Advanced Policy-Based methods. TRPO, PPO

Резяпкин Вячеслав

23.07.2019

Tinkoff.ru

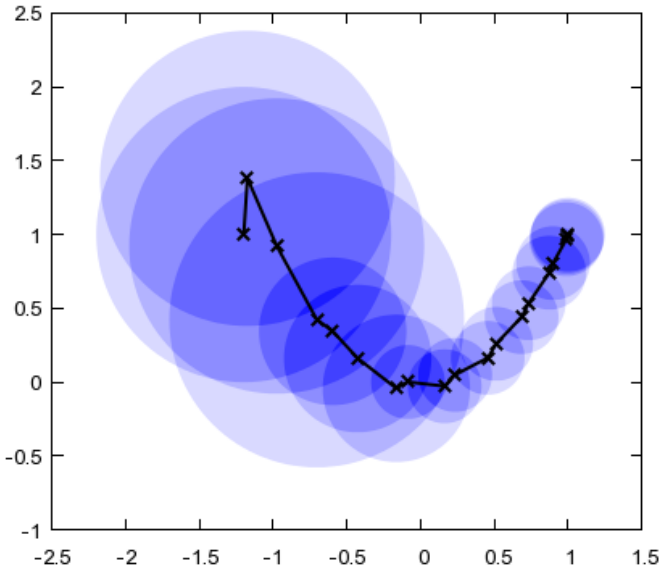
Два типа алгоритмов оптимизации

1. **Line search**
    1. Выбрать направление шага (антиградиент, например)
    2. Выбрать длину шага (learning rate)
2. **Trust region**
    1. Выбрать окрестность для поиска
    2. Выбрать точку в окрестности

Trust region methods

Идея:

- Аппроксимировать исходную сложную функцию $f(x)$ (нейронную сеть) более простой $m(x) \cong f(x)$
- Найти минимум простой приближающей функции $m(x)$ в области, где она хорошо приближает $f(x)$



Trust region methods

- В качестве $m(x)$ часто выбирается разложение функции в ряд Тейлора до второго порядка

$$f(x) \cong m(x) = f(x_k) + (x - x_k)\nabla f(x_k) + \frac{1}{2}(x - x_k)\nabla^2 \mathrm{f}(\mathrm{x_k})(x - x_k)$$

Почему? Потому что задача оптимизации квадратичных функций хорошо изучена

- В качестве trust region используется Δ -окрестность точки x_k

$$||x - x_k|| < \Delta$$

Алгоритм:

Повторять

1. *решить* $x_{k+1} = \underset{||x-x_k||<\Delta}{\operatorname{argmin}}\ m(x)$
2. *изменить* Δ *при необходимости*

Trust region methods

Если гессиан $\mathbf{H} = \boldsymbol{\nabla}^{\mathbf{2}}\mathbf{f}(\mathbf{x_k})$ положительно определенный, то это хорошо, ведь:

- Задача выпуклая
- Есть аналитическое решение $\mathrm{x}^* = x_k - \mathrm{H}^{-1}\nabla f(x_k)$

$$m(x) = f(x_k) + (x - x_k)\nabla f(x_k) + \frac{1}{2}(x - x_k)\mathrm{H}(x - x_k)$$

$$\nabla m(x) = \nabla f(x_k) + \mathrm{H}(x - x_k) = 0$$

$$\mathrm{x}^* = x_k - \mathrm{H}^{-1}\nabla f(x_k)$$

Иначе задача невыпуклая, но существуют эффективные методы, основанные на градиентном спуске

Policy gradients

Обучение модели:
1. Сэмплирование траекторий
2. Обновление параметров
$$\theta_{k+1} = \theta_k + \alpha \nabla_\theta J(\theta)$$

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[R(\tau)] = \mathbb{E}_{s_0 \sim p_0}[V(s_0)] \to max$$

Собираем траектории и оцениваем градиент

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \sum_t r_t * \sum_t \nabla \log \pi(s_t, a_t)$$

Reward-to-go

$$= \mathbb{E}_{\tau \sim \pi_\theta} \sum_{t=1}^{T} \nabla \log \pi(s_t, a_t) * \sum_{t'=t}^{T} r_t$$

Определение Q-функции

$$= \mathbb{E}_{\tau \sim \pi_\theta} \sum_t Q(s_t, a_t) * \nabla \log \pi(s_t, a_t) =$$

