Universität Potsdam
Institut für Informatik
Lehrstuhl Maschinelles Lernen

Reinforcement Learning 2

Uwe Dick

Inhalt

- Erinnerung:
  - Bellman-Gleichungen, Bellman-Operatoren
  - Policy Iteration
- Sehr große oder kontinuierliche Zustandsräume
- Monte-Carlo Sampling, UCT
- Diskretisierung
- Approximate Policy Iteration
  - Bellman Residual Minimization
  - Least Squares Temporal Difference

Literatur

- Reinforcement Learning. An Introduction.
  von Richard S. Sutton und Andrew G. Barto
  http://www.cse.iitm.ac.in/~cs670/book/the-book.html

- Tutorials auf *videolectures.net*
  - z.B. von Csaba Szepesvari oder Satinder Singh

Große und unendliche Zustandsräume

- In realistischen Anwendungen sind Zustandsräume i.A. sehr groß bzw. kontinuierlich.
- Bisherige Annahme: tabellarische Repräsentation der Value Function.

- Mögliche Lösungen:
  - Planen:
    - Monte-Carlo Sampling
    - Diskretisierung und anschließend z.B. Value Iteration
  - Approximation der Value Function durch Funktionsapproximationsmethoden.
  - Direktes Lernen der Policy.

Approximation

- Arten von Approximation
  - Repräsentation, z.B.
    - Value Function $\hat{Q}(s,a;\theta) = \phi^T(s,a)\theta$
    - Policy $\pi(s,a;\theta) = h(\phi^T(s,a)\cdot\theta)$
  - Sampling
    - Online Lernen durch Interaktion
    - Samplen aus generativem Modell der Umgebung
  - Maximierung
    - Finden der besten Aktion in einem Zustand

Funktionsapproximation

- Modellierung
- Value Iteration
  - Least-Squares-Methoden
  - Gradienten-Methoden
- Policy Iteration
  - Least-Squares-Methoden
  - Gradienten-Methoden
- Policy Gradient
- Actor-Critic-Methoden

Funktionsapproximation

- Darstellen der Value Function als parametrisierte Funktion aus dem Funktionsraum $\mathcal{F}$ mit Parametervektor θ.

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

- Vorhersageproblem: Finde Parametervektor θ, so dass V^π, bzw. Q^π am besten approximiert wird.

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

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

Funktionsapproximation

- Generatives Modell
  - Annahme: Es kann jederzeit aus P und R gesamplet werden.
  - Nicht aber $P(s'|s,a)$ abgefragt werden.

- Das Reinforcement Learning Problem:
  - Beispiele <s_t, a_t, R_t, s_{t+1}> aus Interaktion mit der Umgebung.
  - Mögliche Annahme: Interaktion folgt der zu lernenden Policy
    - ★ On-policy-Verteilung von Zuständen $\mu(s)$.

Batch Reinforcement Learning

- Episode gesamplet nach π_b
- Zum Trainingszeitpunkt nur Zugang zu dieser einen Episode.

Approximate Policy Iteration

- Initialisiere Policy π_0
- Iteriere
  - Approximate Policy Evaluation
    Finde $\hat{Q}$ aus $\mathcal{F}$, eine Approximation von Q^{π}, durch
    - Interaktion
    - fester Trainingsmenge
  - Policy Improvement
    Finde π_{t+1} , so dass $\pi_{t+1} \approx \arg\max_a \hat{Q}(s,a),$ für alle s

Approximate Policy Iteration

- Falls Samples von $Q^{\pi}(s,a)$ bekannt, lerne Q^{π} vom Trainingssample mit Hilfe einer überwachten Regressionsmethode.

- Problem: Oft off-policy, d.h. Trainingsbeispiele werden beobachtet während einer Verhaltenspolicy gefolgt wird.
  - Sample Selection Bias (Unterschiedliche Training- und Testverteilungen $p(s,a)$)

Approximate Policy Evaluation

- Idee: Minimiere den quadratischen Abstand zwischen Q und TQ.
  - Mean Squared Bellman Error
- Leider ist TQ nicht notwendigerweise in $\mathcal{F}$.
- Besser: Minimiere $\|Q - \Pi T^{\pi} Q\|_{\mu}^{2}$
  - Mean Squared Projected Bellman Error

Bellman-Residuen-Minimierung

- Temporal Difference Methode.
- Bellman-Gleichung als Fixpunkt-Gleichung.

