~~Model-Based and Model-Free Cost Reduction in a Simulated Multi-Echelon Supply Chain~~

2022 International System Dynamics Conference

James Paine
jpaine@mit.edu
http://jpaine.mit.edu

0

Value of Putting the 'B' in 'BOM'

(...in policy development in dynamic decision-making environments)

2022 International System Dynamics Conference

James Paine
jpaine@mit.edu
http://jpaine.info

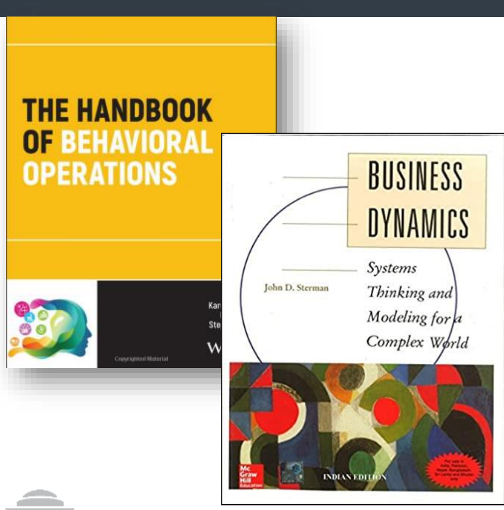
1

Behavioral Operations Management

- Recognition that supply chains consist of, and exist for, often non-rational entities
- (Often) Focus on comparing observed decision to 'optimal'
  - Ignores larger dynamic decision-making environment in which people operate outside of single problem at hand
  - Misses opportunity to develop policies that incorporate, rather than critique, these 'non-optimal'

2

This Work

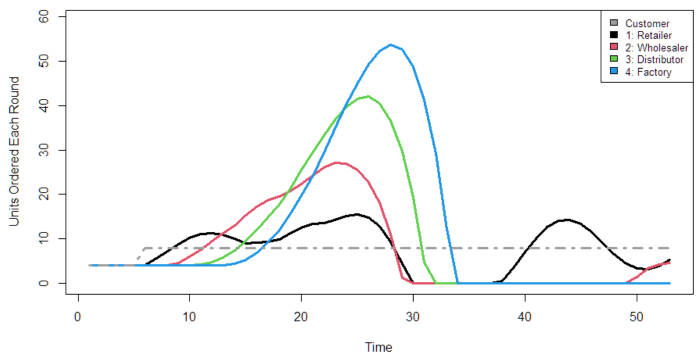


- 'Challenge the clouds' of typical OM/OR by outright expecting non-optimal responses in crafting cost reduction policy
- Incorporate dynamic learning into these policies

3

3

A dynamic decision-making environment with limited information availability

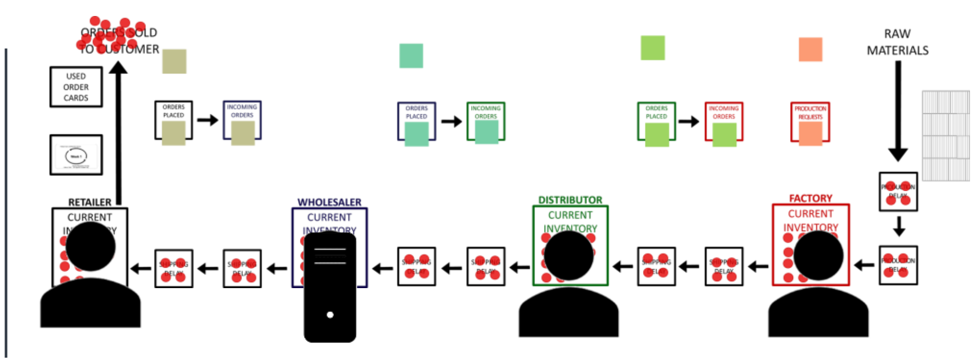


Bullwhip is *structurally induced* but *behaviorally amplified* in this environment

4

4

Modeling Framework: THE BEER GAME



5

5

Behavioral Ordering Models

Modular Simulation Model of the Beer Game

- Discrete time simulation
- Modularized to allow different ordering rules from prior literature
- E.g. Sterman 89 Ordering rule based on four parameters:

$$O_t = MAX(0, \widehat{L_t} + \alpha_S(S' - S_t - \beta\, SL_t) + \varepsilon_t)$$

$$where\ \widehat{L_t} = \theta L_t + (1-\theta)\hat{L}_{t-1}$$

O = order placed at time t

L^ = smoothed interpolation of the expected outflow of inventory

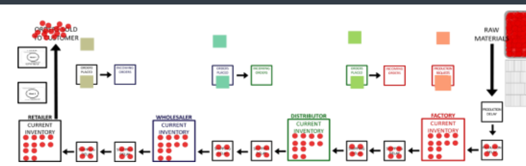
Θ = smoothing parameter

SL = total inbound supply line of inventory

S = current on-hand inventory (or stock)

S' = analogous to the desired or goal on-hand inventory of the player

β = weight of supply line




```
#debug
Entity_Index = 1

or (i in 1:4) {

Entity_Index = i

#Observe supply line
SL = sum(Shipping_flows[Entity_Index,,t]) #Total supply line inbound

#L[Entity_Index,t] = Incoming_Order[Entity_Index]
L[Entity_Index,t] = max(min(Incoming_Order[Entity_Index],OH_Inventory[Entity_Index,t]),Order_flows[Entity_Index,1,t-1]
L[Entity_Index,t] = Order_flows[Entity_Index,1,t-1]

if (t==1) {
  L_hat[Entity_Index,t] = Order_flows[Entity_Index,1,t]
} else if (t==2) {
  L_hat[Entity_Index,t-1] = Order_flows[Entity_Index,1,t-1]
  L[Entity_Index,t-1] = L_hat[Entity_Index,t-1]
}

if (t==20) {
  t = t
}

L_hat[Entity_Index,t] = theta[Entity_Index]*L[Entity_Index,t] + (1-theta[Entity_Index])*L_hat[Entity_Index,t-1]

S = OH_Inventory[Entity_Index,t] #Stock as of current time
```

