Semi-Supervised PageRank Model Learning with Gradient-Free Optimization Methods

Lev Bogolubsky[3], Pavel Dvurechensky[1,2], **Alexander Gasnikov**[1,2], Andrei Raigorodskii[1,3,4], Maxim Zhukovskii[1,3]

[1]MIPT, [2]IITP RAS, [3]Yandex, [4]MSU

20.05.2015

“The fifth International conference on network analysis”
Nizhny Novgorod

Outline

Outline

1. Learning problem formulation
2. Random gradient-free methods with inexact oracle

Outline

1. Learning problem formulation
2. Random gradient-free methods with inexact oracle
3. Bi-level method for learning problem

Outline

Model formulation: random walks

1. Query number $q \in 1, \ldots, Q$.

Model formulation: random walks

1. Query number $q \in 1, \ldots, Q$.
2. S_q – user session which is started from q.

Model formulation: random walks

1. Query number $q \in 1, \ldots, Q$.
2. S_q – user session which is started from q.
3. User browsing graph $G_q = (V_q, E_q)$: $V_q = V_q^1 \sqcup V_q^2$, V_q^1 – queries, V_q^2 – pages, $|V_q^1| = p_q$, $|V_q^2| = n_q$.

Model formulation: random walks

1. Query number $q \in 1, \dots, Q$.
2. S_q – user session which is started from q.
3. User browsing graph $G_q = (V_q, E_q)$: $V_q = V_q^1 \sqcup V_q^2$, V_q^1 – queries, V_q^2 – pages, $|V_q^1| = p_q$, $|V_q^2| = n_q$.
4. NB: $n_q \cong 10^9$

Model formulation: random walks

1. Query number $q \in 1, \ldots, Q$.
2. S_q – user session which is started from q.
3. User browsing graph $G_q = (V_q, E_q)$: $V_q = V_q^1 \sqcup V_q^2$, V_q^1 – queries, V_q^2 – pages, $|V_q^1| = p_q$, $|V_q^2| = n_q$.
4. NB: $n_q \cong 10^9$
5. $\varphi = (\varphi_1, \varphi_2)^T \in \mathbb{R}^{m_1+m_2}$ – unknown vector of parameters which helps to convert web-sites properties to their importance.

Model formulation: random walks

1. Query number $q \in 1, \ldots, Q$.
2. S_q – user session which is started from q.
3. User browsing graph $G_q = (V_q, E_q)$: $V_q = V_q^1 \sqcup V_q^2$, V_q^1 – queries, V_q^2 – pages, $|V_q^1| = p_q$, $|V_q^2| = n_q$.
4. NB: $n_q \cong 10^9$
5. $\varphi = (\varphi_1, \varphi_2)^T \in \mathbb{R}^{m_1+m_2}$ – unknown vector of parameters which helps to convert web-sites properties to their importance.
6. Example [Gao, Liu, Huazhong, Wang, Li, 2011]: $V_i \in \mathbb{R}^l$, $E_{ij} \in \mathbb{R}^s$ – such factors as number of visits, average time spent on a page, number of transitions, etc.

Model formulation: random walks

1. Query number $q \in 1, \ldots, Q$.
2. S_q – user session which is started from q.
3. User browsing graph $G_q = (V_q, E_q)$: $V_q = V_q^1 \sqcup V_q^2$, V_q^1 – queries, V_q^2 – pages, $|V_q^1| = p_q$, $|V_q^2| = n_q$.
4. NB: $n_q \cong 10^9$
5. $\varphi = (\varphi_1, \varphi_2)^T \in \mathbb{R}^{m_1+m_2}$ – unknown vector of parameters which helps to convert web-sites properties to their importance.
6. Example [Gao, Liu, Huazhong, Wang, Li, 2011]: $V_i \in \mathbb{R}^l$, $E_{ij} \in \mathbb{R}^s$ – such factors as number of visits, average time spent on a page, number of transitions, etc.
7. Importance given by $f_q(\varphi_1, i) = \langle \varphi_1, V_i \rangle$ and $g_q(\varphi_2, i \to j) = \langle \varphi_2, E_{ij} \rangle$.

Model formulation: random walks

1. Query number $q \in 1, \ldots, Q$.
2. S_q – user session which is started from q.
3. User browsing graph $G_q = (V_q, E_q)$: $V_q = V_q^1 \sqcup V_q^2$, V_q^1 – queries, V_q^2 – pages, $|V_q^1| = p_q$, $|V_q^2| = n_q$.
4. NB: $n_q \cong 10^9$
5. $\varphi = (\varphi_1, \varphi_2)^T \in \mathbb{R}^{m_1+m_2}$ – unknown vector of parameters which helps to convert web-sites properties to their importance.
6. Example [Gao, Liu, Huazhong, Wang, Li, 2011]: $V_i \in \mathbb{R}^l$, $E_{ij} \in \mathbb{R}^s$ – such factors as number of visits, average time spent on a page, number of transitions, etc.
7. Importance given by $f_q(\varphi_1, i) = \langle \varphi_1, V_i \rangle$ and $g_q(\varphi_2, i \to j) = \langle \varphi_2, E_{ij} \rangle$.
8. $m = m_1 + m_2 \cong 10^3$.

Markov chain

Probability for choosing query i, being at any vertex:

$$[\pi_q^0(\varphi)]_i = \frac{f_q(\varphi_1, i)}{\sum_{\tilde{i}\in V_q^1} f_q(\varphi_1, \tilde{i})}$$

Markov chain

Probability for choosing query i, being at any vertex:

$$[\pi_q^0(\varphi)]_i = \frac{f_q(\varphi_1, i)}{\sum_{\tilde{i} \in V_q^1} f_q(\varphi_1, \tilde{i})}$$

Probability of transition $\tilde{i} \to i$:

$$\frac{g_q(\varphi_2, \tilde{i} \to i)}{\sum_{j:\tilde{i} \to j} g_q(\varphi_2, \tilde{i} \to j)}$$

Markov chain

Probability for choosing query i, being at any vertex:

$$[\pi_q^0(\varphi)]_i = \frac{f_q(\varphi_1, i)}{\sum_{\tilde{i} \in V_q^1} f_q(\varphi_1, \tilde{i})}$$

Probability of transition $\tilde{i} \to i$:

$$\frac{g_q(\varphi_2, \tilde{i} \to i)}{\sum_{j: \tilde{i} \to j} g_q(\varphi_2, \tilde{i} \to j)}$$

Finally, probability of being at i at the step $t+1$, $t = 0, 1, \ldots$ equals

$$[\pi_q(t+1)]_i = \alpha \frac{f_q(\varphi_1, i)}{\sum_{\tilde{i} \in V_q^1} f_q(\varphi_1, \tilde{i})} + (1-\alpha) \sum_{\tilde{i}: \tilde{i} \to i \in E_q} \frac{g_q(\varphi_2, \tilde{i} \to i)}{\sum_{j: \tilde{i} \to j} g_q(\varphi_2, \tilde{i} \to j)} [\pi_q(t)]_{\tilde{i}}$$

Markov chain

Probability for choosing query i, being at any vertex:

$$[\pi_q^0(\varphi)]_i = \frac{f_q(\varphi_1, i)}{\sum_{\tilde{i}\in V_q^1} f_q(\varphi_1, \tilde{i})}$$

Probability of transition $\tilde{i} \to i$:

$$\frac{g_q(\varphi_2, \tilde{i} \to i)}{\sum_{j:\tilde{i}\to j} g_q(\varphi_2, \tilde{i} \to j)}$$

Finally, probability of being at i at the step $t+1$, $t = 0, 1, ...$ equals

$$[\pi_q(t+1)]_i = \alpha \frac{f_q(\varphi_1, i)}{\sum_{\tilde{i}\in V_q^1} f_q(\varphi_1, \tilde{i})} + (1-\alpha) \sum_{\tilde{i}:\tilde{i}\to i\in E_q} \frac{g_q(\varphi_2, \tilde{i} \to i)}{\sum_{j:\tilde{i}\to j} g_q(\varphi_2, \tilde{i} \to j)} [\pi_q(t)]_{\tilde{i}}$$

Stationary distribution of Markov chain defines the p-th web-page rank: $[\pi_q^*(\varphi)]_p$.

$$\pi_q^*(\varphi) = \alpha\pi_q^0(\varphi) + (1-\alpha)P_q^T(\varphi)\pi_q^*(\varphi).$$

Learning problem

- We have some pool of experts who give score from 1 to k to web-pages for Q queries.

