Constrained Optimization in RL

Basic knowledge

Descent direction

1. Lemma 1: Given $x \in \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$, among all directions from x, the direction $d = -\nabla f(x)$ gives the maximum rate of decrease in terms of the value of f.
2. Lemma 2: Given $x \in \mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$, any direction $d \in \mathbb{R}^n$ satisfying $\langle d, \nabla f(x) \rangle < 0$ is a descent direction.
3. Lemma 3: Given $x \in \mathbb{R}^n$, $f : \mathbb{R}^n \to \mathbb{R}$, $S \in \mathbb{S}^n_{++}$, the direction $-S\nabla f(x) \in \mathbb{R}^n$ is a descent direction at x.

4. Different local approximation:
    1. $f(x) \approx f\left(x_k\right)+\nabla f\left(x_k\right)^{\top}\left(x-x_k\right)+\frac{1}{2 \alpha_k}\left(x-x_k\right)^{\top} S^{-1}\left(x-x_k\right)$
    2. $x_{k+1}=x_k-\alpha_k \cdot S_k \nabla f\left(x_k\right)$, Where $S_k \in \mathbb{S}_{++}^n$
        1. Gradient descent: $S_k=I_n$
        2. Newton direction: $S_k=\left[\nabla^2 f(x_k)\right]^{-1}$
        3. Scaled Newton direction: $S_k=\operatorname{diag}(\left[\nabla^2 f(x_k)\right]^{-1})$
        4. Quasi-Newton direction: approximation of $\left[\nabla^2 f(x_k)\right]^{-1}$
        5. Regularized Newton direction: $S_k=\left[\nabla^2 f\left(x_k\right)+\mu_k I\right]^{-1}$

Stepsize

1. Assumption 1: L-Lipschitz continuos gradient: $\|\nabla f(x)-\nabla f(y)\|_2 \leq L\|x-y\|_2$
2. Lemma 4: if f has L-Lipschitz continuos gradient and convex, then (s and n) $f(y) \leq f(x)+\nabla f(x)^{\top}(y-x)+\frac{L}{2}\|y-x\|_2^2$.
3. $\alpha=\frac{1}{L}$, then $f\left(x_k\right)-f^{\star} \leq \frac{\left\|x_0-x^{\star}\right\|_2^2}{2 \alpha k}$

Project Subgradient

1. First-order condition for convex function: $f(y) \geq f(x)+\nabla f(x)^{\top}(y-x)$
2. Subgradient: $f(x) \geq f\left(x_0\right)+y^{\top}\left(x-x_0\right)$
    1. $y \in \mathbb{R}^n$ is called a subgradient
    2. The set $\partial f\left(x_0\right)$ of all subgradients is the subdifferential of f at x_0
    3. Supporting hyperplane
        1. Definition 3 (Supporting hyperplane). Consider a nonempty set $C \subseteq \mathbb{R}^n$ and a boundary point $x_0 \in bd(C)$. If $a \neq 0$ in $\mathbb{R}^n$ satisfies $a^{\top} x \leq a^{\top} x_0, \forall x \in C$, then $\left\{x \mid a^{\top} x=a^{\top} x_0\right\}$ is called a supporting hyperplane to C at x_0.
        2. Theorem 1 (Weak Separating Hyperplane Theorem). Consider any convex set $C \subseteq \mathbb{R}^n$ and a point $x_0 \in \mathbb{R}^n / C$. Then, there exist $a \neq 0$ (in $\mathbb{R}^n$) and $b \in \mathbb{R}$ with $a^{\top} x \leq b \text { and } a^{\top} x_0 \geq b, \forall x \in C$
        3. For convex function, $\partial f\left(x_0\right) \neq \emptyset$
3. Orthogonal Projection
    1. $P_C(x) \equiv \underset{y \in C}{\arg \min }\|y-x\|$
    2. The projection $y^* \in C$ is unique
    3. non-expansive:
        1. $P_C(x)-P_C(y) \leq\|x-y\|$
4. Projection gradient
    1. $x_{k+1}=P_C(x_k-\alpha_k \nabla f(x_k))$
    2. equal to optimize the local approximation:
        1. $x_{k+1}=\arg \min_{x \in C}\{f(x_k)+\nabla_k f(x_k)^T(x-x_k)+\frac{L}{2}\|x-x_k\|^2\}$

