Improving the Numerical Performance of BLP Static and Dynamic Discrete Choice Random Coefficients Demand Estimation[*]

Jean-Pierre Dubé

The University of Chicago Booth School of Business

University of Chicago

Jeremy T. Fox

Department of Economics

University of Chicago and

NBER

Che-Lin Su

The University of Chicago Booth School of Business

University of Chicago

February 2009

Abstract

The widely-used estimator of Berry, Levinsohn and Pakes (1995) produces consistent, instrumental-variables estimates of consumer preferences from a discrete-choice demand model with random coefficients, market-level demand shocks and endogenous regressors (prices). The nested-fixed-point algorithm typically used for estimation is computationally intensive, largely because a system of market-share equations must be repeatedly numerically inverted. We provide numerical theory results that characterize the properties of typical nested-fixed-point implementations. We use these results to discuss several problems with typical computational implementations and, in particular, cases which can lead to incorrect parameter estimates. As a solution, we recast estimation as a mathematical program with equilibrium constraints (MPEC). In some instances, MPEC is faster. It also avoids the numerical issues associated with nested inner loops. Several Monte Carlo and real-data experiments support our numerical concerns about NFP and the advantages of MPEC. We also discuss estimating static BLP using maximum likelihood instead of GMM. Finally, we show that MPEC is particularly attractive for forward-looking demand models where both Bellman's equation and the market share equations must be repeatedly solved.

[*]We thank Daniel Ackerberg, John Birge, Amit Gandhi, Lars Hansen, Panle Jia, Kyoo il Kim, Kenneth Judd, Sven Leyffer, Denis Nekipelov, Aviv Nevo, Jorge Nocedal, Hugo Salgado, Azeem Shaikh and Richard Waltz for helpful discussions and comments. We also thank workshop participants at CREST-INSEE / ENSAE, EARIE, the ESRC Econometrics Study Group Conference, the Econometric Society, the Federal Trade Commission, INFORMS, the International Industrial Organization Conference, Northwestern University, the Portuguese Competition Commission, the Stanford Institute for Theoretical Economics, the UK Competition Commission, the University of Chicago, and the University of Rochester. Dubé is grateful to the Kilts Center for Marketing and the Neubauer Faculty Fund for research support. Fox thanks the NSF, grant 0721036, the Olin Foundation, and the Stigler Center for financial support. Su is grateful for the financial support from the NSF (award no. SES-0631622) and the University of Chicago Booth School of Business. Our email addresses are jdube@chicagogsb.edu, fox@uchicago.edu and Che-Lin.Su@chicagogsb.edu.

1 Introduction

The discrete-choice class of demand models has become popular in the demand estimation literature due to the models' ability to accommodate rich substitution patterns between a potentially large array of products. The method-of-moments estimator developed in Berry, Levinsohn and Pakes (1995), hereafter BLP, made an important contribution to this literature by accommodating controls for the endogeneity of product characteristics (namely prices) without sacrificing the flexibility of these substitution patterns. BLP consider a random-coefficients, discrete-choice model with market-level demand shocks that correlate with prices. They construct moment conditions with which they can address price endogeneity using standard instrumental-variables methods. The approach has had a large impact: as of October 2008, BLP generated more than 1000 citations in Google Scholar and the approach has been used in many important empirical studies. However, the estimator is difficult to program and can take a long time to run on a desktop computer. More importantly, many current implementations of the estimator are sufficiently vulnerable to numerical inaccuracy that they may produce incorrect parameter estimates. We summarize some of these computational problems and propose an alternative procedure that is robust to these sources of numerical inaccuracy.

An important component of BLP's contribution consists of a computationally feasible approach to constructing the moment conditions. As in Berry (1994), the main idea is to invert the nonlinear system of market share equations. BLP and Berry suggest nesting this inversion step directly into the parameter search. For complex specifications such as random coefficients, this inversion step may not have an analytic inverse. BLP propose a contraction-mapping routine to solve this system of equations. This step nests an inner-loop contraction mapping into the parameter search. Following the publication of Nevo's (2000b) "A Practitioner's Guide" to implementing BLP, numerous studies have emerged using the BLP approach to estimating discrete choice demand systems with random coefficients.

Our first objective consists of exploring the numerical properties of BLP's contraction-mapping approach. We call this contraction-mapping approach the nested-fixed point, or NFP, approach. The GMM objective function can be called hundreds of times during a numerical optimization over structural parameters; each call to the objective function requires a call to the nested inner loop. Therefore, it may be tempting to use a less stringent stopping criterion for the inner loop in order to speed up estimation. We use numerical theory to show how a less stringent stopping criterion for the inner loop may cause two types of errors in the parameter estimates. First, in contrast with closed-form derivatives, numerical derivatives for the optimization routine are dominated by inner-loop numerical error. This error may cause the routine to search in the wrong direction and to report incorrect parameter estimates. Second, even methods with closed-form derivatives may falsely stop at a point which is numerically close to a local minimum of the objective function, but is in fact not a local minimum. To illustrate these problems, we construct examples based on real data and data generated from a known model. In these examples, the errors in the parameter estimates from using loose tolerances for the NFP inner loop are large.

Our second objective is to propose a new computational method for implementing the BLP estimator that eliminates the inner loop entirely and, thus, eliminates the potential for numerical inaccuracy discussed above. Following Su and Judd (2007), we recast the BLP problem as a Mathematical Program with Equilibrium Constraints (MPEC). The MPEC method minimizes the GMM objective function subject to a system of nonlinear constraints requiring that the predicted shares from the model equal the observed shares in the data. The minimization of an objective function subject to nonlinear constraints is a standard exercise in nonlinear programming. We prefer the MPEC approach over the existing NFP approach for four reasons. First, there is no numerical error from nested calls. This aspect eliminates the potential for the minimization routine to converge to a point that is not a local minimum of the true GMM objective function, subject to the constraints within a feasibility tolerance, usually set by default to be 10^{-6}. Second, by eliminating the nested calls, the procedure may be faster than the contraction mapping method proposed by BLP. Third, the MPEC algorithm allows the user to relegate all the numerical operations to a single outer loop that can consist of a call to a state-of-the-art optimization package. Fourth, the MPEC approach applies to more general demand models, where no current theoretical result shows a contraction mapping exists.

From a statistical perspective, the MPEC algorithm generates the same estimator as the correctly-implemented NFP approach. Conceptually, MPEC is just an alternative algorithm to programming BLP's original estimator. Therefore, the theoretical results on consistency and statistical inference in Berry, Linton and Pakes (2004) apply equally to the contraction mapping and MPEC methods. The related work on identification of the BLP model by Berry and Haile (2008) and Fox and Gandhi (2008) is also agnostic to the actual computational method used in estimation. Our purpose is therefore not to criticize rich structural methods for being too complicated. On the contrary, we view structural methods as a valuable tool in empirical work. Our purpose is to to discuss the potential numerical problems that can arise with a complex structural demand model and to offer a practical approach that avoids these problems.

To illustrate the empirical relevance of the numerical theory, we provide three sets of numerical examples for static BLP. First, we document cases where a loose tolerance for the contraction mapping in the traditional approach leads to incorrect parameter estimates and/or the failure for the optimization routine's search over parameters to report convergence. The errors in the estimated own-price elasticities are also found to be large in both real data and in simulated data, and with optimization routines that use closed form and numerical derivatives. Further, in one example we show that the parameter estimates will always converge to the same incorrect point (i.e. a point that is not even a local minimum), so that a researcher using multiple starting values will not be able to diagnose the presence of numerical errors in the parameters. We also use this example to show that the alternative Nelder-Meade or simplex algorithm also always converges to the wrong solution.

In a second set of numerical examples, we explore the relative speed of the traditional NFP approach (correctly implemented) and MPEC. BLP is an empirical method and the speeds of various implementations are sensitive to the properties of the data being used. We use numerical theory to show that the speed of the NFP contraction mapping is bounded above by a function that is linear in what is known as a Lipschitz constant. We derive an analytic expression for the Lipschitz constant that depends on the data and the parameter values from the demand model. By playing around with features of the data and the parameter values that increase the magnitude of the Lipschitz constant, we conjecture that we can make the NFP inner loop slower. For example, in our sampling experiments we find that decreasing the outside good share (by raising the utility intercept) decreases the speed of the NFP estimator. We also find that the speed of NFP in these sampling experiments decreases dramatically when the Lipschitz constant gets closer to its theoretical upper bound, 1. By contrast, MPEC's speed is relatively invariant to the value of the Lipschitz constant. We can construct examples where MPEC is several times faster than NFP. A concern some might have with MPEC is that its speed scales poorly with the number of markets in the data. We show numerically that this need not be true. In contrast, in our Monte Carlo results, we find that NFP scales poorly when the number of statistical observations increases.

An advantage of the contraction mapping is that is is guaranteed to converge from any starting value. Alternative methods for the inner loop inversion, that are not guaranteed to converge from any starting point, may be infeasible in practice. Because the NFP inner loop may be called hundreds or thousands of times during the parameter search, it would be impractical to check convergence on each call. Gandhi (2008) derives existence and uniqueness theorems for demand models where the contraction mapping theorem is unlikely to apply. We expect that MPEC will work in cases where Gandhi's existence theorem (or a generalization of that theorem) applies but his uniqueness theorem does not.

The numerical concerns we raise with the inner loop are magnified as new literatures generalize BLP demand estimators to economically richer models of consumer behavior. As an extension, we consider the discrete choice demand system with forward-looking consumers. We look at cases, such as durable or semi-durable goods markets, where consumers can alter the timing of their purchase decision based on expectations about future products and prices (Melnikov 2002, Carranza 2006, Hendel and

Nevo 2007, Gowrisankaran and Rysman 2007, Nair 2007 and Dube, Hitsch and Chintagunta 2008). Estimating demand using the NFP algorithm now involves three numerical loops, adding yet another source of numerical error: the outer optimization routine, the inner inversion of the market share equations, and the inner evaluation of the consumers' value functions (the Bellman equation) for each of the many heterogeneous consumer types. The consumer's dynamic programming problem is typically solved with a contraction mapping with the same slow rate of convergence as the BLP market share inversion. Furthermore, Gowrisankaran and Rysman point out that the recursion proposed by BLP may no longer be a contraction mapping for some specifications of dynamic discrete choice models. Hence, the market share inversion is not guaranteed to converge to a solution, which, in turn, implies that the outer optimization routines may not produce the GMM objective function value.

We show that MPEC extends naturally to the case with forward-looking consumers. We optimize the statistical objective function subject to the constraints that Bellman's equation is satisfied at all consumer states and that the market share equations hold. Our approach eliminates both inner loops, thereby reducing these two sources of numerical error. We produce benchmark results that show that MPEC can be faster than NFP under realistic data generating processes. We expect the relative performance of MPEC to improve for even more complex dynamic demand models that nest even more calls to inner loops (Lee 2008 and Schiraldi 2008).

Our work herein focuses primarily on numerical algorithms for computing the BLP GMM estimator.[1] A recent stream of literature has also studied the statistical efficiency of the GMM estimator itself and has explored likelihood-based approaches that use additional structure on the joint distribution of demand and supply (c.f. Yang et al 2003 and Jiang et al 2008). Using aggregate data, Jiang et al (2008) propose an alternative Bayesian approach to BLP using Markov Chain Monte Carlo methods and report much better small sample properties than the GMM estimates. In general, likelihood-based approaches still require the numerical inversion of the system of market shares,[2] subjecting them to this additional source of numerical error that MPEC avoids. In our extensions, we discuss how one could use MPEC to estimate demand parameters by maximizing the joint likelihood of shares and prices.

Our assessment of BLP's numerical properties is also broadly related to the recent work by Knittel and Metaxoglou (2008). They explore the potential multiple local minima property of the BLP objective function. Our goal is to study the numerical accuracy and speed of finding an apparent

[1]Our work on the BLP estimator operates in parallel to Petrin and Train's (2008) control-function approach, which avoids the inner loop by utilizing additional non-primitive assumptions relating equilibrium prices to the demand shocks. Our proposed MPEC approach also avoids the need for numerical inversion while remaining agnostic about the underlying process (involving the supply side) generating prices and demand shocks – the approach is statistically the same as BLP's original formulation.

[2]The transformation-of-variables theorem involves the evaluation of a Jacobian that requires computing the demand shocks numerically.

global minimum. However, with using multiple starting points, our fakedata Monte Carlo experiments routinely find that BLP can recover the true structural parameters using data generated by the model. In a short digression, we examine the same cereal dataset used by Knittel and Metaxoglou. Using a state-of-the-art solver, we choose 50 starting values and find the same local minimum each time, which is the global minimum found by Knittel and Metaxoglou.

The remainder of the paper is organized as follows. We discuss BLP's model in Section 2 and their statistical estimator in Section 3. Section 4 provides a theoretical analysis of the numerical properties of BLP's traditional NFP algorithm. Section 5 presents examples of practices leading to errors in the estimates of parameters. Section 6 presents our alternative MPEC algorithm. Section 7 provides Monte Carlo evidence about the relative performances of the NFP and MPEC algorithms. The last two sections discuss extensions where MPEC's advantages over NFP are magnified. First, we discuss maximum likelihood estimation, where the need to compute the Jacobian makes MPEC especially useful. Second, we discuss the burgeoning literature on dynamic consumer demand.

2 The Demand Model

In this section, we present the standard random-coefficients, discrete-choice demand model. In most empirical applications, the researcher has access to market shares for each of the available products, but does not have consumer-level information.[3] The usual modeling solution is to build a system of market shares that is consistent with an underlying population of consumers independently making discrete choices among the various products. The population is in most instances assumed to consist of a continuum of consumers with known mass.

Formally, each market $t = 1, ..., T$ has a mass M_t of consumers who each choose one of the $j = 1, ..., J$ products available, or opt not to purchase. Each product j is described by its characteristics $(x_{j,t}, \xi_{j,t}, p_{j,t})$. The vector $x_{j,t}$ consists of K product attributes; let x_t be the collection of the vectors $x_{j,t}$ for all J products, The scalar $\xi_{j,t}$ is a vertical characteristic that is observed by the consumers and firms, but is unobserved by the researcher. $\xi_{j,t}$ can be seen as a market- and product-specific demand shock that is common across all consumers in the market. For each market, we define the J-vector $\xi_t = (\xi_{1,t}, ..., \xi_{J,t})'$. Finally, we denote the price of product j by $p_{j,t}$ and the vector of the J prices by p_t,

Consumer i in market t obtains the following indirect utility from purchasing product j

$$u_{i,j,t} = \beta_i^0 + x'_{j,t}\beta_i^x - \beta_i^p p_{j,t} + \xi_{j,t} + \varepsilon_{i,j,t}. \tag{1}$$

[3]See Berry, Levinsohn and Pakes (2004) as well as Petrin (2002) for methods incorporating consumer-level data.

The utility of the outside good, or "no-purchase" option, is $u_{i,0,t} = \varepsilon_{i,0,t}$. The consumer i's preferences include of the parameter vector β_i^x, the tastes for the K characteristics, and the parameter β_i^p, the marginal utility of income, $i's$ "price sensitivity". Finally, $\varepsilon_{i,j,t}$ is an additional idiosyncratic product-specific shock. Let $\varepsilon_{i,t}$ be the vector of all $J+1$ product-specific shocks for consumer i.

Each consumer is assumed to pick the product j that gives her the highest utility. If tastes, $\beta_i = \left(\beta_i^0, \beta_i^x, \beta_i^p\right)$ and $\varepsilon_{i,t}$, are independent draws from the distributions $F_\beta\left(\beta;\theta\right)$, with unknown parameters θ, and $F_\epsilon(\epsilon)$, respectively, the market share of product j is

$$s_j\left(x_t, p_t, \xi_t; \theta\right) = \int_{\left\{\beta_i, \varepsilon_i \mid u_{i,j} \geq u_{i,j'}\ \forall j' \neq j\right\}} dF_\beta\left(\beta;\theta\right) dF_\epsilon\left(\varepsilon\right).$$

To simplify aggregate demand estimation, we follow the convention in the literature and assume ε is distributed type I extreme value, enabling one to integrate it out analytically,

$$s_j\left(x_t, p_t, \xi_t; \theta\right) = \int_\beta \frac{\exp\left(\beta^0 + x_{j,t}'\beta^x - \beta^p p_{j,t} + \xi_{j,t}\right)}{1 + \sum_{k=1}^J \exp\left(\beta^0 + x_{k,t}'\beta^x - \beta^p p_{k,t} + \xi_{k,t}\right)} dF_\beta\left(\beta;\theta\right). \tag{2}$$

This is the random coefficient logit model.

In BLP, the goal is to estimate the parameters θ characterizing the distribution of consumer random coefficients, $F_\beta\left(\beta;\theta\right)$. McFadden and Train (2000) prove that a flexible choice of the family $F_\beta\left(\beta;\theta\right)$ (combined with a polynomial in $x_{j,t}$ and $p_{j,t}$) allows the random coefficient logit model to approximate arbitrarily any vector of choice probabilities (market shares) originating from a random utility model with an observable linear index (meaning no $\xi_{j,t}$ term). Bajari, Fox, Kim and Ryan (2008) prove the nonparametric identification (no finite-dimensional parameter θ) of $F_\beta\left(\beta\right)$ in the random coefficient logit model without aggregate demand shocks, using data on market shares and product characteristics. Berry and Haile (2008) prove the nonparametric identification of the distribution of utilities in each market (but not structural functional forms like the distribution of the parameters β in $x_{j,t}^{'}\beta$ or u in a nonparametric $u\left(x_{j,t}\right)$) in the entire BLP demand model, including allowing for aggregate shocks. Fox and Gandhi (2008) have an alternative identification proof for heterogeneity that does identify β in in $x_{j,t}^{'}\beta$ or u in a nonparametric $u\left(x_{j,t}\right)$, and which can be adapted for market level demand shocks in the same way as Berry and Haile. However, in most applications, more structure is imposed on the family of distributions characterizing $F_\beta\left(\beta;\theta\right)$ through the choice of the family $F_\beta\left(\beta;\theta\right)$, with each family member indexed by the estimable finite vector of parameters θ. For example, BLP assume that $F_\beta\left(\beta;\theta\right)$ is the product of K independent normals, with $\theta = \left(\mu, \sigma\right)$, the vectors of means and standard deviations for each component of the K normals.

Typically, the integrals in (2) are evaluated by Monte Carlo simulation. The idea is to generate

ns draws of (β, α) from the distribution $F_\beta(\beta; \theta)$ and to simulate the integrals as

$$\hat{s}_j(x_t, p_t, \xi_t; \theta) = \frac{1}{ns} \sum_{r=1}^{ns} \frac{\exp\left(\beta^{0,r} + x'_{j,t}\beta^{x,r} - \beta^{p,r} p_{j,t} + \xi_{j,t}\right)}{1 + \sum_{k=1}^{J} \exp\left(\beta^{0,r} + x'_{k,t}\beta^{x,r} - \beta^{p,r} p_{k,t} + \xi_{k,t}\right)}. \tag{3}$$

In principle, many other numerical methods could be used to evaluate the market-share integrals (Judd 1998, Chapters 7–9).

While a discrete-choice model with heterogeneous preferences dates back at least to Hausman and Wise (1978), the inclusion of the aggregate demand shock, $\xi_{j,t}$, was introduced by Berry (1994) and BLP. The demand shock $\xi_{j,t}$ is the natural generalization of demand shocks in the textbook linear supply and demand model. We can see in (2) that without the shock, $\xi_{j,t} = 0 \, \forall \, j$, market shares are deterministic functions of the x's and p's. In consumer level data applications, the econometric uncertainty is typically assumed to arise from randomness in consumer tastes, ε. This randomness washes out in a model that aggregates over a sufficiently large number of consumer choices (here a continuum). A model without market-level demand shocks will not be able to fit data on market shares across markets, as the model does not give full support to the data. In the next section, we discuss estimation challenges that arise when $\xi_{j,t}$ is included in the model.

