ECEN 743: Reinforcement Learning

RL with Function Approximation

Dileep Kalathil
Assistant Professor
Department of Electrical and Computer Engineering
Texas A&M University

References

- [SB, Chapter 9-11]
- [BDP, Chapter 6]

Tabular Form to Function Approximation

- RL should be able to solve large problems

Tabular Form to Function Approximation

- RL should be able to solve large problems
  - Backgammon: 10^{20}

Tabular Form to Function Approximation

- RL should be able to solve large problems
  - Backgammon: 10^{20}
  - Go: 10^{170}

Tabular Form to Function Approximation

- RL should be able to solve large problems
  - Backgammon: 10^{20}
  - Go: 10^{170}
  - Robotics: continuous states

Tabular Form to Function Approximation

- RL should be able to solve large problems
  - Backgammon: 10^{20}
  - Go: 10^{170}
  - Robotics: continuous states
- Tabular representation of the value and policy is infeasible

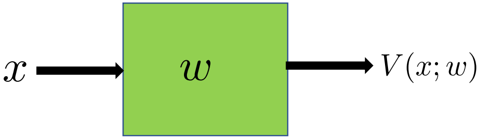
Tabular Form to Function Approximation

- RL should be able to solve large problems
  - Backgammon: 10^{20}
  - Go: 10^{170}
  - Robotics: continuous states
- Tabular representation of the value and policy is infeasible
- How do we learn to control large scale systems?

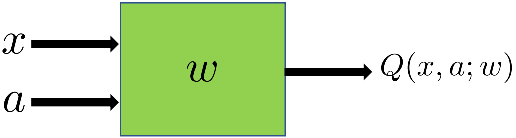
Tabular Form to Function Approximation

- RL should be able to solve large problems
  - Backgammon: 10^{20}
  - Go: 10^{170}
  - Robotics: continuous states
- Tabular representation of the value and policy is infeasible
- How do we learn to control large scale systems?
- Function Approximation

$V(x) \approx V(x; w)$



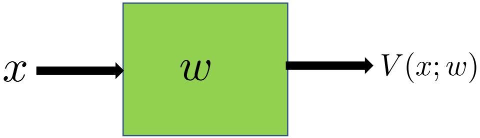
$Q(x, a) \approx Q(x, a; w)$



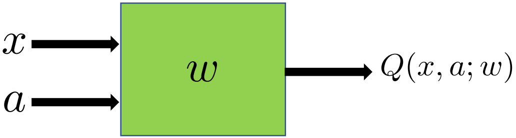
Tabular Form to Function Approximation

- RL should be able to solve large problems
  - Backgammon: 10^{20}
  - Go: 10^{170}
  - Robotics: continuous states
- Tabular representation of the value and policy is infeasible
- How do we learn to control large scale systems?
- Function Approximation

$V(x) \approx V(x; w)$



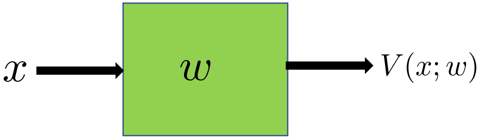
$Q(x, a) \approx Q(x, a; w)$



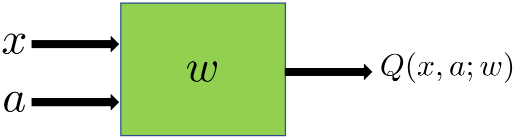
Tabular Form to Function Approximation

- RL should be able to solve large problems
  - Backgammon: 10^{20}
  - Go: 10^{170}
  - Robotics: continuous states
- Tabular representation of the value and policy is infeasible
- How do we learn to control large scale systems?
- Function Approximation

$V(x) \approx V(x;w)$



$Q(x,a) \approx Q(x,a;w)$



Function approximation will extrapolate/generalize the value/policy from seen states to unseen states

Linear Function Approximation

- Feature vector:

$$\text{For each } s \in \mathcal{S},\ \phi(s) \in \mathbb{R}^d,\ \phi(s) = (\phi_1(s), \phi_2(s), \ldots, \phi_d(s))^\top$$

Linear Function Approximation

- Feature vector:

$$\text{For each } s \in \mathcal{S},\ \phi(s) \in \mathbb{R}^d,\ \phi(s) = (\phi_1(s), \phi_2(s), \ldots, \phi_d(s))^\top$$

- Value function is approximated as, $V(s; w) = w^\top \phi(s)$, where $w \in \mathbb{R}^d$ is the weight vector.

Linear Function Approximation

- Feature vector:

$$\text{For each } s \in \mathcal{S},\ \phi(s) \in \mathbb{R}^d,\ \phi(s) = (\phi_1(s), \phi_2(s), \ldots, \phi_d(s))^\top$$

- Value function is approximated as, $V(s;w) = w^\top \phi(s)$, where $w \in \mathbb{R}^d$ is the weight vector.
- Typically $d \ll |\mathcal{S}|$

Linear Function Approximation

- Feature vector:

$$\text{For each } s \in \mathcal{S},\ \phi(s) \in \mathbb{R}^d,\ \phi(s) = (\phi_1(s), \phi_2(s), \ldots, \phi_d(s))^\top$$

- Value function is approximated as, $V(s; w) = w^\top \phi(s)$, where $w \in \mathbb{R}^d$ is the weight vector.
- Typically $d \ll |\mathcal{S}|$
- Value function approximation is linear in w

Linear Function Approximation

- Feature vector:

$$\text{For each } s \in \mathcal{S},\ \phi(s) \in \mathbb{R}^d,\ \phi(s) = (\phi_1(s), \phi_2(s), \ldots, \phi_d(s))^\top$$

- Value function is approximated as, $V(s; w) = w^\top \phi(s)$, where $w \in \mathbb{R}^d$ is the weight vector.
- Typically $d \ll |\mathcal{S}|$
- Value function approximation is linear in w
- Feature vector is typically assumed to be known

Linear Function Approximation

- Feature vector:

$$\text{For each } s \in \mathcal{S},\ \ \phi(s) \in \mathbb{R}^d,\ \ \phi(s) = (\phi_1(s), \phi_2(s), \ldots, \phi_d(s))^\top$$

- Value function is approximated as, $V(s; w) = w^\top \phi(s)$, where $w \in \mathbb{R}^d$ is the weight vector.
- Typically $d \ll |\mathcal{S}|$
- Value function approximation is linear in w
- Feature vector is typically assumed to be known
- Learning algorithm will find the w that gives the best approximation

Linear Function Approximation

- Feature vector:

$$\text{For each } s \in \mathcal{S},\ \phi(s) \in \mathbb{R}^d,\ \phi(s) = (\phi_1(s), \phi_2(s), \ldots, \phi_d(s))^\top$$

- Value function is approximated as, $V(s; w) = w^\top \phi(s)$, where $w \in \mathbb{R}^d$ is the weight vector.
- Typically $d \ll |\mathcal{S}|$
- Value function approximation is linear in w
- Feature vector is typically assumed to be known
- Learning algorithm will find the w that gives the best approximation
- $(\phi_i)_{i=1}^d$ are often assumed to be independent. So, they are also called basis functions

Feature Vector: Examples

- Polynomial basis functions

Feature Vector: Examples

- Polynomial basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^2$, so that we represent $s = (s_1, s_2)^\top$. We can consider feature vectors of many polynomial forms

$$\phi(s) = (s_1, s_2)^\top, \;\; \phi(s) = (1, s_1, s_2, s_1 s_2)^\top$$
$$\phi(s) = (1, s_1, s_2, s_1 s_2, s_1^2, s_2^2, s_1 s_2^2, s_1^2 s_2, s_1^2 s_2^2)^\top$$

Feature Vector: Examples

- Polynomial basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^2$, so that we represent $s = (s_1, s_2)^\top$. We can consider feature vectors of many polynomial forms

$$\phi(s) = (s_1, s_2)^\top, \; \phi(s) = (1, s_1, s_2, s_1 s_2)^\top$$
$$\phi(s) = (1, s_1, s_2, s_1 s_2, s_1^2, s_2^2, s_1 s_2^2, s_1^2 s_2, s_1^2 s_2^2)^\top$$

- Sinusoidal basis functions

Feature Vector: Examples

- Polynomial basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^2$, so that we represent $s = (s_1, s_2)^\top$. We can consider feature vectors of many polynomial forms

$$\phi(s) = (s_1, s_2)^\top, \ \phi(s) = (1, s_1, s_2, s_1 s_2)^\top$$
$$\phi(s) = (1, s_1, s_2, s_1 s_2, s_1^2, s_2^2, s_1 s_2^2, s_1^2 s_2, s_1^2 s_2^2)^\top$$

- Sinusoidal basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^n$, and $\phi(s) \in \mathbb{R}^d$. We can consider sinusoidal feature vectors as $\phi_i(s) = \cos(c_i^\top s)$, where $c_i \in \mathbb{R}^n$ is known (fixed a priori)

Feature Vector: Examples

- Polynomial basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^2$, so that we represent $s = (s_1, s_2)^\top$. We can consider feature vectors of many polynomial forms

$$\phi(s) = (s_1, s_2)^\top, \ \phi(s) = (1, s_1, s_2, s_1 s_2)^\top$$
$$\phi(s) = (1, s_1, s_2, s_1 s_2, s_1^2, s_2^2, s_1 s_2^2, s_1^2 s_2, s_1^2 s_2^2)^\top$$

- Sinusoidal basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^n$, and $\phi(s) \in \mathbb{R}^d$. We can consider sinusoidal feature vectors as $\phi_i(s) = \cos(c_i^\top s)$, where $c_i \in \mathbb{R}^n$ is known (fixed a priori)

- Radial basis functions

Feature Vector: Examples

- Polynomial basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^2$, so that we represent $s = (s_1, s_2)^\top$. We can consider feature vectors of many polynomial forms

$$\phi(s) = (s_1, s_2)^\top,\ \phi(s) = (1, s_1, s_2, s_1 s_2)^\top$$
$$\phi(s) = (1, s_1, s_2, s_1 s_2, s_1^2, s_2^2, s_1 s_2^2, s_1^2 s_2, s_1^2 s_2^2)^\top$$

- Sinusoidal basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^n$, and $\phi(s) \in \mathbb{R}^d$. We can consider sinusoidal feature vectors as $\phi_i(s) = \cos(c_i^\top s)$, where $c_i \in \mathbb{R}^n$ is known (fixed a priori)

- Radial basis functions
  - Assume that $\mathcal{S} \subset \mathbb{R}^n$, and $\phi(s) \in \mathbb{R}^d$. Radial basis functions are defined as

$$\phi_i(s) = \exp\left(-\frac{\|s - \mu_i\|^2}{2\sigma_i^2}\right),$$

where μ_i, σ_i are fixed a priori

Matrix Notation

- We can write $V(w)(s) = (\Phi w)(s)$, where

$$\Phi = \begin{bmatrix} \phi_1(s_1) & \phi_2(s_1) & \cdots & \phi_d(s_1) \\ \phi_1(s_2) & \phi_2(s_2) & \cdots & \phi_d(s_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(s_{|\mathcal{S}|}) & \phi_2(s_{|\mathcal{S}|}) & \cdots & \phi_d(s_{|\mathcal{S}|}) \end{bmatrix} = \begin{bmatrix} | & | & \cdots & | \\ \boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots & \boldsymbol{\phi}_d \\ | & | & \cdots & | \end{bmatrix}$$

