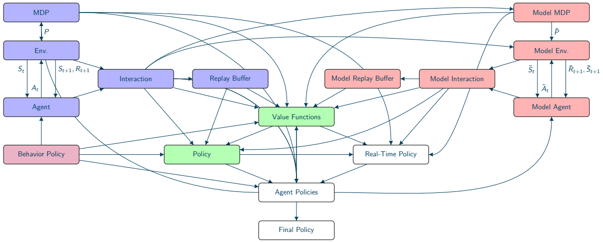
Reinforcement Learning
Reinforcement Learning: Approximation of the Value Functions

Erwan Le Pennec
Erwan.Le-Pennec@polytechnique.edu

M2DS - Reinforcement Learning – Fall 2024

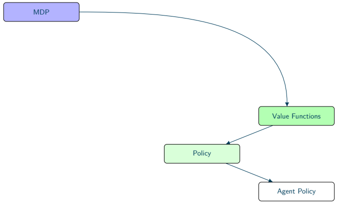
RL: What Are We Going To See?



Outline

- Operations Research and MDP.
- Reinforcement learning and interactions.
- More tabular reinforcement learning.
- Reinforcement and approximation of value functions.
- Actor/Critic: a Policy Point of View

Operations Research and MDP

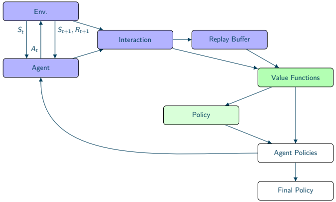


How to find the best policy knowing the MDP?

- Is there an optimal policy?
- How to estimate it numerically?

- Finite states/actions space assumption (tabular setting).
- Focus on interative methods using value functions (dynamic programming).
- Policy deduced by a statewise optimization of the value function over the actions.
- Focus on the discounted setting.

Reinforcement Learning and Interactions

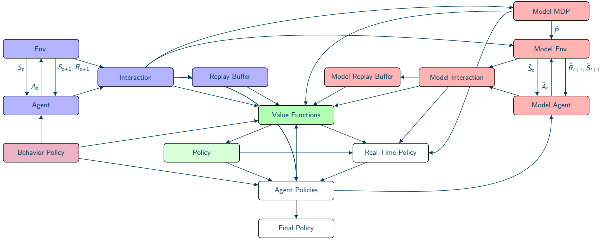


How to find the best policy not knowing the MDP?

- How to interact with the environment to learn a good policy?
- Can we use a Monte Carlo strategy outside the episodic setting?
- How to update value functions after each interaction?

- Focus on stochastic methods using tabular value functions (Q learning, SARSA...)
- Policy deduced by a statewise optimization of the value function over the actions.

More Tabular Reinforcement Learning

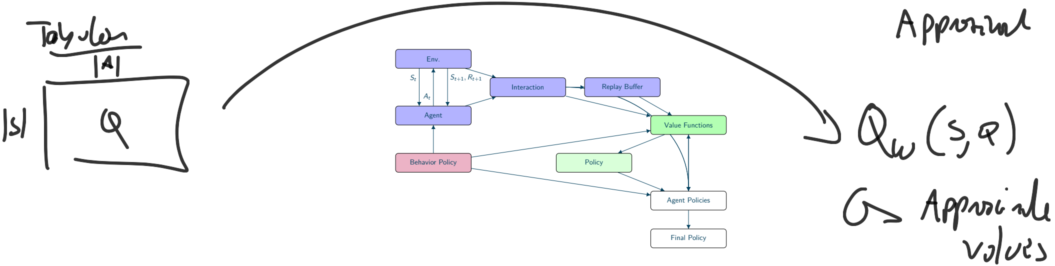


Can We Do Better?

- Is there a gain to wait more than one step before updating?
- Can we interact with a different policy than the one we are estimating?
- Can we use an estimated model to plan?
- Can we plan in real-time instead of having to do it beforehand?

- Finite states/actions space setting (tabular setting).

Reinforcement and Approximation of Value Functions



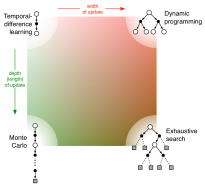
How to Deal with a Large/Infinite states/action space?

- How to approximate value functions?
- How to estimate good approximation of value functions?

- Finite action space setting.
- Stochastic algorithm (Deep Q Learning. . .).
- Policy deduced by a statewise optimization of the value function over the actions.

Outline

Approximation?



Tabular Setting

- Require to store the state(-action) values (a table).
- Requirement in both OR and RL.

Approximation!

- Use instead approximated value functions.
- What is a good approximation?
- How to use them?

- Focus on value-functions. . .

Outline

Approximated Value Functions

$$V(s) \Longrightarrow V_{\boldsymbol{w}}(s)$$
$$Q(s,a) \Longrightarrow Q_{\boldsymbol{w}}(s,a)$$

Parametric Model

- Reduce dimensionality by storing $\boldsymbol{w}$ instead of all the values.
- Linear: $V_{\boldsymbol{w}}(s) = \langle \Phi(s), \boldsymbol{w} \rangle$ and $Q_{\boldsymbol{w}}(s,a) = \langle \Phi(s,a), \boldsymbol{w} \rangle$
  - $\Phi(s)$ and $\Phi(s,a)$ are features associated to the states(-actions).
  - Tabular setting corresponds to $(\Phi)_{s'(,a')}(s(,a)) = \mathbf{1}_{s'=s(,a'=a)}$.
  - Often used in theoretical analysis.
- Deep Learning: $V_{\boldsymbol{w}}(s) = \mathrm{NN}_{\boldsymbol{w}}(\Phi(s))$ and $Q_{\boldsymbol{w}}(s,a) = \mathrm{NN}_{\boldsymbol{w}}(\Phi(s,a))$
  - NN is any (deep) learning network.
  - Often used in practice.

- Other parametrization (or even non parametric coding) could be used (at least in theory...).

