Dynamic Inefficiencies of Vertical Separation

Evidence from the Soft Drinks Industry

Màxim Sandiumenge-i-Boy
Toulouse School of Economics

March 13, 2025

Motivation

- Increased attention to Vertical Mergers due to
  - Can solve key inefficiencies
  - Can also lead to increased market power
  - Concerning trade-off due to rise of market power and emergence of disruptive innovations
  - Concerns that industry dynamics can exacerbate effects of lenient policy

Motivation

- Increased attention to Vertical Mergers due to
  - Can solve key inefficiencies
  - Can also lead to increased market power
  - Concerning trade-off due to rise of market power and emergence of disruptive innovations
  - Concerns that industry dynamics can exacerbate effects of lenient policy

- Inefficiencies of Vertical Separation
  - Static: Double Marginalization
  - In the presence of dynamics (**brand loyalty**), intertemporal linkages
    - **Magnify consequences** of a given above-cost input price by distorting incentives to invest
    - **Discipline** firms: adjust prices to preserve future demand and their ability to compete

Research Question and Contribution

Research question: How do industry dynamics affect the magnitude of coordination problems and pass-through after integration?

Research Question and Contribution

Research question: How do industry dynamics affect the magnitude of coordination problems and pass-through after integration?

This paper:

- Provides motivating evidence that habit formation in demand affects pricing and merger outcomes
- Quantifies the
  - Magnitude of vertical separation inefficiencies in dynamic context
  - Welfare consequences of mergers
  - For different market structures and level of intertemporal linkages

In this presentation:

- Overview of the industry and data
- Estimation of state dependence in demand at the regional level
- Structural model of vertical relations with state dependence in demand
- Estimation of structural model

Empirical Application: The Soft Drinks Industry

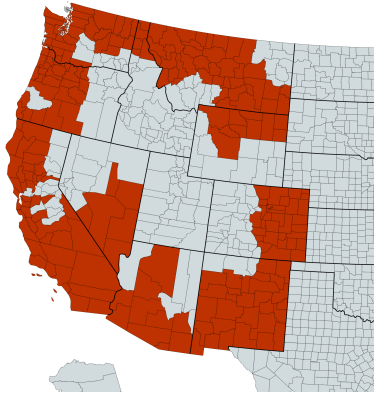
- **Firms**
  - Upstream: concentrate manufacturers (e.g., Coca-Cola, Pepsi)
  - Downstream: Bottlers

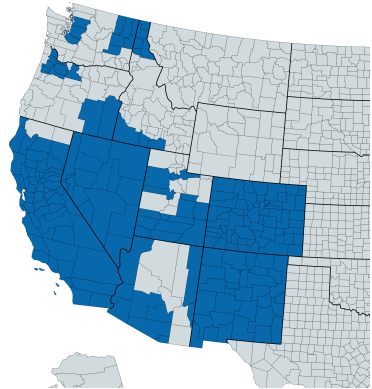
- **Industry**
  - Contracts: stationary long-term linear prices with exclusive territories
  - Source of dynamics: habit formation in demand (e.g., brand loyalty)
  - Vertical Mergers during 2009-2010

Empirical Application: The Soft Drinks Industry

- **Firms**
  - Upstream: concentrate manufacturers (e.g., Coca-Cola, Pepsi)
  - Downstream: Bottlers
- **Industry**
  - Contracts: stationary long-term linear prices with exclusive territories
  - Source of dynamics: habit formation in demand (e.g., brand loyalty)
  - Vertical Mergers during 2009-2010
- **Data**
  - NielsenIQ - Kilts Datasets
    - Retail data: sales, characteristics, store advertisement
    - Scanner data: history of purchases, individual characteristics
  - Territory maps of the U.S. bottling system (Luco and Marshall, 2020)
  - Documents FTC's investigation of the mergers

Territory Maps for Brands that Merged





Color areas: bottlers integrated.

Gray areas: bottlers remained independent.

Overview Dynamic Model of Vertical Relations

- Setting
  - Sets $\mathcal{N}^U$ and $\mathcal{N}^D$ of upstream and downstream firms
  - Market structure and vertical relations are exogenously given
  - Mutually exclusive relations and perpetual
  - Intertemporal linkages: habit formation in demand

Overview Dynamic Model of Vertical Relations

- Setting
  - Sets $\mathcal{N}^U$ and $\mathcal{N}^D$ of upstream and downstream firms
  - Market structure and vertical relations are exogenously given
  - Mutually exclusive relations and perpetual
  - Intertemporal linkages: habit formation in demand
- Upstream prices
  - Before industry starts, firms set margins simultaneously
  - Firms maximize (induced) discounted flow of profits
  - Vector of marginal costs upstream $\boldsymbol{c^U}$

Overview Dynamic Model of Vertical Relations

- Setting
  - Sets $\mathcal{N}^U$ and $\mathcal{N}^D$ of upstream and downstream firms
  - Market structure and vertical relations are exogenously given
  - Mutually exclusive relations and perpetual
  - Intertemporal linkages: habit formation in demand
- Upstream prices
  - Before industry starts, firms set margins simultaneously
  - Firms maximize (induced) discounted flow of profits
  - Vector of marginal costs upstream $\boldsymbol{c^U}$
- Downstream competition
  - Firms compete setting prices each period
  - Vector of marginal costs upstream $\boldsymbol{c^D}$

Model of downstream competition

- Setting
  - Time is discrete and infinite horizon: $t \in \{0, 1, \ldots\}$
  - Firm $i \in \{1, \ldots, N^D\}$ with constant component of marginal cost $\tilde{c}_i$ (common knowledge) makes pricing decisions at each period
  - Industry-wide state s_t: last period's vector of market shares
  - **Privately observed** iid shocks to the marginal cost of production: $\nu_{it} \sim \mathcal{N}(0, \sigma^2_{\nu_i})$

Model of downstream competition

- Setting
  - Time is discrete and infinite horizon: $t \in \{0, 1, \ldots\}$
  - Firm $i \in \{1, \ldots, N^D\}$ with constant component of marginal cost $\tilde{c}_i$ (common knowledge) makes pricing decisions at each period
  - Industry-wide state $\boldsymbol{s}_t$: last period's vector of market shares
  - **Privately observed** iid shocks to the marginal cost of production: $\nu_{it} \sim \mathcal{N}(0, \sigma^2_{\nu_i})$
- Timing
  1. Given $\boldsymbol{s}_t$, each firm i receives a shock to the marginal cost of production ν_{it}
  2. Firms make pricing decisions simultaneously: $\boldsymbol{p}_t = (p_{1t}, ..., p_{Nt}) \in \mathbb{R}^N$
  3. Demands for all firms are realized. Firms receive period payoffs: $\pi_i(\boldsymbol{p}_t, \boldsymbol{s}_t, \nu_{it})$
  4. Industry state evolves according the the law of motion: $\boldsymbol{s}_{t+1} = D(\boldsymbol{p}_t, \boldsymbol{s}_t; \theta^D)$

