Carnegie Mellon
School of Computer Science

Deep Reinforcement Learning and Control

# Pathwise derivatives, DDPG

CMU 10-703

Katerina Fragkiadaki

# $\nabla_\theta \log \pi_\theta(a)$ for Gaussian policy

- Policy is Gaussian $a \sim \mathcal{N}(\mu(s,\theta), \sigma^2 I)$
- Variance may be fixed $\sigma^2$, or can also parameterized
- Remember: univariete Gaussian $\mathcal{N}(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(a-\mu)^2}{\sigma^2}}$
- $\nabla_\theta \log \pi_\theta(a \mid s) = \text{const.} \cdot \frac{(\mu_\theta(s) - a)}{\sigma^2} \nabla_\theta \mu_\theta(s)$

$$\hat{g} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t^{(i)} \mid s_t^{(i)})(G_t^i - b(s_t^{(i)}))$$

# $\nabla_\theta \log \pi_\theta(a)$ for softmax policy

- Policy is the output of a softmax

$$\pi_\theta(a \mid s) = \frac{e^{h_\theta(s,a)}}{\sum_b e^{h_\theta(s,b)}}$$

$$\begin{aligned}
\nabla_\theta \log \pi_\theta(a) = \; & \nabla_\theta h_\theta(s,a) - \nabla_\theta \log \sum_b e^{h_\theta(s,b)} \\
& \nabla_\theta h_\theta(s,a) - \frac{1}{\sum_b e^{h_\theta(s,b)}} \nabla_\theta \sum_b e^{h_\theta(s,b)} \\
& \nabla_\theta h_\theta(s,a) - \frac{1}{\sum_b e^{h_\theta(s,b)}} \sum_b \nabla_\theta e^{h_\theta(s,b)} \\
& \nabla_\theta h_\theta(s,a) - \frac{1}{\sum_b e^{h_\theta(s,b)}} \sum_b e^{h_\theta(s,b)} \nabla_\theta h_\theta(s,b) \\
& \nabla_\theta h_\theta(s,a) - \sum_b \frac{e^{h_\theta(s,b)}}{\sum_b e^{h_\theta(s,b)}} \nabla_\theta h_\theta(s,b) \\
& \nabla_\theta h_\theta(s,a) - \sum_b \pi_\theta(s,b) \nabla_\theta h_\theta(s,b)
\end{aligned}$$

$$\hat{g} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t^{(i)} \mid s_t^{(i)})(G_t^i - b(s_t^{(i)}))$$

# 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:

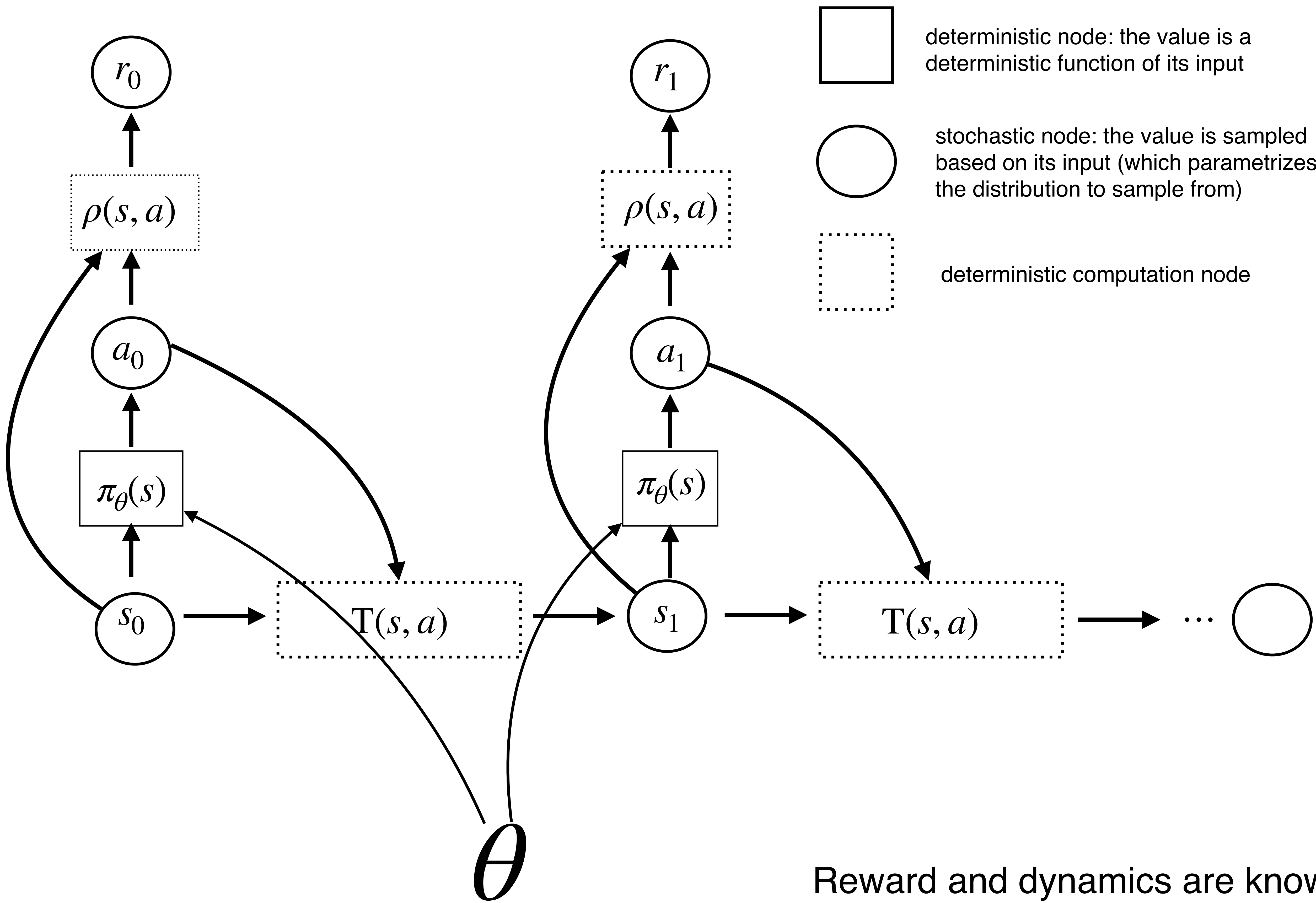
$$\nabla_\theta \mathbb{E}_{z \sim \mathcal{N}(0,1)} f(x(\theta), z) = \mathbb{E}_{z \sim \mathcal{N}(0,1)} \nabla_\theta f(x(\theta), z) = \mathbb{E}_{z \sim \mathcal{N}(0,1)} \frac{df(x(\theta), z)}{dx} \frac{dx}{d\theta}$$

