Inventory Control with Lost Sales and Lead Times

Matthew Ford & Anna Poulton
05/17/21

Inventory Control

- **Positive lead times**: after an order for more inventory is placed, there's a delay before it is received
  - Delay = L time steps
- **Lost sales:** demand that is not fulfilled disappears from the system and cannot be met at a later date
  - p = penalty (per unit) of lost sales

Additionally, inventory has a holding cost: penalty of h=1 per unit of inventory on hand at the end of time t

Source: [2]

State/Action Space

I_t = inventory on hand

$x_t = (x_{1,t},...,x_{L,t})$ = 'pipeline' vector of orders that will arrive in the future

$f_t^{\pi}(I_t, x_t)$ = action (order placed at time t, depending on policy π)

State space: I_t , x_t must be non-negative

Action space: order placed must be non-negative

Source: [2]

MDP

1. Update inventory on hand: $\tilde{I}_t = I_t + x_{1,t}$
2. Simulate demand: $D_t \sim Exp(\lambda)$ ($\lambda = 1$)
3. Determine post-demand inventory: $I_{t+1} = \max(0, \tilde{I}_t - D_t)$
4. Incur costs: $C_t^\pi = h(\tilde{I}_t - D_t)^+ + p(\tilde{I}_t - D_t)^- = h * \max(0, \tilde{I}_t - D_t) + p * \max(0, -(\tilde{I}_t - D_t))$
5. Update pipeline vector: let $x_{L,t+1} = f_t^\pi(\mathbf{x}_t, I_t)$ and $x_{i,t+1} = x_{i+1,t}$ for $i = 1, \ldots, L-1$.

Goal: minimize long run average cost

$$C(\pi) = \limsup_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[C_t^\pi]$$

Source: [2]

Simple Policy

Constant-order policy: the same amount of inventory is always ordered

- characterized by a single parameter ***r***

Xin & Goldberg (2016): best constant-order policy becomes asymptotically optimal as lead time grows

Source: [2]

Optimality Bounds for Constant-Order Policy

Table A.1. When $h = 1$, values of (9) under different p and L.

Evaluation of (9)	$L=1$	$L=4$	$L=10$	$L=20$	$L=30$	$L=50$	$L=70$	$L=100$
$p=1/4$	2.13	1.08	1.00	1.00	1.00	1.00	1.00	1.00
$p=1$	3.36	1.89	1.15	1.01	1.00	1.00	1.00	1.00
$p=4$	6.42	3.99	2.62	1.72	1.34	1.08	1.02	1.00
$p=9$	12.26	6.77	4.43	3.12	2.45	1.73	1.38	1.15
$p=39$	62.26	27.60	14.86	9.62	7.62	5.75	4.75	3.81
$p=99$	204.50	85.21	41.77	24.43	18.20	12.92	10.49	8.49

Figure 6: Table A.1. from Xin & Goldberg (2016). The entries correspond to the upper bound on $C(\pi_{r_\infty})/OPT(L)$

Source: [2]

PPO and Parameterization

- Used PPO from Stable Baselines3
- Feature extractor
  - Output is fed as input into policy and value models
  - 2 layers with 64 units each
  - Used default randomized initialization
- Policy
  - Linear model -> add Gaussian noise -> pass through squashing function
  - Initialized weights to be 0 and bias to be optimal constant order amount
  - Decreased initial standard deviation of noise from default of 1 to 1/e
- Value
  - Linear model
  - Used default randomized initialization

Source: [1]

Training/Evaluation Process

- Used rollouts of length 2048 during training on a single episode until 500000 time steps had been simulated
- Evaluated and saved current policy every ~25000 simulated time steps on 8 episodes of length 20000
- Repeated this process again for all problems
- Chose the policy with the lowest evaluation cost
- Did a final evaluation on 50 episodes of length 20000 to generate our table

Our Results

	L = 1	L = 4	L = 10	L = 20	L = 30	L = 50	L = 70	L = 100
p = 1/4	1.000732	0.998664	0.979373	0.957875	0.967347	0.906137	0.909529	0.910072
p = 1	1.012899	0.995453	0.996600	0.999953	1.001167	0.814216	0.976300	0.904669
p = 4	1.110758	1.026892	1.003398	0.970293	0.993897	0.997304	0.992948	0.992391
p = 9	1.258596	1.079733	1.027929	0.950493	0.966457	0.990411	0.968652	0.944635
p = 39	1.777229	1.407220	1.205311	1.064118	0.870776	0.942677	0.955548	0.909879
p = 99	2.392048	1.823024	1.471593	1.271550	1.139031	0.992178	0.927584	0.936422