6

6

Single-Shot Cost Reduction

For some entity index i

- Fix $\theta_{n\neq i}, \alpha_{n\neq i}, S'_{n\neq i}, \beta_{n\neq i}$
- Introduce a definition of 'cost'

$$Cost_{inventory-based} = \sum_{t=1}^{T}\sum_{entity=1}^{N}(C_{bo} * Backorders_{t,n} + C_{inv} * Inventory_{t,n})$$

- Vary $\theta_i, \alpha_i, S'_i, \beta_i$ to minimize cost function

$Exogenous\ O_{t,c}$

$$O_{t,1} = MAX[0, f(\theta_1) + \alpha_1(S'_1 - S_{t,1} - \beta_1\, SL_{t,1})]$$

$$O_{t,2} = MAX[0, f(\theta_2) + \alpha_2(S'_2 - S_{t,2} - \beta_3\, SL_{t,2})]$$

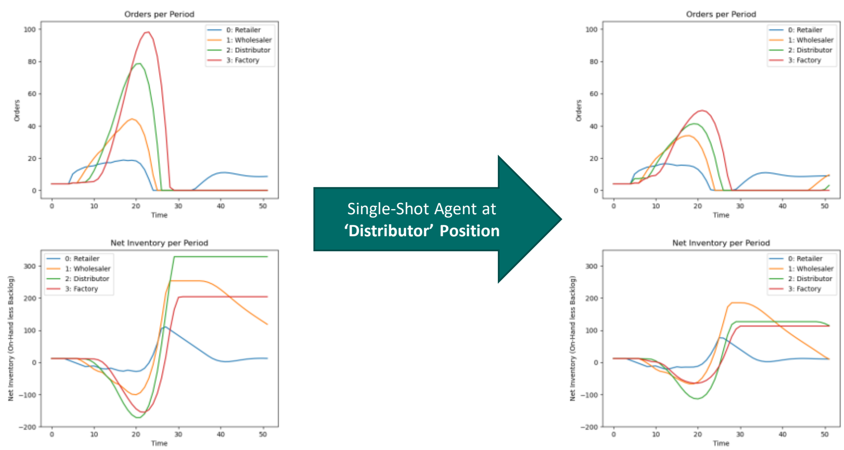
$$O_{t,3} = MAX[0, f(\theta_3) + \alpha_3(S'_3 - S_{t,3} - \beta\_3\, SL_{t,3})]$$

$$O_{t,4} = MAX[0, f(\theta_4) + \alpha_4(S'_4 - S_{t,4} - \beta_4\, SL_{t,4})]$$

7

7

Illustrative Model-Based Result against Step Change in Customer Orders



For 52 simulated weeks: Baseline Costs = 5184
Costs with Single-Shot Optimized Entity = 2,479 **(-52%)**

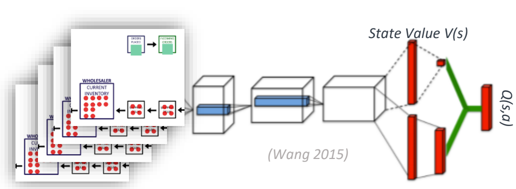
Removing the Model from the Agent: DQN

Framework

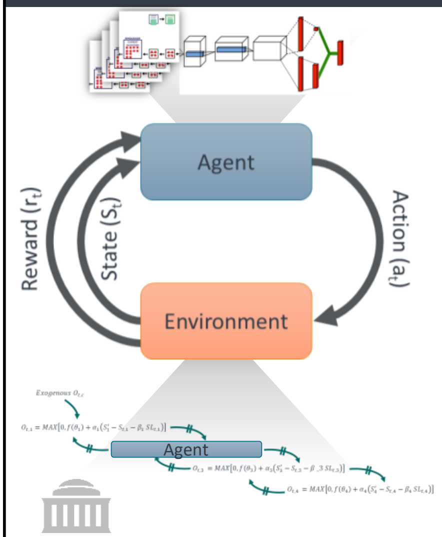
- OpenAI Gym custom environment based on same functional-form of Beer Game developed in Model-based optimization
- Environment consisting of:
  - All supply chain positions in parallel (versus transfer learning in sequence)
  - Randomly drawn models of real teams (from Sterman '89)
  - Random but bounded simulation horizons
  - Noisy realizations of order decisions

DQN Architecture

- Dual DQN network (split Action / State Q values)
- Three sequential dense layers with ReLu activation
- Order-plus action space guided by prior model-optimization
- Combination of epsilon-greedy and Boltzmann policy (Wiering 1999)
- Observation space limited to data available in Beer Game (x4 window for sequential memory)



Model-Informed (but still Model-Free) Approach



Training Environment

- Directly built off of a model of human ordering
- 'Physics' of this model are unknown to the agent

Order-Plus Action Space

- Limits ability for agent to deviate from orders received from supply chain partners and creates tractable action space
- Informed by θ observation in model-based approach