Firms' Dynamic Problem

- Restrict attention to Markov Perfect Equilibria
- Let the profile of Markov strategies be denoted by: $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_N)$
- The value function of a firm, given strategy profile of other firms, can be written recursively as

$$V_i(\boldsymbol{s}, \nu_i; \boldsymbol{\sigma}) \;=\; \max_{p} \Big[\; \pi_i\big(p, \boldsymbol{\sigma}_{-i}(\mathbf{s}, \boldsymbol{\nu}), \mathbf{s}, \nu_i\big) \;+\; \beta\, \mathbb{E}\left[V_i(\mathbf{s}', \nu_i'; \boldsymbol{\sigma}) \right] \Big],$$

Firms' Dynamic Problem

- Restrict attention to Markov Perfect Equilibria
- Let the profile of Markov strategies be denoted by: $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_N)$
- The value function of a firm, given strategy profile of other firms, can be written recursively as

$$V_i(\boldsymbol{s}, \nu_i; \boldsymbol{\sigma}) \;=\; \max_{p} \Big[\; \pi_i\big(p, \boldsymbol{\sigma}_{-i}(\mathbf{s}, \boldsymbol{\nu}), \mathbf{s}, \nu_i\big) \;+\; \beta\, \mathbb{E}\left[V_i(\mathbf{s}', \nu_i'; \boldsymbol{\sigma}) \right] \Big],$$

where

- Law of motion of state variable: $\boldsymbol{s}_{t+1} = D(\boldsymbol{p}_t, \boldsymbol{s}_t; \theta^D)$
- Period profit: $\pi_i(\boldsymbol{p}_t, \boldsymbol{s}_t, \nu_{it}) \;=\; \left[p_{it} - (\tilde{c}_i + \nu_{it}) \right] D\left(\boldsymbol{p}_t, \boldsymbol{s}_t; \theta^D\right)$

Upstream prices

- Upstream firms $i \in \{1, \ldots, N^U\}$ make simultaneous TIOLI offers before downstream competition starts
- Firm i sets (stationary) margin m to maximize

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \, m \, D(\boldsymbol{\sigma}(\boldsymbol{s}_t, \boldsymbol{\nu}_t), \boldsymbol{s}_t; \theta^D) \,\middle|\, s_0;\, \theta^S\right]$$

Upstream prices

- Upstream firms $i \in \{1, \ldots, N^U\}$ make simultaneous TIOLI offers before downstream competition starts
- Firm i sets (stationary) margin m to maximize

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \, m \, D(\boldsymbol{\sigma}(\boldsymbol{s}_t, \boldsymbol{\nu}_t), \boldsymbol{s}_t; \theta^D) \,\middle|\, s_0;\, \theta^S\right]$$

- For each downstream firm $i \in \mathcal{N}^D$ supplied by $j \in \mathcal{N}^U$, the constant component of i's marginal cost is

$$\tilde{c}_i = c_j^U + c_i^D + m_j,$$

- where $c_j^U + c_i^D = c_{ji}^{U+D}$ represents the fixed production cost of that vertical structure.

Upstream prices

- Upstream firms $i \in \{1, \ldots, N^U\}$ make simultaneous TIOLI offers before downstream competition starts
- Firm i sets (stationary) margin m to maximize

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \, m \, D(\boldsymbol{\sigma}(\boldsymbol{s}_t, \boldsymbol{\nu}_t), \boldsymbol{s}_t; \theta^D) \,\middle|\, s_0;\, \theta^S \right]$$

- For each downstream firm $i \in \mathcal{N}^D$ supplied by $j \in \mathcal{N}^U$, the constant component of i's marginal cost is

$$\tilde{c}_i = c_j^U + c_i^D + m_j,$$

- where $c_j^U + c_i^D = c_{ji}^{U+D}$ represents the fixed production cost of that vertical structure.
- Let $\mu\left(\boldsymbol{c}^{\boldsymbol{U+D}}, \boldsymbol{\sigma_\nu}\right)$ be the equilibrium upstream margins.

Upstream prices

- Upstream firms $i \in \{1, \ldots, N^U\}$ make simultaneous TIOLI offers before downstream competition starts
- Firm i sets (stationary) margin m to maximize

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \beta^t \, m \, D(\boldsymbol{\sigma}(\boldsymbol{s}_t, \boldsymbol{\nu}_t), \boldsymbol{s}_t; \theta^D) \,\middle|\, s_0;\, \theta^S \right]$$

- For each downstream firm $i \in \mathcal{N}^D$ supplied by $j \in \mathcal{N}^U$, the constant component of i's marginal cost is

$$\tilde{c}_i = c_j^U + c_i^D + m_j,$$

- where $c_j^U + c_i^D = c_{ji}^{U+D}$ represents the fixed production cost of that vertical structure.
- Let $\mu\left(\boldsymbol{c}^{\boldsymbol{U+D}}, \boldsymbol{\sigma_\nu}\right)$ be the equilibrium upstream margins.
- Model parameters: $\theta = (\theta^D, \theta^S)$, where $\theta^S = (\boldsymbol{c}^{\boldsymbol{U+D}}, \boldsymbol{\sigma_\nu})$

Infeasibility of Exact Solution Methods More

- Need for a suboptimal solution
  - Exact Solution Method infeasible due to continuous state and action spaces
  - Solution: Approximate Dynamic Programming with Neural Networks

Infeasibility of Exact Solution Methods More

- Need for a suboptimal solution
  - Exact Solution Method infeasible due to continuous state and action spaces
  - Solution: Approximate Dynamic Programming with Neural Networks

- Rewrite the problem with Action-value function

$$Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma}) = \pi_i(p, \boldsymbol{\sigma}_{-i}(\mathbf{s}, \boldsymbol{\nu}), \mathbf{s}, \boldsymbol{\nu}) + \beta \mathbb{E}\left[V_i(\mathbf{s}', \boldsymbol{\nu}'; \boldsymbol{\sigma})\right]$$

$$V_i(\mathbf{s}, \boldsymbol{\nu}; \boldsymbol{\sigma}) = \max_p Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma})$$

