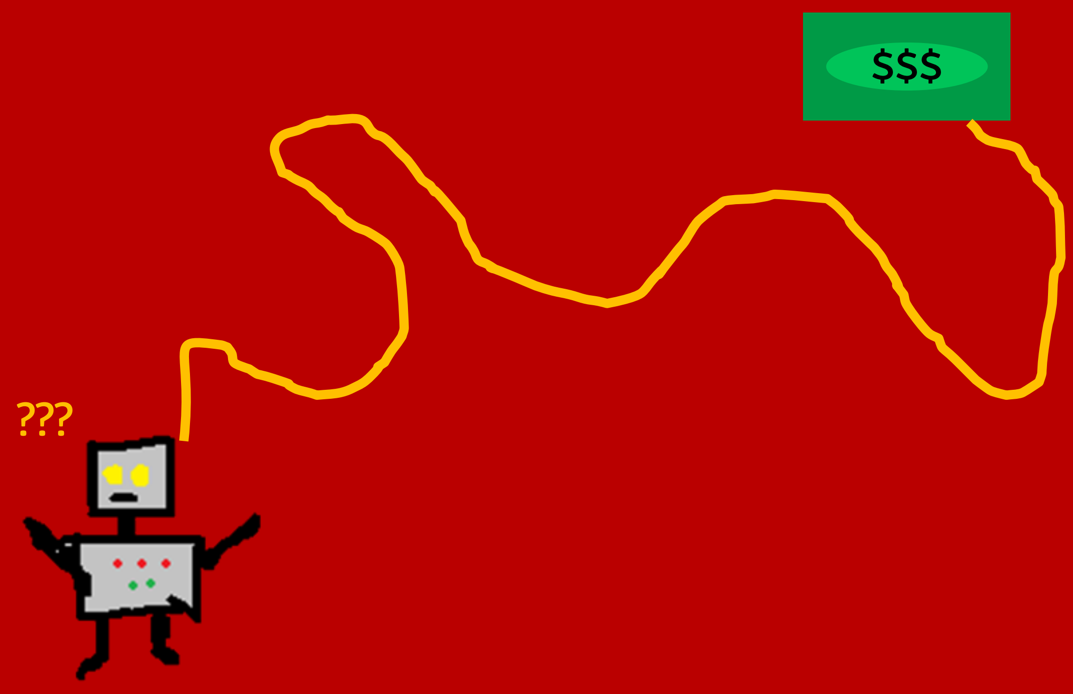


REINFORCEMENT LEARNING

Gergely Neu

Universitat Pompeu Fabra Barcelona

REINFORCEMENT LEARNING

Gergely Neu
Universitat Pompeu Fabra Barcelona

and now

and now
the moment you all
have been waiting for

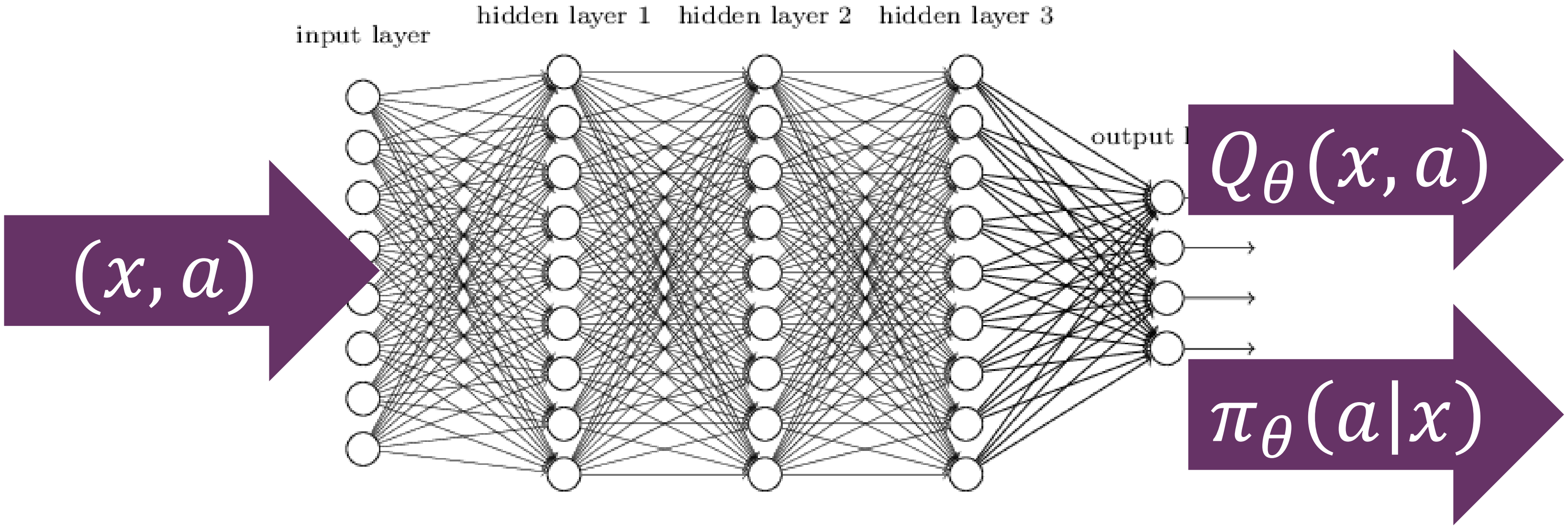
and now
the moment you all
have been waiting for

DEEP REINFORCEMENT LEARNING

…well, or at least a little teaser

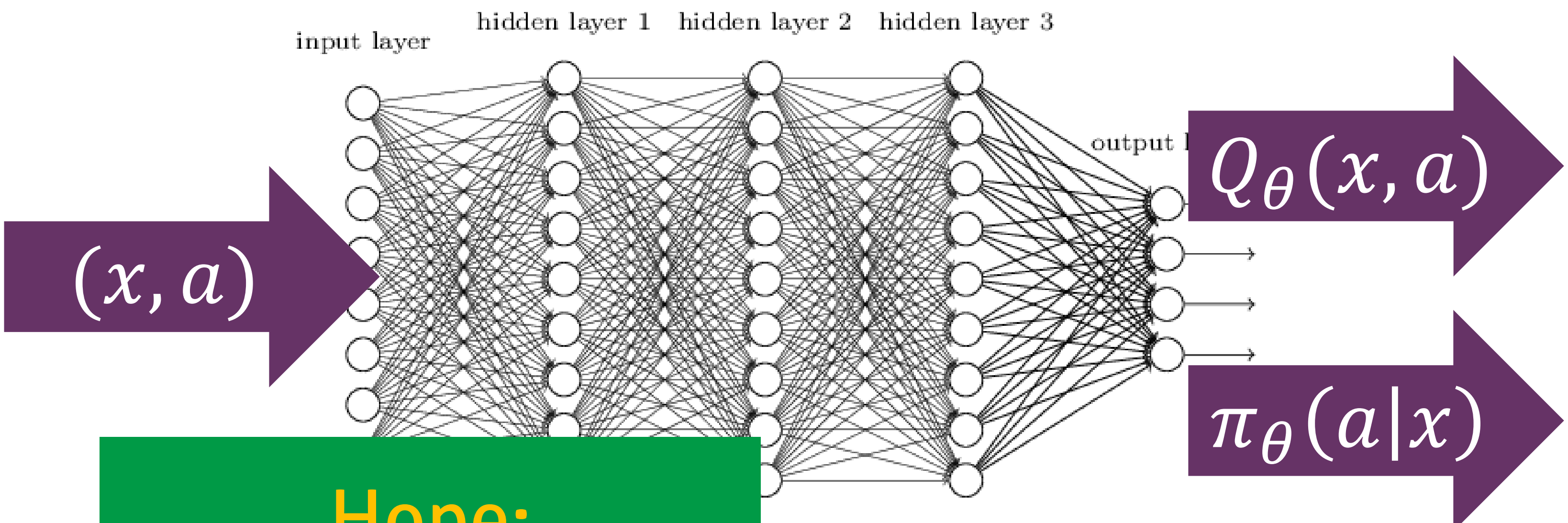
THE PROMISE OF DEEP REINFORCEMENT LEARNING

Parametrize Q-function/policy by a deep net



THE PROMISE OF DEEP REINFORCEMENT LEARNING

Parametrize Q-function/policy by a deep net

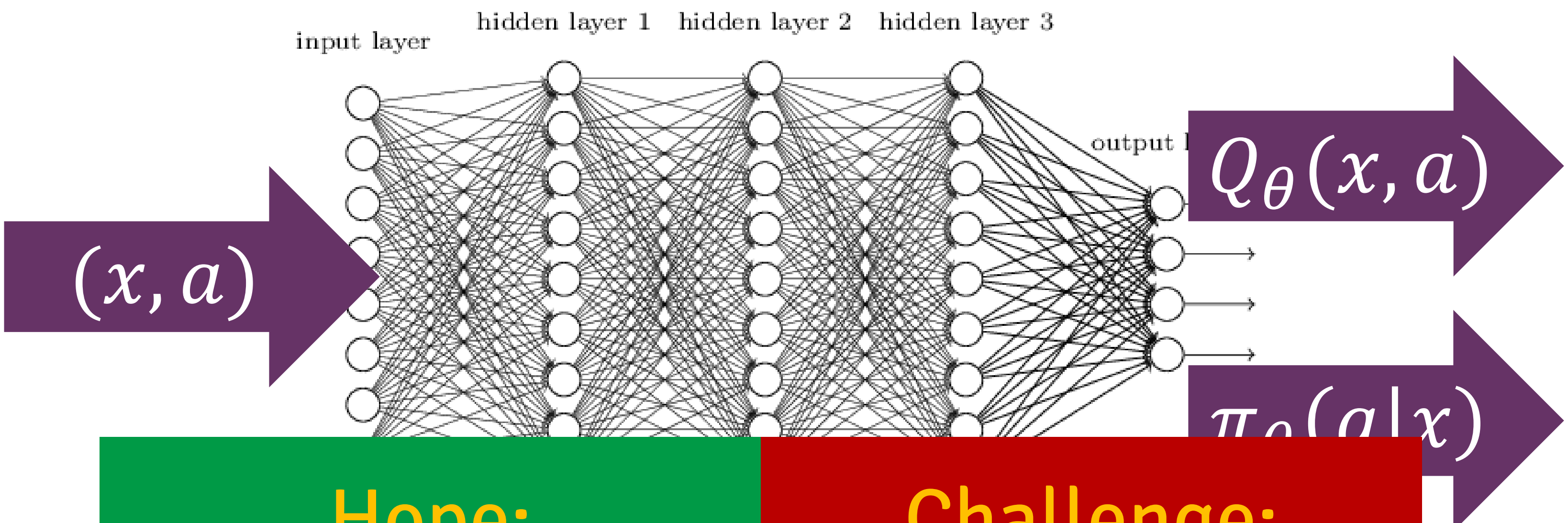


Hope:

Take advantage of representation power!

THE PROMISE OF DEEP REINFORCEMENT LEARNING

Parametrize Q-function/policy by a deep net



Hope:
Take advantage of representation power!

Challenge:
Existing RL methods difficult to generalize

Lecture 8:

The path towards deep reinforcement learning

The path towards deep RL

1. Least-squares TD
   - Linear function approx.
   - General function approx.
   - Deep Q networks
2. Policy optimization
   - The likelihood-ratio trick
   - The policy gradient theorem
   - Actor-critic algorithms

The path towards deep RL

1. Least-squares TD
   - Linear function approx.
   - General function approx.
   - Deep Q networks
2. Policy optimization
   - The likelihood-ratio trick
   - The policy gradient theorem
   - Actor-critic algorithms

WHERE DOES TD(0) CONVERGE TO?

