Optimal and Learning Control

for

Autonomous Robots

Lecture 8

A D R L

Jonas Buchli
Agile & Dexterous Robotics Lab

FNSNF

ETH Zürich

Class logistics

Interviews on Friday!

Exercise 2 hand-out next week!

Chapter 2 of script today

Lecture 8 Goals

★Policy improvement theorem
★Policy iteration, evaluation, value iteration
★Model free RL, Monte Carlo

Policy improvement

A policy π' is better than policy π if

$$\forall\, x \in \mathbf{X} : \qquad V^{\pi'}(x) \geq V^{\pi}(x)$$

$$\exists\, x \in \mathbf{X} : \qquad V^{\pi'}(x) > V^{\pi}(x)$$

Policy improvement theorem

Assume two policies $\pi \quad \pi'$

Identical for all states, but x $\pi'(x) \neq \pi(x)$

If in 1(!) chosen state x $Q^\pi(x, \pi'(x)) \geq V^\pi(x)$

then π' is a better policy than π

Policy improvement theorem:

what if improvement in all states $\overset{?}{\Longrightarrow}$ $V^{\pi'} \geq V^\pi$

A D R L

Policy improvement theorem - proof

Preliminaries:

consider deterministic policy

$$\mu(x) = a_s, \qquad \text{where: } \pi(a_s \mid x) = 1.$$

To show:

$$Q^\pi(x, \mu'(x)) \geq V^\pi(x) \quad \Rightarrow \quad V^{\pi'} \geq V^\pi$$

Policy improvement theorem - proof

$$V^\pi(x) \leq Q^\pi(x, \mu'(x)) = E_\pi\left\{r_n + \alpha V^\pi(x_{n+1}) | u_n = \mu'(x), x_n = x\right\}$$

substitute $V^\pi(x_{n+1})$ by $Q^\pi(x_{n+1}, \mu'(x_{n+1}))$

greedy

$$V^\pi(x) \leq E_\pi\left\{r_n + \alpha\, Q^\pi(x_{n+1}, \mu'(x_{n+1}))\ |\ u_n = \mu'(x), x_n = x\right\}$$

$$V^\pi(x) \leq E_\pi\left\{r_n + \alpha\, E_\pi\left\{r_{n+1} + \alpha V^\pi(x_{n+2})\ |\ u_{n+1} = \mu'(x'), x_{n+1} = x'\right\} |u_n = \mu'(x), x_n = x\right\}$$

$$V^\pi(x) \leq E_\pi\left\{r_n + \alpha r_{n+1} + \alpha^2 V^\pi(x_{n+2}) | u_{n+1} = \mu'(x'), x_{n+1} = x', u_n = \mu'(x), x_n = x\right\}$$

simplify notation

$$V^\pi(x) \leq E_{\substack{u_{[n,n+1]}\sim\pi' \\ u_{[n+2,\dots]}\sim\pi}}\left\{r_n + \alpha r_{n+1} + \alpha^2 V^\pi(x_{n+2}) | x_n = x\right\}$$

repeat argument to infinity

$$V^\pi(x) \leq E_\pi\left\{r_n + \alpha r_{n+1} + \alpha^2 r_{n+2} + \alpha^3 r_{n+3} + \dots\ |\ u_{[n,n+1,n+2,\dots]} \sim \pi', x_n = x\right\}$$

Policy improvement theorem - proof - conclusion

Policy π' is followed for all time steps

thus omit condition on $u_{[n,n+1,n+2,...]} \sim \pi'$ yields $V^{\pi'}$

$$V^{\pi}(x) \leq E_{\pi'}\left\{r_n + \alpha r_{n+1} + \alpha^2 r_{n+2} + \alpha^3 r_{n+3} + ... \mid x_n = x\right\} = V^{\pi'}(x)$$

$$V^{\pi}(x) \leq V^{\pi'}(x)$$

QED

Greedy policy update

Consider the following greedy policy update rule

$$\begin{aligned}\pi'(x) &= \operatorname*{argmax}_u Q^\pi(x,u) \\ &= \operatorname*{argmax}_u E\left\{r_n + \alpha V^\pi(x_{n+1}) | x_n = x, u_n = u\right\}\end{aligned}$$

PIT: greedy policy update will always yield better policy

Convergence of update

What if new policy is strictly equal to the old one:

$$V^{\pi'} = V^{\pi}$$

using policy update rule

$$V^{\pi'}(x) = \max_u E\left\{r_n + \alpha V^{\pi'}(x_{n+1}) | x_n = x, u_n = u\right\}$$

$$= \max_u \sum_{x'} \mathcal{P}^u_{xx'}[\mathcal{R}^u_{xx'} + \alpha V^{\pi'}(x')]$$

→ yields optimal Bellman Equation

⇒ PI will give better policy, unless Policy is already optimal

A D R L

ETH Zürich

Policy iteration

Using Policy Evaluation and Policy improvement, find optimal policy iteratively

Policy iteration

1) Policy evaluation (complete!)
2) Policy improvement
3) repeat

need to visit all s in an iteration ('full backup')

$$\pi_0 \xrightarrow{\mathrm{E}} V^{\pi_0} \xrightarrow{\mathrm{I}} \pi_1 \xrightarrow{\mathrm{E}} V^{\pi_1} \xrightarrow{\mathrm{I}} \pi_2 \xrightarrow{\mathrm{E}} \cdots \xrightarrow{\mathrm{I}} \pi^* \xrightarrow{\mathrm{E}} V^*$$

discount factor < 1 or
termination under policy
(convergence of cost-to-go!)

[SB] Ch 4.1

Algorithm 2 Policy Iteration

1. Initialization

select $V(x) \in \Re$ and $\pi(x) \in \mathbf{U}$ arbitrarily for all $x \in \mathbf{X}$

2. Policy evaluation

repeat

$\Delta \leftarrow 0$

for each: $x \in \mathcal{X}$

$v \leftarrow V(x)$

$V(x) \leftarrow \sum_u \pi(x,u) \sum_{x'} \mathcal{P}_{xx'}^{\pi(x)} [\mathcal{R}_{xx'}^u + \alpha V(x')]$

$\Delta \leftarrow \max(\Delta, |v - V(x)|)$

until $\Delta < \theta$ (a small positive number)

3. Policy Improvement

$policyIsStable \leftarrow true$

for $x \in \mathcal{X}$ **do**

$b \leftarrow \pi(x)$

$\pi(x) \leftarrow \text{argmax}_u \sum_{x'} \mathcal{P}_{xx'}^u [\mathcal{R}_{xx'}^u + \alpha V(x')]$

if $b \neq \pi(x)$ **then**

$policyIsStable \leftarrow false$

end if

end for

if $policyIsStable$ **then**

stop

else

go to 2

end if

Return: a policy, π, such that: $\pi(x) = \arg\max_u \sum_{x'} \mathcal{P}_{xx'}^u [\mathcal{R}_{xx'}^u + \alpha V(x')]$

Visits all states! repeatedly

Policy evaluation

Policy improvement

Visits all states!

ADRL

ETH Zürich

Value iteration

Can we get away without several ('infinite') # of sweeps in the Policy Evaluation step?

In reality: truncate PE after a finite number of steps

Idea: truncate after ONE iteration

→ PE and PI can be merged into a single update rule:

$$V_{k+1}(x) = \boxed{\max_u} \sum_{x'} \mathcal{P}^u_{xx'} [\mathcal{R}^u_{xx'} + \alpha V_k(x')]$$

Value iteration

Algorithm 3 Value Iteration

Initialization: $V(x) \in \Re$ and $\pi(x) \in \mathbf{U}$ arbitrarily for all $x \in \mathbf{X}$

repeat

  $\Delta \leftarrow 0$

  for $x \in \mathbf{X}$ **do**

    $v \leftarrow V(x)$

    $V(x) \leftarrow \max_u \sum_{x'} \mathcal{P}^u_{xx'}[\mathcal{R}^u_{xx'} + \alpha V(x')]$

    $\Delta \leftarrow \max(\Delta, |v - V(x)|)$

  end for

until $\Delta < \theta$ (a small positive number)

Return: a policy, π, such that: $\pi(x) = \arg\max_u \sum_{x'} \mathcal{P}^u_{xx'}[\mathcal{R}^u_{xx'} + \alpha V(x')]$

