it at the same time.
one. Furthermore, the usually cited advantages of Bayesian estimation over the classical estimation methods apply here as well. That is, 1) the conditions for the convergence of the MCMC algorithm are in general weaker than the conditions for the global maximum of the Maximum Likelihood (ML) estimator; 2) The Bayesian estimation is an exact likelihood method, in contrast to ML methods that rely on large sample asymptotics and 3) In MCMC, standard errors can be derived straightforwardly as a byproduct of the estimation routine, whereas in ML estimation, standard errors have to be computed usually in the following two ways. One is by inverting the numerically calculated Information Matrix, which is only valid in large sample world. The other is by repeatedly bootstrapping and reestimating the model, which is computationally demanding.

In Section 2, we present a simple model of entry and exit that we later use for estimation. In Section 3, we explain our Bayesian DP algorithm. In Section 4, we present estimation results of several experiments applied to the model of entry and exit. Finally, in Section 5, we conclude and briefly discuss future direction of this research.

2 Model

We estimate a simple dynamic discrete choice model of entry and exit, with firms in competitive environment.³ The firm is either an incumbent (I) or a potential entrant (O). If the incumbent firm chooses to stay, its per period return is,

\[ R_{I,IN}(K_t, \epsilon_t, \theta) = \alpha K_t + \epsilon_{1t}, \]

where \( K_t \) is the capital of the firm, \( \epsilon_t = (\epsilon_{1t}, \epsilon_{2t}) \) is a vector of random shocks, and \( \theta \) is the vector of parameter values. If it chooses to exit, its per period return is,

\[ R_{I,OUT}(K_t, \epsilon_t, \theta) = \delta_x + \epsilon_{2t} \]

where \( \delta_x \) is the exit value to the firm. Similarly, if the potential entrant chooses to enter, its per period return is,

\[ R_{O,IN}(K_t, \epsilon_t, \theta) = -\delta_E + \epsilon_{1t} \]

and if it decides to stay out, its per period return is,

\[ R_{O,OUT}(K_t, \epsilon_t, \theta) = \epsilon_{2t}. \]

We assume the random component of the current period returns to be distributed i.i.d normal as follows.

\[ \epsilon_{1t} \sim N(0, \sigma_l) \]

The level of capital \( K_t \) evolves as follows.

³For an estimation exercise based on the model, see Roberts and Tybout (1997).
\[
\ln K_{t+1} = b_1 + b_2 \ln K_t + u_{t+1},
\]
where
\[
u_t \sim N(0, \sigma_u),
\]
if the incumbent firm decides to stay in, and

\[
k_{t+1} = k_0,
\]
if the potential entrant decides to enter.

Now, consider a firm who is an incumbent at the beginning of period \(t\). The Bellman equations for the optimal choice of the incumbent is,

\[
V_I(K_t, \epsilon_t, \theta) = \text{Max}\{V_{I,IN}(K_t, \epsilon_t, \theta), V_{I,OUT}(K_t, \epsilon_t, \theta)\}.
\]

where,

\[
V_{I,IN}(K_t, \epsilon_t, \theta) = R_{I,IN}(K_t, \epsilon_{1t}, \theta) + \beta E_{t+1}V_I(K_{t+1}(K_t, u_{t+1}), \epsilon_{t+1}, \theta)
\]
is the value of staying in during period \(t\). Similarly,

\[
V_{I,OUT}(K_t, \epsilon_t, \theta) = R_{I,OUT}(K_t, \epsilon_{1t}, \theta) + \beta E_{t+1}V_O(K_{t+1}(0, \epsilon_{t+1}, \theta)
\]
is the value of exiting during period \(t\) and \(V_O\) is given by the Bellman equation for the optimal choice of the potential entrant:

\[
V_O(K_t, \epsilon_t, \theta) = \text{Max}\{V_{O,IN}(K_t, \epsilon_t, \theta), V_{O,OUT}(K_t, \epsilon_t, \theta)\}.
\]

where,

\[
V_{O,IN}(K_t, \epsilon_t, \theta) = R_{O,IN}(K_t, \epsilon_{1t}, \theta) + \beta E_{t+1}V_I(K_0, \epsilon_{t+1}, \theta),
\]
is the value of entering during period \(t\) and

\[
V_{O,OUT}(K_t, \epsilon_t, \theta) = R_{O,OUT}(K_t, \epsilon_{1t}, \theta) + \beta E_{t+1}V_O(0, \epsilon_{t+1}, \theta)
\]
is the value of staying out during period \(t\).

The parameter vector \(\theta\) of the model is \((\delta_x, \delta_E, \alpha, \beta, \sigma_1, \sigma_2, \sigma_u, b_1, b_2)\). The state variables are the capital stock \(K\), the parameter vector \(\theta\) and the status of the firm, \(i_{t,t} \in \{IN, OUT\}\), that is, whether the firm is an incumbent or a potential entrant.
3 Bayesian DP Algorithm

We denote any variables with \(d\) superscript to be the data. We assume that for each firm, we only observe the capital stock, profit and the entry/exit status over \(T\) periods. That is, we know

\[
\{K_{i,t}^d, \pi_{i,t}^d, i_{t,t}^d\}_{t=1}^{T}
\]

where,

\[
\pi_{i,t}^d = \alpha K_{i,t}^d + \varepsilon_{1t},
\]

if the firm stays in and 0 otherwise.

We assume the prior of the exit value and entry cost to be normally distributed as follows.

\[
\delta_x \sim N(\delta_x, A_x^{-1})
\]

\[
\delta_E \sim N(\delta_E, A_E^{-1})
\]

where \(\delta_x, \delta_E\) are the prior means and \(A_x, A_E\) are the prior precision (inverse of variance) of the exit value and the entry cost, respectively.