3 The BLP GMM Estimator and the MPEC Alternative

We now briefly discuss the GMM estimator typically used to estimate the vector of structural parameters, θ. Like the textbook supply and demand model, the demand shocks, $\xi_{j,t}$, force the researcher to deal with the potential simultaneous determination of price and quantity. To the extent that firms observe $\xi_{j,t}$ and condition on it when they set their prices, the resulting correlation between $p_{j,t}$ and $\xi_{j,t}$ will complicate the estimation of (2). This correlation introduces endogeneity bias.

BLP address the endogeneity of price in demand with a vector of D instrumental variables, $z_{j,t}$. They propose a GMM estimator based on the D moment conditions, $E[\xi_{j,t} \mid z_{j,t}] = 0$. These instruments can be product-specific cost shifters, although frequently other instruments are used because of data availability. Typically the K non-price characteristics in $x_{j,t}$ are also assumed to be mean independent of $\xi_{j,t}$ and hence to be valid instruments, although this is not a requirement of the statistical theory. The estimator does not impose a parametric distributional assumption on the demand shocks $\xi_{j,t}$, besides the identifying assumption $E[\xi_{j,t} \mid z_{j,t}] = 0$. To summarize, the data the researcher has are $\left\{(x_{j,t}, p_{j,t}, s_{j,t}, z_{j,t})_{j=1}^{J}\right\}_{t=1}^{T}$ for J products in each of T markets.

To form the empirical analog of $E[\xi_{j,t} \mid z_{j,t}]$ or the often implemented moments $E[\xi_{j,t} z_{j,t}]$, the researcher needs to find the implied values of the demand shocks, $\xi_{j,t}$, corresponding to a guess for

θ. The system of market shares, (2), defines a mapping between the vector of demand shocks and the market shares : $S_t = s\left(x_t, p_t, \xi_t; \theta\right)$, or $S_t = s\left(\xi_t; \theta\right)$ for short. Berry (1994) and Gandhi (2008) prove that s has an inverse, s^{-1}, such that any observed vector of shares can be explained by a unique vector $\xi_t\left(\theta\right) = s^{-1}\left(S_t; \theta\right)$. An individual demand shock $\xi_{j,t}$ given by this vector is $s^{-1}_{j,t}\left(S_t; \theta\right)$. For the random coefficients logit specification in BLP, we can compute ξ_t using the contraction mapping proposed in BLP. We discuss the properties of the contraction mapping in the next section.

A GMM estimator can now be constructed by using the empirical analog of the D moment conditions,

$$g\left(\xi\left(\theta\right)\right) = \frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{J}\xi_{j,t}\left(\theta\right)z_{j,t} = \frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{J}s^{-1}_{j,t}\left(S_t; \theta\right)z_{j,t},$$

Let S be the vector of market shares in all markets and let $s^{-1}\left(S; \theta\right)$ be the implied demand shocks in all markets. For some weighting matrix, W, we define the GMM estimator as the vector, θ^{GMM}, that solves the problem

$$\min_{\theta} g\left(s^{-1}\left(S; \theta\right)\right)' W g\left(s^{-1}\left(S; \theta\right)\right). \tag{4}$$

The statistical efficiency of the GMM estimator can be improved by using other, nonlinear functions of $z_{j,t}$ in the vector of moments, finding more instruments, using an optimal weighting matrix in a second step, or using an efficient one-step method such as continuously-updated GMM or empirical likelihood. However, as we show in the following sections, the numerical precision of the algorithms used to compute $Q\left(\theta\right)$ may be equally or more important from a practical perspective than matters of statistical efficiency.

To avoid any suspense, our alternative MPEC approach is the problem

$$\begin{array}{ll} \min\limits_{\theta,\xi} & g\left(\xi\right)' W g\left(\xi\right) \\ \text{subject to} & s\left(\xi; \theta\right) = S \end{array},$$

where $g\left(\xi\right) = \frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{J}\xi_{j,t}z_{j,t}$. In our constrained minimization formulation, we minimize over both the structural parameters θ and the demand shocks ξ. Below we will prove that any local minimum to (4) is a local minimum to the MPEC problem, if $s^{-1}\left(S; \theta\right)$ exists and is unique. However, first we will explore some of the problems with NFP.

4 A Theoretical Analysis of the NFP Algorithm

In this section, we theoretically analyze the numerical properties of BLP's method. The GMM estimator described in section 3 consists of an outer loop to minimize the objective function, $Q\left(\theta\right)$, and an inner loop to evaluate this function. Each evaluation of the GMM objective function, $Q(\theta)$, nests a call to a contraction mapping. We call the complete GMM estimator that nests the inner loop the nested fixed point, or NFP, method. Each time the minimization routine calls $Q(\theta)$, the contraction mapping is called T times, once for each market t. If the researcher does not calculate the first and second derivatives of $Q(\theta)$ analytically, many local minimization routines approximate the gradient and Hessian using finite differences. The use of numerical derivatives will require many additional calls to $Q(\theta)$ and hence the contraction mapping, proportionately to the dimension of θ.

From a practical perspective, the speed of optimization is determined almost entirely by the number of calls to the contraction mapping and the computation time associated with each run of the inner loop. For these reasons, some practical applications have used a fairly loose convergence criteria to improve speed. In the subsections below, we first provide formal results on the speed of convergence of the inner loop.[4] We then show formally how numerical error from the inner loop can propagate into the outer loop, potentially leading to incorrect parameter estimates. One goal of this section is to provide guidelines for researchers in their selection of convergence criteria for the numerical algorithms used to estimate θ. We also theoretically analyze the speed of the NFP algorithm and discuss when it is likely to be slow.

4.1 The Convergence Rate of the NFP Contraction Mapping

In this section, we derive the rate of convergence of the contraction mapping proposed by BLP to invert the demand system. Recall from section 3 that the evaluation of the GMM criterion, $Q(\theta)$, requires us to evaluate the inverse: $\xi_t(\theta) = s^{-1}(S_t; \theta)$. For a given θ, the inner loop of the NFP estimator solves the share equations for the demand shocks ξ by iterating the contraction mapping

$$\xi_t^{h+1} = \xi_t^h + \log S_t - \log s\left(\xi_t^h; \theta\right), \quad t = 1, \ldots, T, \tag{5}$$

until the successive iterates ξ_t^{h+1} and ξ_t^h are sufficiently close.[5] Formally, we choose a small number, for example 10^{-8} or 10^{-10}, for ϵ_{in} as the inner loop tolerance level and require ξ_t^{h+1} and ξ_t^h to satisfy the stopping rule

$$\left\|\xi_t^h - \xi_t^{h+1}\right\| \leq \epsilon_{\text{in}} \tag{6}$$

[4]Davis (2006) suggests another inner-loop method based on a nested optimization problem that might converge faster than BLP's contraction mapping. One could replace the contraction mapping with Newton's method or some other solver for a system of nonlinear equations. However, any inner loop approach has the potential to propagate numerical error into the objective function, and hence potentially the parameters.

[5]In our implementation of NFP, we iterate over $\exp(\xi)$ to speed up the computation because taking logarithms in MATLAB is slow. However, depending on the magnitude of ξ, the use of the exponentiated form $\exp(\xi)$ in a contraction mapping can lose 3 to 5 digits of accuracy in ξ, and as a result, introduce an additional source of numerical error. For example, if $\left|\xi_t^h\right| = -8$ and $\left|\exp\left(\xi_t^h\right) - \exp\left(\xi_t^{h+1}\right)\right| = 10^{-10}$, then $\left|\xi_t^h - \xi_t^{h+1}\right| = 2.98 \times 10^{-7}$.

for the iteration $h+1$ where we terminate the contracting mapping (5).[6] Let $\xi_t(\theta, \varepsilon_{\text{in}})$ denote the first ξ_t^{h+1} such that the stopping rule (6) is satisfied. The researcher then uses $\xi_t(\theta, \varepsilon_{\text{in}})$ to approximate $\xi_t(\theta)$.

Researchers often find it tempting to loosen the inner loop tolerance if the NFP contraction mapping is slow. Below, we derive formally the theoretical rate of convergence of the inner loop call to the contraction mapping in terms of the economic parameters of the BLP demand model. Numerical theory proves that the convergence of a contraction mapping is linear at best. Linearly convergent algorithms are typically considered to be slow compared to alternative methods, such as Newton's method, for solving nonlinear equations. The numerical performance of a contraction mapping is also sensitive to the stopping tolerance criteria ϵ_{in}. We now state the contraction-mapping theorem and discuss how to calculate the linear convergence rate for the inner-loop contraction mapping (5) of the BLP estimator.

Theorem 1. *Let $\mathcal{T}_\theta : R^n \to R^n$ be an iteration function and let $S_r = \left\{\xi \mid \left\|\xi - \xi^0\right\| < r\right\}$ be a ball of radius r around a given starting point $\xi^0 \in R^n$. Assume that $\mathcal{T}_\theta$ is a contraction mapping in S_r, meaning*

$$\xi, \tilde{\xi} \in S_r \Rightarrow \left\|\mathcal{T}_\theta(\xi) - \mathcal{T}_\theta\left(\tilde{\xi}\right)\right\| \leq L(\theta)\left\|\xi - \tilde{\xi}\right\|,$$

where $L(\theta) < 1$ is called a Lipschitz constant. Then if

$$\left\|\xi^0 - \mathcal{T}_\theta\left(\xi^0\right)\right\| \leq (1 - L(\theta))\, r,$$

the multidimensional equation $\xi = \mathcal{T}_\theta(\xi)$ has a unique solution ξ^* in the closure of S_r, $\bar{S}_r = \{\xi \mid \|\xi - \xi_0\| \leq r\}$. This solution can be obtained by the convergent iteration process $\xi^{h+1} = \mathcal{T}_\theta\left(\xi^h\right)$, for $h = 0, 1, \ldots$ *The error at the h^{th} iteration is bounded:*

$$\left\|\xi^h - \xi^*\right\| \leq \left\|\xi^h - \xi^{h-1}\right\| \frac{L(\theta)}{1 - L(\theta)} \leq \left\|\xi^1 - \xi^0\right\| \frac{L(\theta)^h}{1 - L(\theta)}.$$

The Lipschitz constant, $L(\theta)$, is a measure of the rate of convergence. At every iteration, the upper bound for the norm of the error is multiplied by a factor equal to $L(\theta)$. A proof of this theorem can be found in many textbooks, such as Dahlquist and Björck (2008). The following theorem shows how a Lipschitz constant for a mapping $\mathcal{T}_\theta(\xi)$ can be expressed in terms of $\nabla\mathcal{T}_\theta(\xi)$, the Jacobian of $\mathcal{T}_\theta(\xi)$. We then use the Lipschitz constant result to assess an upper bound for the performance of the BLP NFP estimator.

Theorem 2. *Let the function $\mathcal{T}_\theta(\xi) : R^n \to R^n$ be differentiable in a convex set $D \subset R^n$. Then*

[6] $\|(a_1, \ldots, a_b)\|$ is a distance measure, such as $\max(a_1, \ldots, a_b)$.

$L(\theta) = \max_{\xi \in D} \|\nabla \mathcal{T}_\theta(\xi)\|$ *is a Lipschitz constant for* $\mathcal{T}$.

The contraction mapping in the BLP estimator is

$$\mathcal{T}_\theta(\xi) = \xi + \log S - \log s(\xi; \theta).$$

We define a Lipschitz constant for the BLP contraction mapping $\mathcal{T}$ given structural parameters θ as

$$L(\theta) \quad = \max_{\xi \in D} \|\nabla \mathcal{T}_\theta(\xi)\| = \max_{\xi \in D} \|I - \nabla(\log s_j(\xi_t; \theta))\|,$$

where

$$\frac{\partial \log\left(s_j(\xi_t;\theta)\right)}{\partial \xi_{lt}} = \begin{cases} \dfrac{\sum_{r=1}^{ns}\left[\left(\frac{\exp\left(\beta^{0,r}+x'_{j,t}\beta^{x,r}-\beta^{p,r}p_{j,t}+\xi_{j,t}\right)}{1+\sum_{k=1}^{J}\exp\left(\beta^{0,r}+x'_{k,t}\beta^{x,r}-\beta^{p,r}p_{k,t}+\xi_{k,t}\right)}\right)-\left(\frac{\exp\left(\beta^{0,r}+x'_{j,t}\beta^{x,r}-\beta^{p,r}p_{j,t}+\xi_{j,t}\right)}{1+\sum_{k=1}^{J}\exp\left(\beta^{0,r}+x'_{k,t}\beta^{x,r}-\beta^{p,r}p_{k,t}+\xi_{k,t}\right)}\right)^2\right]}{\sum_{r=1}^{ns}\frac{\exp\left(\beta^{0,r}+x'_{j,t}\beta^{x,r}-\beta^{p,r}p_{j,t}+\xi_{j,t}\right)}{1+\sum_{k=1}^{J}\exp\left(\beta^{0,r}+x'_{k,t}\beta^{x,r}-\beta^{p,r}p_{k,t}+\xi_{k,t}\right)}.}, & \text{if} \quad j = l \\ \dfrac{-\sum_{r=1}^{ns}\left[\left(\frac{\exp\left(\beta^{0,r}+x'_{j,t}\beta^{x,r}-\beta^{p,r}p_{j,t}+\xi_{j,t}\right)}{1+\sum_{k=1}^{J}\exp\left(\beta^{0,r}+x'_{k,t}\beta^{x,r}-\beta^{p,r}p_{k,t}+\xi_{k,t}\right)}\right)\left(\frac{\exp\left(\beta^{0,r}+x'_{l,t}\beta^{x,r}-\beta^{p,r}p_{l,t}+\xi_{l,t}\right)}{1+\sum_{k=1}^{J}\exp\left(\beta^{0,r}+x'_{k,t}\beta^{x,r}-\beta^{p,r}p_{k,t}+\xi_{k,t}\right)}\right)\right]}{\sum_{r=1}^{ns}\frac{\exp\left(\beta^{0,r}+x'_{j,t}\beta^{x,r}-\beta^{p,r}p_{j,t}+\xi_{j,t}\right)}{1+\sum_{k=1}^{J}\exp\left(\beta^{0,r}+x'_{k,t}\beta^{x,r}-\beta^{p,r}p_{k,t}+\xi_{k,t}\right)}}, & \text{if} \quad j \neq l. \end{cases}.$$

For a given vector of structural parameters θ, $L(\theta)$ is the Lipschitz constant for the NFP inner loop. It is difficult to get precise intuition for the Lipschitz constant as it is the norm of a matrix. But, roughly speaking, the Lipschitz constant is related to the matrix of own and cross demand elasticities for the demand shocks, ξ, as the jth element along the main diagonal is $\frac{\partial s_{j,t}}{\partial \xi_{j,t}} \frac{1}{s_{j,t}}$. In section 7.3 below, we use the Lipschitz constant to distinguish between simulated datasets where we expect the contraction mapping to perform relatively slowly or fast.

4.2 Determining the Stopping Criteria for the Outer Loop in NFP

This subsection provides guidance on how to select the outer-loop tolerance to ensure the outer loop will converge for a given inner-loop tolerance. In particular, we show how numerical error from the inner loop can propagate into the outer loop. We characterize the corresponding numerical inaccuracy in the criterion function, $Q(\theta)$, and its gradient. This analysis then informs the decision of what tolerance to use for the outer-optimization loop to ensure that the optimization routine is able to report convergence. This subsection focuses on ensuring the outer loop can actually converge given the numerical inaccuracy of the inner loop. In a later section, we show how this numerical inaccuracy in $Q(\theta)$ and its gradient can generate numerical inaccuracy in the parameter estimates of θ. In some instances, this inaccuracy could imply that the reported estimates are not a true local minimum of $Q(\theta)$.

Recall that the outer loop of the BLP estimator consists of minimizing the GMM objective function (4). The convergence of this outer loop depends on the choice of an outer-loop tolerance level, denoted by ϵ_{out}. In theory, ϵ_{out} should be set to a small number, such as 10^{-5} or 10^{-6} . In practice, we have found cases in the BLP literature where 10^{-2} was used, possibly to offset the slow performance or non-convergence of the minimization routine. As we illustrate in our Monte Carlo simulations below, a loose stopping criterion for the outer loop can cause the routine to terminate early and produce incorrect point estimates. In some instances, these estimates may not even satisfy the first-order conditions for a local minimizer.

We denote by $\xi(\theta, \epsilon_{\text{in}})$ the approximated demand shock corresponding to a given value for θ and an inner-loop tolerance ϵ_{in} that determines the inner-loop stopping rule, (6). We also denote the true demand shock as $\xi(\theta, 0)$. We let $Q(\xi(\theta, \epsilon_{\text{in}}))$ be the programmed GMM objective function with the inner-loop tolerance ϵ_{in}. This more general notation allows us to examine numerical inaccuracy with the programmed inner loop, which is not present in the statistical theory of GMM.

First, we characterize the bias in evaluating the GMM objective function and its gradient at any structural parameters, θ, when there exist inner-loop numerical errors. In a duplication of notation, let $Q(\xi)$ be the GMM objective function for an arbitrary guess of ξ. We also use the big-O notation, which is a convention in the mathematics literature and is described in many textbooks such as van der Vaart (2000).

Theorem 3. *Let $L(\theta)$ be the Lipschitz constant for the inner-loop contraction mapping. For any structural parameters θ and given a the inner-loop tolerance ϵ_{in},*

1. $\left|Q(\xi(\theta, \epsilon_{\text{in}})) - Q(\xi(\theta, 0))\right| = O\left(\frac{L(\theta)}{1-L(\theta)}\epsilon_{\text{in}}\right)$
2. $\left\|\nabla_\theta Q(\xi(\theta))\left|_{\xi=\xi(\theta,\epsilon_{\text{in}})} - \nabla_\theta Q(\xi(\theta))\right|_{\xi=\xi(\theta,0)}\right\| = O\left(\frac{L(\theta)}{1-L(\theta)}\epsilon_{\text{in}}\right)$,

assuming both $\left\|\frac{\partial Q(\xi)}{\partial \xi}\Big|_{\xi=\xi(\theta,0)}\right\|$ *and* $\left\|\frac{\partial \nabla_\theta Q(\xi(\theta))}{\partial \xi}\Big|_{\xi=\xi(\theta,0)}\right\|$ *are bounded.*

The proof is in the appendix. Theorem 3 states that the biases in evaluating the GMM objective function and its gradient at any structural parameters are of the same order as the inner-loop tolerance adjusted by the Lipschitz constant for the inner-loop contraction mapping. Recall that a smooth optimization routine reports convergence when the gradient of the objective function is close to zero. In the next theorem, we analyze the numerical properties of the gradient. The theorem indicates circumstances in which the outer loop might report convergence despite a numerically inaccurate inner loop.[7] We also show that the choice of the outer-loop tolerance, ϵ_{out}, should depend on the inner-loop tolerance ϵ_{in} and the Lipschitz constant $L(\theta)$. This is important because using a tight outer loop tolerance eliminates spurious local minima.

[7]The numerical error in the gradient convergence test may encourage some researchers to use non-smooth optimization methods. See section 5.6.

Theorem 4. *Let $L(\theta)$ be the Lipschitz constant of the inner-loop contraction mapping for a given θ and let ϵ_{in} be the inner-loop tolerance. Let $\hat{\theta}(\epsilon_{in}) = \arg\max_{\theta} \{Q(\xi(\theta, \epsilon_{\text{in}}))\}$. In order for the outer-loop GMM minimization to converge, the outer-loop tolerance ϵ_{out} should be chosen to satisfy $\epsilon_{\text{out}} = O(\epsilon_{\text{in}})$, assuming $\left\| \nabla^2_{\theta} Q(\xi) \Big|_{\xi=\xi(\hat{\theta}(\epsilon_{in}),0)} \right\|$ is bounded.*

The proof of this theorem shows that if ϵ_{in} is large (the inner loop is loose), then the gradient will be numerically inaccurate. Therefore, ϵ_{out} needs to be large (the outer loop must be loose) for the optimization routine to converge. The proof is in the appendix.

