Estimating Dynamic Discrete-Choice Games of Incomplete Information*

Michael Egesdal† Zhenyu Lai‡ Che-Lin Su§

October 24, 2012

Abstract

We investigate the estimation of models of dynamic discrete-choice games of incomplete information, formulating the maximum-likelihood estimation exercise as a constrained optimization problem which can be solved using state-of-the-art constrained optimization solvers. Under the assumption that only one equilibrium is played in the data, our approach avoids repeatedly solving the dynamic game or finding all equilibria for each candidate vector of the structural parameters. We conduct Monte Carlo experiments to investigate the numerical performance and finite-sample properties of the constrained optimization approach for computing the maximum-likelihood estimator, the two-step pseudo maximum-likelihood estimator and the nested pseudo-likelihood estimator, implemented by both the nested pseudo-likelihood algorithm and a modified nested pseudo-likelihood algorithm.

Keywords: dynamic discrete-choice games of incomplete information, maximum-likelihood estimator, constrained optimization, nested pseudo-likelihood estimator

*We have benefited greatly from discussions with Timothy B. Armstrong, Steven T. Berry, Xiaohong Chen, Jean-Pierre Dubé, Philip A. Haile, Kenneth L. Judd, Harry J. Paarsch, Ariel Pakes, and seminar participants at Yale University. All errors are our own.

†Department of Economics, Harvard University. Email: egesdal@fas.harvard.edu.

‡Department of Economics, Harvard University. Email: zlai@fas.harvard.edu.

§The University of Chicago Booth School of Business. Email: Che-Lin.Su@ChicagoBooth.edu. This author is grateful for financial support from the Neubauer Family Faculty Fellowship.

1 Introduction

Empirical models of dynamic games of incomplete information are an important framework within which to study firms' strategic behavior. In the past decade, developing econometric methods to estimate these models has become an active research topic in the empirical industrial organization and applied econometrics literatures. As models of dynamic games become increasingly sophisticated, estimating the underlying structural parameters and decision policies adopted by firms becomes increasingly challenging computationally. The high computational costs of solving dynamic games during the estimation stage has motivated researchers to propose econometric methods that provide consistent estimates in large-sample theory, and that are computationally light and easy to implement in practice. Most of these computationally simple methods belong to the class of two-step estimators. For example, see Bajari, Benkard, and Levin (2007); Pakes, Ostrovsky, and Berry (2007); Pesendorfer and Schmidt-Dengler (2008) as well as Arcidiacono and Miller (2011). The potential drawbacks of two-step estimators are that their estimates can have large biases in finite samples because insufficient data exist to obtain precise estimates in the first step, and that researchers might not use an appropriate criterion function in the second step; see the discussion in Pakes, Ostrovsky, and Berry (2007). To address these issues, Aguirregabiria and Mira (2007) have proposed the nested pseudo-likelihood (NPL) estimator and the NPL algorithm to compute the NPL estimator. Using Monte Carlo experiments, Aguirregabiria and Mira demonstrated that the NPL estimator is less biased than the two-step pseudo maximum-likelihood (2S-PML) estimator.

Pakes, Ostrovsky, and Berry (2007); Pesendorfer and Schmidt-Dengler (2008); Pesendorfer and Schmidt-Dengler (2010); and Su (2012) have shown that the NPL algorithm can frequently fail to converge. Even worse, the NPL algorithm may not provide consistent estimates. Kasahara and Shimotsu (2012) analyzed the convergence properties of the NPL algorithm and suggested modifications in implementing the NPL algorithm to improve its convergence. Using a simplified version of a dynamic game model derived from Aguirregabiria and Mira (2007), they illustrated that their modified NPL (NPL-Λ) algorithm indeed converged and performed well in a Monte Carlo experiment, while the original NPL algorithm failed.

Su and Judd (2012) have proposed a constrained optimization approach to estimating structural models, while Dubé, Fox, and Su (2012) applied the constrained optimization approach to the estimation of random-coefficients logit demand models, successfully solving examples having tens of thousands of variables and constraints. Su (2012) illustrated that the constrained optimization approach can be applied to estimating static games of incomplete information with multiple equilibria (under the assumption that only one equilibrium is played in each market in the data) and that it performed better than the NPL estimator in Monte Carlo experiments. Even so, some researchers remain unsure whether the constrained optimization approach is practical to estimate dynamic games because it requires solving high-dimensional optimization problems, which can be computationally demanding.

Following Su and Judd (2012) as well as Su (2012), we have formulated the maximum-likelihood (ML) estimation problem of dynamic discrete-choice games of incomplete information as a constrained optimization problem. Using the dynamic game model of Aguirregabiria and Mira (2007), we have conducted Monte Carlo experiments to investigate the finite-sample properties and the numerical performance of the 2S-PML estimator, the NPL estimator implemented by the NPL and

NPL-Λ algorithms, and the ML estimator implemented by the constrained optimization approach. Our Monte Carlo results suggest that the constrained optimization approach is more robust and reliable than both the NPL and the NPL-Λ algorithms. Indeed, the constrained approach converged for all data sets in all experiments, while the performance of the NPL and NPL-Λ algorithms varied. In some cases, the NPL and NPL-Λ algorithms failed to converge. The constrained approach is also faster than either the NPL or the NPL-Λ algorithm when the size of the state space in the model increases, but the state transition matrix remains sparse. Although the 2S-PML estimator always converged in our experiments, this estimator is much less accurate than the ML estimator under the constrained optimization approach. Overall, when compared to alternative estimators, using the constrained optimization approach to ML estimation offers valuable returns: reliable convergence, computational speed, and accurate estimates.

We have organized the remainder of the paper as follows: In Section 2, we describe a model of dynamic games proposed by Aguirregabiria and Mira (2007), while in Section 3, we present the constrained optimization formulation for the ML estimation of these games and discuss alternative likelihood-based estimators and their associated estimation algorithms. In Section 4, we describe the design of our Monte Carlo experiments and present numerical results. We conclude in Section 5 with a summary as well as a brief description of potential future work.

2 Model

We consider a model of discrete-time, infinite-horizon dynamic games based on the research of Aguirregabiria and Mira (2007). In each period $t = 1, 2, ..., \infty$, N players exist, each indexed by $i \in \mathcal{I} = \{1, ..., N\}$; the players operate in a market characterized by size $s^t \in \mathcal{S} = \{s_1, \ldots, s_L\}$. We assume that market size is observed by all players and evolves according to the exogenous stationary transition probability $f_{\mathcal{S}}(s^{t+1}|s^t)$, where $s^t, s^{t+1} \in \mathcal{S}$.

At the beginning of each period t, player i observes a vector of common-knowledge state variables $\boldsymbol{x}^t$ and private shocks $\boldsymbol{\varepsilon}_i^t$. Players then simultaneously choose whether to be active in the market. Let $a_i^t \in \mathcal{A} = \{0, 1\}$ denote player i's action in period t and $\boldsymbol{a}^t = (a_1^t, \ldots, a_N^t)$ denote the collection of all players' actions. The common-knowledge state variables, $\boldsymbol{x}^t$ consist of market size and all players' actions in the previous period, namely, $\boldsymbol{x}^t = \left(s^t, \boldsymbol{a}^{t-1}\right) \in \mathcal{X} = \{\mathcal{S}, \times_{i \in \mathcal{I}} \mathcal{A}\}$. Each player i also observes privately $\boldsymbol{\varepsilon}_i^t = \left\{\varepsilon_i^t\left(a_i^t\right)\right\}_{a_i^t \in \mathcal{A}}$, a vector of choice-contingent shocks to per-period payoffs. We assume that $\varepsilon_i^t\left(a_i^t\right)$ has a type-I extreme value distribution that is independent as well as identically distributed across actions and players as well as over time, and that opposing players do not observe the realization of $\boldsymbol{\varepsilon}_i^t$, but know only its probability density function $g(\boldsymbol{\varepsilon}_i^t)$.

The state variables $(\boldsymbol{x}^t, \boldsymbol{\varepsilon}_i^t)$ evolve after the decisions $\boldsymbol{a}^t$ have been made, so their evolution is described by the exogenous probability distribution function $p(\boldsymbol{x}^{t+1}, \boldsymbol{\varepsilon}_i^{t+1}|\boldsymbol{x}^t, \boldsymbol{\varepsilon}_i^t, \boldsymbol{a}^t)$. We further impose the *conditional independence* assumption.[1] That is,

$$p\left[\boldsymbol{x}^{t+1} = (s', \boldsymbol{a}'), \boldsymbol{\varepsilon}_i^{t+1}|\boldsymbol{x}^t = (s, \tilde{\boldsymbol{a}}), \boldsymbol{\varepsilon}_i^t, \boldsymbol{a}^t\right] = f_{\mathcal{S}}(s'|s)\mathbf{1}\{\boldsymbol{a}' = \boldsymbol{a}^t\}g(\boldsymbol{\varepsilon}_i^{t+1}), \tag{1}$$

where $\mathbf{1}$ is the indicator function.

[1] This is Assumption 2 in Aguirregabiria and Mira (2007).

Denote by $\boldsymbol{\theta}$ the vector of structural parameters and by $\boldsymbol{a}^t_{-i} = (a^t_1, \ldots, a^t_{i-1}, a^t_{i+1}, \ldots, a^t_N)$ the current actions of all players other than i in period t. We specify player i's per-period payoff function as $\tilde{\Pi}_i\left(a^t_i, \boldsymbol{a}^t_{-i}, \boldsymbol{x}^t, \boldsymbol{\varepsilon}^t_i; \boldsymbol{\theta}\right) = \Pi_i\left(a^t_i, \boldsymbol{a}^t_{-i}, \boldsymbol{x}^t; \boldsymbol{\theta}\right) + \varepsilon^t_i\left(a^t_i\right)$, which is additively separable in a common-knowledge component and a private shock. Here, the common-knowledge component $\Pi_i\left(a^t_i, \boldsymbol{a}^t_{-i}, \boldsymbol{x}^t; \boldsymbol{\theta}\right)$ depends on the current actions of all players $\boldsymbol{a}^t$, publicly-observed state variables $\boldsymbol{x}^t$, and $\boldsymbol{\theta}$. Let $\beta \in (0,1)$ denote the discount factor. Given the current state $(\boldsymbol{x}^t, \boldsymbol{\varepsilon}^t_i)$, player i chooses a sequence of decisions to maximize the following total expected discounted payoff:

$$\max_{\{a^t_i, a^{t+1}_i, a^{t+2}_i, \ldots\}} \mathbb{E}\left[\sum_{\tau=t}^{\infty} \beta^{\tau-t} \tilde{\Pi}_i\left(a^\tau_i, \boldsymbol{a}^\tau_{-i}, \boldsymbol{x}^\tau, \boldsymbol{\varepsilon}^\tau_i; \boldsymbol{\theta}\right) \Big| (\boldsymbol{x}^t, \boldsymbol{\varepsilon}^t_i)\right]$$

where the expectation is taken over the state evolution $p(\boldsymbol{x}^{t+1}, \boldsymbol{\varepsilon}^{t+1}_i | \boldsymbol{x}^t, \boldsymbol{\varepsilon}^t_i, \boldsymbol{a}^t)$ given in equation (1) and beliefs about how other players choose their actions.