Proximal Algorithms

1. Proximal operator: $\operatorname{prox}_f(x)=\arg \min_{u c \in \mathbb{R}^n}\left\{f(u))+\frac{1}{2}\|u-x\|^2\right\}$
    1. Consider optimizing the composite function $f(x)+g(x)$

1. Project gradient:

$$
\begin{aligned}
x_{k+1} &= \underset{x \in \mathbb{R}^n}{\arg\min} \left\{ f\left(x_k\right) + \nabla f\left(x_k\right)^{\top}\left(x - x_k\right) + g(x) + \frac{1}{2\alpha_k}\left\|x - x_k\right\|^2 \right\} \\
&= \underset{x \in \mathbb{R}^n}{\arg\min} \left\{ \alpha_k g(x) + \frac{1}{2}\left\|x - \left(x_k - \alpha_k \nabla f\left(x_k\right)\right)\right\|^2 \right\} \\
&= \operatorname{prox}_{\alpha_k g}\left(x_k - \alpha_k \nabla f\left(x_k\right)\right)
\end{aligned}
$$

2. gradient descent for $f(x_k)$, then project to $f(x_k) + g(x)$

2. Proximal gradient descent
   1. $T_L^{f,g}\{x) = \operatorname{prox}_{\frac{1}{L} g}\left(x - \frac{1}{L}\nabla f(x)\right)$
3. The Augment Lagrangian Methods
   1. $H^\star = \min\{H(x,z) \equiv h_1(x) + h_2(z) \mid Ax + Bz = c\}$
   2. Lagrangian function:

$$
\begin{aligned}
L(x,z,y) &= h_1(x) + h_2(z) + y^\top(Ax + Bz - c) \\
&= h_1(x) + y^\top Ax + h_2(z) + y^\top Bz - y^\top c
\end{aligned}
$$

   3. dual function: $g(y) = \min_{x,z}\ \{h_1(x) + y^\top Ax + h_2(z) + y^\top Bz - y^\top c\}$
      1. minimize the $-g(y)$ using proximal methods:
         1. $y_{k+1} = \arg\min_y\{-g(y) + \frac{1}{2\rho}\|y - y_k\|^2\}$
         2. stationary point: $y_{k+1} = y_k + \rho\nabla_y g(y_{k+1})$
            1. $y_{k+1} = y_k + \rho(A^T x_{k+1} + B^T z_{k+1} - c)$
            2. $x_{k+1} = \arg\min_x h_1(x) + y_{k+1}^T Ax$
               1. stationary point: $0 = A^T(y_k + \rho(A^T x_{k+1} + B^T z_{k+1} - c)) + \partial_x h_1(x_{k+1})$
            3. $z_{k+1} = \arg\min_z h_2(z) + y_{k+1}^T Bz$
               1. stationary point: $0 = B^T(y_k + \rho(A^T x_{k+1} + B^T z_{k+1} - c)) + \partial_z h_2(z_{k+1})$
         3. x_{k+1}, z_{k+1} equal to solve:
            1. $H(x,z) = h_1(x) + h_2(z) + \frac{\rho}{2}\|Ax + Bz - c + \frac{1}{\rho}y_k\|^2$
   4. algorithms:
      1. $x_{k+1}, z_{k+1} = \arg\min_{x,z} H(x,z)$
      2. $y_{k+1} = y_k + \rho(A^T x_{k+1} + B^T z_{k+1} - c)$
   5. ADMM
      1. $x_{k+1} = \arg\min_x H(x, z_k)$
      2. $z_{k+1} = \arg\min_z H(x_{k+1}, z)$
      3. $y_{k+1} = y_k + \rho(A^T x_{k+1} + B^T z_{k+1} - c)$

Extra-gradient

1. projection gradient descent:
   1. $x_{k+1} = P_C(x_k - \alpha_k \nabla_x f(x_k))$
   2. optimality condition: $x^* = P_C(x^* - \alpha_k \nabla_x f(x^*))$
2. Extrapolation
   1. $\bar{x}_k = x_k + \beta(x_k - x_{k-1})$

2. $x_{k+1} = x_k - \alpha \nabla_x f(\bar{x}_k)$
3. Project extra-gradient:
   1. $\bar{x}_{k+1} = P_C(x_k - \alpha_k \nabla_x f(x_k))$
   2. $x_{k+1} = P_C(x_k - \alpha_k \nabla_x f(\bar{x}_{k+1}))$

