Tempo adaptation in non-stationary RL

Hyunin Lee [1] Yuhao Ding [1] Jongmin Lee [1]
Ming Jin[2] Javad Lavaei[1] Somayeh Sojoudi[1]

[1]UC Berkeley

[2]Virginia Tech

September 22, 2023

Contents

1. Overlooked issue: time synchronization
2. Time-desynchronized environment
3. `ProsT` framework

Table of Contents

1 Overlooked issue: time synchronization

2 Time-desynchronized environment
- Time-elapsing Markov Decision Process
- Time-elapsing variation budget

3 ProsT framework

Conventional non-stationary RL environment

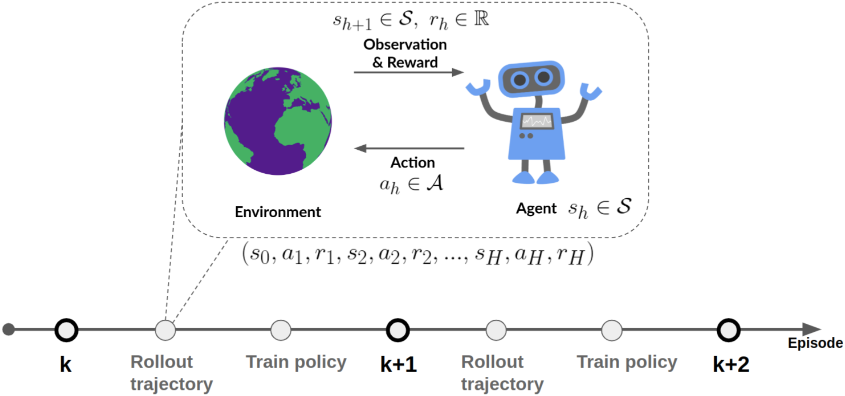


Figure 1: Iterative process: collect data, train policy

- Agent's timeline = Environment's timeline = episode

Time synchronization assumption

- Agent's timeline → episode
- Enviroment's timeline → wall-clock time

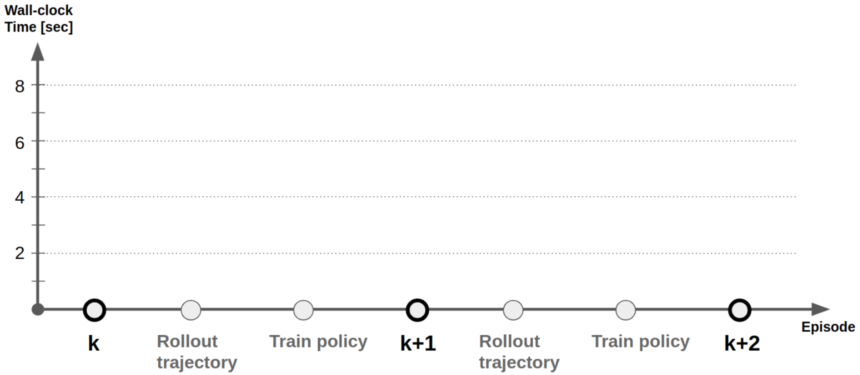


Figure 2: Environment has its own wall-clock time

Time synchronization assumption

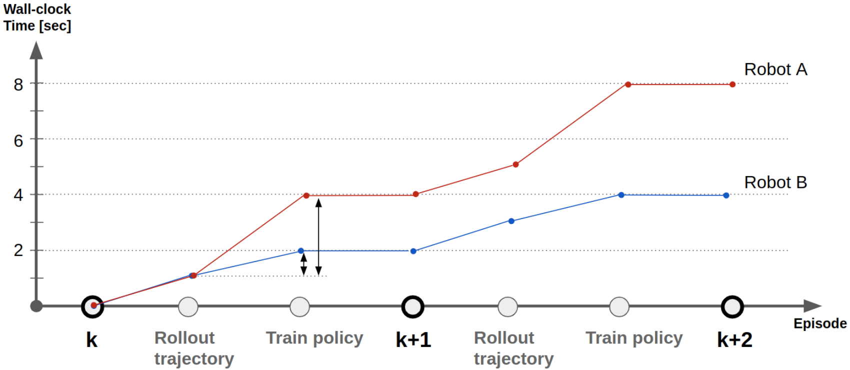
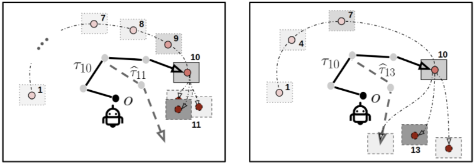