бейзлайн

$$\mathbb{E}_{\tau \sim \pi_\theta} \sum_t \big(Q(s_t, a_t) - V(s_t)\big) * \nabla \log \pi(s_t, a_t) =$$

Определение Advantage

$$\mathbb{E}_{\tau \sim \pi_\theta} \sum_t A(s_t, a_t) * \nabla \log \pi(s_t, a_t)$$

Policy gradients. Baseline

$$\mathbb{E}_{\tau \sim \pi_\theta} \sum_t \quad Q(s_t, a_t) * \nabla \log \pi(s_t, a_t) =$$

$$\mathbb{E}_{\tau \sim \pi_\theta} \sum_t \quad \big(Q(s_t, a_t) - V(s_t)\big) * \nabla \log \pi(s_t, a_t)$$

бейзлайн

$$\mathbb{E}_{\tau \sim \pi_\theta} \sum_t V(s_t) * \nabla \log \pi(s_t, a_t) =$$

$$\sum_t \mathbb{E}_{s_t \sim p, \pi_\theta} \mathbb{E}_{a_t \sim \pi_\theta(s_t)} V(s_t) * \nabla \log \pi(s_t, a_t) =$$

$$\sum_t \mathbb{E}_{s_t \sim p, \pi_\theta} V(s_t) \mathbb{E}_{a_t \sim \pi_\theta(s_t)} \nabla \log \pi(s_t, a_t) =$$

$$\sum_t \mathbb{E}_{s_t \sim p, \pi_\theta} V(s_t) \nabla \mathbb{E}_{a_t \sim \pi_\theta(s_t)} \pi(s_t, a_t) =$$

$$\sum_t \mathbb{E}_{s_t \sim p, \pi_\theta} V(s_t) \nabla 1 \quad = 0$$

Trusted Region Policy Optimization(TRPO)

Можно доказать равенство – связь оценок двух разных политик:

$$\mathrm{J}(\tilde{\pi}) = J\left(\pi_{old}\right) + \mathbb{E}_{\tau\sim\tilde{\pi}}\left[\sum_t A_{\pi_{old}}\left(s_t, a_t\right)\right]$$

Ожидаемая награда с новой политикой

Ожидаемая награда со старой политикой

Траектории сэмплируются новой политикой

Оценка Advantage функции по старой политике

Доказывать не будем, но убедимся, что оно выполняется при равенстве политик

$$A(s,a) = Q(s,a) - V(s)$$
$$\mathbb{E}_{a\sim\pi_\theta(s)}A(s,a) = \mathbb{E}_{a\sim\pi_\theta(s)}[Q(s,a) - V(s)] =$$
$$\mathbb{E}_{a\sim\pi_\theta(s)}[Q(s,a)] - V(s) = V(s) - V(s) = 0$$

Trusted Region Policy Optimization(TRPO)

Можно доказать равенство – связь оценок двух разных политик:

$$\mathrm{J}(\tilde{\pi}) = J\left(\pi_{old}\right) + \mathbb{E}_{\tau\sim\tilde{\pi}}\left[\sum_t A_{\pi_{old}}\left(s_t, a_t\right)\right]$$

Ожидаемая награда с новой политикой

Ожидаемая награда со старой политикой

Траектории сэмплируются новой политикой

Оценка Advantage функции по старой политике

Как это максимизировать?
Мат.ожидание берется по траекториям новой политики, которых у нас еще нет.
Зато есть старые траектории. Можно ли как-то оценить эту функцию на старых траекториях?

Trusted Region Policy Optimization(TRPO)

Предыдущее неравенство можно переписать как:

$$\mathrm{J}(\tilde{\pi}) = J\left(\pi_{old}\right) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_{\pi}(s,a)$$

Discounted visitation frequency

$$\rho_{\pi}(s) = P\left(s_0 = s\right) + \gamma P\left(s_1 = s\right) + \gamma^2 P + \cdots$$

Trusted Region Policy Optimization(TRPO)

$$\mathrm{J}(\tilde{\pi}) = J\left(\pi_{old}\right) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_{\pi}(s,a)$$

1 2

Discounted visitation frequency

$$\rho_{\pi}(s) = P\left(s_0 = s\right) + \gamma P\left(s_1 = s\right) + \gamma^2 P + \cdots$$

