Gradient-based optimization methods for neural network training

Ph.D. in Industrial Engineering
Università di Firenze, A.A. 2020/2021

Simone Rebegoldi
Dipartimento di Ingegneria Industriale
Università di Firenze

Expected and empirical risk minimization

Optimization problems in Machine Learning consist in minimizing:

1. either an *Expected Risk*

$$f(x) \overset{\text{def}}{=} \int_{\mathbb{R}^d \times \mathbb{R}^p} \ell(h(a;x),b) dP(a,b) = \mathbb{E}[\ell(h(a;x),b)], \quad f : \mathbb{R}^n \to \mathbb{R}$$

$P(a,b)$ representing the probability distribution of inputs $a \in \mathbb{R}^d$ and outputs $b \in \mathbb{R}^p$

2. or an *Empirical Risk*

$$f(x) \overset{\text{def}}{=} \frac{1}{N} \sum_{i=1}^{N} \ell(h(a_i;x),b_i), \quad f : \mathbb{R}^n \to \mathbb{R}$$

$\{(a_i,b_i)\}_{i=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}^p$ being a set of independently drawn input-output samples.

Gradient Descent (GD) Method

$$\min_{x \in \mathbb{R}^n} f(x), \quad f : \mathbb{R}^n \to \mathbb{R} \text{ differentiable.}$$

- Start with some initial guess x_0.
- Generate a new guess x_1 by moving in the negative gradient direction:

$$x_1 = x_0 - \alpha_0 \nabla f(x_0).$$

- Repeat to successively refine the guess:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad k = 0, 1, 2, \ldots$$

where $\alpha_k > 0$ is called step-size or steplength or *learning rate* in the machine learning community.

Gradient Descent (GD) Method

$$\min_{x \in \mathbb{R}^n} f(x), \quad f : \mathbb{R}^n \to \mathbb{R} \text{ differentiable.}$$

- Start with some initial guess x_0.
- Generate a new guess x_1 by moving in the negative gradient direction:

$$x_1 = x_0 - \alpha_0 \nabla f(x_0).$$

- Repeat to successively refine the guess:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k), \quad k = 0, 1, 2, \ldots$$

where $\alpha_k > 0$ is called step-size or steplength or *learning rate* in the machine learning community.

$-\nabla f(x_k)$ is a **descent direction** for f at x_k.

==========

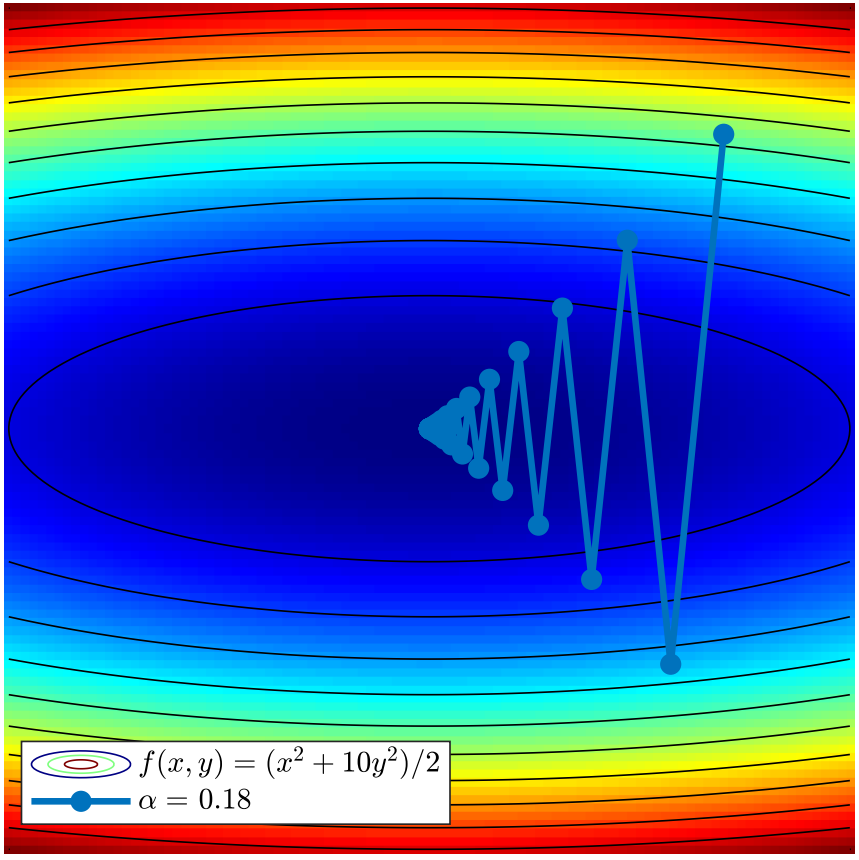
$p \in \mathbb{R}^n$ is a **descent direction** if