Learning problem

- We have some pool of experts who give score from 1 to k to web-pages for Q queries.
- For every query q we have sets of pages $P_q^1, P_q^2, ..., P_q^k$ which are ordered from the most relevant to irrelevant pages. $\sum_{i=1}^{k} \left|P_q^i\right| = r_q$.

Learning problem

- We have some pool of experts who give score from 1 to k to web-pages for Q queries.
- For every query q we have sets of pages $P_q^1, P_q^2, ..., P_q^k$ which are ordered from the most relevant to irrelevant pages. $\sum_{i=1}^{k} \left|P_q^i\right| = r_q$.
- We choose loss function $h(i,j,x) = \max\{x + b_{ij}, 0\}^2$, where $1 \le i < j \le k$, $b_{ij} > 0$ is some threshold.

Learning problem

- We have some pool of experts who give score from 1 to k to web-pages for Q queries.
- For every query q we have sets of pages $P_q^1, P_q^2, ..., P_q^k$ which are ordered from the most relevant to irrelevant pages. $\sum_{i=1}^{k} \left|P_q^i\right| = r_q$.
- We choose loss function $h(i,j,x) = \max\{x + b_{ij}, 0\}^2$, where $1 \le i < j \le k$, $b_{ij} > 0$ is some threshold.
- The idea is that loss is positive if the MC ranking differs from experts' ranking.

Learning problem

- We have some pool of experts who give score from 1 to k to web-pages for Q queries.
- For every query q we have sets of pages $P_q^1, P_q^2, ..., P_q^k$ which are ordered from the most relevant to irrelevant pages. $\sum_{i=1}^{k} \left|P_q^i\right| = r_q$.
- We choose loss function $h(i,j,x) = \max\{x + b_{ij}, 0\}^2$, where $1 \le i < j \le k$, $b_{ij} > 0$ is some threshold.
- The idea is that loss is positive if the MC ranking differs from experts' ranking.
- To find φ we minimize

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q}\sum_{1\le i<j\le k}\sum_{p_1\in P_q^i, p_2\in P_q^j} h(i,j,[\pi_q]_{p_2} - [\pi_q]_{p_1})$$

Problem reformulation

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q \pi_q^*(\varphi) + b_q)_+\|_2^2 \to \min$$

$$\pi_q^*(\varphi) = \alpha \left[I - (1-\alpha) P_q^T(\varphi)\right]^{-1} \pi_q^0(\varphi) \Leftrightarrow \|\pi - \pi_q^*(\varphi)\|_1 \to \min\,.$$

Problem reformulation

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q \pi_q^*(\varphi) + b_q)_+\|_2^2 \to \min$$

$$\pi_q^*(\varphi) = \alpha \left[I - (1-\alpha)P_q^T(\varphi)\right]^{-1} \pi_q^0(\varphi) \Leftrightarrow \|\pi - \pi_q^*(\varphi)\|_1 \to \min\,.$$

[Nemirovski, Nesterov, 2012]: $\|\tilde{\pi}_q^N(\varphi) - \pi_q^*(\varphi)\|_1 \le 2(1-\alpha)^{N+1}$ holds for

$$\tilde{\pi}_q^N(\varphi) = \frac{\alpha}{1-(1-\alpha)^{N+1}} \sum_{i=0}^{N} (1-\alpha)^i \left[P_q^T(\varphi)\right]^i \pi_q^0(\varphi)$$

Problem reformulation

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q \pi_q^*(\varphi) + b_q)_+\|_2^2 \to \min$$

$$\pi_q^*(\varphi) = \alpha\left[I - (1-\alpha)P_q^T(\varphi)\right]^{-1}\pi_q^0(\varphi) \Leftrightarrow \|\pi - \pi_q^*(\varphi)\|_1 \to \min.$$

[Nemirovski, Nesterov, 2012]: $\|\tilde{\pi}_q^N(\varphi) - \pi_q^*(\varphi)\|_1 \le 2(1-\alpha)^{N+1}$ holds for

$$\tilde{\pi}_q^N(\varphi) = \frac{\alpha}{1-(1-\alpha)^{N+1}}\sum_{i=0}^{N}(1-\alpha)^i\left[P_q^T(\varphi)\right]^i \pi_q^0(\varphi)$$

To obtain vector $\tilde{\pi}_q^N(\varphi)$ s.t. $\|\tilde{\pi}_q^N(\varphi) - \pi_q^*(\varphi)\|_1 \le \Delta$ we need $\frac{s_q(p_q+n_q)}{\alpha}\ln\frac{2}{\Delta}$ a.o. and

Problem reformulation

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q \pi_q^*(\varphi) + b_q)_+\|_2^2 \to \min$$

$$\pi_q^*(\varphi) = \alpha \left[I - (1-\alpha) P_q^T(\varphi)\right]^{-1} \pi_q^0(\varphi) \Leftrightarrow \|\pi - \pi_q^*(\varphi)\|_1 \to \min\,.$$

[Nemirovski, Nesterov, 2012]: $\|\tilde{\pi}_q^N(\varphi) - \pi_q^*(\varphi)\|_1 \le 2(1-\alpha)^{N+1}$ holds for

$$\tilde{\pi}_q^N(\varphi) = \frac{\alpha}{1-(1-\alpha)^{N+1}} \sum_{i=0}^{N} (1-\alpha)^i \left[P_q^T(\varphi)\right]^i \pi_q^0(\varphi)$$

To obtain vector $\tilde{\pi}_q^N(\varphi)$ s.t. $\|\tilde{\pi}_q^N(\varphi) - \pi_q^*(\varphi)\|_1 \le \Delta$ we need $\frac{s_q(p_q+n_q)}{\alpha} \ln \frac{2}{\Delta}$ a.o. and

$$f_\delta(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q \tilde{\pi}_q^N(\varphi) + b_q)_+\|_2^2$$

satisfies $|f_\delta(\varphi) - f(\varphi)| \le \Delta\sqrt{2r}(2\sqrt{2r} + 2b)$, where $r = \max_q r_q,\ b = \max_q \|b_q\|_2$

.

Outline

Notation

1. E – m-dimensional real vector space,

Notation

1. E – m-dimensional real vector space,
2. $\|\cdot\|$ – Euclidean norm on E, $\|\cdot\|_*$ is its dual:

$$\|x\| = \sqrt{\langle x, x\rangle}, \quad x \in E, \quad \|g\|_* = \sqrt{\langle g, g\rangle}, \quad g \in E^*.$$

Notation

1. E – m-dimensional real vector space,
2. $\|\cdot\|$ – Euclidean norm on E, $\|\cdot\|_*$ is its dual:

$$\|x\| = \sqrt{\langle x, x\rangle}, \quad x \in E, \quad \|g\|_* = \sqrt{\langle g, g\rangle}, \quad g \in E^*.$$

3. $f \in C_L^{1,1}$ if $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$, $x \in E$.

