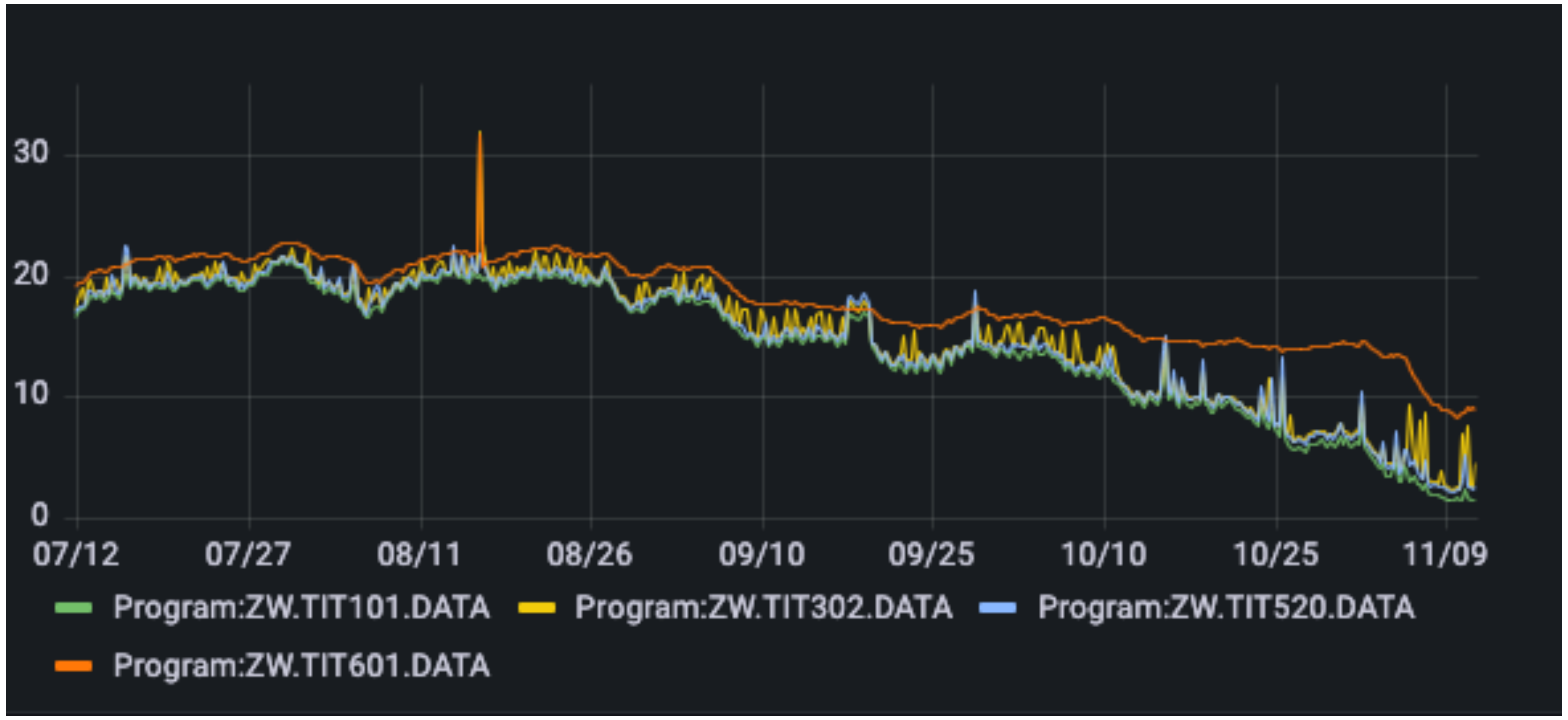
An Approach for Non-Stationarity in Lifelong Learning

Paper Review and General Discussion

Jordan Coblin, Nov 15, 2022

Water Treatment

- Sensor data (**state**), chemical dosing (**action**), water quality (**reward**)
- Continual/lifelong learning
- Non-stationarity



Continual Learning Approaches

- Elastic weight consolidation
- Replay/rehearsal
- Leveraging shared structure
- …
- Context detection -> LILAC falls under this approach