Figure 3: Different traning time makes agent encounteres different environment

Motivation example

- For fixed wall clock time duration 0 [sec] ~ 10 [sec], robot (a),(b) are reaching a gray box that is moving for every wall-clock time.
- robot strategy : predict the future box position and execute optimized policy.



t = [1,2,3,,...,10]
(a)

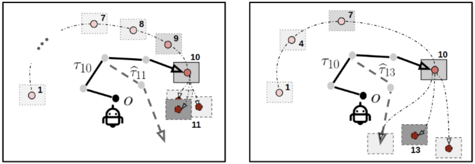
t = [1,4,7,10]
(b)

	K	Model error	sub-optimal gap
Robot (a)	10	↓	↑
Robot (b)	4	∧ ↑	∨ ↓

Figure 4: Trade-off between Model accuracy and policy accuracy

Motivation example

- For fixed wall clock time duration 0 [sec] ~ 10 [sec], robot (a),(b) are reaching a gray box that is moving for every wall-clock time.
- robot strategy : predict the future box position and execute optimized policy.



t = [1,2,3,,...,10]
(a)

t = [1,4,7,10]
(b)

	K	Model error	sub-optimal gap
Robot (a)	10	↓	↑
Robot (b)	4	∧ ↑	∨ ↓

Figure 4: Trade-off between Model accuracy and policy accuracy

- What is optimal k?

Contribution & Overview

- Environmental changes occur over wall-clock time ($\mathfrak{t}$) rather than episode progress (k), where wall-clock time signifies the actual elapsed time within the fixed duration $\mathfrak{t} \in [0, T]$.

Contribution & Overview

- Environmental changes occur over wall-clock time ($\mathfrak{t}$) rather than episode progress (k), where wall-clock time signifies the actual elapsed time within the fixed duration $\mathfrak{t} \in [0, T]$.
- In existing works, at episode k, the agent rollouts a trajectory and trains a policy before transitioning to episode $k+1$.

Contribution & Overview

- Environmental changes occur over wall-clock time ($\mathfrak{t}$) rather than episode progress (k), where wall-clock time signifies the actual elapsed time within the fixed duration $\mathfrak{t} \in [0, T]$.
- In existing works, at episode k, the agent rollouts a trajectory and trains a policy before transitioning to episode $k+1$.
- In the context of the time-desynchronized environment, however, the agent at time $\mathfrak{t}_k$ allocates $\Delta\mathfrak{t}$ for trajectory generation and training, subsequently moves to the next episode at $\mathfrak{t}_{k+1} = \mathfrak{t}_k + \Delta\mathfrak{t}$.

Contribution & Overview

- Environmental changes occur over wall-clock time ($\mathfrak{t}$) rather than episode progress (k), where wall-clock time signifies the actual elapsed time within the fixed duration $\mathfrak{t} \in [0, T]$.
- In existing works, at episode k, the agent rollouts a trajectory and trains a policy before transitioning to episode $k+1$.
- In the context of the time-desynchronized environment, however, the agent at time $\mathfrak{t}_k$ allocates $\Delta\mathfrak{t}$ for trajectory generation and training, subsequently moves to the next episode at $\mathfrak{t}_{k+1} = \mathfrak{t}_k + \Delta\mathfrak{t}$.
- Despite a fixed total episode (K), the agent accumulates different trajectories influenced by the choice of *interaction times* $(\mathfrak{t}_1, \mathfrak{t}_2, ..., \mathfrak{t}_K)$, significantly impacting the sub-optimality gap of policy.

