Constrained Optimization Approaches to Structural Estimation

Che-Lin Su
University of Chicago
Graduate School of Business
csu@chicagogsb.edu

Institute for Computational Economics
The University of Chicago
July 28 – August 8, 2008

Outline of Three Lectures

1. Introduction to Structural Estimation

Outline of Three Lectures

1. Introduction to Structural Estimation
2. Estimation of Demand Systems

Outline of Three Lectures

1. Introduction to Structural Estimation
2. Estimation of Demand Systems
3. Estimation of Dynamic Programming Models of Individual Behavior

Outline of Three Lectures

1. Introduction to Structural Estimation
2. Estimation of Demand Systems
3. Estimation of Dynamic Programming Models of Individual Behavior
4. Estimation of Games

Part I

Random-Coefficients Demand Estimation

Structural Estimation

- Great interest in estimating models based on economic structure
  - DP models of individual behavior: Rust (1987) – NFXP
  - Nash equilibria of games – static, dynamic: Ag-M (2007) – PML
  - Demand Estimation: BLP(1995), Nevo(2000)
  - Auctions: Paarsch and Hong (2006), Hubbard and Paarsch (2008)
  - Dynamic stochastic general equilibrium
  - Popularity of structural models in empirical IO and marketing
- Model sophistication introduces computational difficulties
- General belief: Estimation is a major computational challenge because it involves solving the model many times
- Our goal: Propose a **unified**, **reliable**, and **more computational efficient** way of estimating structural models
- Our finding: Many supposed computational "difficulties" can be avoided by using constrained optimization methods and software

Current Views on Structural Estimation

Tulin Erdem, Kannan Srinivasan, Wilfred Amaldoss, Patrick Bajari, Hai Che, Teck Ho, Wes Hutchinson, Michael Katz, Michael Keane, Robert Meyer, and Peter Reiss, "Theory-Driven Choice Models", *Marketing Letters* (2005)

> *Estimating structural models can be computationally difficult. For example, dynamic discrete choice models are commonly estimated using the nested fixed point algorithm (see Rust 1994). This requires solving a dynamic programming problem thousands of times during estimation and numerically minimizing a nonlinear likelihood function....[S]ome recent research ... proposes computationally simple estimators for structural models ... The estimators ... use a two-step approach. ....The two-step estimators can have drawbacks. First, there can be a loss of efficiency. .... Second, stronger assumptions about unobserved state variables may be required. .... However, two-step approaches are computationally light, often require minimal parametric assumptions and are likely to make structural models accessible to a larger set of researchers.*

Optimization and Computation in Structural Estimation

- Optimization often perceived as 2nd-order importance to research agenda
- Typical computational methohd is Nested fixed-point problem: fixed-point calculation embedded in calculation of objective function
  - compute an "equilibrium"
  - invert a model (e.g. non-linearity in disturbance)
  - compute a value function (i.e. dynamic model)
- Mis-use of optimization can lead to the "wrong answer"
  - naively use canned optimization algorithms – e.g., fmincon
  - use the default settings
  - adjust default-settings to improve speed not accuracy
  - assume there is a unique fixed-point
  - CHECK SOLVER OUTPUT MESSAGE!!!
    - KNITRO: LOCALLY OPTIMAL SOLUTION FOUND.
    - Filter-MPEC: Optimal Solution Found.
    - SNOPT: Optimal Solution Found.

Random-Coefficients Logit Demand

- Berry, Levinsohn and Pakes (1995): Logit with endogenous regressors and unobserved heterogeneity
- Estimated frequently in empirical IO and marketing
- Utility of consumer i from purchasing product j in market t

$$u_{ijt} = \beta_i^0 + x_{jt}\beta_i^x - \beta_i^p p_{jt} + \xi_{jt} + \varepsilon_{ijt}$$

  - ξ_{jt}: not observed
  - x_{jt}, p_{jt} observed; $cov(\xi_{jt}, p_{jt}) \neq 0$
  - β: individual-specific taste coefficients to be estimated; $\beta \sim F_\beta(\beta;\theta)$
- Predicted market share

$$s_j(x_t, p_t, \xi_t, ; \theta) = \int_\beta \frac{\exp(\beta^0 + x_{jt}\beta^x - \beta^p p_{jt} + \xi_{jt})}{1 + \sum_{k=1}^J \exp(\beta^0 + x_{kt}\beta^x - \beta^p p_{kt} + \xi_{kt})} dF_\beta(\beta;\theta)$$

Random-Coefficients Logit Demand: GMM Estimation

- Assume $E\left[\xi_{jt} z_{jt} | z_{jt}\right] = 0$ for some vector of instruments z_{jt}
  - Empirical analog $g\left(\theta\right) = \frac{1}{TJ} \sum_{t=1}^{T} \sum_{j=1}^{J} \xi'_{jt} zjt$
  - Estimate $\theta^{GMM} = \underset{\theta}{argmin} \left\{ g\left(\theta\right)' W g\left(\theta\right) \right\}$
- Cannot compute ξ_j analytically
  - "Invert" ξ_t from system of predicted market shares numerically

$$\begin{array}{rcl} S_t & = & s\left(x_t, p_t, \xi_t; \theta\right) \\ \Rightarrow \quad \xi_t\left(\theta\right) & = & s^{-1}\left(x_t, p_t, S_t; \theta\right) \end{array}$$

  - BLP propose contraction-mapping for inversion, i.e., fixed-point calculation
  - Inversion nested into parameter search ... NFP
- inner-loop: fixed-point calculation, $\xi_t(\theta)$
- outer-loop: minimization, θ^{GMM}

BLP/NFP Estimation Algorithm

- Outer loop: $\min_{\theta} g(\theta)' W g(\theta)$
  - Guess θ parameters to compute $g(\theta) = \frac{1}{TJ} \sum_{t=1}^{T} \sum_{j=1}^{J} \xi_{jt}(\theta)' zjt$
  - Stop when $\|\nabla_{\theta}(g(\theta)' W g(\theta))\| \leq \epsilon_{out}$