Windowed Observation Space

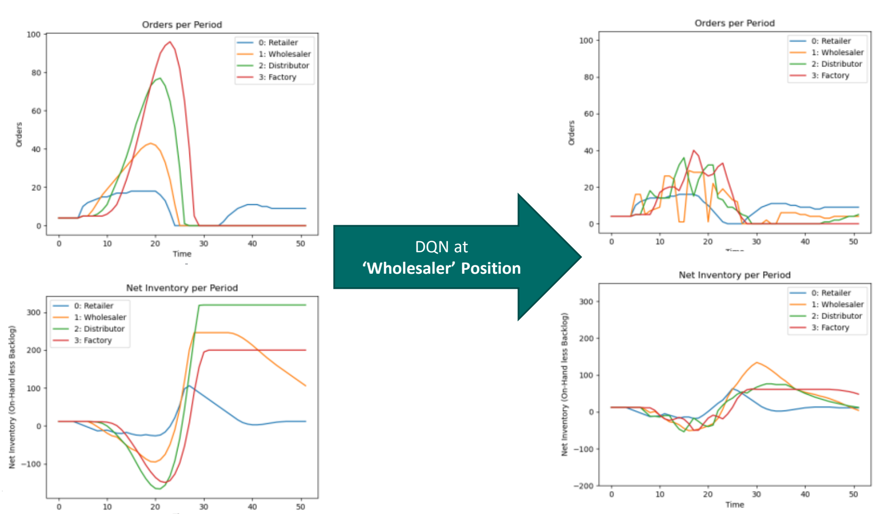
- Observation directly incorporates window of past states
- Size of windowed observations is same as maximum delivery delay for the system
- Informed by β observation in model-based approach

Agent Structure

- Discrete order quantities supported by DQN approach
- Dueling Structure value implied prior behavioral model research
  - Prior work shows small deviations from rational ordering shown to induce bullwhip
  - Actions may have similar value, but need to keep track of state and action separately to avoid inducing bullwhip

10

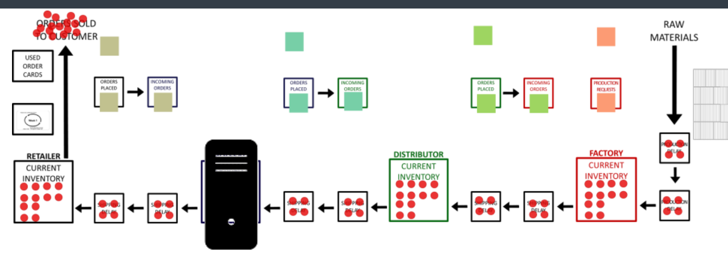
Illustrative DQN Model-Free Result against Step Change in Customer Orders



11

11

Single-Shot Cost Reduction is Model-Constrained, but *not Model-Based*



Exogenous $O_{t,c}$

$$O_{t,1} = MAX[0, f(\theta_1) + \alpha_1(S_1' - S_{t,1} - \beta_1\, SL_{t,1})]$$

$$O_{t,2} = MAX[0, f(\theta_2) + \alpha_2(S_2' - S_{t,2} - \beta_3\, SL_{t,2})]$$

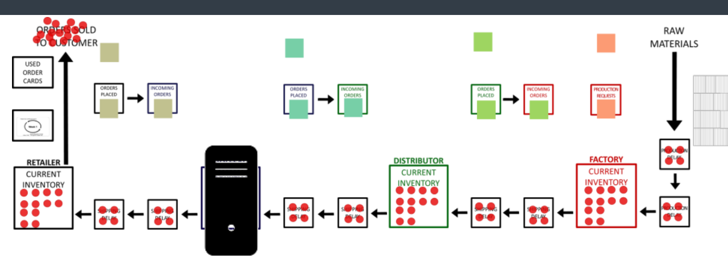
$$O_{t,3} = MAX[0, f(\theta_3) + \alpha_3(S_3' - S_{t,3} - \beta\_3\, SL_{t,3})]$$

$$O_{t,4} = MAX[0, f(\theta_4) + \alpha_4(S_4' - S_{t,4} - \beta_4\, SL_{t,4})]$$

12

12

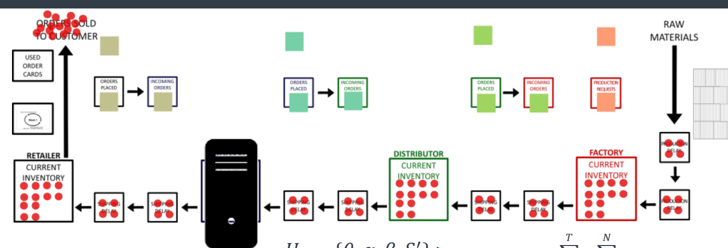
Single-Shot Cost Reduction is Model-Constrained, but *not Model-Based*



Black Box Function

13

Single-Shot Cost Reduction is Model-Constrained, but *not* *Model-Based*



$U_2 = \{\theta, \alpha, \beta, S'\}$

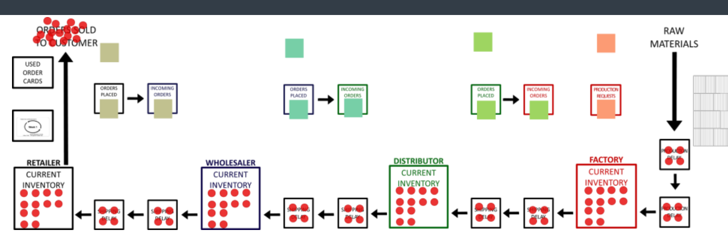
$$\sum_{t=1}^{T} \sum_{entity=1}^{N} (C_{bo} * Backorders_{t,n} + C_{inv} * Inventory_{t,n})$$

$$O_{t,2} = MAX[0, f(\theta_2) + \alpha_2(S'_2 - S_{t,2} - \beta_3\, SL_{t,2})]$$

Black Box Function

14

Incorporate System Structure via Online Parameter Estimation (e.g. MPC)



