<table>
<thead>
<tr>
<th>1</th>
<th>Sequential Decisions, MDP and Policies</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Decision Process and Markov Decision Process</td>
<td></td>
</tr>
<tr>
<td>• Returns and Value Functions</td>
<td></td>
</tr>
<tr>
<td>• Prediction and Planning</td>
<td></td>
</tr>
<tr>
<td>• Operations Research and Reinforcement Learning</td>
<td></td>
</tr>
<tr>
<td>• Control</td>
<td></td>
</tr>
<tr>
<td>• Survey</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2</th>
<th>Operations Research: Prediction and Planning</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Prediction and Bellman Equation</td>
<td></td>
</tr>
<tr>
<td>• Prediction by Dynamic Programming and Contraction</td>
<td></td>
</tr>
<tr>
<td>• Planning, Optimal Policies and Bellman Equation</td>
<td></td>
</tr>
<tr>
<td>• Linear Programming</td>
<td></td>
</tr>
<tr>
<td>• Planning by Value Iteration</td>
<td></td>
</tr>
<tr>
<td>• Planning by Policy Iteration</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3</th>
<th>Reinforcement Learning: Prediction and Planning in the Tabular Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Prediction with Monte Carlo</td>
<td></td>
</tr>
<tr>
<td>• Planning with Monte Carlo</td>
<td></td>
</tr>
<tr>
<td>• Prediction with Temporal Differentials</td>
<td></td>
</tr>
<tr>
<td>• Link with Stochastic Approximation</td>
<td></td>
</tr>
<tr>
<td>• Planning with Value Iteration</td>
<td></td>
</tr>
<tr>
<td>• Planning with Policy Improvement</td>
<td></td>
</tr>
<tr>
<td>• Exploration vs Exploitation</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4</th>
<th>Reinforcement Learning: Advanced Techniques in the Tabular Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>• n-step Algorithms</td>
<td></td>
</tr>
<tr>
<td>• Eligibility Traces</td>
<td></td>
</tr>
<tr>
<td>• Off-policy vs on-policy</td>
<td></td>
</tr>
<tr>
<td>• Bandits</td>
<td></td>
</tr>
<tr>
<td>• Model Based Approach</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5</th>
<th>Reinforcement Learning: Approximation of the Value Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Approximation Target(s)</td>
<td></td>
</tr>
<tr>
<td>• Gradient and Pseudo-Gradient</td>
<td></td>
</tr>
<tr>
<td>• Linear Approximation and LSTD</td>
<td></td>
</tr>
<tr>
<td>• Off-Policy Prediction and Control</td>
<td></td>
</tr>
<tr>
<td>• Two-Scales Algorithms</td>
<td></td>
</tr>
<tr>
<td>• Deep Q Learning</td>
<td></td>
</tr>
<tr>
<td>• Continuous Actions</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6</th>
<th>Reinforcement Learning: Policy Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Policy Gradient Theorems</td>
<td></td>
</tr>
<tr>
<td>• Monte Carlo Based Policy Gradient</td>
<td></td>
</tr>
<tr>
<td>• Actor / Critic Principle</td>
<td></td>
</tr>
<tr>
<td>• 3 SOTA Algorithms</td>
<td></td>
</tr>
<tr>
<td>• Average Return</td>
<td></td>
</tr>
</tbody>
</table>

| 7 | References |
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Decision or Decisions

Sequential Decisions, MDP and Policies

Source: W. Powell
Sequential Decision Setting

- In many (most?) settings, not a single decision but a sequence of decisions.
- Need to take into account the (not necessarily immediate) consequences of the sequence of decisions/actions rather than of each decision.
- Different framework than supervised learning (no immediate feedback here) and unsupervised learning (well defined goal here).
From Sequential Decision to Reinforcement Learning

Sequential Decision

- Sequence of action $A_t$ as a response of an environment $S_t$
- Feedback through a reward $R_t$

Actions?

- Is my current way of choosing actions good?
- How to make it better?
From Sequential Decision to Reinforcement Learning

Sequential Decisions, MDP and Policies

**Markov Decision Process Modeling**
- Specific modeling of the environment.
- Goal as as a (weighted) sum of a scalar reward.

**Actions?**
- Is my current way of choosing actions good?
- How to make it better?
From Sequential Decision to Reinforcement Learning

Sequential Decision

MDP Modeling

Reinforcement Learning

Reinforcement Learning

- Same modeling...
- But no direct knowledge of the MDP.

Actions?

- Is my current way of choosing actions good?
- How to make it better?
Sequential Decision Settings

- MDP / Reinforcement Learning:
  \[ \max \pi \mathbb{E}_\pi \left[ \sum_t R_t \right] \]

- Optimal Control:
  \[ \min_u \mathbb{E} \left[ \sum_t C(x_t, u_t) \right] \]

- (Stochastic) Search:
  \[ \max \theta \mathbb{E}[F(\theta, W)] \]

- Online Regret:
  \[ \max \sum_k \mathbb{E}[F(\theta_k, W)] \]
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.  
Wiley, 2005

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

W. Powell.  
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
Decision Process and Environment

- At time step $t \in \mathbb{N}$:
  - State $S_t \in \mathcal{S}$: representation of the environment
  - Action $A_t \in \mathcal{A}(S_t)$: action chosen
  - Reward $R_{t+1} \in \mathcal{R}$: instantaneous reward
  - New state $S_{t+1}$

- Focus on the discrete setting, i.e. $\mathcal{S}$ finite, $\mathcal{A}(s)$ finite and $\mathcal{R}$ finite.
- Extension: Non finite bounded $\mathcal{R}$: easy / Non finite $\mathcal{S}$: hard / Non finite $\mathcal{A}$: harder.
Stochastic Model

- Dynamic defined by:

\[ \mathbb{P}(S_{t+1} = s', R_{t+1} = r | (S_{t'}, A_{t'}, R_{t'}), t' \leq t) \]

\[ = \mathbb{P}(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a, H_t) \]

where \( H_t = (R_t, S_{t-1}, A_{t-1}, R_{t-1}, S_{t-2}, \ldots) \) is the past and \( (S_t, A_t) \) the present.
Markov Decision Process and Environment

- **Markovian Dynamic Assumption:** \( S_{t+1} \) and \( R_{t+1} \) are independent of the past \( H_t = (R_t, S_{t-1}, A_{t-1}, R_{t-1}, S_{t-2}, \ldots) \) conditionally to the present \((S_t, A_t)\).

- Dynamic entirely defined by state-reward transition probabilities

\[
\mathbb{P}(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a, H_t) = \mathbb{P}(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a) = p(s', r | s, a)
\]

in the discrete setting.

- Informally, this means that \( S_t \) encodes all the information related to the past.
Markov Decision Process and State-Action

- State-Reward transition probabilities for a given state-action:
  \[ P(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a, H_t) = P(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a) = p(s', r | s, a) \]

Induced State-action laws

- State transition probabilities for a given state-action:
  \[ P(S_{t+1} = s' | S_t = s, A_t = a, H_t) = P(S_{t+1} = s' | S_t = s, A_t = a) = p(s' | s, a) = \sum_r p(s', r | s, a) \]

- Expected reward for a given state-action:
  \[ E[R_{t+1} | S_t = s, A_t = a, H_t] = E[R_{t+1} | S_t = s, A_t = a] = r(s, a) = \sum_r \sum_{s'} r p(s', r | r, a) \]

- From now on, we will always assume that the Markovian property holds for the environment.
### Examples

### Sequential Decisions, MDP and Policies

| $s$ | $a$     | $s'$   | $p(s'|s,a)$ | $r(s,a,s')$ |
|-----|---------|--------|-------------|-------------|
| high | search | high  | $\alpha$    | $r_{\text{search}}$ |
| high | search | low   | $1 - \alpha$| $r_{\text{search}}$ |
| low  | search | high  | $1 - \beta$ | $-3$        |
| low  | search | low   | $\beta$     | $r_{\text{search}}$ |
| high | wait   | high  | $1$         | $r_{\text{wait}}$ |
| high | wait   | low   | $0$         | $-$         |
| low  | wait   | high  | $0$         | $-$         |
| low  | wait   | low   | $1$         | $r_{\text{wait}}$ |
| low  | recharge | high  | $1$     | $0$         |
| low  | recharge | low   | $0$     | $-$         |

![Diagram showing a Markov Decision Process (MDP)](image)

Sources: Sutton and Barto / learndatasci.com / danijar.com
Decision Process, Agent and Policy

Agent

- Interact with the environment by choose the action given the past.

Policy $\Pi$ : specification of how to choose the action

- General stochastic policy $\Pi = (\pi_0, \pi_1, \ldots, \pi_t, \ldots)$:
  \[
  \Pi_t(A_t = a) = \pi_t(A_t = a | S_t = a, A_t = a, H_t)
  \]

- General deterministic policy $\Pi = (\pi_0, \pi_1, \ldots, \pi_t, \ldots)$ (with as slight abuse of notation):
  \[
  \Pi_t(A_t = a) = 1_{A_t = \pi_t(S_t = a, A_t = a, H_t)}
  \]
Sequential Decisions, MDP
and Policies

Agent

- Interact with the environment by choose the action given the past.

Policy Π: specification of how to choose the action

- History dependent stochastic policy $\Pi = (\pi_0, \pi_1, \ldots, \pi_t, \ldots)$:
  $$\Pi_t(A_t = a) = \pi_t(A_t = a | S_t = s, H_t)$$

- Markovian stochastic policy $\Pi = (\pi_0, \pi_1, \ldots, \pi_t, \ldots)$:
  $$\Pi_t(A_t = a) = \pi_t(A_t = a | S_t = s) = \pi_t(a | s)$$

- Stationary Markovian stochastic policy $\Pi = (\pi, \pi, \ldots, \pi, \ldots)$:
  $$\Pi_t(A_t = a) = \pi(A_t = a | S_t = s) = \pi(a | s)$$

- Similar deterministic policy definition.

- Partially Observed Markov Decision Process extension: the Agent has only access to a partial observation $O_t$ at each time step... (not the focus of the lectures)
Trajectories

- Trajectory \((S_0, A_0, R_1, S_1, A_1, \ldots)\) defined by
  - an initial distribution \(P_0\) for \(S_0\),
  - a policy \(\Pi = (\pi_0, \pi_1, \ldots, \pi_t, \ldots)\) specifying
    \[
    \Pi_t(A_t = a) = \pi_t(A_t = a | S_t, H_t)
    \]
  - an environment specifying
    \[
    P(S_{t+1}, R_{t+1} | S_t, A_t, H_t)
    \]
Decision Process and Trajectories

Trajectories

- Induced probability:

\[ \mathbb{P}(S_0 = s_0, A_0 = a_0, R_1 = r_1, S_1 = s_1, A_1 = a_1, \ldots S_t = s_t, R_t = r_t) \]

\[ = \mathbb{P}_0(S_0 = s_0) \times \pi_0(A_0 = a_0 | S_0) \mathbb{P}(S_1, R_1 | S_0, A_0) \pi_1(A_1 = a_1 | S_1 = s_1, H_1) \]

\[ \times \cdots \times \mathbb{P}(S_t = s_t, R_t = r_t | S_{t-1} = s_{t-1}, A_{t-1} = a_{n-1}, H_{t-1}) \]
Markov Decision Process and Trajectories

Trajectories
- Trajectory \((S_0, A_0, R_1, S_1, A_1, \ldots)\) defined by
  - an initial distribution \(P_0\) for \(S_0\),
  - a policy \(\Pi = (\pi_0, \pi_1, \ldots, \pi_t, \ldots)\) specifying
    \[ \Pi_t(A_t = a) = \pi_t(A_t = a | S_t, H_t) \]
  - a Markovian environment specifying
    \[ P(S_{t+1}, R_{t+1} | S_t, A_t) \]
Markov Decision Process and Trajectories

Trajectories

- Induced probability:
\[
P(S_0 = s_0, A_0 = a_0, R_1 = r_1, S_1 = s_1, A_1 = a_1, \ldots S_t = s_t, R_t = r_t)
= P_0(S_0 = s_0)
\times \pi_0(A_0 = a_0|S_0) P(S_1, R_1|S_0, A_0) \pi_1(A_1 = a_1|S_1 = s_1, H_1)
\times \cdots \times P(S_t = s_t, R_t = r_t|S_{t-1} = s_{t-1}, A_{t-1} = a_{t-1})
\]
Markov Decision Process and Trajectories

Markovian Trajectories only if the policy is Markovian

\[ \mathbb{P}(R_{t+1}, S_{t+1}, A_{t+1}, R_{t+2}, S_{t+2}, \ldots R_{t+k}, S_{t+k} | S_t, A_t, H_t) = \mathbb{P}(R_{t+1}, S_{t+1}, A_{t+1}, R_{t+2}, S_{t+2}, \ldots R_{t+k}, S_{t+k} | S_t, A_t) \]
\[ = \mathbb{P}(S_{t+1} | S_t, A_t) \pi_{t+1}(A_{t+1} | S_{t+1}) \times \cdots \times \mathbb{P}(S_{t+k} | S_{t+k-1}, A_{t+k-1}) \]

- Stationary if the policy is stationary.
Outline

1 Sequential Decisions, MDP and Policies
   • Decision Process and Markov Decision Process
   • Returns and Value Functions
     • Prediction and Planning
     • Operations Research and Reinforcement Learning
     • Control
     • Survey

2 Operations Research: Prediction and Planning
   • Prediction and Bellman Equation
   • Prediction by Dynamic Programming and Contraction
   • Planning, Optimal Policies and Bellman Equation
   • Linear Programming
   • Planning by Value Iteration
   • Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   • Prediction with Monte Carlo
   • Planning with Monte Carlo
   • Prediction with Temporal Differences
   • Link with Stochastic Approximation
   • Planning with Value Iteration
   • Planning with Policy Improvement
   • Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   • n-step Algorithms
   • Eligibility Traces
   • Off-policy vs on-policy
   • Bandits
   • Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   • Approximation Target(s)
   • Gradient and Pseudo-Gradient
   • Linear Approximation and LSTD
   • On-Policy Prediction and Control
   • Off-Policy and Deadly Triad
   • Two-Scales Algorithms
   • Deep Q Learning
   • Continuous Actions

6 Reinforcement Learning: Policy Approach
   • Policy Gradient Theorems
   • Monte Carlo Based Policy Gradient
   • Actor / Critic Principle
   • 3 SOTA Algorithms
   • Average Return

7 References
Rewards and Total Return

- MDP: Rewards $R_t$ encode all the feedbacks!
- Quality of a policy $\Pi$ measured from the remaining total return:

$$G_t = \sum_{t'=t+1}^{\infty} R_{t'}$$

- Expected total return following $\Pi$ starting from $s$:

$$\mathbb{E}_\Pi[G_t|S_t = s] = \sum_{t'=t+1}^{\infty} \mathbb{E}[R_{t'}|S_t = s]$$
Total Return: Issue and Fixes

**Issues**

- $G_t$ is a limiting process and thus may not be defined!
- Can diverge to $\pm \infty$ and not converge at all.

**Fixes?**

- Finite horizon: $G^T_t = \sum_{t'=t+1}^{T} R_{t'}$
- Episodic setting: the average duration before all $R_{t'} = 0$ is finite: $G_t = \sum_{t'=t+1}^{\infty} R_{t'}$
- Discounted setting: for $0 < \gamma < 1$, $G^\gamma_t = \sum_{t'=t+1}^{\infty} \gamma^{t'-t+1} R_{t'}$
- Average return: $\bar{G}_t = \lim_{T \to \infty} \frac{1}{T} \sum_{t'=t+1}^{t+T} R_{t'}$
Finite Horizon

\[ G_T^t = \sum_{t'=t+1}^{T} R_{t'} \]

**Finite Horizon Setting**

- Always well defined and easy to interpret.
- Loss of Markovianity as we need to know the time step...

- Can be put in a classical Markov framework!
  - Define an absorbing state \( s_{abs} \) (a state that cannot be escaped and from which the reward is always 0).
  - Extend the state space \( S \) to \( (S \times \{0, \ldots, T\}) \cup \{s_{abs}\} \).
  - Define an state reward transition probability:

\[
p(\tilde{s}', r|\tilde{s}, a) = \begin{cases} 
  p(s', t|s, a) & \text{if } \tilde{s} = (s, t), t < T \text{ and } \tilde{s}' = (s', t + 1) \\
  1 & \text{if } \tilde{s} = (s, t), t = T, \tilde{s}' = s_{abs} \text{ and } r = 0 \\
  1 & \text{if } \tilde{s} = s_{abs}, \tilde{s}' = s_{abs} \text{ and } r = 0 \\
  0 & \text{otherwise}
\end{cases}
\]
Episodic Setting

\[ G_t = \sum_{t'=t+1}^{\infty} R_{t'} \]

Assume that for any policy \( \Pi \) the average duration before \( R_t = 0 \) is finite:

\[ \mathbb{E}_\Pi \left[ \min_{t,R_{t'}=0,\forall t' \geq t} t \right] < +\infty \]

Strong assumption...

Easy to interpret.

Equivalent definition by replacing all the states from which \( R_t \) remains equal to 0 whatever the policy by a single absorbing state \( s_{\text{abs}} \) and assume that the average duration is the expectation of stopping time to reach this state is finite

\[ \mathbb{E}_\Pi \left[ \min_{t,S_t=s_{\text{abs}}} t \right] < +\infty \]
Sequential Decisions, MDP and Policies

Discounted

\[ G_t^\gamma = \sum_{t'=t+1}^{T} \gamma^{t'-(t+1)} R_{t'} \]

- Always defined but not that easy to interpret.
- Easiest theoretical setting!

- Equivalent to an episodic setting if one adds an absorbing state \( s_{abs} \) and change all state-reward transition probabilities to:

\[
p(s', r | s, a) = \begin{cases} 
\gamma p(s', r | s, a) & \text{if } s' \neq s_{abs}, s \neq s_{abs} \\
(1 - \gamma) & \text{if } s' = s_{abs}, r = 0, s \neq s_{abs} \\
1 & \text{if } s' = s_{abs}, r = 0, s = s_{abs} \\
0 & \text{otherwise}
\end{cases}
\]
Average Return Setting

\[ \bar{G}_t = \lim_{T \to \infty} \frac{1}{T} \sum_{t'=t+1}^{t+T} R_{t'} \]

**Average Return**

- Not always defined. (Cesaro Average)
- Always equal to 0 in the episodic setting!
- Natural definition in a *stationary* setting...
- Complex theoretical analysis!

- Under a strict stationarity assumption \((R_t \sim R_{t'})\), link with discounted setting as

\[
\mathbb{E}_\pi[G^\gamma_t] = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_\pi[R_{t+1}] = \frac{1}{1 - \gamma} \mathbb{E}_\pi[R_t] = \frac{1}{1 - \gamma} \mathbb{E}_\pi[\bar{G}_t]
\]
State Value Functions

- **Return expectation for a policy \( \Pi \) starting from \( s \) at time \( t \)**
  - **Finite horizon setting:**
    \[
    v_{t,\Pi}^T(s) = \mathbb{E}_\Pi[G_t^T | S_t = s] = \sum_{t'=t+1}^T \mathbb{E}_\Pi[R_{t'} | S_t = s]
    \]
  - **Episodic setting:**
    \[
    v_{t,\Pi}(s) = \mathbb{E}_\Pi[G_t | S_t = s] = \sum_{t'=t+1}^\infty \mathbb{E}_\Pi[R_{t'} | S_t = s]
    \]
  - **Discounted:**
    \[
    v_{t,\Pi}^\gamma(s) = \mathbb{E}_\Pi[G_t^\gamma | S_t = s] = \sum_{t'=t+1}^\infty \gamma^{t'-(t+1)} \mathbb{E}_\Pi[R_{t'} | S_t = s]
    \]
  - **Average return setting:**
    \[
    \overline{v}_{t,\Pi}(s) = \mathbb{E}_\Pi[\overline{G}_t | S_t = s] = \lim_{T \to \infty} \frac{1}{T} \sum_{t'=t+1}^{t+T} \mathbb{E}_\Pi[R_{t'} | S_t = s]
    \]

- Depends on \( t \) for a history dependent policy!
Markovian Policy and State Value Functions

State Value Functions

- Return expectation for a Markovian policy $\Pi$ starting from $s$ at time $t$.
  - Finite horizon setting (with time extended state space):
    \[
    v^T_{t,\Pi}(s) = \mathbb{E}_\Pi[G_t^T | S_t = s] = \sum_{t'=t+1}^{T} \mathbb{E}_\Pi[R_{t'} | S_t = s]
    \]
  - Episodic setting:
    \[
    v_{t,\Pi}(s) = \mathbb{E}_\Pi[G_t | S_t = s] = \sum_{t'=t+1}^{\infty} \mathbb{E}_\Pi[R_{t'} | S_t = s]
    \]
  - Discounted:
    \[
    v^\gamma_{t,\Pi}(s) = \mathbb{E}_\Pi[G_t^\gamma | S_t = s] = \sum_{t'=t+1}^{\infty} \gamma^{t'-(t+1)} \mathbb{E}_\Pi[R_{t'} | S_t = s]
    \]
  - Average return setting:
    \[
    \bar{v}_{t,\Pi}(s) = \mathbb{E}_\Pi[\bar{G}_t | S_t = s] = \lim_{T \to \infty} \frac{1}{T} \sum_{t'=t+1}^{t+T} \mathbb{E}_\Pi[R_{t'} | S_t = s]
    \]

- Becomes independent on $t$ if the policy is stationary and Markovian the generic case (except in the finite horizon setting).
State-Action Value Functions

State Value Functions

- Return expectation for a policy $\Pi$ starting from $s$ and an action $a$ at time $t$.
  - Finite horizon setting:
    $$ q^T_{t,\Pi}(s, a) = \mathbb{E}_{\Pi}[G^T_t \mid S_t = s, A_t = a] = \sum_{t'=t+1}^T \mathbb{E}_{\Pi}[R_{t'} \mid S_t = s, A_t = a] $$
  - Episodic setting:
    $$ q_t,\Pi(s, a) = \mathbb{E}_{\Pi}[G_t \mid S_t = s, A_t = a] = \sum_{t'=t+1}^\infty \mathbb{E}_{\Pi}[R_{t'} \mid S_t = s, A_t = a] $$
  - Discounted:
    $$ q^\gamma_{t,\Pi}(s, a) = \mathbb{E}_{\Pi}[G^\gamma_t \mid S_t = s, A_t = a] = \sum_{t'=t+1}^\infty \gamma^{t'-(t+1)} \mathbb{E}_{\Pi}[R_{t'} \mid S_t = s, A_t = a] $$
  - Average return setting:
    $$ \bar{q}_{t,\Pi}(s, a) = \mathbb{E}_{\Pi}[\bar{G}_t \mid S_t = s, A_t = a] = \lim_{T \to \infty} \frac{1}{T} \sum_{t'=t+1}^{t+T} \mathbb{E}_{\Pi}[R_{t'} \mid S_t = s, A_t = a] $$

- Different strategy for action at time $t$ than after...
- Independent of $t$ for a Markovian policy except for the finite horizon setting!
State Value Function vs State-Action Value Functions

State vs State-Action

- Performance measure of a policy $\Pi$:
  - starting from a state $s$ for the state value function,
  - starting from a state $s$ and an action $a$ (not necessarily related to $\Pi$) for the state-action value function.

- State value function at time $t$ from state-action value function:
  \[ v_{t,\Pi}(s) = \sum_a \Pi_t(a) q_t(s, a) \]
Do We Really Need The History Dependent Policies?

Equivalent Markovian policy in terms of value function

- **Thm:** For any policy $\Pi$ and any initial distribution $P_0(S_0)$, it exists a Markovian policy $\tilde{\Pi}$ such that

\[
\forall t, \forall s, \nu_{t,\Pi}(s) = \nu_{t,\tilde{\Pi}}(s).
\]

- Relies on the Markovian environment.
- Possible choice:

\[
\tilde{\pi}_t \{ A_t = a_t | S_t = s_t \} = \mathbb{E}_{P, P_0}[\pi_t(A_t = a_t | S_t = s_t, H_t) | S_t = s_t, S_0]
\]

- **No need to consider non Markovian policy** if the goal is entirely defined in terms of value functions.
Outline

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
Goals

**Prediction**

- What is the performance of a given policy?

**Planning**

- What is the *best* policy?

- Planning is harder than Prediction.
Prediction

- What is the performance of a given policy?
- Compute/Approximate/Estimate
  \[ v_{t,\Pi}(s) = \mathbb{E}_\Pi[G_t | S_t = s] \]
- Well defined provided the expectation exists.
Planning

What is the \textit{best} policy?

A possible definition: \[ \arg\max_{\pi} \sum_{s,t} \mu(s, t) v_{t, \pi}(s) \]

Not necessarily well defined...\]

Several choices for \( \mu \)!

More realistic goal: find a \textit{good} policy...
Sequential Decisions, MDP and Policies
1. Decision Process and Markov Decision Process
2. Returns and Value Functions
3. Prediction and Planning
4. Operations Research and Reinforcement Learning
   - Control
   - Survey

Operations Research: Prediction and Planning
1. Prediction and Bellman Equation
2. Prediction by Dynamic Programming and Contraction
3. Planning, Optimal Policies and Bellman Equation
4. Linear Programming
5. Planning by Value Iteration
6. Planning by Policy Iteration

Reinforcement Learning: Prediction and Planning in the Tabular Setting
1. Prediction with Monte Carlo
2. Planning with Monte Carlo
3. Prediction with Temporal Differences
4. Link with Stochastic Approximation
5. Planning with Value Iteration
6. Planning with Policy Improvement
7. Exploration vs Exploitation

Reinforcement Learning: Advanced Techniques in the Tabular Setting
1. n-step Algorithms
2. Eligibility Traces
3. Off-policy vs on-policy
4. Bandits
5. Model Based Approach

Reinforcement Learning: Approximation of the Value Functions
1. Approximation Target(s)
2. Gradient and Pseudo-Gradient
3. Linear Approximation and LSTD
4. On-Policy Prediction and Control
5. Off-Policy and Deadly Triad
6. Two-Scales Algorithms
7. Deep Q Learning
8. Continuous Actions

Reinforcement Learning: Policy Approach
1. Policy Gradient Theorems
2. Monte Carlo Based Policy Gradient
3. Actor / Critic Principle
4. 3 SOTA Algorithms
5. Average Return

References
What Do We Know?

