A Q Learning

Q learning is an RL algorithm that learns an estimate of the *action-value function* $Q^*(s,a)$. This action-value function gives the expected total reward of taking action a at state s and using the optimal policy function in the future. Once $Q^*(s,a)$ has been learned, the optimal policy is

$$\pi^*(s) \in \arg\max_{a \in A} Q^*(s,a). \tag{2}$$

A Q-learning algorithm maintains an $|S| \times |A|$ Q-matrix, Q_t, representing the estimate of $Q^*(s,a)$ at step t. Usually, this matrix is randomly initialized. At each step t, the algorithm takes action a_t that with probability $1-\varepsilon_t$ is optimal according to its current Q-matrix Q_t, and with probability ε_t chosen uniformly at random from the set of avalilable actions. We call ε_t the *exploration rate*. The entry (s_t, a_t) of Q_t is updated based on feedback via a convex combination of its previous value and the reward $r(s_t, a_t)$ attained from the action plus the discounted value of the state s_{t+1}:

$$Q_{t+1}(s_t,a_t) = (1-\alpha)Q_t(s_t,a_t) + \alpha[r(s_t,a_t) + \delta \max_{a \in A} Q_t(s_{t+1},a)]. \tag{3}$$

Parameter $\alpha \in [0,1]$ is the *learning rate*. When the environment is stationary and Markovian, and under suitable assumptions on the learning rate and exploration rate, the Q-matrix is guaranteed to converge in the limit to the action-value function $Q^*(s,a)$ and the policy to the optimal policy.

B Proof of Proposition 1

Proposition 1 *The optimal Stackelberg POMDP policy $\pi^*_{n_e,n_r}$, for an equilibrium phase with $n_e \geq 1$ steps and a reward phase with $n_r \geq 1$ steps, maximizes $CS(\pi)$, for seller behavior induced after n_e steps when $n_r = T^*$.*

Proof. Let $\tau \sim q_\pi(\tau)$ denote a generic trajectory determined by executing policy π in the Stackelberg POMDP environment. We have

$$\pi^*_{n_e,n_r} \in \text{argmax}_\pi \mathbb{E}_{\tau \sim q_\pi(\tau)} \left[\sum_{t=n_e+1}^{n_e+n_r} r(s_t,a_t) \right], \tag{4}$$

recognizing $r(s_t,a_t) = 0$ if $t \leq n_e$. After replacing $r(s_t,a_t)$ with $U(p_t;\pi(p_t))$, we can rewrite the objective in (4) as

$$\mathbb{E}_{\tau \sim p_\pi(\tau)} \left[\sum_{t=n_e+1}^{n_e+n_r} U(p_t;\pi(p_t)) \right],$$

which is equal to (1) when $n_r = T^*$.

C Proof of Proposition 2

Proposition 2 *For any $\epsilon > 0$, there exists a threshold platform rule π such that $CS(\pi) \geq CS(\pi^*) - \sum_t \delta^t \epsilon$ under a subgame perfect Nash equilibrium (SPE) of the infinitely-repeated continuous pricing game induced by platform rule π.*

Proof. For any $\eta > 0$, we study the stage game with continuous prices induced by the threshold platform rule that sets threshold $\tau = c + \eta$ for each price profile p. We show that this stage game has a unique Nash equilibrium in which each seller sets a price $p_i = p^*$, for some $p^* \in (c, c+\eta]$. Given this, we have that every seller pricing at $p^* \in (c, c+\eta]$ in every period is an SPE of any infinitely repeated game between sellers, since this is an open-loop Nash equilibrium profile (and thus SPE by the single-deviation principle). Moreover (as it is a continuous function over price profiles) the consumer surplus comes arbitrarily close, for a small enough $\eta > 0$, to the maximum consumer surplus, which coincides with every seller pricing at cost.