Contribution & Overview

- Environmental changes occur over wall-clock time ($\mathfrak{t}$) rather than episode progress (k), where wall-clock time signifies the actual elapsed time within the fixed duration $\mathfrak{t} \in [0, T]$.
- In existing works, at episode k, the agent rollouts a trajectory and trains a policy before transitioning to episode $k+1$.
- In the context of the time-desynchronized environment, however, the agent at time $\mathfrak{t}_k$ allocates $\Delta\mathfrak{t}$ for trajectory generation and training, subsequently moves to the next episode at $\mathfrak{t}_{k+1} = \mathfrak{t}_k + \Delta\mathfrak{t}$.
- Despite a fixed total episode (K), the agent accumulates different trajectories influenced by the choice of *interaction times* $(\mathfrak{t}_1, \mathfrak{t}_2, ..., \mathfrak{t}_K)$, significantly impacting the sub-optimality gap of policy.
- We propose a Proactively Synchronizing Tempo (`ProST`) framework that computes optimal $\{\mathfrak{t}_1, \mathfrak{t}_2, ..., \mathfrak{t}_K\}(= \{\mathfrak{t}\}_{1:K})$.

Contribution & Overview

- Environmental changes occur over wall-clock time ($\mathfrak{t}$) rather than episode progress (k), where wall-clock time signifies the actual elapsed time within the fixed duration $\mathfrak{t} \in [0, T]$.
- In existing works, at episode k, the agent rollouts a trajectory and trains a policy before transitioning to episode $k+1$.
- In the context of the time-desynchronized environment, however, the agent at time $\mathfrak{t}_k$ allocates $\Delta\mathfrak{t}$ for trajectory generation and training, subsequently moves to the next episode at $\mathfrak{t}_{k+1} = \mathfrak{t}_k + \Delta\mathfrak{t}$.
- Despite a fixed total episode (K), the agent accumulates different trajectories influenced by the choice of *interaction times* $(\mathfrak{t}_1, \mathfrak{t}_2, ..., \mathfrak{t}_K)$, significantly impacting the sub-optimality gap of policy.
- We propose a Proactively Synchronizing Tempo (`ProST`) framework that computes optimal $\{\mathfrak{t}_1, \mathfrak{t}_2, ..., \mathfrak{t}_K\}(= \{\mathfrak{t}\}_{1:K})$.
- Our main contribution is that we show optimal $\{\mathfrak{t}\}_{1:K}$ strikes a balance between the policy training time (**agent tempo**) and how fast the environment changes (**environment tempo**).

Table of Contents

1 Overlooked issue: time synchronization

2 Time-desynchronized environment
- Time-elapsing Markov Decision Process
- Time-elapsing variation budget

3 ProsT framework

Time-elapsing Markov Decision Process

Conventional MDP

- MDP at episode k is $\mathcal{M}_k \coloneqq \langle \mathcal{S}, \mathcal{A}, H, P_k, R_k \rangle$
- For total episode K, agent interacts with $\{\mathcal{M}_1, \mathcal{M}_2, ..., \mathcal{M}_K\}$

Time elapsing MDP

- Wall clock time : $\mathfrak{t} \in [0, T]$
- MDP at wall cock time $\mathfrak{t}$ is $\mathcal{M}_{\mathfrak{t}} = \langle \mathcal{S}, \mathcal{A}, H, P_{\mathfrak{t}}, R_{\mathfrak{t}} \rangle$
- For given T, agent chooses K, then chooses $\{\mathfrak{t}_1, \mathfrak{t}_2, ..., \mathfrak{t}_K\} \in [0, T]$, then interacts with $\{\mathcal{M}_{\mathfrak{t}_1}, \mathcal{M}_{\mathfrak{t}_2}, ..., \mathcal{M}_{\mathfrak{t}_K}\}$

Time-elapsing variation budget

Conventional variation budget (VB)