Value iteration vs. Policy evaluation

VI

$$V_{k+1}(x) = \max_u E\left\{r_n + \alpha V_k(x') \mid x_n = x, u_n = u\right\}$$

$$= \max_u \sum_{x'} \mathcal{P}^u_{xx'}[\mathcal{R}^u_{xx'} + \alpha V_k(x')]$$

evaluate most rewarding successor state

PE

$$V_{k+1}(x) = E_\pi\left\{r_n + \alpha V_k(x') \mid x_n = x\right\}$$

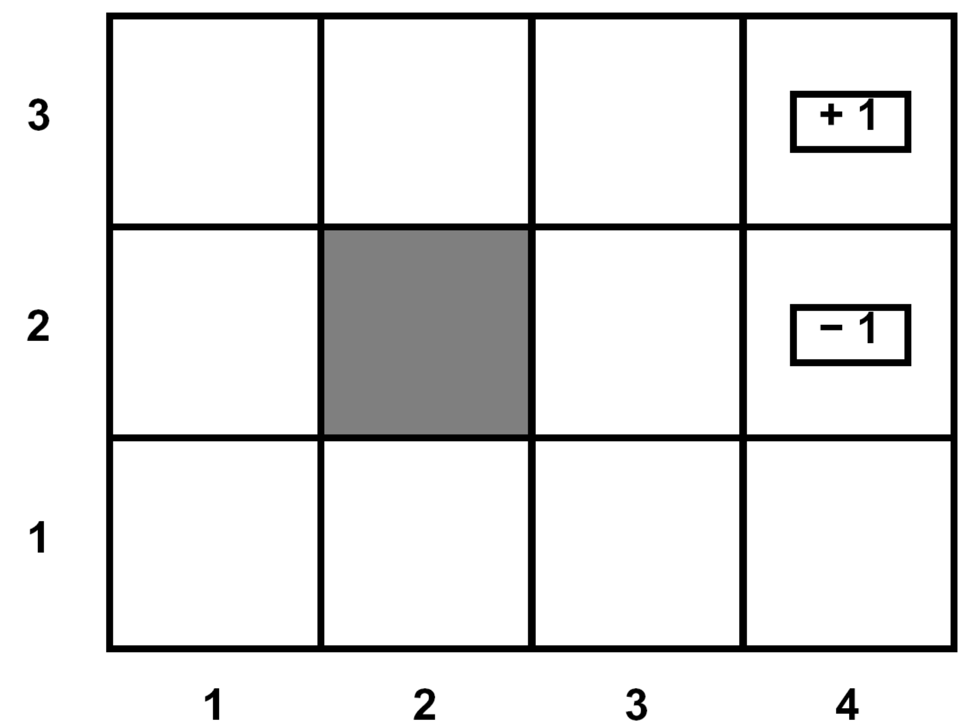
$$= \sum_u \pi(x,u) \sum_{x'} \mathcal{P}^u_{xx'}[\mathcal{R}^u_{xx'} + \alpha V_k(x')]$$

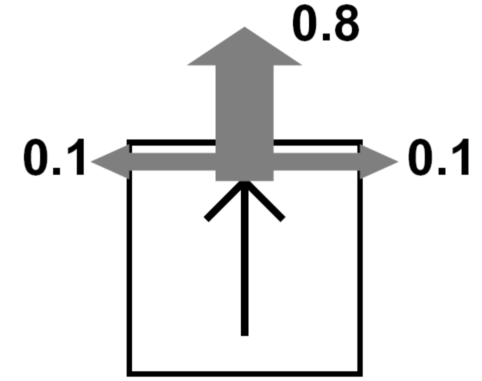
evaluate all successor states

(PI $\quad \pi(x) \leftarrow \operatorname*{argmax}_u \sum_{x'} \mathcal{P}^u_{xx'}[\mathcal{R}^u_{xx'} + \alpha V(x')]$)

Canonical Example: Grid World

- The agent lives in a grid
- Walls block the agent's path
- The agent's actions do not always go as planned:
  - 80% of the time, the action North takes the agent North (if there is no wall there)
  - 10% of the time, North takes the agent West; 10% East
  - If there is a wall in the direction the agent would have been taken, the agent stays put
- Big rewards come at the end





Value Iteration in Gridworld

noise = 0.2, γ =0.9, two terminal states with R = +1 and -1

0.8*0.9*1
+0.1*0.9*0.0
+0.1*0.9*0

= 0.72

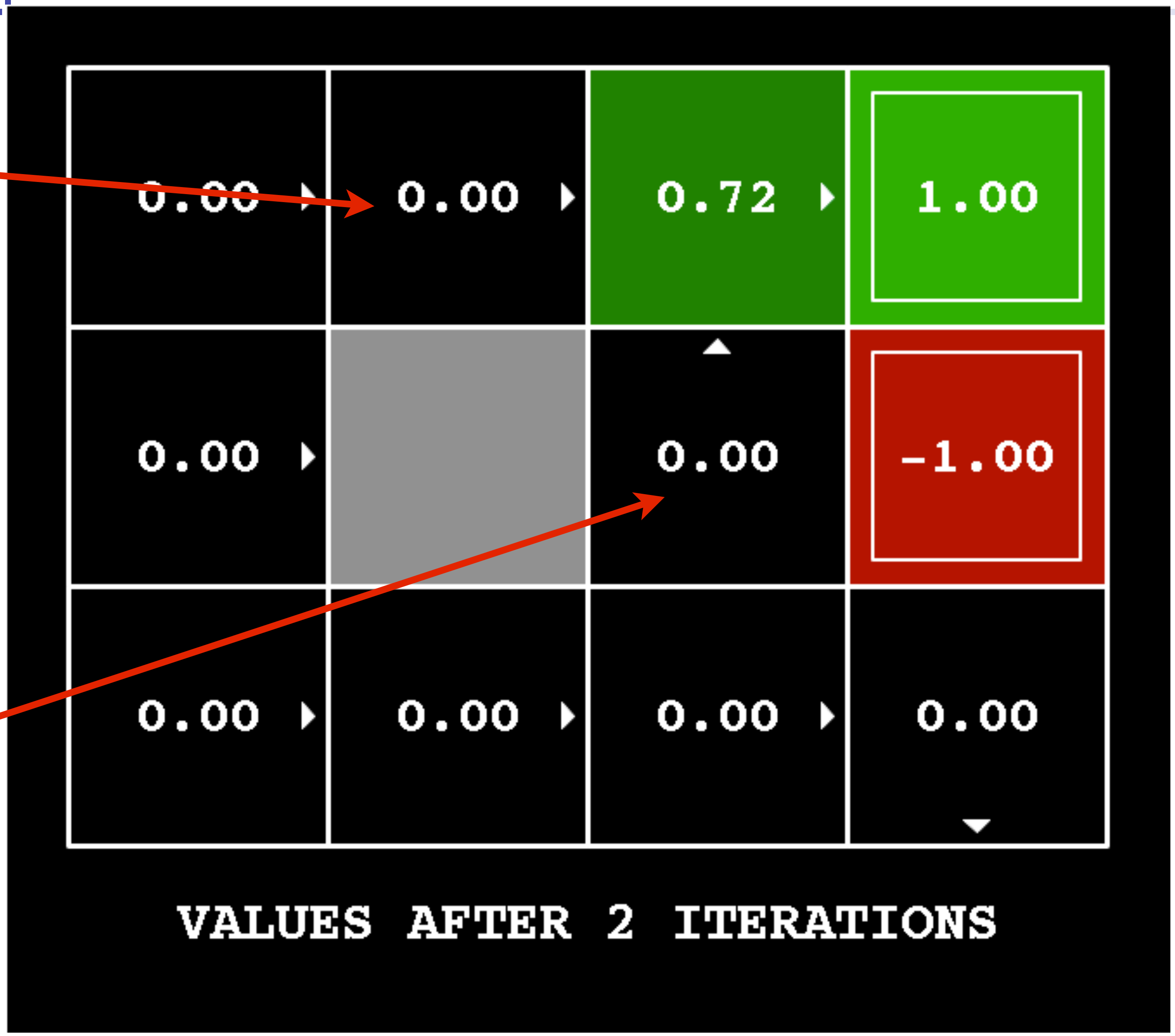
$$V_{k+1}(s) = \max_a \sum_{s'} \mathcal{P}^a_{ss'} [\mathcal{R}^a_{ss'} + \gamma \; V_k(s')]$$

