Carnegie Mellon
School of Computer Science

Deep Reinforcement Learning and Control

Determinist PG, Pathwise derivatives

Spring 2020, CMU 10-403

Katerina Fragkiadaki

Computing Gradients of Expectations

Policy objective:

$$\max_{\theta} . \ \mathbb{E}_{\tau \sim P_\theta(\tau)} \left[R(\tau) \right]$$

Likelihood ratio gradient estimator:

$$\mathbb{E}_{\tau \sim P_\theta(\tau)} \left[\nabla_\theta \log P_\theta(\tau) R(\tau) \right]$$

$$\mathbb{E}_{s \sim d^0(s),\ a \sim \pi_\theta(a|s)} \nabla_\theta \log \pi_\theta(a \mid s) \left[Q(s, a, \mathbf{w}_1) - V(s, \mathbf{w}_2) \right]$$

- Do we have access to the reward function $R(\tau)$?
- Do we have access to the analytic gradients of rewards $R(\tau)$ w.r.t. actions a?
- For continuous actions a, do we have access to the analytic gradients of Q(s,a,w) or Q(s,a,w_1)-V(s,w_2) w.r.t. actions a?
- Have we used the later anywhere?

What if we have a deterministic policy?

Q: does this expectation depend on theta?

$$a = \pi_\theta(s)$$

$$\max_\theta . \ \mathbb{E} \sum_{t=1}^{T} R(s_t, a_t)$$

$$a = \pi_\theta(s)$$

$$\max_\theta . \ \mathbb{E} \sum_{t=1}^{T} Q(s_t, a_t)$$

Qs:

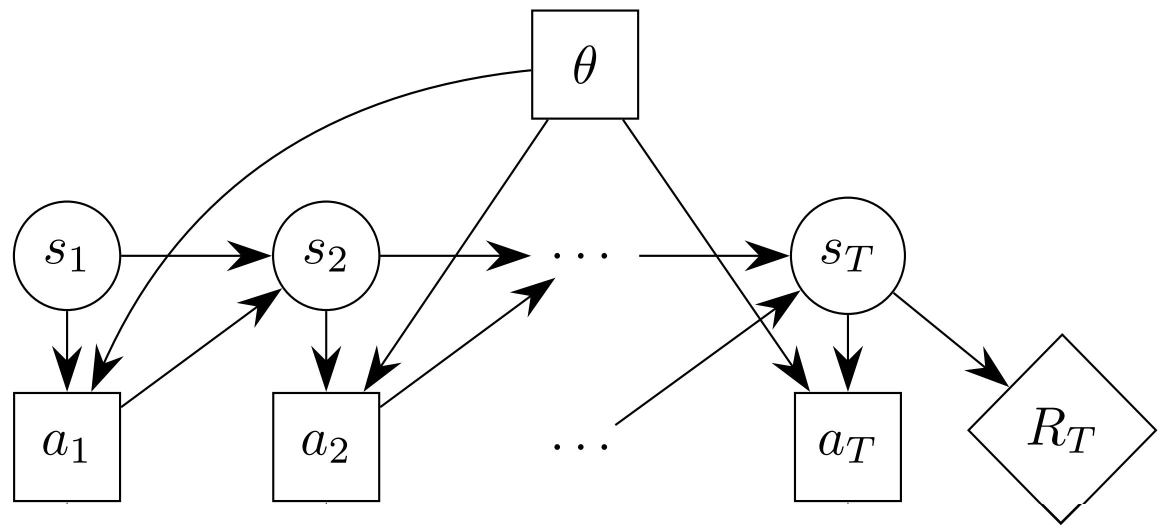
- Can we backpropagate through R?

Qs:

- Can we backpropagate through Q?

$$\mathbb{E} \sum_t \frac{dQ(s_t, a_t)}{d\theta} = \mathbb{E} \sum_t \frac{dQ(s_t, a_t)}{da_t} \frac{da_t}{d\theta}$$

Deep Deterministic Policy Gradients

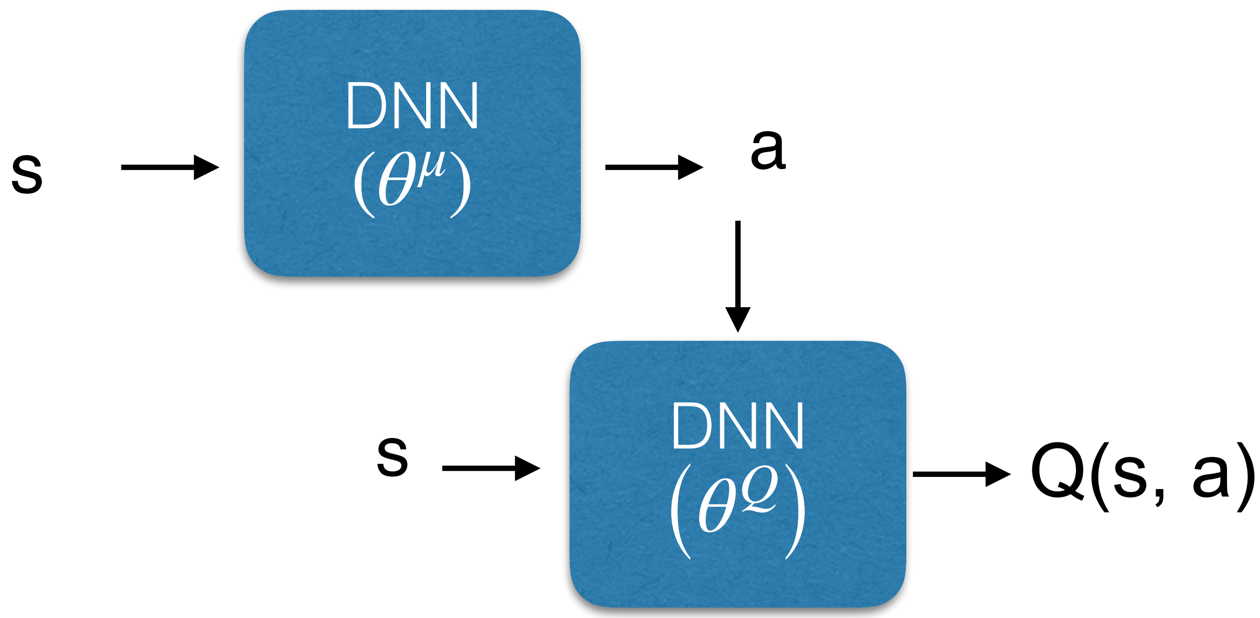


$$a = \pi_\theta(s)$$

$$\mathbb{E} \sum_t \frac{dQ(s_t, a_t)}{d\theta} = \mathbb{E} \sum_{t=1}^{T} \frac{dQ(s_t, a_t)}{da_t} \frac{da_t}{d\theta}$$

Continuous control with deep reinforcement learning, *Lilicrap et al. 2016*

Deep Deterministic Policy Gradients



We are following a stochastic behavior policy to collect data.
DDPG :Deep Q learning for continuous actions

Continuous control with deep reinforcement learning, *Lilicrap et al. 2016*

Deep Deterministic Policy Gradients

Algorithm 1 DDPG algorithm

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state s_1
  for t = 1, T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
    Execute action a_t and observe reward r_t and observe new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in R
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s_i}$$

    Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  end for
end for

Computing Gradients of Expectations

When the variable w.r.t. which we are differentiating appears in the distribution:

$$\nabla_\theta \mathbb{E}_{x \sim P_\theta(x)} f(x) = \mathbb{E}_{x \sim P_\theta(x)} \nabla_\theta \log P_\theta(x) f(x)$$

likelihood ratio gradient estimator

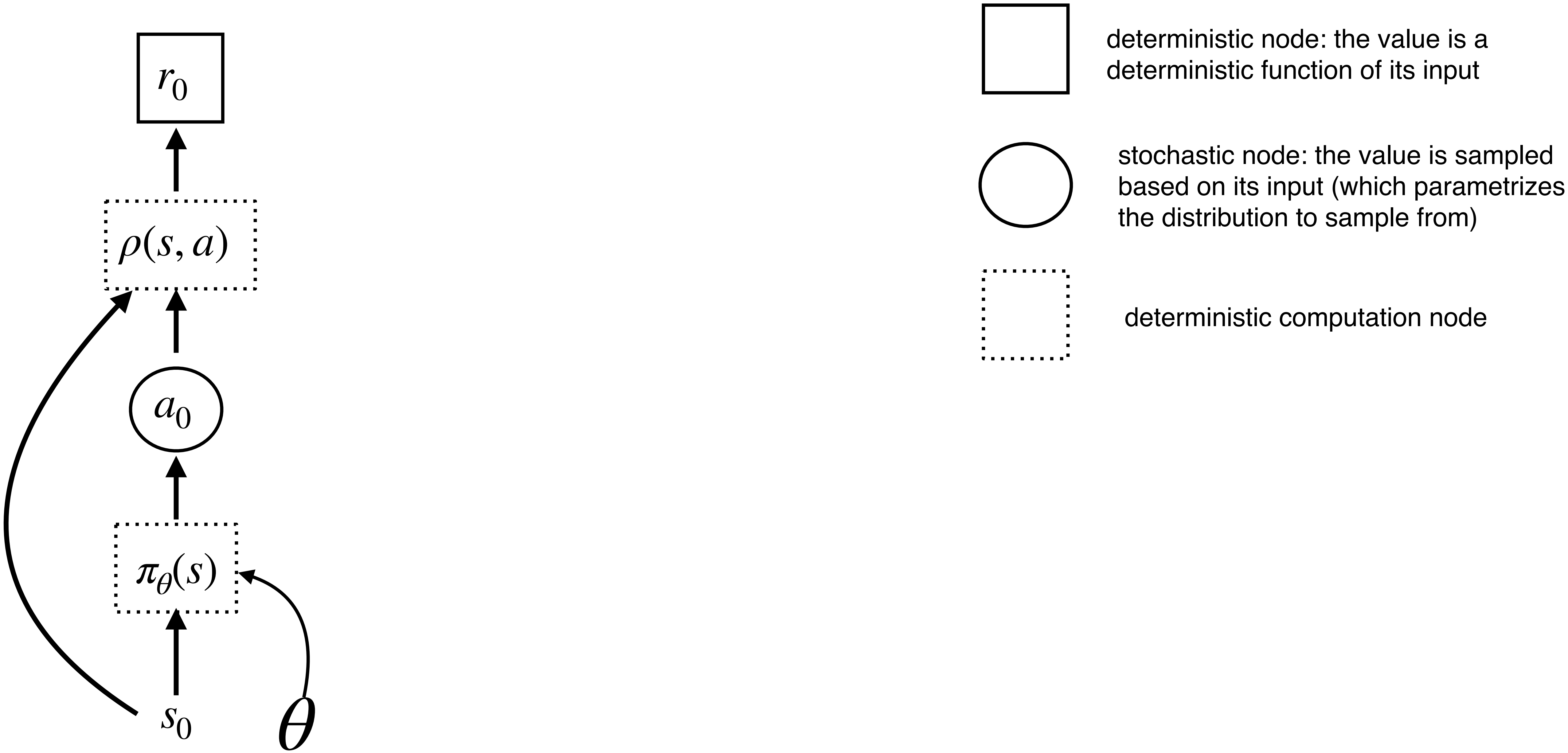
When the variable w.r.t. which we are differentiating appears inside the expectation:

$$\nabla_\theta \mathbb{E} f(x(\theta)) = \mathbb{E}_{x \sim P(x)} \nabla_\theta f(x(\theta)) = \mathbb{E}_{x \sim P(x)} \frac{df(x(\theta))}{dx} \frac{dx}{d\theta}$$

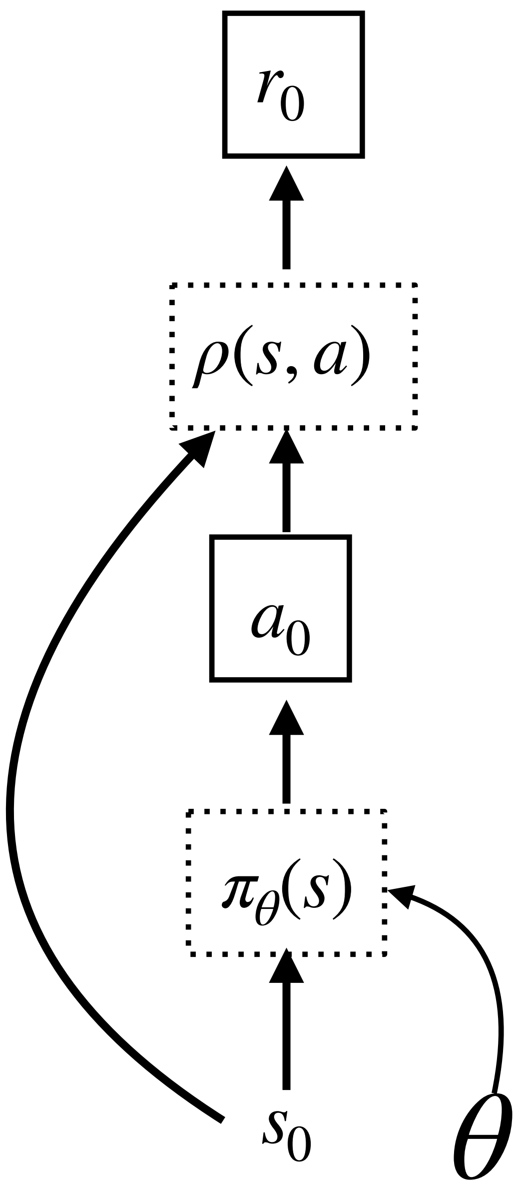
Re-parametrization trick: For some distributions $P_\theta(x)$ we can switch from one gradient estimator to the other.

Q: From which to which? Why would we want to do so?

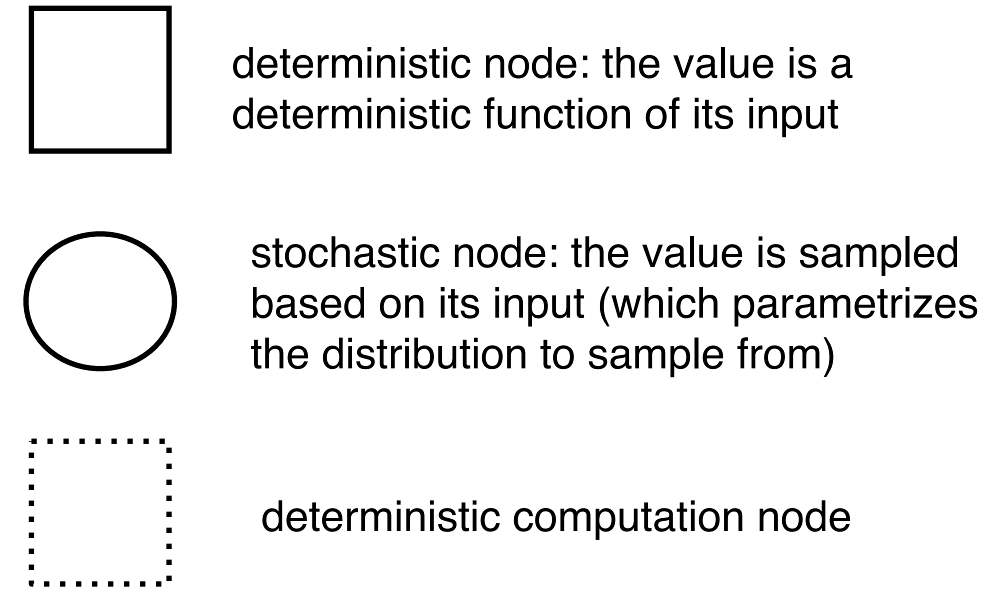
Imagine we knew the reward function $\rho(s, a)$



Deterministic policy



$a = \pi_\theta(s)$



I want to learn θ to maximize the average reward obtained.

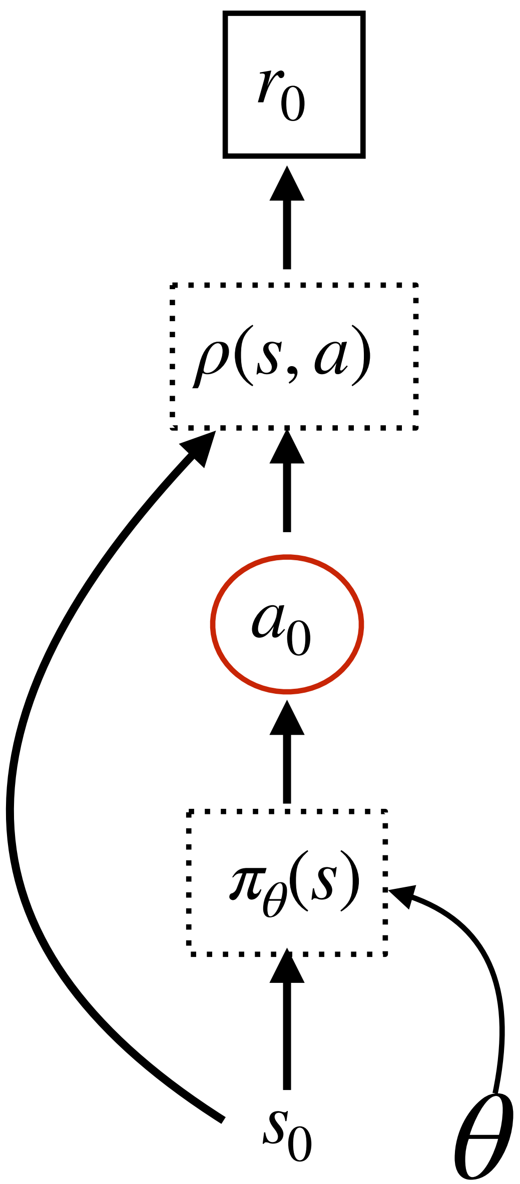
$$\max_{\theta} . \quad \rho(s_0, a)$$

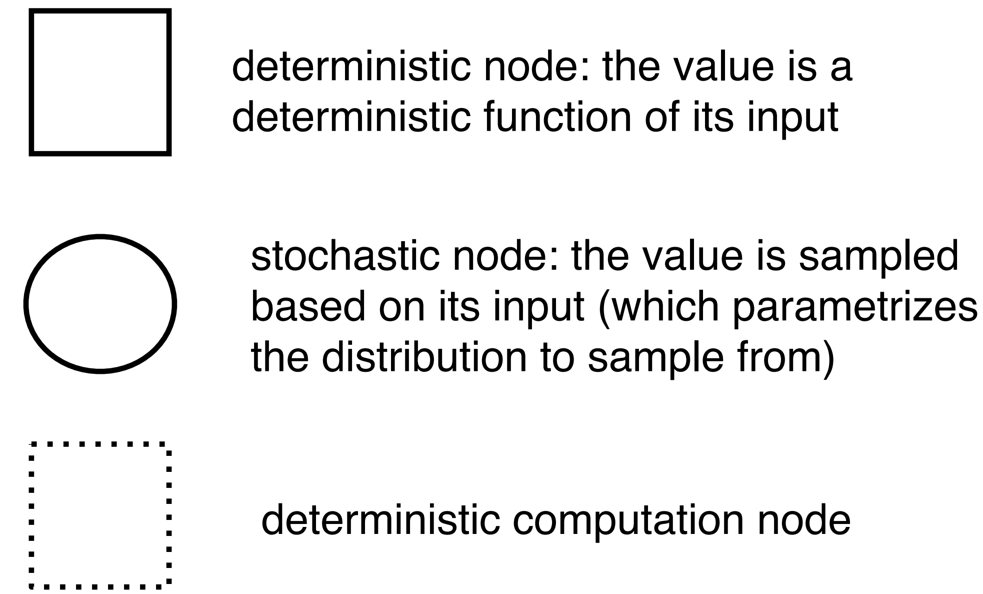
I can compute the gradient with the chain rule.

