Simpler is (Sometimes) Better

A Comparison of Cost Reducing Agent Architectures in a Simulated Behaviorally Driven Multi-Echelon Supply Chain

James E. Paine
jpaine@mit.edu

ABSTRACT

Operations Management endeavors to identify polices that improve real-world operating systems. However, these systems often consist of actors that do not behave fully rationally and to improve performance managers must predict how such boundedly-rational actors will behave. Approaches to this problem range from myopic, limited information decision rules to more modern, but data intensive machine learning methods. This work compares approaches to reducing costs in a simulated, behaviorally driven, multi-echelon supply chain by experimentally changing the features of polices, including complexity, adaptability, incentive structure, and information availability. Results show that that relatively simple ordering policies, especially when accounting for behavioral features of other agents, in these systems can have large cost reducing effects only marginally behind more complex methods. Additionally, for some conditions, greedy, myopic, and limited information decision rules can be cost reducing globally. Under plausible conditions, decision makers in supply chains with other behavioral actors need not be perfectly rational and can be locally focused while achieving global benefits.

Note: All code used to produce the results in the main article and the supporting Appendix document are available at: https://github.com/jpain3/Bullwhip-Policy-Architectures

WORKING PAPER DO NOT DISTRIBUTE

Last Revised: 10/10/2022

1. Introduction

Supply chains consist of, and ultimately exist for, people who derive some value from the movement of goods and services and other value-added activities these systems bring. The study of these goods and service delivery systems mechanisms to either reduce costs or increase value along the chain has been an active area of inquiry since at least the early 20th century with the work of Fredrick Taylor and Frank and Lillian Gilbreth (Krenn, 2011; Payne et al., 2006), and the field of operations management has increasingly followed its peers in economics, marketing, and finance by recognizing the influence of human heuristic-based decision rules and incorporating these behavioral observations into the models of supply chains and inventory management (Gino & Pisano, 2008).

Recent work has endeavored to more clearly and explicitly define 'Behavioral Operations Management' as a subfield of Operations Management and Operations Research, in part, by incorporating observations of human decision heuristics into operations management modeling and policies (Croson et al., 2013; Cui & Wu, 2018; Gino & Pisano, 2008; Größler et al., 2008). The research in this field often features comparisons of an observed decision to a respective optimal decision. Such comparisons of 'optimal' decision making to 'non-rational' or even more broadly defined 'heuristic' decision making are not necessarily fair, or can be outright misleading, in both highlighting the dynamic decision-making mechanisms at play, and in developing appropriate policy interventions in the real operating systems.

Additionally recent Operations Research literature has begun to incorporate increasingly complex policy architectures into polices. Model predictive control schemes (see Åström et al., 2001; Seborg et al., 2016 for overviews) have been used in operational contexts but can abstract away from the human decision making context (see Ciocan & Farias, 2012; Pannek & Frazzon, 2014; Secomandi, 2008; Vossen et al., 2022 for recent supply chain applications of MPC). Alternatively, research can focus on more detailed representation of human decision heuristics, but often emphasizes defining the gap between optimality and observed reality of decision makers over specific policy interventions (see Bendoly, 2013; Huang et al., 2013; Morrison & Oliva, 2018; J. Sterman, 1989 among others). More recently, there has been an expansion of the use of neural network architectures that, perhaps, over emphasize the policy outcomes at the expense of simplicity and interpretability (see Chaharsooghi et al., 2008; Oroojlooyjadid et al., 2021 for two recent supply chain examples).

However, there is also reason to suspect that minimal policies may still be valid in this space, if not outright more applicable. Indeed, other fields that Behavioral Operations Management borrow heavily from such as psychology and other behavioral sciences have already noted that more complex models do not necessarily generate more robust or accurate outcomes. Simple linear models even with imperfectly (but consistently applied) weights can more consistently perform in behaviorally rich settings versus human decision makers (Dawes & Corrigan, 1974; Kahneman et al., 2021; Yu & Kuncel, 2020).

This work endeavors to expand on existing Behavioral Operations Management literature by not further defining the gap between 'rational' and 'irrational' outcomes, but instead by exploring the features of policies that have high leverage in operating systems that are based on, and out-right expect, non-optimal heuristic decision making from other actors in the environment. Specifically, this work considers a dynamic decision-making environment with limited information availability, a setting common to many operational and supply chain settings, to identify the features of policies that minimize costs in these settings when placed alongside (models of) human decision makers. In net, this work endeavors to first *assume* that behavioral heuristics that may significantly depart from optimal or fully rational exist in the decision-making process in operating systems, and then given that assumption, identify the degree of policy complexity necessary to achieve a superior outcome.

One of the more studied consequences of the interaction of behavioral heuristics and supply chain structure is the emergence of instability as embodied by the 'bullwhip effect' in multi-echelon supply chains (see (X. Wang & Disney, 2016) for an excellent review of research on the origins of this effect from both a structural and behavioral perspective from the early 1960's through the mid-2010's). Bullwhip refers to the increasing amplitudes in both orders and on-hand inventory positions of members of a multi-echelon supply chain the further one moves away from a source of order variability. The bullwhip effect is responsible for both excessive strain on real world inventory management systems, stock outs, and unnecessary capital reservation through safety stock building (Ellram, 2010). This phenomena is also not necessarily restricted to any one industry, but rather present in varying forms whenever ordering decisions being made in moderately complex and interlinking environments such as multi-echelon supply chains (Lee et al., 2004; J. Sterman, 1989; J. D. Sterman, 2000). While often viewed as a 'classical' problem in Operations Management, recent worldwide experiences with supply chain shortages and excesses induced by the Coronavirus pandemic for consumer goods, foodstuffs and medical supplies, have catapulted the term 'bullwhip' into the popular consciousness

(Bamakan et al., 2021; Evenett, 2021; Hockett, 2020; Johnson, 2021; Shih, 2020; Stank et al., 2021)

Optimal control policies in multi-echelon supply chains are well understood and well-studied. Work by Clark and Scarf demonstrated that an optimal control policy can be applied via a base-stock ordering system when the final customer demand distribution is known (Clark & Scarf, 1960). Their algorithm was later generalized and operationalized to both multi-echelon supply chains with imperfect local information and stationary demand patterns (Chen, 1999; Chen & Samroengraja, 2009; Lee et al., 2004).

Behavioral causes of ordering and inventory amplification have been thoroughly explored. A key behavioral bias that leads to bullwhip is commonly identified as 'supply-chain underweighting' (Croson & Donohue, 2006; Narayanan & Moritz, 2015; J. Sterman, 1989) and emerges as part of a larger anchoring and adjustment heuristic employed by decision makers in a multi-echelon supply chain (J. Sterman, 1989; Tversky & Kahneman, 1974). Mitigation of bullwhip has focused on adjusting the structure of the supply chain itself, the information availability along the supply chain (Croson et al., 2014; Croson & Donohue, 2006; Wu & Katok, 2006), and on the instruction and training strategies of supply chain managers (Croson et al., 2014; Martin et al., 2004; Wu & Katok, 2006). While mitigation is possible, the underlying ordering heuristics that drive the emergence of bullwhip remain in many of these studies.

Dynamic decision making in multi-echelon supply chains, and specifically investigations of policies that reduce costs associated with bullwhip, provides both a theoretically relevant, and managerially important, environment for this work. The Beer Game (J. Sterman, 1989) is a classical inventory management dynamic system learning tool for exactly such an environment, in which a multi-agent decentralized supply chain is modeled much like real decentralized inventory management systems described above, and is a framework that has been used extensively in prior related research (notably Chaharsooghi et al., 2008; Croson & Donohue, 2006; Narayanan & Moritz, 2015; Oliva et al., 2022; Oroojlooyjadid et al., 2017; J. Sterman, 1989; J. D. Sterman & Dogan, 2015 among others). First developed by Jay Forrester at MIT, the game has been used since the 1950's to illustrate system thinking concepts and the prevalence of the bullwhip effect. The original purpose of the simulation was to illustrate the difficulty of rational thinking during time-delayed and non-linear information feedback loops, value of information sharing, and most classically the bullwhip effect in inventory management.

This same setting is used in this work, allowing the author to develop generalizable insights on how to react in a dynamic decision-making environment, and how to incorporate learning given limited visibility of the full state-space of this complex system.

Using this framework, this paper presents the results of incorporating different agent architectures within a model of a multi-echelon supply chain inhabited by simulated human decision makers. In doing so, the degree of additional value as function of agent complexity is explored, and within those policies that achieve a good trade-off of complexity and performance, the specific features that drive that performance are further identified. In developing these different architectures this work presents two methodological contributions to cost-reducing supply chain agent modeling: an extension of existing DQN architectures that builds off of recent prior literature in this space, and also a model-predictive learning architecture that incorporates features of control theory with features of behavioral modeling. However, these methodological contributions are secondary to the main contribution of this work, namely that the most complex agent architectures are not significantly better than simpler and more directly interpretable alternatives. Furthermore, within those simple alternatives, agents can be locally focused with limited information availability about the supply chain and still achieve comparable levels of performance to those agents with near total information about their environment and a more global view of the supply chain.

