Introduction to stochastic optimization for large-scale problems

Dr. Ir. Valentin Leplat

Skoltech
Center for Artificial Intelligence and Technology

Email: v.leplat@skoltech.ru
Website: https://sites.google.com/view/valentinleplat

ISP Seminar - Part 1

23 th Jan. 2023

Table of Contents

1. General Introduction
2. Formal Optimization Problem Statements
3. First-order methods
4. Stochastic vs Batch Opti. methods
5. Theoretical analysis of SGD
6. Project proposals

Why ?

- **History**: optimization has always played an important role in data sciences and in almost all discipline of engineering !
- **Observation**: recent and rapid development of machine learning methods with very successful real-world applications
- **Need**: fast optimization methods for training high-dimensional models over large datasets

What ?

- we introduce state-of-the-art first-order stochastic gradient methods for solving large-scale optimization problems,
- review their theoretical background on convergence rate analysis,
- present some applications to observe these powerful methods at work !

How ?

Evaluation: presentation of a personal project, it could be:

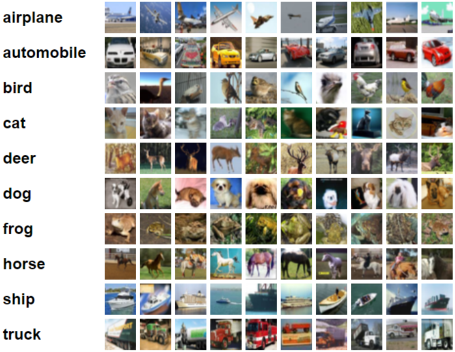
- the implementation of stochastic optimization methods for solving your problem of interest, or a proposed subject.
- an original and detailed presentation of research papers in relationship with stochastic optimization
- the construction and the analysis of a new stochastic optimization method for solving an important problem,

Rules of the game:

1. 23th and 24th January - 9am-1pm : lectures (approx. 6 hours)
2. 27th January: 20 minutes presentation to the class (slides to be shared)
3. project can be done alone or in group of two to three.
4. Estimated personal workload: 8-12 hours
5. learn a lot and practice a lot

A large-scale machine learning example

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.



Source ▸ Link

A large-scale machine learning example

- Denote the images in the data set by:

$$\{x_i\}_{i=1}^{n} \quad \text{with } x_i = \underbrace{(x_i^{(1)}, \cdots, x_i^{(d_x)})}_{\text{pixels in the image}} \in \mathbb{R}^{d_x}$$

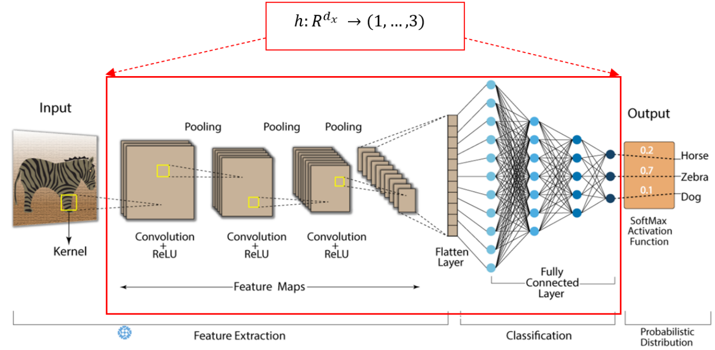
- Denote the labels by:

$$\{y_i\}_{i=1}^{n} \quad \text{with } y_i \in (1, \cdots, 10) \in \mathbb{R}$$

- We call the set $\{(x_1, y_1), \cdots, (x_n, y_n)\} \in \mathbb{R}^{d_x} \times \mathbb{R}$ as the **training points**.
- For CIFAR-10: $n = 60000$ and $d_x = 32 \times 32 \times 3 = 3072$ since every pixel has three values for RGB colors.

A large-scale machine learning example

Goal: construct a classifier, i.e., find a *prediction function* $h : \mathbb{R}^{d_x} \to (1, \cdots, 10)$ such that $h(x_i) = y_i$ for most i $(1 \leqslant i \leqslant n)$.



Source ▸ Link

Table of Contents

1. General Introduction
2. Formal Optimization Problem Statements
3. First-order methods
4. Stochastic vs Batch Opti. methods
5. Theoretical analysis of SGD
6. Project proposals

Blanket assumptions

This course: underlying space is the Euclidean space $\mathbb{R}^d$, a particular case of Hilbert space $\mathcal{W}$ of finite dimension d, that is, a Banach space equipped with:

- an inner product $\langle .,. \rangle$, here we consider the dot product $\langle x, y \rangle = \sum_i^d x^{(i)} y^{(i)}$ for $x, y \in \mathbb{R}^d$,
- induced norm $\|.\| = \sqrt{\langle .,. \rangle}$

Notation: $x^{(i)}$ denotes the i-th component of x.