Natural Gradient

1. When optimizing distribution: $p(x|\theta)$
   1. distance between different distribution: $\mathrm{KL}[p|q] = \int p(x) \log \frac{p(x)}{q(x)}$
   2. local approximation of KL divergence:
      1. $KL[p(x|\theta)|p(x|\theta+\delta)] \approx \int p(x|\theta)[\log p(x|\theta) - (\log p(x|\theta) + \delta \nabla \log p(x|\theta) + \frac{1}{2}\delta^2 \nabla^2 \log p(x|\theta))]$
      2. $KL \approx \frac{1}{2}\delta E_p[\nabla^2 \log p(x|\theta)]\delta$
      3. $H = E_p[\nabla^2 \log p(x|\theta)] = E_p[\nabla \log p(x|\theta) \nabla \log p(x|\theta)^T]$
   3. Descent direction in the trust region:
      1. $\min_{D_{KL}[p_\theta \| \theta+\delta\theta] \leq \epsilon} L(\theta + \delta\theta)$
      2. Lagrangian: $\min L(\theta + \delta\theta) + \lambda \left(D_{KL}\left[p\left(\theta \| p_{\theta+\delta\theta}\right)\right] - \epsilon\right)$
      3. Approximation: $L(\theta) + \nabla L(\theta)^T \delta\theta + \lambda \left(\frac{1}{2}\delta\theta^T H \delta\theta - \epsilon\right)$
      4. Direction: $\delta\theta^* = -\frac{1}{\lambda} H^{-1} \nabla_\theta L(\theta)$

Conjugate gradient

1. Gradient descent
   1. exact step size search
      1. residue: $r_k = b - Qx_k$
      2. stepsize: $\alpha_k = \frac{r_k^T r_k}{r_k^T Q r_k}$
      3. update: $x_{k+1} = x_k + \alpha_k + r_k$
   2. When Q is ill-conditioned, converge slowly.
2. Conjugate
   1. $x_i Q x_j = 0$, then x_i, x_j are conjugate vectors of $Q \in S^+$
      1. $Q = I \rightarrow x_i x_j = 0$, orthogonal is a special case of conjugate
   2. only n independent conjugate vectors for Q
3. Gram-Schmidt Orthogonalization
   1. use n independent basis vector $\{a_i\}$ to construct orthogonal basis $\{q_i\}$
      1. $q_k = a_k + \sum_{j=1}^{k} b_{kj} q_j$,
      2. construct k using basis a_i with $i \leq k$
      3. create sequential dependence.
   2. weights $b_k = -\sum_{i=1}^{k-1} \frac{\langle a_k, q_i \rangle}{\langle q_i, q_i \rangle}$
      1. proof by inner product: $\langle q_k, q_i \rangle = \langle a_k, q_i \rangle + \sum_{j=1}^{k} b_{kj} \langle q_j, q_i \rangle$
      2. $\langle q_k, q_i \rangle = \langle a_k, q_j \rangle + b_{ki} \langle q_i, q_i \rangle$
4. Solving KKT condition: $\nabla f(x) = Qx - b = 0$
   1. find n conjugate vectors $p_i{}_{i=1}^m$ to combine $x^* = x_0 + \sum_{i=1}^m \alpha_i p_i$
   2. Given initial point x_0 and direction $p_k = \nabla f(x_0) = b - Qx_0$

3. line-search α_k. in linear case, $\alpha_k = \frac{r_k^T r_k}{r_k^T Q r_k}$
4. update descent direction through Gram-Schmidt Orthogonalization
    1. $r_k = \nabla f(x_k) = b - Qx_k$
    2.
5. Next point $x_{k+1} + \alpha_k p_k$
6. $x_{k+1} = x_k + \alpha p_k \rightarrow Qx_{k+1} - b = Qx_k - b + \alpha p_k$
    1. $\nabla f(x_{k+1}) = \nabla f(x_k) + \alpha Q p_k$

Bregman Divergence and Mirror Descent

1. Bregman Divergence
    1. Generalize squared Euclidean distance
    2. Definition 1 (Bregman divergence) Let $\psi : \Omega \rightarrow \mathbb{R}$ be a function that is: a) strictly convex, b) continuously differentiable, c) defined on a closed convex set Ω. Then the Bregman divergence is defined as

       $$\Delta_\psi(x, y) = \psi(x) - \psi(y) - \langle \nabla \psi(y), x - y \rangle, \quad \forall x, y \in \Omega$$