Model
- Able to use the MDP transition probabilities.
- Probability world.

Only Observations
- No access to the MDP transition probabilities.
- Reinforcement Learning.
- Statistic world.

- Reinforcement Learning is harder than Markov Decision Process / Operations Research.
MDP / OR

- Stochastic setting in which the world is known.
- MDP model assumption.
- Probability world / Idealized setting...
- Lots of insight for the RL problem.
Reinforcement Learning

- Stochastic setting in which the world is observed through interactions.
- Still MDP model assumption.
- More realistic setting?
- More difficult theoretical analysis.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differentials
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
# MDP vs Discrete Control

## MDP
- State $s$ and action $a$
- Dynamic model: $\mathbb{P}(s'|s, a)$
- Reward $r$ defined by $\mathbb{P}(r|s', s, a)$.
- Policy $\Pi$: $a_t = \pi_t(S_t, H_t)$
- Goal: $\max \mathbb{E}_\Pi \left[ \sum_t R_t \right]$  

## Discrete Control
- State $x$ and control $u$
- Dynamic model: $x' = f(x, u, W)$ with $W$ a stochastic perturbation.
- Cost: $C(x, u, W)$.
- Control strategy $U$: $u_t = u(x_t, H_t)$
- Goal: $\min_U \mathbb{E}_U \left[ \sum_t C(x_t, u_t, W_t) \right]$  

- Almost the same setting but with a different vocabulary!
1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
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
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 interactive methods using value functions (dynamic programming).
- Policy deduced by a statewise optimization of the value function over the actions.
- Most results for 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.
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.
Could We Directly Parameterized the Policy?

- How to parameterize a policy?
- How to optimize this policy?
- Can we combine parametric policy and approximated value function?

- State Of The Art Algorithms (DPG, PPO, SAC...)
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3. Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

4. Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

5. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References

Replay Buffer and Prioritized Sweeping
Real Time Planning
Reinforcement Learning: Approximation of the Value Functions
Approximation Target(s)
Gradient and Pseudo-Gradient
Linear Approximation and LSTD
On-Policy Prediction and Control
Off-Policy and Deadly Triad
Two-Scales Algorithms
Deep Q Learning
Continuous Actions
Reinforcement Learning: Approximation of the Value Functions
"
Known MDP model

Focus on the finite horizon setting

\[ G^T_t = \sum_{t'=t+1}^{T} R_{t'} \]

and the discounted setting:

\[ G^\gamma_t = \sum_{t'=t+1}^{\infty} \gamma^{t'-(t+1)} R_{t'} \]

We will later consider the other settings.
Finite horizon: emphasis on Markovian policies

\[ \Pi_t(A_t = a_t) = \pi_t(A_t = a_t | S_t = s_t) = \pi_t(a_t | s_t) \]

Discounted return: emphasis on stationary Markovian policies

\[ \Pi_t(A_t = a_t) = \pi(A_t = a_t | S_t = s_t) = \pi(a_t | s_t) \]
Prediction

How to efficiently evaluate the quality of a policy

\[ v_{t, \Pi}(s) = \mathbb{E}_\Pi \left[ \sum_{t'=t+1}^{T} \gamma^{t'-(t+1)} R_{t'} \mid S_t = s \right] \]

when we can ensure that the sum is finite?

- \( v_{t, \Pi} \) independent of \( t \) in the discounted setting if the policy is stationary.
How to find a policy $\pi$ such that

$$\sum_{s,t} \mu(s, t) v_{t, \Pi}(s)$$

is as large as possible?

- Emphasis on $\mu(s, t) = 0$ if $t \neq 0$ and $\mu(s, 0) = P_0(S_0 = s_0)$. 
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3. Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

4. Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differencing
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

5. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Bellman Equation

\[ v_{t, \pi}(s) = \sum_a \pi_t(a|s) \sum_{s', r} p(s', r|s, a) \left( r + \gamma v_{t+1, \pi}(s') \right) \]

\[ = \sum_a \pi_t(a|s) r(s, a) + \gamma \sum_{s'} \sum_a p(s'|s, a) \pi_t(a|s) v_{t+1, \pi}(s') \]

Bellman Equation

- Link between \( v_{t, \pi} \) and \( v_{t+1, \pi} \).
- Straightforward consequence of

\[ G_t = \sum_{t'=t+1}^T \gamma^{t'-(t+1)} R_{t'} = R_{t+1} + \gamma \sum_{t'=t+2}^T \gamma^{t'-(t+2)} R_{t'} = R_{t+1} + \gamma G_{t+1} \]

and thus

\[ \mathbb{E}[G_t|S_t = s] = \mathbb{E}[R_{t+1}|S_t = s] + \gamma \mathbb{E}[\mathbb{E}[G_{t+1}|S_{t+1}]|S_t = s] \]
Bellman Operator

\[ \mathcal{T}^{\pi_t} : \mathbb{R}^{|S|} \rightarrow \mathbb{R}^{|S|} \]

\[ \mathcal{T}^{\pi_t} v(s) = \sum_a \pi_t(a|s) r(s, a) + \gamma \sum_{s'} p(s'|s, a) \sum_a \pi_t(a|s) v(s') \]

- Affine operator from the space of state value functions to the space of state value functions.
- By construction,

\[ v_{t, \Pi} = \mathcal{T}^{\pi_t} v_{t+1, \Pi} \]

- \( r_{\pi_t} \) is the vector of average immediate rewards using policy \( \pi_t \) while \( P^{\pi_t} \) is the one step state transition matrix using policy \( \pi_t \).
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differencing
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Finite Horizon: Naive Approach

\[ v_{t, \Pi}(s) = \sum_{a_t, r_{t+1}, s_{t+1}, \ldots, r_T} \left( \sum_{t'=t+1}^{T} r_{t'} \right) \mathbb{P}_{\Pi}(A_t = a_t, \ldots, R_T = r_T | S_t = s) \]

\[ = \sum_{a_t, r_{t+1}, s_{t+1}, \ldots, r_T} \left( \sum_{t'=t+1}^{T} r_{t'} \right) \pi_t(a_t | s) \times \cdots \times p(s_T, r_T | s_{T-1}, a_{T-1}) \]

- Exhaustive exploration of the trajectories.
- Complexity of order \((|A| \times |S| \times |R|)^{T-t}\) for the value function at time \(t\).

- Complexity can be reduced to \((|A| \times |S|)^{T-t}\) by noticing that

\[ v_{t, \Pi}(s) = \sum_{a_t, s_{t+1}, \ldots, s_{T-1}, a_{T-1}} \left( \sum_{t'=t+1}^{T} r(s_t, a_t) \right) \pi_t(a_t | s) \times \cdots \times p(s_T | s_{T-1}, a_{T-1}) \]
Finite Horizon: Recursive Prediction

\[ v_{T,\Pi}^T = 0 \]
\[ v_{t-1,\Pi}^T = \mathcal{T}^{\pi_{t-1}} v_{t,\Pi}^T \]

- After time \( T \), the finite horizon return \( G_t^T = 0 \) hence \( v_{T,\Pi}^T = 0 \) whatever the policy.
- The Bellman equation yields second equation.
- Equivalent rewriting
  \[ v_{t-1,\Pi}(s) = r_{\pi_{t-1}}(s) + \sum_{s'} P_{\pi_{t-1}}(s, s') v_{t,\Pi}^T \]
- Complexity of order only \( T \times |S|^2(|\mathcal{A}| + |S|) \) to compute all the value functions.
Finite Horizon: Prediction by Value Iteration

**input:** MDP model $\langle (S, A, R), P \rangle$ and policy $\Pi$

**parameter:** Horizon $T$

**init:** $v_T^T(s) = 0 \forall s \in S, t = T$

**repeat**

1. $t \leftarrow t - 1$
2. for $\forall s \in S$ do
   3. $v_t^T(s) \leftarrow \sum_{a \in A} \pi_t(a|s) \left( r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) v_{t+1}^T(s') \right)$
   4. end

until $t = 0$

**output:** Value functions $v_t^T$

- Most classical formulation
Discounted: Naive Approach

\[ v_t^{\gamma, \Pi}(s) = \sum_{t'=t+1}^{\infty} \gamma^{t'-(t+1)} \mathbb{E}_{\Pi}[R_{t'}|S_t = s] \simeq \sum_{t'=t+1}^{T} \gamma^{t} \mathbb{E}_{\Pi}[R_{t'}|S_t = s] = v_t^{\gamma, T}(s) \]

\[ v_t^{\gamma, T}(s) = \sum_{a_t, s_{t+1}, \cdots, s_{t-1}, a_{t-1}} \left( \sum_{t'=t+1}^{T} \gamma^{t'-(t+1)} r(s_t, a_t) \right) \pi_t(a_t|s) \times \cdots \times p(s_T|s_{t-1}, a_{t-1}) \]

Naive approach

- Exhaustive exploration of truncated trajectories.
- Back to the finite horizon setting...

**Prop:** Control on the error as \[ |v^{\gamma, \Pi} - v_t^{\gamma, T}|_{\infty} \leq \frac{\gamma^{T+1-t}}{1-\gamma} \max_{r \in \mathcal{R}} |r| \]

- Relation between the error \( \epsilon \simeq \gamma^{T-t} \) and the numerical complexity \( C = (|A| \times |S|)^{T-t} \) of order \( C \simeq \epsilon^{-1} \).
Discounted: Recursive Prediction with Naive Initialization

\[ v_{T, \Pi}^\gamma \simeq v_{T', \Pi}^\gamma = \tilde{v}_{T, \Pi} \]
\[ v_{t-1, \Pi} = \mathcal{T}^{\pi_{t-1}} v_{t, \Pi}^\gamma \simeq \tilde{v}_{t-1, \Pi} = \mathcal{T}^{\pi_{t-1}} \tilde{v}_{t, \Pi} \]

**Recursive Prediction**

- Requires an initialization at time \( T \) with a horizon \( T' \).
- The Bellman equation yields the second equation.
- Complexity of order only \( T \times |S|^2(|A| + |S|) \) to compute all the value functions after the initialization of cost \( (|A| \times |S|)^{T'-T} \).

**Prop:** If the approximation error between \( v_{T, \Pi}^\gamma \) and \( v_{T', \Pi}^\gamma \) is bounded by \( \epsilon \) then

\[ \| v_{t, \Pi}^\gamma - \tilde{v}_{t, \Pi} \|_\infty \leq \gamma^{T-t} \epsilon, \quad \forall t \leq T \]
Discounted and stationary: Bellman Equation

\[ v_\pi = \mathcal{T}_\pi v_\pi \]

\[ v_\pi(s) = \sum_a \pi(a|s)r(s, a) + \gamma \sum_{s'} \sum_a p(s'|s, a)\pi(a|s)v_\pi(s') \]

**Bellman Equation**

- Time independent value function \( v_\pi \).
- **Prop:** Unique solution of the linear equation \( v_\pi = \mathcal{T}_\pi v_\pi \)
- Complexity of order \((|A| + |S|) \times |S|^2 \) to obtain the solution.
Discounted and stationary: Recursive Implementation

\[ v_\Pi = \mathcal{T}^\pi v_\Pi \]

\[ v_{k+1} = \mathcal{T}^\pi v_k \quad \text{with arbitrary } v_0 \]

**Bellman Iteration**

- **Prop:** Unique fixed point of the Bellman operator \( v \mapsto \mathcal{T}^\pi v \).
- **Prop:** The iterates \( v_{k+1} = \mathcal{T}^\pi v_k \) converges toward \( v_\Pi \) and
  \[ \| v_k - v_\Pi \|_\infty \leq \gamma^k \| v_0 - v_\Pi \|_\infty \]
- Complexity of order \((k + |A||S|^2)\) to obtain the \( k \)th iterate.
- Exponential decay of the error with respect to the complexity.
Bellman Operator and Contraction

\[ \| T^\pi v - T^\pi v' \|_\infty \leq \gamma \| v - v' \|_\infty \]

**Proof**

- By definition

\[ \| T^\pi v - T^\pi v' \|_\infty = \gamma \| P^\pi (v - v') \|_\infty \]

- It suffices then to notice that \( P^\pi \) is a transition matrix, so that

\[ \sum_j P^\pi_{i,j} = 1 \]

and thus

\[ \left| \sum_j P^\pi_{i,j} z_j \right| \leq \max |z_j| \]

**Consequences**

- Unicity of the solution of \( T^\pi v = v \).
- Linear decay \( \gamma^k \) of the error with the iterates.
Bellman Operator and Bellman Equation Solution

\[ v_\Pi = \left( \sum_{k=0}^{\infty} \gamma^k (P^\pi)^k \right) r_\pi \]

A Closed Formula for the State Value Function

- \[ v_\Pi = T^\pi v_\Pi \Leftrightarrow (I - \gamma P^\pi) v_\Pi = r_\pi \]
- As \( P^\pi \) is a transition matrix, its eigenvalues are smaller than 1 and thus \( (I - \gamma P^\pi) \) is invertible of inverse

\[ (I - \gamma P^\pi)^{-1} = \sum_{k=0}^{\infty} \gamma^k (P^\pi)^k \]

- Could have been obtained without the Bellman equation as the \( (P^\pi)^k \) is, by construction, the probability of being at state \( s' \) at time \( k \) starting from \( s \) at time 0 and following \( \Pi \).
Discounted and stationary: Value Iteration

Discounted: Prediction by Value Iteration

input: MDP model \( \langle (S, A, R), P \rangle \), discount factor \( \gamma \), and stationary policy \( \pi \)
init: \( \tilde{v}(s) \forall s \in S \)
repeat
\[
\tilde{v}_{\text{prev}} \leftarrow \tilde{v}
\]
for \( s \in S \) do
\[
\tilde{v}(s) \leftarrow \sum_{a \in A} \pi(a|s) \left( r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) \tilde{v}_{\text{prev}}(s') \right)
\]
end
output: Value function \( \tilde{v} \)

- When to stop?
Discounted and stationary: Value Iteration

Discounted: Prediction by Value Iteration

**input:** MDP model \(((S, A, R), P)\), discount factor \(\gamma\), and stationary policy \(\pi\)

**parameter:** \(\delta > 0\) as accuracy termination threshold

**init:** \(\tilde{v}(s) \forall s \in S\)

**repeat**

\[\tilde{v}_{\text{prev}} \leftarrow \tilde{v}\]

\[\Delta \leftarrow 0\]

**for** \(s \in S\) **do**

\[\tilde{v}(s) \leftarrow \sum_{a \in A} \pi(a|s) \left( r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) \tilde{v}_{\text{prev}}(s') \right)\]

\[\Delta \leftarrow \max(\Delta, |\tilde{v}(s) - \tilde{v}_{\text{prev}}(s)|)\]

**end**

**until** \(\Delta < \delta\)

**output:** Value function \(\tilde{v}\)

- **Prop:** when the algorithms stops

\[\|\tilde{v} - v_\Pi\|_\infty \leq \frac{2\delta}{1 - \gamma}\]
Discounted and stationary: Value Iteration

Discounted: Prediction by Value Iteration - Gauss-Seidel Version

**input:** MDP model $\langle (S, A, R), P \rangle$, discount factor $\gamma$, and stationary policy $\pi$

**parameter:** $\delta > 0$ as accuracy termination threshold

**init:** $\tilde{v}(s) \forall s \in S$

repeat

1. $\Delta \leftarrow 0$
2. for $s \in S$ do
   - $\tilde{v}_{prev} \leftarrow \tilde{v}(s)$
   - $\tilde{v}(s) \leftarrow \sum_{a \in A} \pi(a|s) \left( r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a)\tilde{v}(s') \right)$
   - $\Delta \leftarrow \max(\Delta, |\tilde{v}(s) - \tilde{v}_{prev}|)$
3. end

until $\Delta < \delta$

**output:** Value function $\tilde{v}$

- Gauss-Seidel variation mostly used in practice.
- No need to store the previous value function.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - Off-Policy Prediction and Control
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
An optimal policy $\Pi^\star$ should be better than any other policies:

$$\forall s, \forall t, v_{t, \Pi^\star}(s) = \sup_{\pi} v_{t, \Pi}(s)$$

### Several Questions

- Do this policy exist?
- Is it unique?
- How to characterize it?
- How to obtain it?

Even the $\sup$ above could be an issue if it is not attained!
Finite Horizon and Optimal Policy

Explicit Recursive Solution

- After horizon $T$, any policy leads to a 0 return.
- At time $T - 1$,
  - the total return $G_T$ is the immediate return at time $T$ and thus
    \[
    v_{T, \pi^*}(s) = \sup_{\pi(a|s)} \sum_a \pi(a|s) r(a, s) = \sup_a r(a, s)
    \]
  - the optimal policy $\pi^*_{T-1}$ exists and is deterministic.
- By recursion,
  - the total return at time $t - 1$ is the immediate return at time $t$ plus the total return at time $t - 1$ and thus
    \[
    v_{t-1, \pi^*}(s) = \sup_{\pi(a|s)} \sum_a \pi(a|s) \left( r(a, s) + \sum_{s'} p(s'|s, a) v_{t, \pi^*} \right)
    \]
    \[
    = \sup_a \left( r(a, s) + \sum_{s'} p(s'|s, a) v_{t, \pi^*} \right)
    \]
  - the optimal policy $\pi^*_{t-1}$ exists and is deterministic.
Discounted Setting and Optimal Stationary Policy

Heuristic

- Optimal policy: $v^\Pi^*(s) = \sup_{\pi} v^\Pi(s)$
- Stationary solution:
  $v^\Pi^*(s) = \sup_{\pi} (T^\pi v^\Pi^*) (s)$

$$ = \sup_{\pi_t(\cdots | s)} \sum_a \pi(a | s) \left( r(a, s) + \sum_{s'} p(s' | s, a) v^\Pi^*(s') \right)$$

$$ = \sup_a \left( r(a, s) + \sum_{s'} p(s' | s, a) v^\Pi^*(s') \right)$$

- Optimal deterministic policy: $\pi^*(s) \in \arg\max (r(a, s) + \sum_{s'} p(s' | s, a) v^\Pi^*(s'))$.

Is everything well defined? Yes but one has to be more cautious!
Optimal Value Function and Bellman Operator

Optimal Value Function

- Optimal value function: $v_*(s) = \sup_\Pi v_\Pi(s)$
- Defined state by state so that it is not necessarily attained by a single $\Pi^*$

Optimal Bellman operator

- Similar to the Bellman operator but do not depend on a policy:
  \[
  \mathcal{T}^* v(s) = \sup_a \left( r(a, s) + \sum_{s'} p(s'|s, a) v(s') \right)
  \]

Link between the two

- $v \geq \mathcal{T}^* v$ implies $v \geq v_*$.
- $v \leq \mathcal{T}^* v$ implies $v \leq v_*$. 

Discounted
Prop: $\mathcal{T}^*$ is a $\gamma$-contraction for the sup-norm and thus it exists a unique $v_{\star\star}$ such that $v_{\star\star} = \mathcal{T}^* v_{\star\star}$.

Fixed Point and Optimal Value Function

Prop: $v_{\star} = v_{\star\star}$ and is thus the unique fixed point of $\mathcal{T}^*$.

Proof: $v_{\star\star} = \mathcal{T}^* v_{\star\star}$ and thus $v_{\star\star} = v_{\star}$ according the link between the optimal value function and the Bellman operator.

Does this mean something about policies?
Optimal Policy and Bellman Operator

**Bellman Operator and Policy**

**Prop:** For any \( v \), any policy \( \pi_v \) satisfying

\[
\pi_v(s) \in \operatorname{argmax}_a \left( r(a, s) + \sum_{s'} p(s'|s, a) v(s') \right)
\]

is such that \( \mathcal{T}^* v(s) = \sup_\pi \mathcal{T}^\pi v(s) = \mathcal{T}^{\pi_v} v(s) \)

---

**Bellman Operator and Optimal Policy**

**Prop:** Any stationary policy \( \pi_* \) satisfying

\[
\pi_*(s) \in \operatorname{argmax}_a \left( r(a, s) + \sum_{s'} p(s'|s, a) v^*(s') \right)
\]

is optimal.

**Proof:** Indeed by construction, \( \mathcal{T}^* v_* = \mathcal{T}^{\pi_*} v_* \) and thus, as \( \mathcal{T}^* v_* = v_* \), \( v_{\pi_*} = v_* \).
**Summary**

- It exists a unique $v_\star$ such that $T_\star v_\star = v_\star$
- $\forall s$, $v_\star(s) = \sup_\pi v_\pi(s)$
- Any policy $\pi_\star$ satisfying:

$$
\forall s, \pi_\star(s) \in \arg\max_a \left( r(a, s) + \sum_{s'} p(s'|s, a) v_\star(s') \right)
$$

is optimal as $\forall s$, $v_{\pi_\star}(s) = v_\star(s) = \sup_\pi v_\pi(s)$

- Existence result but not (yet) a constructive algorithm!
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Explicit Resolution of the Equations?

- **Prediction:**
  - Simple linear system for $v_\pi$.
  - Already mentioned before...
  - Complexity of order $(|A| + |S|)|S|^2$.
- **Planning:**
  - More complex linear programming system for $v_\star$ due to the max operator.
  - Optimal policy easily deduced from $v_\star$.
  - Complexity of order $(|A||S|)^3$. 

\[
\begin{align*}
  v_\pi &= \mathcal{T}_\pi v_\pi \\
  v_\star &= \mathcal{T}_\star v_\star
\end{align*}
\]
Linear Programming

From $\forall s, \nu(s) = \sup_a r(s, a) + \gamma \sum_{s'} p(s'|s, a) \nu(s')$

to $\min \sum_s \mu(s) \nu(s)$

such that $\forall (s, a), \nu(s) \geq r(s, a) + \gamma \sum_{s'} p(s'|s, a) \nu(s')$

Different formulations but same solution

- Using $\nu \geq T^* \nu \iff \nu \geq \nu_*$, the condition implies $\nu \geq \nu_*$
- Now for any $\mu$ satisfying $\mu(s) > 0$, $\sum_s \mu(s) \nu(s) \geq \sum_s \mu(s) \nu_*(s)$ as soon as the condition is satisfied, hence $\nu_*$ is a solution.
- If for any state $\nu(s) > \nu_*(s)$ then $\sum_s \mu(s) \nu(s) > \sum_s \mu(s) \nu_*(s)$ and thus $\nu_*$ is the unique minimizer.
Primal Problem

Primal: \( \min_{\nu} \sum_{s} \mu(s)\nu(s) \)

such that \( \forall(s, a), \nu(s) \geq r(s, a) + \gamma \sum_{s'} p(s'|s, a)\nu(s') \)

Some properties

- Can be solved with a linear programming solver.
- Unicity of solution (and thus independence with respect to \( \mu \)) can be proved without using \( \nu_* \).

