Sample Efficient RL

Valdinei Freire

(EACH - USP)

Planejamento e Aprendizado por Reforço

- Value Iteration e Policy Iteration (operador de Bellman em todos estados)

- LAO* e LRTDP (operador de Bellman em estados alcançáveis)

- Monte Carlo Tree Search (amostragem das transições)

- Q-Learning e Sarsa(λ) Tabular (Programação Dinâmica Estocástica)

- Q-Learning e Sarsa(λ) Aproximado (Estados, Ações contínuas, otimização por iteração)

- REINFORCE e Actor-Critic (Estados, Ações contínuas, amostragem por iteração)

- Problema: experiências são utilizadas uma única vez

Abordagens

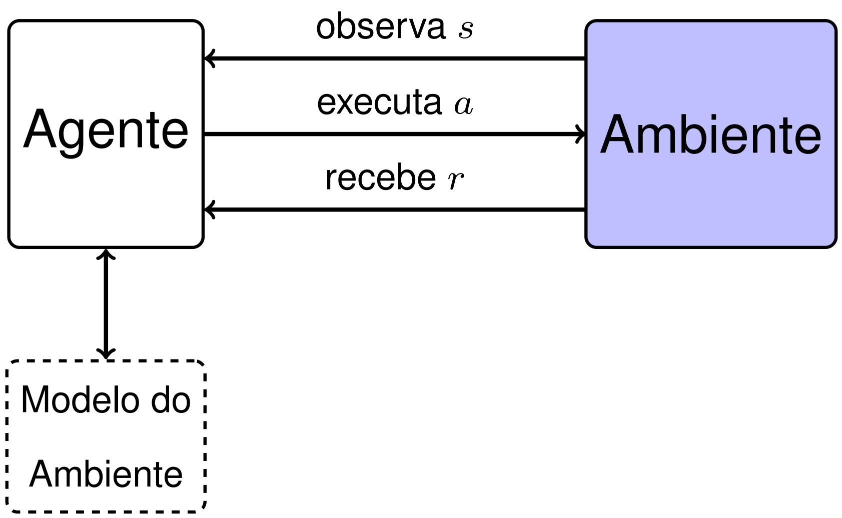
Model-based: o conjunto de experiências em $\mathcal{B}$ definem um MDP aproximado $\widehat{M}$

Reuso de Experiências: considere um algoritmo incremental e reapresente as experiências

Batch: transforme o problema de aprendizado em um problema de regressão segundo alguma função objetivo.

Arquitetura Dyna

Richard Sutton. Integrated Architectures for Learning, Planning, and Reacting Based on Approximating Dynamic Programming. 1990.



Arquitetura Dyna

Repete:

1. Observa estado do mundo e escolha uma ação reativamente
2. Observe recompensa e estado resultante
3. Aplique aprendizado por Reforço a esta experiência
4. Atualize modelo do Ambiente com esta experiência
5. Repita K vezes:
   (a) Escolha um estado e ação hipotético
   (b) Simule recompensa e estado resultante com o modelo do Ambiente
   (c) Aplique aprendizado por Reforço a esta experiência hipotética

Modelo do Ambiente Discreto

- Contagem
  - Armazena uma contagem para cada experiência $N(s, a, r, s')$
  - $\widehat{T}(s, a, s') = \dfrac{\sum_r N(s, a, r, s')}{\sum_{r,s'} (s, a, r, s')}$
  - $\widehat{R}(s, a) = \dfrac{\sum_{r,s'} r \times N(s, a, r, s')}{\sum_{r,s'} (s, a, r, s')}$

Modelo do Ambiente Contínuo

Deisenroth e Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. 2011.

- $\mathbf{x}_t$ estado do ambiente, $\mathbf{u}_t$ controle

- Dinâmica do Sistema: $\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) + \epsilon$

