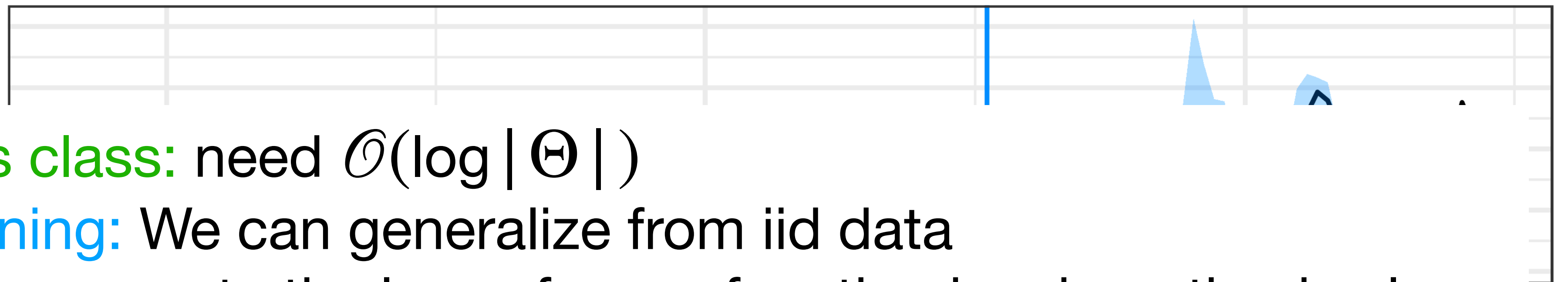
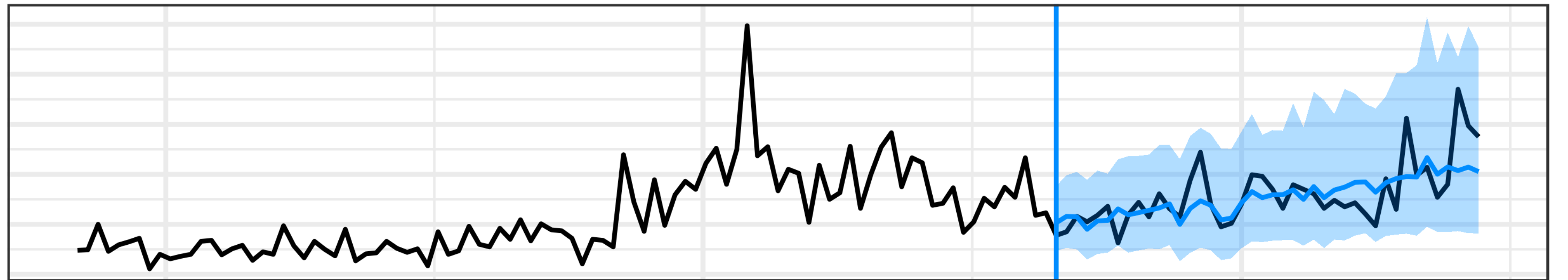
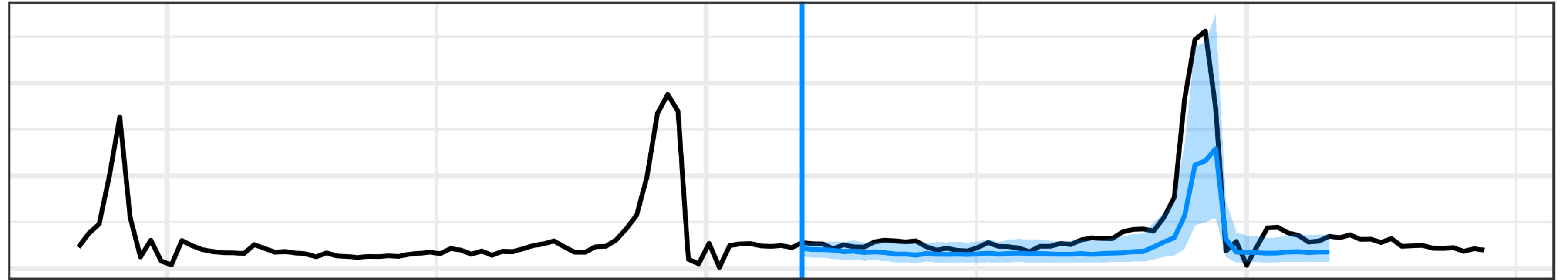