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

- Linke Seite als Fehler interpretieren: Bellman Residuum. μ stationäre Verteilung von Zuständen.

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

- Empirisch:

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

Bellman-Residuen-Minimierung

- Problem: Schätzer $\hat{L}_{BRM}$ nicht erwartungstreu.

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

- Denn

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

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

- Es folgt:

$$L_{BRM}(Q;\pi)$$

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

Bellman-Residuen-Minimierung

- Aber für $E[\hat{L}(Q;\pi)] = \hat{L}(Q;\pi,n)$, für $n \to \infty$ gilt:

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

- Es gilt aber für Erwartungswerte über Zufallsvariablen X:

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

Bellman-Residuen-Minimierung

- Anwendung auf inneren Erwartungswert:

$$E_{s'}\left[\left(Q(s,a) - \left(R(s,a) + \gamma Q(s', \pi(s'))\right)\right)^2\right]$$

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

- Aber: Schätzung aus Samples ist

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

Bellman-Residuen-Minimierung

- Alternative Herleitung für BRM mit Funktionsapproximation.
  - Optimierungskriterium

$$J(\theta)$$

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

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

  - 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-Residuen-Minimierung

- Daher müssen eigentlich immer zwei unabhängige Nachfolgezustände gezogen werden, um erwartungstreue Schätzung zu bekommen
  $\rightarrow$ Unpraktisch und oft unrealistisch

- Stattdessen wurde vorgeschlagen, einfach nur ein Sample zu nehmen und die Updates damit zu machen.
- Damit konvergiert der Algorithmus allerdings nicht mehr gegen den Fixpunkt von $Q^\pi - T^\pi Q^\pi = 0$

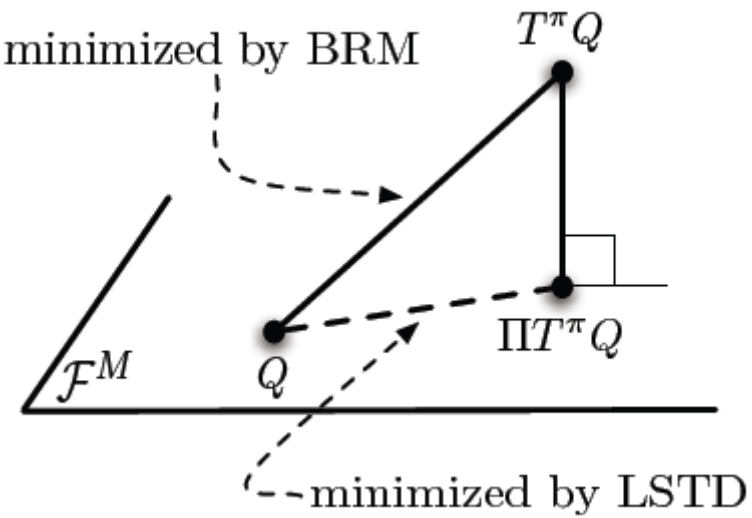
Residual Gradient

- Die neue Lösung nennt man Residual Gradient (RG) Lösung
- I.A. schlechtere Lösung als genaue BRM
- Aber dafür kann ein stochastischer „echter“ Gradient berechnet werden (allerdings nicht für das ursprünglich gewünschte Optimierungskriterium)
  - (Kommt später)

BRM

- Vorschlag: [Antos et. al. 07]
  Erwartungstreue durch Einführung einer Hilfsfunktion $h \in \mathcal{F}$.

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



Least-Squares Temporal Difference

- Q ist aus Funktionsraum $\mathcal{F}$.
- $T^\pi Q$ aber nicht notwendigerweise.
- LSTD minimiert den quadratischen Abstand zwischen Q und der Projektion von $T^\pi Q$ auf $\mathcal{F}$.

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

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

- Unbiased.
- LSTD oft bessere Ergebnisse.