$$\nabla_\theta \rho(s, a) = \frac{d\rho}{da}\frac{da}{d\theta}$$

Derivative of the *known* reward function w.r.t. the action

Stochastic policy



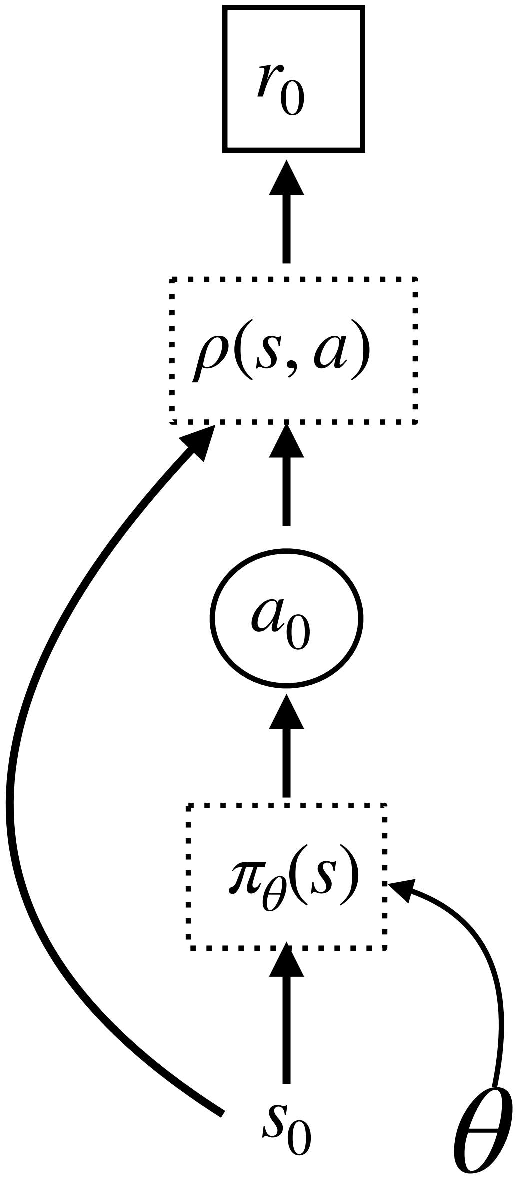


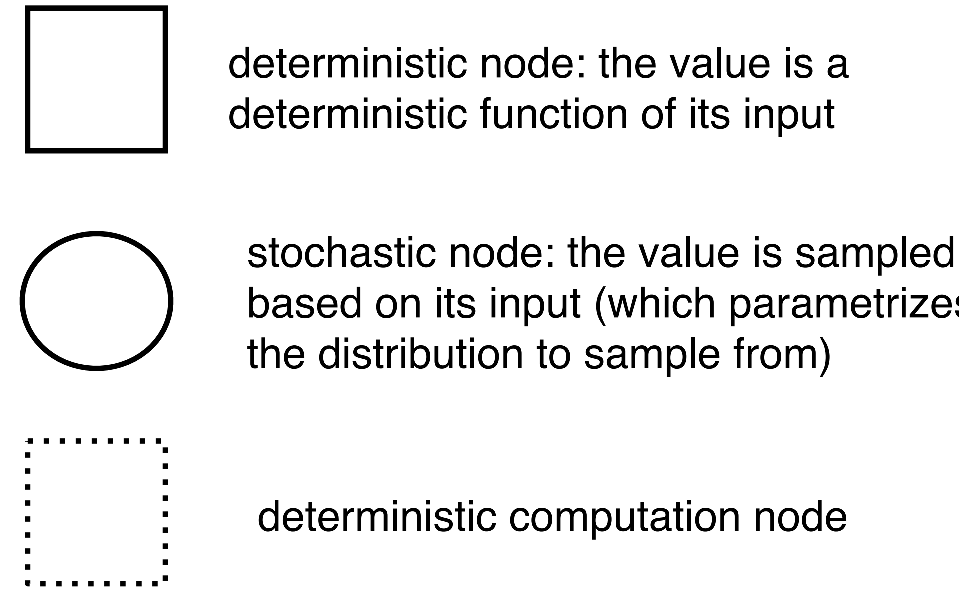
I want to learn θ to maximize the average reward obtained.

$$\max_{\theta} . \quad \mathbb{E}_a \rho(s_0, a)$$

$$\nabla_\theta \mathbb{E}_a \rho(s_0, a)$$

Stochastic policy





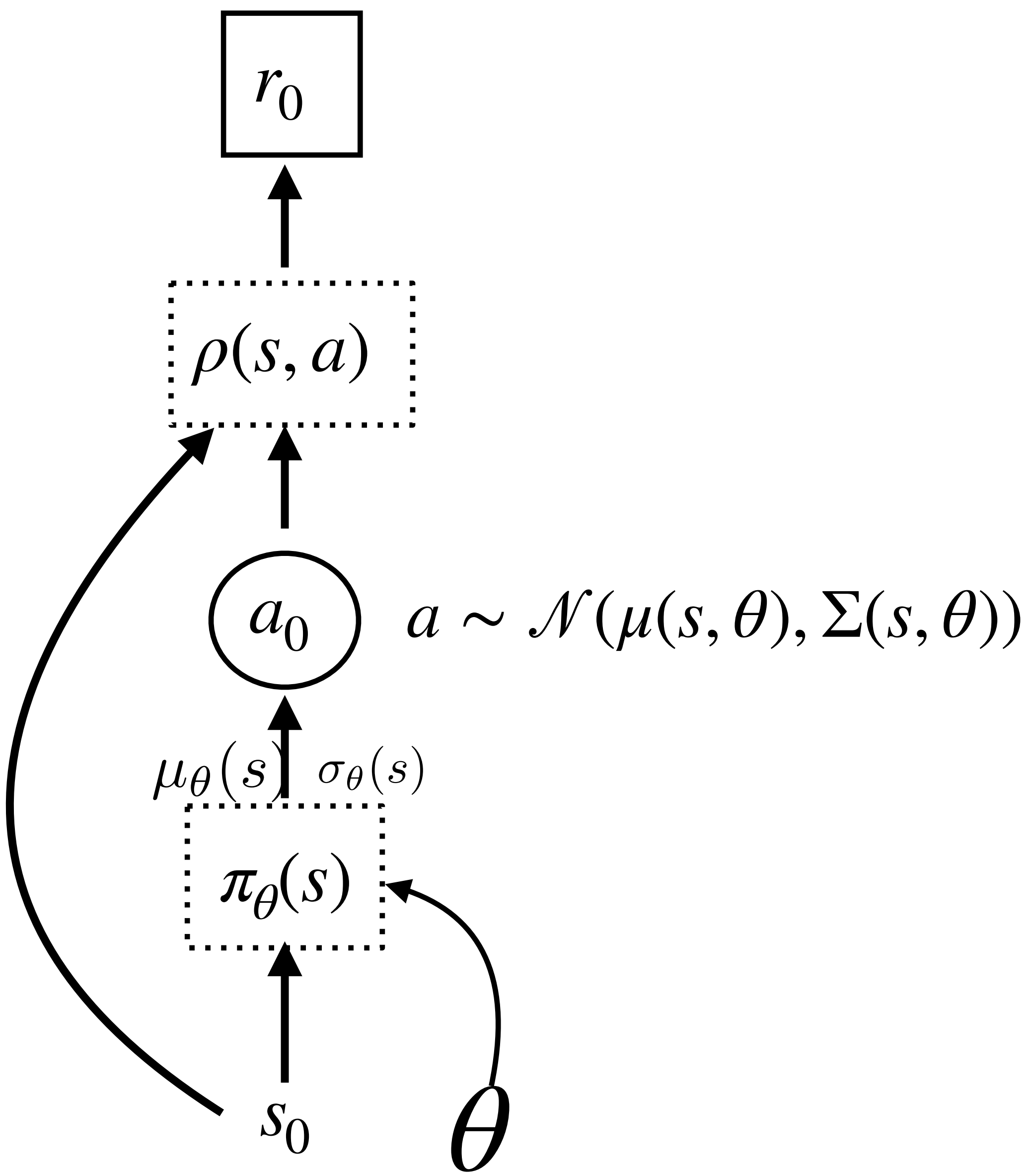
I want to learn θ to maximize the average reward obtained.