$$\nabla f_p(x_k) = \nabla f(x_k)^{\mathrm{T}} p < 0.$$

Gradient Descent (GD) Method

Constant step-size $\alpha_k = \alpha > 0$

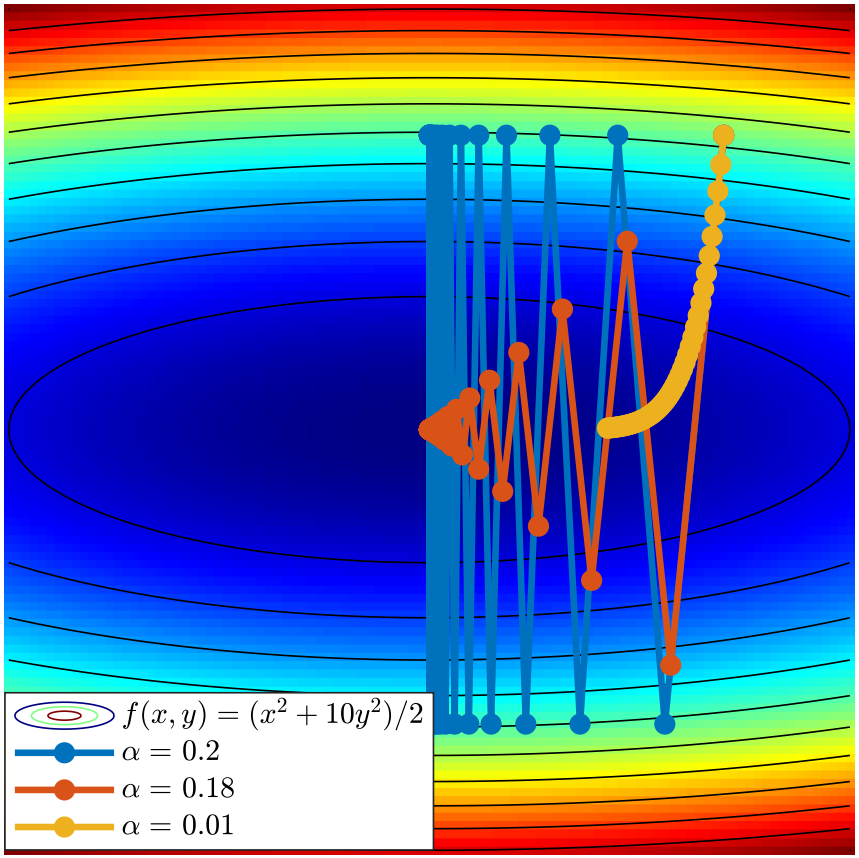
$$x_{k+1} = x_k - \alpha \nabla f(x_k), \quad k = 0, 1, 2, \ldots$$



Gradient Descent (GD) Method

Constant step-size $\alpha_k = \alpha > 0$

$$x_{k+1} = x_k - \alpha \nabla f(x_k), \quad k = 0, 1, 2, \ldots$$



- α needs to be "sufficiently small" in order to guarantee convergence.
- However, if α is "too small", the convergence might be too slow (expensive!)

Gradient Descent: convergence properties

We analyze the gradient descent assuming gradient of f is Lipschitz continuous:

There exists an $L > 0$ such that for all x and y we have

$$\|\nabla f(x) - \nabla f(y)\| \leqslant L\|x - y\|.$$

L is called the Lipschitz constant of the gradient.

Descent Lemma

If ∇f is Lipschitz continuous with constant L, then we have

$$f(y) \leqslant f(x) + \nabla f(x)^T(y - x) + \frac{L}{2}\|y - x\|^2, \quad \forall\ x, y \in \mathbb{R}^n.$$