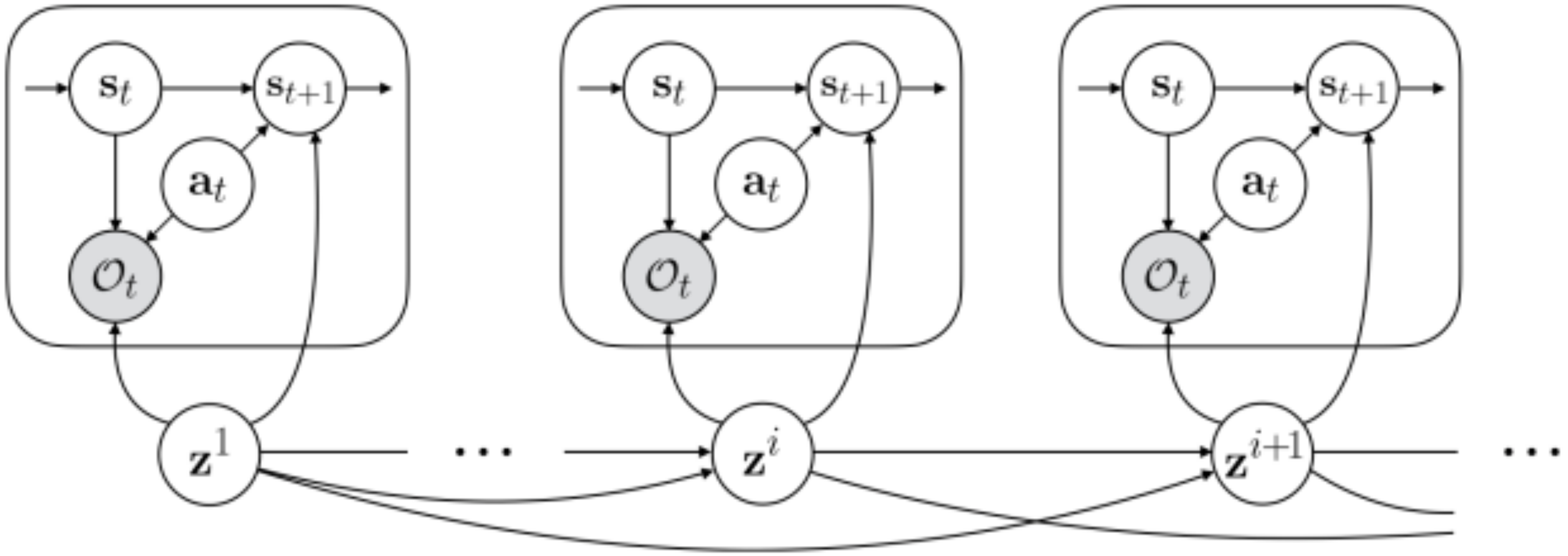
Paper Overview

Lifelong Latent Actor-Critic (LILAC)

Annie Xie, James Harrison, Chelsea Finn, Deep Reinforcement Learning amidst Continual Structured Non-Stationarity, http://proceedings.mlr.press/v139/xie21c/xie21c.pdf, 2020.

- MDP model for non-stationarity
- RL as inference
- Combining into PGM

Dynamic Parameter MDP



Dynamic Parameter MDP

- Episodic, new MDP presented each episode
- Unobserved task parameters $z \in Z$ define dynamics $p_s(s_{t+1} \mid s_t, a_t; z)$ and reward function $r(s_t, a_t; z)$
- z sampled from $p_z(z^{i+1} \mid z^{1:i})$

Dynamic Parameter MDP

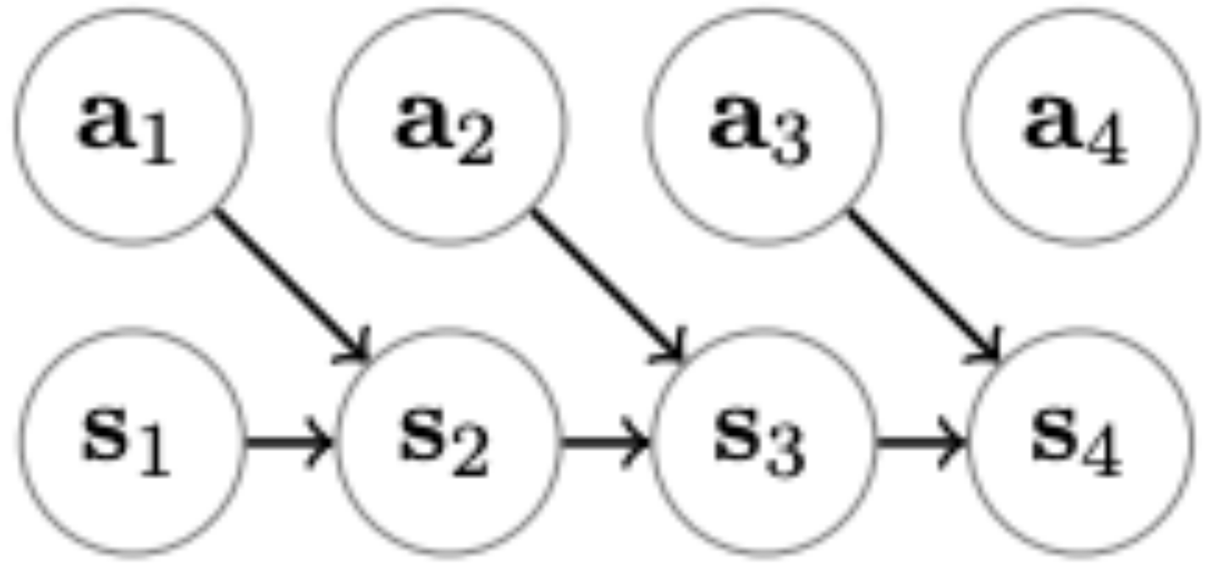
- POMDP
  - + Hidden params can handle non-stationary
  - - Too general, difficult to compute
- Bayes Adaptive MDP, Hidden Parameter MDP
  - + Hidden parameters underlying MDP
  - - Parameters not modelled sequentially

RL as Inference

- Use probability theory + probabilistic inference to model the RL problem.
- **Why?** Leverage tools from PGMs and approximate inference.
- **Goal:** formulate PGM s.t. more probable trajectories correspond to better policies.