       That is, the difference between the value of ψ at x and the first order Taylor expansion of ψ around y evaluated at point x.
    3. examples:
        1. Euclidean distance: $\psi(x) = \frac{1}{2}\|x\|^2$
        2. KL divergence: $\psi(x) = \sum_i x_i \log x_i$
        3. L_p norm: $\psi(x) = \frac{1}{2}\|x\|_q^2, \frac{1}{p} + \frac{1}{q} = 1$
        4. strong convex case: $\psi(x) \geq \psi(y) + \langle \nabla \psi(y), x - y \rangle + \frac{\sigma}{2}\|x - y\|^2$
            1. $\Delta_\psi(x, y) \geq \frac{\sigma}{2}\|x - y\|^2$
    4. Property:
        1. Strict convexity
        2. Non-negativity
        3. Asymmetry
        4. Generalized triangle inequality
        5. gradient: $\frac{\partial}{\partial x}\Delta_\psi(x, y) = \nabla \psi(x) - \nabla \psi(y)$
    5. Projection: $x^* = \underset{x \in C}{\operatorname{argmin}} \Delta_\psi\left(x, x_0\right)$
        1. Pythagorean Theorem: $\Delta_\psi\left(y, x_0\right) \geq \Delta_\psi\left(y, x^*\right) + \Delta_\psi\left(x^*, x_0\right)$
        2. Proximal operator with Bregman Divergence $x^* = \underset{x \in C}{\operatorname{argmin}}\left\{L(x) + \Delta_\psi\left(x^*, x_0\right)\right\}$
            1. If $L(x)$ is convex:
                1. $L(y) + \Delta_\psi\left(y, x_0\right) \geq L\left(x^*\right) + \Delta_\psi\left(x^*, x_0\right) + \Delta_\psi\left(y, x^*\right)$
2. Mirror Descent:
    1. Local approximation under L-2 distance:
        1. $f(x) \approx f\left(x_k\right) + \nabla f\left(x_k\right)^\top\left(x - x_k\right) + \frac{1}{2\alpha_k}\left(x - x_k\right)^\top S^{-1}\left(x - x_k\right)$
        2. Gradient descent: $-\alpha_k \nabla f(x_k)$
    2. Approximation with Bregman divergence

1. $x_{k+1}=\underset{x \in C}{\operatorname{argmin}}\left\{f\left(x_k\right)+\left\langle g_k, x-x_k\right\rangle+\frac{1}{\alpha_k} \Delta_\psi\left(x, x_k\right)\right\}$
2. unconstrained case:
    1. $x_{k+1}=(\nabla \psi)^{-1}\left(\nabla \psi\left(x_k\right)-\alpha_k g_k\right)$

Constrained MDP

1. Maximize reward: $J^R(\pi) \doteq \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{\infty} \gamma^t R\left(s_t, a_t\right)\right]$
2. Constrained by total cost of constraints violation: $J^C(\pi) \doteq \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{\infty} \gamma^t C\left(s_t, a_t\right)\right] \leq h$
3. Policy improvement theorem: $J^R\left(\pi^{\prime}\right)-J^R(\pi)=\frac{1}{1-\gamma} \mathbb{E}_{\substack{s \sim d^{\pi^{\prime}} \\ a \sim \pi^{\prime}}}\left[A_R^\pi(s, a)\right]$
4. Solving framework
    1. Linear programming
    2. Lagrangian methods
        1. primal-dual methods
    3. Trust region optimization
        1. CPO
    4. Lyapunov functions

Trust-region-based

1. bound for policy update:
    1. for non-parametric moving average policy:
        1. $\eta\left(\pi_{\text {new }}\right) \leq L_{\pi_{\text {old }}}\left(\pi_{\text {new }}\right)+\frac{2 \epsilon \gamma}{(1-\gamma)^2} \alpha^2$
        2. $\epsilon=\epsilon=\max _s\left|\mathbb{E}_{a \sim \pi^{\prime}(a \mid s)}\left[A_\pi(s, a)\right]\right|$
        3. $\pi_{\text {new }}(a \mid s)=(1-\alpha) \pi_{\text {old }}(a \mid s)+\alpha \pi^{\prime}(a \mid s)$
        4. consider the change of state visitation probability
    2. for parametric policy with KL divergence bounded:
        1. $\eta(\tilde{\pi}) \leq L_\pi(\tilde{\pi})+C D_{\mathrm{KL}}^{\max }(\pi, \tilde{\pi}), \text { where } C=\frac{2 \epsilon \gamma}{(1-\gamma)^2}$
        2. $D_{\mathrm{TV}}^{\max }(\pi, \tilde{\pi})=\max _s D_{T V}(\pi(\cdot \mid s) \| \tilde{\pi}(\cdot \mid s))$
        3. ignore the change of state visitation probability
2. Approximation