Notation

1. E – m-dimensional real vector space,
2. $\|\cdot\|$ – Euclidean norm on E, $\|\cdot\|_*$ is its dual:

$$\|x\| = \sqrt{\langle x, x\rangle}, \quad x \in E, \quad \|g\|_* = \sqrt{\langle g, g\rangle}, \quad g \in E^*.$$

3. $f \in C_L^{1,1}$ if $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$, $x \in E$. This is equivalent to

$$|f(x) - f(y) - \langle \nabla f(y), x - y\rangle| \leq \frac{L}{2}\|x - y\|^2, \quad x, y \in E$$

Notation

1. E – m-dimensional real vector space,
2. $\|\cdot\|$ – Euclidean norm on E, $\|\cdot\|_*$ is its dual:
$$\|x\| = \sqrt{\langle x, x\rangle}, \quad x \in E, \quad \|g\|_* = \sqrt{\langle g, g\rangle}, \quad g \in E^*.$$
3. $f \in C_L^{1,1}$ if $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$, $x \in E$. This is equivalent to
$$|f(x) - f(y) - \langle \nabla f(y), x - y\rangle| \leq \frac{L}{2}\|x - y\|^2, \quad x, y \in E$$
4. $f(x)$ is smooth strongly convex function if for any $x, y \in E$
$$f(x) \geq f(y) + \langle \nabla f(y), x - y\rangle + \frac{\mu}{2}\|x - y\|^2,$$

Problem formulation

The main problem we are going to consider is

$$\min_{x \in E} f(x),$$

where

Problem formulation

The main problem we are going to consider is

$$\min_{x \in E} f(x),$$

where

1. $f(x) \in C_L^{1,1}$ and either,

Problem formulation

The main problem we are going to consider is

$$\min_{x \in E} f(x),$$

where

1. $f(x) \in C_L^{1,1}$ and either,
   1. convex

Problem formulation

The main problem we are going to consider is

$$\min_{x \in E} f(x),$$

where

1. $f(x) \in C_L^{1,1}$ and either,
    1. convex
    2. strongly convex

Problem formulation

The main problem we are going to consider is

$$\min_{x \in E} f(x),$$

where

1. $f(x) \in C_L^{1,1}$ and either,
   1. convex
   2. strongly convex
2. we use only function values measured with error

$$f_\delta(x) = f(x) + \tilde{\delta}(x),$$

$\tilde{\delta}(x)$ – oracle error satisfying $|\tilde{\delta}(x)| \le \delta \ \forall x \in E$.

Problem formulation

The main problem we are going to consider is

$$\min_{x \in E} f(x),$$

where

1. $f(x) \in C_L^{1,1}$ and either,
   1. convex
   2. strongly convex
2. we use only function values measured with error

   $$f_\delta(x) = f(x) + \tilde{\delta}(x),$$

   $\tilde{\delta}(x)$ – oracle error satisfying $|\tilde{\delta}(x)| \leq \delta \ \forall x \in E$.
3. Sometimes we additionally assume that $\tilde{\delta}(x) \equiv \tilde{\delta}$ and is a random variable which is independent on everything.

Some history of random gradient-free methods

1. Such methods are well known since 1960-s. (e.g. J.Matyas, 1965)

Some history of random gradient-free methods

1. Such methods are well known since 1960-s. (e.g. J.Matyas, 1965)
2. Fast Automatic Differentiation (e.g. Yu. G. Evtushenko, Yu.E. Nesterov et al.).

Some history of random gradient-free methods

1. Such methods are well known since 1960-s. (e.g. J.Matyas, 1965)
2. Fast Automatic Differentiation (e.g. Yu. G. Evtushenko, Yu.E. Nesterov et al.).
3. Some estimates for the rate of convergence: V.G. Karmanov (1975), B.T. Polyak (1983).

Some history of random gradient-free methods

1. Such methods are well known since 1960-s. (e.g. J.Matyas, 1965)
2. Fast Automatic Differentiation (e.g. Yu. G. Evtushenko, Yu.E. Nesterov et al.).
3. Some estimates for the rate of convergence: V.G. Karmanov (1975), B.T. Polyak (1983).
4. Some notes on the errors of the oracle in such methods by A.S. Nemirovski, D.B. Yudin (1979), B.T. Polyak (1983)

Some history of random gradient-free methods

1. Such methods are well known since 1960-s. (e.g. J.Matyas, 1965)
2. Fast Automatic Differentiation (e.g. Yu. G. Evtushenko, Yu.E. Nesterov et al.).
3. Some estimates for the rate of convergence: V.G. Karmanov (1975), B.T. Polyak (1983).
4. Some notes on the errors of the oracle in such methods by A.S. Nemirovski, D.B. Yudin (1979), B.T. Polyak (1983)
5. The current state of the art is covered by A.Conn, K.Scheinberg, and L.Vicente (2009).

Some history of random gradient-free methods

1. Such methods are well known since 1960-s. (e.g. J.Matyas, 1965)
2. Fast Automatic Differentiation (e.g. Yu. G. Evtushenko, Yu.E. Nesterov et al.).
3. Some estimates for the rate of convergence: V.G. Karmanov (1975), B.T. Polyak (1983).
4. Some notes on the errors of the oracle in such methods by A.S. Nemirovski, D.B. Yudin (1979), B.T. Polyak (1983)
5. The current state of the art is covered by A.Conn, K.Scheinberg, and L.Vicente (2009).
6. Our work based on the article by Yu. Nesterov (2011), here the fast gradient scheme also was proposed.

Some history of random gradient-free methods

1. Such methods are well known since 1960-s. (e.g. J.Matyas, 1965)
2. Fast Automatic Differentiation (e.g. Yu. G. Evtushenko, Yu.E. Nesterov et al.).
3. Some estimates for the rate of convergence: V.G. Karmanov (1975), B.T. Polyak (1983).
4. Some notes on the errors of the oracle in such methods by A.S. Nemirovski, D.B. Yudin (1979), B.T. Polyak (1983)
5. The current state of the art is covered by A.Conn, K.Scheinberg, and L.Vicente (2009).
6. Our work based on the article by Yu. Nesterov (2011), here the fast gradient scheme also was proposed.
7. Main our contribution - considering oracle error.

Smoothing the function

Consider smoothing:

$$f_\tau(x) = \mathbb{E}_b f(x + \tau b) = \frac{1}{V_B} \int_{\mathcal{B}} f(x + \tau b) db,$$

where

Smoothing the function

Consider smoothing:

$$f_\tau(x) = \mathbb{E}_b f(x+\tau b) = \frac{1}{V_B}\int_{\mathcal{B}} f(x+\tau b)db,$$

where

1. b is a uniformly distributed over unit ball $\mathcal{B} = \{x \in E : \|x\| \leq 1\}$ random vector,

Smoothing the function

Consider smoothing:

$$f_\tau(x) = \mathbb{E}_b f(x + \tau b) = \frac{1}{V_B} \int_{\mathcal{B}} f(x + \tau b) db,$$

where

1. b is a uniformly distributed over unit ball $\mathcal{B} = \{x \in E : \|x\| \leq 1\}$ random vector,
2. V_B is the volume of the unit ball $\mathcal{B}$,

Smoothing the function

Consider smoothing:

$$f_\tau(x) = \mathbb{E}_b f(x + \tau b) = \frac{1}{V_B} \int_{\mathcal{B}} f(x + \tau b) db,$$

where

1. b is a uniformly distributed over unit ball $\mathcal{B} = \{x \in E : \|x\| \leq 1\}$ random vector,
2. V_B is the volume of the unit ball $\mathcal{B}$,
3. $\tau \geq 0$ is the smoothing parameter.