2

importance sampling

$$\begin{aligned} E_{x\sim p(x)}[f(x)] &= \int p(x) f(x) dx \\ &= \int \frac{q(x)}{q(x)} p(x) f(x) dx \\ &= \int q(x) \frac{p(x)}{q(x)} f(x) dx \\ &= E_{x\sim q(x)}\left[\frac{p(x)}{q(x)} f(x)\right] \end{aligned}$$

2 $\mathbb{E}_{a\sim\pi_{new}(s)} A_{\pi_{old}}(s,a) = \mathbb{E}_{a\sim\pi_{old}(s)} \frac{\pi_{new}(s)}{\pi_{old}(s)} A_{\pi_{old}}(s,a)$

Trusted Region Policy Optimization(TRPO)

$$\mathrm{J}(\tilde{\pi}) = J\left(\pi_{old}\right) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_{\pi}(s,a)$$

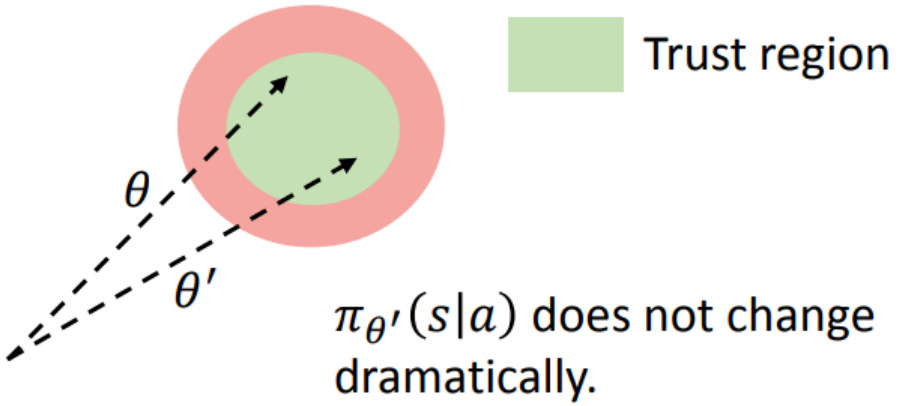
$$\mathrm{L}(\tilde{\pi}) = J\left(\pi_{old}\right) + \sum_s \rho_{\pi}(s) \sum_a \tilde{\pi}(a|s) A_{\pi}(s,a)$$

L – локальная аппроксимация J

Trusted Region Policy Optimization(TRPO)

$$\mathrm{L}(\tilde{\pi}) = J\left(\pi_{old}\right) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$

L – локальная аппроксимация J
в доверительной области параметров



Trusted Region Policy Optimization(TRPO)

$$\mathrm{J}(\tilde{\pi}) = J\left(\pi_{old}\right) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$

Trust Region trick:

If $E_s\left[KL(\pi \,||\, \tilde{\pi})\right]$ is small,

$$J(\tilde{\pi}) \approx J(\pi) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$

Дивергенция Кульбака-Лейблера:

$$D_{KL}(p||q) = \int_X p(x) \log \frac{p(x)}{q(x)} dx$$

$$J(\pi_{old}) = L(\pi_{old})$$
$$\nabla J\Big|_{\pi=\pi_{old}} = \nabla L\Big|_{\pi=\pi_{old}}$$

дз

Trusted Region Policy Optimization(TRPO)

$$\mathrm{L}(\tilde{\pi}) = J(\pi_{old}) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$

Можно показать, что выполняется неравенство

$$\mathrm{J}(\tilde{\pi}) \geq L(\tilde{\pi}) - C D_{KL}^{\max}(\pi, \tilde{\pi})$$

Дивергенция Кульбака-Лейблера:

$$D_{KL}(p||q) = \int_X p(x) \log \frac{p(x)}{q(x)} dx$$

$$\mathrm{C} = \frac{4\gamma}{(1-\gamma)^2} \max_{s,a} |A(s,a)|$$

$$J(\pi_{old}) = L(\pi_{old})$$
$$\nabla J\Big|_{\pi=\pi_{old}} = \nabla L\Big|_{\pi=\pi_{old}}$$

дз

Это означает, что максимизируя правую часть неравенства,
мы автоматически максимизируем левую часть