**Proof:** let \( \nu_1 \) a solution for \( \mu_1 \) and \( \nu_2 \) a solution for \( \mu_2 \) then \( \min(\nu_1, \nu_2) \) satifies the constraints. Furthermore if exists \( \nu_2(s) < \nu_1(s) \) then \( \min(\nu_1, \nu_2) \) is a strictly better solution for \( \mu_2 \) which is impossible.
Dual Problem

Primal: \( \min_v \sum_s \mu(s) v(s) \)

such that \( \forall (s, a), v(s) \geq r(s, a) + \gamma \sum_{s'} p(s'|s, a)v(s') \)

Dual: \( \max_{\lambda(s, a) \geq 0} \sum_s \sum_a \lambda(s, a)r(s, a) \)

such that \( \forall s, \sum_a \lambda(s, a) = \mu(s) + \gamma \sum_{s',a} p(s'|s', a)\lambda(s', a) \)

Derivation

- Usual derivation through the Lagrangian:

\[
\mathcal{L}(v, \lambda) = \sum_s \mu(s)v(s) + \sum_{s,a} \lambda(s,a) \left( r(s,a) + \gamma \sum_{s',a} p(s'|s', a)v(s') - v(s) \right)
\]

- Strong duality as Slater condition holds when \( \gamma < 1 \) with \( v = \frac{1+\epsilon}{1-\gamma} \max_{s,a} r(s, a) \).
**Dual and Interpretation**

**Dual:**
\[
\max_{\lambda(s,a) \geq 0} \sum_{s,a} \lambda(s,a)r(s,a)
\]

such that \(\forall s, \sum_a \lambda(s,a) = \mu(s) + \gamma \sum_{s',a} p(s'|s,a)\lambda(s',a)\)

**Interpretation:**
\[
\max_{\pi} \sum_{k=0}^{\infty} \gamma^k \sum_{s,a} \mathbb{P}(S_t = a, A_t = a|S_0 \sim \mu, \pi) r(s,a)
\]

**Interpretation in terms of policy**

- For any feasible \(\lambda\), define \(u(s) = \sum_a \lambda(s,a)\) and the policy \(\pi(a|s) = \lambda(s,a)/u(s)\).
- **Prop:** \(u = (\text{Id} - \gamma P^\pi)\mu = \sum_{k=0}^{\infty} \gamma^k (P^\pi)^k \mu\).
- **Prop:** \(\lambda(s,a) = \pi(a|s)u(s) = \sum_{k=0}^{\infty} \gamma^k \mathbb{P}(S_t = a, A_t = a|S_0 \sim \mu, \pi)\)
- Conversely for any \(\pi\) they is a feasible \(\lambda\).
- Any optimal \(\lambda^*\) (and thus policy) satisfies \(\lambda^*(s,a) = 0\) if \(v^*(s) > r(s,a) + \gamma \sum_{s'} p(s'|s,a) v^*(s')\) (optimal policy support)
Outline

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

4 Reinforcement Learning: Advanced Techniques
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
Finite Horizon: Planning by Value Iteration

input: MDP model $\langle (S, A, R), P \rangle$

parameter: Horizon $T$

init: $v_T^T(s) = 0 \forall s \in S, t = T$

repeat

\[
\begin{align*}
    t & \leftarrow t - 1 \\
    \text{for } s \in S \text{ do} & \\
    \quad v_t^T(s) & \leftarrow \max_{a \in A} \left( r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) v_{t+1}^T(s') \right)
\end{align*}
\]

end

until $t = 0$

output: Deterministic policy $\pi_t(s) \in \arg\max_{a \in A} \left( r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) v_{t+1}^T(s') \right)$

- Algorithm used to prove the existence of an optimal policy.
- No necessarily unique as argmax may not be unique.
Optimal Value Function, Fixed Point and Contraction

\[ v_* = \mathcal{T}^* v_* \quad \text{and} \quad \| \mathcal{T}^* v - \mathcal{T}^* v' \|_\infty \leq \gamma \| v - v' \|_\infty \]

\[ \implies v_{k+1} = \mathcal{T}^* v_k \to v_* \]

Bellman Operator

- Properties of Optimal Bellman Operator:
  - \( v_* \) is a fixed point of \( \mathcal{T}^* \).
  - \( \mathcal{T}^* \) is a \( \gamma \)-contraction for the \( \| \cdot \|_\infty \) norm.
- Classical fixed point theorem setting.
- Practical algorithm to approximate \( v_* \).
Value Iteration Algorithm

Discounted: Value Iteration Planning

**input:** MDP model $\langle (S,A,R), P \rangle$, and discount factor $\gamma$

**parameter:** $\delta > 0$ as accuracy termination threshold

**init:** $v(s) \forall s \in S$

**repeat**

\[
\begin{align*}
\tilde{v}_{\text{prev}} &\leftarrow \tilde{v} \\
\Delta &\leftarrow 0 \\
\text{for } s \in S \text{ do} \\
&\quad \tilde{v}(s) \leftarrow \max_{a \in A} r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a)\tilde{v}_{\text{prev}}(s') \\
&\quad \Delta \leftarrow \max (\Delta, |\tilde{v}(s) - \tilde{v}_{\text{prev}}(s)|) \\
\text{end}
\end{align*}
\]

**until** $\Delta < \delta$

**output:** Value function $\tilde{v}$

- Same convergence criterion (and similar proof) than in the planning case.
- Which policy?
Value Iteration Algorithm

Discounted: Value Iteration Planning

**input:** MDP model \( \langle (S, A, R), P \rangle \), and discount factor \( \gamma \)

**parameter:** \( \delta > 0 \) as accuracy termination threshold

**init:** \( \tilde{v}(s) \forall s \in S \)

**repeat**

\[
\tilde{v}_{\text{prev}} \leftarrow \tilde{v} \\
\Delta \leftarrow 0 \\
\text{for } s \in S \text{ do} \\
\quad \tilde{v}(s) \leftarrow \max_{a \in A} r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a)\tilde{v}_{\text{prev}}(s') \\
\quad \Delta \leftarrow \max (\Delta, |\tilde{v}(s) - \tilde{v}_{\text{prev}}(s)|) \\
\text{end}
\]

**until** \( \Delta < \delta \)

**output:** Deterministic policy \( \tilde{\pi}(s) \in \arg\max r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a)\tilde{v}(s') \)

• Natural idea: define a policy using the argmax of the existence proof.
• Do we have a convergence guarantee on the resulting policy?
\[ \hat{\pi}(s) \in \arg\max_a r(s, a) + \gamma \sum_{s'} p(s' | s, a) \tilde{v}(s') \]

\[ \implies \| \tilde{v}_{\hat{\pi}} - v_* \|_\infty \leq \frac{2\gamma}{1 - \gamma} \| \tilde{v} - v_* \|_\infty \]

**Value and argmax Policy**

- Bound on the loss of the final policy!
- Rely on the fact that, by construction, \( T_{\hat{\pi}} \tilde{v} = T^* \tilde{v} \)

**Proof:**

\[ \| \tilde{v}_{\hat{\pi}} - v_* \|_\infty = \| T_{\hat{\pi}} \tilde{v}_{\hat{\pi}} - T_{\hat{\pi}} \tilde{v} + T^* \tilde{v} - T^* v_* \|_\infty \]

\[ \leq \| T_{\hat{\pi}} \tilde{v}_{\hat{\pi}} - T_{\hat{\pi}} \tilde{v} \|_\infty + \| T^* \tilde{v} - T^* v_* \|_\infty \]

\[ \leq \gamma \| \tilde{v} - \tilde{v} \|_\infty + \gamma \| \tilde{v} - v_* \|_\infty \]

\[ \leq \gamma \| \tilde{v} - v_* \|_\infty + 2\gamma \| \tilde{v} - v_* \|_\infty \]
Value Iteration Algorithm

Discounted: Value Iteration Planning

**input:** MDP model \( \langle S, A, R, P \rangle \), and discount factor \( \gamma \)

**parameter:** \( \delta > 0 \) as accuracy termination threshold

**init:** \( \tilde{v}(s) \forall s \in S \)

**repeat**

\[
\begin{align*}
\tilde{v}_{\text{prev}} & \leftarrow \tilde{v} \\
\Delta & \leftarrow 0 \\
\text{for } s \in S & \text{ do} \\
\tilde{v}(s) & \leftarrow \max_{a \in A} r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) \tilde{v}_{\text{prev}}(s') \\
\Delta & \leftarrow \max (\Delta, |\tilde{v}(s) - \tilde{v}_{\text{prev}}(s)|) \\
\text{end}
\end{align*}
\]

**until** \( \Delta < \delta \)

**output:** Deterministic policy \( \tilde{\pi}(s) \in \arg\max_a r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) \tilde{v}(s') \)

**Prop:** \( \| v_{\tilde{\pi}} - v_* \|_\infty \leq \frac{4 \gamma \delta}{1 - \gamma} \)
From State Value to State-Action Value Functions

\[ v_\pi(s) = \mathbb{E}_\pi \left[ \sum_k \gamma^k R_t \mid S_0 = s \right] \]

\[ q_\pi(s, a) = \mathbb{E}_\pi \left[ \sum_k \gamma^k R_t \mid S_0 = s, A_0 = a \right] \]

\[ \mathcal{T}^\pi v(s) = \sum_a \pi(a | s) \left( r(s, a) + \sum_{s'} p(s' | s, a) v(s') \right) \]

\[ \mathcal{T}^\pi q(s, a) = r(s, a) + \sum_{s'} p(s' | s, a) \sum_a \pi(a | s') q(s', a) \]

\[ \mathcal{T}^* v(s) = \max_a r(s, a) + \sum_{s'} p(s' | s, a) v(s') \]

\[ \mathcal{T}^* q(s, a) = r(s, a) + \sum_{s'} p(s' | s, a) \max_a q(s', a) \]

Two equivalent point of view?

- Everything could have been defined using the state-action point of view.
- Knowing \( v_\pi \) is equivalent to knowing \( q_\pi \) as

\[ v_\pi(s) = \sum_a \pi(s | a) q_\pi(s, a) \quad \text{and} \quad q_\pi(s, a) = r(s, a) + \sum_{s'} p(s' | s, a) v_\pi(s'). \]
State-Action Bellman Operators

\[ T^\pi q(s, a) = r(s, a) + \sum_{s'} p(s'|s, a) \sum_a \pi(a|s') q(s', a) \]

\[ T^* q(s, a) = r(s, a) + \sum_{s'} p(s'|s, a) \max_a q(s', a) \]

**Properties**

- **Prop:** \( T^\pi \) and \( T^* \) are \( \gamma \) contractions for the \( \| \cdot \|_\infty \) norm.
- **Prop:** \( q_\pi \) is the unique solution of \( T^\pi q = q \)
- **Prop:** \( q_\star \) defined \( q_\star(s, a) = \sup_\pi q_\pi(s, a) \) is the unique solution of \( q = T^* q \) and is attained for any policy \( \pi_\star \) satisfying \( \pi_\star(s) \in \arg\max q_\star(s, a) \).
- **Prop:** Any such policy satisfies: \( v_{\pi_\star}(s) = q_{\pi_\star}(s, \pi_\star(s)) = v_\star(s) \).
State-Action Value Iteration Algorithm

Discounted: Planning by State-Action Value Iteration

| input: | MDP model \(\langle S, A, R, P \rangle\), and discount factor \(\gamma\) |
| parameter: | \(\delta > 0\) as accuracy termination threshold |
| init: | \(\tilde{q}(s, a) \forall (s, a) \in S \times A\) |
| repeat | \(\tilde{q}_{\text{prev}} \leftarrow \tilde{q}\) |
| | \(\Delta \leftarrow 0\) |
| for | \(s \in S\) do |
| for | \(a \in A\) do |
| | \(\tilde{q}(s, a) \leftarrow \left( r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a) \max_{a'} \tilde{q}_{\text{prev}}(s', a') \right) \) |
| | \(\Delta \leftarrow \max(\Delta, |\tilde{q}(s, a) - \tilde{q}_{\text{prev}}(s, a)|)\) |
| end | |
| end | until \(\Delta < \delta\) |
| output: | Deterministic policy \(\tilde{\pi}(s) \in \arg\max_a \tilde{q}(s, a)\) |

- Same complexity but more storage than with state value function. . .
- but will be useful later!
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Value Function vs Policy Point of View

\[ v, q \rightarrow \Pi \quad \text{or} \quad \Pi \rightarrow v, q? \]

### Planning

- Focus so far on value-function point of view!
- Heuristic: find a good approximation of the optimal value function and deduce a good policy.
- Can we work directly on the policy itself?

- For prediction, only the policy point of view makes sense!
∀s, π⁺(s) ∈ argmax_a q_π(s, a) ⇒ ∀v_{π⁺}(s) ≥ v_π(s)

**Classical Policy Improvement Lemma**

- **Prop:** Given a policy π and its q value-function, one can obtain a better policy with the argmax operator.
- **Prop:** If no improvement is possible, it means that π is already optimal.
- **Proof:** Use T^{π⁺}v_π = T^*v_π ≥ T^πv_π = v_π to prove (T^{π⁺})^k v_π ≥ v_π which implies the result by letting k goes to +∞.

- Leads to a sequential improvement algorithm...
Policy Improvement Lemma

\[
\mathbb{E}[v_{\pi'}(S_0)] - \mathbb{E}[v_{\pi}(S_0)] = \sum_{k=0}^{\infty} \gamma^k \mathbb{E}_{\pi'} \left[ \sum_{a} \pi'(a|S_t) \left( q_{\pi}(S_t, a) - v_{\pi}(S_t) \right) \right]
\]

\[
= \sum_{k=0}^{\infty} \gamma^k \mathbb{E}_{\pi'} \left[ \sum_{a} \left( \pi'(a|S_t) - \pi(a|S_t) \right) q_{\pi}(S_t, a) \right]
\]

A Generic Improvement Lemma

- No assumptions on \( \pi \) and \( \pi' \)!
- Easy proof.
- Imply the previous lemma as \( \max_a Q_\pi(s, a) - v_\pi(s) \geq 0 \).
- Show that improvement choices are possible.

- Will prove to be useful later...
Policy Iteration

Discounted: Planning by Policy Iteration

input: MDP model $\langle (S, A, R), P \rangle$, and discount factor $\gamma$
parameter: Initial policy $\tilde{\pi}$
repeat
  Compute $q_{\tilde{\pi}}$.
  for $s \in S$ do
    for $a \in A$ do
      $\tilde{\text{pol}}(s) \leftarrow \text{argmax} \ q_{\tilde{\pi}}(s, a)$
    end
  end
output: Deterministic policy $\tilde{\pi}$.

Some issues

- How to obtain $q_{\pi}$?
- When to stop?
## Discounted: Planning by Policy Iteration

**input:** MDP model $\langle (S, A, R), P \rangle$, and discount factor $\gamma$

**parameter:** Initial policy $\tilde{\pi}$

**repeat**

1. $stable \leftarrow 0$
2. Compute $q_{\tilde{\pi}}$.
3. **for** $s \in S$ **do**
   1. $old - action \leftarrow \tilde{\pi}(s)$
   2. $\tilde{\pi}(s) \leftarrow \text{argmax } q_{\tilde{\pi}}(s, a)$
4. **if** $\tilde{\pi}(s) \neq old - action$ **then**
   1. $stable \leftarrow 0$
5. **end**

**until** $stable == 1$

**output:** Deterministic policy $\tilde{\pi}$.

---

## Finite Setting

- Finite set of action-states implies a finite set of policy.
- Convergence of the algorithm in finite time!
Policy Iteration

Convergence Rate

- **Crude analysis:**
  - Bound after $k$ steps of the algorithm
    \[
    \|v_{\pi_k} - v_*\|_\infty \leq \gamma \|v_{\pi_{k-1}} - v_*\|_\infty \leq \gamma^k \|v_{\pi_0} - v_*\|_\infty
    \]
    \[
    \|v_{\pi_k} - v_*\|_\infty \leq \frac{\gamma}{1 - \gamma} \|v_{\pi_k} - v_{\pi_{k-1}}\|_\infty
    \]
  - Not much better than value iteration but much higher complexity as $q_{\pi_k}$ is obtained by solving the Bellman equation!

- **Much faster in practice...**
- **Clever analysis (Puterman):**
  - Under some mild assumptions and provided $\|P_{\pi_k} - P_*\| \leq K \|v_{\pi_k} - v_*\|_\infty$ then
    \[
    \|v_{\pi_k} - v_*\|_\infty \leq \frac{K \gamma}{1 - \gamma} \|v_{\pi_{k-1}} - v_*\|_\infty^2
    \]
  - May explain the better convergence in practice!
Outline

1 Sequential Decisions, MDP and Policies
   • Decision Process and Markov Decision Process
   • Returns and Value Functions
   • Prediction and Planning
   • Operations Research and Reinforcement Learning
   • Control
   • Survey

2 Operations Research: Prediction and Planning
   • Prediction and Bellman Equation
   • Prediction by Dynamic Programming and Contraction
   • Planning, Optimal Policies and Bellman Equation
   • Linear Programming
   • Planning by Value Iteration
   • Planning by Policy Iteration

3 Optimization Interpretation
   • Approximation and Stability
   • Generalized Policy Iteration
   • Infinite, Episodic and Average setting

4 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   • Prediction with Monte Carlo
   • Planning with Monte Carlo
   • Prediction with Temporal Differences
   • Link with Stochastic Approximation
   • Planning with Value Iteration
   • Planning with Policy Improvement
   • Exploration vs Exploitation

5 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   • n-step Algorithms
   • Eligibility Traces
   • Off-policy vs on-policy
   • Bandits
   • Model Based Approach

6 Reinforcement Learning: Approximation of the Value Functions
   • Approximation Target(s)
   • Gradient and Pseudo-Gradient
   • Linear Approximation and LSTD
   • On-Policy Prediction and Control
   • Off-Policy and Deadly Triad
   • Two-Scales Algorithms
   • Deep Q Learning
   • Continuous Actions

7 Reinforcement Learning: Policy Approach
   • Policy Gradient Theorems
   • Monte Carlo Based Policy Gradient
   • Actor / Critic Principle
   • 3 SOTA Algorithms
   • Average Return

8 References
Value Iteration: (Relaxed) First Order Method

Value Iteration

- **Iteration:**
  \[ v_k = T^* v_{k-1} \]
  \[ = v_{k-1} + (T^* - \text{Id}) v_{k-1} \]

- **Relaxation**
  \[ v_k = v_{k-1} - \alpha (\text{Id} - T^*) v_{k-1} \]
  can be proved to converge for any \( \alpha < \frac{2}{1+\gamma} \).
- Can be interpreted as a first order method with pseudo-gradient \((T^* - \text{Id}) v_{k-1}\).
- No function corresponding to this gradient!

- Is there a better choice for \( \alpha \) than \( \alpha = 1 \)?
- No as the resulting operator is a contraction of constant
  \[ |1 - \alpha| + \alpha \gamma \geq \gamma \]
Policy Iteration: Newton-Raphson Method

Policy Iteration

- **Explicit iteration:**

  Solve \( v_{\pi_{k-1}} = \mathcal{T}^{\pi_k} v_{\pi_{k-1}} \)

  Let \( \pi_k \) such that \( \mathcal{T}^{\pi_k} v_{\pi_{k-1}} = \mathcal{T}^* v_{\pi_{k-1}} \)

- **Implicit iteration on \( v_{\pi_k} \):**

  \[
  v_{\pi_k} = (\text{Id} - \gamma P^{\pi_k})^{-1} r_{\pi_k}
  \]

  \[
  = (\text{Id} - \gamma P^{\pi_k})^{-1} (r_{\pi_k} + (\gamma P^{\pi_k} - \text{Id}) v_{\pi_{k-1}} + (\text{Id} - \gamma P^{\pi_k}) v_{\pi_{k-1}})
  \]

  \[
  = v_{\pi_{k-1}} - (\text{Id} - \gamma P^{\pi_k})^{-1} (\text{Id} - \mathcal{T}^{\pi_k}) v_{\pi_{k-1}}
  \]

- Can be interpreted as a second order method with pseudo-gradient

  \[
  (\text{Id} - \mathcal{T}^{\pi_k}) v_{\pi_{k-1}} = (\text{Id} - \mathcal{T}^*) v_{\pi_{k-1}} \]

  and pseudo-Hessian \( (\text{Id} - \gamma P^{\pi_k}) \).

- Not a formal analysis but give a good insight on the better convergence of policy iteration.
Outline

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
Stability of Value and Policy Iteration

Ideal Value and Policy Iteration?

- Iterative algorithms.
- Convergence proofs assume perfect computation.
- What happens if we make a (small) error at each step?

- Particularly important for Policy Iteration in which one resolves a linear system at each step!
Value Iteration Stability

\[ v_k = T^* v_{k-1} + \epsilon_{k-1} \]

\[ \implies \| v_k - v_* \|_\infty \leq \gamma^k \| v_0 - v_* \|_\infty + \frac{\max_{0 \leq k' < k} \| \epsilon_{k'} \|_\infty}{1 - \gamma} \]

\[ \implies \| v_{\pi_k} - v_* \|_\infty \leq \frac{2\gamma^{k+1}}{1 - \gamma} \| v_0 - v_* \|_\infty + \frac{2\gamma \max_{0 \leq k' < k} \| \epsilon_{k'} \|_\infty}{(1 - \gamma)^2} \]

**Stability with respect to the error**

- Proof relies on the contraction property of \( T^* \) (hence similar results for \( T^\pi \)).
- Error term \( \frac{\max_{0 \leq k' < k} \| \epsilon_{k'} \|_\infty}{1 - \gamma} \) can be replaced by \( \sum_{k' = 0}^{k-1} \gamma^{k-k'} \| \epsilon_{k'} \|_\infty \)
- Convergence if \( \| \epsilon_k \|_\infty \) tends to 0.
- Remains in a neighborhood of the optimal solution if \( \| \epsilon_k \|_\infty \) is bounded.
Policy Iteration

\[ v_{k-1} = v_{\pi_{k-1}} + \epsilon_{k-1} \quad \text{and} \quad \mathcal{T}^{\pi_k} v_{k-1} = \mathcal{T}^* v_{k-1} \]

\[ \implies \| v_{\pi_k} - v_* \|_\infty \leq \gamma^k \| v_{\pi_0} - v_* \|_\infty + \frac{\gamma(2 - \gamma) \max_{0 \leq k' < k} \| \epsilon_{k'} \|_\infty}{(1 - \gamma)^2} \]

**Stability with respect to the error**

- Quite involved proof but crude results.
- Error term \( \max_{0 \leq k' < k} \| \epsilon_{k'} \|_\infty \) can be replaced by \( \sum_{k'=0}^{k-1} \gamma^{k-k'} \| \epsilon_{k'} \|_\infty \)
- Convergence if \( \| \epsilon_k \|_\infty \) tends to 0.
- Remains in a neighborhood of the optimal solution if \( \| \epsilon_k \|_\infty \) is bounded.
- Policy Iteration only requires an approximate estimate of \( v_{\pi_{k-1}} \), for instance obtained by Bellman iteration...
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

   - Infinite, Episodic and Average setting
   - Bellman Equation

6. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

7. Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

8. References
Modified Policy Iteration

Discounted: Planning by Generalized Policy Iteration

**input:** MDP model $\langle (S, A, \mathcal{R}), P \rangle$, and discount factor $\gamma$

**parameter:** Initial $q$

**repeat**

```
for $s \in S$ do
  $\tilde{\pi}(s) \leftarrow \text{argmax } q(s, a)$
end

repeat
  $q_{\text{prev}} \rightarrow q$
  for $(s, a) \in S \times A$ do
    $q(s, a) \leftarrow r(s, a) + \gamma \sum_{s, a'} p(s'|s, a) \tilde{\pi}(a'|s) q_{\text{prev}}(s, a')$
  end
```

**output:** Deterministic policy $\tilde{\pi}$.

- Algorithm driven by $q$.
- Flexibility in the number of prediction steps after each policy improvement steps.
- Special cases:
  - Large number: Policy Iteration with (small) error.
  - One: Value Iteration!
\[ T^{\pi_k} v_k = T^{*} v_k \quad \text{and} \quad v_{k+1} = (T^{\pi_k})^{m_k} v_k \]

\[ \implies \| v_{k+1} - v_* \|_\infty \leq \gamma \left( \frac{1 - \gamma^m_k}{1 - \gamma} \| P^{\pi_k} - P^{*} \| + \gamma^m_k \right) \| v_k - v_* \|_\infty \]

**Convergence Results**

- Quite technical proof.
- Valid only under the mild assumption \( T^{*} v_0 \geq v_0 \).
- Very fast decay provided \( \| P^{\pi_k} - P^{*} \| \) is small.

- No stability with arbitrary errors...
General Policy Iteration

- Two simultaneous interacting processes:
  - One forcing the policy to correspond to the current value function (Policy Improvement)
  - One trying to make the current value function coherent with the current policy (Policy Evaluation)
- Several variations possible on the two processes.

- In GPI, the policy is driven by the value function.
- Typically, stabilizes only if one reaches the optimal value/policy pair.
Discounted: Prediction by Value Iteration - State Update Order

**input:** MDP model \( \langle (S, A, R), P \rangle \), discount factor \( \gamma \), and stationary policy \( \pi \)

**init:** \( \tilde{v}(s) \forall s \in S \)

**repeat**
- \( \tilde{v}_{\text{prev}} \leftarrow \tilde{v} \)
- for \( s \in S' \subset S \) do
  - \( \tilde{v}(s) \leftarrow \sum_{a \in A} \pi(a|s) \left( r(s, a) + \gamma \sum_{s' \in S} p(s'|s, a)\tilde{v}_{\text{prev}}(s') \right) \)

**output:** Value function \( \tilde{v} \)

Classical strategies

- \( S' = S \): classical iteration
- \( S' = \{s\} \): Gauss-Seidel
- \( S' = \{s, |T^\pi \tilde{v}(s) - \tilde{v}(s)| > \epsilon\} \): Prioritized sweeping

- Converges provided all states are visited infinitely often...
- Gain in term of storage or focus on most interesting states...
Policy Improvement Variation

Greedy: \( \pi(s) \in \arg\max_a q(s, a) \iff \pi(\cdot|s) \in \arg\max_{\tilde{\pi}} \sum_a \tilde{\pi}(a)q(s, a) \)

Restricted: \( \pi(\cdot|s) \in \arg\max_{\tilde{\pi} \in \tilde{\Pi}_\epsilon} \sum_a \tilde{\pi}(a)q(s, a) \)

Regularized: \( \pi(\cdot|s) \in \arg\max_{\tilde{\pi}} \sum_a \tilde{\pi}(a)q(s, a) + \epsilon P(\tilde{\pi}) \)

Classical Variations

- \( \epsilon \)-greedy: Restrict \( \tilde{\pi} \) to the set of policy s.t. \( \tilde{\pi}(a) \geq \epsilon \)
  - Explicit solution: \( \pi(a|s) = \epsilon + (1 - \epsilon) \arg\max q(s, a) \)
  - Policy improvement property if \( \epsilon \) decreases.

- Soft-max: Regularize by \( \epsilon H(\tilde{\pi}) \) where \( H \) is the entropy.
  - Explicit solution: \( \pi(a|s) \propto \exp(q(s, a)/\epsilon) \)
  - No classical policy improvement...

- Tends to greedy when \( \epsilon \) goes to 0.
- Will proved to be interesting later...
Outline

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Infinite, Episodic and Average setting
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Reinforcement Learning: Prediction and Planning in the Tabular Setting
     - Prediction with Monte Carlo
     - Planning with Monte Carlo
     - Prediction with Temporal Differences
     - Link with Stochastic Approximation
     - Planning with Value Iteration
     - Planning with Policy Improvement
     - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
Infinite Setting

- No issue with the rewards as only the expectation is used.
- All the theory remains valid if the states are countable, but there is an issue in the algorithms as we need to store/update an infinite number of states.
- The proof of existence of an optimal policy requires the max to be attained, which cannot be ensured in an infinite (even countable setting).

Some results...

**Thm:** If $S$ is countable, there exists an $\epsilon$-optimal (stationary) policy for any $\epsilon > 0$.

**Thm:** If $S$ is a Polish space (completely metrizable topological space),
  - there exists a $(P, \epsilon)$-optimal (stationary policy) for any $\epsilon > 0$.
  - if each $A_s$ is countable, there exists an $\epsilon$-optimal (stationary) policy for any $\epsilon > 0$.
  - if each $A_s$ is finite, there exists an optimal (stationary) policy.
  - if each $A_s$ is a compact metric space, $r(s, a)$ is a bounded u.s.c. function on $A_s$ and $p(B|s, a)$ is continuous in $a$ for each Borel subset $B$ and any $s$, then exist an optimal (stationary) policy.

- **Mainly technical difficulties...**
Total Reward

\[ v_\pi(s) = E_\pi \left[ \sum_{t'=1}^{+\infty} R_{t+1} \mid S_0 = s \right] \]

\[ = E_\pi \left[ \sum_{t'=1}^{+\infty} \max(0, R_{t+1}) \mid S_t = s \right] - E_\pi \left[ \sum_{t'=t+1}^{+\infty} \max(0, -R_{t+1}) \mid S_t = s \right] \]

- Total reward not necessarily well defined!
- Need to **assume** this is the case!