Smoothing the function

Consider smoothing:

$$f_\tau(x) = \mathbb{E}_b f(x+\tau b) = \frac{1}{V_B}\int_{\mathcal{B}} f(x+\tau b)db,$$

where

1. b is a uniformly distributed over unit ball $\mathcal{B} = \{x \in E : \|x\| \le 1\}$ random vector,
2. V_B is the volume of the unit ball $\mathcal{B}$,
3. $\tau \ge 0$ is the smoothing parameter.

It turns out that

$$\nabla f_\tau(x) = \frac{m}{\tau}\mathbb{E}_s(f(x+\tau s) - f(x))s = \frac{m}{\tau V_S}\int_{\mathcal{S}}(f(x+\tau s) - f(x))sd\sigma(s),$$

where

Smoothing the function

Consider smoothing:

$$f_\tau(x) = \mathbb{E}_b f(x+\tau b) = \frac{1}{V_B}\int_{\mathcal{B}} f(x+\tau b)db,$$

where

1. b is a uniformly distributed over unit ball $\mathcal{B} = \{x \in E : \|x\| \le 1\}$ random vector,
2. V_B is the volume of the unit ball $\mathcal{B}$,
3. $\tau \ge 0$ is the smoothing parameter.

It turns out that

$$\nabla f_\tau(x) = \frac{m}{\tau}\mathbb{E}_s(f(x+\tau s) - f(x))s = \frac{m}{\tau V_S}\int_{\mathcal{S}}(f(x+\tau s) - f(x))sd\sigma(s),$$

where

1. s is a uniformly distributed over unit sphere $\mathcal{S} = \{x \in E : \|x\| = 1\}$ random vector,

Smoothing the function

Consider smoothing:

$$f_\tau(x) = \mathbb{E}_b f(x+\tau b) = \frac{1}{V_B}\int_{\mathcal{B}} f(x+\tau b)db,$$

where

1. b is a uniformly distributed over unit ball $\mathcal{B} = \{x \in E : \|x\| \le 1\}$ random vector,
2. V_B is the volume of the unit ball $\mathcal{B}$,
3. $\tau \ge 0$ is the smoothing parameter.

It turns out that

$$\nabla f_\tau(x) = \frac{m}{\tau}\mathbb{E}_s(f(x+\tau s) - f(x))s = \frac{m}{\tau V_S}\int_{\mathcal{S}}(f(x+\tau s) - f(x))s d\sigma(s),$$

where

1. s is a uniformly distributed over unit sphere $\mathcal{S} = \{x \in E : \|x\| = 1\}$ random vector,
2. V_S is the volume of the unit sphere $\mathcal{S}$,

Smoothing the function

Consider smoothing:

$$f_\tau(x) = \mathbb{E}_b f(x+\tau b) = \frac{1}{V_B}\int_{\mathcal{B}} f(x+\tau b)db,$$

where

1. b is a uniformly distributed over unit ball $\mathcal{B} = \{x \in E : \|x\| \leq 1\}$ random vector,
2. V_B is the volume of the unit ball $\mathcal{B}$,
3. $\tau \geq 0$ is the smoothing parameter.

It turns out that

$$\nabla f_\tau(x) = \frac{m}{\tau}\mathbb{E}_s(f(x+\tau s) - f(x))s = \frac{m}{\tau V_S}\int_{\mathcal{S}}(f(x+\tau s) - f(x))s d\sigma(s),$$

where

1. s is a uniformly distributed over unit sphere $\mathcal{S} = \{x \in E : \|x\| = 1\}$ random vector,
2. V_S is the volume of the unit sphere $\mathcal{S}$,
3. $d\sigma(s)$ is unnormalized spherical measure.

Some properties

1. $f_\tau(x) \geq f(x), \quad \forall x \in E.$

Some properties

1. $f_\tau(x) \geq f(x), \quad \forall x \in E.$
2. If $f(x)$ is convex, then $f_\tau(x)$ is also convex.

Some properties

1. $f_\tau(x) \geq f(x), \quad \forall x \in E.$
2. If $f(x)$ is convex, then $f_\tau(x)$ is also convex.
3. If $f \in C_L^{1,1}$ then $f_\tau \in C_L^{1,1}$.

Some properties

1. $f_\tau(x) \geq f(x), \quad \forall x \in E.$
2. If $f(x)$ is convex, then $f_\tau(x)$ is also convex.
3. If $f \in C_L^{1,1}$ then $f_\tau \in C_L^{1,1}$.
4. If $f \in C_L^{1,1}$ then $|f_\tau(x) - f(x)| \leq \frac{L\tau^2}{2}, \quad \forall x \in E.$

Random gradient-free oracle

Define random gradient-free oracle

$$g_\tau(x) = \frac{m}{\tau}(f(x+\tau s) - f(x))s,$$

where s is uniformly distributed vector over the unit sphere $\mathcal{S}$.

Random gradient-free oracle

Define random gradient-free oracle

$$g_\tau(x) = \frac{m}{\tau}(f(x+\tau s) - f(x))s,$$

where s is uniformly distributed vector over the unit sphere $\mathcal{S}$.
One can show that

$$\mathbb{E}_s g_\tau(x) = \nabla f_\tau(x).$$

Random gradient-free oracle

Define random gradient-free oracle

$$g_\tau(x) = \frac{m}{\tau}(f(x+\tau s) - f(x))s,$$

where s is uniformly distributed vector over the unit sphere $\mathcal{S}$.
One can show that

$$\mathbb{E}_s g_\tau(x) = \nabla f_\tau(x).$$

Due to error we can calculate only

$$g_{\tau,\delta}(x) = \frac{m}{\tau}(f_\delta(x+\tau s) - f_\delta(x))s.$$

Some properties

Let $f \in C_L^{1,1}$. Then

- $$\begin{aligned} \|g_{\tau,\delta}(x)\|_*^2 &\leq \\ &\leq m^2\tau^2L^2 + 4m^2(\langle \nabla f(x), s\rangle)^2 + \tfrac{8\delta^2 m^2}{\tau^2} \\ &\leq m^2\tau^2L^2 + 4m^2\|\nabla f(x)\|_*^2 + \tfrac{8\delta^2 m^2}{\tau^2} \end{aligned}$$

Some properties

Let $f \in C_L^{1,1}$. Then

- $$\begin{aligned} \|g_{\tau,\delta}(x)\|_*^2 &\leq \\ &\leq m^2\tau^2L^2 + 4m^2(\langle \nabla f(x), s\rangle)^2 + \tfrac{8\delta^2 m^2}{\tau^2} \\ &\leq m^2\tau^2L^2 + 4m^2\|\nabla f(x)\|_*^2 + \tfrac{8\delta^2 m^2}{\tau^2} \end{aligned}$$
- $\mathbb{E}_s\|g_{\tau,\delta}(x)\|_*^2 \leq m^2\tau^2L^2 + 4m\|\nabla f(x)\|_*^2 + \frac{8\delta^2 m^2}{\tau^2}$.

Some properties

Let $f \in C_L^{1,1}$. Then

- $$\begin{aligned} \|g_{\tau,\delta}(x)\|_*^2 &\le \\ &\le m^2\tau^2L^2 + 4m^2(\langle \nabla f(x), s\rangle)^2 + \frac{8\delta^2 m^2}{\tau^2} \\ &\le m^2\tau^2L^2 + 4m^2\|\nabla f(x)\|_*^2 + \frac{8\delta^2 m^2}{\tau^2} \end{aligned}$$

- $\mathbb{E}_s\|g_{\tau,\delta}(x)\|_*^2 \le m^2\tau^2L^2 + 4m\|\nabla f(x)\|_*^2 + \frac{8\delta^2 m^2}{\tau^2}$.