Reinforcement Learning for Supply Chains

Dhruv Madeka^{*}, Dean Foster^{*}, Sham M. Kakade^{^*}

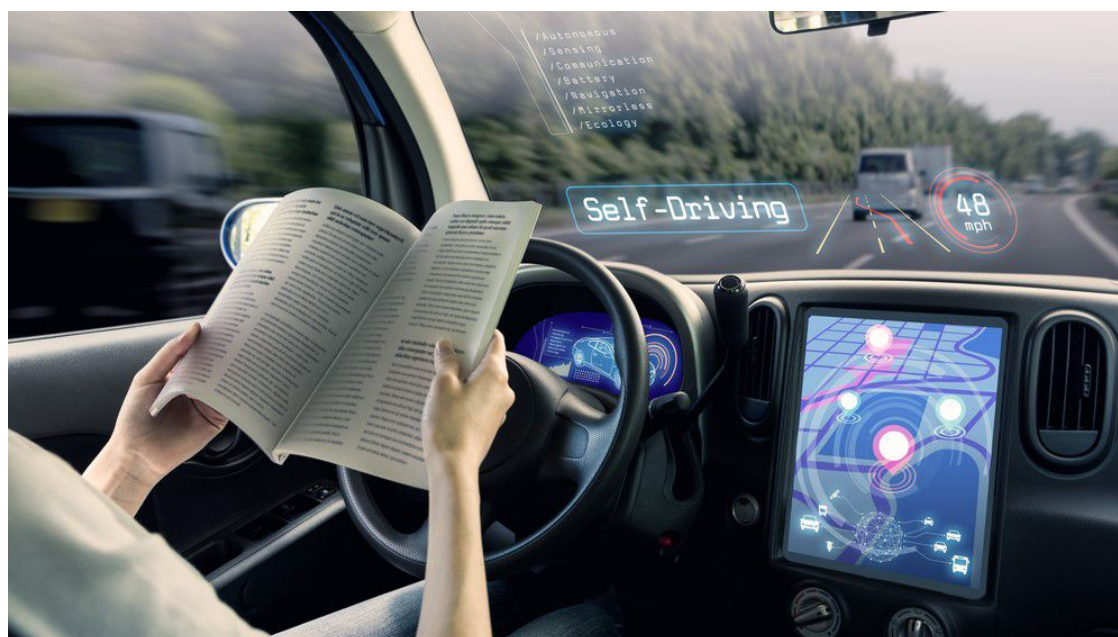
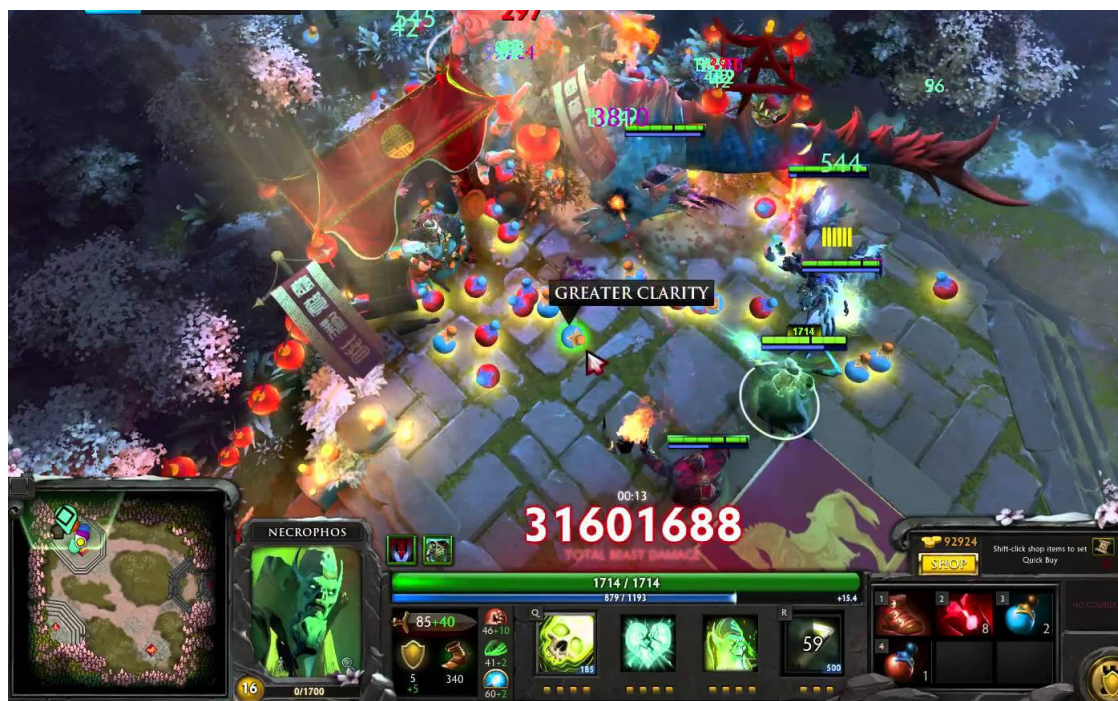
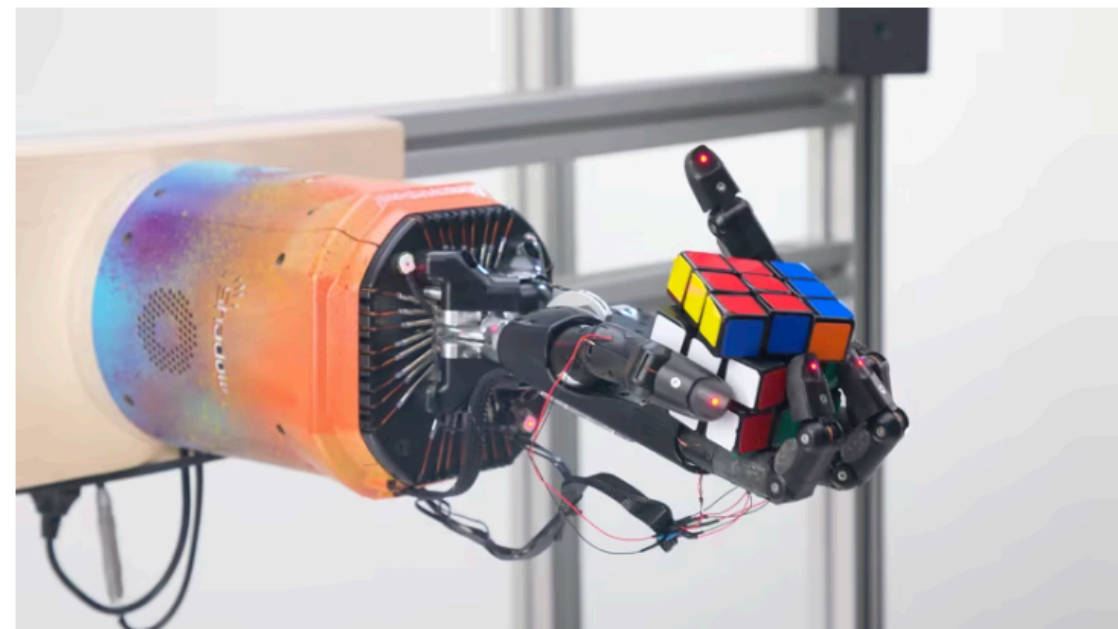
^{*}Amazon [^]Harvard University

First, let's look at supervised learning at Amazon.



See Wen, Torkkola,
Narayanaswamy, [M.](#)
(2017)
Eisenach, Patel, [M.](#)
(2020)

- **Finite hypothesis class:** need $\mathcal{O}(\log |\Theta|)$
 - **Supervised Learning:** We can generalize from iid data
- Data reuse:** We can compute the loss of every function in a hypothesis class



Real-world RL is hard.

The core challenges Amazon faces are sequential decision making problems.

Can RL help in this space?



amazon prime Deliver to Sham Boston 02118 Electronics

HiSense Roku TV Hisense 40-Inch Class H4 Series LED Roku Smart TV with... 8,970

Electronics > Television & Video > Televisions > OLED TVs

LG C2 Series 77-Inch Class OLED evo Gallery Edition Smart TV OLED77C2PUA, 2022 - AI-Powered 4K TV, Alexa Built-in

Visit the LG Store 663 ratings | 268 answered questions

Deal -5% \$2,496⁹⁹ Was: \$2,627.05

Size: 77 inch 42 inch 48 inch 55 inch 65 inch 77 inch 83 inch

Style: TV Only TV + \$65Q TV + \$75Q TV + \$80Q TV Only TV + \$90Q

TV wall mounting options: Get expert TV wall mounting Details

Without expert wall mounting Expert wall mounting +\$200.00 per unit

12 monthly payments: \$208.09/mo. (\$2,496.99 / 12 mo.)

One-time payment: \$2,496⁹⁹

FREE Inside Entryway delivery as soon as Saturday, November 26, 9 AM - 12 PM

In Stock Qty: 1

Add to Cart Buy Now

RL is hard!

- Sample complexity **can be as large** as $\min(|\Theta|, 2^T)$

Dexterous Robotic Hand Manipulation

OpenAI, '19

- Large state/action spaces
- Exploration
- Credit assignment problem



The Supply Chain Problem

- Supply Chain is about buying, storing, and transporting goods.
- Amazon has been running it's Supply Chain for decades now
 - There is a lot of historical “off-policy” data
 - How do we use it?
 - Concern: counterfactual issue?
- This talk: how can we **use this data** to solve the inventory management problem?

