Empirical IO: Problem Set 4

Due date: Nov 30, 2020

Your answers should be produced in LaTeX, or Jupyter, and should include all relevant graphs and code. Code should be in the appropriate verbatim environment and properly documented. My python code is in the GitHub folder and should be more than enough if you are stuck.

The data are `rust_data_2020.csv`. The fields are `bus_id`, `period_id`, `y_it` (replacement decision) and `x_it` (mileage state).

Part 0 shouldn't take more than 5 minutes. Part 3 is the second easiest. Part 1 and 2 are challenging.

Part 0: Pre-Estimation

1. Compute $P(x'|x, y = 0) = Pr(\Delta x_{it}|y_{it} = 0)$ in a separate first stage – **you should have 5 of them**.
2. Populate a transition probability matrix with states of $x_{it} \in \{0, \ldots, 200\}$.
   (Adjust your matrix so that rows sum to one!)

Part 1: Estimation MLE

Estimate the model using the NFXP approach of Rust. Computers are fast enough you can probably get away with a finite differences for the gradient, but analytic gradients will be more reliable in addition to faster. If you are feeling ambitious `Jax` can do the gradient for you.

1. Compute $EV(x, \theta)$ for a given guess of the parameters via the fixed point.
2. Construct the CCP given your $EV(x, \theta)$ for every possible mileage state.
3. Construct the likelihood and its gradient contribution with respect to θ for every possible mileage state.
4. Finally, apply your likelihood and gradient contributions to observed data.

Part 2: Estimation MPEC

Estimate the model using the MPEC method of Su and Judd. If you are feeling ambitious, you can try a modeling language like `pyMC` or `JuMP` or `AMPL` or `GAMS`. (I've solved this in `AMPL` but have no idea if other approaches work).

1. You will probably want to work out the analytic Jacobian where EV_θ is a free parameter.
2. Compare the results in a table, including the nonparametric answers below and discuss the results.
3. Plot the $EV(\cdot)$ you have obtained for both estimators.

Part 3: The Stata Estimator

This is taken from Han Hong's problem set at Stanford, the idea is that we can use the arguments in Hotz-Miller (1993), or Pesendorfer Schmidt-Dengler (2008) to construct an optimization free method to recover the utility parameters in the Rust problem.

We began by defining the choice specific value function with ϵ_{it} i.i.d. and EV.

$$\begin{aligned} v(x,d) &= u(x,d) + \beta \int \log\left(\sum_{d' \in D} \exp(v(x',d'))\right) p(x'|x,d)dx' \\ v(x,d) &= u(x,d) + \beta \int \log\left(\sum_{d' \in D} \exp(v(x',d') - v(x',1))\right) p(x'|x,d)dx' + \beta \int v(x,1)p(x'|x,1)dx' \end{aligned}$$

1. Estimate $p(x'|x,d)$ non parametrically or parametrically (for example as a set of multinomial with n outcomes or an exponential distribution). Call your estimate $\hat{p}(x'|x,d)$.
2. Estimate $p(d|x)$ (the CCP) non-parametrically. You can use the binomial logit model with a basis function (increasing number of terms) or you can use a kernel. (Extra Credit: Impose monotonicty on CCP's).
3. Now use the Hotz-Miller inversion to estimate: $\hat{v}(x,d) - \hat{v}(x,1) = \log \hat{p}(d|x) - \log \hat{p}(1|x)$
4. Normalize $u(x,1) = 0$ and so for $= 1$ we have that

$$\begin{aligned} v(x,1) &= \beta \int v(x',1)p(x'|x,1)dx' + \beta \int \log\left(\sum_{d' \in D} \exp(\hat{v}(x',d') - \hat{v}(x',1))\right) \hat{p}(x'|x,1)dx' \\ &= \beta \int v(x',1)p(x'|x,1)dx' - \beta \int \log\left(\hat{p}(1|x')\right) \hat{p}(x'|x,1)dx' \end{aligned}$$

   This defines a fixed point that we can iterate on to obtain a nonparametric estimate of $\hat{v}(x,1)$. Add this to $\hat{v}(x,d) - \hat{v}(x,1)$ to recover the choice specific value functions for $d = 1, \ldots, D$.
5. Once we know $\hat{v}(x,d)$ for all $d \in D$ we can recover the nonparametric estimate of $u(x,d)$ for $d \geq 2$ by

$$\hat{u}(x,d) = \hat{v}(x,d) - \beta \int \log\left(\exp(\hat{v}(x',d'))\right) \hat{p}(x'|x,d)dx'$$

This estimator should be very simple to implement (and only requires one fixed point) so we could do inference via the bootstrap if we wanted to.