Left to prove is that every seller pricing at p^* is a Nash eq (NE) of the stage game. First, pricing $p_i > \tau(p_t) = c + \eta$ provides zero profit to a seller because the seller is not part of the displayed set of sellers. Similarly, pricing $p_i = c$ provides zero profit. Now dropping the time period t, because we

study a generic stage game, and considering prices $p = (p_1, \ldots, p_n) \in \times_{i=1}^{n}(c, c+\eta]$, so that all sellers are displayed, and with seller profit $\rho_i(p; \mathcal{N})$ for price profile p, we have

$$\frac{\partial}{\partial p_i}\rho_i(p;\mathcal{N}) = -(p_i - c)\frac{D_i(p;\mathcal{N})(1 - D_i(p;\mathcal{N}))}{\mu} + D_i(p;\mathcal{N}).$$

By first-order optimality conditions, we have $\frac{\partial}{\partial p_i}\rho_i(p;\mathcal{N}) = 0$, for each i, only when $p_i = \hat{p}$, for some $\hat{p} > c$ (Anderson and De Palma, 1992). Furthermore, $\rho_i(p_i, p_{-i}; \mathcal{N})$ is concave. Thus, if $c + \eta \geq \hat{p}$, there is only one Nash equilibrium of the stage game, where each seller i sets price $p^* = \hat{p} \leq c + \eta$. On the other hand, if $c + \eta < \hat{p}$, we have that $\rho_i(p; \mathcal{N})$ is strictly increasing when $p \in \times_{i=1}^{n}(c, c+\eta]$. This means that a generic seller i can always increase its utility by quoting $p_i + \varepsilon$ instead of p_i. This happens unless p_i is equal to $c + \eta$: in this case, increasing p_i would drive the agent out of the buybox (zeroing its utility). Thus, there is a unique Nash equilibrium where each seller quotes price $p^* = c + \eta$.

D Variations on Setting Parameters

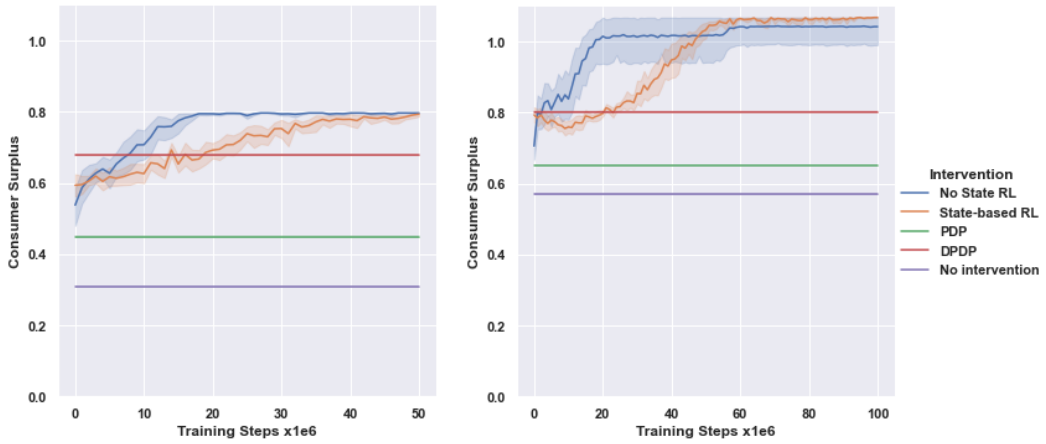

Figure 4: "Learning in the wild" performance of No State RL and State-based RL with different environment settings when (left) the horizontal differentiation μ is lowered from 0.25 to 0.05, and when (right) μ is increased from 0.25 to 0.40 . The results are averaged over 10 runs, and the shaded regions show 95% confidence intervals.

Following Calvano et al. (2020a) and Johnson et al. (2021), we also consider the learning performance of our RL algorithms in settings with different horizontal differentiation parameter μ. Specifically, Figure 4 displays our training curves when the horizontal differentiation is lowered from 0.25 to 0.05 and increased from 0.25 to 0.40, and our policies are learned "in the wild." We notice that, in both scenarios, our RL policies bring consumer welfare close to its optimal level (0.8 when $\mu = 0.05$ and 1.08 when $\mu = 0.4$) outperforming all the baselines.