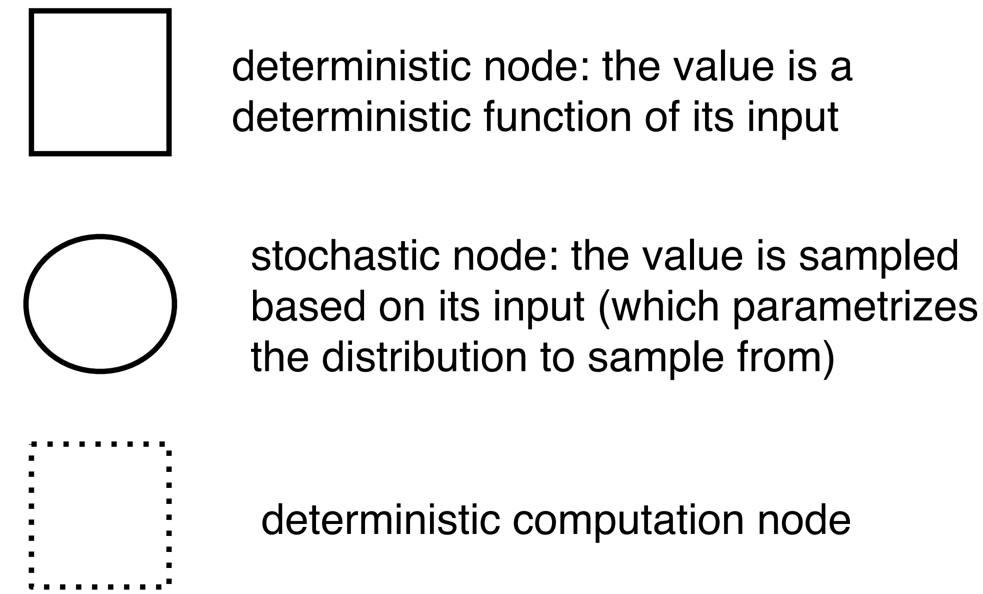
$$\max_{\theta} . \quad \mathbb{E}_a \rho(s_0, a)$$

Likelihood ratio estimator, works for both continuous and discrete actions

$$\mathbb{E}_a \nabla_\theta \log \pi_\theta(s) \rho(s_0, a)$$

Example: Gaussian policy





I want to learn θ to maximize the average reward obtained.

$$\max_{\theta}. \quad \mathbb{E}_a \rho(s_0, a)$$

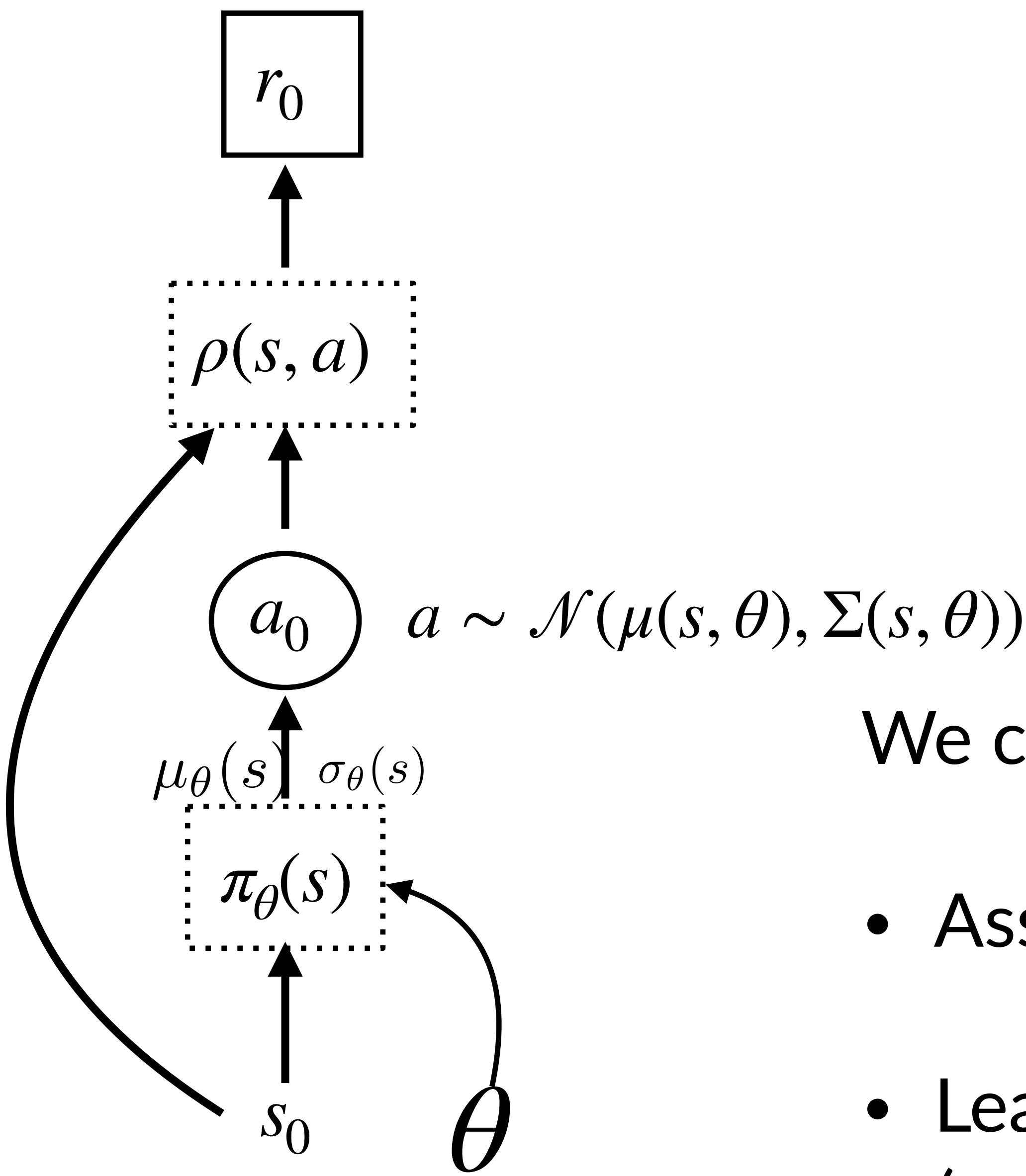
Likelihood ratio estimator, works for both continuous and discrete actions

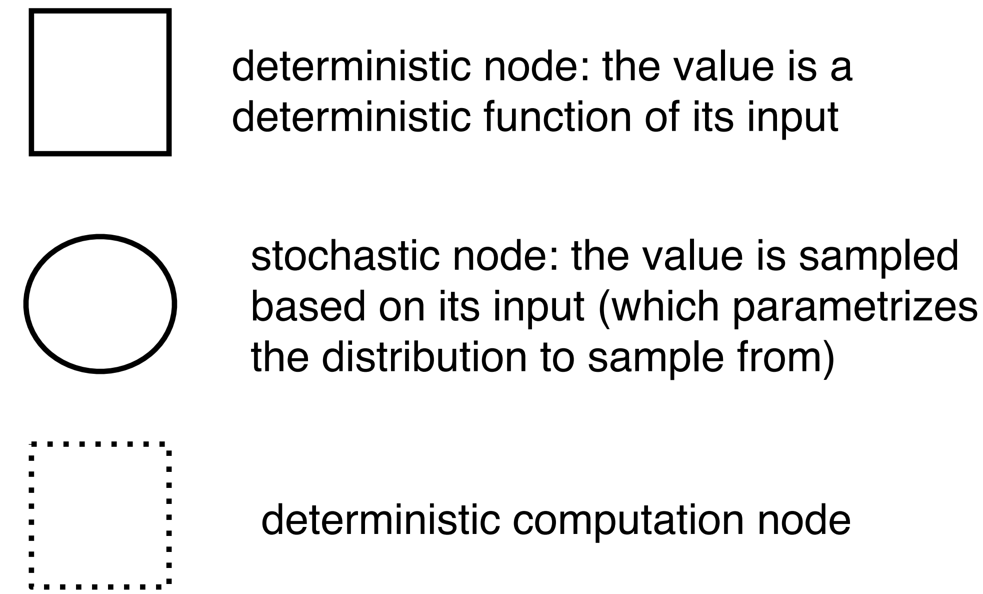
$$\mathbb{E}_a \nabla_\theta \log \pi_\theta(s) \rho(s_0, a)$$

If σ^2 is constant:

$$\nabla_\theta \log \pi_\theta(s, a) = \frac{(a - \mu(s;\theta))\frac{\partial \mu(s;\theta)}{\partial \theta}}{\sigma^2}$$