Approximated Value Functions Usage

$$v_\pi(s) \simeq V_{\mathbf{w}_\pi}(s)$$
$$q_\pi(s,a) \simeq Q_{\mathbf{w}_\pi}(s,a)$$
$$\operatorname*{argmax}_a q_\pi(s,a) \simeq \operatorname*{argmax}_a Q_{\mathbf{w}_\pi}(s,a)$$

$$v_\star(s) \simeq V_{\mathbf{w}_\star}(s)$$
$$q_\star(s,a) \simeq Q_{\mathbf{w}_\star}(s,a)$$
$$\operatorname*{argmax}_a q_\star(s,a) \simeq \operatorname*{argmax}_a Q_{\mathbf{w}_\star}(s,a)$$

Approximated Value Functions Usage

- *Drop-in* replacements for all the value functions?
- Prediction and Planning?
- Quality and Stability?

Approximation Quality

$$v_\pi(s) \simeq V_{\mathbf{w}_\pi}(s) \qquad v_\star(s) \simeq V_{\mathbf{w}_\star}(s)$$

$$q_\pi(s,a) \simeq Q_{\mathbf{w}_\pi}(s,a) \qquad q_\star(s,a) \simeq Q_{\mathbf{w}_\star}(s,a)$$

$$\operatorname*{argmax}_a q_\pi(s,a) \simeq \operatorname*{argmax}_a Q_{\mathbf{w}_\pi}(s,a) \qquad \operatorname*{argmax}_a q_\star(s,a) \simeq \operatorname*{argmax}_a Q_{\mathbf{w}_\star}(s,a)$$

Approximation Quality Norm

- Ideal loss:

$$\|v - V_{\mathbf{w}}\|_\infty \quad \text{or} \quad \|q - Q_{\mathbf{w}}\|_\infty$$

  as this is the error used in all the previous analysis.

- Practical loss:

$$\|v - V_{\mathbf{w}}\|_{\mu,p}^p = \sum_s \mu(s)|v(s) - V_{\mathbf{w}}(s)|^p$$

$$\text{or} \quad \|q - Q_{\mathbf{w}}\|_{\mu,p}^p = \sum_{s,a} \mu(s,a)|q(s,a) - Q_{\mathbf{w}}(s,a)|^p$$

  often with $p = 2$ and μ related to the behavior policy.

Approximation Target(s)

$$\mathcal{T}Q(s_t, a_t) = \mathbb{E}\left[R_{t+1} + \gamma Q(s_{t+1}, a_{t+1})\right]$$

$$q(s,a) = \mathcal{T}q(s,a) \sim Q_{\mathbf{w}}(s,a) \longrightarrow \begin{cases} \|q - Q_{\mathbf{w}}\|_{\mu,p} \text{ small} \\ \|\mathcal{T}Q_{\mathbf{w}} - Q_{\mathbf{w}}\|_{\mu,p} \text{ small} \end{cases}$$

Approximation Targets(s)

- Direct measurement.
- Bellman residual error.

Extended Measurement

- Projection (with linear parametrization): $\|P_\Phi\left(\mathcal{T}Q_{\mathbf{w}} - Q_{\mathbf{w}}\right)\|_{\mu,p}$ small
- Probes Z:

$$\mathbb{E}_Z[|\langle \mathcal{T}Q_{\mathbf{w}} - Q_{\mathbf{w}}, Z\rangle|^p]$$

- Lots of freedom but hard to link with optimality of derived policy!

Outline

Prediction, Approximation and Gradient Descent

$$\min_{\boldsymbol{w}} \sum_{s,a} \mu_b(s,a) \left| q_\pi(s,a) - Q_{\boldsymbol{w}}(s,a) \right|^2$$

Prediction, Approximation and Gradient Descent

- Prediction objective:

$$\overline{\mathrm{VE}}(\boldsymbol{w}) = \sum_{q} \mu_b(s,a) \left| q_\pi(s,a) - Q_{\boldsymbol{w}}(s,a) \right|^2$$

- Gradient:

$$\nabla \overline{\mathrm{VE}}(\boldsymbol{w}) = -2 \sum_{s,a} \mu_b(s,a) \left(q_\pi(s,a) - Q_{\boldsymbol{w}}(s,a) \right) \nabla Q_{\boldsymbol{w}}(s,a)$$

- Stochastic gradient:

$$\widehat{\nabla} \overline{\mathrm{VE}}(\boldsymbol{w}) = -2 \left(q_\pi(S_t, A_t) - Q_{\boldsymbol{w}}(S_t, A_t) \right) \nabla Q_{\boldsymbol{w}}(S_t, A_t)$$

- Not a practical algorithm as q_π is unknown.

Prediction, Approximation and MC

$$\mathbf{w}_{t+1} = \mathbf{w}_t + 2\alpha_t \left(G_t - Q_{\mathbf{w}_t}(S_t, A_t)\right) \nabla Q_{\mathbf{w}_t}(S_t, A_t)$$

Monte Carlo Approach

- Replace $q_\pi(S_t, A_t)$ by its Monte Carlo estimate G_t.
- Still a Stochastic Gradient of the original problem with limit (if it exists) satisfying

$$\begin{aligned}\mathbb{E}_\pi[(G_t - Q_{\mathbf{w}_\infty}(S_t, A_t)) \nabla Q_{\mathbf{w}_\infty}(S_t, A_t)] \\ = \mathbb{E}[(q_\pi(S_t, A_t) - Q_{\mathbf{w}_\infty}(S_t, A_t)) \nabla Q_{\mathbf{w}_\infty}(S_t, A_t)] = 0\end{aligned}$$

- Convergence ensured for the linear parametrization as it is a convex problem.