An immediate consequence of these results is that the researcher may be tempted to select tolerances based on the convergence of the algorithms, rather than the precision of the estimates themselves. In situations where the inner loop is slow, a researcher may loosen the inner-loop tolerance, ϵ_{in}, to speed up the convergence of the contraction mapping. By Theorem 4, the resulting imprecision in the gradient could prevent the optimization routine from detecting a local minimum and converging. In turn, the researcher may be tempted to loosen the outer-loop tolerance to ensure convergence of the minimization routine. Besides concerns about imprecision in the estimates, raising ϵ_{out} could also generate an estimate that is not in fact a local minimum.

4.3 Finite-Sample Bias in Parameter Estimates from the Inner-Loop Numerical Error

In this section, we discuss the small-sample biases associated with inner-loop numerical error. Assume, given ϵ_{in}, that we have chosen ϵ_{out} to ensure that the algorithm is able to report convergence. Let $\theta^* = \arg\max_{\theta} \{Q(\xi(\theta, 0)\}$ be the maximizer of the finite-sample objective function without numerical error. We study the upper bound on the numerical errors in the final estimates, $\hat{\theta}(\epsilon_{\text{in}}) - \theta^*$, from using an inner loop-tolerance ϵ_{in}.

Theorem 5. *Assuming $\left\| \nabla_{\xi} Q(\xi) \Big|_{\xi=\xi(\hat{\theta}(\epsilon_{\text{in}}),0)} \right\|$ is bounded, the difference between the finite-sample maximizers with and without inner loop error satisfies*

$$O\left(\left\|\hat{\theta}(\epsilon_{\text{in}}) - \theta^*\right\|^2\right) \leq \left|Q\left(\xi\left(\hat{\theta}(\epsilon_{\text{in}}), \epsilon_{\text{in}}\right)\right) - Q(\xi(\theta^*, 0))\right| + O\left(\frac{L\left(\hat{\theta}(\epsilon_{\text{in}})\right)}{1 - L\left(\hat{\theta}(\epsilon_{\text{in}})\right)}\epsilon_{\text{in}}\right).$$

The proof is in the appendix. The right side of the inequality provides an upper bound for the order of the square of the numerical error in the parameters. There are two terms in this upper bound. The first term, $|Q(\xi(\epsilon_{\text{in}}, \epsilon_{\text{in}})) - Q(\xi(\theta^*, 0))|$, is the bias in the GMM function evaluated at the finite-sample true and estimated parameter values.[8] The second term arises from $\xi\left(\hat{\theta}(\epsilon_{\text{in}}), \epsilon_{\text{in}}\right) - \xi\left(\hat{\theta}(\epsilon_{\text{in}}), 0\right)$,

[8]This term is related to a formalization of folk knowledge in the numerical-optimization literature that if a function

the bias in demand shocks with and without the inner-loop error, is of the same of as $\frac{L\left(\hat{\theta}(\epsilon_{\text{in}})\right)}{1-L\left(\hat{\theta}(\epsilon_{\text{in}})\right)}\epsilon_{\text{in}}$. The "2" in the exponent of $\left\|\hat{\theta}\left(\epsilon_{\text{in}}\right)-\theta^*\right\|^2$ implies that in the worst case, the significant digits of the parameter estimates are about only half of that of bias in the objective values or that of the inner-loop tolerance.[9]

While the bound in Theorem 5 is not always sharp or large, in our Monte Carlo experiments below, we will show that the parameter errors associated with improper minimization and failure to detect convergence are more severe issues in practice.[10]

4.4 Large Sample Bias from the Inner-Loop Numerical Error

The previous section focused only on numerical errors for a finite data set. We now use statistical theory to examine the large-sample properties of the BLP estimator using the NFP algorithm. Before, θ^* was the true minimizer of the finite-sample GMM objective function without any inner-loop numerical errors. Now instead consider θ^0, the true parameters in the data generating process. Even a researcher with a perfect computer program will not be able to recover θ^0 because of statistical sampling error. Here we explore how numerical errors in the inner loop affect the consistency of the BLP estimator.

Recall that $\hat{\theta}\left(\epsilon_{\text{in}}\right)$ corresponds to the minimizer of $Q\left(\xi\left(\theta,\epsilon_{\text{in}}\right)\right)$, the biased GMM objective function with the inner-loop tolerance $\epsilon_{\text{in}.}$ Let $\bar{Q}\left(\xi\left(\theta,\epsilon_{\text{in}}\right)\right)=E\left[Q\left(\xi\left(\theta,\epsilon_{\text{in}}\right)\right)\right]$ be the probability limit of $Q\left(\xi\left(\theta,\epsilon_{\text{in}}\right)\right)$, as either $T\rightarrow\infty$ or $J\rightarrow\infty$, as in Berry, Linton and Pakes (2004). Clearly, $\theta^0=\arg\min\bar{Q}\left(\xi\left(\theta,0\right)\right)$ if the BLP model is identified.

Let asymptotics be in the number of markets, T, and let each market be an iid observation. By standard consistency arguments (Newey and McFadden 1994), θ^* will converge to θ^0 if $Q\left(\xi\left(\theta,0\right)\right)$ converges to $\bar{Q}\left(\xi\left(\theta,0\right)\right)$ uniformly, which is usually the case with a standard GMM estimator. Further, the rate of convergence of the estimator without numerical error from the inner loop is the standard parametric rate, $\sqrt{T}$. By the triangle inequality,

$$\left\|\hat{\theta}\left(\epsilon_{\text{in}}\right)-\theta^0\right\|\leq\left\|\hat{\theta}\left(\epsilon_{\text{in}}\right)-\theta^*\right\|+\left\|\theta^*-\theta^0\right\|\leq O\left(\sqrt{\left|Q\left(\xi\left(\hat{\theta}\left(\epsilon_{\text{in}}\right),\epsilon_{\text{in}}\right)\right)-Q\left(\xi\left(\theta^*,0\right)\right)\right|+\frac{L\left(\hat{\theta}\left(\epsilon_{\text{in}}\right)\right)}{1-L\left(\hat{\theta}\left(\epsilon_{\text{in}}\right)\right)}\epsilon_{\text{in}}}\right)+O\left(1/\sqrt{T}\right) \tag{7}$$

to be optimized has error η, then the minimizer of the function with error could have error of $\sqrt{\eta}$; see Chapter 8 in Gill, Murray and Wright (1981).

[9] Ackerberg, Geweke and Hahn (Theorem 2, 2009) studied the case where the objective function is differentiable in the equivalent of inner-loop error and found a linear rate. The differentiability of the objective function with respect to inner-loop error has not been established for the NFP objective function and assuming differentiability could potentially result in different conclusions. We allow for non-differentiability and find a square-root upper bound.

[10] An even tighter inner-loop tolerance is required when finite-difference numerical derivatives are used instead of analytic derivatives, as we show below. Also, we discussed above about why tighter tolerances are needed when the contraction mapping uses $\exp\left(\xi_j\right)$ instead of ξ_j directly. In our experiments, we use $\epsilon_{\text{in}}=10^{-14}$ for the latter reason.

The asymptotic bias due to numerical error in the inner loop persists and does not shrink asymptotically. This is intuitive: inner-loop error would introduce numerical errors in the parameter estimates even if the population data were used.

4.5 Loose Inner-Loop Tolerances and Numerical Derivatives

Most scholars use gradient-based optimization routines, as perhaps they should given that the GMM objective function is smooth.[11] Gradient-based optimization requires derivative information, by definition. One approach is to derive algebraic expressions for the derivatives and then to code them manually. Our results above assume that the researcher's optimizer has information on the exact derivatives. However, in many applications, such as the dynamic demand model we study below, calculating and coding derivatives can be very time consuming. In these situations, researchers may choose to use numerical derivatives. The gradient is approximated by

$$\nabla_d Q\left(\xi\left(\theta, \epsilon_{\text{in}}\right)\right) = \left\{ \frac{Q\left(\xi\left(\theta + de_k, \epsilon_{\text{in}}\right)\right) - Q\left(\xi\left(\theta - de_k, \epsilon_{\text{in}}\right)\right)}{2d} \right\}_{k=1}^{|\theta|}, \tag{8}$$

where d is a perturbation to an element of θ and and e_k is a vector of 0's, except for a 1 in the kth position of e_k. As $d \to 0$, $\nabla_d Q\left(\xi\left(\theta, \epsilon_{\text{in}}\right)\right)$ converges to $\nabla Q\left(\xi\left(\theta, \epsilon_{\text{in}}\right)\right)$, the numerically accurate derivative of $\nabla Q\left(\xi\left(\theta, \epsilon_{\text{in}}\right)\right)$. However, the ultimate goal of estimation is to minimize the objective function without numerical error. For this, we need the derivatives without numerical error, $\nabla Q\left(\xi\left(\theta, 0\right)\right)$, although that object is not available on the computer.

Lemma 9.1 in Nocedal and Wright (2006) shows that the numerical error in the gradient is bounded,

$$\left\| \nabla_d Q\left(\xi\left(\theta, \epsilon_{\text{in}}\right)\right) - \nabla Q\left(\xi\left(\theta, 0\right)\right) \right\|_\infty \leq O\left(d^2\right) + \frac{1}{d} O\left(\frac{L\left(\theta\right)}{1 - L\left(\theta\right)} \epsilon_{\text{in}}\right).$$

There are two terms in this bound. The $O\left(d^2\right)$ term represents the standard error that arises from numerical differentiation, (8). As $d \to 0$, the $O\left(d^2\right)$ term converges to 0. The second term $\frac{1}{d} O\left(\frac{L(\theta)}{1-L(\theta)} \epsilon_{\text{in}}\right)$ arises from the numerical error in the objective function, for a given $\epsilon_{\text{in}} > 0$. The $O\left(\frac{L(\theta)}{1-L(\theta)} \epsilon_{\text{in}}\right)$ term comes from part 1 of Theorem 3. If $\frac{1}{d} O\left(\frac{L(\theta)}{1-L(\theta)} \epsilon_{\text{in}}\right)$ is relatively large, as it is when the inner-loop tolerance is loose, then the bound on the error in the gradient is large. In this case, a gradient-based routine can search in the wrong direction, and end up stopping at a parameter far from a local minimum. Therefore, combining loose inner-loop tolerances and numerical derivatives may produce an extremely unreliable solver. Note that setting $d \to 0$ will send the term $\frac{1}{d} O\left(\frac{L(\theta)}{1-L(\theta)} \epsilon_{\text{in}}\right) \to \infty$. So setting d to be extremely small will only exacerbate the numerical error arising from using a loose inner-loop tolerance.

[11] See section 5.6 for why we do not prefer non-gradient methods.

5 Parameter Errors from Loose Inner-Loop Tolerances in the NFP Algorithm

This section presents evidence that using loose tolerances in the NFP inner loop can lead to incorrect parameter estimates. We show that parameter errors can arise both from fake data and field data. We use the fake data to show that a combination of numerical derivatives and loose inner-loop tolerances can lead to grossly incorrect parameter estimates. We use the field data to show that wrong parameter estimates can arise even with closed-form derivatives. Both the fake data and real data results are examples: there is no guarantee that any given dataset will combine with a poor implementation of the NFP algorithm to produce incorrect parameter estimates. However, we only need examples in order to show that NFP with loose inner-loop tolerances can produce incorrect parameter estimates.

5.1 NFP Algorithm Implementations

For all NFP implementations, we examine the one-step GMM estimator with Nevo's (2000) suggestion of using the weighting matrix $W = (Z'Z)^{-1}$, where Z is the $TJ \times D$ matrix of instruments $z_{j,t,k}$.[12] We use one fake dataset and one real dataset to show that NFP with loose inner loop tolerances can lead to incorrect parameter estimates.

We use three implementations of NFP for our real data and fake data tests. We use the same data and set of starting values for all three implementations. We use our numerical theory results from section 4 to guide us in the selection of inner- and outer-loop tolerances for the NFP algorithm. To assess the importance of those findings, we construct three scenarios which we examine for each Monte Carlo experiment. In the first scenario, we explore the implications of a tight outer-loop tolerance, set at $\epsilon_{\text{out}} = 10^{-6}$, and a loose inner-loop tolerance, set at $\epsilon_{\text{in}} = 10^{-4}$. The former outer-loop tolerance is the default setting for most state-of-the-art optimization algorithms. However, from our numerical-theory results, we know the latter inner-loop tolerance is too large. One could think of this scenario as representing the "frustrated researcher" who loosens the inner loop to speed the apparent rate of convergence. In the second scenario, we explore the results from Theorem 4, whereby the loose inner-loop tolerance could, in turn, prevent the outer loop from converging. Specifically, we keep $\epsilon_{\text{in}} = 10^{-4}$ and set $\epsilon_{\text{out}} = 10^{-2}$. One can think of this scenario as representing the attempt of the researcher to loosen the outer loop to force it to converge, even though in practice the converged point may not actually satisfy the first-order conditions. In our third scenario, we implement the "best practice" settings for the NFP algorithm with $\epsilon_{\text{in}} = 10^{-14}$ and $\epsilon_{\text{out}} = 10^{-6}$.[13]

[12]We choose a simple weighting matrix because our focus is on comparing algorithms, not finding the most statistically efficient estimator.

[13]We choose 10^{-14} because it is tighter than the usual default outer-loop tolerance, 10^{-6}. We are looser than 10^{-12} for the reasons stated in footnote 5.

For all implementations of NFP, we use the same programming environment (MATLAB) and the same optimization package (KNITRO using the TOMLAB interface). We selected MATLAB because this is a commonly-used software package among practitioners. We also selected the KNITRO optimization package instead of MATLAB's built-in optimization routines as the former is a highly-respected, state-of-the-art solver in the optimization community (Byrd, Nocedal and Waltz 1999).[14] For our fake data example, we use numerical derivatives. For our real data example, we also supply derivatives for each algorithm because all local optimization methods improve if the user supplies exact derivatives of the objective function.[15]

We also customized several aspects of the NFP algorithm to increase speed. In the case of NFP, the most notable speed improvements came from exploiting as much as possible the built-in linear-algebra operations ("vectorization") for the inner loop. In addition, we exploited the normality assumption for $F_\beta(\beta;\theta)$ to concentrate the means out of the parameter search under the NFP algorithm, as suggested in Nevo (2000b). Therefore, the NFP algorithm can be recast to search only over the standard deviation of the random coefficients, rather than both the means and standard deviations. Relaxing the normality assumption would prevent the use of this simplification (except perhaps in other location and scale families), which could improve the relative speed performance of MPEC over NFP even further.

5.2 The Fake Data Generating Process

We use the demand model from section 2. In this section we describe a data-generating process. We allow for $K = 3$ observed characteristics, in addition to prices. We also estimate a random coefficient on the intercept, β_i^0, which models the relative attractiveness of purchasing any of the products instead of the outside good. β_i^p, the price coefficient, is also random.

We focus on markets with a fairly large number of products, $J = 25$, to ensure that our results are not due to sampling error. We also consider a large number of statistically independent markets, here $T = 50$. Although not reported, we noticed large biases in the mean and standard deviation of the intercept, β_i^0, as well as functions of the parameters (like price elasticities) when a small number of markets was used. Intuitively, the moments of β_i^0 are identified in part from the share of the outside good, and more markets are needed to observe more variation in the outside good's shares.

[14]We found KNITRO is more likely to converge to a valid local minimum than MATLAB's included solver, fminunc.

[15]Another option is to use automatic differentiation software (Griewank and Corliss 1992). Automatic differentiation software is automatically used by some languages, such as AMPL, and can be accessed with the TOMLAB interface for MATLAB. Software packages like AMPL are impractical for NFP algorithms because AMPL is a problem definition language, not a general purpose programming language like MATLAB. Therefore, we use MATLAB for all our empirical analysis. However, in practice, many users may find AMPL more convenient for the MPEC implementation. One warning is that the automatic differentiation overhead in AMPL uses lots of computer memory.

For product j in market t, let

$$\begin{bmatrix} x_{1,j,t} \\ x_{2,j,t} \\ x_{3,j,t} \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & -0.8 & 0.3 \\ -0.8 & 1 & 0.3 \\ 0.3 & 0.3 & 1 \end{bmatrix}\right).$$

Likewise, $\xi_{j,t} \sim N\left(0, \sigma_\xi^2\right)$, with the default $\sigma_\xi^2 = 1$. Price is

$$p_{j,t} = \left| 0.5 \cdot \xi_{j,t} + e_{j,t} + 1.1 \cdot \sum_{k=1}^{3} x_{k,j,t} \right|,$$

where $e_{j,t} \sim N(0,1)$ is an innovation that enters only price. Prices are always positive. Prices are endogenous as $\xi_{j,t}$ enters price. For each product j in market t, there is a separate vector $z_{j,t}$ of $D = 6$ instruments. A powerful instrument must be correlated with $p_{j,t}$ and a valid instrument must not be correlated with $\xi_{j,t}$. Each instrument $z_{j,t,d} \sim U(0,1) + \frac{1}{4}\left(e_{j,t} + 1.1 \cdot \sum_{k=1}^{3} x_{k,j,t}\right)$, where $U(0,1)$ is a standard continuous-uniform random variable.

The goal is to estimate the parameter vector θ in $F_\beta(\beta; \theta)$, the distribution of the random coefficients. To maintain consistency with the application in BLP (1995) and the related empirical literature, we assume independent normal random coefficients on each product characteristic and the intercept. Thus, $F_\beta(\beta; \theta)$ is the product of five independent normal distributions ($K = 3$ attributes, price and the intercept) characterized by means and standard deviations contained in θ. The true values of the moments of the random coefficients $\beta_i = \left\{\beta_i^0, \beta_i^1, \beta_i^2, \beta_i^3, \beta_i^p\right\}$ are $E[\beta_i] = \{-1.0, 1.5, 1.5, 0.5, -3.0\}$ and $\mathrm{Var}[\beta_\mathrm{i}] = \{0.5, 0.5, 0.5, 0.5, 0.2\}$.

Our focus is not on numerical integration error, so we use the same set of 100 draws to compute market shares in the data generating and estimation phases using (3). By using the same draws, we have no simulation error. By using 100 draws, our code is proportionately faster, allowing us to try more starting values and, in a later section, more Monte Carlo replications. The computational times reported in a later section should be inflated by the ratio of the number of draws that would be used in real-data applications (say 4000) to the number we use, 100.[16]

5.3 The Nevo (2000b) Cereal Data

We use the cereal dataset from Nevo (2000b) to assess whether NFP with loose inner-loop tolerances can produce incorrect parameter estimates with real data. We refer the reader to Nevo (2000b) for

[16]The use of simulation in BLP is not the same as in the traditional method of simulated moments. In BLP, one simulates an integral that is nested within the inner loop, rather than simulating the entire model by calculating its solution for each of several draws of all error terms.

a description of these data. Interestingly, Knittel and Metaxoglou (2008) recently used these same data to study a related potential concern with BLP. They report that the parameter estimates are extremely sensitive to the starting values used for the NFP algorithm because many true local minima exist in the GMM objective function. We agree that the BLP problem is not convex and, therefore, may potentially generate multiple optima. As a check for this problem, we use our NFP code with 50 different starting points to estimate the demand model with Nevo's cereal data set. We set the inner loop tolerance to be 10^{-14}. To replicate Knittel and Metaxoglou, we pick our starting values by taking 50 draws from the standard normal distribution.[17] For each of the 50 runs, our NFP code finds the same objective function value, 4.5615, which is also the lowest objective value found by Knittel and Metaxoglou (2008).[18] With multiple starting values, careful implementation of the numerical procedures, and state-of-the-art optimization solvers, we find the BLP GMM estimator produces reliable estimates.[19]

5.4 Fake Data, Numerical Derivatives and False Parameter Estimates

For NFP, the numerical theory in section 4 raises several concerns about the common practice of setting the tolerance, ϵ_{in}, to be too high (too loose). Section 4.5 shows that a combination of a loose inner loop, numerical derivatives and a smooth optimization routine can produce incorrect parameter estimates. Also recall that Theorem 4 shows that if ϵ_{in} is too loose, ϵ_{out} must be set to be too loose in order for the routine to be able to report convergence.