Example: Gaussian policy

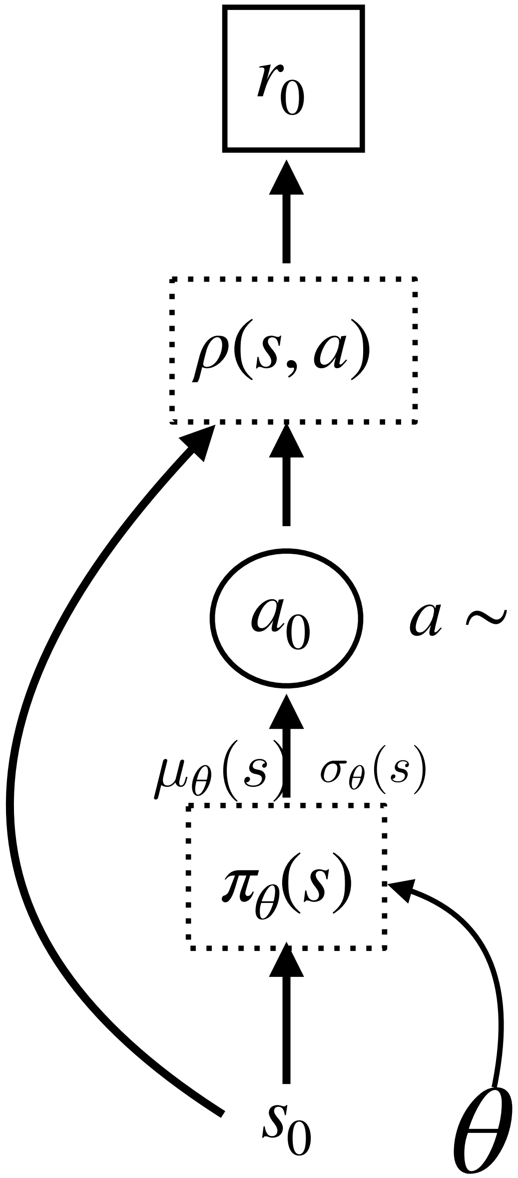




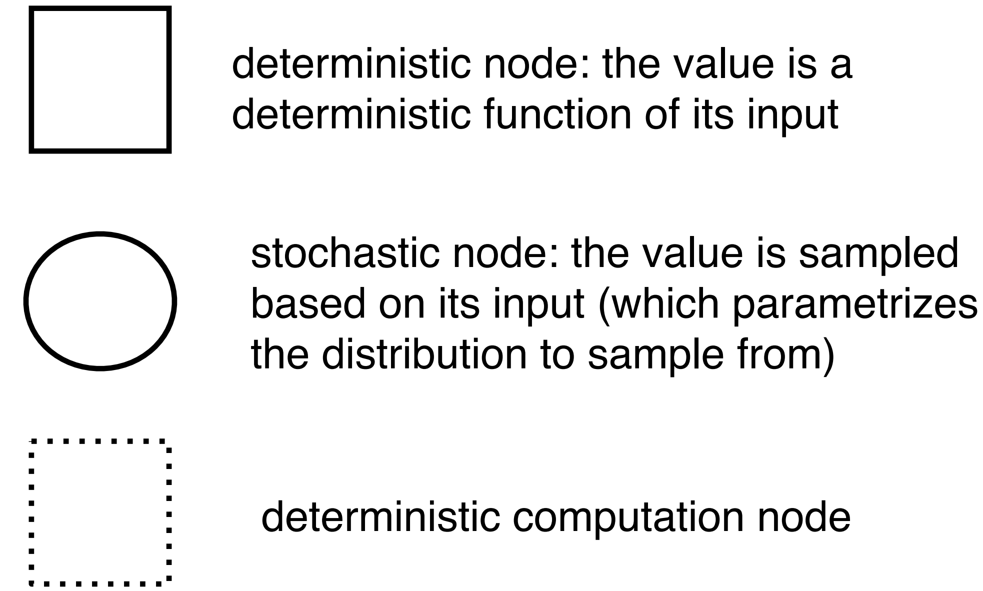
We can either:

- Assume σ fixed (not learned)
- Learn $\sigma(s, \theta)$ one value for all action coordinates (spherical or isotropic Gaussian)
- Learn $\sigma^i(s, \theta), i = 1 \cdots n$ (diagonal covariance)
- Learn a full covariance matrix $\Sigma(s, \theta)$

Example: Gaussian policy



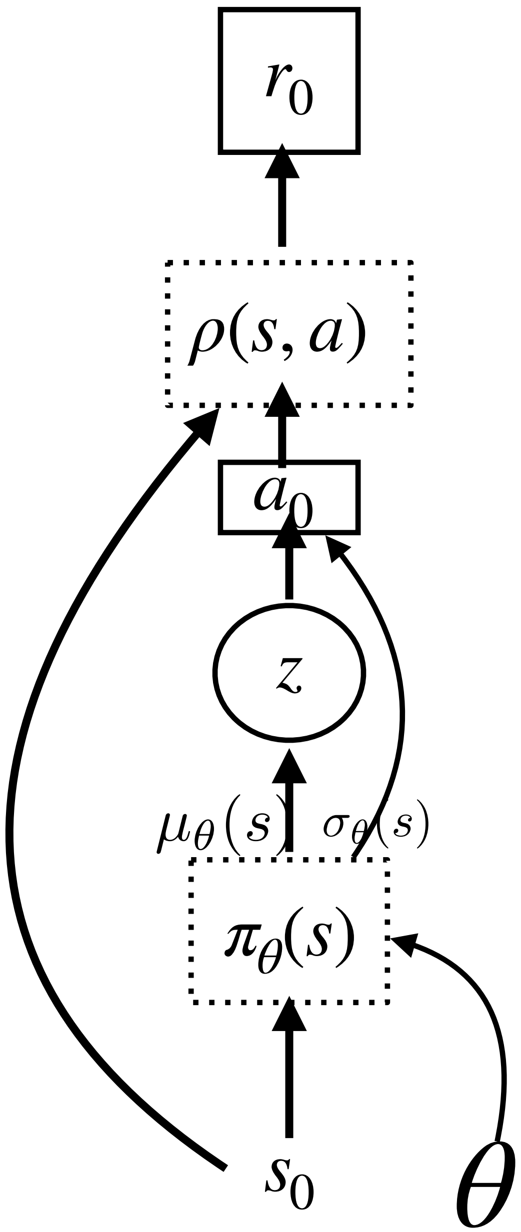
$a \sim \mathcal{N}(\mu(s, \theta), \Sigma(s, \theta))$



We can either:

- Assume σ fixed (not learned)
- Learn $\sigma(s, \theta)$ one value for all action coordinates (spherical or isotropic Gaussian)
- Learn $\sigma^i(s, \theta), i = 1 \cdots n$ (diagonal covariance)
- Learn a full covariance matrix $\Sigma(s, \theta)$

Re-parametrization for Gaussian



Instead of: $a \sim \mathcal{N}(\mu(s,\theta), \Sigma(s,\theta))$

We can write: $a = \mu(s,\theta) + z\sigma(s,\theta) \;\; z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$

Because: $\mathbb{E}_z(\mu(s,\theta) + z\sigma(s,\theta)) = \mu(s,\theta)$

$$\mathrm{Var}_z(\mu(s,\theta) + z\sigma(s,\theta)) = \sigma(s,\theta)^2\mathbf{I}_{n\times n}$$

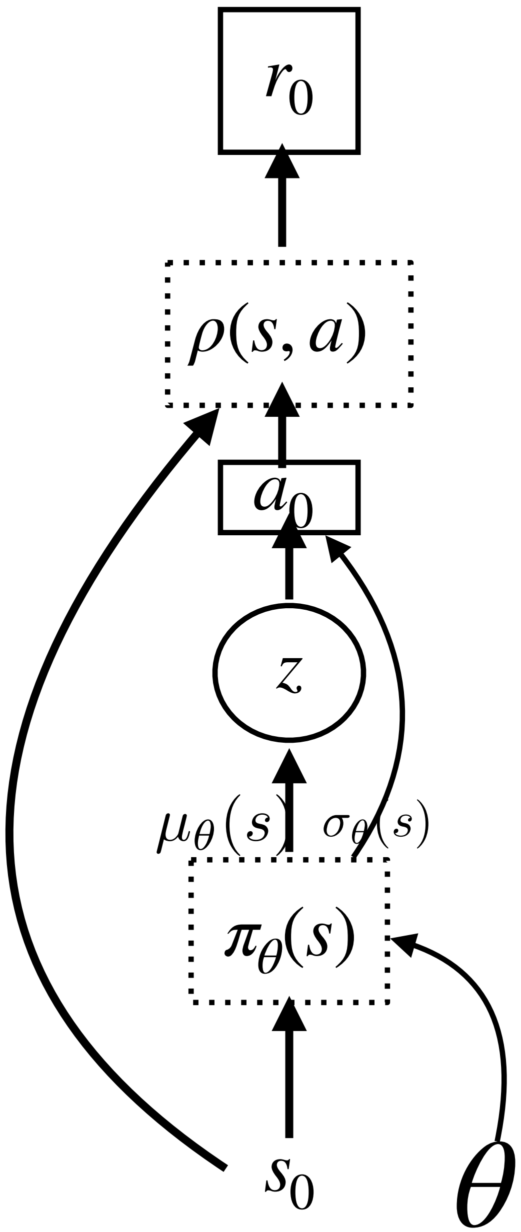
$$\max_\theta . \quad \mathbb{E}_a \rho(s_0, a)$$

$$\max_\theta . \quad \mathbb{E}_z \rho(s_0, a(z))$$

Qs:

- Does a depend on θ ?
- Does z depend on θ ?

Re-parametrization for Gaussian



Instead of: $a \sim \mathcal{N}(\mu(s,\theta), \Sigma(s,\theta))$

We can write: $a = \mu(s,\theta) + z\sigma(s,\theta) \quad z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$

What do we gain?