- Processo Gaussianos:

$$\mathrm{Pr}(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_{t-1}) = \mathcal{N}(\mathbf{x}_t | \mu_t, \Sigma_t)$$

$$\mu_t = \mathbf{x}_t + \mathrm{E}_f[\Delta_t]$$

$$\Sigma_t = \mathrm{Var}_f[\Delta_t]$$

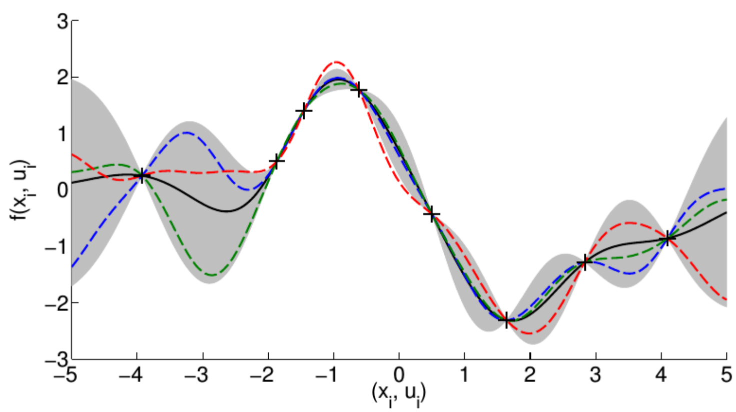
Modelo do Ambiente Contínuo

- Observação, $\mathbf{x}_t$, $\mathbf{u}_t$, e

$$\Delta_t = \mathbf{x}_t - \mathbf{x}_{t-1}$$

- Covariância: Squared Exponential Kernel

$$k(\mathbf{x}, \mathbf{x}') = \text{exp}\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top \Lambda^{1} (\mathbf{x} - \mathbf{x}')\right)$$



Reuso de Experiências

- Armazena as experiências e reapresenta quando na simulação

- Quantidade de Experiências versus Quantidade de Parâmetros

- Informação *a priori* versus *tabula rasa*

Mnih et. al. Human-level control through deep reinforcement learning, Nature, 2015

- Armazena apenas experiências mais recentes

- Considera duas funções Q: aprendizado e referência

Batch Reinforcement Learning

Considere um MDP e uma política estacionária μ.

Considere um conjunto de experiências $\mathcal{B}$ com quadruplas $\langle s_i, a_i, r_i, s_i' \rangle$ obtidas ao executar μ.

Prediction: dada uma política π, encontrar a função valor V^π com base nas experiências em $\mathcal{B}$

Control: encontrar a política ótima π^* com base nas experiências em $\mathcal{B}$

Algoritmos Off-Policy

On-policy: função valor aprendida depende da função valor executada - SARSA(λ)

Off-policy: função valor aprendida independe da função valor executada - Q-learning

Problema: convergência com aproximação

Distribuição Limite

Considere que μ gere uma cadeia de Markov Ergódica e uni-chain, então existe uma distribuição limite:

$$d_s = \lim_{t\to\infty} P(s_t = s)$$

Considere a matriz $D \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|}$ cuja diagonal são d_s.

Defina então a norma quadrática para $v \in \mathbb{R}^{|\mathcal{S}|}$ com pesos d_s, isto é:

$$\| v \|_D^2 = v^\top D v = \sum_{s\in\mathcal{S}} d_s(v(s))^2$$

Função Objetivo

Considere uma função para aproximar $V(s)$ parametrizada em $\theta \in \Omega$, isto é, $\widehat{V}_\theta(s)$.