Set of assumptions on functions

Through this course:

- focus on the minimization of a differentiable function F without constraints.
- important to always keep in mind the set of assumptions made on the functions, in particular on F (the primal objective function).
- The most important assumptions will be highlighted in red when needed
- The derived results (mainly linked to the convergence results of the algorithms discussed in the present document) are only valid in the set of assumptions considered (= the paradigm).

Prediction and Loss functions

- Assumption: h has a *fixed form* and is parameterized by a real vector $w \in \mathbb{R}^d$ (variables)
- **Formally**: for given $h(.,.) : \mathbb{R}^{d_x} \times \mathbb{R}^d \to \mathbb{R}^{d_y}$, we consider the family of prediction functions:

$$H := \{h(., w) : w \in \mathbb{R}^d\} \tag{1}$$

- **Goal**: find $h \in H$ that min. the losses incurred from inaccurate predictions. $h(x_i, w)$ is the model that explains the data using the parameters w.
- **How ?**: we assume a given real-valued loss function $l : \mathbb{R}^{d_y} \times \mathbb{R}^{d_y} \to \mathbb{R}$ such that for a given input-output pair (x_i, y_i), yields the cost

$$l(h(x_i; w), y_i)$$

where $h(x_i; w)$ and y_i are resp. the predicted and true outputs, e.g., $l(z, y) = \frac{1}{2}\|z - y\|_2^2$

Expected Risk

- **Ideally:** w chosen to min. the *expected* loss that could be incurred from *any* input-output pair, but how ?
- **Formal approach**: assume that losses are measured w.r.t. probability distribution $P(x,y)$, that is the *true* relationship between inputs and outputs.
- Assume: input-output space $\mathbb{R}^{d_x} \times \mathbb{R}^{d_y}$ is endowed with $P : \mathbb{R}^{d_x} \times \mathbb{R}^{d_y} \rightarrow [0,1]$ (simultaneously represents the distribution $P(x)$ of inputs as well as the conditional probability $P(y|x)$ of the label y being appropriate for an input x.)
- **Goal**: we want to solve

$$\min_{w \in \mathbb{R}^d} \mathbb{E}[l(h(x;w),y)] := \int_{\mathbb{R}^{d_x} \times \mathbb{R}^{d_y}} l(h(x;w),y) dP(x,y) \qquad (2)$$

- Further: denote $\mathbb{E}[l(h(x;w),y)] = R(w)$, the *expected risk*

Empirical Risk

- **A little snag**: that would great to solve Problem (2)... but it is untenable since not enough info about P, then cannot compute $\mathbb{E}[]$.
- **In practice:** seeks solution that involves an **estimate** of R.
- **For supervised learning**: we have a set of $n \in \mathbb{N}$ i.i.d. input-output samples $\{(x_i, y_i)\}_{i=1}^{n}$.
  Assumption of i.i.d. samples → samples do not depend on the optimization variable w.
- **Problem**: we want to min. the *empirical risk* function $R_n : \mathbb{R}^d \to \mathbb{R}$:

$$\min_{w \in \mathbb{R}^d} R_n(w) := \frac{1}{n} \sum_{i=1}^{n} l(h(x_i; w), y_i) \tag{3}$$

Empirical Risk - remarks

- Min. of R_n may be considered as the practical optimization problem of interest,
- We consider the unregularized formulation (3), in practice we generally want to min.

$$\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} l(h(x_i; w), y_i) + g(w) \tag{4}$$

  where $g(w)$ is a regularization/penalty function, used to whether promote structure for the solution such as sparsity ($g(w) = \|w\|_1$) or limit the so-called over-fitting phenomena ($g(w) = \frac{1}{2}\|w\|_2^2$).
- **However**: the optimization methods discussed in this course can be applied readily when a smooth regularization term is included.
- **and**: if time, we will analyze the "Proximal stochastic gradient".