Trusted Region Policy Optimization(TRPO)

$$\mathrm{L}(\tilde{\pi}) = J\left(\pi_{old}\right) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$

$$\mathrm{J}(\tilde{\pi}) \geq L(\tilde{\pi}) - C D_{KL}^{\max}(\pi, \tilde{\pi})$$

Гарантированное улучшение политики:

$$\pi = \arg\max_{\tilde{\pi}} \left[L(\tilde{\pi}) - C D_{KL}^{\max}(\pi, \tilde{\pi}) \right]$$

Trusted Region Policy Optimization(TRPO)

На практике используется средняя дивергенция вместо максимальной

$$\underset{\theta}{\text{maximize}}\, L(\theta) - C.\overline{D_{KL}}(\theta_{old}, \theta)$$

Trust region оптимизационная задача:

Приближение L Приближение KL

$$\underset{\theta}{\text{maximize}}\, g\,.\,(\theta - \theta_{old}) - \frac{c}{2}(\theta - \theta_{old})^T F(\theta - \theta_{old})$$

$$\text{where, } g = \frac{\partial}{\partial\theta} L(\theta)|_{\theta=\theta_{old}}, \qquad F = \frac{\partial^2}{\partial^2\theta}\overline{D_{KL}}(\theta_{old}, \theta)|_{\theta=\theta_{old}}$$

Где член первого порядка ряда Тейлора у дивергенции и нулевого порядка у обоих членов?

Trusted Region Policy Optimization(TRPO)

$$\max_{\theta} imize\ g\ .\ (\theta - \theta_{old}) - \frac{c}{2}(\theta - \theta_{old})^T F(\theta - \theta_{old})$$

where, $g = \frac{\partial}{\partial\theta} L(\theta)|_{\theta=\theta_{old}}$, $\quad F = \frac{\partial^2}{\partial^2\theta} \overline{D_{KL}}(\theta_{old}, \theta)|_{\theta=\theta_{old}}$

Решение

$$\theta - \theta_{old} = \frac{1}{c} F^{-1} g$$

Не хочется обращать гессиан
Воспользуемся тем, что мы хотим посчитать произведение $F^{-1}g$
Оно равняется решению уравнения $Fx = g$
Эффективно решается с помощью метода сопряженных градиентов

Trusted Region Policy Optimization(TRPO)

Как выбирать C?

- Unconstrained problem: $\max_{\theta} imize\ L(\theta) - C.\overline{D_{KL}}(\theta_{old}, \theta)$
- Constrained problem: $\max_{\theta} imize\ L(\theta)$ subject to $C.\overline{D_{KL}}(\theta_{old}, \theta) \leq \delta$
- δ is a hyper-parameter, remains fixed over whole learning process
- Solve constrained quadratic problem: compute $F^{-1}g$ and then rescale step to get correct KL
  - $\max_{\theta} imize\ g\ .\ (\theta - \theta_{old})$ subject to $\frac{1}{2}(\theta - \theta_{old})^T F(\theta - \theta_{old}) \leq \delta$
  - Lagrangian: $\mathcal{L}(\theta, \lambda) = g\ .\ (\theta - \theta_{old}) - \frac{\lambda}{2}[(\theta - \theta_{old})^T F(\theta - \theta_{old}) - \delta]$
  - Differentiate wrt θ and get $\theta - \theta_{old} = \frac{1}{\lambda}F^{-1}g$
  - We want $\frac{1}{2}s^T Fs = \delta$
  - Given candidate step $s_{unscaled}$ rescale to $s = \sqrt{\frac{2\delta}{s_{unscaled}.(Fs_{unscaled})}}\, s_{unscaled}$

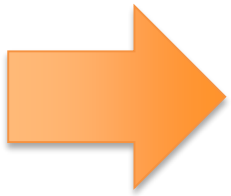
Linesearch

Из-за
- перехода от C к δ
- приближенных вычислений (сопряженные градиенты, Тейлор)
- семплирования вместо матожиданий
- …

данный шаг может оказаться слишком большим

Проверим необходимые условия и, если они не выполняются, уменьшим длину, не меняя направления

$$\theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{g^T H^{-1} g}} H^{-1} g$$



$$\theta_{k+1} = \theta_k + \alpha^j \sqrt{\frac{2\delta}{g^T H^{-1} g}} H^{-1} g$$

j – мин. целое при котором:

$$\mathcal{L}(\theta_k, \theta_{k+1}) > 0$$
$$\overline{D}_{KL}(\theta_{k+1} || \theta_k) \leq \delta$$

Trusted Region Policy Optimization(TRPO)

Algorithm 1 Trust Region Policy Optimization

Input: initial policy parameters θ_0, initial value function parameters ϕ_0

Hyperparameters: KL-divergence limit δ, backtracking coefficient α, maximum number of backtracking steps K

for $k = 0, 1, 2, ...$ **do**

Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.