Since state transition is stationary, we adopt Markov perfect equilibrium as the equilibrium concept. Thus, we can drop the time index t. It is also convenient to characterize the equilibrium in terms of the observed state $\boldsymbol{x}$. Let $P_i(a_i|\boldsymbol{x})$ be the conditional choice probability of player i choosing action $a_i \in \mathcal{A}$ at state $\boldsymbol{x}$. Given $P_j(a_j|\boldsymbol{x}), \forall j \neq i$, the expected payoff of the common-knowledge component $\Pi_i\left(a_i, \boldsymbol{a}_{-i}, \boldsymbol{x}; \boldsymbol{\theta}\right)$ for player i from choosing action a_i at state $\boldsymbol{x}$ is

$$\pi_i\left(a_i|\boldsymbol{x}, \boldsymbol{\theta}\right) = \sum_{\boldsymbol{a}_{-i} \in \mathcal{A}^{N-1}} \left\{ \left[\prod_{a_j \in \boldsymbol{a}_{-i}} P_j\left(a_j|\boldsymbol{x}\right) \right] \Pi_i\left(a_i, \boldsymbol{a}_{-i}, \boldsymbol{x}; \boldsymbol{\theta}\right) \right\}. \tag{2}$$

We denote by $V_i(\boldsymbol{x})$ the expected value function for player i at state $\boldsymbol{x}$ and define $\boldsymbol{P} = \{P_i(a_i|\boldsymbol{x})\}_{i \in \mathcal{I}, \boldsymbol{x} \in \mathcal{X}}$ and $\boldsymbol{V} = \{V_i(\boldsymbol{x})\}_{i \in \mathcal{I}, \boldsymbol{x} \in \mathcal{X}}$. A Markov perfect equilibrium for this game is a tuple $(\boldsymbol{V}, \boldsymbol{P})$ that satisfies the following two systems of nonlinear equations.

I. Bellman Optimality. $\forall i \in \mathcal{I}, \boldsymbol{x} \in \mathcal{X}$

$$V_i\left(\boldsymbol{x}\right) = \sum_{a_i \in \mathcal{A}} P_i\left(a_i|\boldsymbol{x}\right) \left[\pi_i\left(a_i|\boldsymbol{x}, \boldsymbol{\theta}\right) + e^{\boldsymbol{P}}_i\left(a_i, \boldsymbol{x}\right)\right] + \beta \sum_{\boldsymbol{x}' \in \mathcal{X}} V_i\left(\boldsymbol{x}'\right) f^{\boldsymbol{P}}_{\mathcal{X}}\left(\boldsymbol{x}'|\boldsymbol{x}\right). \tag{3}$$

The first system of nonlinear equations specifies that for each i and $\boldsymbol{x}$, given conditional choice probabilities of all players $\boldsymbol{P}$, the expected value function $V_i(\boldsymbol{x})$ satisfies the Bellman equation. Here, $f^{\boldsymbol{P}}_{\mathcal{X}}\left(\boldsymbol{x}'|\boldsymbol{x}\right)$ denotes the state transition probability of $\boldsymbol{x}$, given $\boldsymbol{P}$. Specifically,

$$f^{\boldsymbol{P}}_{\mathcal{X}}\left[\boldsymbol{x}' = (s', \boldsymbol{a}') | \boldsymbol{x} = (s, \tilde{\boldsymbol{a}})\right] = \left[\prod_{j=1}^{N} P_j\left(a'_j|\boldsymbol{x}\right)\right] f_{\mathcal{S}}(s'|s). \tag{4}$$

Given the assumption that $\varepsilon_i\left(a_i\right)$ follows a type-I extreme value distribution with the scale parameter σ, we have[2]

$$e^{\boldsymbol{P}}_i\left(a_i, \boldsymbol{x}\right) = \text{Euler's constant} - \sigma \log\left[P_i\left(a_i|\boldsymbol{x}\right)\right]. \tag{5}$$

[2]For additional details, see the discussion on page 10 of Aguirregabiria and Mira (2007).

II. Bayes–Nash Equilibrium.

The second system of equations characterizes a Bayes–Nash equilibrium in conditional choice probabilities $\boldsymbol{P}$. First, we define player i's conditional choice-specific expected value function as

$$v_i\left(a_i|\boldsymbol{x}\right) = \pi_i\left(a_i|\boldsymbol{x},\boldsymbol{\theta}\right) + \beta \sum_{\boldsymbol{x}'\in\mathcal{X}} V_i\left(\boldsymbol{x}'\right) f_i^{\boldsymbol{P}}\left(\boldsymbol{x}'|\boldsymbol{x},a_i\right), \tag{6}$$

where $f_i^{\boldsymbol{P}}\left(\boldsymbol{x}'|\boldsymbol{x},a_i\right)$ denotes the state transition probability conditional on the current state $\boldsymbol{x}$, player i's action a_i, and his beliefs $\boldsymbol{P}$ over the conditional choice probabilities of all other players. Specifically,

$$f_i^{\boldsymbol{P}}\left[\boldsymbol{x}' = (s',\boldsymbol{a}')|\boldsymbol{x} = (s,\tilde{\boldsymbol{a}}),a_i\right] = f_{\mathcal{S}}\left(s'|s\right)\mathbf{1}\left\{a_i' = a_i\right\}\prod_{j\in\mathcal{I}\setminus i} P_j\left(a_j'|\boldsymbol{x}\right). \tag{7}$$

After the private shocks $\boldsymbol{\varepsilon}_i = [\varepsilon_i(0),\varepsilon_i(1)]$ are observed, player i chooses action $a_i = j$ if and only if

$$j \in \arg\max_{k\in\mathcal{A}}\left\{v_i\left(a_i = k|\boldsymbol{x}\right) + \varepsilon_i\left(a_i = k\right)\right\}.$$

The conditional choice probability is then defined as

$$P_i\left(a_i = j|\boldsymbol{x}\right) = \Pr\left[\boldsymbol{\varepsilon}_i \left| v_i\left(a_i = j|\boldsymbol{x}\right) + \varepsilon_i\left(a_i = j\right) > \max_{k\in\mathcal{A}\setminus j}\left\{v_i\left(a_i = k|\boldsymbol{x}\right) + \varepsilon_i\left(a_i = k\right)\right\}\right.\right].$$

The assumption of a type-I extreme value distribution for $\boldsymbol{\varepsilon_i}$ yields the following closed-form expression to characterize a Bayes–Nash equilibrium in conditional choice probabilities $\boldsymbol{P}$:

$$P_i\left(a_i = j|\boldsymbol{x}\right) = \frac{\exp\left[v_i\left(a_i = j|\boldsymbol{x}\right)\right]}{\sum_{k\in\mathcal{A}}\exp\left[v_i\left(a_i = k|\boldsymbol{x}\right)\right]}, \qquad \forall i\in\mathcal{I}, j\in\mathcal{A}, \boldsymbol{x}\in\mathcal{X}, \tag{8}$$

where $v_i\left(a_i = j|\boldsymbol{x}\right)$ is defined in equation (6).

This second system of nonlinear equations specifies that across all states, players' conditional choice probabilities are in mutual best response, given that each player's beliefs are consistent with the choice-specific expected value functions of all players.

To simplify the notation, we denote the two systems of equations (3) and (8) that characterize a Markov perfect equilibrium by

$$\boldsymbol{V} = \Psi^{\boldsymbol{V}}\left(\boldsymbol{V},\boldsymbol{P},\boldsymbol{\theta}\right), \tag{9}$$

$$\boldsymbol{P} = \Psi^{\boldsymbol{P}}\left(\boldsymbol{V},\boldsymbol{P},\boldsymbol{\theta}\right). \tag{10}$$

3 Estimation

In this section, we first describe the data generating process and then present a constrained optimization approach for ML estimation of this dynamic game. Finally, we discuss other likelihood-based estimators proposed in the literature.

3.1 Data Generating Process

The data consist of observations from M independent markets over T periods. We assume these M markets follow the same exogenous process $f_{\mathcal{S}}(s'|s)$ for the market-size transitions and that players' decisions are independent across these markets. In each market m and time period t, researchers observe the common-knowledge state variables $\bar{\boldsymbol{x}}^{mt}$ and players' actions $\bar{\boldsymbol{a}}^{mt} = (\bar{a}_1^{mt}, \ldots, \bar{a}_N^{mt})$. Let $\boldsymbol{Z} = \left\{\bar{\boldsymbol{a}}^{mt}, \bar{\boldsymbol{x}}^{mt}\right\}_{m\in\mathcal{M},t\in\mathcal{T}}$ denote the collection of data observed across markets and time.

Denote by $\boldsymbol{\theta}^0$ the true value of structural parameters in the population. The vector $(\boldsymbol{V}^0, \boldsymbol{P}^0)$ contains the corresponding expected value functions and conditional choice probabilities that simultaneously solve equations (9) and (10) at $\boldsymbol{\theta}^0$. In a dynamic game, multiple Markov perfect equilibria can exist. Hence, multiple pairs of $(\boldsymbol{V}^0, \boldsymbol{P}^0)$ that satisfy equations (9) and (10) can exist. If multiple equilibria exist at the true parameter vector $\boldsymbol{\theta}^0$, then we assume that only one equilibrium is played across all markets in the data, a common assumption in the literature; see Aguirregabiria and Mira (2007); Bajari, Benkard, and Levin (2007); Pakes, Ostrovsky, and Berry (2007); as well as Pesendorfer and Schmidt-Dengler (2008).[3] Thus, the data $\boldsymbol{Z} = \left\{\bar{\boldsymbol{a}}^{mt}, \bar{\boldsymbol{x}}^{mt}\right\}_{m\in\mathcal{M},t\in\mathcal{T}}$ are generated from *only one* Markov perfect equilibrium $(\boldsymbol{V}^0, \boldsymbol{P}^0)$ at the true parameter values $\boldsymbol{\theta}^0$.