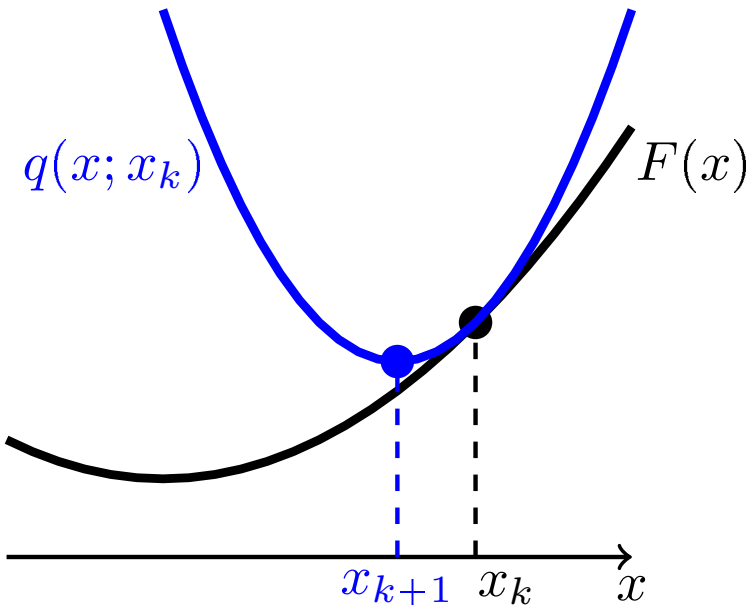
If $y = x + \alpha p$, $\alpha > 0$, $p \in \mathbb{R}^n$, then:

$$f(x + \alpha p) \leqslant f(x) + \alpha \nabla f(x)^T p + \frac{L}{2}\alpha^2\|p\|^2.$$

Gradient descent: case $\alpha_k = 1/L$

By the Descent Lemma, we have

$$\begin{aligned} x_{k+1} &= x_k - \frac{1}{L}\nabla f(x_k) \\ &= \operatorname*{argmin}_{y \in \mathbb{R}^n} \frac{L}{2}\|y - (x_k - \frac{1}{L}\nabla f(x_k))\|^2 \\ &= \operatorname*{argmin}_{y \in \mathbb{R}^n} \underbrace{f(x_k) + \nabla f(x_k)^T(y - x_k) + \frac{L}{2}\|y - x_k\|^2}_{=q(y;x_k) \geqslant f(y)}. \end{aligned}$$



Majorization-Minimization (MM)
At each iteration, replace $f(x)$ with a quadratic majorizer $q(x; x_k)$ at x_k and minimize $q(x; x_k)$.

Gradient descent: case $\alpha_k = 1/L$

$$x_{k+1} = x_k - \frac{1}{L}\nabla f(x_k), \quad k = 0, 1, 2, \ldots$$

Decrease of the objective function

- By the Descent Lemma, we have

$$f(x_{k+1}) = f(x_k - (1/L)\nabla f(x_k)) \leqslant f(x_k) - \frac{1}{2L}\|\nabla f(x_k)\|^2.$$

If $\nabla f(x_k) \neq 0$, this implies $f(x_{k+1}) < f(x_k)$.

(Weak) convergence to critical points

- Still from the Descent Lemma, it follows that

$$\|\nabla f(x_k)\|^2 \leqslant 2L(f(x_{k+1}) - f(x_k)).$$

Taking the limit for $k \to \infty$ yields $\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$.

The same convergence properties hold for any $0 < \alpha_k < 2/L$.

However, the knowledge of the Lipschitz constant L is needed.

Gradient descent: case $\alpha_k = 1/L$

Assume gradient of f Lipschitz continuous and $f \geqslant f_{low}$ bounded from below;

- Given $\epsilon_g > 0$, $\alpha_k = \alpha \leqslant 1/L$, the gradient descent method achieves

$$\|\nabla f(x_k)\| \leqslant \epsilon$$

in at most

$$k = \left\lceil \epsilon_g^{-2} \sqrt{\frac{2(f(x_0) - f_{low})}{\alpha}} \right\rceil - 1 \quad \text{iterations.}$$

In other words, $\|\nabla f(x_k)\|^2$ decreases as $O(1/k)$.

- Assume f convex and let $f(x^*)$ be the optimal value.
Given $\epsilon_f > 0$, $\alpha \leqslant 1/L$, the gradient descent method achieves

$$f(x_k) - f(x^*) \leqslant \epsilon_f$$

in at most

$$k = \left\lceil \epsilon_f^{-1} \frac{\|x_0 - x^*\|^2}{2\alpha} \right\rceil \quad \text{iterations.}$$

In other words, $f(x_k) - f(x^*)$ decreases as $O(1/k)$.