- VB quantifies the environment's non-stationarity
  - $B_p \coloneqq \sum_{k=1}^{K-1} \sup_{s,a} ||P_{k+1}(\cdot\ |s,a) - P_k(\cdot\ |s,a)||_1$
  - $B_r \coloneqq \sum_{k=1}^{K-1} \sup_{s,a} |R_{k+1}(s,a) - R_k(s,a)|$

Time elapsing variation budget

- Assume policy training time Δ_π = interval $\Delta \mathfrak{t} = \mathfrak{t}_{k+1} - \mathfrak{t}_k, \forall k$
  - $B_p(\Delta_\pi) \coloneqq \sum_{k=1}^{K-1} \sup_{s,a} ||P_{\mathfrak{t}_{k+1}}(\cdot\ |s,a) - P_{\mathfrak{t}_k}(\cdot\ |s,a)||_1$
  - $B_r(\Delta_\pi) \coloneqq \sum_{k=1}^{K-1} \sup_{s,a} |R_{\mathfrak{t}_{k+1}}(s,a) - R_{\mathfrak{t}_k}(s,a)|$

Assumption (Drifting environment)

$\exists c, \alpha_p, \alpha_r \geq 0$ that satisfies $B_p(c\Delta_\pi) = c^{\alpha_p} B_p(\Delta_\pi)$, $B_r(c\Delta_\pi) = c^{\alpha_r} B_r(\Delta_\pi)$. We call α_p, α_r as drifting constants

Table of Contents

1 Overlooked issue: time synchronization

2 Time-desynchronized environment
- Time-elapsing Markov Decision Process
- Time-elapsing variation budget

3 ProsT framework

Overview

For given $t \in [0, T]$, ProST framework computes K^*, $\{t_1^*, t_2^*, .., t_{K^*}^*\}$, then $\{\pi_{t_1^*}, \pi_{t_2^*}, .., \pi_{t_{K^*}^*}\}$ into two components

- Time optimizer
- Future policy optimizer

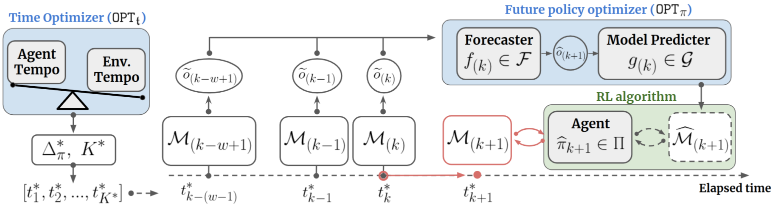


Figure 5: ProST framework

Preliminary

Definition (Value function)

For any given policy π and the MDP $\mathcal{M}_{(k)}$, We denote the state value function at episode k as $V^{\pi,k} : \mathcal{S} \to \mathbb{R}$

$$V^{\pi,k}(s) \coloneqq \mathbb{E}_{\mathcal{M}_{(k)},\pi}\left[\sum_{h=0}^{H-1} \gamma^h r_h^k \;\middle|\; s_0 = s\right]$$

Definition (Dynamic regret)

In non-stationary MDPs, the optimality of the policy is evaluated in terms of dynamic regret $\mathfrak{R}\left(\{\pi_{(1)}, \pi_{(2)}, \dots, \pi_{(K)}\}, K\right)$.

$$\mathfrak{R}\left(\{\pi_k\}_{(1:K)}, K\right) \coloneqq \sum_{k=1}^{K}\left(V^{*,k}(s_0) - V^{\pi^k,k}(s_0)\right)$$

Future policy optimizer

For given $\mathfrak{t}_k$, $\mathfrak{t}_{k+1}$, it computes a near-optimal policy of $\mathfrak{t}_{k+1}$ at time $\mathfrak{t}_k$

Assumption (Observable non-stationary set $\mathcal{O}$)

Non-stationarity of $\mathcal{M}_{\mathfrak{t}_k}$ be fully characterized by $o_{\mathfrak{t}_k} \in \mathcal{O}$.