Exogenous $O_{t,c}$

$$O_{t,1} = MAX[0, f(\theta_1) + \alpha_1(S'_1 - S_{t,1} - \beta_1\, SL_{t,1})]$$

$$O_{t,2} = MAX[0, f(\theta_2) + \alpha_2(S'_2 - S_{t,2} - \beta_3\, SL_{t,2})]$$

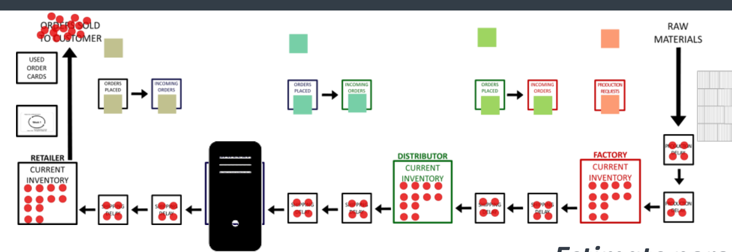
$$O_{t,3} = MAX[0, f(\theta_3) + \alpha_3(S'_3 - S_{t,3} - \beta\_3\, SL_{t,3})]$$

$$O_{t,4} = MAX[0, f(\theta_4) + \alpha_4(S'_4 - S_{t,4} - \beta_4\, SL_{t,4})]$$

15

15

Incorporate System Structure via Online Parameter Estimation (e.g. MPC)



Exogenous $O_{t,c}$

$$\widehat{O_{t,1}} = MAX[0, f(\widehat{\theta_1}) + \widehat{\alpha_1}(\widehat{S_1'} - \widetilde{S_{t,1}} - \widehat{\beta_1}\widetilde{SL_{t,1}})]$$

$$\widehat{O_{t,2}} = MAX[0, f(\widehat{\theta_2}) + \widehat{\alpha_2}(\widehat{S_2'} - \widetilde{S_{t,2}} - \widehat{\beta_2}\widetilde{SL_{t,2}})]$$

$$\widehat{O_{t,3}} = MAX[0, f(\widehat{\theta_3}) + \widehat{\alpha_3}(\widehat{S_3'} - \widetilde{S_{t,3}} - \widehat{\beta_3}\widetilde{SL_{t,3}})]$$

$$\widehat{O_{t,4}} = MAX[0, f(\widehat{\theta_4}) + \widehat{\alpha_4}(\widehat{S_4'} - \widetilde{S_{t,4}} - \widehat{\beta_4}\widetilde{SL_{t,4}})]$$

***Estimate* parameters that best fit observations to an *Assumed* structural model**

16

16

Model-Aware Agent Structure Pseudo Algorithm

Assume system model structure

Calibrate to **observed history**

Optimize based **on assumed and estimated model**

t = 0

Assume Structural and Dynamic Model of System
Define Agent position in System model
Define observable space for Agent
Populate initial assumption of parameterization and initializations
Define backward calibration memory and forward optimization horizon

for t in 1:horizon

Calibrate System Model given history

ArgMin{System Parameter Estimate}

Error (Observed space of simulation of System Model, Actual Observed space)

Return estimated parameters of System Model

Optimize forward given System Model estimate

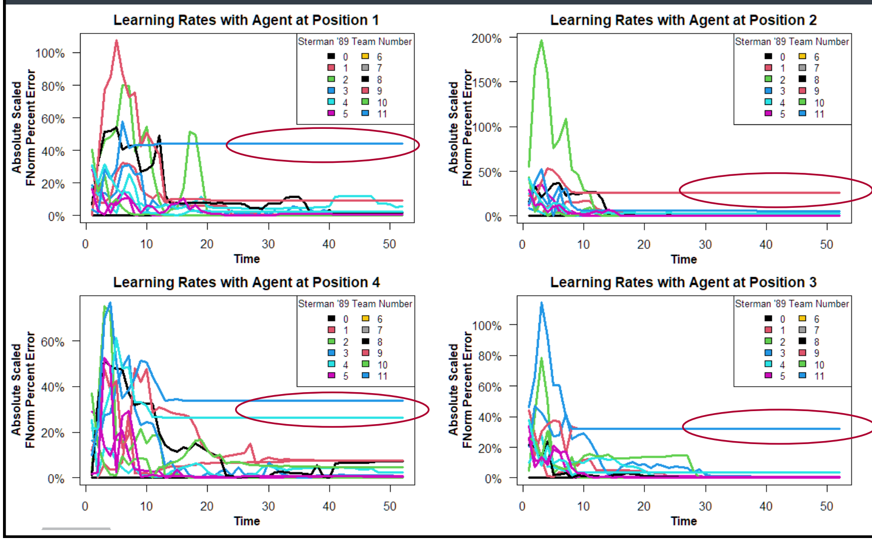
ArgMax{Agent Decision Rule}

Over t:(t+opt horizon): Reward from t:horizon given System Model estimate

17

17

Agent 'Discovers' Environment



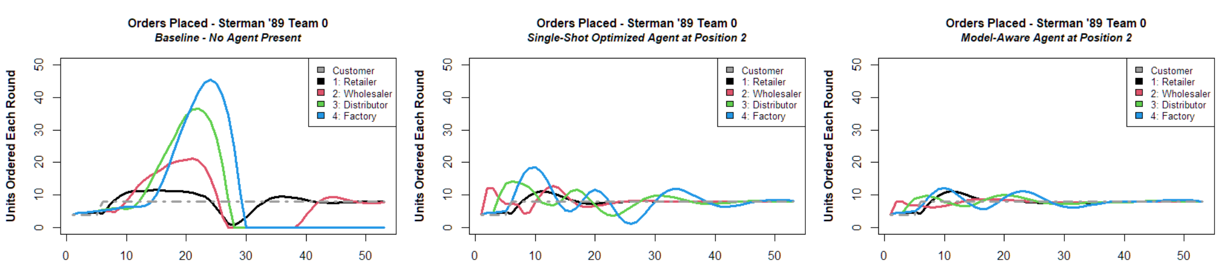
Agent develops estimate of parameters of other supply chain entities