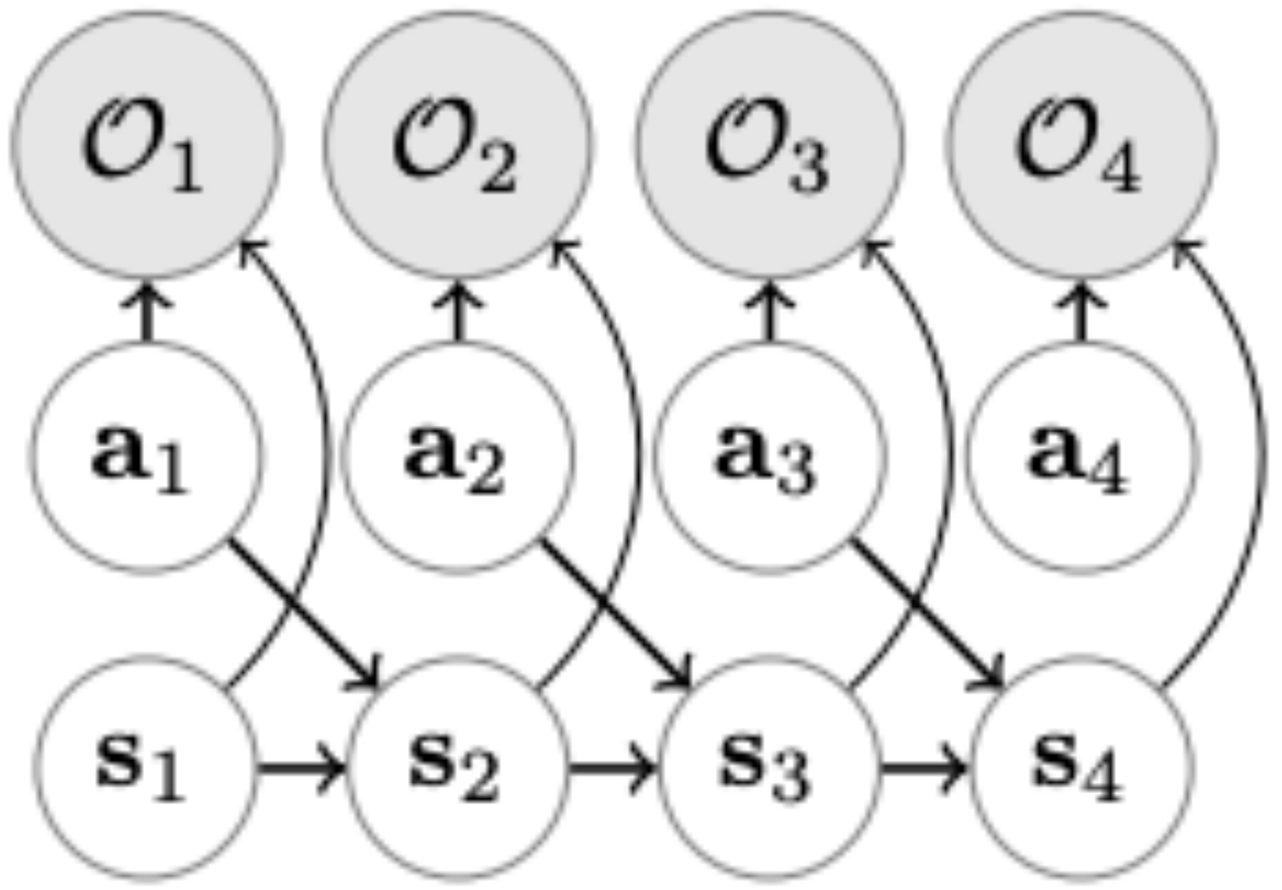
RL as Inference

- Basic PGM with factors $p(s_{t+1} | s_t, a_t)$
- No notion of reward



RL as Inference

- Optimality variable $\mathcal{O}_t$ with $p(\mathcal{O}_t = 1 \mid s_t, a_t) = \exp(r(s_t, a_t))$ with $r(s_t, a_t) > 0$
- Infer optimal trajectories: $p(\tau \mid \mathcal{O}_{1:T} = 1) \propto \left[p(s_1) \prod_{t=1}^{T} p(s_{t+1} \mid s_t, a_t) \right] \exp \left(\sum_{t=1}^{T} r(s_t, a_t) \right)$
- Infer optimal policy: $p(a_t \mid s_t, \mathcal{O}_{t:T} = 1) = \pi(a_t \mid s_t)$



RL as Inference

- How to do inference?

1. Compute backward message $\beta_t(s_t, a_t) = p(\mathcal{O}_{t:T} | s_t, a_t)$
2. Compute policy $p(a_t | s_t, \mathcal{O}_{1:T})$
3. Compute forward messages $\alpha_t(s_t) = p(s_t | \mathcal{O}_{1:t-1})$ -> useful for inverse RL

RL as Inference

- Exact inference issues:
  - High-dim/continuous state space
  - Transition probabilities not known
- Need to do approximate inference

Variational Inference

- **(Bayesian) Inference:** learn conditional distribution $p(z\,|\,\boldsymbol{x})$
- **Variational**: approximate posterior, optimization over functions

$$q*(z) = \arg\min_{q(z)\in\mathscr{D}} KL(q(z)\,||\,p(z\,|\,\boldsymbol{x}))$$

$$p(z\,|\,\boldsymbol{x}) = \frac{p(z,\boldsymbol{x})}{p(x)} = \frac{p(z,\boldsymbol{x})}{\int p(z,\boldsymbol{x})dz}$$ denominator = “evidence” -> often intractable