Fitted Policy Evaluation mit Samples

- Q = 0.
- Ziehe N Samples s,a aus $\mu(s)$,p(a). Ziehe M Samples von R und Nachfolgezustand s‘ entsprechend Modell.
- Iteriere:
  - Mit diesen Samples <s, a, R, s‘> wird ein Bellman-Update-Schritt durchgeführt:

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

  - Dann least-squares Fitting:

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

Projected Bellman Equation

- Lineare Approximation der Q-Funktion

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

- Fixpunkt der Bellman Gleichung $Q = \Pi \circ T \circ Q$
- Projektion: Least-Squares
- Definiere

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

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

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

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

Projected Bellman Equation

- Lineare Approximation, also $\mathbf{Q} = \phi\theta$
- Definiere $\mathbf{T}^{\boldsymbol{\pi}}(\mathbf{Q}) = \mathbf{r} + \gamma \mathbf{p}\boldsymbol{\pi}\mathbf{Q}$

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

- Dann $\mathbf{Q} = \Pi\mathbf{T}^{\boldsymbol{\pi}}(\mathbf{Q})$

- Mit $\Gamma = \phi^T\phi, \quad \Lambda = \phi^T\mathbf{p}\boldsymbol{\pi}\phi, \quad z = \phi^T\mathbf{r}$

  kann θ^π als Lösung von $\Gamma\theta^\pi = \gamma\Lambda\theta^\pi + z$ berechnet werden.

LSTD-Q

- Gegeben eine Menge von n Samples (s_i, a_i, s_i')

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

For i=1...n

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

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

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

Löse nach $\hat{\theta}^\pi$

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

LSPE-Q

- Gegeben eine Menge von n Samples (s_i, a_i, s_i')

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

For i=1...n

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

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

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

$$\theta_i \leftarrow \theta_{i-1} + \alpha(\theta_i^h - \theta_{i-1}) \text{ wobei } \frac{1}{i}\Gamma_i\theta_i^h = \gamma\frac{1}{i}\Lambda_i\theta_{i-1} + \frac{1}{i}z_i$$

$$\hat{\theta}^\pi = \theta_n$$

Konvergenz

- Für $n \to \infty$ konvergieren beide Lösungen $\hat{\theta}_i^h$ gegen die Lösung der projected Bellman Gleichung θ_i^h . Konvergenz für beide Algorithmen bewiesen, falls Samples nach h gesamplet wurden.
- Intuitiv dadurch, dass

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

- Aber h darf nicht deterministisch sein. Es kann also z.B. eine e-greedy Policy verwendet werden.

Least-Squares-Methoden

- LSPE-Q ist ein inkrementeller Algorithmus, kann also als online On-Policy Methode verwendet werden und z.B. von einer guten Startlösung $\hat{\theta}_0^h$ profitieren.
- LSTD-Q auf der anderen Seite berechnet die Lösung in einem Schritt, was bessere Stabilität im Vergleich zu LSPE bedeuten kann, falls sich die Policy h während der Evaluierung ändert.

Gradientenbasierte Policy Evaluation

- Eine andere Möglichkeit von (online) Policy Evaluation sind gradientenbasierte Methoden.

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

- $Q^\pi(s_t, a_t)$ ist unbekannt. Deshalb wird es approximiert durch eine stochastische Approximation eines Bellman Updates (1 Sample)

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

TD(0)

- Daraus ergibt sich die Update-Regel für TD(0):

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

- Spezialfall lineare Funktionsapproximation

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

- Auch hier gilt wieder, dass die Policy nicht deterministisch sein darf.

- Aber: Kein Sample eines echten Gradienten!

Gradientenbasierte Policy Evaluation

- „Echter“ Gradient ist Residual Gradient

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

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

- Empirischer Gradient (1 Sample)

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

$$\cdot (\gamma\phi^T(s_{t+1}, a_{t+1}) - \phi(s_t, a_t))$$

Konvergenz

- Konvergenz für TD(0) ist unter bestimmten Annahmen und einem On-Policy-Sample-Prozess bewiesen.
- Wie immer muss zum On-Policy Lernen eine stochastische Policy verwendet werden.
- Um eine (möglichst) deterministische Policy zu lernen, wird oft z.B. das Epsilon einer epsilon-greedy Policy nach und nach immer kleiner gewählt.
- Leider ist für sehr kleine Epsilon die Konvergenz nicht mehr gewährleistet!

Konvergenz

- Residual Gradient hat bessere Konvergenzeigenschaften, vor allen Dingen dadurch, dass ein „echter“ Gradient verwendet wird.