In this subsection, we explore empirically the problems with loose inner loop tolerances and numerical derivatives. We create one simulated fake dataset, using the data-generating process from section 5.2. Holding the simulated data fixed, we first compare the estimates produced from 100 randomly-chosen starting values for the own-price demand elasticities. We run each of the three NFP implementations described in section 5.1 for each of the 100 vectors of starting values. Each starting value is has a uniform $(0, 7)$ distribution. Table 1 reports the results for the 100 different starting values. The first row reports the fraction of runs for which the routine reports convergence. As Theorem 4 shows, if the inner loop tolerance is a loose $\epsilon_{\text{in}} = 10^{-4}$ and the outer loop tolerance a standard value of $\epsilon_{\text{out}} = 10^{-6}$, the routine will never report convergence. Column one confirms this finding as none of the runs with a loose inner loop and tight outer loop converge. In contrast, column two indicates that

[17]We also experimented with multiplying the starting values by the solution reported in Nevo (2000b). The results were similar.

[18]We also use the MATLAB code by Nevo (2000b), except that we use the KNITRO solver as the search algorithm. For 50 starting points, KNITRO only converges for 25 of the 50 runs. However, all 25 successful runs converge to the same solution with objective value 4.5615.

[19]We also use MATLAB's genetic algorithm routine for one run. The genetic algorithm finds a point with the objective function value 98.4270, which is clearly an order of magnitude higher than 4.5615, the minimum we find using the gradient-based method. We then start KNITRO from this point found by the generic algorithm and KNITRO finds the solution with objective value 4.5615. So the genetic algorithm does not always find a local minimum.

Table 1: Three NFP Implementations: Varying Starting Values for One Fake Dataset, with Numerical Derivatives

	NFP Loose Inner	NFP Loose Both	NFP Tight	Truth
Fraction Convergence	0.0	0.54	0.95	
Frac.< 1% > "Global" Min.	0.0	0.0	1.00	
Mean Own Price Elasticity	-7.24	-7.49	-5.77	-5.68
Std. Dev. Own Price Elasticity	5.48	5.55	~0	
Lowest Objective	0.0176	0.0198	0.0169	
Elasticity for Lowest Obj.	-5.76	-5.73	-5.77	-5.68

We use 100 starting values for one fake dataset. The NFP loose inner-loop implementation has $\epsilon_{\text{in}} = 10^{-4}$ and $\epsilon_{\text{out}} = 10^{-6}$. The NFP loose-both implementation has $\epsilon_{\text{in}} = 10^{-4}$ and $\epsilon_{\text{out}} = 10^{-2}$. The NFP-tight implementation has $\epsilon_{\text{in}} = 10^{-14}$ and $\epsilon_{\text{out}} = 10^{-6}$. We use numerical derivatives using KNITRO's built-in procedures.

the algorithm is more likely to converge (54% of the runs) when we also loosen the tolerance on the outer loop. As we will show below, this semblance of convergence is merely an artifact of numerical imprecision that leads to misleading estimates. Finally, NFP with tight tolerances converges in 95% of the runs.

To diagnose the quality of the estimates, the second row of Table 1 shows the fraction of runs where the reported GMM objective function value was within 1% of the lowest objective function that we numerically found across all three NFP implementations and all 100 starting values (300 cases). We call this value the "global" minimum, although of course we cannot prove we have found the true global minimum. In the first two columns, corresponding to the scenario with a loose inner loop and the scenario with a loose inner and outer loop respectively, none of the 100 starting values lead to finding the global minimum.[20] In contrast, NFP tight found the global minimum all of the time. The perfect performance of NFP on this metric is peculiar to this one example, as NFP tight should not find the global minimum every time, because a gradient-based optimization routine may indeed converge to a true local minimum that is not the global minimum.

The third and fourth rows of Table 1 provide measures to assess the economic implications of our different implementations. We use estimated price elasticities to show how naive implementations could produce misleading economic predictions. In the third row, we report the mean own-price elasticity, across all $H = 100$ starting values, all J products and all T markets:

$$\frac{1}{H}\sum_{h=1}^{H}\frac{1}{T}\sum_{t=1}^{T}\frac{1}{J}\sum_{j=1}^{J}\eta_{j,t}^{p}\left(\hat{\theta}^{h}\right),$$

where $\hat{\theta}^h$ is the vector of parameter estimates for the hth starting value and $\eta_{j,t}^{p}\left(\hat{\theta}^{h}\right)$ is the own price-

[20]Even with loose inner-loop tolerances, the GMM objective value function is accurate to a few decimal places. The values reported in the fifth row of Table 1 are identical whether the reported parameter estimates are evaluated at a NFP objective function with $\epsilon_{\text{in}} = 10^{-4}$ or $\epsilon_{\text{in}} = 10^{-14}$.

elasticity of firm j in market t, at those parameters. The fourth row reports the standard deviation of the mean own-price elasticity across all 100 starting values.

Beginning with the third row, first note that in the final column we report the own-price demand elasticity evaluated at the true parameter values: -5.68. As we hoped, NFP with a tight tolerance produces an estimate near the truth, -5.77. BLP is capable of recovering the true parameters, with good data. On the other hand, we immediately see that the loose-tolerance implementations of NFP produce mean elasticities that are not nearly as close to the truth as NFP with a tighter inner-loop tolerance. The mean of the NFP loose-inner implementation is -7.24, higher in absolute value the true value of -5.68. The loose-both mean is -7.49. The standard deviations of own-price elasticities for the loose inner-loop tolerances are huge: 5.48 and 5.55. With a loose inner-loop tolerance and numerical derivatives, section 4 shows that there is no reason to expect the NFP algorithm to produce correct parameter estimates.

One question is whether a researcher who tried 100 starting values could get close to the true estimates. If "close" is defined by getting one significant digit of the mean own-price elasticity correct, the answer for this particular dataset is "no". However, the reported elasticities for NFP with loose inner loops are -5.76 and -5.73, compared to the numerically correct -5.77 from NFP tight. What is happening is that the NFP implementations with the loose inner-loop tolerances tend to stop near the starting values. By using 100 starting values, the researcher is exploring 100 regions of the objective function. It is equivalent to just evaluating the objective function at 100 points and taking the final estimates to be the minimum. However, there is no guarantee that 100 starting values will lead to "close" estimates in other datasets. Indeed, in the next section we show an example where even the elasticities corresponding to the lowest objective function value have economically important numerical error in them.

5.5 Parameter Errors with the Cereal Data and Closed-Form Derivatives

There are at least three concerns one might have with the previous section's fake-data example. First, perhaps real data does not have the problems we found. Second, the example relied on numerical derivatives; perhaps coding closed-form derivatives eliminates all concerns with achieving incorrect parameter estimates with NFP with loose inner-loop errors. Third, the incorrect elasticity estimates in Table 1 were really variable across starting values. A researcher who tried even a few starting values and found the wildly different elasticity estimates would diagnose that something is wrong, or at least falsely conclude that the BLP objective function has an enormity of local minima. A careful researcher might then explore settings like the inner-loop tolerance, and eventually fix the implementation error. This section uses Nevo's cereal data to produce an example of incorrect parameter estimates that is

Table 2: Three NFP Implementations: Varying Starting Values for Nevo's Cereal Dataset, with Closed-Form Derivatives

	NFP Loose Inner	NFP Loose Both	NFP Tight	NFP Tight Simplex
Fraction Reported Convergence	0.0	0.76	1.00	1.00
Frac. Obj. Fun. < 1% Greater than "Global" Min.	0.0	0.0	1.00	0.0
Mean Own Price Elasticity Across All Runs	-3.82	-3.69	-7.43	-3.84
Std. Dev. Own Price Elasticity Across All Runs	0.4	0.07	˜0	0.35
Lowest Objective Function Value	0.00213	0.00683	0.00202	0.00683
Elasticity for Run with Lowest Obj. Value	-6.71	-3.78	-7.43	-3.76

We use the same 50 starting values for each implementation. The NFP loose inner loop implementation has $\epsilon_{\text{in}} = 10^{-4}$ and $\epsilon_{\text{out}} = 10^{-6}$. The NFP loose both implementation has $\epsilon_{\text{in}} = 10^{-4}$ and $\epsilon_{\text{out}} = 10^{-2}$. The NFP tight implementation has $\epsilon_{\text{in}} = 10^{-14}$ and $\epsilon_{\text{out}} = 10^{-6}$. The Nelder-Meade or simplex method uses a tighter inner loop tolerance of $\epsilon_{\text{in}} = 10^{-14}$ and MATLAB's default values for the simplex convergence criteria. We manually code closed-form derivatives for all methods other than for Nelder-Meade, which does not use derivative information.

robust to these concerns.

The results in Table 2 are of the same format as Table 1. We try out 50 starting values on the same dataset; the same starting values are used for each method. We also report results for the Nelder-Meade or simplex algorithm, which we will discuss in the next section. As Theorem 4 predicts, in row 1 we find that 0% of the NFP-loose inner-loop starting values converge. Loosening the outer loop is one approach to finding convergence; the second column finds that 76% of starting values report convergence when this is done. 100% of the starting values converge for NFP tight. The second row shows that 100% of the NFP-tight starting values find the apparent global minimum, 0.00202, in Nevo's cereal data. None of the NFP-loose tolerance implementations find the global minimum.

The loose inner-loop and loose-both methods find a mean own-price elasticity of -3.82 and -3.69, respectively. This is about half the value of -7.43 found with NFP tight. Further, the estimates are all tightly clustered around the same points. With standard deviations of 0.40 and 0.07 for the loose inner loop methods, the answers are for the most part consistently wrong across runs. The fifth row shows the smallest objective function values found by the loose-inner-loop and loose-both routines are 0.00213 and 0.00683, respectively. The second result is far from the truth of 0.00202. We manually inspected all 50 starting values and found that 1 out of 50 of the loose-inner-loop=only runs was anywhere near the apparent global minimum. This is 1 out of 50 result is somewhat of a fluke, because with other sets of 50 starting values, the loose-inner-loop method replicated the performance of the loose-both method and had no starting values converge to objective-function values anywhere near the apparent global minimum.

What is going on here numerically? The points that the loose inner loop methods are converging to look somewhat like local minima, but are indeed not minima. When the objective function is numerically imprecise, the optimization routine thinks that is has found a local minimum, and stops there, whether or not it reports convergence. Thus, the Nevo cereal data have a dangerous point that mimics a local minimum but is not actually a minimum.

These results show that a naive but otherwise careful researcher might feel that his or her estimates were correct because even trying fifty different starting values always produces around the same estimates. Even if the researcher correctly coded the derivatives in closed form and used a high-quality, professional optimizer like KNITRO, the NFP loose inner and loose both implementations can consistently converge to the wrong elasticity, and the elasticity can be half of the true value. Thus, there is no diagnostic that a researcher can do that will detect all types of numerical error. With Nevo's cereal dataset, an inner loop tolerance that is too loose will lead to replicable but wrong own-price elasticity estimates. Only using an a priori theoretically correct setting, like a tight inner-loop tolerance, will avoid these errors.

5.6 Nelder-Meade or Simplex Algorithm

In the previous parts of this section, we have pointed out errors in parameter estimates that loose inner-loop methods cause for search routines that use numerical derivatives (Table 1) and even closed-form derivatives (Table 2). One logical response would be to look for methods that do not use information on derivatives at all. Indeed, there is no particular theoretical reason to think that inner-loop error from a loose inner-loop tolerance will cause numerical problems for search methods that rely on numerical derivatives.

The most commonly used non-derivative method in econometrics, at least in our informal surveys (for example, it was used in BLP's original paper), is the Nelder-Meade or simplex algorithm. This algorithm does not use derivative information. Our message will be that the simplex algorithm can fail to find a global minimum, even in situations without numerical error in the objective function, meaning with a tight inner loop. To see this, look at the fourth column of Table 2, which applies the simplex algorithm to the Nevo cereal dataset.[21] We use the same starting values as we used for the three implementations of the gradient-based solver and a tight inner loop. We see that the simplex algorithm fails to find the global minimum for all fifty starting values. Further, the elasticity estimate of -3.76 is around half of the numerically correct -7.43, and the elasticity's standard deviation across the fifty starting values is a relatively tight 0.35. The simplex algorithm converges to roughly the same wrong solution for each of the fifty runs. We conclude the simplex algorithm estimates are incorrect (i.e. not local optima) in two ways. First, NFP with a tight tolerance never converged to these solutions. Second, if we use any one of the simplex method's converged point estimates as a starting value for NFP tight, the latter does not report that start value as a local optimum. In summary, the

[21]We use the version of the simplex algorithm in MATLAB, which is fminsearch.

simplex algorithm appears to generate false solutions in our empirical examples.

The numerical analysis literature has come to the theoretical and experimental conclusion that the Nelder-Meade method is a poor alternative to traditional gradient-based solvers. The only theoretical convergence results we know of for simplex methods are due to Lagarias et al. (1998), who show convergence for only convex optimization problems with one or two unknowns. McKinnon (1998) presents an example where the simplex method converges to a point that is not a valid local minimum. Extensive numerical experiments by Wright (1995) show that the simplex algorithm can converge to an acceptably accurate solution with substantially fewer function evaluations than a gradient-based method with finite-difference gradients. Unfortunately, the results also show that the Nelder-Mead method can be "horrifically inefficient and unreliable." The range in performance of the Nelder-Mead method is huge. As we have shown, it simply does not find the lowest objective-function value (the one found by the gradient-based method all fifty times) on the Nevo dataset, even with fifty starting values. Also, the simplex routine tends to converge to around same wrong answer, so looking at the variability of estimates across starting values cannot diagnose the failure to find the global minimum or even a valid local minimum.

6 MPEC: A Constrained Optimization Approach

We have established that only NFP with a tight inner-loop tolerance can produce reliable parameter estimates. According to Theorem 5, if we wish to achieve the default numerical precision in the outer loop of 10^{-6}, we need to set the NFP inner loop-tolerance to 10^{-12} or tighter, for reliable parameter estimates. Using a tight inner loop means NFP may be slow. Further, in the previous section, we established that the NFP method's inner loop converges linearly and can be slow when the Lipschitz constant is close to 1. A slow inner loop might cause researchers to choose loose tolerances for the inner loop, which might lead to problems in establishing the convergence of the outer loop as well as errors in the reported parameter estimates.[22]

In this section, we propose an alternative algorithm based on Su and Judd's (2007) constrained optimization approach for estimating structural models. Below we show that the MPEC approach generates the same solution as NFP. MPEC can save computation time while completely avoiding issues of numerical precision by eliminating the inner loop of the NFP algorithm. MPEC can also be used for demand models that do not have a contraction-mapping solution. In their original paper, Su and Judd focus more on solving for the unknown variables in economic models, such as value

[22]Alternative methods to a contraction mapping for solving systems of nonlinear equations with faster rates of convergence typically have other limitations. For instance, the traditional Newton's method is only guaranteed to converge if the starting values are close to a solution, unless one includes line-search or trust-region procedure subject to some technical assumptions. In general, most practitioners would be daunted by the task of nesting a hybrid Newton method customized to a specific demand problem inside the outer optimization over structural parameters.

functions in single-agent dynamic-programming problems and the entry probabilities of rival firms in static entry games with multiple equilibria. We apply this insight to the recovery of the unobserved demand shocks that enter the criterion function during estimation of a structural model. In particular, we present a constrained-optimization formulation for random-coefficients demand estimation.

If W is the GMM weighting matrix, our constrained-optimization formulation is

$$\begin{array}{cc} \min\limits_{\theta,\xi} & g(\xi)' W g(\xi) \\ \text{subject to} & s(\xi;\theta) = S \end{array} . \tag{9}$$

The moment condition term $g(\xi)$ is just $g(\xi) = \frac{1}{T}\sum_{t=1}^{T} \xi_t z_t$. In MPEC, the market share equations are introduced as nonlinear constraints to the optimization problem. The objective function is specified primitively as a function of the demand shocks ξ. The main difference compared to the traditional NFP method is that we optimize over both the aggregate demand shocks ξ and the structural parameters θ. We do not use NFP's inner loop to enforce $\xi = \xi(\theta)$ for every guess of θ; rather we impose that the predicted shares equal the actual shares in the data only at the solution to the minimization problem.

The next theorem shows the equivalence of the first-order conditions between the NFP method (4) and the constrained optimization formulation (9). Hence, any first-order stationary point of (4) is also a stationary point of (9), and vice versa. This means that MPEC and NFP are the same estimators, statistically. Any statistical property of the original BLP estimator applies to the estimator when computed via MPEC. Hypothesis teats and standard errors are the same between both methods. There is no need to use the statistical theory for equality-constrained estimators to derive the statistical properties of the MPEC estimator.

Theorem 6. *Let the BLP demand model admit a contraction mapping. The set of first order conditions to the MPEC minimization problem in* (9) *is equivalent to the set of first order conditions to the true (no numerical error) GMM inner loop / outer loop method that minimizes* (4)*.*

The main benefit of the MPEC formulation is that it circumvents the need for the inner loop. By eliminating the inner loop, MPEC is less prone to numerical errors and is potentially faster. We discuss these benefits below.

The constrained optimization problem defined by (9) can be solved using modern nonlinear solvers developed by researchers in numerical optimization. Unlike the NFP algorithm, where users need to exercise caution in the choice of tolerance levels for both inner and outer loops, the defaults on feasibility and optimality tolerances in nonlinear solvers for constrained optimization are usually sufficient. These default tolerances have been established to work well in hundreds or thousands of papers in the numerical analysis literature. The default tolerances are usually sufficient because the market share equations and GMM objective function (without an inner loop) are exposed to the optimization routine. In short, MPEC lets a state-of-the-art optimization algorithm handle all of the computational aspects of the problem. In contrast, with NFP, the researcher needs to customize a nested-fixed-point calculation, which could result in naive errors.

In addition to simplifying implementation, bypassing the inner loop reduces several sources of numerical error that could, possibly, lead to non-convergence. We have detected some common practices with the coding of the inner loop that could naively lead to numerical error. These include loose choices for the inner-loop tolerance (as discussed previously) and an adjustable inner-loop tolerance that is loosened for parameter values deemed "far" from the solution to the outer loop.[23] MPEC relegates all numerical calculations to a single call to the outer loop, which is solved using a state-of-the-art optimization package, rather than the user's own customized algorithm.

Our approach can also create substantial speed advantages. As we showed in the previous section, the contraction mapping in the NFP algorithm might be slow as the Lipschitz constant approaches one. By contrast, the MPEC method does not nest any contraction mappings and so we expect its speed to be relatively invariant to the Lipschitz constant. Most optimization solvers for smooth problems use Newton-type methods to solve the Karush-Kuhn-Tucker system of the first-order optimality conditions. Newton's method is quadratically convergent, faster than the linear rate of the contraction mapping that is the NFP inner loop. Another potential source of acceleration of speed comes from the fact that our approach allows constraints to be violated during the solving process. In contrast, the NFP algorithm requires solving the share equation (2) *exactly* for every parameter θ examined in the outer, optimization loop. Modern numerical optimization solvers do not enforce that the constraints are satisfied at every iteration; it suffices that the constraints hold at the solution. This flexibility avoids wasting computational time on iterates away from the true parameters. Still another potential speed advantage is that the outer algorithm has more information: the optimization routine is exposed to the constraints, the derivatives of the constraints and of the objective function, and the sparsity pattern of the constraints. On sparsity, recall that demand shocks for market t do not enter the constraints for market $t+1$. Therefore, this constrained optimization problem is highly sparse.

