Online Appendix to Squeezing more Juice out of Lime: A Novel High-dimensional Pricing Algorithm

Richard Faltings*

March 17, 2024

A Additional Content

A.1 Companion Notebook

Section 2.1 is reproduced in a publicly accessible notebook on Google Colab at the following link: `https://colab.research.google.com/drive/1cMJtJbQ1jfibjtclYfh32V4R7poTqqzp?usp=sharing`

This notebook allows anyone to easily reproduce the results in that section of the paper, and to test the algorithm under alternative parameters.

A.2 Optimal Firm Relocation

The following is a formalization of the firm's relocation problem, which is not solved for in the counterfactuals. As discussed in Section 4.2, I model relocation of vehicles (and charging) as taking place overnight. Here, I further assume that the firm's relocation policy takes the form of a fixed initial vehicle distribution in the morning period, denoted $\check{Q}_f^*$. This is motivated by some of the firms' business model, which adopts a "gig economy" approach to charging and relocation. This lets workers sign up through a mobile application and find

*The University of Texas at Austin: richard.faltings@utexas.edu

vehicles needing to be charged. These firms generally require vehicles that are taken for charging to be redeployed the following morning at specific locations.[1]. Given this business model, a firm would need to commit to its deployment plan so gig workers can plan their routes more efficiently.

Intuitively, the firm must then choose its initial deployment such as to balance the value of a particular distribution of vehicles with its expected cost of relocating vehicles. Both the value and the cost of relocation will depend on its pricing policy, which affects how its vehicles will move over the course of the day.

Formally, I model the choice of initial deployment $\check{Q}_f^*$ as:

$$\check{Q}_f^* = \underset{\check{Q}_f}{\operatorname{argmax}} \mathbb{E}_{\bar{Q}_f|p_f^*,\check{Q}_f}\left[V_f(\check{Q}_f, h=1|p_f^*) - C^*(\check{Q}_f, \bar{Q}_f|C^r)\right] \tag{A.1}$$

Where $\bar{Q}_f$ is the firm's vehicle distribution at the end of the day, which is a random variable whose probability distribution depends both on the pricing policy and the initial vehicle distribution. $V_f(\check{Q}_f, h=1|p_f^*)$ is the firm's value of having the given vehicle distribution at the start of the day, defined according to the pricing problem in 7. I let $C^*(Q', Q|C^r)$ be the cost of optimally relocating vehicles from a distribution Q to a distribution Q', obtained from a canonical optimal transport problem with movement cost matrix C^r (also known as the earth mover's distance). I also use this to define $V_R(Q)$, the firm's value of ending the day with a particular vehicle distribution, which will require relocation:

$$V_R(Q) = C^*(\check{Q}_f^*, Q|C^r) \tag{A.2}$$

A.3 Counterfactual Algorithm Details

The algorithm described here solves for the parametric pricing equilibrium described in Section 4.3. For each firm $f \in \mathcal{F}$, the algorithm computes the optimal parameters θ_f^* for the firm's parametric pricing function $p_{\theta_{f,P}}(Q_f, h)$.

In addition to the iteration of the pricing function described in Equation 4 in Section 6.1, the full algorithm also makes use of a value function to reduce the variance of the estimated gradients. The intuition is that part of the expected gradient involves the profit of the firm earned at a particular state (Q_f, h), as seen in Equation 2. The profit multiplies the gradient

[1]See: https://help.bird.co/hc/en-us/articles/360031785072-24-7-Nest-Availability

on the probability of the realized demand. Thus the gradient effectively puts a greater weight on realizations of q with higher profits, making these states more likely. However, since the profits are non-negative, no states are ever *penalized* with a negative weight. Instead, we can change the weighting to an *advantage* which subtracts the expected profit from the current state. This has the effect of centering the weights, leading to lower variance in the estimated gradient and better convergence of the stochastic gradient ascent but does not modify the optimum of the problem. This closely follows the concept of Generalized Advantage Estimation of Schulman et al. (2015). The algorithm thus uses a parametric approximation $V_{\phi_f}(Q_f, h)$ for each firm $f \in \mathcal{F}$.

