Universität Potsdam
Institut für Informatik
Lehrstuhl Maschinelles Lernen

Reinforcement Learning (Part 2)

Uwe Dick

Markov Decision Processes

- (Finite) Markov Decision Process: Tuple (S, A, R, P)
- S : Finite state space (set of states).
- A : Finite action space (set of actions).
- P : Transition probabilities.

$$P(s'|s,a) \;\; s, s' \in S, a \in A$$

- R : Expected Immediate Reward.

$$R : (S \times A) \to \mathbb{R}$$

- Discount factor $0 \leq \gamma < 1$.

Function Approximation

- Modelling
- Value Iteration
  - Least-Squares Methods
  - Gradient Methods
- Policy Iteration
  - Least-Squares Methods
  - Gradient Methods
- Policy Gradient
- Actor-Critic Methods

Function Approximation

- Represent Value Function as parametric function from function space F with parameter vector µ.

$$\hat{V}(s;\theta) \qquad \hat{Q}(s,a;\theta)$$

- Approximation e.g. with linear function
  - $\hat{Q}(s,a;\theta) = \phi_{s,a}^{\top}\theta$ with feature vector $\phi_{s,a}$ and parameter vector θ.
  - $\hat{V}(s;\theta) = \phi_{s}^{\top}\theta$ with feature vector ϕ_s and parameter vector θ.
- Or you could learn a deep neural net.

Function Approximation

- Prediction problem: Find parameter vector μ, such that V^π, or Q^π resp., will be approximated best.

$$\theta^\pi = \arg\min_\theta \sum_s \mu(s) \left| V^\pi(s) - \hat{V}(s;\theta) \right|^2$$

$$\theta^\pi = \arg\min_\theta \sum_s \sum_a \mu(s) \left| Q^\pi(s,a) - \hat{Q}(s,a;\theta) \right|^2$$

- Unfortunately, we normally do not know V^π, or Q^π, resp.

Function Approximation

- Generative Model
  - Assumption: We can sample from P and R at time.
  - But we cannot query $P(s‘|s,a)$ directly.

- The Reinforcement Learning Problem:
  - Examples $< s_t, a_t, R_t, s_{t+1} >$ from interaction with the environment.
  - One possibility: Interaction according to policy we are currently learning.
    - On-policy distribution of states $\mu(s)$.

Batch Reinforcement Learning

- Episode sampled according to (behavior) policy π_b.
- At time of training only this one episode is available.

Approximate Policy Iteration

- Initialize Policy π_0

- Iterate
  - Approximate Policy Evaluation

    Find $\hat{Q}$, an element of F, as an approximation of Q^{π} via
    - Interaction
    - Fixed training set

  - Policy Improvement

    Find π_{t+1}, such that $\pi_{t+1} \approx \arg\max_a \hat{Q}(s,a), \text{ for all } s$

Approximate Policy Iteration

- If samples of $Q^{\pi}(s,a)$ are known, learn Q^{π} based on training sample with the help of a supervised regression method.

- Problem: Often training examples are sampled off-policy, that is, examples are observed when following a (different) behavior policy.
  - Sample Selection Bias (Differing training and test distributions $p(s,a)$)

Approximate Policy Evaluation

- Idea: Minimize the quadratic difference between Q and TQ.
  - Mean Squared Bellman Error

- Unfortunately TQ is not necessarily in F.

- Better: Minimize

$$\|Q - \Pi T^{\pi} Q\|_{\mu}^{2}$$

More on that later!

  - Mean Squared Projected Bellman Error

Bellman Residual Minimization

- Temporal Difference method.
- Bellman equation as fixed point equation.

$$Q^\pi - T^\pi Q^\pi = 0$$

- Interpret left hand side as error: Bellman Residual. μ is stationary distribution of states.

$$L_{BRM}(Q;\pi) = \|Q - T^\pi Q\|_\mu^2$$

- Empirical: Draw samples $< s_t, a_t, R_t, s_{t+1} >$, e.g. by using π in the environment.

$$\begin{aligned}&\hat{L}_{BRM}(Q;\pi,n)\\ &= \frac{1}{n|A|}\sum_{t=1}^{n}\left[Q(s_t,a_t) - \left(R(s_t,a_t) + \gamma Q(s_{t+1},\pi(s_{t+1}))\right)\right]^2\end{aligned}$$

Bellman Residual Minimization

- Problem: Estimation $\hat{L}_{BRM}$ is biased.

$$E[\hat{L}_{BRM}(Q;\pi,n)] \neq L_{BRM}(Q;\pi)$$

- Because

$$L_{BRM}(Q;\pi) = E_{s\sim\mu,a}\left[\left(Q(s,a) - T^{\pi}Q(s,a)\right)^2\right]$$

$$(T^{\pi}Q)(s,a) = E_{s'\sim P}\left[R(s,a) + \gamma Q(s',\pi(s'))\right]$$

- It follows:

$$\begin{aligned} & L_{BRM}(Q;\pi) \\ = \quad & E_{s\sim\mu,a}\left[\left(Q(s,a) - E_{s'\sim P}\left[R(s,a) + \gamma Q(s',\pi(s'))\right]\right)^2\right] \end{aligned}$$

Bellman Residual Minimization

- But let $E[\hat{L}(Q;\pi)] = \hat{L}(Q;\pi,n)$, für $n \to \infty$. Then

$$\begin{aligned} & E_{s\sim\mu,a,s'\sim P}\left[\hat{L}_{BRM}(Q;\pi)\right] \\ = \quad & E_{s\sim\mu,a,s'\sim P}\left[\left(Q(s,a) - R(s,a) + \gamma Q(s',\pi(s'))\right)^2\right] \\ = \quad & E_{s\sim\mu,a}\left[E_{s'\sim P}\left[\left(Q(s,a) - R(s,a) + \gamma Q(s',\pi(s'))\right)^2\right]\right] \end{aligned}$$