Most constrained optimization solvers are based on sequential quadratic programming or interior point methods. As stated earlier, these solvers use Newton-based methods. Economists are often

[23]The trick consists of using a loose inner loop tolerance when the parameter estimates appear "far" from the solution and switching to a tighter inner loop tolerance when the parameter estimates are "close" to the solution. The switch between the loose and tight inner loop tolerances is usually based on the difference between the successive parameter iterates, e.g, if $\left\|\theta^{k+1} - \theta^{k}\right\| \leq 0.01$, then $\epsilon_{\text{in}} = 10^{-8}$; otherwise, $\epsilon_{\text{in}} = 10^{-6}$. Suppose that the following sequence of iterates occur: $\left\|\theta^{k+1} - \theta^{k}\right\| \geq 0.01$ ($\epsilon_{\text{in}} = 10^{-6}$), $\left\|\theta^{k+2} - \theta^{k+1}\right\| \leq 0.01$ ($\epsilon_{\text{in}} = 10^{-8}$), and $\left\|\theta^{k+2} - \theta^{k+1}\right\| \geq 0.01$ ($\epsilon_{\text{in}} = 10^{-6}$). The NFP objective value can oscillate because of the use of two different inner loop tolerances. This oscillation can prevent the NFP approach from converging.

skeptical about Newton's method because it might not converge if the starting point is far away from the solution. While this perception is true for the purest textbook version of Newton's method, modern Newton-like methods incorporate a line-search or a trust-region strategy to give more robustness to the choice of starting values. We refer readers to Nocedal and Wright (2006) and Kelley (1995, 1999, 2003) for further details on modern optimization methods for smooth objectives and constraints. In any case, the researcher should always use several guesses of starting values for real empirical research.

Our implementation of MPEC for the BLP model is slightly more sophisticated than the simple explanation in (9). We actually treat the moments as separate parameters, so that the problem being solved is

$$\begin{array}{ll} \min\limits_{\theta,\xi,\eta} & \eta' W \eta \\ \text{subject to} & g(\xi) = \eta \\ & s(\xi;\theta) = S \end{array} . \tag{10}$$

The solution to this new problem is the same as (9). The objective function is now a simple quadratic, $\eta' W \eta$, rather than a more complex, direct function of ξ; the additional constraint $g(\xi) - \eta = 0$ is linear in both ξ and η and, hence, does not add additional difficulties to the original problem. Computationally, the advantage with this equivalent formation is that we increase the sparsity of the constraint Jacobian and the Hessian of the Lagrangian function by adding the additional variables and constraints. In numerical optimization, it is often easier to solve a large but sparse problem than a small but dense problem. Another advantage of MPEC over NFP is that the objective function and constraints in MPEC are likely more "smooth" or less "nonlinear" in the unknowns than the NFP objective function is in θ. In NFP, the mapping from θ to the objective function value uses the highly nonlinear inner-loop transformation $s^{-1}(S_t;\theta)$, while no such inner loops are used by MPEC. Thus, MPEC may be a "smoother" nonlinear programming problem.

A common response to practitioners when they first hear about MPEC is that maximizing over a large number of parameters is a numerically daunting challenge. Below we show that this need not be the case. Indeed, the performance comparison of MPEC and NFP may be relatively constant as the number of products and markets increases.

7 Speed Comparisons of MPEC and NFP

NFP with a tight inner loop will produce correct parameter estimates if many starting values are used. However, NFP can be slow on some datasets. This section uses fake data and the Nevo cereal data to compare the speed of MPEC and NFP. We present examples where MPEC performs better than NFP. This is not meant to be a theorem: there could be cases where NFP is faster than MPEC. We now show that, in many situations, NFP may be computationally impractical in terms of execution time. In contrast, we will show that MPEC's execution time appears to be relatively invariant across these situations. Our approach exploits the Lipschitz constant for the BLP contraction mapping derived in section 4.1. We conjecture that data with a higher Lipschitz constant, and hence a higher upper bound on the rate of convergence of the inner loop, may slow NFP estimation. The idea will be to manipulate various components of the data-generating process in order to measure their respective impacts on the Lipschitz constant. We have no reason to believe cases exist where MPEC grows really slow with some equivalent of a Lipschitz constant. Therefore, we suspect that MPEC will be more robust against extremely slow performance. Keep in mind that it is these slow-performing cases where a researcher will be tempted to loosen the inner-loop tolerance, leading to the problem of incorrect parameter estimates that we earlier highlighted.

7.1 NFP and MPEC Implementations

We code NFP and MPEC using closed-form derivatives. As the proof of Theorem 6 shows, the components of these derivatives are the same for both methods. We use the quadratic form of MPEC in (10). We give the sparsity pattern of the constraints to the optimization routine, for MPEC.

An important point for our speed comparison is the choice of starting values. We always use five starting values, which are uniform random numbers. For MPEC, we effectively use the same starting values as we do for NFP; we pick values such that the two algorithms are initialized to have the same objective function value.[24] For each NFP starting value, we run the inner loop once and use this vector of demand shocks and mean taste parameters as starting values for MPEC. This is our attempt to equalize the starting values across NFP and MPEC.[25]

As before, we use 100 simulation draws for both MPEC and NFP. For the fake data experiments, we use the same 100 simulation draws to generate the data and to estimate the model. This shuts down simulation error. Raising the number of simulation draws to a more reasonable number, say 10,000, would increase the CPU times of both MPEC and NFP by about 100 times. So the reported times below are 100 times too slow, compared to an actual empirical investigation.

7.2 Base Fake-Data Case

Here we define a base fake-data case, which is then perturbed to vary the Lipschitz constants in the examples that follow. The model is nearly the same as Section 5.2. We use $T = 50$ to speed the

[24]Our MATLAB code is parallelized across multiple cores to some extent. However, we report CPU times and not clock times.

[25]Adding the NFP inner loop takes two lines of code once MPEC has been coded, so it is not unreasonable to expect a practitioner to be able to reproduce our choice of MPEC starting values.

Table 3: Lipschitz Constants for the NFP Algorithm

Parameter Scale		Std. Dev. of Shocks ξ		# of Markets T		Mean of Intercept $E\left[\beta_i^0\right]$	
Altered Value	Mean Lipschitz	Altered Value	Mean Lipschitz	Altered Value	Mean Lipschitz	Altered Value	Mean Lipschitz
0.01	0.985	0.1	0.808	25	0.860	-2	0.771
0.1	0.971	0.25	0.813	50	0.871	-1	0.871
0.50	0.887	0.5	0.832	100	0.888	0	0.936
0.75	0.865	1	0.871	200	0.888	1	0.971
1	0.871	2	0.934			2	0.988
1.5	0.911	5	0.972			3	0.996
2	0.938	20	0.984			4	0.998
3	0.970						
5	0.993						

runs somewhat. The mean of the random coefficients is $E[\beta_i] = (0.1, 1.5, 1.5, 0.5, -3.0)$. The prices are $p_{j,t} = 3 + \xi_{j,t} \cdot 1.5 + u_{j,t} + \sum_{k=1}^{3} x_{k,j,t}$, where $u_{j,t}$ is a uniform$(0,5)$ random variable. Likewise, $z_{j,t,d} = \tilde{u}_{j,t,d} + \frac{1}{4}\left|u_{j,t} + 1.1 \cdot \sum_{k=1}^{3} x_{k,j,t}\right|$, where $\tilde{u}_{j,t,d}$ is another uniform$(0,1)$ random variable and $u_{j,t}$ is the same variable as before. For each table below, we calculate 20 different fake datasets, and reported means are across these 20 replications.

7.3 Lipschitz Constants

Recall that the Lipschitz constant derived in section 4.1 is related to the demand sensitivity to the unobserved quality, $\xi_{j,t}$. Moreover, this demand sensitivity is roughly related to the degree of asymmetry in market shares. Therefore, we experiment with different features of the data-generating process that affect the degree of share asymmetry. Table 3 reports the Lipschitz constant for the base-case data-generating process of section 5.2. Each cell reports the mean of the Lipschitz constant evaluated at the true parameter values across 30 data sets / replications.

In our first experiment, reported in the first column of Table 3, we manipulate the scale of the parameters, β_i. We multiply the β_i of each of our ns simulated consumers in the data-generating process by one of the constants listed in the table. We find that the Lipschitz constant is non-monotone in the scale, with the constant first falling and then rising again. This non-monotonicity comes from the fact that our manipulation also changes the levels of the market shares. Nevertheless, holding the sample size fixed, we see fairly large changes in the upper bound on the rate of convergence of the contraction mapping.

The second column of Table 3 increases the standard deviation of the product-and-market-specific demand shocks, $\xi_{j,t}$. When these shocks are more variable, products become more vertically differentiated. Over the range of values we investigate, increases in the standard deviation of the demand shocks increase the Lipschitz constant. The third column of Table 3 changes the number of markets. The number of markets has little impact on the Lipschitz constant. Finally, the fourth column of Table 3 increases the mean of the intercept, $E\left[\beta_i^0\right]$, which changes the value of the inside goods relative to the outside good. As the inside good share increases, the Lipschitz constant increases.

7.4 Monte Carlo: Varying the Lipschitz Constant

Having established that different parameter settings can change the Lipschitz constant of the contraction mapping, we now explore whether there is an implication for execution time. We compare performances as we vary the mean of the intercept, $E\left[\beta_i^0\right]$, from -1.9 to 3.1. As we saw in Table 3, increasing $E\left[\beta_i^0\right]$ makes the Lipschitz constant higher. For each scenario, we run 20 replications of the data. For each data replication, we estimate the GMM parameters using our two numerically-accurate algorithms, NFP with a tight inner loop and MPEC. Because a local optimization routine may only converge to a local minimum, we follow what a rigorous researcher should do and we use multiple starting values for each algorithm and fake dataset. We run each algorithm five times per replication, using five independently-drawn starting values. We take the final point estimates for each algorithm as the run with the lowest objective function value. In all cases, the lowest objective function corresponded to a case where the algorithm reported that an optimal local solution had been found. We assess the estimates by looking at the own-price elasticities, computed as a mean across products within each market and then across markets. For each algorithm, we report the total CPU time required for all five starting values. The results are reported in Table 4. All numbers in Table 4 are means across the 20 replications.

Turning to Table 4, we can see that our numerical theory prediction holds in practice. As expected, NFP with a tight inner-loop tolerance and MPEC converge in all scenarios. We also find that MPEC and NFP almost always generate identical point estimates, as one would expect since they are statistically the same estimator (Theorem 6). Across the 20 runs, MPEC and NFP produce identical estimates for the first four values of $E\left[\beta_i^0\right]$. For the last two values of $E\left[\beta_i^0\right]$, MPEC and NFP are nearly identical. With only five starting values, by happenstance in one or two of the 20 replications, MPEC and NFP found different local minima. Using ten or fifteen replications will likely make MPEC and NFP always find the same global minimum. We compute the root mean-squared error (RMSE) and the bias of the own-price elasticities. For a parameter θ_1, the bias is $E\left[\hat{\theta}_1\right] - \theta_1$, where θ_1 is the true value and the expectation is taking over many estimates with independent samples. Likewise, the RMSE is $\sqrt{E\left[\left(E\left[\hat{\theta}_1\right] - \theta_1\right)^2\right]}$. In all cases, the RMSE is low and the bias is moderate at around 0.2 off of a base elasticity of around -12, suggesting that the BLP estimator is capable of recovering true demand elasticities. To our knowledge, this is the most comprehensive Monte Carlo performed on BLP in the literature.

Run times vary dramatically for NFP tight with the level of the Lipschitz constant. For the low Lipschitz case with $E\left[\beta_i^0\right] = -1.9$, the average run time across the 20 replications (using five starting values for each replication) is roughly 17 minutes for NFP and for MPEC. However, as we increase the intercept, we see the run times for NFP increase, while the run times for MPEC change little. When $E\left[\beta_i^0\right] = 3.1$, the highest Lipschitz case, a single run with five starting values of NFP takes, on average, 60 minutes, whereas MPEC takes only 13 minutes. Remarkably, in this example the speed of MPEC actually increases slightly as $E\left[\beta_i^0\right]$ decreases, although the big effect is, as expected, the decrease in the speed of NFP.

One might not be so impressed if these numbers represent datasets which have unusual outside good shares. Our mean own-price elasticity of around -12 is representative of a pretty competitive industry, say one producing a less differentiated product. The shares of the outside good range from 90% to 47%, all large and realistic values. 90% could represent the US auto industry sampled at an annual rate, where most families do not buy a car every year. 47% could represent purchasing staples, such as orange juice, at weekly shopping trips.

Thus, as predicted by the numerical theory, it is easy to find cases where NFP tight could be extremely slow to run due to the slow rate of convergence of the inner loop. In contrast, MPEC is fairly robust in terms of run times across scenarios. This relationship to run time highlights our earlier concern about the choice of the inner-loop tolerance. For real applications with many more products and/or markets (e.g. 25 products and 450 market/quarters in Nevo (2000, 2002) and 250 products and 10 years in BLP (1995)), run times could be considerably slower than in our Monte Carlo experiments with only 25 products and 50 markets. As we demonstrated previously, loosening the inner-loop tolerance to speed the convergence of the inner loop could prevent the outer-loop optimization from converging. This, in turn, might lead the researcher to loosen the outer-loop tolerance, which could produce highly variable point estimates that may not even constitute local minima. We therefore recommend MPEC as a safer and more reliable algorithm for the estimation of the BLP GMM estimator.

7.5 Varying the Number of Markets

In the previous section, we demonstrated that MPEC has a speed advantage over NFP when the Lipschitz constant is high. However, some readers may be concerned that MPEC may not be practical as one increases the number of products or the number of markets. The reason is that there is one nuisance optimization parameter, $\xi_{j,t}$, for each product j and market t combination. As the number of markets T (or the number of products J) increases, there will be more $\xi_{j,t}$'s over which to optimize

Table 4: Monte Carlo Results Varying the Lipschitz Constant

Intercept	Lipschitz	Implementation	Runs Converged	CPU Time (s)	Elasticities			Outside
$E\left[\beta_i^0\right]$	Constant		(fraction)		Bias	RMSE	Value	Share
-1.9	0.789	NFP tight	1	1012.9	-0.200	0.265	-12.00	0.900
		MPEC	1	981.0	-0.200	0.265	-12.00	0.900
-0.9	0.858	NFP tight	1	1365.9	-0.203	0.266	-11.98	0.845
		MPEC	1	1015.2	-0.203	0.266	-11.98	0.845
0.1	0.913	NFP tight	1	1608.4	-0.205	0.266	-11.97	0.775
		MPEC	1	1001.4	-0.205	0.266	-11.97	0.775
1.1	0.952	NFP tight	1	2057.7	-0.201	0.256	-11.96	0.687
		MPEC	1	832.4	-0.201	0.256	-11.96	0.687
2.1	0.976	NFP tight	1	2544.8	-0.202	0.256	-11.95	0.583
		MPEC	1	810.2	-0.199	0.254	-11.96	0.583
3.1	0.989	NFP tight	1	3730.3	-0.195	0.252	-11.97	0.472
		MPEC	1	767.5	-0.202	0.254	-11.96	0.472

There are 20 replications for each experiment. Each replication uses five starting values to ensure a global minimum is found. The NFP-tight implementation has $\epsilon_{\text{in}} = 10^{-14}$ and $\epsilon_{\text{out}} = 10^{-6}$. There is no inner loop in MPEC; $\epsilon_{\text{out}} = 10^{-6}$ and $\epsilon_{\text{feasible}} = 10^{-6}$. The same 100 simulation draws are used to generate the data and to estimate the model.

and, correspondingly, more constraints. The next set of Monte Carlo experiments compare estimation with differing numbers of markets to see whether MPEC's speed advantage is related to having a small number of demand shocks.

Table 5 returns to the base specification, and varies only the number of markets, T. The mean intercept is $E\left[\beta_i^0\right] = 1.1$. As the number of markets increases, not surprisingly both methods take longer. MPEC takes only $\frac{1373}{555} = 40\%$ of the time as NFP for $T = 25$, 41% of the time for $T = 50$, and only 27% of the time for $T = 100$. We conclude that, in this example, the performance advantage of MPEC over NFP actually increases as the number of demand shocks increase. We do not have a theoretical prediction that MPEC's speed advantage should increase, but we suspected MPEC's speed advantage would not decrease. The modified Newton method used for MPEC has a quadratic rate of convergence whereas NFP has a linear rate of convergence for the inner loop. This means that MPEC should have a fairly easy time accommodating more parameters. Keep in mind that we report the sparsity pattern of the constraints to our solver for MPEC: the solver knows demand shocks for market 1 do not enter the market share equations for market 2.

7.6 Speed Comparisons of MPEC and NFP Using Nevo's Cereal Data

One potential criticism of our analysis above is that our Monte Carlo experiments were based on better-quality data than typical field datasets. In section 5.3, we used NFP with a tight tolerance to establish the reliability of the BLP GMM estimator for Nevo's cereal data. We now compare the speed of NFP and MPEC on this data. Like NFP, MPEC converged to the same local minimum with

Table 5: Monte Carlo Results Varying the Number of Markets

# Markets T	Lipschitz Constant	Implementation	Runs Converged (fraction)	CPU Time (s)	Elasticities			Outside
					Bias	RMSE	Value	Share
25	0.903	NFP tight	1	1372.9	-0.265	0.385	-12.16	0.640
		MPEC	1	555.2	-0.269	0.389	-12.16	0.640
50	0.952	NFP tight	1	2060.6	-0.201	0.256	-11.96	0.687
		MPEC	1	839.0	-0.201	0.256	-11.96	0.687
100	0.956	NFP tight	1	8068.2	-0.092	0.174	-12.30	0.893
		MPEC	1	2143.6	-0.106	0.225	-12.29	0.893

There are 20 replications for each experiment. Each replication uses five starting values to ensure a global minimum is found. The NFP-tight implementation has $\epsilon_{\text{in}} = 10^{-14}$ and $\epsilon_{\text{out}} = 10^{-6}$. There is no inner loop in MPEC; $\epsilon_{\text{out}} = 10^{-6}$ and $\epsilon_{\text{feasible}} = 10^{-6}$. The same 100 simulation draws are used to generate the data and to estimate the model.

an objective function value of 4.5615 for 48 out of 50 starting values. For only two of the runs, MPEC converged to a different local minimum with a higher objective-function value. In terms of run time for one starting value, we find that MPEC required an average CPU time of only 544 seconds whereas NFP required an average CPU time of 763 seconds. In short, the relative performance of MPEC and NFP documented in our Monte Carlo experiments appears to hold in the context of field data.

8 Other Computational Issues with BLP

8.1 Standard Errors

After obtaining point estimates, researchers need to compute standard errors to assess precision. Berry, Linton and Pakes (2004, Theorem 2) describe the sampling distribution of the BLP GMM estimator for fixed T and $J \to \infty$. As NFP and BLP are two computational implementations for the same estimator (they produce exactly the same objective function values and parameter estimates), both methods have the same sampling distribution.

One of the components of this formula requires derivatives of the mean of the moment conditions with respect to θ, or

$$\frac{\partial \left(\frac{1}{T} \sum_{t=1}^{T} \sum_{j=1}^{J} \xi_{j,t} (\theta)' \, z_{j,t} \right)}{\partial \theta}.$$

This requires differentiating the inner loop. Berry, Linton, and Pakes (page 632) suggest numerical derivatives (finite derivatives) as one method of computing an estimate of this derivative. However, any attempt to numerically differentiate an inner loop has the potential to introduce substantial numerical error: numerical derivatives are often numerically inaccurate even when the function being differentiated itself has little numerical error. A more numerically accurate approach is to program the derivatives. These derivatives are found, for example, in the appendix of Nevo (2000b). The components of the NFP derivatives are also components of the MPEC derivatives, so coding the MPEC derivatives makes it easy to code the standard errors.[26] In the interests of simplicity and commonality across researchers, we recommend users conducting $J \to \infty$ asymptotics code the asymptotic distribution in Theorem 2 of Berry, Linton and Pakes, using closed form derivatives. Users conducting $T \to \infty$ asymptotics and using a large enough number of simulation draws can use the standard GMM asymptotic-variance formula.