Matrix Notation

- We can write $V(w)(s) = (\Phi w)(s)$, where

$$\Phi = \begin{bmatrix} \phi_1(s_1) & \phi_2(s_1) & \cdots & \phi_d(s_1) \\ \phi_1(s_2) & \phi_2(s_2) & \cdots & \phi_d(s_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(s_{|\mathcal{S}|}) & \phi_2(s_{|\mathcal{S}|}) & \cdots & \phi_d(s_{|\mathcal{S}|}) \end{bmatrix} = \begin{bmatrix} | & | & \cdots & | \\ \boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots & \boldsymbol{\phi}_d \\ | & | & \cdots & | \end{bmatrix}$$

- We assume that $(\boldsymbol{\phi}_i)_{i=1}^d$ are independent

Matrix Notation

- We can write $V(w)(s) = (\Phi w)(s)$, where

$$\Phi = \begin{bmatrix} \phi_1(s_1) & \phi_2(s_1) & \cdots & \phi_d(s_1) \\ \phi_1(s_2) & \phi_2(s_2) & \cdots & \phi_d(s_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(s_{|\mathcal{S}|}) & \phi_2(s_{|\mathcal{S}|}) & \cdots & \phi_d(s_{|\mathcal{S}|}) \end{bmatrix} = \begin{bmatrix} | & | & \cdots & | \\ \boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots & \boldsymbol{\phi}_d \\ | & | & \cdots & | \end{bmatrix}$$

- We assume that $(\boldsymbol{\phi}_i)_{i=1}^d$ are independent
- So, this linear function approximation consider the value functions in a d-dimensional subspace

$$\mathcal{V}_\phi = \{\Phi w, w \in \mathbb{R}^d\}$$

Matrix Notation

- We can write $V(w)(s) = (\Phi w)(s)$, where

$$\Phi = \begin{bmatrix} \phi_1(s_1) & \phi_2(s_1) & \cdots & \phi_d(s_1) \\ \phi_1(s_2) & \phi_2(s_2) & \cdots & \phi_d(s_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(s_{|\mathcal{S}|}) & \phi_2(s_{|\mathcal{S}|}) & \cdots & \phi_d(s_{|\mathcal{S}|}) \end{bmatrix} = \begin{bmatrix} | & | & \cdots & | \\ \boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots & \boldsymbol{\phi}_d \\ | & | & \cdots & | \end{bmatrix}$$

- We assume that $(\boldsymbol{\phi}_i)_{i=1}^d$ are independent
- So, this linear function approximation consider the value functions in a d-dimensional subspace

$$\mathcal{V}_\phi = \{\Phi w, w \in \mathbb{R}^d\}$$

- The actual value functions may be in an $|\mathcal{S}|$-dimensional space $\mathcal{V} \subset \mathbb{R}^{|\mathcal{S}|}$

Approximation as Projection

- How do we find the *best* approximation for $V \in \mathcal{V}$ in the set $\mathcal{V}_\phi = \{\Phi w, w \in \mathbb{R}^d\}$?

$$V_{\text{approx}} = \min_{\tilde{V} \in \mathcal{V}_\phi} \left\| V - \tilde{V} \right\|$$

Approximation as Projection

- How do we find the *best* approximation for $V \in \mathcal{V}$ in the set $\mathcal{V}_\phi = \{\Phi w, w \in \mathbb{R}^d\}$?

$$V_{\text{approx}} = \min_{\tilde{V} \in \mathcal{V}_\phi} \left\| V - \tilde{V} \right\|$$

  - We will consider a Hilbert space where the norm is defined using an inner product

Approximation as Projection

- How do we find the *best* approximation for $V \in \mathcal{V}$ in the set $\mathcal{V}_\phi = \{\Phi w, w \in \mathbb{R}^d\}$?

$$V_{\text{approx}} = \min_{\tilde{V} \in \mathcal{V}_\phi} \left\| V - \tilde{V} \right\|$$

  - We will consider a Hilbert space where the norm is defined using an inner product
- Best approximation can be thought as a projection of V on to the subspace $\mathcal{V}_\phi$. We denote it as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$

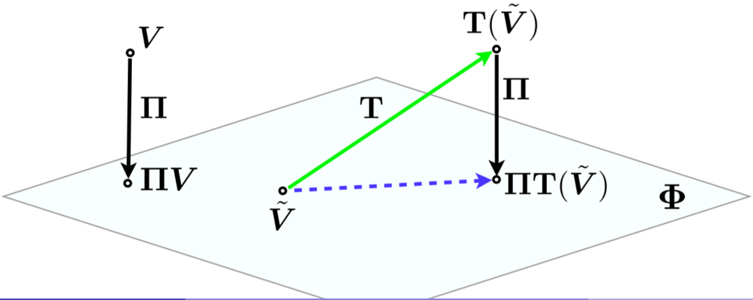
Approximation as Projection

- How do we find the *best* approximation for $V \in \mathcal{V}$ in the set $\mathcal{V}_\phi = \{\Phi w, w \in \mathbb{R}^d\}$?

$$V_{\text{approx}} = \min_{\tilde{V} \in \mathcal{V}_\phi} \left\| V - \tilde{V} \right\|$$

  - We will consider a Hilbert space where the norm is defined using an inner product
- Best approximation can be thought as a projection of V on to the subspace $\mathcal{V}_\phi$. We denote it as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$



Some Facts about Projection

- Best approximation can be thought as a projection of V on to the subspace $\mathcal{V}_\phi$. We denote it as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$

Some Facts about Projection

- Best approximation can be thought as a projection of V on to the subspace $\mathcal{V}_\phi$. We denote it as
$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$
- Projection error vector, $(V - \Pi V)$, is orthogonal to *all* vectors in the subspace $\mathcal{V}_\phi$
$$\langle (V - \Pi V), \tilde{V} \rangle = 0, \ \forall \tilde{V} \in \mathcal{V}_\phi$$

Some Facts about Projection

- Best approximation can be thought as a projection of V on to the subspace $\mathcal{V}_\phi$. We denote it as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$

- Projection error vector, $(V - \Pi V)$, is orthogonal to *all* vectors in the subspace $\mathcal{V}_\phi$

$$\langle (V - \Pi V), \tilde{V} \rangle = 0, \ \forall \tilde{V} \in \mathcal{V}_\phi$$

- $\Pi(\Pi V) = \Pi V$

Some Facts about Projection

- Best approximation can be thought as a projection of V on to the subspace $\mathcal{V}_\phi$. We denote it as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$

- Projection error vector, $(V - \Pi V)$, is orthogonal to *all* vectors in the subspace $\mathcal{V}_\phi$

$$\langle (V - \Pi V), \tilde{V} \rangle = 0, \ \forall \tilde{V} \in \mathcal{V}_\phi$$

- $\Pi(\Pi V) = \Pi V$
- Projection mapping is linear: $\Pi(V_1 + V_2) = \Pi V_1 + \Pi V_2$

Some Facts about Projection

- Best approximation can be thought as a projection of V on to the subspace $\mathcal{V}_\phi$. We denote it as
$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$
- Projection error vector, $(V - \Pi V)$, is orthogonal to *all* vectors in the subspace $\mathcal{V}_\phi$
$$\langle (V - \Pi V), \tilde{V} \rangle = 0, \ \forall \tilde{V} \in \mathcal{V}_\phi$$
- $\Pi(\Pi V) = \Pi V$
- Projection mapping is linear: $\Pi(V_1 + V_2) = \Pi V_1 + \Pi V_2$
- Pythagorean theorem: If V_1 and V_2 are orthogonal, $\|V_1 + V_2\|^2 = \|V_1\|^2 + \|V_2\|^2$

Approximate Dynamic Programming

Policy Evaluation with Linear Function Approximation

- How do we find the value of a policy, assuming that the model is known?

Policy Evaluation with Linear Function Approximation

- How do we find the value of a policy, assuming that the model is known?
  - Policy evaluation iteration: $V_{k+1} = T_\pi V_k$. Then, $V_k \to V_\pi$

Policy Evaluation with Linear Function Approximation

- How do we find the value of a policy, assuming that the model is known?
  - Policy evaluation iteration: $V_{k+1} = T_\pi V_k$. Then, $V_k \to V_\pi$
- We want to represent value functions as $V \approx V(w)$

Policy Evaluation with Linear Function Approximation

- How do we find the value of a policy, assuming that the model is known?
  - Policy evaluation iteration: $V_{k+1} = T_\pi V_k$. Then, $V_k \to V_\pi$
- We want to represent value functions as $V \approx V(w)$
- We will focus on linear function approximation, $V(w) = \Phi w$

Policy Evaluation with Linear Function Approximation

- How do we find the value of a policy, assuming that the model is known?
  - Policy evaluation iteration: $V_{k+1} = T_\pi V_k$. Then, $V_k \to V_\pi$
- We want to represent value functions as $V \approx V(w)$
- We will focus on linear function approximation, $V(w) = \Phi w$
- Define the projection mapping Π as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$

Policy Evaluation with Linear Function Approximation

- How do we find the value of a policy, assuming that the model is known?
  - Policy evaluation iteration: $V_{k+1} = T_\pi V_k$. Then, $V_k \to V_\pi$
- We want to represent value functions as $V \approx V(w)$
- We will focus on linear function approximation, $V(w) = \Phi w$
- Define the projection mapping Π as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$

- Define the projected policy evaluation iteration with function approximation as

$$V_{k+1} = \Pi T_\pi V_k$$

Policy Evaluation with Linear Function Approximation

- How do we find the value of a policy, assuming that the model is known?
  - Policy evaluation iteration: $V_{k+1} = T_\pi V_k$. Then, $V_k \to V_\pi$
- We want to represent value functions as $V \approx V(w)$
- We will focus on linear function approximation, $V(w) = \Phi w$
- Define the projection mapping Π as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$

- Define the projected policy evaluation iteration with function approximation as

$$V_{k+1} = \Pi T_\pi V_k$$

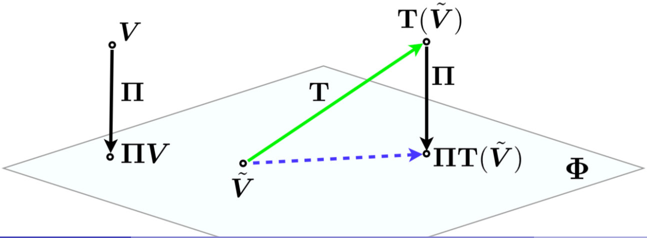
Policy Evaluation with Linear Function Approximation