Infeasibility of Exact Solution Methods More

- Need for a suboptimal solution
  - Exact Solution Method infeasible due to continuous state and action spaces
  - Solution: Approximate Dynamic Programming with Neural Networks

- Rewrite the problem with Action-value function

$$Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma}) = \pi_i(p, \boldsymbol{\sigma}_{-i}(\mathbf{s}, \boldsymbol{\nu}), \mathbf{s}, \boldsymbol{\nu}) + \beta \mathbb{E}\left[V_i(\mathbf{s}', \boldsymbol{\nu}'; \boldsymbol{\sigma})\right]$$

$$V_i(\mathbf{s}, \boldsymbol{\nu}; \boldsymbol{\sigma}) = \max_p Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma})$$

- For each firm
  - Value Function Approximation: minimize loss $J(\rho) = \mathbb{E}[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))^2]$

Infeasibility of Exact Solution Methods More

- Need for a suboptimal solution
  - Exact Solution Method infeasible due to continuous state and action spaces
  - Solution: Approximate Dynamic Programming with Neural Networks

- Rewrite the problem with Action-value function

$$Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma}) = \pi_i(p, \boldsymbol{\sigma}_{-i}(\mathbf{s}, \boldsymbol{\nu}), \mathbf{s}, \boldsymbol{\nu}) + \beta \mathbb{E}\left[V_i(\mathbf{s}', \boldsymbol{\nu}'; \boldsymbol{\sigma})\right]$$

$$V_i(\mathbf{s}, \boldsymbol{\nu}; \boldsymbol{\sigma}) = \max_p Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma})$$

- For each firm
  - Value Function Approximation: minimize loss $J(\rho) = \mathbb{E}[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))^2]$
  - $Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$: Neural Network
  - True Value Function $Q(\mathbf{s}, \boldsymbol{\nu}, p)$ replaced by target: $y = \pi_i(p, \boldsymbol{\sigma}, \mathbf{s}, \boldsymbol{\nu}) + \beta \max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p')$

Infeasibility of Exact Solution Methods More

- Need for a suboptimal solution
  - Exact Solution Method infeasible due to continuous state and action spaces
  - Solution: Approximate Dynamic Programming with Neural Networks

- Rewrite the problem with Action-value function

$$Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma}) = \pi_i(p, \boldsymbol{\sigma}_{-i}(\mathbf{s}, \boldsymbol{\nu}), \mathbf{s}, \boldsymbol{\nu}) + \beta \mathbb{E}\left[V_i(\mathbf{s}', \boldsymbol{\nu}'; \boldsymbol{\sigma})\right]$$

$$V_i(\mathbf{s}, \boldsymbol{\nu}; \boldsymbol{\sigma}) = \max_p Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma})$$

- For each firm
  - Value Function Approximation: minimize loss $J(\rho) = \mathbb{E}[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))^2]$
  - $Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$: Neural Network
  - True Value Function $Q(\mathbf{s}, \boldsymbol{\nu}, p)$ replaced by target: $y = \pi_i(p, \boldsymbol{\sigma}, \mathbf{s}, \boldsymbol{\nu}) + \beta \max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p')$
  - Problematic with continuous control $\rightarrow$ Solution: learn a maximizer

Infeasibility of Exact Solution Methods More

- Need for a suboptimal solution
  - Exact Solution Method infeasible due to continuous state and action spaces
  - Solution: Approximate Dynamic Programming with Neural Networks

- Rewrite the problem with Action-value function

$$Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma}) = \pi_i(p, \boldsymbol{\sigma}_{-i}(\mathbf{s}, \boldsymbol{\nu}), \mathbf{s}, \boldsymbol{\nu}) + \beta \mathbb{E}\left[V_i(\mathbf{s}', \boldsymbol{\nu}'; \boldsymbol{\sigma})\right]$$

$$V_i(\mathbf{s}, \boldsymbol{\nu}; \boldsymbol{\sigma}) = \max_p Q(\mathbf{s}, \boldsymbol{\nu}, p; \boldsymbol{\sigma})$$

- For each firm
  - Value Function Approximation: minimize loss $J(\rho) = \mathbb{E}[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))^2]$
  - $Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$: Neural Network
  - True Value Function $Q(\mathbf{s}, \boldsymbol{\nu}, p)$ replaced by target: $y = \pi_i(p, \boldsymbol{\sigma}, \mathbf{s}, \boldsymbol{\nu}) + \beta \max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p')$
  - Problematic with continuous control $\rightarrow$ Solution: learn a maximizer (Policy Function Approximation)

Estimation of supply-side parameters More

- From downstream solver we have: $\sigma_i^{\phi_i}(\boldsymbol{s}, \nu_i; \tilde{\boldsymbol{c}}, \boldsymbol{\sigma_\nu}))$
- From upstream solver we have: $\mu_i^{\psi_i}(\boldsymbol{c^{U+D}}, \boldsymbol{\sigma_\nu}))$

Estimation of supply-side parameters More

- From downstream solver we have: $\sigma_i^{\phi_i}(\boldsymbol{s}, \nu_i; \tilde{\boldsymbol{c}}, \boldsymbol{\sigma_\nu}))$
- From upstream solver we have: $\mu_i^{\psi_i}(\boldsymbol{c^{U+D}}, \boldsymbol{\sigma_\nu}))$
- Estimation by MSM with prices implied by the model given by

$$\sigma_i^{\phi_i}\left(\boldsymbol{s}, \nu_i;\ \boldsymbol{c^{U+D}} + \mu_i^{\psi_i}(\boldsymbol{c^{U+D}}, \boldsymbol{\sigma_\nu}), \boldsymbol{\sigma_\nu}\right)$$

- Moments used so far: average price and standard deviation.

