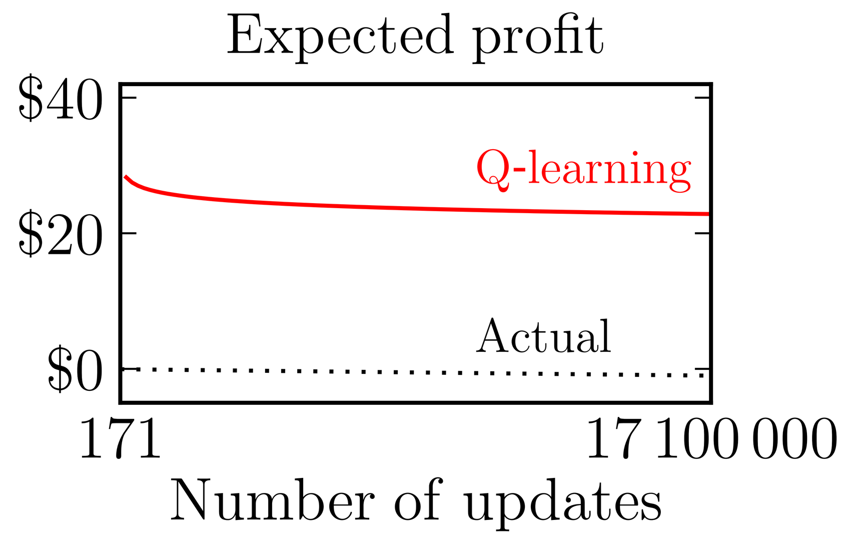
# Double Q-learning

## Or: How to solve Q-learning's gambling issues

Hado van Hasselt
Centrum voor Wiskunde en Informatica, Amsterdam, Netherlands

### Q-learning: bad performance in some noisy settings

- For instance, roulette (1 state, 171 actions)




Average action values for Q-learning on roulette with synchronous updates, $\alpha = 1/n$ and $\gamma = 0.95$.

- Q-learning after 17 million updates:
  - Each dollar will yield between \$22.60 and \$22.70
  - Better to gamble than to walk away

### Why?

- At time $t$: in state $s$ do action $a$, observe reward $r$ and state $s'$
- Q-learning:

$$Q_{t+1}(s,a) \leftarrow Q_t(s,a) + \alpha\left(r + \gamma \max_b Q_t(s',b) - Q_t(s,a)\right)$$

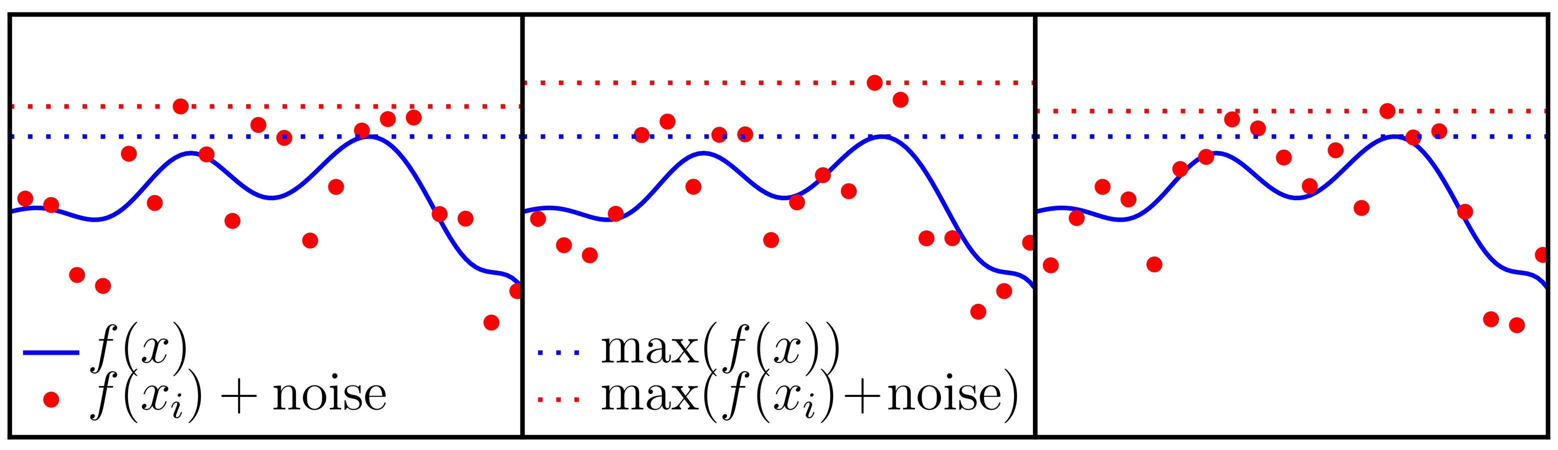
- $Q_t$ is **noisy approximation** of optimal $Q^*$
- But, in presence of noise:

$$E\left\{\max_b Q_t(s',b)\right\} > \max_b E\left\{Q_t(s',b)\right\}$$

- In other words: **Q-learning is biased**
- Bias is cumulative per update

### In general: single estimator

- Let $A_i$ denote an unbiased noisy sample of $x_i$
- In general: $E\{\max_i A_i\} \geq \max_i E\{A_i\} = \max_i x_i$
- Therefore, often: $\max_i A_i > \max_i x_i$




$f(x_i) + noise$ is unbiased sample of $f(x_i)$
$\max(f(x_i) + noise) > max(f(x))$ in all three examples

- Q-learning does this
  - Sarsa too, depending on policy
  - Value iteration too, but less noise $\longrightarrow$ less bias

### In general: double estimator

- Idea: use two sets of unbiased estimates: $A$ and $B$
- Select maximizing argument from one set: $a^* = \arg\max_i A_i$
- $A_{a^*}$ is **biased**: $E\{A_{a^*}\} \geq \max_i x_i \geq x_{a^*}$
- $B_{a^*}$ is **unbiased**: $E\{B_{a^*}\} = x_{a^*}$
- Unfortunately, unbiased for $x_{a^*}$, not for $\max_i x_i$
- In general: $E\{B_{a^*}\} = x_{a^*} \leq \max_i x_i$

### Double Q-learning

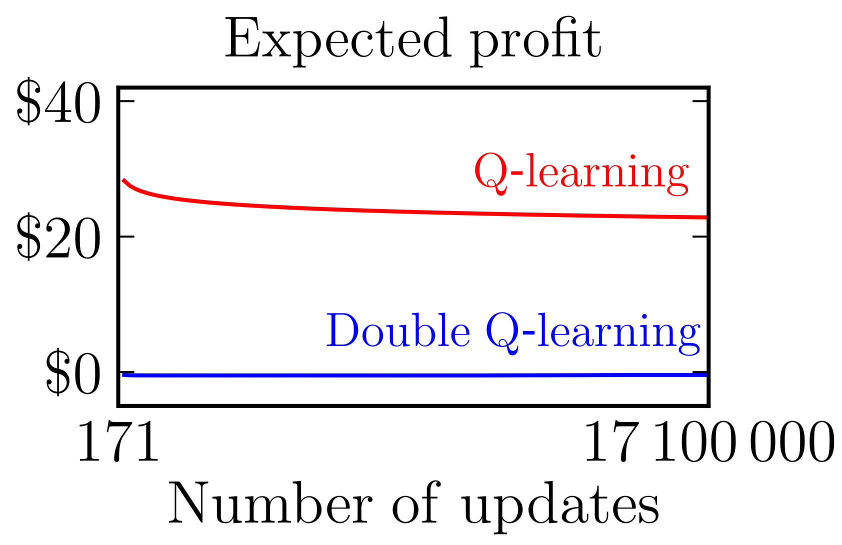
- Idea: apply double estimator to Q-learning
- Use two Q-functions: $Q^A$ and $Q^B$

$$Q^A_{t+1}(s,a) \leftarrow Q^A_t(s,a) + \alpha\left(r + \gamma Q^B_t(s',a^*) - Q^A_t(s,a)\right) \quad \text{or}$$
$$Q^B_{t+1}(s,a) \leftarrow Q^B_t(s,a) + \alpha\left(r + \gamma Q^A_t(s',b^*) - Q^B_t(s,a)\right)$$
$$\text{where} \quad a^* = \arg\max_a Q^A(s',a)$$
$$b^* = \arg\max_a Q^B(s',a)$$