3.2 ML Estimation

If the observed actions $\bar{\boldsymbol{a}}^{mt}$ at observed state $\bar{\boldsymbol{x}}^{mt}$ are generated by $\boldsymbol{P}$ for all m and t, then the logarithm of the likelihood function, given data $\boldsymbol{Z} = \left\{\bar{\boldsymbol{a}}^{mt}, \bar{\boldsymbol{x}}^{mt}\right\}_{m\in\mathcal{M},t\in\mathcal{T}}$, is

$$\mathcal{L}\left(\boldsymbol{Z}, \boldsymbol{P}\right) = \frac{1}{M}\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{t=1}^{T}\log P_i\left(\bar{a}_i^{mt}|\bar{\boldsymbol{x}}^{mt}\right). \tag{11}$$

Note that only the conditional choice probabilities $\boldsymbol{P}$ appear in the logarithm of the likelihood function. To ensure that the conditional choice probabilities $\boldsymbol{P}$ and expected value functions $\boldsymbol{V}$ are consistent with Bayes–Nash equilibrium at the given structural parameters $\boldsymbol{\theta}$, we impose equations (9) and (10) as constraints. Thus, a constrained optimization formulation of the ML estimation problem of this dynamic game is

$$\begin{aligned} \max_{(\boldsymbol{\theta},\boldsymbol{P},\boldsymbol{V})} \quad & \mathcal{L}\left(\boldsymbol{Z}, \boldsymbol{P}\right) \\ \text{subject to} \quad & \boldsymbol{V} = \Psi^{\boldsymbol{V}}\left(\boldsymbol{V}, \boldsymbol{P}, \boldsymbol{\theta}\right) \\ & \boldsymbol{P} = \Psi^{\boldsymbol{P}}\left(\boldsymbol{V}, \boldsymbol{P}, \boldsymbol{\theta}\right). \end{aligned} \tag{12}$$

Let the vector $(\boldsymbol{\theta}^{MLE}, \boldsymbol{P}^{MLE}, \boldsymbol{V}^{MLE})$ be a solution of the constrained optimization problem defined above. Then $\boldsymbol{\theta}^{MLE}$ denotes the ML estimator.

While it has been stated in the literature that when computing a ML estimator researchers must solve for all the Markov perfect equilibria at each candidate of the structural parameter vector (see for example, Aguirregabiria and Mira (2007, p. 16) as well as Kasahara and Shimotsu (2012)),

[3] Moment inequality estimators, such as those investigated in Ciliberto and Tamer (2009) or Pakes, Porter, Ho, and Ishii (2011) or Tamer (2003), do not require this assumption.

this statement is true only when researchers use the nested fixed-point algorithm. When the constrained optimization approach is used, one does not need to solve for all Markov perfect equilibria at every guess of the structural parameters. Two reasons exist: first, modern constrained optimization solvers do not force the constraints to be satisfied during the iteration process; constraints are satisfied (and an equilibrium solved) only when the iterates converge to a (local) solution; second, the constrained optimization approach only needs to find those equilibria together with structural parameters that are local solutions and satisfy the corresponding first-order conditions of the constrained optimization problem (12). Any pair of a vector of structural parameters and a corresponding equilibrium that does not satisfy the first-order conditions of (12) is not a solution to the ML estimation problem. This characterization permits one to eliminate a large set of equilibria together with structural parameters that do not need to be solved by the constrained optimization approach.

3.3 Alternative Dynamic Games Estimators

Hotz and Miller (1993) proposed a two-step estimation strategy within the context of single-agent dynamic models. The main insight of Hotz and Miller was to estimate the expected value function directly from the data without solving the Bellman equation, hence reducing the computational burden of estimating dynamic models. Subsequently, researchers have generalized this idea to estimate dynamic games and have developed various two-step estimators that are computationally light and easy to implement. For example, see Bajari, Benkard, and Levin (2007); Pakes, Ostrovsky, and Berry (2007); as well as Pesendorfer and Schmidt-Dengler (2008). A potential drawback of two-step estimators is that they can be more biased than the ML estimator in finite samples, particularly when the first-step estimates are imprecise, or if a suitable criterion function is not used in the second step; see the discussion in Pakes, Ostrovsky, and Berry (2007).

In an effort to reduce finite-sample bias associated with two-step estimators, Aguirregabiria and Mira (2007) proposed the NPL estimator and the NPL algorithm, a recursive computational procedure over the two-step PML estimator to compute the NPL estimator. While the NPL estimator performed well in their Monte Carlo experiments, convergence of the NPL algorithm can be a problem. Furthermore, the NPL algorithm may converge to the wrong estimates if the data are generated by an equilibrium that is unstable under best response iterations; see Pesendorfer and Schmidt-Dengler (2010) for such an example and Su (2012) on the performance of the NPL algorithm in a static discrete-choice game. Kasahara and Shimotsu (2012) provided theoretical analyses of the convergence of the NPL algorithm and proposed modified NPL algorithms to alleviate the convergence issue of the NPL algorithm.

We describe three approaches to estimating dynamic games: the 2S-PML estimator, the NPL algorithm of Aguirregabiria and Mira (2007), and the NPL-Λ algorithm of Kasahara and Shimotsu (2012) for computing the NPL estimator.[4] We do not discuss other two-step estimators such as those of Bajari, Benkard, and Levin (2007); Pakes, Ostrovsky, and Berry (2007) as well as Pesendorfer and Schmidt-Dengler (2008). Instead, we focus on comparing the performance of the ML estimator with that of alternative likelihood-based estimators in our Monte Carlo experiments.

[4]Kasahara and Shimotsu (2012) also proposed a recursive projection method and a q-NPL algorithm. We do not consider these two methods here because they are more computationally demanding.

3.3.1 Two-Step Pseudo-Maximum Likelihood

In the first step of a two-step estimator, one can nonparametrically estimate the conditional choice probabilities from the observed data $\boldsymbol{Z}$ using, for example, the frequency estimator. Denote by $\widehat{\boldsymbol{P}}$ a consistent estimator of the true conditional choice probabilities $\boldsymbol{P}^0$. This nonparametric estimate $\widehat{\boldsymbol{P}}$ is then fixed and used to evaluate the right-hand side of equations (9) and (10):[5]

$$\begin{aligned} \boldsymbol{V} &= \Psi^{\boldsymbol{V}}\left(\boldsymbol{V}, \widehat{\boldsymbol{P}}, \boldsymbol{\theta}\right) \\ \boldsymbol{P} &= \Psi^{\boldsymbol{P}}\left(\boldsymbol{V}, \widehat{\boldsymbol{P}}, \boldsymbol{\theta}\right). \end{aligned}$$

The second step of the 2S-PML estimator involves solving the following optimization problem:

$$\begin{aligned} \max_{(\boldsymbol{\theta},\boldsymbol{P},\boldsymbol{V})} \quad & \mathcal{L}\left(\boldsymbol{Z}, \boldsymbol{P}\right) \\ \text{subject to} \quad & \boldsymbol{V} = \Psi^{\boldsymbol{V}}\left(\boldsymbol{V}, \widehat{\boldsymbol{P}}, \boldsymbol{\theta}\right) \\ & \boldsymbol{P} = \Psi^{\boldsymbol{P}}\left(\boldsymbol{V}, \widehat{\boldsymbol{P}}, \boldsymbol{\theta}\right) \end{aligned}$$

or, equivalently,

$$\begin{aligned} \max_{(\boldsymbol{\theta},\boldsymbol{V})} \quad & \mathcal{L}\left(\boldsymbol{Z}, \Psi^{\boldsymbol{P}}\left(\boldsymbol{V}, \widehat{\boldsymbol{P}}, \boldsymbol{\theta}\right)\right) \\ \text{subject to} \quad & \boldsymbol{V} = \Psi^{\boldsymbol{V}}\left(\boldsymbol{V}, \widehat{\boldsymbol{P}}, \boldsymbol{\theta}\right). \end{aligned} \tag{13}$$

From equation (3), we can see that once $\boldsymbol{P}$ is fixed at $\widehat{\boldsymbol{P}}$, the variables $\boldsymbol{V}$ and $\boldsymbol{\theta}$ are additively separable. Define $\boldsymbol{V}_i = [V_i(x)]_{\boldsymbol{x}\in\mathcal{X}} \in \mathbb{R}^{|\mathcal{X}|}$, $\boldsymbol{F}_{\mathcal{X}}^{\widehat{\boldsymbol{P}}} = \left[f_{\mathcal{X}}^{\widehat{\boldsymbol{P}}}\left(\boldsymbol{x}'|\boldsymbol{x}\right)\right]_{\boldsymbol{x},\boldsymbol{x}'\in\mathcal{X}} \in \mathbb{R}^{|\mathcal{X}|\times|\mathcal{X}|}$, $\widehat{\boldsymbol{P}}_i(a_i) = [\widehat{P}_i(a_i|\boldsymbol{x})]_{\boldsymbol{x}\in\mathcal{X}} \in \mathbb{R}^{|\mathcal{X}|}$, $\boldsymbol{e}_i^{\widehat{\boldsymbol{P}}}(a_i) = [e_i^{\widehat{\boldsymbol{P}}}(a_i, \boldsymbol{x})]_{\boldsymbol{x}\in\mathcal{X}} \in \mathbb{R}^{|\mathcal{X}|}$, and $\boldsymbol{\pi}_i(a_i, \boldsymbol{\theta}) = [\pi_i(a_i|\boldsymbol{x}, \boldsymbol{\theta})]_{\boldsymbol{x}\in\mathcal{X}} \in \mathbb{R}^{|\mathcal{X}|}$. Equation (3) can then be rewritten in matrix notation as

$$\left[\mathbf{I} - \beta \boldsymbol{F}_{\mathcal{X}}^{\widehat{\boldsymbol{P}}}\right] \boldsymbol{V}_i = \sum_{a_i\in\mathcal{A}} \left[\widehat{\boldsymbol{P}}_i(a_i) \circ \boldsymbol{\pi}_i(a_i, \boldsymbol{\theta})\right] + \sum_{a_i\in\mathcal{A}} \left[\widehat{\boldsymbol{P}}_i(a_i) \circ \boldsymbol{e}_i^{\widehat{\boldsymbol{P}}}(a_i)\right]$$

where $\mathbf{I}$ is an identity matrix in $\mathbb{R}^{|\mathcal{X}|\times|\mathcal{X}|}$ and the notation $\boldsymbol{A} \circ \boldsymbol{B}$ denotes the Hadamard product of two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$. Thus, one can explicitly express $\boldsymbol{V}_i$ in terms of structural parameters $\boldsymbol{\theta}$:

$$\boldsymbol{V}_i = \left[\mathbf{I} - \beta \boldsymbol{F}_{\mathcal{X}}^{\widehat{\boldsymbol{P}}}\right]^{-1} \left\{ \sum_{a_i\in\mathcal{A}} \left[\widehat{\boldsymbol{P}}_i(a_i) \circ \boldsymbol{\pi}_i(a_i, \boldsymbol{\theta})\right] + \sum_{a_i\in\mathcal{A}} \left[\widehat{\boldsymbol{P}}_i(a_i) \circ \boldsymbol{e}_i^{\widehat{\boldsymbol{P}}}(a_i)\right] \right\}, \quad \text{for all } i \in \mathcal{I}, \tag{14}$$

or in a compact notation

$$\boldsymbol{V} = \boldsymbol{\Gamma}(\boldsymbol{\theta}, \widehat{\boldsymbol{P}}). \tag{15}$$