Value Iteration in Gridworld

noise = 0.2, γ =0.9, two terminal states with R = +1 and -1

0.8*0.9*0.72

= 0.52



0.8*0.9*0.72
+ 0.1*0.1*-1

= 0.43

0.8*0.9*1
+0.1*0.9*0.72
+0.1*0.9*0

= 0.78

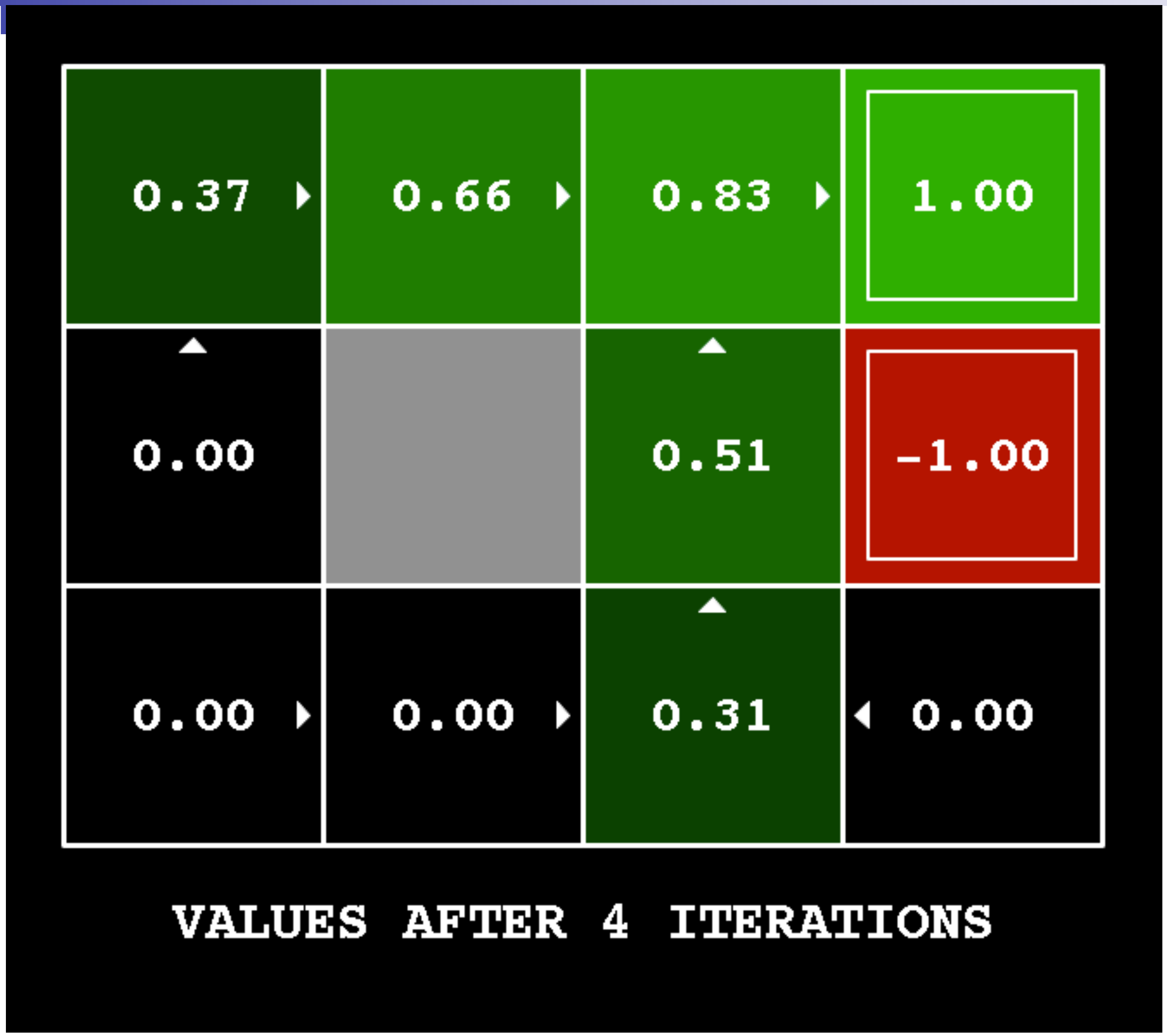
$$V_{k+1}(s) = \max_a \sum_{s'} \mathcal{P}^a_{ss'} [\mathcal{R}^a_{ss'} + \gamma \ V_k(s')]$$