2. Methods

A discrete time model of the Beer Game was developed in both the R and Python scripting languages and was based on the board-game configuration of the game as illustrated in Figure 1 below. This figure also shows the typical starting layout for the game as used in numerous previous studies using this modeling framework (Croson & Donohue, 2006; Narayanan & Moritz, 2015; J. Sterman, 1989), which is started with 12 units of inventory on hand for each player, and 4 units of inventory in transit at each stage in the shipping system, and 4 orders moving through the order chain.
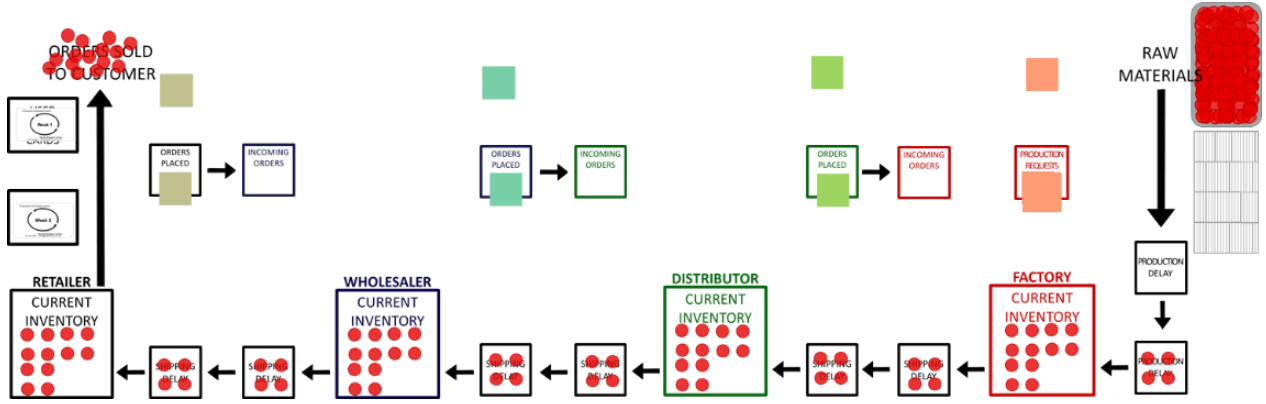
Figure 1. Example of Beer Game Board Layout

Each round of the game proceeds as follows:

1. Receiving inventory and advance shipping delays – Each entity receives the units in the shipping delay immediately to their right. The contents of the furthest shipping delay to the right are moved up

2. Fill orders – Entity 1 (retailer) views the customer order, all others examine the 'incoming orders' and orders, inclusive of any outstanding backorders, are filled to the extended inventory allows

3. Record inventory or backlog

4. Advance order slips – the order slips further to the left are moved up

5. Place orders – Each entity decides what to order and places it the 'orders placed' box to their right

The stated goal of the game is to reduce the amount of *total cost of the entire team* over some time horizon *T,* subject to some known inventory holding and backorder/stockout costs. Backorders do not expire under the traditional interpretation of this game and must be filled from existing stock prior to meeting any new demand. In the prior studies referenced above, and in this work, the cost of holding inventory, C_{inv}, is \$0.50 per unit per period, and the cost of backorders, C_{bo}, is \$1.00 per unit per period.

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

Typically, real human players are placed into this system to make inventory purchasing and management decisions. Within a few rounds of ordering, the bullwhip in inventory and backorders appears, amplifying over time along the simulated supply chain as each player acts

to reserve inventory to satisfy their own myopic forecasts and needs. As discussed above, exact solutions for optimal ordering quantities have been developed, such as the base-sock method (Chen & Samroengraja, 2009; Clark & Scarf, 1960), but require all agents to be acting rationally and consistently, and for specific costing structures to be present (notably increasing costs along the supply chain). Additionally, while these optimal ordering methods presume stationary customer order patterns, which this simulation satisfies, the human participants themselves have no knowledge *a priori* of the distribution of the customer order pattern.

The model was made as both a self-contained simulation of the system over a given time horizon, and as a callable function that takes a given state-action pair and returns an updated state, given an ordering rule for the entities in the system. The order rule utilized by each entity in the supply chain itself is modular, and is able to generate environments with agents based on multiple models from prior work, notably (Croson & Donohue, 2006; Oliva et al., 2022; J. Sterman, 1989; J. D. Sterman & Dogan, 2015) and also classical and fully rational base-stock replenishment strategies (Clark & Scarf, 1960).

While the framework developed here was purposely designed to be modular, allowing for different assumptions of agent and player decision rules to be tested, this paper focuses on agent response in the context of other entities utilizing the ordering heuristic introduced in (J. Sterman, 1989) and (Martin et al., 2004), and summarized in expressions (2) and (3) below:

$$O_t = MAX\big(0, \hat{L}_t + \alpha_S(S' - S_t - \beta \, SL_t) + \varepsilon_t\big) \tag{2}$$

$$where \ \hat{L}_t = \ \theta L_t + (1 - \theta)\hat{L}_{t-1} \tag{3}$$

In the above, *O* is the order placed at time *t* given the information observed in the right-hand side of the above expression. In that expression $\hat{L}$ is a smoothed interpolation of the expected outflow of inventory, subject to a smoothing parameter *θ*. *SL* refers to the total inbound supply line of inventory heading towards the player. *S* is the current on-hand inventory (or stock), and *S'* is a parameter that can be considered analogous to the desired or goal on-hand inventory of the player. Thus, we have an expression with four parameters: θ, α, β, and S'. As conceptualized in (J. Sterman, 1989), the above parameters are bounded as $0 \leq \theta, \alpha, \beta \leq 1$ and $0 \leq S'$, and that paper also provides fitted values for expressions (2) and (3) for a set of real human teams, along with a set of parameter values that best fit the overall behavior of all teams.

Furthermore, ordering data from three recent real-world runs of the Beer Game in Fall of 2021 and Spring of 2022 were collected and used to fit additional models of teams assuming

the ordering heuristics shown in expressions (2) and (3). Combined with the original results printed in (J. Sterman, 1989), this yields 49 total models of human ordering behavior in this simulated multi-echelon supply chain based on real observed ordering and provides the testing bed for subsequent agent development and optimizations presented in this paper.

For the cost-reducing agent itself, three different fundamental architectures are considered and compared below, ranging from the most conceptually and structurally simple to the most complex:

1. 'Single-Shot' Cost Reducing Agents – These agents operate by fixing their ordering decision rule *a priori*, either by fixing some generally cost-reducing value of the four parameters of expressions (2) and (3) (e.g. 'behavioral' agents), or by following a base-stock replenishment policy with a fixed order-up to level (e.g. 'base-stock' agents).

2. Model-Predictive 'Learning' Agents – These agents follow a scheme similar to that found in model predictive control literature (Åström et al., 2001; Sutton & Barto, 2014), starting from the assumed cost reducing rule utilized by the 'single-shot' agents defined above, but then adjusting over time based on observations of the evolution of the environment

3. Deep Q-Network Agents – These agents are trained using reinforcement learning, which at its core features a 'perception-action-learning' loop (Sutton & Barto, 2014) in which a model-free agent interacts repeatedly with its environment in order to construct an action policy that maximizes received rewards.

Each of the above architecture is described in more detail below:

'Single-Shot' Cost Reducing Agents

For the 'single-shot' agents, costs for the entire team of four entities are reduced by fixing the ordering parameters of all entities save one (the agent) in a nested set of these expressions and finding a set of parameters for the remaining entity that are cost reducing. This routine is illustrated in Figure 2.
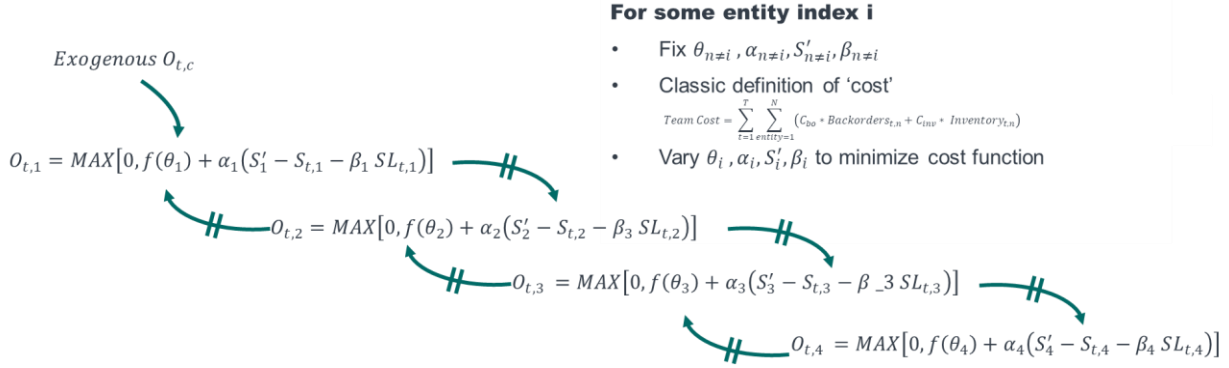
For some entity index i

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

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

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

Exogenous $O_{t,c}$

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

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

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

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

Figure 2. Cost Minimization Routine for the Model-Based Approach

The general learned parameters for a horizon of $t = 52$ periods utilizing a bounded BFGS method to reduce costs (Byrd et al., 2005) against the 'average' team reported in (J. Sterman, 1989) and subject to the classic order pattern seen in that paper (a four-period sequence of orders of 4 units each, followed by a step to 8 units ordered per round until he end of the game) is shown in Table 1.