- Each time step: update only one (e.g., random pick $Q^A$ or $Q^B$)
- Use average of $Q^A$ and $Q^B$ to choose actions
- Same time and space complexity as Q-learning
- Provably convergent

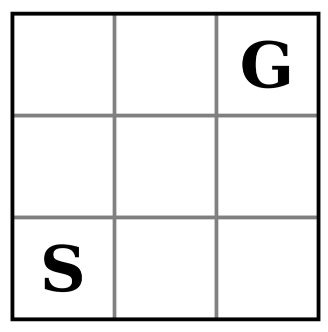
### Results: roulette

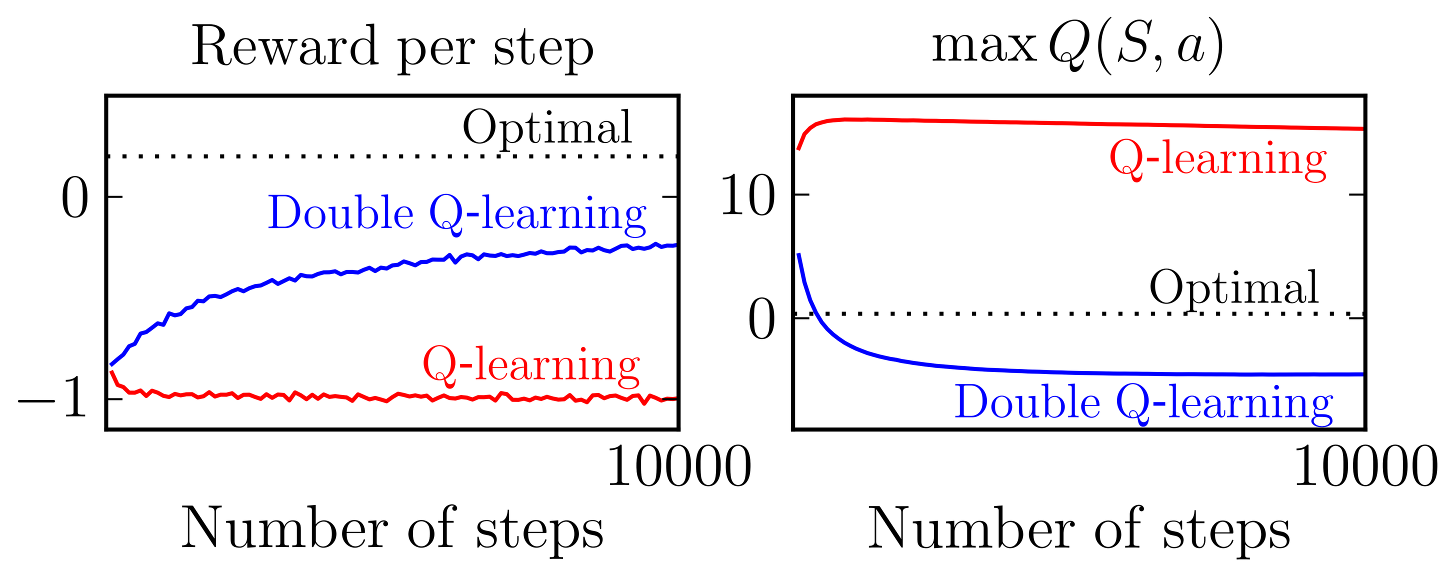
- Double Q-learning:
  - Much better than Q-learning
  - Better to walk away than to gamble




### Results: small grid world

- 9 states, 4 actions per state
- Start in S
- Reward in $s \neq G$: $-12$ or 10 (random)
- Reward in $s = G$: 5 (end episode)




Left: average reward per step. Right: maximal value in start state S.
$\epsilon$-greedy exploration, $\epsilon = 1/\sqrt{n}$, $\alpha = 1/n$, $\gamma = 0.95$

- Reward per step
  - Q-learning: very poor
  - Double Q-learning: much better, not perfect
- Maximum value in start state S
  - Q-learning: overestimation
  - Double Q-learning: underestimation

### Conclusion

- Q-learning is biased: sometimes (huge) overestimations
- Double Q-learning: sometimes underestimations
- Empirical evidence:
  Double Q-learning better in (some) noisy settings
- Future work: unbiased Q-learning?



http://homepages.cwi.nl/~hasselt/ H.van.Hasselt@cwi.nl