8.2 Nonnegativity Constraints on Parameters

BLP (1995) and most subsequent empirical work uses a set of independent normal distributions for $F_\beta(\beta;\theta)$, the distribution of the random coefficients. Under normality, θ includes the standard deviation of each product characteristic's random coefficient. The normal is a symmetric distribution. Therefore, if a guess for the standard deviation of characteristic 1's random coefficient is σ_1, $-\sigma_1$ should produce the same objective-function value for NFP and both the same objective function and constraint values for MPEC. Any failure of this equivalence of σ_1 and $-\sigma_1$ under normality results from simulation error: (3) is not an accurate approximation to (2). Disregarding simulation error, the model is not identified unless the researcher constrains each standard-deviation parameter to be nonnegative. If one of the standard-deviation parameters is in truth zero, then Andrews (2002) shows how to conduct asymptotically valid hypothesis tests. The limiting distribution of the parameter on the boundary will be half-normal, as we know a standard deviation cannot be negative.

9 Extension: Maximum Likelihood Estimation

In this section, we outline how a researcher would adapt MPEC to a likelihood-based estimation of random-coefficients-logit demand. Some researchers prefer to work with likelihood-based estimators and, more specifically, with Bayesian MCMC estimators (c.f. Yang et al 2003 and Jiang et al. 2008) based on the joint density of observed prices and market shares.[27]Besides efficiency advantages, the ability to evaluate the likelihood of the data could be useful for testing purposes. The trade-off relative to GMM is the need for additional modeling structure which, if incorrect, could lead to biased

[26]Because MPEC and NFP are the same estimator, it is not necessary to refer to results on equality-constrained estimators. Still, for users who adopt MPEC, it may be possible to use the constrained distributions derived for GMM in Andrews (2002). Such a procedure requires simulating the asymptotic distribution: a realization of a normal random variable is drawn and then a constrained-optimization problem is solved for each draw. We examined the results on extremum estimators with equality constraints in Gourieroux and Monfort (1995, Chapter 10). They derive the distribution of the constrained estimators as a function of the unconstrained estimators. For MPEC, the finite-sample objective function without constraints will always be minimized at $\xi = 0$. Therefore, the unconstrained limiting distribution is degenerate, and the proof technique in Gourieroux and Monfort does not apply.

[27]One can also think of Jiang et al. (2008) as an alternative algorithm for finding the parameters. The MCMC approach is a stochastic search algorithm that might perform well if the BLP model produces many local optima because MCMC will not be as likely to get stuck on a local flat region. Because our goal is not to study the role of multiple local minima, we do not explore the properties of a Bayesian MCMC algorithm.

parameter estimates. Like GMM, the calculation of the density of market shares still requires inverting the system of market share equations. Once again, MPEC can be used to circumvent the need for inverting the shares, thereby offsetting a layer of computational complexity and a potential source of numerical error. Note that we focus herein on classical MLE and not on a Bayesian approach. Below we outline the estimation of a limited information approach that models the data-generating process for prices in a "reduced form" (this motivation is informal as we do not specify a supply-side model and solve for a reduced form). However, one can easily adapt the estimator to accommodate a structural (full-information) approach that models the data-generating process for supply-side variables, namely prices, as the outcome of an equilibrium in a game of imperfect competition (assuming the equilibrium exists and is unique).

Recall that the system of market shares is defined as follows:

$$s_j\left(x_t, p_t, \xi_t; \theta\right) = \int_\beta \frac{\exp\left(\beta^0 + x'_{j,t}\beta^x - \beta^p p_{j,t} + \xi_{j,t}\right)}{1 + \sum_{k=1}^{J} \exp\left(\beta^0 + x'_{k,t}\beta^x - \beta^p p_{k,t} + \xi_{k,t}\right)} dF_\beta\left(\beta; \theta\right). \tag{11}$$

We assume, as in a triangular system, that the data-generating process for prices is

$$p_{j,t} = z'_{j,t}\gamma + \eta_{j,t}, \tag{12}$$

where $z_{j,t}$ is a vector of price-shifting variables and $\eta_{j,t}$ is a mean-zero, i.i.d. shock. To capture the potential endogeneity in prices, we assume the supply and demand shocks have the following joint distribution: $\left(\xi_{j,t}, \eta_{j,t}\right)' \equiv u_{j,t} \sim N(0, \Omega)$ where $\Omega = \begin{bmatrix} \sigma_\xi^2 & \sigma_{\xi,\eta} \\ \sigma_{\xi,\eta} & \sigma_\eta^2 \end{bmatrix}$.

The system defined by equations (11) and (12) has the joint density function

$$f_{s,p}\left(s_t, p_t; \Theta\right) = f_{\xi|\eta}\left(s_t \mid x_t, p_t; \theta, \Omega\right) \left|J_{\xi \rightarrow s}\right| f_\eta\left(p_t \mid z_t; \gamma, \Omega\right),$$

where $\Theta = \left(\theta, \gamma, \sigma_\xi^2, \sigma_{\xi,\eta}, \sigma_\eta^2\right)$ is the vector of model parameters, $f_{\xi|\eta}(\cdot|\cdot)$ is the marginal density of ξ conditional on η, $f_\eta(\cdot|\cdot)$ is a Gaussian density with variance σ_η^2, and $J_{\xi\rightarrow s}$ is the Jacobian matrix corresponding to the transformation of variables of $\xi_{j,t}$ to shares. The density of $\xi_{j,t}$ conditional on $\eta_{j,t}$ is

$$f_{\xi|\eta}\left(s_t \mid x_t, p_t; \theta, \Omega\right) = \prod_{j=1}^{J} \frac{1}{\sqrt{2\pi}\sigma_\xi\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2}\frac{\left(\xi_{j,t} - \rho\frac{\sigma_\xi}{\sigma_\eta}\eta_{j,t}\right)^2}{\sigma_\xi^2\left(1-\rho^2\right)}\right).$$

Note that the evaluation of $\xi_{j,t}$ requires inverting the market share equations, (2).

The element $J_{j,k}$ in row l and column k of the Jacobian matrix, $J_{\xi\to s}$, is

$$J_{j,l}=\begin{cases}\int_{\alpha,\beta}\left(1-\frac{\exp\left(\beta^0+x_{j,t}'\beta^x-\beta^p p_{j,t}+\xi_{j,t}\right)}{1+\sum_{k=1}^J\exp\left(\beta^0+x_{k,t}'\beta^x-\beta^p p_{k,t}+\xi_{k,t}\right)}\right)\frac{\exp\left(\beta^0+x_{j,t}'\beta^x-\beta^p p_{j,t}+\xi_{j,t}\right)}{1+\sum_{k=1}^J\exp\left(\beta^0+x_{k,t}'\beta^x-\beta^p p_{k,t}+\xi_{k,t}\right)}dF\left(\theta\right) & , \quad j=l \\ \\ -\int_{\alpha,\beta}\frac{\exp\left(\beta^0+x_{j,t}'\beta^x-\beta^p p_{j,t}+\xi_{j,t}\right)}{1+\sum_{k=1}^J\exp\left(\beta^0+x_{k,t}'\beta^x-\beta^p p_{k,t}+\xi_{k,t}\right)}\frac{\exp\left(\beta^0+x_{l,t}'\beta^x-\beta^p p_{l,t}+\xi_{l,t}\right)}{1+\sum_{k=1}^J\exp\left(\beta^0+x_{k,t}'\beta^x-\beta^p p_{k,t}+\xi_{k,t}\right)}dF\left(\theta\right) & , \quad j\neq l\end{cases}.$$

Standard maximum likelihood estimation would involve searching for parameters, Θ^{LISML}, that maximize the following log-likelihood function

$$l\left(\Theta\right)=\sum_{t=1}^{T}\log\left(f_{s,p}\left(s_t,p_t;\Theta\right)\right).$$

This would consist of a nested inner-loop to compute the demand shocks, $\xi_{j,t}$, via numerical inversion (the NFP contraction-mapping).

The equivalent MPEC approach entails searching for the vector of parameters (Θ,ξ) that maximizes the constrained optimization problem

$$\begin{array}{rcl} l^{\text{MPEC}}\left(\Theta,\xi\right) & = & \sum_{t=1}^{T}\log\left(f_{\xi|\eta}\left(s_t\mid x_t,p_t;\theta,\Omega\right)\left|J_{\xi\to s}\right|f_{\eta}\left(p_t\mid z_t;\gamma,\Omega\right)\right) \\ \text{subject to} & & s(\xi;\theta)=S \end{array}. \quad (13)$$

10 Extension: Dynamic Demand Models

Starting with Melnikov (2000), a new stream of literature has considered dynamic analogs of BLP with forward-looking consumers making discrete choice purchases of durable goods (Nair 2007, Gordon 2007, Carranza 2008, Gowrisankaran and Rysman 2008, Dubé, Hitsch and Chintagunta 2008, Lee 2008, Schiraldi 2008). The typical implementation involves a nested-fixed-point approach with two nested inner loops. The first inner loop is the usual numerical inversion of the demand system to obtain the demand shocks, ξ. The second inner loop is the iteration of the Bellman equation to obtain the consumer's value function. In this section, we describe how MPEC can once again serve as a computationally more attractive solution than NFP,

As an example, we work with a simple model of demand for a durable good with falling prices over time, with two competing products. Prices often fall over time for newly-introduced goods. There is a mass M of potential consumers at date $t=1$. Consumers are assumed to drop out of the market once they make a purchase. Abstracting from supply-side specifics, we assume that prices evolve over time as a function of the lagged prices of both firms according to the rule

$$p_{j,t} = \rho_{j,0} + \rho_{j,1} p_{j,t-1} + \rho_{j,2} p_{-j,t-1} + \psi_{j,t} = p'_{t-1}\rho_j + \psi_{j,t},\ j = 1, ..., 2 \tag{14}$$

where $\psi_{j,t}$ is a random supply shock. For the remainder of our discussion, we assume that this supply shock is jointly distributed with the demand shock: $(\xi_{j,t}, \psi_{j,t}) \sim N(0, \Omega)$ and is independent across times, firms and markets. We assume that consumers have rational expectations in the sense that they use the true price process (14) to forecast future prices.

On the demand side, forward-looking consumers now have a real option associated with not purchasing because they can delay adoption to the future, when prices are expected to be lower. A consumer r's expected value of waiting is[28]

$$\begin{aligned} v_0^r(p_t;\theta^r) &= \delta \int \max \left\{ \begin{array}{c} v_0^r(p_t'\rho_j + \psi; \theta^r) + \epsilon_0 \\ \max_j \left\{ \beta_j^r - \alpha^r (p_t'\rho_j + \psi) + \xi_j + \epsilon_j \right\} \end{array} \right\} dF_\epsilon(\epsilon) dF_{\psi,\xi}(\psi,\xi) \\ &= \delta \int \left(\log \left(\exp\left(v_0^r(p_t'\rho_j + \psi;\theta^r)\right) + \sum_j \exp\left(\beta_j^r - \alpha^r(p_t'\rho_j + \psi) + \xi_j\right) \right) \right) dF_{\psi,\xi}(\psi,\xi). \end{aligned} \tag{15}$$

To simplify the calculation of the expected value of waiting, we approximate it with Chebyshev polynomials (Judd 1998). We outline the Chebyshev approximation in Appendix C. Note that by focusing on the expected value of waiting, rather than the consumer's value function, we are merely exploiting the special structure of this model.

We use a discrete distribution with R mass points to characterize the consumer population's tastes at date $t = 1$,

$$\theta^h \equiv \begin{bmatrix} \beta^h \\ \alpha^h \end{bmatrix} = \begin{cases} \theta^1, & \Pr(1) = \lambda_1 \\ \vdots & \vdots \\ \theta^R, & \Pr(R) = 1 - \sum_{r=1}^{R-1} \lambda_r \end{cases}.$$

This heterogeneity implies that certain types of consumers will systematically purchase earlier than others. The mass of consumers of a given type r at the beginning of period t, M_t^r, is

$$M_t^r = \begin{cases} M\lambda_r & ,\ t = 1 \\ M_{t-1}^r S_0^r(X_{t-1};\Theta^r) & ,\ t > 1 \end{cases}.$$

In a given period t, the market share of product j is

$$s_j(p_t;\theta) = \sum_{r=1}^{R} \lambda_{t,r} \frac{\exp(\beta_j^r - \alpha^r p_{j,t} + \xi_{j,t})}{\exp\left(v_0^r(p_t;\theta^r)\right) + \sum_{k=1}^{J} \exp\left(\beta_k^r - \alpha^r p_{k,t} + \xi_{k,t}\right)},\ j = 1, ..., 2 \tag{16}$$

[28]Here we make the normalization that the location parameter of the Type I Extreme Value distribution equals -0.577.

where

$$\lambda_{t,r} = \begin{cases} \lambda_r & t = 1 \\ \frac{M_t^r}{\sum_r M_t^r} & , t > 1 \end{cases}$$

is the probability mass associated with type r consumers still in the market at date t. The finite-types assumption eases dynamic programming because there is only one unknown value-of-waiting function for each type.

The empirical model consists of the system (14) and (16), which we write more compactly as

$$u_t \equiv \begin{bmatrix} \psi_t \\ \xi_t \end{bmatrix} = \begin{bmatrix} p_t - p_{t-1}'\rho \\ s^{-1}(p_t, S_t; \Theta) \end{bmatrix}.$$

The multivariate normal distribution of u_t induces the density on the observable outcomes, $Y_t = (p, S_t)$,

$$f_Y(Y_t; \theta, \rho, \Omega) = \frac{1}{(2\pi)^J |\Omega|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} u_t' \Omega^{-1} u_t\right) |J_{t,u \to Y}|$$

where $J_{t,u\to Y}$ is the $(2J \times 2J)$ Jacobian matrix corresponding to the transformation-of-variables from u_t to Y_t. We provide the derivation of the Jacobian in Appendix D.

An NFP approach to maximum likelihood estimation of the model parameters amounts to solving the optimization problem

$$\max_{\{\theta,\rho,\Omega\}} \prod_{t=1}^{T} f_Y(Y_t; \theta, \rho, \Omega). \tag{17}$$

This problem nests two inner loops. For each stage of the outer loop to maximize the likelihood function in (17), one needs to solve for a fixed point of the contraction mapping, (15), in order to obtain the expected value of waiting. In addition, one needs to solve the fixed point of the BLP contraction mapping, 5, to compute the demand shocks ξ_t (i.e. the inversion). Numerical error from both these inner loops can potentially propagate into the outer loop. Thus, the numerical concerns regarding inner loop convergence tolerance discussed for static BLP are exacerbated with dynamic analogs of BLP.

Let D be the support of the state variables. An MPEC approach to maximum likelihood estimation of the model parameters amounts to solving the optimization problem

$$\begin{array}{lll} \max\limits_{\{\theta,\rho,\Omega,\xi,v\}} & \prod\limits_{t=1}^{T} \frac{1}{(2\pi)^J |\Omega|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} u_t' \Omega_u^{-1} u_t\right) |J_{t,u\to Y}| \\ \text{subject to} & s(\xi_t; \theta) = S_t \, \forall \, t = 1, \ldots, T \\ \text{and} & v_0^r(p_d) = \delta \log \left(\begin{array}{c} \exp\left(v_0^r(p_d'\rho_j + \psi)\right) + ... \\ \sum\limits_j \exp\left(\beta_j^r - \alpha^r \left(p_d'\rho_j + \psi\right) + \xi_j\right) \end{array} \right) dF_{\psi,\xi}(\psi,\xi) \\ & \forall \, d \in D, \, r = 1, \ldots, R. \end{array}$$

In this formulation, we now optimize over the demand shocks, ξ, and the expected value of waiting evaluated at each point, $\nu^r(p_d)$. In this case, $\mathrm{D} \subset \mathbb{R}_+^2$, which is the support of the two products' prices. While this approach increases the number of parameters in the outer-loop optimization problem substantially compared to NFP, MPEC completely eliminates the two inner loops. Note that we reduce the dimension of this problem substantially by using Chebyshev approximation. Rather than searching over the value function at each point in a discretized state space, we search over the weights of the Chebyshev approximation.

To assess the relative performance of MPEC versus NFP in the context of our dynamic durable goods example, we construct the following Monte Carlo experiments. In the first experiment, we assume there is only a single consumer type, $R = 1$. It is easy to show that in this case, ξ_t can be computed analytically by log-linearizing the market shares, (16).[29] We begin with this case because it only involves a nested call to the calculation of the expected value of waiting. Below we will allow for more consumer types to see what happens when we also require a nested call to the numerical inversion of the shares. We assume that the consumers' preferences are: $(\beta_1, \beta_2, \alpha) = (4, 3, -1)$. It is straightforward to show that the speed of the contraction mapping associated with the consumer's expected value of waiting is related to the discount factor. Therefore, we compare performance with the two different discount factors, $\delta = 0.99$ and $\delta = 0.96$. For an annual interest rate of 4%, $\delta = 0.96$ would be the corresponding annual discount rate and $\delta = .99$ would be the corresponding quarterly discount rate. We assume that the density of prices has the transition rules

$$\begin{bmatrix} p_{1,t} & = & 5 + .8p_{1,t-1} + 0.0p_{2,t-1} + \psi_{1,t} \\ p_{2,t} & = & 5 + 0.0p_{1,t-1} + 0.8p_{2,t-1} + \psi_{2,t} \end{bmatrix}.$$

Note how the lagged price of product 2 could potentially affect the price of product 1, and vice versa. Finally, we assume the supply and demand shocks satisfy $(\psi_{j,t}, \xi_{j,t}) \sim N\left(0, \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}\right)$ and are independent across markets and time periods. For our Chebyshev approximation, we use six nodes and a fourth order polynomial. For the NFP algorithm, we use an inner loop tolerance of 10^{-14} for the calculation of the expected value of waiting.

It is very difficult to derive analytic expressions for the Jacobian of the outer-loop optimization associated with dynamic BLP, both under NFP and MPEC.[30] As we discussed in section 4.5, the use of numerical derivatives introduces yet another source of numerical error into the outer-loop optimization. However, due to its formulation as a standard constrained-optimization problem, the MPEC formulation can potentially exploit automatic differentiation to obtain exact derivatives for

[29]See Berry (1994) on how to invert the demand shocks in the homogeneous logit model.

[30]We computed the derivatives in Mathematica, and they ran to 100 pages of algebraic expressions.

Table 6: Monte Carlo Results for Dynamic BLP with One Consumer Type for $\delta = 0.96$: NFP versus MPEC

	MPEC		NFP		
Speeds	335.55 secs.		553.50 secs.		
Parameters	Mean	RMSE	Mean	RMSE	Truth
Utility intercept product 1	3.9557	0.1780	3.9556	0.1780	4.0000
Utility intercept product 2	2.9572	0.2015	2.9572	0.2015	3.0000
Utility price coefficient, type 1	-1.0030	0.0101	-1.0030	0.0101	-1.0000
Price, product 1, constant	0.2111	0.0345	0.2111	0.0345	0.2000
Price, product 1, lagged price of product 1	0.7962	0.0136	0.7962	0.0136	0.8000
Price, product 1, lagged price of product 2	0.0026	0.0098	0.0026	0.0098	0.0000
Price, product 2, constant	0.2071	0.0378	0.2071	0.0378	0.2000
Price, product 2, lagged price of product 1	0.0037	0.0168	0.0037	0.0168	0.0000
Price, product 2, lagged price of product 2	0.7935	0.0156	0.7935	0.0156	0.8000
Demand shocks, Cholesky variance term	0.9958	0.0173	0.9958	0.0173	1.0000
Covariance btw supply and demand, Cholesky variance term	0.5015	0.0215	0.5015	0.0215	0.5000
Supply shocks, Cholesky variance term	0.8647	0.0152	0.8647	0.0152	0.8660

There are 20 replications for each of MPEC and NFP. The same fake data are used for both MPEC and NFP. Each replication uses five starting values to do a better job at finding a global minimum. The NFP implementation has $\epsilon_{\text{in}}^{\xi} = 10^{-14}$, $\epsilon_{\text{in}}^{V} = 10^{-14}$ and $\epsilon_{\text{out}} = 10^{-6}$. There is no inner loop in MPEC; $\epsilon_{\text{out}} = 10^{-6}$ and $\epsilon_{\text{feasible}} = 10^{-6}$. The data have $T = 50$ periods and $M = 20$ distinct markets. Each market has two competing products. The Chebyshev regression approximation to the value function uses a fourth-order polynomial and five interpolation nodes. The numerical integration of future states uses Gauss-Hermite quadrature with three nodes. NFP uses numerical derivatives, as coding the derivatives of dynamic BLP is infeasible for many problems and it is not clear automatic differentiation works with nested inner loops. MPEC uses automatic differentiation in the form of the package MAD.

the outer loop (Griewank and Corliss 1992). To the best of our knowledge, it would be non-standard to apply automatic differentiation to an NFP problem because of the nested calls to fixed-point calculations. Therefore, in our Monte Carlo experiments, we compare the performance of NFP using numerical differentiation and MPEC using automatic differentiation.[31]

Results from twenty replications of this first experiment are reported in Table 6, where we use the discount factor δ=0.96. We report the average point estimate and RMSE associated with each of the structural parameters, for MPEC and NFP respectively. With very tight inner-loop settings and allowing for 5 different starting values, we find that MPEC and NFP produce identical point estimates. Indeed, inspection of the replications found that MPEC and NFP found the same solution for all replications. However, in terms of speed, MPEC is roughly 60% faster in terms of CPU time than NFP.