- Correspond exactly to the tabular MC prediction algorithm for the tabular parametrization.
- For the linear parametrization:

  Limiting equation: $\mathbb{E}_\pi[q_\pi(S_t, A_t)\Phi(S_t, A_t)] = \mathbb{E}_\pi\left[\Phi(S_t, A_t)\Phi(S_t, A_t)^\top\right] \mathbf{w}_\infty$

Prediction, Approximation and TD

$$\mathbf{w}_{t+1} = \mathbf{w}_t + 2\alpha_t \left(R_{t+1} + \gamma Q_{\mathbf{w}_t}(S_{t+1}, A_{t+1}) - Q_{\mathbf{w}_t}(S_t, A_t)\right) \nabla Q_{\mathbf{w}_t}(S_t, A_t)$$

Temporal Differencies Approach

- Replace $q_\pi(S_t, A_t)$ by $R_{t+1} + \gamma Q_{\mathbf{w}_t}(S_{t+1}, A_{t+1})$.
- Not a Stochastic Gradient of the original problem but a Stochastic Approximation algorithm with limit (if it exists) satisfying

$$\begin{aligned}\mathbb{E}_\pi[(R_t + \gamma Q_{\mathbf{w}_\infty}(S_{t+1}, A_{t+1}) - Q_{\mathbf{w}_\infty}(S_t, A_t)) \nabla Q_{\mathbf{w}_\infty}(S_t, A_t)] \\ = \mathbb{E}_\pi[((\mathcal{T}^\pi Q_{\mathbf{w}_\infty} - Q_{\mathbf{w}_\infty})(S_t, A_t)) \nabla Q_{\mathbf{w}_\infty}(S_t, A_t)] = 0\end{aligned}$$

- No simple argument to justify the convergence...

- In general, no straightforward relation with Bellman operator.
- Correspond exactly to the tabular TD prediction algorithm for the tabular parametrization.

$$\mathbf{w}_{t+1} = \mathbf{w}_t + 2\alpha_t \left(\tilde{G}_t - Q_{\mathbf{w}_t}(S_t, A_t)\right) \nabla Q_{\mathbf{w}_t}(S_t, A_t)$$

Temporal Differencies Approach

- Replace $q_\pi(S_t, A_t)$ by any advanced return $\tilde{G}_t$.
- Not a Stochastic Gradient of the original problem but a Stochastic Approximation algorithm with limit (if it exists) satisfying

$$\begin{aligned}
&\mathbb{E}_\pi\left[\left(\tilde{G}_t - Q_{\mathbf{w}_t}(S_t, A_t)\right) \nabla Q_{\mathbf{w}_\infty}(S_t, A_t)\right] \\
&= \mathbb{E}_\pi\left[\left((\widetilde{\mathcal{T}}^\pi Q_{\mathbf{w}_\infty} - Q_{\mathbf{w}_\infty})(S_t, A_t)\right) \nabla Q_{\mathbf{w}_\infty}(S_t, A_t)\right] = 0
\end{aligned}$$

- No simple argument to justify the convergence. . .

- In general, no straightforward relation with Bellman operator.
- Correspond exactly to the tabular TD prediction algorithm for the tabular parametrization.

Prediction, Approximation and Eligibility Trace

$$
\begin{aligned}
z_t &= \gamma \lambda z_{t-1} + \nabla Q_{\mathbf{w}_t}(S_t, A_t) \\
\delta_t &= R_{t+1} + \gamma Q_{\mathbf{w}_t}(S_{t+1}, A_{t+1}) - Q_{\mathbf{w}_t}(S_t, A_t) \\
\mathbf{w}_{t+1} &= \mathbf{w}_t + \alpha_t \delta_t z_t
\end{aligned}
$$

Eligibility Trace

- Rewrite the TD(λ) updates using the backward point of view.
- No strict equivalence due to time evolution of the parameterization.
- Stochastic Approximation with limit (if it exists) satisfying

$$
\begin{aligned}
&\mathbb{E}_\pi[(R_{t+1} + \gamma Q_{\mathbf{w}_\infty}(S_{t+1}, A_{t+1}) - Q_{\mathbf{w}_\infty}(S_t, A_t))\, z_t] \\
&= \mathbb{E}_\pi[(\mathcal{T}^\pi Q_{\mathbf{w}_\infty} - Q_{\mathbf{w}_\infty})\,(S_t, A_t) z_t] = 0
\end{aligned}
$$

- No simple argument to justify the convergence.

Outline

Linear Parametrization

$$Q_{\boldsymbol{w}}(S_t, A_t) = \Phi(S_t, A_t)^\top \boldsymbol{w} \quad \text{and} \quad \nabla Q_{\boldsymbol{w}}(S_t, A_t) = \Phi(S_t, A_t)$$

Linear Parametrization

- Extension of the tabular setting.
- Derivative is independent of $\boldsymbol{w}$.
- Analysis of Stochastic Approximation often possible!
- More than a toy model as an algorithm not converging in the linear case will almost certainly not converge in a more general setting.

Linear Parametrization and MC

$\frac{dw}{dt} = -C(w - \tilde{w})$

$w = \bar{w} + e^{-Ct}$

$$\text{Iteration:} \; \boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha_t (G_t - \Phi(S_t, A_t)^\top \boldsymbol{w}_t) \Phi(S_t, A_t)$$

$$\text{Limiting equation:} \; \mathbb{E}_\pi [q_\pi(S_t, A_t) \Phi(S_t, A_t)] = \mathbb{E}_\pi \left[\Phi(S_t, A_t) \Phi(S_t, A_t)^\top \right] \boldsymbol{w}_\infty$$

$$\text{ODE:} \; \frac{d\boldsymbol{w}}{dt} = -\mathbb{E}_\pi \left[\Phi(S_t, A_t) \Phi(S_t, A_t)^\top \right] (\boldsymbol{w} - \boldsymbol{w}_\infty)$$

Linear Parametrization and MC

- Limiting equation is a linear equation.
- Under asymptotic stationarity assumption, convergence of ODE as $\mathbb{E}_\pi \left[\Phi(S_t, A_t) \Phi(S_t, A_t)^\top \right]$ is a Gram Matrix with positive eigenvalues (provided Φ is not redundant and under an ergodicity assumption).
- Need to explore all state-action pairs!