TD(0) with LFA

Input: arbitrary param. vector $\theta_0 \in \mathbf{R}^d$

For $t = 0,1, \ldots,$

$$\delta_t(\theta) = r_t + \gamma\theta^\top \phi(x_{t+1}) - \theta^\top \phi(x_t)$$

$$\theta_{t+1} = \theta_t + \alpha_t \delta_t(\theta_t)\phi(x_t)$$

WHERE DOES TD(0) CONVERGE TO?

TD(0) with LFA

Input: arbitrary param. vector $\theta_0 \in \mathbf{R}^d$

For $t = 0,1, \dots,$

$$\delta_t(\theta) = r_t + \gamma\theta^\top\phi(x_{t+1}) - \theta^\top\phi(x_t)$$

$$\theta_{t+1} = \theta_t + \alpha_t\delta_t(\theta_t)\phi(x_t)$$

In the limit, TD(0) finds a θ^* such that

$$\mathbf{E}[\delta_t(\theta^*)\phi(x_t)] = 0$$

WHERE DOES TD(0) CONVERGE TO?

Idea: given a finite trajectory, approximate the TD fixed point by solving

$$\mathbf{E}[\delta_t(\theta)\phi(x_t)] \approx \frac{1}{T}\sum_{t=1}^{T} \delta_t(\theta)\phi(x_t) = 0$$

WHERE DOES TD(0) CONVERGE TO?

Idea: given a finite trajectory, approximate the TD fixed point by solving

$$\mathbf{E}[\delta_t(\theta)\phi(x_t)] \approx \frac{1}{T}\sum_{t=1}^{T}\delta_t(\theta)\phi(x_t) = 0$$

Equivalently:

$$\frac{1}{T}\sum_{t=1}^{T}\phi(x_t)\big(\phi(x_t) - \gamma\phi(x_{t+1})\big)^{\top}\theta = \frac{1}{T}\sum_{t=1}^{T} r_t\phi(x_t)$$

WHERE DOES TD(0) CONVERGE TO?

This is a linear system

$$A_T \theta = b_T$$

Solution:

Equivalently:

$$\underbrace{\frac{1}{T}\sum_{t=1}^{T} \phi(x_t)\big(\phi(x_t) - \gamma\phi(x_{t+1})\big)^{\top}}_{A_T} \theta = \underbrace{\frac{1}{T}\sum_{t=1}^{T} r_t \phi(x_t)}_{b_T}$$

WHERE DOES TD(0) CONVERGE TO?

This is a linear system

$$A_T \theta = b_T$$

Solution: $\theta_T = A_T^{-1} b_T$

Equivalently:

$$\underbrace{\frac{1}{T}\sum_{t=1}^{T} \phi(x_t)\big(\phi(x_t) - \gamma\phi(x_{t+1})\big)^{\top}}_{A_T}\theta = \underbrace{\frac{1}{T}\sum_{t=1}^{T} r_t \phi(x_t)}_{b_T}$$

LEAST-SQUARES TEMPORAL DIFFERENCE LEARNING (LSTD)

LSTD(0)

Input: trajectory $(x_t, a_t, r_t)_{t=1}^{T}$

$$\theta_T = A_T^{-1} b_T$$

$$\hat{V}_T = \theta_T^\top \phi$$

LEAST-SQUARES TEMPORAL DIFFERENCE LEARNING (LSTD)

LSTD(0)

Input: trajectory $(x_t, a_t, r_t)_{t=1}^{T}$

$$\theta_T = A_T^{-1} b_T$$
$$\hat{V}_T = \theta_T^{\top} \phi$$

☺ converges to same θ^* as TD(0) ☺

☺ no need to set step sizes α_t ☺

LEAST-SQUARES TEMPORAL DIFFERENCE LEARNING (LSTD)

LSTD(0)

Input: trajectory $(x_t, a_t, r_t)_{t=1}^{T}$

$$\theta_T = A_T^{-1} b_T$$
$$\hat{V}_T = \theta_T^\top \phi$$

☺ converges to same θ^* as TD(0) ☺

TD(0): $O(Td)$

☺ no need to set step sizes α_t ☺

☹ computational complexity: $O(Td^2 + d^3)$ ☹

☹ A_T^{-1} may not exist for small T ☹

THE CONVERGENCE OF TD(0) AND LSTD(0)

Theorem

In the limit $T \to \infty$, LSTD(0) and TD(0) both minimize the projected Bellman error

$$L(V) = \mathbf{E}_{x\sim\mu}\left[\left(\Pi_{\phi}[T^{\pi}V(x)] - V(x)\right)^2\right]$$

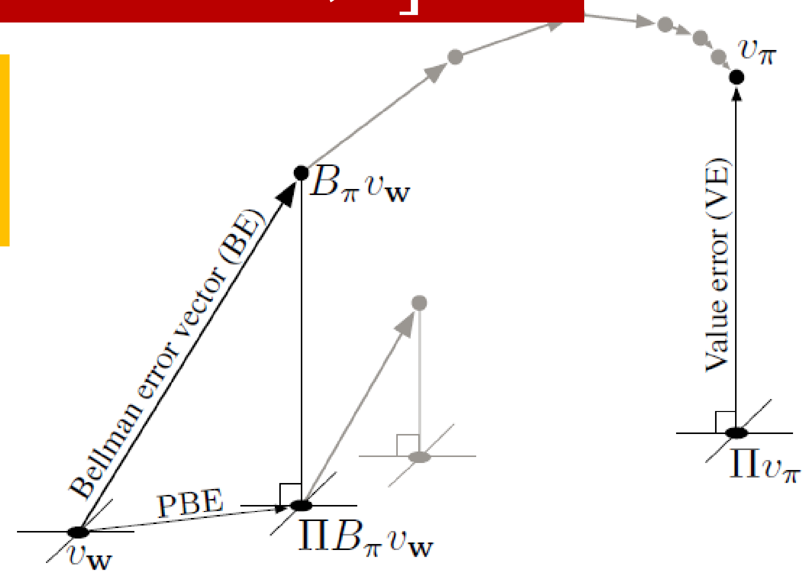
THE CONVERGENCE OF TD(0) AND LSTD(0)

Theorem

In the limit $T \to \infty$, LSTD(0) and TD(0) both minimize the projected Bellman error

$$L(V) = \mathbf{E}_{x\sim\mu}\left[\left(\Pi_\phi[T^\pi V(x)] - V(x)\right)^2\right]$$

Projection onto span of features



The path towards deep RL

1. Least-squares TD
   - Linear function approx.
   - General function approx.
   - Deep Q networks
2. Policy optimization
   - The likelihood-ratio trick
   - The policy gradient theorem
   - Actor-critic algorithms

LEAST-SQUARES TEMPORAL DIFFERENCE LEARNING (LSTD)

LSTD(0)

Input: trajectory $(x_t, a_t, r_t)_{t=1}^{T}$

$$\theta_T = A_T^{-1} b_T$$
$$\hat{V}_T = \theta_T^{\top} \phi$$

☹☹☹

Idea not directly applicable to non-linear function approximation!

☹☹☹

LSTD FOR NON-LINEAR FUNCTION APPROXIMATION?

Can we optimize Bellman error

$$L(\theta) = \mathbf{E}_{x\sim\mu}\left[\left(T^{\pi}V_{\theta}(x) - V_{\theta}(x)\right)^{2}\right]$$

by stochastic gradient descent????

LSTD FOR NON-LINEAR FUNCTION APPROXIMATION?

Can we optimize Bellman error

$$L(\theta) = \mathbf{E}_{x\sim\mu}\left[\left(T^{\pi}V_{\theta}(x) - V_{\theta}(x)\right)^{2}\right]$$

by stochastic gradient descent????

NO!!

Bellman error involves a double expectation:

$$L(\theta) = \mathbf{E}_X[\ell(\theta; X, \mathbf{E}_Y[Y|X])]$$

can't get unbiased gradients!

LSTD FOR NON-LINEAR FUNCTION APPROXIMATION?

Can we optimize Bellman error

$L(\theta) = \mathbf{E}_{x\sim\mu}\left[\left(\ldots \right)^2\right]$

by stochastic g... ?

The infamous "double sampling" issue of RL

NO!!

Bellman error involves a double expectation:

$$L(\theta) = \mathbf{E}_X[\ell(\theta; X, \mathbf{E}_Y[Y|X])]$$

can't get unbiased gradients!

WHEN DO STOCHASTIC UPDATES WORK?

Stochastic optimization for least squares:

$$L(\theta) = \mathbb{E}_{X,Y}[\ell(\theta; X, Y)] = \mathbb{E}_{X,Y}[(X^\top \theta - Y)^2]$$

SGD for linear regression

Input: arbitrary param. vector $\theta_0 \in \mathbf{R}^d$

For $t = 0,1, \dots,$

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha_t \nabla \ell(\theta_t; X_t, Y_t) \\ &= \theta_t + \alpha_t (X_t^\top \theta_t - Y_t) X_t \end{aligned}$$

WHEN DO STOCHASTIC UPDATES WORK?

Stochastic optimization for least squares:

$$L(\theta) = \mathbb{E}_{X,Y}[\ell(\theta; X, Y)] = \mathbb{E}_{X,Y}[(X^\top\theta - Y)^2]$$

SGD for linear regression

Input: arbitrary param. vector $\theta_0 \in \mathbf{R}^d$

For $t = 0, 1, \ldots,$

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha_t \nabla\ell(\theta_t; X_t, Y_t) \\ &= \theta_t + \alpha_t (X_t^\top \theta_t - Y_t) X_t \end{aligned}$$

Stochastic gradient is unbiased estimate of gradient:

$$\mathbb{E}[\nabla\ell(\theta_t; X_t, Y_t)] = \nabla\mathbb{E}[\ell(\theta_t; X_t, Y_t)]$$

THE DOUBLE SAMPLING PROBLEM

Bellman error:

$$L(\theta) = \mathbf{E}_{x\sim\mu}\left[\left(T^{\pi}V_{\theta}(x) - V_{\theta}(x)\right)^2\right]$$

$$= \mathbf{E}_{x\sim\mu}\left[\left(r(x) + \gamma\mathbf{E}_{x'}[V_{\theta}(x')|x] - V_{\theta}(x)\right)^2\right]$$

THE DOUBLE SAMPLING PROBLEM

Bellman error:

$$L(\theta) = \mathbf{E}_{x\sim\mu}\left[\left(T^{\pi}V_\theta(x) - V_\theta(x)\right)^2\right]$$

$$= \mathbf{E}_{x\sim\mu}\left[\left(r(x) + \gamma\mathbf{E}_{x'}[V_\theta(x')|x] - V_\theta(x)\right)^2\right]$$

Idea: replace expectations with samples:

$$\hat{L}_t(\theta) = \left(r(x_t) + \gamma V_\theta(x_{t+1}) - V_\theta(x_t)\right)^2$$

THE DOUBLE SAMPLING PROBLEM

Bellman error:

$$L(\theta) = \mathbf{E}_{x \sim \mu}\left[\left(T^{\pi} V_\theta(x) - V_\theta(x)\right)^2\right]$$

$$= \mathbf{E}_{x \sim \mu}\left[\left(r(x) + \gamma \mathbf{E}_{x'}[V_\theta(x')|x] - V_\theta(x)\right)^2\right]$$

Idea: replace expectations with samples:

$$\hat{L}_t(\theta) = \left(r(x_t) + \gamma V_\theta(x_{t+1}) - V_\theta(x_t)\right)^2$$

TERRIBLE IDEA!!!

$$\mathbf{E}\left[\hat{L}_t(\theta)\right] = L(\theta) + \mathrm{Var}[r(x_t) + \gamma V_\theta(x_{t+1})]$$

$$\mathbf{E}\left[\nabla \hat{L}_t(\theta)\right] = \nabla L(\theta) + (\text{bias})$$