Gradient descent: case $\alpha_k = 1/L$

- Assume f strongly μ-convex, i.e. there exists a constant $\mu > 0$ such that

$$f(x) \geqslant f(y) + \langle \nabla f(y)^T (x - y) + \frac{\mu}{2} \|x - y\|^2 \quad \text{for all} \quad x, y \in \mathbb{R}^n,$$

then, if $0 < \alpha < 2/(\mu + L)$, SG achieves linear convergence

$$f(x_k) - f(x^*) = O(\rho^k),$$

with ρ depending on L/μ, $\rho \in (0, 1)$.

✓ Logistic regression + ℓ_2-regularization is strongly convex and has Lipschitz continuous gradient.

Finite sum minimization: subsampling

Consider a finite-sum minimization problem:

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{N} \sum_{i=1}^{N} \phi_i(x)$$

where N is large.

- Training sets often show redundancy in the data
  $\Rightarrow$ using all the sample data in every optimization iteration is inefficient
- Idea: work with small samples (at least initially)
- Methods in literature use subsampled f and/or ∇f and/or $\nabla^2 f_M$

Finite sum minimization: subsampling

Consider a finite-sum minimization problem:

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{N} \sum_{i=1}^{N} \phi_i(x)$$

where N is large.

Subsampling

- $\mathcal{S}_k \subseteq \{1, \ldots, N\}$ randomly and uniformly selected
- $|\mathcal{S}_k|$: sample size at iteration k

$$\nabla_{\mathcal{S}_k} f(x_k) \quad = \quad \frac{1}{|\mathcal{S}_k|} \sum_{i \in \mathcal{S}_k} \nabla \phi_i(x_k)$$

Stochastic optimization methods

$$\min_{x \in \mathbb{R}^n} f(x), \qquad f : \mathbb{R}^n \to \mathbb{R}$$

where f

$$f(x) = \begin{cases} \mathbb{E}[\phi(x, \xi)] \\ \text{or} \\ \dfrac{1}{N} \displaystyle\sum_{i=1}^{N} \phi_i(x) \end{cases}$$

Algorithm : Stochastic Gradient Methods

1. Choose $x_0 \in \mathbb{R}^n$

2. For $k = 0, 1, \ldots$ do

2.1 Generate a realization of the random variable ξ_k.

2.2 Compute a stochastic gradient $g(x_k, \xi_k)$.

2.3 Choose a stepsize $\alpha_k > 0$.

2.4 Set $x_{k+1} = x_k - \alpha_k g(x_k, \xi_k)$.

Ingredients

- A mechanism for generating a realization ξ_k of ξ:
  ξ_k may represent the choice of a single training sample or a set of samples.
- A mechanism for forming $g(x_k, \xi_k)$.
- A mechanism for computing a scalar stepsize $\alpha_k > 0$,
  e.g., fixed stepsizes or diminishing stepsizes.

Stochastic process

The generated sequence $\{x_k\}$ is not determined uniquely by f, the starting point x_0 and the sequence of stepsizes $\{\alpha_k\}$ unlike the deterministic gradient descent method.

Rather, $\{x_k\}$ is a stochastic process whose behaviour is determined by the random sequence $\{\xi_k\}$.

Stochastic gradient: subsampling

$$f(x) = \mathbb{E}[\phi(x, \xi)]$$

Let ξ_k represent the independent choice of a single training sample or of a set $\{\xi_{k,i}\}$ of samples (according to the distribution P)

$$g(x_k, \xi_k) = \begin{cases} \nabla\phi(x_k, \xi_k) \\ \dfrac{1}{N_k} \displaystyle\sum_{i=1}^{N_k} \nabla\phi_i(x_k, \xi_{k,i}), \quad N_k \in \mathbb{N} \end{cases}$$

Stochastic Gradient method (SG): a single item is drawn.

Mini-batch SG: a (small) set $\{\xi_{k,i}\}$ of samples is drawn.

Stochastic gradient: subsampling

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} \phi_i(x)$$

Let ξ_k represent the independent choice of a single training sample or a set of samples in $\{1, \ldots, N\}$

$$g(x_k, \xi_k) = \begin{cases} \nabla \phi_{i_k}(x_k) \\ \dfrac{1}{N_k} \displaystyle\sum_{i \in \mathcal{S}_k} \nabla \phi_i(x_k), \quad N_k = card(\mathcal{S}_k), \quad 1 \leqslant N_k \ll N \end{cases}$$

Stochastic Gradient method (SG)

Choose randomly and uniformly the index $i_k \in \{1, \ldots, N\}$ and compute

$$x_{k+1} = x_k - \alpha_k \nabla \phi_{i_k}(x_k), \quad k = 0, 1, 2, \ldots$$

Mini-batch Stochastic gradient (Mini-batch SG)