- But the following holds for expected values of random variables X:

$$E[X^2] = E[X]^2 + \mathrm{Var}[X]$$

Bellman Residual Minimization

- Application of inner expected value:

$$
\begin{aligned}
&E_{s'}\left[\Big(Q(s,a)-\big(R(s,a)+\gamma Q(s',\pi(s'))\big)\Big)^2\right] \\
&= \quad E_{s'}\Big[Q(s,a)-\big(R(s,a)+\gamma Q(s',\pi(s'))\big)\Big]^2 \\
&\qquad +\mathrm{Var}_{s'}\Big[Q(s,a)-\big(R(s,a)+\gamma Q(s',\pi(s'))\big)\Big] \\
&= \quad \Big(Q(s,a)-(T^\pi Q)(s,a)\Big)^2 \\
&\qquad +\mathrm{Var}_{s'}\Big[R(s,a)+\gamma Q(s',\pi(s'))\Big]
\end{aligned}
$$

- It follows that the estimation based on samples as above is biased.

$$
\begin{aligned}
&\hat{L}_{BRM}(Q;\pi,n) \\
&= \quad \frac{1}{n|A|}\sum_{t=1}^{n}\Big[Q(s_t,a_t)-\big(R(s_t,a_t)+\gamma Q(s_{t+1},\pi(s_{t+1}))\big)\Big]^2
\end{aligned}
$$

Bellman Residual Minimization

- How can we get an unbiased sample of

$$L_{BRM}(Q;\pi) = \|Q - T^{\pi}Q\|_{\mu}^{2} \quad ?$$

- Define the sampled temporal difference error

$$\delta_t(\theta) = \hat{Q}(s_t, a_t; \theta) - (R(s_t, a_t) + \gamma \hat{Q}(s_{t+1}, a_{t+1}; \theta))$$

Bellman Residual Minimization

- Alternative derivation for BRM with function approximation.
  - Optimization criterion. Minimize

$$J(\theta)$$

$$= E_{s_t \sim \mu, a}[(\hat{Q}(s_t, a_t; \theta) - (R(s_t, a_t) + E_{s_{t+1}}[\gamma \hat{Q}(s_{t+1}, a_{t+1}; \theta)]))^2]$$

$$= E_{s_t \sim \mu}[E_{a, s_{t+1}}[\delta_t(\theta) \mid s_t]^2]$$

  - Gradient

$$\frac{1}{2} \nabla J(\theta)$$

$$= E_{s_t, a, s_{t+1}}[\delta_t(\theta) \nabla \hat{Q}(s_t, a_t; \theta)]$$

$$- E_{s_t}[E_{a, s_{t+1}}[\delta_t(\theta) \mid s_t] E_{s_{t+1}}[\gamma \nabla \hat{Q}(s_{t+1}, a_{t+1}; \theta) \mid s_t]]$$

Bellman Residual Minimization

- That is why we should draw two independent next states in order to get an unbiased estimate
  $\rightarrow$ Not very practical and often unrealistic.

- Instead, it was proposed to just draw one sample of next state and make the updates with only this sample.
- But then the algorithm does not converge towards the fixed point of T^π.

  $$Q^\pi - T^\pi Q^\pi = 0$$

  any more.

Residual Gradient

- This new solution is called the Residual Gradient (RG) solution.

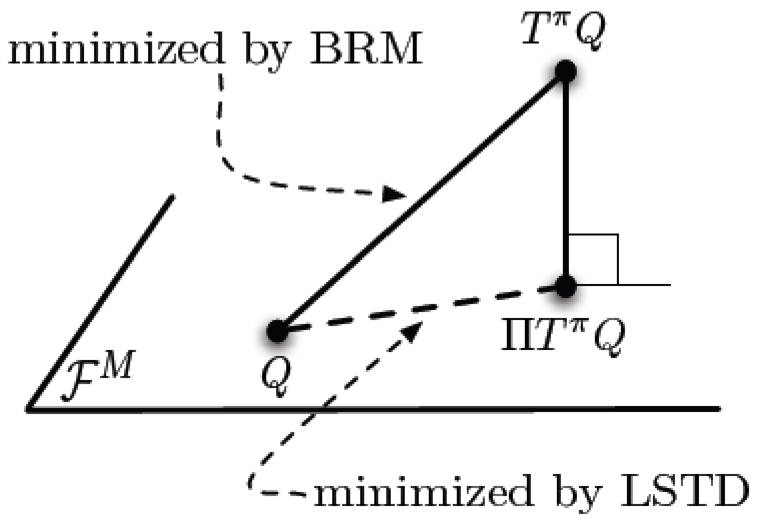
- Usually this solution is not as good as the exact BRM solution.

- On the bright side, we compute a stochastic ‚real' gradient (though not for the original optimization problem)
  - (more on that later)

BRM

- Proposition: [Antos et. al. 07]
  Unbiased estimation with the help of a helper function $h \in F$.

$$L_{BRM}(Q, h; \pi) = \|Q - T^\pi Q\|_\mu^2 - \|h - T^\pi Q\|_\mu^2$$



Least-Squares Temporal Difference

- Q is a member of function space F.
- $T^{\pi}Q$ might not.
- LSTD minimizes the quadratic distance between Q and the projection of on $T^{\pi}Q$ on F.

$$L_{LSTD}(Q;\pi) = \|Q - \Pi T^{\pi} Q\|_{\mu}^{2}$$

$$\Pi f = \arg\min_{h \in \mathcal{F}} \|h - f\|_{\mu}$$

- Unbiased.
- LSTD often gives better results.