Main observation:
If $\nabla f(x^*) = 0$, then we can ensure that $\|g_{\tau,\delta}(x)\|$ decreases as $x \to x^*$ and we can obtain better convergence rate than is given by lower bound for general stochastic convex optimization.

Gradient-type method

We consider the problem

$$\min_{x \in E} f(x).$$

Gradient-type method

We consider the problem

$$\min_{x \in E} f(x).$$

Assume that we know point x_0 and number R such that $\|x_0 - x^*\| \le R$, where x^* is the solution of the problem.

Gradient-type method

We consider the problem

$$\min_{x \in E} f(x).$$

Assume that we know point x_0 and number R such that $\|x_0 - x^*\| \leq R$, where x^* is the solution of the problem.
Define $Q = \{x \in E : \|x - x_0\| \leq 2R\}$.

Gradient-type method

We consider the problem

$$\min_{x \in E} f(x).$$

Assume that we know point x_0 and number R such that $\|x_0 - x^*\| \leq R$, where x^* is the solution of the problem.
Define $Q = \{x \in E : \|x - x_0\| \leq 2R\}$.
Then we can solve the problem

$$\min_{x \in Q} f(x).$$

Gradient-type method

We consider the problem

$$\min_{x \in E} f(x).$$

Assume that we know point x_0 and number R such that $\|x_0 - x^*\| \le R$, where x^* is the solution of the problem.
Define $Q = \{x \in E : \|x - x_0\| \le 2R\}$.
Then we can solve the problem

$$\min_{x \in Q} f(x).$$

Gradient-type method

Input: The point x_0, number R such that $\|x_0 - x^*\| \le R$, stepsize $h > 0$.

Gradient-type method

We consider the problem

$$\min_{x \in E} f(x).$$

Assume that we know point x_0 and number R such that $\|x_0 - x^*\| \le R$, where x^* is the solution of the problem.
Define $Q = \{x \in E : \|x - x_0\| \le 2R\}$.
Then we can solve the problem

$$\min_{x \in Q} f(x).$$

Gradient-type method

Input: The point x_0, number R such that $\|x_0 - x^*\| \le R$, stepsize $h > 0$.
Output: The point x_k.

Gradient-type method

We consider the problem

$$\min_{x \in E} f(x).$$

Assume that we know point x_0 and number R such that $\|x_0 - x^*\| \le R$, where x^* is the solution of the problem.
Define $Q = \{x \in E : \|x - x_0\| \le 2R\}$.
Then we can solve the problem

$$\min_{x \in Q} f(x).$$

Gradient-type method

Input: The point x_0, number R such that $\|x_0 - x^*\| \le R$, stepsize $h > 0$.
Output: The point x_k.

1. Generate s_k and corresponding $g_{\tau,\delta}(x_k)$.

Gradient-type method

We consider the problem

$$\min_{x \in E} f(x).$$

Assume that we know point x_0 and number R such that $\|x_0 - x^*\| \leq R$, where x^* is the solution of the problem.
Define $Q = \{x \in E : \|x - x_0\| \leq 2R\}$.
Then we can solve the problem

$$\min_{x \in Q} f(x).$$

Gradient-type method

Input: The point x_0, number R such that $\|x_0 - x^*\| \leq R$, stepsize $h > 0$.
Output: The point x_k.

1. Generate s_k and corresponding $g_{\tau,\delta}(x_k)$.
2. Calculate $x_{k+1} = \pi_Q(x_k - h g_{\tau,\delta}(x_k))$.

Convergence rate

Denote $\mathcal{U}_k = (s_0, \dots, s_k)$ the history of realizations of the vectors s_k, generated on each iteration of the method, $\phi_0 = f(x_0)$, and $\phi_k = \mathbb{E}_{\mathcal{U}_{k-1}}(f(x_{k-1}))$, $k \geq 1$.

Convergence rate

Denote $\mathcal{U}_k = (s_0, \ldots, s_k)$ the history of realizations of the vectors s_k, generated on each iteration of the method, $\phi_0 = f(x_0)$, and $\phi_k = \mathbb{E}_{\mathcal{U}_{k-1}}(f(x_{k-1}))$, $k \geq 1$.
Let $f \in C_L^{1,1}$ and the sequence x_k be generated by the Algorithm above with $h = \frac{1}{8mL}$. Then for any $N \geq 0$, we have

$$\frac{1}{N+1}\sum_{i=0}^{N}(\phi_i - f^*) \leq \frac{8mLR^2}{N+1} + \frac{\tau^2 L(m+8)}{8} + \frac{8\delta m R}{\tau} + \frac{\delta^2 m}{L\tau^2}.$$

Convergence rate

Denote $\mathcal{U}_k = (s_0, \dots, s_k)$ the history of realizations of the vectors s_k, generated on each iteration of the method, $\phi_0 = f(x_0)$, and $\phi_k = \mathbb{E}_{\mathcal{U}_{k-1}}(f(x_{k-1}))$, $k \geq 1$.
Let $f \in C_L^{1,1}$ and the sequence x_k be generated by the Algorithm above with $h = \frac{1}{8mL}$. Then for any $N \geq 0$, we have

$$\frac{1}{N+1}\sum_{i=0}^{N}(\phi_i - f^*) \leq \frac{8mLR^2}{N+1} + \frac{\tau^2 L(m+8)}{8} + \frac{8\delta m R}{\tau} + \frac{\delta^2 m}{L\tau^2}.$$

If additionally f is strongly convex, then

$$\phi_N - f^* \leq \frac{1}{2}L\left(\delta_\tau + \left(1 - \frac{\mu}{16mL}\right)^N (R^2 - \delta_\tau)\right),$$

where $\delta_\tau = \frac{\tau^2 L(m+8)}{4\mu} + \frac{16m\delta R}{\mu\tau} + \frac{2m\delta^2}{\mu\tau^2 L}$.

Discussion

To achieve desired accuracy ε we need to choose on average.

Discussion

To achieve desired accuracy ε we need to choose on average.
In convex case with $|\tilde{\delta}(x)| \leq \delta$

$$N = O\left(\frac{mLR^2}{\varepsilon}\right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}}\right), \quad \delta = O\left(\min\left\{\left(\frac{\varepsilon}{m}\right)^{\frac{3}{2}} \cdot \frac{1}{\sqrt{LR^2}}, \frac{\varepsilon}{m}\right\}\right).$$

Discussion

To achieve desired accuracy ε we need to choose on average.
In convex case with $|\tilde{\delta}(x)| \le \delta$

$$N = O\left(\frac{mLR^2}{\varepsilon}\right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}}\right), \quad \delta = O\left(\min\left\{\left(\frac{\varepsilon}{m}\right)^{\frac{3}{2}} \cdot \frac{1}{\sqrt{LR^2}}, \frac{\varepsilon}{m}\right\}\right).$$

In convex case with $\tilde{\delta}(x)$ random and independent

$$N = O\left(\frac{mLR^2}{\varepsilon}\right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}}\right), \quad \delta = O\left(\frac{\varepsilon}{m}\right).$$

Discussion

To achieve desired accuracy ε we need to choose on average.
In convex case with $|\tilde{\delta}(x)| \le \delta$

$$N = O\left(\frac{mLR^2}{\varepsilon}\right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}}\right), \quad \delta = O\left(\min\left\{\left(\frac{\varepsilon}{m}\right)^{\frac{3}{2}} \cdot \frac{1}{\sqrt{LR^2}}, \frac{\varepsilon}{m}\right\}\right).$$

In convex case with $\tilde{\delta}(x)$ random and independent

$$N = O\left(\frac{mLR^2}{\varepsilon}\right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}}\right), \quad \delta = O\left(\frac{\varepsilon}{m}\right).$$