BLP/NFP Estimation Algorithm

- Outer loop: $\min_{\theta} \; g(\theta)' W g(\theta)$
  - Guess θ parameters to compute $g(\theta) = \frac{1}{TJ} \sum_{t=1}^{T} \sum_{j=1}^{J} \xi_{jt}(\theta)' zjt$
  - Stop when $\|\nabla_{\theta}(g(\theta)' W g(\theta))\| \leq \epsilon_{out}$
- Inner loop: compute $\xi_t(\theta)$ for a given θ
  - Solve $s_t(x_j, p_t, \xi_t; \theta) = S_{\cdot t}$ for ξ by contraction mapping:

$$\xi_t^{h+1} = \xi_t^h + \log\, S_t - \log\, s_t(x_j, p_t, \xi_t; \theta)$$

    until $\|\xi_{\cdot t}^{h+1} - \xi_{\cdot t}^h\| \leq \epsilon_{in}$
  - Denote the approximated demand shock by $\xi(\theta, \epsilon_{in})$
- Stopping rules: need to choose tolerance/stopping criterion for both inner loop (ϵ_{in}) and outer loop (ϵ_{out})

Concerns with NFP/BLP

- Inefficient amount of computation
  - we only need to know $\xi(\theta)$ at the true θ
  - NFP solves inner-loop exactly each stage of parameter search
- Stopping rules: choosing inner-loop and outer-loop tolerances
  - inner-loop can be slow (especially for bad guesses of θ): contraction mapping is linear convergent at best
  - tempting to loosen inner loop tolerance ϵ_{in} used
    - often see $\epsilon_{in} = 1.e - 6$ or higher
  - outer loop may not converge with loose inner loop tolerance
    - check solver output message; see Knittel and Metaxoglou (2008)
    - tempting to loosen outer loop tolerance ϵ_{in} to promote convergence
    - often see $\epsilon_{out} = 1.e - 3$ or higher
- Inner-loop error propagates into outer-loop

Numerical Experiment: 100 different starting points

- 1 dataset: 75 markets, 25 products, 10 structural parameters
  - NFP tight: $\epsilon_{in} = 1.e{-}10$ $\epsilon_{out} = 1.e{-}6$
  - NFP loose inner: $\epsilon_{in} = 1.e{-}4$ $\epsilon_{out} = 1.e{-}6$
  - NFP loose both: $\epsilon_{in} = 1.e{-}4$ $\epsilon_{out} = 1.e{-}2$

GMM objective values

Starting point	NFP tight	NFP loose inner	NFP loose both
1	$4.3084e-02$	Fail	$7.9967e+01$
2	$4.3084e-02$	Fail	$9.7130e-02$
3	$4.3084e-02$	Fail	$1.1873e-01$
4	$4.3084e-02$	Fail	$1.3308e-01$
5	$4.3084e-02$	Fail	$7.3024e-02$
6	$4.3084e-02$	Fail	$6.0614e+01$
7	$4.3084e-02$	Fail	$1.5909e+02$
8	$4.3084e-02$	Fail	$2.1087e-01$
9	$4.3084e-02$	Fail	$6.4803e+00$
10	$4.3084e-02$	Fail	$1.2271e+03$

Main findings: Loosening tolerance leads to non-convergence

- Check optimization exit flags!
- algorithm may not produce a local optimum!

Stopping Rules

- Notations:
  - $Q(\xi(\theta, \epsilon_{in}))$: the programmed GMM objective function with ϵ_{in}
  - L: the Lipschitz constant of the inner-loop contraction mapping
- Analytic derivatives $\nabla_\theta Q(\xi(\theta))$ is provided: $\epsilon_{out} = O(\frac{L}{1-L}\epsilon_{in})$
- Finite-difference derivatives are used: $\epsilon_{out} = O(\sqrt{\frac{L}{1-L}\epsilon_{in}})$

MPEC Applied to BLP

- Mathematical Programming with Equilibrium Constraints
  - Su and Judd (2008), application by Vitorino (2008)
  - Use constrained optimization - system defining fixed-point used as constraints
- For our Logit Demand example with GMM:

$$\min_{\theta,\xi} \quad g(\xi)' W g(\xi)$$
$$\text{subject to} \quad s(\xi;\theta) = S$$

  - No inner loop (no contraction-mapping)
    - No need to worry about setting up two tolerance levels
  - Easier to implement
  - Potentially faster than NFP b/c share only needs to hold at solution
  - Even larger benefits for problems with multiple inner-loops (i.e. dynamic demand)

AMPL Model: MPEC_BLP.mod

```
param ns ;    # := 20 ;     # number of simulated "individuals" per market
param nmkt ;  # := 94 ;     # number of markets
param nbrn ;  # := 24 ;     # number of brands per market
param nbrnPLUS1 := nbrn+1;  # number of products plus outside good
param nk1 ;   # := 25;      # of observable characteristics
param nk2 ;   # := 4 ;      # of observable characteristics
param niv ;   # := 21 ;     # of instrument variables
param nz := niv-1 + nk1 -1; # of instruments including iv and X1
param nd ;    # := 4 ;      # of demographic characteristics

set S := 1..ns ;        # index set of individuals
set M := 1..nmkt ;      # index set of market
set J := 1..nbrn ;      # index set of brand (products), including outside good
set MJ := 1..nmkt*nbrn; # index of market and brand
set K1 := 1..nk1 ;      # index set of product observable characteristics
set K2 := 1..nk2 ;      # index set of product observable characteristics
set Demogr := 1..nd;
set DS := 1..nd*ns;
set K2S := 1..nk2*ns;

set H := 1..nz ;        # index set of instrument including iv and X1
```

AMPL Model: MPEC_BLP.mod

```
## Define input data format:
param X1 {mj in MJ, k in K1} ;
param X2 {mj in MJ, k in K2} ;
param ActuShare {m in MJ} ;
param Z {mj in MJ, h in H} ;
param D {m in M, di in DS} ;
param v {m in M, k2i in K2S} ;
param invA {i in H, j in H} ;   # optimal weighting matrix = inv(Z'Z);
param OutShare {m in M} := 1 - sum {mj in (nbrn*(m-1)+1)..(nbrn*m)} ActuShare[mj];
```