- With repeated interactions, develops better understanding of environment
- But not necessarily a 'true' understanding!
  - Here, can find functionally equivalent point in 12-D space to 'ground truth'

18

18

Agent at Position 2 in Sterman '89 Team 0



Baseline Costs = 5,184

Single-Shot Trained Agent Costs = 1,483 **(-71%)**

Model-Aware Agent Costs = 996 **(-81%)**

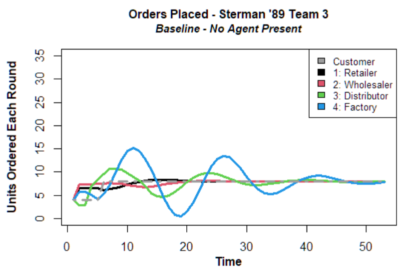
19

19

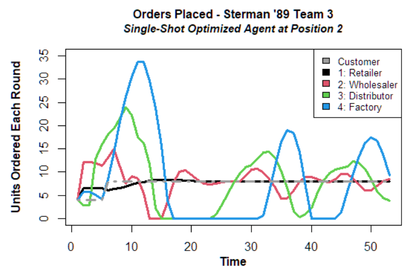
Agent at Position 2 - Wholesaler
(in Sterman '89 Team 3)

Single-Shot Optimization is *not always cost reducing*

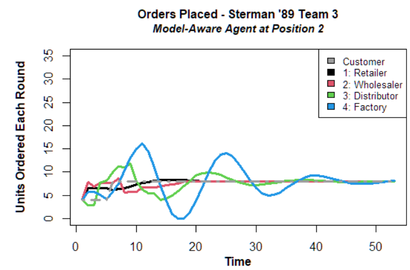
Initial assumptions about model environment



Baseline Costs = 928



Single-Shot Trained Agent Costs = 2473 **(+167%)**



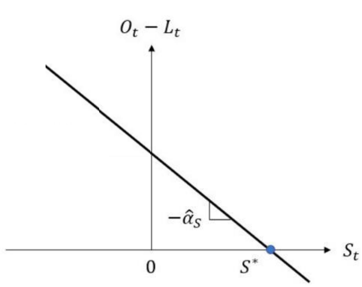
Model-Aware Agent Costs = 872 **(-6%)**

Model-Aware routine can compensate for poor initial assumptions

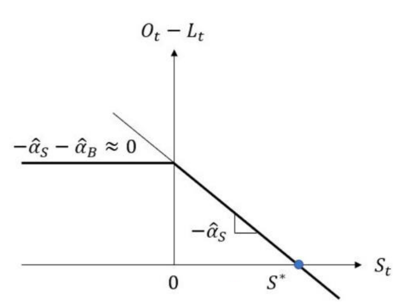
Applicability in Adjacent (but non-identical) environments

Replaced ordering heuristic with that developed by Oliva et al 2021

Sterman '89

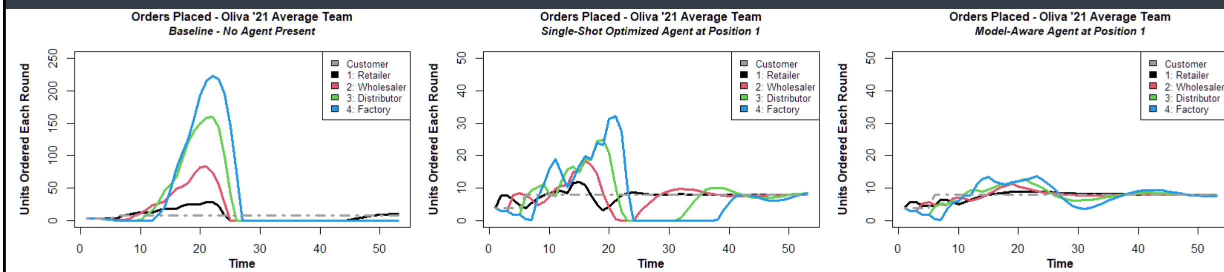


Oliva et al '21



Does the agent trained in one environment *still* reduce costs in an adjacent environment?

Applicability in Adjacent (but non-identical) environments



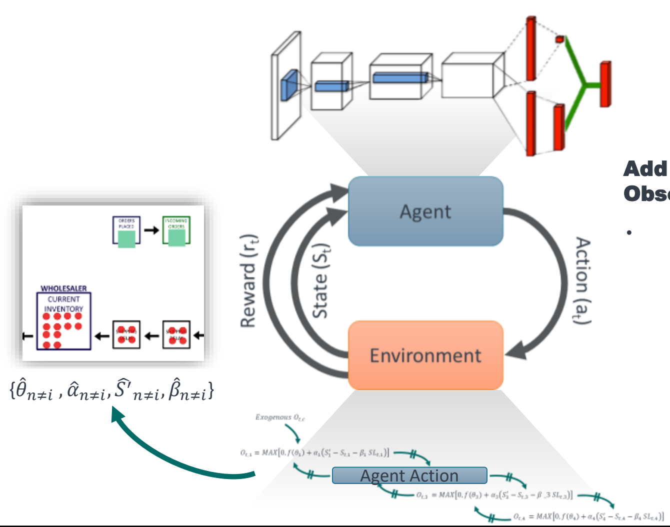
Agent Position		Single-Shot Optimized Agent	Model-Aware Agent
	Baseline Costs	26257	
	1 (Retailer)	1821 (-93.1%)	869 (-96.7%)
	2 (Wholesaler)	3836 (-85.4%)	1040 (-96.0%)
	3 (Distributor)	6681 (-74.6%)	2859 (-89.1%)
	4 (Factory)	16234 (-38.2%)	12274 (-53.3%)