Estimate the future MDP model and optimize.

- At $\mathfrak{t} = \mathfrak{t}_k$
- During $\mathfrak{t} \in (\mathfrak{t}_k, \mathfrak{t}_{k+1})$
  1. $\hat{o}_{(k+1)} = f_{(k)}(\{\tilde{o}\}_{(k-w+1,k)})$
  2. $(\widehat{R}_{(k+1)}(s,a), \widehat{P}_{(k+1)}(\cdot|s,a)) = g_{(k)}(s, a, \hat{o}_{k+1})$
  3. $\widehat{\pi}_{(k+1)} \leftarrow \widehat{\mathcal{M}}_{(k+1)} = \langle \mathcal{S}, \mathcal{A}, H, \widehat{P}_{(k+1)}, \widehat{R}_{(k+1)}, \gamma \rangle$
- At $\mathfrak{t} = \mathfrak{t}_{k+1}$

Time optimizer

For given T,

- Time optimizer computes optimal training time $\Delta_\pi^* \in (0, T)$
- $K^* = \lfloor T/\Delta_\pi^* \rfloor$
- $\mathfrak{t}_k^* = \mathfrak{t}_1 + \Delta_\pi^* \cdot (k-1)$ for all $k \in [K^*]$

Definition (Model prediction error)

At time $\mathfrak{t}_k$, Future policy optimizer generates $\widehat{\mathcal{M}}_{(k+1)}$ and computes $\widehat{\pi}^{(k+1)}$. For any (s,a), denote the state value function and state action value function of $\widehat{\pi}_{(k+1)}$ in $\widehat{\mathcal{M}}_{(k+1)}$ at step $h \in [H]$ as $\widehat{V}_h^{(k+1)}(s)$ and $\widehat{Q}_h^{(k+1)}(s,a)$. Then, we define model prediction error $\iota_h^{k+1}(s,a)$ as follows.

$$\iota_h^{k+1}(s,a) = \left(R_{(k+1)} + \gamma P_{(k+1)} \widehat{V}_{h+1}^{(k+1)} - \widehat{Q}_h^{(k+1)}\right)(s,a)$$

Time optimizer

Strategy : Δ_π^* is a minimizer of the dynamic regret's upperbound

- Analysis on finite space $|\mathcal{S}|, |\mathcal{A}| < \infty \rightarrow$ `ProST-T`

Theorem (ProST-T dynamic regret $\mathfrak{R}$)

Let $\iota_H^K = \sum_{k=1}^{K-1} \sum_{h=0}^{H-1} \iota_h^{k+1}(s_h^{k+1}, a_h^{k+1})$ and $\bar{\iota}_\infty^K := \sum_{k=1}^{K-1} \|\bar{\iota}_\infty^{k+1}\|_\infty$, where ι_H^K is a data-dependent error. For given $p \in (0,1)$, the dynamic regret of the forecasted policies $\{\widehat{\pi}_{k+1}\}_{(1:K-1)}$ of ProST-T is upper bounded with probability $1 - p/2$,

$$\mathfrak{R}\left(\{\widehat{\pi}_{k+1}\}_{(1:K-1)}, K)\right) \leq \mathfrak{R}_I + \mathfrak{R}_{II} + C_I[p] \cdot \sqrt{K-1}$$

where $\mathfrak{R}_I = \bar{\iota}_\infty^K/(1-\gamma) - \iota_H^K$, $\mathfrak{R}_{II} = C_{II}[\Delta_\pi] \cdot (K-1)$, and $C_I[p], C_{II}[\Delta_\pi]$ are functions of p, Δ_π, respectively.

