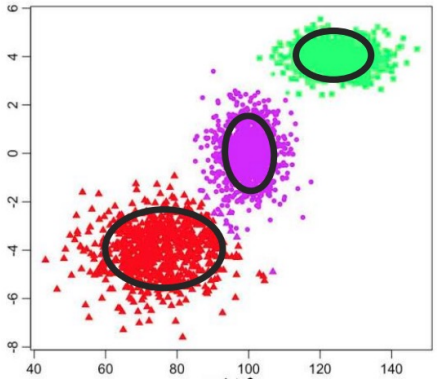
Reinforcement Learning as Probabilistic Inference

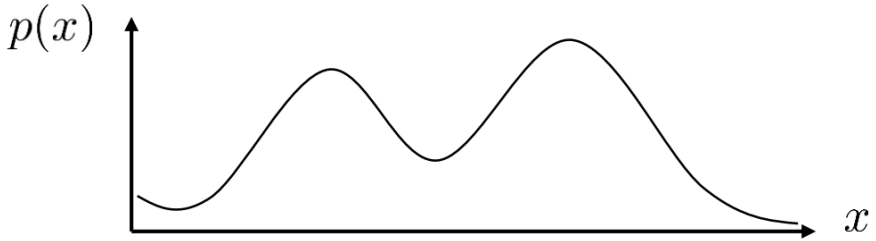
岩延 yany@ucas.ac.cn

Variational Inference

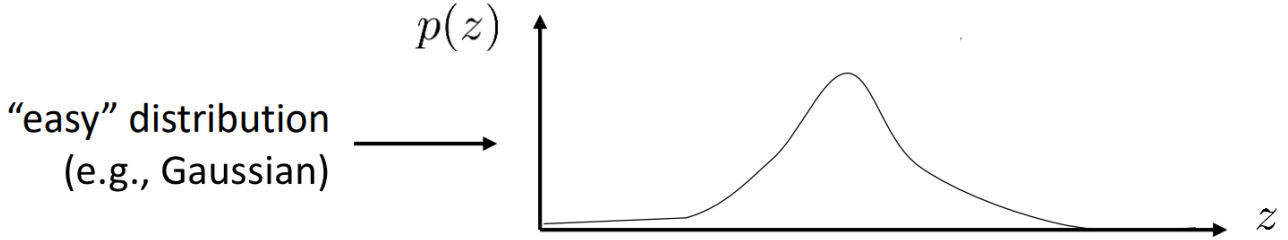
Variational Inference

- **Probabilistic models**
  - $p(x)$
  - $p(y|x)$
- **Latent variable models**
  - $p(x) = \sum_z p(x|z)p(z)$
  - $= \int p(x|z)p(z)dz$
  - $p(y|x) = \sum_z p(y|x,z)p(z)$





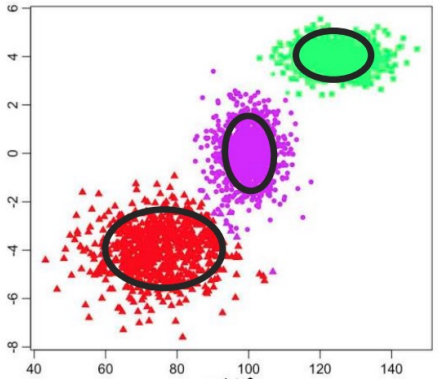
  - $p(z)$
    - Easy distribution (e.g., Gaussian)
  - $p(x|z)$
    - Easy distribution (e.g., Conditional Gaussian)

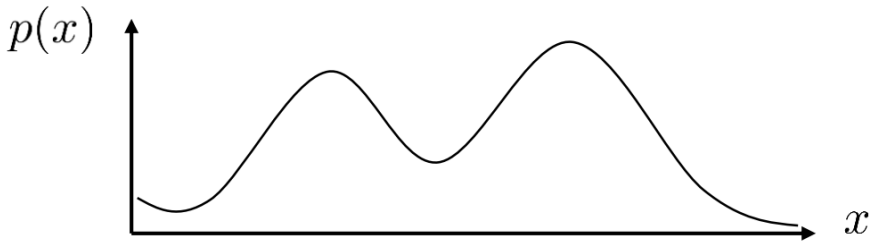


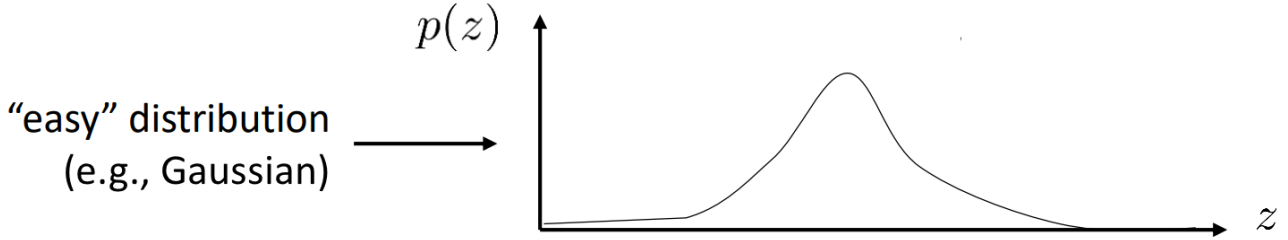
Variational Inference

Latent variable models

- **Latent variable models**
  - Bayes rules
    - $p(z|x) = \frac{p(x,z)}{p(x)}$
    - Where
      - $p(z|x)$ is posterior
      - $p(z)$ is prior

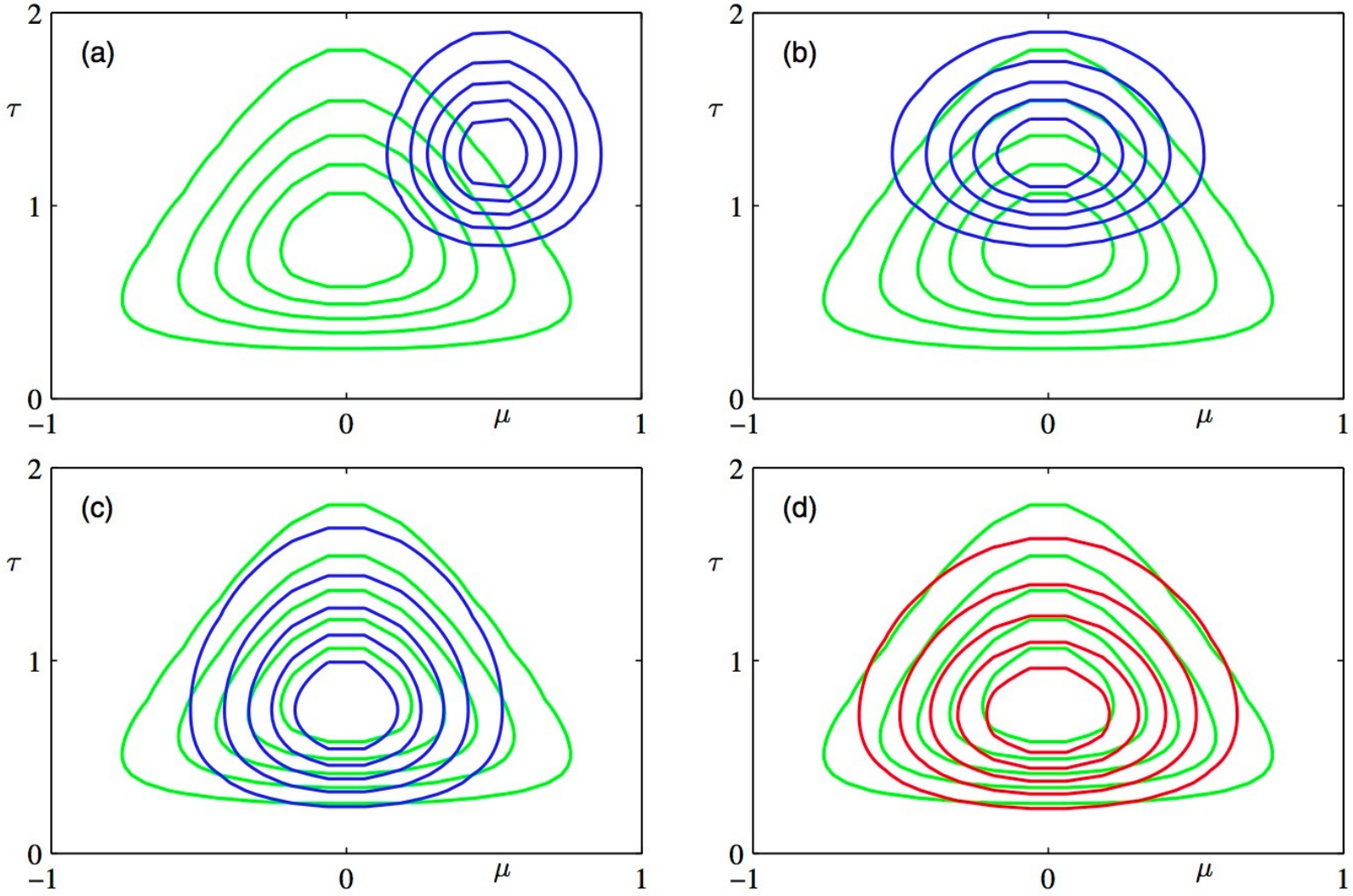






Variational Inference

Variational Inference



Variational Inference

Variational Inference

- **The model**
  - $p_\theta(x)$
- **The data**
  - $D = \{x_1, x_2, \ldots, x_N\}$
- **Maximum likelihood**
  - $\theta \leftarrow \underset{\theta}{\text{argmax}} \frac{1}{N} \sum_i \log p_\theta(x_i)$
- **Latent variable models**
  - $p(x) = \int p(x|z)p(z)dz$
- **Maximum likelihood**
  - $\theta \leftarrow \underset{\theta}{\text{argmax}} \frac{1}{N} \sum_i \log(\int p_\theta(x_i|z)p(z)dz)$
  - Intractable

Variational Inference

Variational Inference

- **Estimating the log-likelihood**
  - Guess the most likely z given x_i
  - There are many possible values of z, so use the distribution $p(z|x_i)$
  - How to calculate $p(z|x_i)$
    - Approximate with $q_i(z) = N(\mu_i, \sigma_i)$
    - Can be used to bound $\log p(x_i)$ for maximum likelihood
- **We can bound** $\log p(x_i)$
  - $p(x_i) = \int p(x_i|z)p(z)dz$
  - $\log p(x_i) = \log \int p(x_i|z)p(z)dz$
  - $= \log \int p(x_i|z)p(z)\frac{q_i(z)}{q_i(z)}dz$
  - $= \log \int q_i(z)\frac{p(x_i|z)p(z)}{q_i(z)}dz$
  - $= \log \mathrm{E}_{z\sim q_i(z)}[\frac{p(x_i|z)p(z)}{q_i(z)}]$

Variational Inference

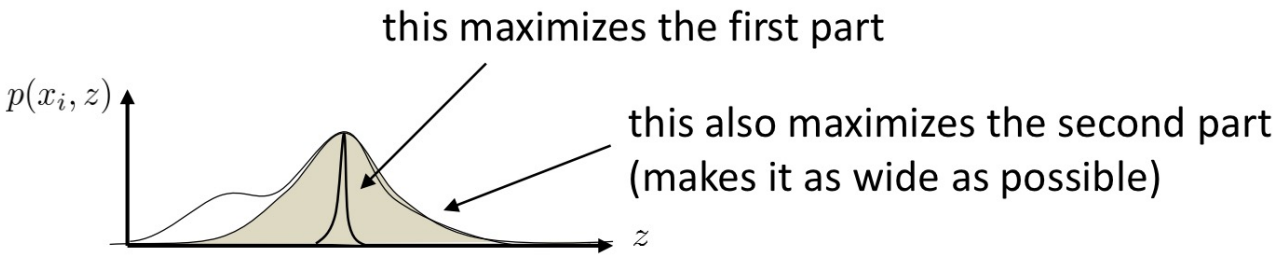
Variational Inference

- **We can bound** $\log p(x_i)$
  - $\log p(x_i) = \log \mathrm{E}_{z\sim q_i(z)}[\frac{p(x_i|z)p(z)}{q_i(z)}]$
  - Use the Jensen's inequality
    - $\log \mathrm{E}[y] \geq \mathrm{E}[\log y]$
  - $\log p(x_i) = \log \mathrm{E}_{z\sim q_i(z)}[\frac{p(x_i|z)p(z)}{q_i(z)}]$
  - $\geq \mathrm{E}_{z\sim q_i(z)}[\log \frac{p(x_i|z)p(z)}{q_i(z)}]$
  - $= \mathrm{E}_{z\sim q_i(z)}[\log p(x_i|z) + \log p(z)] - \mathrm{E}_{z\sim q_i(z)}[\log q_i(z)]$

Variational Inference

Variational Inference

- **We can bound** $\log p(x_i)$
  - Remember the definition of Entropy
    - $H(p) = -\mathrm{E}_{x\sim p(x)}[\log p(x)] = -\int p(x)\log p(x)\,dx$
    - How random is the random variable?
    - How large is the log probability in expectation under itself
  - The we have
    - $\log p(x_i) \geq \mathrm{E}_{z\sim q_i(z)}[\log p(x_i|z) + \log p(z)] + H(q_i)$
    - ELBO（Evidence Lower Bound Objective）
      - $L_i(p, q_i) = \mathrm{E}_{z\sim q_i(z)}[\log p(x_i|z) + \log p(z)] + H(q_i)$



Variational Inference

Variational Inference

- **KL-Divergence Revisit**
  - $D_{KL}(q||p) = \mathrm{E}_{x\sim q(x)}[\log \frac{q(x)}{p(x)}]$
  - $= \mathrm{E}_{x\sim q(x)}[\log q(x)] - \mathrm{E}_{x\sim q(x)}[\log p(x)]$
  - $= -\mathrm{E}_{x\sim q(x)}[\log p(x)] - H(q)$
  - How different are two distributions?
  - How small is the expected log probability of one distribution under another, minus entropy?