Here even single-shot optimized in different (but adjacent) environment is cost reducing. Model-aware agent more-so

22

22

Model-Aware Deep Q-Network Approach



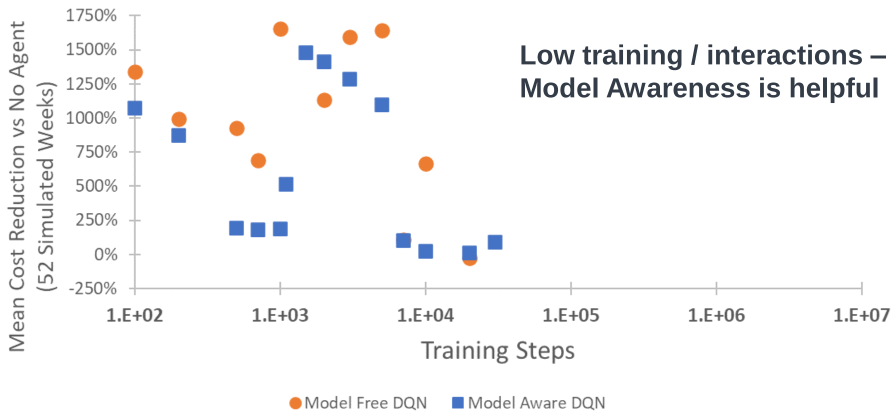
Add *Estimated* System Parameters Observation Space

- Allows for direct incorporation of DQN as cost-reducing agent into Model-Aware algo

23

23

Model-Aware Deep Q-Network Approach



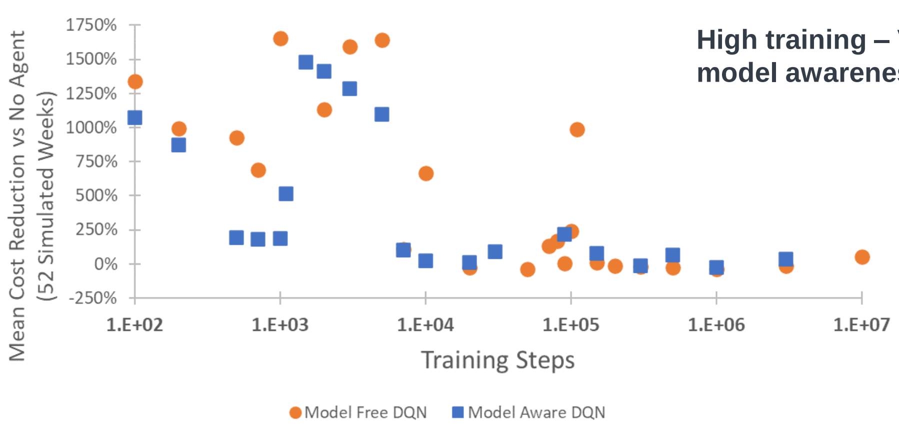
Low training / interactions – Model Awareness is helpful

Trained against bootstrapped teams on full range of heurist rule space. Results shown against 48 teams estimated from original Sterman '89 paper and runs of the Beer Game conducted at MIT Sloan in Fall 2021 and Spring 2022

24

24

Model-Aware Deep Q-Network Approach



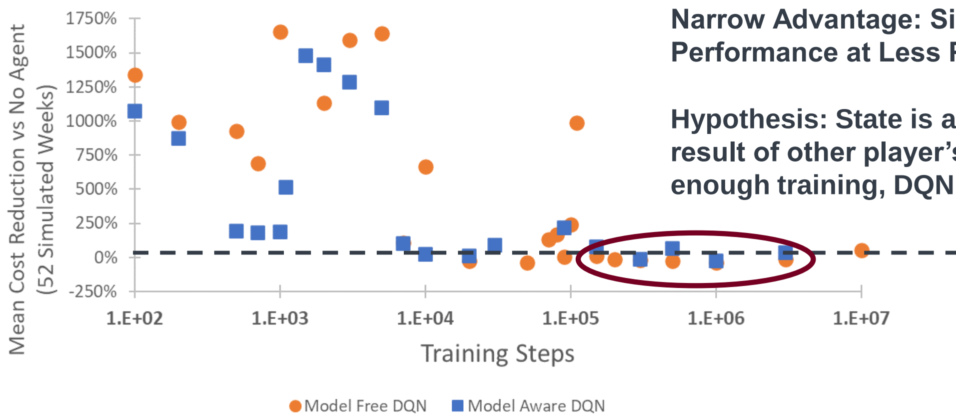
High training – Value of model awareness lost

Trained against bootstrapped teams on full range of heurist rule space. Results shown against 48 teams estimated from original Sterman '89 paper and runs of the Beer Game conducted at MIT Sloan in Fall 2021 and Spring 2022