	L = 1	L = 4	L = 10	L = 20
p = 1/4	[0.998942, 1.002529]	[0.996974, 1.00036]	[0.977414, 0.981339]	[0.955984, 0.959774]
p = 1	[1.010599, 1.015209]	[0.993505, 0.997409]	[0.994277, 0.998935]	[0.998063, 1.00185]
p = 4	[1.107267, 1.11427]	[1.024105, 1.029694]	[1.000466, 1.006346]	[0.967001, 0.973607]
p = 9	[1.254425, 1.262794]	[1.076347, 1.083141]	[1.024301, 1.031582]	[0.9471, 0.95391]
p = 39	[1.770301, 1.784212]	[1.399918, 1.4146]	[1.198947, 1.211743]	[1.058125, 1.070178]
p = 99	[2.377531, 2.406744]	[1.812189, 1.833989]	[1.460911, 1.482432]	[1.260767, 1.282518]
	L = 30	L = 50	L = 70	L = 100
p = 1/4	[0.965783, 0.968915]	[0.904305, 0.907976]	[0.908077, 0.910986]	[0.908201, 0.91195]
p = 1	[0.998846, 1.0035]	[0.812317, 0.816123]	[0.974239, 0.978369]	[0.902875, 0.906471]
p = 4	[0.990942, 0.99687]	[0.993298, 1.001342]	[0.989848, 0.996067]	[0.989381, 0.99542]
p = 9	[0.962164, 0.970789]	[0.986214, 0.994643]	[0.964314, 0.97303]	[0.916613, 0.974425]
p = 39	[0.77531, 0.993054]	[0.920627, 0.96581]	[0.947945, 0.963273]	[0.897742, 0.922348]
p = 99	[1.13209, 1.146057]	[0.981016, 1.003597]	[0.915789, 0.939686]	[0.926192, 0.94688]

Us

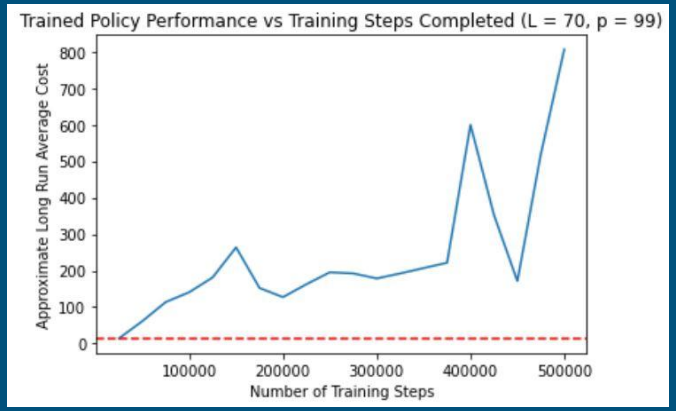
	L = 1	L = 4	L = 10	L = 20	L = 30	L = 50	L = 70	L = 100
p = 1/4	1.000732	0.998664	0.979373	0.957875	0.967347	0.906137	0.909529	0.910072
p = 1	1.012899	0.995453	0.996600	0.999953	1.001167	0.814216	0.976300	0.904669
p = 4	1.110758	1.026892	1.003398	0.970293	0.993897	0.997304	0.992948	0.992391
p = 9	1.258596	1.079733	1.027929	0.950493	0.966457	0.990411	0.968652	0.944635
p = 39	1.777229	1.407220	1.205311	1.064118	0.870776	0.942677	0.955548	0.909879
p = 99	2.392048	1.823024	1.471593	1.271550	1.139031	0.992178	0.927584	0.936422

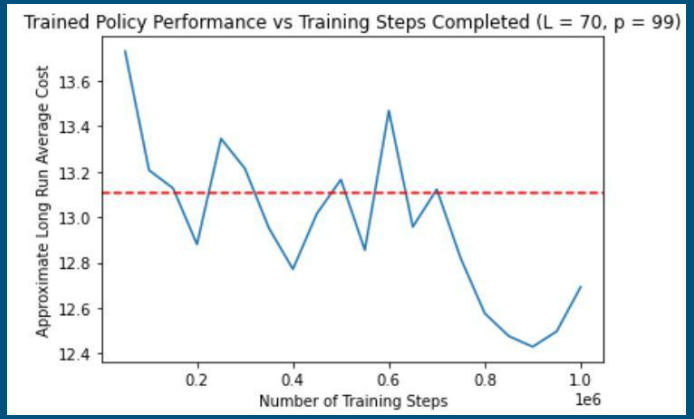
Other Group

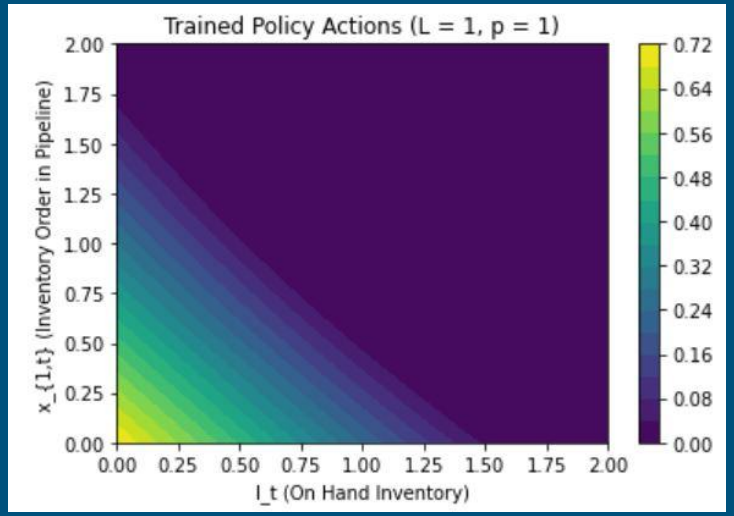
C(const_order)/C(PPO)	L=1	L=4	L=10	L=20	L=30	L=50	L=70	L=100
p=0.25	1.00368	1.00124	0.99923	0.98822	0.90195	0.88395	0.83419	0.85435
p=1	1.00970	1.00112	1.00050	0.99501	0.82632	0.83170	0.79193	0.77658
p=4	1.09048	1.04100	1.00175	0.97143	0.81546	0.81599	0.77624	0.79345
p=9	1.23925	1.13170	1.06793	1.00547	0.82239	0.79777	0.77362	0.77989
p=39	1.83981	1.46979	1.22424	1.17589	0.84864	0.78660	0.77462	0.76134
p=99	2.48444	2.30920	1.45911	1.13117	0.91039	0.81862	0.75929	0.78665

