Proximal Reinforcement Learning

Review of proximal methods, analysis of Gradient TD algorithms

Ahmed Touati

Reinforcement Learning class

Linear value approximation

- Value function: $V(s) = \mathbb{E}\{\sum_{t=0}^{\infty} \gamma^t r_{t+1} | s_0 = s\}$
- Linear approximation: $V_\theta(s) = \theta^T \phi_s$ where $\phi_s \in \mathbb{R}^n$ is a feature vector characterizing state s.
- Conventional linear TD algorithm:
  - We denote by (s_k, s'_k, r_k) the triples of state, next state, and reward with associated feature-vector random variables $\phi_k = \phi_{s_k}$ and $\phi'_k = \phi'_{s_k}$.
  - We define the temporal-difference error:

$$\delta_k = r_k + \gamma \theta^T \phi'_k - \theta^T \phi_k$$

  - parameters update:

$$\theta_{k+1} = \theta_k + \alpha_k \delta_k \phi_k$$

Review of Gradient temporal difference

- TD algorithm was motivated by a semi-gradient of a natural choice of objective function: closeness to the true value:

$$\mathrm{MSE}(\theta) = \sum_s d(s)(V_\theta(s) - V(s))^2 = ||V_\theta - V||_D^2$$

- TD algorithm converges to TD fixed point

$$0 = \mathbb{E}[\delta\phi] = -A\theta + b$$

where $A = \mathbb{E}[(\phi_k - \gamma\phi_k')\phi_k^T]$ and $b = \mathbb{E}[r_k\phi_k]$

- We could view the vector $\mathbb{E}[\delta\phi]$ as an error in the current solution θ. The vector should be zero, so its norm is a measure of how far we are away from the TD solution.

$$J(\theta) = \mathbb{E}[\delta\phi]^T\mathbb{E}[\delta\phi]$$

- this new objective function is called The norm of expected TD update (NEU).

Derivation of Gradient Temporal difference algorithm GTD(0)

- $-\frac{1}{2}\nabla NEU(\theta) = \mathbb{E}[(\phi_k - \gamma\phi'_k)\phi_k^T]\mathbb{E}[\delta\phi]$
- If gradient can be written as a single expectation, it is straightforward to use gradient stochastic gradient descent.
- However, we have a product of two expectations
- The sample product won't be an unbiased estimate of the gradient.
- A trick: introduce a second set of weights $w \in \mathbb{R}^n$ to perform a stochastic approximation of the quantity $\mathbb{E}[\delta\phi]$

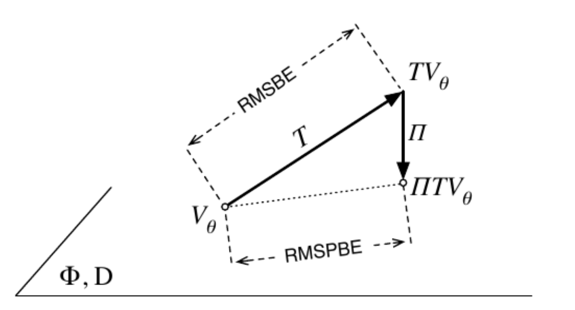
$$w_{k+1} = w_k + \beta_k(\delta_k\phi_k - w_k)$$

- Then, the update of θ would be :

$$\theta_{k+1} = \theta_k + \alpha_k(\phi_k - \gamma\phi'_k)(\phi^T w_k)$$

Projected Bellman error

- Another option: use an objective function representing how closely the approximate value function satisfies the Bellman equation:
$$\text{MSBE}(\theta) = ||V_\theta - TV_\theta||_D^2$$
where $TV = R + \gamma PV$ is the Bellman operator
- T takes you out the space. Π projects you back into
-
$$\text{MSPBE}(\theta) = ||V_\theta - \Pi TV_\theta||_D^2$$



