Optimization Theory

Lecture 16

Fudan University

luoluo@fudan.edu.cn

Outline

Outline

Zeroth-Order Optimization

We study the scheme

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \mathbf{g}_\delta(\mathbf{x}_t; \mathbf{u}_t),$$

where

$$\mathbf{g}_\delta(\mathbf{x}; \mathbf{u}) = \frac{f(\mathbf{x} + \delta\mathbf{u}) - f(\mathbf{x})}{\delta} \cdot \mathbf{u}.$$

1. If $f(\cdot)$ is G-Lipschitz continuous, then

$$\mathbb{E}\,\|\mathbf{g}_\delta(\mathbf{x}; \mathbf{u})\|_2^2 \leq G^2(d+4)^2.$$

2. If $f(\cdot)$ is L-smooth, then

$$\mathbb{E}\,\|\mathbf{g}_\delta(\mathbf{x}; \mathbf{u})\|_2^2 \leq \frac{L^2\delta^2(d+6)^3}{2} + 2(d+4)\,\|\nabla f(\mathbf{x})\|_2^2.$$

Theorem (Nonsmooth Convex)

Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is convex and G-Lipschitz. The iteration

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \mathbf{g}_\delta(\mathbf{x}_t; \mathbf{u}_t)$$

holds that

$$\begin{aligned}
&\frac{1}{\sum_{t=0}^{T-1}\eta_t}\sum_{t=0}^{T-1}\eta_t\mathbb{E}[(f(\mathbf{x}_t) - f(\mathbf{x}^*)] \\
\leq& \delta G\sqrt{d} + \frac{1}{2\sum_{t=0}^{T-1}\eta_t}\left(\|\mathbf{x}_0 - \mathbf{x}^*\|_2^2 + G^2(d+4)^2\sum_{t=0}^{T-1}\eta_t^2\right).
\end{aligned}$$

Zeroth-Order Optimization

Theorem (Smooth Convex)

Suppose $f:\mathbb{R}^d\to\mathbb{R}$ is convex and L-smooth. The iteration

$$\mathbf{x}_{t+1}=\mathbf{x}_t-\eta\mathbf{g}_\delta(\mathbf{x}_t;\mathbf{u}_t)$$

with $\eta=1/(4L(d+4))$ holds that

$$\frac{1}{T}\sum_{t=0}^{T-1}(f(\mathbf{x}_t)-f(\mathbf{x}^*))\le\frac{4L(d+4)\|\mathbf{x}_0-\mathbf{x}^*\|_2^2}{T}+\frac{9L\delta^2(d+4)^2}{25}.$$

Additionally suppose $f(\cdot)$ is μ-strongly convex, then

$$\mathbb{E}\left[\|\mathbf{x}_T-\mathbf{x}^*\|_2^2-\Delta\right]\le\left(1-\frac{\mu}{8L(d+4)}\right)^T\left(\|\mathbf{x}_0-\mathbf{x}^*\|_2^2-\Delta\right),$$

where $\Delta=\dfrac{18\delta^2L(d+4)^2}{25\mu}$.

Zeroth-Order Optimization

The differentiability of $\nabla f_\delta(\cdot)$ and the fact

$$\mathbb{E}[\mathbf{g}_\delta(\mathbf{x};\mathbf{u})] = \nabla f_\delta(\mathbf{x})$$

means the mini-batch version scheme

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \cdot \frac{1}{b}\sum_{i=1}^{b} \mathbf{g}_\delta(\mathbf{x}_t;\mathbf{u}_{t,i})$$

can reduce the iteration numbers.

Zeroth-Order Optimization

The following lemma means we can also apply variance reduction on Gaussian smoothing.

Lemma

For any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ *and some* $\delta > 0$, *it holds that:*

1. *If* $f(\cdot)$ *is* G*-Lipschitz continuous, then*

$$\mathbb{E}\left\|\mathbf{g}_\delta(\mathbf{x}; \mathbf{u}) - \mathbf{g}_\delta(\mathbf{y}; \mathbf{u})\right\|_2^2 \leq \frac{2G^2 d \left\|\mathbf{x} - \mathbf{y}\right\|_2^2}{\delta}.$$

2. *If* $f(\cdot)$ *is* L*-smooth continuous, then*

$$\mathbb{E}\left\|\mathbf{g}_\delta(\mathbf{x}; \mathbf{u}) - \mathbf{g}_\delta(\mathbf{y}; \mathbf{u})\right\|_2^2 \leq \frac{3L^2\delta^2(d+6)^3}{2} + 3L^2(d+4)\left\|\mathbf{x} - \mathbf{y}\right\|_2^2.$$

Outline

Distributed Optimization

We consider the distributed optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) \triangleq \frac{1}{m}\sum_{i=1}^{m} f_i(\mathbf{x}),$$

where $f_i(\mathbf{x}) \triangleq \mathbb{E}_{\xi\sim\mathcal{D}_i} F_i(\mathbf{x};\xi)$ is the smooth local function on the i-th client.

In synchronized (centralized) training, we perform the following steps at each time step:

1. Compute on all client nodes simultaneously.
2. Communicate from client nodes to server node.
3. Compute on server node.
4. Communicate from server node to client nodes.

Synchronous Stochastic Gradient Descent

Algorithm 1 Synchronous SGD

Input: $\mathbf{x}_0$, b and $\{\eta_t\}$
for $t = 0, 1, \dots$ **do**
  for $i = 1, \dots, m$ **do in parallel**
    client:
      receive $\mathbf{x}_t$ from server
      sample $\mathcal{S}_t = \{\xi_{t,1}, \dots, \xi_{t,b}\}$, where $\xi_{t,j} \overset{\text{i.i.d}}{\sim} \mathcal{D}_i$
      compute $\mathbf{x}_{t+1,i} = \mathbf{x}_t - \eta_t \nabla F_i(\mathbf{x}_t; \mathcal{S}_t)$
      send $\mathbf{x}_{t+1,i}$ to server
  end for
  server:
    receive $\mathbf{x}_{t,1}, \dots, \mathbf{x}_{t,m}$ from all of clients
    compute $\mathbf{x}_{t+1} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_{t+1,i}$
    send $\mathbf{x}_{t+1}$ to all of clients
end for

Asynchronous Distributed Optimization

We also consider asynchronous training:

1. Server maintain a time step.
2. Clients receive the parameter from server at any time.
3. Clients can perform computation that requires various duration to finish.
4. Clients can communicate computational results to server
5. Server receives delayed gradient calculations from the clients.

Asynchronous Stochastic Gradient Descent

Algorithm 2 Asynchronous SGD

Input: $\mathbf{x}_0$, b and $\{\eta_t\}$
each client:
  Loop
    receive $\mathbf{x}_t$ from server
    sample $\mathcal{S}_t = \{\xi_{t,1}, \dots, \xi_{t,b}\}$, where $\xi_{t,j} \overset{\text{i.i.d}}{\sim} \mathcal{D}_i$
    compute $\mathbf{g}_i = \nabla F_i(\mathbf{x}_t; \mathcal{S}_t)$
    send $\mathbf{g}_i$ to server
server:
  for $t = 0, 1, \dots$ **do**
    receive $\mathbf{g}_j$ from some node in a queue
    compute $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \mathbf{g}_j$
    send $\mathbf{x}_{t+1}$ to all of clients
  end for