**Classical Assumptions**

- Episodic model: \( \forall \pi, s, E_\pi \left[ \min_{t, \forall t' > t} R_{t'} = 0 \mid S_0 = s \right] < +\infty \)
- Extended Stochastic Shortest Path: \( \exists n, P_\pi(R_t = 0, \forall t \geq n) > 0. \)
- More general assumption: \( \forall \pi, s \) either \( v_{+,\pi}(s) \) or \( v_\pi(s) \) is finite.
Bellman Operator and Optimality Equation

\[
\sup_{\Pi} v_\Pi(s) = v_*(s) = \max_a r(s, a) + \sum_{s'} p(s'|s, a)v_*(s')
\]

- Similar to the discounted setting as:
  - We can focus on Markovian policy.
  - The optimal value \(v_*\) satisfies the Bellman optimality equation.

**But...**

- \(T^*\) is not a contraction and thus there may be several solution of the equation.
- If \(\pi\) is such that \(T^\pi v_* = T^* v_*\), we need to assume that \(\lim \sup (P^\pi)^n v_*(s) \leq 0\) to prove that \(\Pi = (\pi, \pi, \ldots)\) is optimal.
- There may not exists an optimal policy!

- Existence of optimal policies in the finite state-action setting by defining the total reward to the limit of discounted setting when \(\gamma \to 1\) and using the finiteness of the policy set...
Extended Stochastic Shortest Path

A policy is said to be proper if \( \exists n, P_\Pi(R_t = 0, \forall t \geq n) > 0 \)

Assumptions:
- It exists a proper policy.
- For any improper policy, it exists \( s \) such that \( v_\Pi(s) = -\infty \).

Results:
- \( v_* \) is the unique solution of \( v = T^* v \).
- Value Iteration converges and Policy Iteration converges provided \( v_0 \leq T^* v_0 \) (or finite setting).
- If all stationary policy are proper then \( T^* \) is a contraction for a weighted sup-norm.
- Any discounted model can be put in this framework by adding an absorbing state reached at random at each step with probability \( 1 - \gamma \).
### Positive Bounded Models

- $\forall \Pi, s, \ v_{+,\Pi}(s) < \infty$
- $\forall s, \exists a, \ r(s, a) \geq 0$

- Often stronger assumption: $r(s, a) \geq 0$.
- Any discounted model can be put in this framework by adding an absorbing state reached at random at each step with probability $1 - \gamma$.

### Negative Models

- $\forall \Pi, s, \ v_{+,\Pi}(s) = 0$ and $v_{-,\Pi}(s) < \infty$
- There exists a policy $\Pi$ such that $\forall s, v_\Pi(s) > -\infty$

- Maximization of $v_\Pi$ amounts to the minization of $v_{-,\Pi}$ and the negative reward can be interpreted as the opposite of costs.
- Classical Stochastic Shortest Path within this framework.
### Positive Bounded and Negative Models Results

<table>
<thead>
<tr>
<th>Result</th>
<th>Positive Bounded Models</th>
<th>Negative Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimality equation</td>
<td>$v^<em>$ is a minimal solution within $v \leq T^</em> v$</td>
<td>$v^*$ is a maximal solution within $v \geq T v$</td>
</tr>
<tr>
<td>$T^\pi v_\pi = T^* v_\pi \Rightarrow \pi$ optimal</td>
<td>Only if $\limsup(P^\pi)^n v_\pi(s) = 0$</td>
<td>Always</td>
</tr>
<tr>
<td>Existence of optimal stationary policy</td>
<td>$S$ and $A$ finite or existence of optimal policy and $r \geq 0$</td>
<td>$A_s$ finite or $A_s$ compact, $r$ and $p$ continuous with respect to $a$.</td>
</tr>
<tr>
<td>Existence of stationary $\epsilon$-optimal policy</td>
<td>If $v^*$ is bounded</td>
<td>Not always (Always for non stationary policy)</td>
</tr>
<tr>
<td>Value Iteration converges</td>
<td>$0 \leq v_0 \leq v_\pi$</td>
<td>$0 \geq v_0 \geq v_\pi$ and $A_s$ finite or $S$ finite if $v_\pi &gt; -\infty$</td>
</tr>
<tr>
<td>Policy Iteration converges</td>
<td>Yes</td>
<td>Not always</td>
</tr>
<tr>
<td>Modified Policy Iteration converges</td>
<td>$0 \leq v_0 \leq v_\pi$ and $v_0 \leq T^* v_0$</td>
<td>Not always</td>
</tr>
<tr>
<td>Solution by linear programming</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

**Total:** 118
Average Return

\[
\bar{\nu}(s) = \lim_{T \to \infty} \frac{1}{T} \nu_{T, \pi}(s) = \lim_{T \to \infty} \frac{1}{T} \mathbb{E}_\pi \left[ \sum_{t=1}^{T} R_t \bigg| S_0 = s \right]
\]

\[\longrightarrow \bar{\nu}^+(s) = \limsup_{T \to \infty} \frac{1}{T} \nu_{T, \pi}(s)\]

\[\bar{\nu}^-(s) = \liminf_{T \to \infty} \frac{1}{T} \nu_{T, \pi}(s)\]

**Average Return(s)**

- Limit \( \bar{\nu}_\pi \) may not be defined!
- **Prop:** \( \bar{\nu}_\pi \) is well defined if \( \pi \) is stationary and \( \frac{1}{T} \sum_{t=1}^{T} (P^\pi)^{t-1} \) tends to a stochastic matrix.
- Limits \( \bar{\nu}^+_{\pi} \) and \( \bar{\nu}^-_{\pi} \) always defined!
Average Returns and Optimality

\[ \bar{v}_{+,*}(s) = \sup_{\Pi} \bar{v}_{+,\Pi}(s) \quad \text{and} \quad \bar{v}_{-,*}(s) = \sup_{\Pi} \bar{v}_{-,\Pi}(s) \]

**Optimality of \( \Pi_* \)**

- **Average optimal:**
  \[ \bar{v}_{-,\Pi_*} \geq \bar{v}_{+,*,}(s) \]

- **Lim-sup average optimal (best case analysis):**
  \[ \bar{v}_{+,\Pi_*} \geq \bar{v}_{+,*,}(s) \]

- **Lim-inf average optimal (worst case analysis):**
  \[ \bar{v}_{-,\Pi_*} \geq \bar{v}_{-,*,}(s) \]

- More complex setting!
- Let’s start with Prediction...
**Prediction for a Stationary Markov Policy**

\[
\bar{v}_\Pi(s) = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} P_{\pi}^{t-1} r_\pi = \left( \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} P_{\pi}^{t-1} \right) r_\pi = P_{\pi}^\infty r_\pi
\]

**Stochastic Matrix** \( P_{\pi}^\infty \)

- Measures the average amount of time spend on a state \( s' \) starting from state \( s \) at \( t = 0 \) when using policy \( \pi \).
- Structure linked to the properties of the resulting Markov chain:
  - If aperiodic, \( P_{\pi}^\infty = \lim_T P_{\pi}^T \) i.e. \( P_{\pi}^\infty \) is close to the probability of reaching \( s' \) from \( s \) at any large \( T \).
  - If unichain, then \( P_{\pi}^\infty \) has identical rows and corresponds to the stationary distribution.
  - If multichain, then \( P_{\pi}^\infty \) has a diagonal block structure with rows equal withing each block corresponding to the stationary distribution in each chain.

- Implies that \( \bar{v}_\Pi(s) = \bar{v}_\Pi(s') \) in the Markov process is unichain.
- Limit \( P_{\pi}^\infty \) may be hard to compute...
Average Reward and Relative Value Functions

\[ U_\pi(s) = \mathbb{E}_\pi \left[ \sum_{t=1}^{\infty} (R_t - \bar{V}_\pi(S_t)) \right] \bigg| S_0 = s \]

\[ \iff \ U_\pi = (\text{Id} - P_\pi + P_\pi^\infty)^{-1} (\text{Id} - P_\pi^\infty) r_\pi \]

**Link between \( U_\pi \) and \( \bar{V}_\pi \)**

- \( (\text{Id} - P_\pi)\bar{V}_\pi = 0 \)
- \( \bar{V}_\pi + (I - P_\pi)U_\pi = r_\pi \)

**Characterization by a system**

- If \( (\text{Id} - P_\pi)\bar{V} = 0 \) and \( \bar{V} + (I - P_\pi)U = r_\pi \) then
  - \( \bar{V} = \bar{V}_\pi \),
  - \( U = U_\pi + u \) with \( (I - P_\pi)u = 0 \),
  - If \( P_\pi^\infty U = 0 \) then \( u = 0 \).

- Prediction possible by solving this system as we do not need \( U_\pi \).
Optimality Equations

\[ \bar{v}(s) = \max_a \sum_{s'} p(s'|s, a) \bar{v}(s') \]

\[ U(s) + \bar{v}(s) = \max_{a \in B_s} r(s, a) + \sum_{s'} p(s'|s, a) U(s) \text{ with } B_s = \{ a | \sum_{s'} p(s'|s, a) \bar{v}(s') = \bar{v}(s) \} \]

\[ \pi_*(s) \in \arg\max_{a \in B_s} r(s, a) + \sum_{s'} p(s'|s, a) U(s) \]

**Existence**

- If there is a solution \((\bar{v}, U)\) of the system then \(\bar{v} = \bar{v}_*\) and \(\pi_*\) is an optimal policy.
- There may exist other optimal policies not satisfying the \(\arg\max\) property.
- There may not exist solutions to the system.

- Associated relative value iteration and modified policy iteration can be defined.
- Convergence under strong assumptions...
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differentials
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy Prediction and Control
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Reinforcement Learning: Prediction and Planning in the Tabular Setting

From MDP to Interaction

From Probability to Statistics?
- What to do if one has no knowledge of the underlying MDP?
- Only information through interactions!
- Prediction? Planning?
- Focus on the discounted setting
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differentials
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Monte Carlo, i.e. Just Play!

- Most simple way to evaluate a policy.

**Just Play Following Policy \( \Pi \)**

- Play \( N \) episodes following the policy.
- During each episode, compute the (discounted) gain.
- Compute the average gain.

- What is computed?
Average Gain or Value Function

\[ \mathbb{E}[G_0] \quad \text{vs} \quad \nu_{t,\pi}(s) = \mathbb{E}[G_t | S_t = s] \]

Prediction as Value Function Evaluation

- Not the same goal.
- By construction,

\[ \mathbb{E}[G_0] = \sum_s \mu_0(s) \nu_{t,\pi}(s) \]

- Much easier to compute the average gain than the value function (even if we use a stationary policy)

- Average gain is nevertheless the most classical way to evaluate a policy (with a single number).
- Implicit episodic setting if we do not want to use approximated gain.
Episodic: Evaluation by MC

**input:** MDP environment, initial state distribution $\mu_0$, policy $\Pi$ and discount factor $\gamma$

**parameter:** Number of episodes $N$

**init:** $\tilde{V} = 0$, $n = 0$

**repeat**

\[
\begin{align*}
  n &\leftarrow n + 1 \\
  t &\leftarrow 0 \\
  G &\leftarrow 0 \\
  \text{Pick initial state } S_0 \text{ following } \mu_0 \\
  \text{repeat} \\
  &\text{Pick action } A_t \text{ according to } \pi(\cdot|S_t) \\
  &G \rightarrow G + \gamma^t R_{t+1} \\
  &t \leftarrow t + 1 \\
  \text{until } \text{episod ends at time } T \\
  \tilde{V} &\leftarrow \tilde{V} + G \\
\end{align*}
\]

**until** $n == N$

\[
\tilde{V} \leftarrow \tilde{V} / N
\]

**output:** Average gain $\tilde{V}$
Monte Carlo Prediction

- How to estimate $v_{t,\pi}$?

**Just Play Following Policy $\pi$**

- Play $N$ episodes following the policy.
- During episode, record $S_t$ and $R_t$.
- After each episode, compute recursively for each time $t$ the gain $G_t$.
- Estimate $v_{t,\pi}(s)$ by the average $G_t$ over all trajectories such that $S_t = s$

- **May require a lot of game to have a non empty set for each state $s$ at each time $t**
Monte Carlo Prediction

- How to estimate \( v_\Pi \) for a stationary policy?

**Just Play Following Policy \( \Pi \)**

- Play \( N \) episodes following the policy.
- During episode, record \( S_t \) and \( R_t \).
- After each episode, compute recursively for each time \( t \) the gain \( G_t \).
- Estimate \( v_\Pi(s) \) by the average \( G_t \) over all trajectories such that \( S_t = s \), whatever \( t \).

- The same state may be reached several times during a single episode...
- First-visit variant: Use only the first visit of \( s \) for each episode.
Monte Carlo Prediction

Episodic: Prediction by MC

**input:** MDP environment, initial state distribution $\mu_0$, policy $\Pi$ and discount factor $\gamma$

**parameter:** Number of episodes $N$

**init:** $\forall s, \tilde{V}(s), n = 0, N(s) = 0$

**repeat**

$n \leftarrow n + 1$
$t \leftarrow 0$
Pick initial state $S_0$ following $\mu_0$

**repeat**

(If First-visit) $N(S_t) \leftarrow N(S_t) + 1$
Pick action $A_t$ according to $\pi(\cdot|S_t)$
Record $R_{t+1}, S_{t+1}$
$t \leftarrow t + 1$

**until** episod ends at time $T$

$G_{T+1} = 0$
$t \rightarrow T + 1$

**repeat**

$t \leftarrow t - 1$
Compute $G_t = R_{t+1} + \gamma G_{t+1}$
(If First-visit) $\tilde{V}(S_t) = \tilde{V}(S_t) + G_t$

**until** $t = 0$

**until** $n == N$

**for** $s \in S$ **do**

$\tilde{V}(s) \leftarrow \tilde{V}(s)/N(s)$

**end**

**output:** Value function $\tilde{V}$
Monte Carlo Prediction Analysis

First-Visit Variant Analysis

- Straightforward analysis as all the used values for a given state $s$ are independent.
- Variance of order $1/N(s)$ where $N(s)$ is the number of episodes where $s$ is visited.
- Convergence if the number of visits goes to $\infty$.
- Strong assumption is practice as some states may not be visited by a given policy (if we cannot play on the initial state).

- Every-visit works... but not necessarily better!
Outline

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
Monte Carlo Planning

- Can we use a MC approach to find a good policy?

A First Attempt

- Estimate $v_\pi(s)$ by MC.
- Compute $q_\pi(s, a) = r(s, a) + \gamma \sum_{s'} p(s'|s, a)v_\pi(s)$
- Enhance the current policy by setting $\pi(s) = \text{argmax}_a q_\pi(s, a)$

- Inspired by the Operations Research results...
- But unusable as $r$ and $p$ are unknown!
A Second Attempt

- Estimate $q_{\pi}(s, a)$ by MC.
- Enhance the current policy by setting $\pi(s) = \text{argmax}_a q_{\pi}(s, a)$

- Requires that $N(s, a)$ the number of times that an episode contains the state $s$ followed by action $a$ goes to $\infty$.
- Impossible with a deterministic policy!
Monte Carlo Planning

Classical Exploratory Policies...

- Stochastic policies ensuring that any action can occur at any state.
- \( \epsilon \)-exploratory policy: use a deterministic policy and replace it with a random action with probability \( \epsilon \).
- Gibbs policy: use a policy where \( \pi(a|s) \propto e^{G(a,s)} > 0 \).

A Final Attempt

- Start from an exploratory policy.
- Estimate \( q_\pi(s,a) \) by MC.
- Enhance the current policy while remaining an exploratory policy.

- Last step is not straightforward...
- except for \( \epsilon \)-deterministic policy for which the \( \epsilon \)-exploratory policy with \( \pi(s) = \arg\max_a q_\pi(s,a) \) works.
- No convergence proof.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differencies
     - Link with Stochastic Approximation
     - Planning with Value Iteration
     - Planning with Policy Improvement
     - Exploration vs Exploitation
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Replay Buffer and Prioritized Sweeping
   - Real Time Planning

6. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

7. Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

8. References
Advanced Implementation of Monte Carlo Prediction

\[ \tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))(G_t - \tilde{v}_\pi(S_t)) \]

**On-Line Monte Carlo**

- Average for a given state can be updated each time we have the gain \( G_t \) for a state \( S_t \).
- Just use \( \alpha(N) = 1/N \) and increment \( N(S_t) \).
- No need to record the values between episodes...

- We still need to wait until the end of each episode to compute \( G_t \).
- Can we do better?
Reinforcement Learning: Prediction and Planning in the Tabular Setting

Advanced MC Prediction

**Episodic: Prediction by MC**

**input:** MDP environment, initial state distribution $\mu_0$, policy $\Pi$ and discount factor $\gamma$

**parameter:** Number of episodes $N$

**init:** $\forall s, \bar{V}(s), n = 0, N(s) = 0$

**repeat**

$n \leftarrow n + 1$
$t \leftarrow 0$

Pick initial state $S_0$ following $\mu_0$

**repeat**

(If First-visit) $N(S_t) \leftarrow N(S_t) + 1$

Pick action $A_t$ according to $\pi(\cdot|S_t)$

Record $R_{t+1}, S_{t+1}$

$t \leftarrow t + 1$

**until** episod ends at time $T$

$G_{T+1} = 0$

$t \rightarrow T + 1$

**repeat**

$t \leftarrow t - 1$

Compute $G_t = R_{t+1} + \gamma G_{t+1}$

(If First-visit) $\bar{V}(S_t) = \bar{V}(S_t) + \frac{1}{N(S_t)} (G_t - \bar{V}(S_t))$

**until** $t = 0$

**until** $n == N$

**output:** Value function $\bar{V}$

- We still need to wait until the end of each episode to compute $G_t$.
- Can we do better?
Prediction with Temporal Differences

From \( \tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))(G_t - \tilde{v}_\pi(S_t)) \)

to \( \tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))(R_{t+1} + \gamma \tilde{v}_\pi(S_{t+1}) - \tilde{v}_\pi(S_t)) \)

\[ \delta_t \]

Bootstrap Strategy

- Replace \( G_t \) by an instantaneous estimate \( R_{t+1} + \gamma \tilde{v}_\pi(S_{t+1}) \).
- Amounts to replace \( \gamma R_{t+2} + \gamma^2 R_{t+1} \) by an approximation of its expectation given \( S_{t+1} \): \( v_\pi(S_{t+1}) \).
- Bootstrap as we use the current estimate \( \tilde{v}_\pi(S_{t+1}) \) instead of the true value.
- \( \delta_t = R_{t+1} + \gamma \tilde{v}_\pi(S_{t+1}) - \tilde{v}_\pi(S_t) \) is called a temporal difference.

- No need to wait until the end of the episodes!
- Can be used in the discounted setting.
### TD Prediction

**Discounted: Prediction by TD**

**input:** MDP environment, initial state distribution $\mu_0$, policy $\Pi$ and discount factor $\gamma$

**parameter:** Number of step $T$

**init:** $\forall s, \tilde{V}(s), n = 0, N(s) = 0, t' = 0$

**repeat**

$t \leftarrow 0$

Pick initial state $S_0$ following $\mu_0$

**repeat**

$N(S_t) \leftarrow N(S_t) + 1$

Pick action $A_t$ according to $\pi(\cdot | S_t)$

$\tilde{V}(S_t) \leftarrow \tilde{V}(S_t) + \alpha (N(S_t)) \left( R_{t+1} + \gamma \tilde{V}(S_{t+1}) - \tilde{V}(S_t) \right)$

$t \leftarrow t + 1$

**until** episod ends at time $T'$ or $t' == T$

**until** $t' == T$

**output:** Value function $\tilde{V}$

- **But does this work?**
Prediction with Temporal Differentials

\[
\mathbb{E}[\delta_t|S_t] \mathbb{E}[R_{t+1} + \gamma \tilde{v}_\pi(S_{t+1}) - \tilde{v}_\pi(S_t)|S_t] = (\mathcal{T}^\pi - \text{Id}) \tilde{v}_\pi(S_t)
\]

**TD and Bellman Operator**

- TD as an approximate Policy Iteration:
  \[
  \mathbb{E}[\tilde{v}_\pi](S_t) \leftarrow \tilde{v}_\pi + \alpha(N(S_t)) (\mathcal{T}^\pi - \text{Id}) \tilde{v}_\pi(S_t)
  \]
- Proof of convergence of this algorithm to a zero of $\mathcal{T}^\pi - \text{Id}$, i.e. the fixed point of $\mathcal{T}^\pi$!
- Proof requires a mild assumption of $\alpha$ (satisfied by $\alpha(N) = 1/N$) and the strong assumption that $N(s)$ goes to $\infty$.

- MC could be interpreted in a similar way (stochastic approximation) by noticing that
  \[
  \mathbb{E}[G_t - \tilde{v}_\pi(S_t)|S_t] = v_\pi(S_t) - \tilde{v}_\pi(S_t).
  \]
- Often use with a constant $\alpha$
MC vs TD

\[ \tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))(G_t - \tilde{v}_\pi(S_t)) \]

or

\[ \tilde{v}_\pi(S_t) \leftarrow \tilde{v}_\pi(S_t) + \alpha(N(S_t))(R_{t+1} + \gamma \tilde{v}_\pi(S_{t+1}) - \tilde{v}_\pi(S_t)) \]

MC vs TD

- Both are based on stochastic approximation.
- Both converges (under similar assumptions) to the correct value function.
- TD does not require to wait until the end of the episode.
- No theoretical difference in the speed of convergence but often TD is better...
- Solve different approximate problems when used with a finite set of episodes:
  - MC compute the empirical gain from any state.
  - TD compute the value function of the empirical Bellman operator (the one obtained by using the empirical transition probabilities)

- If \( \tilde{v}_\pi \) is kept constant during an episode

\[ G_t - \tilde{v}_\pi(S_t) = \sum \gamma^{t' - t} \delta_t \]
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differencies
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
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:
  - \( \mathbb{E}[\epsilon_k] = 0, \forall \text{var } [\epsilon_k] < \sigma^2, \) 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 \( \alpha \) 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 \( \frac{d\tilde{\theta}}{dt} = H(\tilde{\theta}) \)

ODE Approach

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

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

- \( \alpha_k \) can be interpreted as a time difference allowing to define a time \( t_k = \sum_{t' \leq t} \alpha_k \).

- Equation be interpreted as the derivative at time \( t \in (t_k, t_{k+1}) \) of a piecewise affine function \( \theta(t) \).

- This piecewise function remains close to any solution of the ODE starting from \( \theta_k \) for an arbitrary amount of time provided \( k \) is large enough.
Sketch of Proof

- Difference between $\theta$ and a solution of the ODE with $\tilde{\theta}(t_k) = \theta_k$ at $t_{k+1}$:
  \[
  \theta(t_{k+1}) - \tilde{\theta}(t_{k+1}) = \int_{t_k}^{t_{k+1}} (\theta'(u) - \tilde{\theta}'(u)) \, du
  \]
  \[
  = \sum_{k' = k}^{k+l-1} \int_{t_{k'}}^{t_{k'+1}} \left( H(\theta(t_k)) + \epsilon_k + \eta_k - H(\tilde{\theta}(u)) \right) \, du
  \]
  \[
  = \sum_{k' = k}^{k+l-1} \int_{t_{k'}}^{t_{k'+1}} \left( H(\theta(t_k)) - H(\tilde{\theta}(u)) \right) \, du
  \]
  \[
  + \sum_{k' = k}^{k+l-1} \alpha_{k'} \epsilon_{k'} + \sum_{k' = k}^{k+l-1} \alpha_{k'} \eta_{k'}
  \]

- The last two terms are going to be small by construction...
Sketch of Proof

- Difference between $\theta$ and a solution of the ODE with $\tilde{\theta}(t_k) = \theta_k$ at $t_{k+1}$:

$$\theta(t_{k+1}) - \tilde{\theta}(t_{k+1}) = \sum_{k'=k}^{k+l-1} \int_{t_{k'}}^{t_{k'+1}} \left( H(\theta(t_k)) - H(\tilde{\theta}(u)) \right) du$$

$$+ \sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} + \sum_{k'=k}^{k+l-1} \alpha_{k'} \eta_{k'}$$

- The last two term are going to be small by construction:

$$\mathbb{E} \left[ \sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} \right] = 0 \quad \text{and} \quad \mathbb{V} \text{ar} \left[ \sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} \right] < \sigma^2 \sum_{k'=k}^{k+l-1} \alpha_{k'}^2 \rightarrow 0$$

$$\| \sum_{k'=k}^{k+l-1} \alpha_{k'} \eta_{k'} \| \leq (t_{k+l-1} - t_k) \sup_{k' \geq k} \| \eta_{k'} \|$$

which is small if we assume that $t_{k+l-1} - t_k \leq \Delta$. 
Sketch of Proof

- We can now use a Lipschitz assumption on $H$ to obtain:

$$
\left\| \int_{t_{k'}}^{t_{k'}+1} \left( H(\theta(t_{k'})) - H(\tilde{\theta}(u)) \right) du \right\| \leq L \int_{t_{k'}}^{t_{k'}+1} \| \theta(t_{k'}) - \tilde{\theta}(u) \| du
$$

$$
\leq L \alpha_{k'} \| \theta(t_{k'}) - \tilde{\theta}(t_{k'}) \| + L \int_{t_{k'}}^{t_{k'}+1} \| \theta(t_{k'}) - \tilde{\theta}(u) du \|
$$

$$
\leq L \alpha_{k'} \| \theta(t_{k'}) - \tilde{\theta}(t_{k'}) \| + L \| H \|_\infty \alpha_{k'}^2
$$

- Combinining all the result leads to

$$
\| \theta(t_{k+1}) - \tilde{\theta}(t_{k+1}) \| \leq L \sum_{k' = k}^{k+l-1} \alpha_{k'} \| \theta(t_{k'}) - \tilde{\theta}(t_{k'}) \|
$$

$$
+ L \| H \|_\infty \sum_{k' = k}^{k+l-1} \alpha_{k'}^2 + \sum_{k' = k}^{k+l-1} \alpha_{k'} \epsilon_{k'} + \sum_{k' = k}^{k+l-1} \alpha_{k'} \eta_{k'}
$$

- A Gronwall type Lemma allows to conclude.
Sketch of Proof

- Combinining all the results leads to

\[
\|\theta(t_{k+1}) - \tilde{\theta}(t_{k+1})\| \leq L \sum_{k'=k}^{k+l-1} \alpha_{k'} \|\theta(t_{k'}) - \tilde{\theta}(t_{k'})\|
\]

\[
+ L \|H\|_\infty \sum_{k'=k}^{k+l-1} \alpha_{k'}^2 + \left[ \sum_{k'=k}^{k+l-1} \alpha_{k'} \epsilon_{k'} \right] + \sum_{k'=k}^{k+l-1} \alpha_{k'} \|\eta_{k'}\|
\]

- Using a discrete Gronwall Lemma,

\[
\forall l \leq l'', z_l \leq L \sum_{l'=0}^{l-1} \alpha_{l'} z_{l'} + A \Rightarrow z_{l''} \leq A e^{L \sum_{l'=0}^{l-1} \alpha_{l'}},
\]

we obtain that if \( t_{k+1} - t_k \leq \Delta \)

\[
\|\theta(t_{k+1}) - \tilde{\theta}(t_{k+1})\| \leq \left( L \|H\|_\infty \sum_{k'=k}^{\infty} \alpha_{k'}^2 + \sup_{l' \leq l} \left[ \sum_{k'=k}^{k+l'-1} \alpha_{k'} \epsilon_{k'} \right] + L \sup_{k' \geq k} \|\eta_{k'}\| \right) e^{L \Delta}
\]

\[
\rightarrow 0 \text{ when } k \rightarrow \infty
\]
Asynchronous Update

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

to $\forall i, \theta_{k+1}(i) = \theta_k(i) + \alpha_k(i) h_k(\theta_k)(i)$

Asynchronous Update

- Componentwise action on $\theta$.
- Not necessarily the same stepsize $\alpha_k(i)$ for all components.
- $\alpha_k(i) = 0$ is permitted!
- Previous results hold provided for every component $i$, $\sum_k \alpha_k(i) \to \infty$ and $\sum_k \alpha_k^2(i) < \infty$.
- Exact setting of TD approximation!
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

6. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

7. Replay Buffer and Prioritized Sweeping
   - Real Time Planning

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

9. References
Planning with Temporal Differencies

A State Value Function Attempt

- $V_*$ is the fixed point of $\mathcal{T}^*$.  
- Approximate it as the zero of $\mathcal{T}^* - \text{Id}$.  
- By construction  
  $$\mathcal{T}^* V(S_t) = \max_a \mathbb{E}[R_{T+1} + \gamma V(S_{t+1}) | S_t, a]$$  
- Not an expectation!