In Table 7, we run another twenty replications of the same one-type model using a discount factor of $\delta = 0.99$, corresponding to quarterly data for a possibly fast-changing market. Interestingly, even with five starting values per replication, MPEC appears to perform slightly better overall in terms of

[31]We use the MAD (MATLAB Automatic Differentiation) package, which is part of TOMLAB.

Table 7: Monte Carlo Results for Dynamic BLP with One Consumer Type for $\delta = 0.99$: NFP versus MPEC

	MPEC		NFP		
Speeds	671.49 secs.		1295.50 secs.		
Parameters	Mean	RMSE	Mean	RMSE	Truth
Utility intercept product 1	4.0684	0.7235	3.3907	1.7473	4.0000
Utility intercept product 2	3.0692	0.7316	2.3913	1.7844	3.0000
Utility price coefficient, type 1	-0.9885	0.0380	-0.9987	0.0152	-1.0000
Price, product 1, constant	0.1929	0.0682	0.2171	0.0655	0.2000
Price, product 1, lagged price of product 1	0.8170	0.0532	0.8022	0.0546	0.8000
Price, product 1, lagged price of product 2	-0.0044	0.0295	0.0000	0.0520	0.0000
Price, product 2, constant	0.1770	0.1102	0.2058	0.0813	0.2000
Price, product 2, lagged price of product 1	-0.0195	0.0557	-0.0065	0.0436	0.0000
Price, product 2, lagged price of product 2	0.8330	0.0860	0.8089	0.0585	0.8000
Demand shocks, Cholesky variance term	1.0139	0.0468	1.0053	0.0334	1.0000
Covariance btw supply and demand, Cholesky variance term	0.4985	0.0255	0.5050	0.0219	0.5000
Supply shocks, Cholesky variance term	0.8652	0.0152	0.8640	0.0159	0.8660

There are 20 replications for each of MPEC and NFP. The same fake data are used for both MPEC and NFP. Each replication uses five starting values to do a better job at finding a global minimum. The NFP implementation has $\epsilon_{\text{in}}^{\xi} = 10^{-14}$, $\epsilon_{\text{in}}^{V} = 10^{-14}$ and $\epsilon_{\text{out}} = 10^{-6}$. There is no inner loop in MPEC; $\epsilon_{\text{out}} = 10^{-6}$ and $\epsilon_{\text{feasible}} = 10^{-6}$. The data have $T = 50$ periods and $M = 20$ distinct markets. Each market has two competing products. The Chebyshev regression approximation to the value function uses a fourth-order polynomial and four interpolation nodes. The numerical integration of future states uses Gauss-Hermite quadrature with three nodes. NFP uses numerical derivatives, as coding the derivatives of dynamic BLP is infeasible for many problems and it is not clear automatic differentiation works with nested inner loops. MPEC uses automatic differentiation in the form of the package MAD.

RMSE, especially for the utility intercepts. Furthermore, MPEC is just under twice as fast at NFP in terms of CPU time.[32]

Thus far, we have established that MPEC is faster and, potentially, more reliable than NFP for the one-type case. We now run a final Monte Carlo using MPEC only to illustrate its applicability to a case with consumer heterogeneity. In particular, we allow for a second type of consumer with heterogeneity in the price sensitivity:

$$(\beta_1, \beta_2, \alpha)' = \begin{cases} (4, 3, -1) & \text{, with probability } \lambda{=}0.7 \\ (4, 3, -2) & \text{, with probability } (1-\lambda){=}0.3 \end{cases} .$$

This case now requires constraints corresponding both to the expected value of waiting for each of the $R = 2$ consumer types and for the share equations. To conserve on computer time, we use only

[32]Inspection of the replications found that for two replications of NFP, the optimization routine was not able to diagnose convergence for the greatest likelihood function value out of the five starting values. Indeed, in these two replications NFP found a worse objective function value than MPEC did. All replications of MPEC converged to a valid local maximum. However, for three of the twenty replications, MPEC converged to a worse solution than NFP. For $\delta = 0.99$ and unlike the $\delta = 0.96$ case, it appears five starting values are not enough to always find the same solution with MPEC and NFP.

Table 8: Monte Carlo Results for Dynamic BLP with Two Consumer Types and $\delta = 0.90$ MPEC Only

	MPEC		
Speeds	9397 secs.		
Parameters	Mean	RMSE	Truth
Utility intercept product 1	3.6604	0.5719	4.0000
Utility intercept product 2	2.6980	0.5287	3.0000
Utility price coefficient, type 1	-1.0159	0.0299	-1.0000
Utility price coefficient, type 2	-2.0369	0.2623	-2.0000
Frequency, type 1	0.7907	0.1016	0.7000
Price, product 1, constant	0.1894	0.0818	0.2000
Price, product 1, lagged price of product 1	0.7919	0.0333	0.8000
Price, product 1, lagged price of product 2	-0.0013	0.0274	0.0000
Price, product 2, constant	0.2283	0.0929	0.2000
Price, product 2, lagged price of product 1	-0.0013	0.0262	0.0000
Price, product 2, lagged price of product 2	0.7919	0.0341	0.8000
Demand shocks, Cholesky variance term	0.9001	0.4386	1.0000
Covariance btw supply and demand, Cholesky variance term	0.4537	0.4609	0.5000
Supply shocks, Cholesky variance term	0.6891	0.5616	0.8660

There are 20 replications. Each replication uses two starting values to do a better job at finding a global minimum. There is no inner loop in MPEC; $\epsilon_{\text{out}} = 10^{-6}$ and $\epsilon_{\text{feasible}} = 10^{-6}$. The data have $T = 50$ periods and $M = 5$ distinct markets. Each market has two competing products. The Chebyshev regression approximation to the value function uses a fourth-order polynomial and four interpolation nodes. The numerical integration of future states uses Gauss-Hermite quadrature with three nodes. The code uses automatic differentiation in the form of the package MAD.

$M = 5$ markets per replication. Results are reported in Table 8. Not only do we find that the parameters are recovered quite well, the average run time requires only 2.6 hours of CPU time. In short, these results are encouraging for MPEC as a practical approach to estimating dynamic BLP with unobserved heterogeneity in a broader context.

11 Conclusions

In this paper, we analyzed the numerical properties of the NFP approach proposed by BLP to estimate the random-coefficients-logit demand model. Theoretically, the NFP approach may be slow, as NFP's inner loop is only linearly convergent. NFP is also more vulnerable to error due to the inner loop. We showed the Lipschitz constant is a measure of an upper bound to the convergence rate of NFP's inner loop's contraction mapping. We numerically evaluated the Lipschitz constant for particular data-generating processes and showed when the inner loop is likely to be slow. A researcher is likely to use a loose inner loop tolerance when the speed of NFP is slow. Using both numerical theory and computational examples with both real and fake data, we showed that setting loose inner-loop tolerances can lead to incorrect parameter estimates and a failure of the optimization routine to report that it has converged. Using the cereal data, we showed a case where the estimates with multiple starting values always reported (more or less) the same wrong estimate.

We then proposed a new constrained-optimization formulation, MPEC, for estimating the random-coefficients-logit demand model. MPEC is quicker to compute and avoids numerical errors because it avoids repeatedly inverting the market shares equations numerically. It also allows the researcher to access state-of-the-art constrained optimization solvers.

To assess the practical aspects of MPEC versus NFP, we conducted a number of Monte Carlo experiments. In many instances, we found that the NFP approach could produce accurate estimates of the demand parameters in a reasonable amount of time – so long as it was implemented carefully. However, we find that NFP is comparatively quite slow under several data generating processes. We used the Lipschitz constant to identify cases where NFP was quite slow. Loosening the inner-loop tolerance to improve the speed of NFP, as is often done in practice, leads to a failure to converge unless the optimization method's tolerance is also significantly loosened, which can potentially lead to incorrect estimates.

In contrast, MPEC produces good estimates relatively quickly for all the data-generating processes we considered. The reason is that there is no inner loop in MPEC, so issues like a high Lipschitz constant (or slow contraction mapping) are irrelevant for MPEC. The speed of MPEC was roughly invariant across data generating-processes with different Lipschitz constants, in contrast with NFP. In principle, MPEC can also be used for demand models where there is a unique vector of demand shocks that rationalize the market shares, but no contraction mapping. Moreover, it can be applied to demand models where multiple vectors of demand shocks may solve the system of market-share equations. For the last case, NFP will certainly not work. MPEC is also useful for estimating the BLP demand model using maximum likelihood, where a Jacobian term involving the demand shocks must be computed.

As an extension, we adapt the MPEC approach to a new class of applications with forward-looking consumers. The relative advantage of MPEC is even stronger with dynamics because two inner loops must be solved: the dynamic-programming problem and the market-share inversion. This burdensome collection of three loops (optimization, market shares, dynamic programming) makes the traditional BLP approach nearly untenable in terms of computational time. Current work (Lee 2008, Schiraldi 2008) further extends the number of inner loops being solved in estimation. As demand models become richer, the computational benefits of MPEC over NFP become greater.

References

[1] Ackerberg, D.. and M. Rysman (2005), Unobserved Product Differentiation in Discrete Choice Models: Estimating Price Elasticities and Welfare Effects, *RAND Journal of Economics*, 36 (4):

771–788.

[2] Ackerberg, D., J. Geweke and J. Hahn. (2009) “Comments on 'Convergence Properties of the Likelihood of Computed Dynamic Models' by Fernandez-Villaverde, Rubio-Ramirez and Santos”, *Econometrica.*

[3] Andrews, D. W. K. (2002). Generalized Method of Moments Estimation When a Parameter is on a Boundary, *Journal of Business and Economic Statistics,* 20 (4), 530–544.

[4] Bajari, P., J. T. Fox and S. P. Ryan (2007), “Linear Regression Estimation of Discrete Choice Models with Nonparametric Distributions of Random Coefficients.” *American Economic Review*, 97(2), 459-463

[5] Bajari, P., J. T. Fox, K.-I. Kim and S. P. Ryan (2008), A Simple Nonparametric Estimator for the Distribution of Random Coefficients in Discrete Choice Models. Working paper.

[6] Berry, S. (1994), Estimating Discrete-Choice Models of Product Differentiation. *RAND Journal of Economics*, 25(2): 242–262.

[7] Berry, S. and P. A. Haile (2008), Nonparametric Identification of Multinomial Choice Models with Heterogeneous Consumers and Endogeneity, working paper.

[8] Berry, S., J. Levinsohn, and A. Pakes (1995), Automobile Prices in Market Equilibrium. *Econometrica*, 63(4): 841–890.

[9] Berry, S., J. Levinsohn, and A. Pakes (2004), “Differentiated Products Demand Systems from a Combination of Micro and Macro Data: The New Car Market.” *Journal of Political Economy*, 112(1):68-105.

[10] Berry, S., O. B. Linton, . and A. Pakes (2004), Limit Theorems for Estimating the Parameters of Differentiated Product Demand Systems. *Review of Economic Studies*, 71(3): 613–654.

[11] Byrd, R. H., J. Nocedal and R. A. Waltz (1999), An Interior Point Method for Large Scale Nonlinear Programming. SIAM J. Optimization, 9, 4, 877-990.

[12] Carranza, Juan Esteban. (2008) Product innovation and adoption in market equilibrium: The case of digital cameras. Working paper.

[13] Dahlquist, Germund and Åke Björck. (2008) *Numerical Methods in Scientific Computing.* SIAM, Philadelphia.

[14] Davis, Peter J. (2006) "The Discrete Choice Analytically Flexible (DCAF) Model of Demand for Differentiated Products," CEPR Discussion Papers 5880.

[15] Dubé, Jean-Pierre, Guenter Hitsch and Pradeep Chintagunta (2008). "Tipping and Concentration in Markets With Indirect Network Effects," Working paper, University of Chicago.

[16] Fox, Jeremy T. and Amit Gandhi (2008) Identifying Heterogeneity in Economic Choice and Selection Models Using Mixtures. Working paper.

[17] Gandhi, Amit (2008). "On the Nonparametric Foundations of Product Differentiated Demand Systems". Working paper.

[18] Gill, P. E., W. Murray and M. Wright (1981), *Practical Optimization*, Academic Press.

[19] Gourieroux, Christian and Alain Monfort (1995), *Statistics and Econometric Models,* Vol 1. Cambridge.

[20] Gowrisankaran, G. and M. Rysman (2007), Dynamics of Consumer Demand for New Durable Goods. Working paper.

[21] Griewank, A. and Corliss, G.F. (1992). "Automatic Differentiation of Algorithms: Theory, Implementation, and Application." DTIC Research Report ADA249012.

[22] Hausman, J.A. and D. A. Wise (1976). "A Conditional Profit Model for Qualitative Choice: Discrete Decisions Recognizing Interdependence and Heterogeneous Preferences." *Econometrica*, 46(2), 403–426.

[23] Hendel, I. and Aviv Nevo (2007) Measuring the Implications of Sales and Consumer Inventory Behavior, *Econometrica,* 74(16), 1637-1673.

[24] Jiang, Renna, Puneet Manchanda and Peter E. Rossi (2007), "Bayesian Analysis of Random Coefficient Logit Models Using Aggregate Data." Working Paper.

[25] Judd, Kenneth L. (1992) "Projection methods for solving aggregate growth models." *Journal of Economic Theory*, 58(2), 410–452.

[26] K. L. Judd (1998), *Numerical Methods in Economics.* MIT Press.

[27] C. T. Kelley (1995), *Iterative Methods for Linear and Nonlinear Equations.* SIAM, Philadelphia.

[28] C. T. Kelley (1999), *Iterative Methods for Optimization*, SIAM, Philadelphia.

[29] C. T. Kelley (2003), *Solving Nonlinear Equations with Newton's Method.* SIAM, Philadelphia.

[30] Knittel, Christopher R. and Konstantinos Metaxoglou (2008), "Estimation of Random Coefficient Demand Models: Challenges, Diculties and Warnings." Working paper.

[31] Lee, Robin S. "Vertical Integration and Exclusivity in Platform and Two-Sided Markets." Working paper.

[32] Mas-Colell, Andreu, Michael D. Whinston, and Jerry R. Green. (1995) *Microeconomic Theory.* Oxford University Press.

[33] McFadden, D. and K. Train (2000), Mixed MNL models for discrete response. *Journal of Applied Econometrics*, 15(5): 447–470.

[34] Melnikov, Oleg (2001) Demand for Differentiated Durable Products: The Case of the U.S. Computer Printer Market. Working paper.

[35] Nair, Harikesh. Intertemporal Price Discrimination with Forward-looking Consumers: Application to the US Market for Console Video-Games. *Quantitative Marketing and Economics*, 5(3), 239-292.

[36] Nevo, A. (2000a), Mergers with Differentiated Products: The Case of the Ready-to-Eat Cereal Industry. *RAND Journal of Economics*, 31(3): 395–421.

[37] Nevo, A.(2000b), A Practitioner's Guide to Estimation of Random Coefficients Logit Models of Demand, *Journal of Economics and Management Strategy*, 9(4): 513–548.

[38] Nevo, A. (2001), Measuring Market Power in the Ready-to-Eat Cereal Industry. *Econometrica*, 69(2): 307–342.

[39] Nocedal, J. and S. J. Wright (2006), *Numerical Optimization.* Springer, New York.

[40] Petrin, A. (2002), Quantifying the Benefits of New Products: The Case of the Minivan. *Journal of Political Economy*, 110:705–729.

[41] Petrin, A. and K. Train (2006), Control Function Corrections for Omitted Attributes in Differentiated Products Markets. Working paper.]

[42] Schiraldi, Pasquale (2008). "Automobile Replacement: a Dynamic Structural Approach." Working paper.

[43] Su, C.-L. and K. L. Judd (2007), Constrained Optimization Approaches to Estimation of Structural models. Working paper, CMS-EMS, Kellogg School of Management.

[44] van der Vaart, A. W. (2000). *Asymptotic Statistics.* Cambridge.

A Proofs

In all the proofs below, we assume the sum of the second order and other higher order terms in a Taylor seris expansion is bounded. This is a conventional assumption in the numerical optimization literature and allows us to use the big-O notation with a second order term, e.g., $O\left(\left\|\xi(\hat{\theta}(\epsilon_{in}),\epsilon_{\text{in}})-\xi(\hat{\theta}(\epsilon_{in}),0)\right\|^2\right)$ or $O\left(\left\|\hat{\theta}(\epsilon_{in})-\theta^*\right\|^2\right)$.

A.1 Proof of Theorem 3

By a Taylor series expansion of $Q(\xi)$ around $\xi(\theta,0)$, we have

$$\begin{aligned}
& Q\left(\xi\left(\theta,\epsilon_{\text{in}}\right)\right)-Q\left(\xi\left(\theta,0\right)\right)\\
= & \left[\tfrac{\partial Q(\xi)}{\partial\xi}\left|_{\xi=\xi(\theta,0)}\right.\right]'\left(\xi\left(\theta,\epsilon_{\text{in}}\right)-\xi\left(\theta,0\right)\right)+O\left(\left\|\xi\left(\theta,\epsilon_{\text{in}}\right)-\xi\left(\theta,0\right)\right\|^2\right)\\
& \text{and}\\
& \nabla_\theta Q\left(\xi\right)\left|_{\xi=\xi(\theta,\epsilon_{\text{in}})}\right.-\nabla_\theta Q\left(\xi\right)\left|_{\xi=\xi(\theta,0)}\right.\\
= & \left[\tfrac{\partial\nabla_\theta Q(\xi(\theta))}{\partial\xi}\left|_{\xi=\xi(\theta,0)}\right.\right]'\left(\xi\left(\theta,\epsilon_{\text{in}}\right)-\xi\left(\theta,0\right)\right)+O\left(\left\|\xi\left(\theta,\epsilon_{\text{in}}\right)-\xi\left(\theta,0\right)\right\|^2\right).
\end{aligned}$$

Because $\left\|\xi\left(\theta,\epsilon_{\text{in}}\right)-\xi\left(\theta,0\right)\right\|\leq\frac{L(\theta)}{1-L(\theta)}\epsilon_{\text{in}}$ by Theorem 1, and assuming both $\left\|\frac{\partial Q(\xi)}{\partial\xi}\left|_{\xi=\xi(\theta,0)}\right.\right\|$ and $\left\|\frac{\partial\nabla_\theta Q(\xi(\theta))}{\partial\xi}\left|_{\xi=\xi(\theta,0)}\right.\right\|$ are bounded, we obtain

$$\begin{aligned}
\left|Q\left(\xi\left(\theta,\epsilon_{\text{in}}\right)\right)-Q\left(\xi\left(\theta,0\right)\right)\right| &= O\left(\frac{L(\theta)}{1-L(\theta)}\epsilon_{\text{in}}\right)\\
\left\|\nabla_\theta Q\left(\xi\left(\theta\right)\right)\left|_{\xi=\xi(\theta,\epsilon_{\text{in}})}\right.-\nabla_\theta Q\left(\xi\left(\theta\right)\right)\left|_{\xi=\xi(\theta,0)}\right.\right\| &= O\left(\frac{L(\theta)}{1-L(\theta)}\epsilon_{\text{in}}\right).
\end{aligned}$$

A.2 Proof of Theorem 4

We define $\hat{\theta}(\epsilon_{in})$ to be the numerically incorrect estimates with the inner-loop tolerance ϵ_{in},

$$\hat{\theta}(\epsilon_{in}) = \arg\max_{\theta}\left\{Q\left(\xi\left(\theta,\epsilon_{\text{in}}\right)\right)\right\}.$$

Because $\nabla_\theta Q\left(\xi\right)\Big|_{\xi=\xi(\hat{\theta}(\epsilon_{in}),\epsilon_{\text{in}})}=0$, the application of the second result in Theorem 3 at $\hat{\theta}(\epsilon_{in})$ gives

$$\left\|\nabla_\theta Q\left(\xi\right)\Big|_{\xi=\xi(\hat{\theta}(\epsilon_{in}),0)}\right\|=O\left(\frac{L\left(\hat{\theta}(\epsilon_{in})\right)}{1-L\left(\hat{\theta}(\epsilon_{in})\right)}\epsilon_{\text{in}}\right). \qquad (18)$$

Note that we have evaluated the GMM objective function with no numerical error at the point $\hat{\theta}(\epsilon_{in})$ that minimizes the GMM objective function with inner loop numerical error.