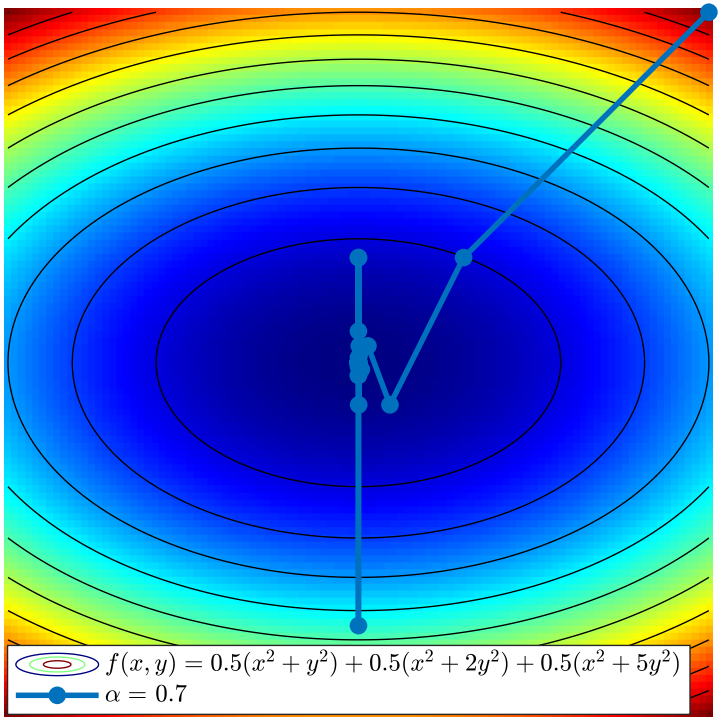
Choose randomly and uniformly the sample $\mathcal{S}_k \subseteq \{1, \ldots, N\}$, let $N_k = card(\mathcal{S}_k)$ be the sample size and compute

$$x_{k+1} = x_k - \frac{\alpha_k}{N_k} \sum_{i \in \mathcal{S}_k} \nabla \phi_i(x_k), \quad k = 0, 1, 2, \ldots$$

Stochastic gradient: subsampling

Constant step-size $\alpha_k = \alpha > 0$

$$x_{k+1} = x_k - \alpha \nabla \phi_{i_k}(x_k), \quad k = 0, 1, 2, \ldots$$



- SG does not guarantee the decrease of the objective function

  $\Rightarrow$ oscillatory behaviour when close to the minimum.

Motivations

- Intuitive Motivation
  In reality, a training set does not consist of exact duplicates of sample data but the data are a large set and redundant.

  This suggests that using all the sample data in every optimization is inefficient.

  Working with small (single) samples can be convenient.

  Using the mini-batch approximation reduces the variance of the stochastic gradient estimate with respect to the true gradient.

- Practical Motivation
  Pros: the cost per-iteration of SG and mini-batch SG is low.

  For the finite-sum minimization problem, an epoch represents N single evaluations $\nabla\phi_i$, i.e., the cost of one full gradient evaluation.

  SG performs N steps per epoch.

  Cons: it is necessary to run the algorithm repeatedly in order to appropriately tune the step-size (learning rate) and thus obtain an efficient solution.

SG for strongly convex f

Lipschitz continuity

$f\colon \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and $\nabla f\colon \mathbb{R}^n \to \mathbb{R}^n$, is Lipschitz-continuous with Lipschitz constant $L > 0$

$$\|\nabla f(x) - \nabla f(\bar{x})\|_2 \leqslant L\|x - \bar{x}\|_2, \quad x, \bar{x} \in \mathbb{R}^n.$$

Strong convexity

There exists a constant $0 < c \leqslant L$ s.t.

$$f(\bar{x}) \geqslant f(x) + \nabla f(x)^T(\bar{x} - x) + \frac{1}{2}c\|\bar{x} - x\|_2^2, \quad \forall x, \bar{x} \in \mathbb{R}^n$$

Hence f has a unique minimizer x_* with $f_* = f(x_*)$.

✓ Logistic regression + ℓ_2-regularization is strongly convex and has Lipschitz continuous gradient.

