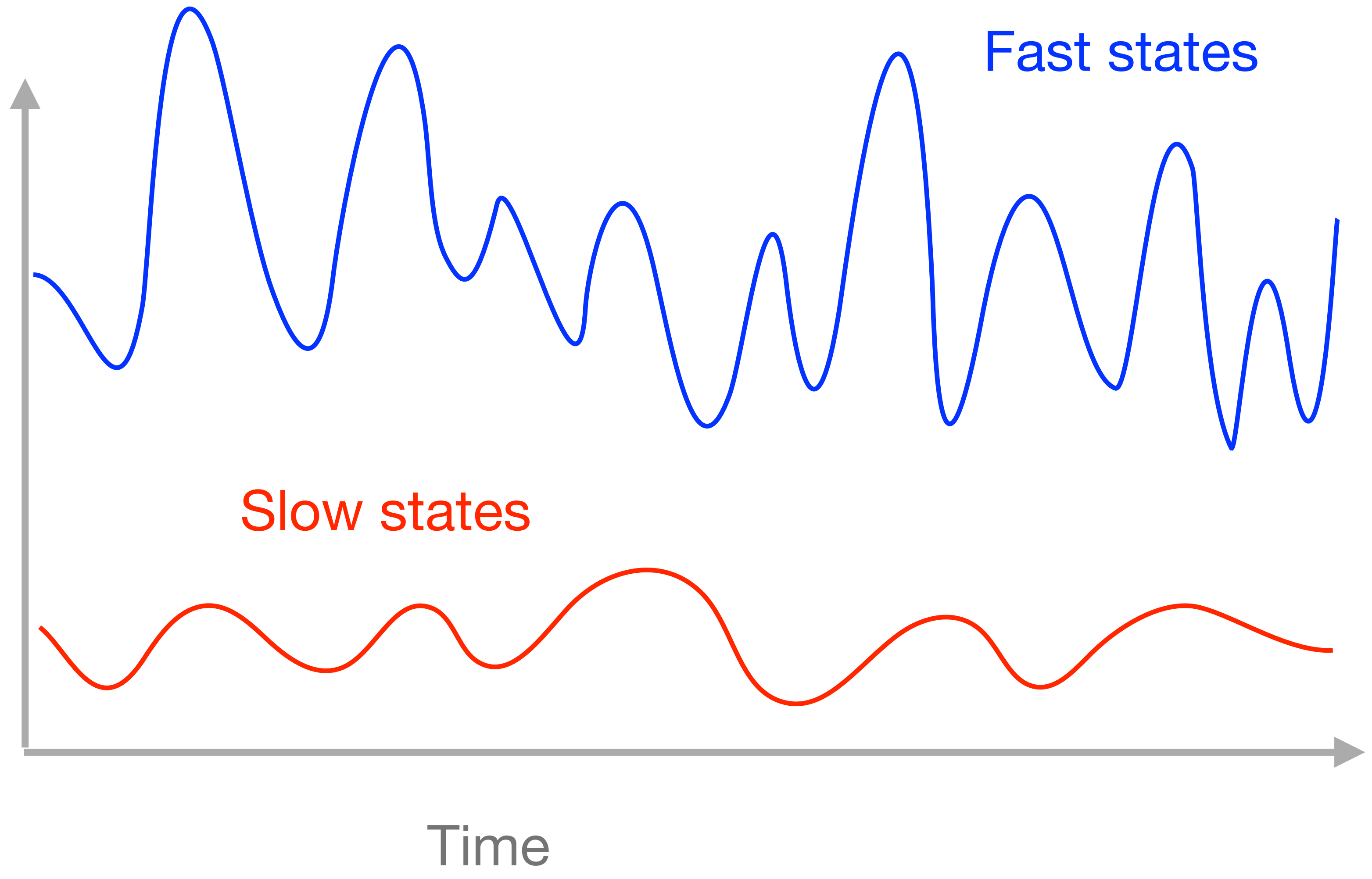
Faster RL by Freezing Slow States

Daniel Jiang (Meta, Applied RL)
joint work with Yijia Wang (Pitt)

International Symposium on Mathematical Programming
Montreal
July 23, 2024

1. Motivation via example applications

Fast and slow states

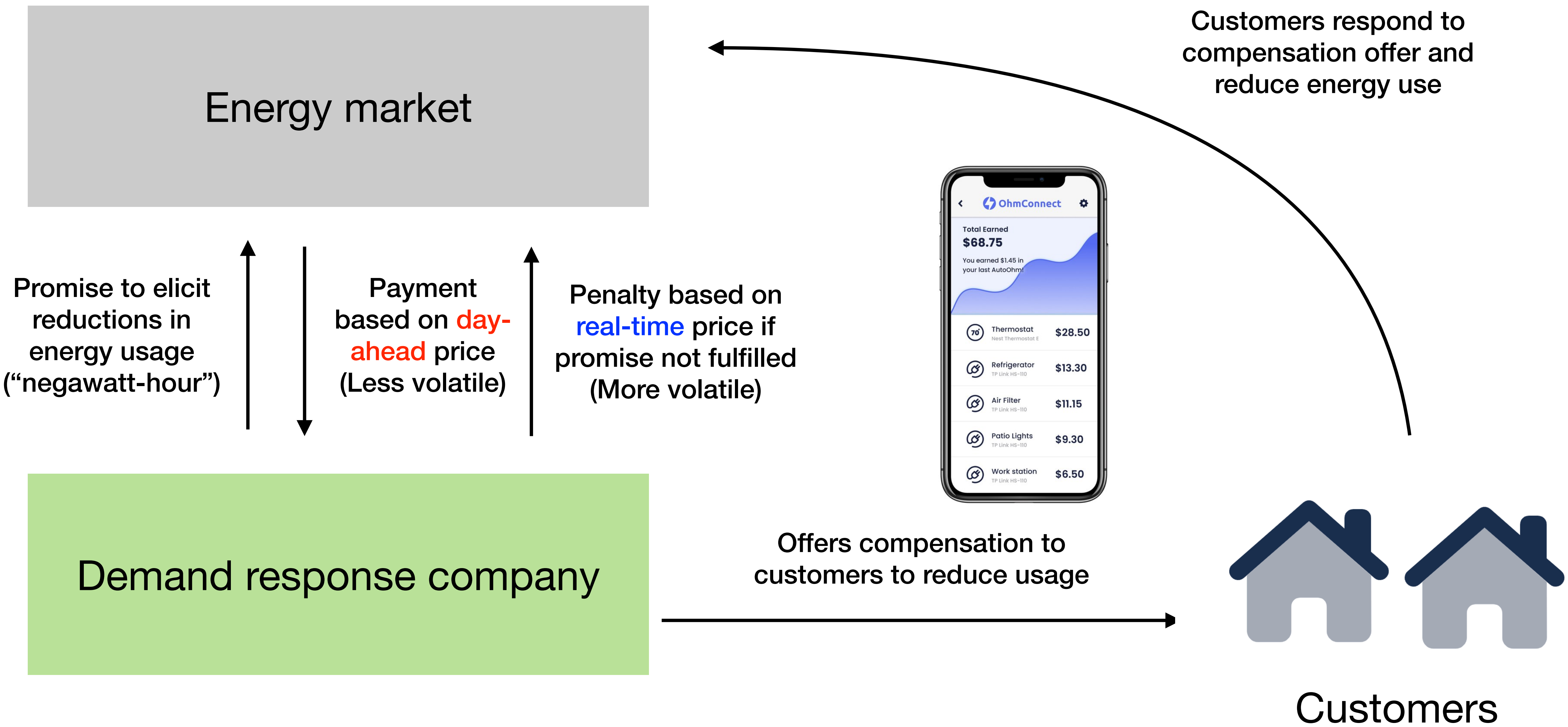


Recommendation systems

- Consider a recommendation system setting where:
  - Users return (i.e., log on) to the platform more often if they are seeing content that is *interesting* (Sumida & Zhou, 2023)
  - Users return to the platform more often if they have a *diverse* recent content history.
  - Users interests can shift over time as a function of the content they see.

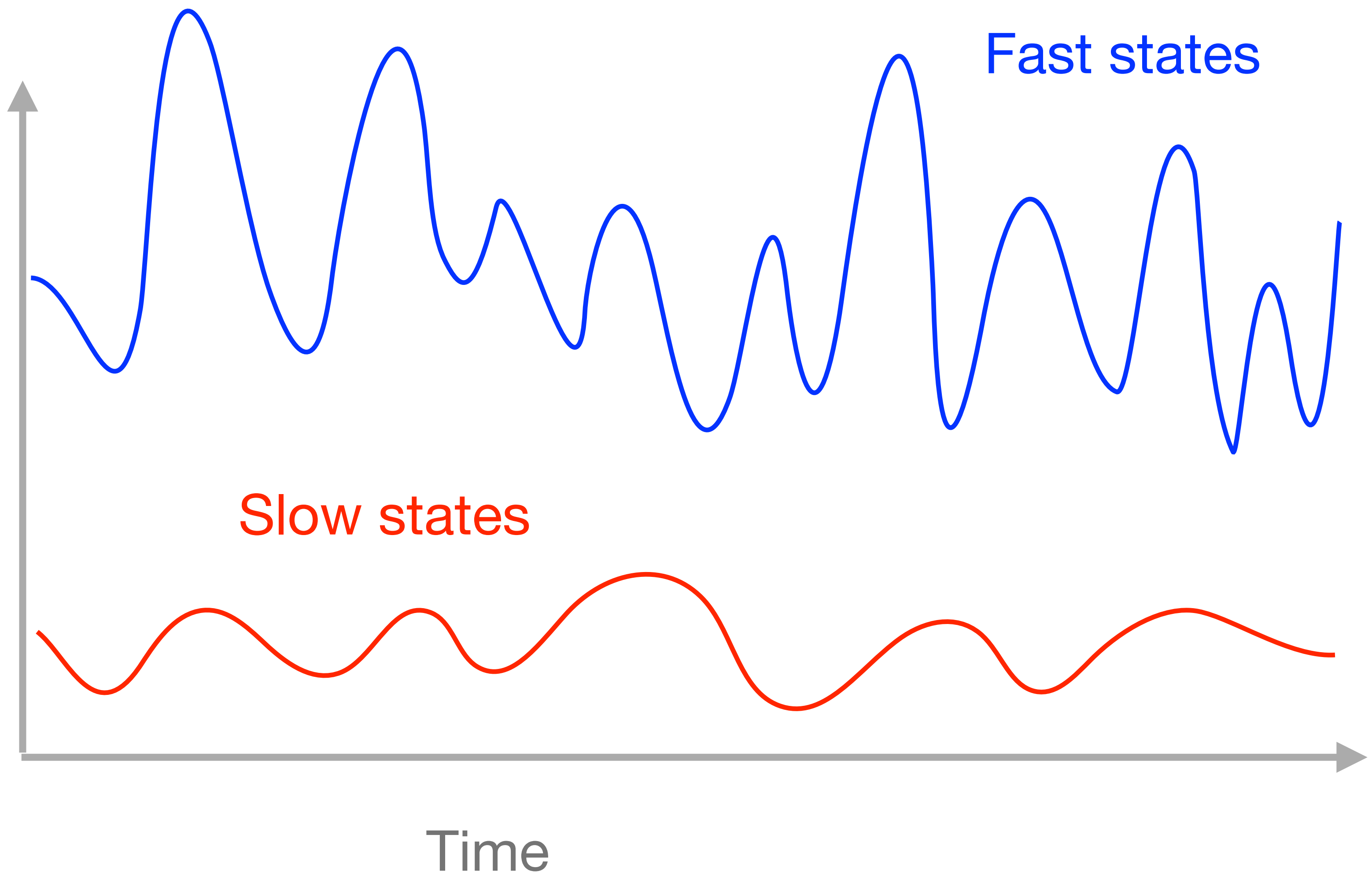
Sumida, Mika, and Angela Zhou. "Optimizing and Learning Assortment Decisions in the Presence of Platform Disengagement." Available at SSRN 4537925 (2023).