Value Iteration in Gridworld

noise = 0.2, γ =0.9, two terminal states with R = +1 and -1

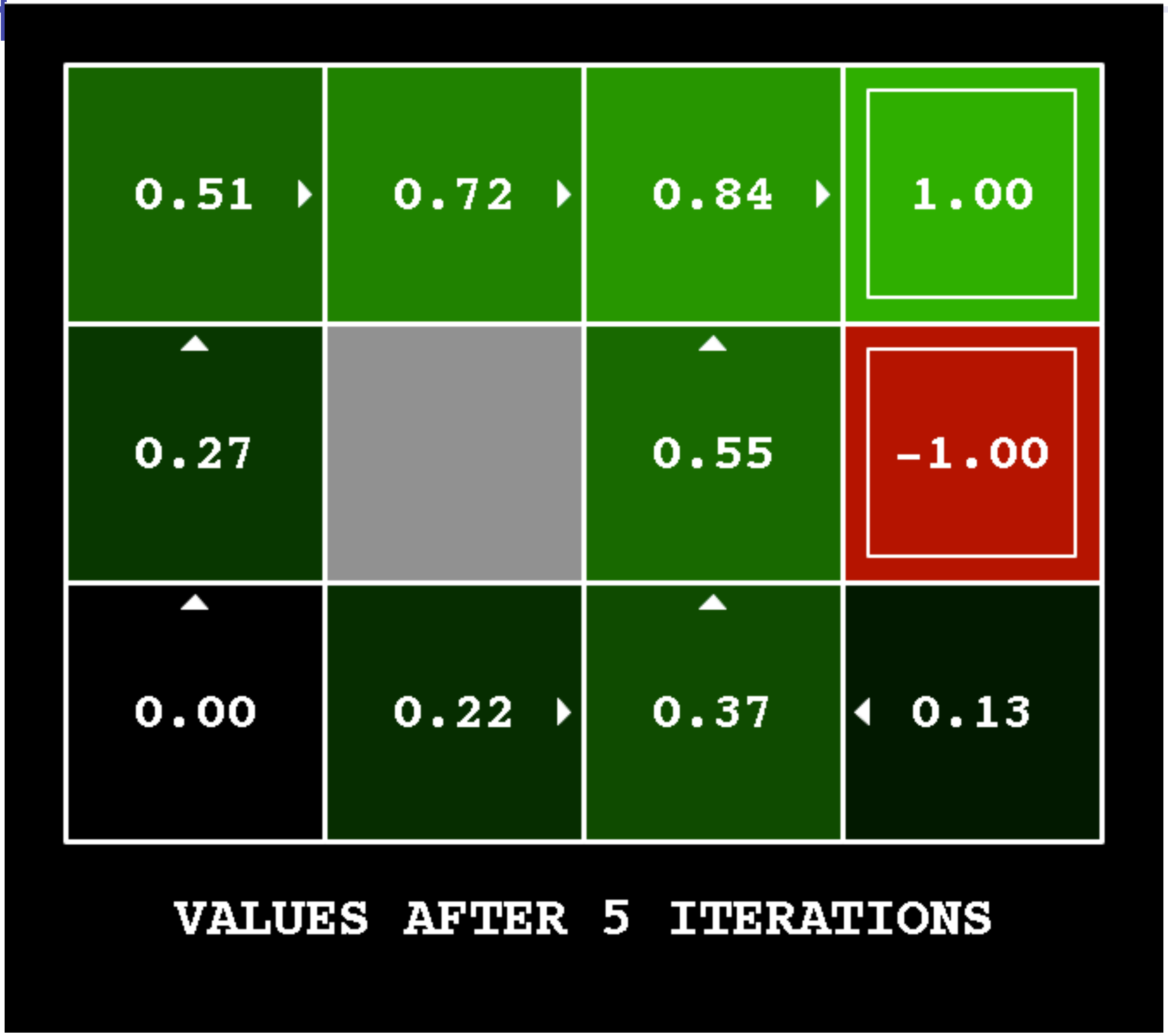
Value Iteration in Gridworld

noise = 0.2, γ =0.9, two terminal states with R = +1 and -1



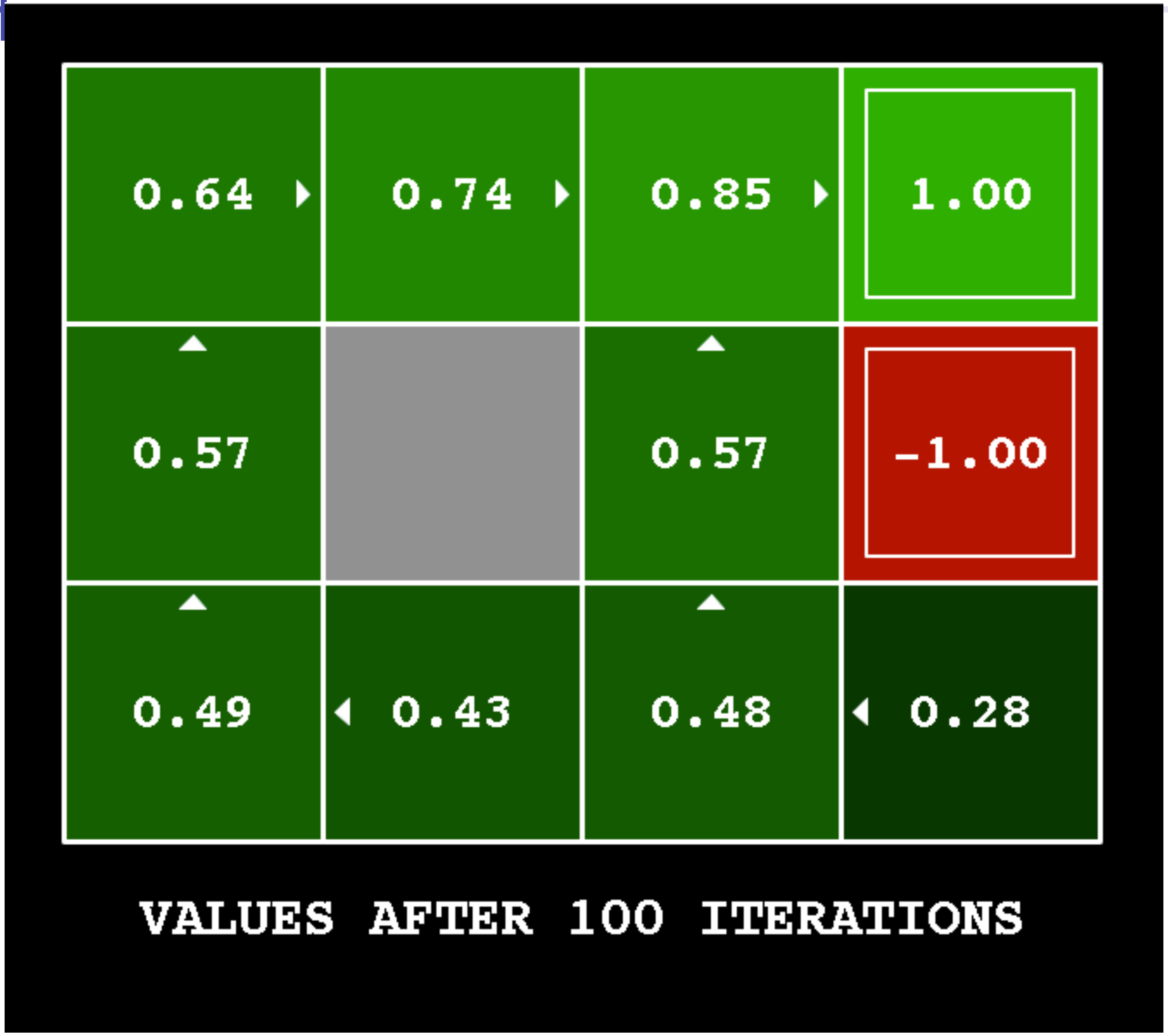
Value Iteration in Gridworld

noise = 0.2, γ =0.9, two terminal states with R = +1 and -1



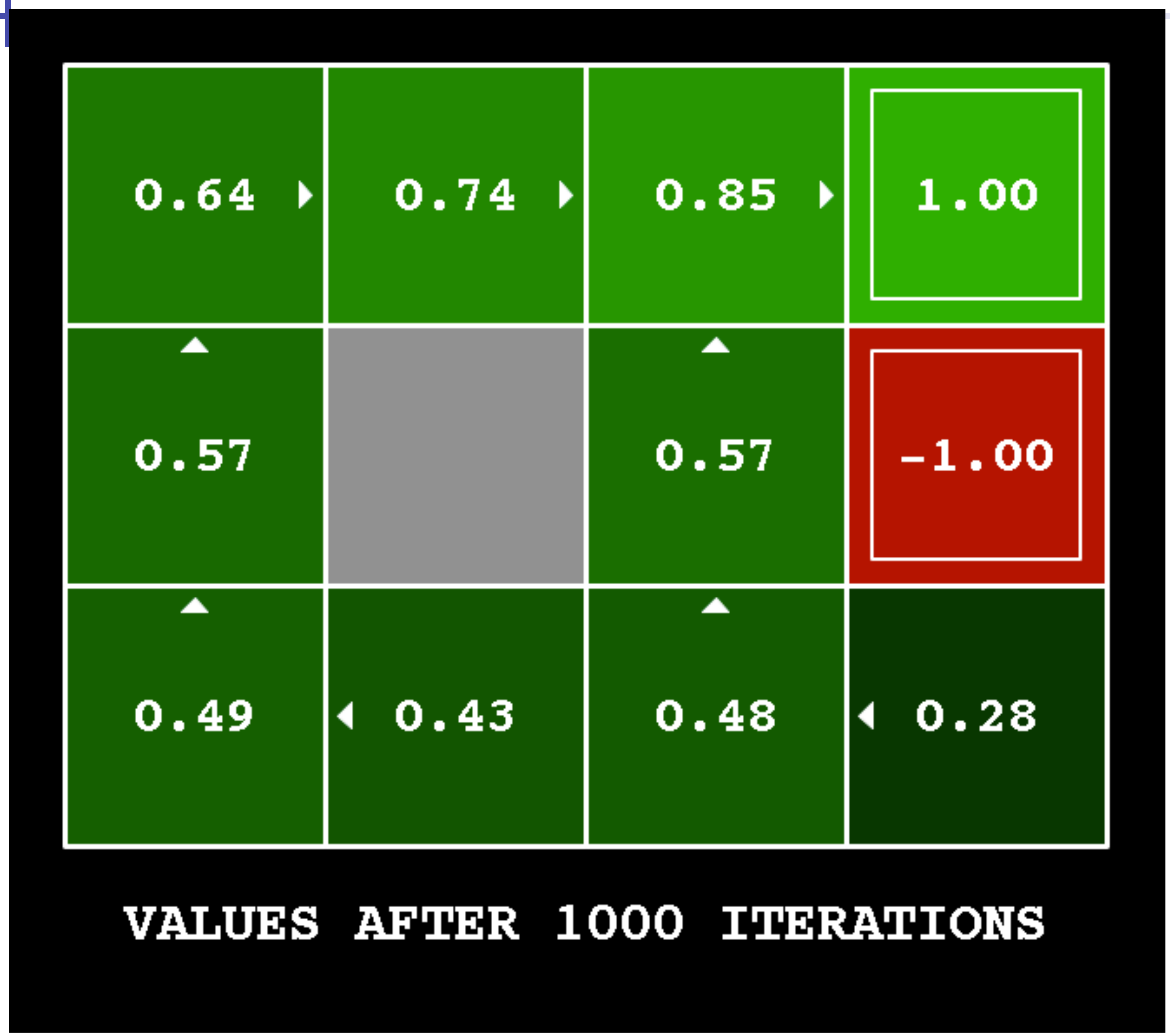
Value Iteration in Gridworld

noise = 0.2, γ =0.9, two terminal states with R = +1 and -1

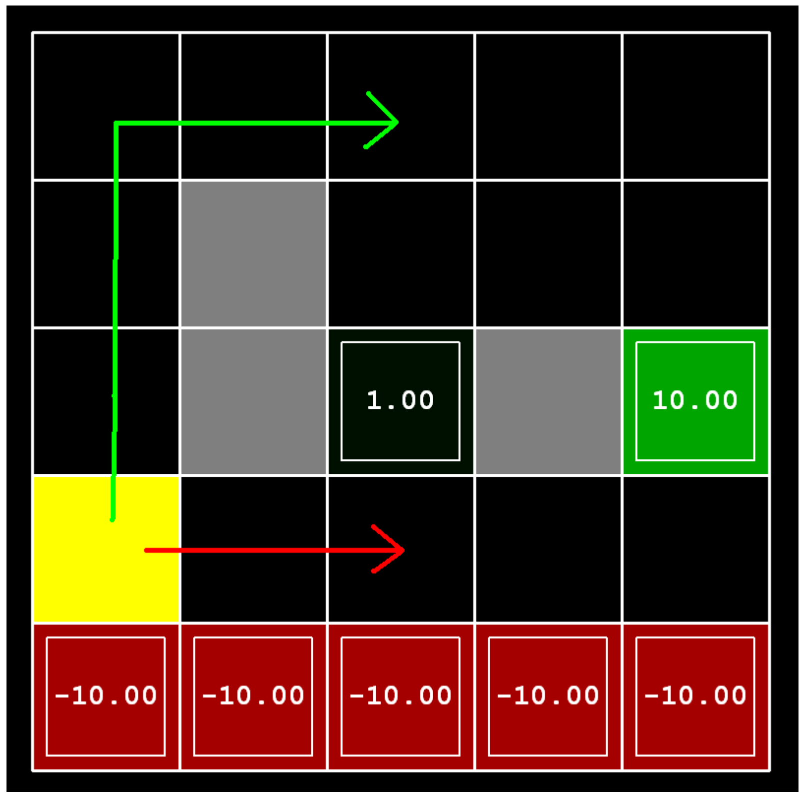


Value Iteration in Gridworld

noise = 0.2, γ =0.9, two terminal states with R = +1 and -1



Exercise 1: Effect of discount, noise



	1	2	3	4
a				
b				
c				
d				

(a) Prefer the close exit (+1), risking the cliff (-10)

(b) Prefer the close exit (+1), but avoiding the cliff (-10)

(c) Prefer the distant exit (+10), risking the cliff (-10)

(d) Prefer the distant exit (+10), avoiding the cliff (-10)