- Für Off-Policy TD(0) Policy Evaluation kann i.A. keine Konvergenz gezeigt werden. (Kann aber trotzdem konvergieren)

Off-Policy Policy Evaluation

- Off-Policy Policy Evaluation bedeutet, dass die Samples (s_t, a_t, s_{t+1}) nach einer Verhaltenspolicy π_b gezogen (generiert) werden, man aber die Value Function von einer anderen Policy π lernen möchte.
- π_b ist i.A. stochastisch, um zu gewährleisten, dass alle Zustands-Aktions-Paare gesehen werden.
- Realisierung abhängig von π
  - Falls π deterministisch, Realisierung durch Subsampling
  - Falls π stochastisch, Realisierung durch Importance Sampling

Off-Policy Policy Evaluation

- Subsampling:
  - Verwerfe alle Beispiele (s_t, a_t, s_{t+1}), die nicht zur Policy π passen

- Importance Sampling:
  - Umgewichten der Beispiele im Update. Z.B. 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) mit Importance Sampling:

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

Divergenz von 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.

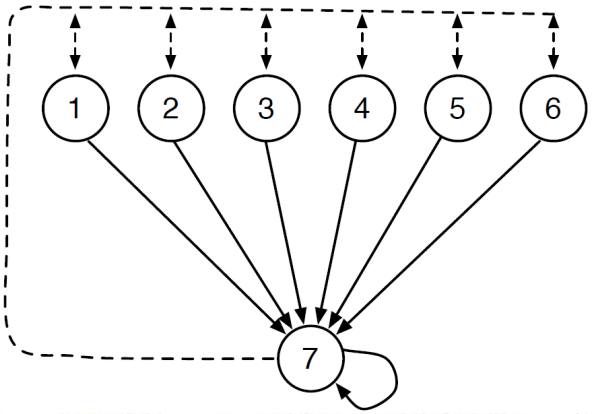


Figure 2.4: The Baird's 7-star MDP. Every transition in this MDP receives zero reward. Each state, has two actions, represented by solid line and dotted line. Solid line action only make transition to state state 7, while dotted line action uniformly make transition to one of states 1-6 with probability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

Off-Policy Policy Evaluation

- Neue Algorithmen, die auch Konvergenz unter Off-Policy-Lernen garantieren.
- Z.B. GTD Algorithmus. (Gradient TD)
- Idee: Neues Optimierungskriterium: Minimiere die L2-Norm des TD(0)-Updates.

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

- Da das Update als der Fehler der derzeigen Lösung θ angesehen werden kann, macht die Minimierung intuitiv Sinn.

GTD-Algorithmus

- Der Gradient kann folgendermaßen berechnet werden:

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

- Bei Approximation basierend auf Samples (für Stochastic Gradient) wird nur einer der beiden Erwartungswerte direkt gesamplet.

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

Policy Improvement

- Falls A endlich und klein, kann die greedy Policy berechnet werden. Ansonsten muss sie approximiert werden.

- Man kann z.B. auch die Policy als eine Funktion von Zustand auf optimale Aktion lernen, etwa indem man das (approximative) max auf Samplemenge bestimmt und damit lernt.

Approximative PI – Theoretische Garantien

- Konvergenz von Approximativen Policy Iteration Algorithmen ist i.A. nicht beweisbar.

- Man kann aber Suboptimalitätsschranken angeben, abhängig von den maximalen Fehlern bei Policy Evaluation und Policy Improvement.

Approximative PI – Theoretische Garantien

- Ist der Fehler bei der Policy Evaluation beschränkt mit

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

- Und der Fehler der Policy Improvement mit

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

- Dann kann der gesamte Fehler mit

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

beschränkt werden

FA für Reinforcement Learning

- Online Updates: Anpassen von θ_t nach jeder Interaktion <s_t, a_t, R_t, s_{t+1}>.

$$\hat{Q}(\cdot;\theta_t) \to Q^{\pi} \qquad t \to \infty$$

$$\hat{Q}(\cdot;\theta_t) \to Q^{*} \qquad t \to \infty$$

- Gradientenabstieg:

$$\begin{aligned} \theta_{t+1} &= \theta_t - \frac{1}{2}\alpha_t \nabla_\theta \left[V^{\pi}(s_t) - \hat{V}(s_t;\theta)\right]^2 \\ &= \theta_t + \alpha_t \left[V^{\pi}(s_t) - \hat{V}(s_t;\theta)\right] \nabla_\theta \hat{V}(s_t;\theta_t) \end{aligned}$$

FA für Reinforcement Learning

- Spezialfall: lineare Methoden.

$$\hat{Q}(\cdot;\theta_t) = \phi^T \theta$$

- Gradientenabstieg:

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

FA für Reinforcement Learning

- Value Function V^{π} unbekannt. Ersetze mit Schätzung.
  - Monte-Carlo: Erwartungstreue Schätzung von V^{π}.
    - ➔ Konvergenz zu lokalem Optimum.
      (Unter Bedingungen für α_t)
  - Temporal Difference (TD(0)): Gebiaste Schätzung.
    - ➔ keine Konvergenz zu lokalem Optimum beweisbar.