pathwise derivative

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

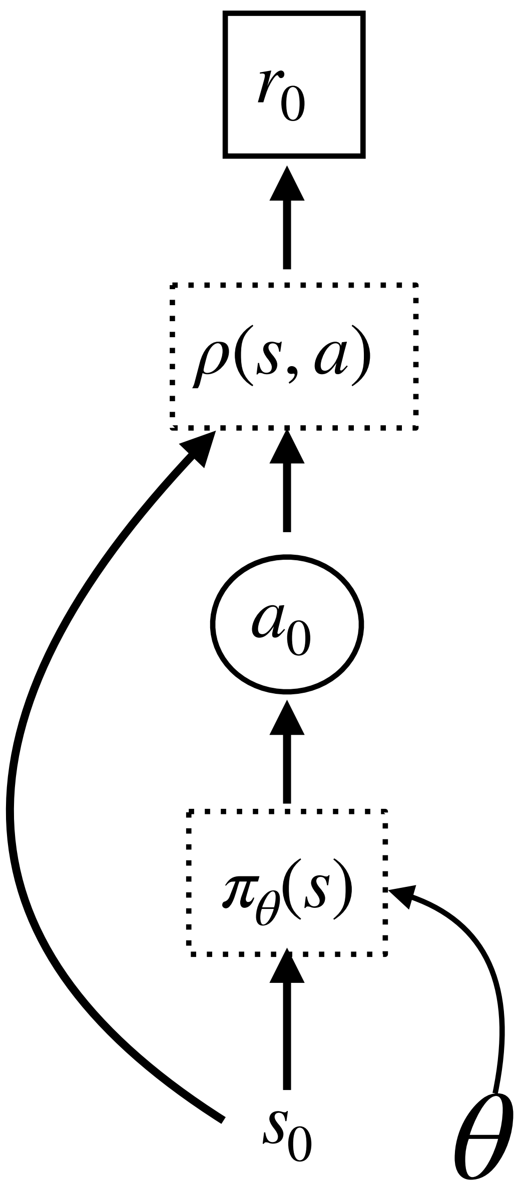
Q: From which to which? Why would we want to do so?

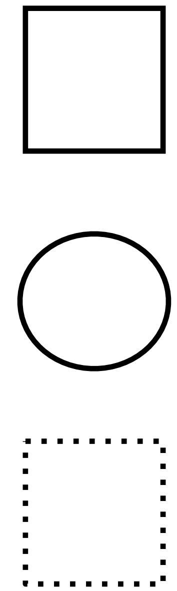
# Known MDP (what if that was possible)




Reward and dynamics are known

# Known MDP-let's make it simpler



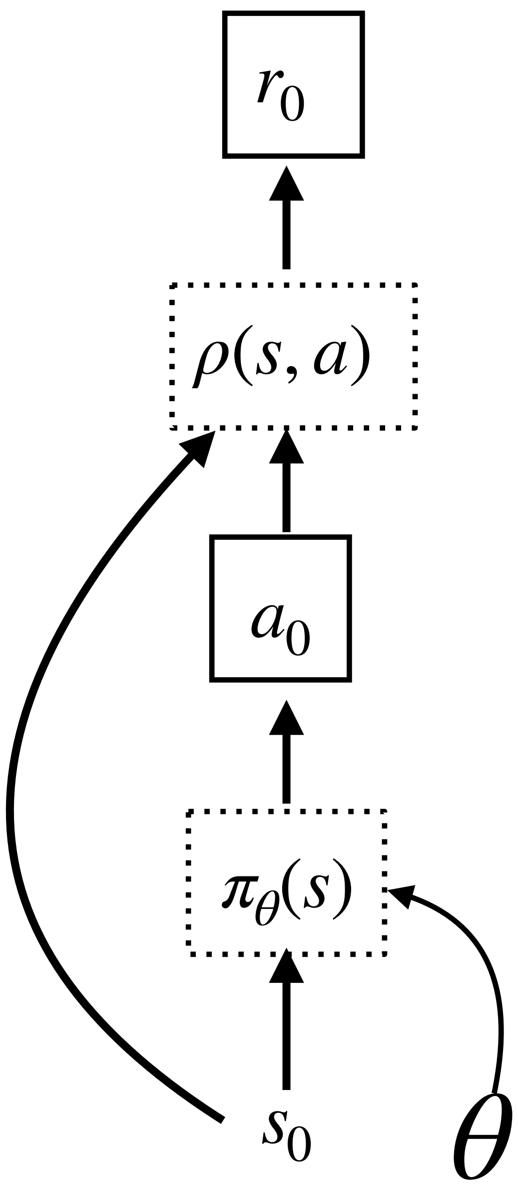
deterministic node: the value is a deterministic function of its input