By replacing the constraint in problem (13) with equation (15), through a simple elimination of variables $\boldsymbol{V}$, the optimization problem (13) is equivalent to the unconstrained optimization problem

$$\max_{\boldsymbol{\theta}} \mathcal{L}\left(\boldsymbol{Z}, \Psi^{\boldsymbol{P}}\left(\boldsymbol{\Gamma}(\boldsymbol{\theta}, \widehat{\boldsymbol{P}}), \widehat{\boldsymbol{P}}, \boldsymbol{\theta}\right)\right).$$

[5] This requires researchers to evaluate $f_{\mathcal{X}}^{\boldsymbol{P}}\left(\boldsymbol{x}'|\boldsymbol{x}\right)$, $e_i^{\boldsymbol{P}}\left(a_i, \boldsymbol{x}\right)$, and $f_i^{\boldsymbol{P}}\left(\boldsymbol{x}'|\boldsymbol{x}, a_i\right)$ using $\widehat{\boldsymbol{P}}$ in equations (4), (5), and (7), respectively.

The 2S-PML estimator is then defined as

$$\boldsymbol{\theta}^{2S-PML} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ \mathcal{L}\left(\boldsymbol{Z}, \Psi^{\boldsymbol{P}}\left(\boldsymbol{\Gamma}(\boldsymbol{\theta}, \widehat{\boldsymbol{P}}), \widehat{\boldsymbol{P}}, \boldsymbol{\theta}\right)\right). \tag{16}$$

The 2S-PML estimator is considered computationally light because it avoids solving Bayes–Nash equilibrium equation (10); also researchers estimate $\widehat{\boldsymbol{P}}$ directly from the data. Nevertheless, solving the optimization problem (16) or, equivalently, solving problem (13) in the second step, although easier than problem (12) for the ML estimator, may not be trivial. Researchers still need to solve the Bellman equation for each player as constraints in (13) or invert the matrix $\left[\mathbf{I} - \beta \boldsymbol{F}_{\mathcal{X}}^{\widehat{\boldsymbol{P}}}\right]$ in (14) for every guess of structural parameters $\boldsymbol{\theta}$ in solving the unconstrained optimization problem (16), a task that can be computationally expensive when the state space $|\mathcal{X}|$ is large.

Note, too, that at the solution $\boldsymbol{\theta}^{2S-PML}$, the first-step estimate $\widehat{\boldsymbol{P}}$ may not satisfy Bayes–Nash equation (10) and is not a Bayes–Nash equilibrium. In finite samples, the bias in the first-step estimate $\widehat{\boldsymbol{P}}$ can potentially lead to large biases in parameter estimates $\boldsymbol{\theta}^{2S-PML}$ in the second step, particularly when the pseudo-likelihood function is used as the criterion function; see the discussion in Pakes, Ostrovsky, and Berry (2007).

3.3.2 NPL Estimator

Aguirregabiria and Mira (2007) proposed an NPL estimator for estimating dynamic discrete-choice games. Any point $(\tilde{\boldsymbol{\theta}}, \widetilde{\boldsymbol{P}})$ that satisfies the conditions below is called an NPL fixed point:

$$\begin{aligned} \tilde{\boldsymbol{\theta}} &= \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ \mathcal{L}\left(\boldsymbol{Z}, \Psi^{\boldsymbol{P}}\left(\boldsymbol{\Gamma}(\boldsymbol{\theta}, \widetilde{\boldsymbol{P}}), \widetilde{\boldsymbol{P}}, \boldsymbol{\theta}\right)\right). \\ \widetilde{\boldsymbol{P}} &= \Psi^{\boldsymbol{P}}\left(\boldsymbol{\Gamma}(\tilde{\boldsymbol{\theta}}, \widetilde{\boldsymbol{P}}), \widetilde{\boldsymbol{P}}, \tilde{\boldsymbol{\theta}}\right). \end{aligned} \tag{17}$$

In principle, more than one NPL fixed point can exist. An NPL estimator $(\boldsymbol{\theta}^{NPL}, \boldsymbol{P}^{NPL})$ is the NPL fixed point that yields the highest objective value in (17). Note that the NPL estimator satisfies Bayes–Nash equilibrium equation (10). Thus, one would expect it to perform better than the 2S-PML estimator in finite samples.

Aguirregabiria and Mira (2007) also proposed a computational procedure referred to as the NPL algorithm to find an NPL fixed point. The NPL algorithm recursively iterates over the 2S-PML estimator and is described as follows. First, choose an initial guess of equilibrium probabilities $\widetilde{\boldsymbol{P}}_0$. For $K \geq 1$, the NPL algorithm iterates the following steps until convergence or until the maximum number of iterations $\bar{K}$ is reached:

Step 1. Given $\widetilde{\boldsymbol{P}}_{K-1}$, solve $\tilde{\boldsymbol{\theta}}_K = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ \mathcal{L}\left(\boldsymbol{Z}, \Psi^{\boldsymbol{P}}\left(\boldsymbol{\Gamma}(\boldsymbol{\theta}, \widetilde{\boldsymbol{P}}_{K-1}), \widetilde{\boldsymbol{P}}_{K-1}, \boldsymbol{\theta}\right)\right)$.

Step 2. Given $\tilde{\boldsymbol{\theta}}_K$, update $\widetilde{\boldsymbol{P}}_K$ by $\widetilde{\boldsymbol{P}}_K = \Psi^{\boldsymbol{P}}\left(\boldsymbol{\Gamma}(\tilde{\boldsymbol{\theta}}_K, \widetilde{\boldsymbol{P}}_{K-1}), \widetilde{\boldsymbol{P}}_{K-1}, \tilde{\boldsymbol{\theta}}_K\right)$; increase K by 1.

If the maximum number of iterations $\bar{K}$ is reached before the NPL algorithm converges, then we declare a failed run and restart with a new initial guess $\widetilde{\boldsymbol{P}}_0$. If the NPL algorithm converges after K iterations ($K \leq \bar{K}$), then $(\tilde{\boldsymbol{\theta}}_K, \widetilde{\boldsymbol{P}}_K)$ satisfies the NPL fixed-point conditions (17).

Researchers have expressed concerns involving the convergence properties of the NPL algorithm. For example, see Pakes, Ostrovsky, and Berry (2007); Pesendorfer and Schmidt-Dengler (2010); and Su (2012). While Aguirregabiria and Mira (2007) reported that they always obtained convergence of the NPL algorithm in their Monte Carlo experiments, examples in Su (2012) indicate that the NPL algorithm often fails to converge, even in static discrete-choice games. Even worse, Pesendorfer and Schmidt-Dengler (2010) demonstrated in a stylized example and Su (2012) demonstrated in Monte Carlo experiments of static discrete-choice games that NPL can converge to the wrong estimates. In a recent paper, Kasahara and Shimotsu (2012) demonstrated that the NPL algorithm will converge, provided that a local stability condition is satisfied at the solution. Without knowing the true parameter values, however, this local stability condition cannot be verified. Therefore, in practice, the theoretical analysis provided by Kasahara and Shimotsu does not inform researchers *a priori* whether the NPL algorithm will converge or whether it converges to the correct estimates. In summary, while the NPL estimator is well defined, the NPL algorithm may fail to converge; even if the NPL algorithm converges, it may fail to recover the true values of the underlying primitives.

3.3.3 A Modified NPL Algorithm

To improve the convergence properties of the NPL algorithm, Kasahara and Shimotsu (2012) introduced the NPL-Λ algorithm to compute an NPL estimator. The NPL-Λ algorithm alters the updating of $\widehat{\boldsymbol{P}}_K$ in Step 2 of the NPL algorithm to

$$\widetilde{\boldsymbol{P}}_K = \left(\Psi^{\boldsymbol{P}}\left(\boldsymbol{\Gamma}(\tilde{\boldsymbol{\theta}}_K, \widetilde{\boldsymbol{P}}_{K-1}), \widetilde{\boldsymbol{P}}_{K-1}, \tilde{\boldsymbol{\theta}}_K\right)\right)^{\lambda}\left(\widetilde{\boldsymbol{P}}_{K-1}\right)^{1-\lambda} \tag{18}$$

where λ is chosen to be between 0 and 1. Note that the NPL-Λ algorithm when λ is one is identical to the NPL algorithm.

The idea behind modifying the second step of the NPL algorithm is similar to that of the successive over- and under-relaxation methods in numerical analysis. Such methods are used to improve the contraction rate of diverging iterative processes; see Ortega and Rheinboldt (1970). The scalar λ represents a partial step length used in the dampening procedure. Ideally, the choice of λ depends on the spectral radius of a Jacobian matrix at the solution $\boldsymbol{\theta}^{NPL}$. In practice, since $\boldsymbol{\theta}^{NPL}$ is unknown prior to estimation, Kasahara and Shimotsu proposed computing the spectral radius of the Jacobian matrix at each iteration of the NPL-Λ algorithm. Computing the spectral radius of a high-dimensional matrix is not a computationally trivial task; repeating this computation at each iteration increases significantly the computational cost of implementing the NPL-Λ algorithm.[6]

3.4 Scalability of the Constrained Optimization Approach

Since the constrained optimization approach requires optimizing over a much larger number of dimensions, one potential concern is its capability to estimate empirically relevant dynamic games; see Aguirregabiria and Nevo (2010) as well as Kasahara and Shimotsu (2012). To address this concern, we demonstrate below that the constraint Jacobian as well as the Hessian of the Lagrangian

[6] In their Monte Carlo experiment, Kasahara and Shimotsu (2012) chose $\lambda = 0.825$, which was obtained by computing the spectral radius of the Jacobian at the true parameter values $\boldsymbol{\theta}^0$.

of the constrained optimization problem (12) are sparse under certain modeling specifications. State-of-the-art constrained optimization solvers use sparse matrix routines to exploit this sparsity. Consequently, the solvers can accelerate the computations, economize on memory usage, and permit users to solve high-dimensional optimization problems, those on the order of 100,000 variables and constraints.[7]

As discussed in Section 3.3, the size of the optimization problem (13) in the second step of the 2S-PML estimator as well as the NPL and NPL-Λ algorithms is half the size of the constrained optimization problem (12) for ML estimation. For large-scale dynamic games, the problem (13) will be high dimensional as well. Without utilizing sparse matrix techniques, researchers will not be able to solve a high-dimensional problem like (13) or invert the high-dimensional matrix $\left[\mathbf{I} - \beta \boldsymbol{F}_{\mathcal{X}}^{\widehat{\boldsymbol{P}}}\right]$ in $\mathbb{R}^{|\mathcal{X}|\times|\mathcal{X}|}$ in equation (14).