Simplified notations

- Expressions (2) and (3) show explicit dependence on the loss function, sample space, sample set, etc.
- We will employ a simplified notation → offers some advantages for generalizing certain algorithmic ideas.
- **Sample representation**: use random seed ξ; realization might be:
  1. a single sample (x, y) from $\mathbb{R}^{d_x} \times \mathbb{R}^{d_y}$,
  2. a set of samples $\{(x_i, y_i)\}_{i=1}^{n}$
- **Loss function notation**: for a given (w, ξ) we use $f(w; \xi)$
  → f is the composition $l \circ h$

Simplified notations

- **Expected Risk**: expected value of f taken w.r.t. distribution of ξ:

$$R(w) = \mathbb{E}[f(w;\xi)] \tag{5}$$

- **Empirical Risk**: given a set of realizations $\{\xi_{[i]}\}_{i=1}^{n}$, the loss incurred for the ith sample:

$$f_i(w) := f(w;\xi_{[i]})$$

  Then:

$$R_n(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w) \tag{6}$$

  ”finite sum of functions” form.

Simplified notations

Important remarks

- We use $\xi_{[i]}$ to denote the i-th element of a fixed set of realizations of a random variable ξ,
- *Later*: ξ_k denotes the k-th element of a sequence of random variables, each ξ_k drawn independently according to the distribution P.

Training and validation sets

Sample points: $\{(x_1, y_1), \ldots, (x_n, y_n)\} \in \mathbb{R}^d_x \times \mathbb{R}$

- Set $\mathcal{A} \subset \{1, \ldots, n\}$
- Training points: $\{(x_i, y_i)\}_{i \in \mathcal{A}}$
- Solve:

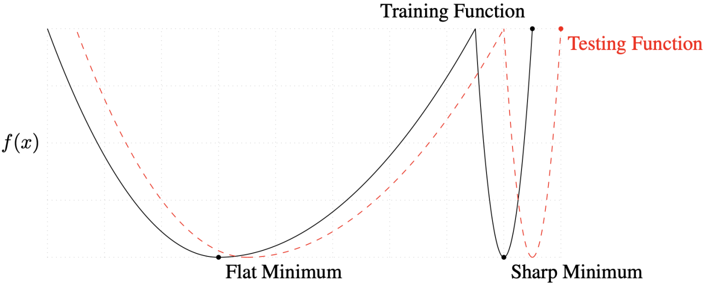
$$\min_w \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} f_i(w)$$

- Set $\mathcal{B} \subset \{1, \ldots, n\} \backslash \mathcal{A}$
- Validation points: $\{(x_i, y_i)\}_{i \in \mathcal{B}}$
- Check:

$$\min_w \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} f_i(w)$$

Flat vs Sharp minima

Confusing goal: we solve a problem in the hope to solve another one (and obtain a good generalization of the model).



From (Keskar et al., 2017).

Example 1

- Problems (3) are ubiquitous when solving a machine learning problem. Let us illustrate this by the example of logistic regression
- **Name**: Maximum likelihood estimator for logistic regression
- **Context**:
  1. consider a classification problem denoted by observations $\{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}^d$ for all i and $y_i \in \{-1, 1\}$.
  2. Each observation is supposed to be independent and there exists a vector $w \in \mathbb{R}^d$ and $w_0 \in \mathbb{R}$ such that for all i, (y_i, x_i) is a realization of the random variable (Y, X) whose law $P(X, Y)$ satisfies:

$$P(Y = 1|X; w, w_0) = \frac{e^{\langle X, w\rangle + w_0}}{1 + e^{\langle X, w\rangle + w_0}}$$

This example can be confusing since we know here a little bit about $P(X, Y)$, that is the *conditional probability*, we are somewhere in between the expected and the empirical risk min., for which the well-known maximum likelihood estimation method can be used.

Example 1 - objectives

1. Show that for all i, $P(Y = y_i | x_i; w, w_0) = \frac{1}{1+e^{-y_i(\langle x_i, w\rangle + w_0)}}$
2. Show that the maximum likelihood estimator is:

$$(w^\star, w_0^\star) = \arg\min_{w,w_0} \sum_{i=1}^{n} \log(1 + e^{-y_i(\langle x_i, w\rangle + w_0)}) \tag{7}$$

3. Denote $f(w, w_0) = \log(1 + e^{-y_i(\langle x_i, w\rangle + w_0)})$, compute $\nabla f(w, w_0)$

Remarks:

- In the exercise, we have $\xi_{[i]} = (x_i, y_i)$.
- Since we have n observations, it is possible to evaluate the objective function.
- However, when n is large, say millions or billions, this can be tedious task!!