$$\begin{aligned}
& \underset{\theta}{\operatorname{maximize}} L_{\theta_{\text {old }}}(\theta) \\
& \text { subject to } \bar{D}_{\mathrm{KL}}^{\rho_{\theta_{\text {old }}}}\left(\theta_{\text {old }}, \theta\right) \leq \delta
\end{aligned}$$

1. max KL divergence -> mean KL divergence for sampling
2. sample-based average
3. importance sampling to reuse samples

$$\begin{aligned}
& \underset{\theta}{\operatorname{minimize}} \mathbb{E}_{s \sim \rho_{\theta_{\text {old }}}, a \sim q}\left[\frac{\pi_\theta(a \mid s)}{q(a \mid s)} Q_{\theta_{\text {old }}}(s, a)\right] \\
& \text { subject to } \mathbb{E}_{s \sim \rho_{\theta_{\text {old }}}}\left[D_{\mathrm{KL}}\left(\pi_{\theta_{\text {old }}}(\cdot \mid s) \| \pi_\theta(\cdot \mid s)\right)\right] \leq \delta
\end{aligned}$$

4. first-order approximation of objective function
    1. $\left[\left.\nabla_\theta L_{\theta_{\text {old }}}(\theta)\right|_{\theta=\theta_{\text {old }}} \cdot\left(\theta-\theta_{\text {old }}\right)\right]$

5. second-order approximation of constraint
    1. KL-divergence
        1. Fish-information matrix: H
        2. $\frac{1}{2}\delta\left\|\theta-\theta_{\text{old}}\right\|^{T} H\left\|\theta-\theta_{\text{old}}\right\| \leq \delta$
        3. compute the H^{-1} using conjugate gradient
    2. L-2 distance
        1. $\frac{1}{2}\left\|\theta-\theta_{\text{old}}\right\|^{2} \leq \delta$

CPO: Constrained Policy Optimization (ICML 2017)

1. Joint optimization

$$
\begin{aligned}
\pi_{k+1} = \arg\max_{\pi \in \Pi_\theta} \underset{\substack{s \sim d^{\pi} k \\ a \sim \pi}}{\mathrm{E}} \left[A^{\pi_k}(s, a)\right] \\
\text{s.t. } J_{C_i}\left(\pi_k\right)+\frac{1}{1-\gamma} \underset{\substack{s \sim d \pi_k \\ a \sim \pi}}{\mathrm{E}}\left[A_{C_i}^{\pi_k}(s, a)\right] \leq d_i \quad \forall i \\
\bar{D}_{KL}\left(\pi \| \pi_k\right) \leq \delta .
\end{aligned}
$$

    1. approximation as TRPO
        1. first-order approximation of objective function
        2. first-order approximation of cost constraint
        3. second-order approximation of KL divergence
    2. backtracking line search is used to ensure surrogate constraint satisfaction

PCPO: Projection-Based Constrained Policy Optimization (ICLR 2020)

1. Two-step algorithm
    1. performs a local reward improvement update
    2. projecting the policy back onto the constraint set
2. Step 1: trust region policy optimization

$$
\begin{aligned}
\pi^{k+\frac{1}{2}} = \arg\max_{\pi} \mathbb{E}_{s \sim d^{\pi}}\left[A_R^{\pi^k}(s, a)\right] \\
\text{s.t. } \mathbb{E}_{s \sim d^{\pi^k}}\left[D_{\mathrm{KL}}\left(\pi \| \pi^k\right)[s]\right] \leq \delta
\end{aligned}
$$

    1. approximation as before
3. Step 2: constraint-satisfying projection

$$
\begin{aligned}
\pi^{k+1} = \arg\min_{\pi} D\left(\pi, \pi^{k+\frac{1}{2}}\right) \\
\text{s.t. } \quad J^C\left(\pi^k\right)+\underset{s \sim d^{\pi} k}{\mathbb{E}}\left[A_C^{\pi^k}(s, a)\right] \leq h
\end{aligned}
$$