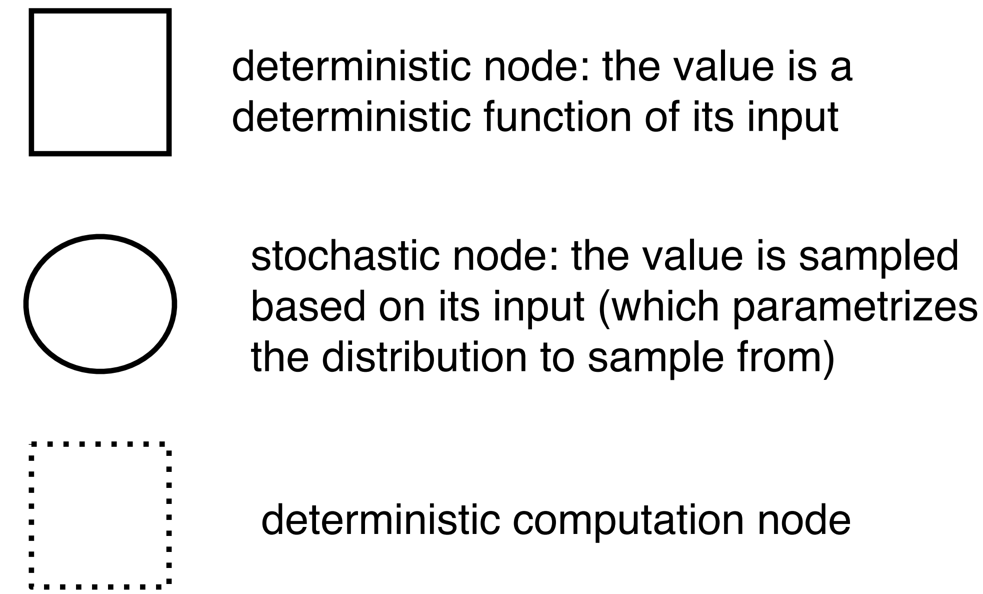
stochastic node: the value is sampled based on its input (which parametrizes the distribution to sample from)

deterministic computation node

I want to learn $\theta$ to maximize the average reward obtained.

# Deterministic policy




I want to learn $\theta$ to maximize the average reward obtained.

I can compute the gradient with backpropagation.

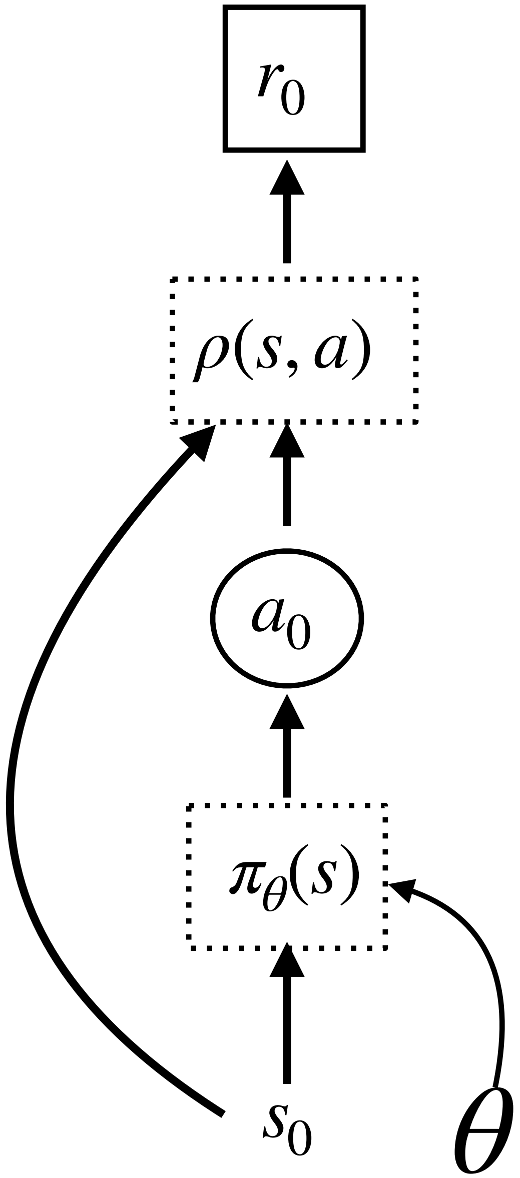
$$a = \pi_\theta(s)$$

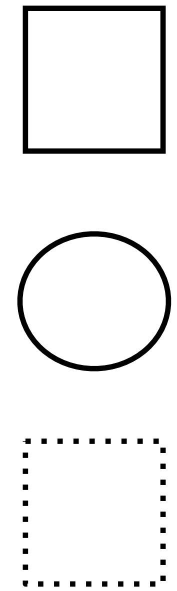
$$\nabla_\theta \rho(s, a) = \rho_a \pi_{\theta_\theta}$$

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

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

# Stochastic policy



deterministic node: the value is a deterministic function of its input

stochastic node: the value is sampled based on its input (which parametrizes the distribution to sample from)

deterministic computation node

I want to learn $\theta$ to maximize the **expected** reward obtained.