We also assume an independent Chi squares prior for the precision of the shocks \(\varepsilon_1\) and \(u\) which is the inverse of their variance, i.e. \(h_{\varepsilon_1} = (\sigma_{\varepsilon_1}^2)^{-1}\), for \(\varepsilon_1\). That is,

\[
s_{\varepsilon_1}^2 h_{\varepsilon_1} \sim \chi^2(\nu_{\varepsilon_1}),
\]

where \(s_{\varepsilon_1}^2\) is a parameter. Similarly,

\[
s_u^2 h_u \sim \chi^2(\nu_u).
\]

Furthermore,

\[
s_{\eta}^2 h_{\eta} \sim \chi^2(\nu_{\eta}),
\]

where \(\eta = \varepsilon_1 - \varepsilon_2\).

Assume that we already know the expected value function (the emax function) of the \(s\) th iteration.

\[
E_\epsilon^t V_\Omega(K, \epsilon, \theta^s),
\]

where \(\Omega = \{I, O\}\), and the expectation is taken with respect to the next period shock \(\epsilon\). That is, the emax function is a function of future capital level \(K\) and the parameters. Also, we denote \(\theta^s\) be the parameter vector at the \(s\) th iteration and \(K_s\) to be the capital stock for iteration \(s\). The following steps are taken to calculate the Emax function of \(s + 1\) th iteration.

**Bellman Equation Step**

In this step, we derive \(V_\Omega(K_s, \epsilon_s, \theta_s)\) and \(K_{s+1}\).
1) We use the random grid integration method of Rust (1997) to deal with the infinite state space for capital stock. That is, we make $M$ i.i.d. draws of capital stock grids $K_m$, $m = 1, \ldots, M$ from a given distribution. Then, calculate,

$$E^s V_{\Omega}(K_{s+1}(K_s, u_{s+1}), \epsilon_s, \theta^s) = \sum_{m=1}^{M} E^s E^s V_{\Omega}(K_m, \epsilon, \theta^s) f(K_m|K_s, \theta^s).$$

2) Draw $\epsilon_s = (\epsilon_{1s}, \epsilon_{2s})$.

3) Given $\epsilon_s$ and $E^s V_{\Omega}(K, \epsilon, \theta_s)$, solve for the Bellman equation, that is, solve the decision of the incumbent (whether to stay or exit) or of the entrant (whether to enter or stay out) and derive the value function corresponding to the optimal decisions:

$$V_{\Omega}(K_s, \epsilon_s, \theta_s) = \max \{R_{\Omega, IN}(K_s, \epsilon_{1s}, \theta^s) + \beta E V_{I}(K_{s+1}(K_s, u_{s+1}), \epsilon, \theta^s),
R_{\Omega, OUT}(K_s, \epsilon_{2s}, \theta^s) + \beta E V_{O}(0, \epsilon, \theta^s)\}.$$

From this optimization problem, we know the status of the firm in the next iteration, that is, whether it is an incumbent or a potential entrant. Also, given the optimal decisions and $K_s$, derive the next period’s capital, $K_{s+1}$ according to the law of evolution of the capital stock specified earlier.

Gibbs Sampling and Data Augmentation Step

Here, we describe how the new parameter vector $\theta^{s+1}$ is drawn. Let the deterministic values for the incumbent be defined as follows:

$$\nabla_{I, IN}(K_t, \theta^s) = \alpha^s K_t + \beta E V_{I}(K_{t+1}, \epsilon, \theta^s),$$

and

$$\nabla_{I, OUT}(K_t, \theta^s) = \delta^s + \beta E V_{O}(0, \epsilon, \theta^s).$$

Similarly, for the potential entrant, we define

$$\nabla_{O, IN}(K_t, \theta^s) = -\delta^s + \beta E V_{I}(K_{0}, \epsilon, \theta^s),$$

and

$$\nabla_{O, OUT}(K_t, \theta^s) = \beta E V_{O}(0, \epsilon, \theta^s).$$

Then, at iteration $s$, we go through the following two steps.
1) Data Augmentation Step on Entry and Exit choice:

Define current revenue difference net of $\alpha^s K_t$ by

$$w_{i,t}^{s+1} = R_{\Omega,OUT}(K_{i,t}^d, \epsilon_2, \theta_s) - R_{\Omega,IN}(K_{i,t}^d, \epsilon_1, \theta_s) + \alpha^s K_{i,t}^d \equiv g(K_{i,t}^d, \epsilon_1, \epsilon_2, \theta_s).$$

The empirical economist does not observe the above statistics directly because he can only obtain data on entry and exit decisions $i_{i,t}$ and profits, not the current revenues themselves. Nonetheless, the empirical economist can indirectly recover $w_{i,t}^{s+1}$ by simulating and augmenting the shock $\eta_{i,t} = \epsilon_1, \epsilon_2$. But the simulation of $\eta_{i,t}$ has to be consistent with the actual choices that the firm makes. That is, if, in the data, the firm $i$ at period $t$ either stays in or enters, that is, $i_{i,t} = IN$, then draw $\eta_{i,t} = \epsilon_1, \epsilon_2$ such that

$$\eta_{i,t} \geq V_{\Omega,OUT}(K_{i,t}^d, \theta_s) - V_{\Omega,IN}(K_{i,t}^d, \theta_s).$$

If, in the data, the firm $i$ either stays out or exits, that is, $i_{i,t} = OUT$, then draw $\eta_{i,t}$ such that

$$\eta_{i,t} < V_{\Omega,OUT}(K_{i,t}^d, \theta_s) - V_{\Omega,IN}(K_{i,t}^d, \theta_s).$$

Once the shock $\eta_{i,t}$ is generated, he can proceed to recover the entry cost and exit value parameters by conducting Bayesian regression of $w_{i,t}^{s+1}$ on entry and exit decisions, using the following linear relationship.