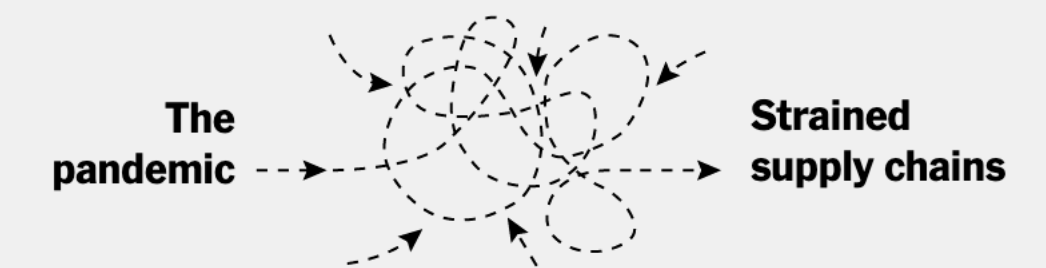


Supply Chain Hurdles Will Outlast Pandemic, White House Says

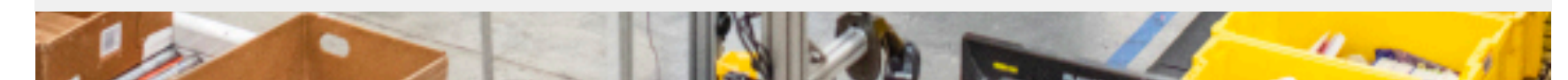
The administration's economic advisers see climate change and other factors complicating global trade patterns for years to come.



The New York Times



How the Supply Chain Crisis Unfolded



Outline

Can we use historical data to solve inventory management problems in supply chain?

- Part I: Utilizing Historical Data
- Part II: Moving to real-world inventory management problems
- Part III: Real World Results

Deep Inventory Management

Dhruv Madeka

Amazon, maded@amazon.com

Kari Torkkola

Amazon, karito@amazon.com

Carson Eisenach

Amazon, ceisen@amazon.com

Anna Luo

Pinterest*, annaluo676@gmail.com

Dean P. Foster

Amazon, foster@amazon.com

Sham M. Kakade

Amazon, Harvard University, shamisme@amazon.com

Largely based on this paper: [arxiv/2210.03137](https://arxiv.org/abs/2210.03137)

I: Utilizing historical data

Warm up: Vehicle Routing

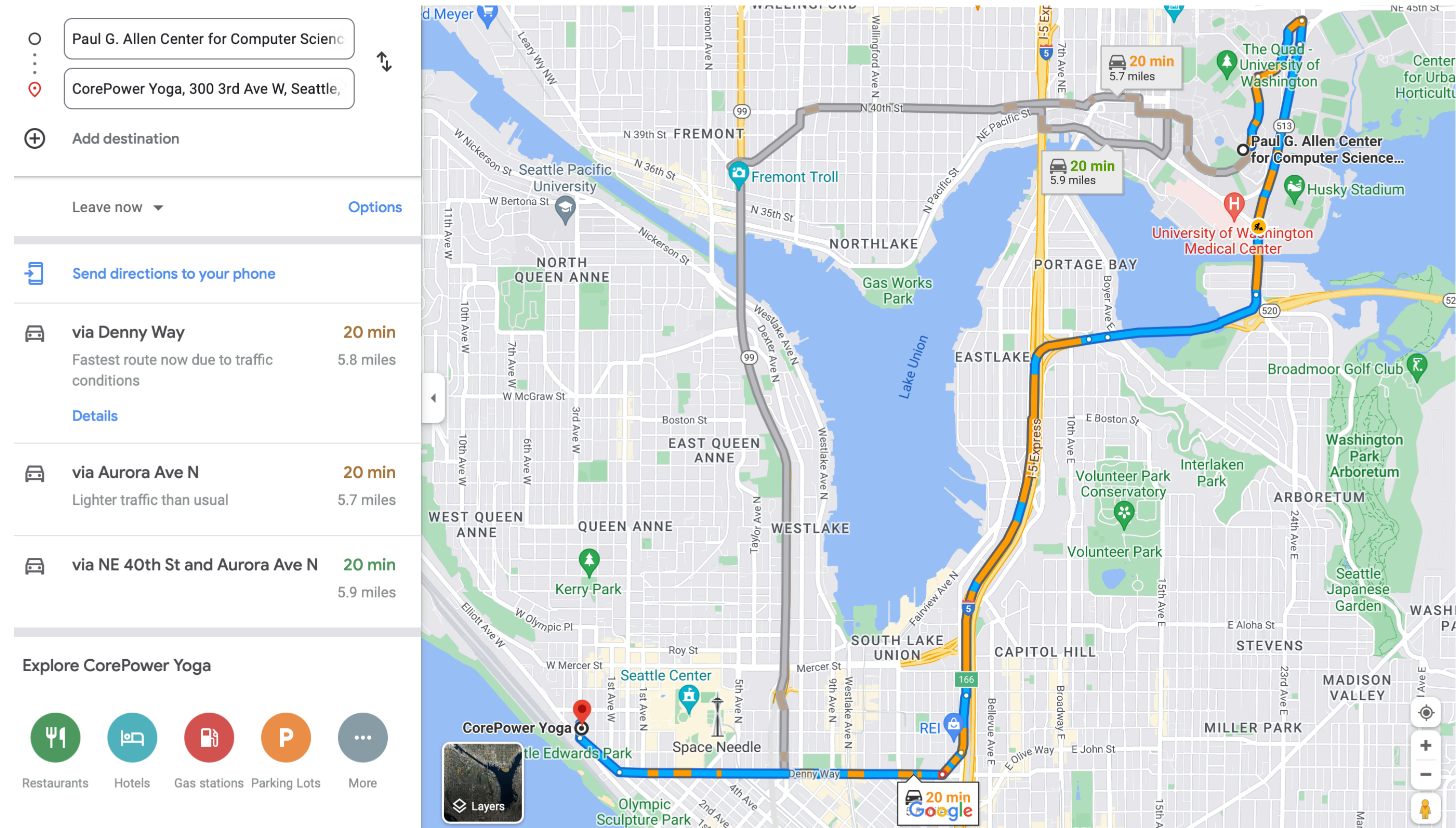
(when using historical data might be ok)

- We want a good policy for routing a single car.

- **Policy π : features \rightarrow directions features:**
time of day, holiday indicators, current traffic, sports games, accidents, location, weather,

- **Historical Data:**
suppose we have logged historical data of features

- **Backtesting policies:**
 - Key idea: a single route minimally affects traffic
 - Counterfactual: with the historical data, we can see what would have happened with another policy.



Warm up 2: Fleet Routing

- We want to route a whole fleet of self-driving taxis.
- Policy π : features \rightarrow directions
 - features:
customer demand, time of day, holiday indicators, current traffic, sports games, accidents, location, weather...
- Historical Data:
suppose we have logged historical data of features
- Backtesting policies:
 - Key idea: a small fleet route may have small affects on traffic.
 - Counterfactual: with the historical data, we can see what would have happened with another policy.