- How do we find the value of a policy, assuming that the model is known?
  - Policy evaluation iteration: $V_{k+1} = T_\pi V_k$. Then, $V_k \to V_\pi$
- We want to represent value functions as $V \approx V(w)$
- We will focus on linear function approximation, $V(w) = \Phi w$
- Define the projection mapping Π as

$$\Pi V = \Phi w_V, \text{ where } w_V = \arg\min_{w \in \mathbb{R}^d} \|V - \Phi w\|$$

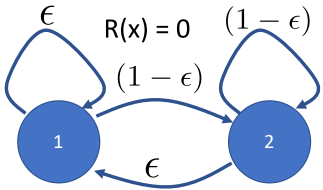
- Define the projected policy evaluation iteration with function approximation as

$$V_{k+1} = \Pi T_\pi V_k$$



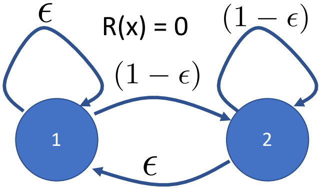
Policy Evaluation Iteration with Linear Function Approximation

- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(s) = 0$
- Use the linear approximation, $V(w)(s) = ws$



Policy Evaluation Iteration with Linear Function Approximation

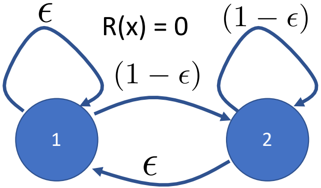
- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(s) = 0$
- Use the linear approximation, $V(w)(s) = ws$



- Optimal value of w is 0

Policy Evaluation Iteration with Linear Function Approximation

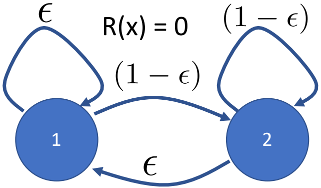
- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(s) = 0$
- Use the linear approximation, $V(w)(s) = ws$



- Optimal value of w is 0
- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$

Policy Evaluation Iteration with Linear Function Approximation

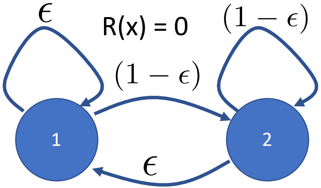
- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(s) = 0$
- Use the linear approximation, $V(w)(s) = ws$



- Optimal value of w is 0
- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$
- $V_k(1) = w_k, V_k(2) = 2w_k$

$$T_\pi V_k(1) =$$
$$T_\pi V_k(2) =$$

Policy Evaluation Iteration with Linear Function Approximation



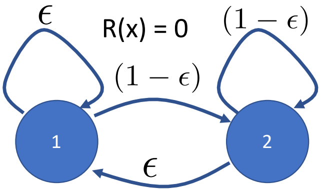
- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(s) = 0$
- Use the linear approximation, $V(w)(s) = ws$

- Optimal value of w is 0
- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$
- $V_k(1) = w_k, V_k(2) = 2w_k$

$$T_\pi V_k(1) = \gamma\epsilon V_k(1) + \gamma(1-\epsilon)V_k(2) = \gamma\epsilon w_k + \gamma(1-\epsilon)2w_k = \gamma(2-\epsilon)w_k$$
$$T_\pi V_k(2) = \gamma\epsilon V_k(1) + \gamma(1-\epsilon)V_k(2) = \gamma\epsilon w_k + \gamma(1-\epsilon)2w_k = \gamma(2-\epsilon)w_k$$

Policy Evaluation Iteration with Linear Function Approximation

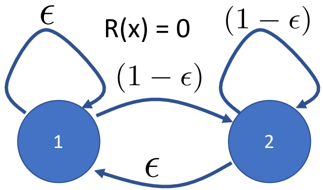
- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(x) = 0$
- Use the linear approximation, $V(w)(s) = ws$
- Optimal value of w is 0



- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$
- $V_k(1) = w_k, V_k(2) = 2w_k$
- $T_\pi V_k(1) = T_\pi V_k(2) = \gamma(2-\epsilon)w_k$

Policy Evaluation Iteration with Linear Function Approximation

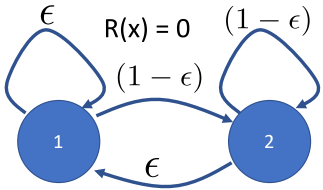
- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(x) = 0$
- Use the linear approximation, $V(w)(s) = ws$
- Optimal value of w is 0



- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$
- $V_k(1) = w_k, V_k(2) = 2w_k$
- $T_\pi V_k(1) = T_\pi V_k(2) = \gamma(2 - \epsilon)w_k$
- To find w_{k+1}, we need to evaluate the projection Π

Policy Evaluation Iteration with Linear Function Approximation

- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(x) = 0$
- Use the linear approximation, $V(w)(s) = ws$
- Optimal value of w is 0

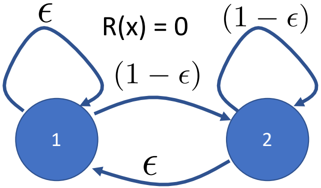


- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$
- $V_k(1) = w_k, V_k(2) = 2w_k$
- $T_\pi V_k(1) = T_\pi V_k(2) = \gamma(2-\epsilon)w_k$
- To find w_{k+1}, we need to evaluate the projection Π

$$\begin{aligned} w_{k+1} &= \arg\min_w \quad ||V - V(w)||_2 \\ &= \arg\min_w \left((V(1)-w)^2 + (V(2)-2w)^2\right) = (V(1)+2V(2))/5 \end{aligned}$$

Policy Evaluation Iteration with Linear Function Approximation

- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(x) = 0$
- Use the linear approximation, $V(w)(s) = ws$
- Optimal value of w is 0

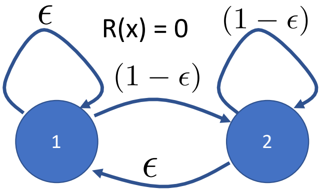


- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$
- $V_k(1) = w_k, V_k(2) = 2w_k$
- $T_\pi V_k(1) = T_\pi V_k(2) = \gamma(2-\epsilon)w_k$
- To find w_{k+1}, we need to evaluate the projection Π

$$
\begin{aligned}
w_{k+1} &= \arg\min_w \; ||V - V(w)||_2 \\
&= \arg\min_w \left((V(1) - w)^2 + (V(2) - 2w)^2\right) = (V(1) + 2V(2))/5
\end{aligned}
$$

- Using $V = T_\pi V_k$, we get $w_{k+1} = \frac{3}{5}\gamma(2-\epsilon)w_k$

Policy Evaluation Iteration with Linear Function Approximation



- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(x) = 0$
- Use the linear approximation, $V(w)(s) = ws$
- Optimal value of w is 0

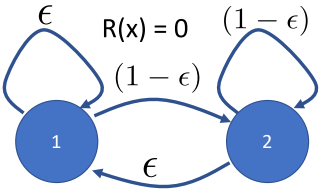
- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$
- $V_k(1) = w_k, V_k(2) = 2w_k$
- $T_\pi V_k(1) = T_\pi V_k(2) = \gamma(2-\epsilon)w_k$
- To find w_{k+1}, we need to evaluate the projection Π

$$\begin{aligned} w_{k+1} &= \arg\min_w \quad ||V - V(w)||_2 \\ &= \arg\min_w \left((V(1) - w)^2 + (V(2) - 2w)^2\right) = (V(1) + 2V(2))/5 \end{aligned}$$

- Using $V = T_\pi V_k$, we get $w_{k+1} = \frac{3}{5}\gamma(2-\epsilon)w_k$
- The above iteration may not converge for all values of γ and ϵ

Policy Evaluation Iteration with Linear Function Approximation

- Consider the Markov chain induced by a policy on an MDP. Clearly, $V_\pi(x) = 0$
- Use the linear approximation, $V(w)(s) = ws$
- Optimal value of w is 0



- Consider the iteration $V_{k+1} = \Pi T_\pi V_k$
- $V_k(1) = w_k, V_k(2) = 2w_k$
- $T_\pi V_k(1) = T_\pi V_k(2) = \gamma(2-\epsilon)w_k$
- To find w_{k+1}, we need to evaluate the projection Π

$$w_{k+1} = \arg\min_w \ ||V - V(w)||_2$$
$$= \arg\min_w \left((V(1) - w)^2 + (V(2) - 2w)^2\right) = (V(1) + 2V(2))/5$$

- Using $V = T_\pi V_k$, we get $w_{k+1} = \frac{3}{5}\gamma(2-\epsilon)w_k$
- The above iteration may not converge for all values of γ and ϵ
- The mapping ΠT_π is not a contraction w.r.t. $\|\cdot\|_2$

Weighted Norm and Contraction

- T_π is a contraction w.r.t. $\|\cdot\|_\infty$, but not w.r.t. $\|\cdot\|_2$

Weighted Norm and Contraction

- T_π is a contraction w.r.t. $\|\cdot\|_\infty$, but not w.r.t. $\|\cdot\|_2$
- Π is non-expansive w.r.t. $\|\cdot\|_2$, but not w.r.t. $\|\cdot\|_\infty$

Weighted Norm and Contraction

- T_π is a contraction w.r.t. $\|\cdot\|_\infty$, but not w.r.t. $\|\cdot\|_2$
- Π is non-expansive w.r.t. $\|\cdot\|_2$, but not w.r.t. $\|\cdot\|_\infty$
- How do we ensure that the iteration $V_{k+1} = \Pi T_\pi V_k$ will converge?

Weighted Norm and Contraction

- T_π is a contraction w.r.t. $\|\cdot\|_\infty$, but not w.r.t. $\|\cdot\|_2$
- Π is non-expansive w.r.t. $\|\cdot\|_2$, but not w.r.t. $\|\cdot\|_\infty$
- How do we ensure that the iteration $V_{k+1} = \Pi T_\pi V_k$ will converge?
- Define the weighted L^2 norm, $||V||_{2,\mu} = \mathbb{E}_{s\sim\mu}[V^2(s)]$ where μ is a probability distribution over $\mathcal{S}$.