$$w_{i,t}^{s+1} = \delta_E I(\Omega_{i,t} = O) + \delta_x I(\Omega_{i,t} = I) + \eta_{i,t}.$$ 

Data Augmentation Step on Profit: If the firm stays out or exits, then its potential profit is not observable. In that case, we simulate the profit as follows:

$$\pi_{i,t} = \alpha^s K_t + \epsilon_{1, i, t}.$$ 

We draw $\epsilon_{1, i, t}$ conditional on $\eta_{i,t}$ as follows:

$$\epsilon_{1, i, t} = \gamma_1 \eta_{i,t} + v_{i,t},$$

where

$$v_{i,t} \overset{\sim}{\sim} N(0, \sigma^2_v).$$
\[
\sigma_v^2 = \sigma_1^2 - \frac{\sigma_4^2}{\sigma_1^2 + \sigma_2^2} = \frac{\sigma_1 \sigma_2}{\sigma_1^2 + \sigma_2^2}
\]

and
\[
\gamma_1 = \frac{\text{Cov}(\epsilon_{1t}, \eta_t)}{\sigma_\eta^2} = \frac{\sigma_1 \sigma_2}{\sigma_1^2 + \sigma_2^2}.
\]

Once the profit for firms who exited or stayed out is recovered, we can recover productivity parameters via a simple Bayesian regression.

2) Draw the new parameter vector \( \theta_{s+1} \) from the posterior distribution.

We denote the matrix \( I \) with \( i(T-1)+t \) th row as follows:
\[
I_{i(T-1)+t} = [I_{i,t}^d(IN), I_{i,t}^d(OUT)].
\]

where \( I_{i,t}^d(IN) = 1 \) if the firm either enters or decides to stay in, and \( I_{i,t}^d(OUT) = 1 \) if the firm either exits or stays out.

We draw \( a_{s+1} = [\delta_{x}^{s+1}, \delta_{E}^{s+1}]' \) conditional on \( (w_{s+1}, h^s_\eta) \) as follows.
\[
a_{s+1}|(w_{s+1}, h^s_\eta) \sim N(\overline{a}, \overline{A}_a),
\]

where,
\[
\overline{A}_a = (A_a + h^s_\eta I'I)^{-1}
\]

and
\[
\overline{a} = \overline{A}_a^{-1}(A_a \overline{a} + h^s_\eta I w_{s+1}).
\]

We draw the posterior distribution of \( h_\eta \) from the following \( \chi^2 \) distribution. That is,
\[
[\hat{\eta}^2 + \sum_{i,t} \hat{\eta}_{i,t}^2] h^s_{\eta} |(w_{s+1}, a_{s+1}) \sim \chi^2(NT + \nu),
\]

where \( \hat{\eta} \) is the “residual”, that is,
\[
\hat{\eta}_{i,t} = w_{s+1,i,t} - \hat{\delta}_E I_{i,t}^d(OUT) - \hat{\delta}_x I_{i,t}^d(IN).
\]

The above Gibbs sampling data augmentation steps are an application of McCulloch and Rossi (1994).
Next, we draw $\alpha^{s+1}$ conditional on $(\pi^{s+1}, h^s)$. Denote

$$k_t = \log(K_t), \quad k_{-1} = [k_{11}, k_{12}, ..., k_{1T-1}, ..., k_{N1}, k_{N2}, ..., k_{NT-1}]$$

and

$$k = [k_{12}, k_{13}, ..., k_{1T}, ..., k_{N2}, k_{N3}, ..., k_{NT}].$$

Then,

$$\alpha^{s+1}|(\pi^{s+1}, h^s) \sim N(\varpi, \mathbf{A}_\alpha),$$

where,

$$\mathbf{A}_\alpha = (A_\alpha + h^s_\alpha k'k)^{-1}$$

and

$$\varpi = \mathbf{A}_\alpha^{-1}(A_\alpha + h^s_\alpha k'\pi).$$

We draw the posterior distribution of $h^s$ from the following $\chi^2$ distribution. That is,

$$[s^{2}_{\varepsilon_1} + \sum_{i,t} \tilde{\varepsilon}_{i,t}^2 |h^s_\varepsilon + 1](\pi^{s+1}, \alpha^{s+1})^{-1}\sim \chi^2(NT + \nu),$$

where $\hat{\varepsilon}_1$ is the “residual”, that is,

$$\hat{\varepsilon}_{i,t} = \pi^{s+1}_{i,t} - \alpha^{s+1}_{i,t} k_{i,t}.$$

Furthermore, $(\sigma^{s+1}_{\varepsilon_2})^2$ or $h^{s+1}_{\varepsilon} = (\sigma^{s+1}_{\varepsilon_2})^{-2}$ can be recovered as follows:

$$(\sigma^{s+1}_{\varepsilon_2})^2 = (h^s_{\eta} + 1)^{-1} - (h^s_{\varepsilon_1} + 1)^{-1}$$

Next, we draw $b^{s+1} = [b^{s+1}_1, b^{s+1}_2]'$ conditional on $(k, h^s)$ as follows.

$$b^{s+1}|(\pi^{s+1}, h^s) \sim N(\bar{b}, \mathbf{A}_b),$$

where,

$$\mathbf{A}_b = (A_b + h^s_b k'_{-1}k_{-1})^{-1}$$

and

$$\bar{b} = \mathbf{A}_b^{-1}(A_b b + h^s_b k'_{-1}k).$$
We draw the posterior distribution of $h_u$ from the following $\chi^2$ distribution. That is,

$$\frac{\bar{u}_{i,t}^2}{s_u^2 + \sum_{i,t} \bar{u}_{i,t}^2} h_u^{s+1} | (u^{s+1}, a^{s+1})^T \chi^2(NT + \nu)$$

where $\bar{u}_{i,t}$ is the “residual”, that is,

$$\bar{u}_{i,t} = k^d_{i,t} - b^s_1 - b^s_2 k^d_{i,t-1}.$$

**Expected Value Function Iteration Step**

Next, we update the expected value function for iteration $s+1$, that is, we derive $E^{s+1}_\Omega(K, \epsilon, \theta_s)$. This is an important step in the algorithm and is closely related to the algorithm of Pakes and McGuire (2001).

