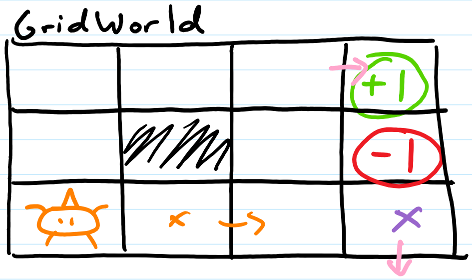
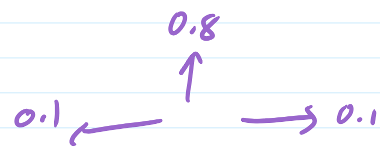
Temporal Difference (TD) Learning





at each step go $\uparrow, \downarrow, \leftarrow, \rightarrow$ with prob 0.8 go in intended dir
0.1 go in perp. dir
0.1 go in other perp. dir

model
$$\begin{cases} P(s, s', a) = \text{prob that from state } s\text{, action } a\text{, end up in state } s' \\ R(s, s', a) = \text{reward obtained going from } s \text{ to } s' \text{ having taken action } a \end{cases}$$

policy : function π : states $\longrightarrow$ action

expected reward given starting in s, following π

value of state s using policy π

$$V_\pi(s) = \begin{cases} \text{value determined by game} & \text{if } s \text{ is terminal} \\ \sum_{s'} P(s, s', \pi(s)) \cdot (R(s, s', \pi(s)) + V_\pi(s')) \end{cases}$$

π_{OPT} : policy π that maximizes $V_\pi(s)$ for each s

$$V(s) = V_{\pi_{OPT}}(s)$$

$Q(s,a)$ = value of taking action a from state s

Value iteration ($\equiv$ introduce time limit)

$$V(s) = \max_a \underbrace{\sum_{s'} P(s,s',a) \cdot (\underbrace{R(s,s',a)}_{\text{immediate reward}} + \gamma V(s'))}_{\text{total reward}}$$

future reward

γ → discount factor

$$\pi_{OPT}(s) = \arg\max_a Q(s,a)$$

$V(r,c,n)$ = value of state (r,c) w/ n steps left

$$= \begin{cases} -1 & \text{if } r,c = (1,3) \\ +1 & \text{if } r,c = (0,3) \\ 0 & \text{if } n = 0 \\ \max_a & .8 \cdot V((r,c)+a, n-1) \end{cases}$$

$$\max_a \; .8 \cdot V((r,c)+a,\, n-1) + .1\, V((r,c)+\tilde{a},\, n-1) + .1\, V((r,c)+\tilde{a}',\, n-1)$$

$V(r,c,n)$ converges to $V(r,c)$ as $n \uparrow$

temporal difference

TD Value Learning

pick some action using ε-greedy (prob. ε pick random action, $1-\varepsilon$ choose greedily)

observe $s \xrightarrow{a} s'$ and reward $R(s, s', a)$

use estimate of discount future reward $\gamma \cdot V(s')$

sampled value of $V(s) = R(s, s', a) + \gamma \cdot V(s')$

learning rate $0 < \alpha < 1$

update estimate $V(s) \leftarrow V(s) + \alpha\left(\underbrace{R(s,s',a) + \gamma \cdot V(s') - V(s)}_{\text{error (surprise)}}\right)$

$$= \underbrace{(1-\alpha) V(s) + \alpha\, R(s,s',a) + \gamma V(s')}_{\text{weighted avg of prev estimate and sample}}$$

Q Learning

$Q(s,a)$ = value of taking action a in state s
(expected reward from taking action a in state s
+ expected discounted future reward from resulting state)

$V(s) = \max_a Q(s,a)$

initialize $Q(s,a) = \begin{cases} R(s) & \text{for terminal } s \\ 0 & \text{otherwise} \end{cases}$

while not done

$s \leftarrow s_0$ initial state

episode

while s not terminal

choose action a ← ε-greedy

observe transition (s, a, r, s') — observed reward (r)

update $Q(s,a) \leftarrow Q(s,a) + \underbrace{\alpha}_{\text{learning rate}} \Big(\overbrace{r + \gamma \cdot \max_a Q(s',a) - Q(s,a)}^{\text{error}} \Big)$

(can go down as episodes ↑)

Q-learning converges $\sum_{t=1}^{\infty} \alpha_t(s,a) = \infty$ (so each s,a must be explored infinitely often)

$\sum_{t=1}^{\infty} \alpha_t^2(s,a) < \infty$ (must decrease α, but not so fast that 1st constraint not met)

Function Approximators

— instead of learning $Q(s,a)$ for every (s,a), learn fxn to approximate them

Linear Approximator

Define features of states or (state, action) pairs

$f_1(s,a)$ on pace to earn upper bonus 1.0 have already earned … 0.0 no chance

$\vdots$

$f_2(s,a)$ is chance unused

$f_3(s,a)$ both LS, SS unused

$\vdots$

$f_4(s,a)$ Y unused

$$Q(s,a) = \underline{w_1} \cdot f_1(s,a) + w_2 \cdot f_2(s,a) + \cdots + w_n \cdot f(s,a)$$

learn w_i instead of $Q(s,a)$ directly

In state s

Choose action a

Observe transition (s, a, r, s')

Update $Q(s,a) \leftarrow Q(s,a) + \alpha(\gamma \max_{a'} Q(s',a') - Q(s,a))$