- $\mathfrak{R}_I \leftarrow$ Forecasting model error $\leftarrow B(\Delta_\pi)$ (rate of environment's change)
- $\mathfrak{R}_{II} \leftarrow$ Policy optimization error $\leftarrow \Delta_\pi$ (rate of agent's adaption)
- $\Delta_{*\pi}$ strikes a balance between $\mathfrak{R}_I$ and $\mathfrak{R}_{II}$

Δ_π bounds for sublinear $\mathfrak{R}_{II}$

Δ_π^* should satisfy sublinear dynamic regret to K

- δ : approximation gap
- τ : entropy regularization parameter
- η : learning rate

Proposition (Δ_π bounds for sublinear $\mathfrak{R}_{II}$)

From the given MDP, we have a fixed horizon H. For any $\epsilon > 0$ that satisfies $H = \Omega\left(\log\left((\widehat{r}_{max} \vee r_{max})/\epsilon\right)\right)$, we choose δ, τ, η to satisfy $\delta = \mathcal{O}(\epsilon)$, $\tau = \Omega(\epsilon / \log|\mathcal{A}|)$, $\eta \leq (1-\gamma)/\tau$. Now, set $\mathbb{N}_{II}$ as follows.

$$\mathbb{N}_{II} = \left\{ n \mid n > \frac{1}{\eta\tau} \log\left(\frac{C_1(\gamma+2)}{\epsilon}\right), n \in \mathbb{N} \right\}$$

Then,

$$\mathfrak{R}_{II} \leq 4\epsilon(K-1).$$

Any $\epsilon = \mathcal{O}((K-1)^{\alpha-1})$ for any $\alpha \in [0,1)$ satisfy sublinear $\mathfrak{R}_I$.

$\mathfrak{R}_I \leftarrow$ Forecasting model error $\leftarrow B(\Delta_\pi)$

SW-LSE : Sliding window regularized LSE

Theorem (Dynamic regret $\mathfrak{R}_I$ when f = SW-LSE)

For given $p \in (0,1)$, if the exploration bonus constant β and regularization parameter λ satisfy $\beta = \Omega(|\mathcal{S}|H\sqrt{\log(H/p)})$, $\lambda \geq 1$, then the $\mathfrak{R}_I$ is bounded with probability $1-p$,

$$\mathfrak{R}_I \leq C_I[B(\Delta_\pi)] \cdot w + C_k \cdot \sqrt{\frac{1}{w}\log\left(1 + \frac{H}{\lambda}w\right)}$$

where $C_I[B(\Delta_\pi)] = (1/(1-\gamma) + H) \cdot B_r(\Delta_\pi) + (1 + H\hat{r}_{max})\gamma/(1-\gamma) \cdot B_p(\Delta_\pi)$, and C_k is a constant on the order of $\mathcal{O}(K)$.

Δ_π bounds for sublinear $\mathfrak{R}_I$

Proposition (Δ_π bounds for sublinear $\mathfrak{R}_I$)

Denote $B(1)$ as environment tempo for one policy iteration update. If environment satisfies $B_r(1) + B_p(1)\hat{r}_{max}/(1-\gamma) = o(K)$ and we choose $w = \mathcal{O}((K-1)^{2/3}/(C_I[B(\Delta_\pi)])^{2/3})$ and set $\mathbb{N}_I$ to be

$$\{n \mid n < K,\ n \in \mathbb{N}\}$$

Then,

$$\mathfrak{R}_I = \mathcal{O}\left(C_I[B(\Delta_\pi)]^{1/3}(K-1)^{2/3}\sqrt{\log\left((K-1)/C_I[B(\Delta_\pi)]\right)}\right)$$

and also satisfies sublinear $\mathfrak{R}_I$

Δ_π^* strikes a balance between $\mathfrak{R}_I$ and $\mathfrak{R}_{II}$

- $\mathfrak{R}_I$ upperbound is increasing on a interval $\mathbb{N}_I \cap \mathbb{N}_{II}$
- $\mathfrak{R}_{II}$ upperbound is decreasing on a interval $\mathbb{N}_I \cap \mathbb{N}_{II}$

Theorem (Optimal tempo Δ_π^*)

Let $k_{Env} = (\alpha_r \vee \alpha_p)^2 C_I[B(1)]$, $k_{Agent} = \log(1/(1-\eta\tau))C_1(K-1)(\gamma+2)$ *where comes from* $\mathbb{N}_{II}$*. Then* Δ_π^* *depends on the environment's drifting constants ;* **case1***:* $\alpha_r \vee \alpha_p = 0$, **case2***:* $\alpha_r \vee \alpha_p = 1$, **case3***:* $0 < \alpha_r \vee \alpha_p < 1$, **case4***:* $\alpha_r \vee \alpha_p > 1$.