Table 1. Learned Parameters for the Behavioral Single Shot Cost Reducing Agent

Entity Optimized	Fitted Parameters				Total Cost (Inventory-Based)
	θ	α	β	S'	
N/A (baseline)	0.3600	0.2600	0.3400	17.0000	9978.44
0 (Retailer)	0.002	0.409	0.975	29.259	1440.45 (-85.56%)
1 (Warehouse)	1.000	0.495	1.000	36.405	1911.77 (-80.84%)
2 (Distributor)	0.747	0.094	0.784	73.721	3225.41 (-67.68%)
3 (Factory)	1.000	1.000	0.048	21.581	4799.45 (-51.9%)

The above forms the baseline for a single-shot agent assuming behavioral response from the other entities in the supply chain. Alternatively, the agent could assume that all other entities in the supply chain are fully rational, following a simple base-stock ordering policy. In which case, the ideal base-stock value for the agent itself to follow is a function of the information structure and cost structure of the system and the distribution of the input orders. As discussed in some recent prior work, the information and costing structure used here does not perfectly match the criteria needed for direct application of the Clark and Scarf algorithm for optimal base-stock values, but that same work provides a blueprint for obtaining near-optimal values via a grid-search method (Oroojlooyjadid et al., 2021). Using that same method, we can obtain the average most cost reducing base-stock values for this specific model of the supply

8

chain. For the examples used in this paper, the pertinent values were found for the original step-input orders (from 4 to 8 units at an unknown time) used in (J. Sterman, 1989) and for gaussian normally drawn orders (mean orders 10 with standard deviation of 4 units) as used in (Chen & Samroengraja, 2009), with the combination of values that were on average most reducing over 50 draws of the order distribution.

Table 2. Average Cost Reducing Full-Team Base-Stock Values

Input Order Distribution	Reference	Base Stock Values			
		Entity 1	Entity 2	Entity 3	Entity 4
$d_0^t \in C(4,8)$	(J. Sterman, 1989)	5	21	4	2
$d_0^t \in \mathbb{N}(10,4)$	(Chen & Samroengraja, 2009)	0	24	8	5

The values in Table 2 are for a team with static base-stock ordering rules that perform best *on average* for 50 draws of the order distribution. For the specific examples presented below, it is possible to obtain even better performance for specific draws of the order distribution as an even more restrictive estimated upper bound on performance (or here lower bound on cost incurred). Starting from the values determined above, a further grid search of +/- 10 units was done for each of the draws of the order string actually faced by the example teams used in the analyses below to determine an order-string specific estimated lower bound on cost that could have been achieved by a team subject to the same order pattern and following a base-stock replenishment policy.

Model Predictive 'Learning' Agents

These agents build on the structure developed in the 'single-shot' agents above by incorporating a concept of learning. This agent still has a static order structure at its root (either based on a heuristic rule like that of expressions (2) and (3), or based on a base-stock replenishment policy) but iteratively estimates the parameters of an assumed model of its environment (the other players in the supply chain), and then optimizes its own ordering rule parameters over a given horizon. Stated generally via the pseudocode below:

```
t = 0
    Assume Structural and Dynamic Model of System
    Define Agent position in System model
    Define observable space for Agent
    Populate initial parameter assumptions
    Define calibration memory and optimization horizon
```

9

```
for t in 1: horizon
    Calibrate System Model given history
    ArgMin {System Parameter Estimate} |
        Error (Expected space of simulation of System Model, Actual
        obs 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
```

This architecture introduces several new hyperparameters (calibration memory and optimization horizon), and also structural choices around the 'observable space' for the agent. In other words, this raises questions about how information conditions affect the performance of the agent.

For this work, the information conditions are summarized below:

- The 'Minimal' condition, the information available as part of the calibration of the agent is the most restrictive, with the agent only having access to directly verifiable and visible information
    - On-hand inventory position of the agent itself an no others
    - The most recent order placed by the agent itself and no others
- For the 'Low Information' condition, the calibration of the agent is given a slightly larger set of information, including information that it itself is not directly in control of, namely inbound shipments
    - On-hand inventory position of the agent itself an no others
    - Inbound shipments to of the agent itself an no others
    - The orders placed by the agent itself and no others
- For the 'Standard Information' condition, the calibration of the agent is based on information that a human player in the traditional playing of Beer Game could view at any given moment, and matches the information available to the players in used in the datasets in this paper:
    - All on-hand inventory positions of all four entities
    - All inbound shipments to all entities
    - The orders placed by the agent itself and no others
- For the 'High Information' condition, the calibration of the agent is omniscient and has access to effectively every state variable in the system, including those only imputed and not directly observable (e.g., backorder):

- o   All on-hand inventory positions of all four entities
- o   The backorder positions of all entities
- o   All inbound shipments to all entities
- o   The entire order and information flows of all four entities

In the routine described above, the agent must first assume a model of the world. Next the agent uses its available information to fit a best estimate of that model. It then projects forward and optimize its own ordering behavior given that estimated model. Two different combinations of assumed models and agent behavior are used here:

- Base-Stock:
  - o   The agent assumes that the other entities in the supply chain are using base-stock ordering rules, and therefore should use a base-stock ordering rule as well for its own behavior.
  - o   Each time step, the agent fits a window of prior observed information about the system, and its own ordering history, to a model that assumes all other agents are following a base-stock policy, updating an estimate of the base-stock value being sought by those other entities. The agent then optimizes its own base-stock value over a forward horizon that would minimize team costs under its fitted model.
- Behavioral Ordering:
  - o   The agent assumes that the other entities in the supply chain are using an ordering rule that can be described by a behavioral heuristic, in this case the rule from Sterman '89. The agent matches the same structure of this rule for its own response.
  - o   Each time step, the agent fits a window of prior observed information about the system, and its own ordering history, to a model that assumes all other agents are following this heuristic ordering rule, updating an estimate of the parameters in that rule for all other entities that would generate the observed behavior of the system. The agent then optimizes the parameters of its own ordering rule over a forward horizon that would minimize team costs under its fitted model.

Finally, the cost being minimized (or similarly the reward being maximized) by the agent in the traditional configuration of this simulation is based on that of the *entire team* as described in (1). However, it is also reasonable to consider the effect of a fully myopic definition of costs in which the agent attempts to minimize costs for itself alone.

Deep-Q Network Agents

The Beer Game itself has been used as a training environment in reinforcement learning applications, most notably utilizing various modifications of Deep Q-Network architectures, including modifications to allow for independent training across entities utilizing pooled reward schemes (Chaharsooghi et al., 2008; Opex Analytics, 2018; Oroojlooyjadid et al., 2021). The Beer Game as a model of a multi-echelon supply chain presents a challenge to direct application of DQN architecture, challenges that are often also found in real integrated supply chains. Specifically challenges emerge from 1) the true 'full state' of information is unknown to any one entity, 2) rewards are communal and not realized until the end of the time horizon, 3) DQN architectures can be 'over optimistic' in even mildly noisy environments (Thrun & Schwartz, 1993), and perhaps most importantly 4) the current overarching quality of the system, e.g. whether bullwhip is in progress or if the supply chain is stable, matters almost as much if not more than any specific action, and

In order to address the above issues, most notably the final point in the above list, this work presents a DQN architecture for use in multi-echelon supply chains like the Beer Game that has the following general architecture: 1) An 'order-plus' action space (Oroojlooyjadid et al., 2021) which both allows for unbounded ordering in absolute terms and follows from observations in the model-based approach above, 2) a dual DQN network (Z. Wang et al., 2016) that separately maintains a value function estimation for both the current overarching combined state of the system and separately for each action, 3) an observation space defined over a window of prior state observations corresponding to the signal delay in the system, 4) a combination of epsilon-greedy and Boltzmann exploration policies (Wiering, 1999), and finally 5) three sequential dense layers with ReLu activations of 256, 128, and 64 free parameters respectively for a total of 448 free parameters.