    1. approximation as before

FOCOPS: First Order Constrained Optimization in Policy Space (NIPS 2020)

$$\begin{aligned}
\underset{\pi_\theta \in \Pi_\theta}{\text{maximize}} \quad & \underset{\substack{s\sim d^{\pi_{\theta_k}} \\ a\sim\pi_\theta}}{\mathbb{E}} \left[A^{\pi_{\theta_k}}(s,a)\right] \\
\text{subject to} \quad & J_C(\pi_{\theta_k}) + \frac{1}{1-\gamma} \underset{\substack{s\sim d^{\pi_{\theta_k}} \\ a\sim\pi_\theta}}{\mathbb{E}} \left[A_C^{\pi_{\theta_k}}(s,a)\right] \le b \\
& \bar{D}_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\theta_k}) \le \delta.
\end{aligned}$$

1. Errors in CPO
    1. Sampling error
    2. Approximation error
    3. conjugate gradient error
2. Two-step algorithm (use formulation in CPO)
    1. Solve in nonparameterized policy space

$$\pi^*(a|s) = \frac{\pi_{\theta_k}(a|s)}{Z_{\lambda,\nu}(s)} \exp\left(\frac{1}{\lambda}\left(A^{\pi_{\theta_k}}(s,a) - \nu A_C^{\pi_{\theta_k}}(s,a)\right)\right)$$

        1. Z: normalization constant
        2. dual variable λ, ν solved by dual function:
            1. $\min_{\lambda,\nu\ge 0} \lambda\delta + \nu\tilde{b} + \lambda \underset{\substack{s\sim d^{\pi_{\theta_k}} \\ a\sim\pi^*}}{\mathbb{E}} \left[\log Z_{\lambda,\nu}(s)\right]$
    2. Project back into the parameterized policy space

$$\mathcal{L}(\theta) = \underset{s\sim d^{\pi_{\theta_k}}}{\mathbb{E}} \left[D_{\mathrm{KL}}\left(\pi_\theta \| \pi^*\right)[s]\right]$$

3. Practical Implementation
    1. approximate dual variable: $\frac{\partial L(\pi^*,\lambda,\nu)}{\partial \nu} = 0$

Lyapunov Optimization in Stochastic Networks

1. Stochastic optimization problem
    1. $\mathcal{P}_2 : \min_{\forall t, \boldsymbol{\alpha}(t)\in\mathcal{A}^m} \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[p(t)]$
    2. $\text{s.t. } \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[y_k(t)\right] \le 0, k \in \{1,\ldots,K\}$
2. Virtual Queues
    1. queue length: $Q_k(t) = \max\{Q_k(t) + y_K(t), 0\}$
        1. $\sum_{t=0}^{T-1} y_k(t) \le Q_k(T) - Q_k(0) = Q_k(T)$
        2. $Q_k(t)^2 \le (Q_k(t) + y_K(t))^2$
        3. $\frac{1}{2}\sum_{k=1}^{K} Q_k(t+1)^2 \le \frac{1}{2}\sum_{k=1}^{K} Q_k(t)^2 + \frac{1}{2}\sum_{k=1}^{K} y_k(t)^2 + \sum_{k=1}^{K} Q_k(t)y_k(t)$
    2. Lyapunov function
        1. $L(K) = \frac{1}{2}\sum_{t=1}^{K} Q(t)^2$
        2. $\Delta L(K) = L(K+1) - L(K) \le \frac{1}{2}\sum_{k=1}^{K} y_k(t)^2 + \sum_{k=1}^{K} Q_k(t)y_k(t)$
        3. Denote B is the upper bound of $\frac{1}{2}\sum_{k=1}^{K} y_k(t)^2$
            1. $\Delta L(K) \le B + \sum_{k=1}^{K} Q_k(t)y_k(t)$
3. Drift-plus-penalty Algorithm

1. $\mathcal{P}_2 : \min_{\forall t, \boldsymbol{\alpha}(t) \in \mathcal{A}^m} \mathbb{E}[\Delta L(K) + Vp(t)]$
    1. approximation of original problem
    2. upper bound: $B + \sum_{k=1}^{K} Q_k(t) y_k(t) + Vp(t)$
4. Performance Analysis
    1. Average goal: $O(\frac{1}{V})$
    2. Average queue: $O(V)$

SDQN: A Lyapunov-based Approach to Safe Reinforcement Learning (NIPS 2018)

1. problem:

> **Problem $\mathcal{OPT}$:** Given an initial state x_0 and a threshold d_0, solve $\min_{\pi \in \Delta} \left\{ \mathcal{C}_\pi(x_0) : \mathcal{D}_\pi(x_0) \leq d_0 \right\}$. If there is a non-empty solution, the optimal policy is denoted by π^*.