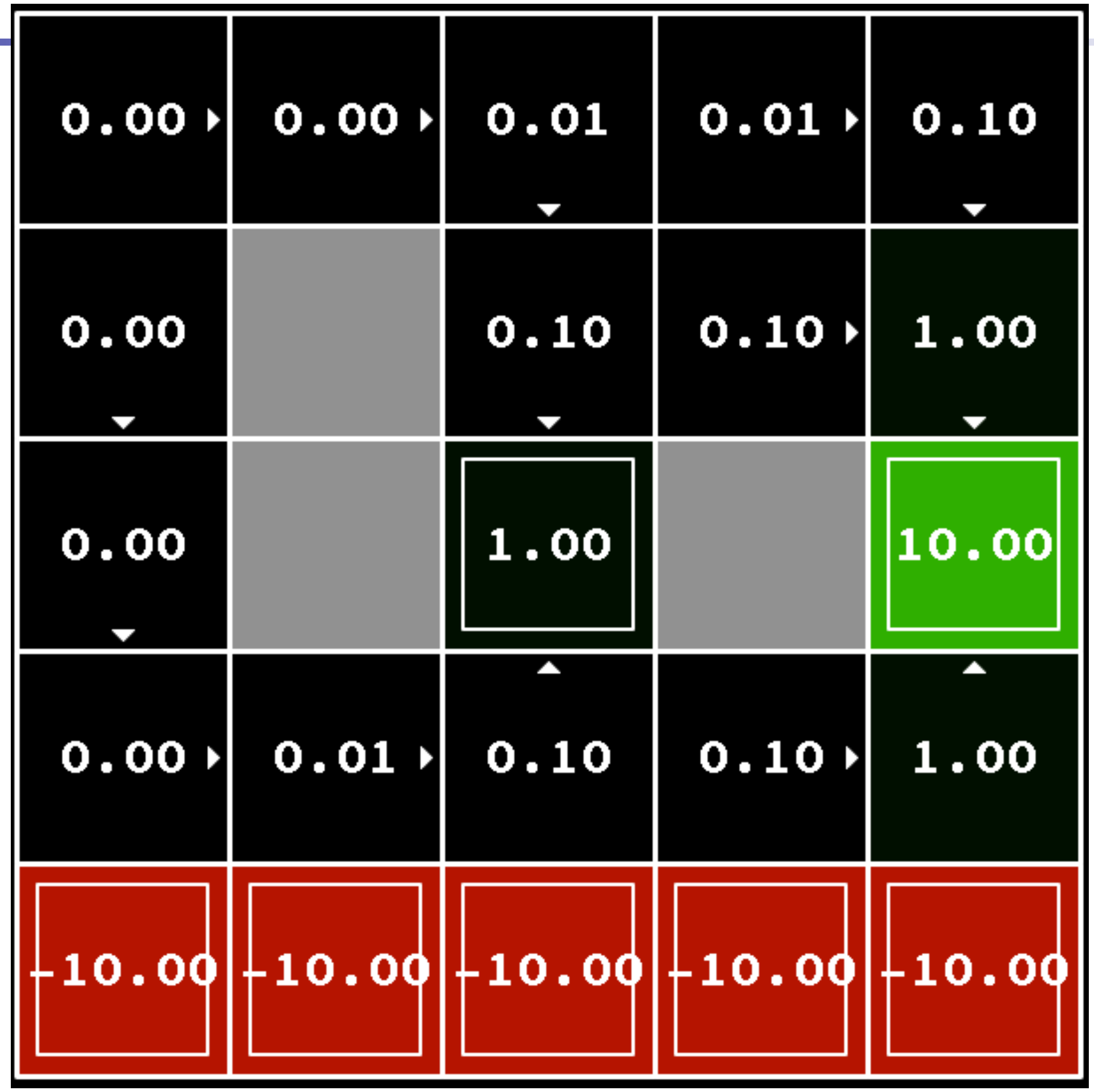
(1) γ = 0.1, noise = 0.5

(2) γ = 0.99, noise = 0

(3) γ = 0.99, noise = 0.5

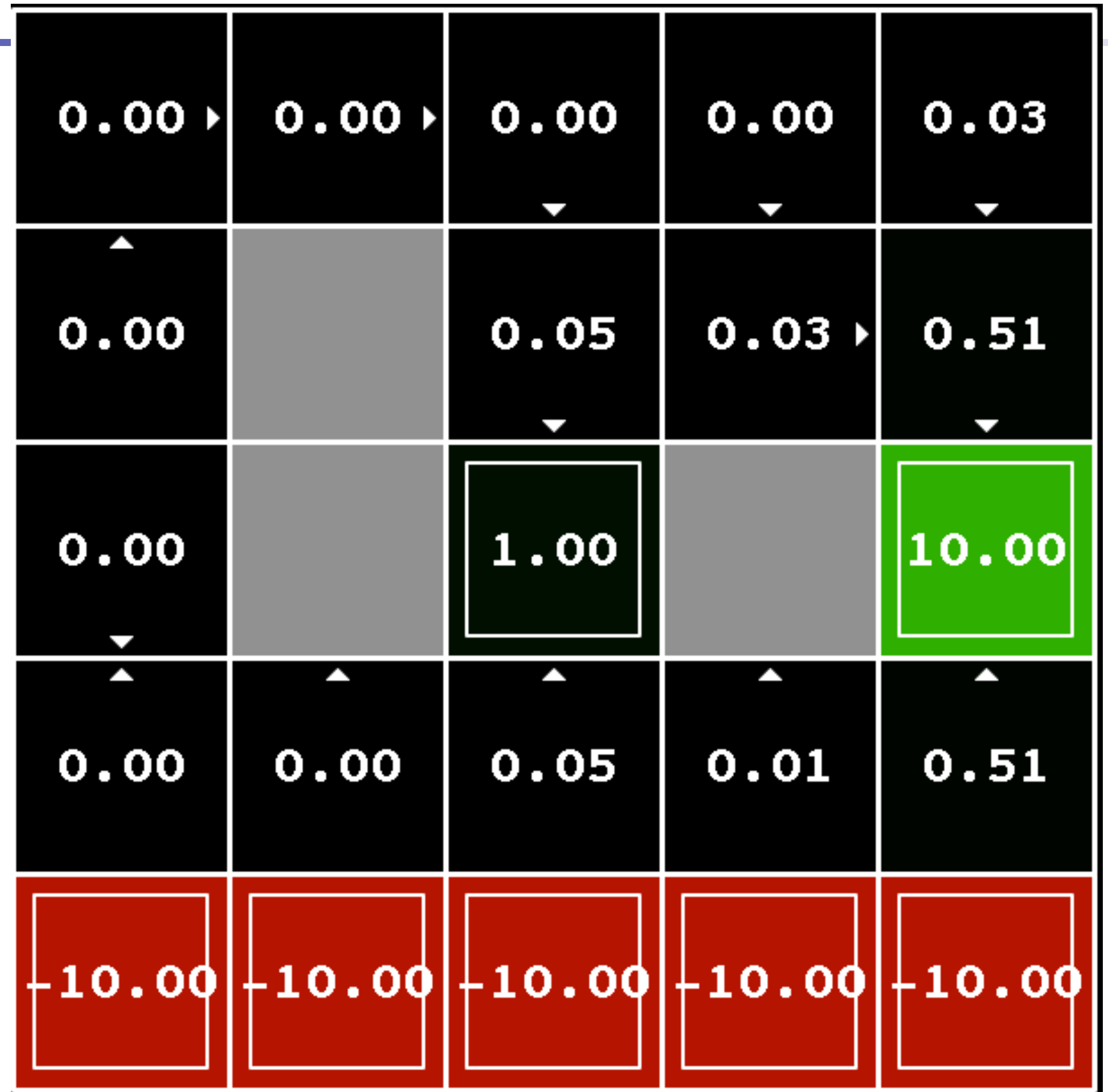
(4) γ = 0.1, noise = 0

Exercise 1 Solution



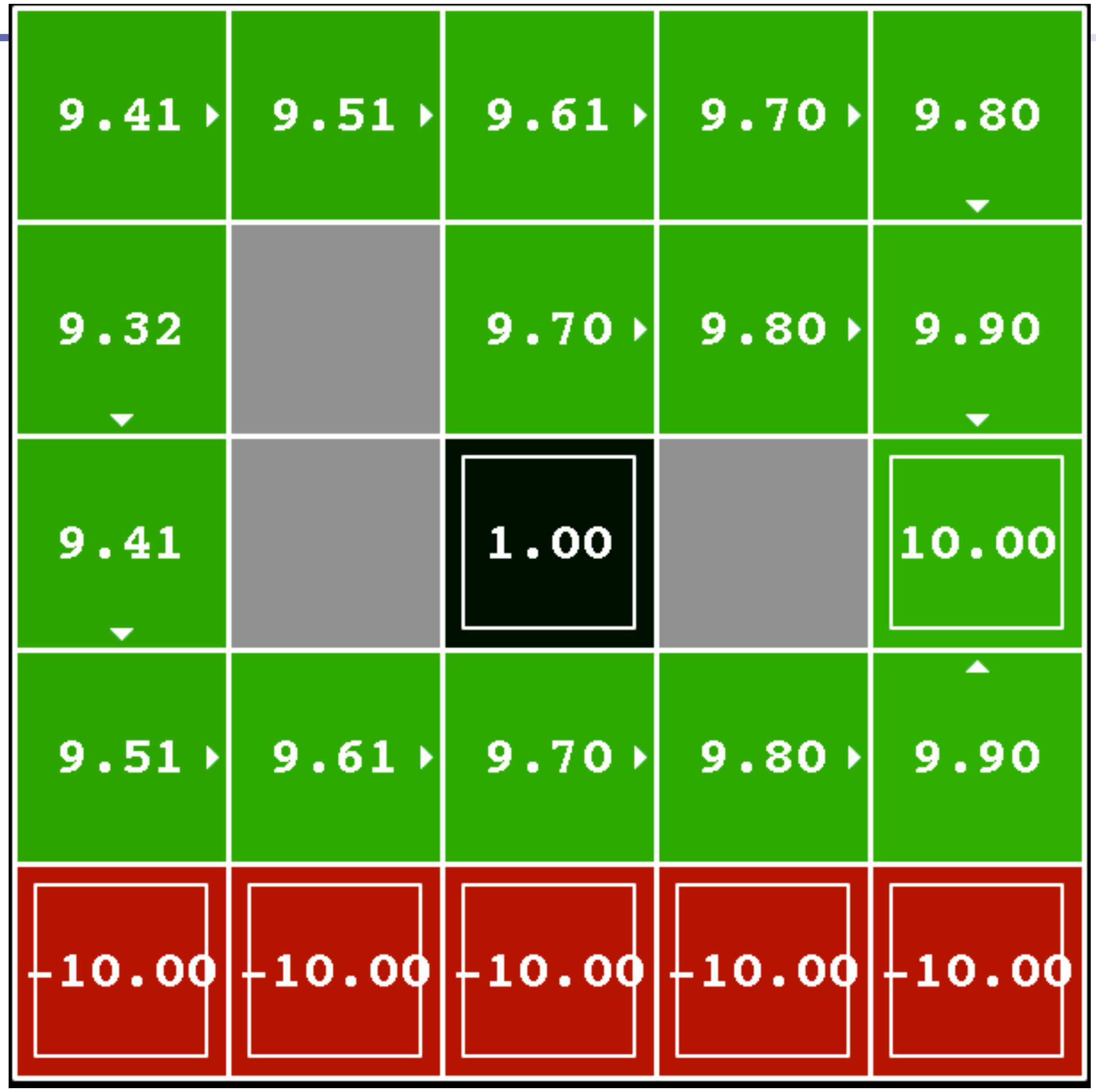
(a) Prefer close exit (+1), risking the cliff (-10) --- γ = 0.1, noise = 0