MSVE: Mean Squared Value Error

$$\text{MSVE}(\theta) = \parallel V - \widehat{V}_\theta \parallel_D^2$$

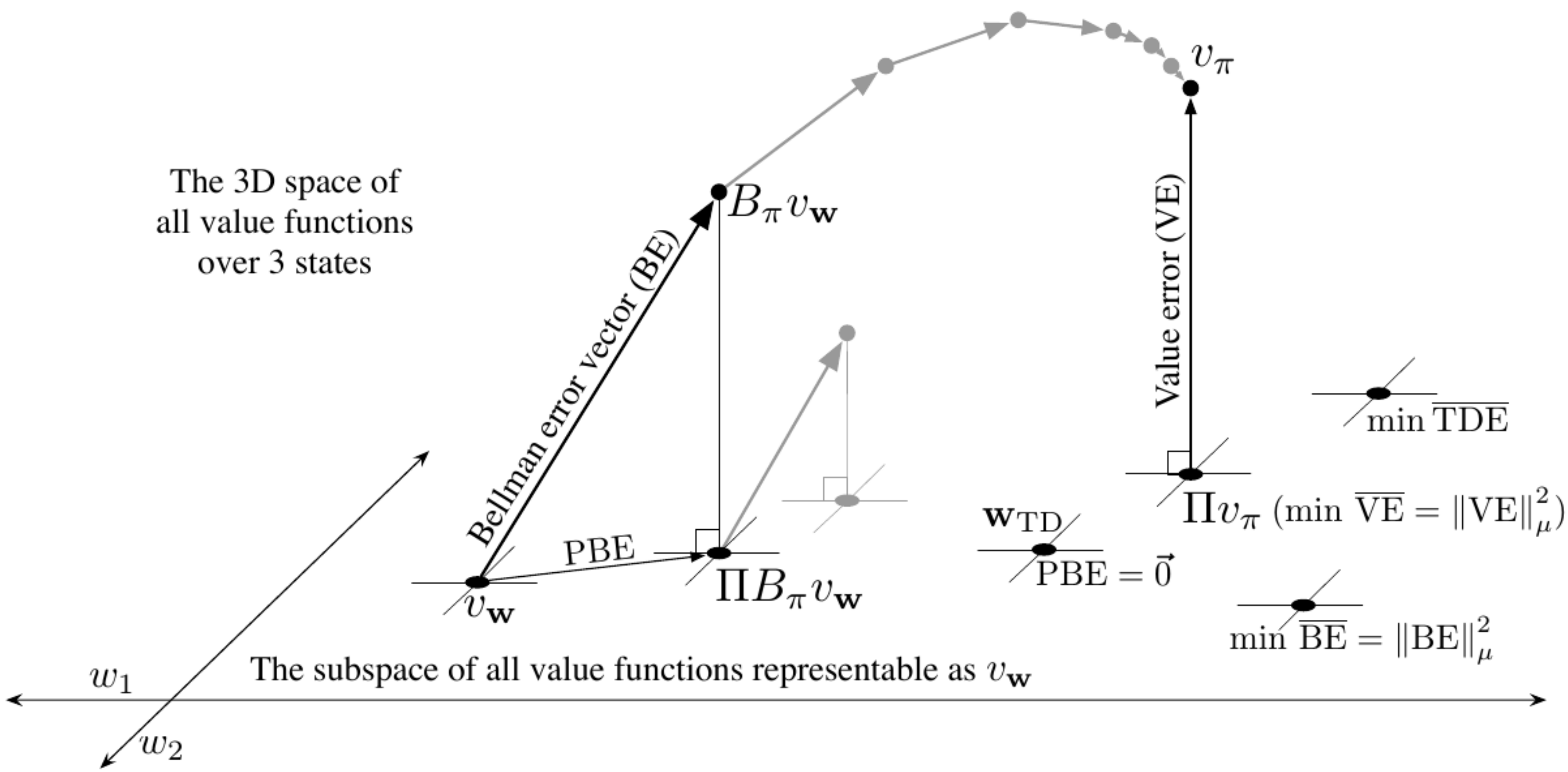
MSBE: Mean Squared Bellman Error

$$\text{MSBE}(\theta) = \parallel \widehat{V}_\theta - \mathcal{T}\widehat{V}_\theta \parallel_D^2$$

MSPBE: Mean Squared projected Bellman Error

$$\text{MSPBE}(\theta) = \parallel \widehat{V}_\theta - \Lambda\mathcal{T}\widehat{V}_\theta \parallel_D^2$$

Função Objetivo



Aproximação Linear

Considere aproximação por função linear, isto é,

$$\widehat{Q}_{\mathbf{w}}(s,a) = \sum_{i=1}^{k} \phi_i(s,a) w_i = \phi(s,a)^\top \mathbf{w}$$

Considere uma função v, a projeção Λv na função linear é dada por:

$$\Lambda v(s,a) = \phi(s,a)^\top \mathbf{w}_v \quad \text{e} \quad \mathbf{w}_v = \arg\min_{\mathbf{w}} \| v - \widehat{Q}_{\mathbf{w}} \|_D^2$$

Considere a representação matricial $\Phi \in \mathbb{R}^k \times \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ para as funções bases $\phi(s,a)$. Pode-se demonstrar que:

$$\Lambda = \Phi^\top (\Phi D \Phi^\top)^{-1} \Phi D$$

Base de Dados

Se estamos interessado em aproximar função valor estado-ação, temos que considerar distribuições $d_{s,a}$.

Considere que $d_{s,a}$ é dado pela distribuição de (s, a) na base de dados $\mathcal{B}$. No limite, cada par estado-ação podem aparecer uma única vez. Considere que cada par (s, a) é único e monte a representação matricial da base de dados $\Phi_{\mathcal{B}}$.

