Mergers and Counterfactual Prices

Chris Conlon

September 24, 2020

Grad IO

Merger Simulation: Two Options

- Partial Merger Simulation
  - Simulate a new price for p_j after acquiring good k holding the prices of all other goods (p_k, p_{-j}) fixed.
  - Repeat for p_k and all other products involved in the merger.
  - Compare price increases to synergies or cost savings.
- Full Merger Simulation
  - Write down the full system of post-merger FOC.
  - Adjust post-merger marginal costs for potential synergies.
  - Solve for all prices at the new (post-merger) equilibrium (p_j, p_k, p_{-j}).

Differentiated Products Bertrand

Recall the multi-product Bertrand FOCs:

$$\arg\max_{p \in \mathcal{J}_f} \pi_f(\mathbf{p}) = \sum_{j \in \mathcal{J}_f} (p_j - c_j) \cdot q_j(\mathbf{p})$$

$$\rightarrow 0 = q_j(\mathbf{p}) + \sum_{k \in \mathcal{J}_f} (p_k - c_k) \frac{\partial q_k}{\partial p_j}(\mathbf{p})$$

It is helpful to define the matrix Δ with entries:

$$\Delta_{(j,k)}(\mathbf{p}) = \left\{ \begin{array}{cc} -\frac{\partial q_j}{\partial p_k}(\mathbf{p}) & \text{for } (j,k) \in \mathcal{J}_f \\ 0 & \text{for } (j,k) \notin \mathcal{J}_f \end{array} \right\}$$

We can re-write the FOC in matrix form:

$$q(\mathbf{p}) = \Delta(\mathbf{p}) \cdot (\mathbf{p} - \mathbf{mc})$$

Merger Simulation

What does a merger do? change the ownership matrix.

- Step 1: Recover marginal costs $\widehat{\mathbf{mc}} = \mathbf{p} + \Delta(\mathbf{p})^{-1} q(\mathbf{p})$.
- Step 1a: (Possibly) adjust marginal cost $\widehat{\mathbf{mc}} \cdot (1-e)$ with some cost efficiency e.
- Step 2: Change the ownership matrix $\Delta^{pre}(\mathbf{p}) \rightarrow \Delta^{post}(\mathbf{p})$.
- Step 3: Solve for $\mathbf{p}^{post}$ via: $\mathbf{p} = \widehat{\mathbf{mc}} - \Delta(\mathbf{p})^{-1} q(\mathbf{p})$.

Merger Simulation

What does a merger do? change the ownership matrix.

- Step 1: Recover marginal costs $\widehat{\mathbf{mc}} = \mathbf{p} + \Delta(\mathbf{p})^{-1} q(\mathbf{p})$.
- Step 1a: (Possibly) adjust marginal cost $\widehat{\mathbf{mc}} \cdot (1-e)$ with some cost efficiency e.
- Step 2: Change the ownership matrix $\Delta^{pre}(\mathbf{p}) \rightarrow \Delta^{post}(\mathbf{p})$.
- Step 3: Solve for $\mathbf{p}^{post}$ via: $\mathbf{p} = \widehat{\mathbf{mc}} - \Delta(\mathbf{p})^{-1} q(\mathbf{p})$.

- The first step is easy (just a matrix inverse).
- The second step is trivial.
- The third step is tricky because we have to solve an implicit system of equations. $\mathbf{p}$ is on both sides.

Partial Merger Analysis

- Hold all other prices p_{-j} fixed at pre-merger prices.
- Adjust the marginal costs for potential efficiencies.
- Consider only the FOC for product j

$$0 = q_j(\mathbf{p}) + \sum_{k \in \mathcal{J}_f} (p_k - c_k) \frac{\partial q_k}{\partial p_j}(\mathbf{p})$$

- Solve for the new p_j given the change in the products controlled by firm f: $\mathcal{J}_f \to \mathcal{J}'_f$
- This is a single Gauss-Jacobi step (only products involved in merger).

Partial Merger Analysis: Why bother?

- We only need own and cross elasticities for products involved in the merger.
- Tends to show smaller price increases than full equilibrium merger analysis.
- Only solving a single equation rather than a system of J nonlinear equations.

Solution Methods

How do we solve: $\mathbf{p} = \widehat{\mathbf{mc}} - \Delta(\mathbf{p})^{-1} q(\mathbf{p})$?

1. Gauss Jacobi: Simultaneous Best Reply $p_j^{k+1}(\mathbf{p^k_{-j}})$.
2. Gauss Seidel: Iterated Best Response $p_j^{k+1}(\mathbf{p^{k+1}_{<j}}, \mathbf{p^k_{>j}})$.
3. Newton's Method: Set $\mathbf{p} - \widehat{\mathbf{mc}} + \Delta(\mathbf{p})^{-1} q(\mathbf{p}) = 0$
   but requires derivatives of $\Delta(\mathbf{p})^{-1} q(\mathbf{p})$
4. Fixed point iteration: $\mathbf{p} \leftarrow \widehat{\mathbf{mc}} - \Delta(\mathbf{p})^{-1} q(\mathbf{p})$
   - Turns out this is not a contraction.
   - But you can get lucky... $\mathbf{p} - \widehat{\mathbf{mc}} + \Delta(\mathbf{p})^{-1} q(\mathbf{p}) = 0$ means you have satisfied FOC's
5. Alternative fixed point.

Exploit the logit formula

For the logit the Δ matrix (for a single market) looks like:

$$\Delta_{(j,k)}(\mathbf{p}) = \left\{ \begin{array}{cc} \int \alpha_i \cdot s_{ij} \cdot (1 - s_{ij})\, \partial F_i & \text{if } j = k \\ -\int \alpha_i \cdot s_{ij} \cdot s_{ik}\, \partial F_i & \text{if } j \neq k \end{array} \right\}$$

Which we can factor into two parts (for plain logit):

$$\Delta(\mathbf{p}) = \underbrace{\text{Diag}\left[\alpha\, \mathbf{s}(\mathbf{p})\right]}_{\Lambda(\mathbf{p})} - \underbrace{\alpha \cdot \mathbf{s}(\mathbf{p})\mathbf{s}(\mathbf{p})'}_{\Gamma(\mathbf{p})}$$

$\Gamma(\mathbf{p})$ and $\Lambda(\mathbf{p})$ are $J \times J$ matrices and $\Lambda(\mathbf{p})$ is diagonal and (j, k) is nonzero in $\Gamma(\mathbf{p})$ only if (j, k) share an owner.

Morrow Skerlos (2010) Fixed Point

- After factoring we can rescale by $\Lambda^{-1}(\mathbf{p})$

$$(\mathbf{p} - \mathbf{mc}) \leftarrow \Lambda^{-1}(\mathbf{p}) \cdot \Gamma(\mathbf{p}) \cdot (\mathbf{p} - \mathbf{mc}) - \Lambda^{-1}(\mathbf{p}) \cdot s(\mathbf{p})$$

- This alternative fixed point is in fact a contraction.
- Moreover the rate of convergence is generally fast and stable (much more than Gauss-Seidel or Gauss-Jacobi).
- Honestly, this is the best way to solve large pricing games. It nearly always wins and doesn't require derivatives.
- Coincidentally, this is what `PyBLP` defaults to.