Compute rewards-to-go $\hat{R}_t$.

Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{ϕ_k}.

Estimate policy gradient as

$$\hat{g}_k = \frac{1}{|\mathcal{D}_k|} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)|_{\theta_k} \hat{A}_t.$$

Use the conjugate gradient algorithm to compute

$$\hat{x}_k \approx \hat{H}_k^{-1} \hat{g}_k,$$

where $\hat{H}_k$ is the Hessian of the sample average KL-divergence.

Update the policy by backtracking line search with

$$\theta_{k+1} = \theta_k + \alpha^j \sqrt{\frac{2\delta}{\hat{x}_k^T \hat{H}_k \hat{x}_k}} \hat{x}_k,$$

where $j \in \{0, 1, 2, ...K\}$ is the smallest value which improves the sample loss and satisfies the sample KL-divergence constraint.

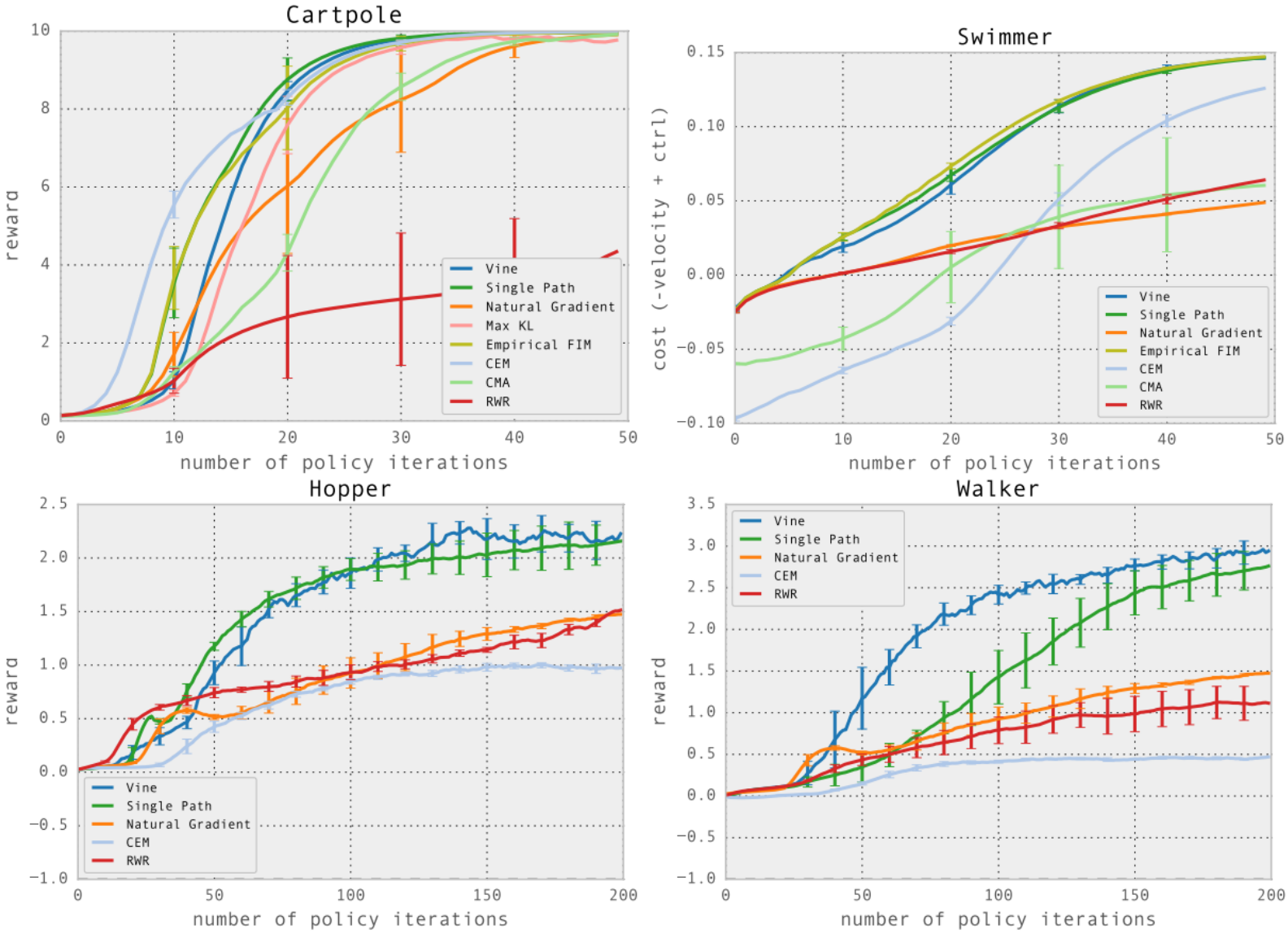
Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_\phi \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2,$$