We derive an upper bound on the density of the constraint Jacobian and Hessian of the Lagrangian, denoted by S_J and S_H respectively, of the constrained optimization problem (12).[8] Recall that $|\mathcal{S}|$ is the number of grid points in the market size state space and $|\theta|$ is the number of structural parameters. Let δ_s denote the maximum incremental change in market size in one period; for example, given s^t, the market size in the next period $s^{t+1} \in \{s^t - \delta_s, s^t - \delta_s + 1, \ldots, s^t, \ldots, s^t + \delta_s - 1, s^t + \delta_s\}$.

Proposition 1 (Density Bounds)**.**

(a) Given binary action space $\mathcal{A} = \{0, 1\}$, we have

$$S_J \leq \frac{2}{9}\left(\frac{2\delta_s + 1}{N \cdot |\mathcal{S}|} + \frac{|\boldsymbol{\theta}| + 1}{N \cdot |\mathcal{S}| \cdot 2^N} + \frac{1}{|\mathcal{S}| \cdot 2^{N-1}}\right),$$

$$S_H \leq \frac{1}{9}\left(\frac{2\delta_s + 1}{N \cdot |\mathcal{S}|} + \frac{4}{|\mathcal{S}| \cdot 2^N} + \frac{4 \cdot (2\delta_s + 1)}{|\mathcal{S}|} + \frac{6 \cdot |\boldsymbol{\theta}|}{N \cdot |\mathcal{S}| \cdot 2^N} + \left(\frac{|\boldsymbol{\theta}|}{N \cdot |\mathcal{S}| \cdot 2^N}\right)^2\right).$$

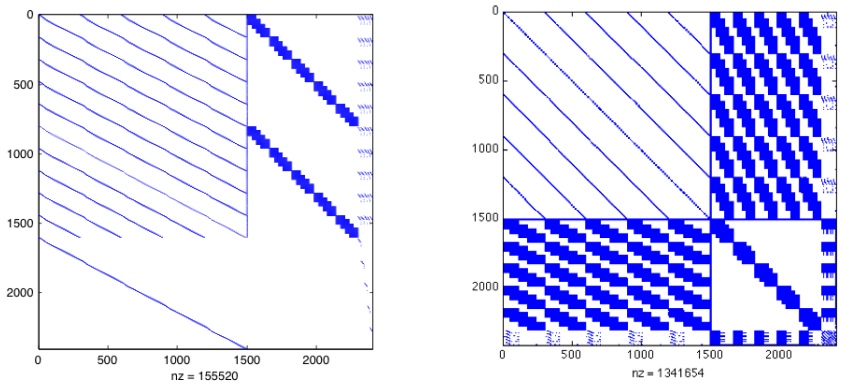
(b) The upper bounds on S_J and S_H are decreasing in $|\mathcal{S}|$ and N.

(c) For fixed $|\mathcal{S}|$ and N, the upper bounds on S_J and S_H decrease when δ_s decreases.

The proof is provided in Appendix A. As the size of the state space grows large, the constraint Jacobian and Hessian matrices become more sparse, which helps to alleviate the increase in computational burden arising from having more variables and constraints. We calculate the density of the constraint Jacobian and Hessian matrices for different values of $|\mathcal{S}|$, N, and $|\theta|$ in Table 7 in Appendix B. Figure 1 illustrates the sparsity pattern of the constraint Jacobian and the Hessian for an example with $N = 5$, $|S| = 5$, and $\delta_s = 1$, which results in a constrained optimization problem with 2400 constraints and 2408 variables. The densities of the corresponding constraint Jacobian and Hessian matrices for this example are around 2.7 percent and 23 percent, respectively.

[7] All researchers need to do is provide sparsity information to optimization solvers.

[8] The density of a matrix is the ratio between the number of nonzero elements and the total number of elements in the matrix.



(a) Constraint Jacobian. (b) Hessian of the Lagrangian function.

Figure 1: Sparsity Pattern of Constraint Matrices with $|\mathcal{S}| = 5$ and $N = 5$

4 Monte Carlo Experiments

We conducted Monte Carlo experiments to investigate the performance of the ML estimator, the 2S-PML estimator, and the NPL estimator implemented by both the NPL algorithm and the NPL-Λ algorithm. We describe the experimental design in Section 4.1 and report the Monte Carlo results in Section 4.2.

4.1 Experimental Design

We considered three experiment specifications, with two cases in each experiment. In the first experiment, we used the example of Kasahara and Shimotsu (2012), which is a simplified version of the example of Aguirregabiria and Mira (2007). In the second experiment, we used the example of Aguirregabiria and Mira (2007). In the third experiment, we increased the set of possible market size values used in the second experiment. We describe the details of our experimental design below.

Experiment 1: Kasahara and Shimotsu (2012) example
This example has $N = 3$ players. The set of possible values for market size is $\mathcal{S} = \{2, 6, 10\}$, and the total number of grid points in the state space is $|\mathcal{X}| = |\mathcal{S}| \times |\mathcal{A}|^N = 3 \times 2^3 = 24$. The common-knowledge component of the per-period payoff Π_i is given as

$$\Pi_i\left(a_i^t, \boldsymbol{a}_{-i}^t, \boldsymbol{x}^t; \boldsymbol{\theta}\right) = \begin{cases} \theta^{RS} \log\left(s^t\right) + \theta^{RN} \log\left(1 + \sum_{j \neq i} a_j^t\right) - \theta_i^{FC} - \theta^{EC}\left(1 - a_i^{t-1}\right), & \text{if } a_i^t = 1, \\ 0 & \text{if } a_i^t = 0, \end{cases}$$

where $\boldsymbol{\theta} = \left(\theta^{RS}, \theta^{RN}, \boldsymbol{\theta}^{FC}, \theta^{EC}\right)$ is the vector of structural parameters with $\boldsymbol{\theta}^{FC} = \{\theta_i^{FC}\}_{i=1}^N$. For this experiment, the ML estimator solves the constrained optimization problem (12) with 216 constraints and 218 variables.

Following Kasahara and Shimotsu (2012), we chose the discount factor $\beta = 0.96$ and the scale parameter of the type-I extreme value distribution $\sigma = 1$. We fixed the values of structural parameters $\boldsymbol{\theta}^{FC} = (1.0, 0.9, 0.8)$ and $\theta^{EC} = 1$, and estimated only θ^{RS} and θ^{RN}.

We considered two sets of parameter values for θ^{RS} and θ^{RN} in this experiment:

$$\begin{aligned} &\text{Case 1:} \quad (\theta^{RN}, \theta^{RS}) = (2, 1); \\ &\text{Case 2:} \quad (\theta^{RN}, \theta^{RS}) = (4, 1). \end{aligned}$$

Experiment 2: Aguirregabiria and Mira (2007) example
This example has $N = 5$ players and five possible values for market size, $\mathcal{S} = \{1, 2, \ldots, 5\}$. The number of points in the state space is $|\mathcal{X}| = |\mathcal{S}| \times |\mathcal{A}|^N = 5 \times 2^5 = 160$. The function Π_i is given as[9]

$$\Pi_i\left(a_i^t, \boldsymbol{a}_{-i}^t, \boldsymbol{x}^t; \boldsymbol{\theta}\right) = \begin{cases} \theta^{RS} s^t + \theta^{RN} \log\left(1 + \sum_{j \neq i} a_j^t\right) - \theta_i^{FC} - \theta^{EC}\left(1 - a_i^{t-1}\right), & \text{if } a_i^t = 1, \\ 0 & \text{if } a_i^t = 0. \end{cases}$$

Following Aguirregabiria and Mira (2007), we fixed $\beta = 0.95$ and $\sigma = 1$. In this experiment, we estimated all the structural parameters $\boldsymbol{\theta}$. For this experiment, the ML estimator solves the constrained optimization problem (12) with 2,400 constraints and 2,408 variables.

We chose $\boldsymbol{\theta}^{FC} = (1.9, 1.8, 1.7, 1.6, 1.5)$ and $\theta^{EC} = 1$ as true parameter values. For θ^{RN} and θ^{RS}, we considered the following two cases:

$$\begin{aligned} &\text{Case 3:} \quad (\theta^{RN}, \theta^{RS}) = (2, 1); \\ &\text{Case 4:} \quad (\theta^{RN}, \theta^{RS}) = (4, 2). \end{aligned}$$

Note that the choices of parameter values in Case 3 are the same as those in Experiment 3 in Aguirregabiria and Mira (2007).

Experiment 3: Examples with increasing $|\mathcal{S}|$, the number of market size values
In this experiment, we considered two sets of market size values:

$$\begin{aligned} &\text{Case 5:} \quad |\mathcal{S}| = 10 \text{ with } \mathcal{S} = \{1, 2, \ldots, 10\}; \\ &\text{Case 6:} \quad |\mathcal{S}| = 15 \text{ with } \mathcal{S} = \{1, 2, \ldots, 15\}. \end{aligned}$$

All other specifications remain the same as those in Case 3 in Experiment 2. Our purpose is to investigate the performance of these estimators when estimating games with a larger number of states. For this experiment, the ML estimator solves the constrained optimization problem (12) with 4,800 constraints and 4,808 variables for $|\mathcal{S}| = 10$, and 7,200 constraints and 7,208 variables for $|\mathcal{S}| = 15$.