AMPL Model: MPEC_BLP.mod

```
## Define variables

var theta1 {k in K1};
var SIGMA {k in K2};
var PI {k in K2, d in Demogr};
var delta {mj in MJ} ;

var EstShareIndivTop {mj in MJ, i in S} = exp( delta[mj]
+ sum {k in K2} (X2[mj,k]*SIGMA[k]*v[ceil(mj/nbrn), i+(k-1)*ns])
+ sum{k in K2, d in Demogr} (X2[mj,k]*PI[k,d]*D[ceil(mj/nbrn),i+(d-1)*ns]) );

var EstShareIndiv{mj in MJ, i in S} = EstShareIndivTop[mj,i] / (1+ sum{
l in ((ceil(mj/nbrn)-1)*nbrn+1)..(ceil(mj/nbrn)*nbrn)} EstShareIndivTop[l, i]);

var EstShare {mj in MJ} = 1/ns * (sum{i in S} EstShareIndiv[mj,i]) ;

var w {mj in MJ} = delta[mj] - sum {k in K1} (X1[mj,k]*theta1[k]) ;

var Zw {h in H} ;  ##  Zw{h in H} = sum {mj in MJ} Z[mj,h]*w[mj];
```

AMPL Model: MPEC_BLP.mod

```
minimize GMM : sum{h1 in H, h2 in H} Zw[h1]*invA[h1, h2]*Zw[h2];

subject to

     conZw {h in H}: Zw[h] = sum {mj in MJ} Z[mj,h]*w[mj] ;

     Shares {mj in MJ}: log(EstShare[mj]) = log(ActuShare[mj]) ;
```

Monte Carlo: Varying the Lipschitz Constant

- 50 markets, 25 products, 30 replications per case
- $E[\beta_i] = \{E[\beta_i^0], 1.5, 1.5, 0.5, -3\}$; $Var[\beta_i] = \{0.5, 0.5, 0.5, 0.5, 0.2\}$
- MPEC: optimality and feasibility tolerances $= 1.e - 6$

Intercept $E[\beta_i^0]$	Lipschitz Constant	Implementation	Runs Converged	CPU Time (sec.)	Elas Bias	Elas RMSE
-2	0.780	NFP tight	30	481.1	0.007	0.316
		MPEC	30	552.1	-0.007	0.358
-1	0.879	NFP tight	30	566.3	0.035	0.364
		MPEC	30	527.5	-0.039	0.330
0.1 (base case)	0.944	NFP tight	30	780.0	0.046	0.385
		MPEC	30	564.7	-0.071	0.360
1	0.973	NFP tight	30	1381.5	0.009	0.370
		MPEC	30	521.7	-0.072	0.367
2	0.989	NFP tight	30	2860.7	0.046	0.382
		MPEC	30	551.6	-0.044	0.344
3	0.996	NFP tight	30	5720.7	0.055	0.406
		MPEC	30	600.7	-0.073	0.370
4	0.998	NFP tight	30	11248.0	0.036	0.349
		MPEC	30	858.3	-0.072	0.375

Monte Carlo Results: Various the # of Markets

- 25 products, 30 replications per case
- Intercept $E[\beta_i^0] = 0.1$

# of Markets	Lipschitz Constant	Stopping Rule	Runs Converged	CPU Time (sec.)	Elas Bias	Elas RMSE
25	0.937	NFP tight	30	258.5	0.060	0.432
		MPEC	30	226.8	-0.055	0.349
50 (base case)	0.944	NFP tight	30	780.0	0.046	0.385
		MPEC	30	564.7	-0.071	0.360
100	0.951	NFP tight	30	2559.6	0.032	0.377
		MPEC	30	2866.0	-0.038	0.216
200	0.953	NFP tight	30	6481.7	0.036	0.313
		MPEC	30	2543.6	-0.039	0.165

Monte Carlo Evidence

BLP/NFP

- Contraction mapping is linear convergent at best
- Needs to be careful at setting inner and outer tolerance
  - With analytic derivatives: $\epsilon_{out} = O(\epsilon_{in})$
  - With finite-difference derivatives: $\epsilon_{out} = O\left(\sqrt{\epsilon_{in}}\right)$
    - Needs very high accuracy from the inner loop in order for the outer loop to converge
  - Lipschitz constant: bound on convergence of contraction-mapping
    - Experiments show datasets with higher Lipschitz converge more slowly

MPEC

- Newton-based methods are locally quadratic convergent
- Two key factors in efficient implementations:
  - Provide analytic-derivatives – huge improvement in speed
  - Exploit sparsity pattern in constraint Jacobian – huge saving in memory requirement

Pattern of Constraint Jacobian

SORTING: Products and then Markets

		Prod=1					Prod=2					Prod=3				
		T1	T2	T3	T4	T5	T1	T2	T3	T4	T5	T1	T2	T3	T4	T5
	T1	X					X					X				
	T2		X					X					X			
Prod=1	T3			X					X					X		
	T4				X					X					X	
	T5					X					X					X
	T1	X					X					X				
	T2		X					X					X			
Prod=2	T3			X					X					X		
	T4				X					X					X	
	T5					X					X					X
	T1	X					X					X				
	T2		X					X					X			
Prod=3	T3			X					X					X		
	T4				X					X					X	
	T5					X					X					X

SORTING: Markets and then Products

		T=1			T=2			T=3			T=4			T=5		
		P1	P2	P3	P1	P2	P3	P1	P2	P3	P1	P2	P3	P1	P2	P3
	P1	X	X	X												
T=1	P2	X	X	X												
	P3	X	X	X												
	P1				X	X	X									
T=2	P2				X	X	X									
	P3				X	X	X									
	P1							X	X	X						
T=3	P2							X	X	X						
	P3							X	X	X						
	P1										X	X	X			
T=4	P2										X	X	X			
	P3										X	X	X			
	P1													X	X	X
T=5	P2													X	X	X
	P3													X	X	X