Weighted Norm and Contraction

- T_π is a contraction w.r.t. $\|\cdot\|_\infty$, but not w.r.t. $\|\cdot\|_2$
- Π is non-expansive w.r.t. $\|\cdot\|_2$, but not w.r.t. $\|\cdot\|_\infty$
- How do we ensure that the iteration $V_{k+1} = \Pi T_\pi V_k$ will converge?
- Define the weighted L^2 norm, $||V||_{2,\mu} = \mathbb{E}_{s\sim\mu}[V^2(s)]$ where μ is a probability distribution over $\mathcal{S}$.
- $||\cdot||_{2,\mu}$ can be expressed as a norm from an inner product norm. Define

$$D = \text{diag}(\mu(s_1), \ldots, \mu(s_{|\mathcal{S}|})), \text{ and, } \langle V_1, V_2 \rangle = v_1^\top D V_2. \text{ Then,}$$
$$||V||_{2,\mu} = \langle V, V \rangle$$

Weighted Norm and Contraction

- T_π is a contraction w.r.t. $\|\cdot\|_\infty$, but not w.r.t. $\|\cdot\|_2$
- Π is non-expansive w.r.t. $\|\cdot\|_2$, but not w.r.t. $\|\cdot\|_\infty$
- How do we ensure that the iteration $V_{k+1} = \Pi T_\pi V_k$ will converge?
- Define the weighted L^2 norm, $||V||_{2,\mu} = \mathbb{E}_{s\sim\mu}[V^2(s)]$ where μ is a probability distribution over $\mathcal{S}$.
- $||\cdot||_{2,\mu}$ can be expressed as a norm from an inner product norm. Define

$$D = \text{diag}(\mu(s_1), \ldots, \mu(s_{|\mathcal{S}|})), \text{ and, } \langle V_1, V_2\rangle = v_1^\top D V_2. \text{ Then,}$$
$$||V||_{2,\mu} = \langle V, V\rangle$$

Weighted Norm and Contraction

- T_π is a contraction w.r.t. $\|\cdot\|_\infty$, but not w.r.t. $\|\cdot\|_2$
- Π is non-expansive w.r.t. $\|\cdot\|_2$, but not w.r.t. $\|\cdot\|_\infty$
- How do we ensure that the iteration $V_{k+1} = \Pi T_\pi V_k$ will converge?
- Define the weighted L^2 norm, $||V||_{2,\mu} = \mathbb{E}_{s\sim\mu}[V^2(s)]$ where μ is a probability distribution over $\mathcal{S}$.
- $||\cdot||_{2,\mu}$ can be expressed as a norm from an inner product norm. Define

$$D = \text{diag}(\mu(s_1), \ldots, \mu(s_{|\mathcal{S}|})), \text{ and, } \langle V_1, V_2\rangle = v_1^\top D V_2. \text{ Then,}$$

$$||V||_{2,\mu} = \langle V, V\rangle$$

Proposition (Contraction property of ΠT_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,

$$||\Pi T_\pi V_1 - \Pi T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$$

Jensen's Inequality

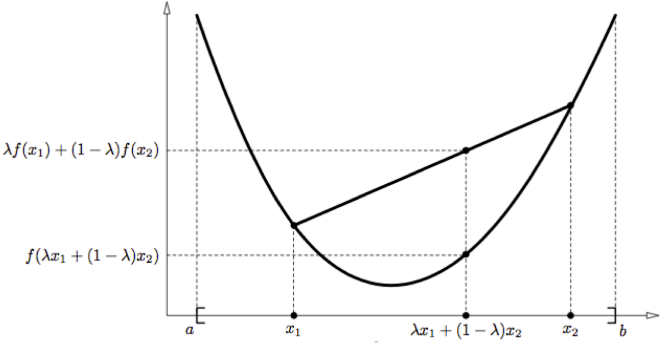
Proposition

Let $f : \mathbb{R}^n \to \mathbb{R}$ *be a convex function and let* X *be a random vector. Then,* $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$.

Jensen's Inequality

Proposition

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function and let X be a random vector. Then, $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$.



Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Proof:

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Proof: We will first show the following: Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then, $||P_\pi V||_{2,\mu} \leq ||V||_{2,\mu}$.

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Proof: We will first show the following: Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then, $||P_\pi V||_{2,\mu} \leq ||V||_{2,\mu}$.

$$||P_\pi V||^2_{2,\mu} = \mathbb{E}_{s \sim \mu}[((P_\pi V)(s))^2] = \sum_{s \in \mathcal{S}} \mu(s) \, (\sum_{s' \in \mathcal{S}} P_\pi(s, s') V(s'))^2$$

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Proof: We will first show the following: Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then, $||P_\pi V||_{2,\mu} \leq ||V||_{2,\mu}$.

$$
\begin{aligned}
||P_\pi V||_{2,\mu}^2 = \mathbb{E}_{s\sim\mu}[((P_\pi V)(s))^2] &= \sum_{s\in\mathcal{S}} \mu(s) \, (\sum_{s'\in\mathcal{S}} P_\pi(s,s')V(s'))^2 \\
&\leq \sum_{s\in\mathcal{S}} \mu(s) \sum_{s'\in\mathcal{S}} P_\pi(s,s')V^2(s')
\end{aligned}
$$

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Proof: We will first show the following: Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then, $||P_\pi V||_{2,\mu} \leq ||V||_{2,\mu}$.

$$
\begin{aligned}
||P_\pi V||_{2,\mu}^2 = \mathbb{E}_{s\sim\mu}[((P_\pi V)(s))^2] &= \sum_{s\in\mathcal{S}} \mu(s) \, (\sum_{s'\in\mathcal{S}} P_\pi(s,s')V(s'))^2 \\
&\leq \sum_{s\in\mathcal{S}} \mu(s) \sum_{s'\in\mathcal{S}} P_\pi(s,s')V^2(s') \\
&= \sum_{s'\in\mathcal{S}} \sum_{s\in\mathcal{S}} \mu(s) P_\pi(s,s') V^2(s')
\end{aligned}
$$

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Proof: We will first show the following: Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then, $||P_\pi V||_{2,\mu} \leq ||V||_{2,\mu}$.

$$
\begin{aligned}
||P_\pi V||^2_{2,\mu} = \mathbb{E}_{s\sim\mu}[((P_\pi V)(s))^2] &= \sum_{s\in\mathcal{S}} \mu(s) \,(\sum_{s'\in\mathcal{S}} P_\pi(s,s')V(s'))^2 \\
&\leq \sum_{s\in\mathcal{S}} \mu(s) \sum_{s'\in\mathcal{S}} P_\pi(s,s')V^2(s') \\
&= \sum_{s'\in\mathcal{S}} \sum_{s\in\mathcal{S}} \mu(s) P_\pi(s,s') V^2(s') \\
&= \sum_{s'\in\mathcal{S}} \mu(s') V^2(s') = ||V||^2_{2,\mu}
\end{aligned}
$$

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then,
$||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Proof: We will first show the following: Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then, $||P_\pi V||_{2,\mu} \leq ||V||_{2,\mu}$.

Weighted Norm and Contraction

Lemma (Contraction property of T_π w.r.t. $\|\cdot\|_{2,\mu}$)

Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then, $||T_\pi V_1 - T_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu}$

Proof: We will first show the following: Let μ be the stationary distribution corresponding to P_π, i.e., $\mu P_\pi = \mu$. Then, $||P_\pi V||_{2,\mu} \leq ||V||_{2,\mu}$. Then,

$$\begin{aligned} ||T_\pi V_1 - T_\pi V_2||_{2,\mu} &= ||(r_\pi + \gamma P_\pi V_1) - (r_\pi + \gamma P_\pi V_2)||_{2,\mu} \\ &= \gamma ||P_\pi V_1 - P_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu} \end{aligned}$$

$\square$

Weighted Norm and Contraction

Lemma (Non-expansive property of $\|\cdot\|_{2,\mu}$ projection)

$$||\Pi V||_{2,\mu} \leq ||V||_{2,\mu}$$

Proof: Proof is left as an exercise! □

Weighted Norm and Contraction

Lemma (Non-expansive property of $\|\cdot\|_{2,\mu}$ projection)

$$||\Pi V||_{2,\mu} \leq ||V||_{2,\mu}$$

Proof: Proof is left as an exercise! □

Proof: (Contraction property of ΠT_π w.r.t. $\|\cdot\|_{2,\mu}$)

$$\begin{aligned} ||\Pi T_\pi V_1 - \Pi T_\pi V_2||_{2,\mu} &\leq ||T_\pi V_1 - T_\pi V_2||_{2,\mu} \\ &= ||(r_\pi + \gamma P_\pi V_1) - (r_\pi + \gamma P_\pi V_2)||_{2,\mu} \\ &= \gamma ||P_\pi V_1 - P_\pi V_2||_{2,\mu} \leq \gamma ||V_1 - V_2||_{2,\mu} \end{aligned}$$

□

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$
- What is the approximation error $\|V_\pi - \bar{V}_\pi\|_{2,\mu}$?

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$
- What is the approximation error $\|V_\pi - \bar{V}_\pi\|_{2,\mu}$?

Proposition (Approximation Error)

$$\|\Pi V_\pi - V_\pi\|_{2,\mu} \leq \|\bar{V}_\pi - V_\pi\|_{2,\mu} \leq \frac{1}{\sqrt{1-\gamma^2}} \|\Pi V_\pi - V_\pi\|_{2,\mu}$$

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$
- What is the approximation error $\|V_\pi - \bar{V}_\pi\|_{2,\mu}$?

Proposition (Approximation Error)

$$\|\Pi V_\pi - V_\pi\|_{2,\mu} \leq \|\bar{V}_\pi - V_\pi\|_{2,\mu} \leq \frac{1}{\sqrt{1-\gamma^2}} \|\Pi V_\pi - V_\pi\|_{2,\mu}$$

Proof: Observe that $\bar{V}_\pi - \Pi V_\pi$ and $V_\pi - \Pi V_\pi$ are orthogonal vectors w.r.t. the inner product $\langle V_1, V_2 \rangle = \sum_s \mu(s) V_1(s) V_2(s)$. Now,

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$
- What is the approximation error $\|V_\pi - \bar{V}_\pi\|_{2,\mu}$?

Proposition (Approximation Error)

$$\|\Pi V_\pi - V_\pi\|_{2,\mu} \leq \|\bar{V}_\pi - V_\pi\|_{2,\mu} \leq \frac{1}{\sqrt{1-\gamma^2}}\|\Pi V_\pi - V_\pi\|_{2,\mu}$$

Proof: Observe that $\bar{V}_\pi - \Pi V_\pi$ and $V_\pi - \Pi V_\pi$ are orthogonal vectors w.r.t. the inner product $\langle V_1, V_2\rangle = \sum_s \mu(s)V_1(s)V_2(s)$. Now,

$$||\bar{V}_\pi - V_\pi||^2_{2,\mu} = ||\bar{V}_\pi - \Pi V_\pi + \Pi V_\pi - V_\pi||^2_{2,\mu}$$

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$
- What is the approximation error $\|V_\pi - \bar{V}_\pi\|_{2,\mu}$?

Proposition (Approximation Error)

$$\|\Pi V_\pi - V_\pi\|_{2,\mu} \leq \|\bar{V}_\pi - V_\pi\|_{2,\mu} \leq \frac{1}{\sqrt{1-\gamma^2}} \|\Pi V_\pi - V_\pi\|_{2,\mu}$$

Proof: Observe that $\bar{V}_\pi - \Pi V_\pi$ and $V_\pi - \Pi V_\pi$ are orthogonal vectors w.r.t. the inner product $\langle V_1, V_2 \rangle = \sum_s \mu(s) V_1(s) V_2(s)$. Now,

$$\begin{aligned} ||\bar{V}_\pi - V_\pi||^2_{2,\mu} &= ||\bar{V}_\pi - \Pi V_\pi + \Pi V_\pi - V_\pi||^2_{2,\mu} \\ &= ||\bar{V}_\pi - \Pi V_\pi||^2_{2,\mu} + 2 \left\langle \bar{V}_\pi - \Pi V_\pi, \Pi V_\pi - V_\pi \right\rangle + ||\Pi V_\pi - V_\pi||^2_{2,\mu} \end{aligned}$$

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$
- What is the approximation error $\|V_\pi - \bar{V}_\pi\|_{2,\mu}$?

