Exercise Set 4

Estimating John Rust (1987) using MPEC

Dynamic Programming Spring 2020,

by Patrick Kofod Mogensen and Maria Juul Hansen

Introduction

This week we will revisit the model form last time. We will try to estimate the parameters using two MPEC as proposed in Su and Judd (2012).

MPEC

As mentioned in John Rust (2000, p. 18), it would seem obvious to solve the problem as a constrained maximization problem

$$\max_{\theta, EV} L(\theta, EV) \text{ subject to } EV = T_\theta(EV).$$

However, this is potentially a quite large problem. To estimate the model, we need to find a θ vector, and at the same time we have to find a function EV satisfying our constraint. Now, the function is actually a vector on our computer, so with N bins, we could think of it as maxizing L with respect to θ and $EV_1, EV_2, \ldots, EV_N$ such that the N constraints $EV_1 = T_\theta(EV_1), EV_2 = T_\theta(EV_2), \ldots, EV_N = T_\theta(EV_N)$ hold. This is potentially a daunting task, since the dimensionality of the problem is huge. However, as Su and Judd (2012) demonstrated, this might not be a problem using modern computers, and efficient interior point solvers for constrained optimization. An interior point method has the advantage that it searches for a solution to the problem in the interior of the constraint set, such that we don't have to strictly worry about the constraints in each iteration. They use both Matlab+kntrlink, and AMPL+KNITRO, but in this class we will use Python and scipy.optmize. Be aware that scipy.optimize is in no way as fast as commercial constrained optimization software, but for our purposes it works fine.

Questions

1. Run the function mpec.sparsity_pattern. The function mpec.sparsity_pattern creates sparse matrices of indicators for where there are elements in the Jacobian of the constraints and Hessian of the likelihood function

   (a) Look at the figures, and talk about what the different elements of the Jacobian of the constraint and Hessian of the likelihood represent. Hints:

i. What are the constraints?
ii. What is the dimension of the likelihood?
iii. What are the dimensions of the Jacobian and Hessian?

2. What is the advantages of handle that the Jacobian and Hessian as sparse matrices?

3. Estimate the model using MPEC. In order to estimate the model, you should understand: mpec.estimate, mpec.ll (don't spend too much time on understanding the gradient and Hessian) and mpec.con_bellman (don't focus too much on computing Jacobian). Note that the implementation does not use the information that the Hessian is sparse

4. Fill in the missing stuff in mpec.ll and mpec.con_bellman, and run the code to check that your results are correct

5. Compare NFXP and MPEC
   (a) Compare the time of NFXP and the time of MPEC, and the time of NFXP and MPEC from the lecture.
   (b) Do we use analytical first-order derivatives?
   (c) What about second-order derivatives?
   (d) What do they do in Su and Judd (2012)?
   (e) Why is our implementation inefficient?

6. How did we get our standard errors using NFXP? How would you calculate them using MPEC?