Recall Equation 3 describing the stochastic gradient used to update the parametric policy.

$$\tilde{\nabla}_\theta J(\theta) = \frac{1}{S}\sum_{s=1}^{S}\Big[\sum_{t=0}^{T_s}\big(\beta^t(\nabla_\theta p_\theta)q_{s,t} + (\nabla_\theta p_\theta)\underbrace{\nabla_p \log g(q_{s,t}|p,Q)}_{\text{log probability gradient}}\underbrace{\sum_{t=0}^{T_s}((p-c)'q_{s,t})}_{\approx V(Q_t,h_t)}\big)\Big] \tag{A.3}$$

The last term to be summed over is essentially the profit obtained from the sequence of demand starting from time t, which has state Q_t and h_t. In expectation, this sequence will yield exactly $V(Q_t, h_t)$, by definition of the value function itself as the continuation value from a particular state. However, there is significant variance involved in drawing the sequences of demand. This is a source of variance in the stochastic gradient itself. As described in Schulman et al. (2015) we can reduce the simulation variance by subtracting the value function from the simulated profit sequence. This doesn't change the value of the gradient in expectation, but reduces its variance. Intuitively, we can think of the sequence of profits as a weight, which is multiplied with the log probability gradient. Both of these are random variables, so the variance of their product will depend on both their covariances and the product of their expectations. By subtracting the value function from the profit weights, we reduce the expectation of the weights to 0, which also has the effect of reducing the variance of their product with the log probability gradient. Given the approximation $V_\phi(Q, h)$, we can thus use a modified gradient as follows:

$$\hat{\nabla}_\theta J(\theta) = \frac{1}{S}\sum_{s=1}^{S}\Big[\sum_{t=0}^{T_s}\Big(\beta^t(\nabla_\theta p_\theta)q_{s,t}+(\nabla_\theta p_\theta)\nabla_p \log g(q_{s,t}|p,Q)\big(\sum_{t=0}^{T_s}((p-c)'q_{s,t})-V_\phi(Q_t,h_t)\big)\Big)\Big] \tag{A.4}$$

where we subtract $V_\phi(Q_t, h_t)$ from the profit sequence $\sum_{t=0}^{T_s}((p-c)'q_{s,t})$.

At the same time, given different sequences of profit, we can also update the value function

approximation itself:

$$\hat{\nabla}_{\phi} J(\phi) = \frac{1}{S} \sum_{s=1}^{S} \Big[\nabla_{\phi} \big(\sum_{t=0}^{T_s} ((p-c)' q_{s,t}) - V_{\phi}(Q_t, h_t) \big) \Big] \tag{A.5}$$

We can now define the full price optimization algorithm used to compute the counterfactual scenarios in Algorithm 1.

Algorithm 1 Stochastic Gradient Price Optimization

for all $f \in \mathcal{F}$ **do**
  $\theta_f^0 \leftarrow$ reasonable random initialization
  $\phi_f^0 \leftarrow$ reasonable random initialization
end for
for k from 1 to K **do**
  for s from 1 to S **do**
    Simulate a full day with prices $p_{\theta_f^k} \forall f \in \mathcal{F}$ (until T_s s.t. $h_{T_s} = \bar{h}$) to obtain: $q_{s,f,t}, \quad \forall f \in \mathcal{F}, t \leq T_s$
  end for
  for all $f \in \mathcal{F}$ **do**
    Stochastic policy update: $\theta_f^{k+1} = \theta_f^k + \alpha_\theta^k \hat{\nabla}_{\theta_f^k} J(\theta_f^k)$
    Stochastic value update: $\phi_f^{k+1} = \phi_f^k + \alpha_\phi^k \hat{\nabla}_{\phi_f^k} J(\phi_f^k)$
  end for
end for

The parameters used for the various counterfactuals are displayed in Table A.1. All the value functions used were approximated by a neural network (one for each firm). Additionally the parameter update steps are performed with the Adam algorithm from Kingma and Ba (2014), which further tempers the step size for better convergence and uses a form of momentum to overcome local optima.

