Best Practices for Differentiated Products Demand Estimation with `pyblp`[*]

Christopher Conlon[†] Jeff Gortmaker[‡]

March 10, 2019

Abstract

Differentiated products demand systems are a workhorse for understanding the price effects of mergers, the value of new goods, and the contribution of products to seller networks. Berry et al. (1995) provide a flexible workhorse model which accounts for the endogeneity of prices and is based on the random coefficients logit. While popular, there exists no standardized generic implementation for the Berry et al. (1995) estimator. This paper reviews and combines several recent advances related to the estimation of BLP-type problems and implements an extensible generic interface via the `pyblp` package. We conduct a number of Monte Carlo experiments and replications which suggest different conclusions than the prior literature: local optima appear to be rare in well-identified problems; it is possible to obtain good performance even in small samples and without exogenous cost-shifters, particularly when "optimal instruments" are employed along with supply-side restrictions.

If Python is installed on your computer, `pyblp` can be installed with the following command:

```
pip install pyblp
```

Up to date documentation for the package is available at `https://pyblp.readthedocs.io`.

[*]Thanks to... Daniel Stackman provided excellent research assistance. Any remaining errors are our own.

[†]New York University, Stern School of Business: cconlon@stern.nyu.edu [Corresponding Author]

[‡]Federal Reserve Bank of New York: jeff@jeffgortmaker.com

1. Introduction

Empirical models of supply and demand for differentiated products are one of the most important achievements of the New Empirical Industrial Organization (NEIO) literature of the last thirty years. The workhorse model is the Berry et al. (1995) approach, which provides both an estimator that allows for flexible substitution patterns across products while also addressing the potential endogeneity of price, and algorithm for recovering that estimator. It is popular, because it addresses many concerns from the previous literature such as fixed markups (as in CES demands), or representative agents (as in Deaton and Muellbauer, 1980). It also has the advantage that it both scales well for large numbers of potential products, and can be applied to both aggregated and dis-aggregated data. The BLP model and its variants have been used in a wide variety of applications: understanding the value of new goods (Petrin, 2002), the price effects of mergers (Nevo, 2001, 2000a), and two-sided markets (Fan, 2013) and (Lee, 2013). The BLP approach has been applied to a wide number of different questions and industries including hospital demand and negotiations with insurers (Ho and Pakes, 2014) and students' choice of schools (Bayer et al., 2007), and even asset pricing (Koijen and Yogo, 2018). Moreover, the BLP approach has been extremely influential in the practice of prospective merger evaluation, particularly in recent years.

The model itself is both quite simple to understand, and quite challenging to estimate. At its core, it involves a nonlinear change of variables from the space of observed marketshares to the space of mean utilities for products. After this nonlinear change of variables, the BLP problem is simply either a single linear instrumental variables regression problem, or a two equation (supply and demand) linear IV regression problem. This means that a wide variety of tools for that class of problems are available to researchers.

The main disadvantage of the BLP estimator is that the nonlinear change of variables means it is a non-linear, non-convex optimization problem with a simulated objective function. The problem must be solved iteratively using nonlinear optimization software, and because of the non-convexity, there is no mathematical guarantee that a solution will always be found. This has led to some frustration with the BLP approach (see Knittel and Metaxoglou, 2014). There is also the fear that when estimation is slow or complicated, researchers may cut corners in undesirable ways and sacrifice modeling richness for computational speed.

There is a growing recent literature which focuses on methodological innovations for BLP-type problems, or the econometric properties of the BLP estimator. For example, Knittel and Metaxoglou (2014) estimate the BLP model many times using the automobile data from Berry et al. (1999) data (with only a demand side specified), and the simulated fake cereal data from Nevo (2000b). The authors apply several different numerical optimization routines to the same data, and obtain a wide range of parameter estimates and implied elasticities suggesting multiple local optima. When we attempt replicate these difficulties with our `pyblp` package, parameter estimates do not appear affected by our choice of optimization algorithm, and appear (for the most part) stable. We attribute this primarily to numerical adjustments specific to our implementation and careful configuration of optimization software.

Relatedly, Armstrong (2016) shows that absent exogenous cost shifting instruments, identification can be weak in these sorts of problems. Conlon (2017) shows that what appears to be a weak instrument problem, may actually be a many instrument problem in the language of Newey and Smith (2004). Recent work has further examined the role of instruments by Gandhi and Houde (2017) and Reynaert and Verboven (2014). The former construct *differentiation IV*, while the latter construct *optimal IV* in the sense of Amemiya (1977) or Chamberlain (1987).[1] We provide routines to construct both sets of instruments. Our results with respect to differentiation IV are mostly consistent with Gandhi and Houde (2017) in that they substantially outperform the traditional *BLP instruments*. Our results with respect to optimal instruments are broadly similar with those of Reynaert and Verboven (2014) in that the performance gains are substantial both in terms of bias and efficiency.

However, our simulations indicate, even absent strong cost-shifting instruments, that it is possible to obtain unbiased (and relatively precise) parameter estimates when one includes a properly specified supply side. Our findings appear to provide a partial solution to the challenges posed by Armstrong (2016) which supports the "folklore" around the original Berry et al. (1995) paper: that supply restrictions were necessary to pin down parameter estimates. Our interpretation is somewhat different from Reynaert and Verboven (2014) which suggest that once optimal demand side instruments are included, the addition of a supply side has limited benefit.[2] Indeed, our simulations indicate with both a correctly specified supply side and optimal instruments together, the finite sample performance of the estimator is good even absent strong cost-shifters.

There is another strand of the literature which focuses on the role of simulation error in the BLP problem. Freyberger (2015) shows how to bias correct parameter estimates for simulation error. Judd and Skrainka (2011) and Heiss and Winschel (2008) consider a variety of integration techniques and suggest *sparse-grid quadrature* as the best alternative. We implement a variety of routines (pseudo-Monte Carlo, quasi-Monte Carlo, quadrature, and sparse-grid quadrature) and our findings mirror those in Heiss and Winschel (2008).

There is also a recent literature of alternative approaches to BLP problems employing different algorithms or statistical estimators, which we do not directly address. For example, Dubé et al. (2012) propose an alternative estimation algorithm based on the mathematical programming with equilibrium constraints (MPEC) method of Su and Judd (2012). While this method has some advantages, it is somewhat difficult to incorporate simultaneous supply and demand, and challenging to include a large number of fixed effects. Conlon (2017) proposes a generalized empirical likelihood version of the MPEC estimator, which has some attractive properties but presents many of the same

[1]It is worth mentioning that the actual optimal IV are well-known to be infeasible. See Berry et al. (1995) or Berry et al. (1999).

[2]This is most likely because those authors only examined scenarios with strong cost-shifters.

challenges. Lee and Seo (2015) provide an *approximate BLP* estimator, which is asymptotically similar to the BLP estimator though differs in finite sample. Salanie and Wolak (2018) propose another approximate estimator that can be estimated with linear IV, and is helpful for constructing good starting values.

Despite all of these methodological improvements, what is still lacking from this literature is a standardized implementation that is sufficiently general to encompass a wide range of potential problems and use cases. Instead, nearly every researcher implements the estimator on their own with problem-specific tweaks and adjustments. This makes replication extremely challenging, and also makes it hard to evaluate various different methodological and statistical improvements to the estimator.

The goal of this paper is to present best practices in the estimation of BLP-type models, some of which are well-known in the literature, others of which are lesser known, and other still are novel to this paper. In addition to presenting these best practices, we provide a common framework, `pyblp`, which offers a general implementation of the BLP approach as a Python 3 package. This software is general, extensible, and open-source so that it can be modified and extended by scholars as new methodological improvements become available. The hope is that these best practices, along with this standardized and extensible software implementation reduce some of the barriers to BLP-type estimation and make these techniques accessible to a wider range of researchers. In addition, we hope that by providing a common framework, this makes structural estimation of demand and supply easier to replicate for researchers and practitioners.

2. Model

Table 1: Notation

j	products
t	markets
i	"individuals"
f	firms
N	number of products across all markets
T	number of markets
J_t	number of products in market t
I_t	number of individuals in market t
F_t	number of firms in market t
$\mathscr{J}_t$	set of products in market t
$\mathscr{J}_{ft}$	set of products owned by f in market t
θ_1	exogenous linear demand parameters
θ_2	endogenous (nonlinear) parameters including α
$\widetilde{\theta}_2$	endogenous (nonlinear) parameters
θ_3	exogenous linear supply parameters
β	exogenous linear demand parameters excluding fixed effects
α	endogenous linear demand parameter on price
γ	exogenous linear supply parameters excluding fixed effects
s_{jt}	calculated market share of j in market t
$\mathcal{S}_{jt}$	observed market share of j in market t
p_{jt}	price of j in market t
c_{jt}	marginal cost of j in market t
u_{ijt}	i's indirect utility of j in market t
δ_{jt}	mean utility of j in market t
μ_{ijt}	i-specific utility of j in market t
ϵ_{ijt}	i's idiosyncratic preferences for j in market t
ξ_{jt}	demand-side structural error
ω_{jt}	supply-side structural error
O	a $J_t \times J_t$ matrix of 0's and 1's where 1 denotes same owners
Ω	a $J_t \times J_t$ matrix of demand derivatives
η_{jt}	markup of j in market t
Z	instruments
g	sample moments
q	objective function
W	weighting matrix

2.1. Demand

Berry et al. (1995) begin with the following problem. An individual i in market $t = 1, \ldots, T$ receives indirect utility from selecting a particular product j:

$$u_{ijt} = \delta_{jt} + \mu_{ijt} + \epsilon_{ijt}. \tag{1}$$

Consumers then choose among $\mathscr{J}_t = \{0, 1, \ldots, J_t\}$ discrete alternatives and select exactly one option which provides the most utility (including the outside alternative, denoted $j = 0$):

$$d_{ij} = \begin{cases} 1 & \text{if } u_{ijt} > u_{ikt} \text{ for } j \neq k, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Aggregate marketshares are given by integrating over heterogeneous consumer choices:

$$s_{jt}(\delta_{\cdot t}) = \int d_{ij}(\delta_{\cdot t}, \mu_{it}) \, d\mu_{it} \, d\epsilon_{it}.$$

When ϵ_{ijt} is distributed IID type I extreme value (Gumbel) this becomes:[3]

$$s_{jt}(\delta_{\cdot t}, \theta_2) = \int \frac{\exp[\delta_{jt} + \mu_{ijt}]}{\sum_{k \in J_t} \exp[\delta_{kt} + \mu_{ikt}]} f(\mu_{it} \mid \widetilde{\theta}_2) \, d\mu_{it}. \tag{3}$$

This is often referred to as a *mixed logit* or *random coefficients logit* because each individual i's demands are given by a multinomial logit kernel, while $f(\mu_{it} \mid \widetilde{\theta}_2)$ denotes the *mixing distribution* over the heterogeneous types i and θ_2 parametrizes this heterogeneity. Indeed, θ_2 contains all of the endogenous parameters of the model (the heterogeneous tastes) as well as the endogenous parameter on price α.

The key insight of Berry (1994) or Berry et al. (1995) is that we can perform a nonlinear change of variables (see Berry and Haile, 2014): $\delta_t = D_t^{-1}(\mathcal{S}_t, \theta_2)$ where $\mathcal{S}_t$ denotes the J_t vector of *observed marketshares*. For each market t, equation (3) represents a $J_t \times J_t$ system of equations in $\delta_{\cdot t}$. Given the $\delta_{\cdot t}(\mathcal{S}_t, \widetilde{\theta}_2)$ which solves that system of equations; along with some covariates x_{jt}, prices p_{jt}, and a structural error ξ_{jt}; under an additivity assumption, one can write the index:

$$\delta_{jt}(\mathcal{S}_t, \widetilde{\theta}_2) = x_{jt}\beta - \alpha p_{jt} + \xi_{jt}. \tag{4}$$

With the addition of some instruments Z_{jt}^D (including the exogenous regressors x_{jt}), one can form moment restriction conditions of the form $E[\xi_{jt}' Z_{jt}^D] = 0$.

2.2. Supply

We can derive an additional set of supply moments from the multi-product differentiated Bertrand first order conditions. Consider the profits of firm f which for a single market t controls several

[3]Identification generally requires normalizing one of the options. Often the outside option $u_{i0t} = 0$ is the preferred normalization.

products J_f and sets prices p_j:

$$\arg\max_{p_j\,:\,j\in\mathcal{J}_f} \sum_{j\in\mathcal{J}_f} (p_j - c_j)\cdot s_j(\mathbf{p}),$$
$$s_j(p) + \sum_{k\in\mathcal{J}_f} \frac{\partial s_k}{\partial p_j}(\mathbf{p})\cdot(p_k - c_k) = 0.$$

It is helpful to write the first order conditions in matrix form so that for a single market t,

$$\begin{aligned} s(\mathbf{p}) &= (O\odot\Omega(\mathbf{p}))\cdot(\mathbf{p}-\mathbf{c}),\\ (\mathbf{p}-\mathbf{c}) &= \underbrace{(O\odot\Omega(\mathbf{p}))^{-1}s(\mathbf{p})}_{\eta}. \end{aligned} \tag{5}$$

Here the multi-product Bertrand markup $\eta(\mathbf{p},\mathbf{s},\theta_2)$ depends on the element-wise (Hadamard) product $\odot$ of two $J_t \times J_t$ matrices: the matrix of demand derivatives $\Omega(\mathbf{p})$ where each (j,k) entry is given by $\frac{\partial s_j}{\partial p_k}$, and the ownership matrix O, given by[4]

$$O_{jk} = \begin{cases} 1 & \text{if } (j,k)\in\mathcal{J}_f \text{ for any } f,\\ 0 & \text{otherwise.}\end{cases}$$

This enables us to recover an estimate of marginal costs $c_{jt} = p_{jt} - \eta_{jt}$, which in turn allows us to construct additional *supply side moments.* We can parametrize marginal cost as[5]

$$f(p_{jt}-\eta_{jt}) = f(c_{jt}) = x_{jt}\gamma_1 + w_{jt}\gamma_2 + \omega_{jt} \tag{6}$$

and construct moment conditions of the form $E[\omega'_{jt}Z^S_{jt}] = 0$. The idea is that we can use observed prices, along with information on demand derivatives $\Omega_t(\theta_2, p_t)$ and firm conduct, to recover markups η_{jt} and then marginal costs c_{jt}. This also imposes a functional form restriction on marginal costs, which depends on both product characteristics x_{jt} and the marginal cost shifters w_{jt} that are excluded from demand.

[4]We can easily consider alternative forms of conduct such as Single Product Oligopoly, Multiproduct Oligopoly, or Multiproduct Monopoly. Miller and Weinberg (2017) consider estimating a single parameter $O(\kappa)$ and Backus et al. (2018) use `pyblp` test various forms of $O(\kappa)$.

[5]The most common choice of $f(\cdot)$ is the identity function, but some authors also consider $f(\cdot) = \log(\cdot)$. In practice this constrains marginal costs to be always positive.

2.3. The Estimator

We can construct a GMM estimator using our supply and demand moments. To do so, we stack their sample analogues to form:[6]

$$g(\theta) = \begin{bmatrix} g^D(\theta) \\ g^S(\theta) \end{bmatrix} = \begin{bmatrix} \frac{1}{N}\sum_{j,t} \xi_{jt} Z_{jt}^D \\ \frac{1}{N}\sum_{j,t} \omega_{jt} Z_{jt}^S \end{bmatrix}$$

and construct a nonlinear GMM estimator for $\theta = [\beta, \alpha, \widetilde{\theta}_2, \gamma]$ with some weighting matrix W:

$$\min_\theta q(\theta) \equiv g(\theta)' W g(\theta) \tag{7}$$

For clarity, we partition the parameter space θ into three parts: the $K_1 \times 1$ vector θ_1 contains the exogenous demand parameters β, the $K_3 \times 1$ vector θ_3 contains the exogenous supply parameters γ, and the remaining (endogenous) parameters, including the price coefficient α, are contained in the $K_2 \times 1$ vector θ_2. The endogenous parameters are endogenous in the literal sense that they appear in both equations for demand and supply.

To be explicit we write the entire program as follows:

$$\begin{aligned} \min_\theta q(\theta) &\equiv g(\theta)' W g(\theta) \\ g(\theta) &= \frac{1}{N}\begin{bmatrix} (Z^D)'\xi \\ (Z^S)'\omega \end{bmatrix} \\ \xi_{jt} &= \delta_{jt} - x_{jt}\beta + \alpha p_{jt} \\ \omega_{jt} &= f(p_{jt} - \eta_{jt}) - x_{jt}\gamma_1 - w_{jt}\gamma_2 \\ \eta_{\cdot t} &= (O \odot \Omega(\mathbf{p_t}, \widetilde{\theta}_2))^{-1} s_t \\ \mathcal{S}_{jt} = s_{jt}(\delta, \theta_2) &\equiv \int \frac{\exp[\delta_{jt} + \mu_{ijt}]}{\sum_{k \in \mathcal{J}_t} \exp[\delta_{kt} + \mu_{ikt}]} f(\mu_{it} | \widetilde{\theta}_2)\, d\mu_{it} \end{aligned} \tag{8}$$

This estimator and its econometric properties are discussed in Berry et al. (1995) and Berry et al. (2004). Our focus is not going to be on the econometric properties of $\hat{\theta}$ but rather various algorithms by which one might obtain $\hat{\theta}$. Technically, we need to solve this program twice. Once to obtain a consistent estimate for $\hat{W}(\hat{\theta})$ and a second time to obtain the efficient GMM estimator.

2.4. The Nested Fixed Point Algorithm

In addition to providing an estimator, Berry et al. (1995) provide an algorithm to solving (8) which attempts to simplify the problem. One can concentrate out $[\theta_1, \theta_3]$ and perform a nonlinear search over just θ_2. Our approach differs somewhat from the previous literature and we provide details in

[6] Here $N = \sum_t \dim(\mathcal{J}_t)$.

Appendix A. The intuition is that the exogenous parameters enter the transformed problem linearly, and thus one can construct $[\hat{\theta}_1(\theta_2), \hat{\theta}_3(\theta_2)]$. This works because the markup term $\eta_{jt}(\mathbf{s_t}, \mathbf{p_t}, \delta_t, \theta_2)$ can be written as a function of only the θ_2 parameters and does not depend on $[\theta_1, \theta_3]$.[7]

In this case parameters are explicit functions of other parameters and can be solved for in a particular order. For each guess of θ_2:

(a) For each market t, solve $\mathcal{S}_{jt} = s_{jt}(\delta_{\cdot t}, \theta_2)$ for $\hat{\delta}_{\cdot t}(\mathcal{S}_{jt}, \theta_2)$.

(b) For each market t, use $\hat{\delta}_{\cdot t}(\mathcal{S}_{jt}, \theta_2)$ to construct $\eta_{\cdot t}(\mathbf{s_t}, \mathbf{p_t}, \hat{\delta}_{\cdot t}(\theta_2), \theta_2)$

(c) For each market t, recover $\hat{c}_{jt}(\hat{\delta}_{\cdot t}(\theta_2), \theta_2) = p_{jt} - \eta_{jt}(\hat{\delta}_{\cdot t}(\theta_2), \theta_2)$

(d) Stack up $\hat{\delta}_{\cdot t}(\mathcal{S}_{jt}, \theta_2)$ and $\hat{c}_{jt}(\hat{\delta}_{\cdot t}(\theta_2), \theta_2)$ and use linear IV-GMM to recover $[\hat{\theta}_1(\theta_2), \hat{\theta}_3(\theta_2)]$ following the recipe in Appendix A. The following is slightly different from the typical formulation so that we can absorb fixed effects:

$$\begin{aligned} \hat{\delta}_{jt}(\mathcal{S}_{jt}, \theta_2) + \alpha p_{jt} &= x'_{jt}\beta + \xi_{jt}, \\ \hat{c}_{jt}(\theta_2) &= [x_{jt}\ w_{jt}]\gamma + \omega_{jt}. \end{aligned} \tag{9}$$

(e) Construct the residuals:

$$\begin{aligned} \hat{\xi}_{jt}(\theta_2) &= \hat{\delta}_{jt}(\theta_2) - x_{jt}\hat{\beta}(\theta_2) + \alpha p_{jt}, \\ \hat{\omega}_{jt}(\theta_2) &= \hat{c}_{jt}(\theta_2) - [x_{jt}\ w_{jt}]\hat{\gamma}(\theta_2). \end{aligned} \tag{10}$$

(f) Stack the sample moments:

$$g(\theta_2) = \begin{bmatrix} \frac{1}{N}\sum_{jt} \hat{\xi}_{jt}(\theta_2) Z^D_{jt} \\ \frac{1}{N}\sum_{jt} \hat{\omega}_{jt}(\theta_2) Z^S_{jt} \end{bmatrix}. \tag{11}$$

(g) Construct the GMM objective $q(\theta_2) = g(\theta_2)' W g(\theta_2)$.

We provide analytic gradients for the BLP problem with supply and demand in Appendix B. One advantage of the BLP algorithm is that is performs a nonlinear search over only K_2 *nonlinear parameters*. Consequently, the Hessian matrix is only $K_2 \times K_2$. This implies relatively minimal memory requirements. Also, the IV-GMM regression in step (d) concentrates out the *linear* parameters $[\theta_1, \theta_3]$. This implies that large numbers of linear parameters β can be estimated essentially for free which is important if one includes a large number of fixed effects such as product or market level fixed effects.[8] In fact, other than (a) the remaining steps are computationally trivial. As is well known, (a)-(c) can be performed separately for each market across multiple processors.

[7]Why? We already know we can invert the shares to solve for $\delta_t(\theta_2)$. Because Ω_t, the matrix of demand derivatives $\partial s_{kt}/\partial p_{jt} = -\int \alpha_i \left[s_{ikt} \cdot I_{j=k} - s_{ikt}s_{ijt}\right] f(\mu_{it}, \alpha_i \mid \theta_2)$ again depends only on θ_2 (which contains the price coefficient α).

[8]For example Nevo (2001) and Nevo (2000b) includes product fixed effects.

The main disadvantage is that all parameters are implicit functions of other parameters, particularly of θ_2. The objective is now a complicated implicit function of θ_2. Once we incorporate any heterogeneous tastes, the resulting optimization problem is non-convex. In general the complexity of this problem grows rapidly with the number of nonlinear θ_2 parameters, while a high number of linear $[\theta_1, \theta_3]$ parameters is more or less for free.

2.5. Nested Logit and RCNL Variants

The random coefficients nested logit (RCNL) model of Brenkers and Verboven (2006) instead places a generalized extreme value (GEV) structure on ϵ_{ijt}. This model is popular in applications where the most important set of characteristics governing substitution is categorical. This has made it popular in studies of alcoholic beverages such as distilled spirits (Conlon and Rao, 2017; Miravete et al., 2018) and beer (Miller and Weinberg, 2017).

Much like the random coefficients model integrates over a heterogeneous distribution where each individual type follows a logit distribution, the RCNL model integrates out over a heterogenous distribution where each individual now follows a *nested logit* (GEV distribution). We expand our definition of the endogenous parameters θ_2 to include the nesting parameter ρ so that $\theta_2 = [\alpha, \rho, \widetilde{\theta}_2]$:[9]

$$u_{ijt} = \delta_{jt}(\theta_1, \alpha) + \mu_{ijt}(\widetilde{\theta}_2) + \epsilon_{ijt}(\rho)$$

The primary difference from the nested logit is that the inclusive value term for all products in nest J_{gt} now depends on the consumer's type i:

$$\begin{aligned} s_{jt}(\delta_{jt}, \theta_2) &= \int \frac{\exp[(\delta_{jt} + \mu_{ijt}(\widetilde{\theta}_2))/(1-\rho)]}{\exp[I_{ig}/(1-\rho)]} \cdot \frac{\exp[I_{ig}]}{\exp[IV_i]} f(\mu_{ijt}|\widetilde{\theta}_2)\, d\mu_{it}, \\ I_{ig}(\theta_1, \theta_2) &= (1-\rho) \ln \sum_{j \in J_{gt}} \exp\left(\frac{\delta_{jt} + \mu_{ijt}(\widetilde{\theta}_2)}{1-\rho}\right), \quad IV_i(\theta_1, \theta_2) = \ln\left(1 + \sum_{g=1}^{G} \exp I_{ig}\right). \end{aligned} \tag{12}$$

The challenge for estimation is that $\ln \delta_{\cdot t} \leftarrow \ln \delta_{\cdot t} + \ln \mathcal{S}_{\cdot t} - \ln \mathbf{s}_{\cdot \mathbf{t}}(\delta_{\cdot t}, \theta_2)$ is no longer a contraction. Instead, the contraction must be dampened so that:[10]

$$\ln \delta_{\cdot t} \leftarrow \ln \delta_{\cdot t} + (1-\rho)\left[\ln \mathcal{S}_{\cdot t} - \ln \mathbf{s}_{\cdot \mathbf{t}}(\delta_{\cdot t}, \theta_2)\right] \tag{13}$$

This creates an additional challenge where the rate of convergence for the contraction in (13) can become arbitrarily slow as $\rho \to 1$.[11] Thus as more consumers substitute within the nest, this

[9]The nesting parameter can also be indexed as ρ_g so as to vary by group. We support both types of nesting parameters in `pyblp`.

[10]See Grigolon and Verboven (2014) for a derivation. The expression in (13) does not precisely match Grigolon and Verboven (2014) because of a typographical error in their final line.

[11]This can be formalized in terms of the modulus of the contraction mapping or the Lipschitz constant. See Dubé et al. (2012) for more details.

model becomes much harder to estimate. Our simulations will demonstrate that this can be quite problematic.

As is well known, the relation in (13) has an analytic solution in the absence of random coefficients: $\mu_{ijt} = 0$ for all (i, j, t). This model reduces to the regular nested logit for which the following expression was derived in Berry (1994):

$$\ln \delta_{jt} \equiv \ln \mathcal{S}_{jt} - \ln \mathcal{S}_{0t} - \rho \ln \mathcal{S}_{j|gt}$$

where $\mathcal{S}_{j|gt}$ is the marketshare of j in its nest g.

3. Algorithmic Improvements in `pyblp`

3.1. Solving for the Shares

The main challenge of the Nested Fixed Point (NFXP) algorithm is solving the system of marketshares: $\mathcal{S}_{jt} = s_{jt}(\delta, \theta_2)$. In the NFXP approach, θ_2 is treated as fixed, which has the immediate implication that rather than solving a system of N nonlinear equations and N unknowns in δ, we solve T systems of J_t equations and J_t unknowns in $\delta_{\cdot t}$. Independent of how we solve each system of equations, we can solve each market t in parallel.[12]

Consider a single market t, where we search for the J_t vector $\delta_{\cdot t}$ which satisfies

$$\mathcal{S}_{jt} = s_{jt}(\delta_{\cdot t} \mid \theta_2) = \int \frac{\exp[\delta_{jt} + \mu_{ijt}]}{\sum_{k \in J_t} \exp[\delta_{kt} + \mu_{ikt}]} f(\mu_{it} \mid \widetilde{\theta}_2)\, d\mu_{it} \tag{1}$$

It is impossible to choose a vector $\delta_{\cdot t}$ which solves (1) exactly. Instead, we must solve the system of equations to some tolerance. We express the tolerance in terms of the log difference in shares:

$$\left\| \ln \mathcal{S}_{jt} - \ln s_{jt}(\delta_{\cdot t}, \theta_2) \right\|_\infty \leq \epsilon^{tol}. \tag{2}$$

If the tolerance is too loose, the numerical error propagates to the rest of the estimation routine, and the tolerance should be chosen as small as possible.[13] It is also possible to set a tolerance which is too tight and thus can never be satisfied. This is particularly problematic when we sum over a large number of elements. We prefer $\epsilon^{tol} \in [10^{-12}, 10^{-14}]$ as the *machine epsilon* for 64-bit doubles is generally $\epsilon^{machine} \in [10^{-16}, 10^{-15}]$.[14]

[12]This same idea provides the sparsity of share constraints in the MPEC approach of Su and Judd (2012).

[13]Tolerance issues abound in Knittel and Metaxoglou (2014). Dubé et al. (2012) show how error propagates from the solution of (1) to the estimates of $\hat{\theta}$. Lee and Seo (2016) provide a more precise characterization of this error when Newton's method is used.

[14]Choosing a smaller tolerance may be impossible for the machine to satisfy when the scaling of the problem changes, and choosing a larger tolerance can lead to numerical errors.

Newton Approaches

A direct approach would be to solve the system of J_t equations and J_t unknowns using Newton-Type methods. Consider Newton-Raphson iteration below:[15]

$$\delta_{\cdot t}^{h+1} \hookleftarrow \delta_{\cdot t}^{h} - \lambda J_s^{-1}(\delta_{\cdot t}^{h}, \widetilde{\theta}_2) \cdot s_t(\delta_{\cdot t}^{h}, \widetilde{\theta}_2) \tag{3}$$

Each Newton-Raphson iteration would require computation of both the J_t vector of marketshares $s_t(\delta_{\cdot t}^{h}, \widetilde{\theta}_2)$, the $J_t \times J_t$ Jacobian matrix $J_s(\delta_{\cdot t}^{h}, \widetilde{\theta}_2) = \frac{\partial s_{\cdot t}}{\partial \delta_{\cdot t}}(\delta_{\cdot t}^{h}, \widetilde{\theta}_2)$, as well as its inverse $J_s^{-1}(\delta_{\cdot t}^{h})$. As in all iterative solution methods there are two primary costs: (a) the cost per iteration and (b) the number of iterations until convergence. The cost per iteration is driven primarily by the cost of computing (rather than inverting) the Jacobian matrix which involves $J_t \times J_t$ numerical integrals.[16]

There are some alternative *quasi-Newton* methods which solve variants of Section 3.1. These variants generally involve modifying the step-size λ or approximating $J_s^{-1}(\delta_{\cdot t}^{h}, \widetilde{\theta}_2)$ in ways that avoid calculating the inverse Jacobian at each step. Quasi-Newton methods are often relatively fast when they work, though they need not converge (they may oscillate, reach a resting point, etc.) and may be sensitive to starting values. Though the BLP problem is a non-convex system of nonlinear equations, there are some features which make it amenable to quasi-Newton methods. The marketshare function is a real-analytic function of $\delta_{\cdot t}$, or in other words, $s_{jt}(\delta_{\cdot,t} \mid \widetilde{\theta}_2)$ is $\mathbb{C}^{\infty}$ and bounded between $(0, 1)$ and thus it agrees with its Taylor approximation at any $\delta_{\cdot t}$. This guarantees that at least within some *basin of attraction*, quasi-Newton methods will be quadratically convergent.[17] The other useful property is that when all shares $s_{jt} < 0.5$, the Jacobian $J_s(\delta_{\cdot t}, \widetilde{\theta}_2)$ is strictly diagonally dominant, which guarantees that it is always non-singular.

Among the quasi-Newton methods considered, we find that *Levenberg-Marquardt*, which minimizes (3) in a least-squares sense using an exact version of the analytic Jacobian provides the fastest and most reliable alternative.[18] We provide details in Appendix C.

Fixed Point Approaches

Berry et al. (1995) also propose a fixed-point approach to solve the $J_t \times J_t$ system of equations in (1). They show that the following is a contraction mapping $f(\delta) = \delta$:

$$f : \delta_{\cdot t}^{h+1} \hookleftarrow \delta_{\cdot t}^{h} + \ln \mathcal{S}_{\cdot t} - \ln s_{\cdot t}(\delta_{\cdot t}^{h}, \widetilde{\theta}_2). \tag{4}$$

[15]In practice it is generally faster to solve the linear system: $J_s(\delta_{\cdot t}^{h}, \widetilde{\theta}_2)\left[\delta_{\cdot t}^{h+1} - \delta_{\cdot t}^{h}\right] = -s_t(\delta_{\cdot t}^{h}, \theta_2)$.