Fitted Policy Evaluation with Samples

- $Q = 0.$
- Draw N samples s, a from $\mu(s)$,$p(a)$. Draw M samples of R and next state s' according to the model.
- Iterate:
  - Use those samples $< s, a, R, s‘ >$ to do a Bellman update step:

$$Q_{k+1}(s,a) \leftarrow \sum_{i=1}^{M} R(s,a) + \gamma Q_k(s', \pi(s'))$$

  - Do least-squares fitting:

$$\hat{Q}_{k+1}(s,a) \leftarrow \arg\min_{f \in \mathfrak{I}} \sum_{i=1}^{M} \left\| Q_{k+1}(s_i, a_i) - f(s_i, a_i) \right\|_2^2$$

Projected Bellman Equation

- Linear approximation of Q function

$$\hat{Q}(s,a;\theta)=\phi^T(s,a)\theta$$

- Fixed point of Bellman equation $Q=\Pi\circ T\circ Q$
- Projection: Least-Squares
- Define

$$\mathbf{Q}\in\mathbb{R}^{|S||A|} \qquad \mathbf{Q}(ij)=Q(s_i,a_j)$$

$$\mathbf{r}\in\mathbb{R}^{|S||A|} \qquad \mathbf{r}(ij)=R(s_i,a_j)$$

$$\mathbf{p}\in\mathbb{R}^{|S||A|\text{x}|\text{S}|} \qquad \mathbf{p}(ij,i')=P(s_{i'}\mid s_i,a_j)$$

$$\boldsymbol{\pi}\in\mathbb{R}^{|\text{S}|\text{x}|\text{S}||A|} \qquad \boldsymbol{\pi}(i',ij)=\pi(a_j\mid s_i),\ \text{if } i=i',\ \text{otherwise}\ \ 0$$

Projected Bellman Equation

- Linear approximation $\mathbf{Q} = \phi\theta$
- Define $\mathbf{T}^{\pi}(\mathbf{Q}) = \mathbf{r} + \gamma \mathbf{p}\boldsymbol{\pi}\mathbf{Q}$

$$\Pi = \phi(\phi^T\phi)^{-1}\phi^T$$

- It follows $\mathbf{Q} = \Pi\mathbf{T}^{\pi}(\mathbf{Q})$

- With the help of $\Gamma = \phi^T\phi,\ \Lambda = \phi^T\mathbf{p}\boldsymbol{\pi}\phi,\ z = \phi^T\mathbf{r}$,

Q^{π} can be computed as the solution of

$$\Gamma\theta^{\pi} = \gamma\Lambda\theta^{\pi} + z$$

LSTD-Q

- Given a set of n samples (s_i, a_i, s_i')

$$\Gamma_0 = 0, \quad \Lambda_0 = 0, \quad z_0 = 0$$

For i=1…n

$$\Gamma_i = \Gamma_{i-1} + \phi(s_i, a_i)\phi^T(s_i, a_i)$$

$$\Lambda_i = \Lambda_{i-1} + \phi(s_i, a_i)\phi^T(s'_i, \pi(s'_i))$$

$$z_i = z_{i-1} + \phi(s_i, a_i) r_i$$

Solve for $\hat{\theta}^\pi$

$$\frac{1}{n}\Gamma_n \hat{\theta}^\pi = \gamma \frac{1}{n}\Lambda_n \hat{\theta}^\pi + \frac{1}{n} z_n$$

LSPE-Q

- Given a set of n samples (s_i, a_i, s_i')

```
$\Gamma_0 = 0, \quad \Lambda_0 = 0, \quad z_0 = 0$
For i=1...n
    $\Gamma_i = \Gamma_{i-1} + \phi(s_i, a_i)\phi^T(s_i, a_i)$
    $\Lambda_i = \Lambda_{i-1} + \phi(s_i, a_i)\phi^T(s'_i, \pi(s'_i))$
    $z_i = z_{i-1} + \phi(s_i, a_i) r_i$
    $\theta_i \leftarrow \theta_{i-1} + \alpha(\theta_i - \theta_{i-1})$ where $\frac{1}{i}\Gamma_i\theta_i = \gamma\frac{1}{i}\Lambda_i\theta_{i-1} + \frac{1}{i}z_i$
$\hat{\theta}^\pi = \theta_n$
```

Convergence

- For $n \to \infty$, both solutions $\hat{\theta}_i^\pi$ converge to the solution θ_i^π of the projected Bellman equation. Convergence can be proven for both algorithm if samples are drawn from π.
- Intuitively correct because

$$\frac{1}{n}\Gamma_n \to \Gamma, \qquad \frac{1}{n}\Lambda_n \to \Lambda, \qquad \frac{1}{n}z_n \to z$$

- But π should not be deterministic. Use e.g. ϵ-greedy policy instead.

Least-Squares Methods

- LSPE-Q is an incremental algorithm. It could be used as an online on-policy method. That way it could e.g. make use of a good starting point $\hat{\theta}_i^{\pi}$.

- LSTD-Q on the other hand computes the solution in one step. This generally means better stability in comparison to LSPE.