In strongly convex case with $|\tilde{\delta}(x)| \le \delta$

$$N = O\left(\frac{mL}{\mu}\ln\frac{LR^2}{\varepsilon}\right), \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}\cdot\frac{\mu}{L}}\right), \delta = O\left(\min\left\{\left(\frac{\varepsilon\mu}{mL}\right)^{\frac{3}{2}} \cdot \frac{1}{\sqrt{LR^2}}, \frac{\varepsilon\mu}{mL}\right\}\right).$$

Discussion

To achieve desired accuracy ε we need to choose on average.
In convex case with $|\tilde{\delta}(x)| \leq \delta$

$$N = O\left(\frac{mLR^2}{\varepsilon}\right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}}\right), \quad \delta = O\left(\min\left\{\left(\frac{\varepsilon}{m}\right)^{\frac{3}{2}} \cdot \frac{1}{\sqrt{LR^2}}, \frac{\varepsilon}{m}\right\}\right).$$

In convex case with $\tilde{\delta}(x)$ random and independent

$$N = O\left(\frac{mLR^2}{\varepsilon}\right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}}\right), \quad \delta = O\left(\frac{\varepsilon}{m}\right).$$

In strongly convex case with $|\tilde{\delta}(x)| \leq \delta$

$$N = O\left(\frac{mL}{\mu}\ln\frac{LR^2}{\varepsilon}\right), \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}\cdot\frac{\mu}{L}}\right), \delta = O\left(\min\left\{\left(\frac{\varepsilon\mu}{mL}\right)^{\frac{3}{2}} \cdot \frac{1}{\sqrt{LR^2}}, \frac{\varepsilon\mu}{mL}\right\}\right).$$

In strongly convex case with $\tilde{\delta}(x)$ random and independent

$$N = O\left(\frac{mL}{\mu}\ln\frac{LR^2}{\varepsilon}\right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{Lm}\cdot\frac{\mu}{L}}\right), \quad \delta = O\left(\frac{\varepsilon\mu}{mL}\right).$$

Fast gradient-type method

We consider the problem

$$\min_{x \in E} f(x),$$

where $f \in C_L^{1,1}$ and is a strongly convex function with parameter $\mu \geq 0$.

Fast gradient-type method

We consider the problem

$$\min_{x \in E} f(x),$$

where $f \in C_L^{1,1}$ and is a strongly convex function with parameter $\mu \geq 0$. We difine $\theta = \frac{1}{64m^2L}$ and $h = \frac{1}{8mL}$ and consider the following method.

Fast gradient-type method

We consider the problem

$$\min_{x \in E} f(x),$$

where $f \in C_L^{1,1}$ and is a strongly convex function with parameter $\mu \geq 0$. We difine $\theta = \frac{1}{64m^2L}$ and $h = \frac{1}{8mL}$ and consider the following method.

Fast Gradient Method Modified

Input: The point x_0, number $\gamma_0 \geq \mu$.
Output: The point x_k.
Set $v_0 = x_0$.

1. Compute $\alpha_k > 0$ satisfying $\frac{\alpha_k^2}{\theta} = (1 - \alpha_k)\gamma_k + \alpha_k \mu \equiv \gamma_{k+1}$.
2. Set $\lambda_k = \frac{\alpha_k}{\gamma_{k+1}}\mu$, $\beta_k = \frac{\alpha_k \gamma_k}{\gamma_k + \alpha_k \mu}$, and $y_k = (1 - \beta_k)x_k + \beta_k v_k$.
3. Generate s_k and corresponding $g_{\tau,\delta}(y_k)$.
4. Calculate $x_{k+1} = y_k - h g_{\tau,\delta}(y_k)$,
   $v_{k+1} = (1 - \lambda_k)v_k + \lambda_k y_k - \frac{\theta}{\alpha_k} g_{\tau,\delta}(y_k)$.

Convergence rate

Define $\kappa = \frac{\mu}{L}$. In the case when $\tilde{\delta}(x)$ is random and independent we have for all $k \geq 0$

$$\mathbb{E}_{\mathcal{U}_{k-1}} f(x_k) - f^* \leq \psi_k \left(f(x_0) - f^* + \frac{\gamma_0}{2} \|x_0 - x^*\|^2 \right) +$$
$$+ C_k \left(\frac{5\tau^2 L}{64} + \frac{\delta^2}{4\tau^2 L} \right) + \tau^2 L,$$

where $\psi_k \leq \min \left\{ \left(1 - \frac{\sqrt{\kappa}}{8m}\right)^k, \left(1 + \frac{k}{16m}\sqrt{\frac{\gamma_0}{L}}\right)^{-2} \right\}$, $C_k \leq \min \left\{ k, \frac{8m}{\sqrt{\kappa}} \right\}$.

Convergence rate

Define $\kappa = \frac{\mu}{L}$. In the case when $\tilde{\delta}(x)$ is random and independent we have for all $k \geq 0$

$$\mathbb{E}_{\mathcal{U}_{k-1}} f(x_k) - f^* \leq \psi_k \left(f(x_0) - f^* + \frac{\gamma_0}{2}\|x_0 - x^*\|^2 \right) +$$
$$+ C_k \left(\frac{5\tau^2 L}{64} + \frac{\delta^2}{4\tau^2 L} \right) + \tau^2 L,$$

where $\psi_k \leq \min\left\{ \left(1 - \frac{\sqrt{\kappa}}{8m}\right)^k, \left(1 + \frac{k}{16m}\sqrt{\frac{\gamma_0}{L}}\right)^{-2} \right\}$, $C_k \leq \min\left\{k, \frac{8m}{\sqrt{\kappa}}\right\}$.
Then for $\mu = 0$ to obtain the accuracy ε we need to choose on average

$$N = O\left(m\sqrt{\frac{LR^2}{\varepsilon}} \right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{mL}\sqrt{\frac{\varepsilon}{LR^2}}} \right), \quad \delta = O\left(\frac{\varepsilon}{m}\sqrt{\frac{\varepsilon}{LR^2}} \right)$$

Convergence rate

Define $\kappa = \frac{\mu}{L}$. In the case when $\tilde{\delta}(x)$ is random and independent we have for all $k \geq 0$

$$\mathbb{E}_{\mathcal{U}_{k-1}} f(x_k) - f^* \leq \psi_k \left(f(x_0) - f^* + \frac{\gamma_0}{2} \|x_0 - x^*\|^2 \right) +$$
$$+ C_k \left(\frac{5\tau^2 L}{64} + \frac{\delta^2}{4\tau^2 L} \right) + \tau^2 L,$$

where $\psi_k \leq \min \left\{ \left(1 - \frac{\sqrt{\kappa}}{8m}\right)^k, \left(1 + \frac{k}{16m}\sqrt{\frac{\gamma_0}{L}}\right)^{-2} \right\}$, $C_k \leq \min \left\{ k, \frac{8m}{\sqrt{\kappa}} \right\}$.
Then for $\mu = 0$ to obtain the accuracy ε we need to choose on average

$$N = O\left(m\sqrt{\frac{LR^2}{\varepsilon}} \right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{mL}\sqrt{\frac{\varepsilon}{LR^2}}} \right), \quad \delta = O\left(\frac{\varepsilon}{m}\sqrt{\frac{\varepsilon}{LR^2}} \right)$$

For $\mu > 0$ to obtain the accuracy ε we need to choose on average

$$N = O\left(m\sqrt{\frac{L}{\mu}} \ln\left(\frac{\mu R^2}{\varepsilon} \right) \right), \quad \tau = O\left(\sqrt{\frac{\varepsilon}{mL}\sqrt{\frac{\mu}{L}}} \right), \quad \delta = O\left(\frac{\varepsilon}{m}\sqrt{\frac{\mu}{L}} \right)$$

Discussion

1. We have considered two random gradient-free methods with error in the oracle value: gradient-type scheme and fast-gradient-type scheme.