The environment itself is built on the same framework utilized in the other architectures described above, with the functionalized form of the Beer Game translated into the commonly used opensource Gym research framework developed by OpenAI (Brockman et al., 2016). This environment allows for training against all positions in the simulated supply chain against randomly bootstrapped assemblages of human-like players, whose ordering rules are drawn from classical supply chain literature (J. Sterman, 1989). Additionally, this agent can be trained against random (but bounded) simulation horizons to avoid over-learning endgame-dependent policies, and even noisy realizations of ordering decisions. An illustration of this framework is shown in Figure 3.
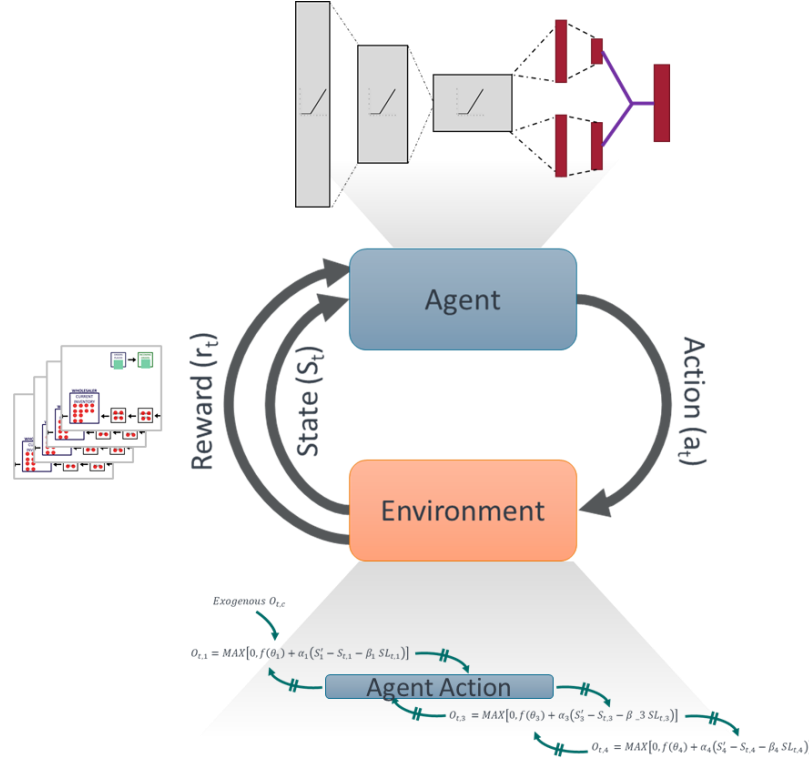
Figure 3. Cost Minimization Framework for the Model-Free DQN Approach

The above structure is largely model-free in the same sense as the 'single-shot' agents described above in so much as the agent is unaware of its surroundings, but rather tunes a set of free parameters ahead of time that allow it to map a current observed state to a desired action. This DQN can be further improved by developing a model aware DQN structure, which is identical to the DQN described above, but in addition to the base set of observations about its environment also incorporates estimates of the ordering parameters of the other agents in the supply chain. In this manner, this model-aware DQN can then be used directly in the model-predictive learning structure described above, with the calibration based on history sill occurring, but then this estimate of the system being used as a state input into the DQN.

Full Design of Experiment

Table 3 takes the features expanded upon above and compactly displays them along with a numbering scheme to describe each experimental feature. For each experiment, 49 runs (one for each simulated team estimated from either prior literature or recent real-world runs of the Beer Game) were conducted.

Table 3. Conditions with the Design of Experiment

(1) Rule Complexity	(2) Degree of Adaptability	(3) Incentive Structure	(4) Information Availability
(.1) Base Stock Ordering	(.1) Single-Shot	(.1) Self (Myopic)	(.1) Minimal
(.2) Behavioral Ordering	(.2) Model-Predictive Learning	(.2) Team (Non-Myopic)	(.2) Low
(.3) Deep-Q Network			(.3) Standard
			(.4) High

Note that while a full factorial experiment would consist of 3x2x2x4 = 48 different combinations of feature choices seen in Table 3, not all combinations are sensible. After level 2 – Degree of Adaptability, levels 3 and 4 only apply to the Model-Predictive Learning agents. A 'single-shot' agent as described in the sections above is pre-trained with a fixed ordering rule (either base-stock or behaviorally based) and thus there is no optimization nor calibration to perform each period and the incentive structure and information availability has no effect on the agent. Additionally, and as discussed in more detail in the Appendix that accompanies this article, while the Deep-Q Network could be configured to explore all subsequent levels of features, its performance is essentially the same independent of feature choices. Once sufficient training as occurred on the Deep-Q Network to achieve cost reducing performance, the change performance as a function of feature space as seen here is insignificant. Thus, the DQN is reserved here as pseudo-upper-baseline on complexity and specific feature choices are largely abstracted, with only a 'single-shot' and single 'model-predictive learning' run explored in the results section below. This is to avoid focusing too much on features of the DQN architecture itself, which not the primary focus of this work, and concentrate on differences induced from the other feature combinations of the full design.

Thus, 20 of the possible 48 full conditions are actually explored in this work. 16 are the full factorial expansion of the Model-Predictive Learning agents, with base stock and behavioral learning, self and team incentives, and all four information levels. In addition are two conditions for the base-stock and behavioral single shot agents, and two conditions for the Deep-Q Network agent. Table 4 gives all 20 conditions, using the same numbering scheme as introduced in Table 3.

Table 4. Fully Enumerated Experimental Conditions

Run Code	Agent Feature Description	Run Code	Agent Feature Description
1.1 - 2.1	Base Stock Ordering, Single-Shot	1.1 - 2.2 - 3.2 - 4.3	Base Stock Ordering, Model-Predictive Learning, Team (Non-Myopic), Standard
1.2 - 2.1	Behavioral Ordering, Single-Shot	1.1 - 2.2 - 3.2 - 4.4	Base Stock Ordering, Model-Predictive Learning, Team (Non-Myopic), High
1.3 - 2.1	Deep-Q Network, Single-Shot	1.2 - 2.2 - 3.1 - 4.1	Behavioral Ordering, Model-Predictive Learning, Self (Myopic), Minimal
1.3 - 2.2	Deep-Q Network, Model-Predictive Learning	1.2 - 2.2 - 3.1 - 4.2	Behavioral Ordering, Model-Predictive Learning, Self (Myopic), Low
1.1 - 2.2 - 3.1 - 4.1	Base Stock Ordering, Model-Predictive Learning, Self (Myopic), Minimal	1.2 - 2.2 - 3.1 - 4.3	Behavioral Ordering, Model-Predictive Learning, Self (Myopic), Standard
1.1 - 2.2 - 3.1 - 4.2	Base Stock Ordering, Model-Predictive Learning, Self (Myopic), Low	1.2 - 2.2 - 3.1 - 4.4	Behavioral Ordering, Model-Predictive Learning, Self (Myopic), High
1.1 - 2.2 - 3.1 - 4.3	Base Stock Ordering, Model-Predictive Learning, Self (Myopic), Standard	1.2 - 2.2 - 3.2 - 4.1	Behavioral Ordering, Model-Predictive Learning, Team (Non-Myopic), Minimal
1.1 - 2.2 - 3.1 - 4.4	Base Stock Ordering, Model-Predictive Learning, Self (Myopic), High	1.2 - 2.2 - 3.2 - 4.2	Behavioral Ordering, Model-Predictive Learning, Team (Non-Myopic), Low
1.1 - 2.2 - 3.2 - 4.1	Base Stock Ordering, Model-Predictive Learning, Team (Non-Myopic), Minimal	1.2 - 2.2 - 3.2 - 4.3	Behavioral Ordering, Model-Predictive Learning, Team (Non-Myopic), Standard
1.1 - 2.2 - 3.2 - 4.2	Base Stock Ordering, Model-Predictive Learning, Team (Non-Myopic), Low	1.2 - 2.2 - 3.2 - 4.4	Behavioral Ordering, Model-Predictive Learning, Team (Non-Myopic), High

3. Results

The results below are reported for the cost reduction achieved by an agent placed into the wholesaler position (position 2 of 4) in the 49 simulated supply chains as described above, and subject to a normally drawn input order signal of mean 10 and standard deviation 4 in a manner similar to (Chen & Samroengraja, 2009). These same analyses were performed with the classic step input signal from (J. Sterman, 1989) with similar results. Position 2 in the supply chain was chosen for reporting here because from preliminary exploration, this position is the most decoupled from its own actions in the supply chain, as partially indicated by the significantly higher base-stock values versus other positions derived in the single-shot optimization task shown in Table 2. This is also supported by recent work in multi-echelon supply chains (such as Moritz et al., 2021; Paine, 2020) that have shown that the extreme ends of the supply chain

(here position 1 – the 'retailer', and position 4 – 'the factory' as described in Figure 1) have a disproportionate influence on the evolution of costs in this supply chain structure.

This paper does not aim to reinforce this already well discussed observation, but rather emphasize the features of agents that are most cost reducing with no direct influence on the ordering behavior of the other entities in the supply chain. Placing the agents developed here in position 2, the wholesaler, serves to isolate the influence of feature choices on cost reduction away from the influence of the agent by virtue of its position in the supply chain alone. Additionally, this simulated supply chain is four positions long by virtue of being based on a real-world supply chain learning environment, the Beer Game. In reality, the supply chain of a specific good could be made up of any number of sequential entities, again only two having the unique feature of being at terminal ends. Thus, while the results below are directionally similar at other positions in the supply chain, they are most meaningful, and possibly of the most value to real supply chain managers, when considered at a central position such as the wholesaler.