TACKLING DOUBLE SAMPLING

- Saddle-point optimization:

$$\min_{\theta} \mathbf{E}[f(\theta; X, \mathbf{E}[Y|X])^2]$$

TACKLING DOUBLE SAMPLING

f: linear function

- Saddle-point optimization:

$$\min_{\theta} \mathbf{E}[f(\theta; X, \mathbf{E}[Y|X])^2]$$

TACKLING DOUBLE SAMPLING

f: linear function

- Saddle-point optimization:

$$\min_{\theta} \mathbf{E}[f(\theta; X, \mathbf{E}[Y|X])^2] =$$

$$\min_{\theta} \max_{z} \mathbf{E}[z(X,Y) \cdot f(\theta; X, \mathbf{E}[Y|X])] - \mathbf{E}[z^2(X,Y)]$$

TACKLING DOUBLE SAMPLING

f: linear function

- Saddle-point optimization:

$$\min_{\theta} \mathbf{E}[f(\theta; X, \mathbf{E}[Y|X])^2] =$$

$$\min_{\theta} \max_{z} \mathbf{E}[z(X,Y) \cdot f(\theta; X, \mathbf{E}[Y|X])] - \mathbf{E}[z^2(X,Y)]$$

$\Rightarrow$ "modified Bellman residual" (Antos et al. 2008), "Gradient TD" methods (Sutton et al. 2009), SBEED (Dai et al., 2018)

TACKLING DOUBLE SAMPLING

f: linear function

- Saddle-point optimization:

$$\min_\theta \mathbf{E}[f(\theta; X, \mathbf{E}[Y|X])^2] =$$

$$\min_\theta \max_z \mathbf{E}[z(X,Y) \cdot f(\theta; X, \mathbf{E}[Y|X])] - \mathbf{E}[z^2(X,Y)]$$

$\Rightarrow$ "modified Bellman residual" (Antos et al. 2008), "Gradient TD" methods (Sutton et al. 2009), SBEED (Dai et al., 2018)

- Iterative optimization schemes

TACKLING DOUBLE SAMPLING

f: linear function

- Saddle-point optimization:

$$\min_{\theta} \mathbf{E}[f(\theta; X, \mathbf{E}[Y|X])^2] =$$

$$\min_{\theta} \max_{z} \mathbf{E}[z(X,Y) \cdot f(\theta; X, \mathbf{E}[Y|X])] - \mathbf{E}[z^2(X,Y)]$$

$\Rightarrow$ “modified Bellman residual” (Antos et al. 2008), “Gradient TD” methods (Sutton et al. 2009), SBEED (Dai et al., 2018)

- Iterative optimization schemes

FITTED POLICY EVALUATION

Idea: compute sequence of value functions by minimizing

$$L_n(\hat{V}; \hat{V}_k) = \frac{1}{n}\sum_{t=1}^{n}\Big(r_t + \gamma \hat{V}_k(x_{t+1}) - \hat{V}(x_t)\Big)^2$$

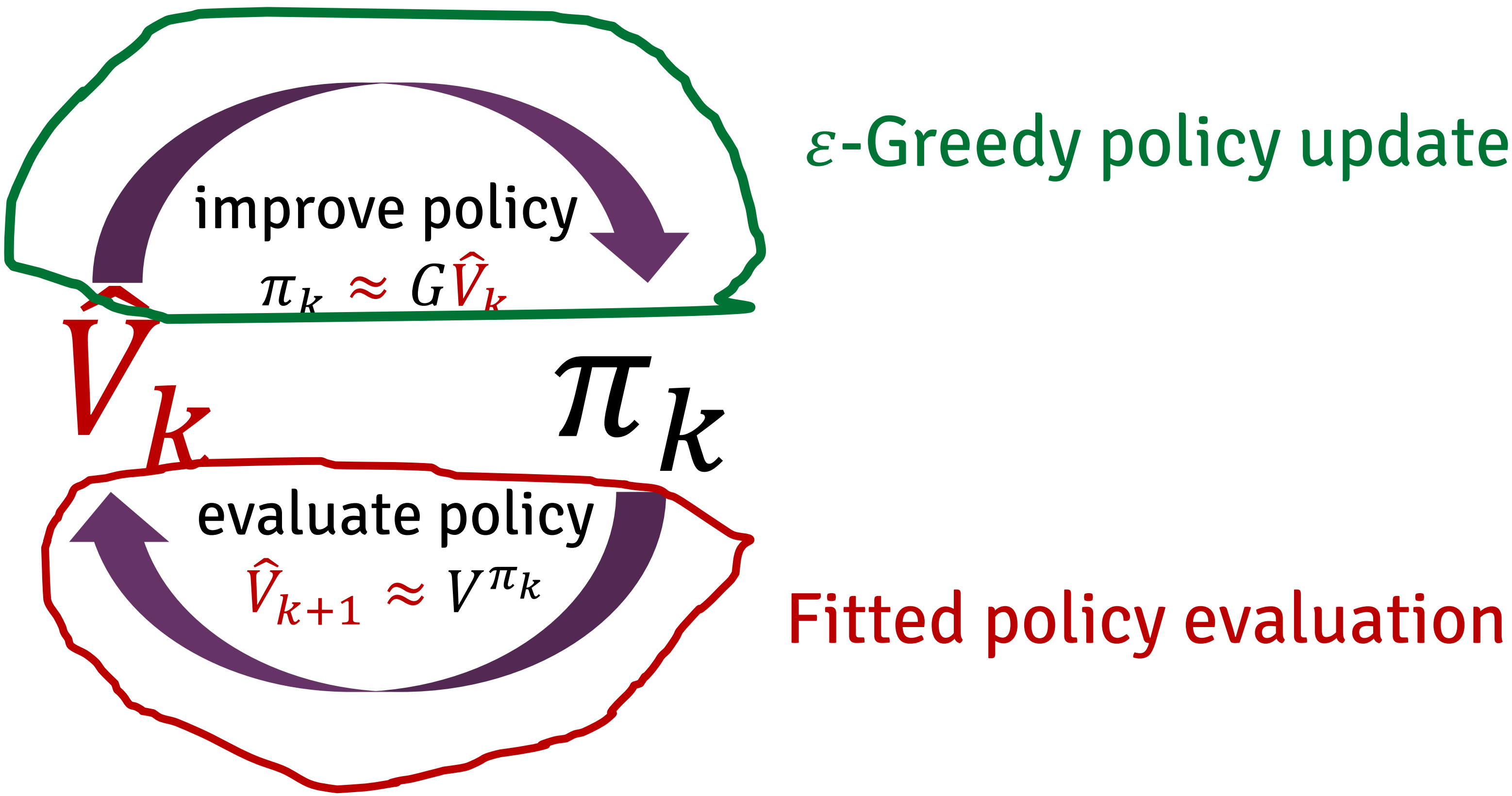
FITTED POLICY EVALUATION

Idea: compute sequence of value functions by minimizing

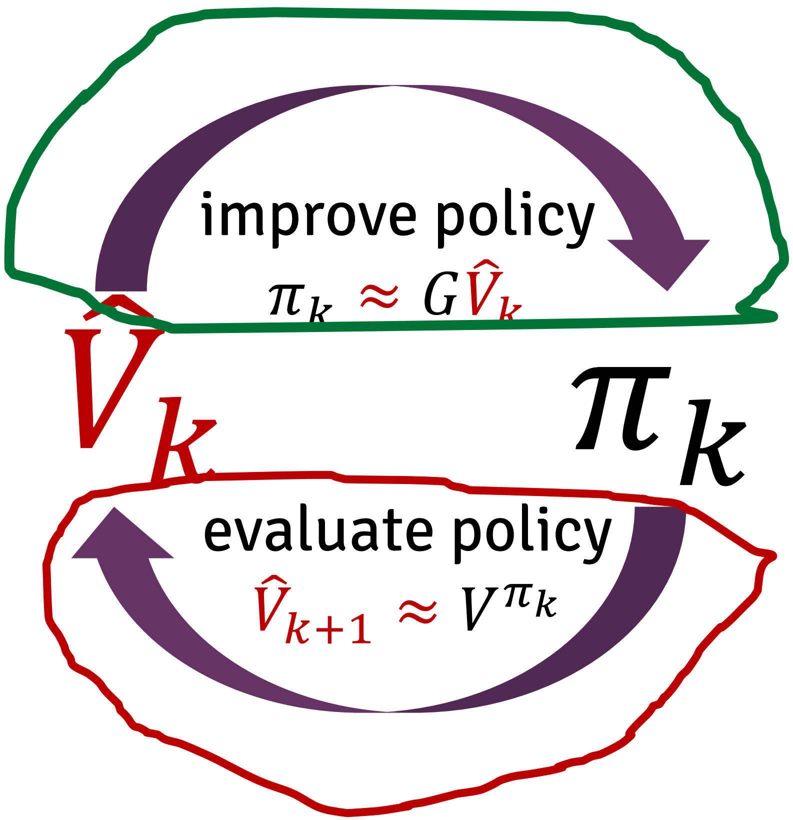
$$L_n(\hat{V}; \hat{V}_k) = \frac{1}{n}\sum_{t=1}^{n}\Big(\underbrace{r_t + \gamma \hat{V}_k(x_{t+1})}_{\text{Target}} - \underbrace{\hat{V}(x_t)}_{\text{Free variable}}\Big)^2$$

This can be finally treated as a regression problem & solved by SGD!

FITTED POLICY ITERATION



FITTED POLICY ITERATION

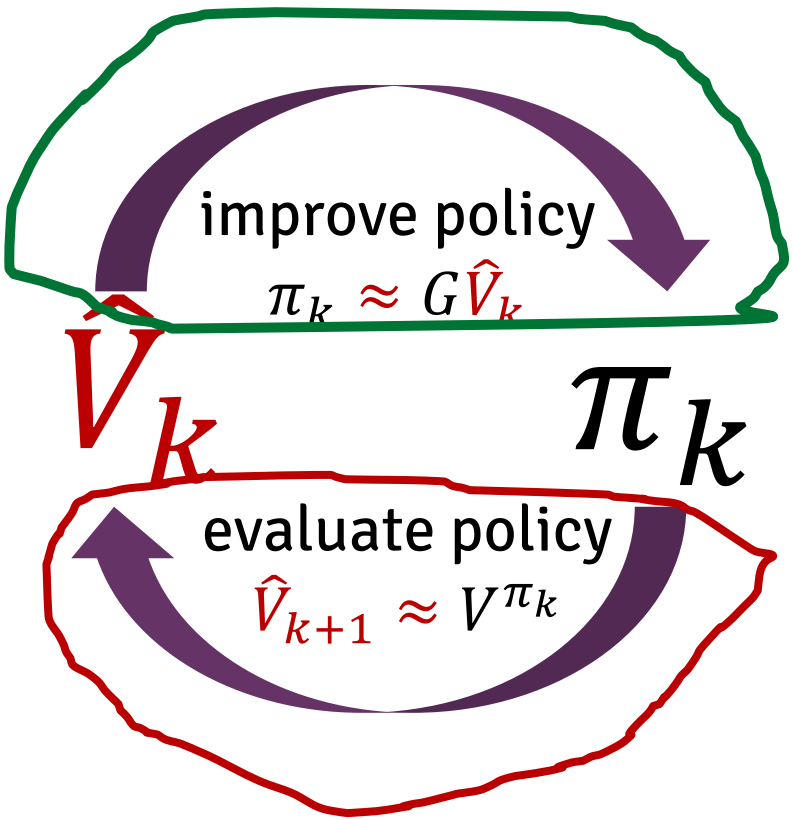


ε-Greedy policy update

Computing policy needs model of P… better use Q-functions!

Fitted policy evaluation

FITTED POLICY ITERATION



ε-Greedy policy update

Computing policy needs model of P… better use Q-functions!