- *Case1:* $\Delta_\pi^* = \infty$, *Case2:* $\Delta_\pi^* = \log_{1-\eta\gamma}(k_{Env}/k_{Agent}) + 1$
- *Case3 & 4:* $\Delta_\pi^* = \exp\left(-W\left[-\frac{\log(1-\eta\tau)}{\max(\alpha_r,\alpha_p)-1}\right]\right)$ *if* $k_{Agent} = (1-\eta\tau)k_{Env}$.

Experiment result

1. Performance : Four benchmark methods VS `ProST`
2. Ablation study : selection of f, g and optimal traning time

Performance

Benchmark methods

- MBPO : state of the art model-based policy optimization.
- Pro-OLS : policy optimization algorithm that predicts future V.
- ONPG : adaptive algorithm that fine-tunes the policy on current data.
- FTRL : adaptive algorithm that maximizes the performance on all previous data.

Table 1: Average reward returns

Speed	$B(G)$	Swimmer-v2					Halfcheetah-v2					Hopper-v2				
		Pro-OLS	ONPG	FTML	MBPO	ProST-G	Pro-OLS	ONPG	FTML	MBPO	ProST-G	Pro-OLS	ONPG	FTML	MBPO	ProST-G
1	16.14	-0.40	-0.26	-0.08	-0.08	**0.57**	-83.79	-85.33	-85.17	-24.89	**-19.69**	**98.38**	95.39	97.18	92.88	92.77
2	32.15	0.20	-0.12	0.14	-0.01	**1.04**	-83.79	-85.63	-86.46	-22.19	**-20.21**	98.78	97.34	**99.02**	96.55	98.13
3	47.86	-0.13	0.05	-0.15	-0.64	**1.52**	-83.27	-85.97	-86.26	-21.65	**-21.04**	97.70	98.18	98.60	95.08	**100.42**
4	63.14	-0.22	-0.09	-0.11	-0.04	**2.01**	-82.92	-84.37	-85.11	-21.40	**-19.55**	98.89	97.43	97.94	97.86	**100.68**
5	77.88	-0.23	-0.42	-0.27	0.10	**2.81**	-84.73	-85.42	-87.02	**-20.50**	-20.52	97.63	99.64	99.40	96.86	**102.48**
A	8.34	1.46	2.10	**2.37**	-0.08	0.57	-76.67	-85.38	-83.83	-40.67	**83.74**	104.72	**118.97**	115.21	100.29	111.36
B	4.68	**1.79**	-0.72	-1.20	0.19	0.20	-80.46	-86.96	-85.59	-29.28	**76.56**	80.83	**131.23**	110.09	100.29	127.74

*Whole training procedure is in Appendix

Ablation study

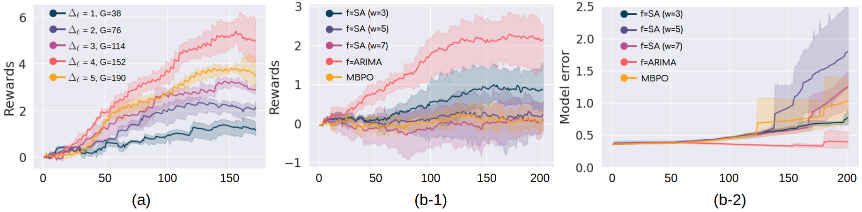


Figure 6: (a) Optimal Δ_π^*. (b-1) Different forecaster f (ARIMA, SA). (b-2) The Mean squared Error (MSE) model loss of four `ProST-G` with different forecasters(ARIMA and three SA) and the MBPO. x-axis are all episodes.