$$E^{s+1}_\Omega(K, \epsilon, \theta_s) = \frac{\sum_{j=\text{Max}(s-J,1)}^s V\Omega(K_j, \epsilon_j, \theta_j) I(K_j = K_s) K_h(\theta_j - \theta_s)}{\sum_{j=\text{Max}(s-J,1)}^s I(K_j = K_s) K_h(\theta_j - \theta_s)},$$

where $I()$ is the indicator function, and $K()$ is the kernel function. We adopt the following Gaussian kernel:

$$K_h(\theta_j - \theta_s) = (2\pi)^{-L/2} \prod_{l=1}^L \exp[-\frac{1}{2} \frac{(\theta_{l,j} - \theta_{l,s})^2}{h_l^2}].$$

Thus, the emax function is updated by taking the average over those past $J$ iterations where the capital $K_j$ was the same as $K_s$ and the parameter vector $\theta_j$ was close to $\theta_s$. The emax function remains the same for the other capital grid points. The similarity with PM is that the expected value functions are approximated by averaging over past values of the algorithm, that is, the continuation values are never explicitly calculated. Also, the optimization problem is solved only once between iterations. The main difference is that past values are weighted according to the distance between their parameter vectors and the current parameter vector: the shorter is the distance, the higher is the weight. Pakes and McGuire (2001) show that the above Bellman equation step and the expected value function iteration step, with the parameter values fixed, overcome the Curse of dimensionality. That is based on the insight that even though the size of the entire state space may be large, and increases exponentially with respect to the dimensions of the state space, the important subset of the state space, which agents revisit frequently, may be small, and only increases linearly with the dimensionality of the state space. Since we adopt their algorithm, our estimation routine is designed to overcome the Curse of Dimensionality. The choice of $J$ could be model specific. The simulation example below indicates that in a simple model, $J = 3000$ is sufficient for a good estimation performance.

After the above Bellman equation step, data augmentation step and the expected value function iteration step, we now have the capital stock $K_{s+1}$, parameter vector $\theta_{s+1}$ and the expected value function $E^{s+1}_\Omega(K_s, \epsilon, \theta_s)$ for $s + 1$ th iteration. We repeat these steps to derive iteration $s + 2$ in the same way as described above for $s + 1$ th iteration.
4 Simulation and Estimation Exercise.

Denote the true values of \( \theta \) by \( \theta^* \). Thus \( \theta^* = (\delta^*_E, \delta^*_x, \sigma^*_1, \sigma^*_2, \sigma^*_u, \alpha^*, b^*_1, b^*_2, \beta^*) \). We set the following parameters for the above model.

\[
\delta^*_E = 0.4, \delta^*_x = 0.4, \sigma^*_1 = 0.4, \sigma^*_2 = 0.4, \sigma^*_u = 0.4, \alpha^* = 0.2, b^*_1 = 0.1, b^*_2 = 0.1, \beta^* = 0.9.
\]

We first solve the DP problem numerically using conventional numerical methods. Next, we generate artificial data based on the above DP solution. Then, using the simulated data, we try to estimate the parameter values using the Bayesian DP estimation method. Below, we briefly explain how we solved for the DP problem to generate the data. Notice that for data generation, we only need to solve for the DP problem once, that is, for a fixed set of parameters. Hence, we took time and made sure that the DP solution is accurate.

Assume that we already know the expected value function of the \( s \) th iteration for all capital grid points.

\[
E_{\epsilon}V(K_m, \epsilon, \theta^*), \quad m = 1, 2, ..., M.
\]

Then, following steps are taken to generate the expected value function for \( s + 1 \) th iteration.

Step 1 Given capital stock \( K \), derive

\[
E^{s}V_{\Omega}(K_{s+1}(K, u_{s+1}), \epsilon_s, \theta^*) = \sum_{m=1}^{M} E_{\epsilon}V_{\Omega}^s(K_m, \epsilon_s, \theta^*) f(K_m|K, \theta^*)
\]

for \( \Omega \in \{I, O\} \). Here, \( K_m (m = 1, ..., M) \) are grid points and \( f(K_m|K_s, \theta^s) \) is the transition probability from \( K \) to \( K_m \).

Step 2 Draw the random shocks \( \epsilon_l \). For a given capital stock \( K \), calculate

\[
V_{\Omega}(K, \epsilon_l, \theta^*) = Max\{R_{\Omega,IN}(K, \epsilon_l, \theta^s) + \beta E^{s}V_{I}(K_{s+1}, \epsilon, \theta^*), \beta E^{s}V_{O}(0, \epsilon, \theta^*) \}
\]

Step 3 Repeat Step 2 \( L \) times and take an average to derive the expected value function for the next iteration.

\[
E^{s+1}_{\epsilon}V_{\Omega}(K, \epsilon, \theta^*) = \frac{1}{L} \sum_{l=1}^{L} V_{\Omega}(K, \epsilon_l, \theta^*).
\]

The above steps are taken for all possible capital grid points, \( K = K_1, ..., K_M \). In our simulation exercise, we set the simulation size \( L \) to be 1000. The total number of capital grid points is set to be 50.
Step 4 Repeat Step 1 to Step 3 until the Emax function converges. That is, for a small $\delta$ (in our case, $\delta = 0.001$),

$$Max_{m=1,\ldots,M}\{E_{\epsilon}^{s+1} V_{\Omega}(K_m, \epsilon, \theta^*) - E_{\epsilon}^{s} V_{\Omega}(K_m, \epsilon, \theta^*) \} < \delta.$$  

We simulate artificial data of capital stock and investment choice $\{K_t, i_t\}$ using the expected value functions derived above. We then estimate the model using the simulated data with our Bayesian DP routine. We do not estimate the discount factor $\beta$. Instead, we set it at the true value $\beta^* = 0.9$.