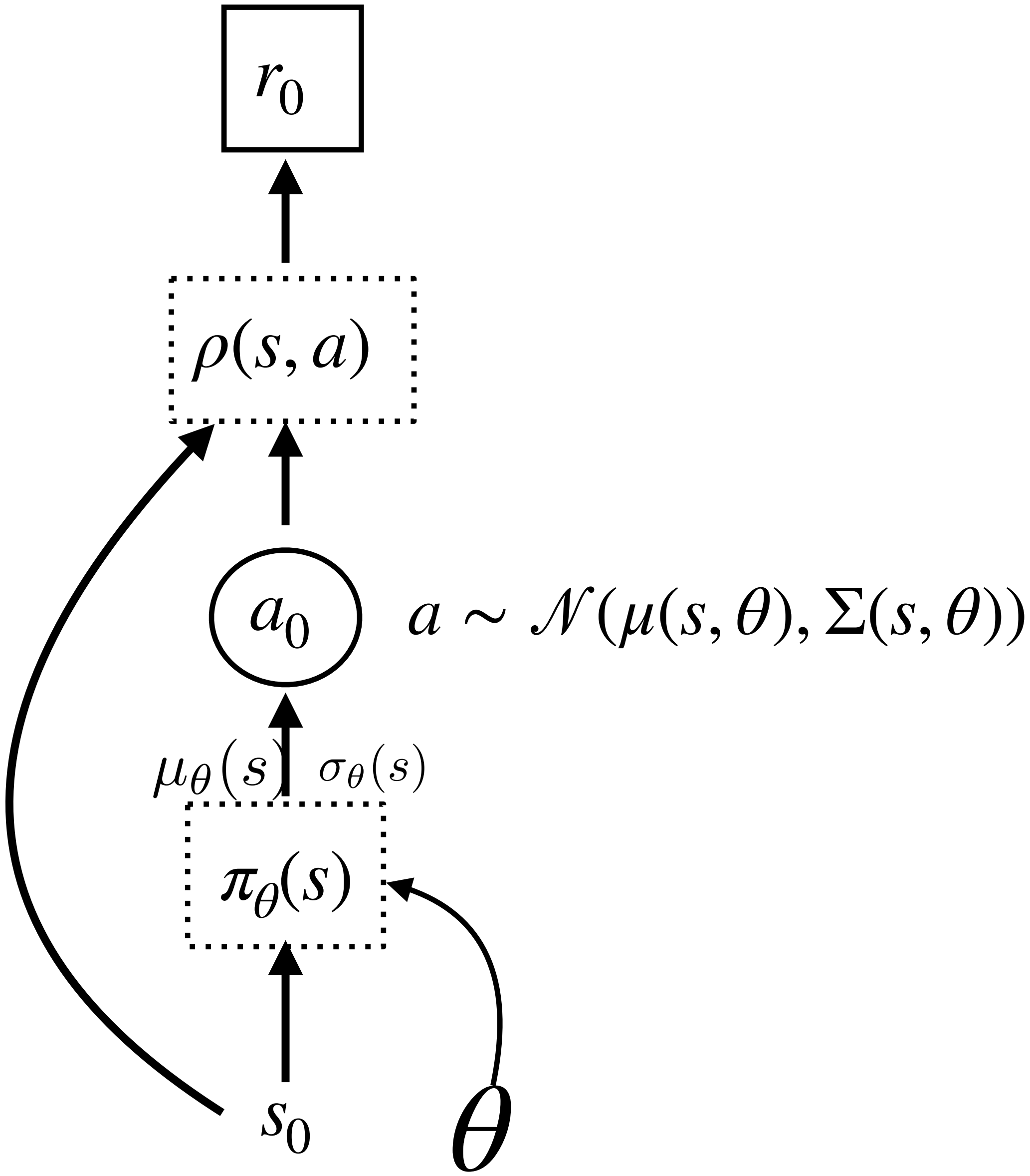
$$\max_\theta . \quad \mathbb{E}_a \rho(s_0, a)$$

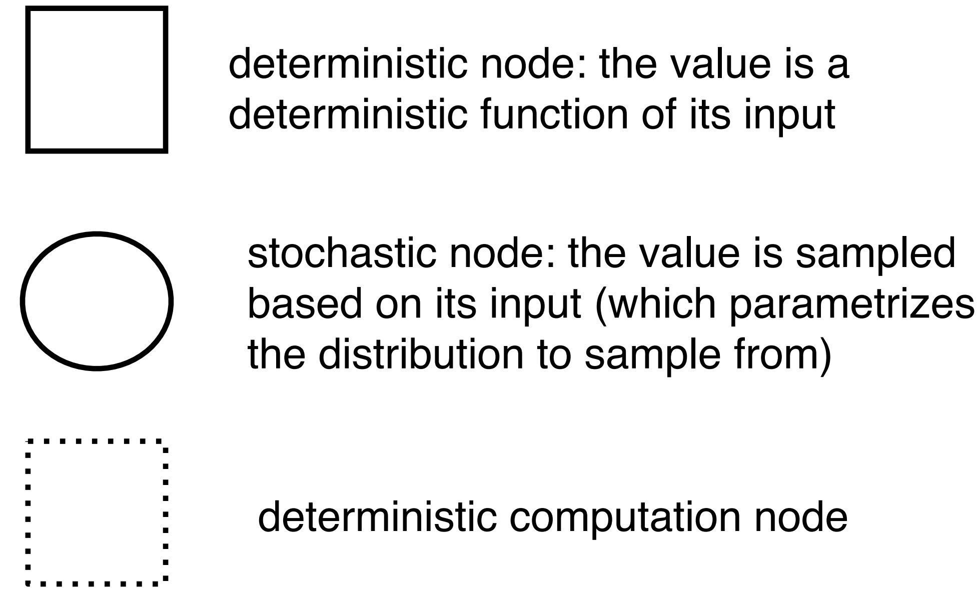
Likelihood ratio estimator, works for both continuous and discrete actions

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

It does not use the derivative of the reward w.r.t. the action.

# Example: Gaussian policy




I want to learn $\theta$ to maximize the average reward obtained.

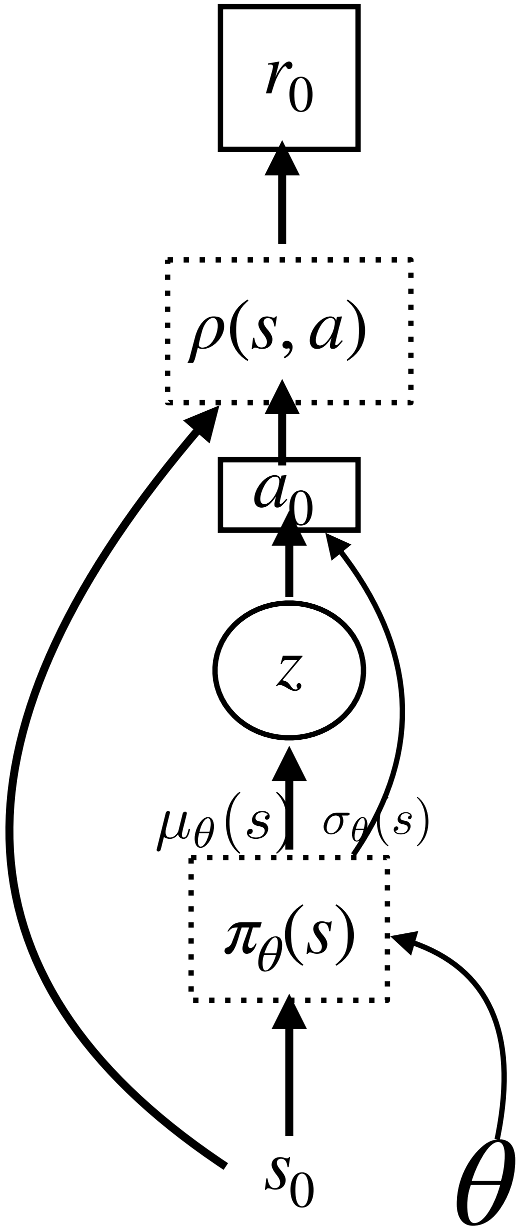
$$\mathbb{E}_a \nabla_\theta \log \pi_\theta(s) \rho(s, a)$$