Linear Parametrization and TD

Iteration: $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha_t (R_{t+1} + \gamma \Phi(S_{t+1}, A_{t+1})^\top \boldsymbol{w}_t - \Phi(S_t, A_t)^\top \boldsymbol{w}_t) \Phi(S_t, A_t)$

Lim. eq.: $\mathbb{E}_\pi[r(S_T, A_t)\Phi(S_t, A_t)] = \mathbb{E}_\pi\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \gamma \Phi(S_{t+1}, A_{t+1})^\top\right)\right] \boldsymbol{w}_\infty$

ODE: $\dfrac{d\boldsymbol{w}}{dt} = -\mathbb{E}_\pi\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \gamma \Phi(S_{t+1}, A_{t+1})^\top\right)\right](\boldsymbol{w} - \boldsymbol{w}_\infty)$

Linear Parametrization and TD

- Convergence of ODE if $\mathbb{E}_\pi\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \gamma \Phi(S_{t+1}, A_{t+1})^\top\right)\right]$ has complex eigenvalues with positive real parts...
- which can be proved to be true under an ergodicity assumption!
- Need to explore all state-action pairs!
- Different solution than MC! Minimization of the Projected Bellman Residual...
- **Prop:**

$$\overline{VE}(\boldsymbol{w}_{\mathrm{TD}}) \leq \frac{1}{1-\gamma} \overline{VE}(\boldsymbol{w}_{\mathrm{MC}}) = \frac{1}{1-\gamma} \min_{\boldsymbol{w}} \overline{VE}(\boldsymbol{w})$$

Least-Squares TD

$$b = \mathbb{E}_\pi[r(S_T, A_t)\Phi(S_t, A_t)] \sim \frac{1}{t}\sum_{t'=0}^{t-1} R_{t'+1}\phi(S_{t'}, A_{t'})$$

$$A = \mathbb{E}_\pi\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \gamma\Phi(S_{t+1}, A_{t+1})^\top\right)\right]$$

$$\sim \frac{1}{t}\sum_{t'=0}^{t-1} \Phi(S_{t'}, A_{t'})\left(\Phi(S_{t'}, A_{t'})^\top - \gamma\Phi(S_{t'+1}, A_{t'+1})^\top\right)$$

Least-Squares TD

- Bypass the Stochastic Approximation scheme by estimating directly its limit:
$$\boldsymbol{w}_\infty = A^{-1}b$$
- Much more sample efficient.
- Recursive implementation possible.
- Recursive implementation maintaining an estimate of A^{-1} is also possible.

Advanced Returns

$$\text{Return: } \tilde{G}_t = \tilde{R}_{t+1} + \tilde{\Phi}_t^\top \boldsymbol{w} \quad \text{(affine formula)}$$

$$\text{Iteration: } \boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha_t(\tilde{R}_t + \tilde{\Phi}_t^\top \boldsymbol{w}_t - \Phi(S_t, A_t)^\top \boldsymbol{w}_t)\Phi(S_t, A_t)$$

$$\text{Lim. eq.: } \mathbb{E}_\pi\left[\tilde{R}_t \Phi(S_t, A_t)\right] = \mathbb{E}_\pi\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \Phi_t{}^\top\right)\right] \boldsymbol{w}_\infty$$

$$\text{ODE: } \frac{d\boldsymbol{w}}{dt} = -\mathbb{E}_\pi\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \tilde{\Phi}_t^\top\right)\right](\boldsymbol{w} - \boldsymbol{w}_\infty)$$

Linear Parametrization and TD

- Convergence of ODE if $\mathbb{E}_\pi\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \tilde{\Phi}_t^\top\right)\right]$ has complex eigenvalues with positive real parts...
- which can be proved to be true for the advanced returns under an ergodicity assumption!

Outline

On-Policy Prediction

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + 2\alpha_t \overbrace{\left(\tilde{G}_t - Q_{\boldsymbol{w}_t}(S_t, A_t)\right)}^{\delta_t} \nabla Q_{\boldsymbol{w}_t}(S_t, A_t)$$

On-line TD Algorithm

- Use the policy Π to obtain the interactions $S_t A_t R_{t+1} S_{t+1} A_{t+1} \ldots$
- Convergence... for linear parametrization under stationarity and coverage assumptions!
- Appear to *converge* even with more complex parametrization.

- Monte Carlo can be used for short episodes.
- Similar observations for elegibility trace.

On-Policy Control

$$\mathbf{w}_{t+1} = \mathbf{w}_t + 2\alpha_t \left(\tilde{G}_t - Q_{\mathbf{w}_t}(S_t, A_t) \right) \nabla Q_{\mathbf{w}_t}(S_t, A_t)$$

$$\pi_{t+1}(s) = \operatorname{argmax} Q_{\mathbf{w}_t}(s, \cdot) \quad \text{(plus exploration)}$$

On-Policy Control

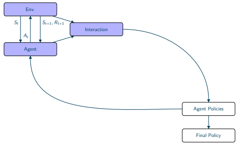
- SARSA type algorithm: update Q values and policy π while using policy π.
- Not a Stochastic Approximation algorithm anymore. . .
- Not approximate policy improvement as no sup-norm control. . .
- No proof of convergence... but appear to work well in practice.

- Non trivial scheduling issue in the definition of $\tilde{G}_t$.
- More constraints with eligibility trace.

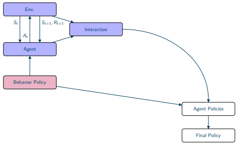
Outline

On-Policy vs Off-Policy

From



to



On-Policy vs Off-Policy

- On-Policy: the policy b used to interact is the same than the policy Π evaluated or optimized.
- Off-Policy: the policy b used to interact may be different from the policy Π evaluated or optimized.

- Off-Policy correction available for the return.

Off-Policy Prediction

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha_t \left(\tilde{G}_t - Q_{\mathbf{w}_t}(S_t, A_t) \right) \nabla Q_{\mathbf{w}_t}(S_t, A_t)$$

Off-policy TD Algorithm

- Use a policy b to obtain the interactions $S_t A_t R_{t+1} S_{t+1} A_{t+1} \dots$
- Compute an (importance-sampling based) corrected return.
- Use it in the algorithm.

- **Can fail spectacularly!**
- Monte Carlo will work.

Off-Policy Divergence

Simplest Example?