Demand response (shifting electricity demand)



K. Khezeli, W. Lin, E. Bitar. Learning to buy (and sell) demand response. *Proceedings of the International Federation of Automatic Control (IFAC) World Congress*, 2017.

What do they have in common?



Fast states from examples:

- Real-time prices
- Recent content history

Shorter timescales

Slow states from examples:

- Day-ahead prices
- Underlying user interests

Longer timescales

Current practice when modeling a new problem

- While modeling an MDP, additional state variables is expensive:
  - Each iteration of value iteration $\mathcal{O}(S^2A)$
- **What do practitioners do (anecdotally)?**
  - If a state is deemed a “slow state” (contexts, environmental variables, etc), they might be *ignored/omitted*
    - e.g. assume costs are deterministic, demand is stationary, weather doesn’t change
- **This work:** A *compromise* between computational tractability and fully ignoring the slow state
  - We propose to *periodically* ignore slow states
  - We give evidence and argue that completely omitting slow states from the decision model is often not a viable heuristic

2. Fast-slow Markov decision processes

Fast-slow Markov decision processes

- A γ-discounted, infinite horizon MDP:
  - States $s \in \mathcal{S}$
  - Actions $a \in \mathcal{A}$
  - Rewards $r(s, a) \in [0, r_{\max}]$
  - Transition function
    - $s_{t+1} = f(s_t, a_t, w_{t+1}), \; w_{t+1} \in \mathcal{W}$

- Fast-slow MDP: Slow Fast
  - States $s = (x, y) \in \mathcal{S} = (\mathcal{X} \times \mathcal{Y})$
  - Actions $a \in \mathcal{A}$
  - Rewards $r(s, a) \in [0, r_{\max}]$
  - Transition function
    - $x_{t+1} = f_{\mathcal{X}}(s_t, a_t, w_{t+1})$
    - $y_{t+1} = f_{\mathcal{Y}}(s_t, a_t, w_{t+1})$

Main assumption ("fast-slow property"):

$$\|y - f_{\mathcal{Y}}(x, y, a, w)\|_2 \leq d_{\mathcal{Y}} \quad \text{and} \quad \|x - f_{\mathcal{X}}(x, y, a, w)\|_2 \leq \alpha d_{\mathcal{Y}} \, .$$

Lipschitz assumptions (let $U^\star(s)$ be the optimal value function):

$$|\, r(s, a) - r(s', a') \,| \leq L_r \, \|(s, a) - (s', a')\|_2,$$

$$\|f(s, a, w) - f(s', a', w)\|_2 \leq L_f \, \|(s, a) - (s', a')\|_2,$$

$$\|U^\star(s) - U^\star(s')\|_2 \leq L_U \|s - s'\|_2.$$

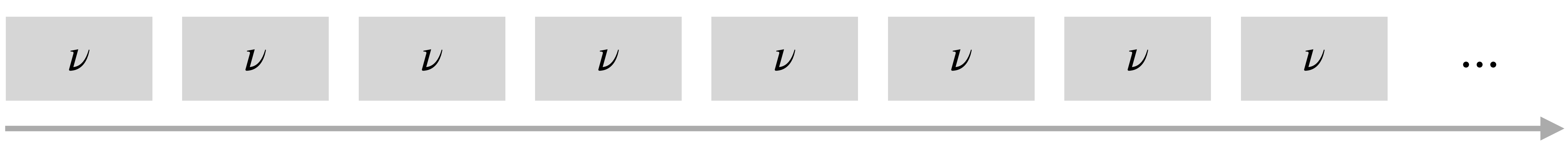
3. Hierarchical reformulation

Hierarchical reformulation (of any MDP)

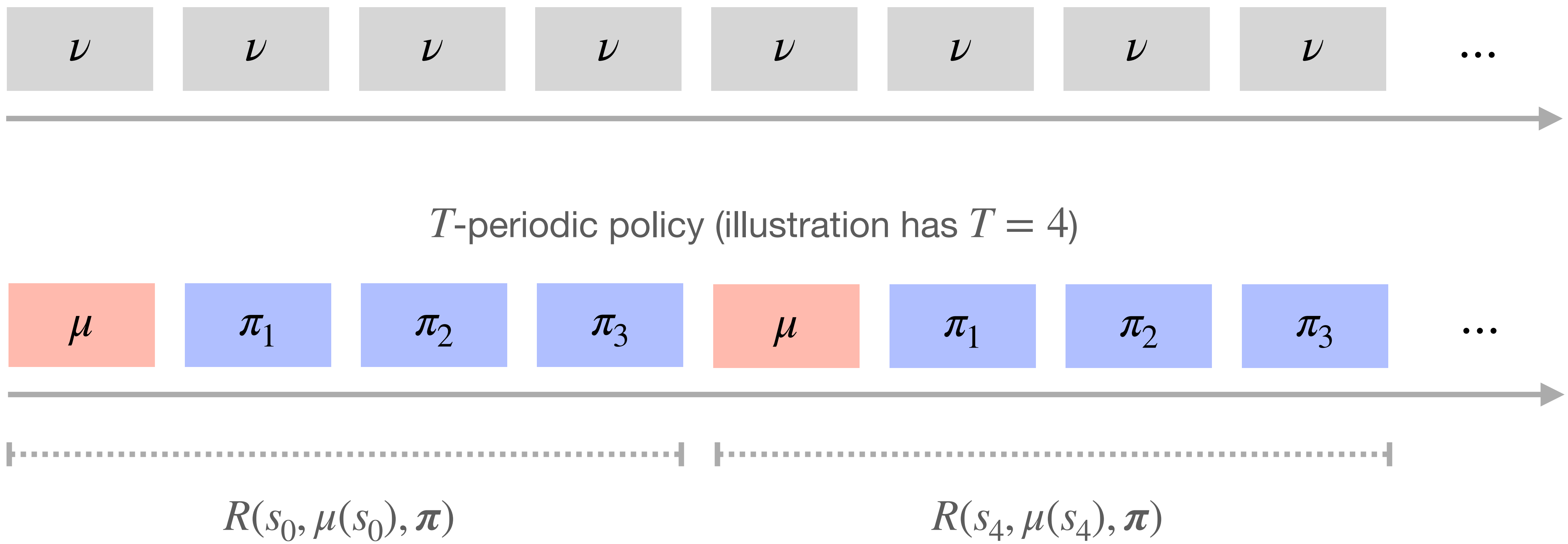
- A hierarchical reformulation is at the basis of our proposed approach
- Consider an MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{W}, f, r, \gamma \rangle$
- Let $\nu : \mathcal{S} \to \mathcal{A}$ be a stationary policy
- The value function is

$$U^\nu(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r\big(s_t, \nu\big) \,\middle|\, s_0 = s\right] = r\big(s, \nu\big) + \gamma\, \mathbb{E}\big[U^\nu(s')\big]$$

- The policy can be thought of as $(\nu, \nu, \ldots)$:



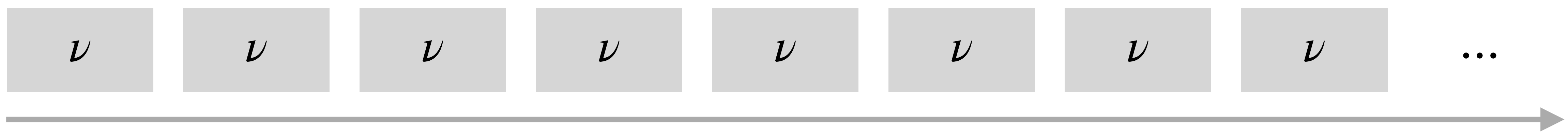
Hierarchical reformulation (of any MDP)



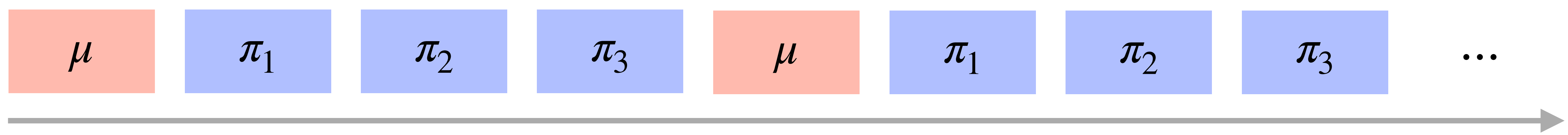
- Given a T-periodic policy $(\mu, \boldsymbol{\pi}) = (\mu, \pi_1, \ldots, \pi_{T-1})$, T-horizon reward is

$$R(s_0, \mu(s_0), \boldsymbol{\pi}) = r(s_0, \mu) + \sum_{t=1}^{T-1} \gamma^t \, r(s_t, \pi_t)$$

Hierarchical reformulation (of any MDP)



T-periodic policy (illustration has $T = 4$)



- Bellman equations of the base model and its hierarchical reformulation are:

$$U^\star(s_0) = \max_{a} \; r(s,a) + \gamma \, \mathbb{E}\big[U^\star(s_1)\big]$$

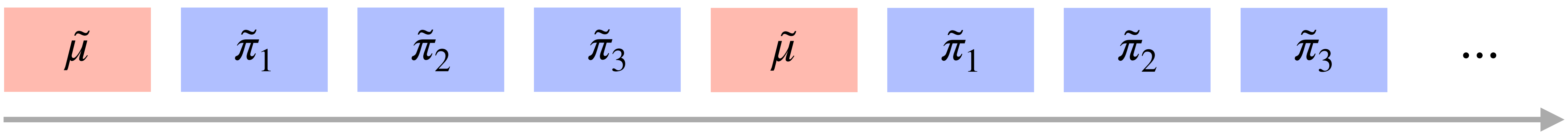
$$\bar{U}^\star(s_0) = \max_{(\mu,\boldsymbol{\pi})} \; \mathbb{E}\big[R(s_0, \mu(s_0), \boldsymbol{\pi}) + \gamma^T \, \bar{U}^\star(s_T)\big]$$

How can we take advantage of this?