A State-Action Value Function Attempt

- $q_*$ is the fixed point of $\mathcal{T}^*$.  
- Approximate it as the zero of $\mathcal{T}^* - \text{Id}$.  
- By construction  
  $$\mathcal{T}^* q(S_t, A_t) = \mathbb{E} \left[ R_{t+1} + \gamma \max_a q(S_{t+1}, a) | S_t, A_t \right]$$  
- An expectation!
Q Learning

Discounted: Planning by Q-Learning

**input:** MDP environment, initial state distribution $\mu_0$, policy $\Pi$ and discount factor $\gamma$

**parameter:** Number of step $T$

**init:** $\forall s, a, \tilde{Q}(s, a), N(s, a) = 0$, $n=0$, $t' = 0$

**repeat**

1. $t \leftarrow 0$
2. Pick initial state $S_0$ following $\mu_0$

**repeat**

1. $N(S_t) \leftarrow N(S_t) + 1$
2. Pick action $A_t$ according to $\pi(\cdot|S_t)$
3. $\tilde{Q}(S_t, A_t) \leftarrow \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t))(R_{t+1} + \gamma \max_a \tilde{Q}(S_{t+1}, a) - \tilde{Q}(S_t, A_t))$
4. $t \leftarrow t + 1$
5. $t' \leftarrow t' + 1$

**until** episod ends at time $T'$ or $t' == T$

**until** $t' == T$

**output:** Deterministic policy $\tilde{\pi}(s) = \arg\max_a \tilde{Q}(s, a)$
Planning with Q Learning

\[ \tilde{Q}(S_t, A_t) = \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t)) \left( R_{t+1} + \gamma \max_a \tilde{Q}(S_{t+1}, a) - \tilde{Q}(S_t, A_t) \right) \]

Q-Learning

- Update is independent of the policy $\Pi$.
- Convergence of the Q-value function provided the policy is such that $N(s, a)$ tends to $\infty$ for any state and any action.
- Implies a convergence of the policy.
- Relies on temporal difference.

- Most classical (tabular) planning algorithm!
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy Prediction and Control
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References

Reinforcement Learning: Prediction and Planning in the Tabular Setting

- Optimization Interpretation
- Approximation and Stability
- Generalized Policy Iteration
- Infinite, Episodic and Average setting
- Replay Buffer and Prioritized Sweeping
- Real Time Planning
- Approximation Target(s)
- Gradient and Pseudo-Gradient
- Linear Approximation and LSTD
- On-Policy Prediction and Control
- Off-Policy Prediction and Control
- Two-Scales Algorithms
- Deep Q Learning
- Continuous Actions
- Average Return

References
Planning with Policy Improvement

from \( \tilde{Q}(S_t, A_t) = \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t)) \)

\( R_{t+1} + \gamma \max_a \tilde{Q}(S_{t+1}, a) - \tilde{Q}(S_t, A_t) \)

\( \delta_t \)

\( \Pi(S_t) = \arg\max_a \tilde{Q}(S_t, a) \) (plus exploration)

to \( \tilde{Q}(S_t, A_t) = \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t)) \)

\( R_{t+1} + \gamma \tilde{Q}(S_{t+1}, A_{t+1}) - \tilde{Q}(S_t, A_t) \)

\( \delta_t \)

Policy Improvement

- More emphasis on the policy with a link between the policy used to play and the optimized policy.
- Almost equivalent to use the current policy in the Q-Learning algorithm.
Discounted: Planning by SARSA

**input:** MDP environment, initial state distribution $\mu_0$, policy $\Pi$ and discount factor $\gamma$

**parameter:** Number of step $T$

**init:** $\forall s, a, \tilde{Q}(s, a), N(s, a) = 0$, $n=0$, $t' = 0$

**repeat**

$t \leftarrow 0$ Pick initial state $S_0$ following $\mu_0$

**repeat**

$N(S_t) \leftarrow N(S_t) + 1$
Pick action $A_t$ according to $\pi(\cdot|S_t)$

$\tilde{Q}_t(S_{t-1}, A_{t-1}) \leftarrow \tilde{Q}(S_{t-1}, A_{t-1}) + \alpha(N(S_{t-1}, A_{t-1}))(R_t + \gamma \tilde{Q}(S_t, A_t) - \tilde{Q}(S_{t-1}, A_{t-1}))$

$\Pi(S_{t-1}) = \text{argmax } \tilde{Q}(S_{t-1}, A_{t-1})$ (plus exploration)

$t \leftarrow t + 1$

$t' \leftarrow t' + 1$

**until** episod ends at time $T'$ or $t' == T$

**until** $t' == T$

**output:** Deterministic policy $\tilde{\pi}(s) = \text{argmax}_a \tilde{Q}(s, a)$

- Does this work?
Reinforcement Learning: Prediction and Planning in the Tabular Setting

SARSA and exploration

$$\Pi(S_t) = \arg\max_a \tilde{Q}(S_t, a) \text{(plus exploration)}$$

- No hope of convergence if we do not explore all possible actions (and states).
- Impossible if the policy used is deterministic.
- Exploration is required!
- Most classical choice: $\epsilon$-greedy policy with a decaying $\epsilon$.

- Convergence proof is harder than for $Q$-Learning.
- Relies on the similarity in the limit (when $\epsilon$ goes to 0) with the $Q$-Learning algorithm.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differencing
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement

4. Exploration vs Exploitation

5. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References

Reinforcement Learning: Prediction and Planning in the Tabular Setting
- Optimization Interpretation
- Approximation and Stability
- Generalized Policy Iteration
- Infinite, Episodic and Average setting
- Real Time Planning
- Replay Buffer and Prioritized Sweeping

Reinforcement Learning: Approximation of the Value Functions
- Approximation Target(s)
- Gradient and Pseudo-Gradient
- Linear Approximation and LSTD
- On-Policy Prediction and Control
- Off-Policy Prediction and Control
- Two-Scales Algorithms
- Deep Q Learning
- Continuous Actions

References
Q-Learning vs SARSA

How different are they?

- In Q-learning, the exploratory policy used is decoupled from the optimized policy.
- This exploratory policy may yield low rewards on average.
- In SARSA, the two policies are linked with the hope on having higher rewards during the learning step.
- Subtle different behavior even if we modify the exploratory policy in Q-Learning.
Exploration vs Exploitation

- Exploration: explore new policies to be able to discover the best ones.
- Exploitation: use good policies to obtain a good return.
- Exploration is a requirement.

- No tradeoff if we optimize only the final result!
- Tradeoff between the two if we consider that the returns during training matters!
- \( Q \)-learning use the first approach and SARSA try to tackle the second.
- Tradeoff if we study a regret:
  \[
  \sum_t \mathbb{E}_{\pi^*}[R_t] - \mathbb{E}_{\pi_t}[R_t]
  \]
  which forces us to be good as fast as possible.
- No natural definition in the discounted setting.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - $n$-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Advanced Tabular Reinforcement Learning

Core idea: Approximate Bellman Operators with Stochastic Approximation...

Advanced Ideas?
- Between MC and TD?
- Off-policy vs on-policy?
- Exploration vs Exploitation?
- Model? Replay?
- Real Time Planning?
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
How many steps before backup?

- One step: TD.
- As many steps as required to end the episod: MC.
- \(n\)-steps: \(n\)-steps TD.

\[
\left( \mathcal{T}^n \right)^n V(s) = \mathbb{E}_\pi \left[ R_{t+1} + \gamma R_{t+2} + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n}) \mid S_t = s \right]
\]

- Family of stochastic approximation algorithms:

\[
V(S_t) \leftarrow V(S_t) + \alpha(N(S_t)) \left( G_{t:t+n} - V(S_t) \right)
\]
\[ V(S_t) \leftarrow V(S_t) + \alpha(N(S_t))(G_{t:t+n} - V(S_t)) \]

**n-steps TD**

- Convergence for prediction.
- Need to be combined with Policy Improvement for planning: \( n \)-steps SARSA.
- \( n \)-steps Q-learning could be an extension of API... but this means following the optimized policy \( \Pi \)... i.e. SARSA!

- Best convergence often for intermediate \( n \).
- No proof beside TD for \( n > 1 \)!
**n-steps TD**

Discounted: Prediction by *n*-steps TD

**input:** MDP environment, initial state distribution \( \mu_0 \), policy \( \Pi \) and discount factor \( \gamma \)

**parameter:** Number of step \( T \)

**init:** \( \forall s, a, \tilde{Q}(s, a), N(s, a) = 0, n=0, t' = 0 \)

**repeat**

\[
t \leftarrow 0 \\
\text{Pick initial state } S_0 \text{ following } \mu_0 \\
\text{repeat}
\]

\[
N(S_t) \leftarrow N(S_t) + 1 \\
\text{Pick action } A_t \text{ according to } \pi(\cdot | S_t) \\
\tilde{Q}(S_{t-n}, A_{t-n}) \leftarrow \tilde{Q}(S_{t-n}, A_{t-n}) + \alpha(N(S_t, A_t))(G_{t-n:t} - \tilde{Q}(S_t, A_t))
\]

\[
t \leftarrow t + 1 \\
t' \leftarrow t' + 1
\]

**until episod ends at time } T' \text{ or } t' == T

**until** \( t' == T \)

**output:** State-Action value function \( \tilde{Q} \)
Expected SARSA

The policy $\Pi$ is known so that we can use it in a formula:

$$R_t + \gamma Q(S_t, A_t) \rightarrow R_t + \gamma \sum_a \pi(a|S_t) Q(S_t, a)$$

- Make the update independent of the action chosen (and thus of the policy used to play).
- Reduce the variance for a computational cost.
- Amount to use the current estimate for $V(S_t)$...
Expected SARSA

Discounted: Prediction by Expected SARSA

input: MDP environment, initial state distribution $\mu_0$, policy $\Pi$ and discount factor $\gamma$

parameter: Number of step $T$

init: $\forall s, a, \tilde{Q}(s, a), N(s, a) = 0$, $n=0$, $t' = 0$

repeat

$t \leftarrow 0$

Pick initial state $S_0$ following $\mu_0$

repeat

$N(S_t) \leftarrow N(S_t) + 1$

Pick action $A_t$ according to $\pi(\cdot|S_t)$

$\tilde{Q}(S_t, A_t) \leftarrow \tilde{Q}(S_t, A_t) + \alpha(N(S_t, A_t)) \left( R_{t+1} + \gamma \sum_a \pi(a|S_t) \tilde{Q}(S_{t+1}, a) - \tilde{Q}(S_t, A_t) \right)$

$t \leftarrow t + 1$

$t' \leftarrow t' + 1$

until episod ends at time $T'$ or $t' == T$

until $t' == T$

output: State-Action value function $\tilde{Q}$
At each time step, use the expected SARSA average over the action while replacing the Q value for the picked action by a deeper estimate.

1-step return (Expected Sarsa)

\[ G_{t:t+1} = R_{t+1} + \gamma \sum_a \pi(a|S_{t+1})Q(S_{t+1}, a) \]

2-step return:

\[ G_{t:t+2} = R_{t+1} + \gamma \sum_{a \neq A_{t+1}} \pi(a|S_{t+1})Q_{t+1}(S_{t+1}, a) \]

\[ + \gamma \pi(A_{t+1}|S_{t+1}) \left( R_{t+2} + \gamma \sum_a \pi(a|S_{t+2})Q(S_{t+2}, a) \right) \]

\[ = R_{t+1} + \gamma \sum_{a \neq A_{t+1}} \pi(a|S_{t+1})Q(S_{t+1}, a) + \gamma \pi(A_{t+1}|S_{t+1})G_{t+1:t+2} \]
n-steps Tree Backup

- 1-step return (Expected Sarsa)
  \[ G_{t:t+1} = R_{t+1} + \gamma \sum_a \pi(a|S_{t+1})Q(S_{t+1}, a) \]

- 2-step return:
  \[
  G_{t:t+2} = R_{t+1} + \gamma \sum_{a \neq A_{t+1}} \pi(a|S_{t+1})Q(S_{t+1}, a) + \gamma \pi(A_{t+1}|S_{t+1}) G_{t+1:t+2} \\
  = R_{t+1} + \gamma \sum_a \pi(a|S_{t+1})Q(S_{t+1}, a) + \gamma \pi(A_{t+1}|S_{t+1}) (G_{t+1:t+2} - Q(S_{t+1}, A_{t+1}))
  \]

- Recursive definition of n-step return:
  \[
  G_{t:t+n} = R_{t+1} + \gamma \sum_a \pi(a|S_{t+1})Q(S_{t+1}, a) \\
  + \gamma \pi(A_{t+1}|S_{t+1}) (G_{t+1:t+n} - Q(S_{t+1}, A_{t+1}))
  \]

- TD update
  \[
  Q(S_{t-n}, A_{t-n}) = Q(S_{t-n}, A_{t-n}) + \alpha (N(S_{t-n}, Q_{t-n})) (G_{t-n:t} - Q(S_{t-n}, A_{t-n}))
  \]
Reinforcement Learning: Advanced Techniques in the Tabular Setting

### Sampling or Averaging

- **Unifying algorithm!**
- **Recursive definition of** $n$-**step return:**

$$G_{t:t+n} = R_{t+1} + \sigma G_{t+1:t+n}$$

$$+ (1 - \sigma) \left( \gamma \sum_a \pi(a|S_{t+1}) Q(S_{t+1}, a) 
+ \gamma \pi(A_{t+1}|S_{t+1}) (G_{t+1:t+n} - Q(S_{t+1}, A_{t+1})) \right)$$
\(\lambda\)-Return

**Averaged \(n\)-steps return?**

- \(n\)-step return:
  \[
  G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})
  \]

- Averaged \(n\)-step return: (compound update)
  \[
  G^\omega_t = \sum_{n=1}^{\infty} \omega_n G_{t:t+n} \quad \text{with} \quad \sum_{i=1}^{\infty} \omega_n = 1
  \]

- TD(\(\lambda\)): specific averaging
  \[
  G^\lambda_t = (1 - \lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n}
  \]
  \[
  = (1 - \lambda) \sum_{n=1}^{T-t} \lambda^{n-1} G_{t:t+n} + \lambda^{T-t} G_t \quad \text{(Episodic)}
  \]
  interpolating between TD (a.k.a. TD(0)) and MC for \(\lambda = 1\).

- Can be mixed with tree backup strategies (TB(\(\lambda\)))
λ-return and Temporality

**True λ-return**
- Require to wait until the end of an episode before we can update.
- Unusable in a non episodic setting!

**Truncated λ-return**
- Truncated λ-return:
  \[
  G_t^\lambda = (1 - \lambda) \sum_{n=1}^{H-t} \lambda^{n-1} G_{t:t+n} + \lambda^{H-t} G_{t:H}
  \]
- The virtual horizon \( H \) may vary during the algorithm.
λ-return and Temporality

**Temporality**

- **n-step return**
  
  \[ G_{t:t+n} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n}) \]
  
  depends on a current estimate \( V \) (or \( Q \))!

- In \( G_\lambda \) should we use
  - an estimate available at time \( t \)?
  - an estimate available at time \( t + n \)?
  - an estimate available at time \( H \)?

- **Off-Line vs On-Line!**
  - Off-line: keep \( V \) constant during the episodes.
  - On-line: Used updated \( V \) when available.
  - True on-line (Sutton and Barto): restart algorithm with a growing horizon.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - $n$-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Forward and Backward Point of View

From a forward view

To a backward one:

Source: Sutton and Barto
Returns and Temporal Differencies

- *n*-step returns:
  \[
  G_{t:t+n} - Q(S_t, A_t) = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} \\
  + \gamma^n Q(S_{t+n}, A_{t+n}) - Q(S_t, A_t)
  \]
  \[
  = \sum_{l=1}^{n} \gamma^{l-1} (R_{t+l} + \gamma Q(S_{t+l}, A_{t+l}) - Q(S_{t+l-1}, A_{t+l-1}))
  \]
  \[
  = \sum_{l=0}^{n-1} \gamma^{l-1} \delta_{t+l}
  \]

- \(\lambda\) return:
  \[
  G_{t}^{\lambda} - Q(S_t, A_t) = (1 - \lambda) \sum_n \lambda^n (G_{t:t+n} - Q(S_t, A_t))
  \]
  \[
  = \sum_{n=0} \lambda^n \gamma^n \delta_{t+n}
  \]
Forward View and Backward View

Forward View

- Updates:
  \[
  Q_t(s, a) = Q_{t-1}(s, a) + 1_{(s,a)=(S_t, A_t)} \alpha_t(s, a) \left( \sum_{t'\geq t} \lambda^{t'-t} \gamma^{t''-t} \delta_{t''} \right)
  \]

- Cumulative updates:
  \[
  Q_t(s, a) = Q_0(s, a) + \sum_{t'\leq t} 1_{(s,a)=(S_{t'}, A_{t'})} \alpha_{t'}(s, a) \left( \sum_{t''\geq t'} \lambda^{t''-t'} \gamma^{t''-t'} \delta_{t''} \right)
  \]

- Limit:
  \[
  Q_\infty(s, a) = Q_0(s, a) + \sum_{t'} 1_{(s,a)=(S_{t'}, A_{t'})} \alpha_{t'}(s, a) \left( \sum_{t''\geq t'} \lambda^{t''-t'} \gamma^{t''-t'} \delta_{t''} \right)
  \]

- Focus on the update place.
Forward View and Backward View

Limit(s)

- Limit:

\[
Q_\infty(s, a) = Q_0(s, a) + \sum_{t'} \mathbf{1}(s, a) = (S_{t'}, A_{t'}) \alpha_{t'}(s, a) \left( \sum_{t'' \geq t'} \lambda^{t'' - t'} \gamma^{t'' - t'} \delta_{t''} \right)
\]

\[
= Q_0(s, a) + \sum_{t''} \delta_{t''} \sum_{t' \leq t''} \mathbf{1}(s, a) = (S_{t'}, A_{t'}) \alpha_{t'}(s, a) \lambda^{t'' - t'} \gamma^{t'' - t'}
\]

- Focus on the update place or and the temporal differences...
### Backward View

- **Same limit with cumulative udpates over temporal differencies**

\[
Q_t(s, a) = Q_0(s, a) + \sum_{t'' \leq t} \delta_{t''} \sum_{t' \leq t''} 1_{(s,a) = (S_{t'}, A_{t'})} \alpha_{t'}(s, a) \lambda^{t''-t'} \gamma^{t''-t'}
\]

- **Updates**

\[
Q_t(s, a) = Q_{t-1}(s, a) + \delta_t \sum_{t' \leq t} 1_{(s,a) = (S_{t'}, A_{t'})} \alpha_{t'}(s, a) \lambda^{t-t'} \gamma^{t'-t'}
\]

- **Pseudo Eligibility trace:**

\[
z_t(s, a) = \sum_{t' \leq t} 1_{(s,a) = (S_{t'}, A_{t'})} \alpha_{t'}(s, a) \lambda^{t-t'} \gamma^{t'-t'}
\]

\[
= \lambda \gamma z_{t-1}(s, a) + \alpha_t(s, a) 1_{(s,a) = (S_t, A_t)}
\]

- **Proof of convergence toward the same target.**
**Eligibility Trace**

\[ Q_t(s, a) = Q_{t-1}(s, a) + \alpha_t \delta_t z_t(s, a) \]

- Focus on temporal differences with simultaneous update on all states.
- TD(\(\lambda\)) eligibility trace: \(z_t(s, a) = \lambda \gamma z_{t-1}(s, a) + 1_{(s,a)=(S_t,A_t)}\)
- Strictly equivalent to the previous scheme for constant stepsize
- Other eligibility trace:
  - Replacing trace:
    \[
    z_t(s, a) = \begin{cases} 
    1 & \text{if } (s, a) = (S_t, A_t) \\
    \lambda \gamma z_{t-1}(s, a) & \text{otherwise}
    \end{cases}
    \]
  - Time dependent trace:
    \[
    z_t(s, a) = c_t \gamma z_{t-1}(s, a) + 1_{(s,a)=(S_t,A_t)}
    \]
    where \(c_t\) is defined in a appropriate way to ensure the convergence of the algorithm.

- Need to store (and update) this information...
Temporal Differencies

\[ \delta_t \]

Basic temporal differencies:
\[ \delta_t = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t) \]

Expected temporal differencies:
\[ \delta_t = R_{t+1} + \gamma V(S_{t+1}) - Q(S_t, A_t) \]
\[ = R_{t+1} + \gamma \sum_a \pi(a|S_{t+1}) Q(S_{t+1}, a) - Q(S_t, A_t) \]

Average of both:
\[ \delta_t = R_{t+1} + \gamma \sigma Q(S_{t+1}, A_{t+1}) + \gamma (1 - \sigma) V(S_{t+1}) - Q(S_t, A_t) \]
\[ = R_{t+1} + \gamma V(S_{t+1}) + \gamma \sigma (Q(S_{t+1}, A_{t+1}) - V(S_{t+1})) - Q(S_t, A_t) \]

- Only expected temporal average is independent of the next action.
- No generic proof of convergence...
Outline

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting
   - Exploration vs Exploitation

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
On-Policy vs Off-Policy

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

- Off-Policy allows in particular to (re)use interactions from previous experiments.
- Q-learning was possible in off-policy setting.
### Importance Sampling

For any law \( p \) and \( q \), and any function \( g \)

\[
\mathbb{E}_p[g(x)] = \mathbb{E}_q \left[ \frac{p(x)}{q(x)} g(x) \right]
\]

provided \( q(x) = 0 \) implies \( p(x) = 0 \).

\( \mathbb{V} \text{ar}_q \left[ \frac{p(x)}{q(x)} g(x) \right] \) may be large with respect to \( \mathbb{V} \text{ar}_p [g(x)] \) if the ratio \( p(x)/q(x) \) is large...