    1. minimize cost only
2. Lyapunov function
    1. analyze the stability of dynamic systems
        1. tracking the energy that a system continually dissipates
    2. represent abstract quantities in a system
        1. steady-state performance of a Markov process
3. Lyapunov for CMDPs
    1. $\mathcal{L}_{\pi_B}(x_0, d_0)$
        1. transient state: $T_{\pi_B,d}[L](x) \leq L(x), \forall x \in \mathcal{X}'$
            1. contraction mapping
        2. terminal state: $L(x) = 0, \forall x \in \mathcal{X} \backslash \mathcal{X}'$
            1. terminal state with 0 cost
        3. initial state: $L(x_0) \leq d_0$
            1. satisfy the constrain threshold
    2. Relation between cost value function and Lyapunov function
        1. exist ϵ that $L_\epsilon(x) = \mathbb{E}\left[\sum_{t=0}^{\mathrm{T}^*-1} d(x_t) + \epsilon(x_t) \mid \pi_B, x\right]$
        2. upper bound of optimal cost value function
    3. Solve ϵ
        1. safety condition: $d_0 \geq L_{\tilde{\epsilon}}(x) \geq T_{\pi_B,d}[L_{\tilde{\epsilon}}](x)$
        2. solve linear programming

$$\tilde{\epsilon} \in \arg\max_{\epsilon : \mathcal{X}' \to \mathbb{R}_{\geq 0}} \left\{ \sum_{x \in \mathcal{X}'} \epsilon(x) : d_0 - \mathcal{D}_{\pi_B}(x_0) \geq \mathbf{1}(x_0)^\top \left(I - \{P(x' \mid x, \pi_B)\}_{x, x' \in \mathcal{X}'} \right)^{-1} \epsilon \right\}$$

4. Safe update:
    1. state-action Lyapunov function:
        1. $Q_L(x, a) = d(x) + \tilde{\epsilon}(x) + \sum_{x'} P(x' \mid x, a) L_{\tilde{\epsilon}'}(x')$
    2. L_{π_B} induced policy set:
        1. $(\pi(\cdot \mid x) - \pi_B(\cdot \mid x))^\top Q_L(x, \cdot) \leq \tilde{\epsilon}(x)$
    3. update policy:

1. $\pi'(\cdot \mid x) \in \arg\min_{\pi\in\Delta} \left\{\pi(\cdot \mid x)^\top Q(x,\cdot)\right\}$
2. Linear programming

SPG: Lyapunov-based Safe Policy Optimization for Continuous Control (ICML 2019)

1. Safe policy optimization:
   1. $\pi'(\cdot \mid x) \in \arg\min_{\pi\in\Delta} \left\{\pi(\cdot \mid x)^\top Q(x,\cdot)\right\}$
   2. $(\pi(\cdot \mid x) - \pi_B(\cdot \mid x))^\top Q_L(x,\cdot) \le \tilde{\epsilon}(x)$
   3. two efficient algorithm
      1. $\theta-$projection
      2. $\alpha-$projection
2. $\theta-$projection
   1. trust region optimization

$$\begin{aligned}\mathcal{C}'_{\pi_\theta}\left(x_0; \pi_{\theta_B}\right) =& \mathcal{C}_{\pi_{\theta_B}}\left(x_0\right) + \beta \bar{D}_{\mathrm{KL}}\left(\theta, \theta_B\right) + \\ & \mathbb{E}_{x\sim\mu_{\theta_B, x_0}, a\sim\pi_\theta}\left[Q_{V_{\theta_B}}(x,a) - V_{\theta_B}(x)\right]\end{aligned}$$

      1. first-order approximation
      2. average constraint surrogate
3. $\alpha-$projection
   1. safety layer
      1. embed the set of Lyapunov constraints into the policy network
         1. project action under Lyapunov constraints
         2. first-order approximation
         3. KKT condition -> OPT-Net
      2. an unconstrained optimization problem

LBPO: Lyapunov Barrier Policy Optimization (2021)