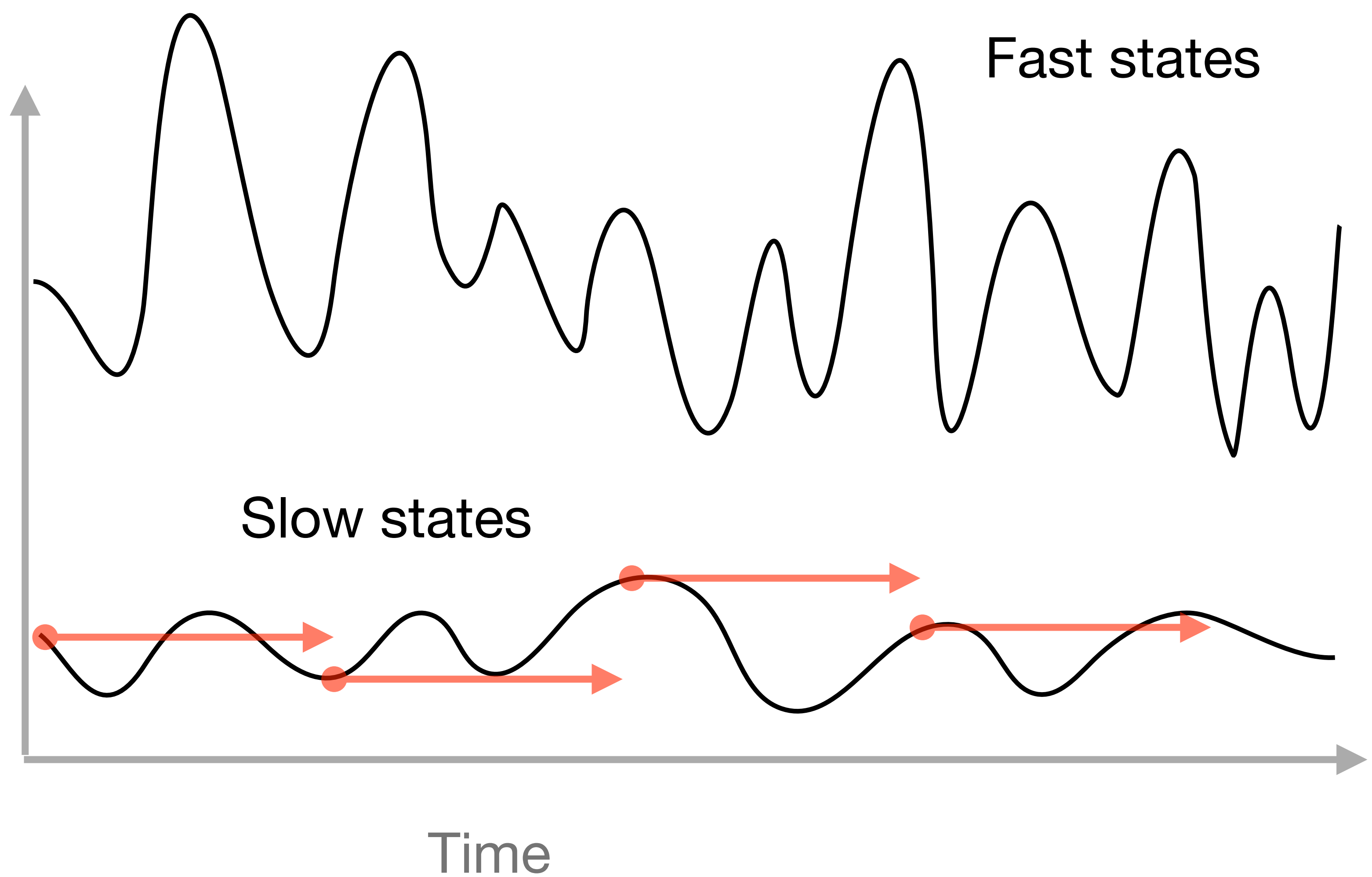
Proposition. The optimal values are equal: $U^\star(s) = \bar{U}^\star(s)$. Therefore, we can use the hierarchical reformulation as a basis for our approximation.

4. Frozen-state approximation and its regret

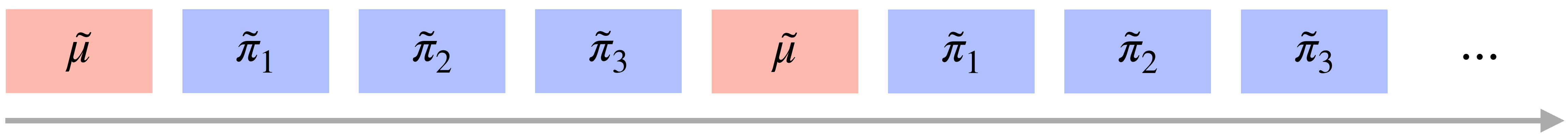
Frozen-state approximation



Main idea: Since slow states don't change much, let's ***freeze*** them for some number (T) of periods. Easier sub-problem with a smaller state space.



Frozen-state approximation



Main idea: Since slow states don't change much, let's ***freeze*** them for some number (T) of periods. Easier sub-problem with a smaller state space.

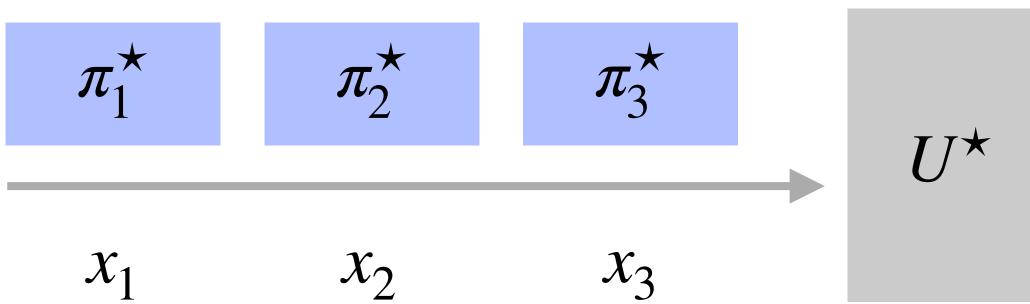
Implementation

1. At $t = 0$, take a "upper-level" action (using $\tilde{\mu}$), i.e., an action that considers the γ^T timescale
2. At $t = 1$, observe slow state and pretend it is frozen until $t = T$ and that $t = T$ is the end of the horizon
3. Solve this *easier* lower-level finite horizon problem.
4. Execute this T-period lower-level policy $(\tilde{\pi}_1, \tilde{\pi}_2, \ldots, \tilde{\pi}_{T-1})$ in the real system
5. Repeat

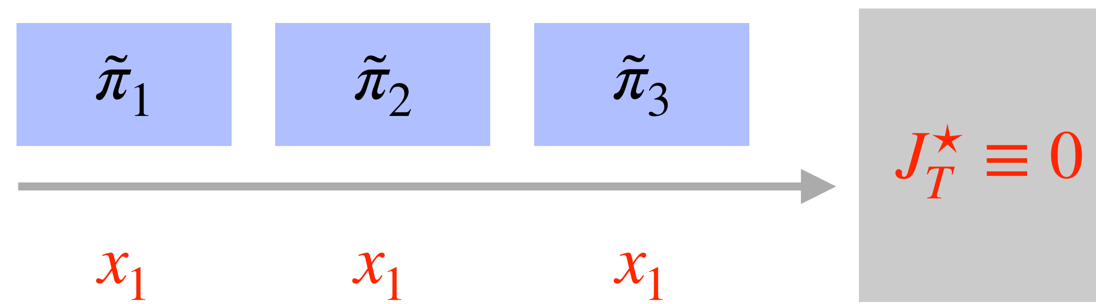
Computation

- **Pre-compute** finite-horizon lower-level policy with frozen slow states
- Re-use pre-computed lower-level policy to solve infinite-horizon upper-level problem, which **takes advantage of** γ^T

Frozen-state, lower-level problem



The true problem



(Or, we can use any VFA we would like as the terminal value.)

Frozen-state lower-level MDP

$$J_1^\star(x, y) = \max_{\tilde{\pi}} \mathbb{E}\left[\sum_{t=1}^{T-1} \gamma^{t-1} r(x_1, y_t, \tilde{\pi}_t) \,\middle|\, (x_1, y_1) = (x, y)\right]$$

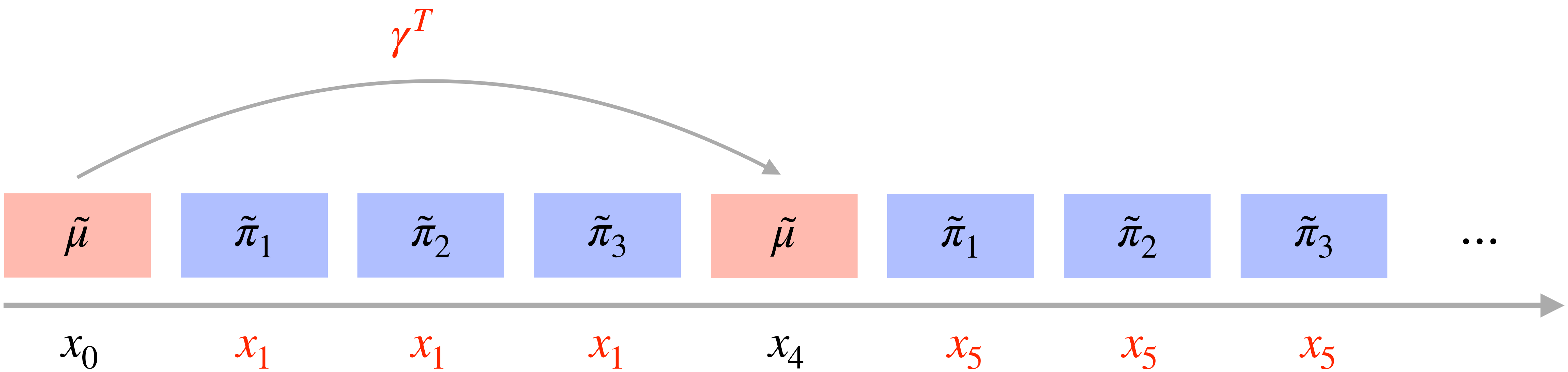
$$J_t^\star(x, y) = \max_a r(x, y, a) + \gamma\, \mathbb{E}\left[J_{t+1}^\star(x, y')\right], \quad J_T^\star \equiv 0$$

$$\tilde{\pi}_t^\star(x, y) = \text{argmax}_a\, r(x, y, a) + \gamma\, \mathbb{E}\left[J_{t+1}^\star(x, y')\right].$$

Computational benefits

- Small number of successor states (since slow state is frozen)
  - $\mathcal{O}(S^2 A) \rightarrow \mathcal{O}(XY^2 A)$
- Independent across x
- Independent from upper-level problem (replaced $U^\star$ by 0)

Frozen-state, upper-level problem



Frozen-state upper-level MDP

Let $(\tilde{\boldsymbol{\pi}}^\star, J_1^\star)$ be the optimal policy/value of the lower-level problem.

$$\tilde{R}(s_0, a, J_1^\star) = r(s_0, a) + \gamma J_1^\star\big(f(s_0, a, w)\big)$$

$$V^\star(s_0, J_1^\star, \tilde{\boldsymbol{\pi}}^\star) = \max_a \mathbb{E}\big[\tilde{R}(s_0, a, J_1^\star) + \gamma^T V^\star(s_T, J_1^\star, \tilde{\boldsymbol{\pi}}^\star)\big]$$ (transitions according to $\tilde{\boldsymbol{\pi}}^\star$)

After solving both levels, let $(\tilde{\mu}^\star, \tilde{\pi}^\star)$ be the solution of the frozen-state approximation.