4.1 Experiment 1: Basic Model

We first describe the prior distributions of parameters. The priors are set to be very diffuse, in order to keep the influence on the outcome of the estimation exercise to a minimum. The exceptions are the priors for the precision parameters, where we impose somewhat tight prior.

$$\delta_x \sim N(\delta_x, A_x^{-1}), \delta_x = 0.4, A_x = 1.0,$$

$$\delta_E \sim N(\delta_E, A_E^{-1}), \delta_E = 0.4, A_E = 1.0$$

$$\alpha \sim N(\alpha, A_{\alpha}^{-1}), \alpha = 0.2, A_{\alpha} = 1.0$$

$$b_1 \sim N(b_1, A_{b_1}^{-1}), b_1 = 0.1, A_{b_1} = 1.0$$

$$b_2 \sim N(b_2, A_{b_2}^{-1}), b_2 = 0.1, A_{b_2} = 1.0$$

$$s^{2}_{\epsilon_1} h_{\epsilon_1} \sim \chi^2(\nu_{\epsilon_1}), (s^{2}_{\epsilon_1})^{-1} = 0.4, v_{\epsilon_1} = 1000.$$  

$$s^{2}_{\eta} h_{\eta} \sim \chi^2(\nu_{\eta}), (s^{2}_{\eta})^{-1} = \sqrt{0.32}, v_{\eta} = 1000.$$  

$$s^{2}_{\mu} h_{\mu} \sim \chi^2(\nu_{\mu}), s^{2}_{\mu} = 1.0, v_{\mu} = 1000.$$  

We set the initial guess of the expected value function to be 0. We set the initial guesses of the parameters to be the true parameter values given by $\theta^*$. The Gibbs sampling was conducted 30,000 times. The Gibbs sampler for the simulation with sample size 10,000 is shown in figures 1.
to 7. In estimation experiments with other sample sizes, the Gibbs sampler converged from around 20,000 iterations as well. The posterior mean and standard errors from the 25,001 th iteration up to 30,000 th iteration are shown in Table 1. The posterior mean of $\delta_x$ and $\delta_E$ are estimated to be below the true values if the sample size is 1000, 4000 and 6000, and above the true values for the sample size 2000, and 10,000. In general, we can see that as the sample size increases, the estimated values becomes closer to the truth (with the exception of sample size 2000, when the estimates for $\delta_x$ and $\delta_E$ are fairly close to the true values.).

Figures 1 and 2 show the Gibbs sampler output of parameters $\delta_x$ and $\delta_E$. Even though the initial guess is set to be the true value, at the start of the Gibbs sampling algorithm, both parameters immediately jump to values very close to zero. Notice that these values are the estimates we should expect to get when we estimate the data generated by a dynamic model using a static model. Because the emax functions are set to zero initially, the future benefit of being in or out is zero. Hence, if either exit value or entry cost were big in value, then either entry or exit choice would dominate most of the time, and thus the model would not predict both choices to be observed in the data.

In Figure 9, we plot the value function at different iterations. As we iterate our Bayesian Dynamic Programming routine, the value function becomes closer to the solution of the Dynamic Programming problem, hence the future values become closer to the true values. Notice that at the same time the estimates of the parameters representing current benefits of the choices, such as $\delta_x$ and $\delta_E$ converge closer to the true value (see Figures 1 and 2). In this sense, we claim that indeed our algorithm solves the Dynamic Programming problem and estimates the parameters simulataneously, and not subsequently.

Figure 3 plots the Gibbs sampler output of the capital coefficient in the profit equation, $\alpha$. We can see that there, the value of the first Gibbs sampler jumps from the true value, 0.2 to 0.2344. The upward bias is due to the inadequate sample selection bias correction. However, immediately after a couple of iterations, the Gibbs sampler estimates the true value quite accurately. That is, our Gibbs sampler algorithm can immediately correct for the sample selection bias even though it has not fully solved the Dynamic programming problem. The Gibbs sampler of the other parameters are reported in Figures 4 to 8. There, we see that all the parameters stay closely around the true value from the start. These results lead us to conjecture that the estimation algorithm is not sensitive to the initial values.
Table 1: Posterior Means and Standard Errors
(standard errors are in parenthesis)