If $\sigma^2$ is constant:

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

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

# Re-parametrization for Gaussian



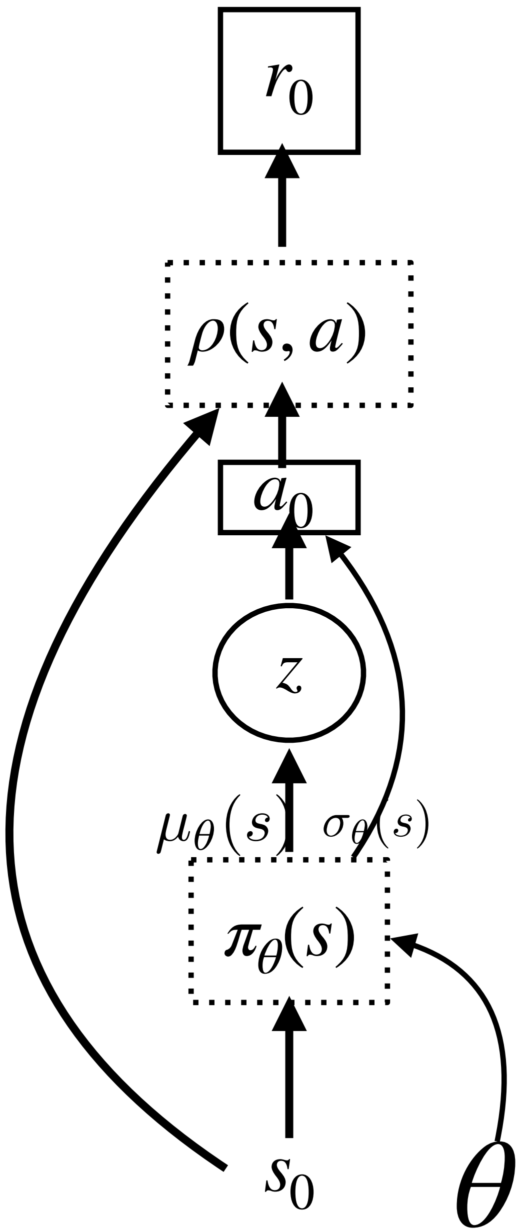

$$\max_{\theta} . \quad \mathbb{E}_a \rho(s_0, a)$$

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

We can either:

- Assume $\sigma$ fixed (spherical or isotropic Gaussian)
- 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




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

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

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

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

Why?

Because:

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

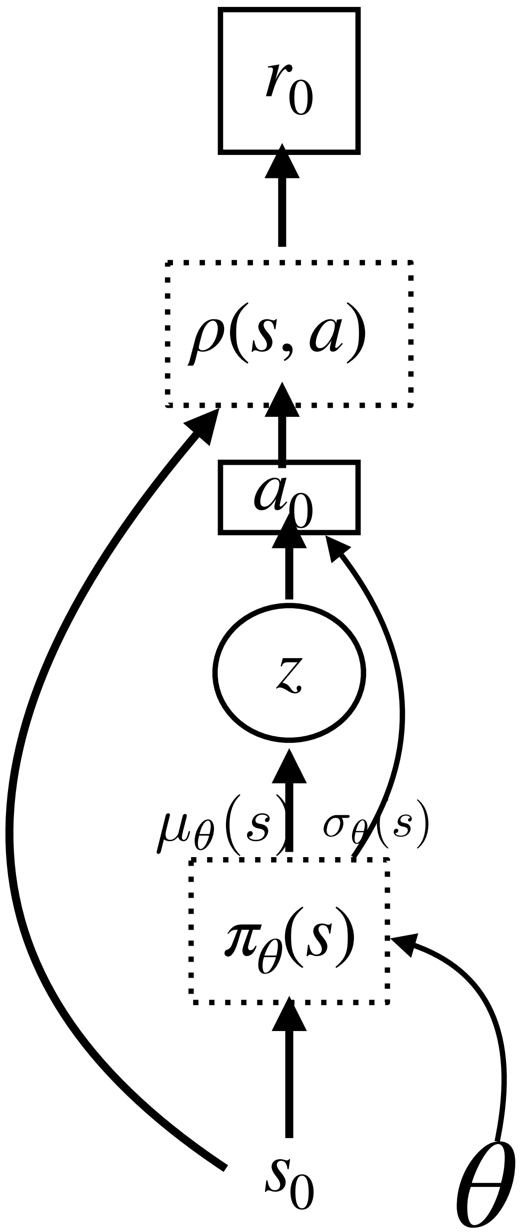
$$\mathrm{Var}_z(\mu(s,\theta) + z\sigma(s,\theta)) = \sigma(s,\theta)^2$$

Qs:

- Does $a$ depend on $\theta$ ?
- Does $z$ depend on $\theta$ ?

$$a = \mu(s,\theta) + L(s,\theta)z, \quad \Sigma = L(s,\theta)L(s,\theta)^\top$$

# Re-parametrization for Gaussian




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

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

Why?

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

$$\mathrm{Var}_z(\mu(s,\theta) + z\sigma(s,\theta)) = \sigma(s,\theta)^2$$

What do we gain?