[9]The first term in Π_i is given as $\theta^{RS} \log\left(s^t\right)$ in equation (48) of Aguirregabiria and Mira (2007); however, their Gauss code `am_econometrica_2007_montecarlo.prg` used the term $\theta^{RS} s^t$. Thus, we decided to follow the specification in their code.

In all three experiments, the market size transition probabilities are given by the following $|\mathcal{S}| \times |\mathcal{S}|$ matrix:

$$f_{\mathcal{S}}(s^{t+1}|s^t) = \begin{pmatrix} 0.8 & 0.2 & 0 & \cdots & 0 & 0 \\ 0.2 & 0.6 & 0.2 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0.2 & 0.6 & 0.2 \\ 0 & 0 & \cdots & 0 & 0.2 & 0.8 \end{pmatrix}.$$

Data Simulation
Given the model primitives, we solved equations (9) and (10) for $(\boldsymbol{V}, \boldsymbol{P})$ at the given structural parameter values specified in each of the six cases above. For each case, we used 100 starting values to find multiple equilibria, but found only a single equilibrium. In their Monte Carlo experiments, Aguirregabiria and Mira (2007) and Kasahara and Shimotsu (2012) also found a unique equilibrium at the true parameter values for these examples. Hence, we do not need to consider equilibrium selection issues in the data generating process.

For each case in Experiments 1 and 2, we constructed data sets of three sizes with $M = 400$ markets and $T = 1, 10$ and 20 periods. For each T, we simulated 100 data sets. For Cases 5 and 6 in Experiment 3, we constructed 50 data sets with $M = 400$ markets and $T = 10$ periods.

Algorithm Implementation
We used AMPL as the programming language and KNITRO, a nonlinear optimization solver, to solve the optimization problem for each estimator. For Experiments 1 and 2, we used 10 starting values for each of the 100 data sets when implementing each estimator. For Experiment 3, we used 5 starting values for each of the 50 data sets. For Experiment 1, we set the maximum number of NPL and NPL-Λ iterations to be $\bar{K} = 250$. For Experiments 2 and 3, we set $\bar{K}$ to 100. For the NPL-Λ algorithm, we chose $\lambda = 0.5$ for the updating in equation (18).

4.2 Numerical Results

In this subsection, we discuss the results of our Monte Carlo experiments.

Experiment 1. In Table 1, we have collected the results for Case 1, with $(\theta_{RN}, \theta_{RS}) = (2, 1)$. In this case, all estimation algorithms converged for all data sets. All estimators produced fairly precise estimates, except for the 2S-PML estimator with $T = 1$. As expected, these estimates become more precise as T increases. Recall that for each data set, we used 10 starting values. The constrained optimization approach converged for around 920 runs for $T = 1$ and 980 runs for $T = 20$; all the other algorithms converged in all 1,000 runs. Note that the constrained approach is faster than either NPL or NPL-Λ; with $T = 1$, the constrained approach took only 0.27 seconds per run compared to 0.45 seconds per run for NPL, a factor of about 1.6. The speed advantage increases as T increases. With $T = 20$, the constrained approach took only 0.15 seconds per run compared to 1.01 seconds per run for NPL, a factor of more than 6.

In Table 2, we have collected the results for Case 2, with $(\theta_{RN}, \theta_{RS}) = (4, 1)$. In this case, both NPL and NPL-Λ failed to converge before reaching the maximum number of iterations $\bar{K} = 250$ for all data sets. In contrast, both the constrained approach and 2S-PML converged for all 100 data

sets, although the constrained optimization approach converged for only 735 out of 1,000 runs for $T = 1$, and 820 runs for $T = 20$. The constrained approach also yielded more precise estimates than the 2S-PML estimator. With $T = 1$, the constrained approach yielded mean estimates of 4.055 for θ^{RN} (standard deviation 0.613) and 1.003 for θ^{RS} (standard deviation 0.158), while 2S-PML gave imprecise mean estimates of 3.107 for θ^{RN} (standard deviation 0.442) and 0.839 for θ^{RS} (standard deviation 0.099).

Experiment 2. In Table 3, we have collected the results of the experiment for Case 3. In this case, NPL converged for only 53 data sets for $T = 1$ to 67 data sets for $T = 20$. The NPL-Λ algorithm worked quite well in this experiment. It significantly improved the convergence properties of the NPL algorithm, converging in all 100 data sets for different T and obtaining more accurate estimates than those of the NPL algorithm for $T = 20$. The mean estimates of the 2S-PML estimator for parameters θ_{RN} and θ_{RS} are quite biased for $T = 1$ and $T = 10$, and are more than two standard deviations away from the true parameter values. The constrained optimization approach converged for all 100 data sets for each T. Its runtime and accuracy of estimates are comparable to those of the NPL-Λ algorithm for $T = 10$, and $T = 20$. For $T = 1$, however, the constrained optimization approach was slow, needing 216 seconds per run, on average; its mean estimates are more biased than those of the NPL-Λ algorithm. Further inspection of the converged likelihood values revealed that the constrained optimization approach yielded higher likelihood values than the NPL-Λ algorithm for all 100 data sets for $T = 1$.

In Table 4, we have collected the results for Case 4. NPL converged for only 2 out of 100 data sets for $T = 1$ and failed to converge for all 100 data sets for $T = 10$ and $T = 20$. NPL-Λ performed better than the NPL algorithm, but failed more frequently than it did in Case 3, converging in 84 out of 100 data sets for $T = 1$ and only 53 data sets for $T = 20$. Similar to the findings in Case 3, the 2S-PML estimator produced inaccurate estimates of parameters θ_{RN} and θ_{RS}, particularly for $T = 1$ and $T = 10$; for instance, with $T = 1$ the mean 2S-PML estimates of θ_{RN} and θ_{RS} are 0.624 (standard deviation 0.393) and 0.759 (standard deviation 0.150), respectively, while the true values are $\theta_{RN} = 4$ and $\theta_{RS} = 2$. For $T = 20$, the mean estimates of θ_{RN} and θ_{RS} of 2S-PML are 4 standard deviations away from the true values. The constrained optimization approach converged for all 100 data sets for different T, although it converged for only 582 runs (out of 1,000) for $T = 1$; it also produced fairly accurate estimates of all structural parameters.

Note that in Experiments 1 and 2, the constrained optimization approach failed more frequently for $T = 1$ than for $T = 20$; its computing time was also longer for $T = 1$ than for $T = 20$. One possible explanation is that with fewer observations in the data ($T = 1$), the likelihood function is flatter than that with more observations ($T = 20$), which makes the optimization problems in the former case more challenging to solve.

Experiment 3. In Table 5, we have collected the results of Case 5. With $|\mathcal{S}| = 10$, the NPL algorithm often failed to converge, finding a solution for only 23 of 50 data sets (or 76 out of 250 runs), and produced highly biased estimates of the parameter θ_{RS} for the converged data sets, with a mean estimate of 1.966 (standard deviation 0.036) versus the true value $\theta_{RS} = 1$. The constrained optimization approach and the NPL-Λ algorithm converged for all 50 data sets and produced similar estimates for all structural parameters. However, with $|\mathcal{S}| = 15$ as shown in Table 6, the NPL algorithm failed to converge for all 50 data sets, and the NPL-Λ algorithm converged for only 1 of 50 data sets. The constrained optimization approach converged for all 50 data sets (or

222 out of 250 runs), and produced accurate estimates of all parameters. In both Cases 5 and 6, the 2S-PML estimator produced highly biased estimates of θ_{RN} and θ_{RS}. In terms of computational speed, the 2S-PML estimator was around 25 to 30 times faster than the constrained optimization approach in this experiment.

Recall that the specifications in Experiment 3 are identical to those in Case 2 in Experiment 2 except that we increased the number of grid points in the market size state space from 5 to 10 and 15 in Case 5 and 6, respectively. It is surprising to see that the NPL and NPL-Λ algorithms failed to converge for all data sets when we simply increased the size of the state space, but fixed the true parameter values in the data generating process. Our findings suggest that, for dynamic games with a higher dimensional state space, NPL and NPL-Λ algorithms may not perform as well as researchers have believed. While the NPL-Λ algorithm does improve the convergence of the NPL algorithm, the NPL-Λ algorithm still fails to converge in several cases, as demonstrated in our Monte Carlo experiments.

Aguirregabiria and Nevo (2012) have argued that with multiple equilibria, it is reasonable to assume that only Lyapunov-stable (or best-response stable) equilibria will be played in the data, in which case the NPL algorithm should converge. As we stated earlier, however, a unique equilibrium exists at the true parameter values in the examples used in our Monte Carlo experiments. The fact that the NPL and NPL-Λ algorithms can fail to converge in all data sets, even when the model has a unique equilibrium at the true parameter values, suggests that in practice these recursive methods are not reliable computational algorithms and should be used with caution.

5 Conclusion

In this paper, we have formulated the ML estimation of dynamic discrete-choice games of incomplete information as a constrained optimization problem. We have compared the numerical performance of our constrained approach to the 2S-PML estimator, which suffers from large finite-sample biases in many cases, and to the NPL and NPL-Λ algorithms, which suffer from convergence issues. The constrained optimization approach for ML estimation produces accurate parameter estimates and has superior convergence properties when compared to the NPL and NPL-Λ algorithms.

Our Monte Carlo experiments have demonstrated that the lack of convergence of NPL and NPL-Λ is not a trivial issue in practice. In contrast, the performance of the constrained optimization approach is robust to changes in both the size of the model and to the true values of the structural parameters. Furthermore, the constrained optimization approach has a speed advantage over NPL and NPL-Λ that is increasing with the number of variables and states, so long as the constraint Jacobian and the Hessian of the constrained optimization problem are relatively sparse. Our results suggest that the constrained optimization approach for ML estimation is, indeed, practical and computationally feasible for estimating dynamic games with a moderate number of grid points in the state space.

For future research, we plan to explore using the constrained approach in games with unobserved heterogeneity, such as the model in Arcidiacono and Miller (2011). We also plan to improve the numerical implementation of the constrained optimization approach to estimate dynamic games with higher dimensional state spaces.

Table 1: Monte Carlo Results for Case 1.