Exercise 1 Solution



(b) Prefer close exit (+1), avoiding the cliff (-10) -- γ = 0.1, noise = 0.5

Exercise 1 Solution



(c) Prefer distant exit (+10) risking the cliff (-10) -- γ = 0.99, noise = 0

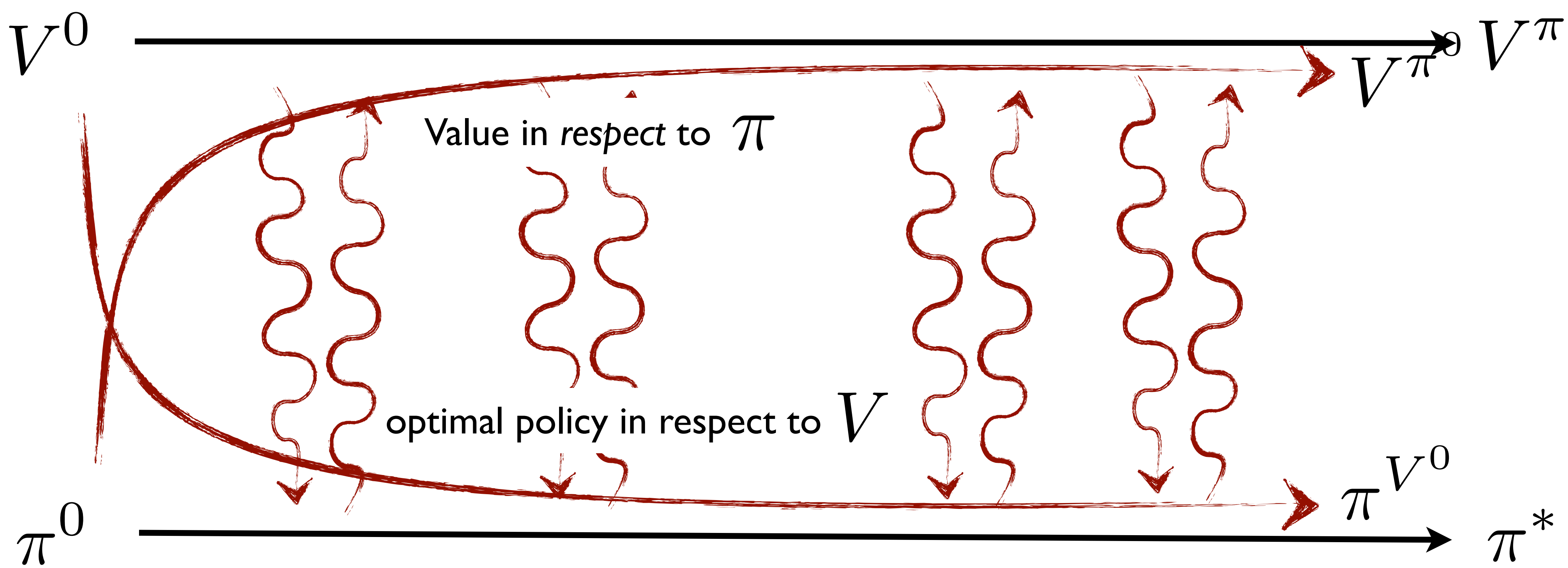
Exercise 1 Solution

(d) Prefer distant exit (+10) avoid the cliff (-10) -- γ = 0.99, noise = 0.5

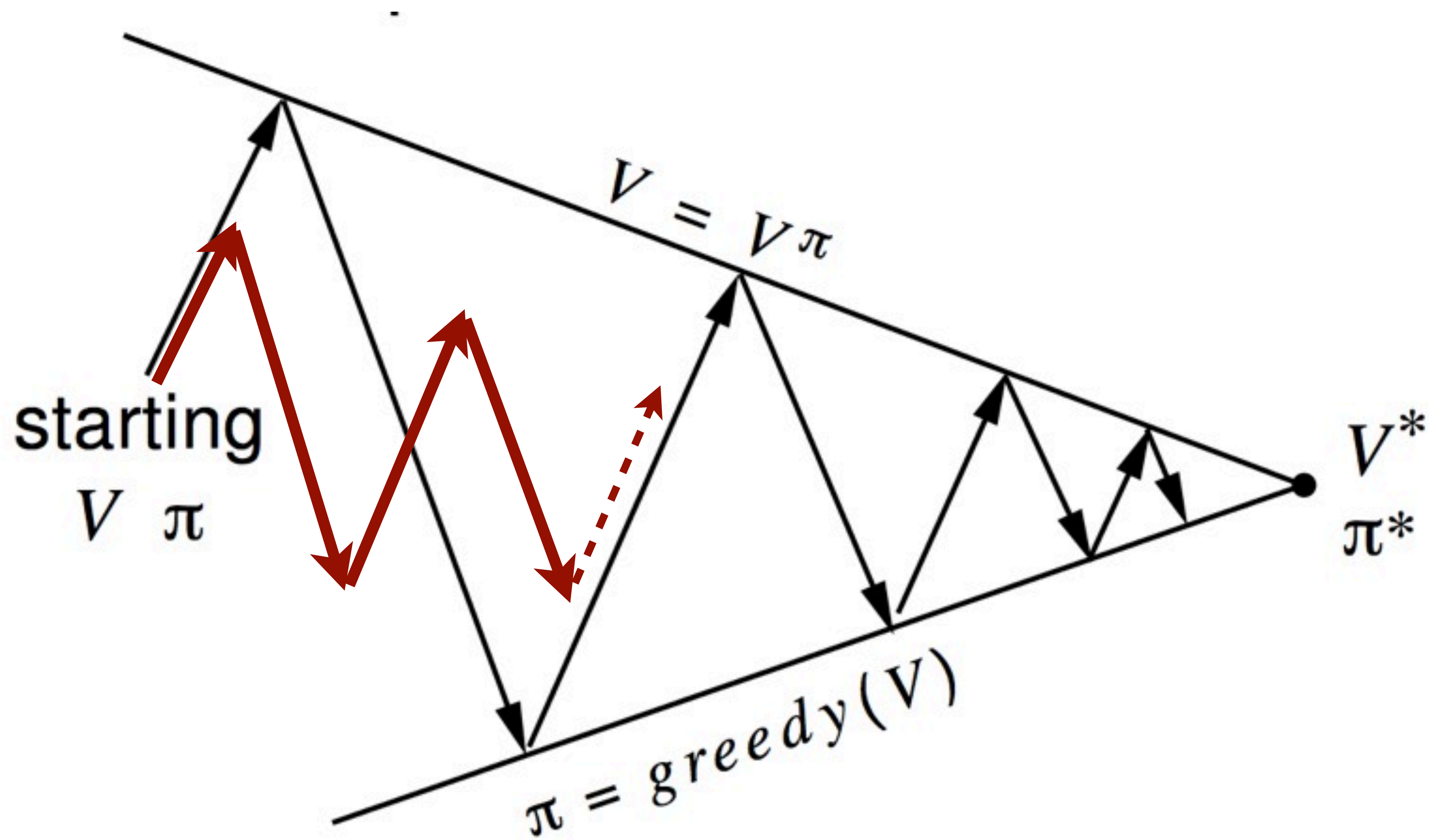
Generalized Policy Iteration

Value and policy interact

policy evaluation

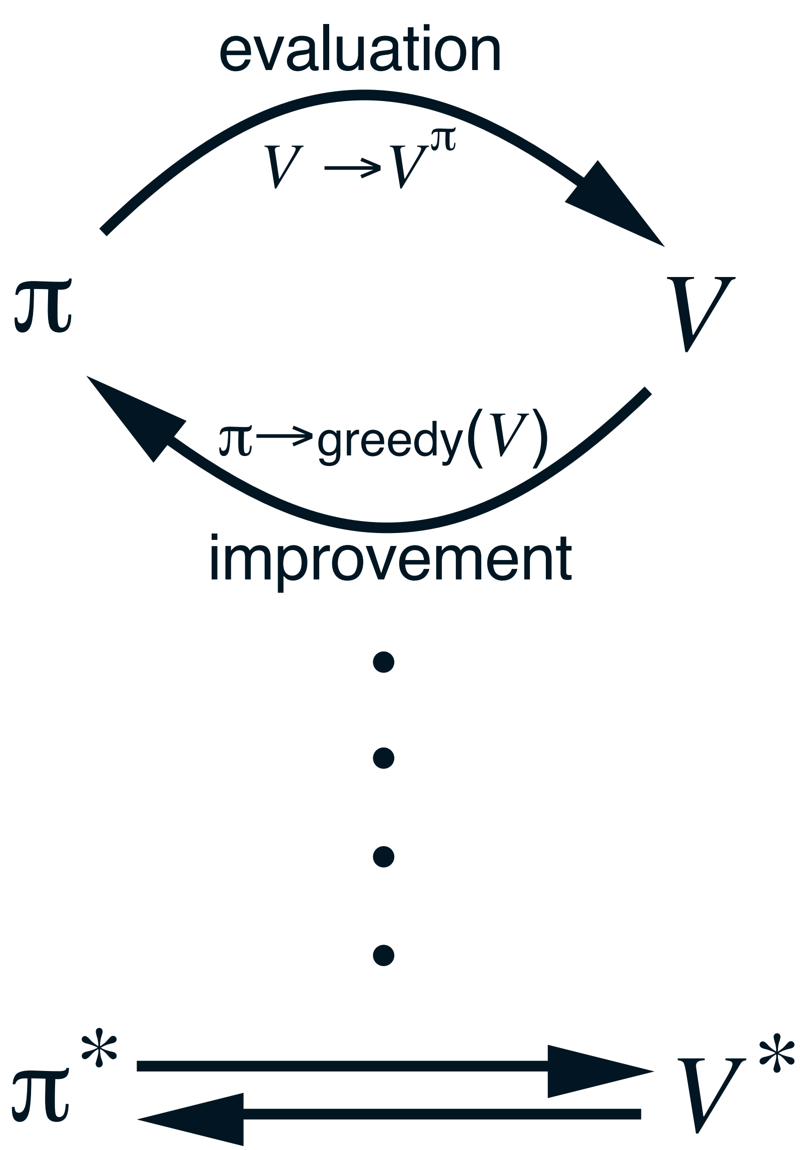


policy improvement



$V = V^{\pi}$
starting
$V \quad \pi$
V^*
π^*
$\pi = greedy(V)$

Generalized Policy Iteration



Prediction problem: Estimate V

Control problem: Find controls

'co-evolution' of control and value function

GPI ‘boundary cases’:

Policy iteration

1) Policy evaluation (complete!)
2) Policy improvement
3) repeat

need to sweep (‘visit all’) x in an iteration (‘full backup’)

$$\pi_0 \xrightarrow{\text{E}} V^{\pi_0} \xrightarrow{\text{I}} \pi_1 \xrightarrow{\text{E}} V^{\pi_1} \xrightarrow{\text{I}} \pi_2 \xrightarrow{\text{E}} \cdots \xrightarrow{\text{I}} \pi^* \xrightarrow{\text{E}} V^*$$

Value iteration

1) locally opt. Value evaluation
2) repeat

need to sweep x in an iteration

$$V_0 \xrightarrow{I} V_1 \xrightarrow{I} V_2 \xrightarrow{I} V_3 \xrightarrow{I} \ldots \xrightarrow{I} V^*$$

A D R L

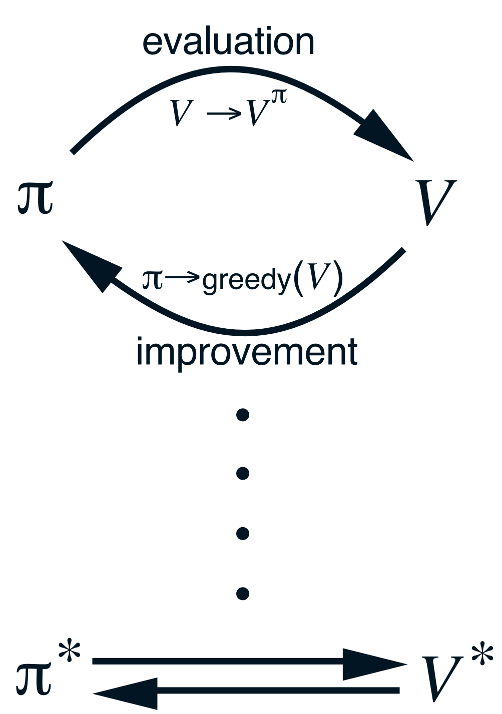
ETH Zürich

Generalized Policy Iteration

have seen two algorithms ('boundary cases'):

1) policy evaluation - improvement - iteration

2) value iteration



many variants to the previously seen 'basic' algorithms

Visit all states???

Both VI and PI have to visit all states!

```
traceroute to duerer.usc.edu (128.125.125.41), 64 hops max, 52 byte packets
 1  192.168.1.1 (192.168.1.1)  1.014 ms  0.992 ms  0.780 ms
 2  * * *