Fitted policy evaluation

This may take too many iterations…

FITTED VALUE ITERATION

Idea: compute sequence of Q-functions by minimizing

$$L_n(\hat{Q}; \hat{Q}_k) = \frac{1}{n}\sum_{t=1}^{n}\Bigg(\underbrace{r_t + \gamma \max_a \hat{Q}_k(x_{t+1}, a)}_{\text{Target}} - \underbrace{\hat{Q}(x_t, a_t)}_{\text{Free variable}}\Bigg)^2$$

FITTED VALUE ITERATION

Idea: compute sequence of Q-functions by minimizing

$$L_n(\hat{Q};\hat{Q}_k)=\frac{1}{n}\sum_{t=1}^{n}\Big(\underbrace{r_t+\gamma\max_a\hat{Q}_k(x_{t+1},a)}_{\text{Target}}-\underbrace{\hat{Q}(x_t,a_t)}_{\text{Free variable}}\Big)^2$$

Familiar?

FITTED VALUE ITERATION

Idea: compute sequence of Q-functions by minimizing

$$L_n(\hat{Q}; \hat{Q}_k) = \frac{1}{n}\sum_{t=1}^{n}\Big(\underbrace{r_t + \gamma \max_a \hat{Q}_k(x_{t+1}, a)}_{\text{Target}} - \underbrace{\hat{Q}(x_t, a_t)}_{\text{Free variable}}\Big)^2$$

Familiar? Q-learning!

Idea: Let's try to
- directly learn about Q^*, and
- improve the policy on the fly!

FITTED VALUE ITERATION

Fitted value iteration

Input: function space F, $\hat{Q}_0 \in F$

For $k = 0,1, \dots,$

- $\pi_k = G_\varepsilon \hat{Q}_k$
- generate trajectory
$$(x_t, a_t, r_t)_{t=1}^{n} \sim \pi_k$$
- compute
$$\hat{Q}_{k+1} = \operatorname*{argmin}_{\hat{Q} \in F} L_n\left(\hat{Q}; \hat{Q}_k\right)$$

FITTED VALUE ITERATION

Fitted value iteration

Input: function space F, $\hat{Q}_0 \in F$

For $k = 0,1, \dots,$

- $\pi_k = G_\varepsilon \hat{Q}_k$
- generate trajectory

$$(x_t, a_t, r_t)_{t=1}^{n} \sim \pi_k$$

- compute

$$\hat{Q}_{k+1} = \underset{\hat{Q} \in F}{\operatorname{argmin}}\, L_n\big(\hat{Q}; \hat{Q}_k\big)$$

Convergence can be guaranteed!

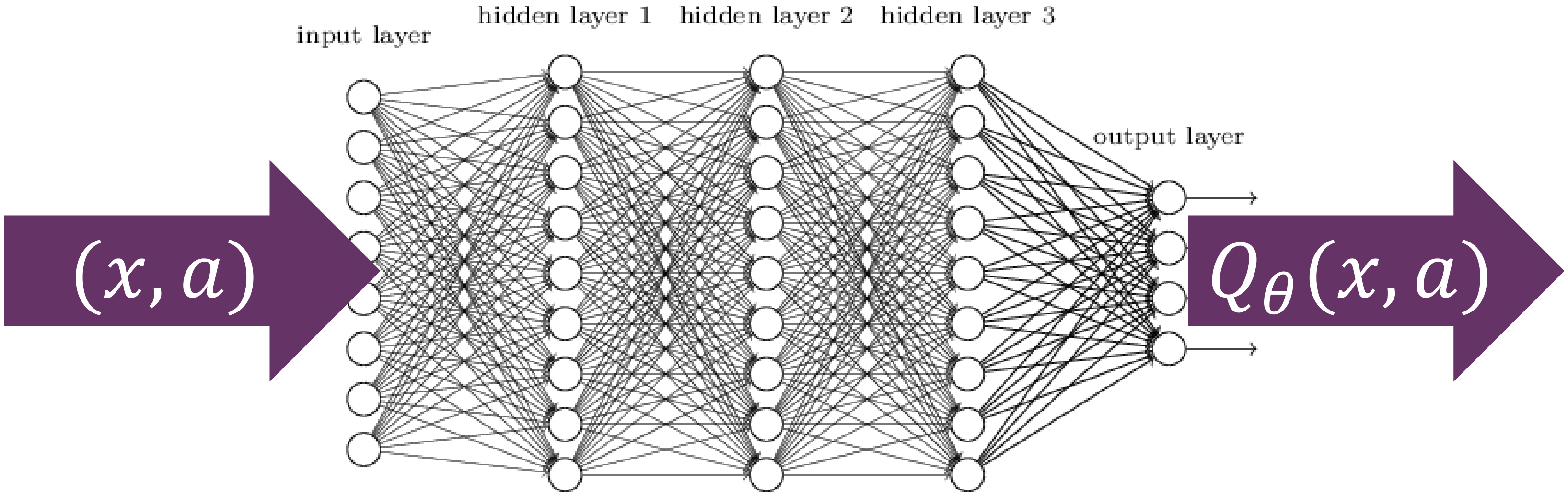
under very technical assumptions…

The path towards deep RL

1. Least-squares TD
   - Linear function approx.
   - General function approx.
   - Deep Q networks
2. Policy optimization
   - The likelihood-ratio trick
   - The policy gradient theorem
   - Actor-critic algorithms

DEEP Q NETWORKS

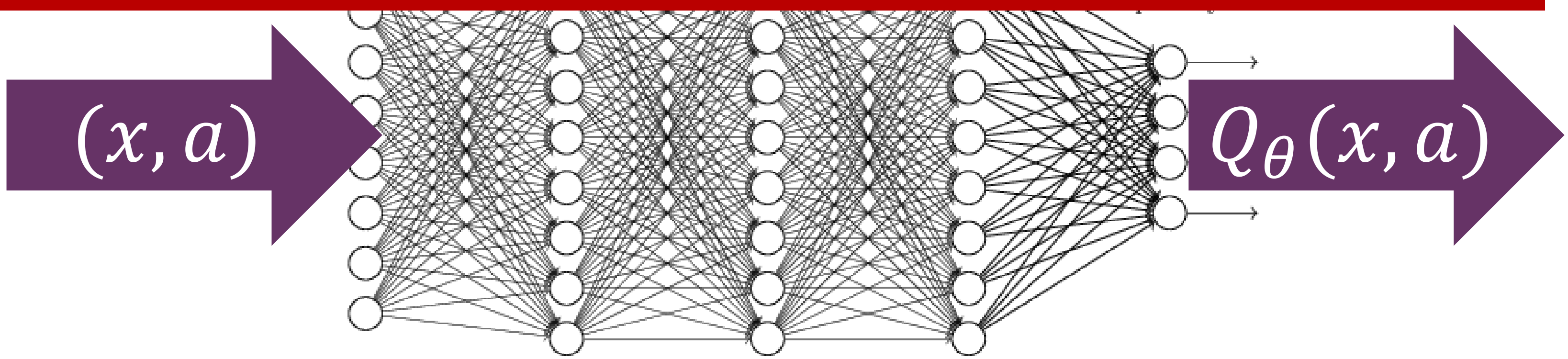
Parametrize Q-function by a deep neural net



DEEP Q NETWORKS

Minimize the loss

$$\mathbf{E}_{(X,A,R,X')\sim D}\left[\left(R+\gamma\max_{b}Q_{\theta_k}(X',b)-Q_{\theta}(X,A)\right)^2\right]$$



DEEP Q NETWORKS

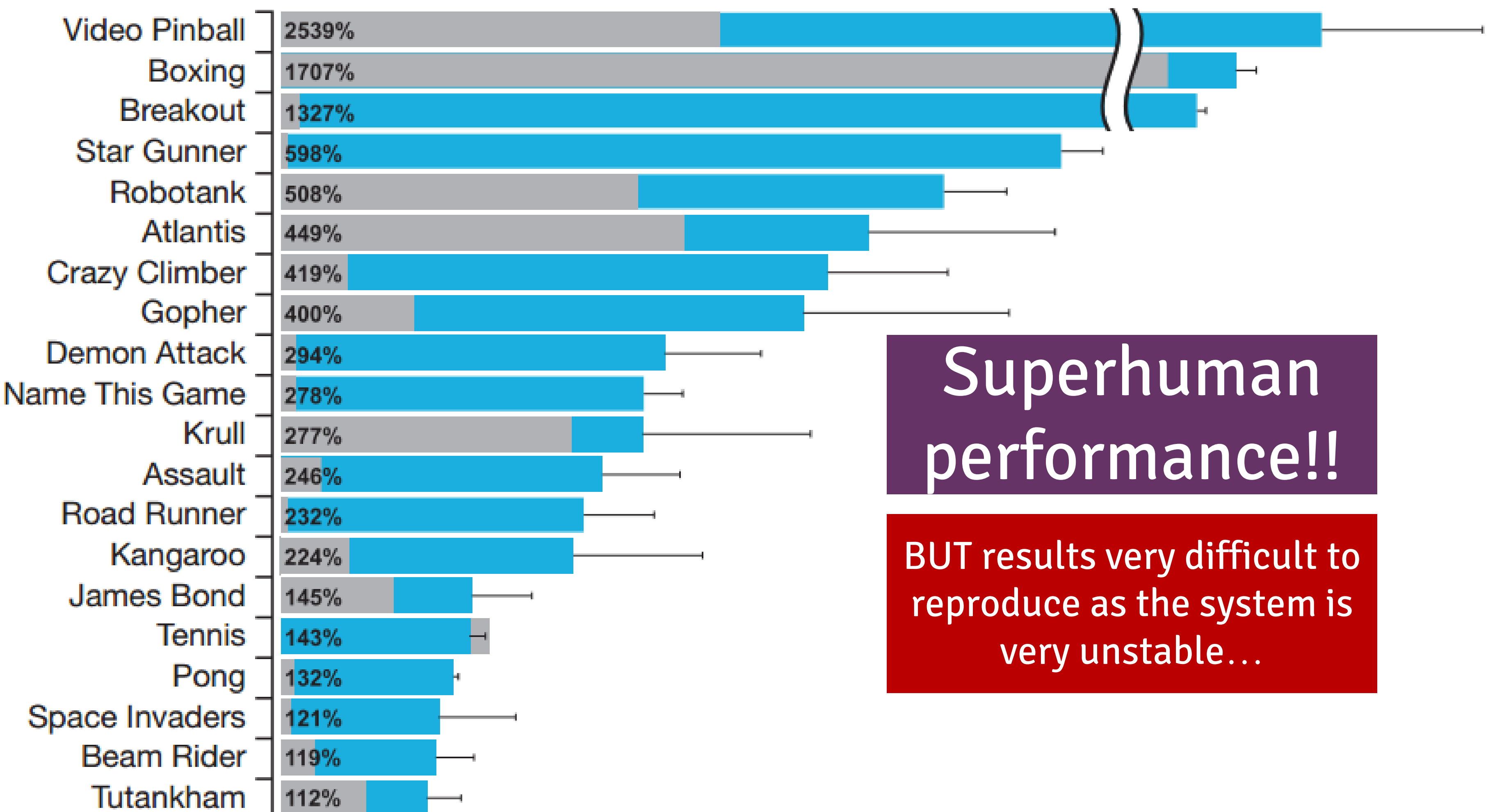
Minimize the loss

$$\mathbf{E}_{(X,A,R,X')\sim D}\left[\left(R+\gamma\max_{b} Q_{\theta_k}(X',b)-Q_\theta(X,A)\right)^2\right]$$

+ training tricks:

- Store transitions (x,a,r,x') in replay buffer D to break dependence on recent samples
- Compute small updates by mini-batch stochastic gradient descent
- Use an older parameter vector θ_{k-m} in target to avoid oscillations
- …