In the exact reformulation, we were maximizing over policies, now it is just a single action.

Per-cycle reward approximation error

Proposition. The difference between true and approximate T-horizon rewards:

$$\Big|\underbrace{\mathbb{E}[R(s_0,a,\boldsymbol{\pi}^\star)]}_{\text{True}}-\underbrace{\mathbb{E}[\tilde{R}(s_0,a,J_1^\star)]}_{\text{Frozen}}\Big|$$

$$\leq \underbrace{\alpha d_{\mathcal{Y}}\left(L_r\sum_{i=1}^{T-2}\gamma^i\sum_{j=0}^{i-1}L_f^j\right)}_{\text{error from freezing}}+\underbrace{\gamma^{T-1}L_U\left[\alpha d_{\mathcal{Y}}\sum_{j=0}^{T-2}L_f^j+\gamma d_{\mathcal{Y}}(\alpha+2)(T-1)\right]}_{\text{end of horizon error}}$$

Main ideas.

1. $\mathbb{E}\left[R(x_0,y_0,a,\boldsymbol{\pi}^*)\right] = \mathbb{E}\left[r(x_0,y_0,a) + \gamma\left(H^{T-1}U^\star\right)(x_1,y_1) - \gamma^T U^\star(x_T,y_T)\right]$

   where $(HU)(x,y) = \max_a r(x,y,a) + \gamma\,\mathbb{E}\left[f(s,a,w))\right]$ *(true Bellman operator)*

2. $\mathbb{E}\left[\tilde{R}(x_0,y_0,a,J_1^\star)\right] = r(x_0,y_0,a) + \gamma\left(\tilde{H}^{T-1}\mathbf{0}\right)(x_1,y_1)$

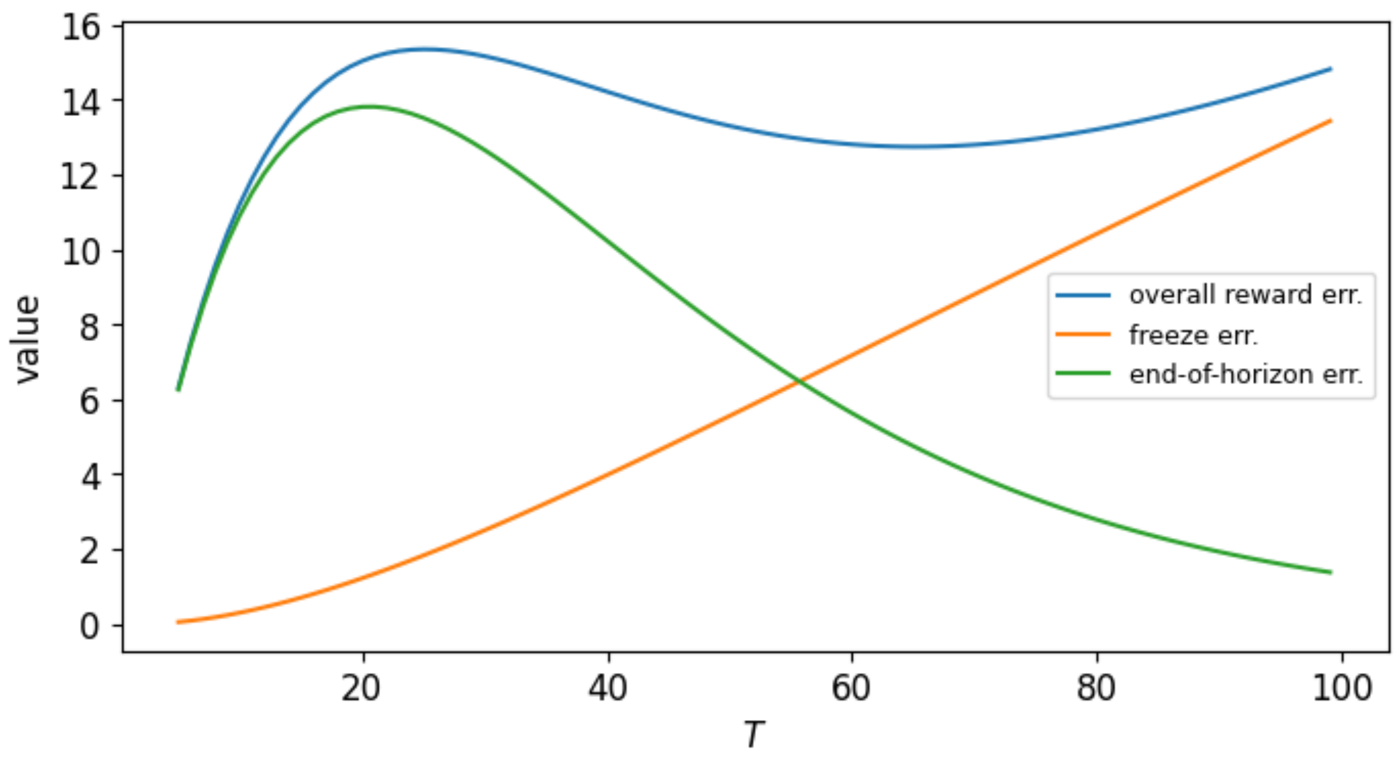
   where $\left(\tilde{H}J_{t+1}\right)(x,y) = \max_a r(x,y,a) + \gamma\,\mathbb{E}\left[J_{t+1}(x,f_{\mathcal{Y}}(x,y,a,w))\right]$ *(frozen Bellman operator)*

Per-cycle reward approximation error

Proposition. The difference between true and approximate T-horizon rewards:

$$\Big|\underbrace{\mathbb{E}[R(s_0, a, \boldsymbol{\pi}^\star)]}_{\text{True}} - \underbrace{\mathbb{E}[\tilde{R}(s_0, a, J_1^\star)]}_{\text{Frozen}}\Big|$$

$$\leq \underbrace{\alpha d_{\mathcal{Y}} \left(L_r \sum_{i=1}^{T-2} \gamma^i \sum_{j=0}^{i-1} L_f^j \right)}_{\text{error from freezing}} + \underbrace{\gamma^{T-1} L_U \left[\alpha d_{\mathcal{Y}} \sum_{j=0}^{T-2} L_f^j + \gamma d_{\mathcal{Y}} (\alpha + 2)(T-1) \right]}_{\text{end of horizon error}}$$



5. A new algorithm: *Frozen-state value iteration*

Standard value iteration on the base model

Recall: Given an MDP and Bellman operator H, where $(HU)(s) = \max_a r(s, a) \, + \, \gamma \mathbb{E} U(f(s, a, w))$, the *value iteration* algorithm is $U^k = H^k U^0$

- Convergence to optimal value function: $\lim_{t \to \infty} H^t U = U^\star$ for any initial estimate V
- $\| U^{\nu_k} - U^* \|_\infty \leq \dfrac{2 r_{\max} \gamma^{k+1}}{(1-\gamma)^2}$, where $\nu^k(s) = \text{argmax}_a \; r(s,a) + \gamma \, \mathbb{E}\big[U^k(f(s,a,w))\big]$

Depends on

- $\| U^k - U^\star \|_\infty \leq \gamma^k \| U^0 - U^\star \|_\infty$
- $\| U^0 - U^\star \|_\infty \leq \dfrac{r_{\max}}{1-\gamma}$
- $\| U^{\nu^k} - U^\star \|_\infty \leq \dfrac{2 \| U^k - U^\star \|_\infty}{1-\gamma}$

Algorithm 1: Exact VI for the Base Model

Input: Initial values U_0, number of iterations k.

Output: Approximation to the optimal policy ν^k.

```
for i = 1, 2, ..., k do
    for s in the state space S do
        U^i(s) = max_a r(s, a) + γ E[U^{i-1}(f(s, a, w))].
    end
end
for s in the state space S do
    ν^k(s) = arg max_a r(s, a) + γ E[U^k(f(s, a, w))].
end
```

Frozen-state value iteration (FSVI)

Algorithm 2: Frozen-State Value Iteration (FSVI)

Input: Initial values $J_T^* \equiv 0$ and V^0, number of iterations k.

Output: Approximation of the T-periodic frozen-state policy $(\tilde{\mu}^k, \tilde{\boldsymbol{\pi}}^*)$ and J_1^*.

```
for t = T − 1, T − 2, ..., 1 do
    for each slow state x ∈ 𝒳 do
        for each fast state y ∈ 𝒴 do
            J*_t(x, y) = max_a r(x, y, a) + γ E[J*_{t+1}(x, f_𝒴(x, y, a, w))].
            π̃*_t(x, y) = argmax_a r(x, y, a) + γ E[J*_{t+1}(x, f_𝒴(x, y, a, w))].
        end
    end
end
for i = 1, 2, ..., k do
    for s_0 = (x_0, y_0) in the state space 𝒳 × 𝒴 do
        V^i(x_0, y_0, J*_1, π̃*) = max_a E[R̃(s_0, a, J*_1) + γ^T V^{i−1}(x_T, y_T, J*_1, π̃*)].
    end
end
for s_0 = (x_0, y_0) in the state space 𝒳 × 𝒴 do
    μ̃^k(x_0, y_0) = argmax_a E[R̃(s_0, a, J*_1) + γ^T V^k(x_T, y_T, J*_1, π̃*)].
end
```

Line 4: $J_t^*(x,y) = \max_a r(x,y,a) + \gamma\,\mathbb{E}\big[J_{t+1}^*(x, f_{\mathcal{Y}}(x,y,a,w))\big]$.

Line 5: $\tilde{\pi}_t^*(x,y) = \arg\max_a r(x,y,a) + \gamma\,\mathbb{E}\big[J_{t+1}^*(x, f_{\mathcal{Y}}(x,y,a,w))\big]$.

Line 11: $V^i(x_0, y_0, J_1^*, \tilde{\boldsymbol{\pi}}^*) = \max_a \mathbb{E}\big[\tilde{R}(s_0, a, J_1^*) + \gamma^T V^{i-1}(x_T, y_T, J_1^*, \tilde{\boldsymbol{\pi}}^*)\big]$.

Line 15: $\tilde{\mu}^k(x_0, y_0) = \arg\max_a \mathbb{E}\big[\tilde{R}(s_0, a, J_1^*) + \gamma^T V^k(x_T, y_T, J_1^*, \tilde{\boldsymbol{\pi}}^*)\big]$.

Note: Freezing the state only happens "within" the algorithm to more efficiently compute $J_1^\star$

Solving the lower level incurs a one time fixed cost

Pre-compute lower-level problem, a finite-horizon DP:

- To solve lower-level DP: $\mathcal{O}(XY^2AT)$
- To compute multi-step transition: $\mathcal{O}(S^2T)$

Upper-level problem (infinite-horizon VI on slow-timescale MDP with γ^T discounting):

- Per upper-level VI iteration: $\mathcal{O}(S^2A)$

$\mathcal{O}(S^2A)$ per iteration is the same as Base VI, but now the discount factor is γ^T instead of γ!