- Simple transition with a reward 0.
- TD error:

$$\begin{aligned}\delta_t &= R_{t+1} + \gamma V_{\boldsymbol{w}_t}(S_{t+1}) - V_{\boldsymbol{w}_t}(S_t) \\ &= 0 + \gamma 2\boldsymbol{w}_t - \boldsymbol{w}_t = (2\gamma - 1)\boldsymbol{w}_t\end{aligned}$$

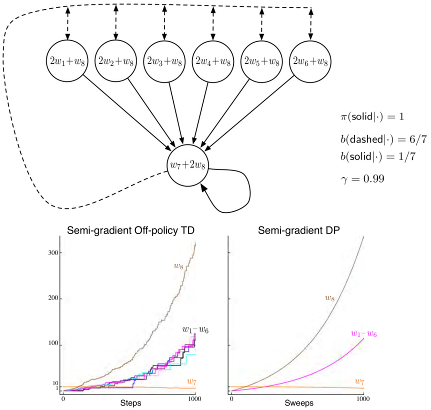
- Off-policy semi-gradient TD(0) update:

$$\begin{aligned}\boldsymbol{w}_{t+1} &= \boldsymbol{w}_t + \alpha_t \rho_t \delta_t \nabla V(S_{t+1}, \boldsymbol{w}_t) \\ &= \boldsymbol{w}_t + \alpha_t \times 1 \times (2\gamma - 1)\boldsymbol{w}_t = (1 + \alpha_t(2\gamma - 1))\boldsymbol{w}_t\end{aligned}$$

- Explosion if this transition is explored without $\boldsymbol{w}$ being update on other transitions as soon as $\gamma > 1/2$.

Discounted

Off-Policy Divergence



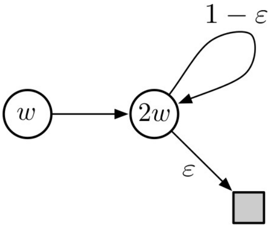
Baird's Counterexample

- Divergence of off-policy algorithm even without sampling, i.e. in Dynamic Programming.

Discounted

Source: Sutton and Barto

Off-Policy Divergence



Tsistiklis and Van Roy's Counterexample

- Exact minimization of bootstrapped $\overline{VE}$ at each step:

$$\begin{aligned} \mathbf{w}_{t+1} &= \operatorname*{argmin}_{\mathbf{w}} \sum_{s} \left(V_{\mathbf{w}_t}(s) - \mathbb{E}_\pi[R_{t+1} + \gamma V_{\mathbf{w}_t}(S_{t+1}) | S_t = s]\right)^2 \\ &= \operatorname*{argmin}_{\mathbf{w}} (\mathbf{w} - \gamma 2\mathbf{w}_t)^2 + (2\mathbf{w} - (1-\epsilon)\gamma 2\mathbf{w}_t)^2 \\ &= \frac{6-4\epsilon}{5}\gamma \mathbf{w}_t \end{aligned}$$

- Divergence if $\gamma > 5/(6-4\epsilon)$.

Linear Parametrization and TD

Iteration: $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha_t (R_{t+1} + \gamma \sum_a \pi(a|S_{t+1}) \Phi(S_{t+1}, a)^\top \boldsymbol{w}_t - \Phi(S_t, A_t)^\top \boldsymbol{w}_t) \Phi(S_t, A_t)$

Lim. eq$\mathbb{E}_b[r(S_T, A_t)\Phi(S_t, A_t)] = \mathbb{E}_b\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \gamma \sum_a \pi(a|S_{t+1})\Phi(S_{t+1}, a)^\top\right)\right] \boldsymbol{w}_\infty$

ODE: $\frac{d\boldsymbol{w}}{dt} = -\mathbb{E}_b\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \gamma \sum_a \pi(a|S_{t+1})\Phi(S_{t+1}, a)^\top\right)\right](\boldsymbol{w} - \boldsymbol{w}_\infty)$

Linear Parametrization and TD

- Convergence of ODE if

$$\mathbb{E}_b\left[\Phi(S_t, A_t)\left(\Phi(S_t, A_t)^\top - \gamma \sum_a \pi(a|S_{t+1})\Phi(S_{t+1}, q^\top\right)\right] = \Phi\Xi(I - \gamma P^\pi)\Phi^\top$$

  (with $\Phi = (\Phi(s, a))$, $\Xi = \text{diag}(\mu_b(s, a)))$ and $P\pi$ the transition matrix associated to π) has complex eigenvalues with positive real parts...
- Proof for on-policy relies on $\mu_b = \mu_\pi$ which satisfies $\mu_\pi{}^\top P_\pi = \mu_\pi{}^\top$.
- Not true anymore with an arbitrary behavior policy!

Discounted

Deadly Triad

Deadly Triad

- **Function approximation**
- **Bootstrapping**
- **Off-policy training**

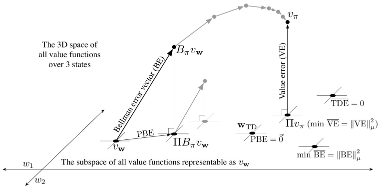
- **Instabilities as soon as the three are present!**

Issue

- Function approximation is unavoidable.
- Bootstrap is much more computational and data efficient.
- Off-policy may be avoided. . . but essential when dealing with extended setting (learn from others or learn several tasks)

- Dead End?

Target?



Linear Parametrization Target?

- Prediction objective $\overline{VE}$:

$$\|q_\pi - Q_{\boldsymbol{w}}\|_\mu^2$$

- Bellman Error $\overline{BE}$:

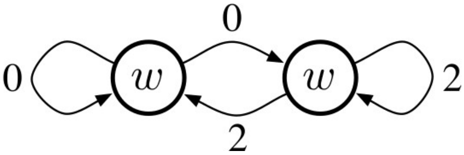
$$\|\mathcal{T}^\pi Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\|_\mu^2$$

- Projected Bellman Error $\overline{PBE}$:

$$\|\operatorname{Proj}\mathcal{T}^\pi Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}}\|_\mu^2$$

with $\operatorname{Proj} = \Phi(\Phi^\top \Xi \Phi)\Phi\,(\Phi)\,\Xi$.