DEEP Q NETWORKS FOR PLAYING ATARI



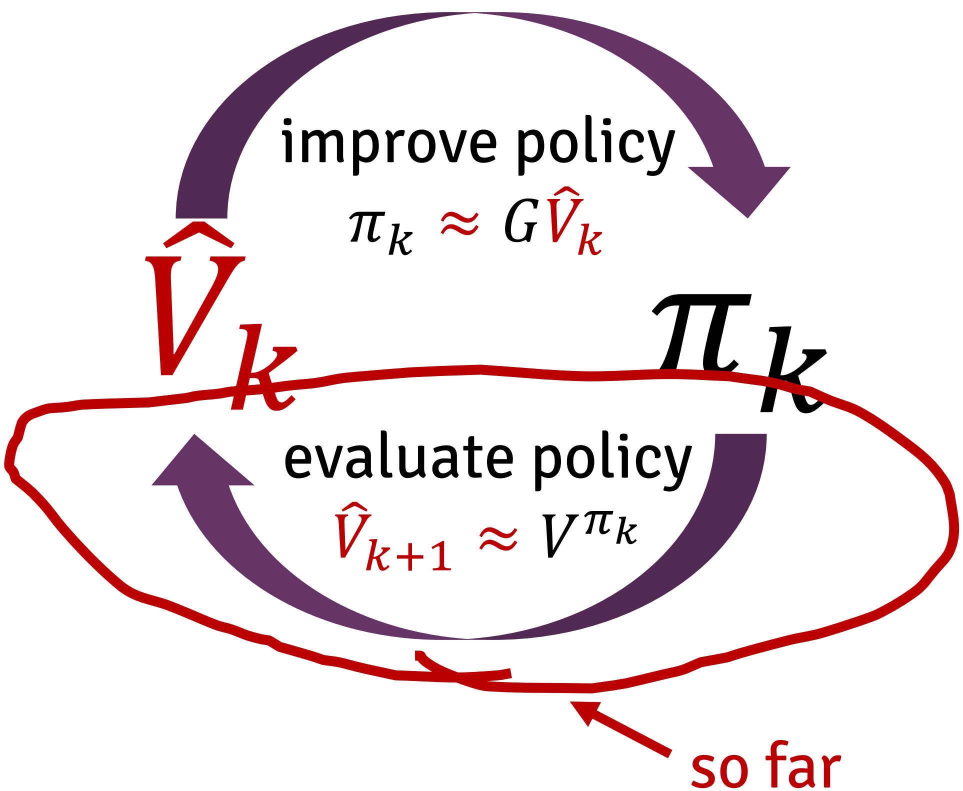
Superhuman performance!!

BUT results very difficult to reproduce as the system is very unstable…

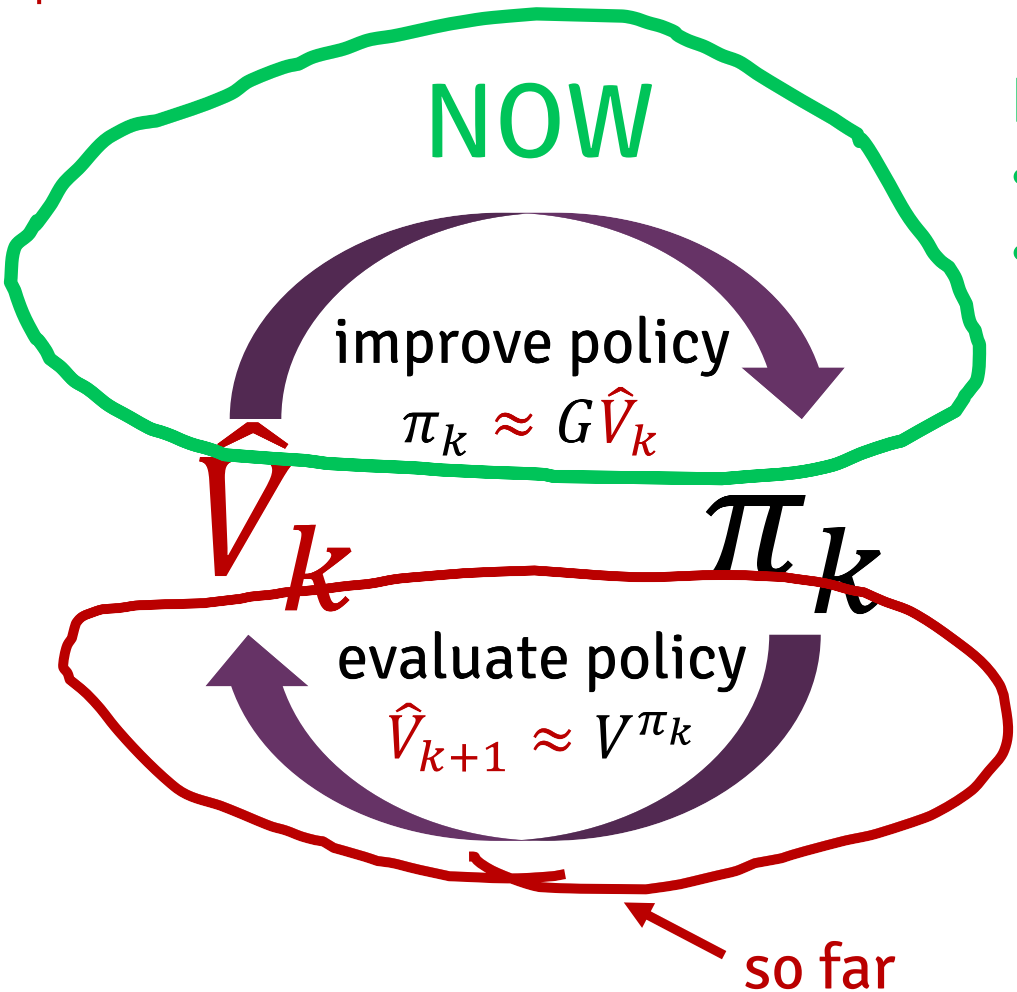
The path towards deep RL

1. Least-squares TD
   - Linear function approx.
   - General function approx.
   - Deep Q networks
2. Policy optimization
   - The likelihood-ratio trick
   - The policy gradient theorem
   - Actor-critic algorithms

DIRECT POLICY OPTIMIZATION



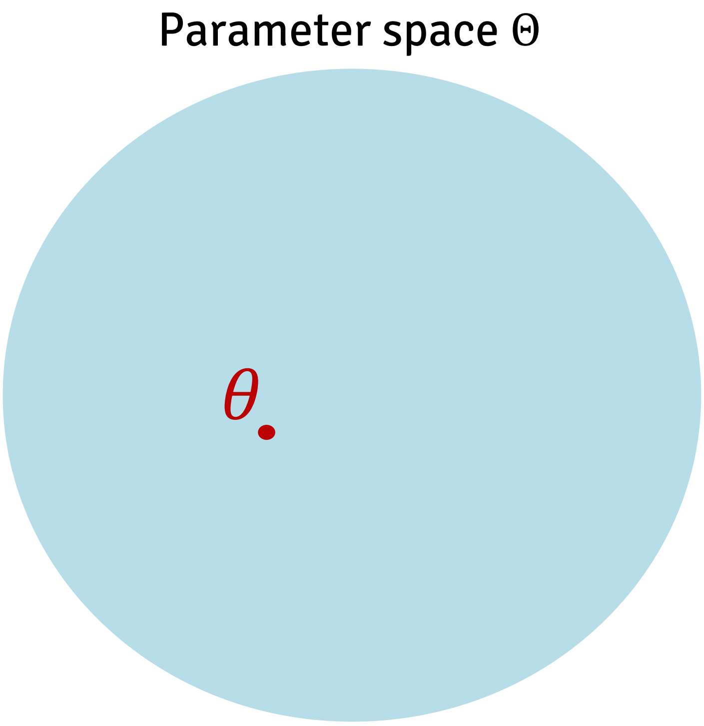
DIRECT POLICY OPTIMIZATION



Idea:

- Parametrize policy π_θ
- Directly improve performance by searching for better parameters θ