Tuning Parameter	Two-Part Tariff	Stock Invariant	Stock Responsive
Value learning rate α_V	5e−4	5e−4	5e−4
Price learning rate α_p	1e−3	5e−3	5e−6
Advantage Estimation Parameter λ	0.5	0.5	0.5
Batch size	150	150	150
Parallel threads	30	30	30
Batches	12000	18000	18200
Value neural network structure	$(22; 64; 64; 64; 1)$	$(22; 64; 64; 64; 1)$	$(22; 64; 64; 64; 1)$
Price neural network structure	N/A	N/A	$(22; 64; 64; 64; 1936)$

Neural networks are all MLPs with structure denoted as inputs, neurons in hidden layers, and outputs
See Schulman et al. (2015) for an explanation of the λ parameter

Table A.1: Training Parameters

Additionally, the expected walking distance, which enters into the utility (see Equation 5), is re-computed in the counterfactuals by modeling the expected walking distance as a function of the number of scooters, with the following functional form:

$$\mathbb{E}(w_{\ell,f}|Q_{\ell,f};\omega) = \frac{\omega^1_{\ell,f}}{\omega^2_{\ell,f}Q_{\ell,f} + \omega^3_{\ell,f}} + \omega^4_{\ell,f} \tag{A.6}$$

When both vehicles and consumers are uniformly distributed, the above formula can be analytically derived with $\omega^3_{\ell,f} = 0$ and $\omega^4_{\ell,f} = 0$. These additional parameters are added to account for non-normality, and all four parameters are fitted on the original data. An example of the fit can be found in Figure A.4.

All computation was carried out on the Texas Advanced Computing Center's Stampede2 cluster, with runtimes of up to 48 hours depending on the pricing function specification.

B Additional Tables

Firm	Unlock fee (SQ)	Minute fee (SQ)	Unlock fee (CF)	Minute fee (CF)
Bird	\$1.00	\$0.39	\$0.95	\$0.08
Jump (Bike)	\$0.00	\$0.25	\$1.12	\$0.07
Jump (Scooter)	\$0.00	\$0.25	\$1.11	\$0.06
Lime	\$1.00	\$0.24	\$1.29	\$0.09
Lyft	\$1.00	\$0.24	\$1.06	\$0.08
Razor	\$1.00	\$0.24	\$1.15	\$0.05
Skip	\$1.00	\$0.25	\$1.22	\$0.09
Spin	\$0.00	\$0.29	\$1.16	\$0.05