Supply Chain Data

Price= \$2

Cost= \$1

Time	Inventory	Demand	Order	Revenue
0	100	20	-	40
0	80	-	10	-10
1	90	20	-	40
1	70	-	50	-50
2	120	60	-	120
2	60	-	10	-10

Backtesting a policy

Price= \$2
Cost= \$1

Time	Inventory	Demand	Order	Revenue
0	100	20	-	40
0	80	-	10 40	-10 -40
1	90 120	20	-	40
1	70 100	-	50 20	-50 -20
2	120	60	-	120
2	60	-	10	-10

- Current order doesn't impact future demand.
 - This allows us to backtest!
 - Empirically, backlog due to unmet demand does not look significant.¹

1. See Verhoef et al (2006)

Formalization of the Supply Chain Problem

- Growing literature around a class of MDPs where a large part of the state is driven by an exogenous noise process [Efroni et al 2021, Sinclair et al 2022]
- A formalization of the model:
 - Action a_t : how much you buy
 - Exogenous random variables: evolving under \Pr and not dependent on our actions
(Demand $_t$, Price $_t$, Cost $_t$, Lead Time $_t$, Covariates $_t$) $:= s_t$
 - Controllable part (inventory) I_t : evolution is dependent on our action.
 - $I_t = \min(I_{t-1} + a_{t-1} - D_t, 0)$ (and suppose we start at I_0).
 - Reward is just the sum of profits: $r(s_t, I_t, a_t) := \text{Price}_t \times \min(\text{Demand}_t, I_t) - \text{Cost}_t \times a_t$
- Learning setting:
 - Data collection: We observe N historical trajectories, where each sequence is sampled $s_1, \dots, s_T \sim \Pr$
 - Goal: maximize our rewards cumulative reward over T periods

$$V_T(\pi) = E_\pi \left[\sum_{t=1}^T r(s_t, I_t, a_t) \right]$$

Why is it an interesting RL problem?

- Lots of time dependence!
 - If you buy too much, you're left with the inventory for months!
 - Your actions (orders) affect the state at a random time later
 - Tons of correlation across time (Demand, Price, Cost)
- We can explore (linear risk in every product)

Theorem: Backtesting in ExoMDPs

Theorem [M., Torkkola, Eisenach, Luo, Foster, Kakade '22]:

Suppose we have a set of K policies $\Pi = \{\pi_1, \dots, \pi_K\}$, and we have N sampled exogenous paths. Then we can accurately backtest up to nearly $K \approx 2^N$ policies.

Formally, for any $\delta \in (0,1)$, with probability greater than $1 - \delta$ - we have that for all $\pi \in \Pi$:

$$|V_T(\pi) - \hat{V}_T(\pi)| \leq T \sqrt{\frac{\log(K/\delta)}{N}}$$

(assuming the reward r_t is bounded by 1).

- **Implications:**
 - We can optimize a **neural policy** on the past data.
 - In the usual RL setting (not exogenous), we would have an **amplification factor of (at least) $\min\{2^T, K\}$** , using historical data due to the counterfactual issue.

II: Real World Inventory Management Problems

Real-world Issue: **Censored** Demand

- When **demand** \geq **inventory**, what customers see:

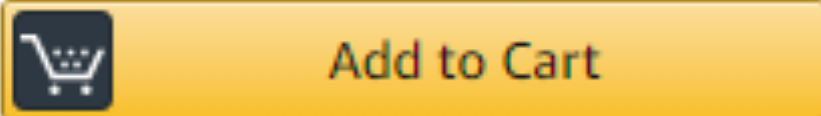
\$19.99
& **FREE Shipping**
Get it Tue, Jan 29 - Thu, Jan 31,
or
Get it Fri, Jan 25 - Fri, Jan 25 if
you choose paid Local Express
Shipping at checkout

**In stock on January 23,
2019.**

Order it now.
Ships from and sold by Vertellis.

Qty: 1 ▾

\$19.99 + Free Shipping




Buy New **\$18.96**
Qty: 1 ▾ List Price:
\$29.99
Save: \$11.03 (37%)

FREE Shipping on orders over \$35.

Temporarily out of stock.
Order now and we'll deliver when
available. [Details](#) ▾

Ships from and sold by Amazon.com.
Gift-wrap available.



[Sign in to turn on 1-click ordering](#)

We only observe **sales** not the **demand**:
Sales := min(Demand, Inventory)

Can we still backtest?

Our historical data is then censored....

Sales := min(Demand, Inventory)

Price= \$2
 Cost= \$1

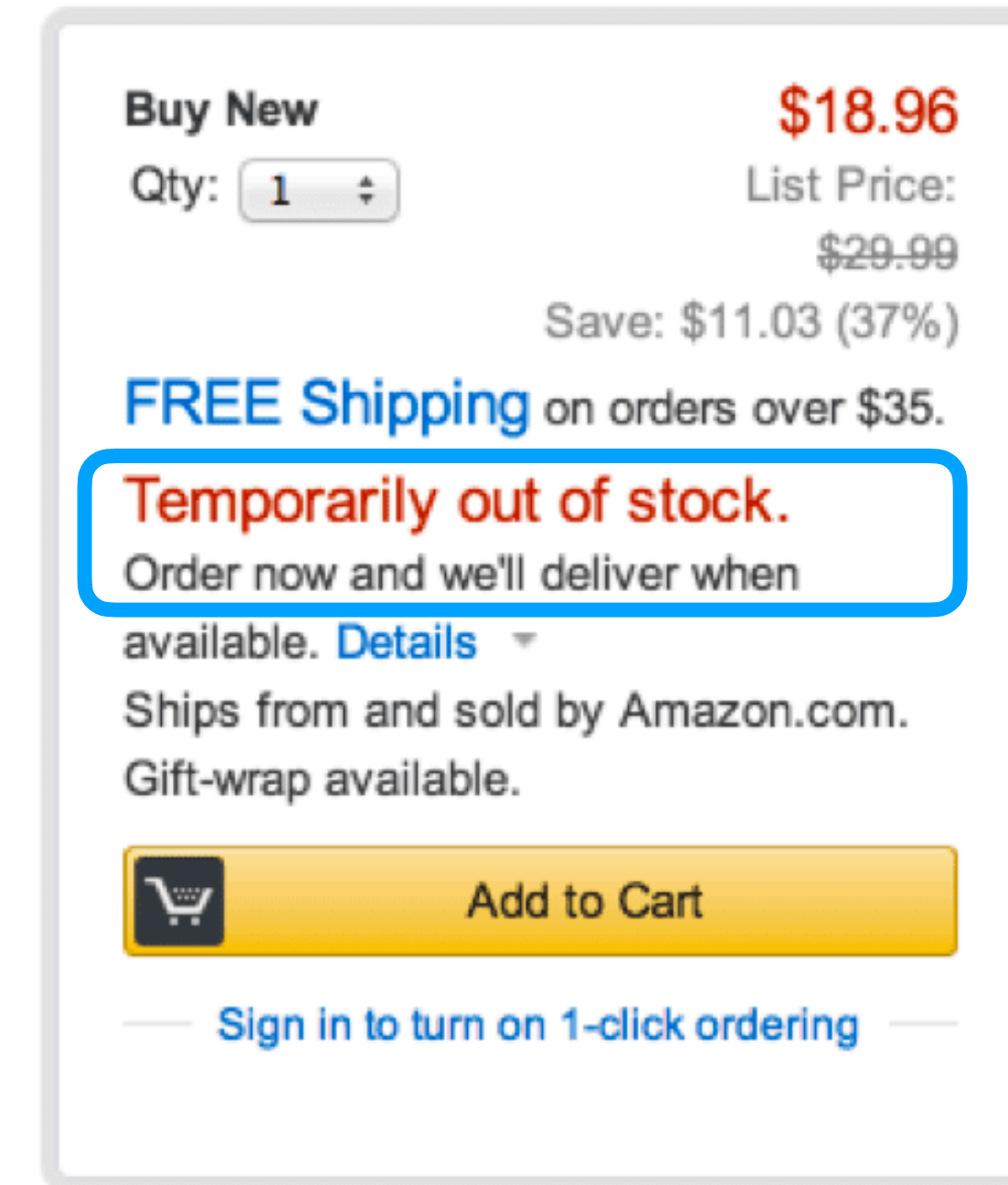
Time	Inventory	True Demand	Sales	Order	Revenue
T	10	??	10	-	20
⋮	⋮				
⋮	⋮				
⋮	⋮				
⋮	⋮				
⋮	⋮				