Instead, optimize the ELBO:

$$\mathrm{ELBO}(q) = \mathbb{E}[\log p(z,\boldsymbol{x})] - \mathbb{E}[\log q(z)]$$

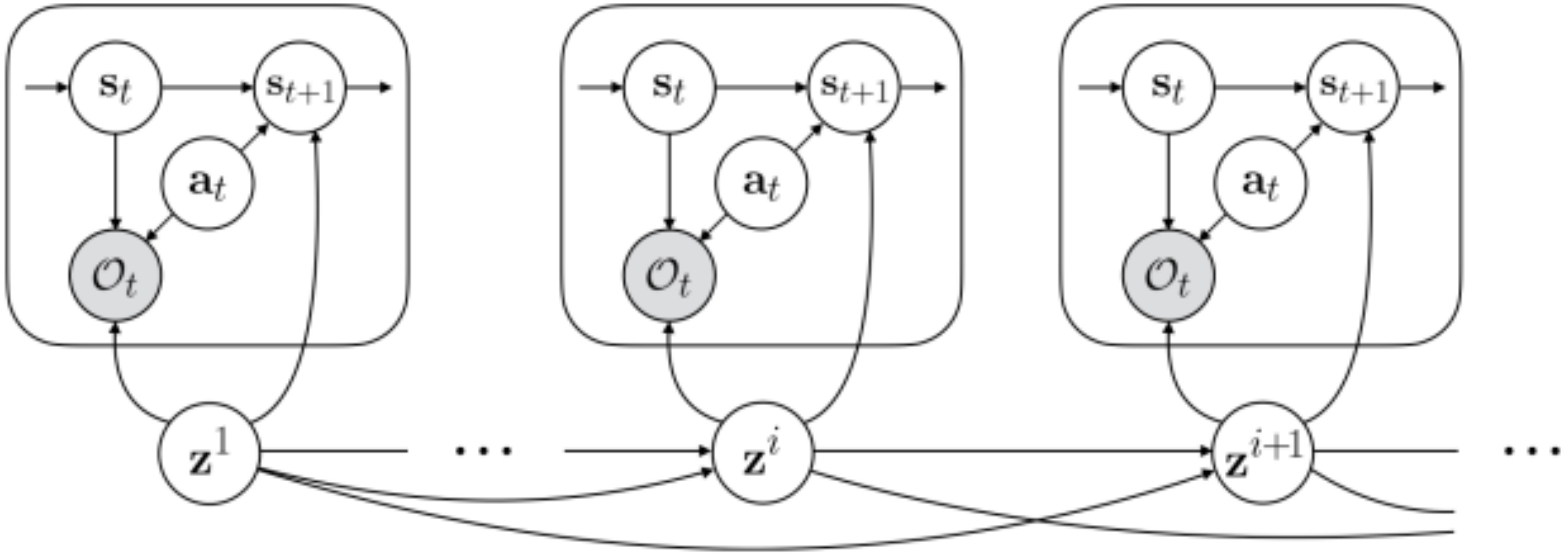
Variational Inference

- ELBO for evidence $\mathcal{O}_{1:T} = 1$ is:

$$\log p(\mathcal{O}_{1:T} = 1) \geq \mathbb{E}_{\pi} \left[\sum_{t=1}^{T} r(s_t, a_t) - \log \pi(a_t, s_t) \right]$$

- Exactly the maximum entropy objective

PGM for Non-stationarity



- Infer $p(a^i_{1:T} \mid \mathcal{O}^i_{1:T} = 1, \tau^{1:i-1})$
- Approximate posterior with $q(z^i \mid \tau^i)$