Baseline Behavioral Responses Greatly Under-Perform vs Base-Stock Responses

Perhaps unsurprisingly the costs incurred for the baseline simulated behaviorally ordering teams were significantly higher than that which would have been incurred by a similar supply chain in which all four entities were following the average cost-reducing base-stock replenishment policy as in Table 2, and much higher than those following an estimated idealized base-stock policy for the specific order pattern each team faced. As seen in Figure 4, the median cost incurred by the 49 simulated behavioral teams was 15,408 while for the static base-stock team exposed to the same 49 input order patters experienced a median cost of 6,419, and for the case of order-string specific base stock rules a median cost of 3,211.
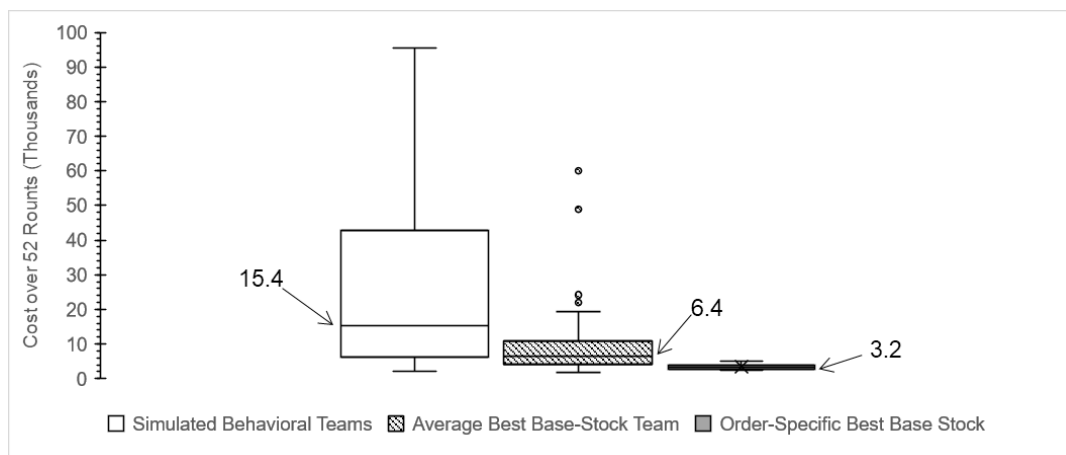

Figure 4. Baseline Simulated Costs of Behavioral Teams vs Base-Stock Teams

However, the purpose of this research is not to simply restate that the behavioral responses identified in the opening paragraphs of this paper result in poorer performance than an optimal decision rule. Rather, this work *assumes* that the other entities in supply chains are acting in a behavioral manner and asks what features of an agent placed into such an environment can help reduce costs overall. Thus, the results in Figure 4 provide two different baselines of comparison for any such agent. While the full team of base-stock agents provides a realistic floor of costs, the simulated behavioral teams do not necessarily represent a true upper bound on costs. Indeed, one or two teams within that sample perform reasonably similarly to the base-stock teams even with simulated behavioral response rules. Therefore, it is reasonable to expect that an agent placed into these already well performing teams could be *destabilizing* and introduce additional costs.

For the remaining results presented below, the performance of each agent architecture is compared relative to the performance of the baseline of the simulated behavioral team (in which it could either reduce costs or increase costs via its presence), rather than to the full base-stock team (in which it would likely still have higher costs as it is in a simulated supply chain of behaviorally ordering entities, just to different degrees). Initially, this comparison is done only for those cases in which the agent is not destabilizing, and then later those cases in which introducing the agent introduces more costs are investigated in more detail.

One of the key differences between each of the above-described architectures is the degree of complexity of the agent. For the 'single-shot' agents, their parameters are determined *a priori* and held fixed over the course of the simulation. For the base-stock 'single-shot' agent this means that agent has only one parameter, the fixed desired safety-stock, while for the behavioral version of this same agent, the number of parameters is equal to the number of free parameters in the behavioral heuristic rule, which for this paper is the four from expressions (2) and (3). For the model-predictive learning agents, the number of free parameters is slightly more difficult to enumerate as the agent is updating the value of the underlying parameters determining its order response each period. For a simple approximation of complexity, we can treat each of these decision points as another set of parameters. Finally, for the DQN, under either architecture, either aware of its surroundings or not, the number and value of free parameters is fixed *a priori* like in the 'single-shot' case, but much larger. For the specific DQN architecture used here, 448 parameters in total across three layers.

Figure 5 compares the mean cost reduction of each architecture type, when not destabilizing, as a function of complexity. Note the logarithmic scale for the x-axis and that the

numbering on the labels corresponds to the experimental condition codes from Table 4. Also note that while in general increasing complexity yields lowered average costs incurred, the three general subclasses of architectures (base-stock, behavioral, and DQN) are offset from one another on this graph. Furthermore, while the DQN method can achieve lower costs than the other methods it does so at the additional expense much greater complexity, especially compared to much simpler single-shot methods that, at this first level of analysis, appear to achieve comparable performance.

This leads to the next level of analysis, in which these simpler (non-DQN) methods are further investigated in order to tease out agent features that yield these improvements.
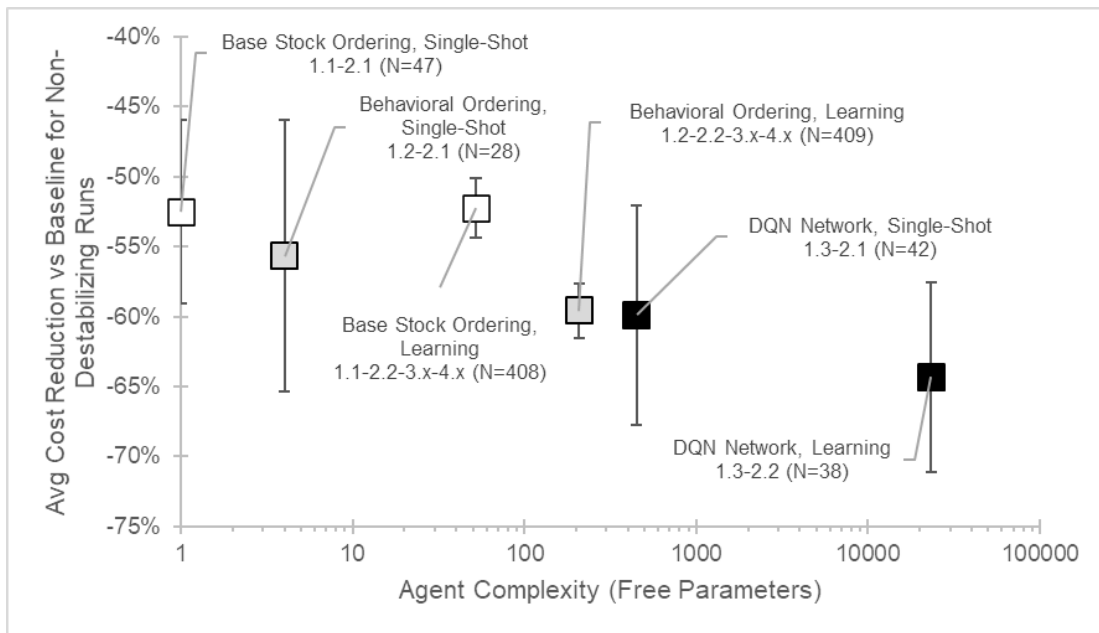

Figure 5. High-Level Comparison of Performance of Agent Features vs Complexity

Comparison of Agent Architectures and Information Availability States

As discussed above, we compare first an agent that is 'single-shot' and placed into the supply chain in Positions 2 (or the Wholesaler role as seen in Figure 1) that either assumes that the environment in which it is placed is populated by other entities acting using either a behavioral or base-stock ordering rule, and responding in kind. Next, we consider the influence of incorporating a model-predictive learning structure (with backward calibration memory of 10 time steps and a forward optimization horizon of 30 time steps). And finally, the influence of both choices together. Table 5 shows a series of regression models that predict the degree of cost reduction achieved versus the baseline simulated behavioral team as a function of these

18

architecture feature choices. As all the independent variables here are factors, the summation of the regression coefficients in each of these regressions represents the predicted percent cost reduction of an agent with all the features used in the model, while the constant indicates the predicted percent cost reduction of the architecture being compared to.

Model (1) in Table 5 shows the result of choosing a behavioral model versus a base-stock model for the agent, independent of if the agent was using a model-predictive learning approach or a static 'single-shot' approach.

$$Model\ (1): \frac{Cost_{Agent_{Team_i}} - Cost_{Baseline\ Team_i}}{Cost_{Baseline\ Team_i}} = \beta_0 + \beta_1 f_{Behavioral} + \epsilon_{Team_i} \qquad (4)$$

The constant is statistically significant and predicts an average reduction in cost of 52% by simply using a static base-stock agent. In other words, when these agents are not destabilizing, the static base-stock policy on average is able to half the cost incurred by the supply chain. The shift to an agent that correctly assumes its environment has behaviorally ordering others (vs the base-stock) but remains static in its response further reduces the cost on average by an additional 7.4%.

Model (2) in Table 5 now considers just the influence of incorporating dynamic learning into the agent. Here, the alternative formulation captured by the constant, which is again significant at 49% cost reduction, is having any agent, behavioral or base-stock but static and single-shot, in the supply chain. Adding the structure necessary to dynamically update the parameters of the ordering rule used by the agent as it learns a model of its environment further reduces the costs incurred by 7.2%.

$$Model\ (2): \frac{Cost_{Agent_{Team_i}} - Cost_{Baseline\ Team_i}}{Cost_{Baseline\ Team_i}} = \beta_0 + \beta_1 f_{learning} + \epsilon_{Team_i} \qquad (5)$$

Finally, Model (3) in Table 5 provides the model with both features: behavioral modeling (versus base-stock modeling), and dynamic model-predictive learning (versus static 'single-shot'), while Model (4) continues this with the full model plus the interaction term. a static base-stock agent in the supply chain is *still* able to cut costs in half on average.

$$Model\ (3): \frac{Cost_{Agent_{Team_i}} - Cost_{Baseline\ Team_i}}{Cost_{Baseline\ Team_i}}$$
$$= \beta_0 + \beta_1 f_{Behavioral} + \beta_2 f_{learning} + \epsilon_{Team_i} \qquad (6)$$

$$Model \ (4): \frac{Cost_{Agent \ Team_i} - Cost_{Baseline \ Team_i}}{Cost_{Baseline \ Team_i}} \tag{7}$$

$$= \beta_0 + \beta_1 f_{Behavioral} + \beta_2 f_{learning} + \beta_3 \left(f_{Behavioral} * f_{learning} \right) + \epsilon_{Team_i}$$

Model (3) and Model (4) carry forward the results of Models (1) and (2), showing that having a behavioral *and* learning agent is significantly cost reducing versus a static base-stock agent. However, and perhaps of most significance, is that a placing. Moreover, while both feature effects are statistically significant in Model (3), the influence of having a behavioral agent is both more significant and larger in magnitude, which is then reinforced by the interaction term in Model (4).