Summary

- Constrained optimization formulation for the random-coefficients demand estimation model is

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

- The MPEC approach is reliable and has speed advantage
- It allows researchers to access best optimization solvers

Part II

Estimation of Dynamic Programming Models

Rust (1987): Zurcher's Data

Bus #: 5297

events	year	month	odometer at replacement
1st engine replacement	1979	June	242400
2nd engine replacement	1984	August	384900

year	month	odometer reading
1974	Dec	112031
1975	Jan	115223
1975	Feb	118322
1975	Mar	120630
1975	Apr	123918
1975	May	127329
1975	Jun	130100
1975	Jul	133184
1975	Aug	136480
1975	Sep	139429

Zurcher's Bus Engine Replacement Problem

- Rust (1987)
- Each bus comes in for repair once a month
  - Bus repairman sees mileage x_t at time t *since last engine overhaul*
  - Repairman chooses between overhaul and ordinary maintenance

$$u(x_t, d_t, \theta^c, RC) = \begin{cases} -c(x_t, \theta^c) & \text{if} \quad d_t = 0 \\ -(RC + c(0, \theta^c)) & \text{if} \quad d_t = 1 \end{cases}$$

  - Repairman solves DP:

$$V_\theta(x_t) = \sup_{\{f_t, f_{t+1}, \ldots\}} E \left\{ \sum_{j=t}^{\infty} \beta^{j-t} \left[u(x_j, f_j, \theta) + \varepsilon_j(f_j) \right] | x_t \right\}$$

- Econometrician
  - Observes mileage x_t and decision d_t, but not cost
  - Assumes extreme value distribution for $\varepsilon_t(d_t)$
- Structural parameters to be estimated: $\theta = (\theta^c, RC, \theta^p)$
  - Coefficients of operating cost function; e.g., $c(x, \theta^c) = \theta_1^c x + \theta_2^c x^2$
  - Overhaul cost RC
  - Transition probabilities in mileages $p(x_{t+1} | x_t, d_t, \theta^p)$

Zurcher's Bus Engine Replacement Problem

- Data: time series $(x_t, d_t)_{t=1}^{T}$
- Likelihood function

$$L(\theta) = \prod_{t=2}^{T} P(d_t | x_t, \theta^c, RC) p(x_t | x_{t-1}, d_{t-1}, \theta^p)$$

with $P(d|x, \theta^c, RC) = \dfrac{exp\{u(x, d, \theta^c, RC) + \beta EV_\theta(x, d)\}}{\sum_{d' \in \{0,1\}} exp\{u(x, d', \theta^c, RC) + \beta EV_\theta(x', d)\}}$

$$EV_\theta(x, d) = T_\theta(EV_\theta)(x, d)$$

$$\equiv \int_{x'=0}^{\infty} \log \left[\sum_{d' \in \{0,1\}} exp\{u(x', d', \theta^c, RC) + \beta EV_\theta(x', d')\} \right] p(dx' | x, d, \theta^p)$$

Nested Fixed Point Algo: Rust (1987)

- Outer loop: Solve likelihood

$$\max_{\theta \geq 0} \prod_{t=2}^{T} P(d_t|x_t, \theta^c, RC) p(x_t|x_{t-1}, d_{t-1}, \theta^p)$$

- Inner loop: Compute expected value function EV_θ for a given θ
  - EV_θ is the implicit expected value function defined by the Bellman equation or the fixed point function

$$EV_\theta = T_\theta(EV_\theta)$$

  - Rust started with contraction iterations and then switched to Newton iterations
- Problem with NFXP: Must compute EV_θ to high accuracy for each θ examined
  - for outer loop to converge
  - to obtain accurate numerical derivatives for the outer loop

MPEC Approach for Solving Zucher Model

- Form augmented likelihood function for data $X = (x_t, d_t)_{t=1}^{T}$

$$\mathcal{L}(\theta, EV; X) = \prod_{t=2}^{T} P(d_t|x_t, \theta^c, RC)p(x_t|x_{t-1}, d_{t-1}, \theta^p)$$

with $P(d|x, \theta^c, RC) = \frac{exp\{u(x, d, \theta^c, RC) + \beta EV(x, d)\}}{\sum_{d' \in \{0,1\}} exp\{u(x, d', \theta^c, RC) + \beta EV(x, d')\}}$

- Rationality and Bellman equation imposes a relationship between θ and EV

$$EV = T(EV, \theta)$$

- Solve constrained optimization problem

$$\begin{array}{ll} \max\limits_{(\theta, EV)} & \mathcal{L}(\theta, EV; X) \\ \text{subject to} & EV = T(EV, \theta) \end{array}$$

MPEC Applied to Zucher: Three-Parameter Estimates

- Synthetic data is better: avoids misspecification
- Use Rust's estimates to generate 2 synthetic data sets of 10^3 and 10^4 data points respectively.
- Rust discretized mileage space into 90 intervals of length 5000 $(N = 91)$
- AMPL program solved on NEOS server using SNOPT

T	N	Estimates RC	θ_1^c	θ_2^c	CPU (sec)	Major Iterations	Evals*	Bell. EQ. Error
10^3	101	1.112	0.043	0.0029	0.14	66	72	3.0E−13
10^3	201	1.140	0.055	0.0015	0.31	44	59	2.9E−13
10^3	501	1.130	0.050	0.0019	1.65	58	68	1.4E−12
10^3	1001	1.144	0.056	0.0013	5.54	58	94	2.5E−13
10^4	101	1.236	0.056	0.0015	0.24	59	67	2.9E−13
10^4	201	1.257	0.060	0.0010	0.44	59	67	1.8E−12
10^4	501	1.252	0.058	0.0012	0.88	35	45	2.9E−13
10^4	1001	1.256	0.060	0.0010	1.26	39	52	3.0E−13

*Number of function and constraint evaluations

MPEC Applied to Zucher: Five-Parameter Estimates

- Rust did a two-stage procedure, estimating transition parameters in first stage. We do full ML