Derivation of the Gradient Temporal difference algorithm GTD2

- $$\text{MSPBE}(\theta) = ||V_\theta - \Pi T V_\theta||_D^2 = \mathbb{E}[\delta\phi]\mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi]$$
- $$-\frac{1}{2}\nabla_\theta \text{MSPBE}(\theta) = \mathbb{E}[(\phi - \gamma\phi')\phi^T]\mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi]$$
- A trick: introduce a second set of weights $w \in \mathbb{R}^n$ to perform a stochastic approximation of the quantity $\mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi]$:

$$w = \mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi]$$

$$\mathbb{E}[\phi\phi^T]w = \mathbb{E}[\delta\phi]$$

$$w_{k+1} = w_k + \beta_k(\delta_k - \phi_k^T w_k)\phi_k$$

- Then:

$$\theta_{k+1} = \theta_k + \alpha_k(\phi_k - \gamma\phi_k')(\phi^T w_k)$$

Some notes about GTD

- All Gradient Temporal difference are asymptotically convergent to TD fixed point.
- The convergence proves use the stochastic approximation and Ordinary differential equation approach.
- Unfortunately, The GTD algorithms are not true stochastic gradient methods with respect to their original objective functions.
- The reason is biased sampling and ad-hoc splitting trick of terms.
- There is not finite-sample analysis (convergence rate) provided for those algorithms.

Proximal perspective of Temporal difference

- Proximal Reinforcement Learning is a new mathematical framework to tackle the difficulties of designing reliable and convergent reinforcement learning algorithms.
- It uses the proximal operator theory to make possible to design ”true” stochastic gradient methods for reinforcement learning in principled way.
- Then, convergence rate analysis could be provided.

Sub-differential review

- Sub-differential:

$$\partial g(x) = \{u \in \mathbb{X}, \forall z, g(x) \geq g(x) + < u, x - z >\}$$

- If g is smooth, then $\partial g(x) = \{\nabla g(x)\}$
- First order conditions:

$$x^* \in \text{argmin}_{x \in \mathbb{X}} g(x) \Leftarrow 0 \in \partial g(x^*)$$

Proximal Operator

- Proximal operator of g is:

$$\text{Prox}_{\gamma g}(x) = \text{argmin}_z\{\frac{1}{2}||z - x||^2 + \gamma g(z)\}$$

- $g(x) = ||x||_1$,

$$\text{Prox}_{\gamma g}(x)_i = max(0, 1 - \frac{\gamma}{|x_i|})x_i$$

- $g(x) = ||x||_0 = |\{i, x_i \neq 0\}|$,

$$\text{Prox}_{\gamma g}(x)_i = \begin{cases} x_i & \text{if } |x_i| \geq \sqrt{2\gamma} \\ 0 & \text{otherwise.} \end{cases}$$

Prox and sub-differential

- Resolvant of ∂g

$$
\begin{aligned}
z = \mathrm{Prox}_{\gamma g}(x) &\Leftrightarrow 0 \in z - x + \gamma \partial g(x) \\
\Leftrightarrow z \in (I + \gamma\partial g)(x) &\leftrightarrow x \in (I + \gamma \partial g)^{-1}(x)
\end{aligned}
$$

- Fixed point:

$$
\begin{aligned}
x^* \in \mathrm{argmin}_{x \in \mathbb{X}} g(x) &\Leftrightarrow 0 \in \partial g(x^*) \\
\Leftrightarrow x^* \in (I + \gamma\partial g)(x^*) &\Leftrightarrow x^* \in (I + \gamma\partial g)^{-1}(x^*) \\
\Leftrightarrow x^* = \mathrm{Prox}_{\gamma g}(x^*)
\end{aligned}
$$

Gradient and Proximal descent

- Gradient descent (smooth function):

$$x_{k+1} = x_k - \gamma_k \nabla g(x_k)$$

- Sub-gradient descent (slow convergence):

$$x_{k+1} = x_k - \gamma_k v_k, v_k \in \partial g(x_k)$$

- Proximal-point algorithm (hard to compute the proximal):

$$x_{k+1} = \mathrm{Prox}_{\gamma g}(x_k)$$

Proximal splitting Methods

- Problem: $min_x E(x)$
- $\mathrm{Prox}_{\gamma E}$ is not available
- Splitting: $E(x) = f(x) + \sum_i g_i(x)$ where f is smooth and g_i is simple.
- $\Rightarrow$ iterative algorithm using ∇f and $\mathrm{Prox}_{\gamma g}$
- Forward backward algorithm solves $f + g$
- Dual-primal algorithm solves $\sum_i g_i \circ A$