Estimation of demand-side parameters

- State dependence in demand:
  - Exogenous: heterogeneity in demographics, historical product availability... Purchase patterns
  - Endogenous: advertisement

- Utility of individual i from product j in period t

$$U_{ijt} = \phi_t - \alpha_i p_{jt} + \xi_{ij} + \boldsymbol{\gamma_i} \mathbb{1}\{j \in s_{it}\} + \epsilon_{ijt}$$

  - s_{it}: set of products purchased last period (observed in the data)

Estimation of demand-side parameters

- State dependence in demand:
  - Exogenous: heterogeneity in demographics, historical product availability... Purchase patterns
  - Endogenous: advertisement

- Utility of individual i from product j in period t

$$U_{ijt} = \phi_t - \alpha_i p_{jt} + \xi_{ij} + \boldsymbol{\gamma_i} \mathbb{1}\{j \in s_{it}\} + \epsilon_{ijt}$$

  - s_{it}: set of products purchased last period (observed in the data)
  - $\boldsymbol{\gamma}$: state dependence coefficient

Estimation of demand-side parameters

- State dependence in demand:
  - Exogenous: heterogeneity in demographics, historical product availability... Purchase patterns
  - Endogenous: advertisement

- Utility of individual i from product j in period t

$$U_{ijt} = \phi_t - \alpha_i p_{jt} + \xi_{ij} + \boldsymbol{\gamma}_i \mathbb{1}\{j \in s_{it}\} + \epsilon_{ijt}$$

  - s_{it}: set of products purchased last period (observed in the data)
  - $\boldsymbol{\gamma}$: state dependence coefficient
  - Random coefficients: $\alpha_i, \xi_{ij}, \gamma_i$
  - ϵ_{ijt}: standard iid EV Type 1 error term
  - Let θ^D denote the vector of demand parameters to be estimated

Conclusion

Research question: How do industry dynamics affect the magnitude of coordination problems and pass-through after integration?

Conclusion

Research question: How do industry dynamics affect the magnitude of coordination problems and pass-through after integration?

In this presentation:

- Overview of the industry and data
- Structural model of vertical relations with state dependence in demand
- Estimation of structural model

Conclusion

Research question: How do industry dynamics affect the magnitude of coordination problems and pass-through after integration?

In this presentation:

- Overview of the industry and data
- Structural model of vertical relations with state dependence in demand
- Estimation of structural model

Next:

- Provide theoretical model to illustrate effect of dynamics on coordination problem and pass-through
- Counterfactuals will
  - Quantify magnitude of coordination problem in dynamic context
  - Show dynamic pass-through incentives
  - Welfare consequences of mergers
  - For different market structures and level of intertemporal linkages

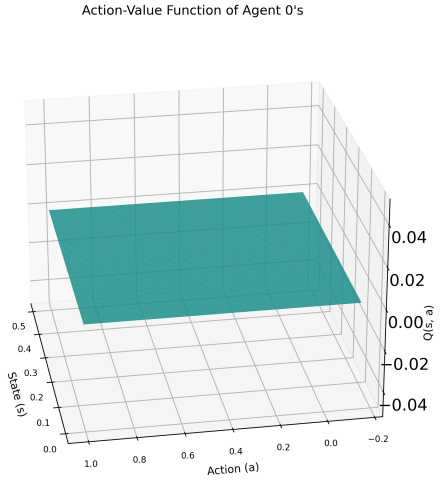
Thank you

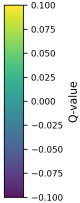
Toulouse School of Economics
maxim.sandiumengeiboy@tse-fr.eu

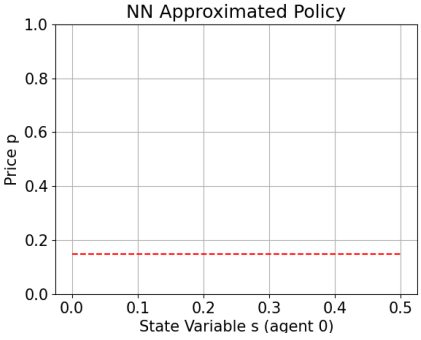
Appendix

Training: initalize neural networks

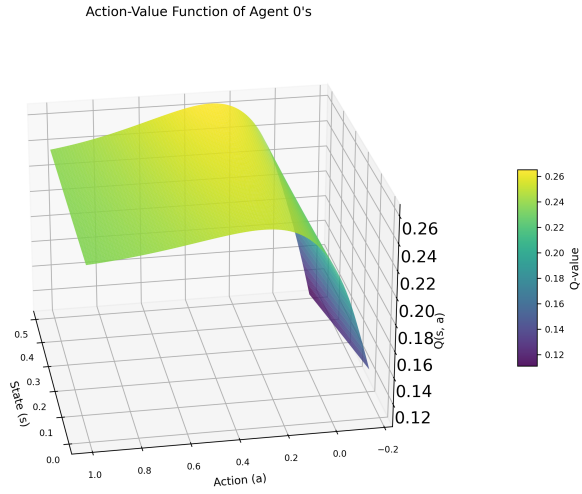
Back



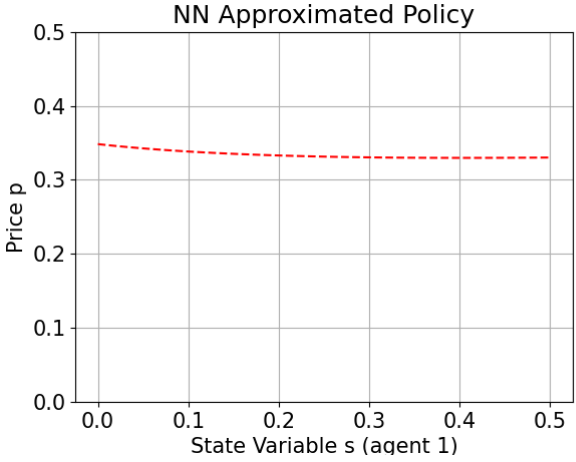




Training: fix policies and train VF until convergence



Training: fix VF and train policy until convergence



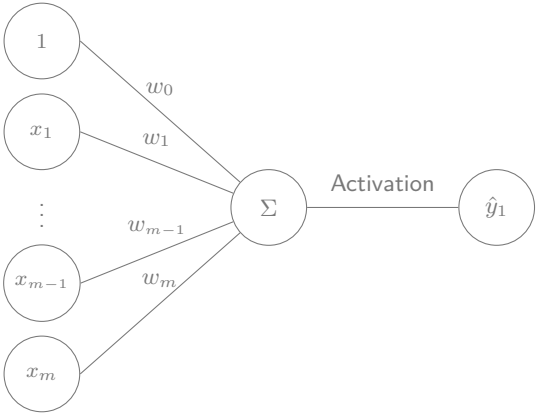
Effect of the merger on prices

Back

$$Y_{jct} = \beta_0 + \beta_1 D_{jct} + \sigma_j + \kappa_c + \lambda_t + \epsilon_{jct}$$

	(1)	(2)
D	-.0124***	-.0111***
	(0.0054)	(0.0029)
Fixed effects	no	yes
R^2 adj	0.0847	0.7429
N	5,730	5,730