<table>
<thead>
<tr>
<th>parameter</th>
<th>estimate</th>
<th>estimate</th>
<th>estimate</th>
<th>true value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta_x$</td>
<td>0.3224 (0.0318)</td>
<td>0.4282 (0.0236)</td>
<td>0.3464 (0.0177)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\delta_E$</td>
<td>0.3165 (0.0436)</td>
<td>0.4103 (0.0301)</td>
<td>0.3262 (0.0215)</td>
<td>0.4</td>
</tr>
<tr>
<td>$a$</td>
<td>0.2084 (0.0048)</td>
<td>0.2065 (0.0041)</td>
<td>0.2049 (0.0031)</td>
<td>0.2</td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>0.3951 (0.0074)</td>
<td>0.3995 (0.0066)</td>
<td>0.4025 (0.0057)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\sigma_2$</td>
<td>0.4065 (0.0192)</td>
<td>0.4050 (0.0187)</td>
<td>0.4031 (0.0195)</td>
<td>0.4</td>
</tr>
<tr>
<td>$b_1$</td>
<td>0.0865 (0.0437)</td>
<td>0.0933 (0.0321)</td>
<td>0.1046 (0.0220)</td>
<td>0.1</td>
</tr>
<tr>
<td>$b_2$</td>
<td>0.0820 (0.0543)</td>
<td>0.0899 (0.0379)</td>
<td>0.0899 (0.0261)</td>
<td>0.1</td>
</tr>
<tr>
<td>$\sigma_u$</td>
<td>0.4005 (0.0076)</td>
<td>0.3977 (0.0064)</td>
<td>0.3968 (0.0051)</td>
<td>0.4</td>
</tr>
<tr>
<td>sample size</td>
<td>1000</td>
<td>2000</td>
<td>4000</td>
<td></td>
</tr>
<tr>
<td>CPU time$^4$</td>
<td>1 hr.32 min.3 sec</td>
<td>1 hrs.58 min.23 sec</td>
<td>3 hrs.31 min.48 sec</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>parameter</th>
<th>estimate</th>
<th>estimate</th>
<th>true value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta_x$</td>
<td>0.3553 (0.0153)</td>
<td>0.4062 (0.0133)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\delta_E$</td>
<td>0.3457 (0.0183)</td>
<td>0.4036 (0.0140)</td>
<td>0.4</td>
</tr>
<tr>
<td>$a$</td>
<td>0.2053 (0.0026)</td>
<td>0.2077 (0.0023)</td>
<td>0.2</td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>0.4021 (0.0049)</td>
<td>0.4029 (0.0042)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\sigma_2$</td>
<td>0.4028 (0.0188)</td>
<td>0.4005 (0.0190)</td>
<td>0.4</td>
</tr>
<tr>
<td>$b_1$</td>
<td>0.1086 (0.0181)</td>
<td>0.1048 (0.0136)</td>
<td>0.1</td>
</tr>
<tr>
<td>$b_2$</td>
<td>0.0905 (0.0214)</td>
<td>0.0921 (0.0170)</td>
<td>0.1</td>
</tr>
<tr>
<td>$\sigma_u$</td>
<td>0.3979 (0.0043)</td>
<td>0.4004 (0.0035)</td>
<td>0.4</td>
</tr>
<tr>
<td>sample size</td>
<td>6,000</td>
<td>10,000</td>
<td></td>
</tr>
<tr>
<td>CPU time$^5$</td>
<td>4 hrs.42 min.44 sec</td>
<td>7 hrs.1 min.51 sec</td>
<td></td>
</tr>
</tbody>
</table>

4.2 Experiment 2: Different Initial Values

We set the initial guess of the expected value function to be 0. Except for $\sigma_2$, which in our estimation exercise is set to be a numeraire at the true value, we set the initial guesses of all the other parameters to be half the true value. We set the sample size to be 10,000 and the Gibbs sampling was conducted 30,000 times. Table 2 describes the posterior mean and standard errors from the 25,001 th iteration up to 30,000 th iteration. Figures 10 to 17 show the Gibbs sampler for the parameters. We can see that the posteriors and the Gibbs sampler are almost the same as those of Experiment 1. The parameters either immediately converge around the true values, or, as is the case with the entry cost and exit value estimates, immediately jump to the static model estimates and thereafter slowly converge to the true values. From these two simulation and estimation exercises, we observe that at least in this example, the performance of

---

$^4$The estimation exercise was done on a Unix Ultra 10 machine.

$^5$The estimation exercise was done on a Unix Ultra 10 machine.
the Dynamic Programming/Gibbs Sampling algorithm is robust to changes in the initial guesses of the parameters.

Table 2: Posterior Means and Standard Errors
(standard errors are in parenthesis)

<table>
<thead>
<tr>
<th>parameter</th>
<th>estimate</th>
<th>true value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta_x$</td>
<td>0.4044 (0.0135)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\delta_E$</td>
<td>0.4023 (0.0139)</td>
<td>0.4</td>
</tr>
<tr>
<td>$a$</td>
<td>0.2078 (0.0024)</td>
<td>0.2</td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>0.4030 (0.0043)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\sigma_2$</td>
<td>0.4020 (0.0191)</td>
<td>0.4</td>
</tr>
<tr>
<td>$b_1$</td>
<td>0.1054 (0.0138)</td>
<td>0.1</td>
</tr>
<tr>
<td>$b_2$</td>
<td>0.0915 (0.0169)</td>
<td>0.1</td>
</tr>
<tr>
<td>$\sigma_u$</td>
<td>0.4004 (0.0035)</td>
<td>0.4</td>
</tr>
<tr>
<td>sample size</td>
<td>10,000</td>
<td></td>
</tr>
<tr>
<td>CPU time$^6$</td>
<td>7 hrs 1 min 14 sec</td>
<td></td>
</tr>
</tbody>
</table>

4.3 Experiment 3: Random Effects

We now report estimation results of a model that includes unobserved heterogeneity. That is, we add to the above entry and exit model a random effect on the profit equation. We assume that the profit coefficient for each firm $i$, $a_i$, is distributed normally with mean $a = 2.0$ and standard error $\sigma_a = 0.04$. Notice that if we use the conventional simulated ML method to estimate the model, for each firm $i$ we need to draw $a_i$ many times, say $M$ times, and for each draw solve the dynamic programming problem. Hence, for a single simulated likelihood evaluation, we need to solve the DP problem $NM$ times, where $N$ is the sample size. This process is computationally so demanding that most researchers so far have only used finite number of types, typically less than 10, as an approximation of the random effect. Since in our Bayesian DP estimation exercise, the computational burden of estimating the dynamic model is roughly equivalent to that of a static model, we can easily accommodate random effects estimation as will be shown below.