Prediction Objective



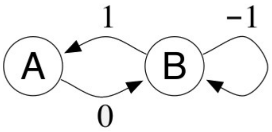
Prediction Objective

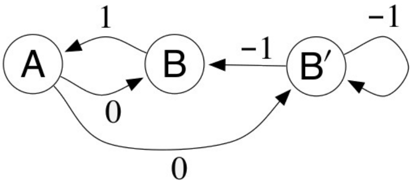
- Two MRP with the same outputs (because of approximation).
- but different $\overline{VE}$.
- Impossibility to learn $\overline{VE}$.
- Minimizer however is learnable:

$$\begin{aligned}\overline{RE}(\boldsymbol{w}) &= \mathbb{E}\left[(G_t - V_{\boldsymbol{w}_t}(S_t))^2\right] \\ &= \overline{VE}(\boldsymbol{w}) + \mathbb{E}\left[(G_t - v_\pi(S_t))^2\right]\end{aligned}$$

- MC method target.

Bellman Error



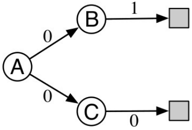


Bellman Error

- Two MRP with the same outputs (because of approximation).
- Different $\overline{BE}$.
- Different minimizer!
- $\overline{BE}$ is not learnable!

TD Error

$$\overline{TDE}(\boldsymbol{w}) = \|\mathbb{E}_\pi\left[\delta_t^2 | S_t, A_t\right]\|_\mu$$



Mean-Squares TD Error

- $\overline{TDE}(\boldsymbol{w}) = \mathbb{E}_b\left[\rho_t \delta^2\right]$
- Gradient: $\nabla \overline{TDE}(\boldsymbol{w}) = \mathbb{E}_b[\rho_t \left(R_t + \gamma Q_{\boldsymbol{w}}(S_{t+1}, A_{t+1})\right) - Q_{\boldsymbol{w}_t}(S_t, A_t)) \left(\gamma \nabla Q_{\boldsymbol{w}_t}(S_{t+1}, A_{t+1}) - \nabla Q_{\boldsymbol{w}_t}(S_t, A_t)\right)]$
- SGD algorithm. . .
- but solutions often converge to not to a *desirable place* even without approximation!

Discounted

Projected Bellman Error

$$\| \operatorname{Proj} \mathcal{T}^\pi Q_{\boldsymbol{w}} - Q_{\boldsymbol{w}} \|_\mu^2 \quad \text{with } \operatorname{Proj} = \Phi(\Phi^\top \Xi \Phi)^{-1} \Phi^\top \Xi.$$

Projected Bellman Error

- Rewriting

$$\begin{aligned}\overline{PBE}(\boldsymbol{w}) &= \| \operatorname{Proj} \mathcal{T}^\pi q_{\boldsymbol{w}} - q_{\boldsymbol{w}} \|_\mu^2 = \| \operatorname{Proj} \delta_{\boldsymbol{w}} \|_\mu^2 \\ &= (\operatorname{Proj} \delta_{\boldsymbol{w}})^\top \Xi (\operatorname{Proj} \delta_{\boldsymbol{w}}) = (\Phi^\top \Xi \delta_{\boldsymbol{w}})^\top \left(\Phi^\top \Xi \Phi\right)^{-1} (\Phi^\top \Xi \delta_{\boldsymbol{w}})\end{aligned}$$

- Gradient:

$$\nabla \overline{PBE}(\boldsymbol{w}) = 2 \nabla (\Phi^\top \Xi \delta_{\boldsymbol{w}})^\top \left(\Phi^\top \Xi \Phi\right)^{-1} (\Phi^\top \Xi \delta_{\boldsymbol{w}})$$

- Expectations:

$$\begin{aligned}\Phi^\top \Xi \delta_{\boldsymbol{w}} &= \mathbb{E}_b[\rho_t \delta_t \Phi(S_t, A_t)] \\ \nabla (\Phi^\top \Xi \delta_{\boldsymbol{w}})^\top &= \mathbb{E}_b\left[\rho_t (\gamma \Phi(S_{t+1}, A_{t+1}) - \Phi(S_t, A_t)) \Phi(S_t, A_t)^\top\right] \\ \Phi^\top \Xi \Phi &= \mathbb{E}_b\left[\Phi(S_t, A_t) \Phi(S_t, A_t)^\top\right]\end{aligned}$$

- Not yet a SGD/SA as the gradient is a product of several terms...

Projected Bellman Error

Gradient and Stochastic Approximation

- Gradient:

$$\nabla \overline{PBE}(\boldsymbol{w}) = 2\mathbb{E}_b\left[\rho_t(\gamma\Phi(S_{t+1}, A_{t+1}) - \Phi(S_t, A_t))\Phi(S_t, A_t)^\top\right] \left(\mathbb{E}_b\left[\Phi(S_t, A_t)\Phi(S_t, A_t)^\top\right]\right)^{-1} \mathbb{E}_b[\rho_t\delta_t\Phi(S_t, A_t)]$$

- Least-squares inside:

$$v = \left(\mathbb{E}_b\left[\Phi(S_t, A_t)\Phi(S_t, A_t)^\top\right]\right)^{-1} \mathbb{E}_b\left[\rho_t\delta_t\Phi(S_t, A_t)^\top\right]$$

$$\Leftrightarrow v = \operatorname*{argmin}_v \mathbb{E}_b\left[\left(\Phi(S_t, A_t)^\top v_t - \rho_t\delta_t\right)^2\right]$$

  which can be estimated by

$$v_{t+1} = v_t + \beta_t\Phi(S_t, A_t)(\delta_t - \rho_t\Phi(S_t, A_t)^\top v_t)$$

- Plugin pseudo gradient (SA):

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - 2\alpha_t\rho_t(\gamma\Phi(S_{t+1}, A_{t+1}) - \Phi(S_t, A_t))\Phi(S_t, A_t)^\top v_t$$

- Same target than Pseudo Gradient but converging algorithm provided $\alpha_t \ll \beta_t$.

Gradient TD Algorithm

GTD

- Simultaneous update:

$$v_{t+1} = v_t + \beta_t \Phi(S_t, A_t)(\delta_t - \rho_t \Phi(S_t, A_t)^\top v_t)$$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - 2\alpha_t \rho_t (\gamma \Phi(S_{t+1}, A_{t+1}) - \Phi(S_t, A_t))\Phi(S_t, A_t)^\top v_t$$

- As $\alpha_t \ll \beta_t$, $\boldsymbol{w}$ is seen as constant by v...