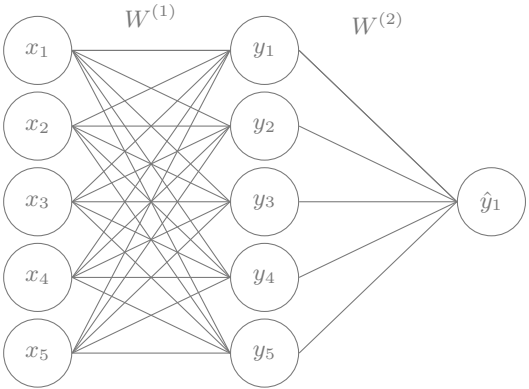
Neural Networks: a single neuron



- Inputs: $x = \{1, x_1, ..., x_m\}$
- Activation function introduces non-linearity: e.g. Sigmoid function $\sigma(z) = \frac{1}{1+e^{-z}}$
- Bias term allows to shift activation
- Output is a weighted sum of inputs and applies a non-linear function

$$\hat{y}_1 = \sigma\left(w_0 + \sum_{i=1}^{m} x_i w_i\right)$$

Neural Networks: Multiple Neurons in a Hidden Layer



- Weights and biases at each layer
- Each neuron in the hidden layer calculates a weighted sum of inputs and applies an activation function.

$$y_i = \sigma\left(w_{0,i}^{(1)} + \sum_{j=1}^{m} x_j w_{j,i}^{(1)}\right)$$

- Output:

$$\hat{y}_1 = \sigma\left(w_{0,i}^{(2)} + \sum_{j=1}^{m_1} y_j w_{j,1}^{(2)}\right)$$

- Trainable parameters: collection of all entries of the weight matrices and bias vectors.

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016)

Value Function Approximation:

- Consider single-region and the problem of firm i for given $\boldsymbol{\sigma}_{-\boldsymbol{i}}$
- Goal: find ρ minimizing distance between approximate VF and true VF

$$J(\rho) = \mathbb{E}[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))^2]$$

- Apply gradient descent (training):

$$\Delta\rho = -\frac{1}{2}\alpha\nabla_\rho J(\rho) = \alpha\mathbb{E}\left[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))\nabla_\rho Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)\right]$$

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016)

Value Function Approximation:

- Consider single-region and the problem of firm i for given $\boldsymbol{\sigma_{-i}}$
- Goal: find ρ minimizing distance between approximate VF and true VF

$$J(\rho) = \mathbb{E}[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))^2]$$

- Apply gradient descent (training):

$$\Delta\rho = -\frac{1}{2}\alpha\nabla_\rho J(\rho) = \alpha\mathbb{E}\left[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))\nabla_\rho Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)\right]$$

- Need to decide:
  - $Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$: Neural Network
  - $Q(\mathbf{s}, \boldsymbol{\nu}, p)$: replace it by a target (TD or MC)

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016)

Value Function Approximation:

- Consider single-region and the problem of firm i for given $\boldsymbol{\sigma_{-i}}$
- Goal: find ρ minimizing distance between approximate VF and true VF

$$J(\rho) = \mathbb{E}[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))^2]$$

- Apply gradient descent (training):

$$\Delta\rho = -\frac{1}{2}\alpha\nabla_\rho J(\rho) = \alpha\mathbb{E}\left[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))\nabla_\rho Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)\right]$$

- Need to decide:
  - $Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$: Neural Network
  - $Q(\mathbf{s}, \boldsymbol{\nu}, p)$: replace it by a target (TD or MC)

$$y = \pi_i(p, \boldsymbol{\sigma}, \mathbf{s}, \boldsymbol{\nu}) + \beta \max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p')$$

- Problematic with continuous control $\rightarrow$ Solution: learn a maximizer

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016)

Value Function Approximation:

- Consider single-region and the problem of firm i for given $\boldsymbol{\sigma_{-i}}$
- Goal: find ρ minimizing distance between approximate VF and true VF

$$J(\rho) = \mathbb{E}[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))^2]$$

- Apply gradient descent (training):

$$\Delta\rho = -\frac{1}{2}\alpha\nabla_\rho J(\rho) = \alpha\mathbb{E}\left[(Q(\mathbf{s}, \boldsymbol{\nu}, p) - Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p))\nabla_\rho Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)\right]$$

- Need to decide:
  - $Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$: Neural Network
  - $Q(\mathbf{s}, \boldsymbol{\nu}, p)$: replace it by a target (TD or MC)

$$y = \pi_i(p, \boldsymbol{\sigma}, \mathbf{s}, \boldsymbol{\nu}) + \beta \max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p')$$

- Problematic with continuous control $\rightarrow$ Solution: learn a maximizer (Policy Function Approximation)

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016) Algorithm

Policy Function Approximation: Alternative

- $\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p') = Q_\rho(\mathbf{s}', \boldsymbol{\nu}', \arg\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p'))$
- $\mu_\phi(\mathbf{s}, \boldsymbol{\nu}) \approx \arg\max_{p} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016) Algorithm

Policy Function Approximation: Alternative

- $\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p') = Q_\rho(\mathbf{s}', \boldsymbol{\nu}', \arg\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p'))$
- $\mu_\phi(\mathbf{s}, \boldsymbol{\nu}) \approx \arg\max_{p} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$
- Goal: find parameters that maximize objective function (measure of the quality of a given policy)

$$\phi = \arg\max_{\phi} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))$$

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016) Algorithm

Policy Function Approximation: Alternative

- $\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p') = Q_\rho(\mathbf{s}', \boldsymbol{\nu}', \arg\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p'))$
- $\mu_\phi(\mathbf{s}, \boldsymbol{\nu}) \approx \arg\max_{p} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$
- Goal: find parameters that maximize objective function (measure of the quality of a given policy)

$$\phi = \arg\max_{\phi} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))$$

$$\Delta\phi = \alpha \nabla_\phi \mathbb{E}\left[Q_\rho(\mathbf{s}, \boldsymbol{\nu}, \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))\right]$$

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016) Algorithm

Policy Function Approximation: Alternative

- $\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p') = Q_\rho(\mathbf{s}', \boldsymbol{\nu}', \arg\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p'))$
- $\mu_\phi(\mathbf{s}, \boldsymbol{\nu}) \approx \arg\max_{p} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$
- Goal: find parameters that maximize objective function (measure of the quality of a given policy)

$$\phi = \arg\max_{\phi} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))$$

$$\Delta\phi = \alpha \nabla_\phi \mathbb{E}\left[Q_\rho(\mathbf{s}, \boldsymbol{\nu}, \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))\right]$$

- New target in VF approximation:

$$y = \pi_i(p, \boldsymbol{\sigma}, \mathbf{s}, \boldsymbol{\nu}) + \beta Q_\rho(\mathbf{s}', \boldsymbol{\nu}', \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))$$