Gradient Based Policy Evaluation

- Another option for (online) policy evaluation are gradient based methods.

$$\theta_{t+1} = \theta_t - \frac{1}{2}\alpha_t \nabla_\theta \left[Q^\pi(s_t, a_t) - \hat{Q}(s_t, a_t; \theta_t) \right]^2$$

$$= \theta_t + \alpha_t \left[Q^\pi(s_t, a_t) - \hat{Q}(s_t, a_t; \theta) \right] \nabla_\theta \hat{Q}(s_t, a_t; \theta_t)$$

- $Q^\pi(s_t, a_t)$ is unknown. We approximate it using a stochastic approximation of a Bellman update (1 Sample)

$$R(s,a) + \gamma \hat{Q}(s', a')$$

TD(0)

- The following update rule for TD(0) follows:

$$\theta_{t+1} = \theta_t + \alpha_t \left[R(s_t, a_t) + \gamma \hat{Q}(s_{t+1}, a_{t+1}; \theta) - \hat{Q}(s_t, a_t; \theta) \right] \nabla_\theta \hat{Q}(s_t, a_t; \theta_t)$$

- Special case for linear function approximation

$$\theta_{t+1} = \theta_t + \alpha_t \left[R(s_t, a_t) + \gamma \phi^T(s_{t+1}, a_{t+1})\theta_t - \phi^T(s_t, a_t)\theta_t \right] \phi(s_t, a_t)$$

- Again, it holds that the policy should not be deterministic.

- But: This is not a sample of a real gradient!

Gradient Based Policy Evaluation

- „Real“ gradient is Residual Gradient

$$E_{s_t, s_{t+1}}[\delta_t(\theta)\nabla\hat{Q}(s_t, a_t; \theta)]$$
$$-E_{s_t}[E_{s_{t+1}}[\delta_t(\theta) \mid s_t]E_{s_{t+1}}[\nabla\hat{Q}(s_{t+1}, a_{t+1}; \theta) \mid s_t]]$$

- Empirical gradient (1 Sample)

$$\left[R(s_t, a_t) + \gamma\phi^T(s_{t+1}, a_{t+1})\theta_t - \phi^T(s_t, a_t)\theta_t\right]$$
$$\cdot(\gamma\phi^T(s_{t+1}, a_{t+1}) - \phi(s_t, a_t))$$

Convergence

- Convergence for TD(0) can be shown under specific assumptions and using an on-policy sample process.
- Like always for on-policy learning: Use stochastic policy.
- In order to learn a (nearly) deterministic policy, the epsilon of an ϵ-greedy policy can be adjusted (lowered) over time.
- Unfortunately, for very small epsilon the convergence is not guaranteed anymore!

Convergence

- Residual Gradient has better convergence properties. One reason for that is that it is a ‚real' gradient.

- For off-policy TD(0) policy evaluation, no convergence can be shown. (It might still converge in practice.)

Off-Policy Policy Evaluation

- Off-policy policy evaluation means that the samples (s_t, a_t, s_{t+1}) are sampled using a behavior policy π_b while we want to learn the value function of another policy π.
- π_b is generally stochastic in order to guarantee that all state-action pairs are observed.

- Practical realization dependent on π_b
  - If π_b deterministic, realization via subsampling.
  - If π_b stochastic, realization via importance sampling.

Off-Policy Policy Evaluation

- Subsampling:
  - Ignore all examples (s_t, a_t, s_{t+1}) that are not in line with policy π.

- Importance Sampling:
  - Reweight each example in the update step. E.g. TD(0): $E[\delta_t(\theta)\phi_t] \rightarrow E[\rho_t \delta_t(\theta)\phi_t]$

$$\rho_t = \frac{\pi(s_t, a_t)}{\pi_b(s_t, a_t)}$$

  - TD(0) with Importance Sampling:

$$\theta_{t+1} = \theta_t + \alpha_t \rho_t \delta_t(\theta)\phi(s_t, a_t)$$

Divergence of Q-Learning

The behavior policy, in this example, chooses the solid line action with probability of $1/7$ and the dotted line action with probability of $6/7$. The goal is to learn the value of a target policy that chooses the solid line more often than the probability of $1/7$. In this example, the target policy choose the solid action with probability of 1.

The value functions are approximated linearly in the form of $V(i) = 2\theta(i) + \theta_0$, for $i \in \{1, ..., 6\}$, and $V(7) = \theta(7) + 2\theta_0$. Here, the discount factor is $\gamma = 0.99$. The TD-solution, in this example, is $\theta(i) = 0$, $i \in \{1, ..., 7\}$, and $\theta_0 = 0$. Both TD(0) and DP (with incremental updates), however, will diverge on this example; that is, their learning parameters will go to $\pm\infty$ as is illustrated in Fig. 2.5.

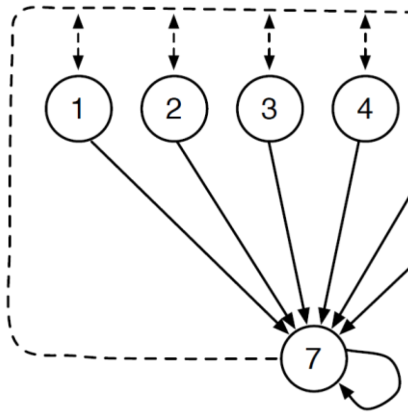


Figure 2.4: The Baird's 7-star MDP. Every transition in this MDP receives zero reward. Each state, has two actions, represented by solid line and dotted line. Solid line action only make transition to state state 7, while dotted line action uniformly make transition to one of states 1-6 with probability of $1/6$.

Off-Policy Policy Evaluation

- New algorithms that also show convergence under off-policy learning.
- E.g. GTD algorithm. (Gradient TD)
- Idea: New Optimization criterion: Minimize the l_2-norm of TD(0) updates.

$$E[\delta_t(\theta)\phi_t]^T E[\delta_t(\theta)\phi_t]$$

- Because update can be considered as error of current solution θ, this minimization criterion makes intuitive sense.