SG for strongly convex f

Assumptions

- $\{x_k\}$ is contained in an open convex set where f is bounded below by a scalar f_*.
- For all $k \in \mathbb{N}$

$$\nabla f(x_k)^T \mathbb{E}_{\xi_k}\left[g(x_k, \xi_k)\right] \geqslant \mu \|\nabla f(x_k)\|_2^2, \quad \text{for some positive } \mu \tag{1}$$

in expectation $-g(x_k, \xi_k)$ is a direction of sufficient descent for f with norm comparable to the norm of $\nabla f(x_k)$ (trivial with $\mu = 1$ if $g(x_k, \xi_k)$ is an unbiased estimate of $\nabla f(x_k)$)

- For all $k \in \mathbb{N}$

$$\mathbb{E}_{\xi_k}\left[\|g(x_k, \xi_k)\|_2^2\right] \leqslant M_1 + M_2 \|\nabla f(x_k)\|_2^2, \quad \text{for some positive } M_1, M_2 \geqslant \mu^2 \tag{2}$$

SG for strongly convex f

For small enough stepsizes, $\{f(x_k)\}$ gets near to the optimal value

Theorem

Suppose that SG is run with a fixed stepsize $\alpha_k = \bar{\alpha}, \forall k$ s.t.

$$0 < \bar{\alpha} \leqslant \frac{\mu}{LM_2}.$$

Then, for all $k \in \mathbb{N}$

$$\begin{aligned} \mathbb{E}[f(x_k) - f(x_*)] \quad &\leqslant \quad \frac{\bar{\alpha}LM_1}{2c\mu} + (1 - \bar{\alpha}c\mu)^k \left(f(x_0) - f(x_*) - \frac{\bar{\alpha}LM_1}{2c\mu} \right) \\ &\xrightarrow{k \to \infty} \quad \frac{\bar{\alpha}LM_1}{2c\mu} \end{aligned}$$

SG for strongly convex f

Consequences:

- If there is no noise in the gradient computation or the noise decays with $\|\nabla f(x_k)\|_2^2$, then $M_1 = 0$.

  Hence, linear convergence to the optimal value occurs.

- The initial optimality gap $f(x_0) - f(x_*)$ appear with an exponentially decreasing factor.

- Selecting a smaller stepsize worsens the contraction constant in the convergence rate but allows one to arrive closer to the optimal values.

SG: f strongly convex and diminishing stepsizes

More general requirements for α_k take the form

$$\sum_{i=0}^{\infty} \alpha_k = \infty, \qquad \sum_{i=0}^{\infty} \alpha_k^2 < \infty,$$

Theorem

Suppose SG is run with a stepsize s.t. for all $k \in \mathbb{N}$

$$\alpha_k = \frac{\beta}{\gamma + k} \quad \text{for some} \quad \beta c\mu > 1 \quad \text{and} \quad \gamma > 0 \quad \text{s.t.} \quad \alpha_0 \leqslant \frac{\mu}{LM_2}$$

Then for all $k \in \mathbb{N}$, the optimality gap satisfies

$$\mathbb{E}[f(x_k) - f(x_*)] \leqslant \frac{\nu}{\gamma + k},$$

where

$$\nu = \max\left\{ \frac{\beta^2 LM_1}{2(\beta c\mu - 1)}, \gamma(f(x_0) - f(x_*)) \right\}.$$

Summary of SG: f strictly convex and diminishing stepsizes

- Constant stepsizes $\bar{\alpha}$ were required to satisfy $\bar{\alpha} \leqslant \frac{\mu}{LM_2}$
- For diminishing stepsizes, α_0 is required to satisfy the same bound.
- Successive steps α_k, $k \geqslant 1$, are of order $O(1/k)$ and depend on the strong convexity parameter c since $\beta > 1/(c\mu)$.
- The rate of convergence of the optimality gap is $\mathbb{E}[f(x_k) - f(x_*)] = O(1/k)$

GD vs SG

GD

$$f(x_k) - f(x_*) = O(\rho^k),\ \rho \in (0,1) \quad \Rightarrow \quad f(x_k) - f(x_*) \leqslant \epsilon, \quad k = O\left(\ln(1/\epsilon)\right)$$

The cost per-iteration is proportional to N (need to compute $\nabla\phi_i$, $1 \leqslant i \leqslant N$)

Total work to obtain ϵ-optimality is proportional to $N \ln(1/\epsilon)$

SG

$$\mathbb{E}[f(x_k) - f(x_*)] = O\left(1/k\right) \quad \Rightarrow \quad \mathbb{E}[f(x_k) - f(x_*)] \leqslant \epsilon, \quad k = O\left(1/\epsilon\right)$$

Unitary per-iteration cost

$$1/\epsilon \geqslant N \ln(1/\epsilon) \qquad \text{for moderate values of } N, \epsilon$$

while

$$1/\epsilon < N \ln(1/\epsilon) \qquad \text{in a big data regime where } N \text{ is large}$$

SG for general f

Recall that $\{f(x_k)\}$ is assumed to be bounded below by a scalar f_*.