Discussion

1. We have considered two random gradient-free methods with error in the oracle value: gradient-type scheme and fast-gradient-type scheme.
2. We have obtained their mean rate of convergence and bounds on the oracle error ($\mu = 0$):

$$\text{PGM}: \quad N = O\left(\frac{mLR^2}{\varepsilon}\right), \quad \delta = O\left(\frac{\varepsilon}{m}\right).$$

$$\text{FGM}: \quad N = O\left(m\sqrt{\frac{LR^2}{\varepsilon}}\right), \quad \delta = O\left(\frac{\varepsilon}{m}\sqrt{\frac{\varepsilon}{LR^2}}\right).$$

Outline

Recall the problem

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q}\|(A_q\pi_q^*(\varphi) + b_q)_+\|_2^2 \to \min$$

$$\pi_q^*(\varphi) = \alpha\left[I - (1-\alpha)P_q^T(\varphi)\right]^{-1}\pi_q^0(\varphi) \Leftrightarrow \|\pi - \pi_q^*(\varphi)\|_1 \to \min\,.$$

Recall the problem

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q \pi_q^*(\varphi) + b_q)_+\|_2^2 \to \min$$

$$\pi_q^*(\varphi) = \alpha \left[I - (1-\alpha) P_q^T(\varphi)\right]^{-1} \pi_q^0(\varphi) \Leftrightarrow \|\pi - \pi_q^*(\varphi)\|_1 \to \min\,.$$

To obtain vector $\tilde{\pi}_q^N(\varphi)$ s.t. $\|\tilde{\pi}_q^N(\varphi) - \pi_q^*(\varphi)\|_1 \le \Delta$ we need $\frac{s_q(p_q+n_q)}{\alpha} \ln \frac{2}{\Delta}$ a.o. and

Recall the problem

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q\pi_q^*(\varphi) + b_q)_+\|_2^2 \to \min$$

$$\pi_q^*(\varphi) = \alpha\left[I - (1-\alpha)P_q^T(\varphi)\right]^{-1}\pi_q^0(\varphi) \Leftrightarrow \|\pi - \pi_q^*(\varphi)\|_1 \to \min\,.$$

To obtain vector $\tilde{\pi}_q^N(\varphi)$ s.t. $\|\tilde{\pi}_q^N(\varphi) - \pi_q^*(\varphi)\|_1 \le \Delta$ we need $\frac{s_q(p_q+n_q)}{\alpha}\ln\frac{2}{\Delta}$ a.o. and

$$f_\delta(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q\tilde{\pi}_q^N(\varphi) + b_q)_+\|_2^2$$

satisfies $|f_\delta(\varphi) - f(\varphi)| \le \Delta\sqrt{2r}(2\sqrt{2r} + 2b)$.

Recall the problem

$$f(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q \pi_q^*(\varphi) + b_q)_+\|_2^2 \to \min$$

$$\pi_q^*(\varphi) = \alpha \left[I - (1-\alpha) P_q^T(\varphi) \right]^{-1} \pi_q^0(\varphi) \Leftrightarrow \|\pi - \pi_q^*(\varphi)\|_1 \to \min .$$

To obtain vector $\tilde{\pi}_q^N(\varphi)$ s.t. $\|\tilde{\pi}_q^N(\varphi) - \pi_q^*(\varphi)\|_1 \le \Delta$ we need $\frac{s_q(p_q+n_q)}{\alpha} \ln \frac{2}{\Delta}$ a.o. and

$$f_\delta(\varphi) = \frac{1}{Q}\sum_{q=1}^{Q} \|(A_q \tilde{\pi}_q^N(\varphi) + b_q)_+\|_2^2$$

satisfies $|f_\delta(\varphi) - f(\varphi)| \le \Delta\sqrt{2r}(2\sqrt{2r} + 2b)$.
Idea: use [Nemirovski, Nesterov, 2012] to calculate $f_\delta(\varphi)$, then use the gradient-type method to make the step using $g_{\mu,\delta}(\varphi)$.

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \le R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \leq R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_{\varphi}\{f(\varphi) : \varphi \in \{\varphi_0, \ldots, \varphi_N\}\}$.

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \le R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_\varphi \{f(\varphi) : \varphi \in \{\varphi_0, \ldots, \varphi_N\}\}$.

1. Define $G = \{\varphi \in \mathbb{R}^m : \|\varphi - \varphi_0\|_2 \le 2R\}$, $N = 32m\frac{LR^2}{\varepsilon}$, $\delta = \frac{\varepsilon^{\frac{3}{2}}\sqrt{2}}{32mR\sqrt{L(m+8)}}$, $\tau = \sqrt{\frac{2\varepsilon}{L(m+8)}}$. Set $k = 0$

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \leq R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_{\varphi}\{f(\varphi) : \varphi \in \{\varphi_0, \dots, \varphi_N\}\}$.

1. Define $G = \{\varphi \in \mathbb{R}^m : \|\varphi - \varphi_0\|_2 \leq 2R\}$, $N = 32m\frac{LR^2}{\varepsilon}$, $\delta = \frac{\varepsilon^{\frac{3}{2}}\sqrt{2}}{32mR\sqrt{L(m+8)}}$, $\tau = \sqrt{\frac{2\varepsilon}{L(m+8)}}$. Set $k = 0$
2. for $k = 0, ..., N$.

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \le R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_{\varphi}\{f(\varphi) : \varphi \in \{\varphi_0, \dots, \varphi_N\}\}$.

1. Define $G = \{\varphi \in \mathbb{R}^m : \|\varphi - \varphi_0\|_2 \le 2R\}$, $N = 32m\frac{LR^2}{\varepsilon}$, $\delta = \frac{\varepsilon^{\frac{3}{2}}\sqrt{2}}{32mR\sqrt{L(m+8)}}$, $\tau = \sqrt{\frac{2\varepsilon}{L(m+8)}}$. Set $k = 0$
2. for $k = 0, ..., N$.
3. Generate random vector s_k uniformly distributed over a unit Euclidean sphere $\mathcal{S}$ in R^m;

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \leq R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_\varphi \{f(\varphi) : \varphi \in \{\varphi_0, \ldots, \varphi_N\}\}$.

1. Define $G = \{\varphi \in \mathbb{R}^m : \|\varphi - \varphi_0\|_2 \leq 2R\}$, $N = 32m\frac{LR^2}{\varepsilon}$, $\delta = \frac{\varepsilon^{\frac{3}{2}}\sqrt{2}}{32mR\sqrt{L(m+8)}}$, $\tau = \sqrt{\frac{2\varepsilon}{L(m+8)}}$. Set $k = 0$
2. for $k = 0, ..., N$.
3. Generate random vector s_k uniformly distributed over a unit Euclidean sphere $\mathcal{S}$ in R^m;
4. Set $\hat{N} = \frac{1}{\alpha} \ln \frac{2\sqrt{2r}(2\sqrt{2r}+2b)}{\delta}$;

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \le R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_{\varphi}\{f(\varphi) : \varphi \in \{\varphi_0, \dots, \varphi_N\}\}$.