We set the prior for $a_i$ as follows.

$$a_i|\mu \sim N(\mu, \tau^2)$$

$$\mu \sim N(\mu_0, h_0^{-1})$$

$$s^{-2} - \chi^2(\nu)$$

Then, if we denote $a' = (a_1, \ldots, a_N)$, $\pi' = (\pi_{11}, \pi_{12}, \ldots, \pi_{1T}, \ldots, \pi_{NT})$ and

$^6$The estimation exercise was done on a Unix Ultra 10 machine.
\[ W = \begin{bmatrix}
  K_1 & 0 & \cdots & 0 \\
  0 & K_2 & \ddots & \vdots \\
  \vdots & \ddots & \ddots & 0 \\
  0 & \cdots & 0 & K_N
\end{bmatrix} \]

where \( K_j = [K_{j1}, K_{j2}, \ldots, K_{jT}] \). Also, \( e_N \) is a \( N \) by 1 vector of ones.

Then, the prior can be expressed as follows.

\[ a \sim N \left( e_N \mu, \tau^2 I_N + h^{-1} e_N e_N' \right) \]

Let \( \theta_{-a}^s \) be defined as parameters not including \( a_i \). Below, we briefly describe the differences between the earlier estimation routine and that which involves random effects.

**Data Augmentation Step on Entry and Exit choice**: For data augmentation, we need to generate

\[ w_{s+1}^{i,t} = R_{\Omega, OUT}(K_{i,t}^d, \epsilon_2, \theta_{-a}^s, a_i^s) - R_{\Omega, IN}(K_{i,t}^d, \epsilon_1, \theta_{-a}^s, a_i^s) + a_i^s K_{i,t}^d \equiv g(K_{i,t}^d, \epsilon_{1,i,t} - \epsilon_{2,i,t}, \theta_{-a}^s, a_i^s). \]

- To draw \( \eta_{i,t} = \epsilon_{1,i,t} - \epsilon_{2,i,t} \) we follow the data augmentation steps below.

\[ \eta_{i,t} \geq V_{\Omega, OUT}(K_{i,t}^d, \theta_{-a}^s, a_i^s) - V_{\Omega, IN}(K_{i,t}^d, \theta_{-a}^s, a_i^s). \]

If, in the data, the firm \( i \) either stays out or exits, that is, \( i_{i,t}^d = OUT \), then draw \( \eta_{i,t} \) such that

\[ \eta_{i,t} < V_{\Omega, OUT}(K_{i,t}^d, \theta_{-a}^s, a_i^s) - V_{\Omega, IN}(K_{i,t}^d, \theta_{-a}^s, a_i^s). \]

As we discussed earlier, once the shock \( \eta_{i,t} \) is generated, he can proceed to recover the entry cost and exit value parameters by conducting Bayesian regression of \( w_{s+1}^{i,t} \) on entry and exit decisions, using the following linear relationship.

\[ w_{s+1}^{i,t} = \delta_E I(\Omega_{i,t} = O) + \delta_x I(\Omega_{i,t} = I) + \eta_{i,t}. \]

In contrast to the earlier case, to evaluate the entry and exit values, we use different \( a_i \) for each firm \( i \).

**Data Augmentation Step on Profit**: If the firm stays out or exits, then its potential profit is not observable. In that case, we simulate the profit:

\[ \pi_{i,t} = a_i^g K_t + \epsilon_{1,i,t}. \]

The only difference from the standard case is that the capital coefficient \( a_i \) is different for each firm \( i \). We skip discussing the rest of the step because it is the same as before.

**Draw the new parameter vector** \( \theta^{s+1} \) **from the posterior distribution** The only difference in the estimation procedure is for drawing the posterior of \( a \), and \( \tau^2 \).
Then, the posterior draw of $a$ for iteration $s$, $a^{s+1}$, can be done from the following distribution.

$$a^{s+1} \mid (\pi^s, K) \sim N \left( \pi, H^{-1}_a \right)$$

with

$$H_a = (\sigma^s_1)^{-2}W'W + \left( \tau^2I_N + h^{-1}_a e_N e'_N \right)^{-1}$$

$$\pi = H_a^{-1} \left[ \left( \tau^2I_N + h^{-1}_a e_N e'_N \right)^{-1} e_N a + (\sigma^s_1)^{-2}W'K \right]$$

Also, the posterior draw of $\tau^{s+1}$ can be done from the below distribution.

$$\left[ s^2 + \sum_i \bar{v}_i^2 \right] (\tau^{s+1})^2 (v^{s+1}, a^{s+1}) \sim \chi^2(N + \nu)$$

where

$$v_i = a_i^{s+1} - \pi$$

**Expected Value Function Iteration Step**

The only difference between the earlier case is that we compute the expected value function for each firm $i$ separately. That is, the expected value function, $E^{s+1}V(K_{s+1}, \epsilon, \theta^{s+1}_s, a_i)$ is derived as follows.

$$E^{s+1}V_\Omega(K_s, \epsilon, \theta^{s+1}_s, a_i) = \frac{\sum_{j=\text{Max}(s-J,1)}^s V_\Omega(K_j, \epsilon_j, \theta_j) I(K_j = K_s)K_h(\theta^j_{-a} - \theta^{s}_s)K_h(a^j - a^s)}{\sum_{j=\text{Max}(s-J,1)}^s I(K_j = K_s)K_h(\theta^j_{-a} - \theta^{s}_s)K_h(a^j - a^s)}$$

This is the part where we have an increase in computational burden, because the above calculation has to be done for each firm. But it turns out that the additional burden is far lighter than those of computing the DP problem again for each firm $i$, for each simulation $m$.

We set the sample size to be 100 firms for 100 periods, and the Gibbs sampling was conducted 15,000 times. Table 3 describes the posterior mean and standard errors from the 14,001 th iteration up to 15,000 th iteration.

**Table 3: Posterior Means and Standard Errors**

(standard errors are in parenthesis)
Notice that most of the parameters are close to the true values. The exceptions are the entry and exit coefficient estimates, that shows upward bias, and the standard error of the random effects parameter, which also is biased upwards. Those biases could be due to the fact that we have not dealt with the initial conditions problem. The computation time is 24 hours, which roughly corresponds to those required for a Bayesian estimation of a reasonably complicated static random effects model.