Approximative Value Iteration

- Analog zu Approximativer Policy Evaluation: Finde Fixpunkt von Bellman Gleichung für das Kontrollproblem.

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

- Bei parametrischer Approximation mit Parametervektor θ

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

Approximative Value Iteration

- Der Bellman-Operator ist definiert als

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

- Und die Projektion wird realisiert durch Least-Squares-Approximation

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

Fitted Value Iteration mit Samples

- [Szepesvári & Munos 05]
- $V = 0$.
- Ziehe N Zustände s aus $\mu(s)$.
- Für jedes s und $a \in A$, Ziehe M Nachfolgezustände s' aus $P(\cdot|s,a)$ und Rewards $R(s,a)$.
- Iteriere:
  - Mit diesen Samples <s, a, R, s'> wird ein Bellman-Update-Schritt durchgeführt:

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

  - Dann least-squares Fitting:

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

Fehlerabschätzung

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

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

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

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

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

Approximatives Q-Learning

- Lineare Parametrisierung der Q-Funktion
- Iterationsschritt:

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

- Analog zu TD(0) wird das unbekannte $Q^*(s_t, a_t)$ approximiert mit einem Bellman-Update

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

Approximatives Q-Learning

- Wieder stochastische Gradient-Methode, es wird also der Erwartungswert durch stochastische Samples approximiert

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

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

- Unter bestimmen Voraussetzungen kann man die Konvergenz von Q-Learning beweisen, allerdings müssen dafür Einschränkungen für Art der Policy oder der Featurevektoren gemacht werden.

Approximative Value Iteration

- Konvergenz-Beweise von Value Iteration basieren i.A. auf der Beweisbarkeit der Nichtexpansionseigenschaften der Projektion und der Funktionsapproximation, damit

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

- D.h., es muss gelten:

und

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

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

Approximative Value Iteration

- $\|Q(\theta) - Q(\theta')\|_\infty \leq \|\theta - \theta'\|_\infty$ kann garantiert werden, indem die Featurevektoren ϕ normalisiert werden. $\sum_{i=1}^{N} \phi_i = 1$

- Projektion schwieriger zu zeigen.

Approximatives SARSA

- On-Policy-Methode für das Kontrollproblem

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

- Konvergenz kann unter vernünftigen Annahmen bewiesen werden. Allerdings gilt die Konvergenz grob gesagt nur, falls nicht zu klare Entscheidungen getroffen werden (z.B. nur bei großem Epsilon in Epsilon-Greedy)

TD(λ)

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

- Updateregel: $V(s_t) \leftarrow (1-\alpha_t)V(s_t) + \alpha_t \Delta V(s_t)$
- TD(λ) Update: $\Delta(\lambda) = \sum_{k=1}^{\infty} (1-\lambda)\lambda^{k-1} \Delta_k V(s_0)$
- $0 \leq \lambda \leq 1$ interpoliert zwischen 1-step und MC.

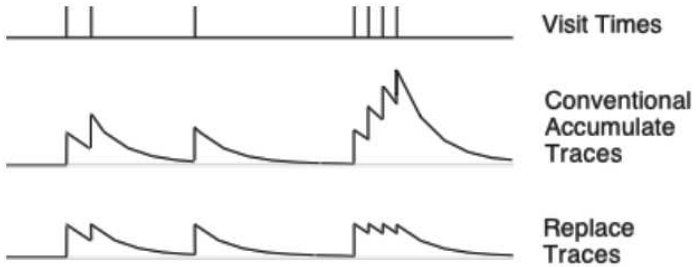
Eligibility Traces

- Algorithmische Sicht auf TD(λ)
- Einführung eines zusätzlichen Speichers $e(s)$ für jeden Zustand $s \in S$.
- Nach Beobachtung <s_t, a_t, R_t, s_{t+1}>, berechne