### Importance Sampling for Trajectories

For any trajectory \( \tau_{t:t'} = S_t, A_t, R_{t+1}, S_{t+1}, \ldots, R_{t'}, S_{t'}, A_{t'}(, R_{t'+1}, S_{t'+1}), \)

\[
\mathbb{P}_\Pi(S_t, A_t, R_{t+1}, S_{t+1}, \ldots, R_{t'}, S_{t'}, A_{t'}(, R_{t'+1}, S_{t'+1})|S_t) = \pi(A_t|S_t) \ldots \pi(A_{t'}|S_{t'})
\]

\[
\mathbb{P}_b(S_t, A_t, R_{t+1}, S_{t+1}, \ldots, R_{t'}, S_{t'}, A_{t'}(, R_{t'+1}, S_{t'+1})|S_t) = b(A_t|S_t) \ldots b(A_{t'}|S_{t'})
\]
Reinforcement Learning: Advanced Techniques in the Tabular Setting

Importance Sampling and Returns

\[
\mathbb{E}_\Pi[g(\tau_{t:t'})|S_t = s] = \mathbb{E}_b[\rho_{t:t'}g(\tau_{t:t'})|S_t = s] \quad \text{with} \quad \rho_{t:t'} = \frac{\pi(A_t|S_t) \cdots \pi(A_{t'}|S_{t'})}{b(A_t|S_t) \cdots b(A_{t'}|S_{t'})}
\]

From \( b \) to \( \Pi \)

- Returns:

\[
\mathbb{E}_\pi[G_{t:t'}|S_t = s] = \mathbb{E}_\pi \left[ \sum_{t'' = t+1}^{t'} \gamma^{t''-t-1} R_{t''} + \gamma^{t'-t} V(S_{t'}) \bigg| S_t = s \right]
\]

\[
= \mathbb{E}_b \left[ \rho_{t:(t'-1)} \left( \sum_{t'' = t+1}^{t'} \gamma^{t''-t-1} R_{t''} + \gamma^{t'-t} V(S_{t'}) \right) \bigg| S_t = s \right]
\]

\[
= \mathbb{E}_b \left[ \sum_{t'' = t+1}^{t'} \rho_{t:(t''-1)} \gamma^{t''-t-1} R_{t''} + \rho_{t:(t'-1)} \gamma^{t'-t} V(S_{t'}) \bigg| S_t = s \right]
\]
Importance Sampling and Returns

\[ \mathbb{E}_\pi[g(\tau_t:t')|S_t, A_t] = \mathbb{E}_b[\rho_{(t+1):t'}g(\tau_t:t')|S_t, A_t] \quad \text{with} \quad \rho_{t:t'} = \frac{\pi(A_t|S_t) \cdots \pi(A_{t'}|S_{t'})}{b(A_t|S_t) \cdots b(A_{t'}|S_{t'})} \]

From \( b \) to \( \Pi \)

- **Returns:**

\[ \mathbb{E}_\pi[G_{t:t'}|S_t, A_t] = \mathbb{E}_\pi \left[ \sum_{t''=t+1}^{t'} \gamma^{t''-t-1} R_{t''} + \gamma^{t'-t} Q(S_{t'}, A_{t'}) \right]_{S_t, A_t} \]

\[ = \mathbb{E}_b \left[ \rho_{(t+1):t'-1} \left( \sum_{t''=t+1}^{t'} \gamma^{t''-t-1} R_{t''} + \gamma^{t'-t} Q(S_{t'}, A_{t'}) \right) \right]_{S_t, A_t} \]

\[ = \mathbb{E}_b \left[ \rho_{(t+1):(t''-1)} \sum_{t''=t+1}^{t'} \gamma^{t''-t-1} R_{t''} + \rho_{(t+1):t'} \gamma^{t'-t} Q(S_{t'}, A_{t'}) \right]_{S_t, A_t} \]

- **No correction if** \( t' = t + 1 \)
### $\lambda$-return

- **Recursive definition of the $\lambda$-return:**
  \[
  G_t^\lambda|S_t = R_{t+1} + \gamma \left( (1 - \lambda)V(S_{t+1}) + \lambda G_{t+1}^\lambda \right)
  \]
  \[
  G_t^\lambda|S_t, A_t = R_{t+1} + \gamma \left( (1 - \lambda)(\sigma Q(S_{t+1}, A_{t+1}) + (1 - \sigma)(\sum_a \pi(a|S_{t+1})Q(S_{t+1}, a)
  
  + \pi(A_{t+1}|S_{t+1}) \left( G_{t+1}^\lambda - Q(S_{t+1}, A_{t+1}) \right))\right) + \lambda G_{t+1}^\lambda
  \]

- **Off-line correction**
  \[
  G_t^\lambda|S_t = \rho_{t:t} \left( R_{t+1} + \gamma \left( (1 - \lambda)V(S_{t+1}) + \lambda G_{t+1}^\lambda \right) \right)
  \]
  \[
  G_t^\lambda|S_t, A_t = R_{t+1} + \gamma \left( (1 - \lambda)(\sigma Q(S_{t+1}, A'_{t+1}) + (1 - \sigma)(\sum_a \pi(a|S_{t+1})Q(S_{t+1}, a)
  
  + \pi(A_{t+1}|S_{t+1}) \left( G_{t+1}^\lambda - Q(S_{t+1}, A_{t+1}) \right))\right)
  
  + \lambda \rho_{t+1:t+1} G_{t+1}^\lambda
  \]

where $A'_{t+1}$ is drawn following $\pi$ (or multiply by $\rho_{t+1:t+1}$ to use $A_{t+1}$).
Temporal Differencies

$\delta_t$?

- **Basic temporal differencies:**
  \[
  \delta_t = R_{t+1} + \gamma Q(S_{t+1}, A'_{t+1}) - Q(S_t, A_t)
  \]
  with $A'_{t+1}$ drawn using $\pi$.

- **Expected temporal differencies:**
  \[
  \delta_t = R_{t+1} + \gamma V(S_{t+1}) - Q(S_t, A_t)
  \]
  \[
  = R_{t+1} + \gamma \sum_a \pi(a|S_{t+1})Q(S_{t+1}, a) - Q(S_t, A_t)
  \]
  without any correction.

- **Average of both:**
  \[
  \delta_t = R_{t+1} + \gamma \sigma Q(S_{t+1}, A_{t+1}) + \gamma (1 - \sigma)V(S_{t+1}) - Q(S_t, A_t)
  \]
  \[
  = R_{t+1} + \gamma V(S_{t+1}) + \gamma \sigma \left( Q(S_{t+1}, A'_{t+1}) - V(S_{t+1}) \right) - Q(S_t, A_t)
  \]
  with $A'_{t+1}$ drawn using $\pi$. 
Off-Policy Algorithm

Off-Policy Correction

- Replace any estimate of the gain by an importance-sampling corrected one.
- Works well for prediction.
- Can be combined with policy improvement (à la SARSA) but less (no?) theoretical guarantees.
Retrace(\(\lambda\))

\[
\tilde{T} Q(s, a) = Q(s, a) + \mathbb{E}_b \left[ \sum_{t=0}^{\infty} \gamma^t \left( \prod_{t'=1}^{t} c_{t'} \right) \delta_t \mid S_0 = s, A_0 = a \right]
\]

\[
c_t = c(A_t, S_t, A_{t-1}, S_{t-1}, \ldots, A_0, S_0)
\]

\[
\mathbb{E}_b[\delta_t \mid S_t, A_t] = \mathbb{E}[R_{t+1} + \gamma \mathbb{E}_\pi[Q(S_{t+1}, \cdot)] - Q(S_t, A_t) \mid S_t, A_t]
\]

Generic Off-Policy Algorithm

- Generic off-line algorithm including
  - Importance sampling: \(c_t = \rho_t = \pi(A_t \mid S_t)/b(A_t \mid S_t)\)
  - TB(\(\lambda\)): \(c_t = \lambda \pi(A_t \mid S_t)\)
  - Retrace(\(\lambda\)): \(c_t = \lambda \min(1, \pi(A_t \mid S_t)/b(A_t \mid S_t))\)

- **Prop**: \(Q_\pi\) is a fixed point as \(\mathbb{E}_b[\delta_t \mid S_t, A_t] = \mathbb{E}[\tilde{T}^\pi Q(S_t, A_t) - Q(S_t, A_t) \mid S_t, A_t].\)

- **Prop**: \(\tilde{T}\) is a contraction provided \(c_t \leq \rho_t = \pi(A_t \mid S_t)/b(A_t \mid S_t)\).

- Convergence for Importance sampling, TB(\(\lambda\)) and Retrace(\(\lambda\)) for any \(b\).

- Partial results for policy improvement under more assumption.

- For \(Q(\lambda)\), \(c_t = \lambda\), convergence if \(||\pi(s) - b(s)||_1 \leq \epsilon\) and \(\lambda \leq (1 - \gamma)/(\gamma \epsilon)\).
Outline

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
Q-Learning vs SARSA

How different are they?

- In Q-learning, the exploratory policy used is decoupled from the optimized policy.
- This exploratory policy may yield low rewards on average.
- In SARSA, the two policies are linked with the hope on having higher rewards during the learning step.
- Subtle different behavior even if we modify the exploratory policy in Q-Learning.
### Exploration vs Exploitation

- **Exploration**: explore new policies to be able to discover the best ones.
- **Exploitation**: use good policies to obtain a good return.
- Exploration is a requirement.

- No tradeoff if we optimize only the final result!
- Tradeoff between the two if we consider that the returns during training matters!
- Q-learning use the first approach and SARSA try to tackle the second.
- Tradeoff if we study a regret:
  \[
  \sum_t \mathbb{E}_{n_{\star}} [R_t] - \mathbb{E}_{n_t} [R_t]
  \]
  which forces us to be good as fast as possible.
- No natural definition in the discounted setting.
Bandits

\[ S = \{0\} \quad \text{and} \quad A = \{1, \ldots, k\} \quad \text{and} \quad r(s, a) = r_a \]

**Bandits**

- Very simple toy model where there is only one state!
- Optimal policy: pick \( a_\star \in \arg\max r_a \).
- Q estimation: estimate \( r_a \) by playing action \( a \).
- Strategy:
  - Every arm has to be played until we are sure they are bad.
  - Best arm should be played as often as possible to maximise the rewards during the learning phase.
- Simple enough setting to obtain result on the regret

\[
r_T = \sum_{t \leq T} (r_{a_\star} - R_t)
\]

- We will use \( \Delta_a = r_{a_\star} - r_a \) and assume that \( R|a \) is 1-subgaussian.
Explore Then Commit

Explore Then Commit (Random Exploration)

- Play the arm successively during $Km$ steps and then play the optimal one during $T - Km$ steps.

- **Prop:**

  $$r_T \leq \min(m, T/K) \sum_{a=1}^{k} \Delta(a) + \max(T - mK, 0) \sum_{a=1}^{k} \Delta(a) \exp\left(-m\Delta(a)^2/4\right)$$

  Furthermore,

  $$\mathbb{P}(a_T = a^*) \geq 1 - \sum_{a \neq a^*} \exp\left(-m\Delta(a)^2/4\right)$$
**ε-greedy Strategy**

- Estimate $Q(a) = r_a$ by MC:
  
  $$Q_t(a) = \frac{\sum_{t'=1}^{t-1} 1_{A_{t'}=a} R_{t'}}{\sum_{i=1}^{t-1} 1_{A_{t'}=a}}$$

- Pick arm $a$ at time $t$ using
  
  $$\pi(a) = \begin{cases} 
  \frac{\epsilon_t}{k} + (1 - \epsilon) & \text{if } a = \text{argmax}_{a'} Q_t(a') \text{ (only the smallest if necessary)} \\
  \frac{\epsilon_t}{k} & \text{otherwise}
  \end{cases}$$

- Prop:
  
  $$r_T \geq \sum_{t=1}^{T} \frac{\epsilon_t}{k} \sum_{a=1}^{k} \Delta(a)$$
**Prop:**

\[ \mathbb{P}(A_T = a_*) \geq 1 - \epsilon_T - \sum_t \exp(-\Sigma_T/(6k)) - \sum_{a \neq a_*} \frac{4}{\Delta(a)^2} e^{-\Delta(a)^2 \Sigma_T/(4k)} \]

with \(\Sigma_T = \sum_{s=1}^{T} \epsilon_s\).

Furthermore,

\[ \mathbb{P}(a_* = \text{argmax } Q_{T,a}) \geq 1 - \sum_t \exp(-\Sigma_T/(6k)) - \sum_{a \neq a_*} \frac{4}{\Delta(a)^2} e^{-\Delta(a)^2 \Sigma_T/(4k)} \]

If \(\epsilon_t = c/t\),

\[ r_T \leq \sum_{a \neq a_*} \left( \Delta(a) \left( c \frac{\log(T) + 1}{k} + C \right) + \frac{4}{\Delta(a)} C' \right) \]

as soon as \(c/(6k) > 1\) and \(c \min_{a \neq a_*} \Delta(a)/4k < 1\).

If \(\epsilon_t = c \log(t)/t\) then

\[ r_T \leq \sum_{a \neq a_*} \left( \Delta(a) \left( c \frac{\log(T)(\log(T) + 1)}{k} + C \right) + \frac{4}{\Delta(a)} C' \right) \]
UCB Strategy

Upper Confidence Bound

- Use an optimistic strategy to pick the best arm
  \[ A_t = \arg\max_a Q_t(a) + \sqrt{\frac{c \log t}{N_t(a)}} \]

- Prop:
  \[ r_n(t) \leq C_c \sum_a \Delta(a) + \sum_a \frac{4c \ln t}{\Delta(a)} \]
  with \( C_c < +\infty \) as soon as \( c > 3/2 \)

Furthermore
  \[ \mathbb{P}(A_t = a_*) \geq 1 - 2kt^{-2c+2} \]
  as soon as \( t \geq \max_a \frac{4c \ln t}{\Delta(a)^2} \).

- Optimal regret!
- Hard to extend to RL setting but shows that \( \epsilon \)-greedy may not be optimal.
## Outline

### 1. Sequential Decisions, MDP and Policies
- Decision Process and Markov Decision Process
- Returns and Value Functions
- Prediction and Planning
- Operations Research and Reinforcement Learning
- Control
- Survey

### 2. Operations Research: Prediction and Planning
- Prediction and Bellman Equation
- Prediction by Dynamic Programming and Contraction
- Planning, Optimal Policies and Bellman Equation
- Linear Programming
- Planning by Value Iteration
- Planning by Policy Iteration

### 3. Reinforcement Learning: Prediction and Planning in the Tabular Setting
- Prediction with Monte Carlo
- Planning with Monte Carlo
- Prediction with Temporal Differences
- Link with Stochastic Approximation
- Planning with Value Iteration
- Planning with Policy Improvement
- Exploration vs Exploitation

### 4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
- n-step Algorithms
- Eligibility Traces
- Off-policy vs on-policy
- Bandits
- Model Based Approach

### 5. Reinforcement Learning: Approximation of the Value Functions
- Approximation Target(s)
- Gradient and Pseudo-Gradient
- Linear Approximation and LSTD
- On-Policy Prediction and Control
- Off-Policy Prediction and Control
- Two-Scales Algorithms
- Deep Q Learning
- Continuous Actions

### 6. Reinforcement Learning: Policy Approach
- Policy Gradient Theorems
- Monte Carlo Based Policy Gradient
- Actor / Critic Principle
- 3 SOTA Algorithms
- Average Return

### 7. References
Model Based Approach

- Use the interactions to learn a model...
- that can be used to learn a good policy.
- This model can be:
  - a MDP,
  - a simulator.
- Often easier to obtain a simulator.
Estimated MDP: back to OR

- MDP can be estimated from trajectories.
- Simple (but maybe slow) even in an off-line setting.
- Once we have an estimated MDP, prediction and planning can be done using OR.
- Implicitly done by TD(0) when doing several passes.
- Model should be checked/improved as much as possible when new trajectories arrive.
Model based and RL

Estimated Simulator: back to RL

- Simulator can be estimated from trajectories.
- Simple (but maybe slow) even in an off-line setting.
- Once we have an estimated simulator, prediction and planning can be done using RL.

- Model should be checked/improved as much as possible when new trajectories arrive.
Model Free and Model Based Approach

- Combine true interactions with simulated ones.
- Simultaneous acting, model learning, OR learning and RL learning.
- Search for a tradeoff between the (slow) learning RL algorithm and the (wrong) model OR algorithm.
- Need to deal with schedule!

Dyna

MDP
- Env.
- Agent

\[ S_t, A_t, R_{t+1}, S_{t+1} \]

Value Functions
- Policy
- Agent Policies
- Final Policy

Model MDP
Model Env.
Model Agent

Model Interaction
Model Replay Buffer

Interactions
Replay Buffer

Combine true interactions with simulated ones.
Simultaneous acting, model learning, OR learning and RL learning.
Search for a tradeoff between the (slow) learning RL algorithm and the (wrong) model OR algorithm.

Need to deal with schedule!
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differentials
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Replay Buffer and Prioritized Sweeping

- Can we reuse previous interactions?
- In which order?
Replay Buffer

- Store previous interactions (trajectories) in a first-in first-out buffer.
- Draw a subsequence from those interactions (trajectories) and use it in a RL algorithm:
  - On-line: if the trajectory comes from the same policy.
  - Off-line: if the trajectory comes from a different policy.
- Similar to a simulator but no arbitrary choice of state or action.
- Often use with on-line algorithm if the policy has only mildly evolved...
Prioritized Sweeping

- Plain Replay Buffer: subsequence drawn uniformly.
- Prioritized Sweeping: subsequence drawn favoring states with large temporal differences.

- Can be combined with a model approach.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differencing
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - \( n \)-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Real Time Planning

6. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

7. References
Real Time Planning

- Can we optimize the policy at the current state?
- Do we need to optimize it everywhere?
- What is required?

- Planning at decision time...
Real-Time Dynamic Programming

- Warmup in Dynamic Programming...

**RT DP**

- Use trajectories to sample the states to update.
- Convergence holds with exploratory policy.
- Optimal policy does not require to specify the action in irrelevant states.
- Convergence holds even without full exploration in some specific cases!

- In practice, seems to be computationally efficient.
Planning At Decision Time

Can we find a good action $A_t$ at $S_t$... without having it precomputed?

Policy Improvement

$A_t = \arg\max Q_t(S_t, \cdot)$

can be seen as a first step.

How to go deeper?

- A model or a simulator will be required!
Heuristic Search

- Requires the knowledge of the MDP and of a heuristic based value function \( V \).

- Strategy:
  - Build a limited depth tree by stopping after a few steps and at some specific states.
  - Backup the heuristic based value function using Dynamic Programming (Optimal Bellman operator).
  - Pick the action having the highest value.

- The deeper the better... but the more expensive due to branching!

- Requires a suitable heuristic...
Rollout Algorithm

Rollout Policy
- Use a MC estimate with a default policy instead of a heuristic.
- Backup those estimates using Dynamic Programming.
- Simulation can even start after the first action (as in Policy Improvement).
- The values are (most of the time) discarded for the next state.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
Monte Carlo Tree Search

- Simultaneous tree growing, rollout and backup by DP.
- Repeat 4 steps:
  - Selection of a sequence of actions according to the current values with a tree policy.
  - Expansion of the tree at the last node without values.
  - Simulation with a rollout policy to estimate the values at this node.
  - Backup of the value by relaxed Dynamic Programming.
- MCTS focuses on promising paths using a UCB approach.
# Outline

1. **Sequential Decisions, MDP and Policies**
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2. **Operations Research: Prediction and Planning**
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3. **Reinforcement Learning: Prediction and Planning in the Tabular Setting**
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. **Reinforcement Learning: Advanced Techniques in the Tabular Setting**
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. **Reinforcement Learning: Approximation of the Value Functions**
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. **References**
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

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References

Reinforcement Learning: Approximation of the Value Functions

- Replay Buffer and Prioritized Sweeping
- Real Time Planning
Approximated Value Functions

\[ v(s) \longrightarrow v_w(s) \]
\[ q(s, a) \longrightarrow q_w(s, a) \]

Parametric Model

- Reduce dimensionality by storing \( w \) instead of all the values.
- Linear: \( V_w(s) = \langle \Phi(s), w \rangle \) and \( Q_w(s, a) = \langle \Phi(s, a), w \rangle \)
  - \( \Phi(s) \) and \( \Phi(s, a) \) are features associated to the states(-actions).
  - Tabular setting corresponds to \( (\Phi)_{s'(a')}((s, a)) = 1_{s'=s, a'=a} \).
  - Often used in theoretical analysis.
- Deep Learning: \( V_w(s) = \text{NN}_w(\Phi(s)) \) and \( Q_w(s, a) = \text{NN}_w(\Phi(s, a)) \)
  - \( \text{NN} \) 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

\[ v_\pi(s) \simeq V_{w_\pi}(s) \]
\[ q_\pi(s, a) \simeq Q_{w_\pi}(s, a) \]
\[ \arg\max_a q_\pi(s, a) \simeq \arg\max_a Q_{w_\pi}(s, a) \]

\[ v_*(s) \simeq V_{w_*}(s) \]
\[ q_*(s, a) \simeq Q_{w_*}(s, a) \]
\[ \arg\max_a q_*(s, a) \simeq \arg\max_a Q_{w_*}(s, a) \]

Approximated Value Functions Usage

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

\[ v_{\pi}(s) \approx V_{w_{\pi}}(s) \]
\[ q_{\pi}(s, a) \approx Q_{w_{\pi}}(s, a) \]
\[ \text{argmax } q_{\pi}(s, a) \approx \text{argmax } Q_{w_{\pi}}(s, a) \]

\[ v_{\ast}(s) \approx V_{w_{\ast}}(s) \]
\[ q_{\ast}(s, a) \approx Q_{w_{\ast}}(s, a) \]
\[ \text{argmax } q_{\ast}(s, a) \approx \text{argmax } Q_{w_{\ast}}(s, a) \]

Approximation Quality Norm

- Ideal loss:
  \[ \| v - V_{W} \|_{\infty} \quad \text{or} \quad \| q - Q_{W} \|_{\infty} \]
  as this is the error used in all the previous analysis.

- Practical loss:
  \[ \| v - V_{W} \|_{\mu,p}^p = \sum_{s} \mu(s) |v(s) - V_{W}(s)|^p \]
  \[ \text{or} \quad \| q - Q_{W} \|_{\mu,p}^p = \sum_{s,a} \mu(s, a) |q(s, a) - Q_{W}(s, a)|^p \]
  often with \( p = 2 \) and \( \mu \) related to the behavior policy.
Approximation Target(s)

\[ q(s, a) = \mathcal{T}q(s, a) \sim Q_w(s, a) \rightarrow \begin{cases} \| q - Q_w \|_{\mu,p} \text{ small} \\ \| \mathcal{T}Q_w - Q_w \|_{\mu,p} \text{ small} \end{cases} \]

Approximation Targets(s)

- Direct measurement.
- Bellman residual error.

Extended Measurement

- Projection (with linear parametrization): \( \| P_\phi (\mathcal{T}Q_w - Q_w) \|_{\mu,p} \text{ small} \)
- Probes \( Z \):

\[ \mathbb{E}_Z [ | \langle \mathcal{T}Q_w - Q_w, Z \rangle |^p ] \]

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

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
     - Linear Approximation and LSTD
     - On-Policy Prediction and Control
     - Off-Policy and Deadly Triad
     - Two-Scales Algorithms
     - Deep Q Learning
     - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
Prediction, Approximation and Gradient Descent

\[
\min_w \sum_{s,a} \mu_\pi(s, a)|q_\pi(s, a) - Q_w(s, a)|^2
\]

- Prediction objective:
  \[
  \bar{\text{VE}}(w) = \sum_q \mu_\pi(s, a)|q_\pi(s, a) - Q_w(s, a)|^2
  \]

- Gradient:
  \[
  \nabla \bar{\text{VE}}(w) = -2 \sum_{s,a} \mu_\pi(s, a)(q_\pi(s, a) - Q_w(s, a)) \nabla Q(s, a)
  \]

- Stochastic gradient:
  \[
  \hat{\nabla \bar{\text{VE}}}(w) = -2 (q_\pi(S_t, A_t) - Q_w(S_t, A_t)) \nabla Q_w(S_t, A_t)
  \]

- Not a practical algorithm as \(q_\pi\) is unknown.
**Prediction, Approximation and MC**

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

**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
  \[
  \mathbb{E}_\pi[(G_t - Q_{\mathbf{w}_\infty}(S_t, A_t))\nabla Q_{\mathbf{w}_\infty}(S_t, A_t)] \\
  = \mathbb{E}[(q_\pi(S_t, A_t) - Q_{\mathbf{w}_\infty}(S_t, A_t))\nabla Q_{\mathbf{w}_\infty}(S_t, A_t)] = 0
  \]
- 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:** \( \mathbb{E}_\pi[q_\pi(S_t)\Phi(S_t, A_t)] = \mathbb{E}_\pi[\Phi(S_t, A_t)\Phi(S_t, A_t)^\top] \mathbf{w}_\infty \)
Prediction, Approximation and TD

\[ w_{t+1} = w_t + 2\alpha_t (R_{t+1} + \gamma Q_w(S_{t+1}, A_{t+1}) - Q_w(S_t, A_t)) \nabla Q_w(S_t, A_t) \]

Temporal Differences Approach

- Replace \( q_\pi(S_t, A_t) \) by \( R_{t+1} + \gamma Q_w(S_{t+1}, A_{t+1}) \).
- Not a Stochastic Gradient of the original problem but a Stochastic Approximation algorithm with limit (if it exists) satisfying
  \[
  \mathbb{E}_\pi [(R_t + \gamma Q_{w_\infty}(S_{t+1}, A_{t+1}) - Q_{w_\infty}(S_t, A_t)) \nabla Q_{w_\infty}(S_t, A_t)] \\
  = \mathbb{E}_\pi [(T^\pi Q_{w_\infty} - Q_{w_\infty})(S_t, A_t)) \nabla Q_{w_\infty}(S_t, A_t)] = 0
  \]
- 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.
\[ w_{t+1} = w_t + 2\alpha_t \left( \tilde{G}_t - Q_{w_t}(S_t, A_t) \right) \nabla Q_{w_t}(S_t, A_t) \]

**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
  \[
  \mathbb{E}_\pi \left[ \left( \tilde{G}_t - Q_{w_t}(S_t, A_t) \right) \nabla Q_{w_\infty}(S_t, A_t) \right] \\
  = \mathbb{E}_\pi \left[ \left( (\tilde{T}_\pi Q_{w_\infty} - Q_{w_\infty})(S_t, A_t) \right) \nabla Q_{w_\infty}(S_t, A_t) \right] = 0
  \]
- 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

\( z_t = \gamma^t \lambda z_{t-1} + \nabla Q_{w_t}(S_t, A_t) \)

\( \delta_t = R_{t+1} + \gamma Q_{w_t}(S_{t+1}, A_{t+1}) - Q_{w_t}(S_t, A_t) \)

\( w_{t+1} = w_t + \alpha_t \delta_t z_t \)

Eligibility Trace

- Rewrite the TD(\( \lambda \)) 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
  \[ E_\pi[(R_{t+1} + \gamma Q_w(S_{t+1}, A_{t+1}) - Q_w(S_t, A_t)) z_t] \]
  \[ = E_\pi[(T^\pi - Q_w)(S_t, A_t)z_t] = 0 \]
- No simple argument to justify the convergence.
Outline

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
Linear Parametrization

\[ Q_w(S_t, A_t) = \Phi(S_t, A_t)^T w \quad \text{and} \quad \nabla Q_w(S_t, A_t) = \Phi(S_t, A_t) \]

- Extension of the tabular setting.
- Derivative is independent of \( w \).
- 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

Iteration: \( w_{t+1} = w_t + \alpha_t (G_t - \Phi(S_t, A_t)^\top w_t) \Phi(S_t, A_t) \)

Limiting equation: \( \mathbb{E}_\pi[q_\pi(S_t, A_t)\Phi(S_t, A_t)] = \mathbb{E}_\pi[\Phi(S_t, A_t)\Phi(S_t, A_t)^\top] w_\infty \)

ODE: \( \frac{dw}{dt} = -\mathbb{E}_\pi[\Phi(S_t, A_t)\Phi(S_t, A_t)^\top] (w - w_\infty) \)

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

Iteration: \( w_{t+1} = w_t + \alpha_t (R_{t+1} + \gamma \Phi(S_{t+1}, A_{t+1})^T w_t - \Phi(S_t, A_t)^T w_t) \Phi(S_t, A_t) \)

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

ODE: \( \frac{dw}{dt} = -\mathbb{E}_\pi \left[ \Phi(S_t, A_t) \left( \Phi(S_t, A_t)^T - \gamma \Phi(S_{t+1}, A_{t+1})^T \right) \right] (w - w_\infty) \)

Linear Parametrization and TD

- Convergence of ODE if \( \mathbb{E}_\pi \left[ \Phi(S_t, A_t) \left( \Phi(S_t, A_t)^T - \gamma \Phi(S_{t+1}, A_{t+1})^T \right) \right] \) 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: \( \bar{VE}(w_{TD}) \leq \frac{1}{1 - \gamma} \bar{VE}(w_{MC}) = \frac{1}{1 - \gamma} \min_w \bar{VE}(w) \)
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:
  \[ \mathbf{w}_\infty = A^{-1} b \]

- Much more sample efficient.
- Recursive implementation possible.
- Recursive implementation maintaining an estimate of \( A^{-1} \) is also possible.
Advanced Returns

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

Iteration: $w_{t+1} = w_t + \alpha_t (\tilde{R}_t + \tilde{\Phi}_t^T w_t - \Phi(S_t, A_t)^T w_t) \Phi(S_t, A_t)$

Lim. eq.: $E_\pi[\tilde{R}_t \Phi(S_t, A_t)] = E_\pi[\Phi(S_t, A_t) (\Phi(S_t, A_t)^T - \Phi_t^T)] w_\infty$

ODE: $\frac{d w}{d t} = -E_\pi[\Phi(S_t, A_t) (\Phi(S_t, A_t)^T - \Phi_t^T)] (w - w_\infty)$

Linear Parametrization and TD

- Convergence of ODE if $E_\pi[\Phi(S_t, A_t) (\Phi(S_t, A_t)^T - \Phi_t^T)]$ has complex eigenvalues with positive real parts...
- which can be proved to be true for the advanced returns under an ergodicity assumption!
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
On-Policy Prediction

\[ w_{t+1} = w_t + 2\alpha_t \left( \tilde{G}_t - Q_{w_t}(S_t, A_t) \right) \nabla Q_{w_t}(S_t, A_t) \]

**On-line TD Algorithm**

- Use the policy \( \Pi \) 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 if the episodes are short.
- Similar observations with eligibility trace.
On-Policy Control

\[ w_{t+1} = w_t + 2\alpha_t \left( \tilde{G}_t - Q_{w_t}(S_t, A_t) \right) \nabla Q_{w_t}(S_t, A_t) \]

\[ \pi_{t+1}(s) = \arg\max Q_{w_t}(s, \cdot) \quad \text{(plus exploration)} \]

**On-Policy Control**

- SARSA type algorithm: update Q values and policy \( \pi \) while using policy \( \pi \).
- 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

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
     - Off-Policy and Deadly Triad
       - Two-Scales Algorithms
       - Deep Q Learning
       - Continuous Actions

6 Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7 References
On-Policy vs Off-Policy

- **On-Policy**: the policy \( b \) used to interact is the same than the policy \( \Pi \) evaluated or optimized.