Regret of a periodic policy $(\mu, \boldsymbol{\pi})$

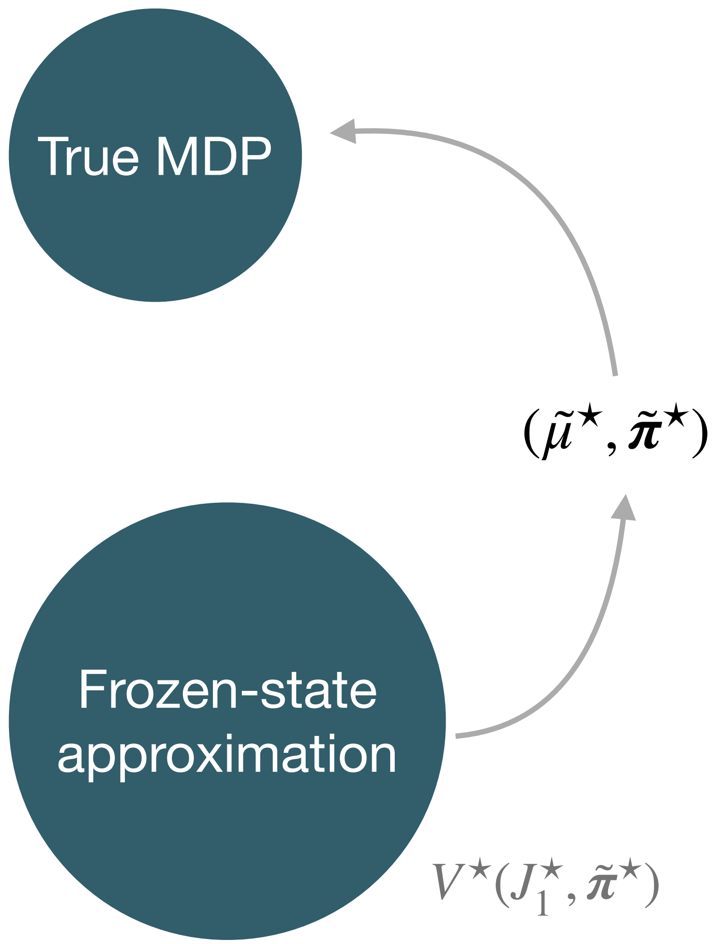
Definition. Suppose the optimal policy is $\nu^\star$. The regret is

$$\mathscr{R}(s, \mu, \boldsymbol{\pi}) = U^{\nu^\star}(s) - \bar{U}^{\mu}(s, \boldsymbol{\pi}) = \bar{U}^\star(s) - \bar{U}^{\mu}(s, \boldsymbol{\pi}) \quad \text{and} \quad \mathscr{R}(\mu, \boldsymbol{\pi}) = \max_s \mathscr{R}(s, \mu, \boldsymbol{\pi}),$$

where we have used the equivalence between the base and hierarchical formulations.

Remarks:

- We always measure regret with respect to the *true* MDP.
    - Although $(\mu, \boldsymbol{\pi})$ is computed ***with the help of frozen states***, it is evaluated in the original MDP with true dynamics.
- Consider $\mathscr{R}(\tilde{\mu}^\star, \tilde{\boldsymbol{\pi}}^\star)$, notice that $V^\star(J_1^\star, \tilde{\boldsymbol{\pi}}^\star)$ does not directly enter the regret definition.
    - It is the optimal value of the approximation, but doesn't reflect the performance of $(\tilde{\mu}^\star, \tilde{\boldsymbol{\pi}}^\star)$ in the true model.



Main idea behind regret analysis

Lemma (Approximation to FSVI).

- Suppose we *approximately* solve the lower-level problem and obtain $\boldsymbol{\pi}, J_1$, instead of the optimal solutions $\boldsymbol{\pi}^\star, U^\star$.
- Suppose we approximately solve the upper-level problem and obtain V instead of $V^\star(J_1, \boldsymbol{\pi})$.
- Let μ be greedy with respect to both J_1 and V:
  - $\mu(s_0) = \text{argmax}_{a \in \mathscr{A}} \mathbb{E}\left[\tilde{R}(s_0, a, J_1) + \gamma^T V(s_T(a, \boldsymbol{\pi}))\right]$.
- Then,

$$\mathscr{R}(\mu, \boldsymbol{\pi}) \leq \left(\frac{2\gamma^T}{(1-\gamma^T)^2} + \frac{2}{1-\gamma^T}\right)\epsilon_r(\boldsymbol{\pi}^\star, J_1) + \left(\frac{2\gamma^{2T}}{(1-\gamma^T)^2} + \frac{2\gamma^T}{1-\gamma^T}\right) L_U\, d(\alpha, d_{\mathscr{Y}}, T) + \frac{2\gamma^T}{1-\gamma^T}\|V^\star(J_1, \boldsymbol{\pi}) - V\|_\infty.$$

Reward error

End of horizon error

V-approximation error

Regret of FSVI

Theorem. The regret of FSVI after k upper-level iterations is:

$$\mathscr{R}(\mu, \boldsymbol{\pi}) \leq \left(\frac{2\gamma^T}{(1-\gamma^T)^2} + \frac{2}{1-\gamma^T}\right)\epsilon_r(\boldsymbol{\pi}^\star, J_1) + \left(\frac{2\gamma^{2T}}{(1-\gamma^T)^2} + \frac{2\gamma^T}{1-\gamma^T}\right)L_U\, d(\alpha, d_{\mathscr{Y}}, T) + \frac{2r_{\max}\gamma^{(k+1)T}}{(1-\gamma)(1-\gamma^T)},$$

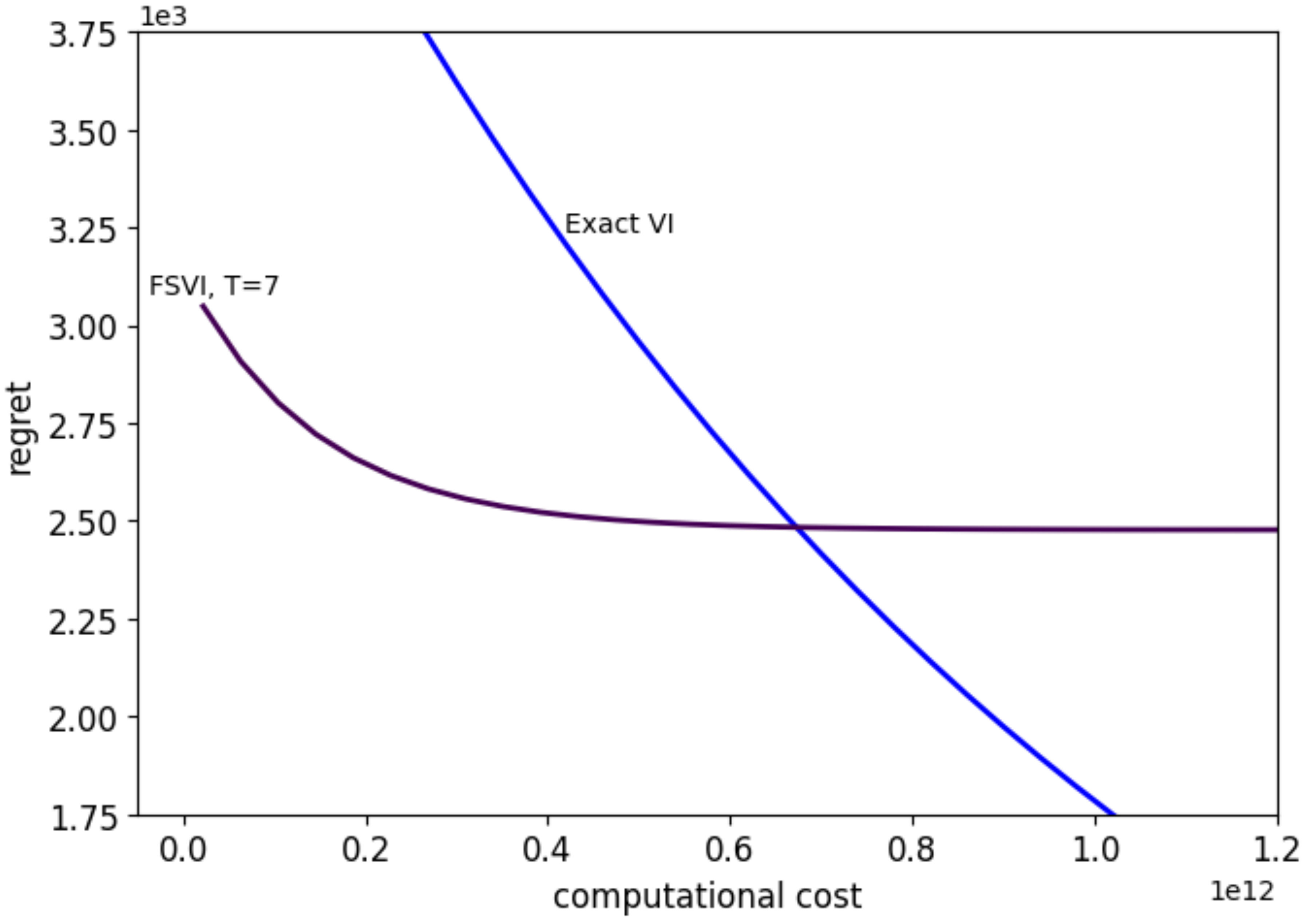
which replaces the V-approximation error term with the VI error.