SQ: Status-quo. CF: Counterfactual

Table A.2: Two-part tariff comparison

C Additional Figures

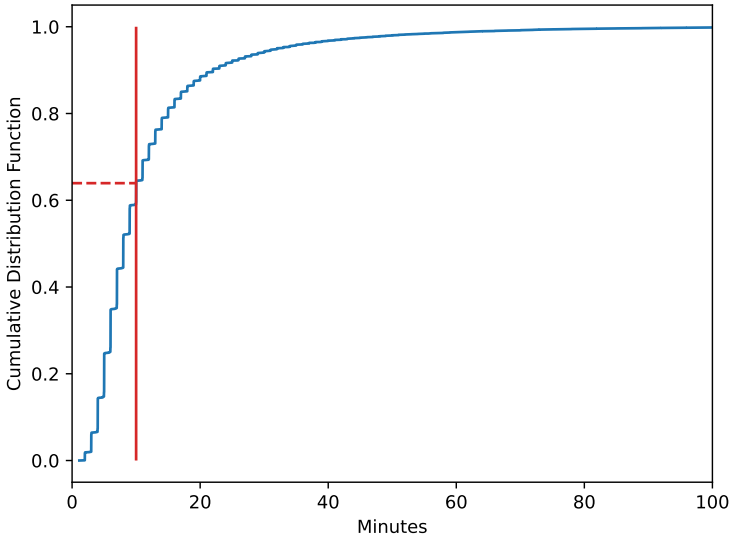


Figure A.1: CDF of trip durations for Jump scooters

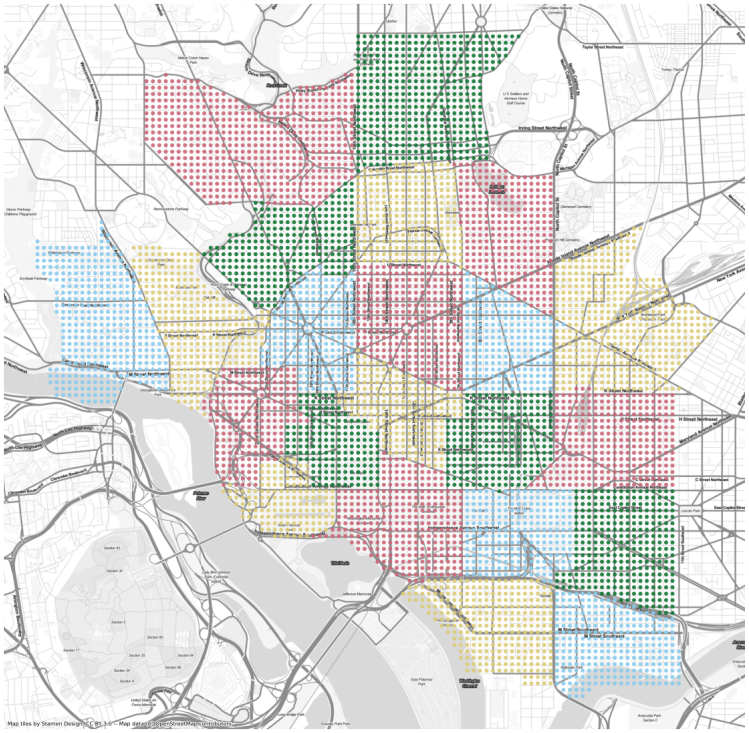


Figure A.2: Fine grid of points used to compute expected walking distances.

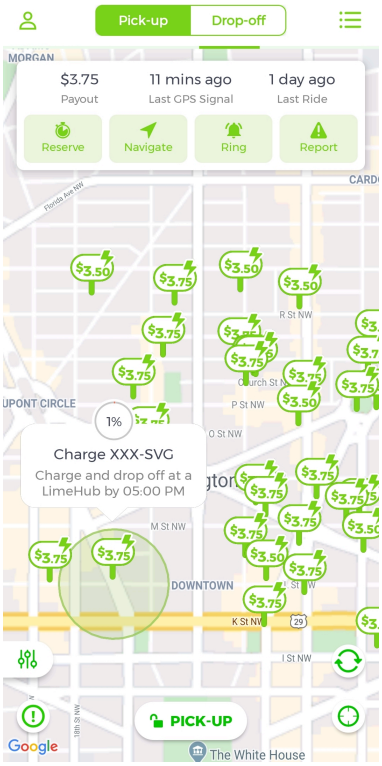


Figure A.3: Charging tasks and rewards for Lime scooters. Dated May 18th, 2020.

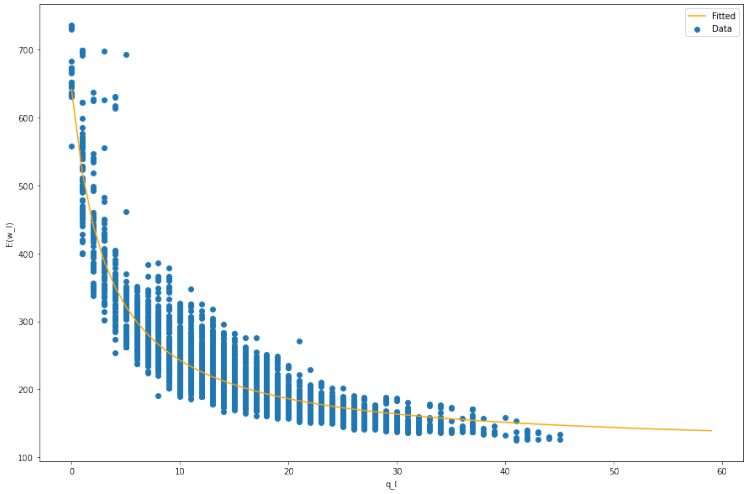


Figure A.4: Example of the fitted walking distance function

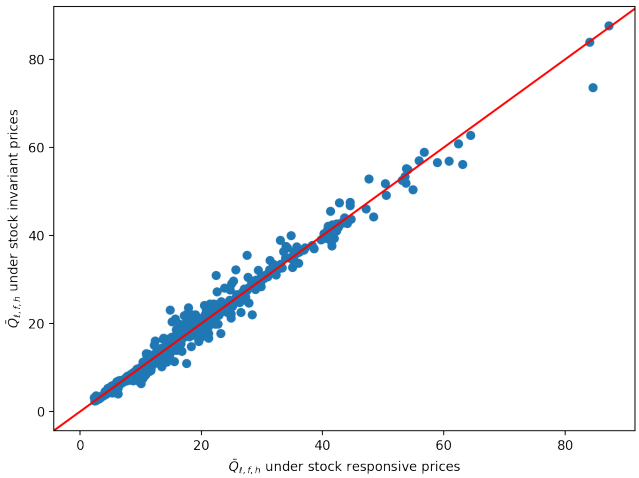


Figure A.5: Comparison of average vehicle stocks by firm, location, and hour of day