[16]A typical entry is $\frac{\partial s_{jt}}{\partial \delta_{kt}} = \int [1(j = k) s_{ijt}(\mu_{it}) - s_{ijt}(\mu_{it}) s_{ikt}(\mu_{it})] f(\mu_{it} \mid \widetilde{\theta}_2)\, d\mu_{it}$. The primary cost arises from the numerical integration over heterogeneous types. Even for a large market with $J_t = 1,000$ products, inverting a $1,000 \times 1,000$ matrix is easy relative to J_t^2 numerical integrals.

[17]For an example of a quasi-Newton solution to Section 3.1, see Houde (2012).

[18]We use the `MINPACK` implementation of both LM and `hybrj` (Powell's Method), along with two `scipy.optimize` implementations of Anderson acceleration and Broyden's method, which approximate the inverse Jacobian. We find that approximate Jacobian methods are slower and less reliable than fixed-point iteration and don't report them in our main results.

This kind of contraction mapping is linearly convergent where the rate of convergence is proportional to $L(\widetilde{\theta}_2)/[1 - L(\widetilde{\theta}_2)]$ where $L(\widetilde{\theta}_2)$ is the Lipschitz constant. Because (4) is a contraction, we know that $L(\widetilde{\theta}_2) < 1$. Dubé et al. (2012) show that for the BLP contraction the Lipschitz constant is defined as $L(\widetilde{\theta}_2) = \max_{\delta \in \Delta} ||I_{J_t} - \frac{\partial \log s_{\cdot t}}{\partial \delta_{\cdot t}}(\delta_{\cdot t}, \widetilde{\theta}_2)||_\infty$.

A smaller Lipschitz constant implies that (4) converges more rapidly. Dubé et al. (2012) show in simulation that all else being equal, a larger outside good share generally implies a smaller Lipschitz constant.[19] Conversely, as the outside good share becomes smaller, the convergence of the fixed point relationship takes increasingly many steps.

Accelerated Fixed Points

Given a fixed point relationship there may be faster ways to obtain a solution to $f(\delta) = \delta$ than direct iteration on the fixed point relation as in (4). There is a large literature on acceleration methods for fixed points. Most of these methods use information from multiple iterations $(\delta^h, \delta^{h+1}, \delta^{h+2}, f(\delta^h), f(f(\delta^h)))$ to approximate $J_{\mathbf{s}}$ or its inverse.[20]

Reynaerts et al. (2012) conduct extensive testing of various fixed point acceleration methods and find that the SQUAREM algorithm of Varadhan and Roland (2008) works well on the BLP contraction in (4). That algorithm is described below:[21]

$$\begin{gathered} \delta_{\cdot t}^{h+1} \leftarrow \delta_{\cdot t}^h - 2\alpha^h r^h + (\alpha^h)^2 v^h, \\ \alpha^h = \frac{(v^h)' r^h}{(v^h)' v^h}, \quad r^h = f(\delta_{\cdot t}^h) - \delta_{\cdot t}^h, \quad v^h = f(f(\delta_{\cdot t}^h)) - 2f(\delta_{\cdot t}^h) + \delta_{\cdot t}^h. \end{gathered} \tag{5}$$

In general the SQUAREM method is 4 to 8 times as fast as direct iteration on the BLP contraction in (4). The idea is to take roughly the same number of steps as Newton-Raphson iteration, but to reduce the cost of steps by avoiding calculating the Jacobian directly. In fact, all of the terms in (5) are computed as a matter of course, because these are just iterations of x^h and $f(x^h)$. Unlike direct iteration on (4), there is technically no convergence guarantee as the iteration on (5) is no longer a contraction.

The are alternative acceleration methods in addition to SQUAREM. Reynaerts et al. (2012) also consider DF-SANE which takes on the form $\delta_{\cdot t}^{h+1} \leftarrow \delta_{\cdot t}^h - \alpha^h f(\delta_{\cdot t}^h)$ with a different choice of the step-size α^h. They find performance is similar to SQUAREM though it can be slightly slower and less robust. Consistent with Reynaerts et al. (2012), we find that accelerated fixed point approaches seem to best balance reducing the number of fixed point iterations with increasing the potential cost per iteration and we use SQUAREM as our default algorithm.

[19]A simple but novel derivation. Consider the matrix $\frac{\partial \log s_{\cdot t}}{\partial \delta_{\cdot t}} = I_{J_t} - \text{diag}^{-1}(s_{\cdot t})\Gamma(\widetilde{\theta}_2)$ where element (j,k) is $1(j = k) - s_{jt}^{-1} \int s_{ijt}(\mu_{it}) s_{ikt}(\mu_{it}) f(\mu_{it} \mid \widetilde{\theta}_2)\, d\mu_{it}$. This implies that $L(\widetilde{\theta}_2) = \max_\delta [\max_j s_{jt}^{-1} \sum_k |\Gamma_{jk}(\delta_{\cdot t}, \widetilde{\theta}_2)|]$. A rough approximation to the term inside the square braces is $\max_j \sum_k |s_{kt}| \cdot |\rho(s_{ijt}, s_{ikt})| < 1 - s_{0t}$.

[20]Many of these algorithms are special cases of *Anderson Mixing.*

[21]pyblp includes a Python port of the SQUAREM package from R.

3.2. Many Fixed Effects

There is a long tradition of extending the demand side utility to incorporate product or market fixed effects. For example Nevo (2001, 2000a) allows for product fixed effects d_j, so that

$$\delta_{jt} = x_{jt}\beta - \alpha p_{jt} + d_j + \Delta\xi_{jt}.$$

These can manageably be incorporated as dummy variables in the linear part of the regression as there are only $J = 24$ products.

Backus et al. (2018) and Conlon and Rao (2017) allow for product (j) store (or chain) (s) specific fixed effects d_{js}. Using weekly UPC-store level Nielsen scanner data it is not uncommon for there to be more that $J = 3,500$ products (in both distilled spirits and ready-to-eat cereal). If one wants to allow for UPC-store fixed effects d_{js} this can explode to 100,000 or more fixed effects. Indeed, in Backus et al. (2018) the authors incorporate more than 50,000 such fixed effects. Similarly, there are approximately $T = 500$ weeks t of Nielsen scanner data from 2006-2016. If one wanted to incorporate store-week fixed effects d_{st} for only 100 stores this too could reach the order of 50,000 such fixed effects.

Clearly, the *least squares dummy variable* (LSDV) approach will not scale with tens or hundreds of thousands of fixed effects. We might consider the *within transformation* to remove the fixed effects, though we can't directly incorporate both a within transformation and a supply side without re-writing the problem because of endogenous prices p_{jt}. We show how to re-write the problem in Appendix A. Define y_{jt}^D, y_{jt}^S, x_{jt}^D, and x_{jt}^S as follows:

$$\begin{aligned} y_{jt}^D &\equiv \hat{\delta}_{jt}(\theta_2) + \alpha p_{jt} = x_{jt}\beta + \xi_t \equiv x_{jt}^D\beta + \xi_{jt}, \\ y_{jt}^S &\equiv \hat{c}_{jt}(\theta_2) \;=\; [x_{jt} \quad w_{jt}]\gamma + \omega_t \equiv x_{jt}^S\gamma + \omega_{jt}. \end{aligned}$$

Stacking the system across observations yields:[22]

$$\underbrace{\begin{bmatrix} y_D \\ y_S \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} X_D & 0 \\ 0 & X_S \end{bmatrix}}_{X} \begin{bmatrix} \beta \\ \gamma \end{bmatrix} + \begin{bmatrix} \xi \\ \omega \end{bmatrix}. \tag{6}$$

After re-arranging terms and re-stacking, this is just a conventional linear IV problem in terms of (Y, X) where the endogenous parameters have been incorporated into Y. This means that the *within transform* can be used to absorb a single dimensional fixed effect.

Consider two dimensions of fixed effects N and T where $N \gg T$:

$$\begin{aligned} \widetilde{y}_{it} &= y_{it} - \overline{y}_{i\cdot} - \overline{y}_{\cdot t}, \\ \widetilde{x}_{it} &= x_{it} - \overline{x}_{i\cdot} - \overline{x}_{\cdot t}. \end{aligned}$$

[22]Performing independent regressions implies the assumption that $\text{Cov}(\xi_{jt}, \omega_{jt}) = 0$.

The simplest approach might be to *iterative demean*: remove $\overline{x}_{i\cdot}$, update x_{it}, remove $\overline{x}_{\cdot t}$, and repeat this process until $\overline{x}_{i\cdot} = \overline{x}_{\cdot t} = 0$. This can be done in a single iteration if $\text{Cov}(\overline{x}_{\cdot t}, \overline{x}_{i\cdot}) = 0$. However, when the two dimensions of fixed effects are correlated this can take many iterations and can be quite burdensome.

The LSDV approach handles the burden of correlation but requires constructing the annihilator matrix to remove all the fixed effects. This approach requires inverting an $(N+T)\times(N+T)$ matrix. Constructing (and inverting) such a large matrix is often infeasible because of memory requirements. Several algorithms have been proposed to deal with this problem. Implemented in the Stata package `reghdfe`, the most popular algorithm is that of Correia (2016), based on the approach of Guimarães and Portugal (2010), which iteratively regresses Y on X in order to residualize Y.

The BLP application is unusual in that we re-use the x_{jt} and w_{jt} variables many times. However, the left hand side variables $\hat{\delta}_{jt}(\theta_2) + \alpha p_{jt}$ and $\hat{c}(\theta_2)$ change with each guess of θ_2, which means the entire procedure needs to be repeated for each update of θ_2.

For two-dimensional fixed effects, `pyblp` uses the algorithm of Somaini and Wolak (2016). This has the advantage that the linear explanatory variables in (X_1, X_3) can be residualized only once, while the left hand side variables (which are $N \times 1$ vectors) still need to be residualized for each guess of θ_2. The savings are particularly large when the dimensions $\dim(X_1) = K_1$ or $\dim(X_3) = K_3$ are large. For three or more dimensional fixed effects, `pyblp` uses the iterative de-meaning algorithm of Rios-Avila (2015) similar to the one described above.

3.3. Optimization and Numerical Issues

As documented by Knittel and Metaxoglou (2014), optimization of the GMM objective function for the BLP problem can be challenging. Best practices involve considering a number of different starting values and optimization routines, verifying that $\hat{\theta}$ satisfies both the first order conditions (the gradient is within a tolerance of zero) and second order conditions (the Hessian matrix has all positive eigenvalues). Both of these are reported by default in `pyblp`. Some optimization routines also allow the user to input constraints on parameters. Sometimes these constraints can speed up estimation or prevent the optimization routine from considering “unreasonable” values.

The BLP problem itself in (8) represents a non-convex optimization problem,[23] so the Hessian need not be positive semidefinite at all values of θ_2. In practice this means that no optimization routine is guaranteed to find a global minimum in a fixed amount of time. This critique applies both to derivative-based quasi-Newton approaches and to simplex-based Nelder-Mead type approaches.

The `pyblp` package uses optimization routines implemented in the open-source SciPy library and also interfaces with commercial routines such as Artleys Knitro. The Knitro software incorporates four different gradient-based search routines, and has been recommended for the BLP problem previously by Dubé et al. (2012). Table 11 lists eight different optimization routines currently sup-

[23]Absent unobserved heterogeneity or random coefficients the problem is globally convex.

ported by `pyblp`. Though non-derivative based routines such as Nelder-Mead have been frequently used in previous papers, they are not recommended.[24]

Indeed, the most important aspect of `pyblp` optimization is that it calculates the analytic gradient for any user-provided model, including models which incorporate both supply and demand moments or fixed effects.[25] Analytic gradients provide both a major speedup in computational time, while also generally improving the chances that the algorithm converges to a valid minimum.

Another important feature is that many routines allow the user to place bounds or *box constraints* on the parameter space. That is, we can restrict components of θ_2 which we call θ_ℓ to $\theta_\ell \in [\underline{\theta}_\ell, \overline{\theta}_\ell]$. Some typical constraints are that demand slopes down and that random coefficients have positive but bounded variances. One of the most common challenges for estimation routines occurs when the optimization software considers such a large value for a random coefficient that $s_{jt}(\delta_{\cdot t}, \theta_2) \to 1$ while $s_{kt}(\delta_{\cdot t}, \theta_2) \to 0$ for $k \neq j$. This constitutes a form of *overflow* error and represents one of the major reasons optimization routines can fail.[26] By restricting variances and covariances from becoming too large we can prevent many of these sorts of errors.

Another way to guarantee overflow safety is to use a protected version of the log-sum-exp (LSE) function: $\text{LSE}(x_1, \ldots, x_K) = \log \sum_k \exp x_k = a + \log \sum_k \exp(x_k - a)$. By choosing $a = \max_k x_k$, use of this function helps ensure overflow safety when evaluating the multinomial logit function. This is a well known trick from the applied mathematics and machine-learning literatures, but there does appear to be evidence of it in the literature on BLP models.

The optimization interface to `pyblp` is generic in the sense that any optimizer which can be implemented as a Python function should work fine. To illustrate this, in the documentation we give an example of such a "custom" routine where we construct a simple brute force solver that searches over a grid of parameter values. This flexibility should allow users to experiment with routines without much difficulty or "upgrade" if better routines are developed.

Our generic recommendation is that researchers try Knitro's `Interior/Direct` algorithm first if available, and then try a `BFGS`-based routine, ideally with constraints on the parameter space such as `L-BFGS-B`.[27] If a user is unwilling to place constraints on the parameter space, we find that the non-bounded `BFGS` routine seems to be the least likely to choose values for which overflow errors are an issue when compared with other free alternatives. As a generic recommendation, we suggest placing relatively restrictive bounds on the parameter space, and do not recommend non-derivative based optimization routines (i.e., Nelder-Mead) under any circumstances. We also recommend trying multiple different optimizers and starting values to check for agreement, though our simulations indicate that when properly configured, most optimizers arrive at the same parameter estimates.

[24]Both Knittel and Metaxoglou (2014) and Dubé et al. (2012) find that derivative-based routines outperform the simplex-based Nelder-Mead routine both in terms of speed and reliability. Our own tests concur with this prior opinion.

[25]See Appendix B.

[26]In this case $s_{jt}(\delta_{\cdot t}, \theta_2) = s_{jt}$ cannot be solved for $\delta_{\cdot t}$, thus the inversion for the mean utilities fails.

[27]The main disadvantage of Knitro is that it is not freely available—it must be purchased and installed by end-users.

3.4. Heterogeneity and Integration

An important aspect of the BLP model is that it incorporates heterogeneity via random coefficients. The challenge is that the integration in (3) needs to be performed numerically:

$$s_{jt}(\delta_t, \theta_2) = \int \frac{\exp[\delta_{jt} + \mu_{ijt}]}{\sum_{k \in J_t} \exp[\delta_{kt} + \mu_{ikt}]} f(\mu_{it}|\theta_2) d\,\mu_{it} \approx \sum_{i=1}^{I_t} w_i \underbrace{\frac{\exp[\delta_{jt} + \hat{\mu}_{ijt}(\nu_{it})]}{\sum_{k \in J_t} \exp[\delta_{kt} + \hat{\mu}_{ikt}(\nu_{it})]}}_{s_{ijt}(\mu_{it}(\theta_2))}. \quad (7)$$

The most common approach is Monte Carlo integration where random draws are taken from some candidate distribution $\nu_{it} \sim f(\nu_{it}|\theta_2)$ and then used to calculate $\mu_{ijt}(\nu_{it}, \theta_2)$. We can then take a weighted sum of individuals $s_{jt}(\delta_t, \theta_2) = \sum_{i=1}^{I_t} w_i s_{ijt}(\delta_t, \mu_{it}(\theta_2))$. The main advantage of Monte-Carlo integration (actually pseudo-Monte Carlo integration) is that it avoids the *curse of dimensionality.* While the accuracy for low dimensional integrals increases slowly in the number of points of approximation, the accuracy does not decline quickly as one increases the dimension of integration.

The second approach is the so-called *Gaussian quadrature* approach. Here the integrand is approximated with a polynomial and then integrated exactly as polynomial integration is trivial. In practice this is far simpler as it still amounts to a weighted sum in (7) over a particular choice of (ν_i, w_i). The main choice the user makes is the *polynomial order* of the rule. In effect the user chooses the order of the polynomial used to approximate the integrand. As the order grows, more nodes are required but the accuracy of the approximation improves.

Gaussian quadrature works best when certain conditions are met: that the integrand is bounded and continuously differentiable. Thankfully, the logit kernel in (7) is always bounded on $(0, 1)$ and is infinitely continuously differentiable (real analytic). This lends itself to quadrature-type approaches. There are a number of different flavors of quadrature rules designed to best approximate integrals under different weighting schemes. The *Gauss-Hermite* family of rules work best when $f(\mu_{it}) \propto \exp[-\mu_{it}^2]$, which (with a change of variables) includes integrals over a normal density. *Gauss-Kronod* rules offer an alternative where for a given level of polynomial accuracy p, they reuse the set of nodes from the $p-1$ polynomial accuracy integration rule. This allows for *adaptive* accuracy without wasting calculations; when the error from numerical integration is large the the polynomial degree can be expanded. Both the advantage and disadvantage of Gaussian quadrature rules is that they do a better job covering the "tail" of the probability distribution. While this increases the accuracy of the approximation, it can also lead to very large values which create *overflow* issues.[28] We prefer to use quadrature rules, and to be careful of potential overflow/underflow issues when computing shares.

The Gaussian quadrature rules apply only to a single dimension. One way to estimate higher dimensional integrals is to construct the product of single dimensional integrals. The disadvantage

[28]Essentially for some simulated individual i we have that $s_{ijt} \to 1$ and $s_{ikt} \to 0$. This problem has been previously documented by Judd and Skrainka (2011) and Skrainka (2012b).

of *product rules* is that if one needs I_t points to approximate the integral in dimension one, then one needs I_t^d points to approximate the integral in dimension d. This is the so-called *curse of dimensionality.*

The curse of dimensionality is a well-known problem in numerical analysis and several off-the-shelf solutions exist. There are several clever algorithms for improving upon the product rule for higher dimensional integration. Judd and Skrainka (2011) explore *monomial cubature rules* while Heiss and Winschel (2008) use *sparse grid* methods to selectively eliminate nodes from the product rules. One disadvantage of these methods is that they often involve weights which are negative (i.e., $w_i < 0$), which can create problems when trying to decompose the distribution of heterogeneity (particularly for counterfactuals).

Though `pyblp` allows for flexible (user-supplied) distributions of random coefficients, by far the most commonly employed choices in the literature are the independent normal and correlated normal distributions for $f(\nu_i \mid \theta_2)$. Here `pyblp` provides some specialized routines to handle these integrals with limited user intervention. There are number of different methods one can use to generate (w_i, ν_i) for the (correlated) normal case:

1. `monte_carlo`: Draw $\nu_{i\ell}$ from the standard normal distribution for each dimension of heterogeneity ℓ. Set $w_i = 1/I_t$ where I_t is the number of simulated "individuals".

2. `product`: Construct the Gauss-Hermite quadrature rule (ν_i, w_i) for a single dimension and build a product rule to obtain a set of weights and nodes in a higher dimension d.

3. `nested_product`: Construct (ν_i, w_i) using a product rule in dimension d but beginning with the basis of Gauss-Kronrod instead of Gauss-Hermite.

4. `grid`: Construct (ν_i, w_i) using a sparse grid rule of Heiss and Winschel (2008) for dimension d with the Gauss-Hermite rule as an initial basis.

5. `nested_grid`: Construct (ν_i, w_i) using a sparse grid rule in dimension d but beginning with the basis of Gauss-Kronrod instead of Gauss-Hermite.

6. Construct (ν_i, w_i) from user-provided data which allows for pseudo-Monte Carlo methods such as *Halton draws* or *Sobol sequences.*

In general, the best practice in low dimensions is probably to use product rules to a relatively high degree of polynomial accuracy. In higher dimensions, sparse grids appear to scale the best both in our own Monte Carlo studies and in those of Judd and Skrainka (2011) and Heiss and Winschel (2008).

3.5. Solving for Pricing Equilibria

Many counterfactuals of BLP-type problems involve perturbing either the market structure, marginal costs, or both, and solving for counterfactual equilibrium prices. Being able to solve for equilibrium prices quickly and accurately is also crucial to generating the optimal instruments in the next section. The Bertrand-Nash first order conditions are defined by (5) for each market t:

$$\mathbf{c} = \mathbf{p} - \underbrace{(O \odot \Omega(\mathbf{p})^{-1} s(\mathbf{p})}_{\eta(\mathbf{p},O)}$$

During estimation, in order to recover marginal costs, one need only invert the $J_t \times J_t$ matrix $O \odot \Omega(p)$. Solving for counterfactual pricing equilibria is more difficult as now we must solve the $J_t \times J_t$ nonlinear system of equations, often after replacing the ownership matrix with a post-merger counterpart, $O \to O^{post}$:

$$\mathbf{p} = \mathbf{c} + \eta^{post}(\mathbf{p}, O^{post}). \tag{8}$$

In general, solving this problem is hard as it represents a non-convex nonlinear system of equations, where one must simulate in order to compute $\eta(p, O^{post})$ and its derivatives.[29] Once one incorporates both multi-product firms and arbitrary coefficients into the problem, both existence and uniqueness of an equilibrium become challenging to establish.[30]

One approach might be to solve the system using Newton's method, which requires calculating the $J_t \times J_t$ Jacobian $\frac{\partial \eta(\mathbf{p})}{\partial \mathbf{p}}$. We provide (possibly novel) analytic expressions for this Jacobian in Appendix B.[31] The expression for the Jacobian involves the Hessian matrix of demand $\frac{\partial^2 s_k}{\partial p_j^2}$ as well as tensor products, and can be computationally challenging.[32]

The second, and perhaps most common approach in the literature is treating (8) as a fixed point and iterating on $p \hookleftarrow c + \eta^{post}(p, O^{post})$.[33] The problem is that while a fixed point of (8) may represent the Bertrand-Nash equilibrium of (6), it is not necessarily a contraction. In fact, as part of Monte Carlo experiments conducted in Armstrong (2016), the author finds that iterating on (8) does not always lead to a solution and at least some fraction of the time leads to cycles. We were able to replicate this finding for similarly constructed Monte Carlo experiments between 1-5% of the time. Were this relation to be a contraction, we should always be able to iterate on $p \hookleftarrow c + \eta^{post}(p, O^{post})$ until we reached the solution.

[29]The first Monte Carlo studies which evaluate price and quantity equilibria as part of the data generating process are likely Armstrong (2016), Skrainka (2012a) and Conlon (2017).

[30]Caplin and Nalebuff (1991) and Gallego et al. (2006) have results which apply to single product firms and linear in price utility under logit demands. Konovalov and Sandor (2010) generalizes these to logit demands with linear in price utility and multi-product firms. With the addition of random coefficients, it is possible that the resulting model will violate the quasi-concavity of the profit function that these results require. Morrow and Skerlos (2010) avoid some of these restrictions but place other restrictions on indirect utilities. Existence and uniqueness are beyond the scope of this paper—we instead focus on calculating solutions to the system of first order conditions, assuming such solutions exist.

[31]For example, Knittel and Metaxoglou (2014) do not update $s(p)$ and thus avoid fully solving the system of equations.

[32]An alternative might be to try a Newton-type approach without providing analytic formulas for the Jacobian. This seems slow and ill-advised as there are J_t markups each with J_t derivatives and each derivative involves integration in order to compute both $\Omega(p)$ and $s(p)$.

[33]See Miravete et al. (2018) for a recent example.

Our preferred approach follows Morrow and Skerlos (2011). This approach does not appear to be well known in the Industrial Organization literature, but in our experiments appears to be quite effective. They reformulate the solution to (5) by breaking up $\Omega(\mathbf{p})$ into two parts: a $J_t \times J_t$ diagonal matrix Λ, and a $J_t \times J_t$ dense matrix Γ, where $\alpha_i = \frac{\partial u_{ijt}}{\partial p_{jt}}$, the marginal (dis)-utility of price:

$$\begin{aligned}
\Omega(\mathbf{p}) &= \Lambda(\mathbf{p}) - \Gamma(\mathbf{p}), \\
\Lambda_{jj} &= \int \alpha_i s_{ijt}(\mu_{it}) f(\mu_{it} \mid \widetilde{\theta}_2)\, d\mu_{it}, \\
\Gamma_{jk} &= \int \alpha_i s_{ijt}(\mu_{it}) s_{ikt}(\mu_{it}) f(\mu_{it} \mid \widetilde{\theta}_2)\, d\mu_{it}.
\end{aligned} \tag{9}$$

Morrow and Skerlos (2011) then reformulate the problem as different fixed point:[34]

$$\mathbf{p} \leftarrow \mathbf{c} + \zeta(\mathbf{p}), \quad \zeta(\mathbf{p}) = \Lambda(\mathbf{p})^{-1}(O^{post} \odot \Gamma(\mathbf{p}))(\mathbf{p} - \mathbf{c}) - \Lambda(\mathbf{p})^{-1} s(\mathbf{p}). \tag{10}$$

The fixed point in (10) is entirely different from that in (8) and coincides only at resting points. Consistent with results reported in Morrow and Skerlos (2011), we find that (10) is around 3-12 times faster than Newton-type approaches and reliably finds an equilibrium.[35]

Perhaps most consequentially, the ability to solve for a pricing equilibrium rapidly and reliably makes it possible to generate the Amemiya (1977) or Chamberlain (1987) *optimal instruments.*

3.6. Optimal Instruments

Though this section may seem pedantic, the expression we derive differs slightly from the prior literature in a way that may have a substantial impact on the usefulness of these instruments.

As a way to improve performance, we can construct optimal instruments in the spirit of Amemiya (1977) or Chamberlain (1987). These optimal instruments were featured in both Berry et al. (1995) and Berry et al. (1999) but are not commonly employed in many subsequent studies using the BLP approach in part because they are challenging to construct. Reynaert and Verboven (2014) show the optimal instruments can improve the econometric performance of the estimator, particularly with respect to the θ_2 parameters. The form we derive is somewhat different from their expression, and incorporates both supply and demand. While the procedure itself is quite involved, the good news is that does not require much in the way of user input, and is fully implemented by the `pyblp` software.

For the GMM problem, Chamberlain (1987) tells us that the optimal instruments are related to the expected Jacobian of the moment conditions where the expectation is taken conditional only

[34]This resembles a well known "folklore" solution to the pricing problem, which is to rescale each equation by its own share s_{jt} (see Skrainka, 2012a). For the plain logit, $\Lambda_{jj}^{-1} = 1/(\alpha s_j)$.

[35]In `pyblp`, iteration is terminated with the *numerical simultaneous stationarity condition* of Morrow and Skerlos (2011) is satisfied: $||\Lambda(p)(p - c - \zeta(p))|| < \epsilon$, where ϵ is a small number and the left hand side is firms' first order conditions.

on the exogenous regressors z_t. We can write this expectation for each product j and market t as $E[D_j(z_t)\Omega^{-1} \mid z_t]$. We derive the components of the optimal instruments under the assumption that (ξ_{jt}, ω_{jt}) are jointly IID:[36]

$$D_j \equiv \underbrace{\begin{bmatrix} \frac{\partial \xi_j}{\partial \beta} & \frac{\partial \omega_j}{\partial \beta} \\ \frac{\partial \xi_j}{\partial \alpha} & \frac{\partial \omega_j}{\partial \alpha} \\ \frac{\partial \xi_j}{\partial \widetilde{\theta}_2} & \frac{\partial \omega_j}{\partial \widetilde{\theta}_2} \\ \frac{\partial \xi_j}{\partial \gamma} & \frac{\partial \omega_j}{\partial \gamma} \end{bmatrix}}_{(K_1+K_2+K_3)\times 2} = \begin{bmatrix} -x_j & 0 \\ \frac{\partial \xi_j}{\partial \alpha} & \frac{\partial \omega_j}{\partial \alpha} \\ \frac{\partial \xi_j}{\partial \widetilde{\theta}_2} & \frac{\partial \omega_j}{\partial \widetilde{\theta}_2} \\ 0 & -x_j \\ 0 & -w_j \end{bmatrix}, \quad \Omega \equiv \underbrace{\begin{bmatrix} \sigma_\xi^2 & \sigma_{\xi\omega} \\ \sigma_{\xi\omega} & \sigma_\omega^2 \end{bmatrix}}_{2\times 2}. \tag{11}$$

A little calculation shows that for each market t and each observation j,

$$D_j\Omega^{-1} = \frac{1}{\sigma_\xi^2\sigma_\omega^2 - \sigma_{\xi\omega}^2} \times \begin{bmatrix} -\sigma_\omega^2 x_j & \sigma_{\xi\omega} x_j \\ \sigma_\omega^2 \frac{\partial \xi_j}{\partial \alpha} - \sigma_{\xi\omega}\frac{\partial \omega_j}{\partial \alpha} & \sigma_\xi^2 \frac{\partial \omega_j}{\partial \alpha} - \sigma_{\xi\omega}\frac{\partial \xi_j}{\partial \alpha} \\ \sigma_\omega^2 \frac{\partial \xi_j}{\partial \widetilde{\theta}_2} - \sigma_{\xi\omega}\frac{\partial \omega_j}{\partial \widetilde{\theta}_2} & \sigma_\xi^2 \frac{\partial \omega_j}{\partial \widetilde{\theta}_2} - \sigma_{\xi\omega}\frac{\partial \xi_j}{\partial \widetilde{\theta}_2} \\ \sigma_{\xi\omega} x_j & -\sigma_\xi^2 x_j \\ \sigma_{\xi\omega} w_j & -\sigma_\xi^2 w_j \end{bmatrix}. \tag{12}$$

Clearly rows 1 and 4 are co-linear. Let Θ be a conformable matrix of zeros and ones such that

$$(D_j\Omega^{-1}) \odot \Theta = \frac{1}{\sigma_\xi^2\sigma_\omega^2 - \sigma_{\xi\omega}^2} \times \begin{bmatrix} -\sigma_\omega^2 x_{jt} & 0 \\ \sigma_\omega^2 \frac{\partial \xi_j}{\partial \alpha} - \sigma_{\xi\omega}\frac{\partial \omega_j}{\partial \alpha} & \sigma_\xi^2 \frac{\partial \omega_j}{\partial \alpha} - \sigma_{\xi\omega}\frac{\partial \xi_j}{\partial \alpha} \\ \sigma_\omega^2 \frac{\partial \xi_j}{\partial \widetilde{\theta}_2} - \sigma_{\xi\omega}\frac{\partial \omega_j}{\partial \widetilde{\theta}_2} & \sigma_\xi^2 \frac{\partial \omega_j}{\partial \widetilde{\theta}_2} - \sigma_{\xi\omega}\frac{\partial \xi_j}{\partial \widetilde{\theta}_2} \\ 0 & -\sigma_\xi^2 x_j \\ \sigma_{\xi\omega} w_j & -\sigma_\xi^2 w_j \end{bmatrix}. \tag{13}$$

We can partition our instrument set by column into "demand" and "supply" instruments:

$$z_j^D \equiv \underbrace{E[(D_j\Omega^{-1} \odot \Theta)_{\cdot 1} \mid z]}_{K_1+K_2+(K_3-K_1)}, \quad z_j^S \equiv \underbrace{E[(D_j\Omega^{-1} \odot \Theta)_{\cdot 2} \mid z]}_{K_2+K_3}. \tag{14}$$

In (14), we have $K_2 + K_3$ instruments for both supply z_j^S and demand z_j^D, though it is evident from (12) that the instruments for the endogenous parameters are not the same.[37]

Notice that when we include simultaneous supply and demand moments we have overidentifying restrictions. As shown in (14), we have $2 \times (K_2 + K_3)$ restrictions and $(K_1 + K_2 + K_3)$ parameters. This gives us $K_2 + (K_3 - K_1)$ overidentifying restrictions. The first set of K_2 overidentifying

[36]We can easily extend this formula to allow for heteroskedastic or clustered standard errors by allowing for the indexes Ω_{jt} rather than Ω for the IID case.