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

- Update für alle Zustände

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



FA für Reinforcement Learning

- TD(λ)
  - Eligibility traces:

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

  - Lineare Methode: Konvergenzgarantie nur für on-policy.
  - Fehlerabschätzung:

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

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

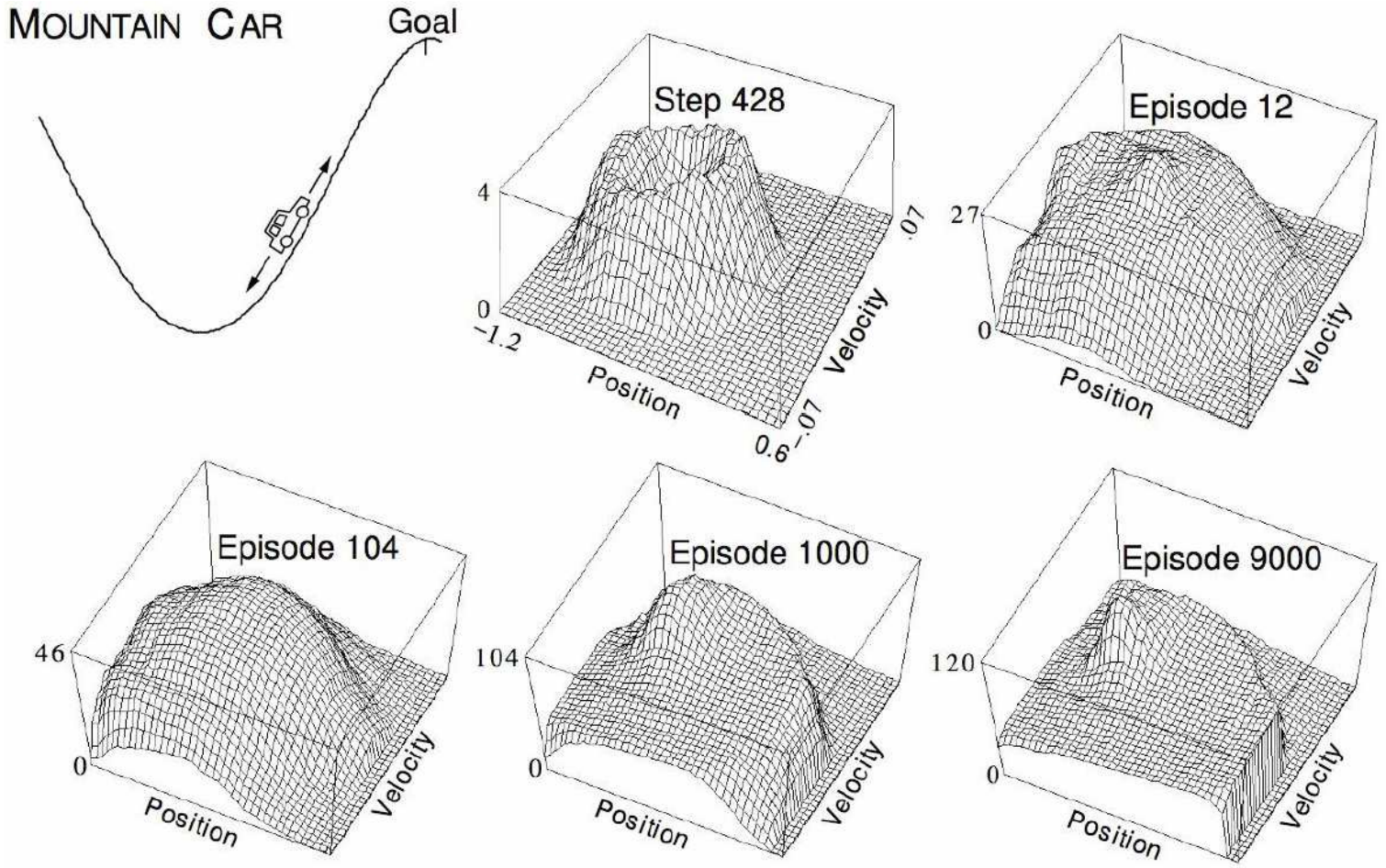
SARSA(λ)

- Kontrollproblem: SARSA(λ) (On-Policy)