GTD-Algorithm

- The gradient can be computed as follows:

$$\nabla_\theta E[\delta(\theta)\phi^T]E[\delta(\theta)\phi] = 2E[\nabla_\theta \delta(\theta)\phi^T]E[\delta(\theta)\phi]$$
$$= 2E[(\gamma\phi' - \phi)\phi^T]E[\delta(\theta)\phi]$$

- For approximations based on samples (for stochastic gradient), only one of the two expectations will be sampled directly.

$$\theta_{t+1} = \theta_t + \alpha_t(\phi_t - \gamma\phi_t{}^T)\phi_t{}^T u_t$$
$$u_{t+1} = u_t + \beta_t(\delta_t\phi_t - u_t)$$

Approximate PI – Theoretical Guarantees

- Convergence of approximate Policy Iteration algorithms is not provable in general.

- It is possible to state suboptimality bounds, depending on errors in policy evaluation and policy improvement steps.

Approximate PI – Theoretical Guarantees

- If the error for Policy Evaluation is upper bounded by

$$\left\| \hat{Q}^{\hat{\pi}_t} - Q^{\hat{\pi}_t} \right\|_\infty \leq \varsigma_Q$$

- If the error for Policy Improvement is upper bounded by

$$\left\| T^{\hat{\pi}_t}(\hat{Q}^{\hat{\pi}_t}) - T(\hat{Q}^{\hat{\pi}_t}) \right\|_\infty \leq \varsigma_\pi$$

- Then the overall error can be upper bounded by

$$\limsup_{t\to\infty} \left\| \hat{Q}^{\hat{\pi}_t} - Q^* \right\|_\infty \leq \frac{\varsigma_\pi + 2\gamma\varsigma_Q}{(1-\gamma)^2}$$

Approximate Value Iteration

- Analoguously to approximate Policy Evaluation: Find fixed point of Bellman equations for the control problem.

$$Q = \Pi \circ T \circ Q$$

- With parametric approximation with parameter vector θ:

$$\theta = \Pi \circ T \circ Q(\theta)$$

Approximate Value Iteration

- The Bellman operator is defined as

$$T(Q)(s,a) = R(s,a) + \gamma \max_a E_{s'}[Q(s',a)]$$

- The projection is realized using least-squares approximation

$$\Pi(Q) = \min_{f \in F} \|Q - f\|_2^2$$

Fitted Value Iteration with Samples

- [Szepesvári & Munos 05]
- $V = 0$.
- Draw N states s according to $\mu(s)$.
- For each s and $a \in A$, draw M next states s‘ from $P(s'|s,a)$ and rewards $R(s,a)$.
- Iterate:
  - Use those sample $< s, a, R, s‘ >$ to perform one Bellman update step:

$$V_k(s) \leftarrow \max_a \left[\frac{1}{M} \sum_{i=1}^{M} R_i(s,a) + \gamma V(s_i') \right]$$

  - least-squares fitting:

$$V \leftarrow \arg\min_{f \in \mathcal{F}} \sum_{i=1}^{N} |V_k(s_i) - f(s_i)|^2$$

Error Estimation

$$\|V^* - V^{\pi_K}\|_{p,\rho} \leq$$

$$\frac{2\gamma}{(1-\gamma)^2} \left\{ C(\mu)^{1/p} \left[d(T\mathcal{F}, \mathcal{F}) + \right. \right.$$

$$c_1 \left(\frac{\mathcal{E}}{N} (\log(N) + \log(K/\delta)) \right)^{1/2p} +$$

$$\left. c_2 \left(\frac{1}{M} (\log(N|A|) + \log(K/\delta)) \right)^{1/2} \right] +$$

$$\left. c_3 \gamma^K K_{\max} \right\}$$

Approximate Q-Learning

- Linear parametrization of Q-function.
- In each iteration, update:

$$\theta_{t+1} = \theta_t - \frac{1}{2}\alpha_t \nabla_\theta \left[Q^*(s_t, a_t) - \hat{Q}(s_t, a_t; \theta_t) \right]^2$$

- Analogously to TD(0), the unknown $Q^*(s_t, a_t)$ will be approximated with a Bellman update

$$Q^*(s_t, a_t) \approx R(s_t, a_t) + \gamma E_{s_{t+1}}[\max \hat{Q}(s_{t+1}, a; \theta_t)]$$

Approximate Q-Learning

- Stochastic gradient method. The expectation is approximated by stochastic samples.

$$\theta_{t+1} = \theta_t + \alpha_t \left[R(s_t, a_t) + \gamma \max_a \hat{Q}(s_{t+1}, a; \theta_t) - \hat{Q}(s_t, a_t; \theta_t) \right] \nabla_\theta \hat{Q}(s_t, a_t; \theta_t)$$

$$= \theta_t + \alpha_t \left[R(s_t, a_t) + \gamma \max_a \phi(s_{t+1}, a)^T \theta_t - \phi(s_t, a_t)^T \theta_t \right] \phi(s_t, a_t)$$

- It is possible to show convergence of approximate Q-Learning under specific assumptions. Restrictions on type of policy and feature vectors have to be met.

Approximate Value Iteration

- Proofs of convergence for Value Iteration are usually based on provability of non-expansion mapping of projection and function approximation, so that

$$\left\|(\Pi \circ T \circ Q)(\theta) - (\Pi \circ T \circ Q)(\theta')\right\|_\infty \leq \gamma \left\|\theta - \theta'\right\|_\infty$$

- I.e.:

$$\left\|Q(\theta) - Q(\theta')\right\|_\infty \leq \left\|\theta - \theta'\right\|_\infty$$

and

$$\left\|\Pi(Q) - \Pi(Q')\right\|_\infty \leq \left\|Q - Q'\right\|_\infty$$

Approximate Value Iteration

- $\left\| Q(\theta) - Q(\theta') \right\|_\infty \le \left\| \theta - \theta' \right\|_\infty$ can be guaranteed by normalizing the feature vectors ϕ.

$$\sum_{i=1}^{N} \phi_i = 1$$

- Projection is harder to show.