[37]This is true except for the knife-edge case where $\frac{\partial \xi_j}{\partial \theta_\ell} / \frac{\partial \omega_j}{\partial \theta_\ell} \propto \frac{\sigma_\xi^2 + \sigma_{\xi\sigma}}{\sigma_\omega^2 + \sigma_{\xi\sigma}}$.

restrictions comes from the fact that in (12) we have two restrictions for each of the endogenous (nonlinear) parameters $[\alpha, \widetilde{\theta}_2]$. The second set of $K_3 - K_1$ overidentifying restrictions comes from the cost shifters w_{jt} which are excluded from demand.[38] With the exception of derivatives with respect to the nonlinear (endogenous) parameters, many of the instruments are simply exogenous regressors re-scaled by covariances.

Remark 1

The set of overidentifying restrictions should be quite intuitive. Both supply and demand moments are informative about the endogenous parameters, and excluded cost shifters provide overidentifying restrictions. It is worth pointing out that our derivation of the optimal instruments appears to vary from derivations in the prior literature. Some of the prior literature using optimal instruments for BLP type problems suggests the resulting problem is *just identified* rather than *over identified.* Reynaert and Verboven (2014) appear to construct their version of optimal instruments by summing across the rows of (12) and excluding either the first or third row. This gives them $K = K_1+K_2+K_3$ instruments and K unknowns so that the model is just identified. However, because they stack (ξ, ω) they effectively have $2 \times N$ rather than N observations. Conceptually, one way to view their formulation is that it imposes $E[\xi_{jt}' z_{jt}]+E[\omega_{jt}' z_{jt}] = 0$ rather than separately imposing $E[\xi_{jt}' z_{jt}] = 0$ and $E[\omega_{jt}' z_{jt}] = 0$.

Remark 2

If one assumes that $\sigma_{\xi\omega} = 0$ or that the supply and demand shocks are uncorrelated, then the expression in (12) clearly shows that the optimal instruments for demand no longer depend on the supply Jacobian $\frac{\partial \xi_j}{\partial \theta_2}$ and likewise that the optimal instruments for supply no longer depend on the demand Jacobian $\frac{\partial \omega_j}{\partial \theta_2}$.[39] We still have overidentifying restrictions in this case as both supply and demand moments provide information on the endogenous parameters.

Constructing Feasible Instruments

The main challenge with implementing the optimal instruments is that the expectations of the Jacobian terms $E[\frac{\partial \xi_j}{\partial \theta_2}, \frac{\partial \omega_j}{\partial \theta_2} \mid z]$ are not directly measured. The most obvious example is that $\frac{\partial \xi_j}{\partial \alpha} = p_j$, but p_j is endogenous, so we must replace it with some estimate of $E[p_j \mid z]$. Here is what Berry et al. (1995) say about optimal instruments:

> *Unfortunately $D_j(z)$ is typically very difficult, if not impossible, to compute. To calculate $D_j(z)$ we would have to calculate the pricing equilibrium for different (ξ_j, ω_j) sequences, take derivatives at the equilibrium prices, and then integrate out over the distribution of such sequences. In addition, this would require an assumption that chooses among*

[38] If not all x_{jt} from the demand side are included in supply we still have overidentifying restrictions from w_{jt}.

[39] In this case the optimal weighting matrix would also have a block diagonal structure with separate blocks for demand and supply moments. The zero covariance restriction is the identification argument in MacKay and Miller (2018).

> *multiple equilibria when they exist, and either additional assumptions on the joint distribution of (ξ, ω), or a method for estimating that distribution.*

The appendix of the NBER working paper version of Berry et al. (1999) is even less positive:

> *Calculating a good estimate of $E[p \mid z]$ then requires (i) knowing or estimating the density of the unobservables and (ii) solving at some initial guess for θ the fixed point that defines equilibrium prices for each (ξ, ω) and then integrating this implicit function with respect to the density of the unknown parameters. This process is too complicated to be practical.*

Even Reynaert and Verboven (2014) focus primarily on the case of perfect competition where $E[p_{jt} \mid z] = E[c_{jt} \mid z] = [x_{jt}\ w_{jt}]\hat{\gamma} + \hat{\omega}_{jt}$. This avoids solving for equilibrium prices (and implicit functions). When they do consider imperfect competition, they construct $E[p_{jt} \mid z_t]$ by regressing the endogenous variable on a series of exogenous regressors (essentially a "first stage" regression).

We follow the possibly more accurate but costly recipe proposed by Berry et al. (1999) and show that with other computational advances in `pyblp` it is feasible to implement. For each market t we can:

1. Obtain an initial estimate $\hat{\theta} = [\hat{\beta}, \hat{\alpha}, \hat{\theta}_2, \hat{\gamma}]$.
2. Obtain an initial estimate of $\hat{\Omega}^{-1}$ by computing covariances of $(\hat{\xi}, \hat{\omega})$.
3. Draw the $J_t \times 2$ matrix of structural errors (ξ_t^*, ω_t^*) according to one of the below options.
4. Compute $\hat{y}_{jt}^S = \hat{c}_{jt} = [x_{jt}\ w_{jt}]\hat{\gamma} + \omega_{jt}^*$ and the exogenous portion of utility $\hat{y}_{jt}^D = x_{jt}\hat{\beta} + \xi_{jt}^*$.
5. Use $(\hat{y}_{jt}^D, \hat{y}_{jt}^S, \hat{\alpha}, x_t, w_t)$ to solve for equilibrium prices and quantities $(\hat{p}_t, \hat{s}_t)$ with the ζ-markup approach in Section 3.5. Note that this does not include any endogenous quantities.
6. Treating $(\hat{p}_t, \hat{s}_t, x_t, w_t)$ as data, solve for $\hat{\xi} = \xi(\hat{p}_t, \hat{s}_t, x_t, w_t, \hat{\alpha}, \hat{\theta}_2)$ and $\hat{\omega}_t = \omega(\hat{p}_t, \hat{s}_t, x_t, w_t, \hat{\alpha}, \hat{\theta}_2)$ following Section 2.1.
7. Construct the Jacobian terms $\frac{\partial \hat{\xi}_j}{\partial \theta_2}(\xi_t^*, \omega_t^*)$, $\frac{\partial \hat{\omega}_j}{\partial \theta_2}(\xi_t^*, \omega_t^*)$, and $\hat{D}_j(\xi_t^*, \omega_t^*)$ using the analytic formulas in Appendix B.
8. Average $\hat{D}_j(\xi_t^*, \omega_t^*)$ over several draws of (ξ_t^*, ω_t^*) to construct an estimate of $E[\hat{D}_j \mid z_t]$.

There are three options for generating (ξ_t^*, ω_t^*) suggested by Berry et al. (1999) and `pyblp` makes all three available:

(a) `approximate`: Replace (ξ_t^*, ω_t^*) with their expectation: $(E[\xi_t], E[\omega_t]) = (0, 0)$. This is what Berry et al. (1999) do.

(b) `asymptotic`: Estimate an asymptotic normal distribution for $(\xi_{jt}, \omega_{jt}) \sim N(0, \hat{\Omega})$ and then draw (ξ_t^*, ω_t^*) from that distribution.

(c) `empirical`: Draw (ξ_t^*, ω_t^*) from the joint empirical distribution of $(\hat{\xi}_{jt}, \hat{\omega}_{jt})$.

In general, `asymptotic` and `empirical` are not believed to be computationally feasible, particularly when there are large numbers of draws for (ξ_t^*, ω_t^*). The costly step is Item 5 above which involves solving for a new equilibrium $(\hat{p}_t, \hat{s}_t)$ for each set of draws. These improved optimal instruments are only really feasible because of the advances in Section 3.5, which drastically reduce the amount of time it takes to solve for equilibria. For relatively large problems, constructing optimal instruments may take several minutes. For smaller problems such as Berry et al. (1995) or Nevo (2000b) it takes only several seconds. Our simulations indicate that `approximate` performs as well as the more expensive `asymptotic` or `empirical` formulations. Updating results with optimal instruments in `pyblp` requires only the two lines of code in Figure 1.

Demand Side Only
The optimal instruments are much easier in the case where no supply side is specified because we no longer have a model for marginal costs and cannot solve for equilibrium $(\hat{p}_t, \hat{s}_t)$. Instead, the user can supply a vector of expected prices $E[p_t \mid z_t]$ or allow `pyblp` to construct the vector with a first-stage regression similar to Reynaert and Verboven (2014).

4. Replication Exercises

Here we provide replications using `pyblp` for the two best-known BLP applications.

Nevo (2000b) Setup
In the first replication we estimate the model of Nevo (2000b). This problem is notable because it includes a combination of observable demographics, unobserved heterogeneity, and product fixed effects.

We demonstrate how to construct the Nevo problem with `pyblp` in Figure 2. We configure the linear portion of demand β, the nonlinear portion of demand Σ, and the demographic interactions Π. We then put all three formulas together with the data.

Replication is straightforward because the original instruments are provided along with the data, and the reported estimates are from one-step GMM with the two-stage least squares (2SLS) weighting matrix $(Z'Z)^{-1}$.

We estimate the model three times. Once using the set of parameters estimated in the original Nevo (2000b) example, a second time where we restrict the interaction between the square of income y and price by setting $\Pi_{p,y^2} = 0$, and a third time using the demand side only optimal instruments described above. We report our results in Table 2. It is well known that the original reported estimates in Nevo (2000b) set the tolerance of the contraction mapping too loose. This explains the discrepancy between our "Replication" results and those originally reported. However, our replication results (with a GMM objective value of $q(\hat{\theta}) = 4.56$) are identical to those reported by Dubé et al. (2012) using the MPEC approach. The "Restricted" and "Optimal Instruments" results have smaller standard errors and lack the multicollinearity problem between income and its square in the demographic interactions.

Berry et al. (1995, 1999) Setup

For our second replication, we consider the problem in Berry et al. (1995). The distinguishing features of this problem are that it lacks demographic interactions and product fixed effects but it adds a supply side. It also allows the price coefficient to vary with individual income. As in the original paper, we estimate the model twice: once to obtain initial consistent estimates of $\hat{\theta}$, and a second time after constructing feasible optimal instruments.

Our configuration for the Berry et al. (1995) problem differs more substantially from the original paper because parts of the original configuration are not included with the data:

(a) We follow Berry et al. (1999) by replacing the original specification's $\log(y_i - p_j)$ term with its first-order linear approximation p_j/y_i.[40] We include the inverse of income, $1/y_i$, as a demographic variable, and configure Π to interact the demographic with prices. Below, we refer to the parameter on this interaction as α. This means that our price parameter is no longer directly comparable with that of the original paper.

(b) The original instruments are not provided, but $\sum_{j \in J_t} x_{jt}$ are nearly collinear. We follow Conlon (2017) by interacting these BLP instruments up to the second degree and projecting them down to the smallest number of principal components that explain at least 99% of the interactions' variance.

(c) We start the optimization routine at the published estimates and the new $\alpha = -10$.

(d) Instead of importance sampling, we use 200 pseudo-Monte Carlo draws in each market.

We provide the `pyblp` formulation of the problem in Figure 3 and the results in Table 3. With the exception of price, which we construct with the first-order approximation of Berry et al. (1999), we obtain broadly similar parameter estimates. Because the precise form of the instruments and weighting matrix are not included with the data itself, it is difficult to fully replicate the original estimates.

5. Monte Carlo Experiments

Here we provide some Monte Carlo experiments to illustrate some of the best practices we laid out in Section 3.

[40]Otherwise there are individuals for whom $p_j > y_i$ which creates a host of problems.

5.1. Monte Carlo Configuration

Our simulation configurations are loosely based on those of Armstrong (2016). The most important aspect of our simulations is that we solve for equilibrium prices and shares. For each configuration, we construct and solve 1,000 different synthetic datasets. There are $F = 5$ firms, and each produces a number of products chosen randomly from $J_f \in \{2, 5, 10\}$. There are $T = 20$ markets, and in each market, the number of firms is chosen randomly from $F_t \in \{F-2, F-1, F\} = \{3, 4, 5\}$. This procedure generates variation in the number of firms and products across markets which can be helpful for identification. Sample sizes are generally within $N \in [200, 600]$.

We draw the structural error terms (ξ_{jt}, ω_{jt}) from a mean-zero bivariate normal distribution with $\sigma_\xi^2 = \sigma_\omega^2 = 0.2$ and $\sigma_{\xi\omega} = 0.1$. Linear demand characteristics are $X_1 = [1, x, p]$ and linear supply characteristics are $X_3 = [1, x, w]$. Nonlinear characteristics are $X_2 = x$ and heterogeneity is parameterized by $\mu_{ijt} = \sigma_x x_{jt} \nu_i$ where we draw ν_i from the standard normal distribution for 1,000 different individuals in each market. We draw the two exogenous characteristics (x_{jt}, w_{jt}) from the standard uniform distribution and compute the endogenous (p_{jt}, s_{jt}) with the ζ-markup approach of Section 3.5.

Demand-side parameters are $[\beta_0, \beta_1, \alpha]' = [-7, 6, -1]'$ and $\sigma_x = 3$. Other linear parameters were chosen to generate realistic outside shares generally within $s_{0t} \in [0.8, 0.9]$. Supply-side parameters $[\gamma_0, \gamma_1, \gamma_2]' = [2, 1, 0.2]'$ enter into a linear functional form for marginal costs: $c = X_3\gamma + \omega$.

In our different Monte Carlo experiments we modify this baseline problem in a number of ways. In most experiments, we consider three variants:

(a) **Simple** is the baseline problem described above.

(a) **Complex** adds a random coefficient on price: $X_2 = [x, p]$ and $\sigma_p = 0.2$.

(a) **RCNL** adds a nesting parameter $\rho = 0.5$. Each of the J_f products produced by a firm is randomly assigned to one of two nesting groups.

Two broad classes of models are estimated: demand-only models, which are estimated with single equation GMM, and models that also include the supply side, which are estimated with multiple equation GMM.

Instruments are constructed in two stages. The initial supply instruments consist of all of the exogenous regressors. The initial demand instruments consist of the exogenous regressor x_{jt}, the cost shifter w_{jt}, and BLP Instruments, which are the sum of product characteristics for products controlled by the same firm and those controlled by competing firms:

$$\begin{aligned} Z_{jt}^{S,BLP} &= \{1, x_{jt}, w_{jt}\}, \\ Z_{jt}^{D,BLP} &= \Big\{1, x_{jt}, w_{jt}, \sum_{k \in \mathcal{J}_{ft}} 1, \sum_{k \notin \mathcal{J}_{ft}} 1, \sum_{k \in \mathcal{J}_{ft}} x_{kt}, \sum_{k \notin \mathcal{J}_{ft}} x_{kt}\Big\}. \end{aligned} \tag{1}$$

For some specifications we include the *differentiation IV* of Gandhi and Houde (2017) in place of the BLP instruments. Pooling products across all markets, for each pair of products (j, k), we construct the difference $d_{jk} = x_k - x_j$ of the exogenous regressor and then form two types of differentiation IV, which for the Simple simulation are[41]

$$\begin{aligned} Z_{jt}^{local} &= \Big\{1, x_{jt}, w_{jt}, \sum_{k \in \mathcal{J}_{ft}} 1(|d_{jk}| < \sigma_d), \sum_{k \notin \mathcal{J}_{ft}} 1(|d_{jk}| < \sigma_d)\Big\}, \\ Z_{jt}^{quad} &= \Big\{1, x_{jt}, w_{jt}, \sum_{k \in \mathcal{J}_{ft}} d_{jk}^2, \sum_{k \notin \mathcal{J}_{ft}} d_{jk}^2\Big\}. \end{aligned} \tag{2}$$

The *differentiation IV* come in two flavors: *Local* and *Quadratic*. In both cases, the differentiation IV are constructed by computing the distance in characteristic space between products j and k. The Local measure counts the number of products within a standard deviation of product j, while the Quadratic measure sums up the aggregate distance between j and other products.

Optimal instruments are constructed in three different variants following Section 3.6. When a supply side is included, the vector of expected prices is computed with the ζ-markup approach of Section 3.5. Absent a supply side, $E[p_{jt} \mid z_t]$ is estimated from a regression of endogenous prices onto all exogenous variables.

To numerically integrate choice probabilities, we use Gauss-Hermite product rules that exactly integrate polynomials of degree 17 or less.[42] In some specifications, we compare quadrature with pseudo-Monte Carlo integration and draws from the Halton sequence.

To solve the standard fixed point for $\delta_{\cdot t}$ in each market, we use the `SQUAREM` acceleration method of Varadhan and Roland (2008) with L^∞ tolerance of 10^{-14} and limit the number of contraction evaluations to 1,000. When evaluating the multinomial logit function, we use the log-sum-exp function of Section 3.1 to improve numerical stability. During the first GMM step, `SQUAREM` starts at the solution to the simple logit (or nested logit) model; in the second step, it starts at the estimated first-stage $\hat{\delta}_{\cdot t}$.

To optimize, we supply objective values and analytic gradients to a bounded limited-memory BFGS (`L-BFGS-B`) routine, which is made available by the open-source SciPy library. We use an L^∞ gradient-based tolerance of 10^{-4} and limit the number of major iterations to 1,000. Drawing different starting values from a uniform distribution with support 50% above and below the true parameter values, we solve each simulation three times and keep the solution with the smallest

[41]Following Gandhi and Houde (2017), for the Complex simulation where there is a random coefficient on price, we also include differentiation IVs constructed with the fitted values from a regression of endogenous prices onto all exogenous variables. Further following the suggestions of Gandhi and Houde (2017) and Berry (1994) for discrete variables, for the RCNL simulation, we include counts of products in the same nest.

[42]In the Simple specification when there is only one nonlinear product characteristic, the product rule is one-dimensional, with $(17 + 1)/2 = 9$ nodes. In the Complex specification, there are 9^2 nodes. If our simulations were of higher dimension, it would be more efficient to use sparse grid integration. We chose to use product rules because for these small simulation problems the gains from sparse grids are minimal. Sparse grids also have negative weights, which can introduce some estimation problems.

objective value. For each simulation, we use box constraints 1,000% above and below the true values with the following exceptions: $\sigma_x > 0$, $\sigma_p > 0$, $\rho \in [0, 0.95]$, and when a supply side is estimated, $\alpha < -0.001$.[43]

5.2. Monte Carlo Results

Fixed Effects

In Table 4 we compare computational time and memory usage for Monte Carlo experiments where we either absorb fixed effects or include them as dummy variables. As one might expect, absorbing fixed effects dramatically reduces memory requirements (by multiple orders of magnitude) and can speed up computation by as much as five times.[44] As expected, the largest improvements for the two-dimensional case are when one dimension is much larger than the other. Even absorbing relatively small numbers of fixed effects (125 in a single dimension) leads to a substantial reduction in memory usage, and substantial speedup in computational time. This is likely an important advantage of `pyblp` going forward.

Fixed Point Iteration Algorithms

To highlight the effects of different iteration schemes from Section 3.1, we explore several methods of solving the system of share equations for $\delta_{\cdot t}(\theta_2)$. We compare the conventional fixed-point iteration scheme proposed by Berry et al. (1995) to two alternative methods of solving the system of nonlinear equations without the Jacobian: `DF-SANE` and `SQUAREM`. We also compare the Jacobian based methods: Powell's method (implemented in `MINPACK` as `HYBRJ`), and Levenberg-Marquardt (LM, implemented in `MINPACK` as `LMDER`).[45]

We focus mainly on computational burden and report the results in Table 5. For the Simple and Complex simulations we vary the coefficient on the constant term β_0. A smaller value of β_0 leads to a larger share for the outside good. As shown by Dubé et al. (2012), a smaller outside good share implies a larger value for the contraction's Lipschitz constant, which generally increases the number of required iterations. Several patterns emerge. As we shrink the outside good share from $s_0 = 0.91 \to 0.26$ (and increase the Lipschitz constant) the number of required iterations increases approximately by a factor of 5 times for the Simple and Complex simulations. As expected, the Jacobian-based routines are unaffected by variation of the Lipschitz constant.[46] With the `SQUAREM` iteration scheme the number of iterations increases, but by only 90%.

In general, we find Powell's method reliable, but slightly slower than LM at lower tolerances, and it struggled with a tolerance of 10^{-14}. The `DF-SANE` algorithm performs substantially better

[43]When the optimization software considers $\alpha = 0$, the matrix of demand derivatives Ω becomes singular.

[44]We solve our simulations with a Intel Xeon E5-2697 v2 processor operating at 2.70 GHz and DDR3 DIMM operating at 1,866 MHz.

[45]We do not report results for other SciPy root-finding methods such as *Broyden's Method* or *Anderson Acceleration* because we found them too slow and unreliable to be worth considering.

[46]This is consistent with the findings of Dubé et al. (2012) using the MPEC method.

than standard fixed point iteration, but less well than our two preferred algorithms: `SQUAREM` (accelerated fixed point iteration) and LM (a Gauss-Newton solver with an analytic Jacobian).

In terms of other speedups, the effects of the `SQUAREM` iteration scheme are substantial. It reduces the number of iterations between 3-8 times without substantially increasing the cost per iteration. This is because it approximates the Newton step without actually computing the costly Jacobian. It performs well across all settings, but does particularly well in the Berry et al. (1995, 1999) example which includes a supply side.

The LM algorithm reduces the number of iterations, but at a higher cost per iteration. On some of the more difficult problems (those with a small outside good share) it performs as much as 10 times faster than fixed point iteration, and at other times roughly as fast as `SQUAREM`. The speedup is particularly large for the RCNL model, where the contraction is dampened by $(1-\rho)$. When $\rho \to 1$ the contraction becomes arbitrarily slow. In our experiments, LM performs somewhat better than the other quasi-Newton routines in Reynaerts et al. (2012) were reported to perform.[47]

Fixed Point Iteration Tricks

We also consider some commonly employed tricks in the literature to speed up the contraction mapping and report our results in Table 6. In all cases, we employ the `SQUAREM` algorithm. We consider two commonly-employed "tweaks" from the literature: working with $\exp \delta_{jt}$ rather than δ_{jt} to avoid taking logs and eliminating a division in the share computation, and using a "hot-start" where the starting value for the iterative procedure is the value of $\delta_{\cdot t}^{n-1}$ which solved the system of equations for the previous guess of θ_2. In addition, we consider the potential cost of our overflow safe modification to the log-sum-exp function.

The results in Table 6 are largely underwhelming. Once we adopt `SQUAREM` acceleration, additional tricks to speed up the problem seem to have little benefit. The "hot-start" approach was able to reduce the number of iterations between by between 10-20% which is possibly worth considering.[48] One clear recommendation is the log-sum-exp trick from Section 3.3, which reduces the chance of overflow problems. This seems relatively low-cost and reduces the possibility that the iteration routine fails to find a solution to the system of equations.

Numerical Integration

Given extensive past work by Heiss and Winschel (2008), Judd and Skrainka (2011), Skrainka (2012b), and Freyberger (2015), it is not surprising that the choice of integration method has a striking effect on the small sample performance of the estimator. We report the results of our experiments in Table 7. We compare a Gauss-Hermite product rule of degree 17 which has $I_t = 9$

[47]One possible explanation is that Reynaerts et al. (2012) employ a standard *Newton-Raphson* solver, whereas the `MINPACK-LMDER` solver uses a modified quasi-Newton routine that is designed to be more robust to poor starting values.

[48]This introduces a potential numerical problem where the GMM objective $q(\theta^h)$ need not evaluate to precisely the same quantity depending on θ^{h-1}, although for sufficient fixed point tolerances this should not be an issue. In practice, we still recommend a tight tolerance of 10^{-14}.

nodes in one dimension and $I_t = 9^2 = 81$ nodes two dimensions with Monte Carlo rules of $I_t = 100$ (roughly the same number of points) and $I_t = 1000$ (roughly the same level of accuracy). With similar numbers of nodes, we see as much as an order of magnitude reduction in bias for the σ parameters and reductions in MAE by a factor of 3-5 times particularly for the random coefficient terms (as one might expect). We see that for the same number of draws, Halton sequences perform better than Monte Carlo integration. When compared to the product rule, ten times as many Monte Carlo or Halton draws provide similar accuracy, but take 3-5 times as long to estimate.

Choice of Instruments

We also consider several different choices of instruments for our Monte Carlo simulations. We consider models which include a supply side, as well as models estimated using the demand side only. We present a slightly different construction of the Chamberlain (1987) optimal instruments for the BLP problem in Section 3.6 than those proposed in Reynaert and Verboven (2014).

In Table 8, we present simulation results using the original BLP instruments from equation (1), both the Local and Quadratic forms of the Gandhi and Houde (2017) differentiation IV from equation (2), and the `approximate` version of the feasible optimal instruments in the spirit of Chamberlain (1987) from equation (14).

We find that in most settings the feasible optimal instruments perform best, which is consistent with the findings of Reynaert and Verboven (2014). We also find that the differentiation IV substantially outperform the original BLP instruments, as Gandhi and Houde (2017) suggest.

We also find that the feasible optimal instruments perform much better when the supply side is included than with the demand side alone, while jointly estimating the supply side seems to have a limited impact under the other forms of instruments. In fact, with both optimal instruments and a supply side, the bias is all but eliminated in most of our Monte Carlo experiments, while the MAE (particularly for the random coefficients) is substantially reduced. This is not match Reynaert and Verboven (2014) who find that including the supply side has little effect once feasible optimal instruments are used.[49] The gains of included a correctly specified supply side are also valuable in estimating nonlinear functions of model parameters such as average elasticities and counterfactual simulations such as price effects of mergers. We provide additional details in Appendix E and F.

We compute the feasible optimal instruments using the three different techniques from Section 3.6: the approximate approach employed by Berry et al. (1999) where $(\xi^*, \omega^*) = (0, 0)$ (their unconditional expectation), an asymptotic approach where 100 draws are taken from the estimated normal distribution for $(\xi_{jt}, \omega_{jt}) \sim N(0, \widehat{\Omega})$, and an empirical approach where samples are taken with replacement from the joint distribution of $(\hat{\xi}_{jt}, \hat{\omega}_{jt})$. We present results in Table 9. Consistent with the prior litteraure, the approximate version of optimal instruments appears to provide nearly all of the benefits of the asymptotic or empirical versions at much lower cost.

[49]A likely important distinction is that their simulations all appear to include "strong cost shifters" in which case all estimators perform quite well, whereas some of our examples include "weak cost shifters".

We also show the value of both the supply side restrictions and the optimal instruments is largest when the exogenous cost shifter w_{jt} is weakest. We report results in Table 10 where we reduce the magnitude of the coefficient γ_2, which governs how responsive marginal costs are to the exogenous cost shifter. As we reduce this coefficient it reduces the correlation between p and w. When the cost-shifting instrument is very weak ($\rho \approx 0.05$), this increases both the bias and the variance of the price parameter α consistent with Armstrong (2016). However, if we include the supply restrictions (under optimal instruments) we are effectively able to eliminate the bias and substantially reduce the variance of the estimates. This finding appears to be novel as Reynaert and Verboven (2014) do not find substantial benefits of including supply side restrictions once optimal instruments are employed.[50]

Consistent with Gandhi and Houde (2017) our recommendation is to start with differentiation IV in the first stage and then compute feasible optimal instruments in the second stage. The substantial small sample benefits of including optimal instruments suggest that they should be employed more widely, particularly when there are multiple random coefficients. Our hope is that `pyblp` makes these instruments available to a wider set of researchers as it substantially reduces computational cost by providing analytic Jacobian calculations and by rapidly solving for the pricing equilibrium needed to generate the instruments.

Optimization Algorithms

Similar to Knittel and Metaxoglou (2014) we consider a broad array of different optimization algorithms and report our results regarding convergence in Table 11. We report our results for parameter estimates from different algorithms in Appendix Table G.13.

Our findings are somewhat different from those in Knittel and Metaxoglou (2014). Whereas those authors found that many different optimization algorithms found a variety of local minima and often failed to converge to a valid minimum, we obtain basically the opposite result. Using a broad array of optimization routines (three Knitro algorithms and five algorithms implemented in SciPy) we find that more than 99% of all simulation runs converge to a local minimum, which we define as having an L^∞ norm of the gradient sufficiently close to zero and a positive semi-definite Hessian matrix (all eigenvalues weakly positive). Even the non-derivative based Nelder-Mead algorithm, which we do not recommend, appears to do reasonably well at finding an optimum.

We observe similar success with the Nevo and BLP problems, which represent a more direct comparison with Knittel and Metaxoglou (2014). We report those results in Appendix D. When we employ multiple optimization routines and multiple starting values, we almost no dispersion in the recovered parameter estimates, objective values, and implied elasticities.[51]

[50]This may have to do with the fact that they construct optimal instruments differently for the simultaneous supply and demand problem so that the model is just identified, whereas our approach constructs them to be overidentified. An important caveat is that simultaneous supply and demand is not the main focus of their work.

[51]An important distinction between our BLP problem and the one in Knittel and Metaxoglou (2014), is that ours uses both supply and demand restrictions as in the original BLP(95/99) papers while Knittel and Metaxoglou (2014)

Our preferred algorithms are Knitro's `Interior/Direct` algorithm and `BFGS`-based algorithms in SciPy as they provide the best speed and reliability. We should caution that our simulations are simple enough to be run thousands of times, so it may not be surprising that most optimization software packages appear to work well. It may also be the case that various numerical fixes and improvements to the fixed point iteration problem may have resolved some of the issues with optimization.

Problem Size
We might also be interested in how the BLP problem scales as we vary it size. We report our results in Table 12. We find that computational time is roughly linear in the number of markets T, while it is grows at a rate closer to $\sqrt{J}$ as we increase the number of products when we use only the demand side. When we include both supply and demand the computational time appears to grow more quickly than J.

Our Monte Carlo exercises are fairly simple, but when using only demand-side restrictions, it appears as if $T = 40$ markets is roughly enough to obtain "reasonable" parameter estimates in terms of bias and efficiency, though our simulation benefit from variation in the number of products per market.

6. Conclusion

Our goal has to be to review recent methodological developments related to the BLP problem, and collect them not only in a single paper, but also in a single software package `pyblp`. We have provided a list of best practices with numerical evidence to support our recommendations. Our hope is that these practices can now be made available to more researchers, and provide a common platform which should facilitate replication.

Some of these results such as those related to numerical integration (sparse grids) are well established in the literature Skrainka (2012b) and Heiss and Winschel (2008). In other cases, we find that other algorithms (such as Levenberg-Marquardt for solving the fixed point relationship) perform as well as best in class algorithms such as the `SQUAREM` approach of Reynaerts et al. (2012) and in some cases (such as the RCNL model) substantially better. We also document how lesser-known methodologies such as the reformulation of the pricing equilibria by Morrow and Skerlos (2011) can substantially improve both speed and reliability.

In addition, we present some methodological results which we believe to be novel. We show how with a slight reformulation of the nested fixed point problem, it is possible to include high dimensional fixed effects in models with simultaneous supply and demand. We also provide a somewhat different expression for optimal instruments than the prior literature (Reynaert and Verboven, 2014) which makes clear the over-identifying restrictions implied by the supply side. Also novel, we find that optimal instruments when combined with a correctly specified supply side

uses a demand side only.

are extremely valuable. Consistent with prior work, we find the gains to optimal instruments to be substantial such that they should nearly always be employed. Thankfully, we have made this process extremely straightforward in `pyblp`.

Figure 1: Optimal Instruments Code

```
instrument_results = results.compute_optimal_instruments(method='empirical')
updated_problem = instrument_results.to_problem()
```

This code demonstrates how to use optimal instruments in **pyblp**. Solving a problem with non-optimal instruments gives a **pyblp.ProblemResults** object, which is used to estimate the optimal instruments (here, the **empirical** technique is employed). This gives a **pyblp.OptimalInstrumentResults** object that is converted to another **pyblp.Problem**, which can be solved like any other.