T	N	Estimates RC	θ_1^c	θ_2^c	θ_1^p	θ_2^p	CPU (sec)	Maj. Iter.	Evals	Bell. Err.
10^3	101	1.11	0.039	0.0030	0.723	0.262	0.50	111	137	6E−12
10^3	201	1.14	0.055	0.0015	0.364	0.600	1.14	109	120	1E−09
10^3	501	1.13	0.050	0.0019	0.339	0.612	3.39	115	127	3E−11
10^3	1001	1.14	0.056	0.0014	0.360	0.608	7.56	84	116	5E−12
10^4	101	1.24	0.052	0.0016	0.694	0.284	0.50	76	91	5E−11
10^4	201	1.26	0.060	0.0010	0.367	0.053	0.86	85	97	4E−11
10^4	501	1.25	0.058	0.0012	0.349	0.596	2.73	83	98	3E−10
10^4	1001	1.26	0.060	0.0010	0.370	0.586	19.12	166	182	3E−10

Observations

- Problem is solved very quickly.
- Timing is nearly linear in the number of states for modest grid size.
- The likelihood function, the constraints, and their derivatives are evaluated only 45-200 times in this example.
- In contrast, the Bellman operator (the constraints here) is solved hundreds of times in NFXP

Parametric Bootstrap Experiment

- For calculating statistical inference, bootstrapping is better and more reliable than asymptotic analysis. However, bootstrap is often viewed as computationally infeasible
- Examine several data sets to determine patterns
- Use Rust's estimates to generate 1 synthetic data set
- Use the estimated values on the synthetic data set to reproduce 20 independent data sets:
    - Five parameter estimation
    - 1000 data points
    - 201 grid points in DP

Maximum Likelihood Parametric Bootstrap Estimates

Table 3: Maximum Likelihood Parametric Bootstrap Results

	Estimates						CPU (sec)	Maj. Ite	Evals	Bell. Err.
	RC	θ_1^c	θ_2^c	θ_1^p	θ_2^p	θ_3^p				
mean	1.14	0.037	0.004	0.384	0.587	0.029	0.54	90	109	8E-09
S.E.	0.15	0.035	0.004	0.013	0.012	0.005	0.16	24	37	2E-08
Min	0.95	0.000	0.000	0.355	0.571	0.021	0.24	45	59	1E-13
Max	1.46	0.108	0.012	0.403	0.606	0.039	0.88	152	230	6E-08

MPEC Approach to Method of Moments

- Suppose you want to fit moments. E.g., likelihood may not exist
- Method then is

$$\begin{array}{cc} \min\limits_{(\theta,\sigma)} & \|m(\theta,\sigma) - M(X)\|^2 \\ \text{subject to} & G(\theta,\sigma) = 0 \end{array}$$

  - Compute moments $m(\theta, EV)$ numerically via linear equations in constraints - no simulation
  - Objective function for the Rust's bus example:

$$\begin{aligned} \mathcal{M}(m,M) \quad &= (m_x - M_x)^2 + (m_d - M_d)^2 + (m_{xx} - M_{xx})^2 + (m_{xd} - M_{xd})^2 \\ &+ (m_{dd} - M_{dd})^2 + (m_{xxx} - M_{xxx})^2 + (m_{xxd} - M_{xxd})^2 \\ &+ (m_{xdd} - M_{xdd})^2 + (m_{ddd} - M_{ddd})^2 \end{aligned}$$

Formulation for Method of Moments

- Constraints imposing equilibrium conditions and moment definitions, transition matrix Π and computes stationary distribution p

$$\max_{(\theta, EV, \Pi, p, m)} \quad \mathcal{M}(m, M)$$

subject to

$$EV = T(\theta, EV), \quad \Pi = H(\theta, EV)$$

$$p^\top \Pi = p^\top, \quad \sum_{x \in Z, d \in \{0,1\}} p_{x,d} = 1$$

$$m_x = \sum_{x,d} p_{x,d}\, x, \;\; m_d = \sum_{x,d} p_{x,d}\, d$$

$$m_{xx} = \sum_{x,d} p_{x,d}\,(x - m_x)^2, \;\; m_{xd} = \sum_{x,d} p_{x,d}\,(x - m_x)(d - m_d)$$

$$m_{dd} = \sum_{x,d} p_{x,d}\,(d - m_d)^2$$

$$m_{xxx} = \sum_{x,d} p_{x,d}\,(x - m_x)^3, \;\; m_{xxd} = \sum_{x,d} p_{x,d}\,(x - m_x)^2(d - m_d)$$

$$m_{xdd} = \sum_{x,d} p_{x,d}\,(x - m_x)(d - m_d)^2, \;\; m_{ddd} = \sum_{x,d} p_{x,d}\,(d - m_d)^3$$

Method of Moments Parametric Bootstrap Estimates

Table 4: Method of Moments Parametric Bootstrap Results

	Estimates						CPU	Major	Evals	Bell
	RC	θ_1^c	θ_2^c	θ_1^p	θ_2^p	θ_3^p	(sec)	Iter		Err.
mean	1.0	0.05	0.001	0.397	0.603	0.000	22.6	525	1753	7E-06
S.E.	0.3	0.03	0.002	0.040	0.040	0.001	16.9	389	1513	1E-05
Min	0.1	0.00	0.000	0.340	0.511	0.000	5.4	168	389	2E-10
Max	1.5	0.10	0.009	0.489	0.660	0.004	70.1	1823	6851	4E-05

- Solving GMM is not as fast as solving MLE
  - the larger size of the moments problem
  - the nonlinearity introduced by the constraints related to moments, particularly the skewness equations.