25

25

Model-Aware Deep Q-Network Approach



Narrow Advantage: Similar Performance at Less Pre-Training

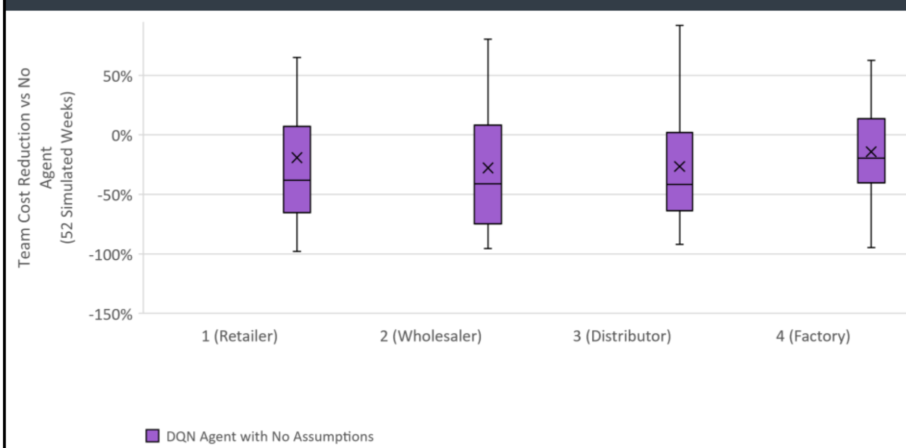
Hypothesis: State is almost entirely a result of other player's heuristics. With enough training, DQN can learn this

Trained against bootstrapped teams on full range of heurist rule space. Results shown against 48 teams estimated from original Sterman '89 paper and runs of the Beer Game conducted at MIT Sloan in Fall 2021 and Spring 2022

26

26

Comparing Policy Assumptions and Structure



- DQN is, on average, cost reducing but can also be destabilizing in some scenarios

Fit Against 48 teams estimated with behavior fitted on Sterman '89
Teams were from original Sterman '89 paper and runs of the Beer Game conducted at MIT Sloan in Fall 2021 and Spring 2022

'Standard' Beer Game setup with step customer order
Simulation Horizon = 52 Periods
For Non DQN Agents, Memory = 5 periods and Forward Horizon = 30 periods
DQN is 'model-free' trained per-position for approx. 1e6 timesteps

27

27

Comparing Policy Assumptions and Structure

- Base-Stock response policy while fitting to a ‘wrong’ model is still surprisingly robust, even though other agents are heuristic
- Still occasionally destabilizing for some simulated teams

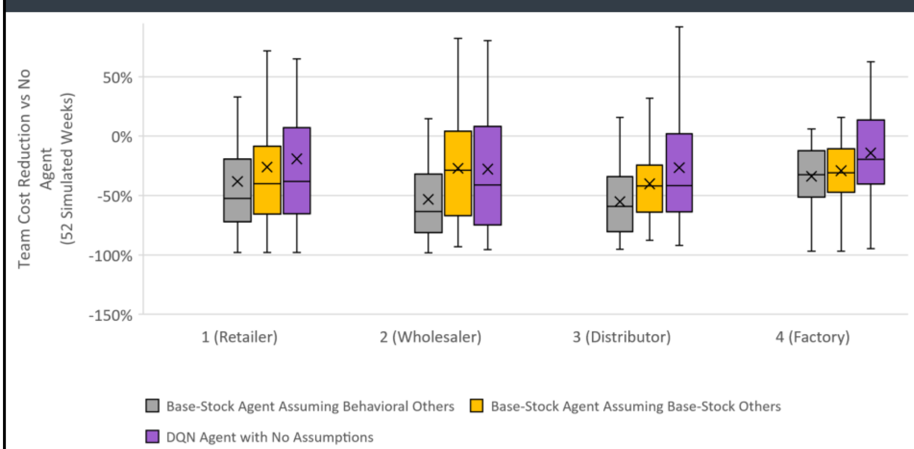
Fit Against 48 teams estimated with behavior fitted on Sterman ’89
Teams were from original Sterman ’89 paper and runs of the Beer Game conducted at MIT Sloan in Fall 2021 and Spring 2022

‘Standard’ Beer Game setup with step customer order
Simulation Horizon = 52 Periods
For Non DQN Agents, Memory = 5 periods and Forward Horizon = 30 periods
DQN is ‘model-free’ trained per-position for approx. 1e6 timesteps

28

28

Comparing Policy Assumptions and Structure



- Fitting a model that better reflects the underlying physics of the system greatly improves cost reduction performance

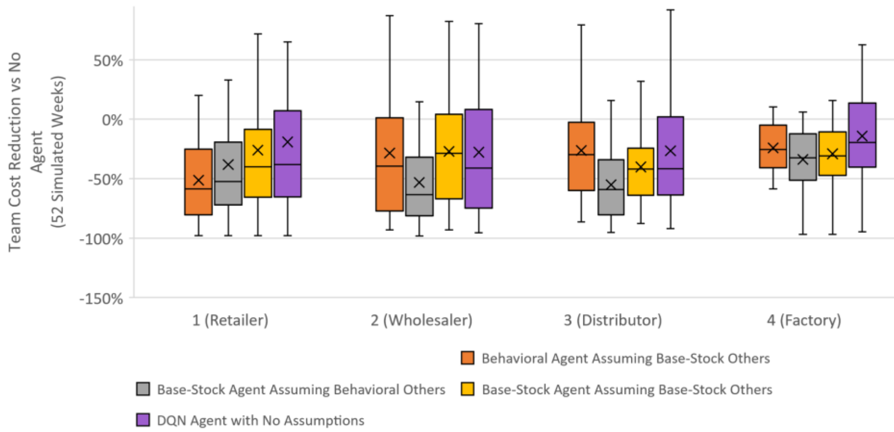
Fit Against 48 teams estimated with behavior fitted on Sterman ’89
Teams were from original Sterman ’89 paper and runs of the Beer Game conducted at MIT Sloan in Fall 2021 and Spring 2022

‘Standard’ Beer Game setup with step customer order
Simulation Horizon = 52 Periods
For Non DQN Agents, Memory = 5 periods and Forward Horizon = 30 periods
DQN is ‘model-free’ trained per-position for approx. 1e6 timesteps