Figure 2: Nevo (2000b) Problem Formulation and Code

```
nevo_problem = pyblp.Problem(
    product_formulations=(
        pyblp.Formulation('0 + prices', absorb='C(product_ids)'),                    # Linear demand
        pyblp.Formulation('1 + prices + sugar + mushy')                              # Nonlinear demand
    ),
    agent_formulation=pyblp.Formulation('0 + income + income_squared + age + child'),  # Demographics
    product_data=pandas.read_csv(pyblp.data.NEVO_PRODUCTS_LOCATION),
    agent_data=pandas.read_csv(pyblp.data.NEVO_AGENTS_LOCATION)
)
print(nevo_problem)

Dimensions:
=======================================
 N     T    K1    K2    D    MD    ED
----  ---  ----  ----  ---  ----  ----
2256  94    1     4     4    20    1
=======================================

Formulations:
===================================================================
       Column Indices:           0           1           2      3
-----------------------------  ------  --------------  -----  -----
 X1: Linear Characteristics    prices
X2: Nonlinear Characteristics    1         prices      sugar  mushy
       d: Demographics         income  income_squared   age   child
===================================================================
```

This code demonstrates how to construct the problem from Nevo (2000b) in **pyblp**. For convenience, the data comes packaged in **pyblp** and can be accessed via the file paths **pyblp.data.NEVO_PRODUCTS_LOCATION** and **pyblp.data.NEVO_AGENTS_LOCATION**. Names in the product and agent formulations correspond to variable names in the datasets, which are loaded into memory with Python package **pandas**. Printing the problem object displays useful information about the configured problem. When solved, the problem gives the results reported under the "Replication" header in Table 2.

Figure 3: BLP (1995/1999) Problem Formulation and Code

```
blp_problem = pyblp.Problem(
    product_formulations=(
        pyblp.Formulation('1 + hpwt + air + mpd + space'),                    # Linear demand
        pyblp.Formulation('1 + prices + hpwt + air + mpd + space'),           # Nonlinear demand
        pyblp.Formulation('1 + log(hpwt) + air + log(mpg) + log(space) + trend')  # Supply
    ),
    agent_formulation=pyblp.Formulation('0 + I(1 / income)'),                 # Demographics
    product_data=pandas.read_csv(pyblp.data.BLP_PRODUCTS_LOCATION),
    agent_data=pandas.read_csv(pyblp.data.BLP_AGENTS_LOCATION)
)
print(blp_problem)

Dimensions:
============================================
 N     T    K1    K2    K3    D    MD    MS
----  ---  ----  ----  ----  ---  ----  ----
2217  20    5     6     6     1    11    12
============================================

Formulations:
=====================================================================================
       Column Indices:           0          1        2       3          4        5
-----------------------------  --------  ---------  -----  --------  ----------  -----
 X1: Linear Characteristics       1        hpwt      air     mpd       space
X2: Nonlinear Characteristics     1       prices    hpwt     air        mpd      space
  X3: Cost Characteristics        1      log(hpwt)   air   log(mpg)  log(space)  trend
       d: Demographics         1/income
=====================================================================================
```

This code demonstrates how to construct a modified version of the problem from Berry et al. (1995) in `pyblp`. For convenience, the data comes packaged in `pyblp` and can be accessed via the file paths `pyblp.data.BLP_PRODUCTS_LOCATION` and `pyblp.data.BLP_AGENTS_LOCATION`. Names in the product and agent formulations correspond to variable names in the datasets, which are loaded into memory with Python package `pandas`. Printing the problem object displays useful information about the configured problem. When solved, the problem gives the results reported under the "Optimal Instruments" header in Table 3.

Table 2: Nevo (2000b) Replication

		Original		Replication		Restricted		Optimal Instruments	
Parameters	Variable	Estimate	SE	Estimate	SE	Estimate	SE	Estimate	SE
Means (β)	Price	-32.433	7.743	-62.730	14.803	-32.019	2.304	-25.756	1.475
Standard Deviations (Σ)	Price	1.848	1.075	3.312	1.340	1.803	0.920	2.586	0.617
	Constant	0.377	0.129	0.558	0.163	0.375	0.120	0.221	0.074
	Sugar	0.004	0.012	-0.006	0.014	-0.004	0.012	0.023	0.007
	Mushy	0.081	0.205	0.093	0.185	0.086	0.193	0.290	0.096
Interactions (Π)	Price $\times$ Income	16.598	172.334	588.325	270.441	4.187	4.638	-9.699	2.406
	Price $\times$ Income^2	-0.659	8.955	-30.192	14.101				
	Price $\times$ Child	11.625	5.207	11.055	4.123	11.755	5.197	3.960	2.401
	Constant $\times$ Income	3.089	1.213	2.292	1.209	3.101	1.054	6.520	0.468
	Constant $\times$ Age	1.186	1.016	1.284	0.631	1.198	1.048	0.029	0.240
	Sugar $\times$ Income	-0.193	0.005	-0.385	0.121	-0.190	0.035	-0.271	0.023
	Sugar $\times$ Age	0.029	0.036	0.052	0.026	0.028	0.032	0.053	0.017
	Mushy $\times$ Income	1.468	0.697	0.748	0.802	1.495	0.648	0.817	0.259
	Mushy $\times$ Age	-1.514	1.103	-1.353	0.667	-1.539	1.107	-0.374	0.214
Mean Own-Price Elasticity (ε_{jj})				-3.618		-3.702		-3.679	
Mean Markup ($(p_{jt} - c_{jt})/p_{jt}$)				0.364		0.360		0.365	
Objective ($g'Wg$)		1.49E+01		4.56E+00		1.54E+01		8.27E-14	

This table reports replication results for the model of Nevo (2000b) described in Section 4. Under "Original," we report estimates from the original paper. Under "Replication," we report our replication results. Discrepancies can be explained by the loose contraction mapping tolerance used in the original paper. Under "Restricted," we report additional results for which we remedy a multicollinearity problem with income and its square by excluding the interaction term of the latter. Under "Optimal Instruments," we report results using optimal instruments constructed in the style of Chamberlain (1987) from initial "Restricted" estimates.

Table 3: Berry et al. (1995, 1999) Replication

		Original		Replication		PC Instruments		Optimal Instruments	
Parameters	Variable	Estimate	SE	Estimate	SE	Estimate	SE	Estimate	SE
Means (β)	Constant	-7.061	0.941	-12.595	1.654	-22.947	6.155	-8.538	1.016
	HP / weight	2.883	2.019	-6.978	6.294	4.406	3.201	1.853	2.300
	Air	1.521	0.891	1.847	3.242	0.485	4.659	1.069	1.154
	MP$	-0.122	0.320	-2.130	1.941	0.744	0.207	-1.649	0.798
	Size	3.460	0.610	4.980	0.378	4.763	0.595	1.590	0.784
Standard Deviations (Σ)	Constant	3.612	1.485	9.049	1.405	17.943	6.765	1.319	1.832
	HP / weight	4.628	1.885	17.271	7.199	2.452	5.722	3.315	5.262
	Air	1.818	1.695	-0.438	7.035	2.236	8.015	0.846	1.624
	MP$	1.050	0.272	2.929	2.247	-1.782	0.589	2.071	0.822
	Size	2.056	0.585	-9.996	1.424	-5.228	1.644	1.209	1.066
Term on Price (α)	$\ln(y-p)$	43.501	6.427	8.409	0.936	8.210	0.773	7.962	2.519
Supply Side Terms (γ)	Constant	0.952	0.194	2.499	0.183	2.470	0.182	2.423	0.178
	ln(HP / weight)	0.477	0.056	0.677	0.120	0.662	0.126	0.760	0.198
	Air	0.619	0.038	0.891	0.069	0.902	0.066	0.948	0.093
	ln(MPG)	-0.415	0.055	-0.564	0.116	-0.580	0.117	-0.616	0.153
	ln(Size)	-0.046	0.081	0.010	0.133	-0.006	0.139	0.249	0.184
	Trend	0.019	0.002	0.014	0.003	0.015	0.003	0.016	0.004
Mean Own-Price Elasticity (ε_{jj})				-3.492		-3.408		-2.939	
Mean Markup ($(p_{jt}-c_{jt})/p_{jt}$)				0.368		0.377		0.419	
Objective ($g'Wg$)				3.25E+02		3.63E+02		9.64E+01	

This table reports replication results for the model of Berry et al. (1995, 1999). Under "Original," we report estimates from the original paper. Under the following headers, we report replication results for a modified version of the original model described in Section 4. Under "Replication," we use traditional BLP instruments. Under "PC Instruments," we remedy a multicollinearity problem with the BLP instruments by following Conlon (2017) and interacting them up to a second degree before projecting them down to the smallest number of principal components that explain at least 99% of the interactions' variance. Under "Optimal Instruments," we report results using optimal instruments constructed in the style of Chamberlain (1987) from initial "PC Instruments" estimates.

Table 4: Fixed Effect Absorption

Problem	FE Dimensions	FE Levels	Absorbed	Seconds	Megabytes
Simple Simulation	1	125	No	20.39	84.75
Simple Simulation	1	125	Yes	12.95	0.66
Simple Simulation	2	125×125	No	28.55	212.31
Simple Simulation	2	125×125	Yes	13.62	1.02
Simple Simulation	2	25×625	No	55.81	642.79
Simple Simulation	2	25×625	Yes	13.67	1.05
Simple Simulation	3	$25 \times 25 \times 25$	No	20.60	37.29
Simple Simulation	3	$25 \times 25 \times 25$	Yes	15.52	1.24
Nevo Example	1	24	No	20.50	0.01
Nevo Example	1	24	Yes	19.37	0.04

This table documents the impact of fixed effect (FE) absorption on estimation speed and memory usage during one GMM step. Reported values are medians across 100 different simulations and 10 identical runs of the Nevo (2000b) example problem. When not absorbed, FEs are included as dummy variables. A single FE is absorbed with de-meaning; two, with the Somaini and Wolak (2016) algorithm; and three, with iterative de-meaning using an infinity norm tolerance of 10^{-14}. To accommodate FEs in the Simple simulation, we first set $T = 625$ and $F_t = J_f = 5$ so that there are $N = 25^3$ products. We then randomly assign $n \in \{1, \ldots, N\}$ to each product and add FEs. The 1-D case has $\beta_{n \bmod 125}$. For the "square" 2-D case, we add $\beta_{n \text{ div } 125}$. The "uneven" 2-D case has $\beta_{n \bmod 25}$ and $\beta_{n \text{ div } 25}$. The 3-D case has $\beta_{n \bmod 25}$, $\beta_{(n \text{ div } 25) \bmod 25}$, and $\beta_{n \text{ div } 125}$.

Table 5: Fixed Point Algorithms

Problem	Median s_0	Algorithm	Jacobian	Termination	Mean Milliseconds	Mean Evaluations	Percent Converged
Simple Simulation ($\beta_0 = -7$)	0.91	Iteration	No	Absolute L^∞	5.02	46.72	100.00%
Simple Simulation ($\beta_0 = -7$)	0.91	DF-SANE	No	Absolute L^∞	2.98	17.55	100.00%
Simple Simulation ($\beta_0 = -7$)	0.91	SQUAREM	No	Absolute L^∞	2.10	17.30	100.00%
Simple Simulation ($\beta_0 = -7$)	0.91	SQUAREM	No	Relative L^2	2.26	16.45	100.00%
Simple Simulation ($\beta_0 = -7$)	0.91	Powell	Yes	Relative L^2	3.07	16.31	33.50%
Simple Simulation ($\beta_0 = -7$)	0.91	LM	Yes	Relative L^2	2.23	9.04	100.00%
Simple Simulation ($\beta_0 = -1$)	0.26	Iteration	No	Absolute L^∞	24.64	230.94	100.00%
Simple Simulation ($\beta_0 = -1$)	0.26	DF-SANE	No	Absolute L^∞	6.22	38.19	100.00%
Simple Simulation ($\beta_0 = -1$)	0.26	SQUAREM	No	Absolute L^∞	4.13	34.35	100.00%
Simple Simulation ($\beta_0 = -1$)	0.26	SQUAREM	No	Relative L^2	4.55	33.68	100.00%
Simple Simulation ($\beta_0 = -1$)	0.26	Powell	Yes	Relative L^2	3.23	17.40	12.50%
Simple Simulation ($\beta_0 = -1$)	0.26	LM	Yes	Relative L^2	2.22	8.94	100.00%
Complex Simulation ($\beta_0 = -7$)	0.91	Iteration	No	Absolute L^∞	8.22	47.79	100.00%
Complex Simulation ($\beta_0 = -7$)	0.91	DF-SANE	No	Absolute L^∞	4.26	17.93	100.00%
Complex Simulation ($\beta_0 = -7$)	0.91	SQUAREM	No	Absolute L^∞	3.14	16.73	100.00%
Complex Simulation ($\beta_0 = -7$)	0.91	SQUAREM	No	Relative L^2	3.23	15.98	100.00%

Continued on the next page.

Continued from the previous page.

Problem	Median s_0	Algorithm	Jacobian	Termination	Mean Milliseconds	Mean Evaluations	Percent Converged
Complex Simulation ($\beta_0 = -7$)	0.91	Powell	Yes	Relative L^2	3.98	15.24	33.30%
Complex Simulation ($\beta_0 = -7$)	0.91	LM	Yes	Relative L^2	2.93	8.99	100.00%
Complex Simulation ($\beta_0 = -1$)	0.27	Iteration	No	Absolute L^∞	38.77	228.40	100.00%
Complex Simulation ($\beta_0 = -1$)	0.27	DF-SANE	No	Absolute L^∞	8.84	38.26	100.00%
Complex Simulation ($\beta_0 = -1$)	0.27	SQUAREM	No	Absolute L^∞	6.29	34.28	100.00%
Complex Simulation ($\beta_0 = -1$)	0.27	SQUAREM	No	Relative L^2	6.66	33.63	100.00%
Complex Simulation ($\beta_0 = -1$)	0.27	Powell	Yes	Relative L^2	4.61	17.60	10.61%
Complex Simulation ($\beta_0 = -1$)	0.27	LM	Yes	Relative L^2	2.92	8.95	100.00%
RCNL Simulation ($\rho = 0.5$)	0.91	Iteration	No	Absolute L^∞	28.98	146.57	94.50%
RCNL Simulation ($\rho = 0.5$)	0.91	DF-SANE	No	Absolute L^∞	10.29	39.31	99.26%
RCNL Simulation ($\rho = 0.5$)	0.91	SQUAREM	No	Absolute L^∞	8.87	41.46	100.00%
RCNL Simulation ($\rho = 0.5$)	0.91	SQUAREM	No	Relative L^2	8.91	38.48	100.00%
RCNL Simulation ($\rho = 0.5$)	0.91	Powell	Yes	Relative L^2	5.04	16.34	58.14%
RCNL Simulation ($\rho = 0.5$)	0.91	LM	Yes	Relative L^2	3.69	10.01	100.00%
RCNL Simulation ($\rho = 0.8$)	0.91	Iteration	No	Absolute L^∞	62.53	314.48	93.58%
RCNL Simulation ($\rho = 0.8$)	0.91	DF-SANE	No	Absolute L^∞	16.80	63.22	99.19%
RCNL Simulation ($\rho = 0.8$)	0.91	SQUAREM	No	Absolute L^∞	14.13	65.23	100.00%
RCNL Simulation ($\rho = 0.8$)	0.91	SQUAREM	No	Relative L^2	14.11	60.24	100.00%
RCNL Simulation ($\rho = 0.8$)	0.91	Powell	Yes	Relative L^2	7.07	23.11	62.26%
RCNL Simulation ($\rho = 0.8$)	0.91	LM	Yes	Relative L^2	4.89	13.50	100.00%
Nevo Example	0.54	Iteration	No	Absolute L^∞	11.20	95.70	99.97%
Nevo Example	0.54	DF-SANE	No	Absolute L^∞	4.87	27.33	100.00%
Nevo Example	0.54	SQUAREM	No	Absolute L^∞	3.44	26.16	100.00%
Nevo Example	0.54	SQUAREM	No	Relative L^2	3.60	24.72	100.00%
Nevo Example	0.54	Powell	Yes	Relative L^2	3.57	17.57	29.69%
Nevo Example	0.54	LM	Yes	Relative L^2	2.58	9.50	100.00%
BLP Example	0.89	Iteration	No	Absolute L^∞	31.74	36.80	100.00%
BLP Example	0.89	DF-SANE	No	Absolute L^∞	16.25	17.35	100.00%
BLP Example	0.89	SQUAREM	No	Absolute L^∞	16.16	18.11	100.00%
BLP Example	0.89	SQUAREM	No	Relative L^2	15.37	16.87	100.00%
BLP Example	0.89	Powell	Yes	Relative L^2	26.71	17.23	36.25%
BLP Example	0.89	LM	Yes	Relative L^2	26.18	11.54	100.00%

This table documents the impact of algorithm choice on solving the nested fixed point. Reported values are medians across 100 different simulations and 10 identical runs of the example problems. We report the number of milliseconds and contraction evaluations needed to solve the fixed point, averaged across all markets and one GMM step's objective evaluations. We also report each algorithm's convergence rate (i.e., the percent of times no numerical errors were encountered and a limit of 1,000 iterations was not reached). We configure simple iteration with an absolute L^∞ norm tolerance of 10^{-14} and compare it with `DF-SANE` and `SQUAREM`. The `MINPACK` implementations of algorithms that do require a Jacobian, a modification of the Powell hybrid method and Levenberg-Marquardt, only support relative L^2 norm tolerances, so for comparison's sake we also include `SQUAREM` with the same termination condition. Simulations are configured as in Section 5.1, except for the coefficient on the constant term, β_0, and the nesting parameter, ρ, which we vary to document the effects of decreasing the outside share s_0 and of dampening the contraction in the RCNL model.

Table 6: Fixed Point Tricks

Problem	Median s_0	Overflow Safe	Type	Initial δ	Mean Milliseconds	Mean Evaluations	Percent Converged
Simple Simulation	0.91	Yes	δ	δ^0	2.10	17.30	100.00%
Simple Simulation	0.91	No	δ	δ^0	1.82	17.30	100.00%
Simple Simulation	0.91	Yes	$\exp(\delta)$	δ^0	2.10	17.30	100.00%
Simple Simulation	0.91	No	$\exp(\delta)$	δ^0	1.85	17.30	100.00%
Simple Simulation	0.91	Yes	δ	δ^{n-1}	1.61	15.48	100.00%
Complex Simulation	0.91	Yes	δ	δ^0	3.14	16.73	100.00%
Complex Simulation	0.91	No	δ	δ^0	2.80	16.73	100.00%
Complex Simulation	0.91	Yes	$\exp(\delta)$	δ^0	3.13	16.73	100.00%
Complex Simulation	0.91	No	$\exp(\delta)$	δ^0	2.78	16.73	100.00%
Complex Simulation	0.91	Yes	δ	δ^{n-1}	2.57	14.78	100.00%
RCNL Simulation	0.91	Yes	δ	δ^0	8.87	41.46	100.00%
RCNL Simulation	0.91	No	δ	δ^0	7.86	41.46	100.00%
RCNL Simulation	0.91	Yes	$\exp(\delta)$	δ^0	8.91	41.46	100.00%
RCNL Simulation	0.91	No	$\exp(\delta)$	δ^0	7.84	41.46	100.00%
RCNL Simulation	0.91	Yes	δ	δ^{n-1}	7.03	33.50	100.00%
Nevo Example	0.54	Yes	δ	δ^0	3.44	26.16	100.00%
Nevo Example	0.54	No	δ	δ^0	3.02	26.16	100.00%
Nevo Example	0.54	Yes	$\exp(\delta)$	δ^0	3.55	26.16	100.00%
Nevo Example	0.54	No	$\exp(\delta)$	δ^0	3.00	26.16	100.00%
Nevo Example	0.54	Yes	δ	δ^{n-1}	2.73	20.95	100.00%
BLP Example	0.89	Yes	δ	δ^0	16.16	18.11	100.00%
BLP Example	0.89	No	δ	δ^0	14.47	18.08	100.00%
BLP Example	0.89	Yes	$\exp(\delta)$	δ^0	16.03	18.11	100.00%
BLP Example	0.89	No	$\exp(\delta)$	δ^0	14.48	18.08	100.00%
BLP Example	0.89	Yes	δ	δ^{n-1}	11.62	13.78	100.00%

This table documents the impact of common tricks used in the literature on solving the nested fixed point. Reported values are medians across 100 different simulations and 10 identical runs of the two example problems. We report the number of milliseconds and contraction evaluations needed to solve the nested fixed point, averaged across all markets and objective evaluations of one GMM step. We also report convergence rates (i.e., the percent of times no numerical errors were encountered a limit of 1,000 iterations was not reached). Overflow safe results are those that use the log-sum-exp (LSE) function. The two fixed point types are the standard linear contraction (over δ) and the exponentiated version (over $\exp\delta$). An initial δ^0 means that the contraction always starts at the solution to the logit model, whereas δ^{n-1} is the "hot-start" version where starting values are those that solved the fixed point for the previous guess of θ_2. We use the `SQUAREM` algorithm with an absolute L^∞ norm tolerance of 10^{-14}. Simulations are configured as in 5.1.

Table 7: Alternative Integration Methods

					True Value				Median Bias				Median Absolute Error			
Simulation	Supply	Integration	I_t	Seconds	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ
Simple	No	Monte Carlo	100	1.5	-1	3			0.165	-0.672			0.280	0.674		
Simple	No	Monte Carlo	1,000	6.2	-1	3			0.156	-0.105			0.251	0.197		
Simple	No	Halton	1,000	5.7	-1	3			0.150	0.010			0.244	0.175		
Simple	No	Product Rule	9^1	1.1	-1	3			0.157	-0.028			0.246	0.181		
Simple	Yes	Monte Carlo	100	5.9	-1	3			0.104	-0.659			0.226	0.660		
Simple	Yes	Monte Carlo	1,000	38.1	-1	3			0.027	-0.090			0.178	0.186		
Simple	Yes	Halton	1,000	36.4	-1	3			0.021	0.036			0.171	0.152		
Simple	Yes	Product Rule	9^1	4.3	-1	3			0.010	0.011			0.171	0.159		
Complex	No	Monte Carlo	100	3.1	-1	3	0.2		0.183	-0.835	-0.050		0.289	0.837	0.101	
Complex	No	Monte Carlo	1,000	15.4	-1	3	0.2		0.079	-0.200	-0.004		0.252	0.282	0.126	
Complex	No	Halton	1,000	16.6	-1	3	0.2		0.062	-0.061	0.026		0.240	0.204	0.179	
Complex	No	Product Rule	9^2	3.8	-1	3	0.2		-0.033	-0.130	0.089		0.272	0.248	0.174	
Complex	Yes	Monte Carlo	100	11.2	-1	3	0.2		0.120	-0.686	-0.135		0.277	0.750	0.153	
Complex	Yes	Monte Carlo	1,000	52.5	-1	3	0.2		0.015	-0.123	-0.049		0.205	0.234	0.128	
Complex	Yes	Halton	1,000	52.6	-1	3	0.2		-0.034	-0.017	0.032		0.194	0.169	0.141	
Complex	Yes	Product Rule	9^2	11.3	-1	3	0.2		-0.063	-0.041	0.044		0.185	0.179	0.136	
RCNL	No	Monte Carlo	100	9.2	-1	3		0.5	0.198	-0.716		0.027	0.311	0.716		0.057
RCNL	No	Monte Carlo	1,000	52.0	-1	3		0.5	0.130	-0.164		-0.001	0.241	0.223		0.025
RCNL	No	Halton	1,000	55.6	-1	3		0.5	0.118	-0.012		-0.005	0.242	0.187		0.025
RCNL	No	Product Rule	9^1	7.6	-1	3		0.5	0.111	-0.012		-0.006	0.221	0.198		0.025
RCNL	Yes	Monte Carlo	100	30.8	-1	3		0.5	0.097	-0.614		0.037	0.129	0.615		0.046
RCNL	Yes	Monte Carlo	1,000	154.2	-1	3		0.5	0.029	-0.099		0.005	0.109	0.186		0.020
RCNL	Yes	Halton	1,000	160.7	-1	3		0.5	0.017	0.010		0.000	0.114	0.173		0.018
RCNL	Yes	Product Rule	9^1	20.5	-1	3		0.5	0.008	-0.013		0.001	0.108	0.171		0.018

This table documents bias and variance of parameter estimates along with estimation speed over 1,000 simulated datasets for different numerical integration methods and numbers of draws I_t. Monte Carlo integration uses pseudo-random draws from the standard normal distribution, Halton integration uses draws from the deterministic Halton sequence, and we use a degree 17 Gauss Hermite Product Rule. For all problems, we use optimal instruments and `L-BFGS-B`. For more details, refer to Section 5.1.

Table 8: Alternative Instruments

					True Value				Median Bias				Median Absolute Error			
Simulation	Supply	Z_D	Z_S	Seconds	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ
Simple	No	$[X_1, w]$	X_3	0.1	-1	3			-0.854	-0.710			0.926	0.943		
Simple	No	$[\text{BLP}, w, X_1]$	X_3	0.8	-1	3			0.111	0.022			0.227	0.418		
Simple	No	$[\text{Local}, w, X_1]$	X_3	0.4	-1	3			0.079	0.021			0.241	0.288		
Simple	No	$[\text{Quadratic}, w, X_1]$	X_3	0.5	-1	3			0.090	-0.026			0.250	0.373		
Simple	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	1.1	-1	3			0.157	-0.028			0.246	0.181		
Simple	Yes	$[X_1, w]$	X_3	0.8	-1	3			-0.216	-0.410			0.337	0.814		
Simple	Yes	$[\text{BLP}, w, X_1]$	X_3	2.2	-1	3			0.111	0.022			0.226	0.418		
Simple	Yes	$[\text{Local}, w, X_1]$	X_3	1.1	-1	3			0.077	0.021			0.240	0.287		
Simple	Yes	$[\text{Quadratic}, w, X_1]$	X_3	1.2	-1	3			0.090	-0.026			0.251	0.373		
Simple	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	4.3	-1	3			0.010	0.011			0.171	0.159		
Complex	No	$[X_1, w]$	X_3	0.2	-1	3	0.2		-0.835	-0.657	-0.027		0.913	0.965	0.062	
Complex	No	$[\text{BLP}, w, X_1]$	X_3	2.8	-1	3	0.2		-0.014	-0.453	-0.016		0.269	0.826	0.200	
Complex	No	$[\text{Local}, w, X_1]$	X_3	1.3	-1	3	0.2		0.010	-0.185	-0.109		0.365	0.364	0.200	
Complex	No	$[\text{Quadratic}, w, X_1]$	X_3	1.2	-1	3	0.2		0.104	0.032	-0.200		0.353	0.403	0.200	
Complex	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	3.8	-1	3	0.2		-0.033	-0.130	0.089		0.272	0.248	0.174	
Complex	Yes	$[X_1, w]$	X_3	1.1	-1	3	0.2		-0.217	-0.352	-0.009		0.349	0.773	0.051	
Complex	Yes	$[\text{BLP}, w, X_1]$	X_3	7.5	-1	3	0.2		-0.030	-0.487	0.032		0.265	0.854	0.200	
Complex	Yes	$[\text{Local}, w, X_1]$	X_3	3.9	-1	3	0.2		-0.001	-0.185	-0.060		0.365	0.363	0.200	
Complex	Yes	$[\text{Quadratic}, w, X_1]$	X_3	3.5	-1	3	0.2		0.096	0.038	-0.200		0.354	0.412	0.200	
Complex	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	11.3	-1	3	0.2		-0.063	-0.041	0.044		0.185	0.179	0.136	
RCNL	No	$[X_1, w]$	X_3	0.4	-1	3		0.5	-0.846	-0.576		0.099	0.992	0.907		0.144
RCNL	No	$[\text{BLP}, w, X_1]$	X_3	5.6	-1	3		0.5	0.139	-0.717		0.028	0.274	1.168		0.063
RCNL	No	$[\text{Local}, w, X_1]$	X_3	3.0	-1	3		0.5	0.126	-0.106		0.009	0.241	0.340		0.047
RCNL	No	$[\text{Quadratic}, w, X_1]$	X_3	3.2	-1	3		0.5	0.142	-0.117		0.011	0.240	0.407		0.047
RCNL	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	7.6	-1	3		0.5	0.111	-0.012		-0.006	0.221	0.198		0.025
RCNL	Yes	$[X_1, w]$	X_3	1.9	-1	3		0.5	-0.233	-0.279		0.026	0.321	0.749		0.119
RCNL	Yes	$[\text{BLP}, w, X_1]$	X_3	11.3	-1	3		0.5	0.139	-0.705		0.028	0.272	1.168		0.063
RCNL	Yes	$[\text{Local}, w, X_1]$	X_3	6.0	-1	3		0.5	0.126	-0.104		0.009	0.240	0.339		0.047
RCNL	Yes	$[\text{Quadratic}, w, X_1]$	X_3	6.5	-1	3		0.5	0.141	-0.116		0.011	0.240	0.407		0.047
RCNL	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	20.5	-1	3		0.5	0.008	-0.013		0.001	0.108	0.171		0.018

This table documents bias and variance of parameter estimates along with estimation speed over 1,000 simulated datasets for different instruments. BLP instruments follow (1). Local and Quadratic instruments follow (2) and Gandhi and Houde (2017). Optimal instruments are the `approximate` version. For all problems, we use a degree 17 Gauss Hermite Product Rule and `L-BFGS-B`. For more details, refer to Section 5.1.

Table 9: Form of Optimal Instruments

				True Value				Median Bias				Median Absolute Error			
Simulation	Supply	Optimality	Seconds	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ
Simple	No	Approximate	1.1	-1	3			0.157	-0.028			0.246	0.181		
Simple	No	Asymptotic	4.2	-1	3			0.157	-0.024			0.246	0.181		
Simple	No	Empirical	4.2	-1	3			0.157	-0.020			0.245	0.180		
Simple	Yes	Approximate	4.3	-1	3			0.010	0.011			0.171	0.159		
Simple	Yes	Asymptotic	20.0	-1	3			0.029	0.014			0.180	0.154		
Simple	Yes	Empirical	20.1	-1	3			0.012	0.009			0.175	0.159		
Complex	No	Approximate	3.8	-1	3	0.2		-0.033	-0.130	0.089		0.272	0.248	0.174	
Complex	No	Asymptotic	7.8	-1	3	0.2		-0.025	-0.136	0.088		0.268	0.247	0.175	
Complex	No	Empirical	7.8	-1	3	0.2		-0.038	-0.133	0.089		0.266	0.255	0.177	
Complex	Yes	Approximate	11.3	-1	3	0.2		-0.063	-0.041	0.044		0.185	0.179	0.136	
Complex	Yes	Asymptotic	37.9	-1	3	0.2		-0.039	-0.041	0.015		0.229	0.224	0.131	
Complex	Yes	Empirical	37.4	-1	3	0.2		-0.045	-0.045	0.025		0.222	0.211	0.122	
RCNL	No	Approximate	7.6	-1	3		0.5	0.111	-0.012		-0.006	0.221	0.198		0.025
RCNL	No	Asymptotic	12.1	-1	3		0.5	0.108	-0.015		-0.006	0.222	0.195		0.024
RCNL	No	Empirical	12.1	-1	3		0.5	0.104	-0.010		-0.007	0.221	0.195		0.025
RCNL	Yes	Approximate	20.5	-1	3		0.5	0.008	-0.013		0.001	0.108	0.171		0.018
RCNL	Yes	Asymptotic	59.6	-1	3		0.5	0.012	-0.007		0.001	0.113	0.170		0.019
RCNL	Yes	Empirical	58.9	-1	3		0.5	0.014	-0.005		0.001	0.111	0.174		0.018

This table documents bias and variance of parameter estimates along with estimation speed over 1,000 simulated datasets for different forms of optimal instruments. The `approximate` form replaces the structural errors with their expectation. The `asymptotic` form estimates an asymptotic normal distribution for the errors and draws from that distribution. The `empirical` form draws from the joint empirical distribution of the estimated errors. For all problems, we use a degree 17 Gauss Hermite Product Rule and `L-BFGS-B`. For more details, refer to Section 5.1.