- **Off-Policy**: the policy \( b \) used to interact may be different from the policy \( \Pi \) evaluated or optimized.

- Off-Policy correction available for the return.
Off-Policy Prediction

\[ w_{t+1} = w_t + \alpha_t \left( \tilde{G}_t - Q_{w_t}(S_t, A_t) \right) \nabla Q_{w_t}(S_t, A_t) \]

**Off-policy TD Algorithm**

- Use a policy \( b \) to obtain the interactions \( S_tA_tR_{t+1}S_{t+1}A_{t+1} \ldots \)
- 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:
  \[ \delta_t = R_{t+1} + \gamma V_w(S_{t+1}) - V_w(S_t) \]
  \[ = 0 + \gamma 2w_t - w_t = (2\gamma - 1)w_t \]

- Off-policy semi-gradient TD(0) update:
  \[ w_{t+1} = w_t + \alpha_t \rho_t \delta_t \nabla V(S_{t+1}, w_t) \]
  \[ = w_t + \alpha_t \times 1 \times (2\gamma - 1)w_t = (1 + \alpha_t(2\gamma - 1))w_t \]

- Explosion if this transition is explored without \( w \) being update on other transitions as soon as \( \gamma > 1/2 \).
Baird’s Counterexample

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

Tsistiklis and Van Roy’s Counterexample

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

  $$w_{t+1} = \text{argmin}_w \sum_s (V_w(s) - \mathbb{E}_\pi[R_{t+1} + \gamma V_w(S_{t+1})|S_t = s])^2$$

  $$= \text{argmin}_w (w - \gamma 2w_t)^2 + (2w - (1 - \epsilon)\gamma 2w_t)^2$$

  $$= \frac{6 - 4\epsilon}{5} \gamma w_t$$

- Divergence if $\gamma > \frac{5}{6 - 4\epsilon}$. 
Linear Parametrization and TD

Iteration: \( w_{t+1} = w_t + \alpha_t (R_{t+1} + \gamma \sum_a \pi(a|S_{t+1})\Phi(S_{t+1}, a)^\top w_t - \Phi(S_t, A_t)^\top w_t) \Phi(S_t, A_t) \)

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

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

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 \( \Phi = (\Phi(s, a)) \), \( \Xi = \text{diag}(\mu(s, a)) \) and \( P^\pi \) the transition matrix associated to \( \pi \)) has complex eigenvalues with positive real parts.

- Proof for on-policy relies on \( \mu = \mu^\pi \) which satisfies \( \mu^\pi^\top P^\pi = \mu^\pi^\top \).

- Not true anymore with an arbitrary behavior policy!
**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?
Objective?

Linear Parametrization Target?

- Prediction objective $\overline{VE}$:
  \[ \| q_\pi - Q_w \|_\mu^2 \]

- Bellman Error $\overline{BE}$:
  \[ \| T^\pi Q_w - Q_w \|_\mu^2 \]

- Projected Bellman Error $\overline{PBE}$:
  \[ \| \text{Proj} \ T^\pi Q_w - Q_w \|_\mu^2 \]
  with $\text{Proj} = \Phi(\Phi^\top \Xi \Phi)\Phi (\Phi) \Xi$. 

Reinforcement Learning: Approximation of the Value Functions
Prediction Objective

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

$$\bar{\text{RE}}(w) = \mathbb{E} \left[ (G_t - V_{w_t}(S_t))^2 \right] = \bar{\text{VE}}(w) + \mathbb{E} \left[ (G_t - v_{\pi}(S_t))^2 \right]$$

- MC method target.
Bellman Error

- Two MRP with the same outputs (because of approximation).
- Different $BE$.
- Different minimizer!
- $BE$ is not learnable!
Reinforcement Learning: Approximation of the Value Functions

**TD Error**

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

**Mean-Squares TD Error**

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

\[ \| \text{Proj} \mathcal{T}^\pi Q_w - Q_w \|_\mu^2 \text{ with Proj} = \Phi(\Phi^\top \Xi \Phi)^{-1} \Phi^\top \Xi. \]

**Projected Bellman Error**

- **Rewriting**
  \[
  \overline{PBE}(w) = \| \text{Proj} \mathcal{T}^\pi q_w - q_w \|_\mu^2 = \| \text{Proj} \delta_w \|_\mu^2
  \]
  \[
  = (\text{Proj} \delta_w)^\top \Xi (\text{Proj} \delta_w) = (\Phi^\top \Xi \delta_w)^\top (\Phi^\top \Xi \Phi)^{-1} (\Phi^\top \Xi \delta_w)
  \]

- **Gradient:**
  \[
  \nabla \overline{PBE}(w) = 2 \nabla (\Phi^\top \Xi \delta_w)^\top (\Phi^\top \Xi \Phi)^{-1} (\Phi^\top \Xi \delta_w)
  \]

- **Expectations:**
  \[
  \Phi^\top \Xi \delta_w = \mathbb{E}_b[\rho_t \delta_t \Phi(S_t, A_t)]
  \]
  \[
  \nabla (\Phi^\top \Xi \delta_w)^\top = \mathbb{E}_b\left[\rho_t (\gamma \Phi(S_{t+1}, A_{t+1}) - \Phi(S_t, A_t))\Phi(S_t, A_t)^\top\right]
  \]
  \[
  \Phi^\top \Xi \Phi = \mathbb{E}_b\left[\Phi(S_t, A_t)\Phi(S_t, A_t)^\top\right]
  \]

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

Gradient and Stochastic Approximation

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

- Least-squares inside:
\[
\nu = \left( \mathbb{E}_b \left[ \Phi(S_t, A_t) \Phi(S_t, A_t)^\top \right] \right)^{-1} \mathbb{E}_b [\rho_t \delta_t \Phi(S_t, A_t)^\top] \]
\[
\Leftrightarrow \nu = \arg\min_v \mathbb{E}_b \left[ (\Phi(S_t, A_t)^\top v - \rho_t \delta_t)^2 \right]
\]
which can be estimated by
\[
\nu_{t+1} = \nu_t + \beta_t \Phi(S_t, A_t) (\delta_t - \rho_t \Phi(S_t, A_t)^\top \nu_t)
\]

- Plugin pseudo gradient (SA):
\[
w_{t+1} = w_t - 2\alpha_t \rho_t (\gamma \Phi(S_{t+1}, A_{t+1}) - \Phi(S_t, A_t)) \Phi(S_t, A_t)^\top \nu_t
\]

- Same target than Pseudo Gradient but converging algorithm provided \(\alpha_t \ll \beta_t\).
Gradient TD Algorithm

**GTD**

- Simultaneous update:
  \[ v_{t+1} = v_t + \beta_t \Phi(S_t, A_t)(\delta_t - \rho_t \Phi(S_t, A_t)^\top v_t) \]
  \[ w_{t+1} = w_t - 2\alpha_t \rho_t (\gamma \Phi(S_{t+1}, A_{t+1}) - \Phi(S_t, A_t))\Phi(S_t, A_t)^\top v_t \]
- As \( \alpha_t \ll \beta_t \), \( w \) is seen as constant by \( v \)...

**TDC**

- Simultaneous update:
  \[ v_{t+1} = v_t + \beta_t \Phi(S_t, A_t)(\delta_t - \rho_t \Phi(S_t, A_t)^\top v_t) \]
  \[ w_{t+1} = w_t - 2\alpha_t \rho_t (\delta_t \Phi(S_t, A_t) - \gamma \Phi(S_{t+1}, A_{t+1}))\Phi(S_t, A_t)^\top v_t \]
- Obtained by a similar derivation but faster in practice...
- As \( \alpha_t \ll \beta_t \), \( w \) is seen as constant by \( v \)...
- Restricted to the linear setting but interesting insight.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
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:
  - \( \mathbb{E}[\varepsilon_k] = 0, \) \( \text{Var}[\varepsilon_k] < \sigma^2, \) 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 \( \alpha \) 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 \( \frac{d\tilde{\theta}}{dt} = H(\tilde{\theta}) \)

ODE Approach

- General proof showing that the algorithm converges provided the ODE converges.
- Rely on the rewriting the equation \( \frac{\theta_{k+1} - \theta_k}{\alpha_k} = h_k(\theta_k) = H(\theta_k) + \epsilon_k + \eta_k \)
- \( \alpha_k \) can be interpreted as a time difference allowing to define a time \( t_k = \sum_{t' \leq t} \alpha_k \).
- Equation be interpreted as the derivative at time \( t \in (t_k, t_{k+1}) \) of a piecewise affine function \( \theta(t) \).
- This piecewise function remains close to any solution of the ODE starting from \( \theta_k \) for an arbitrary amount of time provided \( k \) is large enough.
Stochastic Approximation

\[
\begin{align*}
\theta_{k+1} &= \theta_k + \alpha_k h_k(\theta_k, \nu_k) \\
\nu_{k+1} &= \nu_k + \beta_k g_k(\theta_k, \nu_k)
\end{align*}
\]  
with  
\[
\begin{align*}
h_k(\theta, \nu) &= H(\theta, \nu) + \epsilon_k + \eta_k \\
g_k(\theta, \nu) &= G(\theta, \nu) + \epsilon'_k + \eta'_k
\end{align*}
\]

\[\Rightarrow \theta_k \to \{\theta, H(\theta, \nu(\theta)) = 0, \nu(\theta) \in \{\nu, G(\theta, \nu) = 0\}\}\]

Stochastic Approximation

- Family of sequential stochastic algorithm converging to a zero of a function.
- Classical assumptions:
  - \(\mathbb{E}[\epsilon_k] = 0, \forall \text{Var} [\epsilon_k] < \sigma^2, \text{and } \mathbb{E}[||\eta_k||] \to 0\),
  - \(\sum_k \alpha_k \to \infty \text{ and } \sum_k \alpha_k^2 < \infty\),
  - \(\sum_k \beta_k \to \infty \text{ and } \sum_k \beta_k^2 < \infty\),
  - \(\alpha_k / \beta_k \to 0 \text{ (two-scales assumption)}\),
  - 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
\[\begin{align*}
\theta_{k+1} &= \theta_k + \alpha_k h_k(\theta_k, \nu_k) \\
\nu_{k+1} &= \nu_k + \beta_k + g_k(\theta_k, \nu_k)
\end{align*}\]

with
\[\begin{align*}
h_k(\theta, \nu) &= H(\theta, \nu) + \epsilon_k + \eta_k \\
g_k(\theta, \nu) &= G(\theta, \nu) + \epsilon'_k + \eta'_k
\end{align*}\]

to
\[\frac{d\tilde{\theta}}{dt} = H(\tilde{\theta}, \tilde{\nu}(\tilde{\theta})) \quad \text{with} \quad \tilde{\nu}(\theta) \text{ the limit of } \frac{d\tilde{\nu}}{dt} = G(\theta, \tilde{\nu})\]

**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

1 Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

2 Operations Research: Prediction and Planning
   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3 Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4 Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5 Reinforcement Learning: Approximation of the Value Functions
   - Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting

6 Reinforcement Learning: Policy Approach
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

7 References

Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

Reinforcement Learning: Policy Approach
   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

References
Simplified Deep Q-Learning

$$w_{t+1} = w_t + \beta_t (R_{t+1} + \gamma \max_a Q_{\nu_t}(S_{t+1}, a) - Q_{w_t}(S_t, A_t)) \nabla Q_{w_t}(S_t, A_t)$$

$$\nu_t = w_{\lceil t/T \rceil T}$$

**Simplified Deep Q-Learning**

- Stochastic Approximation for a fixed $\nu$:
  - Limiting equation:
    $$\mathbb{E}[T^*Q_\nu(S_t, A_t) - Q_{w_\infty}(S_t, A_t)] = 0$$
  - Stochastic Gradient Descent of
    $$\mathbb{E}[T^*Q_\nu(S_t, A_t) - Q_{w}(S_t, A_t)]^2$$
  - $Q_w \to T^*Q_\nu$
- Approximate Value Iteration Scheme!

- Two-scales algorithm flavour as $\nu$ is kept constant.
- Explicit two scales with $\nu_{t+1} = \nu_t + \alpha_t(w_t - \nu_t)$ 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**

\[ w_{t+1} = w_t + \beta_t (R_t + \gamma \max_a Q_{\nu_t}(S_{t+1}, a) - Q_w(S_t, A_t)) \nabla Q_w(S_t, A_t) \]

\[ \nu_t = w[^{\lfloor t/T \rfloor} T] \]

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

### 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 $D$ 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 $D$.
- Use random samples of the buffer $D_t$ (more than one per interaction is OK).
Deep Q-Learning

\[ w_{t+1} = w_t + \beta_t (R_t + \gamma \max_a Q_{\nu_t}(S_{t+1}, a) - Q_w(S_t, A_t)) \nabla Q_w(S_t, A_t) \]

\[ \nu_t = w_{\lfloor t/T \rfloor T} \]

Plus tricks

### Deep Q-Learning Tricks

- Replay buffer
- Double Q-Learning
- Better Exploration
- Advanced Return and Distributional
- Network Architecture
- Rainbow paper...
### 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 an 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 close 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: $R_{s,a} + \max_{a'} Q_w(s', a')$
- Approximation issue: $Q_w(s', a') \sim Q(s, a) + \epsilon(s, a)$
- Consequence: $\mathbb{E}[\max_a Q_w(S_t, a)] \geq \max (Q(s, a) + \mathbb{E}[\epsilon(s, a)])$

Double Q-Learning with two Q functions: $Q_{w_1}$ and $Q_{w_2}$

- Used in a crossed way for the target of $Q_{w_i}$:
  $R_{s,a} + Q_{w_i}(s', \arg\max_{a'} Q_{w_i}(s', a'))$
- Mitigates the bias.

Clipped Q-Learning with several Q functions: $Q_{w_i}$

- Used in a pessimistic way for the target of $Q_{w_i}$:
  $R_{s,a} + \min_{i'} Q_{w_{i'}}(s', \arg\max_{a'} Q_{w_{i'}}(s', a'))$
- Seems even more efficient.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Continuous Action

- Case (almost) not yet considered.
- Most complex theoretical extension.

Prediction

- No algorithmic issue if one can sample $\pi$.
- 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

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Optimize policy directly instead of deriving it from a value function.

Avoid the argmax operator.

Most natural POV?

Pontryagin vs Hamilton-Jacobi(-Bellman) in control!
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy Prediction and Control
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Reinforcement Learning: Policy Approach

Policy and Goal

\[ J_\mu(\pi) = \sum_s \mu(s)v_\pi(s) \]

Goal: average expected return over the states

- Target used to define the linear programming formulation of an optimal policy in the tabular setting.
- \( \mu \) can be the initial distribution of the states (independent of \( \pi \))...
- but may also depends on \( \pi \) (for instance the associated stationary measure)
- Other choices will appear.

- Goal: optimize \( J_\mu(\pi) \) in \( \pi \)! 
Parametric Policy

\[ \pi_\theta(a|s) = \begin{cases} 
\frac{e^{h_\theta(a,s)}}{\sum_a e^{h_\theta(a,s')}} & \text{(softmax)} \\
P_{h_\theta(s)}(a) & \text{(parametric conditional model)} \\
1_{a=h_\theta(s)} & \text{(deterministic)} 
\end{cases} \]

- **Parametric Policy**
  - Restriction of the set of policy to a parametrized one.
  - Most classical parametrizations:
    - Soft-max with a preference function \( h_\theta(a,s) \),
    - Parametric conditional model with parameter \( h_\theta(s) \)
  - To be useful need to be able to sample the distribution.
  - \( h_\theta \): from linear model to deep learning . . .
  - Most of our result will assume that \( \pi_\theta(a|s) \) is differentiable with respect to \( \theta \).

- Deterministic policies will be considered with a different analysis.
Episodic Setting: Gradient of Expected Returns

\[ v_{\pi_\theta}(s) = \mathbb{E}_{\pi_\theta}[G_0|S_0 = s] \]

\[ \nabla_\theta v_{\pi_\theta}(s) = \mathbb{E}_{\pi_\theta} \left[ \left( \sum_{t=0}^{T_\tau - 1} \nabla \log \pi_\theta(A_t|S_t) \right) G_0 \bigg| S_0 = s \right] \]

**Expected Returns**

- Rely on \( v_{\pi_\theta}(s) = \sum_\tau \mathbb{P}_{\pi_\theta}(\tau|S_0 = s) G_0(\tau) \) and

\[ \nabla \mathbb{P}_{\pi_\theta}(\tau|S_0 = s) = \mathbb{P}_{\pi_\theta}(\tau|S_0 = s) \nabla \log \mathbb{P}_{\pi_\theta}(\tau|S_0 = s) \]

\[ = \mathbb{P}_{\pi_\theta}(\tau|S_0 = s) \sum_t (\nabla \log \pi_\theta(A_t|S_t) + \nabla p(R_{t+1}, S_{t+1}|S_t, A_t)) \]

\[ = \mathbb{P}_{\pi_\theta}(\tau|S_0 = s) \sum_t \nabla \log \pi_\theta(A_t|S_t) \]

- In an episodic setting, any trajectory \( \tau \) ends at a finite time \( T_\tau \).
Reinforcement Learning: Policy Approach

**Episodic Setting: Policy Gradient Theorem**

\[
J_{\mu_0}(\pi_\theta) = \sum_s P(S_0 = s) \nu_{\pi_\theta}(s)
\]

\[
\nabla J_{\mu_0}(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ \left( \sum_{t=0}^{T-1} \nabla \log \pi_\theta(A_t|S_t) \right) G_0 \right]
\]

**Policy Gradient Theorem**

- **Natural \(\mu\):** initial state distribution.
- **Gradient is an expectation:** MC type algorithm...
- Can be interpreted as the gradient of a the maximum likelihood of the actions weighted by the return.
- Favors good actions over bad ones.
Baseline and Variance Reduction

\[ J_{\mu_0}(\pi_\theta) = \sum_s P(S_0 = s) \nu_{\pi_\theta}(s) \]

\[ \nabla J_{\mu_0}(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ \left( \sum_{t=0}^{T_{\tau} - 1} \nabla \log \pi_\theta(A_t|S_t) \right) (G_0 - b) \right] \]

Variance Reduction and Baseline

- The previous formulae are valid if one replace \( G_0 \) by any function of \( \tau \).
- For any constant \( b \), this leads to
  \[ \nabla \mathbb{E}_{\pi_\theta}[b] = 0 = \mathbb{E}_{\pi_\theta} \left[ \left( \sum_{t=0}^{T_{\tau} - 1} \nabla \log \pi_\theta(A_t|S_t) \right) b \right] \]
- Optimal value for
  \[ b = \mathbb{E}_{\pi_\theta} \left[ \left( \sum_{t=0}^{T_{\tau} - 1} \nabla \log \pi_\theta(A_t|S_t) \right)^2 \right] G_0 \bigg/ \mathbb{E}_{\pi_\theta} \left[ \left( \sum_{t=0}^{T_{\tau} - 1} \nabla \log \pi_\theta(A_t|S_t) \right)^2 \right] \]
- Most used value \( b = \mathbb{E}_{\pi_\theta}[G_0] \).
Episodic/Discounted: Gradient(s) of Expected Return

\[ v_{\pi_\theta}(s) = \mathbb{E}_{\pi_\theta} \left[ \sum \gamma^t R_t \bigg| S_0 = s \right] \]

\[ \nabla v_{\pi_\theta}(s) = \sum_t \gamma^t \mathbb{E}_{\pi_\theta} \left[ \left( \sum_{t'=0}^{t-1} \nabla \log \pi_\theta(A_{t'}|S_{t'}) \right) R_t \bigg| S_0 = s \right] \]

\[ = \sum_{t'} \mathbb{E}_{\pi_\theta} \left[ \nabla \log \pi_\theta(A_{t'}|S_{t'}) \left( \sum_{t \geq t'} \gamma^t R_t \right) \bigg| S_0 = s \right] \]

\[ = \sum_{t'} \gamma^{t'} \mathbb{E}_{\pi_\theta} \left[ \nabla \log \pi_\theta(A_{t'}|S_{t'}) Q_{\pi_\theta}(S_{t'}, A_{t'}) \bigg| S_0 = s \right] \]

\[ = \sum_{t'} \gamma^{t'} \mathbb{E}_{\pi_\theta} \left[ \nabla \log \pi_\theta(A_{t'}|S_{t'}) \left( Q_{\pi_\theta}(S_{t'}, A_{t'}) - V_{\pi_\theta}(S_{t'}) \right) \bigg| S_0 = s \right] \]

**Expected Returns**

- Several gradients!
Episodic/Discounted: More Gradient(s)

$$\nabla v_{\pi_\theta}(s) = \sum_{t'} \gamma^{t'} E_{\pi_\theta} [\nabla \log \pi_\theta(A_{t'}|S_{t'}) Q_{\pi_\theta}(S_{t'}, A_{t'}) | S_0 = s]$$

$$= \sum_{t'} \gamma^{t'} E_{\pi_\theta} [\nabla \log \pi_\theta(A_{t'}|S_{t'}) A_{\pi_\theta}(S_{t'}, A_{t'}) | S_0 = s]$$

$$= \sum_s \left( \sum_t \gamma^t P_{\pi_\theta}(S_t = s | S_0 = s) \right) \left( \sum_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) Q_{\pi_\theta}(s, a) \right)$$

$$= \sum_s \left( \sum_t \gamma^t P_{\pi_\theta}(S_t = s | S_0 = s) \right) \left( \sum_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) A_{\pi_\theta}(s, a) \right)$$

Focus on states

- More gradients!
Episodic/Discounted: Policy Gradient(s)

\[
J(\mu_0)(\pi_\theta) = \sum_s \mu_0(s) v_{\pi_\theta}(s)
\]

\[
\nabla J_{\mu_0}(\pi_\theta) = \sum_s \left( \sum_t \gamma^t \mathbb{P}_{\pi_\theta}(S_t = s) \right) \left( \sum_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) Q_{\pi_\theta}(s, a) \right) 
\]

\[
= \sum_s \left( \sum_t \gamma^t \mathbb{P}_{\pi_\theta}(S_t = s) \right) \left( \sum_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) (Q_{\pi_\theta}(s, a) - V_{\pi_\theta}(s, a)) \right)
\]

Discounted Setting

- Average (discounted) return from the beginning.
- Focus on early steps in discounted setting...
Policy Improvement Lemma

$$J_{\mu_0}(\pi') - J_{\mu_0}(\pi) = \sum_t \gamma^t P_{\pi'}(S_t = s) \left( \sum_a (\pi'(a|s) - \pi(a|s)) Q_{\pi}(s, a) \right)$$

$$= \sum_t \gamma^t P_{\pi'}(S_t = s) \left( \sum_a (\pi'(a|s) - \pi(a|s)) A_{\pi}(s, a) \right)$$