Main question to answer using the regret analysis

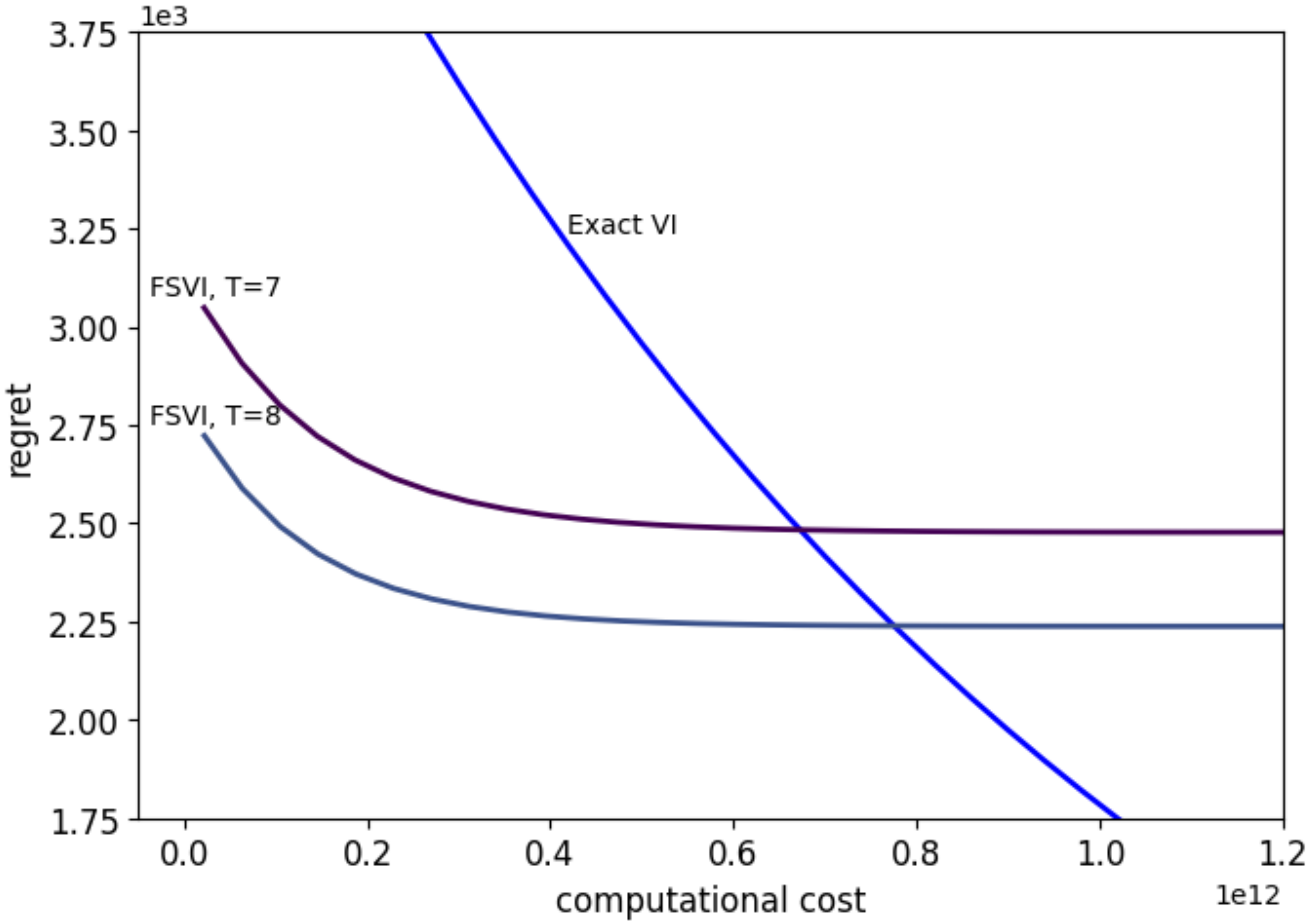
Given some computational budget, should we use the FSVI algorithm at all?

If so, how should we choose T (the number of periods to freeze) and k (number of upper-level steps iterations)?

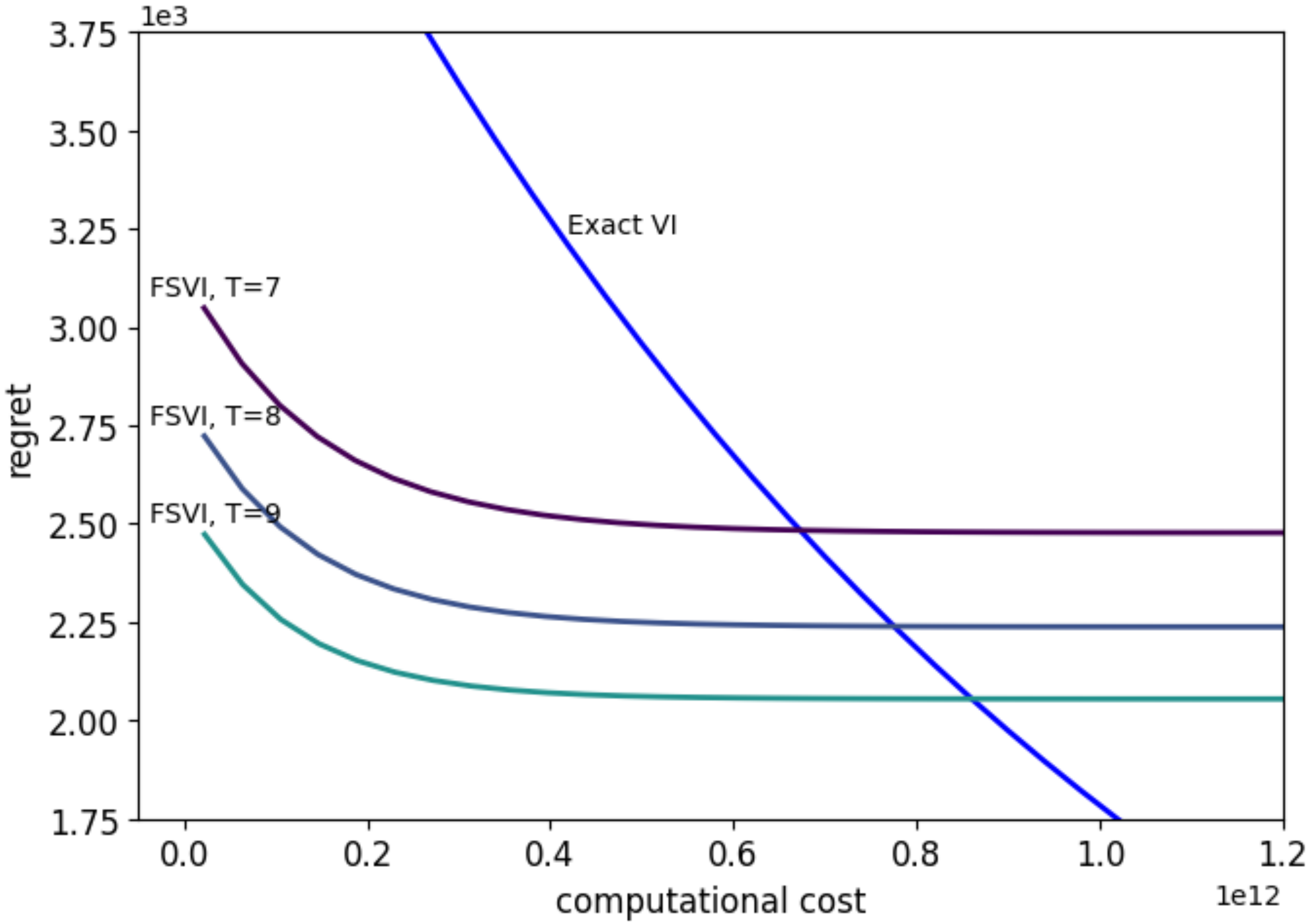
How can we use the bound to choose T?



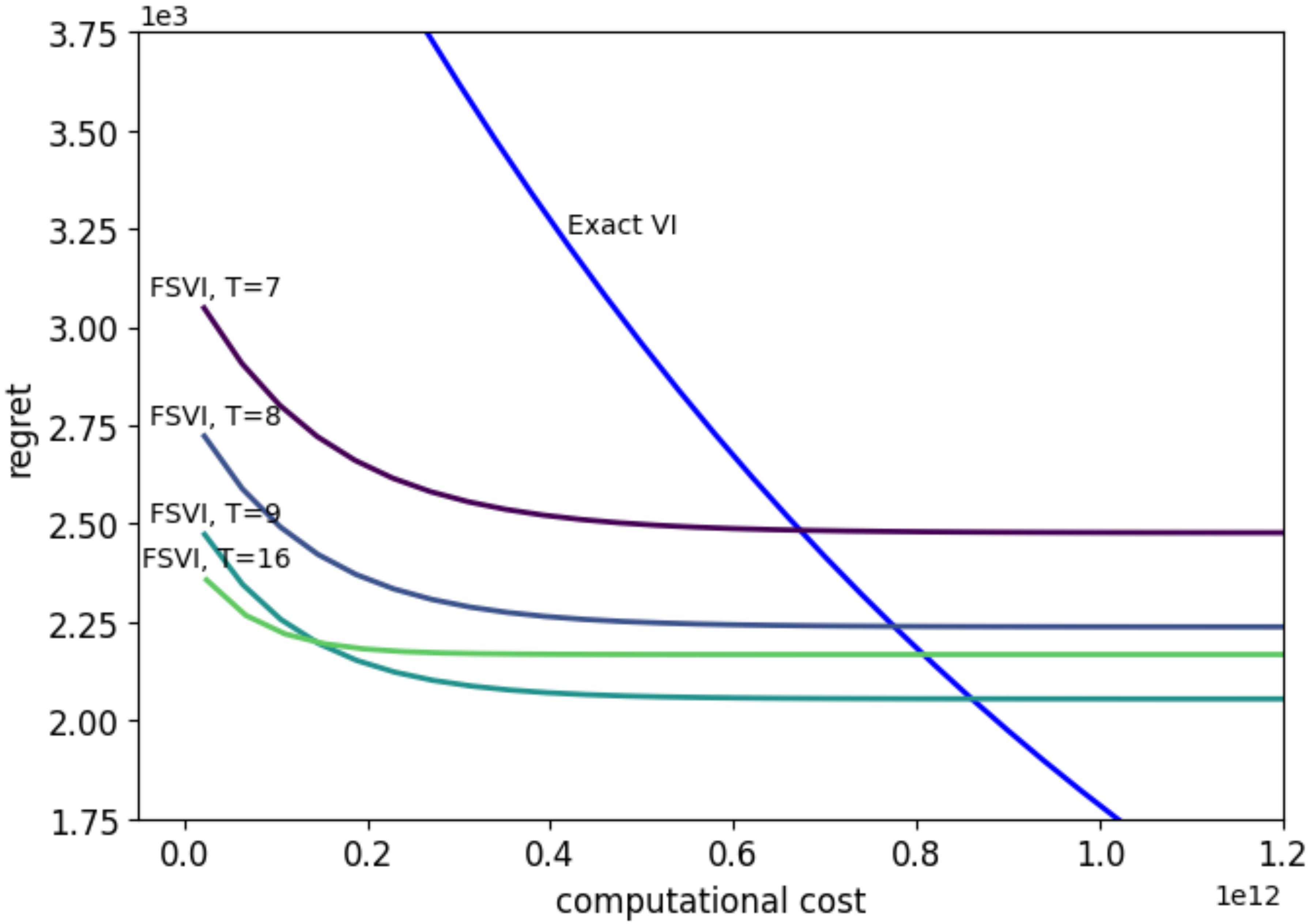
How can we use the bound to choose T?