Table of Contents

1. General Introduction
2. Formal Optimization Problem Statements
3. First-order methods
4. Stochastic vs Batch Opti. methods
5. Theoretical analysis of SGD
6. Project proposals

Generalities

- Problems (6) are large-scale; number of samples n and parameters dimension d are (very) large.
- Most optimization methods designed to solve these problems are first-order methods.
- This means that at each iteration, we only use the information of the gradient.
- **In this section**: we introduce two fundamental first-order methods.

Gradient Descent

The most well-known and simple first-order method or a differentiable function F:

Initialization

Set $w_0 \in \mathcal{W}$.

Iteration $(k \geqslant 0)$:

1. Choose a step size α_k (s.t. $F(w_{k+1}) < F(w_k)$)
2. Compute
$$w_{k+1} \leftarrow w_k - \alpha_k \nabla F(w_k) \tag{8}$$

Many variants of this method: differ from the *step size strategy*

- The sequence $\{\alpha_k\}_{k=0}^{\infty}$ is chosen in advance, ex: $\alpha_k = \alpha$ (constant) or $\alpha_k = \frac{\alpha_0}{\sqrt{k+1}}$
- Full relaxation: $\alpha_k = \arg\min_{\alpha \geqslant 0} F(w_k - \alpha \nabla F(w_k))$
- *Armijo* rule

How to compute the Gradient ?

Using partial derivatives

- We know that the gradient is the vector of all the partial derivatives. Hence, we can compute $\frac{\partial F}{\partial w^{(i)}}(w)$ for all i and reconstruct the vector $\nabla F(w) := (\frac{\partial F}{\partial w^{(1)}}(w), ..., \frac{\partial F}{\partial w^{(d)}}(w))^T$.
- Example: Consider $F(w) = \|Aw - b\|_2^2$ with $A \in \mathbb{R}^{m \times n}$, we can write:

$$F(w) = \sum_{i=1}^{m} (\sum_{j=1}^{n} A^{(i,j)} w^{(j)} - b^{(i)})^2$$

- and so:

$$\frac{\partial F}{\partial w^{(k)}}(w) = 2 \sum_{i=1}^{m} A^{(i,k)} (\sum_{j=1}^{n} A^{(i,j)} w^{(j)} - b^{(i)})$$

- We recognise the components of the vector: $\nabla F(w) = 2A^T(Aw - b)$.

How to compute the Gradient ?

Using the definition

- We compute $F(w+h)$ and try to isolate $F(w)$, a linear term in h and a negligible term.
- Example: Consider $F(w) = \|Aw - b\|_2^2$.
- We write:

$$\begin{aligned} F(w+h) = \|A(w+h) - b\|_2^2 &= \|Aw - b\|_2^2 + 2\langle Aw - b, Ah \rangle + \|Ah\|_2^2 \\ &= F(w) + 2\langle A^T(Aw - b), h \rangle + o(h^2) \end{aligned}$$

How to compute the Gradient ?

Using chain rule Let as an exercise :) (practicing chain rule might be useful for the projects).

Convergence of Gradient Descent

Non-convex functions satisfying some conditions such as:

- $F \in C_M^{2,2}(\mathbb{R}^d)$, that F is 2 times continuously differentiable and $\|\nabla^2 F(x) - \nabla^2 F(y)\| \leqslant M\|x - y\|$ for any x and y
- There exists a local min. $w^\star$ at which Hessian satisfies $\mu I_d \leqslant \nabla^2 F(w^\star) \leqslant L I_d$ with $0 < \mu \leqslant L < \infty$
- w_0 close enough to $w^\star$, that is $r_0 = \|w_0 - w^\star\| < \bar{r} = \frac{2\mu}{M}$

By choosing $\alpha_k = \frac{2}{\mu+L}$, then Gradient descent method converges *linearly* and *locally* as follows:

$$\|w_k - w^\star\| \leqslant \frac{\bar{r} r_0}{\bar{r} - r_0} \left(1 - \frac{2\mu}{L + 3\mu}\right)^k$$

General non-convex functions: convergence to a stationary point given that the step sizes are properly chosen.