Part III

General Formulations

Standard Problem and Current Approach

- Individual solves an optimization problem
- Econometrician observes states and decisions
- Want to estimate structural parameters and equilibrium solutions that are consistent with structural parameters
- Current standard approach
  - Structural parameters: θ
  - Behavior (decision rule, strategy, price): σ
  - Equilibrium (optimality or competitive or Nash) imposes

$$G(\theta, \sigma) = 0$$

  - Likelihood function for data X and parameters θ

$$\max_{\theta} L(\theta; X)$$

where equilibrium can be presented by $\sigma = \Sigma(\theta)$

NFXP Applied to DP – Rust (1987)

- $\Sigma(\theta)$ is single-valued
- Outline of NFXP
  - Given θ, compute $\sigma = \Sigma(\theta)$ by solving $G(\theta, \sigma) = 0$
  - For each θ, define

$$L(\theta; X) = \text{likelihood given } \sigma = \Sigma(\theta)$$

  - Compute

$$\max_{\theta} L(\theta; X)$$

NFXP Applied to Games with Multiple Equilibria

- $\Sigma(\theta)$ is multi-valued
- Outline of NFXP
  - Given θ, compute all $\sigma \in \Sigma(\theta)$
  - For each θ, define

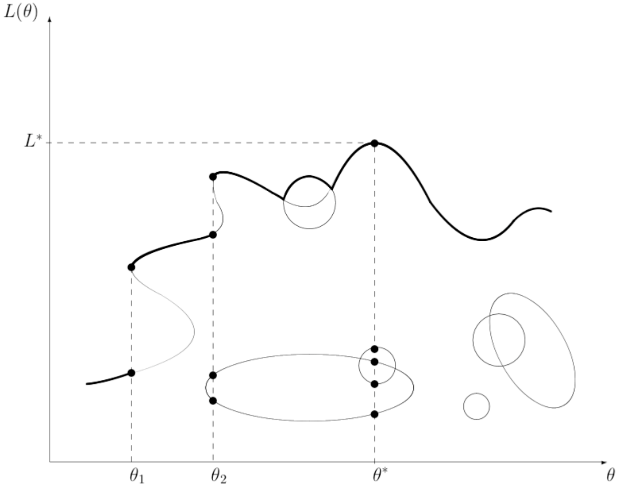
$$L(\theta; X) = \text{max likelihood over all } \sigma \in \Sigma(\theta)$$

  - Compute

$$\max_{\theta} L(\theta; X)$$

- If $\Sigma(\theta)$ is multi-valued, then L can be nondifferentiable and/or discontinuous

NFXP Applied to Games with Multiple Equilibria



MPEC Ideas Applied to Estimation

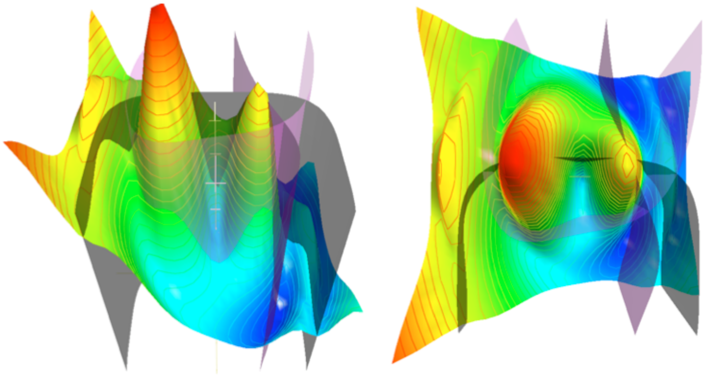
- Structural parameters: θ
- Behavior (decision rule, strategy, price mapping): σ
- Equilibrium conditions impose

$$G(\theta, \sigma) = 0$$

- Denote the *augmented likelihood* of a data set, X, by $\mathcal{L}(\theta, \sigma; X)$
  - $\mathcal{L}(\theta, \sigma; X)$ decomposes $L(\theta; X)$ so as to highlight the seperate dependence of likelihood on θ and σ
  - In fact, $L(\theta; X) = \mathcal{L}(\theta, \Sigma(\theta); X)$
- Therefore, maximum likelihood estimation is

$$\begin{array}{ll} \max\limits_{(\theta,\sigma)} & \mathcal{L}(\theta, \sigma; X) \\ \text{subject to} & G(\theta, \sigma) = 0 \end{array}$$

MPEC Applied to Games with Multiple Equilibria



Our Advantanges

- Both $\mathcal{L}$ and G are smooth functions
- We do not require that equilibrium conditions be defined as a solution to a fixed-point equation
- We do not need to specify an algorithm for computing σ given θ
- We do not need to solve for all equilibria σ for every θ
- Using a constrained optimization approach allows one to take advantage of the best available methods and software (AMPL, KNITRO, SNOPT, filterSQP, PATH, etc)

So ... What is NFXP?

- NFXP is equivalent to nonlinear elimination of variables
- Consider

$$\begin{array}{ll} \max_{(x,y)} & f(x,y) \\ \text{subject to} & g(x,y)=0 \end{array}$$

  - Define $Y(x)$ implicitly by $g(x, Y(x)) = 0$
  - Solve the unconstrained problem

$$\max_{x} f(x, Y(x))$$

- Used only when memory demands are too large
- Often creates very difficult unconstrained optimization problems

Constrained Estimation

- The MPEC approach is an example of constrained estimation, be it maximum likelihood or method of moments.
- Sampling of previous literature

  - Aitchison, J. & S.D. Silvey (1958): Maximum likelihood estimation of parameters subject to restraints. *Annals of Mathematical Statistics*, 29, 813–828.
  - Gallant, A.R., and A. Holly (1980): Statistical inference in an implicit, nonlinear, simultaneous equation model in the context of maximum likelihood estimation. *Econometrica*, 48, 697–720.
  - Gallant, A.R., and G. Tauchen (1989): Seminonparametric estimation of conditionally constrained heterogeneous processes: asset pricing applications. *Econometrica*, 57, 1091–1120.
  - Silvey, S.D. *Statistical Inference*. London: Chapman & Hall, 1970.
  - Wolak, F.A. (1987): An exact test for multiple inequality and equality constraints in the linear regression model. *J. Am. Statist. Assoc.* 82, 782–793.
  - Wolak, F.A. (1989): Testing inequality constraints in linear econometric models. *Journal of Econometrics*, 41, 205–235.