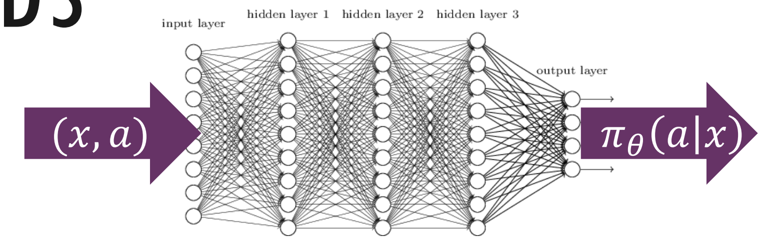
POLICY GRADIENT METHODS



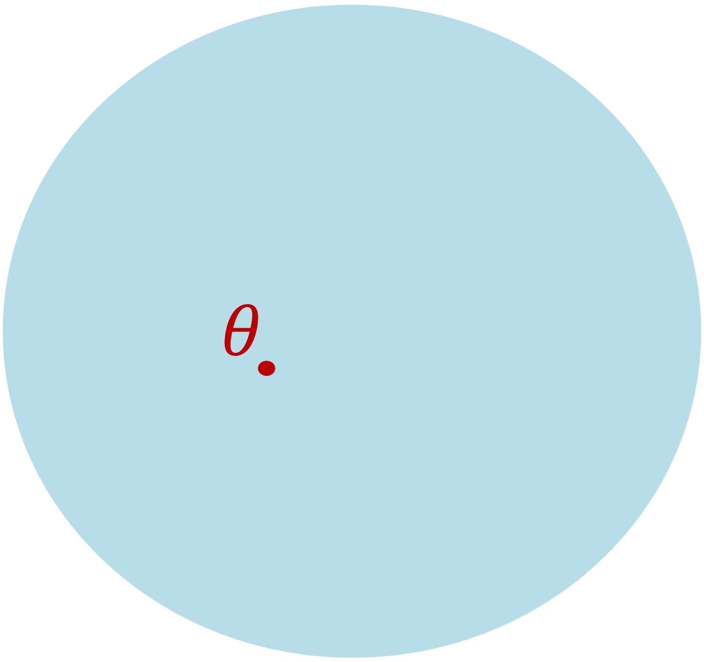
- Construct mapping

$$\theta \mapsto \pi_\theta$$

POLICY GRADIENT METHODS



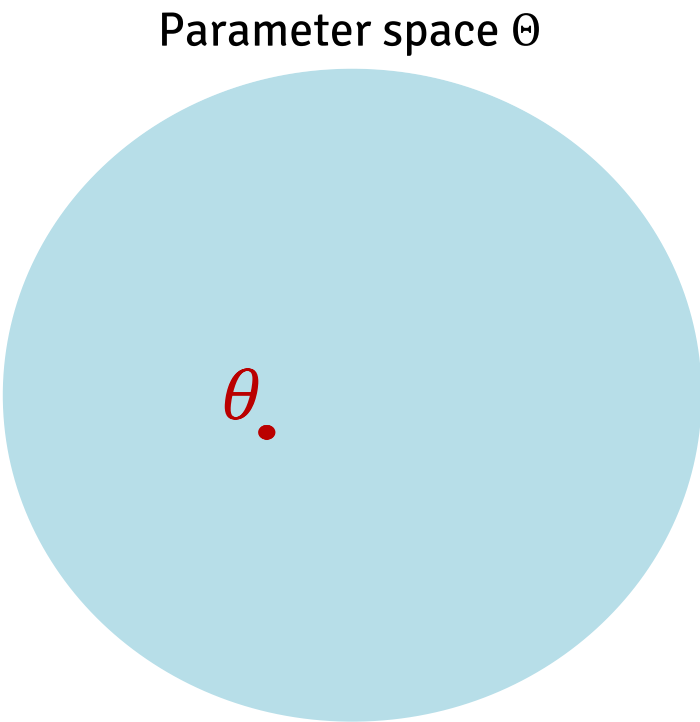
Parameter space Θ

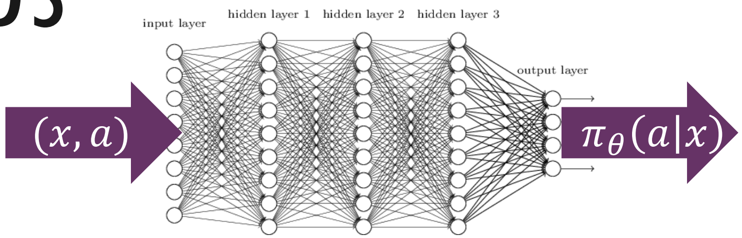


- Construct mapping

$$\theta \mapsto \pi_\theta$$

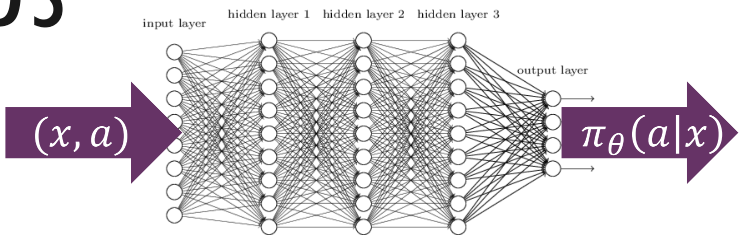
POLICY GRADIENT METHODS

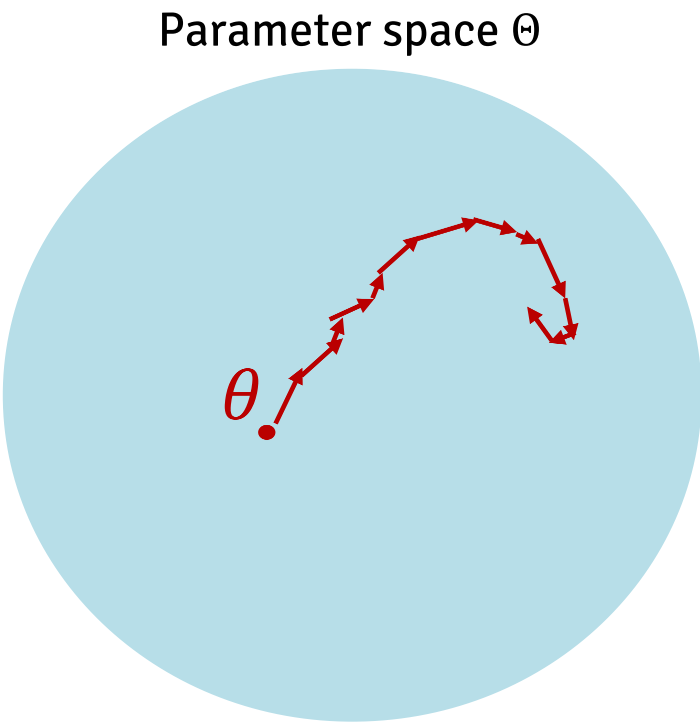




- Construct mapping
$$\theta \mapsto \pi_\theta$$
- Define objective function:
$$\rho(\theta) = V^{\pi_\theta}(x_0)$$

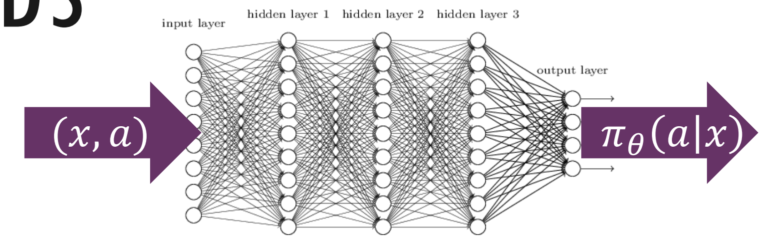
POLICY GRADIENT METHODS



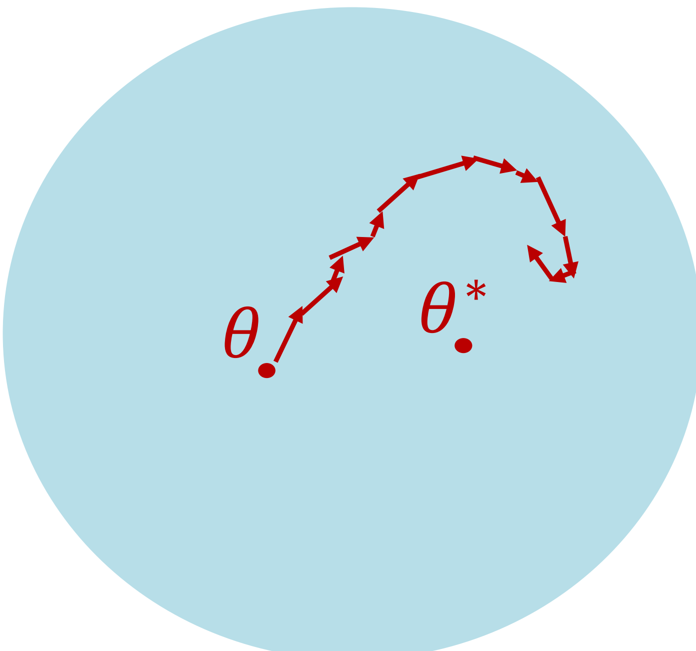


- Construct mapping
$$\theta \mapsto \pi_\theta$$
- Define objective function:
$$\rho(\theta) = V^{\pi_\theta}(x_0)$$
- Update parameters by gradient ascent:
$$\theta_{k+1} = \theta_k + \alpha_k \nabla_\theta \rho(\theta_k)$$

POLICY GRADIENT METHODS



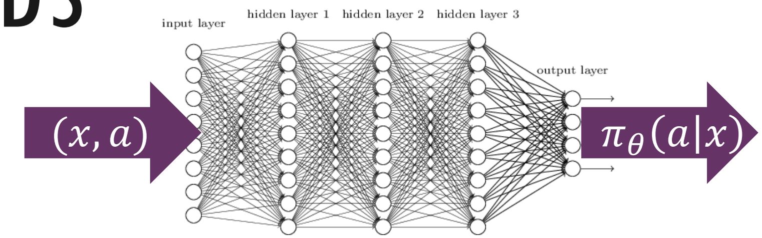
Parameter space Θ



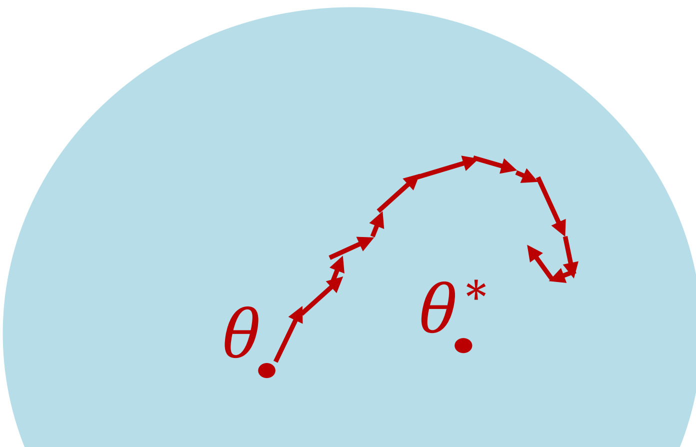
- Construct mapping
$$\theta \mapsto \pi_\theta$$
- Define objective function:
$$\rho(\theta) = V^{\pi_\theta}(x_0)$$
- Update parameters by gradient ascent:
$$\theta_{k+1} = \theta_k + \alpha_k \nabla_\theta \rho(\theta_k)$$

… and hope for convergence

POLICY GRADIENT METHODS



Parameter space Θ



How can we estimate the gradients?

- Construct mapping
$$\theta \mapsto \pi_\theta$$
- Define objective function:
$$\rho(\theta) = V^{\pi_\theta}(x_0)$$
- Update parameters by gradient ascent:
$$\theta_{k+1} = \theta_k + \alpha_k \nabla_\theta \rho(\theta_k)$$

… and hope for convergence

The path towards deep RL

1. Least-squares TD
   - Linear function approx.
   - General function approx.
   - Deep Q networks
2. Policy optimization
   - The likelihood-ratio trick
   - The policy gradient theorem
   - Actor-critic algorithms

SOME DEFINITIONS

Consider a special case:

- Episodic MDP with episode terminating in round T
- No discounting: $\gamma = 1$

Let π_θ be a stochastic policy with

$$\pi_\theta(a|x) = \mathbf{P}[a_t = a|x_t = x]$$

SOME DEFINITIONS

Consider a special case:

- Episodic MDP with episode terminating in round T
- No discounting: $\gamma = 1$

Let π_θ be a stochastic policy with

$$\pi_\theta(a|x) = \mathbf{P}[a_t = a|x_t = x]$$

For a trajectory $\tau_T = (x_1, a_1, \dots, x_T)$, define

$$p_\theta(\tau_T) = \mathbf{P}\,[\tau \text{ is generated by } \pi_\theta]$$

and $R(\tau_T) = \sum_{x_t, a_t \in \tau_T} r(x_t, a_t)$

Notice that

$$\rho(\theta) = V^{\pi_\theta}(x_0) = \mathbf{E}_{\tau_T \sim p_\theta}[R(\tau_T)]$$

THE LIKELIHOOD RATIO TRICK

Theorem

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[R(\tau_T) \nabla_\theta \log p_\theta(\tau_T)]$$

THE LIKELIHOOD RATIO TRICK

Theorem

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[R(\tau_T)\nabla_\theta \log p_\theta(\tau_T)]$$

Proof:

$$\nabla \rho(\theta) = \nabla \mathbf{E}_{\tau \sim p_\theta}[R(\tau)] = \nabla \left(\sum_\tau p_\theta(\tau) R(\tau) \right)$$

THE LIKELIHOOD RATIO TRICK

Theorem

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[R(\tau_T)\nabla_\theta \log p_\theta(\tau_T)]$$

Proof:

$$\nabla \rho(\theta) = \nabla \mathbf{E}_{\tau \sim p_\theta}[R(\tau)] = \nabla \left(\sum_\tau p_\theta(\tau) R(\tau) \right)$$

$$= \sum_\tau \nabla p_\theta(\tau) R(\tau)$$

THE LIKELIHOOD RATIO TRICK

Theorem

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[R(\tau_T)\nabla_\theta \log p_\theta(\tau_T)]$$

Proof:

$$\nabla \rho(\theta) = \nabla \mathbf{E}_{\tau \sim p_\theta}[R(\tau)] = \nabla \left(\sum_\tau p_\theta(\tau) R(\tau) \right)$$

$$= \sum_\tau \nabla p_\theta(\tau) R(\tau) = \sum_\tau p_\theta(\tau) \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)} R(\tau)$$

THE LIKELIHOOD RATIO TRICK

Theorem

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[R(\tau_T) \nabla_\theta \log p_\theta(\tau_T)]$$

Proof:

$$\nabla \rho(\theta) = \nabla \mathbf{E}_{\tau \sim p_\theta}[R(\tau)] = \nabla \left(\sum_\tau p_\theta(\tau) R(\tau) \right)$$

$$= \sum_\tau \nabla p_\theta(\tau) R(\tau) = \sum_\tau p_\theta(\tau) \frac{\nabla p_\theta(\tau)}{p_\theta(\tau)} R(\tau)$$

$$= \sum_\tau p_\theta(\tau) \nabla \log p_\theta(\tau) \, R(\tau)$$

$\nabla \log f = \nabla f / f$ for $f > 0$

HOW DO WE COMPUTE $\nabla_\theta \log p_\theta(\tau_T)$??

Observe that

$$p_\theta(\tau_T) = p_\theta(\tau_{T-1})P(x_T|x_{T-1}, a_{T-1})\pi_\theta(a_{T-1}|x_{T-1})$$

HOW DO WE COMPUTE $\nabla_\theta \log p_\theta(\tau_T)$??