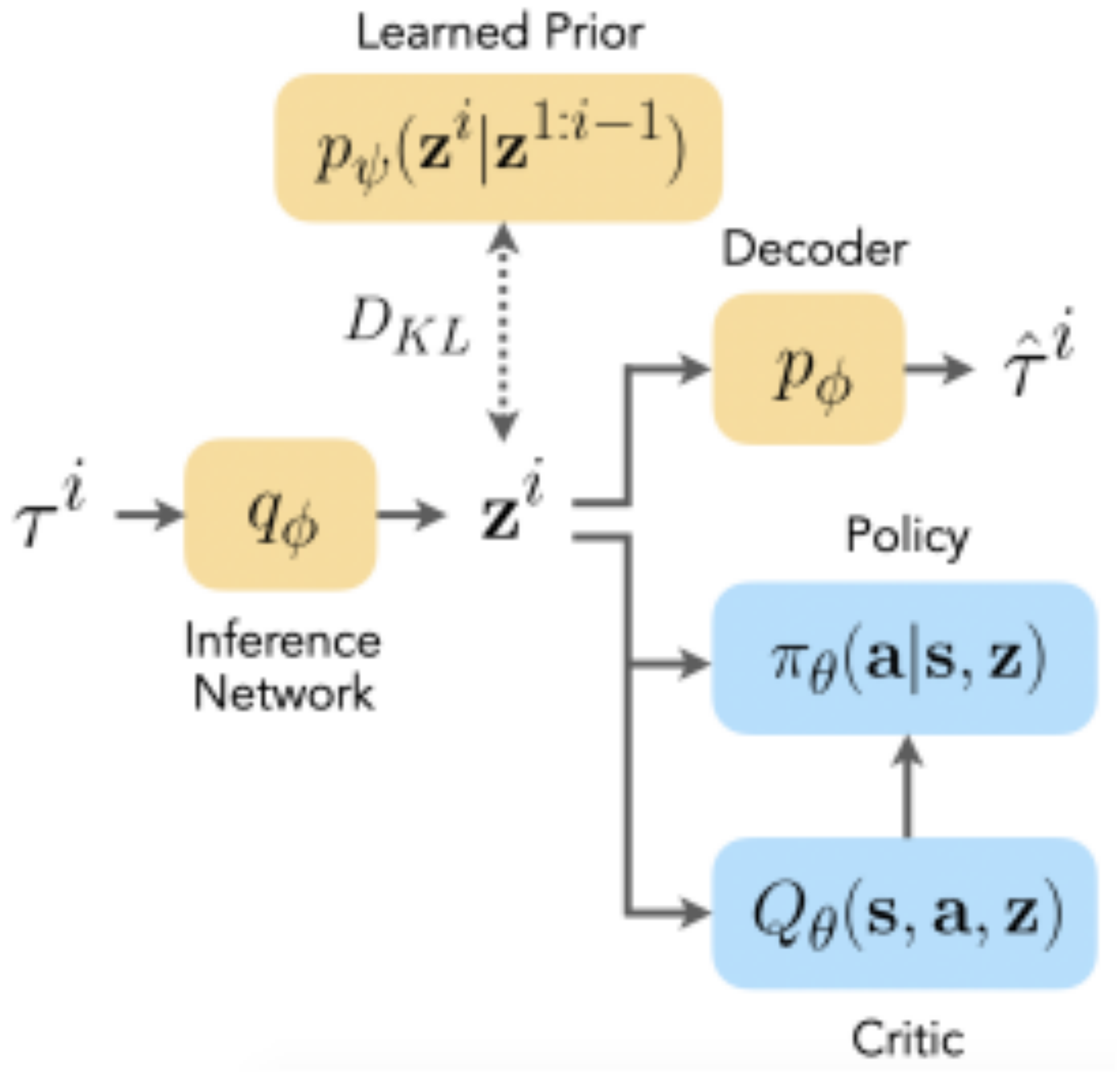
PGM for Non-stationarity

$$\log p(\tau^{1:i-1}, \mathcal{O}^{i}_{1:T} \,|\, a^{1:i-1}_{1:T}) \geq$$

$$\mathbb{E}_q \left[\sum_{i'=1}^{i} \sum_{t=1}^{T} \underbrace{\log p(s_{t+1}, r_t \,|\, s_t, a_t; z^{i'}))}_{\text{Model dynamics \& reward}} - \underbrace{\mathrm{D}_{\mathrm{KL}}(q(z^{i'} \,|\, \tau^{i'}) \,||\, p(z^{i'} \,|\, z^{i'-1})}_{\text{Model latent shifts}} \right]$$

$$+\mathbb{E}_{p(z^i|\tau^{1:i-1}),\pi(a_t|s_t,z^i)} \left[\sum_{i=1}^{T} \underbrace{r(s_t, a_t; z^i) - \log \pi(a_t \,|\, s_t, z^i)}_{\text{Entropy-regularized RL}} \right]$$

LILAC Architecture



LILAC Algorithm

Beginning of each episode:

- Sample $z^i \sim p_\psi(z^i \mid z^{1:i-1})$
- Collect trajectory τ^i from env with $\pi_\theta(a \mid s, z)$

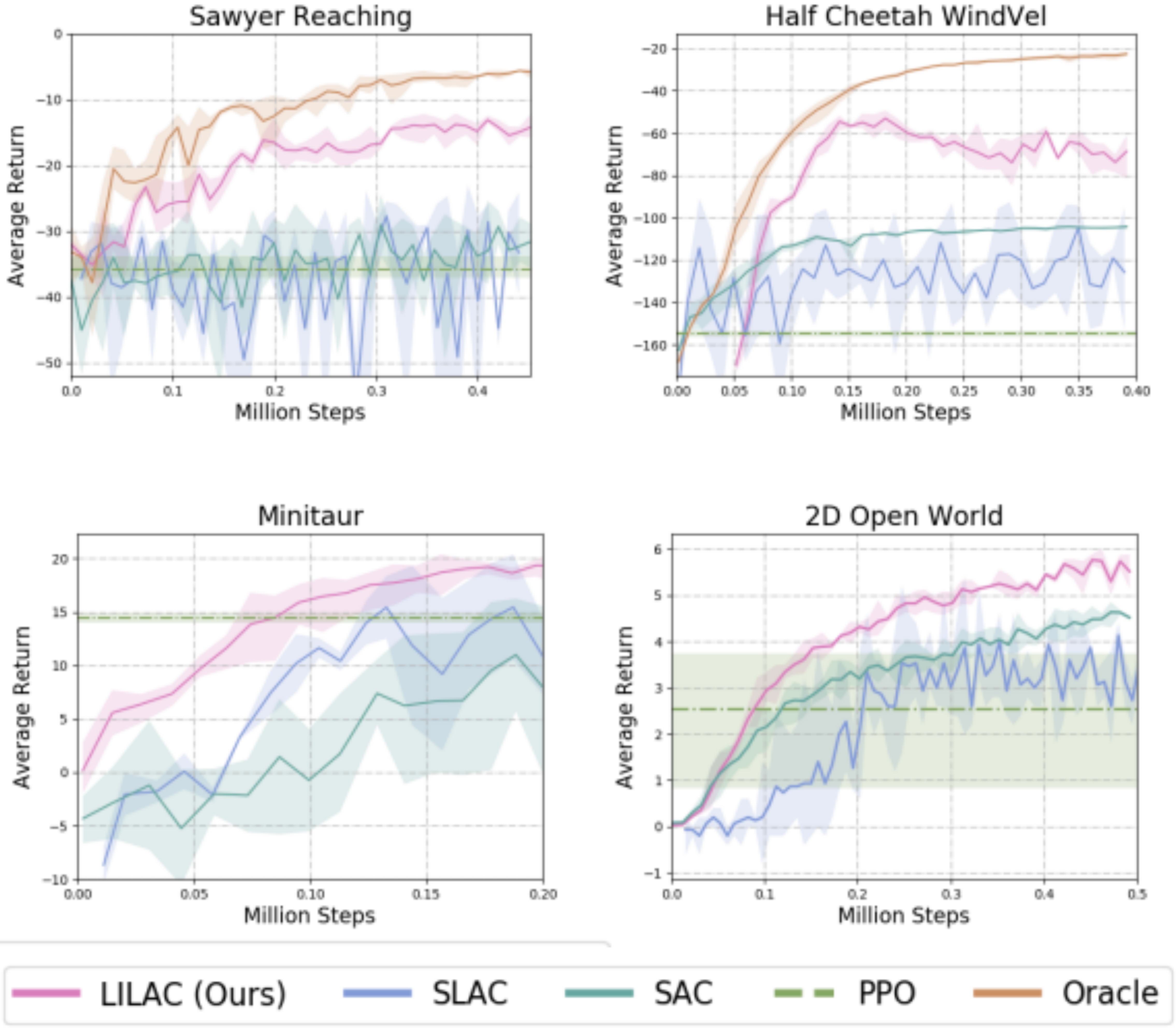
Replay updates:

- Update models by sampling inference network $z^i \sim q_\phi(z^i \mid \tau^i)$

Experiments

- Envs with varying rates of change, *intra*-episodic shifts, and extrapolating environment shifts (i.e. out-of-distribution)
- Sawyer reaching task: target position not observed, moves between eps
- Half-cheetah: change in direction + magnitude of wind forces
- etc.

Results



Sawyer Reaching
Average Return
Million Steps
Half Cheetah WindVel
Average Return
Million Steps
Minitaur
Average Return
Million Steps
2D Open World
Average Return
Million Steps
LILAC (Ours)
SLAC
SAC
PPO
Oracle

Takeaways

- Framing RL problem as inference $\leftrightarrow$ max entropy RL
- Context detection via task latent variable
- Sequential modelling of the task latent variable -> anticipate non-stationarity

Downsides/Questions

- Assumes that contexts are known and context shifts are *observable.*
- Trade-offs of framing RL as inference?
- Applicable/overkill for water treatment?

References

- Sergey Levine, Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review, https://arxiv.org/pdf/1805.00909.pdf

- Annie Xie, James Harrison, Chelsea Finn, Deep Reinforcement Learning amidst Continual Structured Non-Stationarity, http://proceedings.mlr.press/v139/xie21c/xie21c.pdf

Questions?

POMDPs

- Belief state

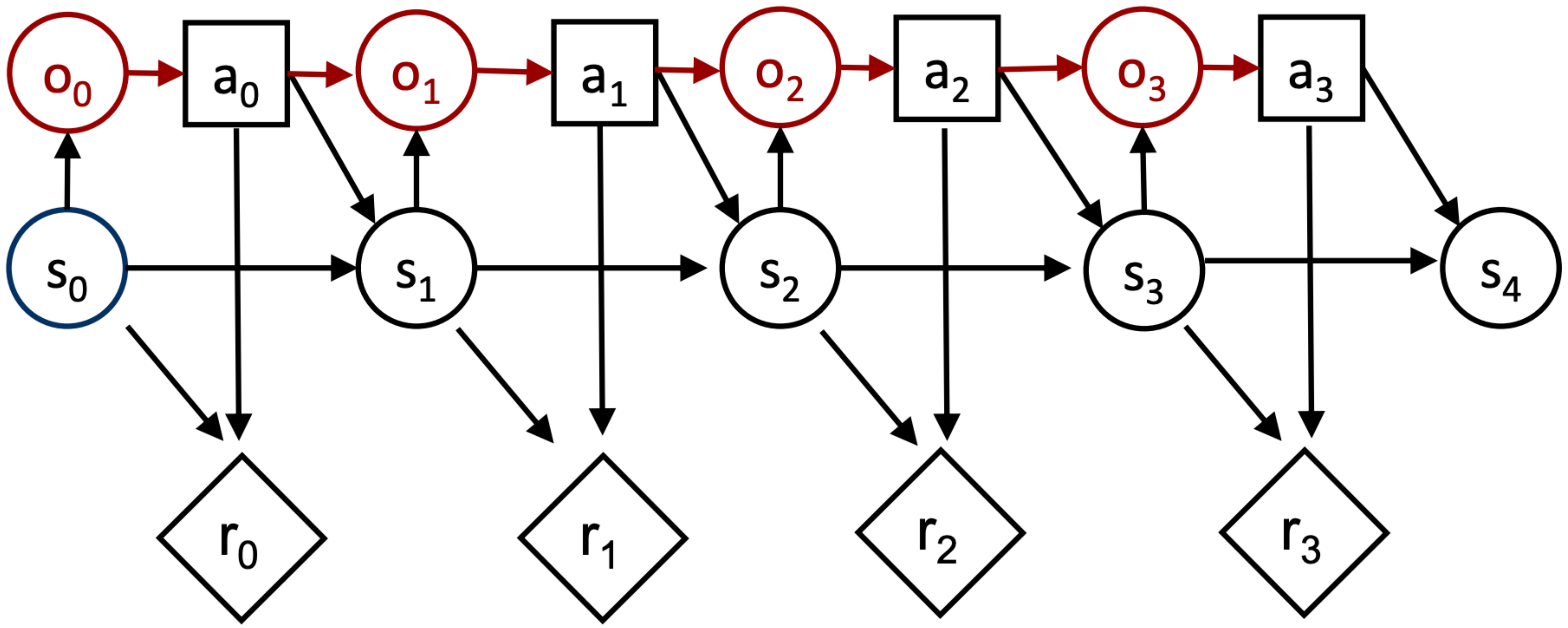
$$b_t(s_t) = Pr(s_t \mid o_{1:t}, a_{1:t})$$