Theorem (f nonconvex, fixed stepsize)

Suppose that SG is run with a fixed stepsize $\alpha_k = \bar{\alpha}$ for all $k \in \mathbb{R}$ s.t.

$$0 < \bar{\alpha} \leqslant \frac{\mu}{LM_2}.$$

Then, for all $K \in \mathbb{N}$

$$\mathbb{E}\left[\frac{1}{K}\sum_{k=1}^{K}\|\nabla f(x_k)\|_2^2\right] \leqslant \frac{\bar{\alpha}LM_1}{\mu} + 2\frac{f(x_0)-f_*}{K\mu\bar{\alpha}} \xrightarrow{K\to\infty} \frac{\bar{\alpha}LM_1}{\mu}$$

This result characterizes the expected average-squared gradients of f

The average norm of the gradients can be made arbitrarily small by selecting a small stepsize (but doing so reduces the speed at which $\|\nabla f(x_k)\|_2$ approaches its limiting distribution).

SG for general f

Recall that $\{f(x_k)\}$ is assumed to be bounded below by a scalar f_*.

Theorem (f nonconvex, diminishing stepsizes)

Suppose that SG is run with a stepsize sequence satisfying

$$\sum_{i=1}^{\infty} \alpha_k = \infty, \qquad \sum_{i=1}^{\infty} \alpha_k^2 < \infty,$$

Then,

$$\liminf_{k \to \infty} \mathbb{E}\left[\|\nabla f(x_k)\|_2^2\right] = 0$$

i.e., gradient norms cannot asymptotically stay far from zero.

If we further assume that f is twice continuously differentiable and $\|\nabla f(x)\|_2^2$ has Lipschitz continuous derivatives then

$$\lim_{k \to \infty} \mathbb{E}\left[\|\nabla f(x_k)\|_2^2\right] = 0$$

Mini-batch SG

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} \phi(x)$$

$$g(x_k, \xi_k) = \frac{1}{N_k} \sum_{i \in S_k} \nabla \phi_i(x_k), \quad card(S_k) = N_k = N_{\text{mb}}, \forall k$$

The variance of the direction is reduced by a factor $1/N_{\text{mb}}$.

Previous results can be extended to mini-batch SG, e.g.,

- SG, f strongly convex, constant and sufficiently small stepsize

$$\mathbb{E}[f(x_k) - f(x_*)] \leqslant \frac{\bar{\alpha} L M_1}{2c\mu} + (1 - \bar{\alpha} c \mu)^k \left(f(x_0) - f(x_*) - \frac{\bar{\alpha} L M_1}{2c\mu} \right)$$

- Mini-batch SG, f strongly convex, constant and sufficiently small stepsize

$$\mathbb{E}[f(x_k) - f(x_*)] \leqslant \underbrace{\frac{\bar{\alpha} L M_1}{2c\mu N_{\text{mb}}}}_{\text{smaller asymptotic gap}} + (1 - \bar{\alpha} c \mu)^k \left(f(x_0) - f(x_*) - \frac{\bar{\alpha} L M_1}{2c\mu N_{\text{mb}}} \right)$$

But the computation of $g(x_k, w_k)$ is N_{mb} times more expensive than in SG.
In general the methods can be comparable but mini-batch method can be parallelized.

Numerical experiment

Given $\{(a_i, b_i)\}_{i=1,\dots,N}$ the Mushrooms dataset[1], $b_i \in \{0, 1\}$, learn the parameters $x \in \mathbb{R}^n$ of the logistic regression by solving

$$\min_{x \in \mathbb{R}^n} f(x) = \frac{1}{N} \sum_{i=1}^{N} \underbrace{\left(b_i - \frac{1}{1 + e^{-a_i^T x}} \right)^2}_{:= \phi_i(x)}.$$

DATA	Training Size N	Test Size	Numb. of Features d
Mushrooms	6503	1621	112

Safe to eat or deadly poison?