Inverse Reinforcement Learning

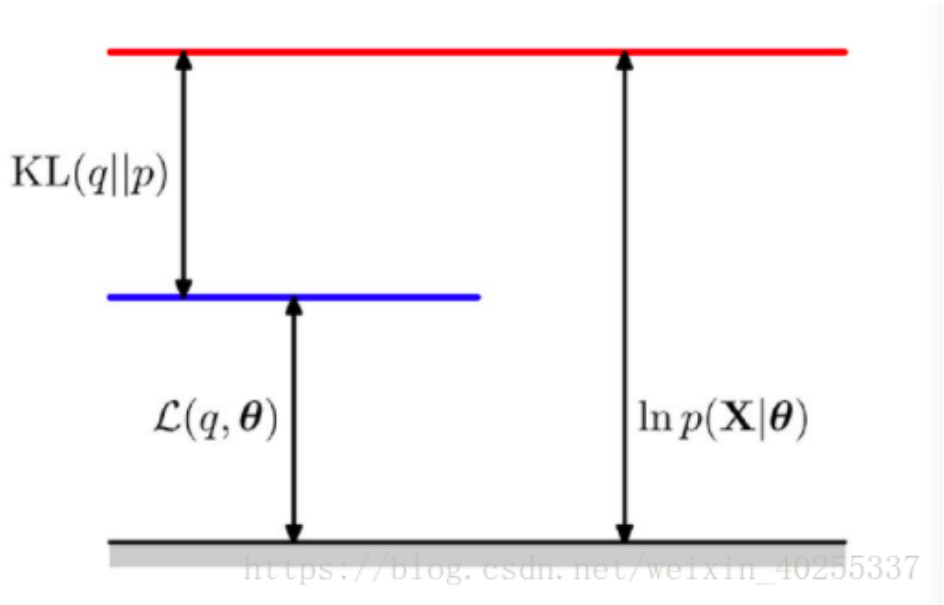
Variational Approximation

- **What makes a good $q_i(z)$?**
  - $q_i(z)$ should approximate $p(z|x_i)$
  - in terms of the KL-divergence
    - $D_{KL}(q_i(z)||p(z|x_i))$
  - $D_{KL}(q_i(\mathrm{z})||p(z|x_i)) = \mathrm{E}_{z\sim q_i(z)}[\log \frac{q_i(z)}{p(z|x_i)}]$
  - $= \mathrm{E}_{z\sim q_i(z)}[\log \frac{q_i(z)p(x_i)}{p(x_i,z)}] = \mathrm{E}_{z\sim q_i(z)}[\log \frac{q_i(z)p(x_i)}{p(x_i|z)p(z)}]$
  - $= -\mathrm{E}_{z\sim q_i(z)}[\log p(x_i|z) + \log p(z)] + \mathrm{E}_{z\sim q_i(z)}[\log q_i(z)] + \mathrm{E}_{z\sim q_i(z)}[\log p(x_i)]$
  - $= -\mathrm{E}_{z\sim q_i(z)}[\log p(x_i|z) + \log p(z)] - H(q_i) + \mathrm{E}_{z\sim q_i(z)}[\log p(x_i)]$
    - Noticed that $\log p(x_i)$ independent of q_i
  - $= -L_i(p, q_i) + \log p(x_i)$
  - Then we have
    - $\log p(x_i) = D_{KL}(q_i(\mathrm{z})||p(z|x_i)) + L_i(p, q_i)$
    - $\log p(x_i) \geq L_i(p, q_i)$

Variational Inference

Variational Inference

- **Maximizing $L_i(p, q_i)$ w.r.t. q_i minimizes KL-divergence**
  - $\log p(x_i) = D_{KL}(q_i(\mathrm{z})||p(z|x_i)) + L_i(p, q_i)$



  - Then our goal is to maximize ELBO $L_i(p, q_i)$
    - $L_i(p, q_i) = \mathrm{E}_{z\sim q_i(z)}[\log p(x_i|z) + \log p(z)] + H(q_i)$

Variational Inference

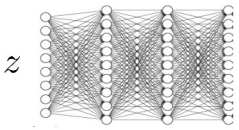
Variational Inference

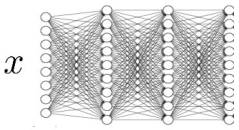
- **Maximum likelihood**
  - $L_i(p, q_i) = \mathrm{E}_{z\sim q_i(z)}[\log p_\theta(x_i|z) + \log p(z)] + H(q_i)$
  - $\theta \leftarrow \underset{\theta}{\operatorname{argmax}} \frac{1}{N}\sum_i \log p_\theta(x_i)$ Change to
    - $\theta \leftarrow \underset{\theta}{\operatorname{argmax}} \frac{1}{N}\sum_i L_i(p, q_i)$
  - For each x_i (or mini-batch)
    - Calculate $\nabla_\theta L_i(p, q_i)$
      - Sample $z\sim q_i(z)$
      - $\nabla_\theta L_i(p, q_i) \approx \nabla_\theta \log p_\theta(x_i|z)$
    - $\theta \leftarrow \theta + \alpha\nabla_\theta L_i(p, q_i)$
    - Update q_i to maximize $L_i(p, q_i)$
      - Let $q_i(z) = N(\mu_i, \sigma_i)$
      - Use gradient $\nabla_{\mu_i} L_i(p, q_i)$ and $\nabla_{\sigma_i} L_i(p, q_i)$ ascent on μ_i, σ_i

Variational Inference

Amortized variational inference

- **What is the problem?**
  - Too many parameters
    - $|\theta| + (|\mu_i| + |\sigma_i|) \times N$
- **How to solve?**
  - $q_i(z)$ should approximate $p(z|x_i)$
  - Learn a network $q_i(z) = q(z|x_i) \approx p_\theta(x_i|z)$





  - $\log p(x_i) \geq L_i\left(p_\theta(x_i|z), q_\phi(z|x_i)\right)$
  - $= \mathrm{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z) + \log p(z)] + H(q_\phi(z|x_i))$
  - For each x_i (or mini-batch)
    - Calculate $\nabla_\theta L_i\left(p_\theta(x_i|z), q_\phi(z|x_i)\right)$
      - Sample $z\sim q_\phi(z|x_i)$
      - $\nabla_\theta L_i(p, q_i) \approx \nabla_\theta \log p_\theta(x_i|z)$
    - $\theta \leftarrow \theta + \alpha \nabla_\theta L_i$
    - $\phi \leftarrow \phi + \alpha \nabla_\phi L_i$
  - How to calculate $\nabla_\phi L_i$?

Variational Inference

Amortized variational inference

- **How to calculate $\nabla_\phi L_i$?**
  - $L_i = \mathrm{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z) + \log p(z)] + H(q_\phi(z|x_i))$
  - $q_\phi(z|x) = N(\mu_\phi(x), \sigma_\phi(x))$
    - $H(q_\phi(z|x_i))$ can be calculated easily
  - Let $\log p_\theta(x_i|z) + \log p(z) = r(x_i, z)$
  - Then we have
    - $J(\phi) = L_i = \mathrm{E}_{z\sim q_\phi(z|x_i)}[r(x_i, z)]$
    - Just use the policy gradient!
      - $J(\phi) \approx \frac{1}{M}\sum_j \nabla_\phi \log q_\phi\left(z_j|x_i\right) r(x_i, z_j)$
  - What's the problem?
    - High variance

Variational Inference

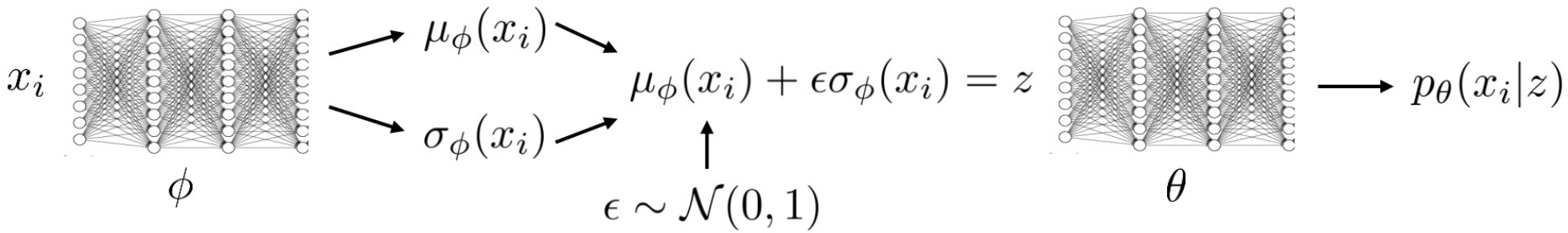
Reparameterization

- **How to calculate** $J(\phi) = \mathrm{E}_{z\sim q_\phi(z|x_i)}[r(x_i, z)]$
  - $q_\phi(z|x) = N(\mu_\phi(x), \sigma_\phi(x))$
  - We can rewrite $z = \mu_\phi(x) + \epsilon\sigma_\phi(x)$ with noise $\epsilon \in N(0,1)$
    - ϵ is independent to ϕ
  - Then we can rewrite
    - $J(\phi) = \mathrm{E}_{z\sim q_\phi(z|x_i)}[r(x_i, z)]$
    - $= \mathrm{E}_{\epsilon\sim N(0,1)}[r(x_i, \mu_\phi(x_i) + \epsilon\sigma_\phi(x_i))]$
  - Then calculate the $\nabla_\phi J(\phi)$
    - Sample $\epsilon_1, \epsilon_2, \ldots, \epsilon_M$ from $N(0,1)$
    - $\nabla_\phi J(\phi) \approx \frac{1}{M}\sum_j \nabla_\phi r(x_i, \mu_\phi(x_i) + \epsilon_j\sigma_\phi(x_i))$

Variational Inference

Reparameterization

- **Think in another way**
  - $L_i = \mathrm{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z) + \log p(z)] + H(q_\phi(z|x_i))$
  - $= \mathrm{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] + \mathrm{E}_{z\sim q_\phi(z|x_i)}[\log p(z)] + H(q_\phi(z|x_i))$
  - $= \mathrm{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - D_{KL}(q_\phi(z|x_i)||p(z))$
  - $= \mathrm{E}_{\epsilon\sim N(0,1)}\left[\log p_\theta(x_i|\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i))\right] - D_{KL}(q_\phi(z|x_i)||p(z))$
  - $\approx \log p_\theta(x_i|\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i)) - D_{KL}(q_\phi(z|x_i)||p(z))$



Variational Inference

Reparameterization

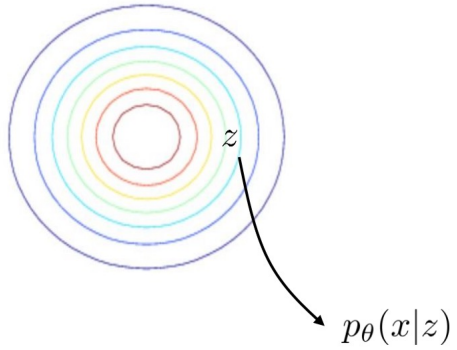
- **Policy gradient**
  - $J(\phi) \approx \frac{1}{M}\sum_j \nabla_\phi \log q_\phi(z_j|x_i)\, r(x_i, z_j)$
  - Can handle both discrete and continuous latent variables
  - High variance, requires multiple samples & small learning rates
- **Reparameterization trick**
  - $\nabla_\phi J(\phi) \approx \frac{1}{M}\sum_j \nabla_\phi r(x_i, \mu_\phi(x_i) + \epsilon_j \sigma_\phi(x_i))$
  - Only continuous latent variables
  - Very simple to implement
  - Low variance