M	T	Estimator	Estimates		CPU Time (in sec.)	Data Sets Converged	Runs Converged	Avg. NPL(-Λ) Iter.
			θ_{RN}	θ_{RS}				
		Truth	2	1	–	–	–	–
400	1	MLE	1.895 (0.580)	0.961 (0.156)	0.27	100	917	–
400	1	2S-PML	1.134 (0.616)	0.753 (0.171)	0.02	100	1000	–
400	1	NPL	1.909 (0.628)	0.964 (0.168)	0.45	100	1000	30
400	1	NPL-Λ	1.909 (0.628)	0.964 (0.168)	0.42	100	1000	28
400	10	MLE	1.970 (0.158)	0.992 (0.042)	0.16	100	964	–
400	10	2S-PML	1.819 (0.236)	0.951 (0.062)	0.03	100	1000	–
400	10	NPL	1.963 (0.191)	0.991 (0.050)	0.61	100	1000	22
400	10	NPL-Λ	1.963 (0.191)	0.991 (0.050)	0.56	100	1000	20
400	20	MLE	2.001 (0.118)	1.000 (0.033)	0.15	100	979	–
400	20	2S-PML	1.923 (0.158)	0.979 (0.042)	0.06	100	1000	–
400	20	NPL	1.999 (0.129)	0.999 (0.036)	1.01	100	1000	22
400	20	NPL-Λ	1.999 (0.129)	0.999 (0.036)	0.91	100	1000	20

Table 2: Monte Carlo Results for Case 2.

M	T	Estimator	Estimates		CPU Time (in sec.)	Data Sets Converged	Runs Converged	Avg. NPL(-Λ) Iter.
			θ_{RN}	θ_{RS}				
		Truth	4	1	–	–	–	–
400	1	MLE	4.055 (0.613)	1.003 (0.158)	0.61	100	735	–
400	1	2S-PML	3.107 (0.442)	0.839 (0.099)	0.02	100	1000	–
400	1	NPL	N/A (N/A)	N/A (N/A)	1.68	0	0	250
400	1	NPL-Λ	N/A (N/A)	N/A (N/A)	1.68	0	0	250
400	10	MLE	4.003 (0.039)	1.000 (0.016)	0.50	100	767	–
400	10	2S-PML	3.902 (0.099)	0.983 (0.025)	0.04	100	1000	–
400	10	NPL	N/A (N/A)	N/A (N/A)	7.61	0	0	250
400	10	NPL-Λ	N/A (N/A)	N/A (N/A)	7.54	0	0	250
400	20	MLE	4.003 (0.032)	1.001 (0.011)	0.47	100	820	–
400	20	2S-PML	3.954 (0.084)	0.992 (0.019)	0.06	100	1000	–
400	20	NPL	N/A (N/A)	N/A (N/A)	12.38	0	0	250
400	20	NPL-Λ	N/A (N/A)	N/A (N/A)	12.41	0	0	250

Table 3: Monte Carlo Results for Case 3.

M	T	Estimator	Estimates								CPU Time (in sec.)	Data Sets Converged	Runs Converged	Avg. NPL(-Λ) Iter.
			$\theta_{FC,1}$	$\theta_{FC,2}$	$\theta_{FC,3}$	$\theta_{FC,4}$	$\theta_{FC,5}$	θ_{EC}	θ_{RN}	θ_{RS}				
		Truth	1.9	1.8	1.7	1.6	1.5	1	2	1	–	–	–	–
400	1	MLE	1.941 (0.272)	1.847 (0.251)	1.765 (0.260)	1.656 (0.266)	1.570 (0.279)	0.959 (0.201)	2.485 (1.542)	1.139 (0.425)	216.31	100	736	–
400	1	2S-PML	1.608 (0.222)	1.496 (0.213)	1.425 (0.214)	1.306 (0.210)	1.196 (0.187)	1.174 (0.141)	0.162 (0.295)	0.433 (0.093)	1.34	100	1000	–
400	1	NPL	1.907 (0.217)	1.815 (0.201)	1.716 (0.203)	1.573 (0.196)	1.473 (0.189)	1.074 (0.111)	1.413 (0.484)	0.843 (0.137)	45.85	53	530	64.44
400	1	NPL-Λ	1.923 (0.241)	1.830 (0.231)	1.740 (0.235)	1.619 (0.237)	1.528 (0.238)	0.997 (0.145)	2.077 (0.994)	1.027 (0.282)	36.78	90	882	49.39
400	10	MLE	1.895 (0.077)	1.794 (0.078)	1.697 (0.075)	1.597 (0.074)	1.495 (0.073)	0.990 (0.046)	2.048 (0.345)	1.011 (0.095)	32.11	100	995	–
400	10	2S-PML	1.884 (0.066)	1.774 (0.069)	1.662 (0.065)	1.548 (0.062)	1.425 (0.057)	1.040 (0.039)	0.805 (0.251)	0.671 (0.068)	1.40	100	1000	–
400	10	NPL	1.894 (0.075)	1.788 (0.077)	1.688 (0.069)	1.581 (0.071)	1.478 (0.073)	1.010 (0.041)	1.812 (0.213)	0.946 (0.061)	52.39	58	580	72.14
400	10	NPL-Λ	1.896 (0.077)	1.795 (0.079)	1.697 (0.076)	1.597 (0.074)	1.495 (0.073)	0.991 (0.044)	2.039 (0.330)	1.008 (0.091)	24.31	100	1000	33.46
400	20	MLE	1.903 (0.056)	1.801 (0.050)	1.701 (0.050)	1.600 (0.049)	1.502 (0.050)	0.996 (0.028)	2.020 (0.241)	1.005 (0.067)	29.74	100	999	–
400	20	2S-PML	1.902 (0.052)	1.795 (0.046)	1.684 (0.042)	1.572 (0.042)	1.459 (0.043)	1.027 (0.025)	1.210 (0.198)	0.785 (0.052)	1.54	100	1000	–
400	20	NPL	1.909 (0.055)	1.805 (0.048)	1.704 (0.050)	1.600 (0.050)	1.498 (0.049)	1.006 (0.028)	1.879 (0.169)	0.969 (0.051)	55.27	67	664	71.54
400	20	NPL-Λ	1.903 (0.055)	1.801 (0.050)	1.701 (0.049)	1.600 (0.048)	1.501 (0.050)	0.996 (0.029)	2.014 (0.250)	1.004 (0.069)	23.75	100	1000	31.50

Table 4: Monte Carlo Results for Case 4.

M	T	Estimator	Estimates								CPU Time (in sec.)	Data Sets Converged	Runs Converged	Avg. NPL(-Λ) Iter.
			$\theta_{FC,1}$	$\theta_{FC,2}$	$\theta_{FC,3}$	$\theta_{FC,4}$	$\theta_{FC,5}$	θ_{EC}	θ_{RN}	θ_{RS}				
		Truth	1.9	1.8	1.7	1.6	1.5	1	4	2	–	–	–	–
400	1	MLE	1.923 (0.267)	1.830 (0.265)	1.723 (0.252)	1.613 (0.245)	1.508 (0.246)	1.023 (0.140)	3.898 (0.680)	1.974 (0.246)	273.88	100	582	–
400	1	2S-PML	1.681 (0.255)	1.595 (0.241)	1.474 (0.241)	1.319 (0.227)	1.073 (0.208)	1.369 (0.144)	0.624 (0.393)	0.759 (0.150)	1.71	100	1000	–
400	1	NPL	1.997 (0.115)	1.891 (0.175)	1.747 (0.230)	1.676 (0.129)	1.389 (0.134)	1.481 (0.069)	1.958 (0.142)	1.340 (0.013)	103.77	2	20	99.47
400	1	NPL-Λ	1.963 (0.273)	1.863 (0.272)	1.759 (0.255)	1.631 (0.258)	1.506 (0.263)	1.056 (0.147)	3.680 (0.739)	1.907 (0.269)	74.22	84	840	69.67
400	10	MLE	1.897 (0.084)	1.797 (0.084)	1.697 (0.082)	1.594 (0.085)	1.496 (0.095)	0.993 (0.045)	4.015 (0.216)	2.004 (0.086)	149.65	100	812	–
400	10	2S-PML	1.934 (0.090)	1.824 (0.085)	1.703 (0.079)	1.556 (0.079)	1.338 (0.085)	1.123 (0.049)	2.297 (0.330)	1.409 (0.117)	1.59	100	1000	–
400	10	NPL	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	102.69	0	0	100
400	10	NPL-Λ	1.900 (0.079)	1.801 (0.081)	1.700 (0.077)	1.600 (0.080)	1.500 (0.091)	0.991 (0.052)	4.023 (0.255)	2.007 (0.098)	78.12	63	630	77.28
400	20	MLE	1.908 (0.057)	1.806 (0.056)	1.707 (0.053)	1.607 (0.055)	1.514 (0.059)	0.991 (0.031)	4.046 (0.137)	2.017 (0.054)	121.71	100	871	–
400	20	2S-PML	1.946 (0.066)	1.840 (0.062)	1.722 (0.059)	1.593 (0.059)	1.413 (0.059)	1.070 (0.039)	2.931 (0.224)	1.635 (0.079)	1.67	100	1000	–
400	20	NPL	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	107.07	0	0	100
400	20	NPL-Λ	1.905 (0.063)	1.804 (0.062)	1.706 (0.058)	1.607 (0.058)	1.517 (0.063)	0.988 (0.038)	4.077 (0.173)	2.027 (0.065)	84.30	53	530	79.30

Table 5: Monte Carlo Results for Case 5.

$\|S\|$	Estimator	Estimates								CPU Time (in sec.)	Data Sets Converged	Runs Converged	Avg. NPL(-Λ) Iter.
		$\theta_{FC,1}$	$\theta_{FC,2}$	$\theta_{FC,3}$	$\theta_{FC,4}$	$\theta_{FC,5}$	θ_{EC}	θ_{RN}	θ_{RS}				
	Truth	1.9	1.8	1.7	1.6	1.5	1	2	1	–	–	–	–
10	MLE	1.882 (0.092)	1.780 (0.087)	1.677 (0.079)	1.584 (0.084)	1.472 (0.068)	0.999 (0.046)	2.031 (0.201)	1.004 (0.048)	231.52	50	240	–
10	2S-PML	1.884 (0.102)	1.792 (0.088)	1.679 (0.082)	1.583 (0.087)	1.469 (0.068)	1.039 (0.048)	1.065 (0.222)	0.755 (0.053)	7.44	50	250	–
10	NPL	1.919 (0.092)	1.810 (0.089)	1.699 (0.068)	1.606 (0.079)	1.485 (0.071)	1.011 (0.050)	1.851 (0.136)	1.966 (0.036)	552.85	23	76	89
10	NPL-Λ	1.884 (0.095)	1.781 (0.089)	1.678 (0.081)	1.584 (0.085)	1.472 (0.070)	0.997 (0.049)	2.032 (0.211)	1.005 (0.051)	289.39	50	241	43

Table 6: Monte Carlo Results for Case 6.