Let $\tilde{\theta}$ be any value of the structural parameters near $\hat{\theta}(\epsilon_{in})$. By first the inverse triangle inequality, then the regular triangle inequality, and then finally a Taylor series expansion, we have

$$\begin{aligned}
& \left\| \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},\epsilon_{\text{in}}\right)} \right\| - \left\| \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right\| \\
\leq \; & \left\| \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},\epsilon_{\text{in}}\right)} - \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right\| \\
= \; & \left\| \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},\epsilon_{\text{in}}\right)} - \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},0\right)} + \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},0\right)} - \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right\| \\
\leq \; & \left\| \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},\epsilon_{\text{in}}\right)} - \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},0\right)} \right\| \\
& + \left\| \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},0\right)} - \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right\| \\
\leq \; & O\left(\tfrac{L(\tilde{\theta})}{1-L(\tilde{\theta})}\epsilon_{\text{in}}\right) + \left\| \nabla_\theta^2 Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right\| \left\| \tilde{\theta} - \hat{\theta}(\epsilon_{in}) \right\| + O\left(\left\| \tilde{\theta} - \hat{\theta}(\epsilon_{in}) \right\|^2 \right).
\end{aligned}$$

As we have assumed $\left\| \nabla_\theta^2 Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right\|$ is bounded, the second order term $O\left(\left\| \tilde{\theta} - \hat{\theta}(\epsilon_{in}) \right\|^2 \right)$ term can be ignored. By rearranging the above inequality, we obtain

$$\begin{aligned}
\left\| \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\tilde{\theta},\epsilon_{\text{in}}\right)} \right\| \; & \leq \; \left\| \nabla_\theta Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right\| + O\left(\tfrac{L(\tilde{\theta})}{1-L(\tilde{\theta})}\epsilon_{\text{in}}\right) + O\left(\left\| \tilde{\theta} - \hat{\theta}(\epsilon_{in}) \right\| \right) \\
& = \; O\left(\tfrac{L(\hat{\theta}(\epsilon_{in}))}{1-L(\hat{\theta}(\epsilon_{in}))}\epsilon_{\text{in}}\right) + O\left(\tfrac{L(\tilde{\theta})}{1-L(\tilde{\theta})}\epsilon_{\text{in}}\right) + O\left(\left\| \tilde{\theta} - \hat{\theta}(\epsilon_{in}) \right\|^2 \right) \\
& = \; O\left(\epsilon_{\text{in}}\right) + O\left(\left\| \tilde{\theta} - \hat{\theta}(\epsilon_{in}) \right\|^2 \right),
\end{aligned}$$

where the first equality uses (18).

A.3 Proof of Theorem 5

We define θ^* to be the true estimate when there are no inner-loop numerical errors ($\epsilon_{\text{in}} = 0$), i.e., $\theta^* = \arg\max_\theta \left\{ Q\left(\xi\left(\theta, 0\right)\right)\right\}$. First, we can quantify the bias between the numerically correct and incorrect objective function values, $Q\left(\xi(\theta(\hat{\epsilon_{in}}), \epsilon_{\text{in}})\right)$ and $Q\left(\xi(\theta^*, 0)\right)$. By two Taylor series expansions, we have

$$\begin{aligned}
& Q\left(\xi(\theta(\hat{\epsilon_{in}}), \epsilon_{\text{in}})\right) - Q\left(\xi(\theta^*, 0)\right) \\
= \; & Q\left(\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{\text{in}})\right) - Q\left(\xi(\hat{\theta}(\epsilon_{in}), 0)\right) + Q\left(\xi(\theta(\hat{\epsilon_{in}}), 0)\right) - Q\left(\xi(\theta^*, 0)\right) \\
= \; & \left[\nabla_\xi Q\left(\xi\left(\theta\right)\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right]' \left(\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{\text{in}}) - \xi(\hat{\theta}(\epsilon_{in}), 0) \right) + O\left(\left\| \xi(\hat{\theta}(\epsilon_{in}), \epsilon_{\text{in}}) - \xi(\hat{\theta}(\epsilon_{in}), 0) \right\|^2 \right) + \\
& \left[\left(\nabla_\theta \xi\left(\theta\right)\right)' \nabla_\xi Q\left(\xi\right)\big|_{\xi=\xi\left(\theta^*,0\right)} \right]' \left(\hat{\theta}(\epsilon_{in}) - \theta^* \right) + O\left(\left\| \hat{\theta}(\epsilon_{in}) - \theta^* \right\|^2 \right) \\
= \; & \left[\nabla_\xi Q\left(\xi\right)\Big|_{\xi=\xi\left(\hat{\theta}(\epsilon_{in}),0\right)} \right]' \left(\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{in}) - \xi(\hat{\theta}(\epsilon_{in}), 0) \right) + O\left(\left\| \xi(\hat{\theta}(\epsilon_{in}), \epsilon_{\text{in}}) - \xi(\hat{\theta}(\epsilon_{in}), 0) \right\|^2 \right) + O\left(\left\| \hat{\theta}(\epsilon_{in}) - \theta^* \right\|^2 \right),
\end{aligned}$$

because $\nabla\xi(\theta^*)'\nabla_\xi Q(\xi(\theta^*)) = 0$ at the true estimates θ^*.

Rearranging the equality involving $Q\left(\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{\text{in}})\right) - Q(\xi(\theta^*, 0))$ to focus on the $O\left(\left\|\hat{\theta}(\epsilon_{in}) - \theta^*\right\|^2\right)$ term, we have

$$\begin{aligned}
O\left(\left\|\hat{\theta}(\epsilon_{in}) - \theta^*\right\|^2\right) &= Q\left(\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{in})\right) - Q(\xi(\theta^*, 0)) - \left[\nabla_\xi Q(\xi)\Big|_{\xi=\xi(\hat{\theta}(\epsilon_{in}),0)}\right]'\left(\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{in}) - \xi(\hat{\theta}(\epsilon_{in}), 0)\right) \\
&\quad - O\left(\left\|\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{in}) - \xi(\hat{\theta}(\epsilon_{in}), 0)\right\|^2\right) \\
&\leq \left|Q\left(\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{in})\right) - Q(\xi(\theta^*, 0))\right| + \left\|\nabla_\xi Q(\xi)\Big|_{\xi=\xi(\theta(\hat{\epsilon_{in}}),0)}\right\| \left\|\xi(\theta(\hat{\epsilon_{in}}), \epsilon_{in}) - \xi(\hat{\theta}(\epsilon_{in}), 0)\right\| \\
&\quad - O\left(\left\|\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{in}) - \xi(\hat{\theta}(\epsilon_{in}), 0)\right\|^2\right).
\end{aligned}$$

Since we assume $\left\|\nabla_\xi Q(\xi)\Big|_{\xi=\xi(\hat{\theta}(\epsilon_{in}),0)}\right\|$ is bounded, the second-order term $O\left(\left\|\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{in}) - \xi(\hat{\theta}(\epsilon_{in}), 0)\right\|^2\right)$ can be ignored. This allows us to focus on the numerical error from the NFP algorithm's inner loop and the bias in objective values. From Theorem 1, we also know $\left\|\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{\text{in}}) - \xi(\hat{\theta}(\epsilon_{in}), 0)\right\| \leq \frac{L(\hat{\theta}(\epsilon_{in}))}{1-L(\hat{\theta}(\epsilon_{in}))}\epsilon_{\text{in}}$. Hence, we obtain

$$O\left(\left\|\hat{\theta}(\epsilon_{in}) - \theta^*\right\|^2\right) \leq \left|Q\left(\xi(\hat{\theta}(\epsilon_{in}), \epsilon_{\text{in}})\right) - Q(\xi(\theta^*, 0))\right| + O\left(\frac{L(\hat{\theta}(\epsilon_{in}))}{1 - L(\hat{\theta}(\epsilon_{in}))}\epsilon_{\text{in}}\right).$$

A.4 Proof of Theorem 6

The NFP method (4) solves the following unconstrained problem

$$\min_\theta Q(\xi(\theta)). \tag{19}$$

The first-order condition of (19) is

$$\frac{\partial Q(\xi(\theta))}{\partial\theta} = \frac{d\xi}{d\theta}'\frac{\partial Q}{\partial\xi} = 0. \tag{20}$$

The constrained optimization formulation of (19) is

$$\begin{aligned}
\min_{(\theta,\xi)} \quad & Q(\xi) \\
\text{s.t.} \quad & s(\xi;\theta) = S.
\end{aligned} \tag{21}$$

The Lagrangian for (21) is $\mathcal{L}(\theta,\xi,\lambda) = Q(\xi) - \lambda^T (S - s(\xi;\theta))$, where λ is the vector of Lagrange multipliers. The first-order conditions of (21) are

$$\begin{aligned}
\frac{\partial \mathcal{L}(\theta,\xi,\lambda)}{\partial \theta} &= - \quad \frac{ds(\xi;\theta)'}{d\theta}\lambda = 0 \\
\frac{\partial \mathcal{L}(\theta,\xi,\lambda)}{\partial \xi} &= \quad \frac{\partial Q}{\partial \xi} - \frac{ds(\xi;\theta)'}{d\xi}\lambda = 0 \\
\frac{\partial \mathcal{L}(\theta,\xi,\lambda)}{\partial \lambda} &= \quad S - s(\xi;\theta) = 0.
\end{aligned} \tag{22}$$

Since the BLP inner loop is a contraction mapping, the matrix $\frac{ds(\xi;\theta)'}{d\xi}$ is invertible.[33] Solving the second set of first order conditions for λ gives $\lambda = \left(\frac{ds(\xi;\theta)'}{d\xi}\right)^{-1}\frac{\partial Q}{\partial \xi}$. Then

$$\frac{\partial \mathcal{L}}{\partial \theta} = -\frac{ds(\xi;\theta)'}{d\theta}\left(\frac{ds(\xi;\theta)'}{d\xi}\right)^{-1}\frac{\partial Q}{\partial \xi} = 0, \tag{23}$$

which is identical to (20), the first-order condition from the NFP formulation. To see the equivalence, note that the implicit function theorem (Theorem M.E.1 in Mas-Collel, Whinston and Green 1995) states

$$\frac{\partial \xi(\theta)}{\partial \theta} = -\left(\frac{ds(\xi;\theta)}{d\xi}\right)^{-1}\frac{ds(\xi;\theta)}{d\theta},$$

so by substitution,

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial \xi(\theta)'}{\partial \theta}\frac{\partial Q}{\partial \xi} = \frac{\partial Q(\xi(\theta))}{\partial \theta}.$$

B Gradients for the MPEC Objective Function and Constraints

Here we derive the gradients of the MPEC objective function and constraints with respect to the optimization parameters in MPEC. These gradients are an important input, for both numerical accuracy and speed. Nevo (2000) lists the gradients for NFP. This section uses the independent normal distribution for each of the random coefficients, as in BLP (1995) and many other empirical papers.

Market Share

$$\begin{aligned}
s_j(\xi_t;\theta) &= \int \frac{\exp\left(x_{j,t}'\overline{\beta} - \bar{\alpha}p_{j,t} + \xi_{j,t} + \sum_k x_{kj,t}'\nu_k\sigma_{\beta_k} - p_{j,t}\nu_{K+1}\sigma_\alpha\right)}{1+\sum_{i=1}^{J}\exp\left(x_{it}'\overline{\beta} - \bar{\alpha}p_{i,t} + \xi_{i,t} + \sum_k x_{k,i,t}'\nu_k\sigma_{\beta_k} - p_{i,t}\nu_{K+1}\sigma_\alpha\right)} dF(\nu) \\
&= \int T_j(\xi_t,\nu;\theta)dF(\nu)
\end{aligned}$$

$$\text{where}\, \theta = (\overline{\beta}, \overline{\alpha}, \sigma_\beta, \sigma_\alpha)', \text{ and}\, \nu \sim N(0, I_{K+1}).$$

MPEC Criterion Function

[33] We thank Ken Judd and John Birge for pointing out this property.

$$\begin{array}{ll} \min_{\theta,\xi} & g(\xi)' W g(\xi) \\ \text{subject to} & s(\xi;\theta) = S \end{array} . \tag{24}$$

$$\text{where } g(\xi) = \frac{1}{T}\sum_{t=1}^{T} \xi_t' z_t$$

Gradients for MPEC

$$\frac{\partial s_j(\xi_t;\theta)}{\partial \overline{\beta_k}} = \int T_j(\xi_t,\nu;\theta)(x_{j,k,t} - \sum_i T_i(\xi_t,\nu;\theta) x_{k,i,t}) dF(\nu)$$

$$\frac{\partial s_j(\xi_t;\theta)}{\partial \overline{\alpha}} = \int T_j(\xi_t,\nu;\theta)(p_{j,k,t} - \sum_i T_i(\xi_t,\nu;\theta) p_{k,i,t}) dF(\nu)$$

$$\frac{\partial s_j(\xi_t;\theta)}{\partial \sigma_{\beta_k}} = \int T_j(\xi_t,\nu;\theta)(x_{j,k,t} - \sum_i T_i(\xi_t,\nu;\theta) x_{k,i,t}) \nu_k dF(\nu)$$

$$\frac{\partial s_j(\xi_t;\theta)}{\partial \sigma_{\alpha}} = \int T_j(\xi_t,\nu;\theta)(p_{j,k,t} - \sum_i T_i(\xi_t,\nu;\theta) p_{k,i,t}) \nu_{K+1} dF(\nu)$$

$$\frac{\partial s_j(\xi_t;\theta)}{\partial \xi_{j,t}} = \int T_j(\xi_t,\nu;\theta)(1 - T_j(\xi_t,\nu;\theta)) dF(\nu)$$

$$\frac{\partial s_j(\xi_t;\theta)}{\partial \xi_{i,t}} = -\int T_j(\xi_t,\nu;\theta)) T_i(\xi_t,\nu;\theta)) dF(\nu)$$

$$\frac{\partial g(\xi)' W g(\xi)}{\partial \xi} = 2 g(\xi)' W \frac{\partial g(\xi)}{\partial \xi}$$

C Chebyshev Approximation of the Expected Value of Waiting

First, we bound the range of prices as follows, $p = (p_1, p_2)^{'} \in [0,b] \times [0,b]$, where b is large (b is 1.5 times the largest observed price in the data). We then approximate the expected value of delaying adoption with Chebyshev polynomials, $v_0^r(p;\theta^r) \approx \gamma^{r'} \Lambda(p)$, where γ^r is a $K \times 1$ vector of parameters and $\Lambda(p)$ is a $K \times 1$ vector of K Chebyshev polynomials. Therefore, we can re-write the Bellman as

$$\gamma^{r'} \Lambda(p) = \delta \int \log \left(\exp\left(\gamma^{r'} \Lambda(p\rho + \psi)\right) + \sum_j \exp\left(\beta_j^r - \alpha^r (p'\rho_j + \psi) + \xi_j\right) \right) dF_{\psi,\xi}(\psi,\xi).$$

To solve for the Chebyshev weights, we use the Galerkin method described in Judd (1992). We define the residual function:

$$\begin{aligned} R(p;\gamma) &= \gamma^{r'}\Lambda(p) - \ldots \\ & \delta \int \log\left(\exp\left(\gamma^{r'}\Lambda(p\rho+\psi)\right) + \sum_j \exp\left(\beta_j^r - \alpha^r\left(p'\rho_j+\psi\right)+\xi_j\right)\right) dF_{\psi,\xi}(\psi,\xi) \end{aligned} \quad . \tag{25}$$

Next, we let X be the matrix of K Chebyshev polynomials at each of the G points on our grid (i.e. G nodes). Our goal is to search for parameters, γ, that set the following expression to zero:

$$X' R(p;\gamma) = 0. \tag{26}$$

We use an iterated least squares approach for NFP.

1. Pick a starting value $\gamma^{r,0}$, $v_0^{r,0}(p;\Theta^r) = \gamma^{r,0'}\rho(p)$
2. Compute $Y\left(p;\gamma^{r,0}\right) = \delta\int\log\left(\exp\left(\gamma^{r,0'}\Lambda(p\rho+\psi)\right)+\sum_j \exp\left(\beta_j^r - \alpha^r\left(p'\rho_j+\psi\right)+\xi_j\right)\right)dF_{\psi,\xi}(\psi,\xi)$ using quadrature
3. solve the least squares problem: $\min_\gamma R(p;\gamma)'R(p;\gamma) \Rightarrow \min_\gamma \left(X\gamma^r - Y\left(p;\gamma^{r,0}\right)\right)'\left(\left(X\gamma^r - Y\left(p;\gamma^{r,0}\right)\right)\right)$
   - for which the solution is: $\gamma^{r,1} = \left(X'X\right)^{-1}X'Y\left(p;\gamma^{r,0}\right)$.
4. Compute $v_0^{r,1}(p;\Theta^r) = \gamma^{r,1'}\Lambda(p)$
5. Repeat steps 2 and 3 until convergence.

D Jacobian of the Density of (p_t, S_t) in the Dynamic BLP model

The Jacobian is defined as follows:

$$J_{t,u\to Y} = \begin{bmatrix} \frac{\partial\psi_t}{\partial p_t} & \frac{\partial\psi_t}{\partial S_t} \\ \frac{\partial\xi_t}{\partial p_t} & \frac{\partial\xi_t}{\partial S_t} \end{bmatrix}.$$

Since $\frac{\partial\psi_t}{\partial\log(p_t)} = I_J$ and $\frac{\partial\psi_t}{\partial\log(p_t)} = 0_J$ (a square matrix of zeros), we only need to compute the matrix of derivatives, $\left[\frac{\partial\xi_t}{\partial S_t}\right]$. We can simplify this calculation by applying the implicit function theorem to the following system

$$G(S_t,\xi_t) = s(p,\xi_t;\Theta) - S_t = 0$$

and computing the lower block of the Jacobian as

$$\begin{aligned} J_{t,\xi\rightarrow S} &= -\left[\frac{\partial G}{\partial \xi_t}\right]^{-1}\left[\frac{\partial G}{\partial S_t}\right] \\ &= \left[\frac{\partial s}{\partial \xi_t}\right]^{-1} \end{aligned},$$

where the (j,k) element of $\frac{\partial s_{j,t}}{\partial \xi_{k,t}}$ is

$$\frac{\partial S_{j,t}}{\partial \xi_{k,t}} = \begin{cases} \sum_r \lambda_{r,t} s_j\left(p_t, \xi_t; \Theta^r\right)\left(1 - s_j\left(p_t, \xi_t; \Theta^r\right)\right) & \text{, if } j = k \\ -\sum_r \lambda_{r,t} s_j\left(p, \xi_t; \Theta^r\right) s_k\left(p, \xi_t; \Theta^r\right) & \text{, otherwise.} \end{cases}$$