1. problem

$$\max_{\pi\in\mathcal{P}}[J_\pi(s_0)] \text{ s.t } D_\pi(s_0) \le d_0$$

2. Update policies inside the L_{π_B} induced policy set
   1. Q-value Evaluation
   2. Safe Policy Improvement
      1. $\pi_+(. \mid s) = \max_{\pi\in\mathcal{P}} J_\pi(s_0)$
      2. $\int_{a\in\mathcal{A}} (\pi(a \mid s) - \pi_B(a \mid s))\, Q_{L_{\pi_B},\hat{\epsilon}}(s,a) da \le \hat{\epsilon}(s)$
      3. convert constrain as log-barrier function

Primal-Dual Optimization

Primal methods

CRPO: A New Approach for Safe Reinforcement Learning with Convergence Guarantee

1. Primal-approach (The alternating mirror descent SA algorithm)

        1. convergence guaranteed
    2. Two-step:
        1. policy evaluation:
            1. reward value function
            2. cost value function
        2. policy update and constrain update
            1. if constraint is satisfied: update policy using reward value function
            2. if not satisfied: update policy using cost value function

Primal-dual methods

PDO: Risk-Constrained Reinforcement Learning with Percentile Risk Criteria

1. Lagrangian Approach and Reformulation
    1. primal-dual descent-ascent algorithm
    2. sample average estimation

RCPO: Reward Constrained Policy Optimization

1. handle discounted sum and mean constraints
2. Lagrangian:
    1. $\min_{\lambda \geq 0} \max_\theta L(\lambda, \theta) = \min_{\lambda \geq 0} \max_\theta \left[J_R^{\pi_\theta} - \lambda \cdot \left(J_C^{\pi_\theta} - \alpha\right)\right]$
3. Penalized reward functions
    1. $\hat{r}(\lambda, s, a) \triangleq r(s, a) - \lambda c(s, a)$
    2. update actor and critic using penalized value function
    3. update λ

OPDOP: Provably Efficient Safe Exploration via Primal-Dual Policy Optimization

1. Lagrangian:
    1. $\min_{\lambda \geq 0} \max_\theta L(\lambda, \theta) = \min_{\lambda \geq 0} \max_\theta \left[J_R^{\pi_\theta} - \lambda \cdot \left(J_C^{\pi_\theta} - \alpha\right)\right]$
2. utility function over K episodes

$$\operatorname{Regret}(K) = \sum_{k=1}^{K} \left(V_{r,1}^{\pi^\star}(x_1) - V_{r,1}^{\pi^k}(x_1)\right)$$
$$\text{Violation }(K) = \sum_{k=1}^{K} \left(b - V_{g,1}^{\pi^k}(x_1)\right)$$

3. Learning process
    1. policy evaluation: Least-Squares Temporal Difference
    2. primal update: KL divergence penalized update
    3. dual update: upper bounded gradient

CPPO: Responsive Safety in Reinforcement Learning by PID Lagrangian Methods

1. Lagrangian approaches are in oscillations and over-shoot

1. apply PID to adjust dual variable

$$
\begin{aligned}
&6: && \Delta \leftarrow J_C - d \\
&7: && \partial \leftarrow (J_C - J_{C,prev})_+ \\
&8: && I \leftarrow (I + \Delta)_+ \\
&9: && \lambda \leftarrow (K_P \Delta + K_I I + K_D \partial)_+ \\
&10: && J_{C,prev} \leftarrow J_C
\end{aligned}
$$

Convergent Policy Optimization for Safe Reinforcement Learning (NIPS 2019)

1. Lagrangian:
2. Successive convex relaxation:
   1. Both value and constraint

Safe layer

Safe Exploration in Continuous Action Spaces

1. only for immediate-constraint functions
2. linearization cost function:
   1. $\bar{c}_i\left(s'\right) \triangleq c_i(s,a) \approx \bar{c}_i(s) + g\left(s; w_i\right)^\top a$
3. project action: convex optimization

$$
\begin{aligned}
a^* = & \arg\min_a \frac{1}{2}\left\|a - \mu_\theta(s)\right\|^2 \\
& \text{s.t. } \bar{c}_i(s) + g\left(s; w_i\right)^\top a \leq C_i \forall i \in [K]
\end{aligned}
$$

Evolutionary approach

Constrained Cross-Entropy Method for Safe Reinforcement Learning (NIPS 2018)

1. Sampling and sorting