typically via some gradient descent algorithm.

end for

Результаты


Cartpole
reward
number of policy iterations
Vine
Single Path
Natural Gradient
Max KL
Empirical FIM
CEM
CMA
RWR
Swimmer
cost (-velocity + ctrl)
number of policy iterations
Vine
Single Path
Natural Gradient
Empirical FIM
CEM
CMA
RWR
Hopper
reward
number of policy iterations
Vine
Single Path
Natural Gradient
CEM
RWR
Walker
reward
number of policy iterations
Vine
Single Path
Natural Gradient
CEM
RWR

Результаты

	B. Rider	*Breakout*	*Enduro*	*Pong*	*Q*bert*	*Seaquest*	*S. Invaders*
Random	354	1.2	0	−20.4	157	110	179
Human (Mnih et al., 2013)	7456	31.0	368	−3.0	18900	28010	3690
Deep Q Learning (Mnih et al., 2013)	4092	168.0	470	20.0	1952	1705	581
UCC-I (Guo et al., 2014)	5702	380	741	21	20025	2995	692
TRPO - single path	1425.2	10.8	534.6	20.9	1973.5	1908.6	568.4
TRPO - vine	859.5	34.2	430.8	20.9	7732.5	788.4	450.2

Адаптивное изменение параметра регуляризации

- Using several epochs of minibatch SGD, optimize the KL-penalized objective

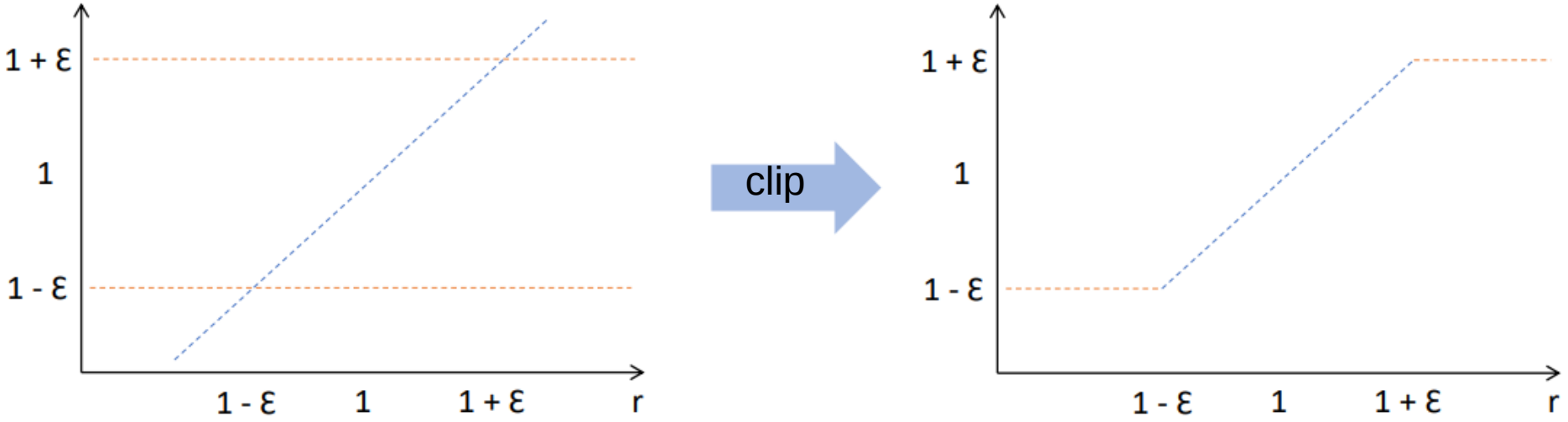
$$L^{KLPEN}(\theta) = \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t - \beta \, \text{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)] \right]$$