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

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

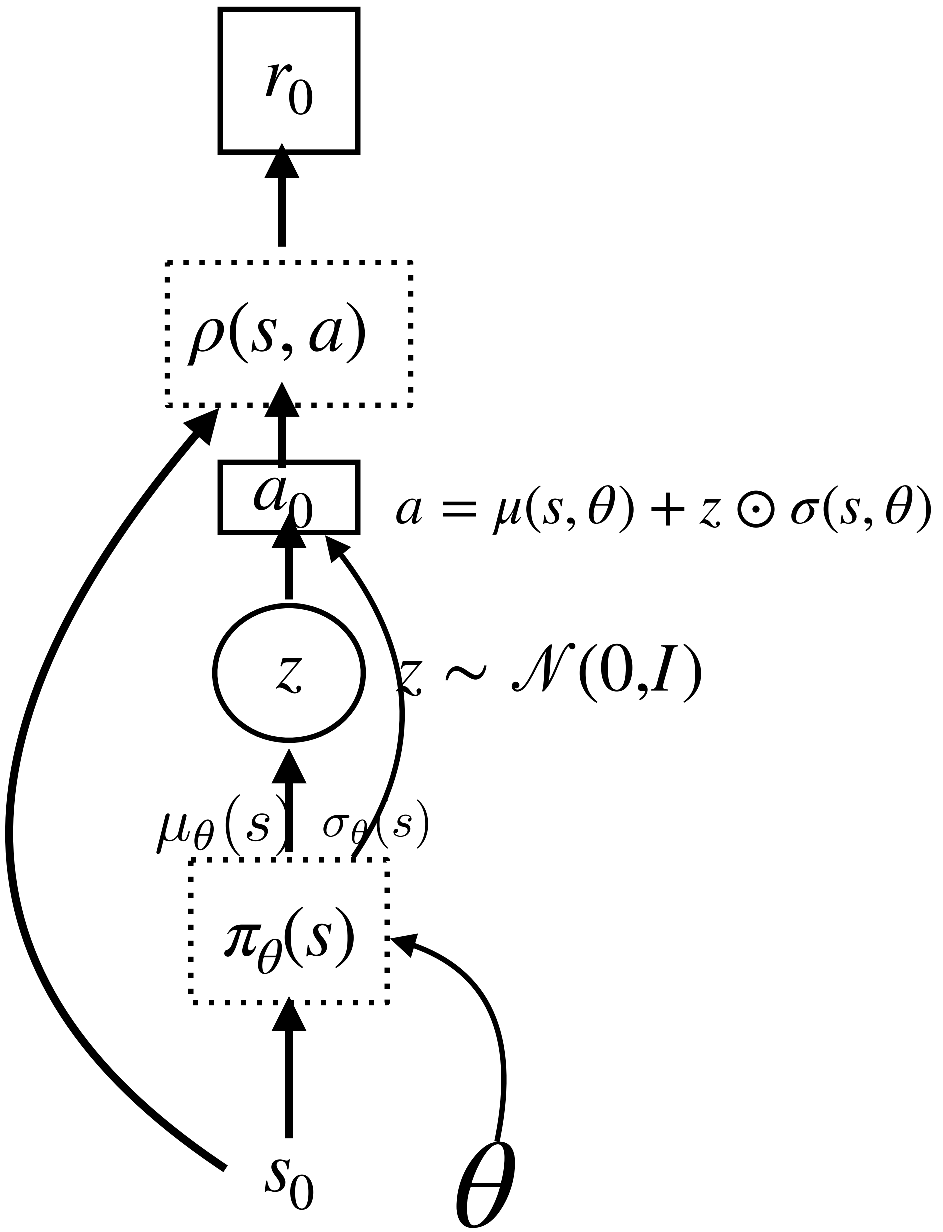
$$\max_\theta . \quad \mathbb{E}_a \rho(s_0, a)$$

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

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:

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

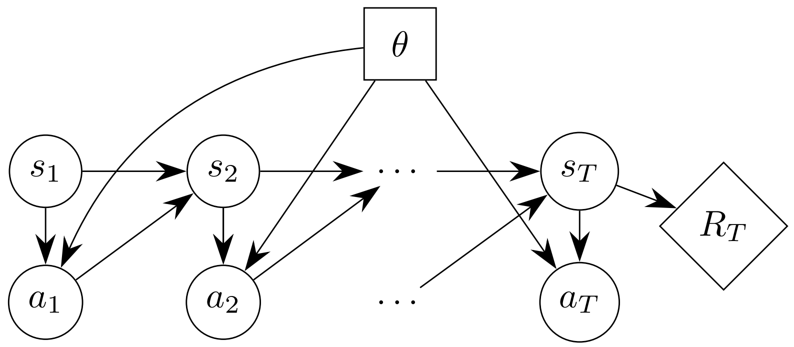
Pathwise derivative:

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

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

# Re-parametrized Policy Gradients

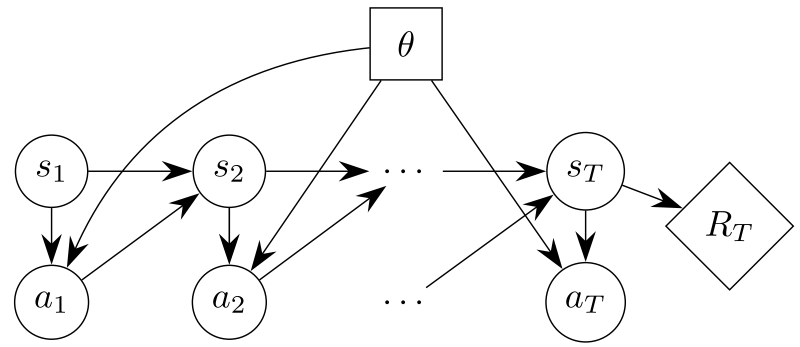
- Episodic MDP:




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

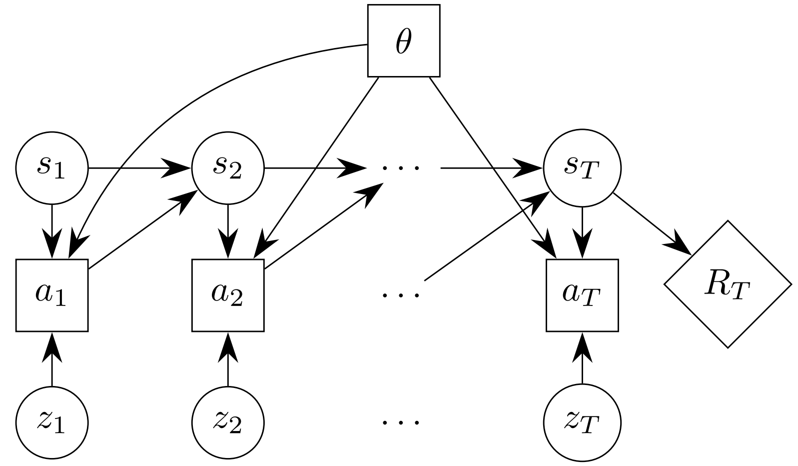
# Re-parametrized Policy Gradients

- Episodic MDP:



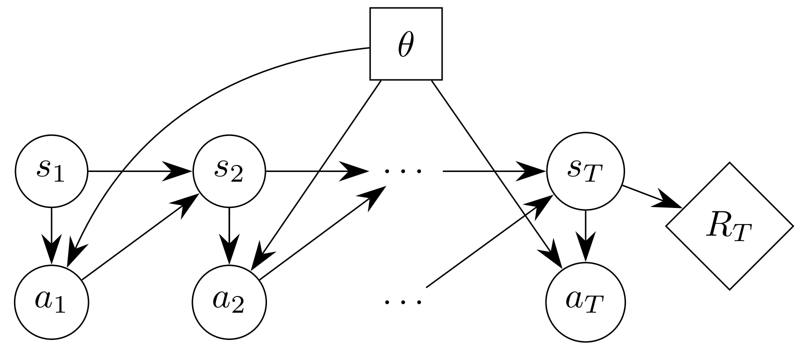

We want to compute: $\nabla_\theta \mathbb{E}[R_T]$

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



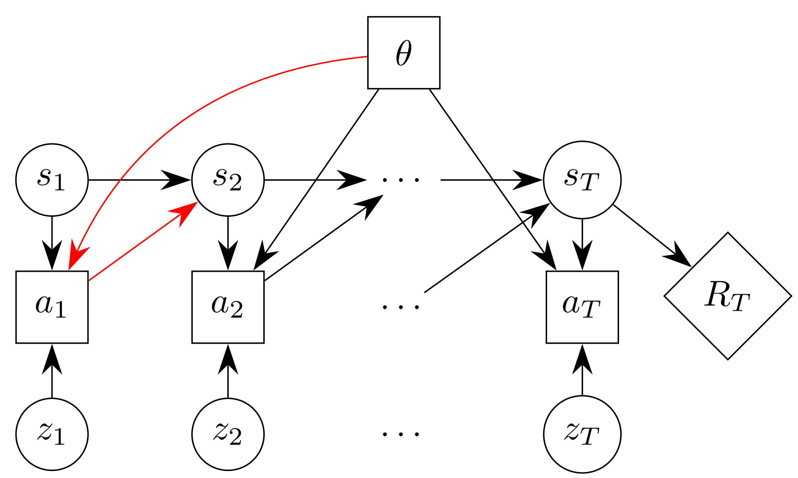
# Re-parametrized Policy Gradients

- Episodic MDP:



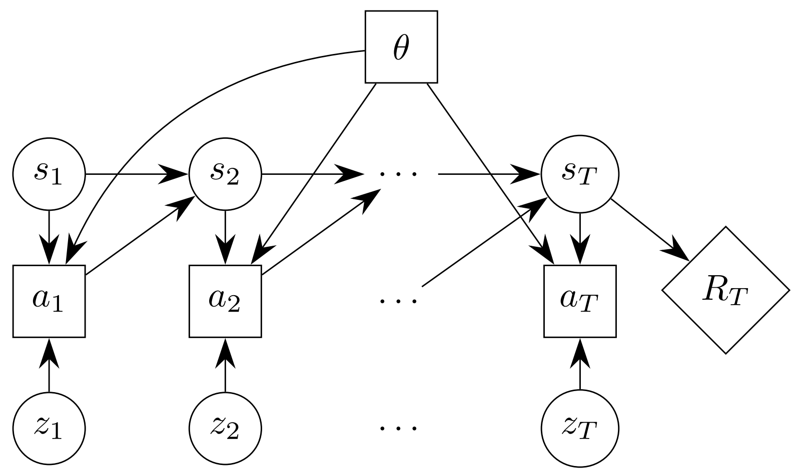

We want to compute: $\nabla_\theta \mathbb{E}[R_T]$

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




For pathwise derivative to work, we need transition dynamics and reward function to be known.

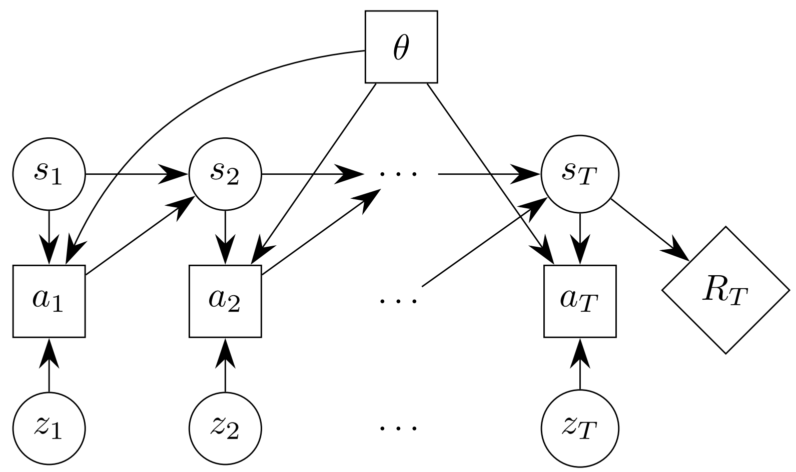
# Re-parametrized Policy Gradients




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

For path wise derivative to work, we need transition dynamics and reward function to be known, or…