 3  217-168-57-101.static.cablecom.ch (217.168.57.101)  14.084 ms  18.090 ms  7.835 ms
 4  84.116.211.25 (84.116.211.25)  189.901 ms  190.472 ms  190.575 ms
 5  84.116.202.241 (84.116.202.241)  183.342 ms  188.042 ms  200.363 ms
 6  84.116.210.217 (84.116.210.217)  183.742 ms  182.100 ms  183.804 ms
 7  fr-par02a-rd1-gi-15-0-0.aorta.net (84.116.130.213)  177.586 ms
    at-vie15a-rd1-xe-4-1-0.aorta.net (84.116.130.193)  184.864 ms  185.053 ms
 8
 9
10  xe-0.equinix.snjsca04.us.bb.gin.ntt.net (206.223.116.12)  183.620 ms  186.715 ms  191.808 ms
11  ae-7.r20.snjsca04.us.bb.gin.ntt.net (129.250.5.52)  186.407 ms  186.333 ms  204.364 ms
12  ae-4.r21.lsanca03.us.bb.gin.ntt.net (129.250.6.10)  198.648 ms  402.438 ms  198.831 ms
13  ae-2.r05.lsanca03.us.bb.gin.ntt.net (129.250.5.86)  192.220 ms  200.324 ms  392.242 ms
14  165.254.21.242 (165.254.21.242)  208.798 ms  193.002 ms  194.576 ms
15  130.152.181.131 (130.152.181.131)  182.530 ms  188.588 ms  181.767 ms
16  rtr30-v255.usc.edu (128.125.251.148)  183.561 ms  189.412 ms  181.406 ms
17  duerer.usc.edu (128.125.125.41)  182.365 ms  183.795 ms  186.961 ms
```

Shortest(?) path: ZH-LA 17 hops

Reduce dependency on model

DP Methods require full sweeps (visit all states) and complete transition probabilities

They are iterative

We can use insight of DP to use similar ‘tricks’ to get rid of requirement for full sweeps and complete transition probabilities

‘get rid of the (full) model’

Sample based RL

Monte Carlo Method

Monte-Carlo Methods

Monte-Carlo Method (Sutton definition): average (values) over random samples of actual returns

Episodic learning

$$R_t = r_t + r_{t+1} + r_{t+2} + \ldots + r_N$$

$$V^\pi(x) = E\{R_n \mid x_n = x\} = E\left\{\sum_{k=0}^{\infty} \alpha^k r_{n+k} \mid x_n = x\right\}$$

Expectation is a weighted average!

