Mini Course on Structural Estimation of Static and Dynamic Games

Junichi Suzuki

University of Toronto

June 1st, 2009

Part II: Estimation of Single-Agent Dynamic Optimization Problem

Motivation

- Most economic agents seem forward-looking ($\beta \neq 0$)
  - Households care current consumption as well as future consumptions
  - Firms make huge investment for developing new technology
- Current decisions often affect future state variables
  - If you spend too much today, you have to spend less in the future
  - If a drug company does not invest enough, it cannot attain a profitable profit

Motivation

- Static analysis is irrelevant when these two conditions hold
- Need to pursue a dynamic analysis
- This section looks at the models of single-agent dynamic optimization problem (i.e., no strategic interaction with others)
- Dynamic programming (DP) is a standard tool to analyze this framework (Stokey, Lucas and Prescott)

Goal

- Overview the very basic of single-agent maximization problem using the model of Rust (1989)
- Present difficulty in estimating this model by straightforward conventional methods
- Show how to apply a two-step method/BBL in this framework

Example: Rust's Engine Replacement Model

- Harold Zurcher is a maintenance guy at a bus company in Madison, Wisconsin
- He is responsible for deciding when each bus replaces its engine
- The chance of engine trouble increases as engines accumulates mileage
- Replacing engine incurs significant expense
- Replacing decisions need to take into account the trade-off between current expenditure and future trouble

Example: Rust's Engine Replacement Model

- Static model is NOT appropriate since
  - Harold Zurcher should be forward-looking
  - Current replacement decision affects future profit

- Need to construct a dynamic model

- Consider a simple version of his model

Model: Primitives

- Consider a particular bus
- Observable state variable: x
- Unobservable state variables: $\epsilon = \{\epsilon(0), \epsilon(1)\}$
- Choice variable: i

$$i = \begin{cases} 1 & \text{if replaced} \\ 0 & \text{otherwise} \end{cases}$$

- Notation
  - x: accumulated mileage on the bus engine at the current period since the last replacement
  - ϵ: cost factor observable to Zurcher but not to econometricians
  - i: replacement decision

Model: Period Profit

- Period profit function

$$u(x, i, \theta_1) = \begin{cases} -[RC + c(0, \theta_1)] + \epsilon(1) & \text{if } i = 1 \\ -c(x_t, \theta_1) + \epsilon(0) & \text{if } i = 0 \end{cases}$$

- Operating cost increases as mileage accumulates

$$\frac{\partial c(x, \theta_1)}{\partial x} > 0$$

- Notations
  - RC: net replacement cost
  - $c(0, \theta_1)$: operating cost

Model: Transition

- Additional mileage is a random draw from a certain probability distribution:

$$p\left(x' | x, i, \theta_3\right) = \begin{cases} g\left(x' - x, \theta_3\right) & \text{if } i = 0 \\ g\left(x' - 0, \theta_3\right) & \text{if } i = 1 \end{cases}$$

- This function represents the uncertainty about
  - when this bus is in operation
  - demand shock etc..

Epsilon as a Structural Error Term

- In structural analysis, error terms ϵ has a direct economic interpretation
- ϵ reflects whatever relevant factors Zurcher observes but econometricians do not
- Large values of $\epsilon(0)$ may represent bus driver's positive report on this bus
- Large values of $\epsilon(1)$ may represent engine inventory

Conditional Independence Assumption (CI)

- To make estimation feasible, need to impose the following restriction on the distribution of ϵ

$$p\left(x', \epsilon' | x, \epsilon, i, , \theta_2, \theta_3\right) = q\left(\epsilon' | x', \theta_2\right) p\left(x' | x, i, \theta_3\right)$$

- Rust calls this assumption Conditional Independence Assumption (CI)
- This assumption implies
  - The value of x' is a sufficient statistic to characterize the distribution of ϵ'
  - The value of ϵ affects the value of x' only through investment i
- Type I extreme valued i.i.d ϵ and above transition function satisfies this condition

Model: Value Function

- Value function calculates the sum of expected discounted profits when Zurcher makes the profit-maximizing decision every period:

$$
\begin{aligned}
& V(x, \epsilon, \theta) \\
= & \max_{i \in \{0,1\}} \left[u(x, i, \theta_1) + \epsilon(i, \theta_2) \right. \\
& + \beta \int \int V(x', \epsilon', \theta)\, p(x' | x, i, \theta_3)\, dF(\epsilon' | x', \theta_2) \\
\text{where } \theta = & (\theta_1, \theta_2, \theta_3)
\end{aligned}
$$

Model: Optimal Strategy

There exists an optimal policy $\sigma(x, \epsilon, \theta)$ that maps the state variable to $\{0, 1\}$

$$
\begin{aligned}
& V(x, \epsilon, \theta) \\
= \; & u(x, \sigma(x, \epsilon), \theta_1) + \epsilon(\sigma(x, \epsilon), \theta_2) \\
& + \beta \int \int V(x', \epsilon', \theta)\, p(x' | x, \sigma(x, \epsilon), \theta_3)\, dF(\epsilon' | x', \theta_2)\, dx'
\end{aligned}
$$