Observe that

$$p_\theta(\tau_T) = p_\theta(\tau_{T-1}) P(x_T | x_{T-1}, a_{T-1}) \pi_\theta(a_{T-1} | x_{T-1})$$

$$= p(x_0) \prod_{t=0}^{T-1} P(x_{t+1} | x_t, a_t) \pi_\theta(a_t | x_t)$$

HOW DO WE COMPUTE $\nabla_\theta \log p_\theta(\tau_T)$??

Observe that

$$p_\theta(\tau_T) = p_\theta(\tau_{T-1}) P(x_T | x_{T-1}, a_{T-1}) \pi_\theta(a_{T-1} | x_{T-1})$$
$$= p(x_0) \prod_{t=0}^{T-1} P(x_{t+1} | x_t, a_t) \pi_\theta(a_t | x_t)$$

So we have

$$\log p_\theta(\tau_T) = \log p(x_0) + \sum_{t=0}^{T-1} \log P(x_{t+1} | x_t, a_t)$$
$$+ \sum_{t=0}^{T-1} \log \pi_\theta(a_t | x_t)$$

HOW DO WE COMPUTE $\nabla_\theta \log p_\theta(\tau_T)$??

Observe that

$$\begin{aligned} p_\theta(\tau_T) &= p_\theta(\tau_{T-1}) P(x_T | x_{T-1}, a_{T-1}) \pi_\theta(a_{T-1} | x_{T-1}) \\ &= p(x_0) \prod_{t=0}^{T-1} P(x_{t+1} | x_t, a_t) \pi_\theta(a_t | x_t) \end{aligned}$$

So we have

$$\begin{aligned} \log p_\theta(\tau_T) = \log p(x_0) &+ \sum_{t=0}^{T-1} \log P(x_{t+1} | x_t, a_t) \\ &+ \sum_{t=0}^{T-1} \log \pi_\theta(a_t | x_t) \end{aligned}$$

Only this part depends on θ!!

THE LIKELIHOOD-RATIO GRADIENT

The gradient then becomes

$$\nabla_\theta \log p_\theta(\tau_T) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t | x_t)$$

THE LIKELIHOOD-RATIO GRADIENT

The gradient then becomes

$$\nabla_\theta \log p_\theta(\tau_T) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t | x_t)$$

Theorem

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[R(\tau_T) \nabla_\theta \log p_\theta(\tau_T)]$$

THE LIKELIHOOD-RATIO GRADIENT

The gradient then becomes

$$\nabla_\theta \log p_\theta(\tau_T) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|x_t)$$

Theorem

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[R(\tau_T) \nabla_\theta \log p_\theta(\tau_T)]$$

$$= \mathbf{E}_{\tau_T \sim p_\theta}\left[R(\tau_T) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|x_t)\right]$$

WHY IS THIS GOOD?

Theorem

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}\left[R(\tau_T)\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t | x_t)\right]$$

Gradient can be written as an expectation!!!!
Can be used for SGD!!!

REINFORCE: A STOCHASTIC POLICY GRADIENT ALGORITHM

REINFORCE

Input: arbitrary initial θ_0

For $k = 0,1, \ldots$

- Draw a sample trajectory $(x_t, a_t, r_t)_{t=1}^{T}$
- Let $R_k = \sum_{t=1}^{T} r_t$
- Let $g_k = R_k \sum_{t=1}^{T} \nabla_\theta \log \pi_{\theta_k}(a_t | x_t)$
- Update $\theta_{k+1} = \theta_k + \alpha_k g_k$

REINFORCE: A STOCHASTIC POLICY GRADIENT ALGORITHM

REINFORCE

Input: arbitrary in

For $k = 0,1,\ldots$

- Draw a sample trajectory $(x_t, a_t, r_t)_{t=1}^{T}$
- Let $R_k = \sum_{t=1}^{T} r_t$
- Let $g_k = R_k \sum_{t=1}^{T} \nabla_\theta \log \pi_{\theta_k}(a_t | x_t)$
- Update $\theta_{k+1} = \theta_k + \alpha_k g_k$

$$\mathbf{E}[g_k] = \nabla_\theta \rho(\theta_k)$$

$\Rightarrow$ this is a stochastic gradient ascent algorithm

$\Rightarrow$ converges to local optimum

EXAMPLE: SOFTMAX POLICY

Define a feature map $\phi: X \times A \to \mathbb{R}^d$

Softmax policy:

$$\pi_\theta(a|x) = \frac{\exp(\langle\theta, \phi(x,a)\rangle)}{\sum_{a'} \exp(\langle\theta, \phi(x,a')\rangle)} = \frac{\exp(\langle\theta, \phi(x,a)\rangle)}{Z_\theta}$$

Score function:

$$\nabla_\theta \log \pi_\theta(a|x) = \nabla_\theta \langle\theta, \phi(x,a)\rangle - \nabla_\theta \log Z_\theta$$

$$= \phi(x,a) - \frac{\nabla_\theta Z_\theta}{Z_\theta} = \phi(x,a) - \frac{\sum_{a'} \nabla_\theta \exp(\langle\theta, \phi(x,a')\rangle)}{Z_\theta}$$

$$= \phi(x,a) - \frac{\sum_{a'} \phi(x,a') \exp(\langle\theta, \phi(x,a')\rangle)}{Z_\theta}$$

$$= \phi(x,a) - \sum_{a'} \pi_\theta(a'|x)\phi(x,a')$$

ON THE LIKELIHOOD RATIO TRICK

☺ Gives unbiased gradient estimates ☺

☺ Also works for black-box optimization, MCMC,… ☺

ON THE LIKELIHOOD RATIO TRICK

☺ Gives unbiased gradient estimates ☺

☺ Also works for black-box optimization, MCMC,… ☺

☹ HUGE variance ☹

☹ Doesn't make use of the Bellman equations ☹

REDUCING THE VARIANCE

Adding a baseline preserves expectation, but may reduce variance:

Theorem

For any $b \in \mathbb{R}$,

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[(R(\tau_T) - b)\nabla_\theta \log p_\theta(\tau_T)]$$

REDUCING THE VARIANCE

Adding a baseline preserves expectation, but may reduce variance:

Theorem

For any $b \in \mathbb{R}$,

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[(R(\tau_T) - b)\nabla_\theta \log p_\theta(\tau_T)]$$

Proof:

$$\nabla \rho(\theta) = \nabla \mathbf{E}_{\tau \sim p_\theta}[R(\tau) - b]$$

REDUCING THE VARIANCE

Adding a baseline preserves expectation, but may reduce variance:

Theorem

For any $b \in \mathbb{R}$,

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[(R(\tau_T) - b)\nabla_\theta \log p_\theta(\tau_T)]$$

Proof:

$$\begin{aligned}\nabla \rho(\theta) &= \nabla \mathbf{E}_{\tau \sim p_\theta}[R(\tau) - b] \\ &= \nabla \mathbf{E}_{\tau \sim p_\theta}[R(\tau)] - \nabla \mathbf{E}_{\tau \sim p_\theta}[b]\end{aligned}$$

REDUCING THE VARIANCE

Adding a baseline preserves expectation, but may reduce variance:

Theorem

For any $b \in \mathbb{R}$,

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[(R(\tau_T) - b)\nabla_\theta \log p_\theta(\tau_T)]$$

Proof:

$$\begin{aligned}\nabla\rho(\theta) &= \nabla\mathbf{E}_{\tau\sim p_\theta}[R(\tau) - b] \\ &= \nabla\mathbf{E}_{\tau\sim p_\theta}[R(\tau)] - \nabla\mathbf{E}_{\tau\sim p_\theta}[b] \\ &= \nabla\mathbf{E}_{\tau\sim p_\theta}[R(\tau)] - \nabla b\end{aligned}$$

REDUCING THE VARIANCE

Adding a baseline preserves expectation, but may reduce variance:

Theorem

For any $b \in \mathbb{R}$,

$$\nabla_\theta \rho(\theta) = \mathbf{E}_{\tau_T \sim p_\theta}[(R(\tau_T) - b)\nabla_\theta \log p_\theta(\tau_T)]$$

Proof:

$$\begin{aligned}
\nabla\rho(\theta) &= \nabla\mathbf{E}_{\tau\sim p_\theta}[R(\tau) - b] \\
&= \nabla\mathbf{E}_{\tau\sim p_\theta}[R(\tau)] - \nabla\mathbf{E}_{\tau\sim p_\theta}[b] \\
&= \nabla\mathbf{E}_{\tau\sim p_\theta}[R(\tau)] - \nabla b \\
&= \nabla\mathbf{E}_{\tau\sim p_\theta}[R(\tau)]
\end{aligned}$$

The path towards deep RL

1. Least-squares TD
   - Linear function approx.
   - General function approx.
   - Deep Q networks
2. Policy optimization
   - The likelihood-ratio trick
   - The policy gradient theorem
   - Actor-critic algorithms

MAKING USE OF THE BELLMAN EQS: THE POLICY GRADIENT THEOREM

Theorem

For any function $b: X \to R$, we have

$$\nabla_\theta \rho(\theta) = \sum_{t=1}^{T} \mathbf{E}_{\tau_T \sim p_\theta}\left[\nabla_\theta \log \pi_\theta(a_t|x_t)\left(Q^{\pi_\theta}(x_t, a_t) - b(x_t)\right)\right]$$

MAKING USE OF THE BELLMAN EQS: THE POLICY GRADIENT THEOREM

Theorem

For any function $b: X \to R$, we have

$$\nabla_\theta \rho(\theta) = \sum_{t=1}^{T} \mathbf{E}_{\tau_T \sim p_\theta}\left[\nabla_\theta \log \pi_\theta(a_t | x_t) \left(Q^{\pi_\theta}(x_t, a_t) - b(x_t)\right)\right]$$

Likelihood-ratio gradient on steroids:

- Replaces total rewards by Q-function
- State-dependent baseline
- Also works beyond episodic case

REINFORCE WITH THE PG THEOREM

REINFORCE v2

Input: arbitrary initial θ_0

For $k = 0,1, \ldots$

- Draw a sample trajectory $(x_t, a_t, r_t)_{t=1}^{T}$
- Estimate $Q^{\pi_{\theta_k}} \approx \hat{Q}_k$ by Monte Carlo
- Estimate $\nabla_\theta \rho(\theta_k) \approx g_k$ by the average of
  $$g_{k,t} = \nabla_\theta \log \pi_{\theta_k}(a_t|x_t)\hat{Q}_k(x_t, a_t)$$
- Update $\theta_{k+1} = \theta_k + \alpha_k g_k$