Table 5. Feature Influence: Behavioral and Learning Agent

	Dependent variable:			
	Performance Improvement over Baseline			
	(1)	(2)	(3)	(4)
Behavioral Agent	-0.074***		-0.071***	0.088
	(0.018)		(0.018)	(0.061)
Learning Agent		-0.072**	-0.060*	0.003
		(0.031)	(0.031)	(0.039)
Behavioral x Learning				-0.173***
				(0.064)
Constant	-0.522***	-0.494***	-0.469***	-0.525***
	(0.012)	(0.030)	(0.030)	(0.037)
Observations	817	817	817	817
Log Likelihood	-32.62	-38.62	-30.71	-27.06
McFadden R^2	0.210	0.064	0.256	0.324
Residual Std. Error	0.252 (df = 815)	0.254 (df = 815)	0.252 (df = 814)	0.251 (df = 813)
F Statistic	17.436*** (df = 1; 815)	5.291** (df = 1; 815)	10.651*** (df = 2; 814)	9.588*** (df = 3; 813)

Note:	*p<0.1; **p<0.05; ***p<0.01

The regression results in Table 5 illustrate in part that having an agent that dynamically learns its environment and optimizes forward can be further cost reducing. As discussed in the Methods section above, this model-predictive learning agent introduces the opportunity to explore how both the choice of reward (myopic focus on self-cost reduction or focus on total team cost reduction) and on information space (minimal versus high, near omniscient) can affect the performance of the system. Prior literature on model predictive control has the embedded assumption that additional information about the state space can only improve outcomes via a more accurate representation of the 'reality' of the system by implying that the main tradeoff in these schemes is between computational time and system performance (Mayne, 2014; Seborg et al., 2016).

Table 6 shows a series of regression models expanding on the models seen in Table 5. Here, the agent is a model-predictive learning agent. In other words, the feature of 'Learning' is assumed, and the data is filtered to only include experiments that utilized a model-predictive learning agent. Now the features under investigation relate to the structural in and information availability choices of that learning agent. that predict the degree of cost reduction achieved versus the baseline simulated behavioral team as a function of these architecture feature choices. For all three sub-models, the comparison between a base-stock architecture and a behavioral architecture is maintained and independent of other feature choices the influence of choosing a behavioral architecture remains significant and cost reducing.

However, what is most striking about the regression results in Table 6 is the *lack* of significance of other feature choices. In Model (5) and the full Model (7), the influence of moving towards a non-myopic reward (e.g., having an agent that is attempting to reduce the cost for the entire supply chain and not just itself), while of the correct sign is insignificant. Similarly, the difference in moving from a hyper limited set of information for the calibration of the model the agent is using towards a near omniscience set of information is insignificant.

$$Model\ (5): \left.\frac{Cost_{Agent_{Team_i}} - Cost_{Baseline\ Team_i}}{Cost_{Baseline\ Team_i}}\right|_{Learning} \tag{8}$$

$$= \beta_0 + \beta_1 f_{Behavioral} + \beta_2 f_{NonMyopic} + \epsilon_{Team_i}$$

$$Model\ (6): \left.\frac{Cost_{Agent_{Team_i}} - Cost_{Baseline\ Team_i}}{Cost_{Baseline\ Team_i}}\right|_{Learning} \tag{9}$$

$$= \beta_0 + \beta_1 f_{Behavioral} + \beta_2 f_{LowInfo} + \beta_2 f_{StandardInfo} + \beta_2 f_{HighInfo} + \epsilon_{Team_i}$$

$$Model\ (7): \frac{Cost_{Agent_{Team_i}} - Cost_{Baseline\ Team_i}}{Cost_{Baseline\ Team_i}}\Bigg|_{Learning}$$

$$= \beta_0 + \beta_1 f_{Behavioral} + \beta_2 f_{NonMyopic} + \beta_3 f_{LowInfo} + \beta_4 f_{StandardInfo}$$

$$+ \beta_5 f_{HighInfo} + \epsilon_{Team_i}$$

(10)

This is surprising as these results, taken in combination with the more general models summarized in Table 5, indicate that having a behavioral learning agent that is totally focused on its own cost reduction and having only the most narrow understanding of its environment is sufficient to achieve most of the cost savings seen versus the baseline of full behavioral ordering. Note that the full regression with all interactions was also performed (and is in the Appendix to this article), but no interaction terms were significant and thus only the main effects are reported here.

Table 6. Feature and Information State Influence within a Learning Agent

	Dependent variable:		
	Performance Improvement over Baseline		
	(5)	(6)	(7)
Behavioral Agent	-0.084***	-0.085***	-0.084***
	(0.018)	(0.018)	(0.018)
Non-Myopic Agent	-0.028		-0.028
	(0.018)		(0.018)
Low Information		-0.016	-0.016
		(0.026)	(0.026)
Standard Information		-0.018	-0.019
		(0.026)	(0.026)
High Information		-0.017	-0.017
		(0.026)	(0.026)
Constant	-0.508***	-0.509***	-0.495***
	(0.016)	(0.021)	(0.023)
Observations	744	744	744
Log Likelihood	-23.04	-23.91	-22.72
McFadden R^2	0.344	0.313	0.345
Residual Std. Error	0.250 (df = 741)	0.251 (df = 739)	0.250 (df = 738)
F Statistic	11.870*** (df = 2; 741)	5.481*** (df = 4; 739)	4.865*** (df = 5; 738)
Note:			*p<0.1; **p<0.05; ***p<0.01

This model-predictive learning agent can perform this cost reduction not necessarily by perfectly learning the ordering rules that govern its environment, but rather by simply having a model to learn in the first place. Figure 6 shows a measure of the error between the estimation of the environment learned by the agent and the actual environment ordering rules. As seen in the top of this figure, even when exposed to near perfect information about its environment, the agent may eventually learn a near accurate representation of its environment. However, when compared to the minimal information case as seen on the bottom of that same figure, the minimal case is both more likely to learn an incorrect representation of the environment but also is significantly less stable in its underlying model. However, even with this unstable and less accurate model of its surroundings, this minimal information case with a relative erroneous model of the environment *still* can perform statistically similarly to the near total information case.
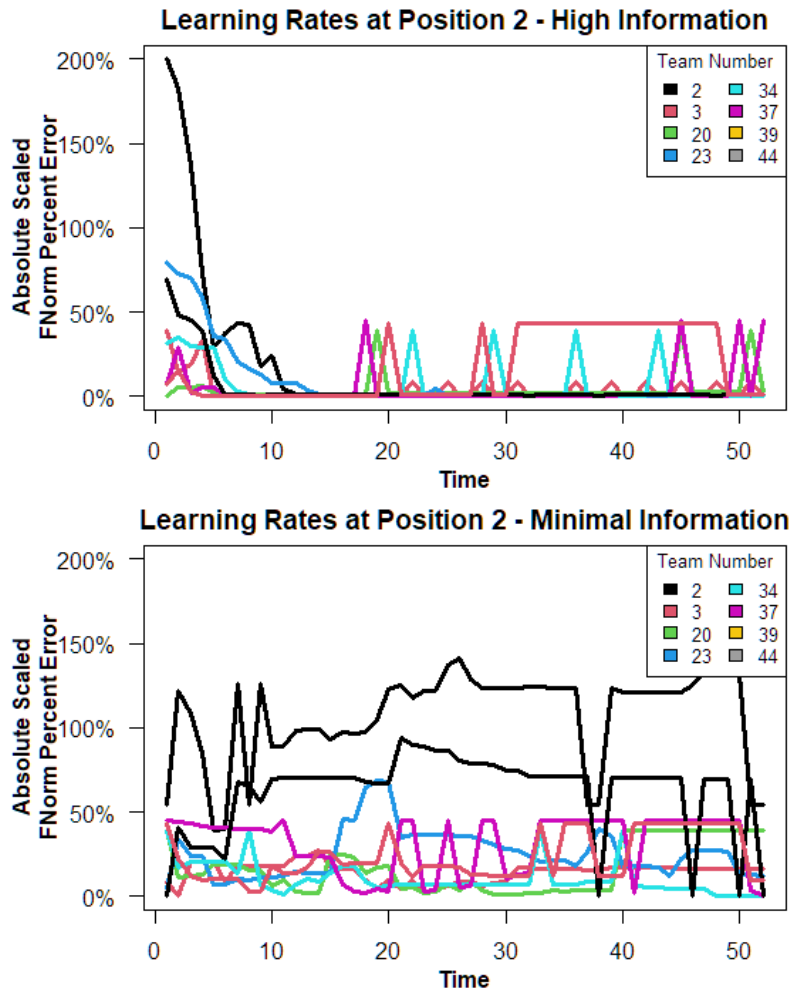

Figure 6. Sample of Learning Rates for Model-Predictive Learning Agents

Stability of Different Architecture Choices

The above analyses about the influence of different agent architectures and information choices were all based on those runs in which the presence of the agent was not destabilizing. In other words, of the 882 total runs (49 runs each for each across 18 different feature and information configurations), 65 total resulted in *higher* costs being realized with the presence of the agent versus the baseline performance of the simulated team, resulting in the *N = 817* seen in Table 5. Similarly, among the subset of 784 runs that had model-predictive learning features, 40 were destabilizing resulting in the *N = 744* seen in Table 6.