Proposition (Approximation Error)

$$\|\Pi V_\pi - V_\pi\|_{2,\mu} \leq \|\bar{V}_\pi - V_\pi\|_{2,\mu} \leq \frac{1}{\sqrt{1-\gamma^2}} \|\Pi V_\pi - V_\pi\|_{2,\mu}$$

Proof: Observe that $\bar{V}_\pi - \Pi V_\pi$ and $V_\pi - \Pi V_\pi$ are orthogonal vectors w.r.t. the inner product $\langle V_1, V_2 \rangle = \sum_s \mu(s) V_1(s) V_2(s)$. Now,

$$\begin{aligned}
||\bar{V}_\pi - V_\pi||_{2,\mu}^2 &= ||\bar{V}_\pi - \Pi V_\pi + \Pi V_\pi - V_\pi||_{2,\mu}^2 \\
&= ||\bar{V}_\pi - \Pi V_\pi||_{2,\mu}^2 + 2 \left\langle \bar{V}_\pi - \Pi V_\pi, \Pi V_\pi - V_\pi \right\rangle + ||\Pi V_\pi - V_\pi||_{2,\mu}^2 \\
&= ||\Pi T_\pi \bar{V}_\pi - \Pi T_\pi V_\pi||_{2,\mu}^2 + ||\Pi V_\pi - V_\pi||_{2,\mu}^2
\end{aligned}$$

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$
- What is the approximation error $\|V_\pi - \bar{V}_\pi\|_{2,\mu}$?

Proposition (Approximation Error)

$$\|\Pi V_\pi - V_\pi\|_{2,\mu} \leq \|\bar{V}_\pi - V_\pi\|_{2,\mu} \leq \frac{1}{\sqrt{1-\gamma^2}} \|\Pi V_\pi - V_\pi\|_{2,\mu}$$

Proof: Observe that $\bar{V}_\pi - \Pi V_\pi$ and $V_\pi - \Pi V_\pi$ are orthogonal vectors w.r.t. the inner product $\langle V_1, V_2 \rangle = \sum_s \mu(s) V_1(s) V_2(s)$. Now,

$$\begin{aligned}
||\bar{V}_\pi - V_\pi||^2_{2,\mu} &= ||\bar{V}_\pi - \Pi V_\pi + \Pi V_\pi - V_\pi||^2_{2,\mu} \\
&= ||\bar{V}_\pi - \Pi V_\pi||^2_{2,\mu} + 2\left\langle \bar{V}_\pi - \Pi V_\pi, \Pi V_\pi - V_\pi \right\rangle + ||\Pi V_\pi - V_\pi||^2_{2,\mu} \\
&= ||\Pi T_\pi \bar{V}_\pi - \Pi T_\pi V_\pi||^2_{2,\mu} + ||\Pi V_\pi - V_\pi||^2_{2,\mu} \\
&\leq \gamma^2 ||\bar{V}_\pi - V_\pi||^2_{2,\mu} + ||\Pi V_\pi - V_\pi||^2_{2,\mu}
\end{aligned}$$

Approximation Error

- ΠT_π is a contraction mapping. So, it has a unique fixed point $\bar{V}_\pi$. The iteration $V_{k+1} = \Pi T_\pi V_k$ will converge to $\bar{V}_\pi$
- What is the approximation error $\|V_\pi - \bar{V}_\pi\|_{2,\mu}$?

Proposition (Approximation Error)

$$\|\Pi V_\pi - V_\pi\|_{2,\mu} \leq \|\bar{V}_\pi - V_\pi\|_{2,\mu} \leq \frac{1}{\sqrt{1-\gamma^2}} \|\Pi V_\pi - V_\pi\|_{2,\mu}$$

Proof: Observe that $\bar{V}_\pi - \Pi V_\pi$ and $V_\pi - \Pi V_\pi$ are orthogonal vectors w.r.t. the inner product $\langle V_1, V_2 \rangle = \sum_s \mu(s) V_1(s) V_2(s)$. Now,

$$\begin{aligned}
||\bar{V}_\pi - V_\pi||^2_{2,\mu} &= ||\bar{V}_\pi - \Pi V_\pi + \Pi V_\pi - V_\pi||^2_{2,\mu} \\
&= ||\bar{V}_\pi - \Pi V_\pi||^2_{2,\mu} + 2\left\langle \bar{V}_\pi - \Pi V_\pi, \Pi V_\pi - V_\pi \right\rangle + ||\Pi V_\pi - V_\pi||^2_{2,\mu} \\
&= ||\Pi T_\pi \bar{V}_\pi - \Pi T_\pi V_\pi||^2_{2,\mu} + ||\Pi V_\pi - V_\pi||^2_{2,\mu} \\
&\leq \gamma^2 ||\bar{V}_\pi - V_\pi||^2_{2,\mu} + ||\Pi V_\pi - V_\pi||^2_{2,\mu}
\end{aligned}$$

This implies, $\|\bar{V}_\pi - V_\pi\|_{2,\mu} \leq \frac{1}{\sqrt{1-\gamma^2}} \|\Pi V_\pi - V_\pi\|_{2,\mu}$.

TD Learning with Function Approximation

Tabular TD to Function Approximation

- Tabular TD learning

$$V_{t+1}(s_t) = V_t(s_t) + \alpha_t(r_t + \gamma V_t(s_{t+1}) - V_t(s_t)), \text{ and, } V_{t+1}(s) = V_t(s) \text{ for all } s \neq s_t$$

Tabular TD to Function Approximation

- Tabular TD learning

$$V_{t+1}(s_t) = V_t(s_t) + \alpha_t(r_t + \gamma V_t(s_{t+1}) - V_t(s_t)), \text{ and, } V_{t+1}(s) = V_t(s) \text{ for all } s \neq s_t$$

- TD Error: $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$

Tabular TD to Function Approximation

- Tabular TD learning

$$V_{t+1}(s_t) = V_t(s_t) + \alpha_t(r_t + \gamma V_t(s_{t+1}) - V_t(s_t)), \text{ and, } V_{t+1}(s) = V_t(s) \text{ for all } s \neq s_t$$

- TD Error: $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$
  - The goal is to minimize the expected TD error

Tabular TD to Function Approximation

- Tabular TD learning

$$V_{t+1}(s_t) = V_t(s_t) + \alpha_t(r_t + \gamma V_t(s_{t+1}) - V_t(s_t)), \text{ and, } V_{t+1}(s) = V_t(s) \text{ for all } s \neq s_t$$

- TD Error: $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$
  - The goal is to minimize the expected TD error
- TD learning with function approximation: Approximate the value function as $V(w_t)$, where w_t is the parameter at time t

Tabular TD to Function Approximation

- Tabular TD learning

$$V_{t+1}(s_t) = V_t(s_t) + \alpha_t(r_t + \gamma V_t(s_{t+1}) - V_t(s_t)), \text{ and, } V_{t+1}(s) = V_t(s) \text{ for all } s \neq s_t$$

- TD Error: $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$
  - The goal is to minimize the expected TD error
- TD learning with function approximation: Approximate the value function as $V(w_t)$, where w_t is the parameter at time t

Tabular TD to Function Approximation

- Tabular TD learning

$$V_{t+1}(s_t) = V_t(s_t) + \alpha_t(r_t + \gamma V_t(s_{t+1}) - V_t(s_t)), \text{ and, } V_{t+1}(s) = V_t(s) \text{ for all } s \neq s_t$$

- TD Error: $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$
  - The goal is to minimize the expected TD error
- TD learning with function approximation: Approximate the value function as $V(w_t)$, where w_t is the parameter at time t
- Define the TD error $\delta_t(w) = r_t + \gamma V(w_t)(s_{t+1}) - V(w)(s_t)$

Tabular TD to Function Approximation

- Tabular TD learning

$$V_{t+1}(s_t) = V_t(s_t) + \alpha_t(r_t + \gamma V_t(s_{t+1}) - V_t(s_t)), \text{ and, } V_{t+1}(s) = V_t(s) \text{ for all } s \neq s_t$$

- TD Error: $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$
  - The goal is to minimize the expected TD error
- TD learning with function approximation: Approximate the value function as $V(w_t)$, where w_t is the parameter at time t
- Define the TD error $\delta_t(w) = r_t + \gamma V(w_t)(s_{t+1}) - V(w)(s_t)$
- Update the parameter in order to minimize the squared TD error;

$$w_{t+1} = w_t - \alpha_t \nabla_w \delta_t^2(w)|_{w=w_t}, \text{ where,}$$

$$\nabla_w \delta_t^2(w)|_{w=w_t} = -\delta_t(w_t) \nabla_w V(w)(s_t)|_{w=w_t}$$

TD Learning with Linear Function Approximation

- TD learning with function approximation: Define the TD error at time step t as $\delta_t(w) = r_t + \gamma V(w_t)(s_{t+1}) - V(w)(s_t)$. Update the parameter as

$$w_{t+1} = w_t + \alpha_t \delta_t(w_t) \nabla_w V(w)(s_t)|_{w=w_t}$$

TD Learning with Linear Function Approximation

- TD learning with function approximation: Define the TD error at time step t as $\delta_t(w) = r_t + \gamma V(w_t)(s_{t+1}) - V(w)(s_t)$. Update the parameter as

$$w_{t+1} = w_t + \alpha_t \delta_t(w_t) \nabla_w V(w)(s_t)|_{w=w_t}$$

- TD learning with **linear** function approximation: Approximate the value function as $V(w)(s) = w^\top \phi(s)$, where $\phi(s) = (\boldsymbol{\phi}_1(s), \boldsymbol{\phi}_2(s), \ldots, \boldsymbol{\phi}_d(s))^\top$ is the feature vector. Then, update the parameter as

$$w_{t+1} = w_t + \alpha_t \delta_t(w_t) \phi(s_t)$$

TD Learning with Linear Function Approximation

- TD learning with function approximation: Define the TD error at time step t as $\delta_t(w) = r_t + \gamma V(w_t)(s_{t+1}) - V(w)(s_t)$. Update the parameter as

$$w_{t+1} = w_t + \alpha_t \delta_t(w_t) \nabla_w V(w)(s_t)|_{w=w_t}$$

- TD learning with **linear** function approximation: Approximate the value function as $V(w)(s) = w^\top \phi(s)$, where $\phi(s) = (\boldsymbol{\phi}_1(s), \boldsymbol{\phi}_2(s), \ldots, \boldsymbol{\phi}_d(s))^\top$ is the feature vector. Then, update the parameter as

$$w_{t+1} = w_t + \alpha_t \delta_t(w_t) \phi(s_t)$$

Theorem (Convergence of TD learning with linear function approximation)

Let w^* be such that $V(w^*) = \Pi T_\pi V(w^*)$. Then $w_t \to w^*$ almost surely, where w_t is given by the TD learning update equation above.