Approximate SARSA

- On-Policy method for control problem.

$$\begin{aligned}\theta_{t+1} &= \theta_t + \alpha_t \left[R(s_t, a_t) + \gamma \hat{Q}(s_{t+1}, a_{t+1}; \theta_t) - \hat{Q}(s_t, a_t; \theta_t) \right] \nabla_\theta \hat{Q}(s_t, a_t; \theta_t) \\ &= \theta_t + \alpha_t \left[R(s_t, a_t) + \gamma \phi(s_{t+1}, a_{t+1})^T \theta_t - \phi(s_t, a_t)^T \theta_t \right] \phi(s_t, a_t)\end{aligned}$$

- Convergence can be shown under mild assumptions. But convergence only holds if decisions are not ‚too deterministic'. (e.g. only with large enough epsilon in ϵ-greedy policies)

TD(¸)

$$\begin{array}{lllllll}
 & & R_0 & R_1 & R_2 & R_3 \quad \dots & R_k \\
(1-\lambda) & \Delta_1 : & R_0 + \gamma V(s_1) & & & & \\
(1-\lambda)\lambda & \Delta_2 : & R_0 + \gamma R_1 + & \gamma^2 V(s_2) & & & \\
(1-\lambda)\lambda^2 & \Delta_3 : & R_0 + \gamma R_1 + & \gamma^2 R_2 + & \gamma^3 V(s_3) & & \\
 & & & \vdots & & & \\
(1-\lambda)\lambda^{k-1} & \Delta_k : & R_0 + \gamma R_1 + & \gamma^2 R_2 + & \gamma^3 R_3 & + \dots & + \gamma^k V(s_k) \\
 & & & \vdots & & &
\end{array}$$

- Update rule: $V(s_t) \leftarrow (1-\alpha_t)V(s_t) + \alpha_t \Delta V(s_t)$
- TD(¸) update: $\Delta(\lambda) = \sum_{k=1}^{\infty} (1-\lambda)\lambda^{k-1} \Delta_k V(s_0)$
- 0·¸·1 interpolates between 1-step and MC.

Eligibility Traces

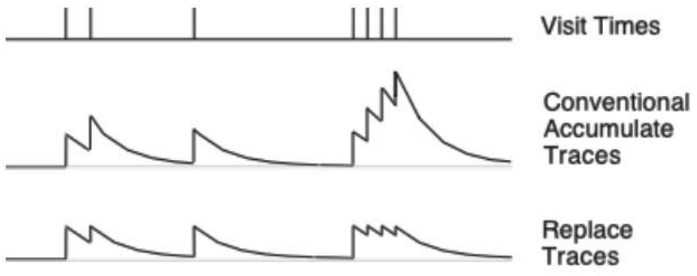
- Algorithmic view on TD(¸)
- Use additional variable e(s) for every state s2S.

- After observation <st,at,Rt,st+1>, compute

$$\begin{aligned} \delta_t &\leftarrow R_t + \gamma V(s_{t+1}) - V(s_t) \\ e(s_t) &\leftarrow e(s_t) + 1 \end{aligned}$$

- Update for all states

$$\begin{aligned} V(s) &\leftarrow V(s) + \alpha_t \delta_t e(s) \\ e(s) &\leftarrow \lambda\gamma e(s) \end{aligned}$$



FA for Reinforcement Learning

- TD(¸)
  - Eligibility traces:

$$\begin{aligned}\delta_t &\leftarrow R_t + \gamma V(s_{t+1}) - V(s_t)\\ e_t &\leftarrow \gamma\lambda e_{t-1} + \nabla_\theta V(s_t)\\ \theta_{t+1} &\leftarrow \alpha_t \delta_t e_t\end{aligned}$$

  - Linear method: convergence guarantee only on-policy.
  - Error estimation:

$$\lim_{t\to\infty} \sum_s \mu(s)|V^\pi(s) - \hat{V}(s;\theta_t)|^2$$

$$\leq \frac{1-\gamma\lambda}{1-\gamma} \sum_s \mu(s)|V^\pi(s) - \hat{V}(s;\theta^*)|^2$$