Então, o operador de projeção sob a base de dados é dado por:

$$\Lambda_{\mathcal{B}} = \Phi_{\mathcal{B}}^{\top}(\Phi_{\mathcal{B}}\Phi_{\mathcal{B}}^{\top})^{-1}\Phi_{\mathcal{B}}$$

Como é subentendido a base de dados $\mathcal{B}$, utilizaremos simplesmente:

$$\Lambda = \Phi^{\top}(\Phi\Phi^{\top})^{-1}\Phi$$

Fitt Q

Martin Riedmiller. Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method. 2005.

Considere o seguinte Mean Squared Bellman Error:

$$\text{MSBE}(\theta; \theta') = \| \widehat{V}_\theta - \mathcal{T}\widehat{V}_{\theta'} \|_D^2$$

1. escolha $\mathbf{w}_0$ arbitrário
2. faça para $t \geqslant 1$:
   (a) $\mathbf{w}_t \leftarrow \arg\min_{\mathbf{w} \in \mathbb{R}^k} \text{MSBE}(\mathbf{w}; \mathbf{w}_{t-1})$
   (b) $res = \| \mathbf{w}_t - \mathbf{w}_{t-1} \|$
3. enquanto $res > \epsilon$

Fitt Q

Para cada tupla $\langle s, a, r, s' \rangle$ da base de dados $\mathcal{B}$, adicione ações $a' = \arg\max_{a \in \mathcal{A}} \widehat{Q}_{\mathbf{w}_{t-1}}(s', a)$. Então, é bem definida a matriz $\Phi'_{\mathbf{w}_{t-1}}$.

Se $\widehat{Q}_{\mathbf{w}}$ é linear temos que:

$$\text{MSBE}(\mathbf{w}) = \left\| \Phi^\top \mathbf{w} - \mathbf{r} - \gamma (\Phi'_{\mathbf{w}_{t-1}})^\top \mathbf{w}_{t-1} \right\|^2$$

e

$$\mathbf{w} = (\Phi\Phi^\top)^{-1} \Phi \left(\mathbf{r} + \gamma (\Phi'_{\mathbf{w}_{t-1}})^\top \mathbf{w}_{t-1} \right)$$

Fitted Q

Geoffrey Gordon. Stable Function Approximation in Dynamic Programming. 1995.

Se as funções bases ϕ_i pertencer à classe de *Averagers*, o algoritmo Fitted Q converge.

$$\widehat{V}(s_i) = \beta_i k_i + \sum_j \beta_{ij} V(s_j),$$

tal que $\beta_i + \sum_j \beta_{ij} = 1$, $\beta_i \geqslant 0$, $\beta_{ij} =\geqslant 0$

Uma classe de *Averagers* são estados agregados. Nesse caso, ϕ_i são funções binárias tal que para todo $s \in \mathcal{S}$:

$$\sum_i \phi_i(s, a) = 1$$

Least Square Policy Iteration

Lagoudakis and Parr. Least-Squares Policy Iteration. 2003.

Considera política inicial π_0

Avalia:

- MSBE

$$\mathbf{w}_\pi = \left(\left(\mathbf{\Phi} - \gamma \mathbf{\Phi}'_\pi\right)\left(\mathbf{\Phi}^\top - \gamma(\mathbf{\Phi}'_\pi)^\top\right)\right)^{-1}\left(\mathbf{\Phi} - \gamma(\mathbf{\Phi}'_\pi)\right)\mathbf{r}$$

- MSPBE

$$\mathbf{w}_\pi = \left(\mathbf{\Phi}\left(\mathbf{\Phi}^\top - \gamma(\mathbf{\Phi}'_\pi)^\top\right)\right)^{-1}\mathbf{\Phi}\mathbf{r}$$

Next

Sutton et al. Fast Gradient-Descent Methods for Temporal-Difference Learning with Linear Function Approximation. 2009.

Maei et al. Convergent Temporal-Difference Learning with Arbitrary Smooth Function Approximation. 2009.

Maei et al. Toward Off-Policy Learning Control with Function Approximation. 2010.

Degris at al. Off-Policy Actor-Critic. 2012.

Silver et a. Deterministic Policy Gradient Algorithms. 2014.

Munos et al. Safe and efficient off-policy reinforcement learning. 2016.