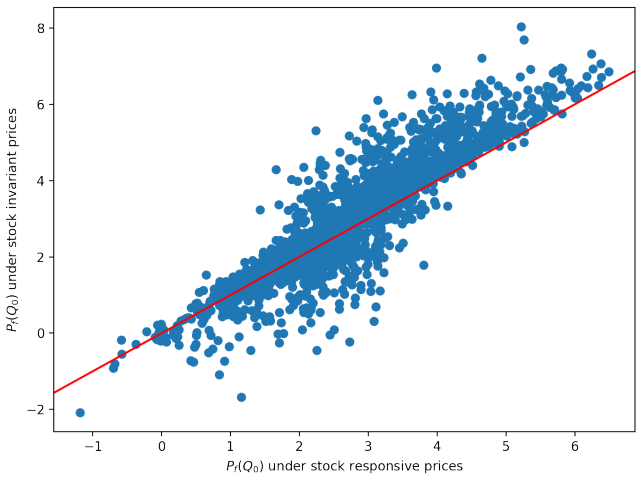
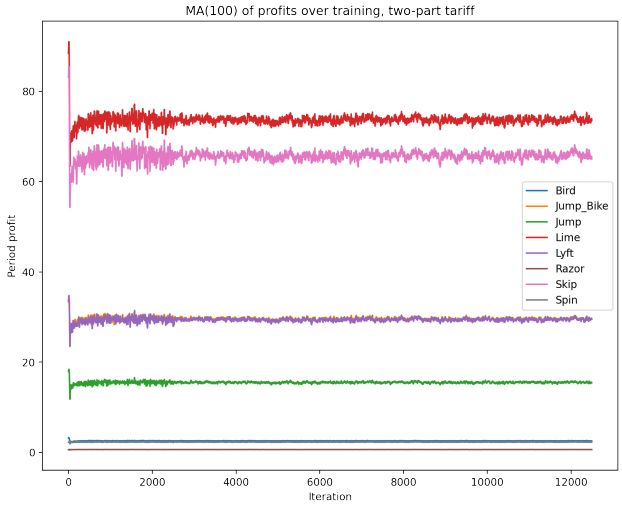
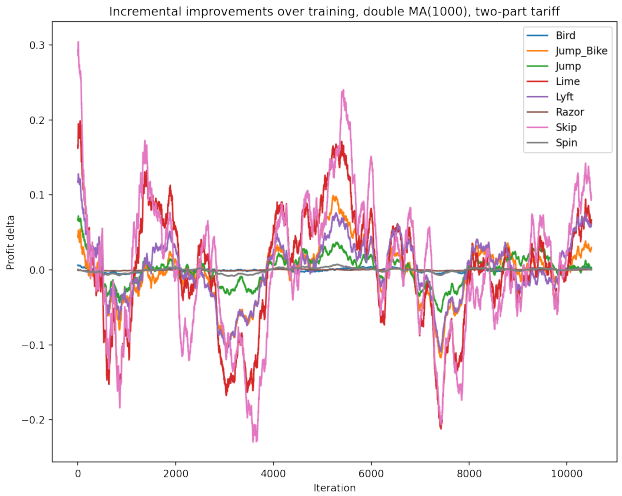


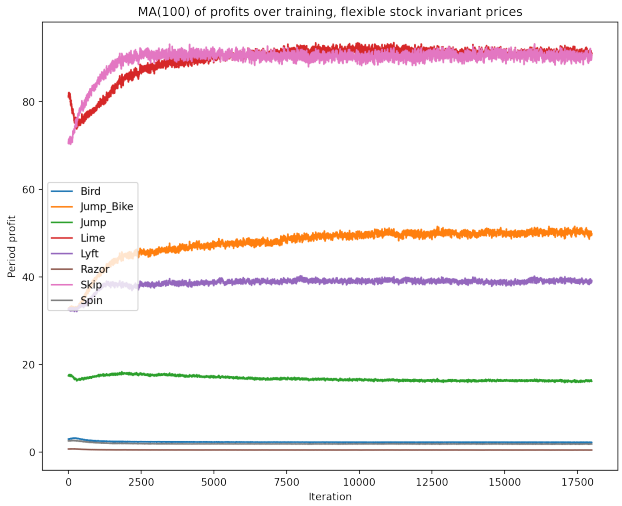
Figure A.6: Comparison of prices under initial vehicle distribution, Jump