- Compute $d = \hat{\mathbb{E}}_t[\text{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]]$
  - If $d < d_{\text{targ}}/1.5$, $\beta \leftarrow \beta/2$
  - If $d > d_{\text{targ}} \times 1.5$, $\beta \leftarrow \beta \times 2$

Proximal Policy Optimization(PPO)

По-прежнему будем максимизировать приближение L, но теперь проще

$$\underset{\theta}{\text{maximize}} \quad \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right] \qquad r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$



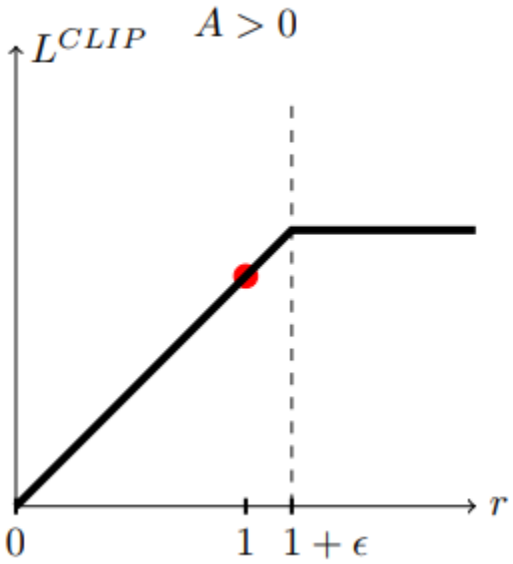
$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right]$$

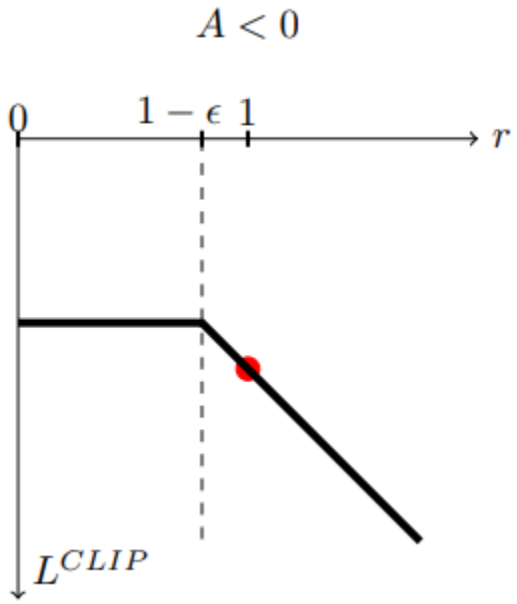
Proximal Policy Optimization(PPO)

Будем ограничивать шаги в сторону улучшения политики

$$\underset{\theta}{\text{maximize}} \quad \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t \right] \qquad r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$

$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t \left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right]$$





Результаты

algorithm	avg. normalized score
No clipping or penalty	-0.39
Clipping, $\epsilon = 0.1$	0.76
Clipping, $\epsilon = 0.2$	**0.82**
Clipping, $\epsilon = 0.3$	0.70
Adaptive KL $d_{\text{targ}} = 0.003$	0.68
Adaptive KL $d_{\text{targ}} = 0.01$	0.74
Adaptive KL $d_{\text{targ}} = 0.03$	0.71
Fixed KL, $\beta = 0.3$	0.62
Fixed KL, $\beta = 1.$	0.71
Fixed KL, $\beta = 3.$	0.72
Fixed KL, $\beta = 10.$	0.69

Table 1: Results from continuous control benchmark. Average normalized scores (over 21 runs of the algorithm, on 7 environments) for each algorithm / hyperparameter setting . β was initialized at 1.

Сравнение

TRPO	PPO
• Works for smaller models • Second-order optimization	• Works for big models • First-order optimization