TDC

- Simultaneous update:

$$v_{t+1} = v_t + \beta_t \Phi(S_t, A_t)(\delta_t - \rho_t \Phi(S_t, A_t)^\top v_t)$$

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - 2\alpha_t \rho_t (\delta_t \Phi(S_t, A_t) - \gamma \Phi(S_{t+1}, A_{t+1}))\Phi(S_t, A_t)^\top v_t$$

- Obtained by a similar derivation but faster in practice...
- As $\alpha_t \ll \beta_t$, $\boldsymbol{w}$ is seen as constant by v...

- Restricted to the linear setting but interesting insight.

Outline

Stochastic Approximation

$$\theta_{k+1} = \theta_k + \alpha_k h_k(\theta_k) \quad \text{with} \quad h_k(\theta) = H(\theta) + \epsilon_k + \eta_k$$
$$\implies \theta_k \to \{\theta, H(\theta) = 0\}$$

Stochastic Approximation

- Family of sequential stochastic algorithm converging to a zero of a function.
- Classical assumptions:
  - $\mathbb{E}[\epsilon_k] = 0$, $\mathbb{V}\text{ar}\,[\epsilon_k] < \sigma^2$, and $\mathbb{E}[\|\eta_k\|] \to 0$,
  - $\sum_k \alpha_k \to \infty$ and $\sum_k \alpha_k^2 < \infty$,
  - the algorithm converges if we replace h_k by H.

- Convergence toward a neighborhood if α is kept constant (as often in practice).
- Most famous example are probably Robbins-Monro and SGD.
- Proof quite technical in general.
- The convergence with H is easy to obtain for a contraction.

Stochastic Approximation and ODE

From $\theta_{k+1} = \theta_k + \alpha_k h_k(\theta_k)$ with $h_k(\theta) = H(\theta) + \epsilon_k + \eta_k$

to $\dfrac{d\tilde{\theta}}{dt} = H(\tilde{\theta})$

ODE Approach

- General proof showing that the algorithm converges provided the ODE converges.
- Rely on the rewriting the equation

$$\frac{\theta_{k+1} - \theta_k}{\alpha_k} = h_k(\theta_k) = H(\theta_k) + \epsilon_k + \eta_k$$

- α_k can be interpreted as a time difference allowing to define a time $t_k = \sum_{t' \leq t} \alpha_k$.
- $\theta(t)$ is piecewise affine and defined through its derivative at time $t \in (t_k, t_{k+1})$.
- This piecewise function remains close to any solution of the ODE starting from θ_k for an arbitrary amount of time provided k is large enough.

- More general proofs based on martingale.

Stochastic Approximation

$$\begin{cases} \theta_{k+1} = \theta_k + \alpha_k h_k(\theta_k, \nu_k) \\ \nu_{k+1} = \nu_k + \beta_k g_k(\theta_k, \nu_k) \end{cases} \quad \text{with} \quad \begin{cases} h_k(\theta, \nu) = H(\theta, \nu) + \epsilon_k + \eta_k \\ g_k(\theta, \nu) = G(\theta, \nu) + \epsilon'_k + \eta'_k \end{cases}$$

$$\Longrightarrow \theta_k \to \{\theta, H(\theta, \nu(\theta)) = 0, \nu(\theta) \in \{\nu, G(\theta, \nu) = 0\}\}$$

Stochastic Approximation

- Family of sequential stochastic algorithm converging to a zero of a function.
- Classical assumptions:
  - $\mathbb{E}[\epsilon_k] = 0$, $\mathbb{V}\text{ar}\,[\epsilon_k] < \sigma^2$, and $\mathbb{E}[\|\eta_k\|] \to 0$,
  - $\sum_k \alpha_k \to \infty$ and $\sum_k \alpha_k^2 < \infty$,
  - $\sum_k \beta_k \to \infty$ and $\sum_k \beta_k^2 < \infty$,
  - $\alpha_k/\beta_k \to 0$ (two-scales assumption),
  - the algorithm converges if we replace h_k and g_k by H and G.

- Convergence toward a neighborhood if $\alpha \ll \beta$ are kept constant (as often in practice).

Stochastic Approximation and ODE

From $\begin{cases} \theta_{k+1} = \theta_k + \alpha_k h_k(\theta_k, \nu_k) \\ \nu_{k+1} = \nu_k + \beta_k + g_k(\theta_k, \nu_k) \end{cases}$ with $\begin{cases} h_k(\theta, \nu) = H(\theta, \nu) + \epsilon_k + \eta_k \\ g_k(\theta, \nu) = G(\theta, \nu) + \epsilon'_k + \eta'_k \end{cases}$

to $\dfrac{d\tilde{\theta}}{dt} = H(\tilde{\theta}, \tilde{\nu}(\tilde{\theta}))$ with $\tilde{\nu}(\theta)$ the limit of $\dfrac{d\tilde{\nu}}{dt} = G(\theta, \tilde{\nu})$

ODE Approach

- General proof showing that the algorithm converges provided the two ODE converge.
- Quite generic setting and source of new algorithm or insight on existing ones.
- Importance of having two scales. . .

- Can be used to prove the convergence of GTD and TDC!

Outline

Simplified Deep Q-Learning

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \beta_t \left(R_{t+1} + \gamma \max_a Q_{\nu_t}(S_{t+1}, a) - Q_{\mathbf{w}_t}(S_t, A_t)\right) \nabla Q_{\mathbf{w}_t}(S_t, A_t)$$

$$\nu_t = \mathbf{w}_{\lceil t/T \rceil T}$$

Simplified Deep Q-Learning

- Stochastic Approximation for a fixed ν:
  - Limiting equation:
  $$\mathbb{E}_b[(\mathcal{T}^\star Q_\nu(S_t, A_t) - Q_{\mathbf{w}_\infty}(S_t, A_t)) \nabla Q_{\mathbf{w}_\infty}(S_t, A_t)] = 0$$
  - Stochastic Gradient Descent of
  $$\mathbb{E}_b\left[(\mathcal{T}^\star Q_\nu(S_t, A_t) - Q_{\mathbf{w}}(S_t, A_t))^2\right]$$
  - $Q_{\mathbf{w}} \to \mathcal{T}^\star Q_\nu$

  $Q_{\nu_{t+T}} \approx \mathcal{T}^\star Q_t$
- Approximate Value Iteration Scheme!