Proof in the discounted setting rely on

$$v_{\pi'} - v_{\pi} = r_{\pi'} + \gamma P_{\pi'} v_{\pi'} - r_{\pi} - \gamma P_{\pi} v_{\pi}$$

$$= r_{\pi'} - r_{\pi} + \gamma (P_{\pi'} - P_{\pi}) v_{\pi} + \gamma P_{\pi'} (v_{\pi'} - v_{\pi})$$

$$v_{\pi'} - v_{\pi} = (I - \gamma P_{\pi'})^{-1} \left( r_{\pi'} - r_{\pi} + \gamma (P_{\pi'} - P_{\pi}) v_{\pi} \right)$$
Approximate Policy Improvement Lemma

\[
\left| J_{\mu_0}(\pi') - J_{\mu_0}(\pi) - \sum_t \gamma^t P_\pi(S_t = s) \left( \sum_a (\pi'(a|s) - \pi(a|s)) A_\pi(s, a) \right) \right|
\]

\[
= \left| \sum_t \gamma^t (P_{\pi'}(S_t = s) - P_\pi(S_t = s)) \left( \sum_a (\pi'(a|s) - \pi(a|s)) A_\pi(s, a) \right) \right|
\]

\[
\leq \sum_t \gamma^t 2t \max_s \| \pi'(\cdot|s) - \pi(\cdot|s) \|^2_{1} \max_{s,a} |A_\pi(s, a)|
\]

Approximate Policy Improvement Lemma

- If \( \max_s \| \pi'(\cdot|s) - \pi(\cdot|s) \|^1 \leq \epsilon \)
  
  \[
P_{\pi'}(S_t = s) = (1 - \epsilon)^t P_\pi(S_t = s) + (1 - (1 - \epsilon)^t) P_{\text{mistake}}(S_t = s)
  \]
  
  \[
  \rightarrow |P_{\pi'}(S_t = s) - P_\pi(S_t = s)| \leq 2(1 - (1 - \epsilon)^t) \leq 2\epsilon t
  \]
Approximate Policy Improvement Lemma

\[
\left| J_{\mu_0}(\pi') - J_{\mu_0}(\pi) - \sum_t \gamma^t P_{\pi}(S_t = s) \left( \sum_a (\pi'(a|s) - \pi(a|s)) A_{\pi}(s, a) \right) \right| \\
\leq \sum_t \gamma^t 2t \max_s \|\pi'(s) - \pi(s)\|_2 \max_{s,a} |A_{\pi}(s, a)|
\]

Approximate Policy Improvement Lemma and Policy Gradient Theorem

- Let \( \pi' = \pi_{\theta+h} \) and \( \pi_{\theta} \)
  - \( \pi_{\theta+h}(a|s) - \pi_{\theta}(a|s) = \pi_{\theta}(a|s)\langle \nabla \log \pi_{\theta}(a|s), h \rangle + O(\|h\|^2) \)
  - \( \|\pi_{\theta+h}(s|\cdot) - \pi_{\theta}(s|\cdot)\|_1 \leq \|h\| \max_a \|\nabla \log \pi_{\theta}(a|s)\| + O(\|h\|^2) \)
- Implies Policy Gradient Theorem:
  \[
  J_{\mu_0}(\pi_{\theta+h}) \]
  \[
  = J_{\mu_0}(\pi_{\theta}) + \sum_t \gamma^t P_{\pi_{\theta}}(S_t = s) \left( \sum_a \pi_{\theta}(a|s)\langle \nabla \log \pi_{\theta}(s, a), h \rangle A_{\pi}(s, a) \right) + O(\|h\|^2)
  \]
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differencies
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - $n$-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Monte Carlo Approach

Monte Carlo
- Replace every return by an empirical estimate along episodes.
- Need to wait until the end of the episodes.

\[ G_t = \sum_{t' \geq t} R_{t+1} \]

\[ Q_{t,\pi_\theta}(s, a) = \mathbb{E}[G_t | S_t = s, A_t = a] \]
REINFORCE: Monte Carlo Based Policy Gradient

\[ J_{\mu_0}(\pi_\theta) = \sum_s \mathbb{P}(S_0 = s) v_{\pi_\theta}(s) \]

\[ \nabla J_{\mu_0}(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ \left( \sum_{t=0}^{T_\tau-1} \nabla \log \pi_\theta(A_t|S_t) \right) G_0 \right] \]

\[ = \sum_s \left( \sum_t \mathbb{P}_{\pi_\theta}(S_t = s) \right) \left( \sum_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) Q_{\pi_\theta}(s,a) \right) \]

\[ \hat{\nabla} J_{\mu_0}(\pi_\theta) = \left( \sum_{t=0}^{T_\tau-1} \nabla \log \pi_\theta(A_t|S_t) \right) G_0 \quad \text{or} \quad \hat{\nabla} J_{\mu_0}(\pi_\theta) = \sum_t \nabla \log \pi_\theta(A_t|S_t) G_t \]

REINFORCE

- Plain MC (SGD) algorithm.
- Need to wait until the end of the episodes.
- Convergence guarantees (even in off-line setting with importance sampling).
REINFORCE with Baseline

\[ \nabla J_{\mu_0}(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ \sum_{t=0}^{T_\tau-1} \nabla \log \pi_\theta(A_t|S_t) \right] (G_0 - b) \]

\[ = \sum_s \left( \sum_t \mathbb{P}_{\pi_\theta}(S_t = s) \right) \left( \sum_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) (Q_{\pi_\theta}(s, a) - b(s)) \right) \]

\[ \widehat{\nabla J}_{\mu_0}(\pi_\theta) = \left( \sum_{t=0}^{T_\tau-1} \nabla \log \pi_\theta(A_t|S_t) \right) (G_0 - b) \]

or \[ \widehat{\nabla J}_{\mu_0}(\pi_\theta) = \sum_t \nabla \log \pi_\theta(A_t|S_t) (G_t - b(S_t)) \]

REINFORCE with baseline

- Several choices for \( b \)...
- and for \( b(s) \) which can be any function (a crude estimate of \( V_{t,\pi}(s) \) for instance)!
- Convergence guarantees (even in off-line setting with importance sampling).
Discounted REINFORCE?

\[
\nabla J_{\mu_0}(\pi_\theta) = E_{\pi_\theta} \left[ \left( \sum_{t=0}^{T_\tau-1} \nabla \log \pi_\theta(A_t | S_t) \right) (G_0 - b) \right]
\]

\[
= \sum_s \left( \sum_t \gamma^t \mathbb{P}_{\pi_\theta}(S_t = s) \right) \left( \sum_a \pi_\theta(a | s) \nabla \log \pi_\theta(a | s) (Q_{\pi_\theta}(s, a) - b(s)) \right)
\]

\[
\hat{\nabla} J_{\mu_0}(\pi_\theta) = \left( \sum_{t=0}^{T_\tau-1} \nabla \log \pi_\theta(A_t | S_t) \right) (G_0 - b)
\]

or \[
\hat{\nabla} J_{\mu_0}(\pi_\theta) = \sum_t \gamma^t \nabla \log \pi_\theta(A_t | S_t) (G_t - b(S_t))
\]

Discounted REINFORCE

- Can be defined...
- but still requires an episodic setting for the discounted return \( G_t \) to be computed.
Discounted Measure?

\[ \hat{\nabla} J_{\mu_0}(\pi_\theta) = \sum_t \gamma^t \nabla \log \pi_\theta(A_t|S_t) (G_t - b(S_t)) \]

\[ \rightarrow \hat{\nabla} J_{\mu_{\pi_\theta}}(\pi_\theta) = \frac{1}{1 - \gamma} \nabla \log \pi_\theta(A_t|S_t) (G_t - b(S_t)) \]

**Discounted Measure?**

- Much less weights for later states!
- Probability independent of \(t\) if the initial distribution is the stationary distribution \(\mu_{\pi_\theta}\) corresponding to \(\pi_\theta\) (it it exists).
- Approximately true after a burning stage if we reach stationarity...
- Better handled by the average return!

- More on this later...
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
Actor/Critic

- Actor: Parametric policy $\pi_\theta$ used.
- Critic: $Q$-value function $Q_w(\cdot, \cdot)$ approximating $Q_{\pi_\theta}$.
- Critic follows the Actor, which is optimized using the Critic.

- In Value Approximation, the Actor follows the Critic (through the argmax operator).
- In on-line methods, the Actor is used to interact with the environment.
Actor/Critic

\[ J(\mu_0)(\pi_\theta) = \sum_s \mu_0(s) v_{\pi_\theta}(s) \]

\[ \nabla J_{\mu_0}(\pi_\theta) = \sum_s \left( \sum_t \gamma^t P_{\pi_\theta}(S_t = s) \right) \left( \sum_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s)(Q_{\pi_\theta}(s, a) - V_{\pi_\theta}(s, a)) \right) \]

\[ \hat{\nabla} J_{\mu_0}(\pi_\theta) = \sum_t \gamma^t \pi_\theta(A_t|S_t) \nabla \log \pi_\theta(A_t|S_t) \left( Q_{\pi_\theta}(S_t, A_t) - \sum_a \pi(a|S_t) Q_{\pi_\theta}(S_t, A_t) \right) \]

\[ \simeq \sum_t \gamma^t \pi_\theta(A_t|S_t) \nabla \log \pi_\theta(A_t|S_t) \left( Q_w(S_t, A_t) - \sum_a \pi(a|S_t) Q_w(S_t, A_t) \right) \]

**Actor/Critic**

- Critic update: Stochastic Policy Gradient with plugin.
- Actor update: any \( Q \)-value methods estimating \( Q_{\pi_\theta} \).
- Requires a two-scales algorithm so that \( Q_w \) is always a good estimate of \( Q_{\pi_\theta} \).

- Is this a real algorithm in a non episodic setting?
Actor/Critic

\[ J_{\mu_{\pi_{\theta}}} (\pi_{\theta}) = \sum_s \mu_{\pi_{\theta}} (s) v_{\pi_{\theta}} (s) \]

\[ \nabla J_{\mu_{\pi_{\theta}}} (\pi_{\theta}) = \sum_s \frac{1}{1 - \gamma} P_{\pi_{\theta}} (S_t = s) \left( \sum_a \pi_{\theta} (a|s) \nabla \log \pi_{\theta} (a|s) (Q_{\pi_{\theta}} (s, a) - V_{\pi_{\theta}} (s, a)) \right) \]

\[ \hat{\nabla} J_{\mu_{\pi_{\theta}}} (\pi_{\theta}) \approx \frac{1}{1 - \gamma} \pi_{\theta} (A_t|S_t) \nabla \log \pi_{\theta} (A_t|S_t) \left( Q_{w} (S_t, A_t) - \sum_a \pi (a|S_t) Q_{w} (S_t, A_t) \right) \]

**Actor/Critic**

- Critic update: Stochastic Policy Gradient with plugin.
- Actor update: any Q-value methods estimating \( Q_{\pi_{\theta}} \).
- Requires a two-scales algorithm so that \( Q_{w} \) is always a good estimate of \( Q_{\pi_{\theta}} \).

- Require the existence of a stationary measure... and that this stationary measure is reached *quickly*.
- Much harder to do off-policy algorithm as the stationary measure is not known!
Critic in Actor/Critic

\[ Q_w \approx Q_{\pi\theta} \]

- On-line TD learning with interaction following \( \pi_\theta \).
- Off-Policy TD learning is possible if the policy used for any action is stored.
- Approximate off-policy TD learning is possible using a replay buffer providing \( \pi_\theta \) is changing slowly.

- May lead to 3 scales algorithm (Actor/Critic Target/Critic)
- As mentioned in the previous slide, much harder to do off-line update for the actor.
Off-Line Actor

$$J'_\mu(\pi) = \sum_s \mu(s) v_\pi(s)$$

- Idea proposed in 2012.
- Key lemma in the paper
  $$\nabla J'_\mu(\pi_\theta) \approx \sum_s \mu(s) \sum_a \pi_\theta(a|s) \nabla \pi_\theta(a|s) Q_{\pi_\theta}(s, a)$$
  turns out to be wrong!
- Still used as a heuristic justification of many algorithms!
- Explicit formula for $\nabla J'_\mu(\pi_\theta)$ can be obtained but much harder to use...
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differences
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy Prediction and Control
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
PPO: Minorize-Majorization Algorithm

\[ J_{\mu_0}(\pi') \geq J_{\mu_0}(\pi) + \sum_t \gamma^t P_{\pi}(S_t = s) \left( \sum_a (\pi'(s|a) - \pi(s|a)) A_{\pi}(s, a) \right) \]

\[ - \sum_t \gamma^t 2t \max_s \|\pi'(\cdot|s) - \pi(\cdot|s)\|_2^2 \max_{s,a} |A_{\pi}(s, a)| \]

**Ideal Minorize-Majorization Algorithm**

- At step $k$, find $\theta_{k+1}$ maximizing

\[ J_{\mu_0}(\pi_{\theta} | \pi_{\theta_k}) = \sum_s \sum_t \gamma^t P_{\pi_{\theta_k}}(S_t = s) \left( \sum_a (\pi_{\theta}(s|a) - \pi_{\theta_k}(s|a)) A_{\pi_{\theta_k}}(s, a) \right) \]

\[ - \sum_t \gamma^t 2t \max_s \|\pi_{\theta}(\cdot|s) - \pi_{\theta_k}(\cdot|s)\|_2^2 \max_{s,a} |A_{\pi_{\theta_k}}(s, a)| \]

- By construction, $J_{\mu_0}(\pi_{\theta_{k+1}}) \geq J_{\mu_0}(\pi_{\theta_k})$

- Sample efficient algorithm as the same trajectory can be (re)used in the optimization.
PPO: Optimization

\[ J_{\mu_0}(\pi_\theta) \geq J_{\mu_0}(\pi_{\theta_k}) + \sum_s \sum_t \gamma^t \mathbb{P}_{\pi_{\theta_k}}(S_t = s) \left( \sum_a (\pi_\theta(s|a) - \pi_{\theta_k}(s|a)) A_{\pi_{\theta_k}}(s, a) \right) \]

\[ - \sum_t \gamma^t 2t \max_s ||\pi_\theta(\cdot|s) - \pi_{\theta_k}(\cdot|s)||_2^2 \max_{s,a} |A_{\pi_{\theta_k}}(s, a)| \]

**Optimization**

- Gradient descent is possible.
- Gradient of the first term is

\[ \sum_s \sum_t \gamma^t \mathbb{P}_\pi(S_t = s) \left( \sum_a \pi_\theta \nabla \pi_\theta(s|a) A_{\pi_{\theta_k}}(s, a) \right) \]

- Gradient of the second term more involved.

- Simpler (TRPO like) strategy: optimize

\[ \sum_s \sum_t \gamma^t \mathbb{P}_{\pi_{\theta_k}}(S_t = s) \left( \sum_a (\pi_\theta(s|a) - \pi_{\theta_k}(s|a)) A_{\pi_{\theta_k}}(s, a) \right) \]

under \( \max_s ||\pi_\theta(\cdot|s) - \pi_{\theta_k}(\cdot|s)||_2^2 \leq \epsilon \) and reduce \( \epsilon \) there is no gain.
PPO: KL Relaxation

\[ J_{\mu_0}(\pi_\theta) \geq J_{\mu_0}(\pi_{\theta_k}) + \sum_s \sum_t \gamma^t P_{\pi_{\theta_k}}(S_t = s) \left( \sum_a (\pi_\theta(s|a) - \pi_{\theta_k}(s|a)) A_{\pi_{\theta_k}}(s, a) \right) \]

\[ - \frac{2\gamma R_{\text{max}}}{(1 - \gamma)^3} \max_s \text{KL}(\pi_{\theta_k}(\cdot|s), \pi_\theta(\cdot|s)) \]

TRPO/PPO Optimization

- Replace the \( \ell_1 \) norm by a KL divergence.
- In practice, replace the max by an average and replace \( \frac{2\gamma R_{\text{max}}}{(1 - \gamma)^3} \) by parameter \( \beta \).
- PPO: Gradient descent of the relaxed goal.
- TRPO: Constrained optimization.

- Adaptive scheme to set \( \beta \).
- Can be used with continuous action.
Reinforcement Learning: Policy Approach

PPO: Clipped Objective

\[
\sum_{s} \sum_{t} \gamma^t P_{\pi_{\theta_k}} (S_t = s) \left( \sum_a \pi_{\theta_k}(s|a) \min \left( \frac{\pi_{\theta}(s|a)}{\pi_{\theta_k}(s,a)}, \text{clip}(1 - \epsilon, \frac{\pi_{\theta}(s|a)}{\pi_{\theta_k}(s,a)}), 1 + \epsilon \right) \right)
\]

Clipped Objective

- Insight by (re)substracting \( \sum_a \pi_{\theta_k}(s|a)A_{\theta_k}(s, a) = 0 \):
  \[
  \sum_a \min \left( (\pi_{\theta}(s|a) - \pi_{\theta_k}(s, a)) A_{\pi_{\theta_k}}(s, a), \text{clip}(-\epsilon, \pi_{\theta}(s|a) - \pi_{\theta_k}(s, a), \epsilon) A_{\pi_{\theta_k}}(s, a) \right)
  = \sum_a \text{clip}(-\epsilon \pi_{\theta_k}(s, a), \pi_{\theta}(s|a) - \pi_{\theta_k}(s, a), \epsilon \pi_{\theta_k}(s, a)) A_{\pi_{\theta_k}}(s, a)
  - \max \left( 0, - (\pi_{\theta}(s|a) - \pi_{\theta_k}(s, a)) A_{\pi_{\theta_k}}(s, a) - \epsilon \pi_{\theta_k}(s, a) | A_{\pi_{\theta_k}}(s, a)| \right)
  \]

- First term amount to replace \( \pi_{\theta} \) by a policy
  \[
  \tilde{\pi}_{\theta}(a|s) = \text{clip}(\pi_{\theta_k}(a|s)(1 - \epsilon), \pi_{\theta}(a|s), \pi_{\theta_k}(a|s)(1 + \epsilon)) + \eta_s \pi_{\theta_k}(a|s)
  \]
  where \( \eta \) is so that \( \tilde{\pi} \) is a probability for all \( s \) and \( \| \tilde{\pi}_{\theta}(\cdot, s) - \pi_{\theta_k}(\cdot, s) \|_1 \leq \epsilon \)

- Second term is a hinge loss type penalization of the policy \( \pi_{\theta} \) penalizing bad actions.

Very efficient for discrete actions.
PPO: Stationary Objective

\[
\sum_{s} \mathbb{P}_{\pi_{\theta_{k}}} (S_t = s) \left( \sum_{a} \left( \pi_{\theta}(s|a) - \pi_{\theta_{k}}(s|a) \right) A_{\pi_{\theta_{k}}}(s, a) \right) - \beta \max_{s} KL(\pi_{\theta_{k}}(\cdot|s), \pi_{\theta}(\cdot|s)) \\
= \sum_{s} \mathbb{P}_{\pi_{\theta_{k}}} (S_t = s) \left( \sum_{a} \pi_{\theta_{k}}(s|a) \min \left( \frac{\pi_{\theta}(s|a)}{\pi_{\theta_{k}}(s, a)}, \text{clip}(1 - \epsilon, \frac{\pi_{\theta}(s|a)}{\pi_{\theta_{k}}(s, a)}, 1 + \epsilon) A_{\pi_{\theta_{k}}}(s, a) \right) \right)
\]

Stationary Objective

- Amount to replace \( J_{\mu_0}(\pi) \) by \( J_{\mu_\pi}(\pi) \)
- Most common implementation of PPO...
- But no way to justify it mathematically!
- May explain the (possible) instabilities of PPO.
DPG: Deterministic Policy Gradient

\[ \pi_\theta(a|s) = 1_{a=h_\theta(s)} \quad (\text{deterministic policy}) \]

\[ J_{\mu_0}(\pi_\theta) = \sum_s \mu_0(s) v_{\pi_\theta}(s) \]

\[ \nabla J_{\mu_0}(\pi_\theta) = \sum_s \sum_t \gamma^t P_{\pi_\theta}(S_t = s) \nabla_a Q(S_t, h_\theta(S_t)) \]

**Deterministic Policy Gradient**

- Deterministic policy replaced by a randomized one centered on \( h_\theta(s) \) in the interactions!
- Critic trained with a TD variant of DQN.
- Gradient can be obtained by use a policy \( \pi_\theta = \mathcal{N}(h_\theta(s), \sigma^2 \text{Id}) \) and letting \( \sigma \) goes to 0.
- Off-Policy as claimed?
- Yes for the actor but no theoretical justification for the critic!
- In practice, the buffer contains only samples using a policy close to the current one...
SAC: A New Goal

\[ R_t \rightarrow R_t + \lambda H(\pi(S_t)) \]

A Modified Reward

- Modification of the reward to favor high entropy policy:
  \[ R_t \rightarrow R_t + \lambda H(\pi(S_t)) \]

- Goal:
  \[ J(\pi) = \sum_t (R_t + \lambda H(\pi(S_t))) \]

- Soft value function implicitly defined as the fixed point of
  \[ T^\pi Q_\pi(s, a) = r_\pi(s, a) + \sum_{s'} p(s'|s, a)V_\pi(s') \]

  \[ V_\pi(s, a) = \sum_a \pi(a|s)(Q_\pi(s, a) - \log \pi(a|s)) \]
SAC: Policy Improvement and Optimal Policy

\[ R_t \rightarrow R_t + \lambda \mathcal{H}(\pi(S_t)) \]

A Modified Policy Improvement Lemma

- Policy improvement rule:
  \[
  \pi^+ (\cdot | s) = \text{argmax}_{\pi (\cdot | s)} \sum_a \pi (a | s) (q(s, a) - \lambda \log(\pi(a | s)))
  \]
  \[
  \pi^+ (a | s) \propto \exp\left(-\frac{1}{\lambda} q(s, a)\right)
  \]
  implies \( G_{\pi^+}(s, a) \geq G_\pi(s, a) \).

- At convergence, \( J(\pi^*) \) is optimal!

- Convergence in the finite setting.
SAC: Parametrization

\[ \pi \sim \pi_\theta \quad \text{and} \quad Q(s, a) \sim Q_w \]

**SAC Choices**

- **Fitted TD learning for** \( Q \):
  \[
  w \simeq \arg\min \sum_{(S,A,R,S') \in B} \left( R + \mathbb{E}_{\pi_\theta} [\gamma Q_w(S', a) - \lambda \log \pi_\theta(a|S')] - Q_w(S, A) \right)^2
  \]
  where the trajectory pieces are samples from a replay buffer and \( \overline{w} \) is a slowdown version of \( w \) (two-scales algorithm).

- **Online version rather than batch...**

- **Fitted KL for** \( \pi \):
  \[
  \theta \simeq \arg\min \sum_{(S,A,R,S') \in B} \text{KL}(\pi_\theta(\cdot|S)| \exp -\lambda Q[\overline{w}](S, \cdot)/Z_{\overline{w}}(S))
  \]
  \[
  \simeq \sum_{(S,A,R,S') \in B} \mathbb{E}_{\pi_{\theta}} \left[ \frac{1}{\lambda} \log \pi_{\theta}(a|S) - Q_{\theta}(a|s) \right]
  \]
Continuing Tasks and Average Return

- Most natural performance measure:
  \[ J(\pi) = r(\pi) = \lim_{T \to \infty} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}_\pi[R_t|S_0] \]
  \[ = \sum_s \left( \lim_{T \to \infty} \sum_{t=1}^{T} \mathbb{P}_\pi(S_t = s|S_0) \right) \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)r \]

- \( \mu \) if it exists is such that
  \[ \sum_s \mu(s) \sum_a \pi(a|s)p(s'|s, a) = \mu(s') \]

- Gradient of \( J(\pi_\theta) \):
  \[ \nabla J(\pi_\theta) = \sum_s \mu(s) \sum_a \pi_\theta(a|s) \nabla \log \pi_\pi_\theta(a|s)q_{\pi_\theta}(s, a) \]
  Beware \( q_{\pi_\theta} \) are the relative Q-value functions and not the classical one.
Average Return and Relative Value Functions

\[ r(\pi) = \sum_{s} \mu(s) \sum_{a} \pi(a|s) \sum_{s',r} p(s', r|s, a)r \]

\[ G_t = \sum_{t' \geq t} R_t - r(\pi) \]

\[ V_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] \quad \text{and} \quad Q_\pi(s) = \mathbb{E}_\pi[G_t | S_t = s] \]

- Numerical algorithm to estimate those relative value functions.
- Leads to another family of Policy Gradient algorithm.
Outline

1. Sequential Decisions, MDP and Policies
   - Decision Process and Markov Decision Process
   - Returns and Value Functions
   - Prediction and Planning
   - Operations Research and Reinforcement Learning
   - Control
   - Survey

   - Prediction and Bellman Equation
   - Prediction by Dynamic Programming and Contraction
   - Planning, Optimal Policies and Bellman Equation
   - Linear Programming
   - Planning by Value Iteration
   - Planning by Policy Iteration

3. Optimization Interpretation
   - Approximation and Stability
   - Generalized Policy Iteration
   - Infinite, Episodic and Average setting
   - Reinforcement Learning: Prediction and Planning in the Tabular Setting
   - Prediction with Monte Carlo
   - Planning with Monte Carlo
   - Prediction with Temporal Differentials
   - Link with Stochastic Approximation
   - Planning with Value Iteration
   - Planning with Policy Improvement
   - Exploration vs Exploitation

4. Reinforcement Learning: Advanced Techniques in the Tabular Setting
   - n-step Algorithms
   - Eligibility Traces
   - Off-policy vs on-policy
   - Bandits
   - Model Based Approach

5. Reinforcement Learning: Approximation of the Value Functions
   - Approximation Target(s)
   - Gradient and Pseudo-Gradient
   - Linear Approximation and LSTD
   - On-Policy Prediction and Control
   - Off-Policy and Deadly Triad
   - Two-Scales Algorithms
   - Deep Q Learning
   - Continuous Actions

   - Policy Gradient Theorems
   - Monte Carlo Based Policy Gradient
   - Actor / Critic Principle
   - 3 SOTA Algorithms
   - Average Return

7. References
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.  
Wiley, 2005

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

W. Powell.  
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

Creative Commons Attribution-ShareAlike (CC BY-SA 4.0)

- You are free to:
  - **Share**: copy and redistribute the material in any medium or format
  - **Adapt**: remix, transform, and build upon the material for any purpose, even commercially.
- Under the following terms:
  - **Attribution**: You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - **ShareAlike**: If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - **No additional restrictions**: You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Contributors

- **Main contributor**: E. Le Pennec