Model: Value Function

- Once we know the optimal policy function $\sigma\left(x, \epsilon, \theta\right)$, the chance of engine replacement at this period is

$$\Pr\left(\, i = 1 \middle|\, x\right) = \Pr\left(\sigma\left(x, \epsilon, \theta\right) = 1\right)$$

$$= \frac{\exp\left(u\left(x, 1, \theta_1\right) + \beta \int_0^\infty V\left(x'\right) p\left(x' | x, f\left(x, \theta\right), \theta_3\right) dx'\right)}{\sum_{i' \in \{0,1\}} \exp\left(u\left(x, i', \theta_1\right) + \beta \int_0^\infty V\left(x'\right) p\left(x' | x, i', \theta_3\right) dx\right.}$$

The Data

- Suppose we observe the maintenance record of M different buses $\{x_t^m, i_t^m\}_{m=1}^M$
- For simplicity, assume all M buses are observationaly equal except their mileages
- Want to recover the parameter of θ_1 and θ_3 from the data
- Start with a brute-force method, nested-fixed point algorithm

Step 1: Estimating the Transition Function

- Discretize the space of x into several intervals
- Assume x never decreases without replacing an engine
- Assume an increase in x in one period is no more than two intervals
- These two assumptions imply $x' - x \in \{0, 1, 2\}$
- Using the data, estimate

$$\begin{cases} \theta_{30} = \Pr\left(x' - x = 0 | i = 0\right) \\ \theta_{31} = \Pr\left(x' - x = 1 i = 0\right) \end{cases}$$

- This estimation can be done independently, thanks to CI assumption

Step 2: Evaluating the Likelihood

- Let the value θ_1 as given
- For a given set of parameters, find the optimal policy $f(x, \epsilon, \theta)$ and $V(x, \epsilon)$
  - This step requires solving DP numerically
  - $V(x, \epsilon)$ is often approximated by Chebyshev polynomials
  - See Judd (1989) for its implementation
- Calculate the likelihood of the observed event $L_{it}(\theta)$
- Take log and summing up

$$
\begin{aligned}
\ln L(\theta) &= \sum_i \sum_t \ln L_{it} \\
&= \sum_i \sum_t [1(i_{it} = 0) \ln \Pr(i_{it} = 0|x) \\
&\quad +1(i_{it} = 1) \ln \Pr(i_{it} = 1|x)]
\end{aligned}
$$

Step 3: Maximizing the Likelihood

- Find a set of parameters that maximizes $\ln L(\theta)$
- This algorithm is straightforward but requires heavy computation
- Need to solve the dynamic programming for every set of parameters evaluated
- Estimation can be very slow
- Again, the two-step methods is very useful

Estimating the Policy Function

- Both BBL and NPL require estimating the reduced-form policy function of Zurcher
- Ideally, nonparametric methods are appealing
- In practice, flexible logit are often used

$$
\begin{aligned}
i^* &= \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3 + \upsilon \\
i &= \begin{cases} 0 & \text{if } i^* < 0 \\ 1 & \text{if } i^* \geq 0 \end{cases}
\end{aligned}
$$

- Used to approximate his observed/optimal behavior

Estimation by BBL

- BBL estimates the structural parameters without solving DP even once
- For estimating the parameter, the following optimization condition is exploited:

$$V\left(x,\sigma\right)\geq V\left(x,\tilde{\sigma}\right)\;\text{ for all } x$$

- Generate fake policies $\{\sigma'(x)\}$ that are slightly different from the observed one $\hat{\sigma}(x)$
- Find a set of parameters that let $\sigma(x)$ beat as many as $\hat{\sigma}(x)$ possible

- BBL use the data only to estimate policy function $\hat{\sigma}(x)$

Estimation by BBL

- The basic idea is to transform a dynamic discrete choice problem to the conventional (static) discrete choice problem
- Approximate $EV(x)$ by implementing the forward simulation
- Find a set of parameters that rationalizes the observed policy

Implementing BBL Step by Step

- Generate fake policies $\{\tilde{\sigma}^m(x)\}_{m=1}^{NI1}$
- Pick several initial values $\{x_0^n\}_{n=1}^{NI2}$
- Calculate $EV(x_0, \sigma; \theta)$ by forward simulation
- Calculate $EV(x_0, \tilde{\sigma}; \theta)$ by forward simulation
- Find θ^* that solves

$$\min_{\theta} \sum_{m=1}^{NI1} \sum_{n=1}^{NI2} \left(\min\left\{EV\left(x_0^n, \sigma; \theta\right) - EV\left(x_0^n, \tilde{\sigma}^m; \theta\right), 0\right\}\right)^2$$

Implementing Forward Simulation

- Pick T so that β^T is sufficiently small
- By using $\hat{p}(x'|x)$, $\sigma(x)$ and $F(\epsilon)$, simulate a stream of his period profit for T periods
- Calculate $\sum_{t=1}^{T} \beta^t u(x_t, \sigma(x_t, \epsilon_t), \theta_1)$
- Iterating this process many times, calculate $EV(x_0^n, \sigma; \theta) = E\left[\sum_{t=1}^{T} \beta^t u(x_t, \sigma(x_t, \epsilon_t), \theta_1)\right]$

Summary

- Go over a single-agent dynamic optimization problem by using Rust (1989)
- Nested-fixed point is straightforward but its computational burden can be prohibitive
- Two-step methods are very useful to obtain consistent estimates by maintaining computational burden to practical level