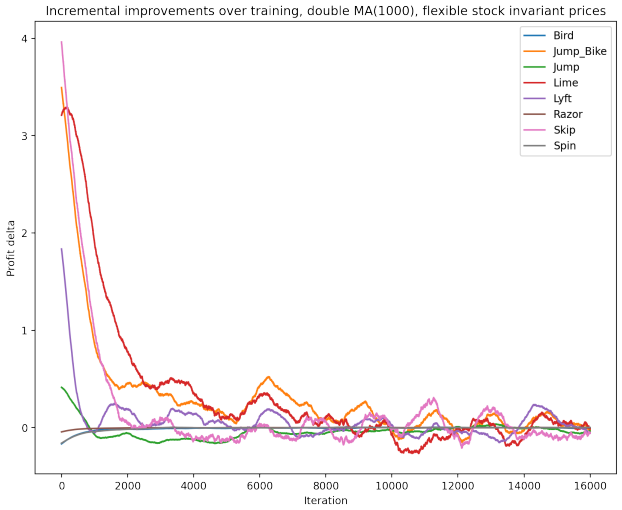
(a) Two-part tariff, profits over training



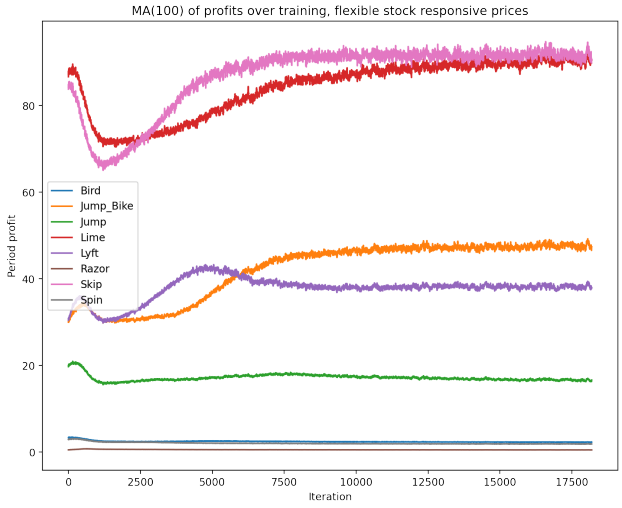
(b) Two-part tariff, training deltas



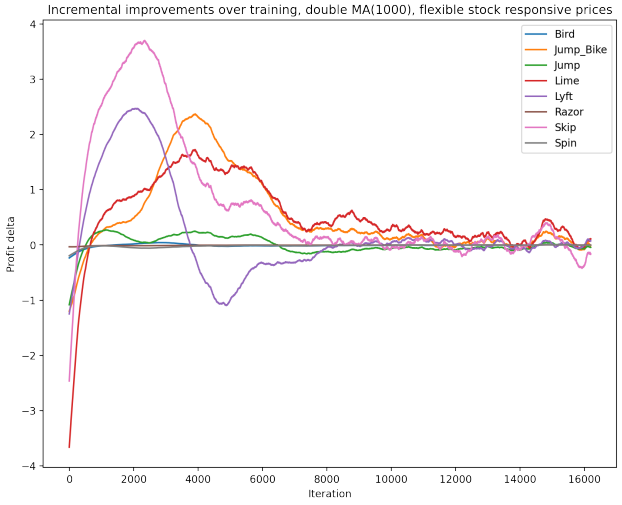
(c) Stock invariant prices, profits over training



(d) Stock invariant prices, training deltas



(e) Stock responsive prices, profits over training



(f) Stock responsive prices, training deltas

Figure A.7: Evolution of profits over training

References

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization." In: *arXiv preprint arXiv:1412.6980*.

Schulman, John et al. (2015). "High-dimensional continuous control using generalized advantage estimation." In: *arXiv preprint arXiv:1506.02438*.