- Use this belief state to perform control, i.e. compute value functions over belief space
- **Issue**: computing value function over belief space (which is continuous and high-dimensional) is often intractable
- From Sutton & Barto:

"… its assumptions and computational complexity scale poorly and we do not recommend it as an approach to artificial intelligence."

POMDPs

- Environment state is *hidden*



CS885 Spring 2018, Pascal Poupart, https://cs.uwaterloo.ca/~ppoupart/teaching/cs885-spring20/slides/cs885-lecture11b.pdf

LILAC Algorithm

Algorithm 1 Lifelong Latent Actor-Critic (LILAC)

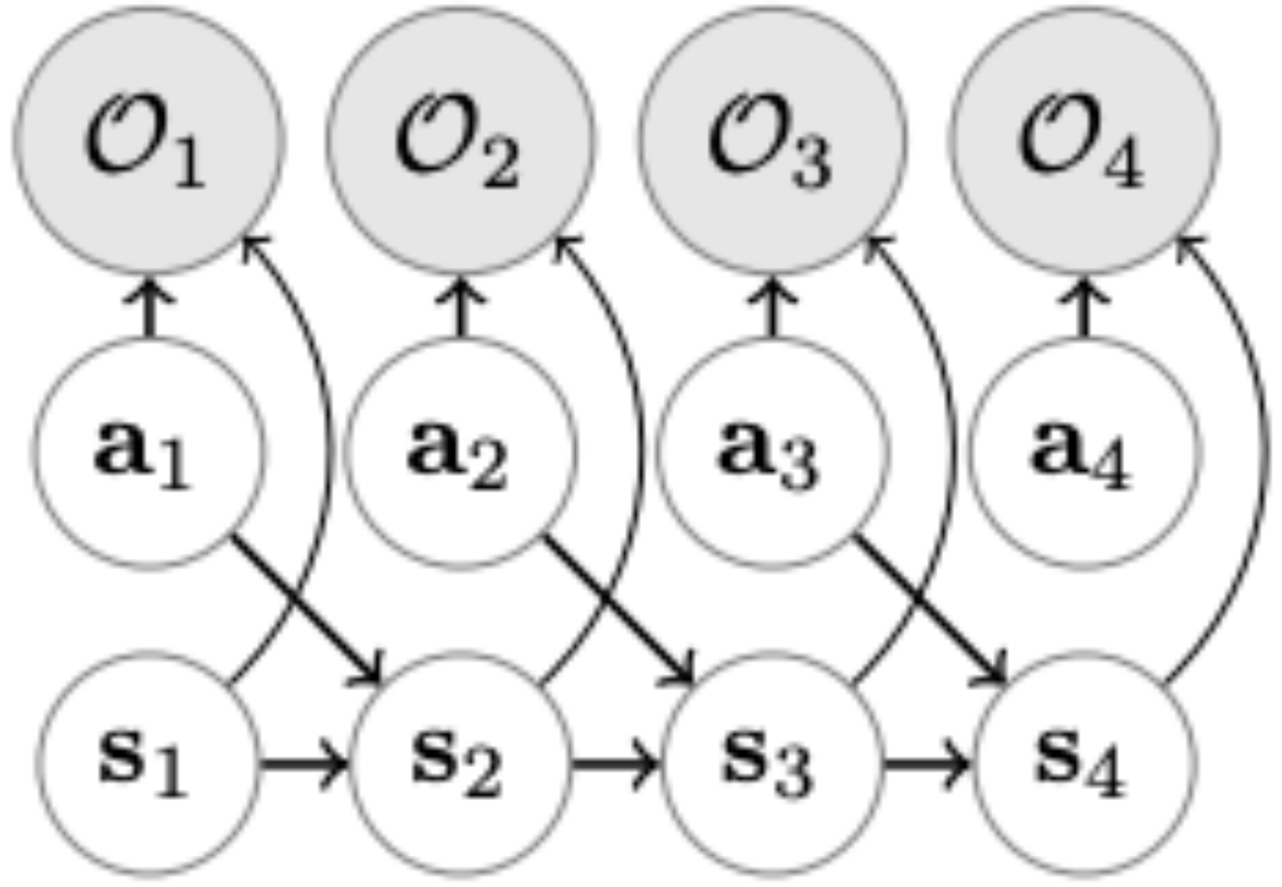
Input: env, α_Q, α_π, α_{enc}, α_{dec}, α_ψ
Randomly initialize θ_Q, θ_π, ϕ_{enc}, ϕ_{dec}, and ψ
Initialize empty replay buffer $\mathcal{D}$
Assign $\mathbf{z}^1 \leftarrow \vec{0}$
for i = 1, 2, … **do**
  Sample $\mathbf{z}^i \sim p_\psi(\mathbf{z}^i | \mathbf{z}^{1:i-1})$
  Collect trajectory τ^i from env with $\pi_\theta(\mathbf{a}|\mathbf{s}, \mathbf{z})$
  Update replay buffer $\mathcal{D}[i] \leftarrow \tau^i$
  for j = 1, 2, …, N **do**
    Sample a batch of episodes E from $\mathcal{D}$
    ▷ Update actor and critic
    $\theta_Q \leftarrow \theta_Q - \alpha_Q \nabla_{\theta_Q} \mathcal{J}_Q$
    $\theta_\pi \leftarrow \theta_\pi - \alpha_\pi \nabla_{\theta_\pi} \mathcal{J}_\pi$
    ▷ Update inference network
    $\phi_{\text{enc}} \leftarrow \phi_{\text{enc}} - \alpha_{\text{enc}} \nabla_{\phi_{\text{enc}}} (\mathcal{J}_{\text{dec}} + \mathcal{J}_{\text{KL}} + \mathcal{J}_Q)$
    ▷ Update model
    $\phi_{\text{dec}} \leftarrow \phi_{\text{dec}} - \alpha_{\text{dec}} \nabla_{\phi_{\text{dec}}} \mathcal{J}_{\text{dec}}$
    $\psi \leftarrow \psi - \alpha_\psi \nabla_\psi \mathcal{J}_{\text{KL}}$
  end for
end for