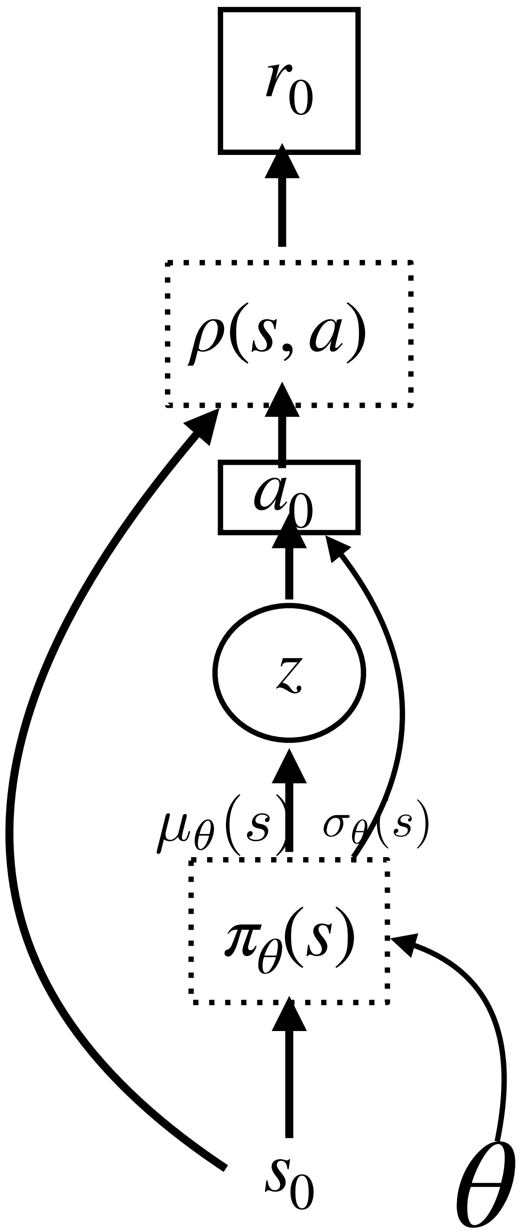
$$\nabla_\theta \mathbb{E}_z \left[\rho\left(a(\theta,z),s\right)\right] = \mathbb{E}_z \frac{d\rho\left(a(\theta,z),s\right)}{da} \frac{da(\theta,z)}{d\theta}$$

$$\frac{da(\theta,z)}{d\theta} = \frac{d\mu(s,\theta)}{d\theta} + z\frac{d\sigma(s,\theta)}{d\theta}$$

$$\max_\theta . \quad \mathbb{E}_a \rho(s_0, a)$$

$$\max_\theta . \quad \mathbb{E}_z \rho(s_0, a(z))$$

Re-parametrization for Gaussian



Instead of: $a \sim \mathcal{N}(\mu(s,\theta), \Sigma(s,\theta))$

We can write: $a = \mu(s,\theta) + z\sigma(s,\theta)$ $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n})$

What do we gain?

$$\nabla_\theta \mathbb{E}_z\left[\rho\left(a(\theta, z), s\right)\right] = \mathbb{E}_z \frac{d\rho\left(a(\theta, z), s\right)}{da} \frac{da(\theta, z)}{d\theta}$$

$$\frac{da(\theta, z)}{d\theta} = \frac{d\mu(s,\theta)}{d\theta} + z\frac{d\sigma(s,\theta)}{d\theta}$$

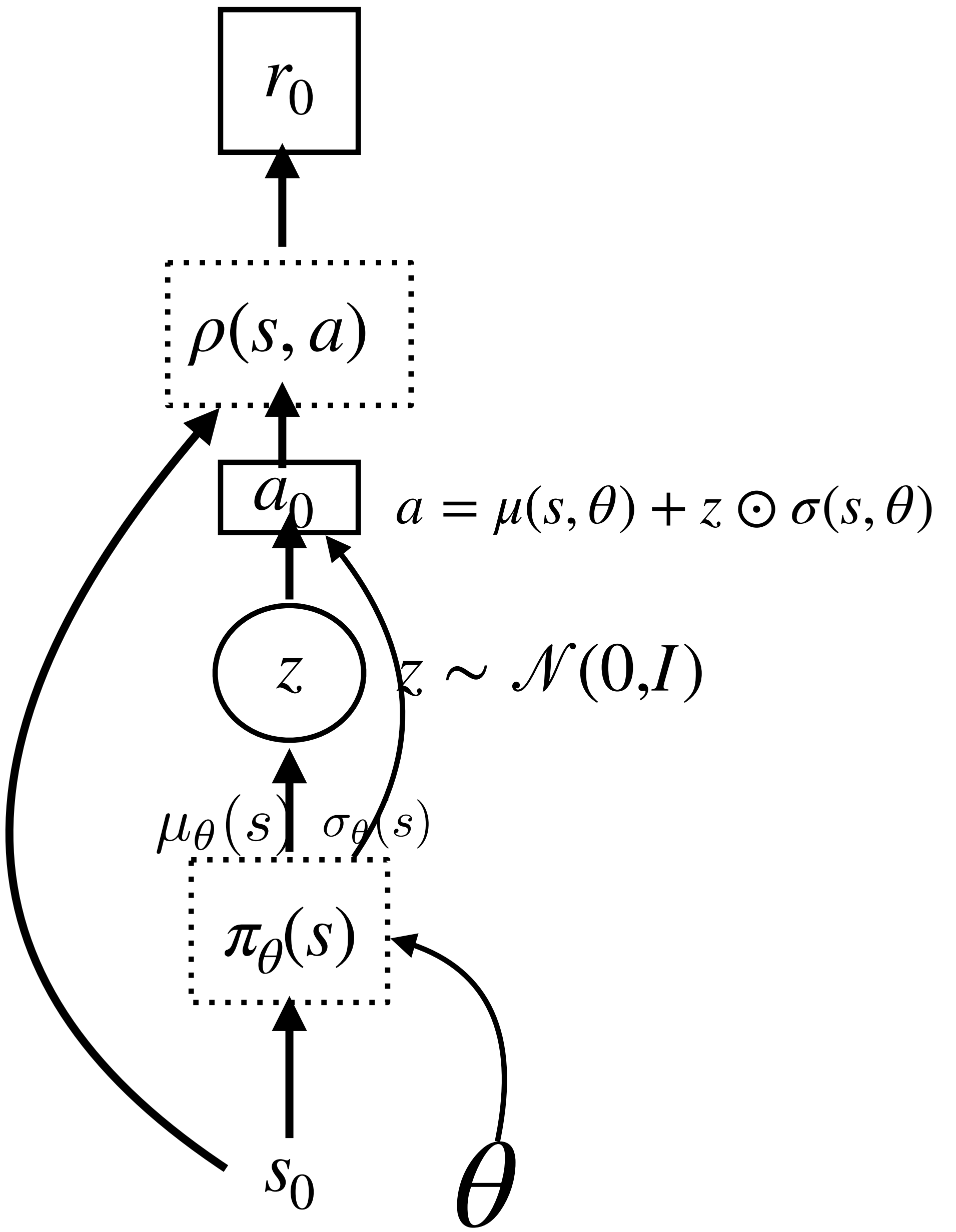
$$\max_\theta . \quad \mathbb{E}_a \rho(s_0, a)$$

$$\max_\theta . \quad \mathbb{E}_z \rho(s_0, a(z))$$

Sample estimate:

$$\nabla_\theta \frac{1}{N}\sum_{i=1}^{N}\left[\rho\left(a(\theta, z_i), s\right)\right] = \frac{1}{N}\sum_{i=1}^{N} \frac{d\rho\left(a(\theta, z), s\right)}{da} \frac{da(\theta, z)}{d\theta}\Big|_{z=z_i}$$

Re-parametrization for Gaussian



Likelihood ratio grad estimator:

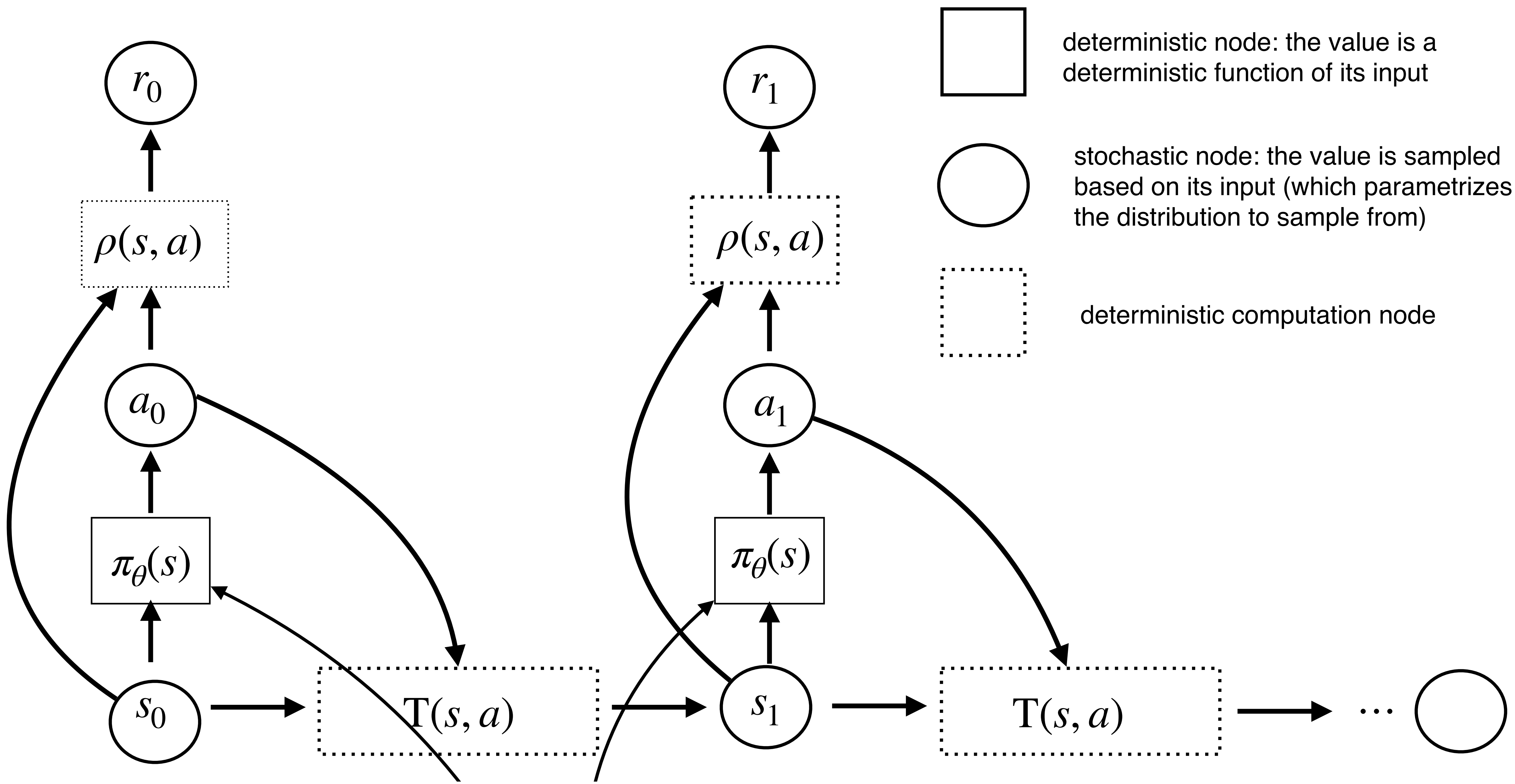
$\mathbb{E}_a \nabla_\theta \log \pi_\theta(s,a)\rho(s,a)$