- Two-scales algorithm flavour as ν is kept constant.
- Explicit two scales with $\nu_{t+1} = \nu_t + \alpha_t(\mathbf{w}_t - \nu_t)$ variation.
- Could be used for prediction with $R_{t+1} + \gamma \sum_a \pi(a|S_{t+1}) Q_{\nu_t}(S_{t+1}, a)$

Deep Q-Learning

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \beta_t(R_t + \gamma \max_a Q_{\nu_t}(S_{t+1}, a) - Q_{\mathbf{w}}(S_t, A_t))\nabla Q_{\mathbf{w}}(S_t, A_t)$$

$$\nu_t = \mathbf{w}_{\lceil t/T \rceil T}$$

- **Who are $S_t, A_t, R_{t+1}, S_{t+1}$?** and thus to what corresponds $\mathbb{E}_b$?

Simplified Deep Q-Learning

- Use a behaviour policy b.
- The current greedy plus exploration Q-policy can be used.

Neural Fitted-Q

- Instead of a policy b, use a fix dataset $\mathcal{D}$ of $S_t, A_t, R_{t+1}, S_{t+1}$.
- Several pass on the data can be made.

Deep Q-Learning

- Use the current greedy plus exploration Q-policy to populate a FIFO buffer $\mathcal{D}$.
- Use random samples of the buffer $\mathcal{D}_t$ (more than one per interaction is OK).

Deep Q-Learning

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \beta_t (R_t + \gamma \max_a Q_{\nu_t}(S_{t+1}, a) - Q_{\mathbf{w}}(S_t, A_t)) \nabla Q_{\mathbf{w}}(S_t, A_t)$$

$$\nu_t = \mathbf{w}_{\lceil t/T \rceil T}$$

Plus tricks

Deep Q-Learning Tricks

- Replay buffer
- Double Q-Learning
- Better Exploration
- Advanced Return and Distributional
- Network Architecture

Rainbow Paper

- Rainbow paper. . .

Replay Buffer

Replay Buffer

- Replace an expectation over real trajectories by an empirical average over past (short) sub-trajectories stored in a replay buffer.
- The empirical average corresponds to uniform sampling.
- If the policy is changing across time, we should use a importance sampling correction to be faithful with the theory. . .
- Not necessary for one-step Q learning but required for most of the other methods where replay buffer is used.
- Often no correction in practice if the policies used in the buffer are closed to the current one.
- Prioritized sweeping variant possible. . .

- Buffer can be constructed in parallel of the learning part.
- Only requires to transmit the *current* greedy plus exploration Q-policy.

Double Q-Learning

Q-Learning and overestimation

- Target: $R_{s,a} + \gamma \max_{a'} Q_{\mathbf{w}}(s', a')$
- Approximation issue: $Q_{\mathbf{w}}(s', a') \sim Q(s, a) + \epsilon(s, a)$
- Consequence: $\mathbb{E}[\max_a Q_{\mathbf{w}}(S_t, a)] \geq \max\left(Q(s, a) + \mathbb{E}[\epsilon(s, a)]\right)$

Double Q-Learning with two Q functions: $Q_{\mathbf{w}_1}$ and $Q_{\mathbf{w}_2}$

- Used in a crossed way for the target of $Q_{\mathbf{w}_i}$:

$$R_{s,a} + \gamma Q_{\mathbf{w}_{i'}}(s', \operatorname*{argmax}_{a'} Q_{\mathbf{w}_i}(s', a'))$$

- Mitigates the bias.

Clipped Q-Learning with several Q functions: $Q_{\mathbf{w}_i}$

- Used in a pessimistic way for the target of $Q_{\mathbf{w}_i}$:

$$R_{s,a} + \gamma \min_{i'} Q_{\mathbf{w}_{i'}}(s', \operatorname*{argmax}_{a'} Q_{\mathbf{w}_i}(s', a'))$$

- Seems even more efficient.

Outline

Continuous Action

- Case (almost) not yet covered in the lectures.
- Most complex theoretical extension.

Prediction

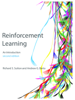
- No algorithmic issue if one can sample π.
- Off-policy can be considered under a domination assumption.

Planning

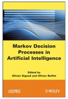
- Main issue is the argmax of the greedy policy (or the sampling of Gibbs policy).
- May be impossible to compute.
- Possible if the parametrization of Q with respect to a is simple (e.g. explicit quadratic dependency in a).
- An alternative could be to approximate the argmax operator, or to learn how to approximate the argmax directly, which is very close to approximating directly the policy itself. . .

Outline

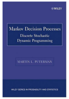
References



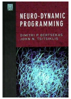
R. Sutton and A. Barto.
Reinforcement Learning, an Introduction (2nd ed.)
MIT Press, 2018



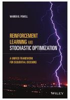
O. Sigaud and O. Buffet.
Markov Decision Processes in Artificial Intelligence.
Wiley, 2010



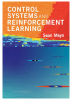
M. Puterman.
Markov Decision Processes. Discrete Stochastic Dynamic Programming.
Wiley, 2005



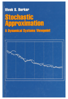
D. Bertsekas and J. Tsitsiklis.
Neuro-Dynamic Programming.
Athena Scientific, 1996



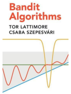
W. Powell.
Reinforcement Learning and Stochastic Optimization: A Unified Framework for Sequential Decisions.
Wiley, 2022



S. Meyn.
Control Systems and Reinforcement Learning.
Cambridge University Press, 2022



V. Borkar.
Stochastic Approximation: A Dynamical Systems Viewpoint.
Springer, 2008



T. Lattimore and Cs. Szepesvári.
Bandit Algorithms.
Cambridge University Press, 2020

Licence and Contributors

Contributors

- Main contributor: E. Le Pennec
- Contributors: S. Boucheron, A. Dieuleveut, A.K. Fermin, S. Gadat, S. Gaiffas, A. Guilloux, Ch. Keribin, E. Matzner, M. Sangnier, E. Scornet.