PGM Derivation

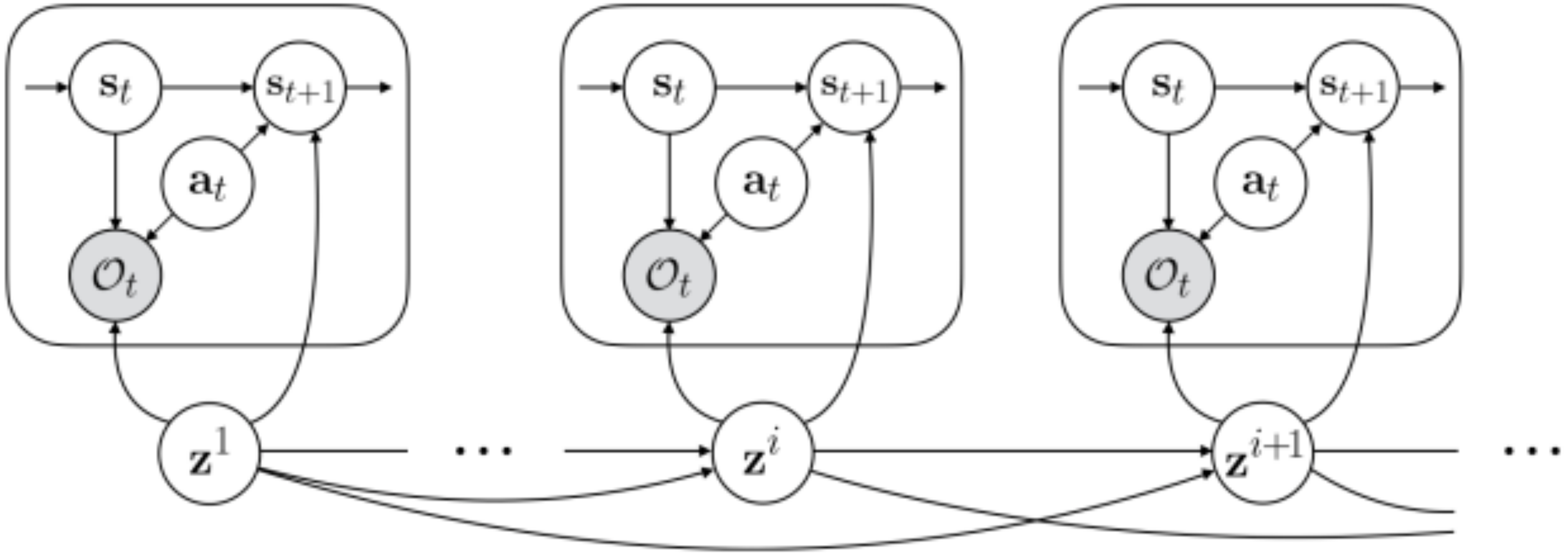
$$p(\tau|\mathbf{o}_{1:T}) \propto p(\tau, \mathbf{o}_{1:T}) = p(\mathbf{s}_1)\prod_{t=1}^{T} p(\mathcal{O}_t = 1|\mathbf{s}_t, \mathbf{a}_t)p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$$= p(\mathbf{s}_1)\prod_{t=1}^{T} \exp(r(\mathbf{s}_t, \mathbf{a}_t))p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$$= \left[p(\mathbf{s}_1)\prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)\right]\exp\left(\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t)\right).$$



PGM for Non-stationarity



- Two tiered model:
    - 1st: sequence of latent variables z^i as a Markov chain (i is episode number)
    - 2nd: MDP corresponding to each z^i
- Model posterior over z