Matrix Notation

- We can write $V(w)(s) = (\Phi w)(s)$, where

$$\Phi = \begin{bmatrix} \phi_1(s_1) & \phi_2(s_1) & \cdots & \phi_d(s_1) \\ \phi_1(s_2) & \phi_2(s_2) & \cdots & \phi_d(s_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(s_{|\mathcal{S}|}) & \phi_2(s_{|\mathcal{S}|}) & \cdots & \phi_d(s_{|\mathcal{S}|}) \end{bmatrix} = \begin{bmatrix} | & | & \cdots & | \\ \boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots & \boldsymbol{\phi}_d \\ | & | & \cdots & | \end{bmatrix}$$

Matrix Notation

- We can write $V(w)(s) = (\Phi w)(s)$, where

$$\Phi = \begin{bmatrix} \phi_1(s_1) & \phi_2(s_1) & \cdots & \phi_d(s_1) \\ \phi_1(s_2) & \phi_2(s_2) & \cdots & \phi_d(s_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(s_{|\mathcal{S}|}) & \phi_2(s_{|\mathcal{S}|}) & \cdots & \phi_d(s_{|\mathcal{S}|}) \end{bmatrix} = \begin{bmatrix} | & | & \cdots & | \\ \boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots & \boldsymbol{\phi}_d \\ | & | & \cdots & | \end{bmatrix}$$

- TD learning with linear function approximation:

$$w_{t+1} = w_t + \alpha_t (r_t + \gamma w_t^\top \phi(s_{t+1}) - w_t^\top \phi(s_t))\ \phi(s_t)$$

Matrix Notation

- We can write $V(w)(s) = (\Phi w)(s)$, where

$$\Phi = \begin{bmatrix} \phi_1(s_1) & \phi_2(s_1) & \cdots & \phi_d(s_1) \\ \phi_1(s_2) & \phi_2(s_2) & \cdots & \phi_d(s_2) \\ \vdots & & \ddots & \vdots \\ \phi_1(s_{|\mathcal{S}|}) & \phi_2(s_{|\mathcal{S}|}) & \cdots & \phi_d(s_{|\mathcal{S}|}) \end{bmatrix} = \begin{bmatrix} | & | & \cdots & | \\ \boldsymbol{\phi}_1 & \boldsymbol{\phi}_2 & \cdots & \boldsymbol{\phi}_d \\ | & | & \cdots & | \end{bmatrix}$$

- TD learning with linear function approximation:

$$w_{t+1} = w_t + \alpha_t(r_t + \gamma w_t^\top \phi(s_{t+1}) - w_t^\top \phi(s_t))\ \phi(s_t)$$

$$w_{t+1} = w_t + \alpha_t\ (r_t + \gamma\ (\Phi w_t)(s_{t+1}) - (\Phi w_t)(s_t))\ \phi(s_t)$$

ODE Approximation

- TD learning update equation

$$w_{t+1} = w_t + \alpha_t \, (r_t + \gamma \, (\Phi w_t)(s_{t+1}) - (\Phi w_t)(s_t)) \, \phi(s_t)$$

ODE Approximation

- TD learning update equation

$$w_{t+1} = w_t + \alpha_t \left(r_t + \gamma \, (\Phi w_t)(s_{t+1}) - (\Phi w_t)(s_t)\right) \, \phi(s_t)$$

- ODE approximation

ODE Approximation

- TD learning update equation

$$w_{t+1} = w_t + \alpha_t \left(r_t + \gamma \, (\Phi w_t)(s_{t+1}) - (\Phi w_t)(s_t)\right) \, \phi(s_t)$$

- ODE approximation

$$\dot{w} = \mathbb{E}\left[\left(r_t + \gamma \, (\Phi w)(s_{t+1}) - (\Phi w)(s_t)\right) \, \phi(s_t)\right]$$

ODE Approximation

- TD learning update equation

$$w_{t+1} = w_t + \alpha_t \, (r_t + \gamma \, (\Phi w_t)(s_{t+1}) - (\Phi w_t)(s_t)) \, \phi(s_t)$$

- ODE approximation

$$\begin{aligned} \dot{w} &= \mathbb{E}\left[(r_t + \gamma \, (\Phi w)(s_{t+1}) - (\Phi w)(s_t)) \, \phi(s_t)\right] \\ &= \mathbb{E}\left[\mathbb{E}\left[(r_t + \gamma \, (\Phi w)(s_{t+1}) - (\Phi w)(s_t)) \, \phi(s_t) | s_t = s\right]\right] \end{aligned}$$

ODE Approximation

- TD learning update equation

$$w_{t+1} = w_t + \alpha_t \left(r_t + \gamma\ (\Phi w_t)(s_{t+1}) - (\Phi w_t)(s_t)\right)\ \phi(s_t)$$

- ODE approximation

$$\begin{aligned}
\dot{w} &= \mathbb{E}\left[(r_t + \gamma\ (\Phi w)(s_{t+1}) - (\Phi w)(s_t))\ \phi(s_t)\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[(r_t + \gamma\ (\Phi w)(s_{t+1}) - (\Phi w)(s_t))\ \phi(s_t) | s_t = s\right]\right] \\
&= \mathbb{E}\left[(r_\pi(s) + \gamma\ (P_\pi \Phi w)(s) - (\Phi w)(s))\ \phi(s)\right]
\end{aligned}$$

ODE Approximation

- TD learning update equation

$$w_{t+1} = w_t + \alpha_t \; (r_t + \gamma \; (\Phi w_t)(s_{t+1}) - (\Phi w_t)(s_t)) \; \phi(s_t)$$

- ODE approximation

$$\begin{aligned}
\dot{w} &= \mathbb{E}\left[(r_t + \gamma \; (\Phi w)(s_{t+1}) - (\Phi w)(s_t)) \; \phi(s_t)\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[(r_t + \gamma \; (\Phi w)(s_{t+1}) - (\Phi w)(s_t)) \; \phi(s_t) | s_t = s\right]\right] \\
&= \mathbb{E}\left[(r_\pi(s) + \gamma \; (P_\pi \Phi w)(s) - (\Phi w)(s)) \; \phi(s)\right] \\
&= \sum_{s \in \mathcal{S}} \; \mu_\pi(s) \; (r_\pi(s) + \gamma \; (P_\pi \Phi w)(s) - (\Phi w)(s)) \; \phi(s)
\end{aligned}$$

ODE Approximation

- TD learning update equation

$$w_{t+1} = w_t + \alpha_t \, (r_t + \gamma \, (\Phi w_t)(s_{t+1}) - (\Phi w_t)(s_t)) \, \phi(s_t)$$

- ODE approximation

$$\begin{aligned}
\dot{w} &= \mathbb{E}\left[(r_t + \gamma \, (\Phi w)(s_{t+1}) - (\Phi w)(s_t)) \, \phi(s_t)\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[(r_t + \gamma \, (\Phi w)(s_{t+1}) - (\Phi w)(s_t)) \, \phi(s_t) | s_t = s\right]\right] \\
&= \mathbb{E}\left[(r_\pi(s) + \gamma \, (P_\pi \Phi w)(s) - (\Phi w)(s)) \, \phi(s)\right] \\
&= \sum_{s \in \mathcal{S}} \mu_\pi(s) \, (r_\pi(s) + \gamma \, (P_\pi \Phi w)(s) - (\Phi w)(s)) \, \phi(s)
\end{aligned}$$

- This can be written as

$$\dot{w} = \Phi^\top D_\pi (r_\pi + \gamma \, (P_\pi \Phi w) - (\Phi w)),$$

where $D_\pi = \text{diag}(\mu(s_1), \ldots, \mu(s_{|\mathcal{S}|}))$

A Simple Result

- Define the inner product $\langle V_1, V_2 \rangle = V_1^\top D_\pi V_2$.

A Simple Result

- Define the inner product $\langle V_1, V_2 \rangle = V_1^\top D_\pi V_2$.

Lemma

Let Π *be an orthogonal projection. Then,* $\langle \Pi V_1, V_2 \rangle = \langle V_1, \Pi V_2 \rangle$

A Simple Result

- Define the inner product $\langle V_1, V_2 \rangle = V_1^\top D_\pi V_2$.

Lemma

Let Π be an orthogonal projection. Then, $\langle \Pi V_1, V_2 \rangle = \langle V_1, \Pi V_2 \rangle$

Proof: For any V_1, V_2, due to orthogonality, $\langle \Pi V_1, (V_2 - \Pi V_2) \rangle = \langle (V_1 - \Pi V_1), V_2 \rangle = 0$. We get the desired result from this. $\square$

Convergence of TD Learning with LFA

Proof:

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.
Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.
Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$\dot{U} = -2(w^* - w)^\top \dot{w}$$

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.
Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$\begin{aligned}\dot{U} &= -2(w^* - w)^\top \dot{w} \\ &= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma \, (P_\pi \Phi w) - (\Phi w))\end{aligned}$$

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.

Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$
\begin{aligned}
\dot{U} &= -2(w^* - w)^\top \dot{w} \\
&= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma\,(P_\pi \Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\; D_\pi\; (T_\pi(\Phi w) - (\Phi w))
\end{aligned}
$$

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.
Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$
\begin{aligned}
\dot{U} &= -2(w^* - w)^\top \dot{w} \\
&= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma\ (P_\pi \Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi(\Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi(\Phi w) - (\Phi w^*) + (\Phi w^*) - (\Phi w))
\end{aligned}
$$

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.

Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$\begin{aligned}
\dot{U} &= -2(w^* - w)^\top \dot{w} \\
&= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma\,(P_\pi \Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi(\Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi(\Phi w) - (\Phi w^*) + (\Phi w^*) - (\Phi w)) \\
&= -2\left[(\Phi w^* - \Phi w)^\top\ D_\pi\ (\Phi w^* - \Phi w) + (\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi \Phi w - \Phi w^*)\right]
\end{aligned}$$

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.

Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$\begin{aligned}
\dot{U} &= -2(w^* - w)^\top \dot{w} \\
&= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma \, (P_\pi \Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top \; D_\pi \; (T_\pi(\Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top \; D_\pi \; (T_\pi(\Phi w) - (\Phi w^*) + (\Phi w^*) - (\Phi w)) \\
&= -2 \left[(\Phi w^* - \Phi w)^\top \; D_\pi \; (\Phi w^* - \Phi w) + (\Phi w^* - \Phi w)^\top \; D_\pi \; (T_\pi \Phi w - \Phi w^*) \right] \\
&= -2 \left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top \; D_\pi \; (T_\pi \Phi w - \Phi w^*) \right]
\end{aligned}$$

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.

Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$\begin{aligned}
\dot{U} &= -2(w^* - w)^\top \dot{w} \\
&= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma\,(P_\pi \Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi(\Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi(\Phi w) - (\Phi w^*) + (\Phi w^*) - (\Phi w)) \\
&= -2\left[(\Phi w^* - \Phi w)^\top\ D_\pi\ (\Phi w^* - \Phi w) + (\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi \Phi w - \Phi w^*)\right] \\
&= -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi \Phi w - \Phi w^*)\right] \\
&\overset{(i)}{=} -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top\ D_\pi\ (\Pi T_\pi \Phi w - \Phi w^*)\right],
\end{aligned}$$

(we get (i) by the previous projection lemma)

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.

Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$\begin{aligned}
\dot{U} &= -2(w^* - w)^\top \dot{w} \\
&= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma\, (P_\pi \Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi(\Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi(\Phi w) - (\Phi w^*) + (\Phi w^*) - (\Phi w)) \\
&= -2\left[(\Phi w^* - \Phi w)^\top\ D_\pi\ (\Phi w^* - \Phi w) + (\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi \Phi w - \Phi w^*)\right] \\
&= -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top\ D_\pi\ (T_\pi \Phi w - \Phi w^*)\right] \\
&\overset{(i)}{=} -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top\ D_\pi\ (\Pi T_\pi \Phi w - \Phi w^*)\right],
\end{aligned}$$

(we get (i) by the previous projection lemma)

$$\overset{(ii)}{\leq} -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 - ||(\Phi w^* - \Phi w)||_{2,\mu}\ ||(\Pi T_\pi \Phi w - \Phi w^*)||_{2,\mu}\right]$$

(we get (ii) by Cauchy–Schwarz inequality)

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.

Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$
\begin{aligned}
\dot{U} &= -2(w^* - w)^\top \dot{w} \\
&= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma\,(P_\pi \Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\; D_\pi\; (T_\pi(\Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\; D_\pi\; (T_\pi(\Phi w) - (\Phi w^*) + (\Phi w^*) - (\Phi w)) \\
&= -2\left[(\Phi w^* - \Phi w)^\top\; D_\pi\; (\Phi w^* - \Phi w) + (\Phi w^* - \Phi w)^\top\; D_\pi\; (T_\pi \Phi w - \Phi w^*)\right] \\
&= -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top\; D_\pi\; (T_\pi \Phi w - \Phi w^*)\right] \\
&\overset{(i)}{=} -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top\; D_\pi\; (\Pi T_\pi \Phi w - \Phi w^*)\right],
\end{aligned}
$$

(we get (i) by the previous projection lemma)

$$
\overset{(ii)}{\leq} -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 - ||(\Phi w^* - \Phi w)||_{2,\mu}\; ||(\Pi T_\pi \Phi w - \Phi w^*)||_{2,\mu}\right]
$$

(we get (ii) by Cauchy–Schwarz inequality)

$$
\leq -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 - \gamma||(\Phi w^* - \Phi w)||_{2,\mu}^2\right]
$$

Convergence of TD Learning with LFA

Proof: Let w^* be such that $\Pi T_\pi V(w^*) = V(w^*)$.

Let $U(t) = ||w^* - w(t)||_2^2$. We will show that $\dot{U} < 0$ for $w(t) \neq w^*$.

$$\begin{aligned}
\dot{U} &= -2(w^* - w)^\top \dot{w} \\
&= -2(w^* - w)^\top \Phi^\top D_\pi (r_\pi + \gamma\, (P_\pi \Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\, D_\pi\, (T_\pi(\Phi w) - (\Phi w)) \\
&= -2(\Phi w^* - \Phi w)^\top\, D_\pi\, (T_\pi(\Phi w) - (\Phi w^*) + (\Phi w^*) - (\Phi w)) \\
&= -2\left[(\Phi w^* - \Phi w)^\top\, D_\pi\, (\Phi w^* - \Phi w) + (\Phi w^* - \Phi w)^\top\, D_\pi\, (T_\pi \Phi w - \Phi w^*)\right] \\
&= -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top\, D_\pi\, (T_\pi \Phi w - \Phi w^*)\right] \\
&\overset{(i)}{=} -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 + (\Phi w^* - \Phi w)^\top\, D_\pi\, (\Pi T_\pi \Phi w - \Phi w^*)\right],
\end{aligned}$$

(we get (i) by the previous projection lemma)

$$\overset{(ii)}{\leq} -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 - ||(\Phi w^* - \Phi w)||_{2,\mu}\, ||(\Pi T_\pi \Phi w - \Phi w^*)||_{2,\mu}\right]$$

(we get (ii) by Cauchy–Schwarz inequality)

$$\begin{aligned}
&\leq -2\left[||\Phi w^* - \Phi w||_{2,\mu}^2 - \gamma ||(\Phi w^* - \Phi w)||_{2,\mu}^2\right] \\
&= -2(1-\gamma)||\Phi w^* - \Phi w||_{2,\mu}^2
\end{aligned}$$

Convergence of TD Learning

Proof: So, we have

$$\dot{U} \leq -2(1-\gamma)||\Phi w^* - \Phi w||_{2,\mu}^2$$

Convergence of TD Learning

Proof: So, we have

$$\dot{U} \leq -2(1-\gamma)||\Phi w^* - \Phi w||^2_{2,\mu}$$

From this we we can see that the distance between w and w^* is non-increasing.

Convergence of TD Learning

Proof: So, we have

$$\dot{U} \leq -2(1-\gamma)||\Phi w^* - \Phi w||_{2,\mu}^2$$

From this we we can see that the distance between w and w^* is non-increasing. If we further assume that the columns of Φ are independent (which is a natural assumption since we can always eliminate redundant features), we can guarantee that $||w - w^*||_{2,\mu}^2$ strictly decreases until $w = w^*$.

Convergence of TD Learning

Proof: So, we have

$$\dot{U} \leq -2(1-\gamma)||\Phi w^* - \Phi w||^2_{2,\mu}$$

From this we we can see that the distance between w and w^* is non-increasing.
If we further assume that the columns of Φ are independent (which is a natural assumption since we can always eliminate redundant features), we can guarantee that $||w - w^*||^2_{2,\mu}$ strictly decreases until $w = w^*$.
Therefore, the ODE approximation of TD learning with linear function approximation will converge to the optimal parameter w^*.

Convergence of TD Learning

Proof: So, we have

$$\dot{U} \leq -2(1-\gamma)||\Phi w^* - \Phi w||^2_{2,\mu}$$

From this we we can see that the distance between w and w^* is non-increasing.
If we further assume that the columns of Φ are independent (which is a natural assumption since we can always eliminate redundant features), we can guarantee that $||w - w^*||^2_{2,\mu}$ strictly decreases until $w = w^*$.
Therefore, the ODE approximation of TD learning with linear function approximation will converge to the optimal parameter w^*.
Now, using the results from stochastic approximation theory, we can argue that TD learning with linear function approximation will converge to the optimal parameter w^*

$\square$

Convergence of RL Algorithms for Policy Evaluation

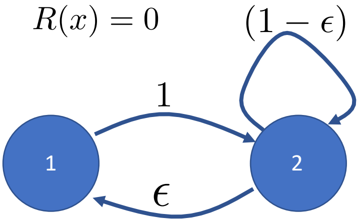
- on-policy policy evaluation
  - TD learning converges in the tabular setting
  - TD learning converges in the linear function approximation setting
  - Convergence is not guaranteed for nonlinear setting

Convergence of RL Algorithms for Policy Evaluation

- on-policy policy evaluation
  - TD learning converges in the tabular setting
  - TD learning converges in the linear function approximation setting
  - Convergence is not guaranteed for nonlinear setting
- off-policy policy evaluation algorithms are not guaranteed to converge

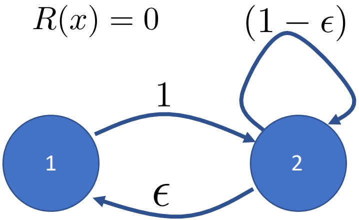
Example of Non-Convergence for Off-Policy Evaluation

- Consider the Markov chain induced by a policy π on an MDP. Clearly, $V_\pi(s) = 0$
- Use linear approximation with $\phi = \begin{bmatrix} 1 & 2 \end{bmatrix}^\top$
- We get $V(w)(s) = ws$. Optimal value of w is 0



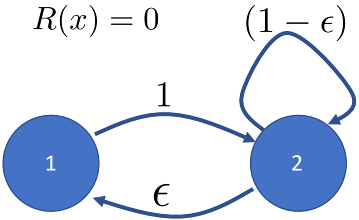
Example of Non-Convergence for Off-Policy Evaluation

- Consider the Markov chain induced by a policy π on an MDP. Clearly, $V_\pi(s) = 0$
- Use linear approximation with $\phi = \begin{bmatrix} 1 & 2 \end{bmatrix}^\top$
- We get $V(w)(s) = ws$. Optimal value of w is 0



- On-policy algorithm collects data, (s_i, a_i, s_{i+1}), according to policy π

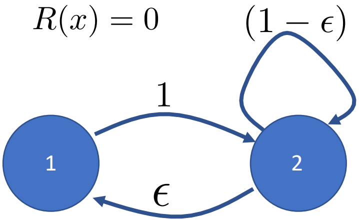
Example of Non-Convergence for Off-Policy Evaluation



- Consider the Markov chain induced by a policy π on an MDP. Clearly, $V_\pi(s) = 0$
- Use linear approximation with $\phi = \begin{bmatrix} 1 & 2 \end{bmatrix}^\top$
- We get $V(w)(s) = ws$. Optimal value of w is 0

- On-policy algorithm collects data, (s_i, a_i, s_{i+1}), according to policy π
- Consider another data collection procedure (off-policy data)

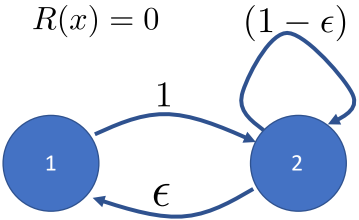
Example of Non-Convergence for Off-Policy Evaluation



- Consider the Markov chain induced by a policy π on an MDP. Clearly, $V_\pi(s) = 0$
- Use linear approximation with $\phi = \begin{bmatrix} 1 & 2 \end{bmatrix}^\top$
- We get $V(w)(s) = ws$. Optimal value of w is 0

- On-policy algorithm collects data, (s_i, a_i, s_{i+1}), according to policy π
- Consider another data collection procedure (off-policy data)
  - Sample state $s \in \{1, 2\}$ uniformly at random. Then get the next state according to the action taken by the policy π

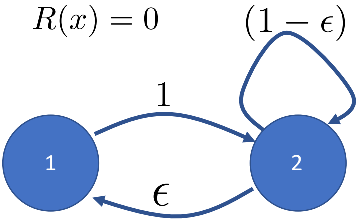
Example of Non-Convergence for Off-Policy Evaluation



- Consider the Markov chain induced by a policy π on an MDP. Clearly, $V_\pi(s) = 0$
- Use linear approximation with $\phi = \begin{bmatrix} 1 & 2 \end{bmatrix}^\top$
- We get $V(w)(s) = ws$. Optimal value of w is 0

- On-policy algorithm collects data, (s_i, a_i, s_{i+1}), according to policy π
- Consider another data collection procedure (off-policy data)
  - Sample state $s \in \{1, 2\}$ uniformly at random. Then get the next state according to the action taken by the policy π
- The distribution of the state-action pairs in the on-policy data and off-policy data will be different

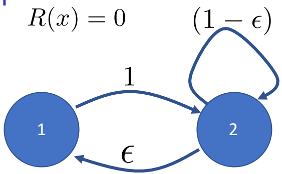
Example of Non-Convergence for Off-Policy Evaluation



- Consider the Markov chain induced by a policy π on an MDP. Clearly, $V_\pi(s) = 0$
- Use linear approximation with $\phi = \begin{bmatrix} 1 & 2 \end{bmatrix}^\top$
- We get $V(w)(s) = ws$. Optimal value of w is 0

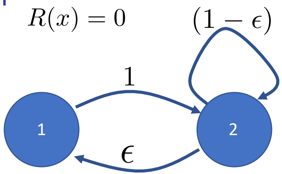
- On-policy algorithm collects data, (s_i, a_i, s_{i+1}), according to policy π
- Consider another data collection procedure (off-policy data)
  - Sample state $s \in \{1, 2\}$ uniformly at random. Then get the next state according to the action taken by the policy π
- The distribution of the state-action pairs in the on-policy data and off-policy data will be different
- How do we perform TD learning using off-policy data obtained as above?

