Lecture 5: BLP and Dynamic Discrete Choice (Rust)

Antoine Chapel

Introduction

- Two econometrics models, which have an algorithmic structure in common
- BLP: a method for demand estimation that allows for random coefficients
- Rust: introduces the dynamic discrete choice framework
- We will briefly present BLP and fully replicate Rust

BLP: A VERY SHORT INTRODUCTION

$$u_{ij} = x_j' \beta_i - \alpha p_j + \xi_j + \epsilon_{ij}$$

- x_j is a K-dimensional vector of characteristics specific to alternative j
- p_j is the price of alternative j
- ξ_j is the unobserved heterogeneity specific to characteristic j
- ϵ_{ij} is the Gumbel econometric error

$$x_j'\beta_i = x_j\overline{\beta} + \sum_{k=1}^{K}\sigma_k$$

- There are K characteristics
- σ_k is the systematic weight given to characteristic k
- $\nu_{ik} \sim \mathcal{N}(0,1)$ is individual i specific taste for characteristic k: drives randomness in β_i
- We want to estimate $\overline{\beta}_k$ and σ_k

$$\begin{aligned} u_{ij} &= x_j\overline{\beta} - \alpha p_j + \xi_j + \sum_{k=1}^{K}\sigma_k x_{jk}\nu_{ik} + \epsilon_{ij} \\ &= \delta_j + \sum_{k=1}^{K}\sigma_k x_{jk}\nu_{ik} + \epsilon_{ij} \\ &= \delta_j + \mu_{ij} \end{aligned}$$

BLP

Let us use the result just derived and the multinomial logit formula to derive the choice probability

$$P_{ij|\nu_i} = \frac{\exp(\delta_j + \sum_{k=1}^{K} \sigma_k x_{jk} \nu_{ik})}{1 + \sum_{h=1}^{J} \exp(\delta_h + \sum_{k=1}^{K} \sigma_k x_{hk} \nu_{ik})}$$

Integrating the uncertainty coming from individual ν_i

$$s_j = \int P_{ij|\nu_i} f_\nu(\nu_i) d\nu_i$$

In the estimation procedure, these market shares will be simulated, by randomly drawing ν_i and evaluating the integral.

BLP

- To avoid spoiling m+e+c, I only give a sketch of the algorithm, which will be revisited using Optimal Transport.
- The algorithm applies an IV GMM method, for which the moment condition is $E[\xi' Z] = 0$ (error must be uncorrelated with the instruments).
- The algorithm has a nested loop procedure:
  1. Given β^j, the inner loop returns a GMM objective: $\left(\frac{\sum_{j,t} z'_{jt}\hat{\xi}_{jt}(\theta)}{JT}\right)' W \left(\frac{\sum_{j,t} z'_{jt}\hat{\xi}_{jt}(\theta)}{JT}\right)$
  2. The outer loop minimizes that GMM objective, using a convex optimization method
  3. More detail on the notebook

BLP

- The algorithm is a bit technical to code by hand
- The pyblp library automates everything
- You can see it in action on the notebook, where we implemented the version of the pyblp tutorial that corresponds to the simple blp case presented earlier

Dynamic Discrete Choice

- Introduction: dynamic discrete choice
- From generality to tractability
- The Bellman equation for dynamic discrete choice:

$$V_\theta(x,\epsilon,d) = \max_{d\in D(x)} \left[u(x,d,\theta) + \epsilon(d) + \beta \int_{x'} \int_{\epsilon'} V_\theta(x,\epsilon) \cdot \pi(x',\epsilon'|x,\epsilon,\theta) dx' d\epsilon' \right]$$

$$V_\theta(x,\epsilon,d) = \max_{d\in D(x)} \left[u(x,d,\theta) + \epsilon(d) + \beta E[V_\theta(x',\epsilon')] \right]$$

Two main assumptions:

1. $\epsilon \sim_{iid}$ T1EV (Gumbel)
2. Separability: $\pi(x'\epsilon'|x,e,\theta) = p(x'|x,d,\theta) \cdot q(\epsilon'|x,\theta)$

Dynamic Discrete Choice

A very useful rule: if there are j alternatives that provide systematic utility U_j and a Gumbel random shock ϵ_j:

$$E\Big[\max\{U_j+\epsilon_j\}\Big] = \gamma + \log\sum_j e^{U_j} \qquad \text{Where } \gamma \approx 0.57$$

So:

$$\begin{aligned}
&E\Big[V_\theta(x)\Big] \\
=&E\Big[\max_{d\in D(x)} u(x,d,\theta)+\epsilon+\beta E[V_\theta(x')]\Big] \\
=&\int_x \Big[\gamma + \log\big[\sum_{d\in D(x)} \exp\{u(x,d,\theta)+\beta E[V_\theta(x')\}\big]\Big] p(x|d,\theta)dx
\end{aligned}$$

Dynamic Discrete Choice

Let's go back to our Bellman equation:

$$\begin{aligned} V_\theta(x) &= \max_{d \in D(x)} \Big[u(x,d,\theta) + \epsilon + \beta E[V_\theta(x')] \Big] \\ &= \max_{d \in D(x)} \Big[u(x,d,\theta) + \epsilon + \beta \int_{x'} \Big[\gamma + \log \big[\sum_{d \in D(x')} \exp\{u(x',d,\theta) \\ &\qquad + \beta E[V_\theta(x'',d)]\}\big] \Big] p(x'|d,\theta) dx' \end{aligned}$$

Since ϵ is Gumbel distributed: guess who's back: the multinomial logit formula !

$$P(d|x,\theta) = \frac{\exp\{u(x,d,\theta) + \beta E[V_\theta(x',d)]\}}{\sum_{k \in D(x)} \exp\{u(x',k,\theta) + \beta E[V_\theta(x',k)]\}}$$

Dynamic Discrete Choice estimation

1. Inner loop: does value function iteration to predict a policy function given θ, and return the likelihood function
2. Outer loop: maximizes the log-likelihood function:

$$\mathcal{L}(\theta) = \sum_t \sum_i \sum_j d_{ijt} \log \left(P(d_{ijt} | x_j, \theta) \right)$$

- Where d_{ijt} is a dummy that denotes the observed choice of individual i to choose alternative j at period t
- $P()$ denotes our predicted probability, given our guess of the value function, for i to make this observed choice
- $\hat{\theta} = \arg\max_\theta \mathcal{L}(\theta)$

Rust model

- You handle a fleet of buses
- Every period, you can replace the bus engine and incur $RC = \theta_1$
- Or you can pay day-to-day reparations cost $c(x, \theta_0) = x\theta_0$
- x is the state, and denotes the bus's mileage

$$u(x, d, \theta) = \begin{cases} -\theta_1 + \epsilon & \text{if } d = 1 \text{ (replace)} \\ -x\theta_0 + \epsilon & \text{if } d = 0 \text{ (not replace)} \end{cases}$$

- Bus mileage is reinitialized if $d = 1$ and increases otherwise

$$p(x'|x, d, \theta) = \begin{cases} g(x' - 0|\theta) & \text{if } d = 1 \\ g(x' - x|\theta) & \text{if } d = 0 \end{cases}$$

Estimation

1. First step: estimate the transition probability. We provide the transition matrix directly in the notebook
2. Second step: code the value function iteration algorithm
3. Third step: compute the conditional choice probabilities
4. Fourth step: Build a likelihood function that sums over the probability for a bus in one given state to take decision j
5. Fifth step: maximize the likelihood function for θ

Let's take this to Python