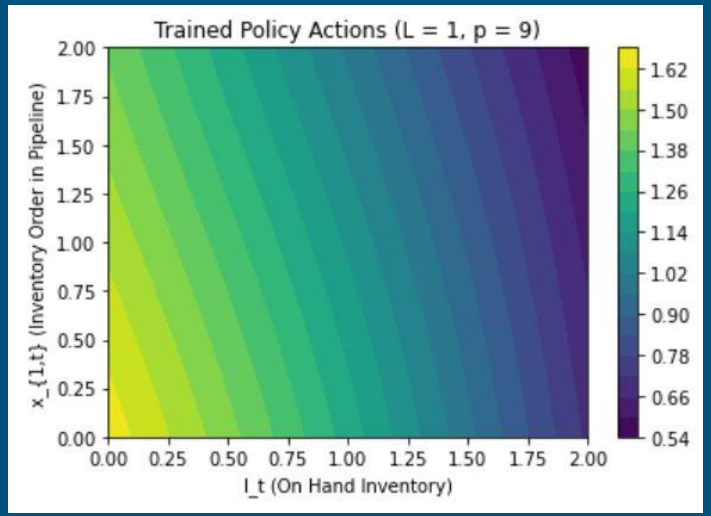
Improving when L is Large (p=99, L=70)

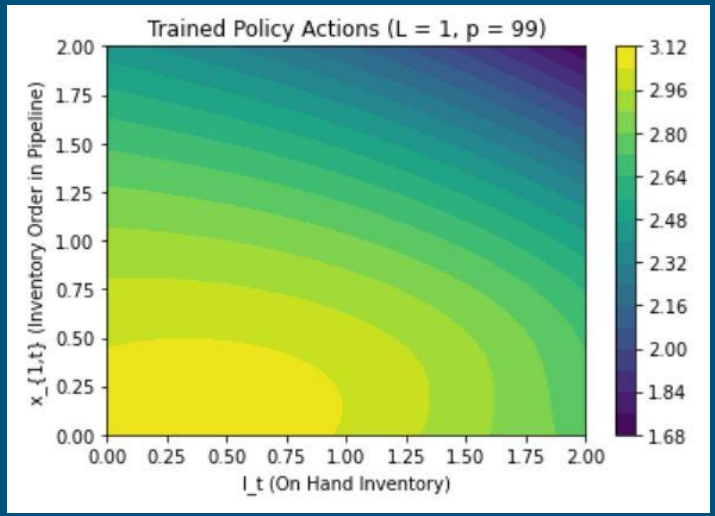
- Decreased initial standard deviation of policy to 1/e^2
- Decreased Adam learning rate from .0003 to .0002
- Increased rollout length to ~49000 during training and total time steps to 2000000
- Evaluated every ~50000 time steps using 20 episodes
- Achieved ratio of 1.0639 with 95% confidence interval [1.0543, 1.0736]






Trained Policy Actions (L = 1, p = 1)
x_{1,t} (Inventory Order in Pipeline)
I_t (On Hand Inventory)
0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75 2.00
0.72 0.64 0.56 0.48 0.40 0.32 0.24 0.16 0.08 0.00


Trained Policy Actions (L = 1, p = 9)
x_{1,t} (Inventory Order in Pipeline)
I_t (On Hand Inventory)
0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75 2.00
1.62 1.50 1.38 1.26 1.14 1.02 0.90 0.78 0.66 0.54


Trained Policy Actions (L = 1, p = 99)
x_{1,t} (Inventory Order in Pipeline)
I_t (On Hand Inventory)
0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75 2.00
3.12 2.96 2.80 2.64 2.48 2.32 2.16 2.00 1.84 1.68

References

1. Raffin, Antonin & Hill, Ashley & Ernestus, Maximilian & Gleave, Adam & Kanervisto, Anssi & Dormann, Noah. (2019). Stable Baselines3. GitHub, GitHub repository, https://github.com/DLR-RM/stable-baselines3
2. Xin, Linwei & Goldberg, David (2016). Optimality Gap of Constant-Order Policies Decays Exponentially in the Lead Time for Lost Sales Models. Operations Research. 64. 10.1287/opre.2016.1514.