Table 10: Varying Instrument Strength

					True Value				Median Bias				Median Absolute Error			
Simulation	γ_2	Corr(p, w)	Supply	Seconds	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ
Simple	0.1	0.052	No	1.1	-1	3			0.266	-0.050			0.350	0.190		
Simple	0.1	0.052	Yes	4.5	-1	3			0.010	0.006			0.206	0.162		
Simple	0.2	0.102	No	1.1	-1	3			0.157	-0.028			0.246	0.181		
Simple	0.2	0.102	Yes	4.3	-1	3			0.010	0.011			0.171	0.159		
Simple	0.4	0.199	No	1.1	-1	3			0.051	-0.001			0.126	0.170		
Simple	0.4	0.199	Yes	4.2	-1	3			0.006	0.017			0.111	0.146		
Simple	0.8	0.376	No	1.1	-1	3			0.014	0.009			0.062	0.167		
Simple	0.8	0.376	Yes	4.3	-1	3			-0.000	0.009			0.060	0.143		
Complex	0.1	0.054	No	3.7	-1	3	0.2		0.065	-0.160	0.092		0.331	0.263	0.182	
Complex	0.1	0.054	Yes	11.5	-1	3	0.2		-0.059	-0.043	0.044		0.210	0.190	0.140	
Complex	0.2	0.104	No	3.8	-1	3	0.2		-0.033	-0.130	0.089		0.272	0.248	0.174	
Complex	0.2	0.104	Yes	11.3	-1	3	0.2		-0.063	-0.041	0.044		0.185	0.179	0.136	
Complex	0.4	0.204	No	3.7	-1	3	0.2		-0.076	-0.097	0.077		0.191	0.230	0.170	
Complex	0.4	0.204	Yes	11.3	-1	3	0.2		-0.061	-0.038	0.053		0.155	0.172	0.129	
Complex	0.8	0.384	No	3.7	-1	3	0.2		-0.089	-0.091	0.067		0.157	0.221	0.167	
Complex	0.8	0.384	Yes	12.5	-1	3	0.2		-0.057	-0.038	0.048		0.129	0.163	0.125	
RCNL	0.1	0.050	No	7.6	-1	3		0.5	0.203	-0.017		-0.010	0.309	0.196		0.027
RCNL	0.1	0.050	Yes	20.3	-1	3		0.5	0.008	-0.010		0.001	0.119	0.170		0.019
RCNL	0.2	0.097	No	7.6	-1	3		0.5	0.111	-0.012		-0.006	0.221	0.198		0.025
RCNL	0.2	0.097	Yes	20.5	-1	3		0.5	0.008	-0.013		0.001	0.108	0.171		0.018
RCNL	0.4	0.189	No	7.8	-1	3		0.5	0.041	-0.012		-0.002	0.130	0.187		0.022
RCNL	0.4	0.189	Yes	19.8	-1	3		0.5	0.008	-0.006		0.001	0.089	0.171		0.019
RCNL	0.8	0.358	No	7.8	-1	3		0.5	0.009	0.005		-0.000	0.078	0.180		0.021
RCNL	0.8	0.358	Yes	19.7	-1	3		0.5	0.005	-0.017		0.002	0.063	0.169		0.017

This table documents bias and variance of parameter estimates along with estimation speed over 1,000 simulated datasets for varying instrument strength. To increase the strength of the cost shifter w, we increase its coefficient in the equation for marginal costs and report the increasing correlation between w and prices. For all problems, we use a degree 17 Gauss Hermite Product Rule and `L-BFGS-B`. For more details, refer to Section 5.1.

Table 11: Optimization Algorithms

							Percent of Runs		Median, First GMM Step			
Simulation	Supply	P	Software	Algorithm	Gradient	Termination	Converged	PSD Hessian	Seconds	Evaluations	$q = \bar{g}'W\bar{g}$	$\|\|\nabla q\|\|_\infty$
Simple	No	1	Knitro	Interior/Direct	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	0.5	8	8.72E-15	3.03E-07
Simple	No	1	Knitro	Active Set	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	0.4	7	5.15E-15	2.27E-07
Simple	No	1	Knitro	SQP	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	0.4	8	5.35E-15	2.39E-07
Simple	No	1	SciPy	L-BFGS-B	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	0.4	6	2.57E-15	1.65E-07
Simple	No	1	SciPy	BFGS	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	0.4	7	2.69E-15	1.72E-07
Simple	No	1	SciPy	TNC	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	0.6	9	4.11E-17	1.79E-08
Simple	No	1	SciPy	TNC	Yes	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	99.9%	100.0%	0.7	11	2.11E-18	2.61E-09
Simple	No	1	SciPy	Nelder-Mead	No	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	63.9%	100.0%	24.2	123	4.75E-19	1.13E-09
Simple	Yes	2	Knitro	Interior/Direct	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	1.8	14	4.10E-01	1.07E-05
Simple	Yes	2	Knitro	Active Set	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	1.5	12	4.12E-01	1.36E-05
Simple	Yes	2	Knitro	SQP	Yes	$\|\|\nabla q\|\|_\infty$	99.9%	100.0%	1.7	13	4.10E-01	1.03E-05
Simple	Yes	2	SciPy	L-BFGS-B	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	1.1	9	4.12E-01	1.19E-05
Simple	Yes	2	SciPy	BFGS	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	1.3	10	4.12E-01	9.45E-06
Simple	Yes	2	SciPy	TNC	Yes	$\|\|\nabla q\|\|_\infty$	99.9%	100.0%	2.5	18	4.12E-01	8.92E-06
Simple	Yes	2	SciPy	TNC	Yes	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	99.9%	100.0%	2.1	15	4.12E-01	8.70E-05
Simple	Yes	2	SciPy	Nelder-Mead	No	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	54.8%	100.0%	47.8	266	4.12E-01	1.94E-07
Complex	No	3	Knitro	Interior/Direct	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	2.1	17	2.52E-12	4.65E-06
Complex	No	3	Knitro	Active Set	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.9%	1.3	14	1.96E-12	4.63E-06
Complex	No	3	Knitro	SQP	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.9%	1.8	16	2.20E-12	4.31E-06
Complex	No	3	SciPy	L-BFGS-B	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.9%	1.1	11	1.86E-12	4.15E-06
Complex	No	3	SciPy	BFGS	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	1.2	13	2.29E-12	4.53E-06
Complex	No	3	SciPy	TNC	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	2.5	24	8.17E-13	2.66E-06
Complex	No	3	SciPy	TNC	Yes	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	100.0%	99.9%	2.1	21	6.27E-11	1.51E-05
Complex	No	3	SciPy	Nelder-Mead	No	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	56.4%	100.0%	47.4	311	2.63E-18	7.97E-09
Complex	Yes	4	Knitro	Interior/Direct	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.3%	6.7	24	5.95E-01	2.24E-05
Complex	Yes	4	Knitro	Active Set	Yes	$\|\|\nabla q\|\|_\infty$	99.6%	99.3%	4.6	20	5.94E-01	2.17E-05
Complex	Yes	4	Knitro	SQP	Yes	$\|\|\nabla q\|\|_\infty$	99.7%	99.6%	7.1	24	5.94E-01	2.30E-05
Complex	Yes	4	SciPy	L-BFGS-B	Yes	$\|\|\nabla q\|\|_\infty$	99.6%	99.3%	3.8	17	5.95E-01	2.20E-05
Complex	Yes	4	SciPy	BFGS	Yes	$\|\|\nabla q\|\|_\infty$	99.6%	99.5%	3.9	18	5.95E-01	2.15E-05
Complex	Yes	4	SciPy	TNC	Yes	$\|\|\nabla q\|\|_\infty$	99.5%	99.1%	7.8	37	5.94E-01	2.00E-05
Complex	Yes	4	SciPy	TNC	Yes	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	99.7%	99.0%	5.5	27	5.96E-01	4.98E-04
Complex	Yes	4	SciPy	Nelder-Mead	No	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	46.8%	99.3%	80.6	1,001	5.93E-01	3.92E-07

Continued on the next page.

Continued from the previous page.

							Percent of Runs		Median, First GMM Step			
Simulation	Supply	P	Software	Algorithm	Gradient	Termination	Converged	PSD Hessian	Seconds	Evaluations	$q = \bar{g}'W\bar{g}$	$\lvert\lvert\nabla q\rvert\rvert_\infty$
RCNL	No	2	Knitro	Interior/Direct	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.6%	4.4	17	1.42E-13	7.86E-06
RCNL	No	2	Knitro	Active Set	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.8%	3.2	14	2.91E-14	3.05E-06
RCNL	No	2	Knitro	SQP	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.6%	4.3	16	1.59E-14	2.43E-06
RCNL	No	2	SciPy	L-BFGS-B	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.4%	2.1	11	6.48E-15	1.56E-06
RCNL	No	2	SciPy	BFGS	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.6%	2.4	13	7.85E-15	1.60E-06
RCNL	No	2	SciPy	TNC	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.4%	5.1	25	1.16E-15	8.12E-07
RCNL	No	2	SciPy	TNC	Yes	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	100.0%	99.5%	3.9	19	6.85E-11	7.20E-05
RCNL	No	2	SciPy	Nelder-Mead	No	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	54.6%	99.9%	100.1	256	9.20E-20	8.37E-09
RCNL	Yes	3	Knitro	Interior/Direct	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	9.6	24	6.05E-01	6.01E-05
RCNL	Yes	3	Knitro	Active Set	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	7.0	21	6.08E-01	2.64E-05
RCNL	Yes	3	Knitro	SQP	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	9.2	24	6.01E-01	2.63E-05
RCNL	Yes	3	SciPy	L-BFGS-B	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	4.6	15	6.08E-01	2.00E-05
RCNL	Yes	3	SciPy	BFGS	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	4.7	15	6.08E-01	1.71E-05
RCNL	Yes	3	SciPy	TNC	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	12.1	34	6.08E-01	2.35E-05
RCNL	Yes	3	SciPy	TNC	Yes	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	100.0%	100.0%	7.2	23	6.08E-01	2.80E-03
RCNL	Yes	3	SciPy	Nelder-Mead	No	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	41.9%	100.0%	149.2	1,001	6.09E-01	9.99E-07

This table documents optimization convergence statistics along with estimation speed over 1,000 simulated datasets for different optimization algorithms. For comparison's sake, we only perform one GMM step and do not set parameter bounds. We report each algorithm's convergence rate (i.e., the percent of times the algorithm reported that it successfully found an optima and the maximum number of iterations, 1,000, was not reached) and the percent of times the final Hessian matrix was positive semidefinite (i.e., that a local minima was found). We also report medians for the number of seconds needed to solve each problem, the number of objective evaluations, the final GMM objective value $q = \bar{g}'W\bar{g}$, and the L^∞ norm of the final gradient. We compare three `KNITRO` algorithms: `Interior/Direct` (an interior-point mehtod), `Active Set` (sequential linear-quadratic programming), and `SQP` (sequential quadratic programming). From the `scipy.optimize` module, we compare `L-BFGS-B` (limited-memory BFGS), `BFGS`, `TNC` (truncated Newton algorithm), and `Nelder-Mead` (Simplex algorithm). All algorithms except for `Nelder-Mead` are configured with a gradient L^∞ norm of 10^{-4}. Since `Nelder-Mead` does not support gradient-based terimation conditions, it is instead configured with a absolute parameter L^∞ norm of 10^{-4}. For comparison's sake, we include a second `TNC` algorithm configured with the same termination condition. For all problems, we use a degree 17 Gauss Hermite Product Rule and optimal instruments. For more details, refer to Section 5.1.

Table 12: Problem Scaling

					True Value				Median Bias				Median Absolute Error			
Simulation	Supply	T	J_f	Seconds	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ
Simple	No	20	$\{2, 5, 10\}$	1.1	-1	3			0.157	-0.028			0.246	0.181		
Simple	No	40	$\{2, 5, 10\}$	2.2	-1	3			0.091	-0.016			0.167	0.115		
Simple	No	100	$\{2, 5, 10\}$	5.8	-1	3			0.040	0.006			0.102	0.078		
Simple	No	20	$\{4, 10, 20\}$	1.3	-1	3			0.087	0.005			0.164	0.141		
Simple	No	20	$\{10, 25, 50\}$	2.0	-1	3			0.037	-0.018			0.117	0.104		
Simple	Yes	20	$\{2, 5, 10\}$	4.3	-1	3			0.010	0.011			0.171	0.159		
Simple	Yes	40	$\{2, 5, 10\}$	8.5	-1	3			0.014	0.008			0.122	0.116		
Simple	Yes	100	$\{2, 5, 10\}$	22.0	-1	3			0.005	0.010			0.079	0.073		
Simple	Yes	20	$\{4, 10, 20\}$	6.8	-1	3			0.008	0.019			0.149	0.134		
Simple	Yes	20	$\{10, 25, 50\}$	45.9	-1	3			0.008	-0.023			0.112	0.093		
Complex	No	20	$\{2, 5, 10\}$	3.8	-1	3	0.2		-0.033	-0.130	0.089		0.272	0.248	0.174	
Complex	No	40	$\{2, 5, 10\}$	7.9	-1	3	0.2		0.017	-0.064	0.034		0.171	0.150	0.113	
Complex	No	100	$\{2, 5, 10\}$	22.7	-1	3	0.2		0.008	-0.026	0.013		0.120	0.096	0.066	
Complex	No	20	$\{4, 10, 20\}$	5.4	-1	3	0.2		-0.047	-0.096	0.071		0.224	0.173	0.173	
Complex	No	20	$\{10, 25, 50\}$	12.1	-1	3	0.2		-0.032	-0.077	0.041		0.168	0.134	0.181	
Complex	Yes	20	$\{2, 5, 10\}$	11.3	-1	3	0.2		-0.063	-0.041	0.044		0.185	0.179	0.136	
Complex	Yes	40	$\{2, 5, 10\}$	23.0	-1	3	0.2		-0.012	-0.019	0.017		0.140	0.117	0.099	
Complex	Yes	100	$\{2, 5, 10\}$	68.6	-1	3	0.2		-0.002	-0.010	0.004		0.092	0.077	0.059	
Complex	Yes	20	$\{4, 10, 20\}$	21.9	-1	3	0.2		-0.067	-0.040	0.049		0.185	0.131	0.148	
Complex	Yes	20	$\{10, 25, 50\}$	145.0	-1	3	0.2		-0.029	-0.043	0.026		0.142	0.090	0.125	
RCNL	No	20	$\{2, 5, 10\}$	7.6	-1	3		0.5	0.111	-0.012		-0.006	0.221	0.198		0.025
RCNL	No	40	$\{2, 5, 10\}$	16.3	-1	3		0.5	0.058	-0.005		-0.003	0.163	0.136		0.016
RCNL	No	100	$\{2, 5, 10\}$	50.8	-1	3		0.5	0.026	0.006		-0.002	0.097	0.085		0.011
RCNL	No	20	$\{4, 10, 20\}$	10.5	-1	3		0.5	0.087	-0.047		0.003	0.178	0.234		0.025
RCNL	No	20	$\{10, 25, 50\}$	19.0	-1	3		0.5	0.066	-0.080		0.009	0.141	0.300		0.032
RCNL	Yes	20	$\{2, 5, 10\}$	20.5	-1	3		0.5	0.008	-0.013		0.001	0.108	0.171		0.018
RCNL	Yes	40	$\{2, 5, 10\}$	46.3	-1	3		0.5	0.006	0.000		0.000	0.076	0.116		0.014
RCNL	Yes	100	$\{2, 5, 10\}$	129.2	-1	3		0.5	-0.001	0.000		0.000	0.050	0.069		0.008
RCNL	Yes	20	$\{4, 10, 20\}$	36.2	-1	3		0.5	0.023	-0.071		0.006	0.122	0.203		0.023
RCNL	Yes	20	$\{10, 25, 50\}$	181.1	-1	3		0.5	0.028	-0.097		0.011	0.121	0.255		0.030

This table documents bias and variance of parameter estimates along with estimation speed over 1,000 simulated datasets for different problem sizes. We separately increase the number of simulated markets T and products per firm J_f. For all problems, we use a degree 17 Gauss Hermite Product Rule and `L-BFGS-B`. For more details, refer to Section 5.1.

References

AMEMIYA, T. (1977): "The Maximum Likelihood and the Nonlinear Three-Stage Least Squares Estimator in the General Nonlinear Simultaneous Equation Model," *Econometrica*, 45, 955–968.

ARMSTRONG, T. (2016): "Large Market Asymptotics for Differentiated Product Demand Estimators with Economic Models of Supply," *Econometrica*, 84, 1961–1980.

BACKUS, M., C. CONLON, AND M. SINKINSON (2018): "Common Ownership and Competition in the Ready-To-Eat Cereal Industry," Working Paper.

BAYER, P., F. FERREIRA, AND R. MCMILLAN (2007): "A Unified Framework for Measuring Preferences for Schools and Neighborhoods," *Journal of Political Economy*, 115, 588–638.

BERRY, S. (1994): "Estimating discrete-choice models of product differentiation," *RAND Journal of Economics*, 25, 242–261.

BERRY, S., J. LEVINSOHN, AND A. PAKES (1995): "Automobile Prices in Market Equilibrium," *Econometrica*, 63, 841–890.

——— (1999): "Voluntary Export Restraints on Automobiles: Evaluating a Trade Policy," *American Economic Review*, 89, 400–430.

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, 613–654.

BERRY, S. T. AND P. A. HAILE (2014): "Identification in Differentiated Products Markets Using Market Level Data," *Econometrica*, 82, 1749–1797.

BRENKERS, R. AND F. VERBOVEN (2006): "Liberalizing a Distribution System: The European Car Market," *Journal of the European Economic Association*, 4, 216–251.

CAPLIN, A. AND B. NALEBUFF (1991): "Aggregation and Imperfect Competition: On the Existence of Equilibrium," *Econometrica*, 59, 25–59.

CHAMBERLAIN, G. (1987): "Asymptotic Efficiency in Estimation with Conditional Moment Restrictions," *Journal of Econometrics*, 34, 305–334.

CONLON, C. (2017): "The MPEC Approach to Empirical Likelihood Estimation of Demand," Working Paper.

CONLON, C. AND N. RAO (2017): "The Price of Liquor is Too Damn High: The Effects of Post and Hold Pricing," Working Paper.

CORREIA, S. (2016): "Linear Models with High-Dimensional Fixed Effects: An Efficient and Feasible Estimator," Tech. rep., working Paper.

DEATON, A. S. AND J. MUELLBAUER (1980): "An Almost Ideal Demand System," *American Economic Review*, 70, 312–26.

DUBÉ, J.-P. H., J. T. FOX, AND C.-L. SU (2012): "Improving the Numerical Performance of BLP Static and Dynamic Discrete Choice Random Coefficients Demand Estimation," *Econometrica*, 80, 2231–2267.

Fan, Y. (2013): "Ownership Consolidation and Product Characteristics: A Study of the US Daily Newspaper Market," *American Economic Review*, 103, 1598–1628.

Freyberger, J. (2015): "Asymptotic theory for differentiated products demand models with many markets," *Journal of Econometrics*, 185, 162 – 181.

Gallego, G., W. T. Huh, W. Kang, and R. Phillips (2006): "Price Competition with the Attraction Demand Model: Existence of Unique Equilibrium and Its Stability," *Manufacturing & Service Operations Management*, 8, 359–375.

Gandhi, A. and J. Houde (2017): "Measuring Substitution Patterns in Differentiated Products Industries," Working Paper.

Grigolon, L. and F. Verboven (2014): "Nested Logit or Random Coefficients Logit? A Comparison of Alternative Discrete Choice Models of Product Differentiation," *The Review of Economics and Statistics*, 96, 916–935.

Guimarães, P. and P. Portugal (2010): "A simple feasible procedure to fit models with high-dimensional fixed effects," *Stata Journal*, 10, 628–649.

Heiss, F. and V. Winschel (2008): "Likelihood approximation by numerical integration on sparse grids," *Journal of Econometrics*, 144, 62 – 80.

Ho, K. and A. Pakes (2014): "Hospital Choices, Hospital Prices, and Financial Incentives to Physicians," *American Economic Review*, 104, 3841–84.

Houde, J.-F. (2012): "Spatial Differentiation and Vertical Mergers in Retail Markets for Gasoline," *American Economic Review*, 102, 2147–82.

Judd, K. L. and B. Skrainka (2011): "High performance quadrature rules: how numerical integration affects a popular model of product differentiation," CeMMAP working papers CWP03/11, Centre for Microdata Methods and Practice, Institute for Fiscal Studies.

Knittel, C. R. and K. Metaxoglou (2014): "Estimation of Random-Coefficient Demand Models: Two Empiricists' Perspective," *Review of Economics and Statistics*, 96.

Koijen, R. S. and M. Yogo (2018): "A Demand System Approach to Asset Pricing," *Journal of Political Economy*, 0, null.

Konovalov, A. and Z. Sandor (2010): "On price equilibrium with multi-product firms," *Economic Theory*, 44, 271–292.

Lee, J. and K. Seo (2015): "A computationally fast estimator for random coefficients logit demand models using aggregate data," *The RAND Journal of Economics*, 46, 86–102.

——— (2016): "Revisiting the nested fixed-point algorithm in BLP random coefficients demand estimation," *Economics Letters*, 149.

Lee, R. S. (2013): "Vertical Integration and Exclusivity in Platform and Two-Sided Markets," *American Economic Review*, 103, 2960–3000.

MacKay, A. and N. Miller (2018): "Demand Estimation in Models of Imperfect Competition," Working Paper.

Miller, N. and M. Weinberg (2017): "Understanding the Price Effects of the MillerCoors Joint Venture," *Econometrica*, 85, 1763–1791.

Miravete, E. J., K. Seim, and J. Thurk (2018): "Market Power and the Laffer Curve," *Econometrica*, 86, 1651–1687.

Morrow, W. R. and S. J. Skerlos (2010): "On the Existence of Bertrand-Nash Equilibrium Prices Under Logit Demand," *CoRR*, abs/1012.5832.

——— (2011): "Fixed-Point Approaches to Computing Bertrand-Nash Equilibrium Prices Under Mixed-Logit Demand," *Operations Research*, 59, 328–345.

Nevo, A. (2000a): "Mergers with differentiated products: the case of the ready-to-eat cereal industry," *RAND Journal of Economics*, 31, 395–421.

——— (2000b): "A Practitioner's Guide to Estimation of Random Coefficients Logit Models of Demand (including Appendix)," *Journal of Economics and Management Strategy*, 9, 513–548.

——— (2001): "Measuring Market Power in the Ready-to-Eat Cereal Industry," *Econometrica*, 69, 307–342.

Newey, W. K. and R. J. Smith (2004): "Higher Order Properties of GMM and Generalized Empirical Likelihood Estimators," *Econometrica*, 72, 219–255.

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

Reynaert, M. and F. Verboven (2014): "Improving the performance of random coefficients demand models: The role of optimal instruments," *Journal of Econometrics*, 179, 83–98.

Reynaerts, J., R. Varadhan, and J. C. Nash (2012): "Enhancing the Convergence Properties of the BLP (1995) Contraction Mapping," Vives discussion paper series 35, Katholieke Universiteit Leuven, Faculteit Economie en Bedrijfswetenschappen, Vives, http://ideas.repec.org/p/ete/vivwps/35.html.

Rios-Avila, F. (2015): "Feasible fitting of linear models with N fixed effects," *The Stata Journal*, 15, 881–898.

Salanie, B. and F. Wolak (2018): "Fast, Robust, and Approximately Correct: Estimating Mixed Demand Systems," Working Paper.

Skrainka, B. (2012a): "Three Essays on Product Differentiation," PhD dissertation, University College London.

Skrainka, B. S. (2012b): "A Large Scale Study of the Small Sample Performance of Random Coefficient Models of Demand," Working Paper.

Somaini, P. and F. Wolak (2016): "An Algorithm to Estimate the Two-Way Fixed Effects Model," *Journal of Econometric Methods*, 5, 143–152.

Su, C.-L. and K. L. Judd (2012): "Constrained optimization approaches to estimation of structural models," *Econometrica*, 80, 2213–2230.

Varadhan, R. and C. Roland (2008): "Simple and Globally Convergent Methods for Accelerating the Convergence of Any EM Algorithm," *Scandinavian Journal of Statistics*, 35, 335–353.

Appendices

A. Concentrating out Linear Parameters

Our objective is to concentrate out $[\hat{\beta}(\theta_2), \hat{\gamma}(\theta_2)]$. Define y_{jt}^D, y_{jt}^S, x_{jt}^D, and x_{jt}^S as follows:

$$\begin{aligned} y_{jt}^D &\equiv \hat{\delta}_{jt}(\theta_2) + \alpha p_{jt} = x_{jt}\beta + \xi_t \equiv x_{jt}^D \beta + \xi_{jt}, \\ y_{jt}^S &\equiv \hat{c}_{jt}(\theta_2) \; = \; [x_{jt} \quad w_{jt}]\gamma + \omega_t \equiv x_{jt}^S \gamma + \omega_{jt}. \end{aligned} \tag{A.1}$$

Stacking the system across observations yields:[52]

$$\underbrace{\begin{bmatrix} y_D \\ y_S \end{bmatrix}}_{2N\times 1} = \underbrace{\begin{bmatrix} X_D & 0 \\ 0 & X_S \end{bmatrix}}_{2N\times(K_1+K_3)} \underbrace{\begin{bmatrix} \beta \\ \gamma \end{bmatrix}}_{(K_1+K_3)\times 1} + \underbrace{\begin{bmatrix} \xi \\ \omega \end{bmatrix}}_{2N\times 1}. \tag{A.2}$$

Adding the $N \times q^D$ instruments for demand z^D and the $N \times q^S$ instruments for supply z^S yields $q = q^D + q^S$ moment restrictions:

$$g(\beta,\gamma) = E_n \begin{bmatrix} z_n^{D\prime}(y_n^D - x_n^{D\prime}\beta) \\ z_n^{S\prime}(y_n^S - x_n^{S\prime}\gamma) \end{bmatrix} = 0$$

$$\underbrace{g_n}_{q\times 1} = \underbrace{\frac{1}{N}\underbrace{\begin{bmatrix} Z_D' & 0 \\ 0 & Z_S' \end{bmatrix}}_{q\times 2N}\underbrace{\begin{bmatrix} y_D \\ y_S \end{bmatrix}}_{2N\times 1}}_{\widetilde{Y}} - \underbrace{\frac{1}{N}\underbrace{\begin{bmatrix} Z_D'X_D & 0 \\ 0 & Z_S'X_D \end{bmatrix}}_{q\times(K_1+K_3)}}_{\widetilde{X}} \underbrace{\begin{bmatrix} \beta \\ \gamma \end{bmatrix}}_{(K_1+K_3)\times 1}. \tag{A.3}$$

Now we can simply perform a GMM regression of $\widetilde{Y}$ on $\widetilde{X}$ where W is the $q \times q$ GMM weighting matrix:[53]

$$\begin{bmatrix} \hat{\beta}(\theta_2) \\ \hat{\gamma}(\theta_2) \end{bmatrix} = (\widetilde{X}'W\widetilde{X})^{-1}\widetilde{X}'W\widetilde{Y}. \tag{A.4}$$

B. Analytic Derivative Calculations

This derivation appears to be novel to the literature for the case of simultaneous estimation of supply and demand.

The gradient of the GMM objective function $q(\theta_2)$ is

$$2G_n(\theta_2)'Wg_n(\theta_2)$$

[52]Note: we cannot perform independent regressions unless we are willing to assume that $\mathrm{Cov}(\xi_{jt}, \omega_{jt}) = 0$.

[53]Observe this is the same GMM weighting matrix as for the overall problem. Also note that we require $q > K_1+K_3$ so that we have overidentifying restrictions, we need at least $K_2 = \dim(\theta_2)$ such restrictions.

The challenging piece here is the Jacobian of the GMM objective

$$G_n(\theta_2) = \frac{1}{N} \underbrace{\begin{bmatrix} Z_D' & 0 \\ 0 & Z_S' \end{bmatrix}}_{q \times 2N} \underbrace{\begin{bmatrix} \frac{\partial \xi}{\partial \theta_2} \\ \frac{\partial \omega}{\partial \theta_2} \end{bmatrix}}_{2N \times K_2},$$

which is a $q \times K_2$ matrix. Because (10) are linear, we can write:

$$\begin{bmatrix} \frac{\partial \xi}{\partial \theta_2} \\ \frac{\partial \omega}{\partial \theta_2} \end{bmatrix} = \begin{bmatrix} \frac{\partial \delta}{\partial \theta_2} \\ -f'(\cdot) \frac{\partial \eta}{\partial \theta_2} \end{bmatrix}. \tag{B.1}$$

For the demand moments, after invoking the implicit function theorem, this has a convenient block structure which can be separated market by market t:[54]

$$\underbrace{\frac{\partial \delta_{\cdot t}}{\partial \theta_2}(\theta_2)}_{J_t \times K_2} = \underbrace{\left[\frac{\partial s_{\cdot t}}{\partial \delta_{\cdot t}}(\theta_2)\right]^{-1}}_{J_t \times J_t} \times \underbrace{\left[\frac{\partial s_{\cdot t}}{\partial \theta_2}(\theta_2)\right]}_{J_t \times K_2}$$

For the supply moments, the $f'(\cdot)$ in (B.1) comes from (6) where $f(\cdot)$ is typically linear, $f'(\cdot) = 1$, or logarithmic, $f'(c) = 1/c$. Differentiating the supply moments is challenging because the matrix of demand derivatives $\Omega(p, \xi(\theta_2), \theta_2)$ depends on both $\xi(\theta_2)$ and θ_2 directly. It can be helpful to write $A(\xi(\theta_2), \theta_2) = O \odot \Omega$, and $\eta = A^{-1} \cdot s$. To avoid tensor product notation, let us consider taking the derivative with respect to an element within θ_2 which we call θ_ℓ:[55]

$$\begin{aligned}
\underbrace{\frac{\partial \eta_{\cdot t}}{\partial \theta_\ell}}_{J_t \times 1} &= -A_t^{-1} \frac{\partial A_t}{\partial \theta_\ell} A_t^{-1} s_{\cdot t} + A_t^{-1} \frac{\partial A_t}{\partial \xi} \frac{\partial \xi_{\cdot t}}{\partial \theta_\ell} A_t^{-1} s_{\cdot t} + A_t^{-1} \underbrace{\frac{\partial s_{\cdot t}}{\partial \theta_\ell}}_{0} \\
&= -\underbrace{A_t^{-1}}_{(J_t \times J_t)} \underbrace{\frac{\partial A_t}{\partial \theta_\ell}}_{(J_t \times J_t)} \underbrace{\eta_{\cdot t}}_{(J_t \times 1)} + \underbrace{A_t^{-1}}_{(J_t \times J_t)} \underbrace{\frac{\partial A_t}{\partial \xi_{\cdot t}}}_{(J_t \times J_t \times J_t)} \underbrace{\frac{\partial \xi_{\cdot t}}{\partial \theta_\ell}}_{(J_t \times 1)} \underbrace{\eta_{\cdot t}}_{(J_t \times 1)}.
\end{aligned}$$

This is expression is complicated because the supply error $\omega_{\cdot t}$ and the markup $\eta_{\cdot t}$ depend both directly on θ_2 and indirectly on θ_2 through $\xi_{\cdot t}$.