E Deterministic Policies During Stackelberg POMDP Episodes

In this section, we test the learning performance of our RL algorithms when policies are not deterministic during the Stackelberg POMDP episodes. As we can see from Figure 5, this variation dramatically affects our learning performance, which only slightly improves during training and leads to final policies that do not outperform our baselines. As discussed in Section 5, this poor performance is caused by the sellers not being able to learn optimal response strategies due to the high variance introduced by the non-deterministic behavior of our policies.

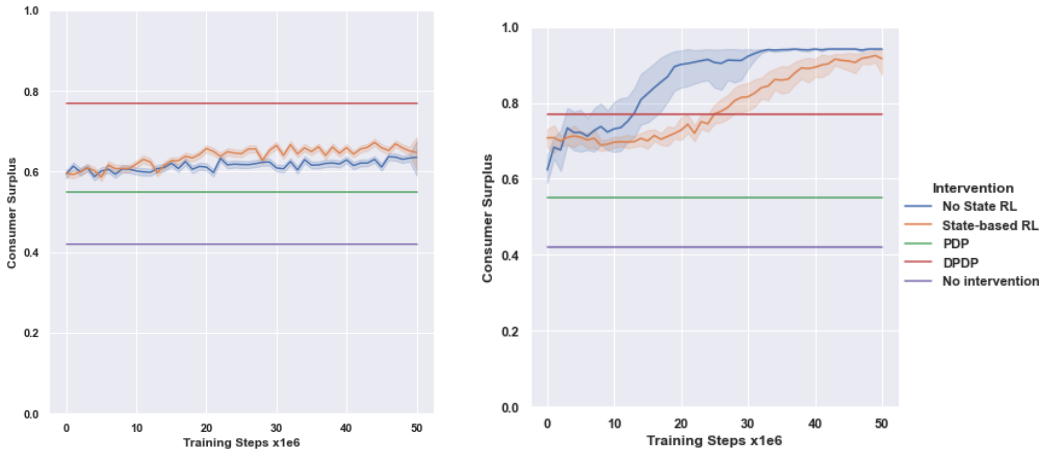

Figure 5: "Learning in the wild" performance of No State RL and State-based RL when policies are non-deterministic (left) and deterministic (right) during Stackelberg POMDP episodes. The results are averaged over 10 runs and shaded regions show 95% confidence intervals.

F Variations on Sellers' Learning Behavior

In our previous experiments, we assumed that both sellers restart their learning processes any time the platform rules change. This is consistent with the original experiments run by Calvano et al. (2020a), which demonstrated seller collusive behavior. However, this assumption may not hold in real-world settings, where sellers can restart their learning processes asynchronously and at any time. This behavior can present new challenges to learning an effective platform policy. Indeed, in this scenario, changes in the sellers' behavior may be caused not only by different platform interventions, but also as a result of learning restarts.

In this section, we evaluate the performance of the Stackelberg POMDP framework in scenarios where sellers randomly restart their exploration rate during training. Specifically, we assume that, at each step of the platform's learning process, each seller restarts its exploration rate with some probability. We set this probability such that, in expectation, each seller restarts its exploration once per Stackelberg POMDP episode (which corresponds to the number of steps between platform

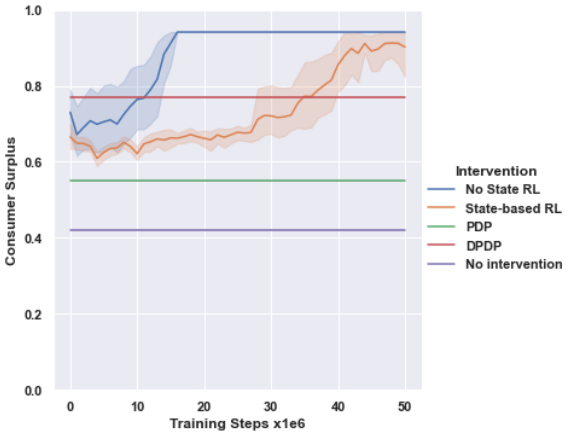

Figure 6: "Learning in the wild" performance of No State RL and State-based RL when sellers restart their exploration asynchronously during Stackelberg POMDP episodes. The results are averaged over 10 runs and shaded regions show 95% confidence intervals.