Approximate E by 'sampling'

$$E(x) = \sum_i P(x_i)x_i \approx \sum_s \frac{1}{N} x_s \qquad x_s \sim P(x)$$

A D R L

ETH Zürich

Approximate V by sampling

- Do N rollouts
- Average return observed after first visit of each state

$$\widetilde{V}_N^\pi(x) \approx \frac{1}{N}\sum_{i=1}^{N} R_n^i(x,u) = \frac{1}{N}\sum_{i=1}^{N}\left(r_n^i + \alpha\, r_{n+1}^i + \alpha^2\, r_{n+2}^i + \dots\right)$$

$$\boxed{V^\pi(x) = E\{R_n \mid x_n = x\} = E\left\{\sum_{k=0}^{\infty}\alpha^k r_{n+k} \mid x_n = x\right\}}$$

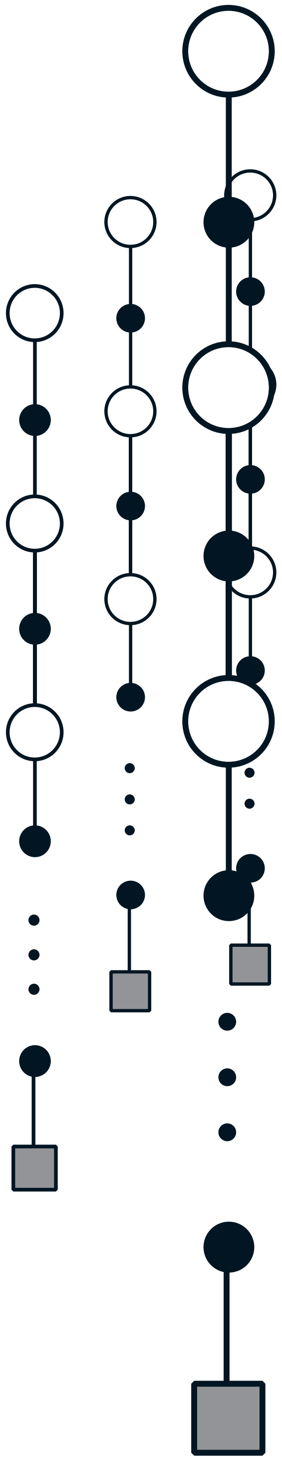
$$\widetilde{V}_N^\pi(x) \approx \frac{1}{N}\sum_{i=1}^{N_R}\sum_{k=0}^{N_T}\alpha^k\, r_{n+k}$$

'sampling approach to calculate expectation'

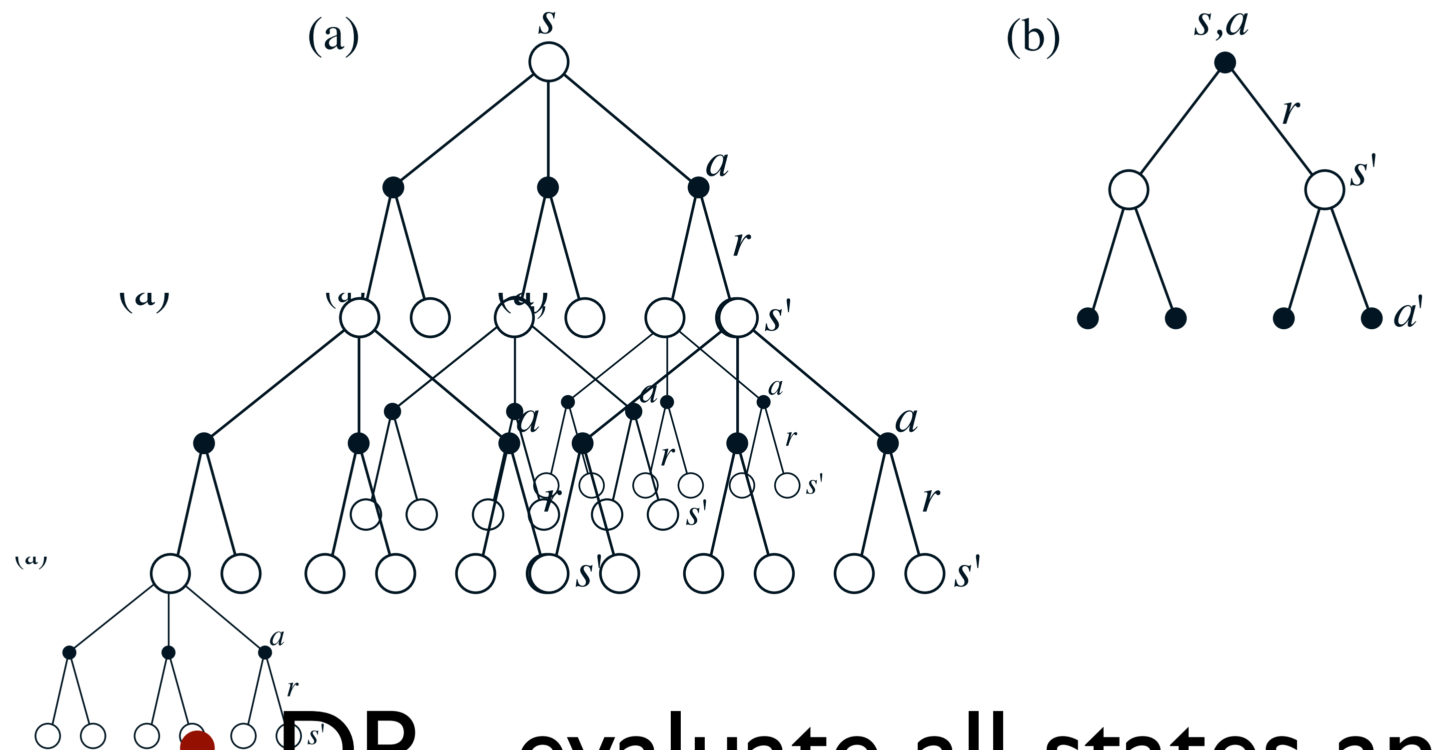
A D R L

ETH Zürich

Tree view on DP/MC



terminal state



- DP - evaluate all states and/or all choices: full backups
- MC - only evaluate states seen in an episode

opportunity and problem: can 'focus' on relevant states, might not explore...

Credits

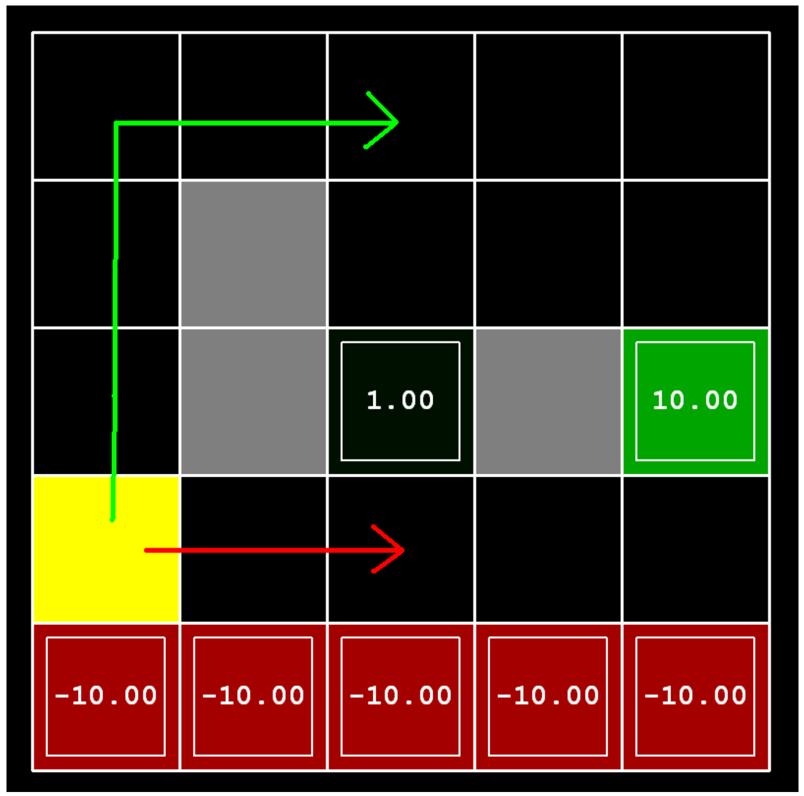
some material from:

Pieter Abbeel's Fall 2012: CS 287 Advanced Robotics @ UC Berkeley

Sutton & Barto's book: http://webdocs.cs.ualberta.ca/~sutton/book/the-book.html

SOLUTION

Exercise 1: Effect of discount, noise



	1	2	3	4
a				x
b	x			
c		x		
d			x	

(a) Prefer the close exit (+1), risking the cliff (-10)

(b) Prefer the close exit (+1), but avoiding the cliff (-10)

(c) Prefer the distant exit (+10), risking the cliff (-10)

(d) Prefer the distant exit (+10), avoiding the cliff (-10)

(1) γ = 0.1, noise = 0.5

(2) γ = 0.99, noise = 0

(3) γ = 0.99, noise = 0.5

(4) γ = 0.1, noise = 0

ADRL

ETH Zürich