[54] $\frac{\partial \xi}{\partial \delta}$ is the identity matrix, so $\frac{\partial \xi}{\partial \theta_2} = \frac{\partial \delta}{\partial \theta_2}$. The matrix inverse of $\frac{\partial s_{\cdot t}}{\partial \delta_{\cdot t}}(\theta_2)$ is guaranteed by the diagonal dominance of system of equations with respect to δ_{jt}. As long as the outside good has a positive share, we have that for each j, $|\frac{\partial s_{jt}}{\partial \delta_{jt}}| > \sum_{k \neq j} |\frac{\partial s_{kt}}{\partial \delta_{jt}}|$. A square matrix and its transpose have the same eigenvalues and thus are both are nonsingular. In practice, as shares become small, there may still be numerical issues.

[55] In the markup η, $s_{\cdot t}$ is data and thus does not depend on parameters.

C. Levenberg-Marquardt Algorithm

The Levenberg-Marquardt (LM) algorithm minimizes the following least-squares problem in order to solve the $J_t \times J_t$ system of nonlinear equations:

$$\min_{\delta} \sum_{j=1}^{J} [\mathcal{S}_j - s_j(\delta_{\cdot t}, \theta_2)]^2.$$

The idea is to update our guess of δ to to $\delta + x$ where x is a $J \times 1$ vector. The LM update is given by the solution to the following linear system of equations:

$$\underbrace{[J_s' J_s + \lambda \operatorname{diag}(J_s' J_s)]}_{\text{approximate Hessian}}{}^{-1} J_s' \, x = J_s' [\mathcal{S}_j - s_j(\delta_{\cdot t}, \theta_2)].$$

This has the advantage that for $\lambda = 0$ the algorithm takes a full Gauss-Newton step, while for λ large it takes a step in the direction of the gradient (gradient-descent). The additional diagonal term also guarantees the invertibility of approximate Hessian.

D. Optimization Algorithms and the Example Problems

We extend our comparison of different optimization algorithms and starting values to the two example problems from Section 4. For each algorithm and example problem, we first report convergence statistics across 100 different starting values in Table D.1.[56]

Results are similar to those from solving the simulated problems. For most algorithms, convergence rates are greater than 99% and second order conditions generally always satisfied. The exceptions are `TNC` with a gradient-based norm and Nelder-Mead, which have trouble satisfying their termination conditions. Regardless, all algorithms terminate at essentially the same objective value with a small gradient norm.

In Figure D.1, we plot the distribution of objective values across algorithms and starting values. In Figures D.2 and D.3, we plot how the small amount of variation in objective values translates into differences in post-estimation outputs. Similar to Knittel and Metaxoglou (2014), we exclude values from routines that obviously failed to find a local minima. We consider the mean own-price elasticity and markup, averaged across all products and markups. As in Knittel and Metaxoglou (2014), we also consider the elasticity and markup of the top and median products in terms of marketshare. We find very little variation in post-estimation outputs.

[56]Like the simulations, we draw starting values from a uniform distribution with support 50% above and below the starting values we use in in the replications.

Table D.1: Optimization Algorithms: Example Problems

						Percent of Runs		Median, First GMM Step			
Example	P	Software	Algorithm	Gradient	Termination	Converged	PSD Hessian	Seconds	Evaluations	$q = \bar{g}'W\bar{g}$	$\lVert\nabla q\rVert_\infty$
Nevo: Replication	13	Knitro	Interior/Direct	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	173.6	150	4.56E+00	5.19E-04
Nevo: Replication	13	Knitro	Active Set	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	68.0	120	4.56E+00	3.66E-04
Nevo: Replication	13	Knitro	SQP	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	95.0	94	4.56E+00	4.13E-04
Nevo: Replication	13	SciPy	L-BFGS-B	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	929.9	1,554	4.56E+00	8.16E-05
Nevo: Replication	13	SciPy	BFGS	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	35.0	60	4.56E+00	3.60E-05
Nevo: Replication	13	SciPy	TNC	Yes	$\lVert\nabla q\rVert_\infty$	8.0%	100.0%	812.9	1,115	4.56E+00	1.78E-05
Nevo: Replication	13	SciPy	TNC	Yes	$\lVert\theta^n - \theta^{n-1}\rVert_\infty$	100.0%	100.0%	71.8	123	4.56E+00	2.29E-01
Nevo: Replication	13	SciPy	Nelder-Mead	No	$\lVert\theta^n - \theta^{n-1}\rVert_\infty$	0.0%	100.0%	3,866.6	10,001	4.56E+00	3.66E-04
Nevo: Restricted	12	Knitro	Interior/Direct	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	163.1	128	1.54E+01	3.49E-04
Nevo: Restricted	12	Knitro	Active Set	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	50.1	90	1.54E+01	3.42E-04
Nevo: Restricted	12	Knitro	SQP	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	77.5	69	1.54E+01	2.66E-04
Nevo: Restricted	12	SciPy	L-BFGS-B	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	421.8	789	1.54E+01	7.85E-05
Nevo: Restricted	12	SciPy	BFGS	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	19.7	36	1.54E+01	3.21E-05
Nevo: Restricted	12	SciPy	TNC	Yes	$\lVert\nabla q\rVert_\infty$	7.0%	100.0%	149.1	270	1.54E+01	3.43E-05
Nevo: Restricted	12	SciPy	TNC	Yes	$\lVert\theta^n - \theta^{n-1}\rVert_\infty$	100.0%	100.0%	30.3	56	1.54E+01	3.35E-01
Nevo: Restricted	12	SciPy	Nelder-Mead	No	$\lVert\theta^n - \theta^{n-1}\rVert_\infty$	14.0%	100.0%	3,506.3	10,005	1.54E+01	6.84E-05
Nevo: Optimal Instruments	12	Knitro	Interior/Direct	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	169.4	143	6.81E-11	2.78E-04
Nevo: Optimal Instruments	12	Knitro	Active Set	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	54.1	96	1.78E-10	3.36E-04
Nevo: Optimal Instruments	12	Knitro	SQP	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	84.0	73	4.05E-11	2.20E-04
Nevo: Optimal Instruments	12	SciPy	L-BFGS-B	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	371.8	678	5.75E-10	8.39E-05
Nevo: Optimal Instruments	12	SciPy	BFGS	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	17.8	33	2.30E-13	2.51E-05
Nevo: Optimal Instruments	12	SciPy	TNC	Yes	$\lVert\nabla q\rVert_\infty$	98.0%	100.0%	137.5	233	3.90E-15	3.55E-06
Nevo: Optimal Instruments	12	SciPy	TNC	Yes	$\lVert\theta^n - \theta^{n-1}\rVert_\infty$	100.0%	100.0%	25.4	50	9.96E-05	5.64E-01
Nevo: Optimal Instruments	12	SciPy	Nelder-Mead	No	$\lVert\theta^n - \theta^{n-1}\rVert_\infty$	10.0%	100.0%	3,635.4	10,004	2.84E-26	1.13E-11
BLP: Replication	6	Knitro	Interior/Direct	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	106.5	28	3.25E+02	3.08E-05
BLP: Replication	6	Knitro	Active Set	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	80.0	25	3.25E+02	2.96E-05
BLP: Replication	6	Knitro	SQP	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	94.6	25	3.25E+02	3.21E-05
BLP: Replication	6	SciPy	L-BFGS-B	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	98.5	30	3.25E+02	3.69E-05
BLP: Replication	6	SciPy	BFGS	Yes	$\lVert\nabla q\rVert_\infty$	100.0%	100.0%	74.8	22	3.25E+02	2.78E-05
BLP: Replication	6	SciPy	TNC	Yes	$\lVert\nabla q\rVert_\infty$	10.0%	94.0%	821.4	200	3.25E+02	3.68E-05
BLP: Replication	6	SciPy	TNC	Yes	$\lVert\theta^n - \theta^{n-1}\rVert_\infty$	100.0%	91.0%	302.7	87	3.25E+02	4.76E-04
BLP: Replication	6	SciPy	Nelder-Mead	No	$\lVert\theta^n - \theta^{n-1}\rVert_\infty$	21.0%	94.0%	5,211.5	10,002	3.25E+02	4.73E-06

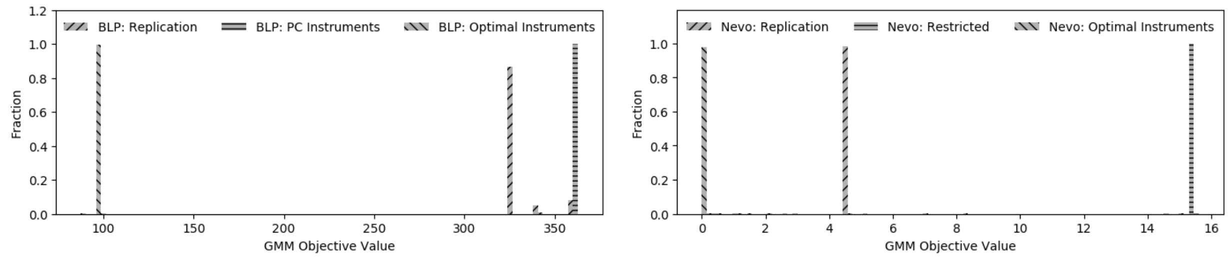
Continued on the next page.

Continued from the previous page.

						Percent of Runs		Median, First GMM Step			
Example	P	Software	Algorithm	Gradient	Termination	Converged	PSD Hessian	Seconds	Evaluations	$q = \bar{g}'W\bar{g}$	$\|\|\nabla q\|\|_\infty$
BLP: PC Instruments	6	Knitro	Interior/Direct	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	92.2	27	3.63E+02	3.38E-05
BLP: PC Instruments	6	Knitro	Active Set	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	84.4	26	3.63E+02	3.03E-05
BLP: PC Instruments	6	Knitro	SQP	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	93.4	26	3.63E+02	3.56E-05
BLP: PC Instruments	6	SciPy	L-BFGS-B	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	102.1	32	3.63E+02	3.99E-05
BLP: PC Instruments	6	SciPy	BFGS	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	83.2	25	3.63E+02	2.77E-05
BLP: PC Instruments	6	SciPy	TNC	Yes	$\|\|\nabla q\|\|_\infty$	67.0%	100.0%	703.9	186	3.63E+02	1.15E-05
BLP: PC Instruments	6	SciPy	TNC	Yes	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	100.0%	100.0%	277.7	82	3.63E+02	5.21E-04
BLP: PC Instruments	6	SciPy	Nelder-Mead	No	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	19.0%	100.0%	5,141.1	10,003	3.63E+02	2.31E-06
BLP: Optimal Instruments	6	Knitro	Interior/Direct	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	98.0%	95.0	24	9.64E+01	4.08E-05
BLP: Optimal Instruments	6	Knitro	Active Set	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	63.4	21	9.64E+01	3.20E-05
BLP: Optimal Instruments	6	Knitro	SQP	Yes	$\|\|\nabla q\|\|_\infty$	99.0%	99.0%	402.3	21	9.64E+01	2.78E-05
BLP: Optimal Instruments	6	SciPy	L-BFGS-B	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	64.2	21	9.64E+01	3.64E-05
BLP: Optimal Instruments	6	SciPy	BFGS	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	54.0	17	9.64E+01	3.26E-05
BLP: Optimal Instruments	6	SciPy	TNC	Yes	$\|\|\nabla q\|\|_\infty$	14.0%	100.0%	780.4	225	9.64E+01	2.87E-05
BLP: Optimal Instruments	6	SciPy	TNC	Yes	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	100.0%	100.0%	214.5	66	9.64E+01	8.44E-04
BLP: Optimal Instruments	6	SciPy	Nelder-Mead	No	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	26.0%	100.0%	3,309.1	10,002	9.64E+01	2.46E-06

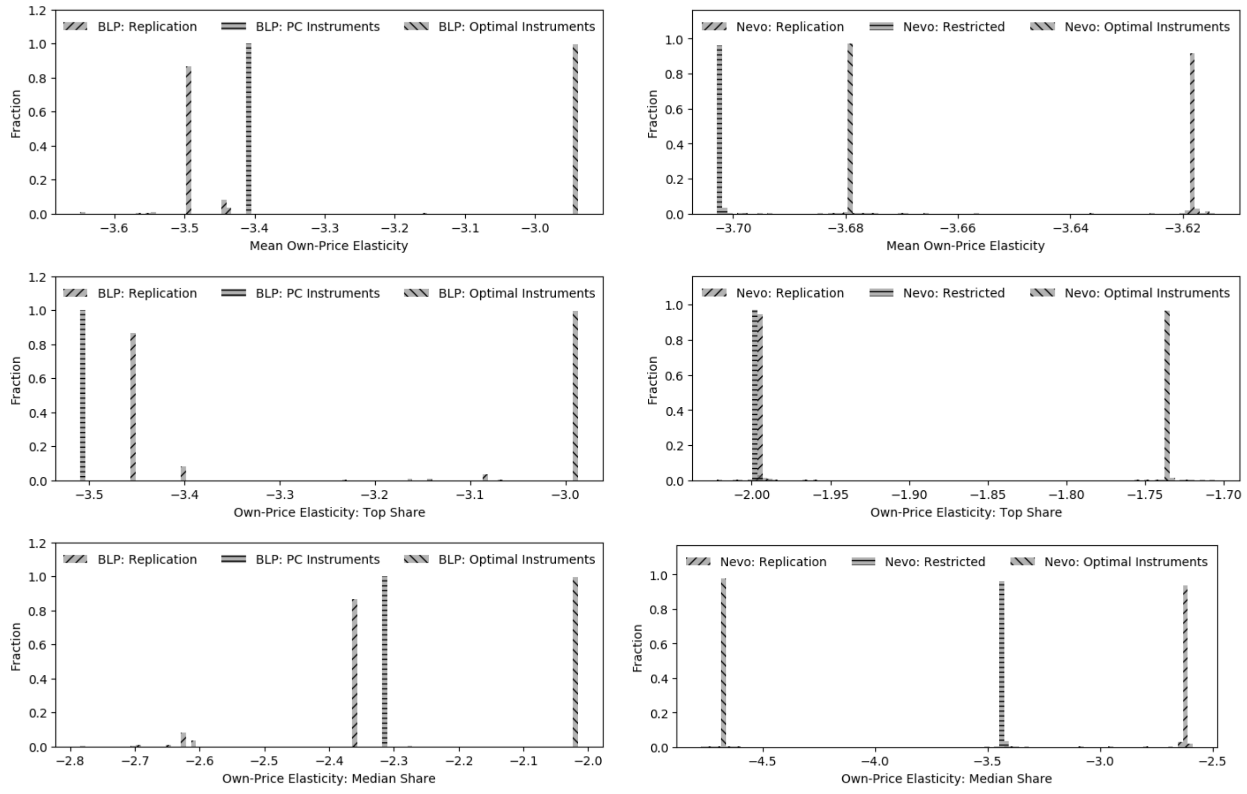
This table documents the same optimization convergence statistics as Table 11, but for the example problems solved with 100 different starting values instead of across different simulated datasets.

Figure D.1: GMM Objective Value Histograms



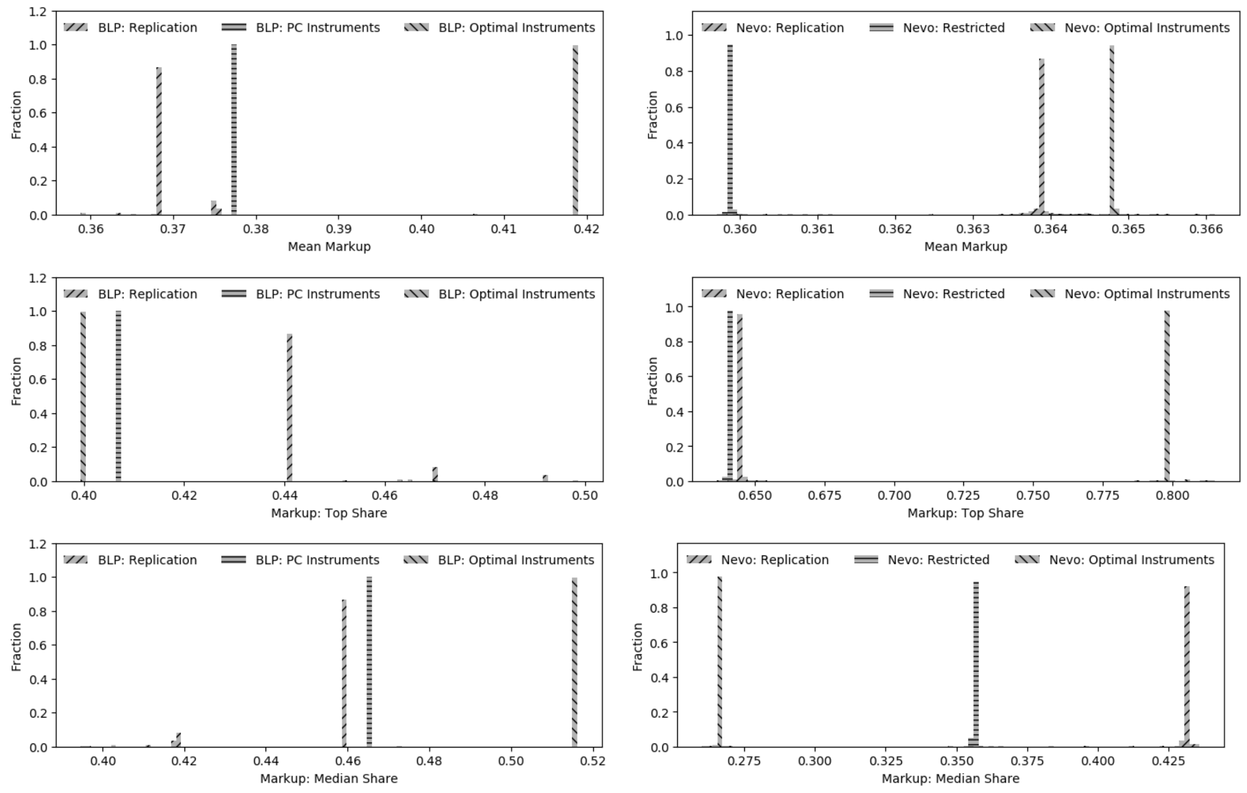
This figure plots the distribution of GMM objective values $q = g'Wg$ from Table D.1. Values for routines that failed to find a local minima are excluded.

Figure D.2: Mean Own-Price Elasticity Histograms



This figure plots distributions of own-price elasticities computed from the results used to create Table D.1. We report elasticities averaged across all products and markets, along with elasticities for the two products considered in Knittel and Metaxoglou (2014): those with the largest and median marketshares across all markets. Values for routines that failed to find a local minima are excluded.

Figure D.3: Mean Markup Histograms



This figure plots distributions of markups computed from the results used to create Table D.1. We report markups averaged across all products and markets, along with markups for the two products considered in Knittel and Metaxoglou (2014): those with the largest and median marketshares across all markets. Values for routines that failed to find a local minima are excluded.

E. Monte Carlo Results for Post-Estimation Outputs

For each set of results used to create the tables in Section 5.2, we report the bias and variance of four post-estimation outputs. Mean own-price elasticities are

$$\bar{\epsilon}_{jj} = \frac{1}{N} \sum_{j,t} \frac{\partial s_{jt}}{p_{jt}} \frac{p_{jt}}{s_{jt}}.$$

Mean aggregate price elasticities are

$$\bar{E} = \frac{1}{N} \sum_{j,t} \frac{s_{jt}(p_{jt} + \Delta p_{jt}) - s_{jt}}{\Delta}$$

where we scale prices by $\Delta = 0.01$. Total producer surplus is

$$\text{PS} = \sum_{j,t} p_{jt} - c_{jt} s_{jt}. \tag{E.1}$$

Total consumer surplus is

$$\text{CS} = \sum_{t,i} w_i \frac{\log(1 + \sum_{j=t}^{J_t} \exp[\delta_{jt} + \mu_{jti}])}{\alpha_i}. \tag{E.2}$$

Table E.2: Alternative Integration Methods: Post-Estimation

					True Median Value				Median Bias				Median Absolute Error			
Simulation	Supply	Integration	I_t	Seconds	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS
Simple	No	Monte Carlo	100	1.5	-3.564	-0.241	2.651	3.653	0.574	0.004	0.333	0.043	0.988	0.065	0.755	1.017
Simple	No	Monte Carlo	1,000	6.2	-3.564	-0.241	2.651	3.653	0.554	0.031	0.461	0.514	0.894	0.056	0.712	0.954
Simple	No	Halton	1,000	5.7	-3.564	-0.241	2.651	3.653	0.533	0.033	0.433	0.502	0.863	0.055	0.688	0.912
Simple	No	Product Rule	9^1	1.1	-3.564	-0.241	2.651	3.653	0.558	0.032	0.463	0.610	0.874	0.055	0.687	0.925
Simple	Yes	Monte Carlo	100	5.9	-3.564	-0.241	2.651	3.653	0.353	-0.007	0.207	-0.060	0.793	0.050	0.574	0.731
Simple	Yes	Monte Carlo	1,000	38.1	-3.564	-0.241	2.651	3.653	0.100	0.004	0.073	0.057	0.633	0.039	0.455	0.611
Simple	Yes	Halton	1,000	36.4	-3.564	-0.241	2.651	3.653	0.076	0.006	0.059	0.042	0.612	0.038	0.442	0.597
Simple	Yes	Product Rule	9^1	4.3	-3.564	-0.241	2.651	3.653	0.041	0.001	0.038	0.079	0.608	0.038	0.446	0.621
Complex	No	Monte Carlo	100	3.1	-3.303	-0.227	2.945	4.500	0.651	0.007	0.507	0.002	0.966	0.066	0.939	1.443
Complex	No	Monte Carlo	1,000	15.4	-3.303	-0.227	2.945	4.500	0.489	0.024	0.472	0.398	0.888	0.055	0.830	1.495
Complex	No	Halton	1,000	16.6	-3.303	-0.227	2.945	4.500	0.477	0.028	0.403	0.424	0.851	0.050	0.782	1.427
Complex	No	Product Rule	9^2	3.8	-3.303	-0.227	2.945	4.500	0.439	0.023	0.343	0.578	0.868	0.051	0.804	1.732
Complex	Yes	Monte Carlo	100	11.2	-3.303	-0.227	2.945	4.500	0.160	-0.019	0.038	-0.553	0.834	0.059	0.714	1.120
Complex	Yes	Monte Carlo	1,000	52.5	-3.303	-0.227	2.945	4.500	0.062	-0.001	0.006	-0.045	0.588	0.038	0.521	0.828
Complex	Yes	Halton	1,000	52.6	-3.303	-0.227	2.945	4.500	0.092	0.004	0.045	0.203	0.547	0.034	0.483	0.874
Complex	Yes	Product Rule	9^2	11.3	-3.303	-0.227	2.945	4.500	0.073	0.001	0.028	0.089	0.525	0.031	0.472	0.908
RCNL	No	Monte Carlo	100	9.2	-5.892	-0.195	1.651	3.198	0.881	0.010	0.195	0.122	1.863	0.057	0.531	0.926
RCNL	No	Monte Carlo	1,000	52.0	-5.892	-0.195	1.651	3.198	0.836	0.018	0.219	0.323	1.494	0.043	0.424	0.763
RCNL	No	Halton	1,000	55.6	-5.892	-0.195	1.651	3.198	0.803	0.023	0.174	0.262	1.478	0.044	0.430	0.767
RCNL	No	Product Rule	9^1	7.6	-5.892	-0.195	1.651	3.198	0.781	0.021	0.216	0.433	1.409	0.041	0.407	0.755
RCNL	Yes	Monte Carlo	100	30.8	-5.892	-0.195	1.651	3.198	0.049	-0.009	0.011	-0.102	0.789	0.020	0.193	0.328
RCNL	Yes	Monte Carlo	1,000	154.2	-5.892	-0.195	1.651	3.198	0.112	0.001	0.028	0.030	0.640	0.019	0.176	0.330
RCNL	Yes	Halton	1,000	160.7	-5.892	-0.195	1.651	3.198	0.093	0.003	0.022	-0.002	0.695	0.021	0.190	0.340
RCNL	Yes	Product Rule	9^1	20.5	-5.892	-0.195	1.651	3.198	0.037	0.001	0.019	0.069	0.636	0.019	0.174	0.342

This table reports bias and variance of post-estimation outputs along with estimation speed over 1,000 simulated datasets for different numerical integration methods and numbers of draws I_t. It is created from the same results as Table 7.

Table E.3: Alternative Instruments: Post-Estimation

					True Median Value				Median Bias				Median Absolute Error			
Simulation	Supply	Z_D	Z_S	Seconds	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS
Simple	No	$[X_1, w]$	X_3	0.1	-3.564	-0.241	2.651	3.653	-3.073	-0.212	-1.344	-1.936	3.283	0.244	1.354	1.975
Simple	No	$[\text{BLP}, w, X_1]$	X_3	0.8	-3.564	-0.241	2.651	3.653	0.401	0.026	0.293	0.519	0.804	0.057	0.649	0.972
Simple	No	$[\text{Local}, w, X_1]$	X_3	0.4	-3.564	-0.241	2.651	3.653	0.285	0.016	0.198	0.314	0.860	0.054	0.656	0.872
Simple	No	$[\text{Quadratic}, w, X_1]$	X_3	0.5	-3.564	-0.241	2.651	3.653	0.319	0.021	0.257	0.437	0.885	0.056	0.670	0.924
Simple	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	1.1	-3.564	-0.241	2.651	3.653	0.558	0.032	0.463	0.610	0.874	0.055	0.687	0.925
Simple	Yes	$[X_1, w]$	X_3	0.8	-3.564	-0.241	2.651	3.653	-0.772	-0.066	-0.487	-0.701	1.206	0.091	0.772	1.199
Simple	Yes	$[\text{BLP}, w, X_1]$	X_3	2.2	-3.564	-0.241	2.651	3.653	0.400	0.026	0.312	0.530	0.804	0.057	0.648	0.972
Simple	Yes	$[\text{Local}, w, X_1]$	X_3	1.1	-3.564	-0.241	2.651	3.653	0.278	0.016	0.208	0.337	0.857	0.054	0.655	0.867
Simple	Yes	$[\text{Quadratic}, w, X_1]$	X_3	1.2	-3.564	-0.241	2.651	3.653	0.319	0.021	0.264	0.443	0.887	0.056	0.669	0.924
Simple	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	4.3	-3.564	-0.241	2.651	3.653	0.041	0.001	0.038	0.079	0.608	0.038	0.446	0.621
Complex	No	$[X_1, w]$	X_3	0.2	-3.303	-0.227	2.945	4.500	-3.080	-0.199	-1.499	-2.508	3.296	0.227	1.521	2.543
Complex	No	$[\text{BLP}, w, X_1]$	X_3	2.8	-3.303	-0.227	2.945	4.500	0.374	0.012	0.377	0.421	0.819	0.054	0.779	1.508
Complex	No	$[\text{Local}, w, X_1]$	X_3	1.3	-3.303	-0.227	2.945	4.500	0.587	0.033	0.589	0.457	0.922	0.051	0.880	1.821
Complex	No	$[\text{Quadratic}, w, X_1]$	X_3	1.2	-3.303	-0.227	2.945	4.500	0.665	0.040	0.541	0.832	0.943	0.056	0.862	1.709
Complex	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	3.8	-3.303	-0.227	2.945	4.500	0.439	0.023	0.343	0.578	0.868	0.051	0.804	1.732
Complex	Yes	$[X_1, w]$	X_3	1.1	-3.303	-0.227	2.945	4.500	-0.800	-0.056	-0.565	-0.984	1.307	0.092	1.001	1.769
Complex	Yes	$[\text{BLP}, w, X_1]$	X_3	7.5	-3.303	-0.227	2.945	4.500	0.368	0.009	0.382	0.464	0.817	0.053	0.770	1.570
Complex	Yes	$[\text{Local}, w, X_1]$	X_3	3.9	-3.303	-0.227	2.945	4.500	0.584	0.032	0.588	0.518	0.922	0.051	0.866	1.833
Complex	Yes	$[\text{Quadratic}, w, X_1]$	X_3	3.5	-3.303	-0.227	2.945	4.500	0.669	0.040	0.545	0.892	0.944	0.056	0.856	1.715
Complex	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	11.3	-3.303	-0.227	2.945	4.500	0.073	0.001	0.028	0.089	0.525	0.031	0.472	0.908
RCNL	No	$[X_1, w]$	X_3	0.4	-5.892	-0.195	1.651	3.198	-5.831	-0.175	-0.876	-1.633	6.282	0.194	0.886	1.660
RCNL	No	$[\text{BLP}, w, X_1]$	X_3	5.6	-5.892	-0.195	1.651	3.198	0.549	0.001	0.140	0.190	1.428	0.049	0.399	0.851
RCNL	No	$[\text{Local}, w, X_1]$	X_3	3.0	-5.892	-0.195	1.651	3.198	0.551	0.019	0.184	0.460	1.341	0.041	0.394	0.742
RCNL	No	$[\text{Quadratic}, w, X_1]$	X_3	3.2	-5.892	-0.195	1.651	3.198	0.646	0.021	0.206	0.509	1.367	0.041	0.382	0.758
RCNL	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	7.6	-5.892	-0.195	1.651	3.198	0.781	0.021	0.216	0.433	1.409	0.041	0.407	0.755
RCNL	Yes	$[X_1, w]$	X_3	1.9	-5.892	-0.195	1.651	3.198	-1.720	-0.051	-0.332	-0.588	2.568	0.070	0.543	1.006
RCNL	Yes	$[\text{BLP}, w, X_1]$	X_3	11.3	-5.892	-0.195	1.651	3.198	0.546	0.001	0.140	0.192	1.424	0.049	0.401	0.845
RCNL	Yes	$[\text{Local}, w, X_1]$	X_3	6.0	-5.892	-0.195	1.651	3.198	0.554	0.019	0.191	0.479	1.334	0.040	0.393	0.742
RCNL	Yes	$[\text{Quadratic}, w, X_1]$	X_3	6.5	-5.892	-0.195	1.651	3.198	0.646	0.021	0.209	0.517	1.364	0.040	0.380	0.758
RCNL	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	20.5	-5.892	-0.195	1.651	3.198	0.037	0.001	0.019	0.069	0.636	0.019	0.174	0.342

This table documents bias and variance of post-estimation outputs along with estimation speed over 1,000 simulated datasets for different instruments. It is created from the same results as Table 8.