Part IV

Estimation of Games

NFXP and Related Methods to Games

- For any given θ, NFXP requires finding all σ that solve $G(\theta, \sigma) = 0$, compute the likelihood at each such σ, and report the max as the likelihood value $L(\theta)$
- Finding all equilibria for arbitrary games is an essentially intractable problem - see Judd and Schmedders (2006)
- One fundamental issue: G-S or G-J type methods (e.g., Pakes-McGuire) are often used to solve for an equilibrium. This implicitly imposes an undesired **equilibrium selection rule**: converge only to equilibria that are stable under best reply

MPEC Approach to Games

- Suppose the game has parameters θ.
- Let σ denote the equilibrium strategy given θ; that is, σ is an equilibrium if and only if for some function G

$$G(\theta, \sigma) = 0$$

- Suppose that likelihood of a data set, X, if parameters are θ and players follow strategy σ is $\mathcal{L}(\theta, \sigma, X)$. Therefore, maximum likelihood is the problem

$$\begin{array}{ll} \max\limits_{(\theta,\sigma)} & \mathcal{L}(\theta, \sigma, X) \\ \text{subject to} & G(\theta, \sigma = 0) \end{array}$$

Example: Pricing Game with Multiple Equilibria

- Bertrand pricing game with 3 types of customers
  - Type 1 customers only want good x

$$Dx_1(p_x) = A - p_x;\ Dy_1 = 0$$

  - Type 3 customers only want good y, and have a linear demand curve:

$$Dx_3 = 0;\ Dy_3(p_y) = A - p_y$$

  - Type 2 customers want some of both. Let n be the number of type 2 customers in a city.

$$\begin{aligned} Dx_2(p_x, p_y) &= n p_x^{-\sigma} \left(p_x^{1-\sigma} + p_y^{1-\sigma}\right)^{\frac{\gamma-\sigma}{-1+\sigma}} \\ Dy_2(p_x, p_y) &= n p_y^{-\sigma} \left(p_x^{1-\sigma} + p_y^{1-\sigma}\right)^{\frac{\gamma-\sigma}{-1+\sigma}} \end{aligned}$$

Example: Pricing Game with Multiple Equilibria

- Total demand for good x (y) is

$$
\begin{aligned}
Dx(p_x, p_y) &= Dx_1(p_x, p_y) + Dx_2(p_x, p_y) \\
Dy(p_x, p_y) &= Dy_2(p_x, p_y) + Dy_3(p_x, p_y)
\end{aligned}
$$

- Let m be the unit cost of production for each firm. Profit for good x (y) is

$$
Rx(p_x, p_y) = (p_x - m)Dx(p_x, p_y)
$$
$$
Ry(p_x, p_y) = (p_y - m)Dy(p_x, p_y)
$$

Example: Pricing Game with Multiple Equilibria

- Let MR_x be marginal profits for good x; similarly for MR_y.

$$MR_x(p_x, p_y) = A - p_x + n\left(p_x^{\sigma}\left(p_x^{1-\sigma} + p_y^{1-\sigma}\right)^{\frac{\gamma-\sigma}{\sigma-1}}\right)^{-1}$$

$$+ (p_x - m)\left(-1 + \frac{n_i(\sigma-\gamma)}{p_x^{2\sigma}\left(p_x^{1-\sigma} + p_y^{1-\sigma}\right)^{1+\frac{\sigma-\gamma}{\sigma-1}}} - \frac{n\sigma}{p_x^{1+\sigma}\left(p_x^{1-\sigma} + p_y^{1-\sigma}\right)^{\frac{\sigma-\gamma}{\sigma-1}}}\right)$$

$$MR_y(p_x, p_y) = A - p_y + n\left(p_y^{\sigma}\left(p_x^{1-\sigma} + p_y^{1-\sigma}\right)^{\frac{\gamma-\sigma}{\sigma-1}}\right)^{-1}$$

$$+ (p_y - m)\left(-1 + \frac{n(\sigma-\gamma)}{p_y^{2\sigma}\left(p_x^{1-\sigma} + p_y^{1-\sigma}\right)^{1+\frac{\sigma-\gamma}{\sigma-1}}} - \frac{n\sigma}{p_y^{1+\sigma}\left(p_x^{1-\sigma} + p_y^{1-\sigma}\right)^{\frac{\sigma-\gamma}{\sigma-1}}}\right)$$

Example: Pricing Game with Multiple Equilibria

- The other parameters are common across markets:

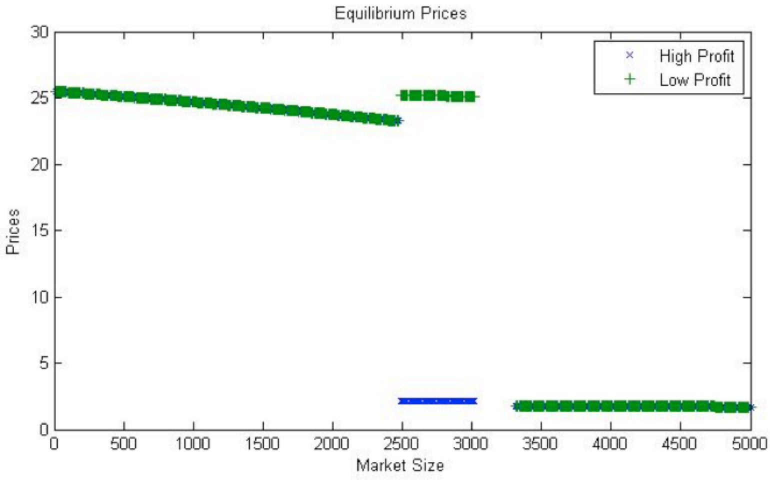
$$\sigma = 3;\ \gamma = 2;\ m = 1;\ A = 50$$

- We solve the FOC

$$MR_x(p_x, p_y) = 0$$
$$MR_y(p_x, p_y) = 0$$

and check the second-order conditions global optimality for each firm in each potential equilibria