While the regression results for the models in Table 5 and Table 6 imply that there is no statistically significant cost reduction from either considering myopic or team-wide costs, or from varying the level of information available to the agent, this is conditioned on the agent already being able to achieve some cost reduction (or at least cost-parity) to begin with.

Table 7 and Table 8 show the fraction of the 784 runs in each condition category of model-predictive learning agents that were destabilizing, resulting in more incurred costs versus the baseline simulated teams. While the absolute numbers of stabilizing runs are low (no more than 5 of 49 runs in each condition), it is nevertheless interesting to observe that generally higher information availability results in lower occurrence of destabilizing runs. This is especially true for non-myopic agents that endeavor to reduce costs across the entire supply chain. For myopic agents focused only on their own cost reduction, there is an interesting trade-off implied among the lowest information availability states. The lowest information state, which as defined above is only the on-hand inventory of the agent and its own placed orders, is less likely to be destabilizing than providing a small additional amount of information in the calibration process (specifically the inbound shipments from the agent's supplier). This countertrend is only represented by a single difference in occurrence and thus cannot be said to be significant but should be noted for future inquiry.

Table 7. Percent Destabilizing for Myopic Agents

		Information State			
		Minimal	*Low*	*Standard*	*High*
Agent Type	*Base-Stock (Myopic Agent)*	6.1%	8.2%	6.1%	6.1%
	Behavioral (Myopic Agent)	8.2%	10.2%	0.0%	0.0%

Note: *N = 49 for all conditions*

Table 8. Fraction Destabilizing for Non-Myopic Agents

		Information State			
		Minimal	*Low*	*Standard*	*High*
Agent Type	*Base-Stock (Non-Myopic Agent)*	10.2%	10.2%	10.2%	6.1%
	Behavioral (Non-Myopic Agent)	0.0%	0.0%	0.0%	0.0%

Note: *N = 49 for all conditions*

4. Discussion

The most striking outcome of the results above is that simply having *any* agent of the types described here has a similar level of cost reduction to more complex control schemes, including the relatively naïve agents described above that simply follow a fixed order response policy (albeit a policy based on the best fully rational base-stock assumptions). This follows from the related work in psychology and behavioral economics referenced in the introduction to this paper that have noticed that consistent and simple policies can often perform as well as more complex ones in that space. As stated by others, Behavioral Operations Management "requires an operations context" while also acknowledging the presence of "potentially non-hyper-rational actors in [that] operational context" (Croson et al., 2013). Thus, it should not be surprising that these results, placed in a realistic operational context, follow similar observations from other fields in which non-hyper-rational decision makers exist.

Furthermore, that the policies that assumed behavioral responses performed better in general than those that assume base-stock responses from their fellow entities in the supply chain is not surprising. At minimum, this is a result of simply better matching the underlying system in which the agent is placed. While not included in this paper for compact presentation, it should be noted that even the behavioral-based agent, when also combined with model-predictive learning structures, is able to perform well when placed in an environment with perfectly rational base-stock other agents. This is because such an agent, while still assuming behavioral responses from the environment, can adapt quickly.

Stated differently and as seen in the results above, even learning a 'wrong' model of the environment can result in improved performance. However, as stated before in this article the focuses of Behavioral Operations Management as a field are policies that exist in an operational context that included non-rational actors, and thus it may be safer to assume behavioral responses in a multi-echelon supply chain context with delays and imperfect information and then correct, rather than assuming the reverse.

While secondary to the points above, this work also builds on recent work illustrating the application of DQN structures in a multi-echelon supply chain setting by leveraging insights developed from directly applying similar structure to the Beer Game (notably Chaharsooghi et al., 2008; Oroojlooyjadid et al., 2021), but while also leveraging a dueling architecture to overcome some of the computational difficulties previously faced. However, while this is of methodological interest, the greater contribution of this paper is the observation that significantly simpler and more directly interpretable polices can achieve similar levels of performance to these more complex methods.

In addition to expanding on the DQN methods described above, and as part of the simpler architectures explored here, this work also introduces a method to incorporate model-predictive control architectures more directly into policies that exist in a behavioral context. The results here show that such model-predictive learning architectures does result in better performance than the simplest single-shot methods, though this improvement is secondary to the initial improvement received simply by having a stable policy to begin with.

'Stable' here does not mean 'static'. The 'one-shot' architecture is both stable and static, while the model-predictive architecture is stable but dynamic. Stable in this context refers to consistent application of a rule to transform a set of observed inputs into a given decision or output. The 'one-shot' agent defined in this article has a pre-determined rule, but one that is follows consistently as the system progresses. The model-predictive learning agent similarly has an assumption that a stable response rule will reduce costs, but not that it necessarily knows that rule a priori.

Of additional interest in this model-predictive context is that the traditional tradeoff of computational time from better model fidelity and performance outcomes may not be as stark as previously implied in other MPC applications. Here, even the most minimal information about the state of the system was adequate to achieve a flexible enough policy that could adapt and shift over time in response to a changing environment.

In part this is because the system itself is nearly fully defined by the small amount of information to which the agent has access. There are no losses of physical material, and the agent has no direct control over how the other entities in the system will respond to orders and shipments, only respond to how their own on-hand inventory is changing over time as a function of their order signals. What more information about the state of the system does do is allow the agent to settle on a stable model estimating their environment more quickly.

The only place this tradeoff between model accuracy and overall performance, conditional on information availability, occurs is in the number of instances in which the agent is destabilizing. In the analyses presented here, such destabilizations were still rare, though did occur more frequently with both less information about the overall state of the system and greater divergences between the assumed model and the underlying reality, and between the goal of the agent and the goal of the actual cost structure. Thus, the tradeoff here is not represented in terms degrees of performance and computational cost, but rather between external validity of the model-predictive process and absolute risk.

Specifically, to what degree is the agent limited to some physical or behavioral reality in a real system? Does the agent have access to the entire state of the environment, or just a more siloed view? For a fully integrated supply chain with entities held to a performance standard defined by overall team outcomes, then perhaps the more near-omniscient agent with global goals is realistic. However, for an agent in a supply chain that is not integrated and rather simply part of a value-add chain of independent organizations, then the locally focused 'greedy' goal with very limited information maybe the only realistic option. What is interesting here is that it is possible, in fact likely, to still be able to implement a model-predictive learning policy in this most restrictive context and still achieve cost reductions versus no policy. This work shows that the agent can be locally focused and achieve global benefits, generally, but not always. The more information sharing and the more global the goal, the less likely the risk for a destabilizing outcome.

5. References

Åström, K., Albertos, P., Blanke, M., Isidori, A., Schaufelberger, W., & Sanz, R. (Eds.). (2001). *Control of Complex Systems*. Springer London. https://doi.org/10.1007/978-1-4471-0349-3

Bamakan, S. M. H., Malekinejad, P., Ziaeian, M., & Motavali, A. (2021). Bullwhip effect reduction map for COVID-19 vaccine supply chain. *Sustainable Operations and Computers*, *2*, 139. https://doi.org/10.1016/J.SUSOC.2021.07.001

Bendoly, E. (2013). Real-time feedback and booking behavior in the hospitality industry: Moderating the balance between imperfect judgment and imperfect prescription. *Journal of Operations Management*, *31*(1–2), 62–71. https://doi.org/10.1016/j.jom.2012.06.003

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). *OpenAI Gym*.

Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. (2005). A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific Computing*, *16*(5), 1190–1208. https://doi.org/10.1137/0916069

Chaharsooghi, S. K., Heydari, J., & Zegordi, S. H. (2008). A reinforcement learning model for supply chain ordering management: An application to the beer game. *Decision Support Systems*, *45*(4), 949–959. https://doi.org/10.1016/j.dss.2008.03.007

Chen, F. (1999). Decentralized supply chains subject to information delays. *Management Science*, *45*(8), 1076–1090. https://doi.org/10.1287/mnsc.45.8.1076

Chen, F., & Samroengraja, R. (2009). The Stationary Beer Game. *Production and Operations Management*, *9*(1), 19–30. https://doi.org/10.1111/j.1937-5956.2000.tb00320.x

Ciocan, D. F., & Farias, V. (2012). Model Predictive Control for Dynamic Resource Allocation. *Mathematics of Operations Research*, *37*(3), 501–525. https://doi.org/10.1287/moor.1120.0548

Clark, A. J., & Scarf, H. (1960). Optimal Policies for a Multi-Echelon Inventory Problem. *Management Science*, *6*(4), 475–490. https://doi.org/10.1287/mnsc.6.4.475

Croson, R., & Donohue, K. (2006). Behavioral causes of the bullwhip effect and the observed value of inventory information. *Management Science*, *52*(3), 323–336. https://doi.org/10.1287/mnsc.1050.0436

Croson, R., Donohue, K., Katok, E., & Sterman, J. (2014). Order stability in supply chains: Coordination risk and the role of coordination stock. *Production and Operations Management*, *23*(2), 176–196. https://doi.org/10.1111/j.1937-5956.2012.01422.x

Croson, R., Schultz, K., Siemsen, E., & Yeo, M. L. (2013). Behavioral operations: The state of the field. *Journal of Operations Management*, *31*(1–2), 1–5. https://doi.org/10.1016/j.jom.2012.12.001

Cui, T. H., & Wu, Y. (2018). Incorporating Behavioral Factors into Operations Theory. In *The Handbook of Behavioral Operations* (pp. 89–119). John Wiley & Sons, Inc. https://doi.org/10.1002/9781119138341.ch3

Dawes, R. M., & Corrigan, B. (1974). Linear models in decision making. *Psychological Bulletin*, *81*(2), 95–106. https://doi.org/10.1037/h0037613

Ellram, L. M. (2010). Introduction to the forum on the bullwhip effect in the current economic climate. *Journal of Supply Chain Management*, *46*(1), 3–4. https://doi.org/10.1111/j.1745-493X.2009.03178.x

Evenett, S. J. (2021). *How big of a vaccine surplus will the US have?* Brookings Institution Reports2. https://www.brookings.edu/blog/future-development/2021/05/04/how-big-of-a-vaccine-surplus-will-the-us-have/

Gino, F., & Pisano, G. (2008). Toward a theory of behavioral operations. *Manufacturing and Service Operations Management*, *10*(4), 676–691. https://doi.org/10.1287/msom.1070.0205

Größler, A., Thun, J. H., & Milling, P. M. (2008). System dynamics as a structural theory in operations management. *Production and Operations Management*, *17*(3), 373–384. https://doi.org/10.3401/poms.1080.0023

Hockett, M. (2020). The Pandemic's Bullwhip Effect on Food Manufacturers' Inventory. *Food Manufacturing*.

Huang, T., Allon, G., & Bassamboo, A. (2013). *Bounded Rationality in Service Systems. 15*(2), 263–279. https://doi.org/10.1287/msom.1120.0417

Johnson, B. (2021). In an Age of Abudance, Why do People Starve? *MIT Technology Review*, 74–79.

Kahneman, D., Sibony, O., & Sunstein, C. R. (2021). *Noise: A flaw in human judgment*. William Collins.

Krenn, M. (2011). From scientific management to homemaking: Lillian M. Gilbreth's contributions to the development of management thought. *Management & Organizational History*, *6*(2), 145–161. https://doi.org/10.1177/1744935910397035

Lee, H. L., Padmanabhan, V., & Whang, S. (2004). Information distortion in a supply chain: The bullwhip effect. *Management Science*, *50*(12 SUPPL.), 1875–1886. https://doi.org/10.1287/mnsc.1040.0266

Martin, M. K., Gonzalez, C., & Lebiere, C. (2004). Learning to make decisions in dynamic environments: ACT-R plays the beer game. In M. Lovett, C. Schunn, C. Lebiere, & P. Munro (Eds.), *Proceedings of the Sixth International Conference on Cognitive Modeling: ICCCM 2004: Integrating Models* (Vol. 420, pp. 178–183). Lawrence Erlbaum Associates Publishers.

Mayne, D. Q. (2014). Model predictive control: Recent developments and future promise. *Automatica*, *50*(12), 2967–2986. https://doi.org/10.1016/j.automatica.2014.10.128

Moritz, B. B., Narayanan, A., & Parker, C. (2021). Unraveling Behavioral Ordering: Relative Costs and the Bullwhip Effect. *Manufacturing & Service Operations Management*, *November*. https://doi.org/10.1287/msom.2021.1030

Morrison, J. B., & Oliva, R. (2018). Integration of Behavioral and Operational Elements Through System Dynamics. In *The Handbook of Behavioral Operations* (pp. 287–321). John Wiley & Sons, Inc. https://doi.org/10.1002/9781119138341.ch8

Narayanan, A., & Moritz, B. B. (2015). Decision Making and Cognition in Multi-Echelon Supply Chains: An Experimental Study. *Production and Operations Management*, *24*(8), 1216–1234. https://doi.org/10.1111/poms.12343

Oliva, R., Abdulla, H., & Gonçalves, P. (2022). Do Managers Overreact When in Backlog? Evidence of Scope Neglect from a Supply Chain Experiment. *Manufacturing & Service Operations Management*. https://doi.org/10.1287/msom.2021.1072

Opex Analytics. (2018). *The Beer Game*.

Oroojlooyjadid, A., Nazari, M., Snyder, L., & Takáč, M. (2017). A Deep Q-Network for the Beer Game: A Deep Reinforcement Learning algorithm to Solve Inventory Optimization Problems. *Arxiv*, 1–38.

Oroojlooyjadid, A., Nazari, M., Snyder, L. V., & Takáč, M. (2021). A Deep Q-Network for the Beer Game: Deep Reinforcement Learning for Inventory Optimization. *Manufacturing & Service Operations Management*, msom.2020.0939. https://doi.org/10.1287/msom.2020.0939

Paine, J. (2020). *Taming the Bull: Algorithmic Intervention to Mitigate Inventory and Ordering Amplification in Multi-Echelon Supply Chains*. Massachusetts Institute of Technology Sloan School of Management. http://dspace.mit.edu/handle/1721.1/7582

Pannek, J., & Frazzon, E. (2014). Supply Chain Optimization via Distributed Model Predictive Control: Supply Chain Optimization via Distributed Model Predictive Control. *PAMM*, *14*(1), 905–906. https://doi.org/10.1002/pamm.201410433

Payne, S. C., Youngcourt, S. S., & Watrous, K. M. (2006). Portrayals of F.W. Taylor across textbooks. *Journal of Management History*, *12*(4), 385–407. https://doi.org/10.1108/17511340610692752

Seborg, D. E., Edgar, T. F., Millichamp, D. A., & Doyle III, F. J. (2016). *Process Dynamics and Control* (4th ed.). Wiley.

Secomandi, N. (2008). An Analysis of the Control-Algorithm Re-solving Issue in Inventory and Revenue Management. *Manufacturing & Service Operations Management*, *10*(3), 468–483. https://doi.org/10.1287/msom.1070.0184

Shih, W. (2020). *COVID-19 And Global Supply Chains: Watch Out For Bullwhip Effects*. Forbes. https://www.forbes.com/sites/willyshih/2020/02/21/covid-19-and-global-supply-chains-watch-out-for-bullwhip-effects/?sh=2fa2b2467195

Stank, T., Goldsby, T., & Saunders, L. (2021). Commentary: Caution—Bullwhip Effect Ahead. *The Wall Street Journal*.

Sterman, J. (1989). Modeling Managerial Behavior: Misperceptions of Feedback in a Dynamic Decision Making Experiment. *Management Science*, *35*(3), 321–339. https://doi.org/10.1287/mnsc.35.3.321

Sterman, J. D. (2000). *Business Dynamics—Systems Thinking and Modeling for a Complex World*. McGraw-Hill Higher Education.

Sterman, J. D., & Dogan, G. (2015). "I'm not hoarding, I'm just stocking up before the hoarders get here." Behavioral causes of phantom ordering in supply chains. *Journal of Operations Management*, *39–40*(1), 6–22. https://doi.org/10.1016/j.jom.2015.07.002

Sutton, R., & Barto, A. (2014). *Reinforcement Learning: An Introduction* (2nd ed.). The MIT Press. https://doi.org/10.1016/S1364-6613(99)01331-5

Thrun, S., & Schwartz, A. (1993). Issues in Using Function Approximation for Reinforcement Learning. *Proceedings of the 1993 Connectionist Models Summer School*, 255–263.

Tversky, A., & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *Science*, *185*, 1124–1131. https://doi.org/10.4324/9781912282562

Vossen, T. W. M., You, F., & Zhang, D. (2022). Finite-horizon approximate linear programs for capacity allocation over a rolling horizon. *Production and Operations Management*, *31*(5), 2127–2142. https://doi.org/10.1111/poms.13669

Wang, X., & Disney, S. M. (2016). The bullwhip effect: Progress, trends and directions. *European Journal of Operational Research*, *250*(3), 691–701. https://doi.org/10.1016/j.ejor.2015.07.022

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Frcitas, N. (2016). Dueling Network Architectures for Deep Reinforcement Learning. *33rd International Conference on Machine Learning, ICML 2016*, *4*(9), 2939–2947.

Wiering, M. (1999). *Explorations in efficient reinforcement learning*. University of Amsterdam.

Wu, D. Y., & Katok, E. (2006). Learning, communication, and the bullwhip effect. *Journal of Operations Management*, *24*(6), 839–850. https://doi.org/10.1016/j.jom.2005.08.006

Yu, M. C., & Kuncel, N. R. (2020). Pushing the Limits for Judgmental Consistency: Comparing Random Weighting Schemes with Expert Judgments. *Personnel Assessment and Decisions*, *6*(2), 1–10. https://doi.org/10.25035/pad.2020.02.002