The image shows two screenshots of an Amazon product page. The left screenshot displays a product priced at \$19.99 with a status message: "In stock on January 23, 2019." This message is highlighted with a blue rounded rectangle. The right screenshot shows the same product at a lower price of \$18.96, with a status message: "Temporarily out of stock." This message is also highlighted with a blue rounded rectangle. Both screenshots include shipping information, quantity selectors, and "Add to Cart" buttons.

If we could fill in the missing demand, then we could still backtest!

We have many observed historical covariates

- **Covariates:** Sales, Web Site, **Glance Views**, Product Text, Reviews
- **Example:** the #times customers look at an item gives us info about the unobserved demand.



- **Let's forecast the missing variables** from the observed covariates!
 $\hat{P}(\text{Missing Data} | \text{Observed Data})$

Uncensoring the data....

$$\text{Sales} := \min(\text{Demand}, \text{Inventory})$$

Price= \$2
Cost= \$1

Time	Inventory	True Demand	Sales	Order	Revenue
T	10	40	10	-	20
⋮	⋮	⋮	⋮		
⋮	⋮	⋮	⋮		
⋮	⋮	⋮	⋮		
⋮	⋮	⋮	⋮		
⋮	⋮	⋮	⋮		
⋮	⋮	⋮	⋮		

Buy New **\$18.96**
Qty: 1
List Price: \$29.99
Save: \$11.03 (37%)
FREE Shipping on orders over \$35.
Temporarily out of stock.
Order now and we'll deliver when available. [Details](#)
Ships from and sold by Amazon.com.
Gift-wrap available.
[Add to Cart](#)
[Sign in to turn on 1-click ordering](#)

Key idea:
Use covariates
(e.g. glance
views) to forecast
missing demand,
vendor lead
times, etc

Theorem: Backtesting in Uncensored ExoMDPs

Theorem [M., Torkkola, Eisenach, Luo, Foster, Kakade 22]:

If we can forecast the missing variables accurately (in a total variation sense), then we can backtest accurately. More formally,

Setting: we have N sampled sequences $\{s_1^i, s_2^i, \dots, s_T^i\}_{i=1}^N$,

where M_i and O_i are the missing and observed exogenous variables in sequence i .

Forecast: $\widehat{\mathbb{P}}^i = \widehat{\Pr}(M_i | O_i)$ is our forecast of $\mathbb{P}^i = \Pr(M_i | O_i)$.

Assume: With pr. 1, forecasting has low error:
$$\frac{1}{N} \sum_{i=1}^N \text{TotalVar}(\mathbb{P}^i, \widehat{\mathbb{P}}^i) \leq \epsilon_{\text{sup}}.$$

Guarantee: For any $\delta \in (0,1)$, with pr. greater than $1 - \delta$, for all $\pi \in \Pi$:

$$|V_T(\pi) - \widehat{V}_T(\pi)| \leq T \left(\epsilon_{\text{sup}} + \sqrt{\frac{\log(K/\delta)}{N}} \right)$$

- Key idea: We can backtest even in the **censored** setting!

III: Training Policies & Empirical Results

The Simulator

- **Collection of historical trajectories:**
 - 1 million products
 - 104 weeks of data per product
- **Uncensoring:**
 - Demand
 - Vendor Lead Times
- **Policy gradient methods in a “gym”:**
 - “gym” ↔ backtesting ↔ simulator
(note the “simulator” isn’t a good world model).
 - The policy can depend on many features.
(seasonality, holiday indicators, demand history, ASIN, text features)



Differentiable Control Problem

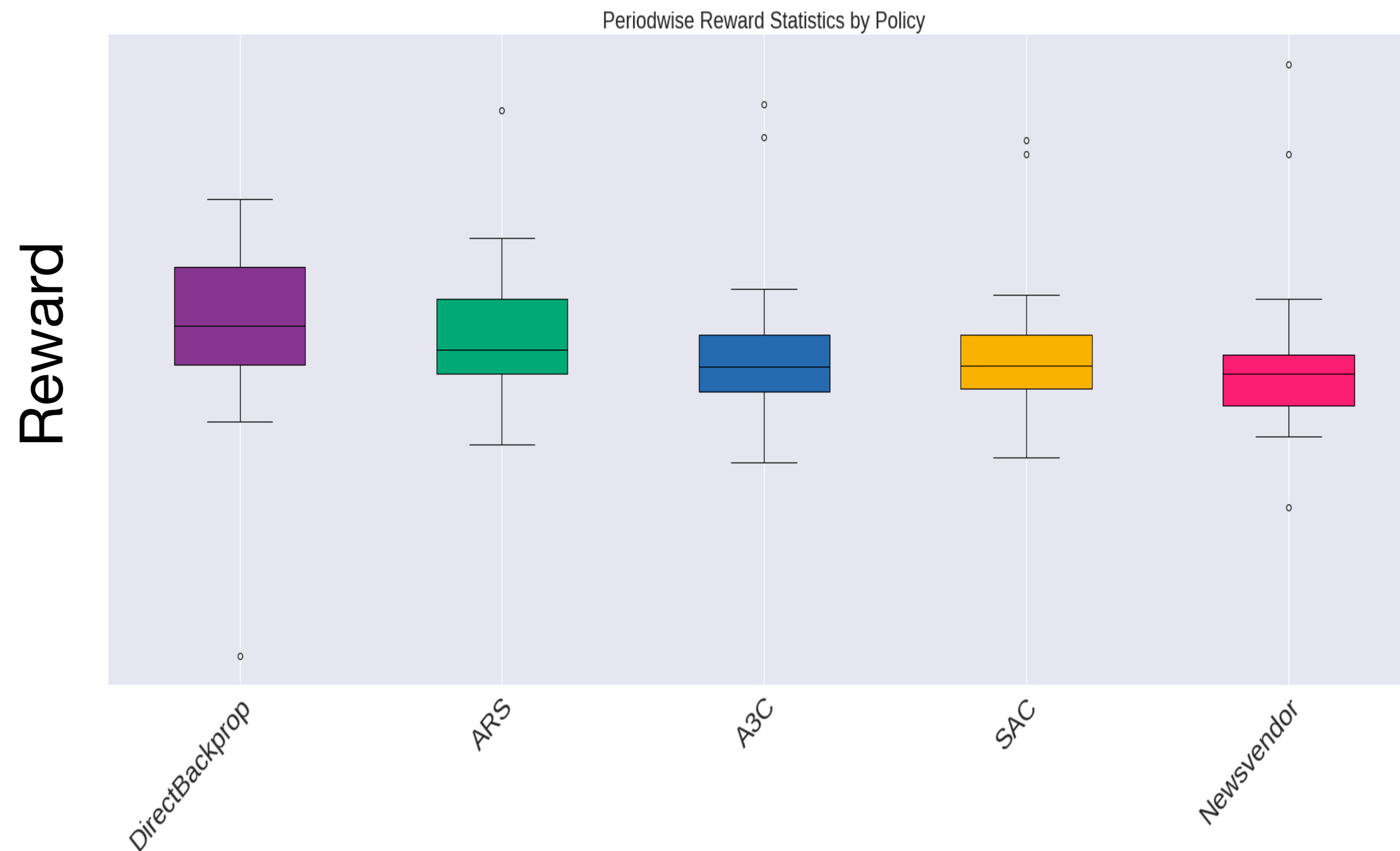
- Note that each term of our state evolution is a [differentiable function](#) of previous actions
- So, we can take gradients directly from our Reward through our policy
- This is our current production policy, called *DirectBackprop*
- Similar in spirit to Perturbation Analysis (Glasserman et al 1995), except it uses a [neural policy](#)