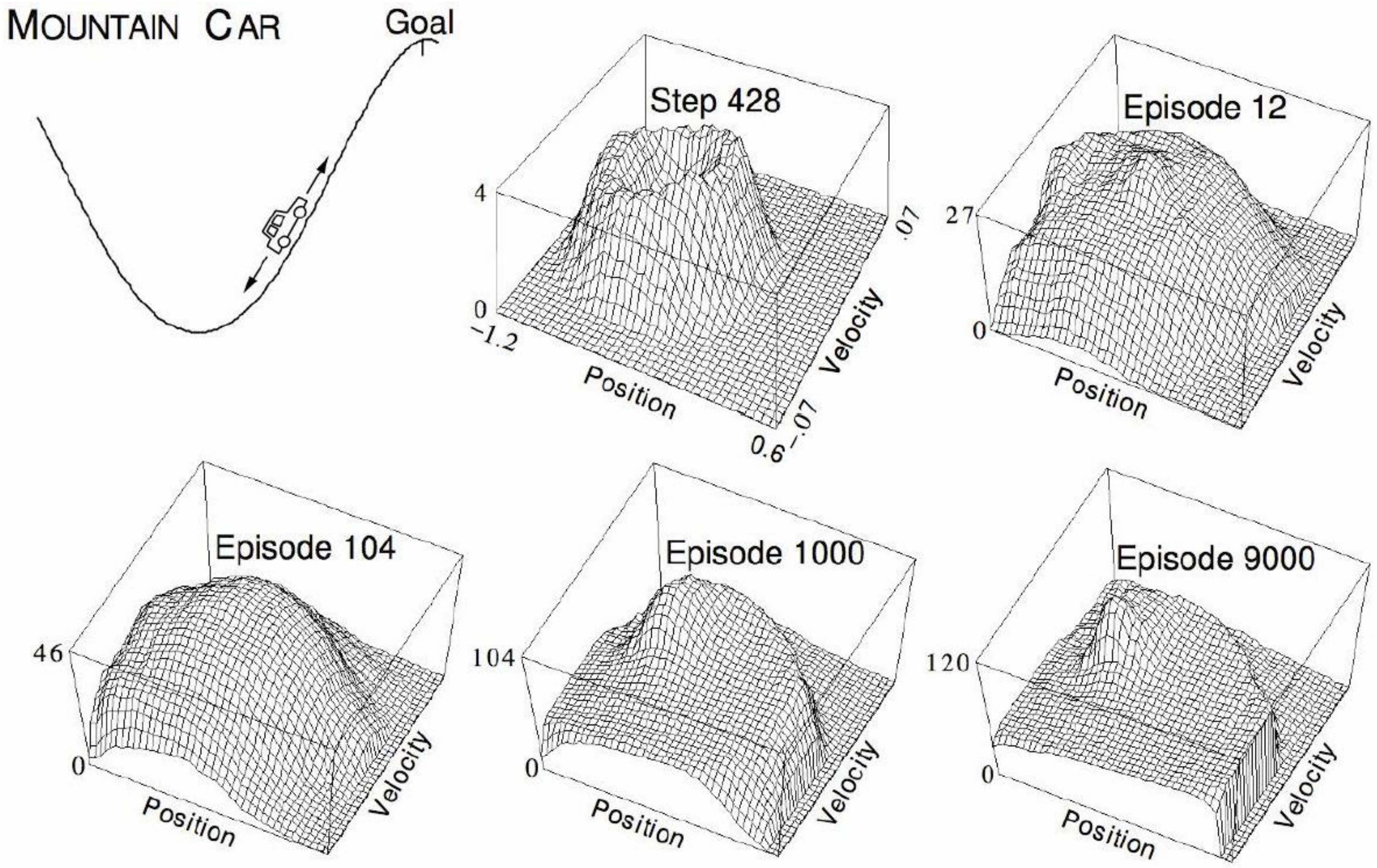
SARSA(¸)

- Control problem: SARSA(¸) (On-Policy)

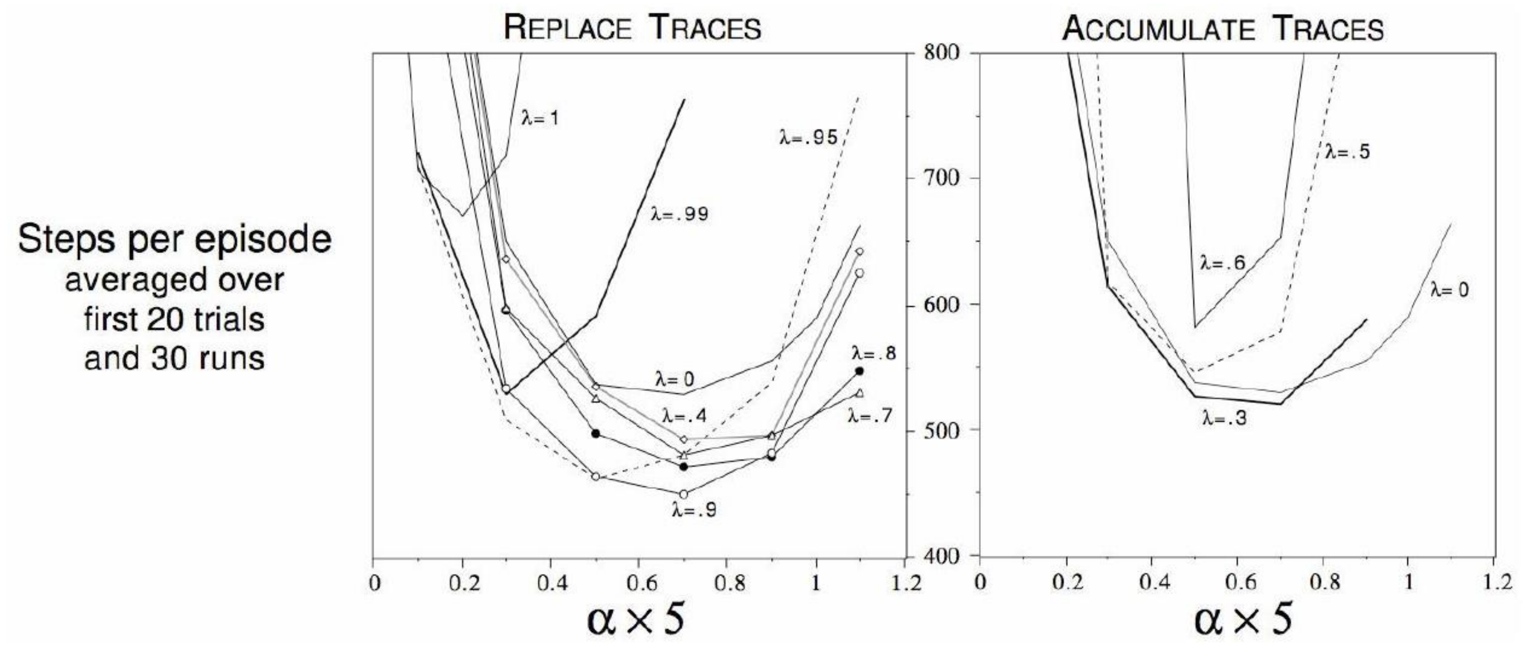
$$\begin{aligned}
\delta_t &\leftarrow R(s_t, a_t) + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \\
e_t &\leftarrow \gamma \lambda e_{t-1} + \nabla_\theta Q(s_t, a_t) \\
\theta_{t+1} &\leftarrow \alpha_t \delta_t e_t
\end{aligned}$$

- Off-policy can diverge.