Forward backward

- Fixed point equation:

$$
\begin{aligned}
x^* \in \text{argmin}_{x\in\mathbb{X}} f(x) + g(x) &\Leftrightarrow 0 \in \nabla f(x^*) + \partial g(x^*) \\
&\Leftrightarrow x^* - \gamma \nabla f(x^*) \in (I + \gamma\partial g)(x^*) \\
&\Leftrightarrow x^* = \text{Prox}_{\gamma g}(x^* - \gamma \nabla f(x^*))
\end{aligned}
$$

- Forward backward update:

$$x_{k+1} = \text{Prox}_{\gamma g}(x_k - \gamma \nabla f(x_k))$$

Convex conjugate

$$f^*(y) = \sup_{x \in dom f} (< y, x > - f(x))$$

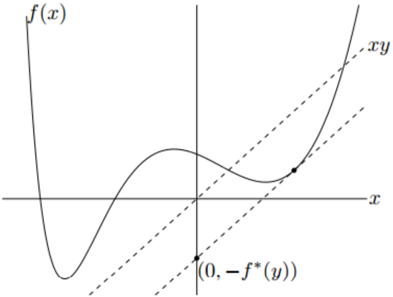


Figure 3.8 A function $f : \mathbf{R} \to \mathbf{R}$, and a value $y \in \mathbf{R}$. The conjugate function $f^*(y)$ is the maximum gap between the linear function yx and $f(x)$, as shown by the dashed line in the figure. If f is differentiable, this occurs at a point x where $f'(x) = y$.

Primal-dual formulation

- Problem $\min_x g(x) + f(Ax)$ with g and f convex and A is linear operator.
- $\min_x g(x) + f(Ax) = \min_x g(x) + \max_y(< Ax, y > -f^*(y))$
- Saddle point formulation:

$$\min_x \max_y \left(g(x) + < Ax, y > -f^*(y)\right)$$

- updates:

$$x_{k+1} = \mathrm{Prox}_{\gamma_k g}(x_k - \gamma_k A^T y_k)$$

$$y_{k+1} = y_k + \gamma_k(Ax_k - \nabla f^*(y_k))$$

Back to Temporal difference

- $$\mathrm{NEU}(\theta) = \mathbb{E}[\delta\phi]^T\mathbb{E}[\delta\phi] = ||\mathbb{E}[\delta\phi]|| = ||b - A\theta||$$
- $$\mathrm{MSPBE}(\theta) = \mathbb{E}[\delta\phi]^T\mathbb{E}[\phi\phi^T]^{-1}\mathbb{E}[\delta\phi] = ||\mathbb{E}[\delta\phi]||_{C^{-1}} = ||b - A\theta||_{C^{-1}}$$

  where $A = \mathbb{E}[(\phi_k - \gamma\phi'_k)\phi_k^T]$, $b = \mathbb{E}[r_k\phi_k]$ and $C = \mathbb{E}[\phi\phi^T]$
- $\Rightarrow$ NEU and MSBPE are square unweighted and weighted C^{-1} by l2 norm of $\mathbb{E}[\delta\phi]$.

- the problem now is: $\min_\theta(\frac{1}{2}||b - A\theta||_{M^{-1}} + g(\theta))$
- If $f(x) = \frac{1}{2}||x||_{M^{-1}}$, then $f^*(x) = \frac{1}{2}||x||_M$
- the saddle-point problem:

$$\min_\theta \max_w \Big(< b - A\theta, w > -\frac{1}{2}||w||_M + g(\theta)\Big)$$

- updates

$$\theta_{k+1} = \text{Prox}_{\gamma_k g}(\theta_k + \gamma_k A^T w_k)$$

$$w_{k+1} = w_k + (b - A\theta - M\theta_k)$$

- If we replace A, B and C by their unbiased estimates, we obtain the update rules of GTD (M=I) and GTD2 (M=C).
- Note that thanks to the dual formulation, we don't need to inverse the matrix C.

Questions?