1. Define $G = \{\varphi \in \mathbb{R}^m : \|\varphi - \varphi_0\|_2 \le 2R\}$, $N = 32m\frac{LR^2}{\varepsilon}$, $\delta = \frac{\varepsilon^{\frac{3}{2}}\sqrt{2}}{32mR\sqrt{L(m+8)}}$, $\tau = \sqrt{\frac{2\varepsilon}{L(m+8)}}$. Set $k = 0$
2. for $k = 0, \dots, N$.
3. Generate random vector s_k uniformly distributed over a unit Euclidean sphere $\mathcal{S}$ in R^m;
4. Set $\hat{N} = \frac{1}{\alpha} \ln \frac{2\sqrt{2r}(2\sqrt{2r}+2b)}{\delta}$;
5. For every q calculate $\tilde{\pi}_q^{\hat{N}}(\varphi_k)$, $\tilde{\pi}_q^{\hat{N}}(\varphi_k + \tau s_k)$ defined above;

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \leq R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_{\varphi}\{f(\varphi) : \varphi \in \{\varphi_0, \ldots, \varphi_N\}\}$.

1. Define $G = \{\varphi \in \mathbb{R}^m : \|\varphi - \varphi_0\|_2 \leq 2R\}$, $N = 32m\frac{LR^2}{\varepsilon}$, $\delta = \frac{\varepsilon^{\frac{3}{2}}\sqrt{2}}{32mR\sqrt{L(m+8)}}$, $\tau = \sqrt{\frac{2\varepsilon}{L(m+8)}}$. Set $k = 0$
2. for $k = 0, ..., N$.
3. Generate random vector s_k uniformly distributed over a unit Euclidean sphere $\mathcal{S}$ in R^m;
4. Set $\hat{N} = \frac{1}{\alpha}\ln\frac{2\sqrt{2r}(2\sqrt{2r}+2b)}{\delta}$;
5. For every q calculate $\tilde{\pi}_q^{\hat{N}}(\varphi_k)$, $\tilde{\pi}_q^{\hat{N}}(\varphi_k + \tau s_k)$ defined above;
6. Calculate $g_{\tau,\delta}(x_k) = \frac{m}{\tau}(f_\delta(\varphi_k + \tau s_k) - f_\delta(\varphi_k))s_k$;

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \leq R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_{\varphi}\{f(\varphi) : \varphi \in \{\varphi_0, \ldots, \varphi_N\}\}$.

1. Define $G = \{\varphi \in \mathbb{R}^m : \|\varphi - \varphi_0\|_2 \leq 2R\}$, $N = 32m\frac{LR^2}{\varepsilon}$, $\delta = \frac{\varepsilon^{\frac{3}{2}}\sqrt{2}}{32mR\sqrt{L(m+8)}}$, $\tau = \sqrt{\frac{2\varepsilon}{L(m+8)}}$. Set $k = 0$
2. for $k = 0, ..., N$.
3. Generate random vector s_k uniformly distributed over a unit Euclidean sphere $\mathcal{S}$ in R^m;
4. Set $\hat{N} = \frac{1}{\alpha} \ln \frac{2\sqrt{2r}(2\sqrt{2r}+2b)}{\delta}$;
5. For every q calculate $\tilde{\pi}_q^{\hat{N}}(\varphi_k)$, $\tilde{\pi}_q^{\hat{N}}(\varphi_k + \tau s_k)$ defined above;
6. Calculate $g_{\tau,\delta}(x_k) = \frac{m}{\tau}(f_\delta(\varphi_k + \tau s_k) - f_\delta(\varphi_k))s_k$;
7. Calculate $\varphi_{k+1} = \Pi_G\left(\varphi_k - \frac{1}{8mL} g_{\tau,\delta}(\varphi_k)\right)$;

The method

Input: The point φ_0, L – Lipschitz constant for the function $f(\varphi)$, number R such that $\|\varphi_0 - \varphi^*\|_2 \leq R$, accuracy $\varepsilon > 0$, numbers r, b defined above.

Output: The point $\hat{\varphi}_N = \arg\min_{\varphi}\{f(\varphi) : \varphi \in \{\varphi_0, \ldots, \varphi_N\}\}$.

1. Define $G = \{\varphi \in \mathbb{R}^m : \|\varphi - \varphi_0\|_2 \leq 2R\}$, $N = 32m\frac{LR^2}{\varepsilon}$, $\delta = \frac{\varepsilon^{\frac{3}{2}}\sqrt{2}}{32mR\sqrt{L(m+8)}}$, $\tau = \sqrt{\frac{2\varepsilon}{L(m+8)}}$. Set $k = 0$
2. for $k = 0, ..., N$.
3. Generate random vector s_k uniformly distributed over a unit Euclidean sphere $\mathcal{S}$ in R^m;
4. Set $\hat{N} = \frac{1}{\alpha} \ln \frac{2\sqrt{2r}(2\sqrt{2r}+2b)}{\delta}$;
5. For every q calculate $\tilde{\pi}_q^{\hat{N}}(\varphi_k)$, $\tilde{\pi}_q^{\hat{N}}(\varphi_k + \tau s_k)$ defined above;
6. Calculate $g_{\tau,\delta}(x_k) = \frac{m}{\tau}(f_\delta(\varphi_k + \tau s_k) - f_\delta(\varphi_k))s_k$;
7. Calculate $\varphi_{k+1} = \Pi_G\left(\varphi_k - \frac{1}{8mL} g_{\tau,\delta}(\varphi_k)\right)$;
8. Set $k = k + 1$;

Complexity

Each iteration of the Algorithm needs approximately $\frac{2Qs(p+n)}{\alpha} \ln \frac{2\sqrt{2r}(2\sqrt{2r}+2b)}{\delta}$ a.o., where $s = \max_q s_q$, $p = \max_q p_q$, $n = \max_q n_q$.

Complexity

Each iteration of the Algorithm needs approximately $\frac{2Qs(p+n)}{\alpha} \ln \frac{2\sqrt{2r}(2\sqrt{2r}+2b)}{\delta}$ a.o., where $s = \max_q s_q$, $p = \max_q p_q$, $n = \max_q n_q$.

Total number of a.o. for the accuracy ε is given by

$$O\left(m(n+p)sQ\frac{LR^2}{\alpha\varepsilon} \ln\left((r+b\sqrt{r})\frac{m^{3/2}R\sqrt{L}}{\varepsilon^{3/2}}\right)\right).$$

Complexity

Each iteration of the Algorithm needs approximately $\frac{2Qs(p+n)}{\alpha}\ln\frac{2\sqrt{2r}(2\sqrt{2r}+2b)}{\delta}$ a.o., where $s=\max_q s_q$, $p=\max_q p_q$, $n=\max_q n_q$.
Total number of a.o. for the accuracy ε is given by

$$O\left(m(n+p)sQ\frac{LR^2}{\alpha\varepsilon}\ln\left((r+b\sqrt{r})\frac{m^{3/2}R\sqrt{L}}{\varepsilon^{3/2}}\right)\right).$$

Fast-gradient-type scheme would give

$$O\left(mnsQ\sqrt{\frac{LR^2}{\alpha^2\varepsilon}}\ln\left((r+b\sqrt{r})\frac{mRL}{\varepsilon}\right)\right).$$

Discussion

Problems

1. $f(\varphi)$ is generally non-convex and we have only local convergence. We have some ideas by Yu. Nesterov on how to reformulate the problem and have convex optimization problem.

Discussion

Problems

1. $f(\varphi)$ is generally non-convex and we have only local convergence. We have some ideas by Yu. Nesterov on how to reformulate the problem and have convex optimization problem.
2. For now we proved convergence rate fast-gradient-type gradient-free method only for independent random error. We are trying to obtain more general result.

Discussion

Problems

1. $f(\varphi)$ is generally non-convex and we have only local convergence. We have some ideas by Yu. Nesterov on how to reformulate the problem and have convex optimization problem.
2. For now we proved convergence rate fast-gradient-type gradient-free method only for independent random error. We are trying to obtain more general result.
3. Unknown or large L. We are trying to use idea of double smoothing from [Duchi, Jordan, Wainwright, Wibisono, 2014].

Thank you!