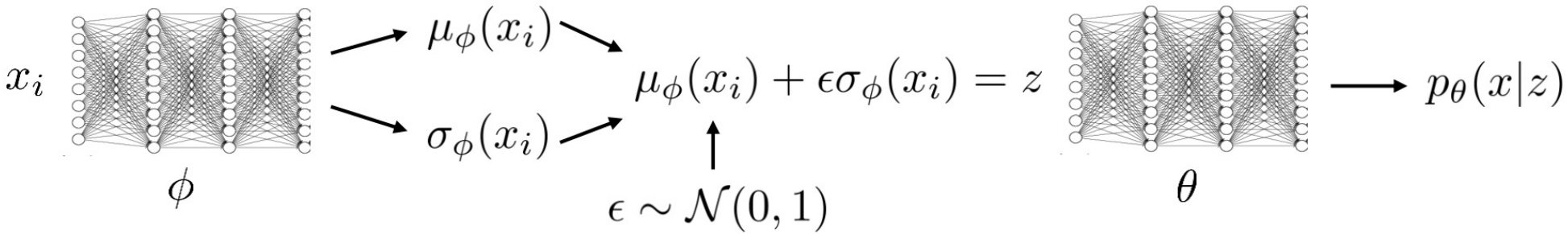
Variational Inference

The variational autoencoder

- **It's a generative model**

$q_\phi(z|x) = \mathcal{N}(\mu_\phi(x), \sigma_\phi(x))$ ϕ θ $p_\theta(x|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta(z))$



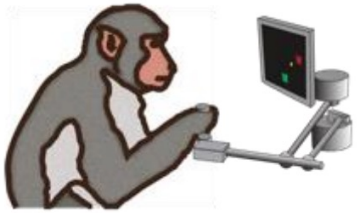


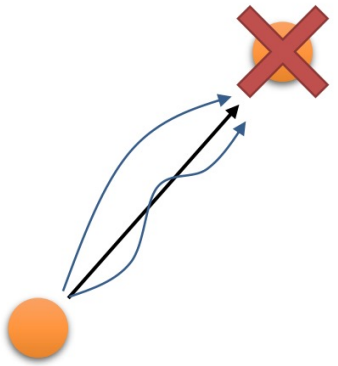
- $\max_{\theta,\phi} \frac{1}{N} \sum_i \log p_\theta(x_i|\mu_\phi(x_i) + \epsilon\sigma_\phi(x_i)) - D_{KL}(q_\phi(z|x_i)||p(z))$
- $p(x) = \int p(x|z)p(z)dz$
  - Sampling $z \sim p(z)$ $x \sim p(x|z)$

Probabilistic Graphical Model

Optimal Control

- **Does reinforcement learning and optimal control provide a reasonable model of human behavior?**
  - $a_1, a_2, \ldots, a_T = \underset{a_1, a_2, \ldots, a_T}{\text{argmax}} \sum_{t=1}^{T} r(s_t, a_t)$
  - $s_{t+1} = p(s_t, a_t)$
  - $\pi = \underset{\pi}{\text{argmax}}\, \mathrm{E}_{s_{t+1} \sim p(s_{t+1}|s_t, a_t), a_t \sim \pi(s_t|a_t)}[r(s_t, a_t)]$
  - $a_t \sim \pi(s_t|a_t)$
- **What if the data is not optimal?**
  - some mistakes matter more than others
  - behavior is stochastic
  - but good behavior is still the most likely

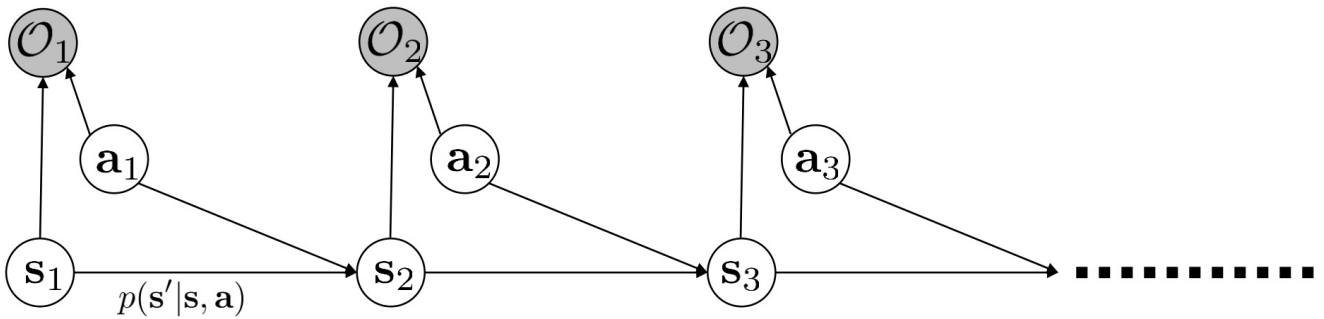




Probabilistic Graphical Model

Probabilistic Graphical Model

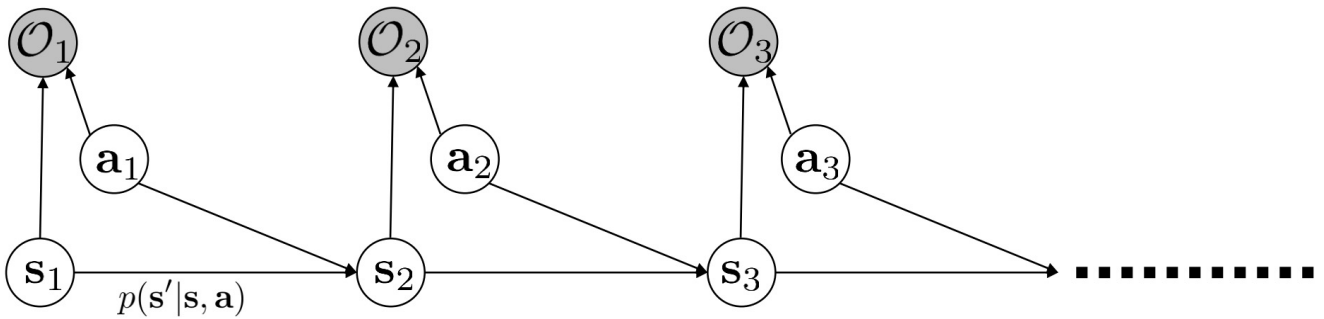
- **Probabilistic Graphical Model**
  - $p(\tau) = p(s_{1:T}, a_{1:T})$ =? ? No assumption of optimal behavior
  - $p(O_t|s_t, a_t) \propto \exp(r(s_t, a_t))$
  - $p(\tau|O_{1:T}) = \frac{p(\tau, O_{1:T})}{p(O_{1:T})} \propto p(\tau, O_{1:T})$
  - $= p(s_1) \prod_t p(O_t|s_t, a_t) p(s_{t+1}|s_t, a_t)$
  - $= p(s_1) \prod_t \exp\big(r(s_t, a_t)\big)\, p(s_{t+1}|s_t, a_t)$
  - $= p(\tau) \prod_t \exp\big(r(s_t, a_t)\big)$
  - $= p(\tau) \exp(\sum_t r(s_t, a_t))$
  - Optimal policy
    - $p(a_t|s_t, O_{t:T} = 1)$



Probabilistic Graphical Model

Probabilistic Graphical Model

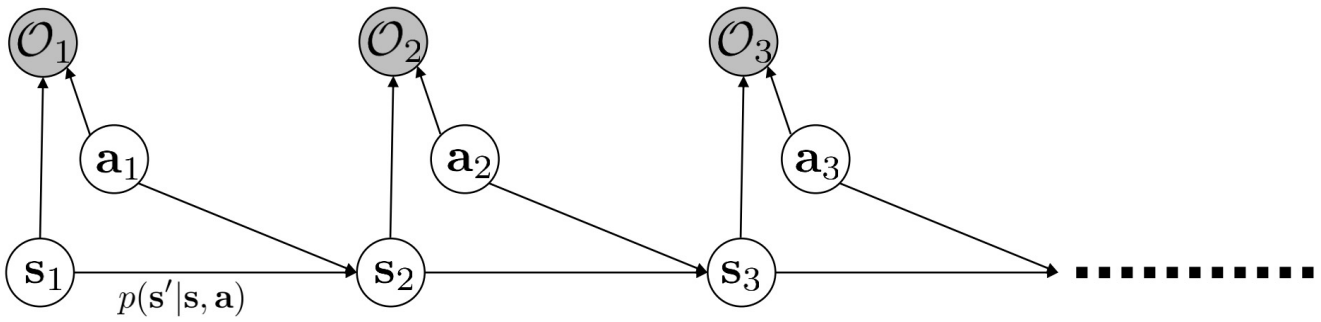
- **Why Probabilistic Graphical Model?**
  - Can model suboptimal behavior (important for inverse RL)
  - Can apply inference algorithms to solve control and planning problems
  - Provides an explanation for why stochastic behavior might be preferred (useful for exploration and transfer learning)



Probabilistic Graphical Model

Inference = Control

- **How to do inference?**
  - Compute backward message $\beta_t(s_t, a_t) = p(O_{t:T}|s_t, a_t)$
  - Compute policy $p(a_t|s_t, O_{1:T})$
  - Compute forward messages $\alpha_t(s_t) = p(s_t|O_{1:t-1})$



Probabilistic Graphical Model

Inference = Control

- **Backward messages**
  - $\beta_t(s_t, a_t) = p(O_{t:T}|s_t, a_t)$
  - $= \int p(O_{t:T}, s_{t+1}|s_t, a_t) ds_{t+1}$
  - $= \int p(O_{t+1:T}|s_{t+1}) p(s_{t+1}|s_t, a_t) p(O_t|s_t, a_t) ds_{t+1}$
  - Let $\beta_t(s_t) = p(O_{t:T}|s_t)$, we have
    - $\beta_t(s_t, a_t) = \int \beta_{t+1}(s_{t+1}) p(s_{t+1}|s_t, a_t) p(O_t|s_t, a_t) ds_{t+1}$
    - $\beta_t(s_t) = p(O_{t:T}|s_t)$
    - $= \int p(O_{t:T}|s_t, a_t) p(a_t|s_t) da_t$
    - $= \int \beta_t(s_t, a_t) p(a_t|s_t) da_t$
      - Where $p(O_{t:T}|s_t, a_t) = \beta_t(s_t, a_t)$
  - Then we have
    - $\beta_T(s_T, a_T) = p(O_T|s_t, a_t) \propto \exp(r(s_T, a_T))$
    - From $t = T - 1$ to 1
      - $\beta_t(s_t, a_t) = p(O_t|s_t, a_t) \mathrm{E}_{s_{t+1} \sim p(s_{t+1}|s_t, a_t)}[\beta_{t+1}(s_{t+1})]$
      - $\beta_t(s_t) = \mathrm{E}_{a_t \sim p(a_t|s_t)}[\beta_t(s_t, a_t)]$

Probabilistic Graphical Model

Backward Messages vs Value Iteration

- **Backward messages**
  - From $t = T - 1$ to 1
    - $\beta_t(s_t, a_t) = p(O_t|s_t, a_t)\mathrm{E}_{s_{t+1}\sim p(s_{t+1}|s_t,a_t)}[\beta_{t+1}(s_{t+1})]$
    - $\beta_t(s_t) = \mathrm{E}_{a_t\sim p(a_t|s_t)}[\beta_t(s_t, a_t)]$
  - Log space message
    - Let $V(s_t) = \log \beta_t(s_t)$, then $\beta_t(s_t) = \exp(V(s_t))$
    - Let $Q(s_t, a_t) = \log \beta_t(s_t, a_t)$, then $\beta_t(s_t, a_t) = \exp(Q(s_t, a_t))$
  - Then we have
    - $V(s_t) = \log \int \exp(Q(s_t, a_t))\, da_t$
    - $V(s_t) \to \max_{a_t} Q(s_t, a_t)$ as $Q(s_t, a_t)$ gets bigger
      - Means **soft max**!
    - $Q(s_t, a_t) = \log p(O_t|s_t, a_t)\mathrm{E}_{s_{t+1}\sim p(s_{t+1}|s_t,a_t)}[\beta_{t+1}(s_{t+1})]$
    - $= r(s_t, a_t) + \log \mathrm{E}[\beta_{t+1}(s_{t+1})]$
    - $= r(s_t, a_t) + \log \mathrm{E}[\exp(V(s_{t+1}))]$
      - For deterministic transition we have
        - $Q(s_t, a_t) = r(s_t, a_t) + V(s_{t+1})$