Table E.4: Form of Optimal Instruments: Post-Estimation

				True Median Value				Median Bias				Median Absolute Error			
Simulation	Supply	Optimality	Seconds	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS
Simple	No	Approximate	1.1	-3.564	-0.241	2.651	3.653	0.558	0.032	0.463	0.610	0.874	0.055	0.687	0.925
Simple	No	Asymptotic	4.2	-3.564	-0.241	2.651	3.653	0.557	0.032	0.462	0.611	0.877	0.055	0.689	0.936
Simple	No	Empirical	4.2	-3.564	-0.241	2.651	3.653	0.564	0.032	0.463	0.608	0.872	0.055	0.688	0.931
Simple	Yes	Approximate	4.3	-3.564	-0.241	2.651	3.653	0.041	0.001	0.038	0.079	0.608	0.038	0.446	0.621
Simple	Yes	Asymptotic	20.0	-3.564	-0.241	2.651	3.653	0.105	0.006	0.081	0.126	0.636	0.040	0.451	0.634
Simple	Yes	Empirical	20.1	-3.564	-0.241	2.651	3.653	0.045	-0.000	0.034	0.061	0.629	0.039	0.463	0.636
Complex	No	Approximate	3.8	-3.303	-0.227	2.945	4.500	0.439	0.023	0.343	0.578	0.868	0.051	0.804	1.732
Complex	No	Asymptotic	7.8	-3.303	-0.227	2.945	4.500	0.457	0.024	0.365	0.462	0.884	0.051	0.826	1.545
Complex	No	Empirical	7.8	-3.303	-0.227	2.945	4.500	0.464	0.024	0.342	0.550	0.891	0.052	0.832	1.577
Complex	Yes	Approximate	11.3	-3.303	-0.227	2.945	4.500	0.073	0.001	0.028	0.089	0.525	0.031	0.472	0.908
Complex	Yes	Asymptotic	37.9	-3.303	-0.227	2.945	4.500	0.083	0.001	-0.010	0.039	0.651	0.039	0.568	1.087
Complex	Yes	Empirical	37.4	-3.303	-0.227	2.945	4.500	0.059	0.002	0.009	0.031	0.606	0.038	0.553	0.984
RCNL	No	Approximate	7.6	-5.892	-0.195	1.651	3.198	0.781	0.021	0.216	0.433	1.409	0.041	0.407	0.755
RCNL	No	Asymptotic	12.1	-5.892	-0.195	1.651	3.198	0.763	0.021	0.217	0.444	1.403	0.041	0.408	0.749
RCNL	No	Empirical	12.1	-5.892	-0.195	1.651	3.198	0.736	0.020	0.211	0.439	1.383	0.040	0.405	0.736
RCNL	Yes	Approximate	20.5	-5.892	-0.195	1.651	3.198	0.037	0.001	0.019	0.069	0.636	0.019	0.174	0.342
RCNL	Yes	Asymptotic	59.6	-5.892	-0.195	1.651	3.198	0.104	0.001	0.035	0.083	0.677	0.020	0.180	0.355
RCNL	Yes	Empirical	58.9	-5.892	-0.195	1.651	3.198	0.094	0.001	0.024	0.075	0.688	0.020	0.177	0.344

This table documents bias and variance of post-estimation outputs along with estimation speed over 1,000 simulated datasets for different forms of optimal instruments. It is created from the same results as Table 9.

Table E.5: Varying Instrument Strength: Post-Estimation

					True Median Value				Median Bias				Median Absolute Error			
Simulation	γ_2	Corr(p, w)	Supply	Seconds	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS
Simple	0.1	0.052	No	1.1	-3.516	-0.243	2.720	3.763	0.934	0.060	0.785	1.115	1.228	0.082	1.169	1.588
Simple	0.1	0.052	Yes	4.5	-3.516	-0.243	2.720	3.763	0.032	0.002	0.026	0.100	0.725	0.047	0.550	0.759
Simple	0.2	0.102	No	1.1	-3.564	-0.241	2.651	3.653	0.558	0.032	0.463	0.610	0.874	0.055	0.687	0.925
Simple	0.2	0.102	Yes	4.3	-3.564	-0.241	2.651	3.653	0.041	0.001	0.038	0.079	0.608	0.038	0.446	0.621
Simple	0.4	0.199	No	1.1	-3.663	-0.237	2.516	3.454	0.184	0.010	0.133	0.210	0.458	0.029	0.315	0.450
Simple	0.4	0.199	Yes	4.2	-3.663	-0.237	2.516	3.454	0.025	-0.000	0.028	0.054	0.406	0.026	0.276	0.384
Simple	0.8	0.376	No	1.1	-3.860	-0.229	2.278	3.082	0.055	0.002	0.033	0.062	0.240	0.016	0.140	0.223
Simple	0.8	0.376	Yes	4.3	-3.860	-0.229	2.278	3.082	0.002	-0.001	0.004	0.024	0.228	0.015	0.136	0.208
Complex	0.1	0.054	No	3.7	-3.264	-0.228	3.016	4.617	0.746	0.042	0.610	0.536	1.224	0.075	1.196	2.451
Complex	0.1	0.054	Yes	11.5	-3.264	-0.228	3.016	4.617	0.060	0.001	0.013	0.238	0.591	0.037	0.548	1.073
Complex	0.2	0.104	No	3.8	-3.303	-0.227	2.945	4.500	0.439	0.023	0.343	0.578	0.868	0.051	0.804	1.732
Complex	0.2	0.104	Yes	11.3	-3.303	-0.227	2.945	4.500	0.073	0.001	0.028	0.089	0.525	0.031	0.472	0.908
Complex	0.4	0.204	No	3.7	-3.381	-0.223	2.818	4.281	0.176	0.006	0.116	0.300	0.494	0.028	0.397	0.866
Complex	0.4	0.204	Yes	11.3	-3.381	-0.223	2.818	4.281	0.040	-0.000	-0.007	0.091	0.410	0.025	0.333	0.723
Complex	0.8	0.384	No	3.7	-3.535	-0.215	2.580	3.877	0.056	0.000	0.014	0.177	0.257	0.015	0.187	0.505
Complex	0.8	0.384	Yes	12.5	-3.535	-0.215	2.580	3.877	0.019	-0.001	-0.019	0.088	0.238	0.015	0.174	0.463
RCNL	0.1	0.050	No	7.6	-5.798	-0.196	1.692	3.302	1.263	0.037	0.377	0.726	1.894	0.057	0.590	1.138
RCNL	0.1	0.050	Yes	20.3	-5.798	-0.196	1.692	3.302	0.057	0.001	0.019	0.069	0.680	0.021	0.198	0.363
RCNL	0.2	0.097	No	7.6	-5.892	-0.195	1.651	3.198	0.781	0.021	0.216	0.433	1.409	0.041	0.407	0.755
RCNL	0.2	0.097	Yes	20.5	-5.892	-0.195	1.651	3.198	0.037	0.001	0.019	0.069	0.636	0.019	0.174	0.342
RCNL	0.4	0.189	No	7.8	-6.079	-0.192	1.571	3.011	0.290	0.008	0.069	0.167	0.788	0.023	0.197	0.373
RCNL	0.4	0.189	Yes	19.8	-6.079	-0.192	1.571	3.011	0.047	0.001	0.014	0.060	0.526	0.016	0.130	0.256
RCNL	0.8	0.358	No	7.8	-6.457	-0.185	1.431	2.690	0.087	0.004	0.022	0.070	0.424	0.012	0.091	0.186
RCNL	0.8	0.358	Yes	19.7	-6.457	-0.185	1.431	2.690	0.014	0.001	0.010	0.044	0.366	0.010	0.081	0.152

This table documents bias and variance of post-estimation outputs along with estimation speed over 1,000 simulated datasets for varying instrument strength. It is created from the same results as Table 10.

Table E.6: Problem Scaling: Post-Estimation

					True Median Value				Median Bias				Median Absolute Error			
Simulation	Supply	T	J_f	Seconds	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS
Simple	No	20	$\{2, 5, 10\}$	1.1	-3.564	-0.241	2.651	3.653	0.558	0.032	0.463	0.610	0.874	0.055	0.687	0.925
Simple	No	40	$\{2, 5, 10\}$	2.2	-3.566	-0.242	5.330	7.402	0.325	0.017	0.503	0.683	0.594	0.037	0.876	1.233
Simple	No	100	$\{2, 5, 10\}$	5.8	-3.567	-0.242	13.356	18.591	0.141	0.006	0.542	0.831	0.366	0.023	1.396	1.874
Simple	No	20	$\{4, 10, 20\}$	1.3	-3.581	-0.315	3.727	5.920	0.312	0.022	0.341	0.501	0.585	0.049	0.624	0.962
Simple	No	20	$\{10, 25, 50\}$	2.0	-3.591	-0.407	5.474	10.180	0.131	0.021	0.203	0.360	0.419	0.049	0.642	1.215
Simple	Yes	20	$\{2, 5, 10\}$	4.3	-3.564	-0.241	2.651	3.653	0.041	0.001	0.038	0.079	0.608	0.038	0.446	0.621
Simple	Yes	40	$\{2, 5, 10\}$	8.5	-3.566	-0.242	5.330	7.402	0.053	0.001	0.074	0.159	0.436	0.027	0.641	0.844
Simple	Yes	100	$\{2, 5, 10\}$	22.0	-3.567	-0.242	13.356	18.591	0.017	-0.001	0.059	0.152	0.282	0.017	1.017	1.377
Simple	Yes	20	$\{4, 10, 20\}$	6.8	-3.581	-0.315	3.727	5.920	0.028	-0.002	0.024	0.035	0.535	0.045	0.536	0.840
Simple	Yes	20	$\{10, 25, 50\}$	45.9	-3.591	-0.407	5.474	10.180	0.028	0.006	0.042	-0.028	0.403	0.047	0.619	1.161
Complex	No	20	$\{2, 5, 10\}$	3.8	-3.303	-0.227	2.945	4.500	0.439	0.023	0.343	0.578	0.868	0.051	0.804	1.732
Complex	No	40	$\{2, 5, 10\}$	7.9	-3.304	-0.228	5.913	9.104	0.292	0.015	0.477	0.857	0.569	0.033	0.982	2.040
Complex	No	100	$\{2, 5, 10\}$	22.7	-3.304	-0.228	14.780	22.763	0.104	0.005	0.409	1.003	0.359	0.023	1.594	2.975
Complex	No	20	$\{4, 10, 20\}$	5.4	-3.338	-0.298	4.054	7.154	0.296	0.015	0.293	0.717	0.634	0.047	0.745	1.857
Complex	No	20	$\{10, 25, 50\}$	12.1	-3.377	-0.387	5.837	12.031	0.102	0.006	0.140	0.487	0.443	0.050	0.763	2.111
Complex	Yes	20	$\{2, 5, 10\}$	11.3	-3.303	-0.227	2.945	4.500	0.073	0.001	0.028	0.089	0.525	0.031	0.472	0.908
Complex	Yes	40	$\{2, 5, 10\}$	23.0	-3.304	-0.228	5.913	9.104	0.046	0.001	0.064	0.288	0.375	0.023	0.657	1.259
Complex	Yes	100	$\{2, 5, 10\}$	68.6	-3.304	-0.228	14.780	22.763	0.023	0.000	0.104	0.397	0.255	0.016	1.108	2.112
Complex	Yes	20	$\{4, 10, 20\}$	21.9	-3.338	-0.298	4.054	7.154	0.047	-0.002	-0.025	0.351	0.509	0.042	0.564	1.393
Complex	Yes	20	$\{10, 25, 50\}$	145.0	-3.377	-0.387	5.837	12.031	0.059	0.000	0.041	0.397	0.380	0.042	0.658	1.987
RCNL	No	20	$\{2, 5, 10\}$	7.6	-5.892	-0.195	1.651	3.198	0.781	0.021	0.216	0.433	1.409	0.041	0.407	0.755
RCNL	No	40	$\{2, 5, 10\}$	16.3	-5.889	-0.194	3.269	6.457	0.414	0.011	0.229	0.505	0.961	0.028	0.537	1.007
RCNL	No	100	$\{2, 5, 10\}$	50.8	-5.893	-0.194	8.200	16.167	0.163	0.003	0.211	0.552	0.611	0.018	0.825	1.533
RCNL	No	20	$\{4, 10, 20\}$	10.5	-6.034	-0.232	1.862	4.611	0.471	0.012	0.151	0.444	1.020	0.036	0.320	0.788
RCNL	No	20	$\{10, 25, 50\}$	19.0	-6.132	-0.273	2.141	6.646	0.249	0.004	0.086	0.239	0.750	0.033	0.260	0.820
RCNL	Yes	20	$\{2, 5, 10\}$	20.5	-5.892	-0.195	1.651	3.198	0.037	0.001	0.019	0.069	0.636	0.019	0.174	0.342
RCNL	Yes	40	$\{2, 5, 10\}$	46.3	-5.889	-0.194	3.269	6.457	0.029	-0.001	0.024	0.072	0.485	0.014	0.256	0.471
RCNL	Yes	100	$\{2, 5, 10\}$	129.2	-5.893	-0.194	8.200	16.167	-0.001	-0.002	0.017	0.119	0.302	0.009	0.398	0.766
RCNL	Yes	20	$\{4, 10, 20\}$	36.2	-6.034	-0.232	1.862	4.611	0.012	-0.004	0.001	0.033	0.712	0.026	0.217	0.511
RCNL	Yes	20	$\{10, 25, 50\}$	181.1	-6.132	-0.273	2.141	6.646	0.016	-0.004	0.009	0.036	0.644	0.029	0.228	0.702

This table documents bias and variance of post-estimation outputs along with estimation speed over 1,000 simulated datasets for different problem sizes. It is created from the same results as Table 12.

F. Monte Carlo Results for Merger Simulation

For each set of results used to create the tables in Section 5.2, we report the bias and variance of three merger simulation outputs.

After a merger of three out of the five firms, we compute post-merger prices and shares with the ζ-markup approach in Section 3.5. We then compute the post-merger change in total producer surplus, PS from (E.1), the change in total consumer surplus, CS from (E.2), and the change in the mean Herfindahl-Hirschman Index,

$$\text{HHI} = \frac{10{,}000}{T} \sum_{f,t} \Big(\sum_{j \in J_{ft}} s_{jt} \Big)^2 .$$

Table F.7: Alternative Integration Methods: Merger

					True Median Value			Median Bias			Median Absolute Error		
Simulation	Supply	Integration	I_t	Seconds	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS
Simple	No	Monte Carlo	100	1.5	1,112.227	0.052	-0.168	36.826	-0.014	0.018	37.165	0.020	0.050
Simple	No	Monte Carlo	1,000	6.2	1,112.227	0.052	-0.168	4.734	0.005	-0.020	9.784	0.012	0.041
Simple	No	Halton	1,000	5.7	1,112.227	0.052	-0.168	1.385	0.007	-0.023	8.421	0.013	0.040
Simple	No	Product Rule	9^1	1.1	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.634	0.012	0.041
Simple	Yes	Monte Carlo	100	5.9	1,112.227	0.052	-0.168	36.663	-0.015	0.027	36.702	0.016	0.038
Simple	Yes	Monte Carlo	1,000	38.1	1,112.227	0.052	-0.168	3.977	-0.002	0.001	9.311	0.009	0.029
Simple	Yes	Halton	1,000	36.4	1,112.227	0.052	-0.168	0.373	0.000	-0.002	7.696	0.010	0.029
Simple	Yes	Product Rule	9^1	4.3	1,112.227	0.052	-0.168	-0.216	-0.001	-0.001	7.878	0.009	0.029
Complex	No	Monte Carlo	100	3.1	1,102.034	0.062	-0.199	52.247	-0.023	0.040	52.850	0.028	0.072
Complex	No	Monte Carlo	1,000	15.4	1,102.034	0.062	-0.199	10.853	-0.005	0.002	17.464	0.019	0.059
Complex	No	Halton	1,000	16.6	1,102.034	0.062	-0.199	1.000	0.002	-0.010	16.690	0.016	0.054
Complex	No	Product Rule	9^2	3.8	1,102.034	0.062	-0.199	0.813	-0.002	0.006	20.219	0.018	0.063
Complex	Yes	Monte Carlo	100	11.2	1,102.034	0.062	-0.199	46.969	-0.022	0.043	48.850	0.024	0.054
Complex	Yes	Monte Carlo	1,000	52.5	1,102.034	0.062	-0.199	6.652	-0.004	0.006	13.531	0.011	0.033
Complex	Yes	Halton	1,000	52.6	1,102.034	0.062	-0.199	-2.627	0.001	-0.007	13.024	0.010	0.032
Complex	Yes	Product Rule	9^2	11.3	1,102.034	0.062	-0.199	-3.711	0.001	-0.005	13.588	0.009	0.033
RCNL	No	Monte Carlo	100	9.2	934.505	0.070	-0.160	21.094	-0.005	0.000	21.604	0.020	0.045
RCNL	No	Monte Carlo	1,000	52.0	934.505	0.070	-0.160	4.637	0.005	-0.013	8.057	0.016	0.036
RCNL	No	Halton	1,000	55.6	934.505	0.070	-0.160	2.118	0.006	-0.015	6.889	0.016	0.037
RCNL	No	Product Rule	9^1	7.6	934.505	0.070	-0.160	-0.354	0.008	-0.021	6.372	0.015	0.036
RCNL	Yes	Monte Carlo	100	30.8	934.505	0.070	-0.160	15.631	-0.006	0.008	16.420	0.009	0.017
RCNL	Yes	Monte Carlo	1,000	154.2	934.505	0.070	-0.160	1.941	-0.000	-0.001	5.406	0.007	0.017
RCNL	Yes	Halton	1,000	160.7	934.505	0.070	-0.160	0.812	0.000	-0.001	4.947	0.008	0.018
RCNL	Yes	Product Rule	9^1	20.5	934.505	0.070	-0.160	-1.395	0.001	-0.003	5.715	0.008	0.017

This table reports bias and variance of merger simulation estimates along with estimation speed over 1,000 simulated datasets for different numerical integration methods and numbers of draws I_t. It is created from the same results as Table 7.

Table F.8: Alternative Instruments: Merger

					True Median Value			Median Bias			Median Absolute Error		
Simulation	Supply	Z_D	Z_S	Seconds	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS
Simple	No	$[X_1, w]$	X_3	0.1	1,112.227	0.052	-0.168	37.222	-0.033	0.088	49.648	0.033	0.089
Simple	No	$[\text{BLP}, w, X_1]$	X_3	0.8	1,112.227	0.052	-0.168	0.000	0.005	-0.018	19.914	0.016	0.045
Simple	No	$[\text{Local}, w, X_1]$	X_3	0.4	1,112.227	0.052	-0.168	0.000	0.003	-0.011	13.479	0.013	0.040
Simple	No	$[\text{Quadratic}, w, X_1]$	X_3	0.5	1,112.227	0.052	-0.168	1.648	0.003	-0.014	17.455	0.014	0.042
Simple	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	1.1	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.634	0.012	0.041
Simple	Yes	$[X_1, w]$	X_3	0.8	1,112.227	0.052	-0.168	18.590	-0.018	0.044	38.260	0.026	0.067
Simple	Yes	$[\text{BLP}, w, X_1]$	X_3	2.2	1,112.227	0.052	-0.168	0.030	0.006	-0.019	19.914	0.016	0.046
Simple	Yes	$[\text{Local}, w, X_1]$	X_3	1.1	1,112.227	0.052	-0.168	0.000	0.004	-0.012	13.423	0.013	0.039
Simple	Yes	$[\text{Quadratic}, w, X_1]$	X_3	1.2	1,112.227	0.052	-0.168	1.648	0.003	-0.015	17.455	0.014	0.042
Simple	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	4.3	1,112.227	0.052	-0.168	-0.216	-0.001	-0.001	7.878	0.009	0.029
Complex	No	$[X_1, w]$	X_3	0.2	1,102.034	0.062	-0.199	35.079	-0.036	0.108	48.896	0.039	0.109
Complex	No	$[\text{BLP}, w, X_1]$	X_3	2.8	1,102.034	0.062	-0.199	19.142	-0.014	0.018	32.075	0.029	0.066
Complex	No	$[\text{Local}, w, X_1]$	X_3	1.3	1,102.034	0.062	-0.199	17.359	-0.007	0.016	29.975	0.027	0.092
Complex	No	$[\text{Quadratic}, w, X_1]$	X_3	1.2	1,102.034	0.062	-0.199	22.604	-0.005	0.005	38.908	0.027	0.082
Complex	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	3.8	1,102.034	0.062	-0.199	0.813	-0.002	0.006	20.219	0.018	0.063
Complex	Yes	$[X_1, w]$	X_3	1.1	1,102.034	0.062	-0.199	17.323	-0.020	0.048	37.699	0.030	0.085
Complex	Yes	$[\text{BLP}, w, X_1]$	X_3	7.5	1,102.034	0.062	-0.199	19.476	-0.014	0.020	32.232	0.029	0.066
Complex	Yes	$[\text{Local}, w, X_1]$	X_3	3.9	1,102.034	0.062	-0.199	17.536	-0.007	0.016	30.587	0.027	0.092
Complex	Yes	$[\text{Quadratic}, w, X_1]$	X_3	3.5	1,102.034	0.062	-0.199	24.256	-0.005	0.005	40.342	0.027	0.086
Complex	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	11.3	1,102.034	0.062	-0.199	-3.711	0.001	-0.005	13.588	0.009	0.033
RCNL	No	$[X_1, w]$	X_3	0.4	934.505	0.070	-0.160	-0.284	-0.036	0.082	39.303	0.037	0.084
RCNL	No	$[\text{BLP}, w, X_1]$	X_3	5.6	934.505	0.070	-0.160	15.128	-0.004	-0.002	30.887	0.021	0.040
RCNL	No	$[\text{Local}, w, X_1]$	X_3	3.0	934.505	0.070	-0.160	-0.763	0.009	-0.021	7.541	0.015	0.035
RCNL	No	$[\text{Quadratic}, w, X_1]$	X_3	3.2	934.505	0.070	-0.160	-0.519	0.010	-0.023	7.577	0.016	0.037
RCNL	No	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	7.6	934.505	0.070	-0.160	-0.354	0.008	-0.021	6.372	0.015	0.036
RCNL	Yes	$[X_1, w]$	X_3	1.9	934.505	0.070	-0.160	-0.111	-0.015	0.030	30.816	0.026	0.052
RCNL	Yes	$[\text{BLP}, w, X_1]$	X_3	11.3	934.505	0.070	-0.160	14.800	-0.004	-0.003	30.397	0.021	0.040
RCNL	Yes	$[\text{Local}, w, X_1]$	X_3	6.0	934.505	0.070	-0.160	-0.869	0.010	-0.022	7.552	0.015	0.035
RCNL	Yes	$[\text{Quadratic}, w, X_1]$	X_3	6.5	934.505	0.070	-0.160	-0.520	0.010	-0.024	7.598	0.016	0.037
RCNL	Yes	$[\text{Optimal}, X_1]$	$[\text{Optimal}, X_3]$	20.5	934.505	0.070	-0.160	-1.395	0.001	-0.003	5.715	0.008	0.017

This table documents bias and variance of merger simulation estimates along with estimation speed over 1,000 simulated datasets for different instruments. It is created from the same results as Table 8.

Table F.9: Form of Optimal Instruments: Merger

				True Median Value			Median Bias			Median Absolute Error		
Simulation	Supply	Optimality	Seconds	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS
Simple	No	Approximate	1.1	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.634	0.012	0.041
Simple	No	Asymptotic	4.2	1,112.227	0.052	-0.168	0.806	0.007	-0.026	8.466	0.012	0.041
Simple	No	Empirical	4.2	1,112.227	0.052	-0.168	0.752	0.007	-0.026	8.538	0.012	0.041
Simple	Yes	Approximate	4.3	1,112.227	0.052	-0.168	-0.216	-0.001	-0.001	7.878	0.009	0.029
Simple	Yes	Asymptotic	20.0	1,112.227	0.052	-0.168	-0.107	-0.000	-0.005	7.797	0.009	0.030
Simple	Yes	Empirical	20.1	1,112.227	0.052	-0.168	-0.230	-0.001	-0.002	7.983	0.009	0.029
Complex	No	Approximate	3.8	1,102.034	0.062	-0.199	0.813	-0.002	0.006	20.219	0.018	0.063
Complex	No	Asymptotic	7.8	1,102.034	0.062	-0.199	0.822	-0.002	0.008	20.424	0.018	0.063
Complex	No	Empirical	7.8	1,102.034	0.062	-0.199	0.199	-0.002	0.005	21.841	0.017	0.062
Complex	Yes	Approximate	11.3	1,102.034	0.062	-0.199	-3.711	0.001	-0.005	13.588	0.009	0.033
Complex	Yes	Asymptotic	37.9	1,102.034	0.062	-0.199	0.000	-0.001	0.001	14.756	0.011	0.041
Complex	Yes	Empirical	37.4	1,102.034	0.062	-0.199	-0.697	-0.001	0.001	14.349	0.012	0.038
RCNL	No	Approximate	7.6	934.505	0.070	-0.160	-0.354	0.008	-0.021	6.372	0.015	0.036
RCNL	No	Asymptotic	12.1	934.505	0.070	-0.160	-0.290	0.008	-0.020	6.326	0.015	0.035
RCNL	No	Empirical	12.1	934.505	0.070	-0.160	-0.558	0.009	-0.021	6.248	0.015	0.035
RCNL	Yes	Approximate	20.5	934.505	0.070	-0.160	-1.395	0.001	-0.003	5.715	0.008	0.017
RCNL	Yes	Asymptotic	59.6	934.505	0.070	-0.160	-1.635	0.001	-0.005	5.778	0.008	0.018
RCNL	Yes	Empirical	58.9	934.505	0.070	-0.160	-1.618	0.001	-0.005	5.724	0.008	0.018

This table documents bias and variance of merger simulation estimates along with estimation speed over 1,000 simulated datasets for different forms of optimal instruments. It is created from the same results as Table 9.

Table F.10: Varying Instrument Strength: Merger

					True Median Value			Median Bias			Median Absolute Error		
Simulation	γ_2	Corr(p, w)	Supply	Seconds	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS
Simple	0.1	0.052	No	1.1	1,114.327	0.054	-0.174	2.613	0.012	-0.045	8.990	0.020	0.067
Simple	0.1	0.052	Yes	4.5	1,114.327	0.054	-0.174	0.000	-0.001	-0.003	7.945	0.011	0.035
Simple	0.2	0.102	No	1.1	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.634	0.012	0.041
Simple	0.2	0.102	Yes	4.3	1,112.227	0.052	-0.168	-0.216	-0.001	-0.001	7.878	0.009	0.029
Simple	0.4	0.199	No	1.1	1,113.231	0.047	-0.158	-0.762	0.001	-0.009	8.251	0.007	0.020
Simple	0.4	0.199	Yes	4.2	1,113.231	0.047	-0.158	-0.787	-0.000	-0.001	7.776	0.006	0.018
Simple	0.8	0.376	No	1.1	1,115.817	0.040	-0.137	-1.635	0.001	-0.004	8.523	0.005	0.012
Simple	0.8	0.376	Yes	4.3	1,115.817	0.040	-0.137	-1.655	0.000	-0.002	7.615	0.004	0.011
Complex	0.1	0.054	No	3.7	1,104.680	0.064	-0.205	5.491	-0.006	0.020	28.757	0.028	0.103
Complex	0.1	0.054	Yes	11.5	1,104.680	0.064	-0.205	-3.654	-0.000	-0.006	13.097	0.011	0.038
Complex	0.2	0.104	No	3.8	1,102.034	0.062	-0.199	0.813	-0.002	0.006	20.219	0.018	0.063
Complex	0.2	0.104	Yes	11.3	1,102.034	0.062	-0.199	-3.711	0.001	-0.005	13.588	0.009	0.033
Complex	0.4	0.204	No	3.7	1,105.469	0.057	-0.187	-0.562	-0.001	0.001	16.677	0.010	0.033
Complex	0.4	0.204	Yes	11.3	1,105.469	0.057	-0.187	-4.688	0.001	-0.005	12.924	0.008	0.024
Complex	0.8	0.384	No	3.7	1,107.039	0.049	-0.167	-1.352	-0.001	-0.001	15.533	0.006	0.017
Complex	0.8	0.384	Yes	12.5	1,107.039	0.049	-0.167	-4.205	0.000	-0.003	12.512	0.006	0.015
RCNL	0.1	0.050	No	7.6	938.357	0.073	-0.165	0.641	0.013	-0.033	6.701	0.024	0.057
RCNL	0.1	0.050	Yes	20.3	938.357	0.073	-0.165	-1.146	0.001	-0.004	5.659	0.008	0.019
RCNL	0.2	0.097	No	7.6	934.505	0.070	-0.160	-0.354	0.008	-0.021	6.372	0.015	0.036
RCNL	0.2	0.097	Yes	20.5	934.505	0.070	-0.160	-1.395	0.001	-0.003	5.715	0.008	0.017
RCNL	0.4	0.189	No	7.8	937.049	0.065	-0.150	-1.507	0.003	-0.009	6.063	0.008	0.018
RCNL	0.4	0.189	Yes	19.8	937.049	0.065	-0.150	-1.948	0.001	-0.003	5.883	0.006	0.013
RCNL	0.8	0.358	No	7.8	937.049	0.056	-0.134	-2.802	0.002	-0.004	6.406	0.004	0.009
RCNL	0.8	0.358	Yes	19.7	937.049	0.056	-0.134	-2.526	0.001	-0.003	5.983	0.004	0.008

This table documents bias and variance of merger simulation estimates along with estimation speed over 1,000 simulated datasets for varying instrument strength. It is created from the same results as Table 10.

Table F.11: Problem Scaling: Merger

					True Median Value			Median Bias			Median Absolute Error		
Simulation	Supply	T	J_f	Seconds	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS
Simple	No	20	$\{2, 5, 10\}$	1.1	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.634	0.012	0.041
Simple	No	40	$\{2, 5, 10\}$	2.2	1,113.283	0.102	-0.342	0.896	0.006	-0.028	6.143	0.016	0.056
Simple	No	100	$\{2, 5, 10\}$	5.8	1,070.906	0.256	-0.845	0.132	0.002	-0.035	4.029	0.025	0.086
Simple	No	20	$\{4, 10, 20\}$	1.3	383.076	0.038	-0.090	0.696	0.002	-0.005	1.789	0.006	0.014
Simple	No	20	$\{10, 25, 50\}$	2.0	68.137	0.013	-0.024	-0.010	0.001	-0.001	0.117	0.002	0.003
Simple	Yes	20	$\{2, 5, 10\}$	4.3	1,112.227	0.052	-0.168	-0.216	-0.001	-0.001	7.878	0.009	0.029
Simple	Yes	40	$\{2, 5, 10\}$	8.5	1,113.283	0.102	-0.342	0.116	-0.001	-0.005	5.902	0.012	0.039
Simple	Yes	100	$\{2, 5, 10\}$	22.0	1,070.906	0.256	-0.845	-0.072	-0.006	-0.005	3.634	0.020	0.063
Simple	Yes	20	$\{4, 10, 20\}$	6.8	383.076	0.038	-0.090	0.495	-0.001	0.000	1.684	0.005	0.012
Simple	Yes	20	$\{10, 25, 50\}$	45.9	68.137	0.013	-0.024	-0.004	0.000	-0.000	0.106	0.002	0.003
Complex	No	20	$\{2, 5, 10\}$	3.8	1,102.034	0.062	-0.199	0.813	-0.002	0.006	20.219	0.018	0.063
Complex	No	40	$\{2, 5, 10\}$	7.9	1,102.163	0.122	-0.404	0.000	0.003	-0.023	11.920	0.022	0.074
Complex	No	100	$\{2, 5, 10\}$	22.7	1,060.386	0.304	-1.002	-0.677	0.004	-0.026	6.395	0.031	0.111
Complex	No	20	$\{4, 10, 20\}$	5.4	379.886	0.044	-0.102	0.205	-0.000	-0.001	5.124	0.008	0.022
Complex	No	20	$\{10, 25, 50\}$	12.1	67.645	0.015	-0.027	0.014	0.000	-0.000	0.406	0.002	0.004
Complex	Yes	20	$\{2, 5, 10\}$	11.3	1,102.034	0.062	-0.199	-3.711	0.001	-0.005	13.588	0.009	0.033
Complex	Yes	40	$\{2, 5, 10\}$	23.0	1,102.163	0.122	-0.404	-1.165	0.001	-0.009	9.697	0.013	0.043
Complex	Yes	100	$\{2, 5, 10\}$	68.6	1,060.386	0.304	-1.002	-0.374	-0.002	-0.007	5.878	0.022	0.072
Complex	Yes	20	$\{4, 10, 20\}$	21.9	379.886	0.044	-0.102	-0.902	0.000	-0.002	4.107	0.006	0.015
Complex	Yes	20	$\{10, 25, 50\}$	145.0	67.645	0.015	-0.027	-0.037	0.000	-0.000	0.318	0.002	0.003
RCNL	No	20	$\{2, 5, 10\}$	7.6	934.505	0.070	-0.160	-0.354	0.008	-0.021	6.372	0.015	0.036
RCNL	No	40	$\{2, 5, 10\}$	16.3	938.052	0.141	-0.324	-1.266	0.011	-0.027	4.785	0.021	0.048
RCNL	No	100	$\{2, 5, 10\}$	50.8	909.372	0.355	-0.813	-1.995	0.011	-0.033	3.180	0.033	0.076
RCNL	No	20	$\{4, 10, 20\}$	10.5	325.245	0.035	-0.063	0.480	0.002	-0.004	1.538	0.005	0.010
RCNL	No	20	$\{10, 25, 50\}$	19.0	58.506	0.008	-0.011	0.080	0.000	-0.000	0.170	0.001	0.001
RCNL	Yes	20	$\{2, 5, 10\}$	20.5	934.505	0.070	-0.160	-1.395	0.001	-0.003	5.715	0.008	0.017
RCNL	Yes	40	$\{2, 5, 10\}$	46.3	938.052	0.141	-0.324	-2.032	0.000	-0.004	4.098	0.010	0.024
RCNL	Yes	100	$\{2, 5, 10\}$	129.2	909.372	0.355	-0.813	-2.266	0.000	-0.009	2.974	0.017	0.041
RCNL	Yes	20	$\{4, 10, 20\}$	36.2	325.245	0.035	-0.063	0.372	-0.000	0.000	1.373	0.004	0.007
RCNL	Yes	20	$\{10, 25, 50\}$	181.1	58.506	0.008	-0.011	0.076	-0.000	0.000	0.151	0.001	0.001

This table documents bias and variance of merger simulation estimates along with estimation speed over 1,000 simulated datasets for different problem sizes. It is created from the same results as Table 12.