$\|S\|$	Estimator	Estimates								CPU Time (in sec.)	Data Sets Converged	Runs Converged	Avg. NPL(-Λ) Iter.
		$\theta_{FC,1}$	$\theta_{FC,2}$	$\theta_{FC,3}$	$\theta_{FC,4}$	$\theta_{FC,5}$	θ_{EC}	θ_{RN}	θ_{RS}				
	Truth	1.9	1.8	1.7	1.6	1.5	1	2	1	–	–	–	–
15	MLE	1.897 (0.098)	1.800 (0.107)	1.694 (0.087)	1.597 (0.093)	1.492 (0.090)	0.983 (0.059)	2.040 (0.311)	1.011 (0.069)	762.78	50	222	–
15	2S-PML	1.792 (0.119)	1.705 (0.123)	1.595 (0.119)	1.506 (0.114)	1.394 (0.114)	1.046 (0.059)	0.766 (0.220)	0.664 (0.053)	31.45	50	242	–
15	NPL	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	N/A (N/A)	1560.08	0	0	100
15	NPL-Λ	1.922 (0.000)	1.821 (0.000)	1.671 (0.000)	1.611 (0.000)	1.531 (0.000)	1.012 (0.000)	1.992 (0.000)	1.007 (0.000)	1658.08	1	3	99.7

A Proof of Proposition 1

We first consider the constraint Jacobian matrix. For any Bellman Optimality constraint gradient row, there are at most $2^N(2\delta_s + 1) + |\boldsymbol{\theta}| + 2N$ non-zero elements. The first term comes from the derivative with respect to $V_i(\boldsymbol{x'})$ for the same player i, the second term comes from the derivative with respect to the structural parameters, and the third term comes from the derivative with respect to choice probabilities of all players across the binary action space at the current state $\boldsymbol{x}$. This applies to $N \cdot |\mathcal{S}| \cdot 2^N$ Bellman Optimality constraints. Similarly, inspecting the Bayes–Nash Equilibrium constraints leads, by the same derivation, to at most $|2|^N (2\delta_s + 1) + |\boldsymbol{\theta}| + 2N$ non-zero elements for each constraint gradient row. This refers nontrivially to each of $N \cdot |\mathcal{S}| \cdot 2^N$ rows. Finally, we add at most 2 non-zero elements per row for each of the remaining $N \cdot |\mathcal{S}| \cdot 2^N$ Bayes–Nash Equilibrium constraints, forcing the choice probabilities to sum to 1. This corresponds to the probabilities of each player's actions at the current state $\boldsymbol{x}$.

Summing up these terms yields the numerator as the number of non-zero elements in the constraint Jacobian matrix. The denominator comes from there being $(2+1) \cdot N \cdot |\mathcal{S}| \cdot 2^N$ constraints and $(2+1) \cdot N \cdot |\mathcal{S}| \cdot 2^N + |\boldsymbol{\theta}|$ variables. This leads to the following upper bound:

$$\begin{aligned} S_J \leq & \frac{N \cdot |\mathcal{S}| \cdot 2^N \cdot \left(2 \cdot \left(2^N (2\delta_s + 1) + |\boldsymbol{\theta}| + 2N\right) + 2\right)}{((2+1) \cdot N \cdot |\mathcal{S}| \cdot 2^N) \cdot ((2+1) \cdot N \cdot |\mathcal{S}| \cdot 2^N + |\boldsymbol{\theta}|)} \\ = & \frac{2}{3} \left(\frac{2^N (2\delta_s + 1) + |\boldsymbol{\theta}| + 2N + 1}{3N \cdot |\mathcal{S}| \cdot 2^N + |\boldsymbol{\theta}|} \right) \\ \leq & \frac{2}{9} \left(\frac{2\delta_s + 1}{N \cdot |\mathcal{S}|} + \frac{|\boldsymbol{\theta}| + 1}{N \cdot |\mathcal{S}| \cdot 2^N} + \frac{1}{|\mathcal{S}| \cdot 2^{N-1}} \right) \end{aligned} \tag{19}$$

The derivation for S_H is similar. We can sum over five terms to construct the numerator of the upper bound. The first corresponds to the $\partial^2 \boldsymbol{V}$ derivative terms, the second corresponds to the $\partial^2 \boldsymbol{P}$ derivative terms, the third corresponds to the $\partial \boldsymbol{V} \partial \boldsymbol{P}$ derivative terms, the fourth corresponds to both the $\partial \boldsymbol{V} \partial \boldsymbol{\theta}$ and $\partial \boldsymbol{P} \partial \boldsymbol{\theta}$ derivative terms, and the fifth corresponds to the $\partial^2 \boldsymbol{\theta}$ derivative terms. The denominator corresponds to the square of the number of variables. Combining these numerator and denominator terms leads to the following expression:

$$\begin{aligned} S_H \leq & \frac{N \cdot |\mathcal{S}| \cdot 2^N \cdot \left(2^N \cdot (2\delta_s + 1)\right) + |\mathcal{S}| \cdot 2^N \cdot (2N)^2 + 4N^2 \cdot |\mathcal{S}| \cdot 2^N \cdot \left(2^N \cdot (2\delta_s + 1)\right) + 2 \cdot \left((2+1) \cdot N \cdot |\mathcal{S}| \cdot 2^N\right) \cdot |\boldsymbol{\theta}| + |\boldsymbol{\theta}|^2}{\left((2+1) \cdot N \cdot |\mathcal{S}| \cdot 2^N + |\boldsymbol{\theta}|\right)^2} \\ \leq & \frac{1}{9} \left(\frac{2\delta_s + 1}{N \cdot |\mathcal{S}|} + \frac{4}{|\mathcal{S}| \cdot 2^N} + \frac{4 \cdot (2\delta_s + 1)}{|\mathcal{S}|} + \frac{6 \cdot |\boldsymbol{\theta}|}{N \cdot |\mathcal{S}| \cdot 2^N} + \left(\frac{|\boldsymbol{\theta}|}{N \cdot |\mathcal{S}| \cdot 2^N} \right)^2 \right) \end{aligned} \tag{20}$$

Proposition 1b and 1c follow immediately.

B Sparsity Information for the Constraint Jacobian and Hessian Matrices

Table 7: Upper Bounds on the Density of Constraint Matrices Varying $|\mathcal{S}|$ and N, with $\delta_s = 1$.

$\|\mathcal{S}\|$	N	$\boldsymbol{\theta}$	Constrained Jacobian		Hessian	
			Non-zero elements	Density S_J	Non-zero elements	Density S_H
5	5	8	187200	3.24%	1676900	28.9%
6	5	8	224640	2.70%	2012260	24.1%
7	5	8	262080	2.31%	2347620	20.7%
8	5	8	299520	2.03%	2682980	18.2%
9	5	8	336960	1.80%	3018340	16.1%
10	5	8	374400	1.62%	3353700	14.5%
11	5	8	411840	1.47%	3689060	13.2%
12	5	8	449280	1.35%	4024420	12.1%
13	5	8	486720	1.25%	4359780	11.2%
14	5	8	524160	1.16%	4695140	10.4%
15	5	8	561600	1.08%	5030500	9.68%
16	5	8	599040	1.01%	5365860	9.08%
17	5	8	636480	0.95%	5701220	8.55%
18	5	8	673920	0.90%	6036580	8.07%
19	5	8	711360	0.85%	6371940	7.65%
20	5	8	748800	0.81%	6707300	7.27%
5	3	6	9360	7.07%	44704	33.4%
5	4	7	42240	4.54%	283601	30.3%
5	5	8	187200	3.24%	1676900	28.9%
5	6	9	829440	2.50%	9388921	28.2%
5	7	10	3682560	2.04%	50337424	27.8%

References

[1] Arcidiacono, P. and R. A. Miller (2011): "CCP Estimation of Dynamic Discrete Choice Models with Unobserved Heterogeneity," *Econometrica*, 79, 1823–1868.

[2] Aguirregabiria, V. and P. Mira (2007): "Sequential Estimation of Dynamic Discrete Games," *Econometrica*, 75, 1–53.

[3] Aguirregabiria, V. and A. Nevo (2012): "Recent Developments in Empirical IO: Dynamic Demand and Dynamic Games," in *Advances in Economics and Econometrics: Theory and Applications. Tenth World Congress of the Econometric Society*. Forthcoming.

[4] Bajari, P., C. L. Benkard, and J. Levin (2007): "Estimating Dynamic Games of Imperfect Competition," *Econometrica*, 75, 1331–1370.

[5] Ciliberto, F. and E. Tamer (2009): "Market Structure and Multiple Equilibria in Airline Markets," *Econometrica*, 77, 1791–1828.

[6] Dubé, J.-P., J. T. Fox, and C.-L. Su (2012): "Improving the Numerical Performance of Static and Dynamic Aggregate Discrete Choice Random Coefficients Demand Estimation," *Econometrica*, 80, 2231-2267.

[7] Hotz, J. and R. A. Miller (1993): "Conditional Choice Probabilities and the Estimation of Dynamic Models," *Review of Economic Studies*, 60, 497–529.

[8] Kasahara, H. and K. Shimotsu (2012): "Sequential Estimation of Structural Models with a Fixed Point Constraint," *Econometrica*, forthcoming.

[9] Ortega, J. M. and W. C. Rheinboldt (1970): *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York and London.

[10] Pakes, A., M. Ostrovsky, and S. Berry (2007): "Simple Estimators for the Parameters of Discrete Dynamic Games, with Entry/Exit Examples," *RAND Journal of Economics*, 38, 373–399.

[11] Pakes, A., J. Porter, K. Ho, and J. Ishii (2011): "Moment Inequalities and Their Applications," Working paper, Harvard University.

[12] Pesendorfer, M. and P. Schmidt-Dengler (2008): "Asymptotic Least Squares Estimators for Dynamic Games," *Review of Economic Studies*, 75, 901–928.

[13] Pesendorfer, M. and P. Schmidt-Dengler (2010): "Sequential Estimation of Dynamic Discrete Games: A Comment," *Econometrica*, 78, 833–842.

[14] Su, C.-L. (2012): "Estimating Discrete-Choice Games of Incomplete Information: A Simple Static Example," Working paper, University of Chicago.

[15] Su, C.-L. and K. L. Judd (2012): "Constrained Optimization Approaches to Estimation of Structural Models," *Econometrica*, 80, 2213–2230.

[16] Tamer, E. (2003): "Incomplete Simultaneous Discrete Response with Multiple Equilibria," *Review of Economic Studies,* 70, 147–165.