Solving the Single-Agent Problem - DDPG (Lillicrap et al. 2016) Algorithm

Policy Function Approximation: Alternative

- $\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p') = Q_\rho(\mathbf{s}', \boldsymbol{\nu}', \arg\max_{p'} Q_\rho(\mathbf{s}', \boldsymbol{\nu}', p'))$
- $\mu_\phi(\mathbf{s}, \boldsymbol{\nu}) \approx \arg\max_{p} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, p)$
- Goal: find parameters that maximize objective function (measure of the quality of a given policy)

$$\phi = \arg\max_{\phi} Q_\rho(\mathbf{s}, \boldsymbol{\nu}, \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))$$

$$\Delta\phi = \alpha \nabla_\phi \mathbb{E}\left[Q_\rho(\mathbf{s}, \boldsymbol{\nu}, \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))\right]$$

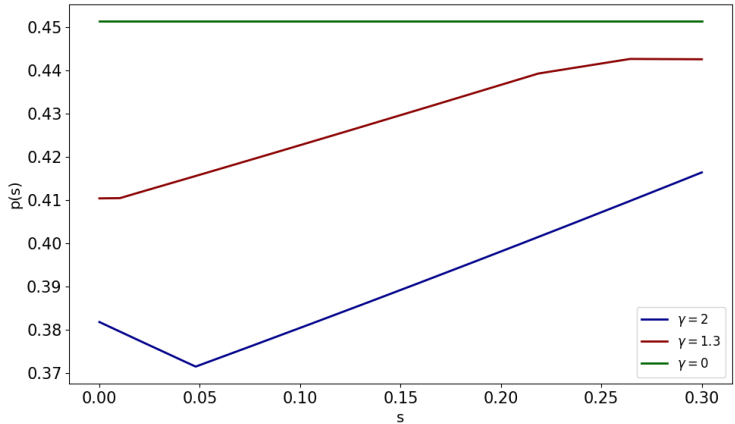
- New target in VF approximation:

$$y = \pi_i(p, \boldsymbol{\sigma}, \mathbf{s}, \boldsymbol{\nu}) + \beta Q_\rho(\mathbf{s}', \boldsymbol{\nu}', \mu_\phi(\mathbf{s}, \boldsymbol{\nu}))$$

Trained policy is the solution of the problem: $\mu_\phi(\mathbf{s}, \boldsymbol{\nu})$, or could also be trained on the supply parameters $\mu_\phi(\mathbf{s}, \boldsymbol{\nu}; \theta^S)$

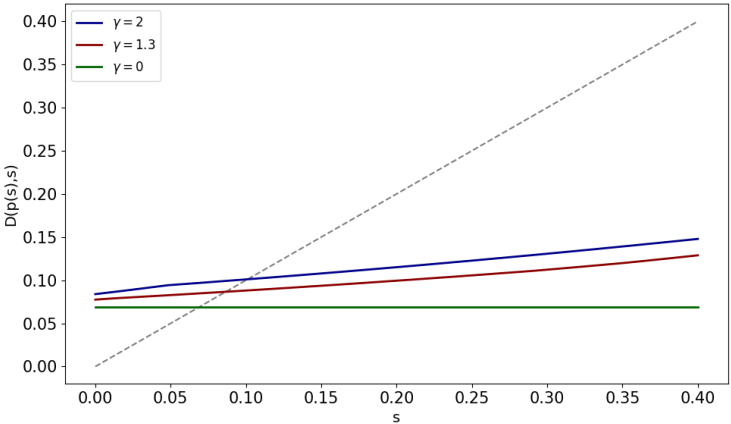
Solution: equilibrium policy

Action VF



Equilibrium policy for different values of state dependence in demand

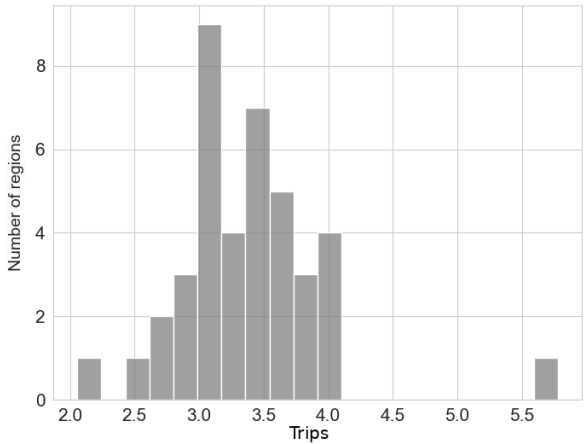
Solution: demand dynamics and stationary point



Demand dynamics for different values of state dependence in demand

Purchase Behavior Across Regions 1/4: re-purchase path duration

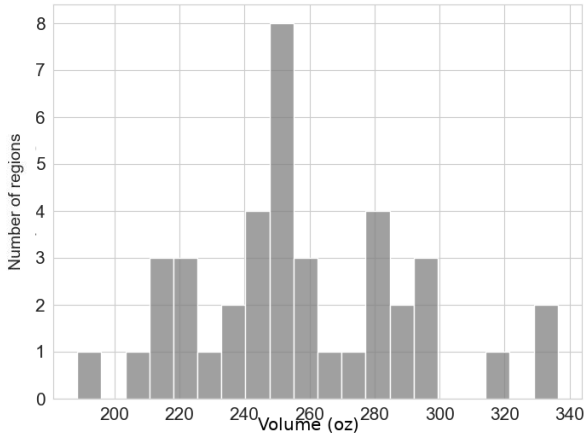
Back



Average Consecutive Repurchase Path Duration Across Regions

Purchase Behavior Across Regions 2/4: purchase volume per trip

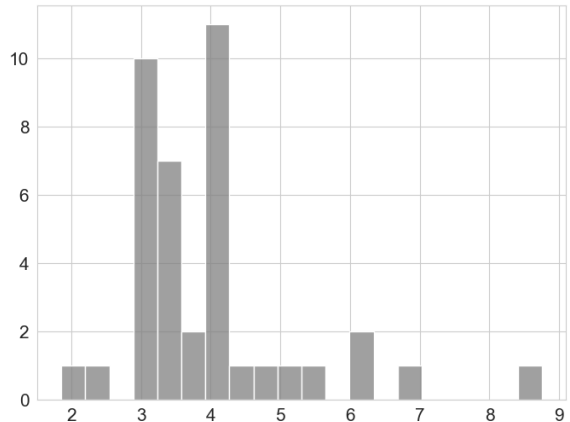
Back



Average Purchase Volume Across Regions (oz)