Probabilistic Graphical Model

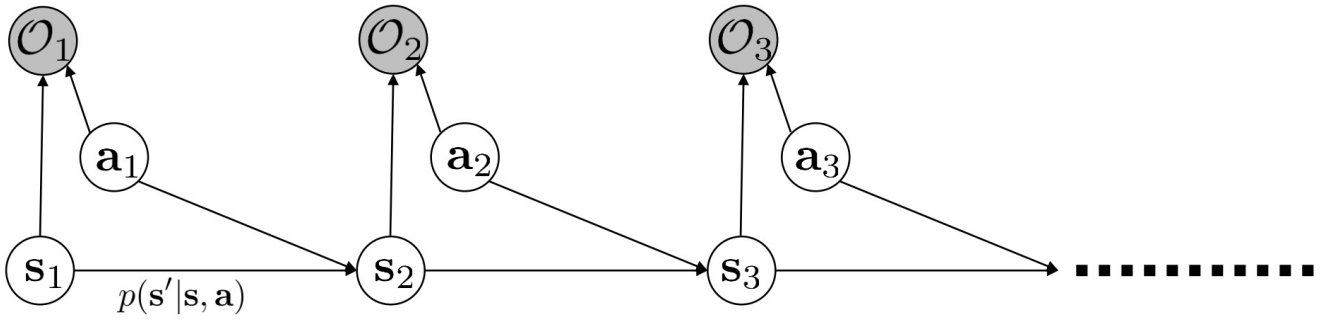
Backward Messages vs Value Iteration

- **Remember value iteration**
  - $V_{k+1}(s) = \max_a \mathrm{E}[R_{t+1} + \gamma V_k(S_{t+1}) | S_t = s, A_t = a]$
  - $V(s) \leftarrow \max_a Q(s,a)$
  - $Q(s,a) \leftarrow r(s,a) + \gamma \mathrm{E}[V(s')]$
- **Backward messages**
  - $V(s_t) = \log \int \exp(Q(s_t, a_t))\, da_t \rightarrow \max_{a_t} Q(s_t, a_t)$
  - $Q(s_t, a_t) = r(s_t, a_t) + \log \mathrm{E}[\exp(V(s_{t+1}))]$
  - $\log \mathrm{E}[\exp(V(s_{t+1}))]$is an optimistic transition, not a good idea
  - A better stochastic model
    - $Q(s_t, a_t) = r(s_t, a_t) + \mathrm{E}[V(s_{t+1})]$

Probabilistic Graphical Model

Backward Pass Summary

- **Backward messages**
  - $\beta_t(s_t, a_t) = p(O_{t:T}|s_t, a_t)$
    - Probability that we can be optimal at step t through T given that we take action a_t in state s_t
  - From $t = T - 1$ to 1
    - $\beta_t(s_t, a_t) = p(O_t|s_t, a_t)\mathrm{E}_{s_{t+1}\sim p(s_{t+1}|s_t,a_t)}[\beta_{t+1}(s_{t+1})]$
    - $\beta_t(s_t) = \mathrm{E}_{a_t\sim p(a_t|s_t)}[\beta_t(s_t, a_t)]$
  - Log space message
    - Let$V(s_t) = \log \beta_t(s_t)$
    - Let $Q(s_t, a_t) = \log \beta_t(s_t, a_t)$
    - Log of $\beta_t$is Q-function-like



Probabilistic Graphical Model

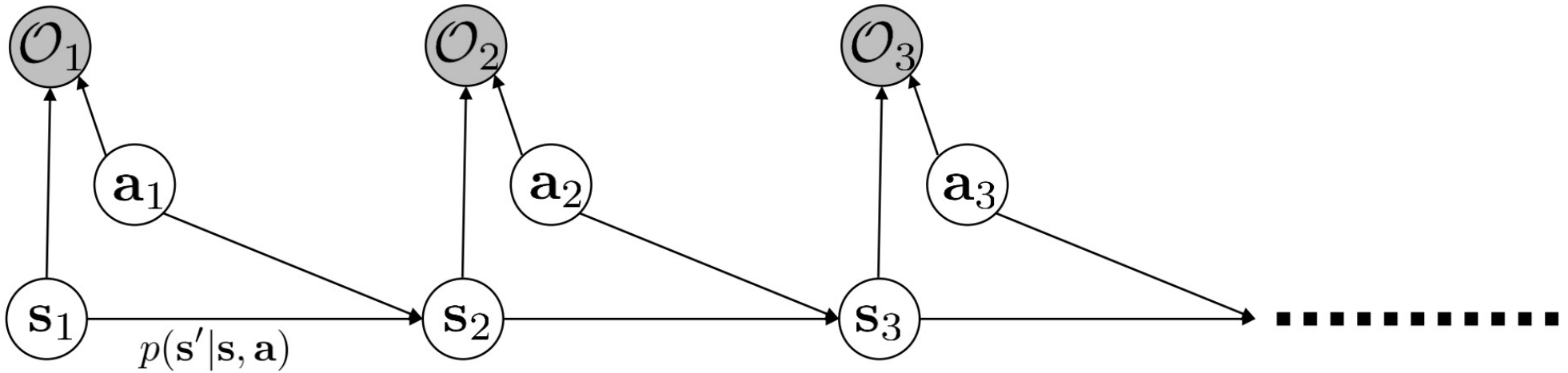
The Action Prior

- **We assumed uniform action prior**
  - $p(O_{t+1:T}|s_{t+1}) = \beta_{t+1}(s_{t+1})$
  - $= \int p(O_{t+1:T}|s_{t+1}, a_{t+1}) p(a_{t+1}|s_{t+1}) da_{t+1}$
- **What if action prior is not uniform?**
  - $V(s_t) = \log \int \exp(Q(s_t, a_t) + \log p(a_t|s_t))\, da_t$
  - $Q(s_t, a_t) = r(s_t, a_t) + \log \mathrm{E}[\exp(V(s_{t+1}))]$
  - Let
    - $\tilde{Q}(s_t, a_t) = r(s_t, a_t) + \log p(a_t|s_t) + \log \mathrm{E}[\exp(V(s_{t+1}))]$
  - Then we have
    - $V(s_t) = \log \int \exp(\tilde{Q}(s_t, a_t))\, da_t$
  - Can always fold the action prior into the reward! uniform action prior can be assumed without loss of generality

Probabilistic Graphical Model

Policy Computation

- **Compute policy** $p(a_t|s_t, O_{1:T})$
  - $\beta_t(s_t, a_t) = p(O_{t:T}|s_t, a_t)$ $\quad\quad$ $\beta_t(s_t) = p(O_{t:T}|s_t)$
  - $p(a_t|s_t, O_{1:T}) = \pi(a_t|s_t) = p(a_t|s_t, O_{t:T})$
  - $= \frac{p(a_t, s_t|O_{t:T})}{p(s_t|O_{t:T})}$
  - $= \frac{p(O_{t:T}|a_t, s_t)\, p(a_t, s_t)/p(O_{t:T})}{p(O_{t:T}|s_t)p(s_t)/p(O_{t:T})} = \frac{p(O_{t:T}|a_t, s_t)p(a_t, s_t)}{p(O_{t:T}|s_t)p(s_t)}$
  - $= \frac{\beta_t(s_t, a_t)}{\beta_t(s_t)} p(a_t|s_t) = \frac{\beta_t(s_t, a_t)}{\beta_t(s_t)}$



Probabilistic Graphical Model

Policy Computation

- **Policy computation with value functions**
  - From $t = T - 1$ to 1
    - $Q(s_t, a_t) = r(s_t, a_t) + \log \mathrm{E}[\exp(V(s_{t+1}))]$
    - $V(s_t) = \log \int \exp(Q(s_t, a_t))\, da_t$
  - $\pi(a_t|s_t) = \frac{\beta_t(s_t, a_t)}{\beta_t(s_t)}$
  - $V(s_t) = \log \beta_t(s_t)$
  - $Q(s_t, a_t) = \log \beta_t(s_t, a_t)$
  - Then we have
    - $\pi(a_t|s_t) = \exp(Q(s_t, a_t) - V(s_t))$
    - $= \exp(A(s_t, a_t))$
      - Better action are more probable

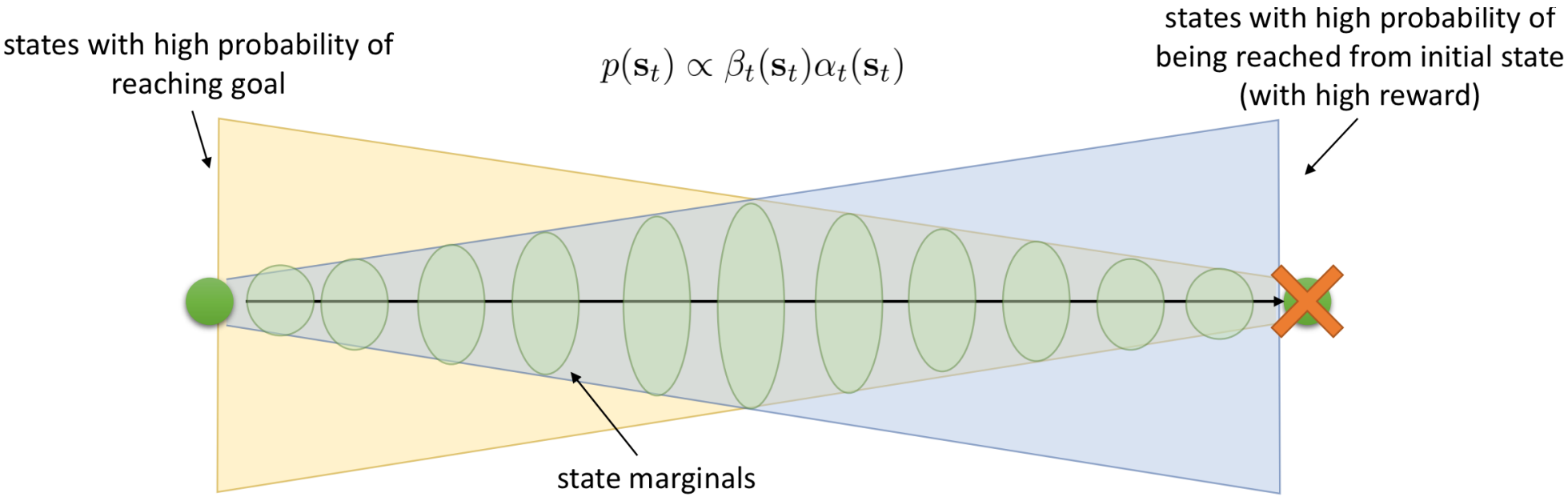
Probabilistic Graphical Model

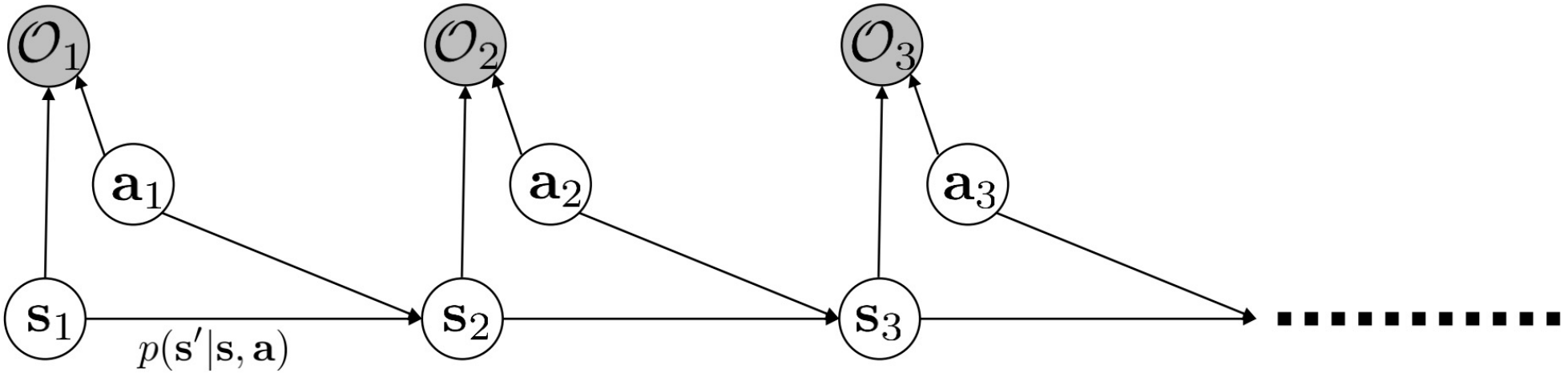
Forward Messages

- **Forward messages**
  - $\alpha_1(s_1) = p(s_1)$
  - $\alpha_t(s_t) = p(s_t|O_{1:t-1})$
  - $= \int p(s_t|s_{t-1}, a_{t-1}) p(a_{t-1}|s_{t-1}, O_{t-1}) p(s_{t-1}|O_{1:t-2}) ds_{t-1} da_{t-1}$
  - $= \int p(s_t|s_{t-1}, a_{t-1}) p(a_{t-1}|s_{t-1}, O_{t-1}) \alpha_{t-1}(s_{t-1}) ds_{t-1} da_{t-1}$
  - Where
    - $p(a_{t-1}|s_{t-1}, O_{t-1}) = \frac{p(O_{t-1}|s_{t-1}, a_{t-1})\, p(a_{t-1}|s_{t-1})}{p(O_{t-1}|s_{t-1})}$
    - Remember $p(O_{t-1}|s_{t-1}, a_{t-1}) \propto \exp(r(s_{t-1}, a_{t-1}))$
    - Then $p(a_{t-1}|s_{t-1}, O_{t-1}) \propto \exp(r(s_{t-1}, a_{t-1}))$
- **We want $p(s_t|O_{1:T})$**
  - $p(s_t|O_{1:T}) = \frac{p(s_t, O_{1:T})}{p(O_{1:T})} = \frac{p(O_{t:T}|s_t)\, p(s_t, O_{1:t-1})}{p(O_{1:T})}$
  - $\propto \beta_t(s_t) \alpha_t(s_t)$