[1] https://www.kaggle.com/uciml/mushroom-classification

Numerical experiment

GD

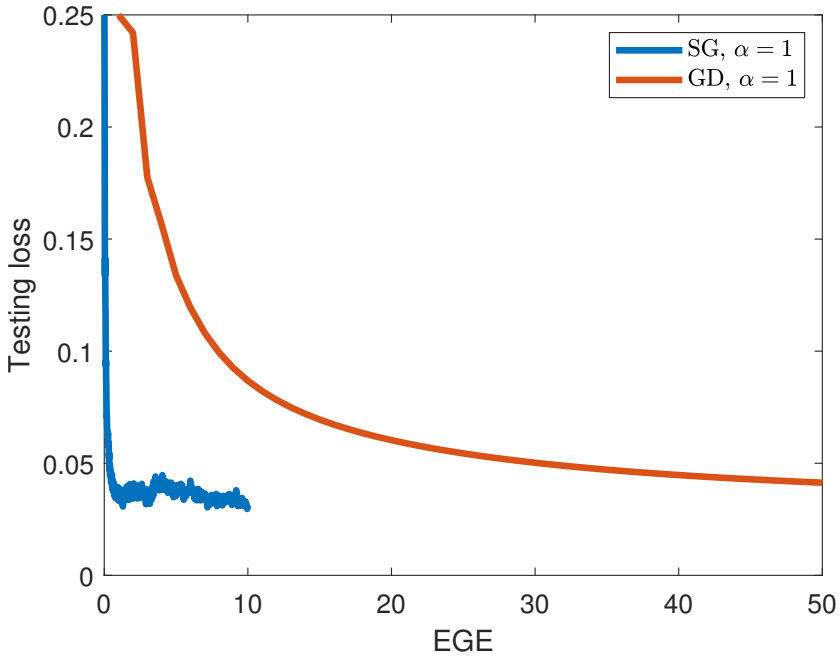
$$x_{k+1} = x_k - \frac{\alpha}{N} \sum_{i=1}^{N} \nabla \phi_i(x_k).$$

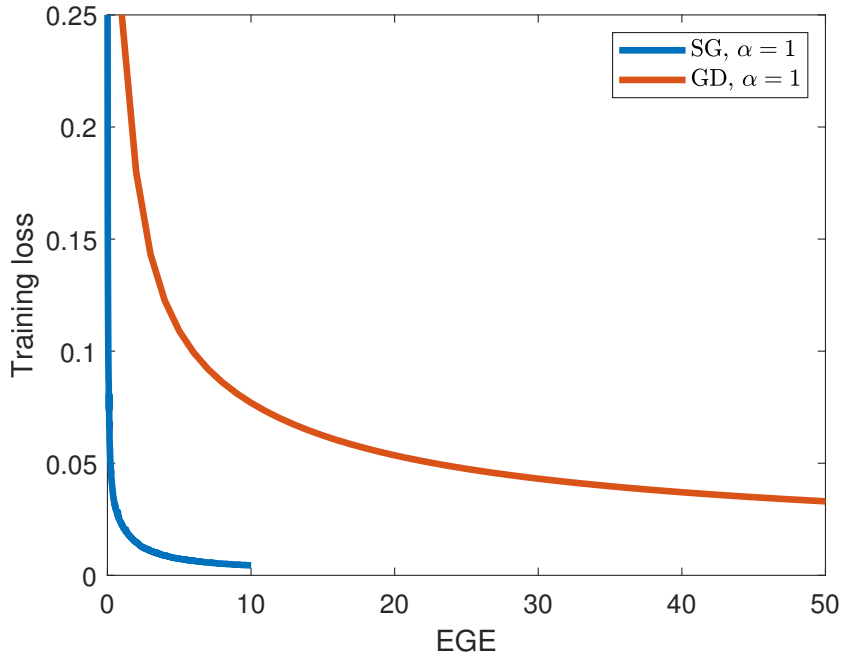
Mini-batch SG

$$x_{k+1} = x_k - \frac{\alpha}{|S|} \sum_{i \in S} \nabla \phi_i(x_k), \quad 1 < |S| << N.$$

For these tests, we set $\alpha = 1$ and $|S| = 65$.
The sample S is computed by random and uniform subsampling.

Numerical experiment





- Training loss (for the training set $\mathcal{T}$)

$$f_N(x) = \frac{1}{N} \sum_{i \in \mathcal{T}} \phi_i(x).$$

- Testing loss (for the testing set $\mathcal{T}_T$)

$$f_N(x) = \frac{1}{N_T} \sum_{i \in \mathcal{T}_T} \phi_i(x).$$