Purchase Behavior Across Regions 3/4: purchase frequency

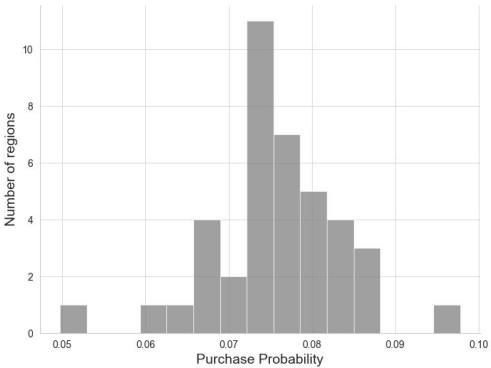
Back



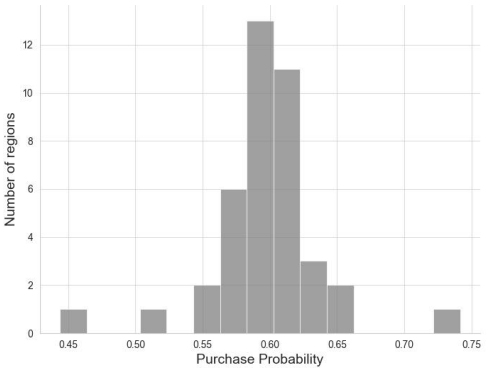
Average Week Difference Between Trips Across Regions

Purchase Behavior Across Regions 4/4: purchase probability

Back



Conditional on no re-purchase



Conditional on re-purchase

Back

1: **Input**: initial policy parameters ϕ, Q-function parameters ρ, empty replay buffer $\mathcal{D}$
2: Set target parameters equal to main parameters $\rho_{\text{targ}} \leftarrow \rho$, $\phi_{\text{targ}} \leftarrow \phi$
3: **repeat**
4: Observe state s and select action $a = \text{clip}(\mu_\phi(s) + \epsilon, a_{\text{low}}, a_{\text{high}})$, where $\epsilon \sim \mathcal{N}$
5: Execute a in the environment
6: Observe next state s' and reward r
7: Store (s, a, r, s') in replay buffer $\mathcal{D}$
8: **if** it's time to update **then**
9: **for** however many updates **do**
10: Randomly sample a batch of transitions, $B = \{(s, a, r, s')\}$ from $\mathcal{D}$
11: Compute targets

$$y(r, s', d) = r + \gamma(1 - d)Q_{\rho_{\text{targ}}}(s', \mu_{\phi_{\text{targ}}}(s'))$$

Back

12: Update Q-function by one step of gradient descent using

$$\nabla_\rho \frac{1}{|B|} \sum_{(s,a,r,s') \in B} \left(Q_\rho(s,a) - y(r,s')\right)^2$$

13: Update policy by one step of gradient ascent using

$$\nabla_\rho \frac{1}{|B|} \sum_{s \in B} Q_\rho(s, \mu_\phi(s))$$

14: Update target networks by Polyak averaging

$$\phi_{\text{targ}} \leftarrow \eta \phi_{\text{targ}} + (1-\eta)\phi$$

$$\rho_{\text{targ}} \leftarrow \eta \rho_{\text{targ}} + (1-\eta)\rho$$

15: **end for**

16: **end if**

17: **until** convergence

Estimation: MSM

Back

- Let $\kappa = \{c, \sigma\}$ be the vector of parameters to be estimated
- Draw a set of marginal cost shocks $\tilde{\nu} = \{\tilde{\nu}_1, ..., \tilde{\nu}_J\}$ from the distribution $\mathcal{N}(0, \sigma^2)$.
- Simulated moments: $\hat{m}(s, \kappa) = \{\hat{m}_1(s, \kappa), ..., \hat{m}_R(s, \kappa)$ with r^{th} element: $\hat{m}_r(\mathbf{s}, \kappa) = \frac{1}{J} \sum_{j=1}^{J} m_r(s, \tilde{\nu}_j, \kappa)$
- $e(\mathbf{s}, \kappa) = \{e_1(\mathbf{s}, \kappa), ..., e_R(\mathbf{s}, \kappa)\}$
- $e_r(\mathbf{s}, \kappa) = \hat{m}_r(\mathbf{s}, \kappa) - m_r(\mathbf{s})$
- $\hat{\kappa}_{MSM} = \arg\min_{\kappa} e(\mathbf{s}, \kappa)' \mathbf{W} e(\mathbf{s}, \kappa)$

Estimation: MSM

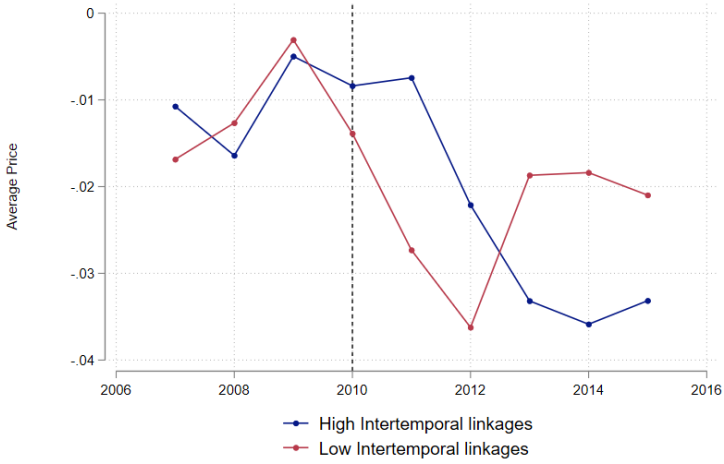
Back

- Let $\kappa_0 = \{c_0, \sigma_0\}$ be the initial guess of the vector of parameters
- Draw a set of marginal cost shocks $\tilde{\nu} = \{\tilde{\nu}_1, ..., \tilde{\nu}_S\}$ from the distribution $\mathcal{N}(0, \sigma^2)$.
- For each state and shock, compute prices implied by the model: $p^{NN}(s, \tilde{\nu}_i, c, \sigma)$
- Compute moments observed in the data and predicted by the moment for each value of the state s
  - $\bar{p}^{NN}(s, \kappa) = \frac{1}{S} \sum_{i=1}^{S} p^{NN}(s, \tilde{\nu}_i, \kappa)$
  - $p_{SD}^{NN}(s, \kappa) = \sqrt{\frac{1}{S} \sum_{i=1}^{S} (p^{NN}(s, \tilde{\nu}_i, \kappa) - \bar{p}^{NN}(s, \kappa))^2}$
- Define the loss function