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

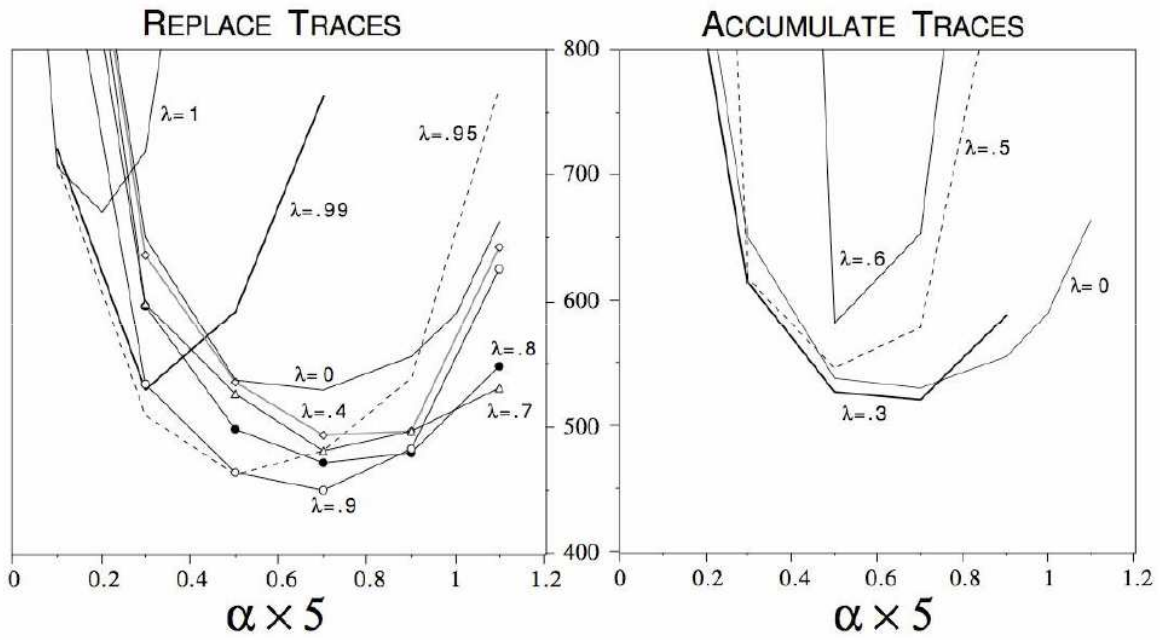
- Off-policy kann divergieren.

SARSA(λ)



SARSA(λ)

Steps per episode
averaged over
first 20 trials
and 30 runs



Policy Gradient

- Lernen einer stochastischen Policy.
- Die Policy wird explizit repräsentiert, z.B. als Gibbs Verteilung

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

- Lerne , so dass $E[\sum_{t=1}^{\infty} \gamma^t r_t | x_0, \pi]$ maximiert wird.

- Idee: (stochastische) Gradientenmethode

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

Policy Gradient

- Frage: Wie wird der Gradient berechnet?
- Antwort: Policy Gradient Theorem

- Definiere zeitlich discounted Zustandswahrscheinlichkeit

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

- Dann

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

Policy Gradient Theorem

- Es gilt: Policy Gradient Theorem:
  - Sei L definiert wie oben. Dann gilt für den Gradienten

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

- D.h. der Gradient berechnet sich auf Grund der gleichen erwarteten Zustandsverteilung und der Value Function

Policy Gradient: Log-Trick

- Der Gradient kann umgeschrieben werden mit Hilfe des „log-Tricks“

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

- Daraus folgt für den empirischen Gradienten

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

d.h. der Gradient wird ohne Bias approximiert, wenn die Samples aus der On-Policy Verteilung stammen.

Policy Gradient: Baseline

- Um die Varianz des Gradienten klein zu halten, wird oft eine Baseline in den Gradienten eingefügt

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

- Die Baseline verändert nicht den Erwartungswert, denn $b^\pi(x) \sum_a \nabla_\theta \pi(a|x;\theta) = 0$

  da $\sum_a \pi(a|x;\theta) \quad = \quad 1$

- Empirischer 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

- Um den Gradienten zu berechnen, brauchen wir die Value Function Q
- Z.B. möglich über MC
- Andere Möglichkeit: Approximieren der Value Function mit Hilfe z.B. einer linearen Funktion
- Werden der Actor (die Policy) und der Critic (die Value Function) beide gelernt, spricht man von Actor-Critic-Methoden

Literatur

- [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