Probabilistic Graphical Model

Forward/backward message intersection





Probabilistic Graphical Model

Summary

- **Probabilistic graphical model for optimal control**
- **Control = inference**
- **Very similar to dynamic programming, value iteration, etc. (but "soft")**

Probabilistic Graphical Model

The Optimism Problem

- **The optimism issue**
  - $Q(s_t, a_t) = r(s_t, a_t) + \log \mathrm{E}[\exp(V(s_{t+1}))]$
  - $\log \mathrm{E}[\exp(V(s_{t+1}))]$is an optimistic transition, not a good idea
- **Inference problem**
  - $p(s_{1:T}, a_{1:T}|O_{1:T})$
  - We get policy $p(a_t|s_t, O_{1:T})$ by marginalizing and conditioning
    - Means given that you obtained high reward, what was your action probability?
  - we can also get $p(s_{t+1}|s_t, a_t, O_{1:T})$ by marginalizing and conditioning
    - $p(s_{t+1}|s_t, a_t, O_{1:T}) \neq p(s_{t+1}|s_t, a_t)$
    - Means given you obtained high reward, what was your transition probability?
  - We want to get right action $p(a_t|s_t, O_{1:T})$

Probabilistic Graphical Model

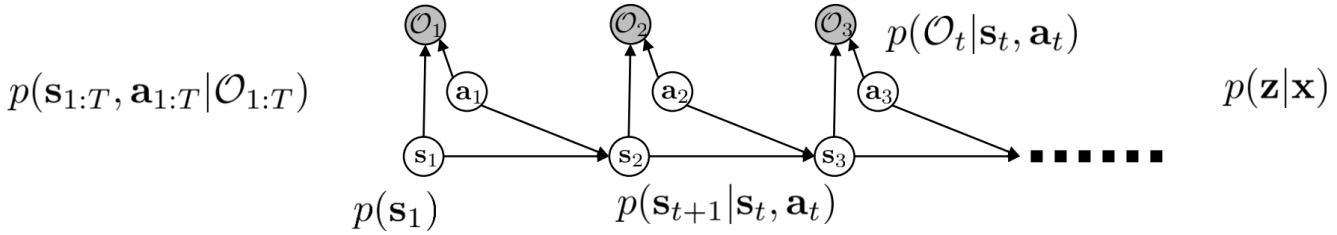
The Optimism Problem

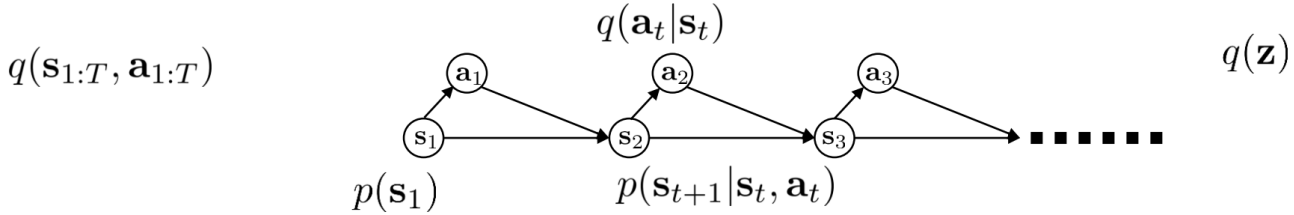
- **The optimism problem**
  - given that you obtained high reward, what was your action probability, given that your transition probability didn't change?
- **How can we solve that?**
  - Find another distribution $q(s_{1:T}, a_{1:T})$ that is close to $p(s_{1:T}, a_{1:T}|O_{1:T})$ but has dynamics $p(s_{t+1}|s_t, a_t)$
  - Can be solved by variational inference
    - Let $x = O_{1:T}$
    - Let $z = s_{1:T}, a_{1:T}$
    - The problem is equal to
      - Find $q(z)$ to approximate $p(z|x)$

Probabilistic Graphical Model

Control via Variational Inference

- **How to solve it?**
  - Let $q(z) = q(s_{1:T}, a_{1:T}) = q(\tau) = p(s_1)\prod_t p(s_{t+1}|s_t, a_t)\,\pi(a_t|s_t)$
  - Let $p(z|x) = p(s_{1:T}, a_{1:T}|O_{1:T}) = p(\tau|O_{1:T})$
  - Remember $p(\tau|O_{1:T}) = \frac{p(\tau, O_{1:T})}{p(O_{1:T})} \propto p(\tau, O_{1:T}) = p(x, z)$
  - $= p(s_1)\prod_t p(O_t|s_t, a_t)p(s_{t+1}|s_t, a_t)$
  - $= p(\tau)\prod_t \exp\big(r(s_t, a_t)\big)$
  - $= p(\tau)\exp(\sum_t r(s_t, a_t))$





Probabilistic Graphical Model

Control via Variational Inference

- **The variational lower bound**
  - Remember
    - $\log p(x) \geq \mathrm{E}_{z\sim q(z)}[\log p(x,z) - \log q(z)]$
  - Then
    - $\log p(x) = \log p(O_{1:T})$
    - $\geq \mathrm{E}_{(s_{1:T},a_{1:T})\sim\pi}[\log p(s_1) + \sum_{t=1}^{T} \log p(s_{t+1}|s_t,a_t) + \sum_{t=1}^{T} \log p(O_t|s_t,a_t)$
    - $- \log p(s_1) - \sum_{t=1}^{T} \log p(s_{t+1}|s_t,a_t) - \sum_{t=1}^{T} \log \pi(a_t|s_t)]$
    - $= \mathrm{E}_{(s_{1:T},a_{1:T})\sim\pi}[\sum_t r(s_t,a_t) - \log \pi(a_t|s_t)]$
    - $= \sum_t \mathrm{E}_{(s_t,a_t)\sim\pi}[r(s_t,a_t) + H(\pi(a_t|s_t))]$
      - **Maximize reward and action entropy**
      - Temperature could be used
        - $\sum_t \mathrm{E}_{(s_t,a_t)\sim\pi}[r(s_t,a_t) + \alpha H(\pi(a_t|s_t))]$

Probabilistic Graphical Model

Control via Variational Inference

- **Optimizing the variational lower bound**
  - $\log p(x) = \log p(O_{1:T}) \geq \sum_t \mathrm{E}_{(s_t,a_t)\sim\pi}[r(s_t,a_t) + H(\pi(a_t|s_t))]$
  -
  - Solve for $\pi(a_T|s_T)$ first
    - $\pi(a_T|s_T) = \operatorname{argmax} \mathrm{E}_{s_T\sim q(s_T)}[\mathrm{E}_{a_T\sim\pi(a_T|s_T)}[r(s_T,a_T)] + H(\pi(a_T|s_T))]$
    - $= \operatorname{argmax} \mathrm{E}_{s_T\sim q(s_T)}[\mathrm{E}_{a_T\sim\pi(a_T|s_T)}[r(s_T,a_T) - \log \pi(a_T|s_T)]]$
    - Solved when $\pi\,(a_T|s_T) \propto \exp(r(s_T,a_T))$
      - $\pi(a_T|s_T) = \frac{\exp(r(s_T,a_T))}{\int \exp(r(s_T,a))da} = \frac{\exp(r(s_T,a_T))}{\exp(V_T(s_T))} = \exp(Q(s_T,a_T) - V(s_T))$
      - Where $V(s_T) = \log \int \exp(Q(s_T,a_T))\,da_T = \log \int \exp(r(s_T,a_T))\,da_T$
    - Then we have
      - $\mathrm{E}_{s_T\sim q(s_T)}[\mathrm{E}_{a_T\sim\pi(a_T|s_T)}[r(s_T,a_T) - \log \pi(a_T|s_T)]]$
      - $= \mathrm{E}_{s_T\sim q(s_T)}[\mathrm{E}_{a_T\sim\pi(a_T|s_T)}[V(s_T)]]$

Probabilistic Graphical Model

Control via Variational Inference

- **Solve recursively**
  - $\pi(a_t|s_t) = \text{argmax}\, \mathrm{E}_{s_t,a_t\sim\pi}[r(s_t,a_t) + H\big(\pi(a_t|s_t)\big)] + \mathrm{E}_{s_{t+1}\sim p(s_{t+1}|s_t,a_t)}[V(s_{t+1})]]$
  - Now define $Q(s_t,a_t) = r(s_t,a_t) + \mathrm{E}[V(s_{t+1})]$
    - Now it's regular Bellman backup
    - A better stochastic model
  - $\pi\,(a_t|s_t) = \text{argmax}\, \mathrm{E}_{s_t\sim q(s_t)}[\mathrm{E}_{a_t\sim\pi(a_t|s_t)}[Q(s_t,a_t)] + H(\pi(a_t|s_t))]$
  - $= \text{argmax}\, \mathrm{E}_{s_t\sim q(s_t)}[\mathrm{E}_{a_t\sim\pi(a_t|s_t)}[Q(s_t,a_t) - \log\pi(a_t|s_t)]]$
    - Solved when $\pi\,(a_t|s_t) \propto \exp(r(s_t,a_t))$
  - $\pi\,(a_t|s_t) = \exp(Q(s_t,a_t) - V(s_t))$
  - $V(s_t) = \log\int \exp(Q(s_t,a_t))\,da_t = \mathrm{E}_{a_t\sim\pi}[Q(s_t,a_t) - \log\pi(a_t|s_t)]$
  - $Q(s_t,a_t) = r(s_t,a_t) + \mathrm{E}[V(s_{t+1})]$

Probabilistic Graphical Model

Backward pass summary - variational

- **Backward pass**
  - From $t = T - 1$ to 1
    - $Q(s_t, a_t) = r(s_t, a_t) + \mathrm{E}[V(s_{t+1})]$
    - $V(s_t) = \log \int \exp(Q(s_t, a_t))\, da_t$
- **Value iteration**
  - $Q(s, a) \leftarrow r(s, a) + \gamma \mathrm{E}[V(s')]$
  - $V(s) \leftarrow \max_a Q(s, a)$
- **Soft value iteration**
  - $Q(s, a) \leftarrow r(s, a) + \gamma \mathrm{E}[V(s')]$
  - $V(s) \leftarrow \text{soft} \max_a Q(s, a)$
- **Discount and temperature**
  - Discounted SOC $Q(s_t, a_t) = r(s_t, a_t) + \gamma \mathrm{E}[V(s_{t+1})]$
  - Explicit temperature $V(s_t) = \alpha \log \int \exp(\frac{1}{\alpha} Q(s_t, a_t))\, da_t$
    - With temperature $\pi(a_t | s_t) = \exp(\frac{1}{\alpha} Q(s_t, a_t) - \frac{1}{\alpha} V(s_t))$
    - $= \exp(\frac{1}{\alpha} A(s_t, a_t))$