Example of Non-Convergence for Off-Policy Evaluation



- Off-policy data: Sample state $s \in \{1, 2\}$ uniformly at random. Then get the next state according to the action taken by the policy π

Example of Non-Convergence for Off-Policy Evaluation

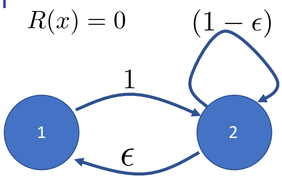


- Off-policy data: Sample state $s \in \{1, 2\}$ uniformly at random. Then get the next state according to the action taken by the policy π

- Recall TD learning with linear function approximation:

$$w_{t+1} = w_t + \alpha_t w_t(\gamma\phi(s_{t+1}) - \phi(s_t))\ \phi(s_t)$$

Example of Non-Convergence for Off-Policy Evaluation



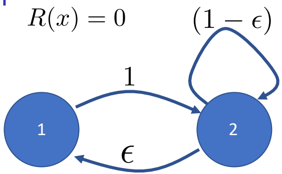
- Off-policy data: Sample state $s \in \{1, 2\}$ uniformly at random. Then get the next state according to the action taken by the policy π

- Recall TD learning with linear function approximation:

$$w_{t+1} = w_t + \alpha_t w_t(\gamma\phi(s_{t+1}) - \phi(s_t))\ \phi(s_t)$$

$$\begin{aligned}\mathbb{E}[w_{t+1}] &= \mathbb{E}[w_t] + \alpha_t \mathbb{E}[w_t]\left(\frac{1}{2}(\gamma\phi(2) - \phi(1))\phi(1)\right) \\ &\quad + \left(\frac{1}{2}(\gamma(\epsilon\phi(1) + (1-\epsilon)\phi(2)) - \phi(2))\phi(2)\right) \\ &= \mathbb{E}[w_t] + \alpha_t \mathbb{E}[w_t]\left(\frac{1}{2}(2\gamma - 1)\right) + \left(\frac{1}{2}(\gamma(4 - 2\epsilon) - 4)\right) \\ &= \mathbb{E}[w_t] + \alpha_t \mathbb{E}[w_t]\frac{1}{2}(6\gamma - 2\epsilon - 5)\end{aligned}$$

Example of Non-Convergence for Off-Policy Evaluation



- Off-policy data: Sample state $s \in \{1, 2\}$ uniformly at random. Then get the next state according to the action taken by the policy π

- Recall TD learning with linear function approximation:

$$w_{t+1} = w_t + \alpha_t w_t(\gamma\phi(s_{t+1}) - \phi(s_t))\ \phi(s_t)$$

$$\begin{aligned}\mathbb{E}[w_{t+1}] &= \mathbb{E}[w_t] + \alpha_t \mathbb{E}[w_t]\left(\frac{1}{2}(\gamma\phi(2) - \phi(1))\phi(1)\right) \\ &\quad + \left(\frac{1}{2}(\gamma(\epsilon\phi(1) + (1-\epsilon)\phi(2)) - \phi(2))\phi(2)\right) \\ &= \mathbb{E}[w_t] + \alpha_t \mathbb{E}[w_t]\left(\frac{1}{2}(2\gamma - 1)\right) + \left(\frac{1}{2}(\gamma(4 - 2\epsilon) - 4)\right) \\ &= \mathbb{E}[w_t] + \alpha_t \mathbb{E}[w_t]\frac{1}{2}(6\gamma - 2\epsilon - 5)\end{aligned}$$

- This will diverge if $\gamma > 5/(6 - \epsilon)$

Q-Learning with Function Approximation

Tabular Q-Learning to Function Approximation

- Tabular Q-learning

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \left(r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t)\right)$$

Tabular Q-Learning to Function Approximation

- Tabular Q-learning

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \ (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$$

- TD Error: $\delta_t = (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$

Tabular Q-Learning to Function Approximation

- Tabular Q-learning

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \; (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$$

- TD Error: $\delta_t = (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$
- Q-Learning learning with function approximation: Approximate the Q-value function as $Q_t(s, a) \approx Q(w_t)(s, a)$, where w_t is the parameter at time t

Tabular Q-Learning to Function Approximation

- Tabular Q-learning

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \ (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$$

- TD Error: $\delta_t = (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$
- Q-Learning learning with function approximation: Approximate the Q-value function as $Q_t(s, a) \approx Q(w_t)(s, a)$, where w_t is the parameter at time t

Tabular Q-Learning to Function Approximation

- Tabular Q-learning

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \ (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$$

- TD Error: $\delta_t = (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$
- Q-Learning learning with function approximation: Approximate the Q-value function as $Q_t(s, a) \approx Q(w_t)(s, a)$, where w_t is the parameter at time t
- Define the TD error $\delta_t(w) = r_t + \gamma \max_b Q(w_t)(s_{t+1}, b) - Q(w)(s_t, a_t)$

Tabular Q-Learning to Function Approximation

- Tabular Q-learning

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha_t \left(r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t)\right)$$

- TD Error: $\delta_t = (r_t + \gamma \max_b Q_t(s_{t+1}, b) - Q_t(s_t, a_t))$
- Q-Learning learning with function approximation: Approximate the Q-value function as $Q_t(s, a) \approx Q(w_t)(s, a)$, where w_t is the parameter at time t
- Define the TD error $\delta_t(w) = r_t + \gamma \max_b Q(w_t)(s_{t+1}, b) - Q(w)(s_t, a_t)$
- Update the parameter in order to minimize the squared TD error;

$$w_{t+1} = w_t - \alpha_t \nabla_w \delta_t^2(w)|_{w=w_t}, \text{ where,}$$

$$\nabla_w \delta_t^2(w)|_{w=w_t} = -\delta_t(w_t) \nabla_w Q(w)(s_t, a_t)|_{w=w_t}$$

Q-Learning with Function Approximation

- Q-learning with function approximation: Define $\delta_t(w) = r_t + \gamma \max_b Q(w_t)(s_{t+1}, b) - Q(w)(s_t, a_t)$. Update the parameter as

$$w_{t+1} = w_t + \alpha_t \delta_t(w_t) \nabla_w Q(w)(s_t, a_t)|_{w=w_t}$$

Q-Learning with Function Approximation

- Q-learning with function approximation: Define $\delta_t(w) = r_t + \gamma \max_b Q(w_t)(s_{t+1}, b) - Q(w)(s_t, a_t)$. Update the parameter as

$$w_{t+1} = w_t + \alpha_t \delta_t(w_t) \nabla_w Q(w)(s_t, a_t)|_{w=w_t}$$

$$w_{t+1} = w_t + \alpha_t \left(r_t + \gamma \max_b Q(w_t)(s_{t+1}, b) - Q(w_t)(s_t, a_t) \right) \nabla_w Q(w)(s_t, a_t)|_{w=w_t}$$

Q-Learning with Function Approximation

- Q-learning with function approximation: Define $\delta_t(w) = r_t + \gamma \max_b Q(w_t)(s_{t+1}, b) - Q(w)(s_t, a_t)$. Update the parameter as

$$w_{t+1} = w_t + \alpha_t \delta_t(w_t) \nabla_w Q(w)(s_t, a_t)|_{w=w_t}$$

$$w_{t+1} = w_t + \alpha_t \left(r_t + \gamma \max_b Q(w_t)(s_{t+1}, b) - Q(w_t)(s_t, a_t) \right) \nabla_w Q(w)(s_t, a_t)|_{w=w_t}$$

- Q-learning with **linear** function approximation: Approximate the value function as $Q(w)(s) = w^\top \phi(s, a)$, where $\phi(s) = (\boldsymbol{\phi}_1(s, a), \boldsymbol{\phi}_2(s, a), \ldots, \boldsymbol{\phi}_d(s, a))^\top$ is the feature vector. Then, update the parameter as

$$w_{t+1} = w_t + \alpha_t \left(r_t + \gamma \max_b Q(w_t)(s_{t+1}, b) - Q(w_t)(s_t, a_t) \right) \phi(s_t, a_t)$$

Convergence of Q-Learning Algorithm with Function Approximation

- on-policy policy evaluation
  - TD learning converges in the tabular setting
  - TD learning converges in the linear function approximation setting
  - Convergence is not guaranteed for nonlinear setting

Convergence of Q-Learning Algorithm with Function Approximation

- on-policy policy evaluation
  - TD learning converges in the tabular setting
  - TD learning converges in the linear function approximation setting
  - Convergence is not guaranteed for nonlinear setting
- off-policy policy evaluation algorithms, such as off-policy TD learning, are not guaranteed to converge

Convergence of Q-Learning Algorithm with Function Approximation

- on-policy policy evaluation
  - TD learning converges in the tabular setting
  - TD learning converges in the linear function approximation setting
  - Convergence is not guaranteed for nonlinear setting
- off-policy policy evaluation algorithms, such as off-policy TD learning, are not guaranteed to converge
- Q-learning
  - Q-learning converges in the tabular setting
  - Q-learning may not converge (even) in the linear function approximation setting
  - Convergence is not guaranteed for nonlinear setting

"Deadly Triad" [SB, Chapter 11]

- RL algorithms shows instability and divergence whenever we combine all of the following three elements:
- Function approximation: a scalable way of generalizing from a state space much larger than the memory and computational resources
- Bootstrapping: Update targets that include existing estimates rather than relying exclusively on actual rewards and complete returns
- Off-policy training: training on a distribution of transitions other than that produced by the target policy