Pathwise derivative:

$\mathbb{E}_z \frac{d\rho\left(a(\theta,z),s\right)}{da}\frac{da(\theta,z)}{d\theta}$

The pathwise derivative uses the derivative of the reward w.r.t. the action!

Known MDP with known deterministic reward and dynamic functions



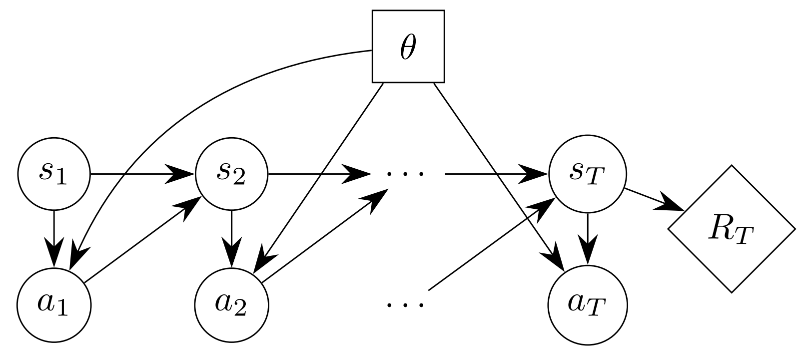
Can we apply the chair rule through deterministic policies?

Can we apply the chain rule through sampled actions?

θ

Re-parametrized Policy Gradients

Episodic MDP:



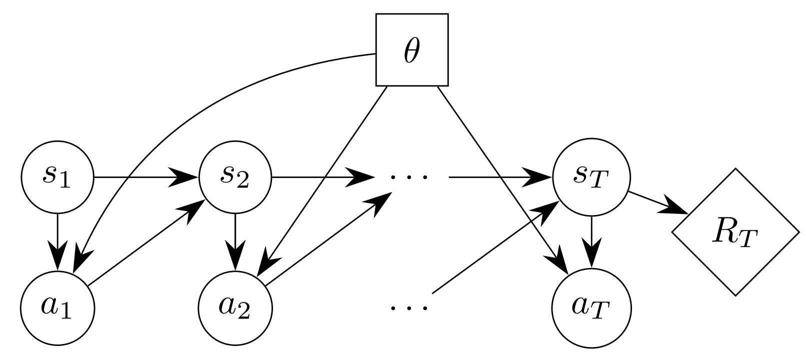
We want to compute: $\nabla_\theta \mathbb{E}[R_T]$

The problem is: we do not know the reward function, neither the dynamics function.

Solution: we will approximate it with the Q function!!!!

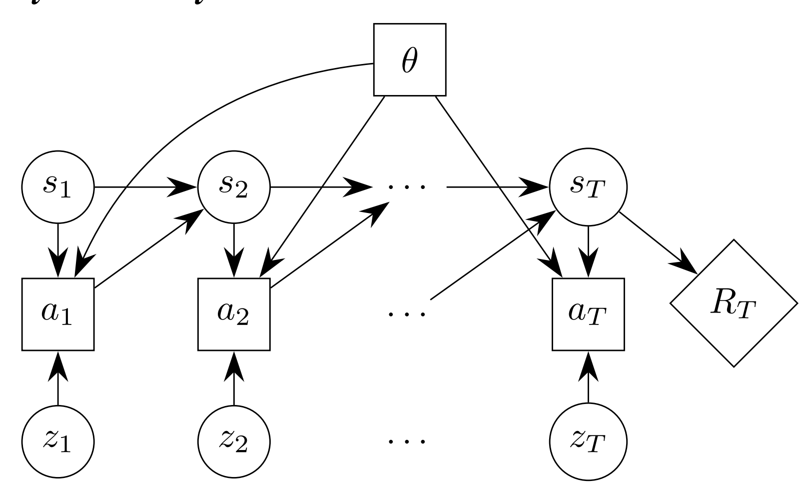
Re-parametrized Policy Gradients

- Episodic MDP:



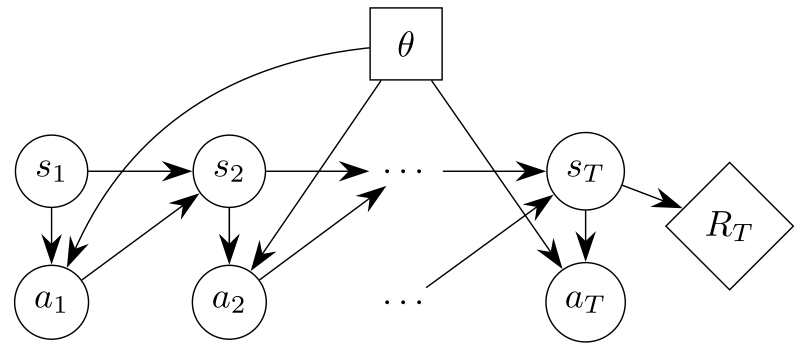
We want to compute: $\nabla_\theta \mathbb{E}[R_T]$

- Reparameterize: $a_t = \pi(s_t, z_t, \theta)$. z_t is noise from fixed distribution



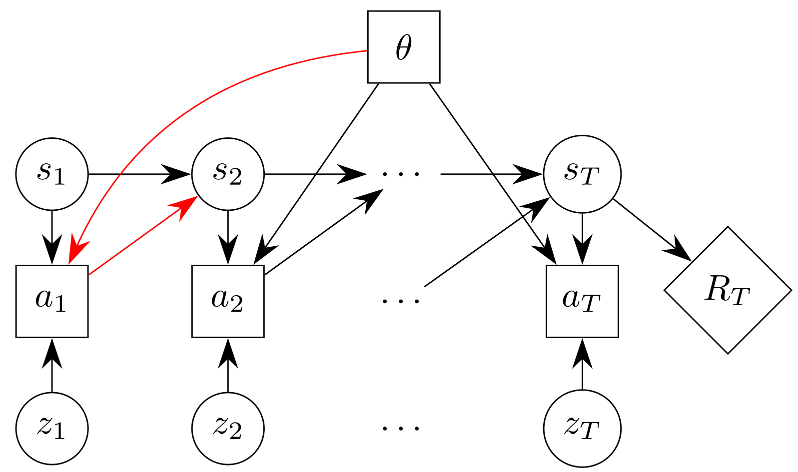
Re-parametrized Policy Gradients

- Episodic MDP:

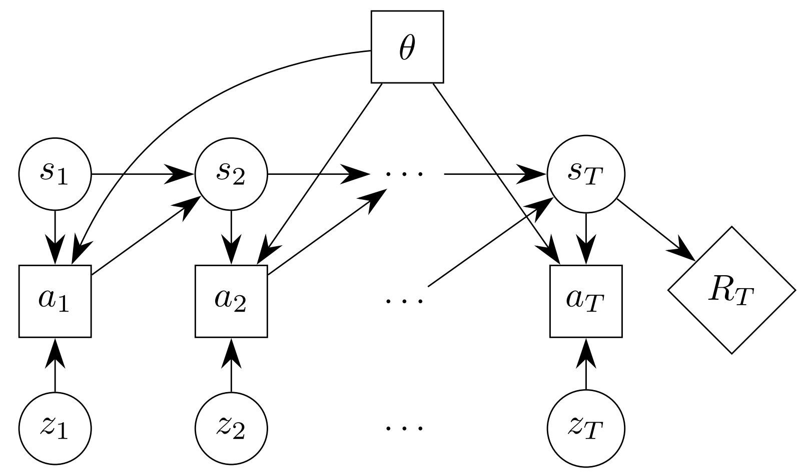


We want to compute: $\nabla_\theta \mathbb{E}[R_T]$

- Reparameterize: $a_t = \pi(s_t, z_t, \theta)$. z_t is noise from fixed distribution