Probabilistic Graphical Model

Q-learning with soft optimality

- **Q-learning**
  - $\phi \leftarrow \phi + \alpha \nabla_\phi Q_\phi(s,a)(r(s,a) + \gamma V(s') - Q_\phi(s,a))$
  - Target: $V(s') = \max_{a'} Q_\phi(s',a')$
- **Q-learning with soft optimality**
  - Soft Q-learning: $\phi \leftarrow \phi + \alpha \nabla_\phi Q_\phi(s,a)(r(s,a) + \gamma V(s') - Q_\phi(s,a))$
  - Target: $V(s') = \text{soft}\max_{a'} Q_\phi(s',a') = \log \int \exp(Q_\phi(s',a'))\, da'$
  - $\pi(a|s) = \exp\left(Q_\phi(s,a) - V(s)\right) = \exp(A(s,a))$
- **Soft DQN**
  - Take some action a_i and observe (s_i, a_i, s_i', r_i), add it to $\boldsymbol{B}$
  - Sample mini-batch of transitions $\{(s_i, a_i, s_i', r_i)\}$ from $\boldsymbol{B}$ uniformly
  - Compute $y_i = r(s_i,a_i) + \gamma \text{soft}\max_{a_i'} Q^\pi_{\phi'}(s_i', a_i')$ using target network $Q^\pi_{\phi'}$
  - $\phi \leftarrow \phi - \alpha \sum_i \frac{dQ^\pi_\phi}{d\phi}(s_i,a_i)\left(Q^\pi_\phi(s_i,a_i) - y_i\right)$
  - Update ϕ': copy ϕ every N steps

Probabilistic Graphical Model

Policy gradient with soft optimality

- **Entropy regularized policy gradient**
  - When $\pi(a|s) = \exp\left(Q_\phi(s,a) - V(s)\right)$
    - It optimize $\sum_t \mathrm{E}_{\pi(s_t,a_t)}[r(s_t,a_t)] + \mathrm{E}_{\pi(s_t)}[H(\pi(a_t|s_t))]$
    - $H(\pi(a_t|s_t))$ is the policy entropy
  - $\pi(a|s) \propto \exp\left(Q_\phi(s,a)\right)$ when π minimizes $D_{KL}(\pi(a|s)||\frac{1}{Z}\exp\left(Q_\phi(s,a)\right))$
    - $D_{KL}(\pi(a|s)||\frac{1}{Z}\exp\left(Q_\phi(s,a)\right)) = \mathrm{E}_{\pi(a|s)}\left[Q_\phi(s,a)\right] + H(\pi)$
  - Entropy regularized policy gradient combats premature entropy collapse
  - Closely related to soft Q-learning

Probabilistic Graphical Model

Soft Actor-Critic

- **Soft Policy Iteration**
  - Still optimize $\sum_t \mathrm{E}_{\pi(s_t,a_t)}[r(s_t,a_t)] + \mathrm{E}_{\pi(s_t)}[H(\pi(a_t|s_t))]$
  - Evaluation
    - $Q(s_t,a_t) = r(s_t,a_t) + \gamma \mathrm{E}_{s_{t+1}\sim p}[V(s_{t+1})]$
    - $V(s_t) = \mathrm{E}_{a_t\sim\pi}[Q(s_t,a_t) - \alpha \log \pi(a_t|s_t)]$
  - Improvement
    - $\pi_{\text{new}} = \underset{\pi'\in\Pi}{\operatorname{argmin}}\, D_{KL}(\pi'(\cdot|s)||\frac{\exp\left(Q_{\pi old}(s_t,\cdot)\right)}{Z_{\pi old}(s_t)})$

Probabilistic Graphical Model

Soft Actor-Critic

- **Soft Actor-Critic**
  - Use neural network $Q_\theta(s_t, a_t)$, $\pi_\phi(a_t|s_t)$, $V_\psi(s_t)$
  - Train value function
    - $J_\psi(V) = \mathrm{E}_{s_t \sim D}[\frac{1}{2}(V_\psi(s_t) - \mathrm{E}_{a_t \sim \pi_\phi}[Q_\theta(s_t, a_t) - \alpha \log \pi_\phi(a_t|s_t)]\ )^2]$
  - Train Q function
    - $J_\theta(Q) = \mathrm{E}_{s_t, a_t \sim D}[\frac{1}{2}(Q_\theta(s_t, a_t) - \hat{Q}_\theta(s_t, a_t))^2]$
      - Where $\hat{Q}_\theta(s_t, a_t) = r(s_t, a_t) + \gamma \mathrm{E}_{s_{t+1} \sim p}[V_\psi(s_{t+1})]$
  - Train Policy
    - $J_\phi(\pi) = \mathrm{E}_{s_t \sim D}[D_{KL}(\pi_\phi(\cdot\,|s_t)||\frac{\exp(Q_\theta(s_t, \cdot))}{Z_\theta(s_t)})]$
  - Reparametrize trick
    - $a_t = f_\phi(\epsilon_t; s_t)$
    - $J_\phi(\pi) = \mathrm{E}_{s_t \sim D, \epsilon_t \sim N}[D_{KL}(\pi_\phi(\cdot\,|s_t)||\frac{\exp(Q_\theta(s_t, \cdot))}{Z_\theta(s_t)})]$

Probabilistic Graphical Model

Soft Actor-Critic

Algorithm 1 Soft Actor-Critic

Initialize parameter vectors ψ, $\bar{\psi}$, θ, ϕ.
for each iteration **do**
  for each environment step **do**
    $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$
    $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$
  end for
  for each gradient step **do**
    $\psi \leftarrow \psi - \lambda_V \hat{\nabla}_\psi J_V(\psi)$
    $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$
    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$
    $\bar{\psi} \leftarrow \tau\psi + (1 - \tau)\bar{\psi}$
  end for
end for

Probabilistic Graphical Model

Review

- **Reinforcement learning can be viewed as inference in a graphical model**
  - Value function is a backward message
  - Maximize reward and entropy (the bigger the rewards, the less entropy matters)
  - Variational inference to remove optimism
- **Soft Q-learning**
- **Entropy-regularized policy gradient**
- **Soft Actor-Critic**

Probabilistic Graphical Model

Benefits of soft optimality

- Improve exploration and prevent entropy collapse
- Easier to specialize (finetune) policies for more specific tasks
- Principled approach to break ties
- Better robustness (due to wider coverage of states)
- Can reduce to hard optimality as reward magnitude increases
- Good model for modeling human behavior (more on this later)

Inverse Reinforcement Learning

Inverse Reinforcement Learning

- **So far**
  - Manually design reward function to define a task
- **What if we want to learn the reward function from observing an expert, and then use reinforcement learning?**
  - Apply approximate optimality model from last week, but now learn the reward
- **Inverse reinforcement learning**
  - Infer reward function from roll-outs of expert policy

Inverse Reinforcement Learning

Inverse Reinforcement Learning

- **Why should we learn the reward?**
  - Alternative: directly mimic the expert (behavior cloning)
    - simply "ape" the expert's motions/actions
    - doesn't necessarily capture the salient parts of the behavior
    - what if the expert has different capabilities?
  - Can we reason about what the expert is trying to achieve instead?

Inverse Reinforcement Learning

Inverse Optimal Control / Inverse Reinforcement Learning:

- **Infer reward function from demonstrations**
  - Given
    - State & action space
    - Samples from π^*
    - Dynamics model (sometimes)
  - Goal
    - Recover reward function
    - Then use reward to get policy
  - Challenges
    - Underdefined problem
    - Difficult to evaluate a learned reward
    - Demonstrations may not be precisely optimal

Inverse Reinforcement Learning

Inverse Reinforcement Learning

- **Forward Reinforcement Learning**
  - Given
    - States $s \in S$ actions $a \in A$
    - (sometimes) Transitions $p(s'|s,a)$
    - Reward function $r(s,a)$
  - Learn $\pi^*(a|s)$
- **Inverse Reinforcement Learning**
  - Given
    - States $s \in S$ actions $a \in A$
    - (sometimes) transitions $p(s'|s,a)$
    - Samples $\{\tau_i\}$ from $\pi^*(\tau)$
  - Learn $r_\psi(s,a)$
    - Could be linear $r_\psi(s,a) = \sum_i \psi_i f_i(s,a) = \psi^T \boldsymbol{f}(s,a)$
    - Or neural net
  - Then use it to learn $\pi^*(a|s)$

Inverse Reinforcement Learning

Feature matching IRL

- **Linear reward function**
  - $r_\psi(s,a) = \sum_i \psi_i f_i(s,a) = \psi^T \boldsymbol{f}(s,a)$
- **If the feature f is important, we could match their expectations**
  - Let π^{r_ψ} be the optimal policy for r_ψ
  - Pick ψ such that $\mathrm{E}_{\pi^{r_\psi}}[\boldsymbol{f}(s,a)] = \mathrm{E}_{\pi^*}[\boldsymbol{f}(s,a)]$
    - $\mathrm{E}_{\pi^{r_\psi}}[\boldsymbol{f}(s,a)]$ is the state-action marginal under π^{r_ψ}
    - $\mathrm{E}_{\pi^*}[\boldsymbol{f}(s,a)]$ is unknown optimal policy approximate using expert samples
  - Using maximum margin principle
    - $\max_{\psi,m} m$
    - Such that
      - $\psi^T \mathrm{E}_{\pi^*}[\boldsymbol{f}(s,a)] \geq \max_{\pi \in \Pi} \psi^T \mathrm{E}_\pi[\boldsymbol{f}(s,a)] + m$

Inverse Reinforcement Learning

Feature matching IRL & maximum margin

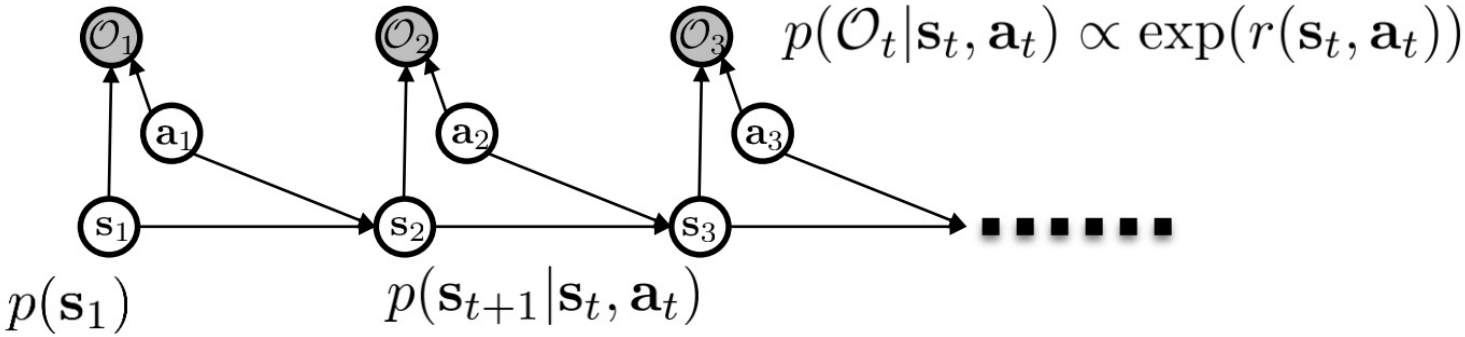
- **Remember the SVM**
  - $\max_{\psi,m} m$ Such that $\psi^T \mathrm{E}_{\pi^*}[\boldsymbol{f}(s,a)] \geq \max_{\pi \in \Pi} \psi^T \mathrm{E}_{\pi}[\boldsymbol{f}(s,a)] + m$
  - $\min_{\psi} \frac{1}{2}||\psi||^2$ such that $\psi^T \mathrm{E}_{\pi^*}[\boldsymbol{f}(s,a)] \geq \max_{\pi \in \Pi} \psi^T \mathrm{E}_{\pi}[\boldsymbol{f}(s,a)] + D(\pi, \pi^*)$
  - Where $D(\pi, \pi^*)$ is difference in feature expectations
  - Problems
    - Maximizing the margin is a bit arbitrary
    - No clear model of expert suboptimality (can add slack variables…)
    - Messy constrained optimization problem – not great for deep learning

Inverse Reinforcement Learning

Probabilistic Graphical Model

- **A probabilistic graphical model of decision making**

$p(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}|\mathcal{O}_{1:T})$



- $p(s_{1:T}, a_{1:T}) = ??$ $\quad p(O_t|s_t, a_t) \propto \exp(r(s_t, a_t))$
- $p(\tau|O_{1:T}) = \frac{p(\tau, O_{1:T})}{p(O_{1:T})} \propto p(\tau) \prod_t \exp\big(r(s_t, a_t)\big) = p(\tau)\exp(\sum_t r(s_t, a_t))$

Inverse Reinforcement Learning

Probabilistic Graphical Model

- **Learning the optimality variable**
  - $p(O_t|s_t, a_t, \psi) \propto \exp(r_\psi(s_t, a_t))$
  - $p(\tau|O_{1:T}, \psi) = \frac{p(\tau, O_{1:T})}{p(O_{1:T})} \propto p(\tau) \prod_t \exp\left(r_\psi(s_t, a_t)\right) \sim \frac{1}{Z} p(\tau) \exp\left(r_\psi(\tau)\right)$
  - Given
    - Samples $\{\tau_i\}$ from $\pi^*(\tau)$
  - We can perform maximum likelihood learning
    - $\max_\psi \frac{1}{N} \sum_{i=1}^N \log p(\tau_i|O_{1:T}, \psi) = \max_\psi \frac{1}{N} \sum_{i=1}^N r_\psi(\tau_i) - \log Z$
    - Where $Z = \int p(\tau) \exp\left(r_\psi(\tau)\right) d\tau$ is the partition function