updates). Note that if restarts occur close to the reward phases, rewards may reflect an out-of-equilibrium behavior of the sellers (even if exploration is paused). To avoid this problem, we generate our plots by logging rewards in an evaluation Stackelberg POMDP episode we run every 100k training steps. These evaluation episodes use the current platform policy and operate it executing the action with the highest weight given each observation. In these episodes, the Q learning processes are run as in the previous sections, without intra-episode restarts.

As we can see from Figure 6, the Stackelberg learning framework allows us to derive close-to-optimal policies even under this less stationary behavior. Given that rewards are collected in evaluation episodes (where policies are operated via highest-weighted actions), the optimal intervention under No State RL is executed much earlier than in the simulations of Figure 2, reaching the maximum reward after only 15M training steps.

G Variations on Policy Rewards

In our main experiments, we assume that the platform can compute the consumer surplus $U(p_t; \mathcal{N}_t) = \mu \cdot \log[\lambda(p_t; \mathcal{N}_t)]$, where $\lambda(p_t; \mathcal{N}_t) = \sum_{j \in \mathcal{N}_t} \exp((\alpha_j - p_{j,t})/\mu) + \exp(\alpha_0/\mu)$ at each step t to reward the policy. We note that, however, to compute this surplus one needs to access the consumers' quality indexes α_is, which may not be available to the platform. However, we note that, to maximize consumer surplus, we can replace $U(p_t; \mathcal{N}_t)$ with any reward function that increases with respect to the number of agents displayed and decreases as prices increase. In this section, we will use $\tilde{U}(p_t; \mathcal{N}_t) = \mu \cdot \log[\tilde{\lambda}(p_t; \mathcal{N}_t)]$, where $\tilde{\lambda}(p_t; \mathcal{N}_t) = \sum_{j \in \mathcal{N}_t} \exp(-p_{j,t}/\mu)$ and show that we can achieve similar learning performance as shown in Figure 7.

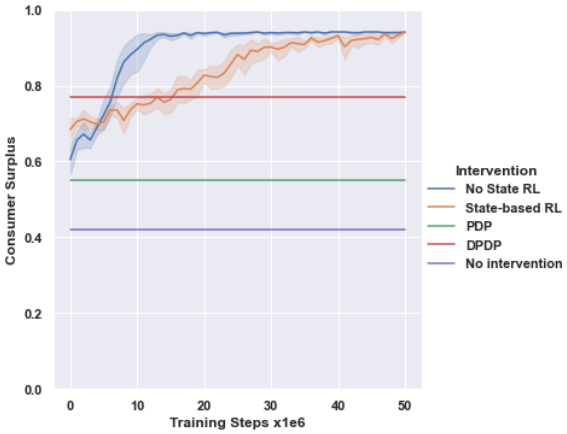

Figure 7: Learning performance under reward that does not access quality indexes.

H Selecting Stackelberg POMDP Parameters

The hyperparameters of our Stackelberg POMDP are selected the following criteria: First, we want to make sure that the equilibrium phase of our Stackelberg POMDP is long enough such that sellers' dynamics produce best responses to platform policies. At the same time, we want to avoid too long equilibrium phases, as this makes rewards too sparse. With 50k steps we have a good trade-off between these two desiderata. Regarding the reward phase, we want to make sure that this reward is representative of the converged policy reached by Q-learners, and their converged behavior is usually a single price profile or a loop of two or three price profiles. We adopt 30 reward steps to be conservative.