Re-parametrized Policy Gradients

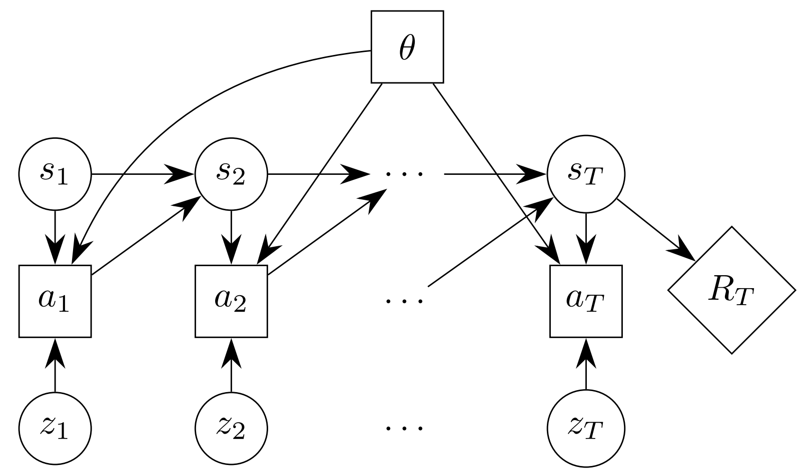


$$\frac{\mathrm{d}}{\mathrm{d}\theta}\mathbb{E}\left[R_T\right] = \mathbb{E}\left[\sum_{t=1}^{T} \frac{\mathrm{d}R_T}{\mathrm{d}a_t}\frac{\mathrm{d}a_t}{\mathrm{d}\theta}\right] = \mathbb{E}\left[\sum_{t=1}^{T} \frac{\mathrm{d}}{\mathrm{d}a_t}\mathbb{E}\left[R_T \,|\, a_t\right]\frac{\mathrm{d}a_t}{\mathrm{d}\theta}\right]$$

The problem is: we do not know the reward function!

Solution: we will approximate it with the Q function!!!!

Re-parametrized Policy Gradients



$$\frac{\mathrm{d}}{\mathrm{d}\theta}\mathbb{E}\left[R_T\right] = \mathbb{E}\left[\sum_{t=1}^{T}\frac{\mathrm{d}R_T}{\mathrm{d}a_t}\frac{\mathrm{d}a_t}{\mathrm{d}\theta}\right] = \mathbb{E}\left[\sum_{t=1}^{T}\frac{\mathrm{d}}{\mathrm{d}a_t}\mathbb{E}\left[R_T \,|\, a_t\right]\frac{\mathrm{d}a_t}{\mathrm{d}\theta}\right]$$

$$= \mathbb{E}\left[\sum_{t=1}^{T}\frac{\mathrm{d}Q\left(s_t, a_t\right)}{\mathrm{d}a_t}\frac{\mathrm{d}a_t}{\mathrm{d}\theta}\right] = \mathbb{E}\left[\sum_{t=1}^{T}\frac{\mathrm{d}}{\mathrm{d}\theta}Q\left(s_t, \pi\left(s_t, z_t; \theta\right)\right)\right]$$

Learn Q_ϕ to approximate $Q^{\pi,\gamma}$, and use it to compute gradient estimates

Learning continuous control by stochastic value gradients, Hees et al.

Stochastic Value Gradients V0

Learn Q_ϕ to approximate $Q^{\pi,\gamma}$, and use it to compute gradient estimates

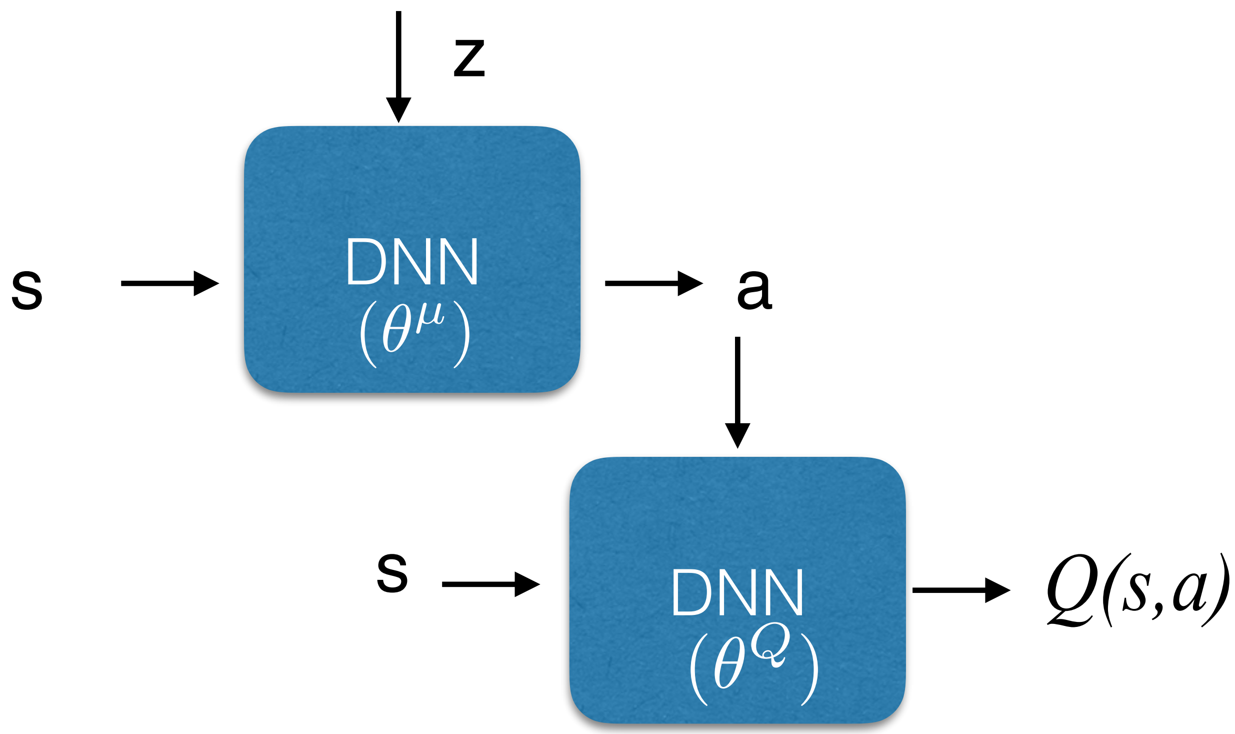
Algorithm:

for iteration$=1, 2, \dots$ **do**
  Execute policy π_θ to collect T timesteps of data
  Update π_θ using $g \propto \nabla_\theta \sum_{t=1}^{T} Q(s_t, \pi(s_t, z_t; \theta))$
  Update Q_ϕ using $g \propto \nabla_\phi \sum_{t=1}^{T} (Q_\phi(s_t, a_t) - \hat{Q}_t)^2$, e.g. with TD($\lambda$)
end for

Learning continuous control by stochastic value gradients, Hees et al.

Stochastic Value Gradients V0

$$z \sim \mathcal{N}(0, 1)$$

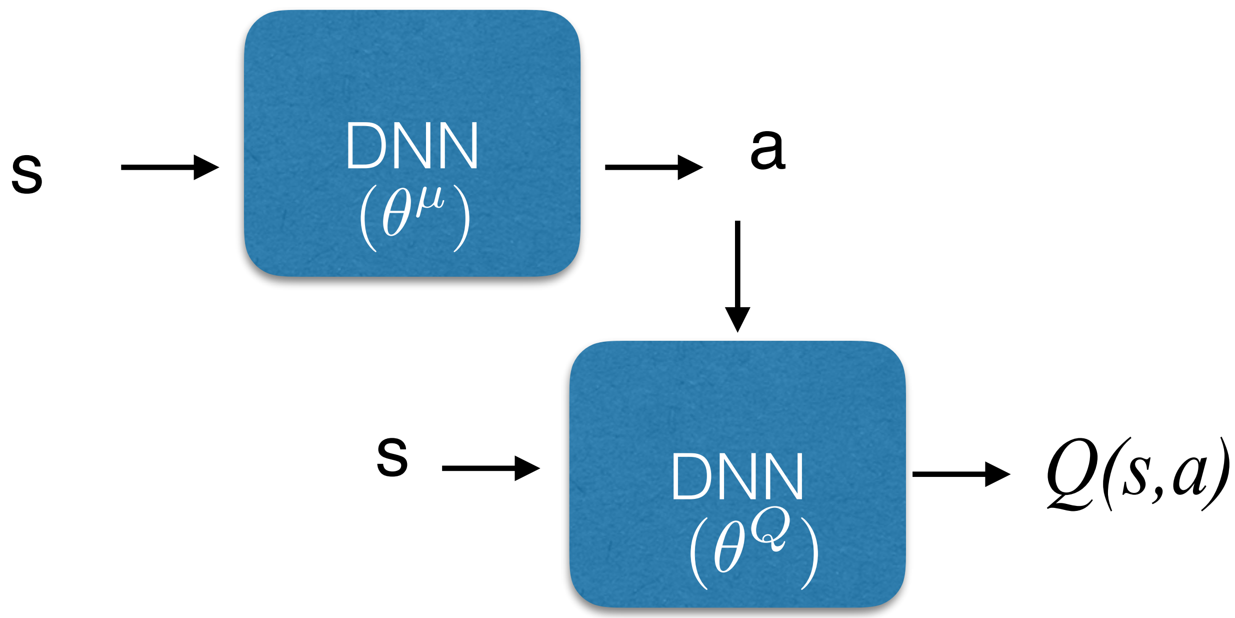


$$a = \mu(s;\theta) + z\sigma(s;\theta)$$

Learning continuous control by stochastic value gradients, Hees et al.

Compare with: Deep Deterministic Policy Gradients

$$a = \mu(\theta)$$



No z!

Learning continuous control by stochastic value gradients, Hees et al.