REINFORCE WITH THE PG THEOREM

REINFORCE v2

Input: arbitrary init

For $k = 0,1,\ldots$

- Draw a sample trajectory $(x_t, a_t, r_t)_{t=1}^{T}$
- Estimate $Q^{\pi_{\theta_k}} \approx \hat{Q}_k$ by Monte Carlo
- Estimate $\nabla_\theta \rho(\theta_k) \approx g_k$ by the average of

$$g_{k,t} = \nabla_\theta \log \pi_{\theta_k}(a_t|x_t)\hat{Q}_k(x_t, a_t)$$

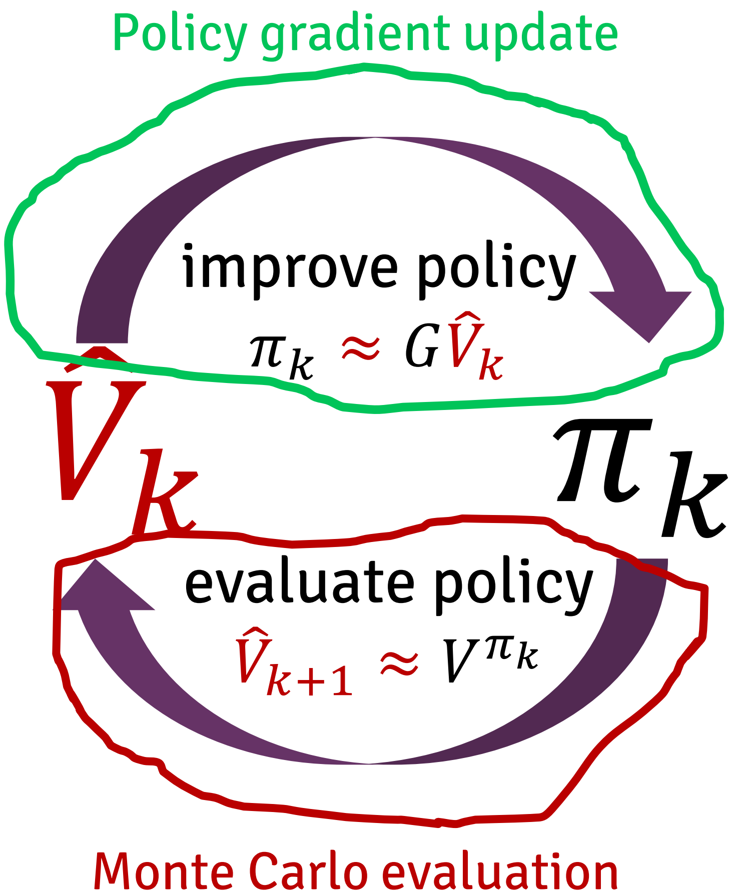
- Update $\theta_{k+1} = \theta_k + \alpha_k g_k$

$$\mathbf{E}[g_k] = \nabla_\theta \rho(\theta_k)$$

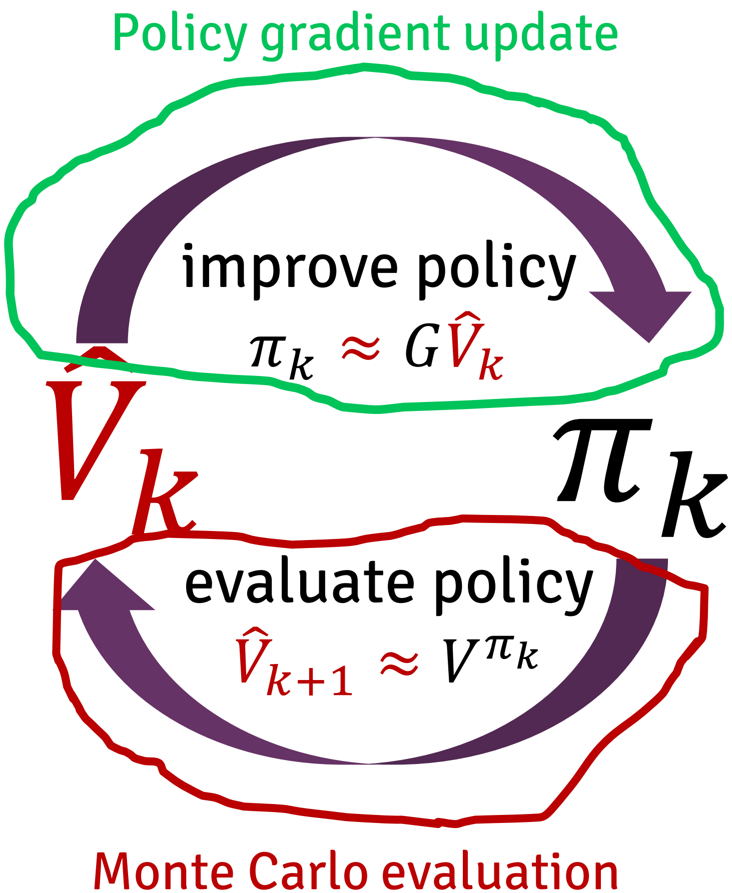
$\Rightarrow$ this is a stochastic gradient ascent algorithm

$\Rightarrow$ converges to local optimum

REINFORCE AS DIRECT POLICY OPTIMIZATION



REINFORCE AS DIRECT POLICY OPTIMIZATION



☺ direct method: no explicit approximation of V^{π} ☺

☺ converges to local optimum ☺

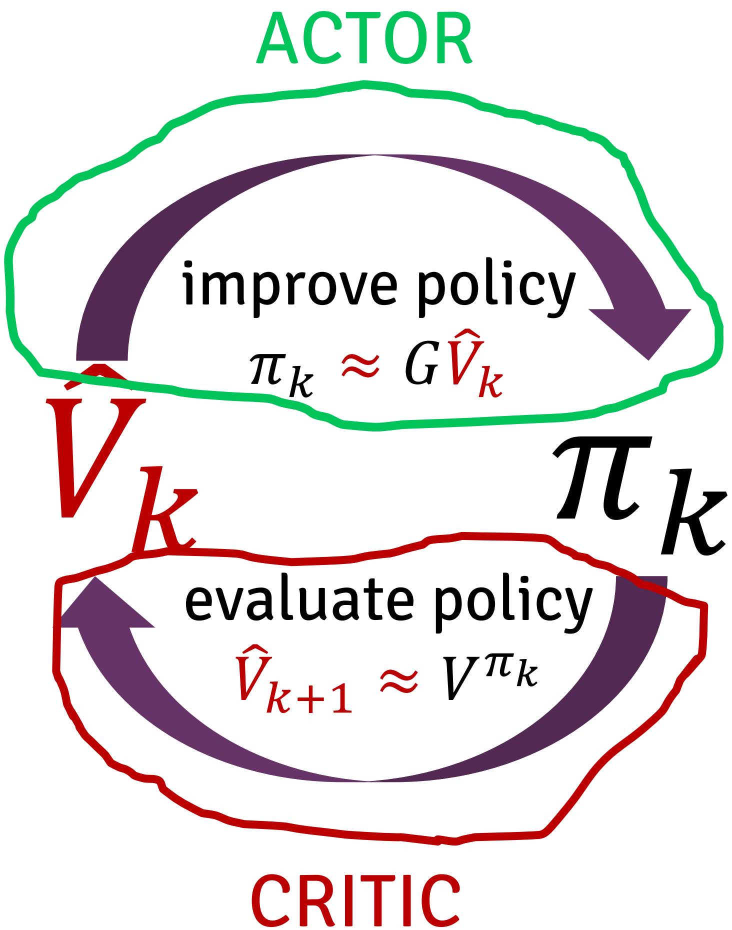
☺ less aggressive updates ☺

☹ large variance of g_k ☹

The path towards deep RL

1. Least-squares TD
   - Linear function approx.
   - General function approx.
   - Deep Q networks
2. Policy optimization
   - The likelihood-ratio trick
   - The policy gradient theorem
   - Actor-critic algorithms

ACTOR-CRITIC METHODS



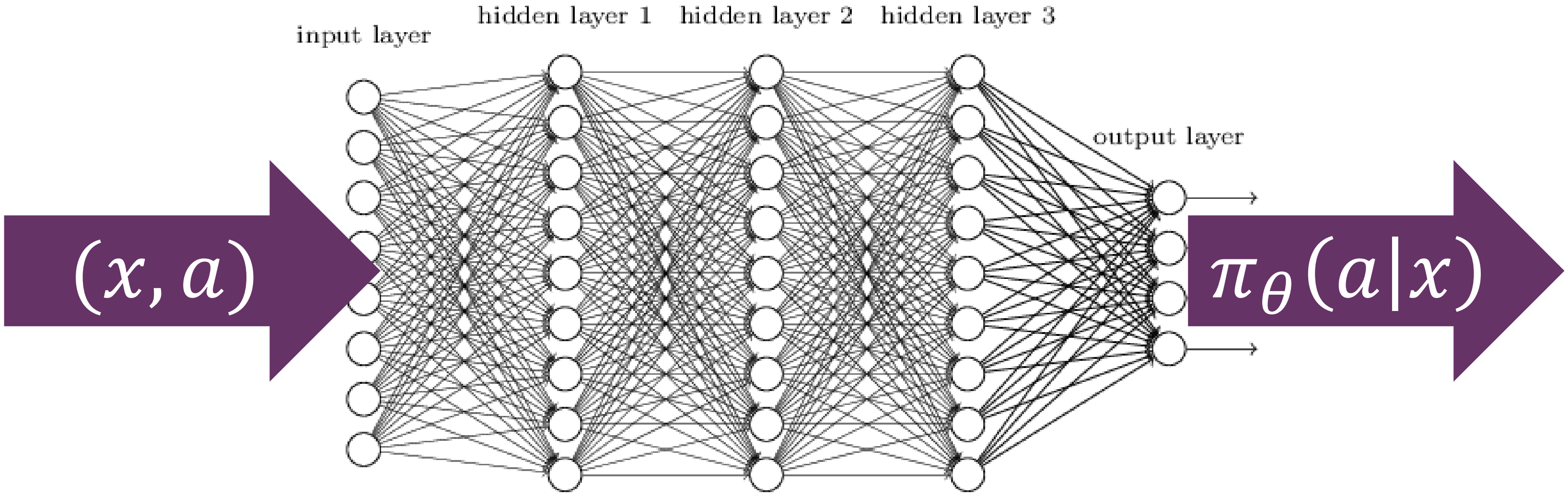
Typical actor:
policy gradient updates

Critic:

- Monte Carlo ⇒ REINFORCE
- TD(λ)
- LSTD(λ)
- DQN, …

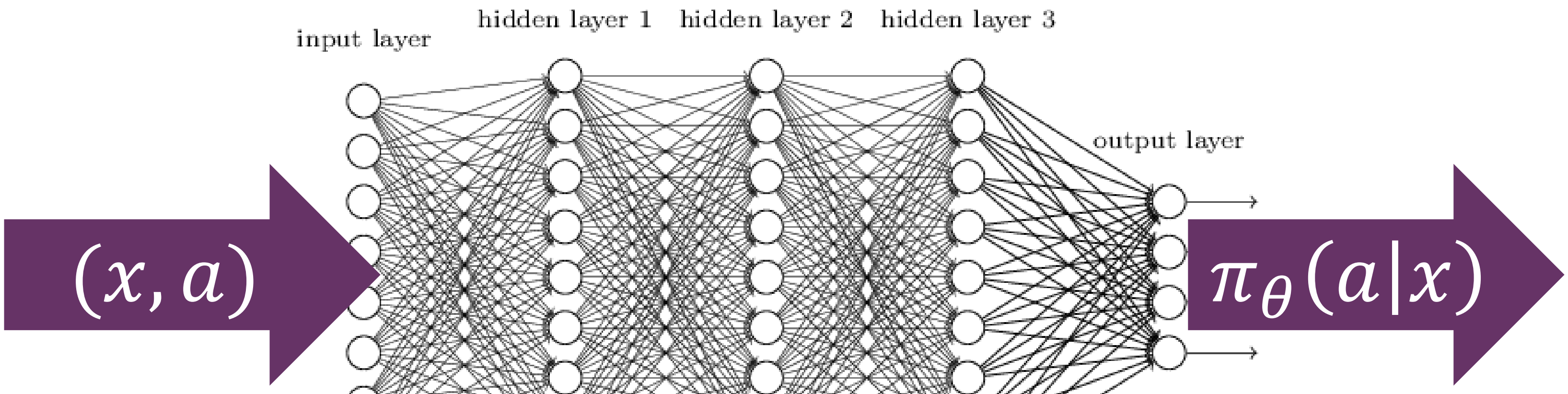
A TYPICAL DEEP RL ARCHITECTURE: A3C

Parametrize policy by a deep neural net



A TYPICAL DEEP RL ARCHITECTURE: A3C

Parametrize policy by a deep neural net



+ another neural net to estimate V^{π_θ} and to estimate Q^{π_θ} by bootstrapped Monte Carlo
+ asynchronous updates
+ entropy regularization of the gradients
+…

Next week:
Robustness and exploration in RL