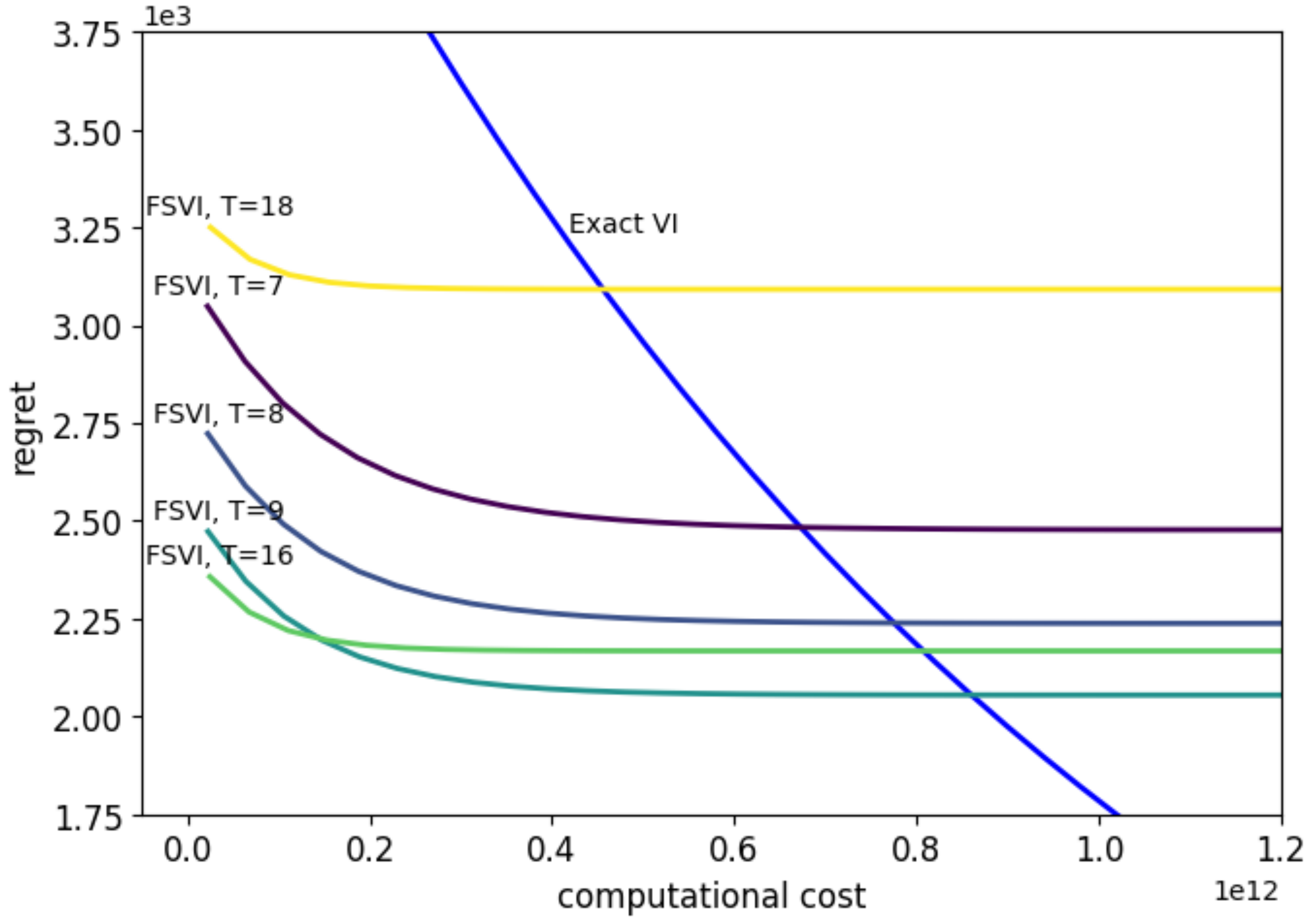
How can we use the bound to choose T?



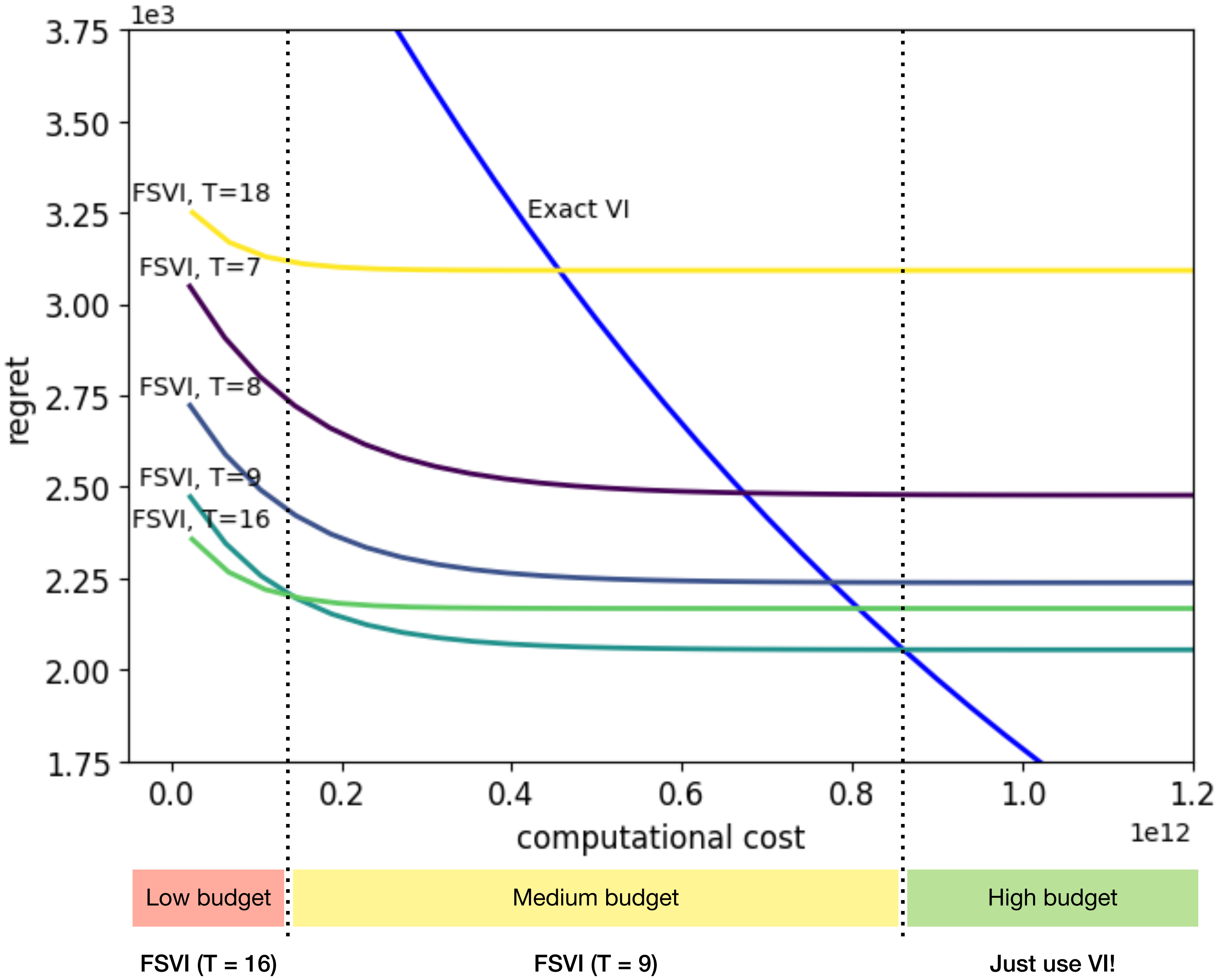
How can we use the bound to choose T?



How can we use the bound to choose T?



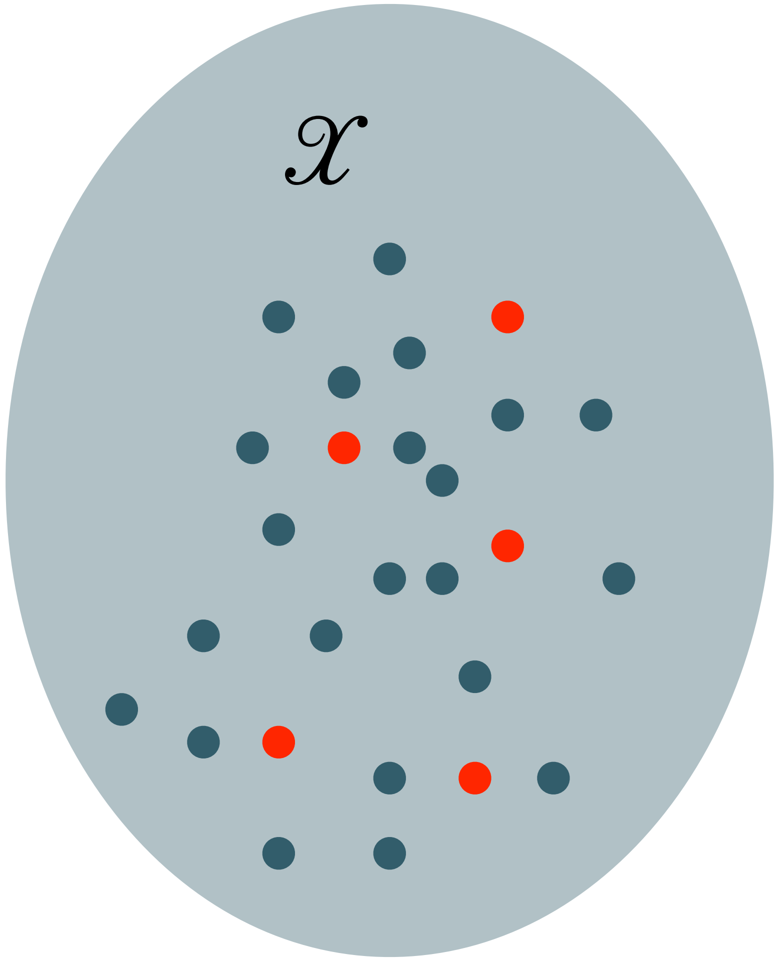
How can we use the bound to choose T?



6. Extensions

Extension: Nominal state version of FSVI

- In FSVI, we have to solve the lower-level MDP for each x
- We can further do approximations by solving the lower-level MDP for a few nominal states only
  - $\mathcal{O}(S^2A) \to \mathcal{O}(XY^2A) \to \mathcal{O}(X_{\text{nom}}Y^2A)$
- Later, extrapolate to nearby nominal states
- Theoretical results can be adapted given an additional assumption on the MDP rewards



Extension: Scaling to larger state spaces using feature-based approximate value iteration

Architecture:

- Consider M pre-selected states $\tilde{\mathcal{S}} = \{s_1, s_2, \ldots, s_M\}$.
- Consider an M-dimensional feature vector $\boldsymbol{\phi}(s)$, where $\boldsymbol{\phi}(s_m)$ are linearly independent.
- Assume there exists $\gamma' \in [\gamma, 1)$ s.t. for any s, there exists $\theta_m(s)$, where
  - $\sum_m |\theta_m(s)| \leq 1$ and $\boldsymbol{\phi}(s) = \frac{\gamma'}{\gamma} \sum_{m=1}^{M} \theta_m(s)\, \boldsymbol{\phi}(s_m)$.
- Lower level: $\hat{J}(s, \boldsymbol{\omega}_t) = \boldsymbol{\phi}^\intercal(s)\boldsymbol{\omega}_t$.
- Upper level: $\hat{V}(s, \boldsymbol{\beta}^k) = \boldsymbol{\phi}^\intercal(s)\boldsymbol{\beta}^k$.
- Update procedure:
  1. Compute Bellman update at pre-selected states only: $y(s_m)$.
  2. Compute next parameter vector ($\boldsymbol{\omega}_{t-1}$ or $\boldsymbol{\beta}^{k+1}$) such that the updated value function evaluated at the pre-selected states is equal: e.g., $\hat{J}(s_m, \boldsymbol{\omega}_{t-1}) = y(s_m)$.