G. Additional Optimization Algorithm Results

Table 11 documents that the vast majority of run converge to a local optima, regardless of optimization routine. For completeness' sake, in this appendix we replicate the table on BLP instruments instead of optimal instruments (Table G.12), and document the impact of algorithm choice on bias and variance of parameter estimates, post-estimation outputs, and merger simulation estimates. Results are very similar across algorithms.

Table G.12: Optimization Algorithms: BLP Instruments

							Percent of Runs		Median, First GMM Step			
Simulation	Supply	P	Software	Algorithm	Gradient	Termination	Converged	PSD Hessian	Seconds	Evaluations	$q = \bar{g}'W\bar{g}$	$\lvert\lvert\nabla q\rvert\rvert_\infty$
Simple	No	1	Knitro	Interior/Direct	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.8%	0.4	6	2.61E-01	5.28E-06
Simple	No	1	Knitro	Active Set	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.9%	0.3	6	2.61E-01	5.83E-06
Simple	No	1	Knitro	SQP	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.7%	0.4	6	2.61E-01	5.39E-06
Simple	No	1	SciPy	L-BFGS-B	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.7%	0.4	6	2.61E-01	5.80E-06
Simple	No	1	SciPy	BFGS	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.9%	0.4	7	2.61E-01	6.58E-06
Simple	No	1	SciPy	TNC	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.9%	0.6	9	2.61E-01	1.20E-06
Simple	No	1	SciPy	TNC	Yes	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	99.6%	99.6%	0.7	11	2.61E-01	1.78E-09
Simple	No	1	SciPy	Nelder-Mead	No	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	78.0%	99.9%	17.9	129	2.61E-01	2.01E-08
Simple	Yes	2	Knitro	Interior/Direct	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	1.3	11	1.25E+00	2.15E-05
Simple	Yes	2	Knitro	Active Set	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	1.2	10	1.25E+00	1.95E-05
Simple	Yes	2	Knitro	SQP	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	1.3	10	1.25E+00	1.95E-05
Simple	Yes	2	SciPy	L-BFGS-B	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	1.2	9	1.25E+00	2.05E-05
Simple	Yes	2	SciPy	BFGS	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	1.3	10	1.25E+00	2.00E-05
Simple	Yes	2	SciPy	TNC	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	2.2	16	1.25E+00	1.25E-05
Simple	Yes	2	SciPy	TNC	Yes	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	100.0%	100.0%	2.1	15	1.25E+00	2.20E-05
Simple	Yes	2	SciPy	Nelder-Mead	No	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	52.2%	100.0%	49.9	268	1.25E+00	7.44E-08
Complex	No	3	Knitro	Interior/Direct	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	98.8%	2.4	22	2.17E-01	2.15E-05
Complex	No	3	Knitro	Active Set	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.3%	2.0	19	2.20E-01	2.25E-05
Complex	No	3	Knitro	SQP	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.4%	2.3	21	2.18E-01	2.04E-05
Complex	No	3	SciPy	L-BFGS-B	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.8%	2.1	21	2.18E-01	1.98E-05
Complex	No	3	SciPy	BFGS	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.4%	2.1	21	2.17E-01	1.98E-05
Complex	No	3	SciPy	TNC	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.7%	4.0	35	2.17E-01	1.31E-05
Complex	No	3	SciPy	TNC	Yes	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	100.0%	99.2%	3.6	34	2.18E-01	2.76E-05
Complex	No	3	SciPy	Nelder-Mead	No	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	71.6%	99.3%	40.7	321	2.17E-01	3.55E-08
Complex	Yes	4	Knitro	Interior/Direct	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.9%	5.7	24	1.07E+00	2.67E-05
Complex	Yes	4	Knitro	Active Set	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.9%	4.0	21	1.07E+00	2.74E-05
Complex	Yes	4	Knitro	SQP	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.9%	5.4	24	1.07E+00	2.74E-05
Complex	Yes	4	SciPy	L-BFGS-B	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	99.9%	4.4	22	1.07E+00	2.78E-05
Complex	Yes	4	SciPy	BFGS	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	3.9	20	1.07E+00	3.10E-05
Complex	Yes	4	SciPy	TNC	Yes	$\lvert\lvert\nabla q\rvert\rvert_\infty$	100.0%	100.0%	9.4	44	1.07E+00	2.44E-05
Complex	Yes	4	SciPy	TNC	Yes	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	100.0%	99.6%	6.6	32	1.07E+00	5.32E-04
Complex	Yes	4	SciPy	Nelder-Mead	No	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	46.9%	100.0%	80.2	1,001	1.07E+00	1.87E-07

Continued on the next page.

Continued from the previous page.

							Percent of Runs		Median, First GMM Step			
Simulation	Supply	P	Software	Algorithm	Gradient	Termination	Converged	PSD Hessian	Seconds	Evaluations	$q = \bar{g}'W\bar{g}$	$\|\|\nabla q\|\|_\infty$
RCNL	No	2	Knitro	Interior/Direct	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	97.1%	3.0	15	1.20E-01	4.89E-05
RCNL	No	2	Knitro	Active Set	Yes	$\|\|\nabla q\|\|_\infty$	99.9%	95.9%	4.6	15	1.20E-01	1.75E-05
RCNL	No	2	Knitro	SQP	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	96.3%	4.2	15	1.21E-01	1.53E-05
RCNL	No	2	SciPy	L-BFGS-B	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	95.8%	3.5	17	1.21E-01	1.42E-05
RCNL	No	2	SciPy	BFGS	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	97.2%	2.9	16	1.20E-01	1.49E-05
RCNL	No	2	SciPy	TNC	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	97.6%	5.8	30	1.21E-01	1.29E-05
RCNL	No	2	SciPy	TNC	Yes	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	100.0%	96.5%	4.8	25	1.21E-01	1.16E-04
RCNL	No	2	SciPy	Nelder-Mead	No	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	63.1%	96.9%	83.5	292	1.21E-01	1.23E-07
RCNL	Yes	3	Knitro	Interior/Direct	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.8%	6.9	21	1.09E+00	5.89E-05
RCNL	Yes	3	Knitro	Active Set	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.8%	6.7	20	1.09E+00	2.92E-05
RCNL	Yes	3	Knitro	SQP	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.8%	7.3	22	1.09E+00	2.66E-05
RCNL	Yes	3	SciPy	L-BFGS-B	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.9%	6.2	20	1.09E+00	2.27E-05
RCNL	Yes	3	SciPy	BFGS	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	100.0%	5.4	17	1.09E+00	2.63E-05
RCNL	Yes	3	SciPy	TNC	Yes	$\|\|\nabla q\|\|_\infty$	100.0%	99.8%	12.2	37	1.09E+00	2.46E-05
RCNL	Yes	3	SciPy	TNC	Yes	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	100.0%	99.8%	9.1	29	1.09E+00	6.01E-04
RCNL	Yes	3	SciPy	Nelder-Mead	No	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	44.0%	99.8%	140.4	1,001	1.09E+00	3.05E-07

Like Table 11, this table also documents optimization convergence statistics along with estimation speed over 1,000 simulated datasets for different optimization algorithms. Instead of optimal instruments, the problems solved for this table use only the traditional BLP instruments.

Table G.13: Optimization Algorithms: Parameter Estimates

						True Value				Median Bias				Median Absolute Error			
Simulation	Supply	Software	Algorithm	Termination	Seconds	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ	α	σ_x	σ_p	ρ
Simple	No	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	1.2	-1	3			0.157	-0.037			0.246	0.193		
Simple	No	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	1.1	-1	3			0.157	-0.028			0.246	0.181		
Simple	No	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	1.2	-1	3			0.157	-0.037			0.246	0.193		
Simple	No	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	1.1	-1	3			0.157	-0.028			0.246	0.181		
Simple	No	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	1.2	-1	3			0.157	-0.028			0.246	0.181		
Simple	No	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	1.6	-1	3			0.157	-0.028			0.246	0.181		
Simple	No	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	1.8	-1	3			0.157	-0.028			0.246	0.181		
Simple	No	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	51.0	-1	3			0.157	-0.028			0.246	0.181		
Simple	Yes	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	4.5	-1	3			0.018	0.005			0.160	0.188		
Simple	Yes	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	4.1	-1	3			0.018	0.006			0.160	0.185		
Simple	Yes	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	4.3	-1	3			0.018	0.002			0.160	0.190		
Simple	Yes	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	3.8	-1	3			0.018	0.006			0.160	0.185		
Simple	Yes	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	4.0	-1	3			0.018	0.006			0.160	0.185		
Simple	Yes	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	6.0	-1	3			0.018	0.006			0.160	0.185		
Simple	Yes	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	5.6	-1	3			0.018	0.006			0.160	0.185		
Simple	Yes	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	99.8	-1	3			0.018	0.006			0.160	0.185		
Complex	No	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	5.0	-1	3	0.2		0.073	-0.093	-0.200		0.236	0.212	0.200	
Complex	No	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	4.0	-1	3	0.2		0.077	-0.080	-0.200		0.235	0.201	0.200	
Complex	No	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	4.7	-1	3	0.2		0.074	-0.086	-0.157		0.235	0.210	0.200	
Complex	No	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	4.1	-1	3	0.2		0.077	-0.079	-0.113		0.237	0.201	0.200	
Complex	No	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	4.1	-1	3	0.2		0.075	-0.079	-0.105		0.237	0.202	0.200	
Complex	No	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	6.7	-1	3	0.2		0.077	-0.078	0.018		0.235	0.202	0.159	
Complex	No	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	6.2	-1	3	0.2		0.074	-0.079	0.018		0.234	0.201	0.159	
Complex	No	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	91.1	-1	3	0.2		0.079	-0.079	0.018		0.238	0.202	0.157	
Complex	Yes	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	13.5	-1	3	0.2		-0.040	-0.051	-0.200		0.182	0.190	0.200	
Complex	Yes	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	10.3	-1	3	0.2		-0.039	-0.052	-0.077		0.181	0.190	0.200	
Complex	Yes	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	13.2	-1	3	0.2		-0.039	-0.051	-0.065		0.182	0.190	0.200	
Complex	Yes	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	10.4	-1	3	0.2		-0.036	-0.053	-0.123		0.182	0.190	0.200	
Complex	Yes	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	10.0	-1	3	0.2		-0.039	-0.051	-0.093		0.181	0.191	0.200	
Complex	Yes	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	18.8	-1	3	0.2		-0.039	-0.052	0.017		0.182	0.191	0.148	
Complex	Yes	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	14.3	-1	3	0.2		-0.038	-0.050	0.012		0.180	0.190	0.155	
Complex	Yes	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	162.0	-1	3	0.2		-0.039	-0.049	0.019		0.181	0.191	0.135	

Continued on the next page.

Continued from the previous page.

Simulation	Supply	Software	Algorithm	Termination	Seconds	True Value α	σ_x	σ_p	ρ	Median Bias α	σ_x	σ_p	ρ	Median Absolute Error α	σ_x	σ_p	ρ
RCNL	No	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	8.8	-1	3		0.5	0.100	-0.013		-0.006	0.227	0.203		0.025
RCNL	No	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	8.3	-1	3		0.5	0.111	-0.013		-0.004	0.222	0.192		0.024
RCNL	No	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	9.3	-1	3		0.5	0.106	-0.006		-0.006	0.226	0.194		0.025
RCNL	No	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	7.1	-1	3		0.5	0.113	-0.012		-0.005	0.221	0.189		0.024
RCNL	No	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	6.8	-1	3		0.5	0.111	-0.013		-0.005	0.222	0.191		0.024
RCNL	No	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	11.9	-1	3		0.5	0.113	-0.013		-0.005	0.221	0.188		0.024
RCNL	No	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	10.0	-1	3		0.5	0.109	-0.010		-0.005	0.221	0.188		0.024
RCNL	No	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	199.1	-1	3		0.5	0.111	-0.012		-0.005	0.224	0.193		0.025
RCNL	Yes	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	20.4	-1	3		0.5	0.011	-0.016		0.000	0.111	0.180		0.021
RCNL	Yes	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	18.0	-1	3		0.5	0.011	-0.017		0.000	0.109	0.180		0.021
RCNL	Yes	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	20.6	-1	3		0.5	0.011	-0.013		0.000	0.109	0.179		0.021
RCNL	Yes	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	14.9	-1	3		0.5	0.011	-0.016		0.000	0.109	0.180		0.021
RCNL	Yes	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	14.2	-1	3		0.5	0.011	-0.018		0.000	0.109	0.180		0.021
RCNL	Yes	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	25.9	-1	3		0.5	0.011	-0.018		0.000	0.109	0.179		0.021
RCNL	Yes	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	19.9	-1	3		0.5	0.011	-0.018		0.000	0.109	0.180		0.021
RCNL	Yes	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	283.4	-1	3		0.5	0.010	-0.019		0.001	0.109	0.180		0.021

This table documents bias and variance of parameter estimates along with estimation speed over 1,000 simulated datasets for different optimization algorithms. It is created from the same results as Table 11 after a second GMM step.

Table G.14: Optimization Algorithms: Post-Estimation

						True Median Value				Median Bias				Median Absolute Error			
Simulation	Supply	Software	Algorithm	Termination	Seconds	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS
Simple	No	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	1.2	-3.564	-0.241	2.651	3.653	0.559	0.032	0.464	0.613	0.875	0.055	0.689	0.925
Simple	No	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	1.1	-3.564	-0.241	2.651	3.653	0.559	0.032	0.464	0.613	0.875	0.055	0.689	0.925
Simple	No	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	1.2	-3.564	-0.241	2.651	3.653	0.559	0.032	0.464	0.613	0.875	0.055	0.689	0.925
Simple	No	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	1.1	-3.564	-0.241	2.651	3.653	0.559	0.032	0.464	0.613	0.875	0.055	0.689	0.925
Simple	No	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	1.2	-3.564	-0.241	2.651	3.653	0.559	0.032	0.464	0.613	0.875	0.055	0.689	0.925
Simple	No	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	1.6	-3.564	-0.241	2.651	3.653	0.559	0.032	0.464	0.613	0.875	0.055	0.689	0.925
Simple	No	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	1.8	-3.564	-0.241	2.651	3.653	0.559	0.032	0.464	0.613	0.875	0.055	0.689	0.925
Simple	No	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	51.0	-3.564	-0.241	2.651	3.653	0.559	0.032	0.464	0.613	0.875	0.055	0.689	0.925
Simple	Yes	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	4.5	-3.564	-0.241	2.651	3.653	0.063	0.001	0.042	0.071	0.571	0.037	0.416	0.567
Simple	Yes	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	4.1	-3.564	-0.241	2.651	3.653	0.063	0.001	0.042	0.069	0.571	0.037	0.417	0.567
Simple	Yes	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	4.3	-3.564	-0.241	2.651	3.653	0.063	0.001	0.042	0.069	0.571	0.037	0.416	0.567
Simple	Yes	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	3.8	-3.564	-0.241	2.651	3.653	0.063	0.001	0.042	0.069	0.571	0.037	0.417	0.567
Simple	Yes	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	4.0	-3.564	-0.241	2.651	3.653	0.063	0.001	0.042	0.069	0.571	0.037	0.417	0.567
Simple	Yes	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	6.0	-3.564	-0.241	2.651	3.653	0.063	0.001	0.042	0.069	0.571	0.037	0.417	0.567
Simple	Yes	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	5.6	-3.564	-0.241	2.651	3.653	0.063	0.001	0.042	0.069	0.571	0.037	0.417	0.567
Simple	Yes	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	99.8	-3.564	-0.241	2.651	3.653	0.063	0.001	0.042	0.069	0.571	0.037	0.417	0.567
Complex	No	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	5.0	-3.303	-0.227	2.945	4.500	0.479	0.027	0.440	0.605	0.830	0.049	0.779	1.374
Complex	No	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	4.0	-3.303	-0.227	2.945	4.500	0.498	0.028	0.443	0.621	0.832	0.050	0.784	1.371
Complex	No	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	4.7	-3.303	-0.227	2.945	4.500	0.482	0.027	0.441	0.603	0.828	0.049	0.779	1.371
Complex	No	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	4.1	-3.303	-0.227	2.945	4.500	0.496	0.028	0.443	0.610	0.831	0.050	0.782	1.371
Complex	No	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	4.1	-3.303	-0.227	2.945	4.500	0.496	0.027	0.442	0.617	0.831	0.049	0.785	1.374
Complex	No	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	6.7	-3.303	-0.227	2.945	4.500	0.496	0.028	0.445	0.605	0.828	0.049	0.784	1.374
Complex	No	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	6.2	-3.303	-0.227	2.945	4.500	0.489	0.027	0.443	0.603	0.828	0.049	0.777	1.361
Complex	No	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	91.1	-3.303	-0.227	2.945	4.500	0.496	0.028	0.445	0.610	0.829	0.049	0.788	1.374
Complex	Yes	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	13.5	-3.303	-0.227	2.945	4.500	0.089	0.002	0.044	0.117	0.485	0.029	0.428	0.865
Complex	Yes	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	10.3	-3.303	-0.227	2.945	4.500	0.089	0.002	0.043	0.118	0.482	0.029	0.428	0.865
Complex	Yes	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	13.2	-3.303	-0.227	2.945	4.500	0.093	0.002	0.041	0.119	0.480	0.029	0.426	0.875
Complex	Yes	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	10.4	-3.303	-0.227	2.945	4.500	0.096	0.003	0.049	0.119	0.485	0.029	0.431	0.857
Complex	Yes	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	10.0	-3.303	-0.227	2.945	4.500	0.089	0.002	0.041	0.117	0.480	0.029	0.425	0.857
Complex	Yes	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	18.8	-3.303	-0.227	2.945	4.500	0.091	0.003	0.045	0.113	0.486	0.029	0.435	0.865
Complex	Yes	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	14.3	-3.303	-0.227	2.945	4.500	0.091	0.003	0.049	0.110	0.485	0.029	0.431	0.867
Complex	Yes	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	162.0	-3.303	-0.227	2.945	4.500	0.091	0.003	0.047	0.113	0.480	0.029	0.428	0.865

Continued on the next page.

Continued from the previous page.

						True Median Value				Median Bias				Median Absolute Error			
Simulation	Supply	Software	Algorithm	Termination	Seconds	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS	$\bar{\varepsilon}_{jj}$	$\bar{E}$	PS	CS
RCNL	No	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	8.8	-5.892	-0.195	1.651	3.198	0.766	0.021	0.219	0.465	1.398	0.041	0.405	0.752
RCNL	No	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	8.3	-5.892	-0.195	1.651	3.198	0.776	0.021	0.221	0.441	1.373	0.041	0.403	0.737
RCNL	No	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	9.3	-5.892	-0.195	1.651	3.198	0.766	0.021	0.216	0.465	1.398	0.041	0.405	0.752
RCNL	No	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	7.1	-5.892	-0.195	1.651	3.198	0.777	0.021	0.222	0.440	1.371	0.040	0.402	0.738
RCNL	No	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	6.8	-5.892	-0.195	1.651	3.198	0.776	0.021	0.219	0.444	1.392	0.041	0.404	0.749
RCNL	No	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	11.9	-5.892	-0.195	1.651	3.198	0.779	0.021	0.222	0.441	1.373	0.040	0.403	0.739
RCNL	No	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	10.0	-5.892	-0.195	1.651	3.198	0.774	0.021	0.219	0.435	1.369	0.040	0.402	0.732
RCNL	No	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	199.1	-5.892	-0.195	1.651	3.198	0.776	0.021	0.219	0.441	1.371	0.040	0.402	0.738
RCNL	Yes	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	20.4	-5.892	-0.195	1.651	3.198	0.075	0.001	0.023	0.074	0.641	0.019	0.172	0.321
RCNL	Yes	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	18.0	-5.892	-0.195	1.651	3.198	0.069	0.001	0.022	0.072	0.640	0.019	0.171	0.318
RCNL	Yes	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	20.6	-5.892	-0.195	1.651	3.198	0.072	0.001	0.020	0.070	0.641	0.019	0.172	0.318
RCNL	Yes	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	14.9	-5.892	-0.195	1.651	3.198	0.069	0.001	0.019	0.071	0.639	0.019	0.171	0.318
RCNL	Yes	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	14.2	-5.892	-0.195	1.651	3.198	0.064	0.001	0.020	0.072	0.640	0.019	0.171	0.318
RCNL	Yes	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	25.9	-5.892	-0.195	1.651	3.198	0.063	0.001	0.019	0.071	0.640	0.019	0.171	0.319
RCNL	Yes	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	19.9	-5.892	-0.195	1.651	3.198	0.064	0.001	0.019	0.072	0.641	0.019	0.171	0.319
RCNL	Yes	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	283.4	-5.892	-0.195	1.651	3.198	0.062	0.001	0.019	0.069	0.640	0.019	0.171	0.319

This table documents bias and variance of post-estimation outputs along with estimation speed over 1,000 simulated datasets for different optimization algorithms. It is created from the same results as Table 11 after a second GMM step.

Table G.15: Optimization Algorithms: Merger

						True Median Value			Median Bias			Median Absolute Error		
Simulation	Supply	Software	Algorithm	Termination	Seconds	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS
Simple	No	Knitro	Interior/Direct	$\lvert\lvert\nabla q\rvert\rvert_\infty$	1.2	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.573	0.012	0.041
Simple	No	Knitro	Active Set	$\lvert\lvert\nabla q\rvert\rvert_\infty$	1.1	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.573	0.012	0.041
Simple	No	Knitro	SQP	$\lvert\lvert\nabla q\rvert\rvert_\infty$	1.2	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.573	0.012	0.041
Simple	No	SciPy	L-BFGS-B	$\lvert\lvert\nabla q\rvert\rvert_\infty$	1.1	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.573	0.012	0.041
Simple	No	SciPy	BFGS	$\lvert\lvert\nabla q\rvert\rvert_\infty$	1.2	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.573	0.012	0.041
Simple	No	SciPy	TNC	$\lvert\lvert\nabla q\rvert\rvert_\infty$	1.6	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.573	0.012	0.041
Simple	No	SciPy	TNC	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	1.8	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.573	0.012	0.041
Simple	No	SciPy	Nelder-Mead	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	51.0	1,112.227	0.052	-0.168	0.830	0.007	-0.026	8.573	0.012	0.041
Simple	Yes	Knitro	Interior/Direct	$\lvert\lvert\nabla q\rvert\rvert_\infty$	4.5	1,112.227	0.052	-0.168	-0.092	-0.000	-0.002	8.370	0.008	0.027
Simple	Yes	Knitro	Active Set	$\lvert\lvert\nabla q\rvert\rvert_\infty$	4.1	1,112.227	0.052	-0.168	-0.081	-0.000	-0.001	8.370	0.008	0.026
Simple	Yes	Knitro	SQP	$\lvert\lvert\nabla q\rvert\rvert_\infty$	4.3	1,112.227	0.052	-0.168	-0.081	-0.000	-0.001	8.407	0.008	0.027
Simple	Yes	SciPy	L-BFGS-B	$\lvert\lvert\nabla q\rvert\rvert_\infty$	3.8	1,112.227	0.052	-0.168	-0.082	-0.000	-0.001	8.370	0.008	0.026
Simple	Yes	SciPy	BFGS	$\lvert\lvert\nabla q\rvert\rvert_\infty$	4.0	1,112.227	0.052	-0.168	-0.081	-0.000	-0.001	8.370	0.008	0.026
Simple	Yes	SciPy	TNC	$\lvert\lvert\nabla q\rvert\rvert_\infty$	6.0	1,112.227	0.052	-0.168	-0.081	-0.000	-0.001	8.371	0.008	0.026
Simple	Yes	SciPy	TNC	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	5.6	1,112.227	0.052	-0.168	-0.082	-0.000	-0.001	8.371	0.008	0.026
Simple	Yes	SciPy	Nelder-Mead	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	99.8	1,112.227	0.052	-0.168	-0.081	-0.000	-0.001	8.370	0.008	0.026
Complex	No	Knitro	Interior/Direct	$\lvert\lvert\nabla q\rvert\rvert_\infty$	5.0	1,102.034	0.062	-0.199	0.851	0.002	-0.010	14.841	0.015	0.053
Complex	No	Knitro	Active Set	$\lvert\lvert\nabla q\rvert\rvert_\infty$	4.0	1,102.034	0.062	-0.199	0.851	0.002	-0.009	14.841	0.015	0.054
Complex	No	Knitro	SQP	$\lvert\lvert\nabla q\rvert\rvert_\infty$	4.7	1,102.034	0.062	-0.199	0.691	0.002	-0.010	14.961	0.015	0.053
Complex	No	SciPy	L-BFGS-B	$\lvert\lvert\nabla q\rvert\rvert_\infty$	4.1	1,102.034	0.062	-0.199	0.621	0.002	-0.010	14.788	0.015	0.053
Complex	No	SciPy	BFGS	$\lvert\lvert\nabla q\rvert\rvert_\infty$	4.1	1,102.034	0.062	-0.199	0.853	0.002	-0.009	15.034	0.015	0.053
Complex	No	SciPy	TNC	$\lvert\lvert\nabla q\rvert\rvert_\infty$	6.7	1,102.034	0.062	-0.199	0.689	0.002	-0.010	14.788	0.015	0.053
Complex	No	SciPy	TNC	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	6.2	1,102.034	0.062	-0.199	0.756	0.002	-0.010	14.944	0.015	0.053
Complex	No	SciPy	Nelder-Mead	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	91.1	1,102.034	0.062	-0.199	0.592	0.002	-0.011	14.645	0.015	0.053
Complex	Yes	Knitro	Interior/Direct	$\lvert\lvert\nabla q\rvert\rvert_\infty$	13.5	1,102.034	0.062	-0.199	-2.553	0.000	-0.006	12.391	0.009	0.031
Complex	Yes	Knitro	Active Set	$\lvert\lvert\nabla q\rvert\rvert_\infty$	10.3	1,102.034	0.062	-0.199	-2.518	0.000	-0.006	12.376	0.009	0.031
Complex	Yes	Knitro	SQP	$\lvert\lvert\nabla q\rvert\rvert_\infty$	13.2	1,102.034	0.062	-0.199	-2.502	0.000	-0.006	12.391	0.009	0.031
Complex	Yes	SciPy	L-BFGS-B	$\lvert\lvert\nabla q\rvert\rvert_\infty$	10.4	1,102.034	0.062	-0.199	-2.502	0.000	-0.006	12.391	0.009	0.031
Complex	Yes	SciPy	BFGS	$\lvert\lvert\nabla q\rvert\rvert_\infty$	10.0	1,102.034	0.062	-0.199	-2.304	0.000	-0.006	12.391	0.009	0.031
Complex	Yes	SciPy	TNC	$\lvert\lvert\nabla q\rvert\rvert_\infty$	18.8	1,102.034	0.062	-0.199	-2.418	0.000	-0.006	12.437	0.009	0.031
Complex	Yes	SciPy	TNC	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	14.3	1,102.034	0.062	-0.199	-2.291	0.000	-0.006	12.398	0.009	0.031
Complex	Yes	SciPy	Nelder-Mead	$\lvert\lvert\theta^n - \theta^{n-1}\rvert\rvert_\infty$	162.0	1,102.034	0.062	-0.199	-2.518	0.000	-0.006	12.391	0.009	0.031

Continued on the next page.

Continued from the previous page.

						True Median Value			Median Bias			Median Absolute Error		
Simulation	Supply	Software	Algorithm	Termination	Seconds	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS	$\Delta\overline{\text{HHI}}$	ΔPS	ΔCS
RCNL	No	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	8.8	934.505	0.070	-0.160	-0.669	0.009	-0.021	6.169	0.015	0.035
RCNL	No	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	8.3	934.505	0.070	-0.160	-0.357	0.008	-0.020	6.160	0.015	0.035
RCNL	No	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	9.3	934.505	0.070	-0.160	-0.593	0.009	-0.021	6.173	0.015	0.035
RCNL	No	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	7.1	934.505	0.070	-0.160	-0.354	0.008	-0.020	6.066	0.015	0.034
RCNL	No	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	6.8	934.505	0.070	-0.160	-0.372	0.008	-0.021	6.130	0.015	0.035
RCNL	No	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	11.9	934.505	0.070	-0.160	-0.351	0.008	-0.021	6.069	0.015	0.034
RCNL	No	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	10.0	934.505	0.070	-0.160	-0.435	0.008	-0.020	6.009	0.015	0.034
RCNL	No	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	199.1	934.505	0.070	-0.160	-0.354	0.008	-0.020	6.160	0.015	0.034
RCNL	Yes	Knitro	Interior/Direct	$\|\|\nabla q\|\|_\infty$	20.4	934.505	0.070	-0.160	-1.598	0.001	-0.004	5.693	0.007	0.017
RCNL	Yes	Knitro	Active Set	$\|\|\nabla q\|\|_\infty$	18.0	934.505	0.070	-0.160	-1.571	0.001	-0.003	5.693	0.007	0.017
RCNL	Yes	Knitro	SQP	$\|\|\nabla q\|\|_\infty$	20.6	934.505	0.070	-0.160	-1.609	0.001	-0.003	5.693	0.007	0.017
RCNL	Yes	SciPy	L-BFGS-B	$\|\|\nabla q\|\|_\infty$	14.9	934.505	0.070	-0.160	-1.590	0.001	-0.003	5.693	0.007	0.017
RCNL	Yes	SciPy	BFGS	$\|\|\nabla q\|\|_\infty$	14.2	934.505	0.070	-0.160	-1.571	0.001	-0.003	5.725	0.007	0.017
RCNL	Yes	SciPy	TNC	$\|\|\nabla q\|\|_\infty$	25.9	934.505	0.070	-0.160	-1.555	0.001	-0.003	5.658	0.007	0.017
RCNL	Yes	SciPy	TNC	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	19.9	934.505	0.070	-0.160	-1.555	0.001	-0.003	5.661	0.007	0.017
RCNL	Yes	SciPy	Nelder-Mead	$\|\|\theta^n - \theta^{n-1}\|\|_\infty$	283.4	934.505	0.070	-0.160	-1.571	0.001	-0.003	5.751	0.007	0.017

This table documents bias and variance of merger simulation estimates along with estimation speed over 1,000 simulated datasets for different optimization algorithms. It is created from the same results as Table 11 after a second GMM step.