# Re-parametrized Policy Gradients




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

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

- Learn $Q_\phi$ 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

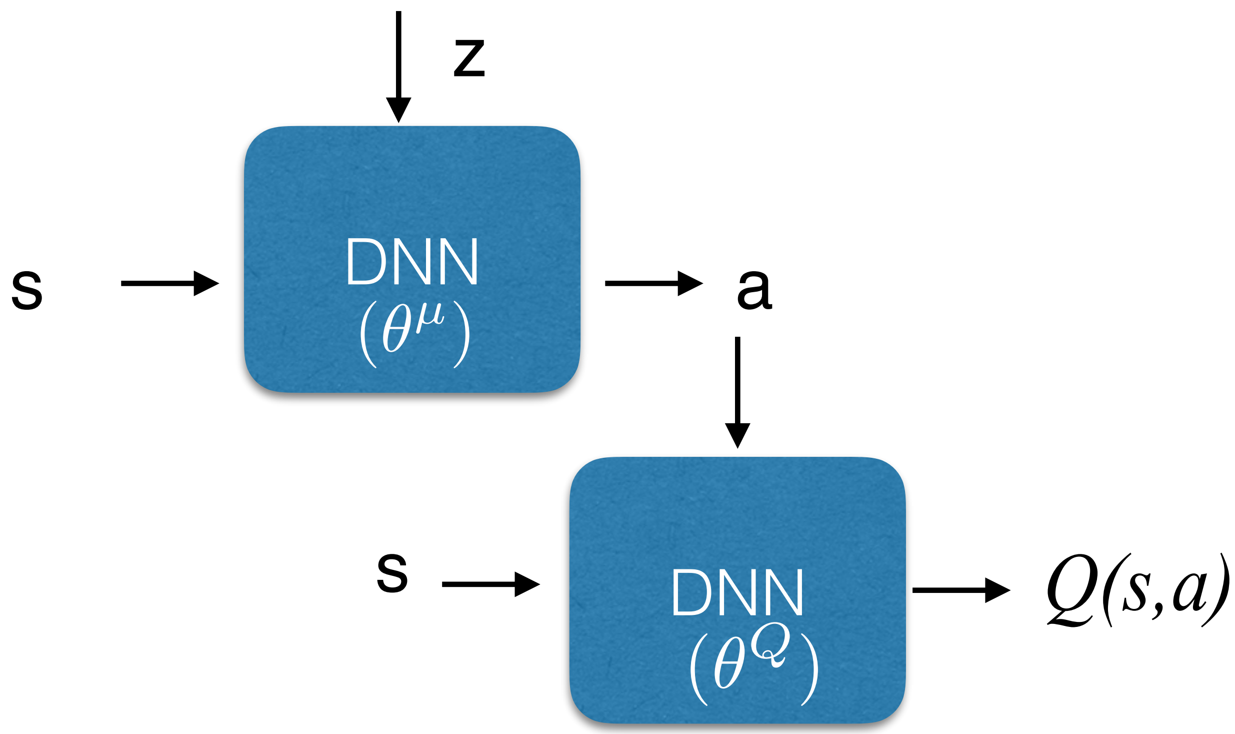
- Learn $Q_\phi$ to approximate $Q^{\pi,\gamma}$, and use it to compute gradient estimates.
- Pseudocode:

  **for** iteration$=1, 2, \ldots$ **do**
  
  Execute policy $\pi_\theta$ to collect $T$ timesteps of data
  
  Update $\pi_\theta$ using $g \propto \nabla_\theta \sum_{t=1}^{T} Q(s_t, \pi(s_t, z_t; \theta))$
  
  Update $Q_\phi$ 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

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

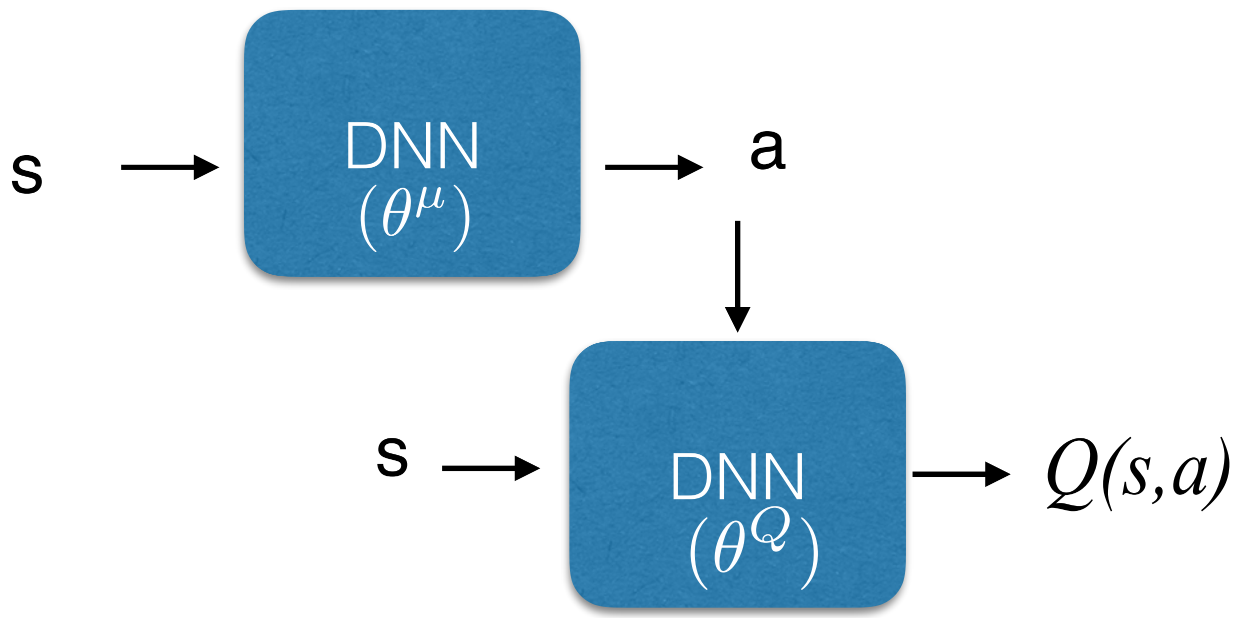



$$a = \mu(s;\theta) + z\sigma(s;\theta)$$

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

# Compare with: Deep Deterministic Policy Gradients

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




No z!