Inverse Reinforcement Learning

The IRL partition function

- **The IRL partition function**
  - $\max_{\psi} \frac{1}{N} \sum_{i=1}^{N} \log p(\tau_i | O_{1:T}, \psi) = \max_{\psi} \frac{1}{N} \sum_{i=1}^{N} r_\psi(\tau_i) - \log Z$
  - $Z = \int p(\tau) \exp\left(r_\psi(\tau)\right) d\tau$
  - Gradient
    - $\nabla_\psi L = \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{Z} \int p(\tau) \exp\left(r_\psi(\tau)\right) \nabla_\psi r_\psi(\tau) d\tau$
    - Remember $p(\tau | O_{1:T}, \psi) \sim \frac{1}{Z} p(\tau) \exp(\sum_t r(s_t, a_t)) = \frac{1}{Z} p(\tau) \exp\left(r_\psi(\tau)\right)$
    - Then we have
      - $\nabla_\psi L = \mathrm{E}_{\tau \sim \pi^*(\tau)}[\nabla_\psi r_\psi(\tau_i)] - \mathrm{E}_{\tau \sim p(\tau | O_{1:T}, \psi)}[\nabla_\psi r_\psi(\tau)]$
      - $\mathrm{E}_{\tau \sim \pi^*(\tau)}[\nabla_\psi r_\psi(\tau_i)]$
        - estimate from expert samples
      - $\mathrm{E}_{\tau \sim p(\tau | O_{1:T}, \psi)}[\nabla_\psi r_\psi(\tau)]$
        - soft optimal policy under current reward

Inverse Reinforcement Learning

The IRL partition function

- **Estimating the expectation**
  - $\mathrm{E}_{\tau \sim p(\tau|O_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)] = \mathrm{E}_{\tau \sim p(\tau|O_{1:T},\psi)}[\nabla_\psi \sum_{t=1}^{T} r_\psi(s_t,a_t)]$
  - $= \sum_{i=1}^{T} \mathrm{E}_{(s_t,a_t)\sim p(s_t,a_t|O_{1:T},\psi)}[\nabla_\psi r_\psi(s_t,a_t)]$
  - Where
    - $p(s_t,a_t|O_{1:T},\psi) = p(a_t|s_t,O_{1:T},\psi)p(s_t|O_{1:T},\psi)$
  - Remember that
    - $p(a_t|s_t,O_{1:T},\psi) = \frac{\beta(s_t,a_t)}{\beta(s_t)}$
    - $p(s_t\,|O_{1:T},\psi) \propto \beta_t(s_t)\alpha_t(s_t)$
  - Then we have
    - $p(s_t,a_t|O_{1:T},\psi) = p(a_t|s_t,O_{1:T},\psi)p(s_t\,|O_{1:T},\psi)$
    - $\propto \beta(s_t,a_t)\alpha_t(s_t)$
  - Let $\mu(s_t,a_t) \propto \beta(s_t,a_t)\alpha_t(s_t)$
    - $\mathrm{E}_{\tau \sim p(\tau|O_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)] = \sum_{t=1}^{T} \int\int \mu(s_t,a_t)\nabla_\psi r_\psi(s_t,a_t) ds_t da_t$
    - $= \sum_{t=1}^{T} \vec{\mu}_t^T \nabla_\psi \vec{r}_\psi$
    - Where $\vec{\mu}_t^T$ is state-action visit probability for each (s_t,a_t)

Inverse Reinforcement Learning

The MaxEnt IRL algorithm

- **The MaxEnt IRL algorithm**
  - Repeat
    - Given ψ, compute backward message $\beta(s_t, a_t)$
    - Given ψ, compute forward message $\alpha_t(s_t)$
    - Compute $\mu(s_t, a_t) \propto \beta(s_t, a_t)\alpha_t(s_t)$
    - Evaluate $\nabla_\psi L = \frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{T}\nabla_\psi r_\psi(s_{i,t}, a_{i,t}) - \sum_{t=1}^{T}\int\int \mu(s_t, a_t)\nabla_\psi r_\psi(s_t, a_t)ds_t da_t$
    - $\psi \leftarrow \psi + \eta\nabla_\psi L$
- **Why it is max entropy?**
  - For $r_\psi(s, a) = \psi^T \boldsymbol{f}(s, a)$
  - It optimizes $\max_\psi H(\pi^{r_\psi})$ such that $\mathrm{E}_{\pi^{r_\psi}}[\boldsymbol{f}(s, a)] = \mathrm{E}_{\pi^*}[\boldsymbol{f}(s, a)]$

Inverse Reinforcement Learning

Deep Inverse Reinforcement Learning

- **What about larger RL problems?**
  - MaxEnt IRL: probabilistic framework for learning reward functions
  - Computing gradient requires enumerating state-action visitations for all states and actions
    - Only really viable for small, discrete state and action spaces
    - Amounts to a dynamic programming algorithm (exact forward backward inference)
  - For deep IRL, we want two things:
    - Large and continuous state and action spaces
    - Effective learning under unknown dynamics

Inverse Reinforcement Learning

Unknown Dynamics & Large State/action Spaces

- **Assume we don't know the dynamics, but we can sample, like in standard RL**
  - $\nabla_\psi L = \mathrm{E}_{\tau\sim\pi^*(\tau)}[\nabla_\psi r_\psi(\tau_i)] - \mathrm{E}_{\tau\sim p(\tau|O_{1:T},\psi)}[\nabla_\psi r_\psi(\tau)]$
  - Learn $p(a_t|s_t, O_{1:T}, \psi)$ using any max-ent RL algorithm
    - $J(\theta) = \sum_t \mathrm{E}_{\pi(s_t,a_t)}[r_\psi(s_t,a_t)] + \mathrm{E}_{\pi(s_t)}[H(\pi(a|s_t))]$
  - Then run this policy to sample $\{\tau_j\}$
  - Then
    - $\nabla_\psi L \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$
    - $\frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i)$ sum over expert samples
    - $\frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$ sum over policy samples

Inverse Reinforcement Learning

Unknown Dynamics & Large State/action Spaces

- **More efficient sample-based updates**
  - $\nabla_\psi L \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{M}\sum_{j=1}^{M} \nabla_\psi r_\psi(\tau_j)$
  - Use lazy policy optimization
- **Problems**
  - Wrong distribution, estimator is now biased
- **Solution**
  - Use importance sampling
    - $\nabla_\psi L \approx \frac{1}{N}\sum_{i=1}^{N} \nabla_\psi r_\psi(\tau_i) - \frac{1}{\sum_j w_j}\sum_{j=1}^{M} w_j \nabla_\psi r_\psi(\tau_j)$
    - Where $w_j = \frac{p(\tau_j | O_{1:T}, \psi)}{\pi(\tau_j)} = \frac{p(\tau_j)\exp(r_\psi(\tau_j))}{\pi(\tau_j)}$
    - $= \frac{p(s_1)\prod_t p(s_{t+1}|s_t,a_t)\exp(r_\psi(s_t,a_t))}{p(s_1)\prod_t p(s_{t+1}|s_t,a_t)\pi(a_t|s_t)} = \frac{\exp(\sum_t r_\psi(s_t,a_t))}{\prod_t \pi(a_t|s_t)}$

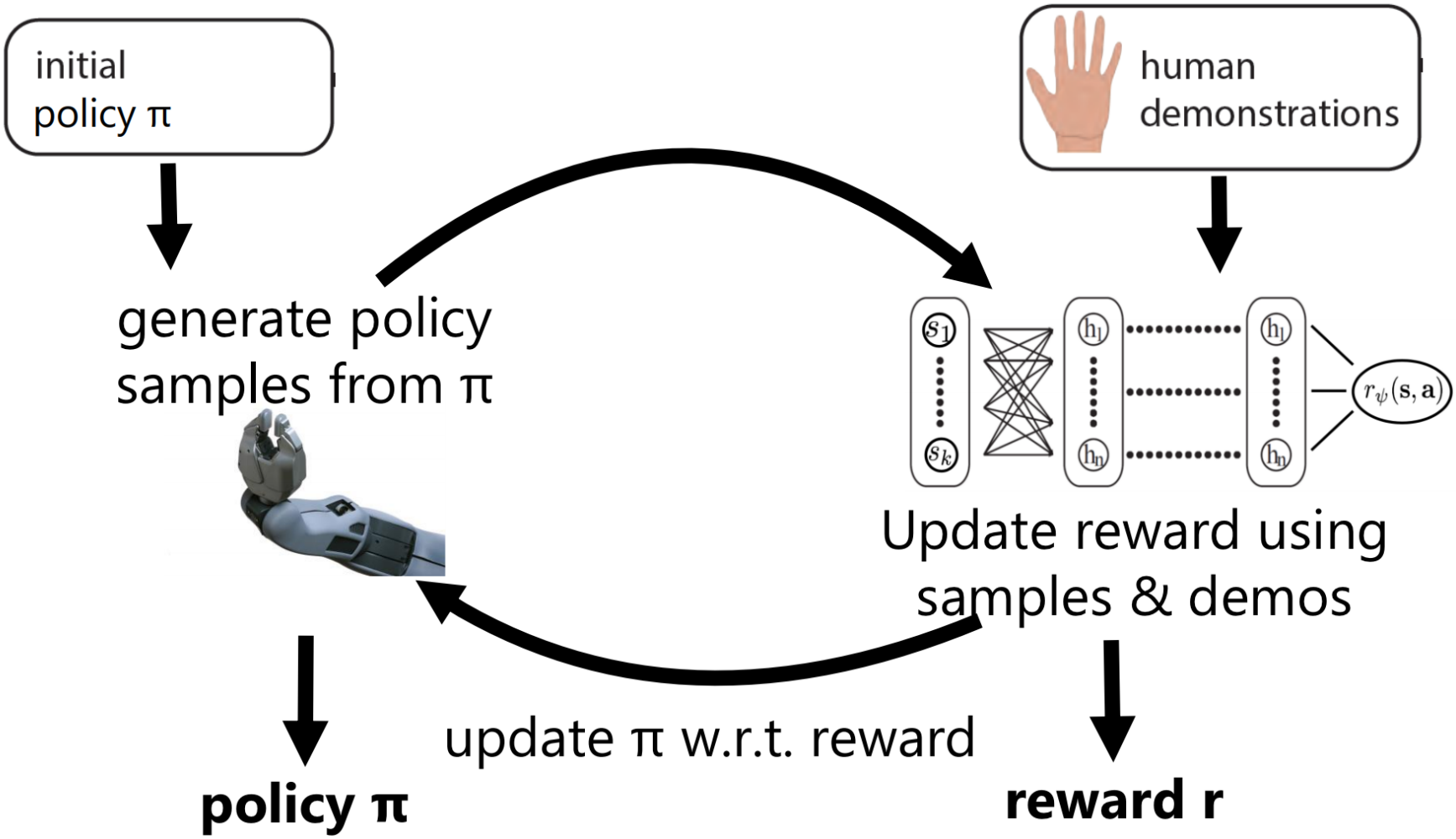
Inverse Reinforcement Learning

Importance Sampling

- **Importance sampling**
  - $w_j = \frac{\exp(\sum_t r_\psi(s_t, a_t))}{\prod_t \pi(a_t|s_t)}$
  - Which sampling distribution $\pi(\tau)$ is best?
    - Optimal IS distribution $q(x)$ for $\mathrm{E}_{p(x)}[f(x)]$ is $q(x) \propto |f(x)|p(x)$
    - Then, optimal π is $\pi(\tau) \propto \exp(r_\psi(\tau))$
      - Max-ent optimal policy for r_ψ
  - Each policy update w.r.t r_ψ bring closer to the optimal distribution