Equilibrium Prices for Different Populations



Example: Pricing Game with Multiple Equilibria

- Strategies for each firm
  - Niche strategy: price high, get low elasticity buyers.
  - Mass market strategy: price low to get type 2 people.
- Equilibrium possibilities for each firm
  - Low population implies both do niche
  - Medium population implies one does niche, other does mass market, but both combinations are equilibria.
  - High population implies both go for mass market

Example: Pricing Game with Multiple Equilibria

- Four markets that differ only in terms of type 2 customer population with $(n_1, n_2, n_3, n_4) = (1500, 2500, 3000, 4000)$
- Unique equilibrium for City 1 and City 4:

$$\begin{array}{lrcl} \text{City 1:} & (p_{x1}, p_{y1}) & = & (24.24, 24.24) \\ \text{City 4:} & (p_{x4}, p_{y4}) & = & (1.71, 1.71) \end{array}$$

- Two equilibria in City 2 and City 3:

$$\begin{array}{lrcl} \text{City 2:} & \left(p^I_{x2}, p^I_{y2}\right) & = & (25.18,\ 2.19) \\ & \left(p^{II}_{x2}, p^{II}_{y2}\right) & = & (2.19,\ 25.18) \\ \\ \text{City 3:} & \left(p^I_{x3}, p^I_{y3}\right) & = & (2.15,\ 25.12) \\ & \left(p^{II}_{x3}, p^{II}_{y3}\right) & = & (25.12,\ 2.15) \end{array}$$

Generating Synthetic Data

- Assume that the equilibria in the four city types are

$$\begin{aligned} \left(p_{x1}^{*}, p_{y1}^{*}\right) &= \ (24.24, 24.24) \\ \left(p_{x2}^{*}, p_{y2}^{*}\right) &= \ (25.18,\ 2.19) \\ \left(p_{x3}^{*}, p_{y3}^{*}\right) &= \ (2.15,\ 25.12) \\ \left(p_{x4}^{*}, p_{y4}^{*}\right) &= \ (1.71, 1.71) \end{aligned}$$

- Econometrician observes price data with measurement errors for 4K cities, with K cities of each type
- We used a normally distributed measurement error $\varepsilon \sim N(0, 50)$ to simulate price data for 40,000 cities, with 10,000 cities of each type ($K =$ 10,000)
- We want to estimate the unknown structural parameters (σ, γ, A, m) as well as equilibrium prices $(p_{xi}, p_{yi})_{i=1}^{4}$ implied by the data in all four cities.

Example: Pricing Game with Multiple Equilibria

- MPEC formulation

$$\min_{(p_{xi},p_{yi},\sigma,\gamma,A,m)} \quad \sum_{k=1}^{K}\sum_{i=1}^{4}\left((p_{xi}^k - p_{xi})^2 + (p_{yi}^k - p_{yi})^2\right)$$

subject to: $\quad p_{xi} \geq 0, \quad p_{yi} \geq 0, \ \forall i$

[FOC:] $\quad MR_x(p_{xi}, p_{yi}) = MR_y(p_{xi}, p_{yi}) = 0, \ \forall\, i$

[sampling global opt:] $\quad (p_{xi} - m)Dx(p_{xi}, p_{yi}) \geq (p_j - m)Dx(p_j, p_{yi}), \ \forall\, i, j$

[sampling global opt:] $\quad (p_{yi} - m)Dy(p_{xi}, p_{yi}) \geq (p_j - m)Dy(p_{xi}, p_j), \ \forall\, i, j$

- We do not impose an equilibrium selection criterion

Game Estimation Results

- Case 1: Estimate only σ and γ and fix $A_x = A_y = 50$ and $m_x = m_y = 1$
- Case 2: Estimate all six structural parameters but impose the symmetry constraints on the two firms: $A_x = A_y$ and $m_x = m_y$
- Case 3: Estimated all six structural parameters without imposing the symmetry constraints

	True	Case 1	Case 2	Case 3
(σ, γ)	$(3, 2)$	$(3.01, 2.02)$	$(2.82, 1.99)$	$(3.08, 2.09)$
(A_x, A_y)	$(50, 50)$		$(50.40, 50.40)$	$(50.24, 49.54)$
(m_x, m_y)	$(1, 1)$		$(0.98, 0.98)$	$(1.08, 0.97)$
(p_{x1}, p_{y1})	$(24.24, 24.24)$	$(24.29, 24.29)$	$(24.44, 24.44)$	$(24.69, 24.24)$
(p_{x2}, p_{y2})	$(25.18, 2.19)$	$(25.19, 2.17)$	$(25.25, 2.14)$	$(25.43, 2.00)$
(p_{x3}, p_{y3})	$(2.15, 25.12)$	$(2.13, 25.14)$	$(2.10, 25.16)$	$(2.24, 24.93)$
(p_{x4}, p_{y4})	$(1.71, 1.71)$	$(1.72, 1.72)$	$(1.73, 1.73)$	$(1.81, 1.65)$

Other Applications of MPEC Approach in Estimation

- Vitorino (2007): Estimation of shopping mall entry
  - Standard analyses assume strategic substitutes to make contraction more likely in NFXP, but complementarities are obviously important
  - Vitorino used MPEC for estimation, and did find complementarities
  - Vitorino used bootstrap methods to compute standard errors.
- Chen, Esteban and Shum (2008): Dynamic equilibrium model of durable good oligopoly
- Hubbard and Paarsch (2008): Low-price, sealed-bid auctions
- Dubé, Su and Vitorino (2008): Empirical Pricing Games
- Dynamic demand estimation
- Estimation of dynamic games
- Estimation of multi-bidder multi-unit auctions (with Paarsch) – PDE constrained optimization

Conclusion

- Structural estimation methods are far easier to construct if one uses the structural equations
- The advances in computational methods (SQP, Interior Point, AD, MPEC) with NLP solvers such as KNITRO, SNOPT, filterSQP, PATH, makes this tractable
- User-friendly interfaces (e.g., AMPL, GAMS) makes this as easy to do as Stata, Gauss, and Matlab
- This approach makes structural estimation *really* accessible to a larger set of researchers