Sim to Real Transfer

- Sim: the backtest of [DirectBackprop](#) improves on Newsvendor.
- Real: [DirectBackprop](#) significantly reduces inventory without significantly reducing total revenue.

Simulation

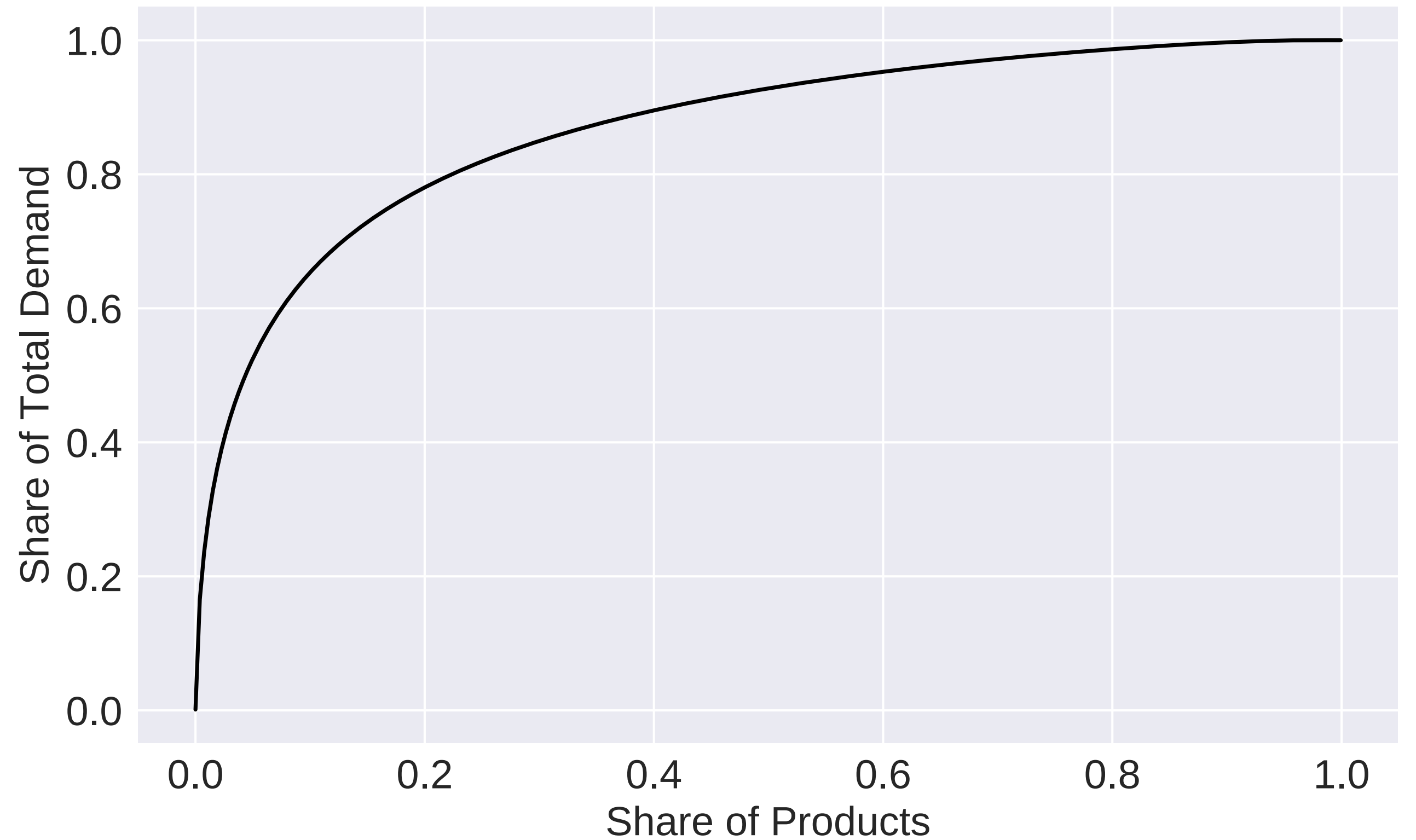
Real World



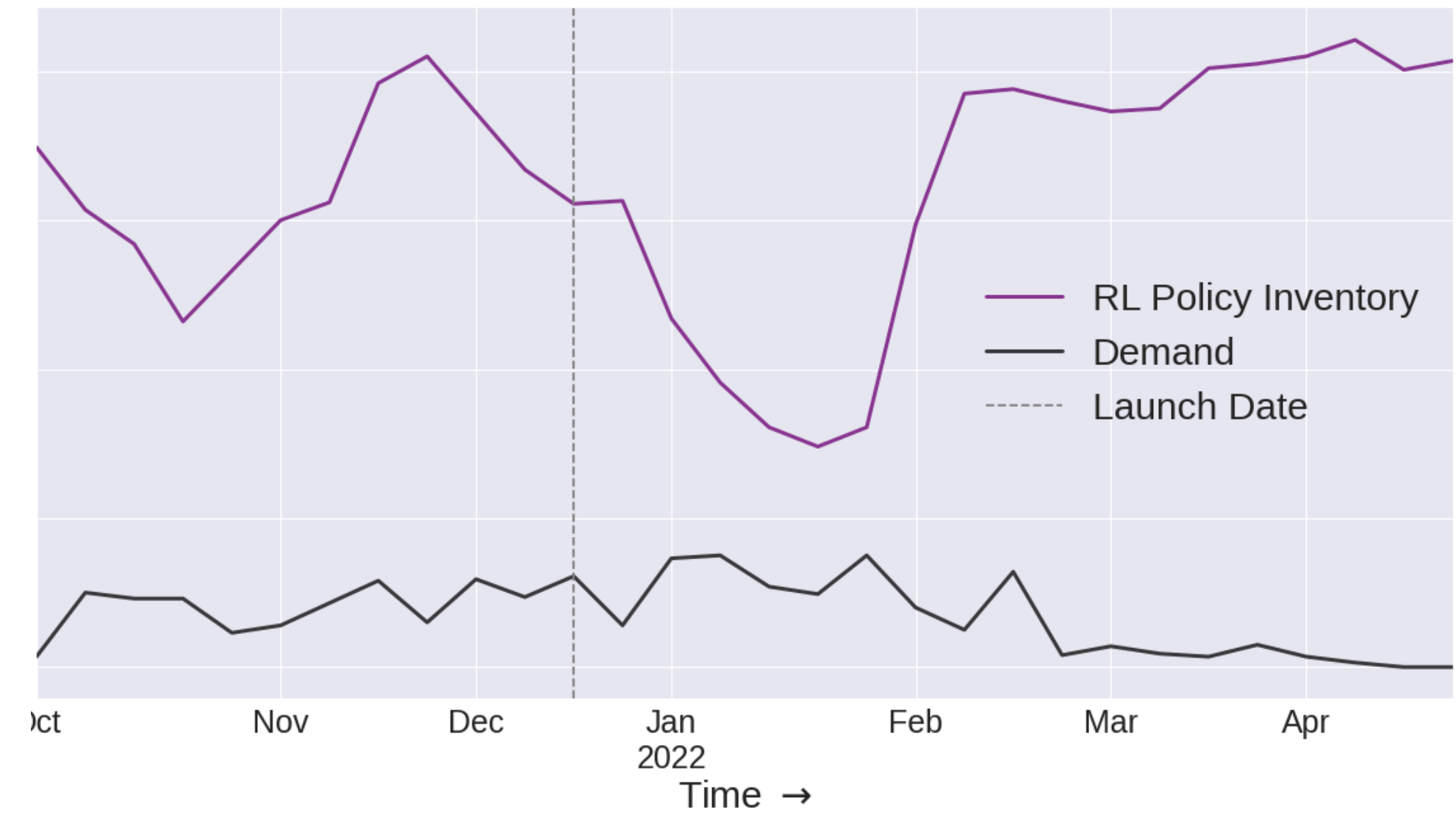
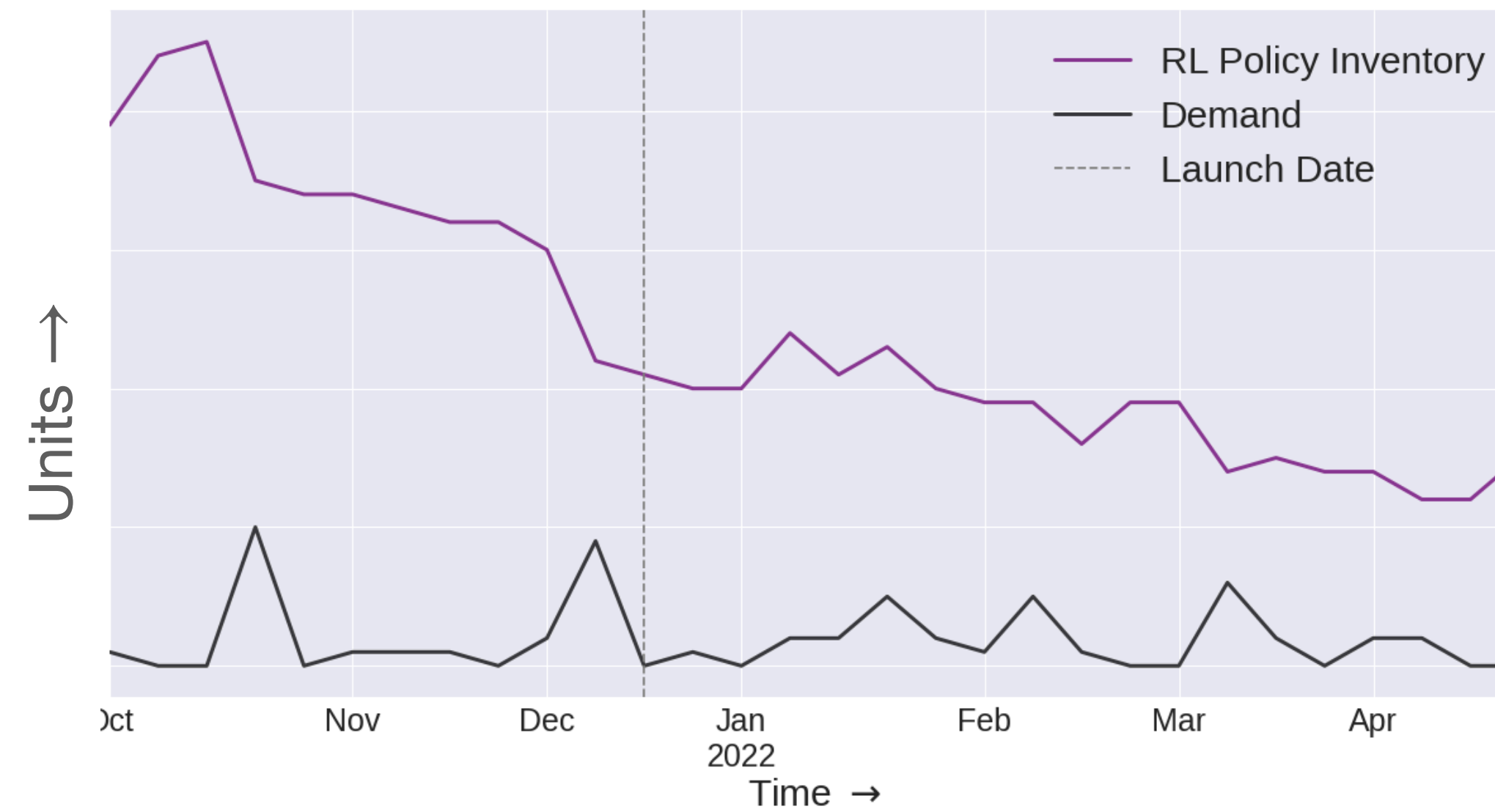
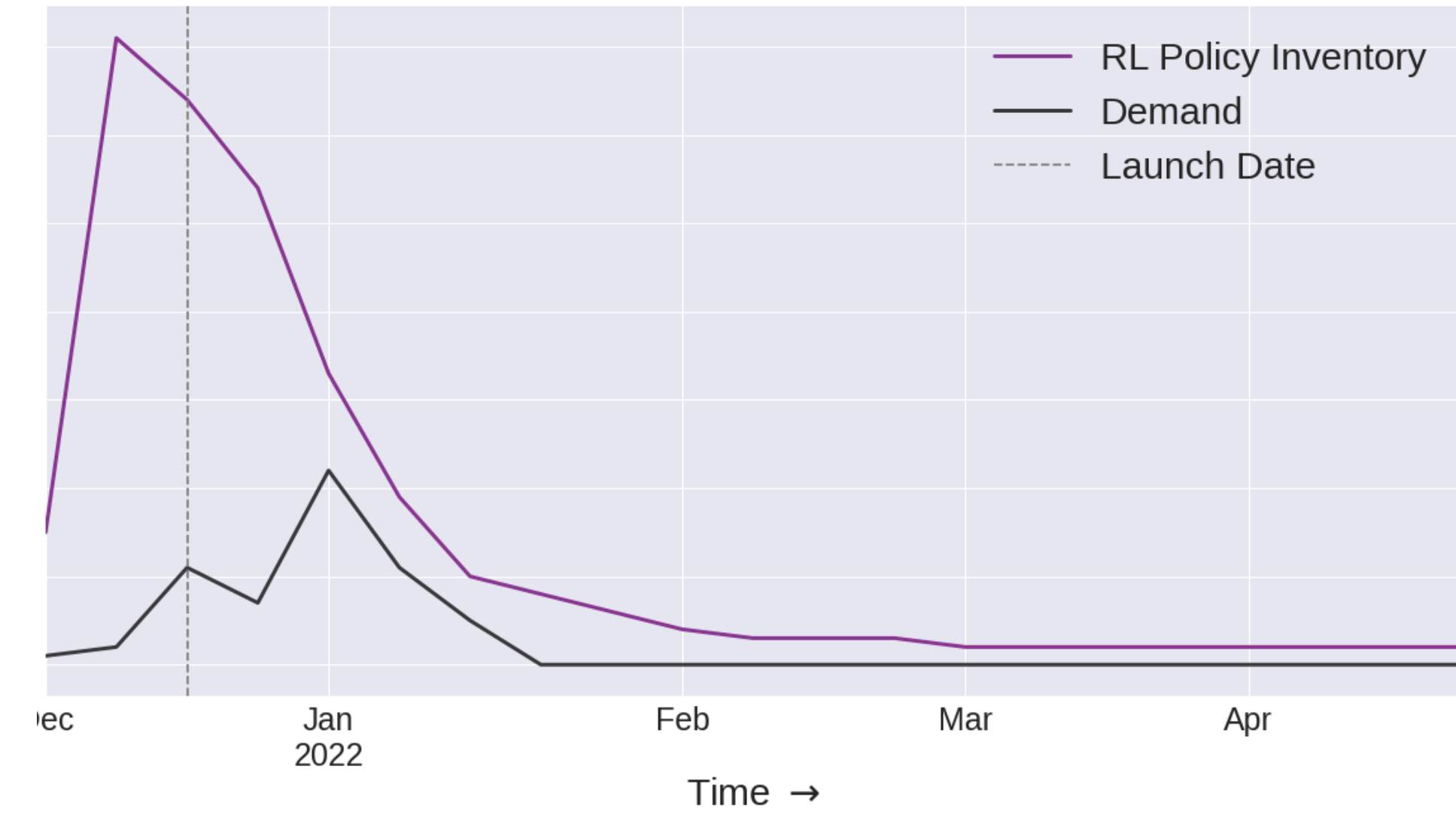
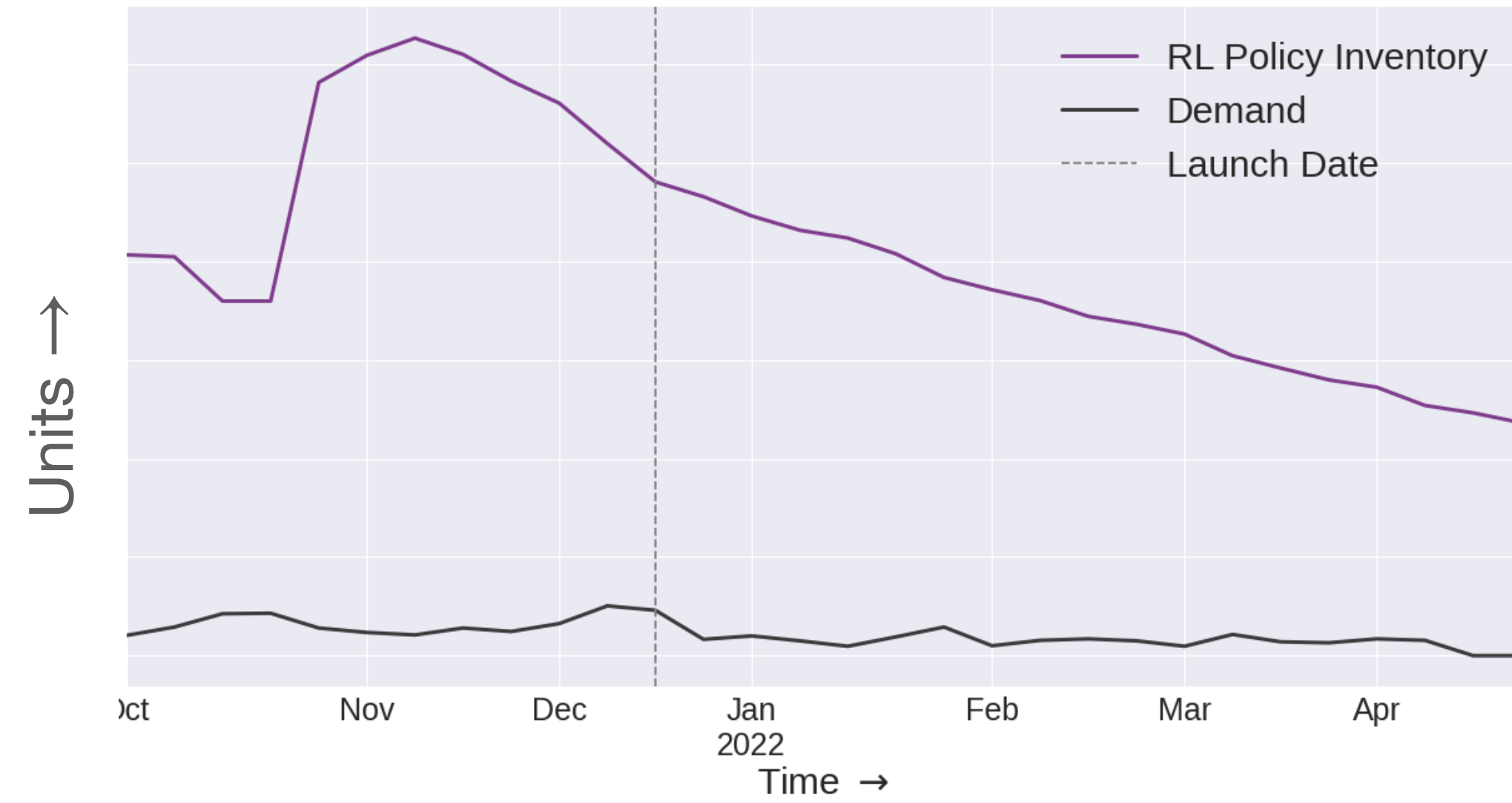
Metrics	% change
Inventory Level	-12 ± 6
Revenue	~

What about in the real world?

- Really hard to measure! (Tripuraneni, [M](#) et. al 2021)
- Heavy tailed data:
 - A few products contribute to most of the reward



Anecdotally, RL has reasonable strategies in the real world...



Real World RL Challenges

- World is **not perfectly** exogenous (some terms may depend on our actions)
- Cross product constraints are **computationally intensive**
- Not **every Supply Chain** problem can be written in this framework

Conclusion

- There are a class of RL Problems that work in the real world!
- The exogenous assumption allows us to backtest **any** policy on historical data
- A large number of classical Operations Research problems fall into this class of Interactive Decision-Making problems



Carson



Kari



Anna



Dean



Sham