SARSA(¸)


MOUNTAIN CAR
Goal
Step 428
4
0
−1.2
Position
0.6
Velocity
.07
−.07
Episode 12
27
0
Episode 104
46
Episode 1000
104
Episode 9000
120

SARSA(¸)


Steps per episode
averaged over
first 20 trials
and 30 runs
Replace Traces
Accumulate Traces
λ=1
λ=.99
λ=.95
λ=0
λ=.4
λ=.9
λ=.8
λ=.7
λ=.5
λ=.6
λ=0
λ=.3
800
700
600
500
400
0
0.2
0.4
0.6
0.8
1
1.2
α×5

Policy Gradient

- Learn stochastic policy.
- The policy will be represented explicitly, e.g. as Gibbs distribution

$$\pi(a|s;\theta) = \frac{e^{\phi_{s,a}\theta}}{\sum_{a'} e^{\phi_{s,a'}\theta}}$$

- Learn π, such that $E[\sum_{t=1}^{\infty} \gamma^t r_t | x_0, \pi]$ will be maximized.

- Idea: (stochastic) gradient method

$$\theta_{t+1} = \theta_t + \alpha_t \nabla_\theta L(\theta_t)$$

Policy Gradient

- The gradient is defined with the help of the Policy Gradient Theorem

- Define discounted state probabilities

$$d^\pi(x) = (1-\gamma)\sum_{t=1}^{\infty} \gamma^{t-1} p(x_t = x|\pi)$$

- Then

$$L(\theta) \quad = \quad \int_x d^\pi(x) \sum_a \pi(a|x;\theta) R(x,a) dx$$

Policy Gradient Theorem

- From the Policy Gradient Theorem:
  - Let L be defined as above. The gradient is defined as

$$\nabla_\theta L(\theta) \quad = \quad \int_x d^\pi(x) \sum_a \nabla_\theta \pi(a|x;\theta) Q(x,a) dx$$

- That is, the gradient will be computed based on the same expected state distribution and the value function.

Policy Gradient: Log-Trick

- The gradient can be rewritten with the help of the „log-trick“

$$\nabla_\theta L(\theta) \quad = \quad \int_x d^\pi(x) \sum_a \pi(a|x;\theta) \nabla_\theta \log \pi(a|x;\theta) Q(x,a) dx$$

- It follows for the empirical gradient

$$\nabla_\theta L(\theta) \quad = \quad \sum_{t=0}^{\infty} \gamma^{t-1} \sum_{x_t^i} \sum_a \nabla_\theta \log \pi(a|x_t^i;\theta) Q(x_t^i, a)$$

The gradient will be approximated without bias, if the samples are drawn from the on-policy distribution.

Policy Gradient: Baseline

- In order to reduce the variance of the gradient, a baseline function can be introduced.

$$\nabla_\theta L(\theta) \quad = \quad \int_x d^\pi(x) \sum_a \nabla_\theta \pi(a|x;\theta)(Q(x,a) - b^\pi(x))dx$$

- The baseline does not change the expected value, because $b^\pi(x) \sum_a \nabla_\theta \pi(a|x;\theta) = 0$.

  This follows from $\sum_a \pi(a|x;\theta) \quad = \quad 1$.

- Empirical gradient:

$$\nabla_\theta L(\theta) \quad = \quad \sum_{t=0}^{\infty} \gamma^{t-1} \sum_{x_t^i} \sum_a \nabla_\theta \log \pi(a|x_t^i;\theta)(Q(x_t^i,a) - b^\pi(x_t))$$

Actor-Critic

- In order to compute the gradient, we need the value function Q.

- Possible to compute e.g. with MC.

- Other possibility: Approximate the Value Function with a linear function.

- Methods in which both the actor (the policy) and the critic (the Value Function) are learned are called Actor-Critic methods.

Literature

- [Auer et al. 02]: P.Auer, N.Cesa-Bianchi and P.Fischer: Finite time analysis of the multiarmed bandit problem. Machine Learning 47, 2002.
- [Kearns et al. 02]: M.J. Kearns, Y. Mansour, A.Y. Ng: A sparse sampling algorithm for near-optimal planning in large Markov decision processes. Machine Learning 49: 2002.
- [Kocsis & Szepesvári 06]: L. Kocsis and Cs. Szepesvári: Bandit based Monte-Carlo planning. ECML, 2006.
- [Rust 97]: J. Rust, 1997, Using randomization to break the curse of dimensionality, Econometrica, 65:487—516, 1997.
- [Szepesvári & Munos 05]: Cs. Szepesvári and R. Munos: Finite time bounds for sampling based fitted value iteration, ICML, 2005.
- [Antos et. al. 07]: A. Antos, Cs. Szepesvari and R. Munos: Learning near-optimal policies with Bellman-residual minimization based fitted policy iteration and a single sample path, Machine Learning Journal, 2007