29

29

Comparing Policy Assumptions and Structure



- Just adding more control parameters to response policy does not always improve performance if underlying assumption of system is flawed

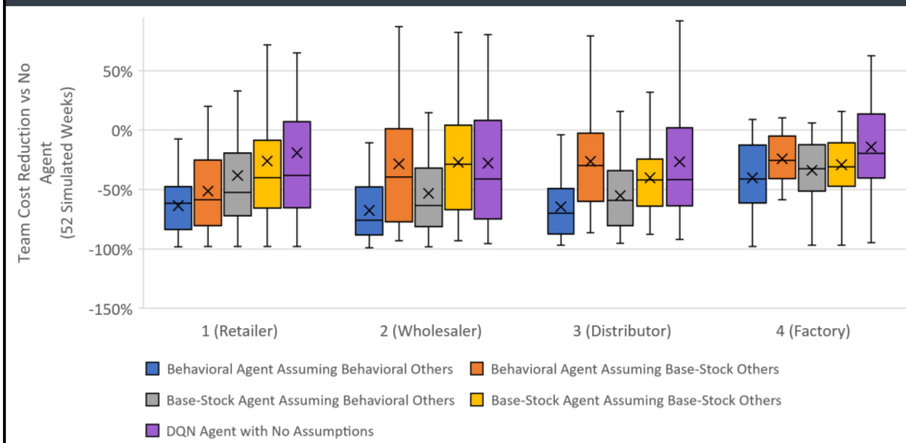
Fit Against 48 teams estimated with behavior fitted on Sterman '89
Teams were from original Sterman '89 paper and runs of the Beer Game conducted at MIT Sloan in Fall 2021 and Spring 2022

'Standard' Beer Game setup with step customer order
Simulation Horizon = 52 Periods
For Non DQN Agents, Memory = 5 periods and Forward Horizon = 30 periods
DQN is 'model-free' trained per-position for approx. 1e6 timesteps

30

30

Comparing Policy Assumptions and Structure



- Combination of a learning heuristic policy paired with a behavioral model of the system is almost universally cost reducing

Fit Against 48 teams estimated with behavior fitted on Sterman '89
Teams were from original Sterman '89 paper and runs of the Beer Game conducted at MIT Sloan in Fall 2021 and Spring 2022

'Standard' Beer Game setup with step customer order
Simulation Horizon = 52 Periods
For Non DQN Agents, Memory = 5 periods and Forward Horizon = 30 periods
DQN is 'model-free' trained per-position for approx. 1e6 timesteps

31

31

Advantage of Heuristic Policy: Interpretability

General Shared Features of the Behavioral Agents

$$O_t = MAX\left(0, \widehat{L_t} + \alpha_S(S' - S_t - \beta\, SL_t) + \varepsilon_t\right)$$

$$where\ \widehat{L_t} = \theta L_t + (1-\theta)\hat{L}_{t-1}$$

Low values of θ for the Retailer and high values of θ for others

- Determines the degree of smoothing in updating each entity's expectation of future orders (Anchoring and Adjustment)
- Low values of θ = slow to update expectations, while high values of θ = is quick to adopt the new order signal
- Retailer dubious about customer orders, all other entities quick to update

Very high values of β (at or near 1.0) throughout

- Directly corresponds to Supply Chain Underweighting
- Matches existing best practices from literature and counteracts largest hypothesized source of Bullwhip

Values of S' resembling (full inventory) base-sock replenishment

- Classic solution to Bullwhip with perfect customer distribution knowledge is base stock replenishment
- S' at or near 36 in all optimizations (and higher in less stable positions), which matches base stock level under uniform random centered around 8 units with total inventory delay of 4 units of time

32

Discussion and Limitations

Notable Limitations

- This is an empirically grounded simulation, but not an empirically verified one (yet)
- Important concerns in real supply chains (integration, cost sharing, etc.) are ignored here. The definition of 'cost' matters!
- Ordering data from real games implies subtle behavioral differences between 'in-person' and online/hybrid runs. This is ignored (for now!)

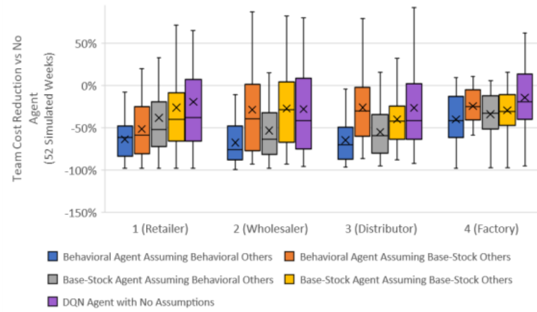
33

33

Discussion and Future Work

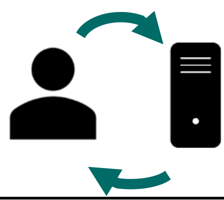
Helping to define the value of the 'behavioral' part of BOM

- Compared Model-Based and Model-Free derived agents (to mitigate Bullwhip)
- Learning model-based methods can compensate for poor initial conditions
- Fully model free DQN less robust, but still able to learn environment when given enough training
- All methods here do not rely on changing behavioral features of others in the supply chain



Next: From Interpretable Behavioral-Based Policy to Simple Managerial Decision Support Tool

- Enumerate cost-reducing heuristic policy for full space
- Translate into simple state-dependent policy for real managers
- Empirical Test and Extension
  1. Does the presence of an algorithmic intervention mitigate bullwhip outside of this model?
  2. Does knowledge of the presence of such a machine modify human ordering behavior?



34

34

Thank You!

Please send questions and comment to:

James Paine

jpaine@mit.edu

jpaine.mit.edu

https://arxiv.org/abs/2202.12786

https://github.com/jpain3/Taming-the-Bull

35

35