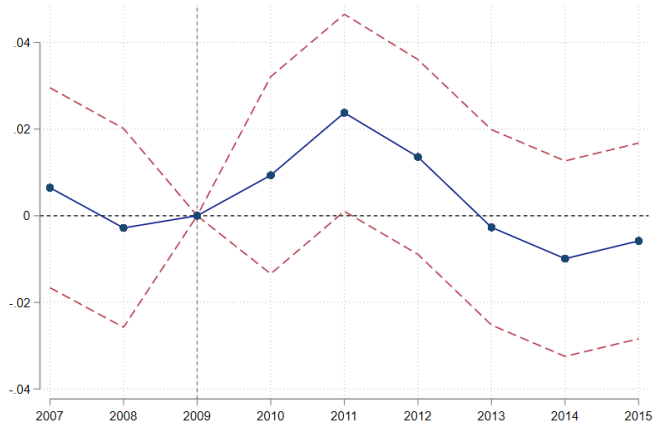
$$L(\kappa) = \sum_{s} \left[(\bar{p}^{NN}(s, \kappa) - \bar{p}^{obs}(s))^2 + (p_{SD}^{NN}(s, \kappa) - p_{SD}^{obs}(s))^2 \right]$$

- Update parameters, go back to first step, until convergence.

Differential trends TD



Diff trends Event Study TD



Event Study TD, by level of intertemporal linkages

Back

- High intertemporal linkages

- Low intertemporal linkages

Triple Difference Estimator

Back

Average Price	(1)	(2)	(3)	(4)	(5)	(6)
D	-0.0150	-0.0290***	-0.0332***	-0.01202	-0.0127***	-0.0107***
	(0.0194)	(0.0103)	(0.0092)	(0.0078)	(0.0041)	(0.0039)
D*γ	0.0016	0.0160*	0.0240***			
	(0.0165)	(0.0088)	(0.0075)			
D*γ^H				-0.0012	0.0035	0.0098*
				(0.0109)	(0.0058)	(0.0051)
Product, county, period FE		✓	✓		✓	✓
Product-by-county FE			✓			✓
Product-by-period FE			✓			✓
County-by-period FE			✓			✓
$R^2 adj$	0.0859	0.7437	0.9173	0.0849	0.7433	0.9171
N	5,730	2,910	2,820	5,730	5,730	5,730

Non-linearities

Average Price	(1)	(2)	(3)	(4)	(5)	(6)
D	0.0144	0.0631	0.1774***	-0.0064	-0.0033	-0.0008
	(0.1079)	(0.0576)	(0.053)	(0.0131)	(0.0070)	(0.0061)
D*γ	-0.0562	-0.1558	-0.3636***			
	(0.1949)	(0.1042)	(0.0967)			
D*γ^2	0.0256	0.0735*	0.1638***			
	(0.0817)	(0.0437)	(0.0407)			
D*$\gamma^{L.2}$				-0.0132	-0.0192*	-0.0169**
				(0.0153)	(0.0082)	(0.0073)
D*$\gamma^{L.3}$				-0.0033	0.0030	0.0073
				(0.0162)	(0.0086)	(0.0073)
Product, county, period FE		✓	✓		✓	✓
Other FE			✓			✓
$R^2 adj$	0.0868	0.7438	0.9178	0.0894	0.7437	0.9175
N	5,730	5,730	5,628	5,730	5,730	5,628

Alternative policy update for Single-Agent

Back

- Bellman equation for firm i, given strategy profile of other firms:

$$V_i(\mathbf{s};\boldsymbol{\sigma}) = \max_{(p_t)} \left[\pi_i(p_i, \boldsymbol{\sigma}_{-i}(\mathbf{s}), \mathbf{s}) + \beta V_i(\mathbf{s}';\boldsymbol{\sigma})\right]. \tag{1}$$

- The first order condition for the optimization problem of firm i is

$$\frac{\partial \pi_i}{\partial p_i}(p_i, \boldsymbol{\sigma}_{-i}, \mathbf{s}) + \beta \frac{dV_i}{d\mathbf{s}'}(\mathbf{s}') \frac{\partial \mathbf{s}'}{\partial p_i}(p_i, \boldsymbol{\sigma}_{-i}, \mathbf{s}) = 0 \tag{2}$$

- Differentiate equation 1 in order to get an equation for the partial derivatives of the value function.

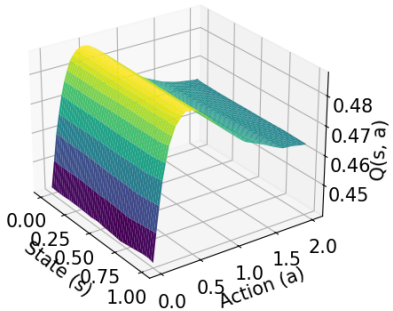
$$\frac{dV_i}{d\mathbf{s}}(\mathbf{s}) = \frac{\partial \pi_i}{\partial \mathbf{s}}(p_i, \boldsymbol{\sigma}_{-i}, \mathbf{s}) + \beta \frac{dV_i}{d\mathbf{s}'}(\mathbf{s}') \frac{\partial \mathbf{s}'}{\partial \mathbf{s}}(p_i, \boldsymbol{\sigma}_{-i}, \mathbf{s}) \tag{3}$$

- One can use to substitute into , in order to substitute back into and get:

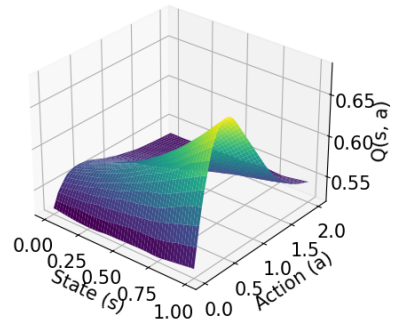
$$\frac{\partial \pi_i}{\partial p_i}(p_i, \boldsymbol{\sigma}_{-i}, \mathbf{s}) + \beta \left[\frac{\partial \pi_i}{\partial \mathbf{s}} - \frac{\partial \pi_i}{\partial p_i}\left(\frac{\partial \mathbf{s}'}{\partial \mathbf{p}}\right)^{-1} \frac{\partial \mathbf{s}'}{\partial \mathbf{s}}\right](p_i', \boldsymbol{\sigma}'_{-i}, \mathbf{s}') \frac{\partial \mathbf{s}'}{\partial p_i}(p_i, \boldsymbol{\sigma}_{-i}, \mathbf{s}) = 0 \tag{4}$$

Solution: action-value function

Back



- No state dependence in demand



- Average state dependence in demand