Limited "expansion" after going to parameter space and back:

$$\|(\Phi\Phi^\dagger)(J) - (\Phi\Phi^\dagger)(J')\|_\infty \leq \kappa\, \|J - J'\|_\infty \quad (\kappa = \gamma'/\gamma)$$

J. N. Tsitsiklis and B. Van Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 1996.

Extension: Scaling to larger state spaces using feature-based approximate value iteration

Algorithm 4: Frozen-State Approximate Value Iteration (FSAVI)

Input: $\tilde{\mathcal{S}} = \{s_1, s_2, \ldots, s_M\}$, $\boldsymbol{\phi}$, initial weights $\boldsymbol{\omega}_T = \boldsymbol{\beta}_0 = \mathbf{0}$, number of iterations k.

Output: Approximation of the T-periodic frozen-state policy $\left(\hat{\mu}_{(\boldsymbol{\beta}^k, \boldsymbol{\omega}^*)}, \hat{\boldsymbol{\pi}}_{\boldsymbol{\omega}^*}\right)$ and $\hat{J}_1(\boldsymbol{\omega}^*)$

for $t = T-1, T-2, \ldots, 1$ **do**
    for *each pre-selected state* $s = (x, y) \in \tilde{\mathcal{S}}$ **do**
        $J_t(x, y) = \max_a r(x, y, a) + \gamma\, \mathbb{E}\big[\hat{J}_{t+1}(x, f_{\mathcal{Y}}(x, y, a, w), \boldsymbol{\omega}_{t+1})\big]$.
    end
    Set remaining entries of J_t to zero. Update parameter vector: $\boldsymbol{\omega}_t^* = \Phi^\dagger J_t$.
end
Let $\hat{\boldsymbol{\pi}}_{\boldsymbol{\omega}^*}$ be greedy with respect to $\hat{J}_t(\boldsymbol{\omega}_t^*) = \Phi \boldsymbol{\omega}_t^*$, similar to (23).
for $i = 1, 2, \ldots, k$ **do**
    for *each pre-selected state* $s_0 \in \tilde{\mathcal{S}}$ **do**
        $V^i(s_0) = \max_a \mathbb{E}\big[\tilde{R}(s, a, \hat{J}_1(\boldsymbol{\omega}_1^*)) + \gamma^T\, \hat{V}(s_T(a, \tilde{\boldsymbol{\pi}}_{\text{avi}}), \boldsymbol{\beta}_{i-1})\big]$.
        Set remaining entries of V^i to zero. Update parameter vector: $\boldsymbol{\beta}_i = \Phi^\dagger V^i$.
    end
end
for s_0 *in the state space* $\mathcal{S}$ **do**
    $\hat{\mu}_{(\boldsymbol{\beta}^k, \boldsymbol{\omega}^*)}(s_0) = \arg\max_a \mathbb{E}\big[\tilde{R}(s_0, a, \hat{J}_1(\boldsymbol{\omega}_1^*)) + \gamma^T \hat{V}(s_T(a, \tilde{\boldsymbol{\pi}}_{\boldsymbol{\omega}^*}), \boldsymbol{\beta}_k)\big]$.
end

Regret of FSAVI

Theorem. The regret of FSAVI after k upper-level iterations is:

$$\mathscr{R}(\mu,\boldsymbol{\pi}) \le \left(\frac{2\gamma^T}{(1-\gamma^T)^2}+\frac{2}{1-\gamma^T}\right)\epsilon_r(\boldsymbol{\pi}^\star,\hat{J}_1(\boldsymbol{\omega}_1^\star))$$

$$+\left(\frac{2\gamma^{2T}}{(1-\gamma^T)^2}+\frac{2\gamma^T}{1-\gamma^T}\right)L_U\, d(\alpha, d_{\mathscr{Y}}, T)+\underbrace{\left(\frac{1+\kappa}{1-\kappa\gamma^T}\right)\varepsilon_{\text{up}}}_{\|V^\star_{\boldsymbol{\omega}^\star}-\hat{V}(\boldsymbol{\beta}^\star)\|_\infty}+\underbrace{(\kappa\gamma^T)^k\left(\frac{\kappa^2-\kappa^2(\kappa\gamma)^{T+1}}{(1-\kappa\gamma^T)(1-\kappa\gamma)}\right)r_{\max}}_{\|\hat{V}(\boldsymbol{\beta}^*)-\hat{V}(\boldsymbol{\beta}_k)\|_\infty},$$

which $\epsilon_r(\boldsymbol{\pi}^\star,\hat{J}_1(\boldsymbol{\omega}_1^\star)) = \epsilon_r(\boldsymbol{\pi}^\star, J_1^\star)+\underbrace{\left(\frac{1+\kappa}{1-\kappa\gamma}-\frac{(\kappa\gamma)^T(1+\gamma)}{\gamma-\kappa\gamma^2}\right)\varepsilon_{\text{low}}}_{\|J_t^\star-\hat{J}_t(\boldsymbol{\omega}_t^\star)\|_\infty}$.

Upper-level feature approximation error

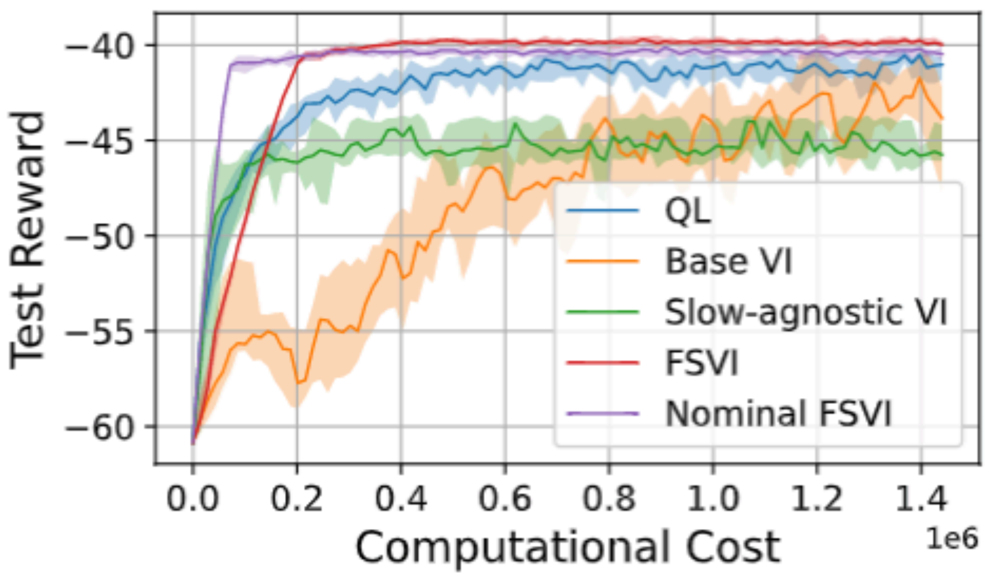
Lower-level feature approximation error

7. Numerical results

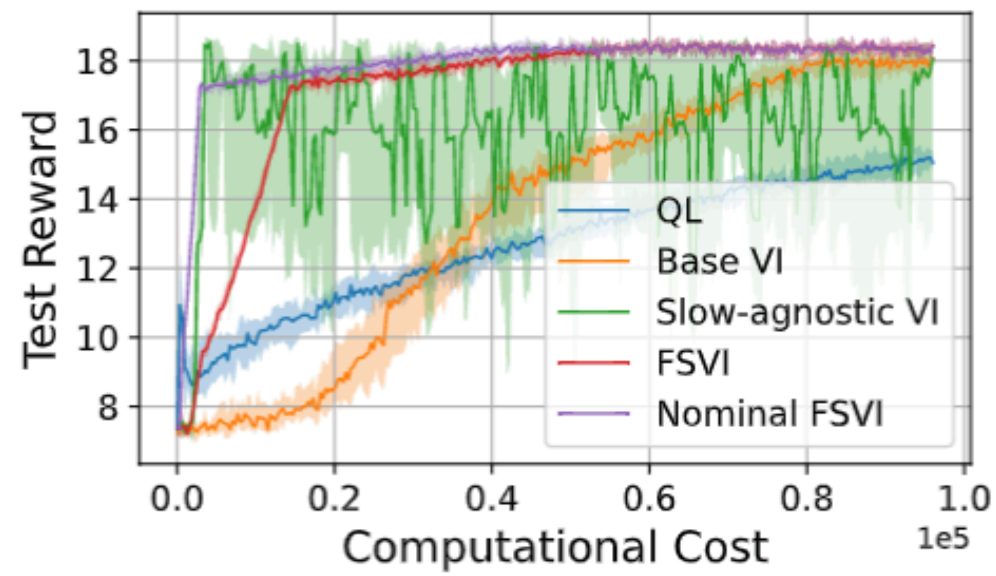
Baseline algorithms

- VI / AVI
- Slow-agnostic VI / AVI
  - Average over slow states during learning
  - Upon implementation, ignore slow state
- Q-learning (QL)
- Deep Q-networks (DQN)
- **Ours: FSVI / Nominal FSVI**
- **Ours: FSAVI / Nominal FSAVI**

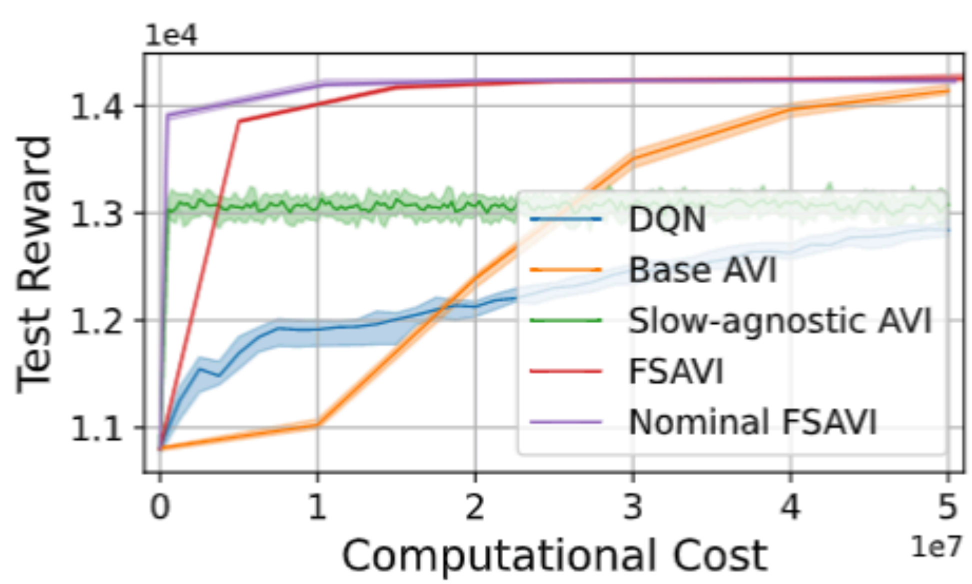
Overall performance comparison



(a) Multi-class service allocation



(b) Restless two-armed bandit



(c) Energy demand response

Questions

Please feel free to email me at danielrjiang@gmail.com for additional comments and discussion.