Inverse Reinforcement Learning

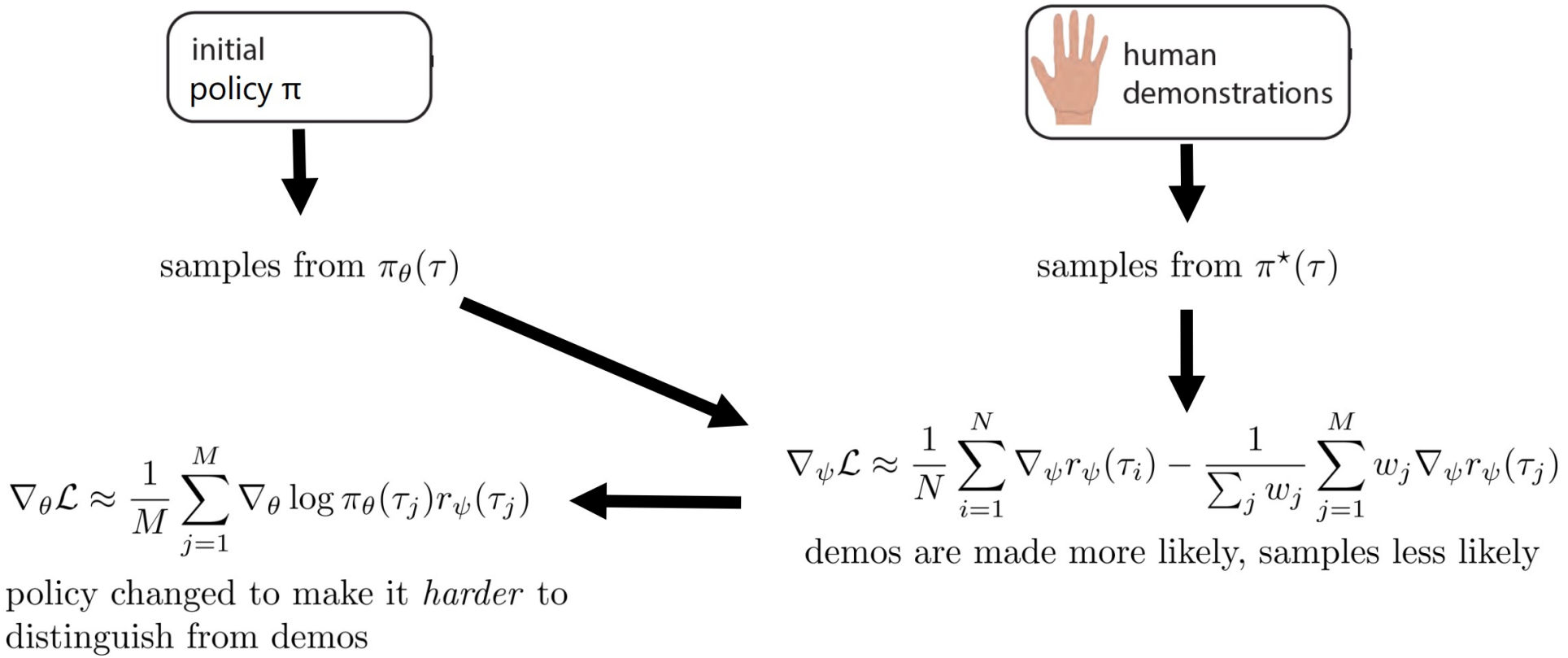
Guided Cost Learning



Inverse Reinforcement Learning

Inverse Reinforcement Learning

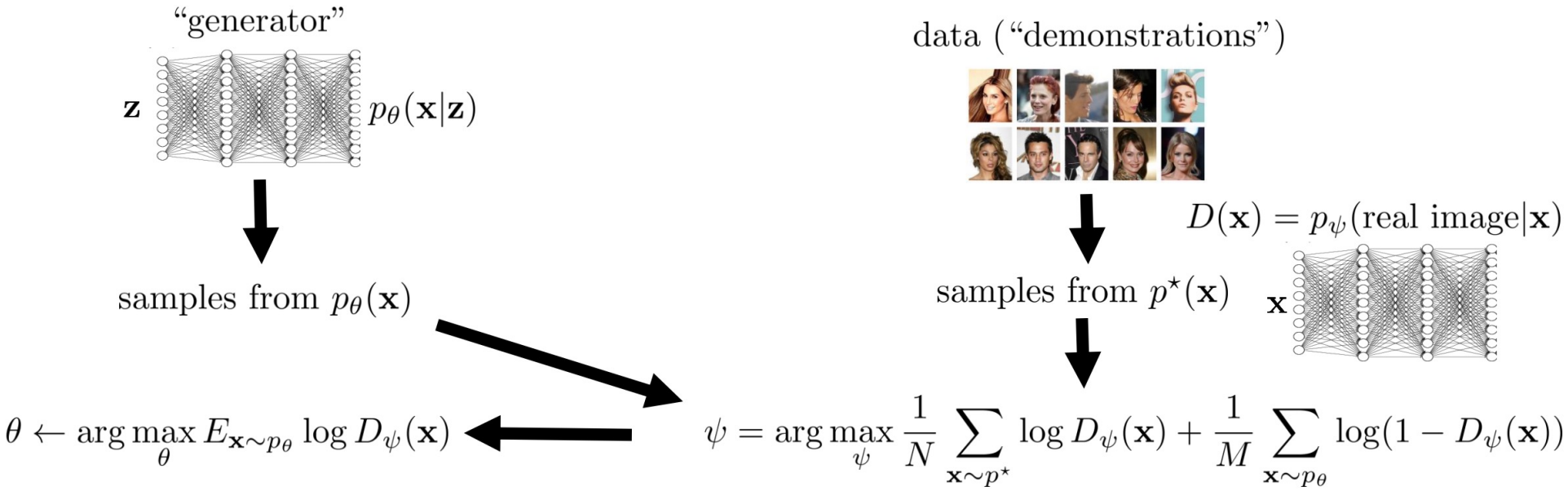
- **Looks like a adversarial game**



Inverse Reinforcement Learning

Generative Adversarial Networks

- **Generative Adversarial Networks**



	trajectory τ	⟷ **sample x**	
Inverse RL	**policy π~q(τ)**	⟷ **generator G**	GANs
	reward r	⟶ **discriminator D**	
	data distribution p		

Inverse Reinforcement Learning

Inverse RL as a GAN

- **GAN discriminator**
  - $L_{\text{Discriminator}}(\psi) = \mathrm{E}_{x\sim p^*}\left[-\log D_\psi(x)\right] + \mathrm{E}_{x\sim p_\theta}\left[-\log(1 - D_\psi(x))\right]$
  - $\psi = \underset{\psi}{\operatorname{argmax}} \frac{1}{N}\sum_{x\sim p^*} \log D_\psi(x) + \frac{1}{M}\sum_{x\sim p_\theta} \log(1 - D_\psi(x))$
  - The best discriminator $D(x) = p_\psi(\text{real image}|x)$ is
    - $D^*(x) = \frac{p^*(x)}{p_\theta(x)+p^*(x)}$
- **IRL**
  - Optimal policy approaches $\pi_\theta(\tau) \propto p(\tau)\exp(r_\psi(\tau))$
  - Then let $p^*(x) \propto p(\tau)\exp(r_\psi(\tau))$
    - $D_\psi(\tau) = \frac{p(\tau)\frac{1}{Z}\exp(r_\psi(\tau))}{p_\theta(\tau)+p(\tau)\frac{1}{Z}\exp(r_\psi(\tau))} = \frac{p(\tau)\frac{1}{Z}\exp(r_\psi(\tau))}{p(\tau)\prod_t \pi_\theta(a_t|s_t)+p(\tau)\frac{1}{Z}\exp(r_\psi(\tau))}$
    - $= \frac{\frac{1}{Z}\exp(r_\psi(\tau))}{\prod_t \pi_\theta(a_t|s_t)+\frac{1}{Z}\exp(r_\psi(\tau))}$
    - $\psi = \underset{\psi}{\operatorname{argmax}} \mathrm{E}_{\tau\sim p^*}\left[\log D_\psi(\tau)\right] + \mathrm{E}_{\tau\sim \pi_\theta}\left[\log(1 - D_\psi(\tau))\right]$

Inverse Reinforcement Learning

Inverse RL as a GAN

- **IRL**
  - Discriminator
    - $L_{\text{Discriminator}}(\psi) = \mathrm{E}_{\tau\sim p^*}\left[-\log D_\psi(\tau)\right] + \mathrm{E}_{\tau\sim\pi_\theta}\left[-\log(1 - D_\psi(\tau))\right]$
    - $\psi = \underset{\psi}{\operatorname{argmax}}\, \mathrm{E}_{\tau\sim p^*}\left[\log D_\psi(\tau)\right] + \mathrm{E}_{\tau\sim\pi_\theta}\left[\log(1 - D_\psi(\tau))\right]$
    - Can be proved that optimize Z w.r.t. same objective as ψ
  - Generator
    - $L_{\text{Generator}}(\theta) = \mathrm{E}_{\tau\sim\pi_\theta}\left[\log\left(1 - D_\psi(\tau)\right) - \log D_\psi(\tau)\right]$
    - $= \mathrm{E}_{\tau\sim\pi_\theta}\left[\log p_\theta(\tau) + \log Z - r_\psi(\tau)\right]$

Inverse Reinforcement Learning

Inverse RL as a GAN

generator/policy

$\pi_\theta(\tau)$

samples from $\pi_\theta(\tau)$

data/demonstrations

samples from $p^\star(\tau)$

$$\psi \leftarrow \arg\max_\psi E_{\tau\sim p^\star}[\log D_\psi(\tau)] + E_{\tau\sim\pi_\theta}[\log(1 - D_\psi(\tau))]$$

$$D_\psi(\tau) = \frac{\frac{1}{Z}\exp(r(\tau))}{\prod_t \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) + \frac{1}{Z}\exp(r(\tau))}$$

$$\nabla_\theta \mathcal{L} \approx \frac{1}{M}\sum_{j=1}^{M} \nabla_\theta \log \pi_\theta(\tau_j) r_\psi(\tau_j)$$

policy changed to make it *harder* to distinguish from demos

Inverse Reinforcement Learning

Inverse RL as a GAN

- **Can we just use a regular discriminator?**
  - $\psi = \underset{\psi}{\text{argmax}}\, \mathrm{E}_{\tau\sim p^*}\left[\log D_\psi(\tau)\right] + \mathrm{E}_{\tau\sim\pi_\theta}[\log(1 - D_\psi(\tau))]$
  - $D_\psi(\tau) =$ standard binary neural net classifier
  - $\nabla_\theta L \approx \frac{1}{M}\sum_{j=1}^{M} \nabla_\theta \log \pi_\theta(\tau_j) \log D_\psi(\tau_j)$
  - Often simpler to set up optimization, fewer moving parts
  - Discriminator knows nothing at convergence
  - Generally cannot reoptimize the "reward"

Inverse Reinforcement Learning

Summary

- **IRL: infer unknown reward from expert demonstrations**
- **MaxEnt IRL: infer reward by learning under the control-as-inference framework**
- **MaxEnt IRL with dynamic programming: simple and efficient, but requires small state space and known dynamics**
- **Sampling-based MaxEnt IRL: generate samples to estimate the partition function**
  - Guided cost learning algorithm
  - Connection to generative adversarial networks
  - Generative adversarial imitation learning (not IRL per se, but similar)

Inverse Reinforcement Learning

Suggested Readings

- Manuel Watter, Embed to Control: a locally linear latent dynamics models for control from raw images
- M Zhang, SOLAR: deep structured latent representation for model-based reinforcement learning
- Todorov. (2006). Linearly solvable Markov decision problems
- Todorov. (2008). General duality between optimal control and estimation
- Kappen. (2009). Optimal control as a graphical model inference problem
- Ziebart. (2010). Modeling interaction via the principle of maximal causal entropy
- Rawlik, Toussaint, Vijaykumar. (2013). On stochastic optimal control and reinforcement learning by approximate inference
- Haarnoja*, Tang*, Abbeel, L. (2017). Reinforcement learning with deep energy based models

Inverse Reinforcement Learning

Suggested Readings

- Nachum, Norouzi, Xu, Schuurmans. (2017). Bridging the gap between value and policy based reinforcement learning
- Schulman, Abbeel, Chen. (2017). Equivalence between policy gradients and soft Q-learning
- Haarnoja, Zhou, Abbeel, L. (2018). Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor
- Christodoulou, Petros. "Soft Actor-Critic for Discrete Action Settings."
- Sallans & Hinton. Using Free Energies to Represent Q-values in a Multiagent Reinforcement Learning Task
- O'Donoghue et al. Combining Policy Gradient and Q-Learning
- Levine. Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review

Inverse Reinforcement Learning

Suggested Readings

- Abbeel & Ng: Apprenticeship learning via inverse reinforcement learning
- Ratliff et al: Maximum margin planning
- Ziebart et al. : Maximum Entropy Inverse Reinforcement Learning
- Wulfmeier et al. Maximum Entropy Deep Inverse Reinforcement Learning
- Wulfmeier et al. Watch This: Scalable Cost-Function Learning for Path Planning in Urban Environments
- Finn et al. Guided Cost Learning
- Finn, Christiano, et al. A Connection Between Generative Adversarial Networks, Inverse Reinforcement Learning, and Energy-Based Models
- Fu et al. Learning Robust Rewards with Adversarial Inverse Reinforcement Learning
- Ho & Ermon. Generative adversarial imitation learning