### Table

<table>
<thead>
<tr>
<th>parameter</th>
<th>estimate</th>
<th>true value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\delta_x$</td>
<td>0.4910 (0.0103)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\delta_E$</td>
<td>0.5001 (0.0128)</td>
<td>0.4</td>
</tr>
<tr>
<td>$a$</td>
<td>0.2142 (0.0044)</td>
<td>0.2</td>
</tr>
<tr>
<td>$\sigma_a$</td>
<td>0.0470 (0.0069)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\sigma_1$</td>
<td>0.4019 (0.0042)</td>
<td>0.4</td>
</tr>
<tr>
<td>$\sigma_2$</td>
<td>0.4004 (0.0079)</td>
<td>0.4</td>
</tr>
<tr>
<td>$b_1$</td>
<td>0.1012 (0.0160)</td>
<td>0.1</td>
</tr>
<tr>
<td>$b_2$</td>
<td>0.1018 (0.0118)</td>
<td>0.1</td>
</tr>
<tr>
<td>$\sigma_u$</td>
<td>0.2004 (0.0028)</td>
<td>0.2</td>
</tr>
<tr>
<td>sample size</td>
<td>$100 \times 100$</td>
<td></td>
</tr>
<tr>
<td>CPU time$^7$</td>
<td>24 hrs 28 min 9 sec</td>
<td></td>
</tr>
</tbody>
</table>

5 Conclusion

In conventional estimation methods of Dynamic Discrete Choice models, such as GMM, Maximum Likelihood or Markov Chain Monte Carlo, at each iteration step, given a new set of parameter values, the researcher first solves the Bellman equation to derive the expected value function, and then uses it to construct the likelihood. That is, during the DP iteration, which typically takes a lot of computational time, the researcher fixes the parameter values and does not "estimate". We propose a Bayesian estimation algorithm where the DP problem is solved and parameters estimated at the same time. In other words, we move parameters during the DP solution. This dramatically increases the speed of estimation. We have demonstrated the effectiveness of our approach by estimating a simple dynamic model of discrete entry-exit choice. Even though we are estimating a dynamic model, the required computational time is in line with the time required for Bayesian estimation of static models. The reason for the speed is clear. The computational burden of estimating dynamic models has been high because the researcher has to repeatedly evaluate the Bellman equation during a single estimation routine, where he keeps the parameter values the same. We move parameters, i.e. "estimate" the model after each Bellman equation evaluation. Since a single Bellman equation evaluation is computationally no different from computing a static model, the speed of our estimation exercise, too, is no different from that of a static model.

There is, however, a caveat to the above statement. Another computational obstacle in the estimation of a Dynamic Discrete Choice model is the Curse of Dimensionality. That is, the computational burden increases exponentially with the increase in the dimension of the state space.

---

$^7$The estimation exercise was done on a Unix Ultra 10 machine.
In our example, we only estimated a simple model of entry-exit behavior of firms, where the state space is only unidimensional. Hence, so far we did not demonstrate the effectiveness of our estimation method in overcoming the Curse of Dimensionality. However, we conjecture that our estimation method could offer a potential solution to this problem, because it is well known from the examples of Pakes and McGuire (2001) that the stochastic Dynamic Programming Algorithm overcomes the Curse of Dimensionality. That is based on the insight that even though the size of the entire state space may be large, and increases exponentially with respect to the dimensions of the state space, the important subset of the state space, which agents revisit frequently, may be small, and only increases linearly with the dimensionality of the state space. Since we adopt the same stochastic algorithm as a component of our estimation routine, we conjecture that our method overcomes the Curse of Dimensionality. We have been experimenting with a simple Oligopoly model that involves multiple firms. Preliminary results are encouraging, suggesting that the Curse of Dimensionality is overcome as well, even though more work needs to be done to finalize the estimation.

Furthermore, since we are locally approximating the expected value function nonparametrically, as we increase the number of parameters, we may face the “Curse of Dimensionality” in terms of the number of parameters to be estimated. But here, we again are optimistic, since the simulation exercises in the above examples show that with a reasonably large sample size, the posterior distribution has small variance, hence the inaccuracy due to nonparametric approximation is not severe.
6 References


Figure 1: Gibbs Sampler Output of Exit Value (True Value: 0.4)
Figure 2: Gibbs Sampler Output of Entry Cost (True Value: 0.4)
Figure 3: Gibbs Sampler Output of Capital Coefficient (True Value: 0.2)
Figure 4: Gibbs Sampler Output of the Profit Shock (True Value: 0.4)
Figure 5: Gibbs Sampler Output of the Entry and Exit Shock (True Value: 0.4)
Figure 6: Gibbs Sampler Output of the Capital Stock Transition Parameter $b_1$ (True Value: 0.1)
Figure 7: Gibbs Sampler Output of Capital Stock Transition Parameter b2 (True Value: 0.1)
Figure 8: Gibbs Sampler Output of the Standard Error of the Capital Stock Transition Shock

$\sigma_{\text{mu}}$

![Plot of $\sigma_{\text{mu}}$ vs. iteration](image)
Figure 9: Value Functions at Different Iterations.
Figure 10: Gibbs Sampler Output of Exit Value (True Value: 0.4)
Figure 12: Gibbs Sampler Output of Capital Coefficient (True Value: 0.2)
Figure 13: Gibbs Sampler Output of the Profit Shock (True Value: 0.4)
Figure 14: Gibbs Sampler Output of the Entry and Exit Shock (True Value: 0.4)
Figure 15: Gibbs Sampler Output of the Capital Stock Transition Parameter $b_1$ (True Value: 0.1)
Figure 16: Gibbs Sampler Output of Capital Stock Transition Parameter b2 (True Value: 0.1)
Figure 17: Gibbs Sampler Output of the Standard Error of the Capital Stock Transition Shock

\( \text{sigma}_u \)