Convergence of Gradient Descent

Theorem 1 - *Global* convergence

Let F be a convex differentiable function that has a minimizer $w^\star$ and whose gradient is L-Lipschitz continuous, that is $F \in C_L^{1,1}(\mathbb{R}^d)$. The gradient method with constant step size $\alpha_k = \frac{1}{L}$ satisfies

$$F(w_k) - F(w^\star) \leqslant \frac{L\|w_0 - w^\star\|^2}{2k} \tag{9}$$

Moreover, if F is μ-strongly convex, then:

$$\begin{aligned} F(w_k) - F(w^\star) &\leqslant (1 - \frac{\mu}{L})^k (F(w_0) - F(w^\star) + \frac{L}{2}\|w_0 - w^\star\|^2) \\ \|w_k - w^\star\|^2 &\leqslant (1 - \frac{\mu}{L})^k (\frac{2}{L}(F(w_0) - F(w^\star)) + \|w_0 - w^\star\|^2) \end{aligned} \tag{10}$$

Proof. We will prove more general results in the following of the course.

Can we do better?

Yes! In fact, much better: The *Accelerated* Gradient Descent Method

Initialization

Set $w_0 \in \mathcal{W}$, some $\alpha_0 \in (0,1)$, and set $y_0 = w_0$.

Iteration $(k \geqslant 0)$:

1. Compute $F(y_k)$ and $\nabla F(y_k)$. Set $w_{k+1} \leftarrow y_k - \frac{1}{L}\nabla F(y_k)$
2. Compute $\alpha_{k+1} \in (0,1)$ from Equation

$$\alpha_{k+1}^2 = (1-\alpha_{k+1})\alpha_k^2 + \frac{\mu}{L}\alpha_{k+1} \tag{11}$$

Set $\beta_k = \frac{\alpha_k(1-\alpha_k)}{\alpha_k^2+\alpha_{k+1}}$ and $y_{k+1} \leftarrow w_{k+1} + \beta_k(w_{k+1} - w_k)$

Accelerated Gradient Descent Method: Convergence

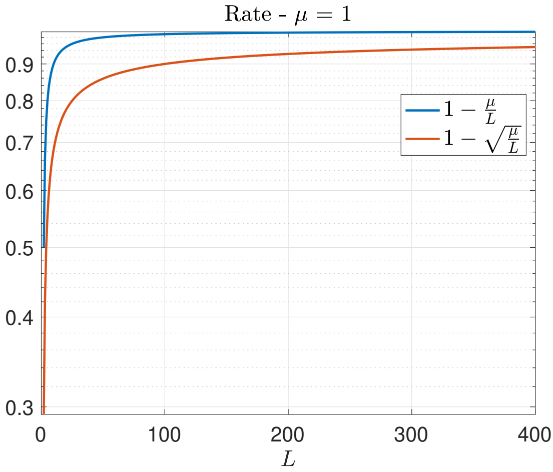
We omit the details in this slide, if requested, the exact forms of convergence rates can be provided :).

- Smooth convex functions: convergence in $O(\frac{1}{k^2})$ instead of $O(\frac{1}{k})$!!
- Smooth μ-strongly convex: convergence in $O((1-\sqrt{\frac{\mu}{L}})^k)$, instead of $O((1-\frac{\mu}{L})^k)$.
- For non-convex: need to use restart to guarantee convergence.

This is a significant acceleration, and it is optimal in the sense that no other first-order method can guarantee a faster convergence rate!
See (Y.Nesterov,2018), section 2.2 (”Optimal Methods”) for more details.

Accelerated Gradient Method: Convergence

Example: $\mu = 1$, $L = 10$: from 0.6838 vs 0.9 !



Accelerated Gradient Descent Method: Convergence

Test case: Given $A \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$, we want to solve: $\min_w 1/2\|Aw - b\|_2^2$. Code ▸ Link, Code Ocean ▸ Link

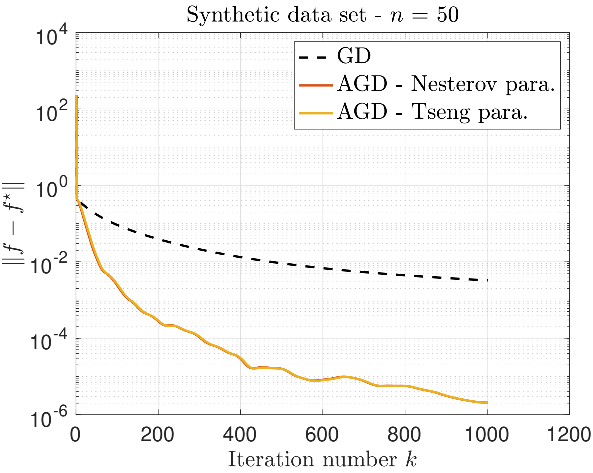


Table of Contents

1. General Introduction
2. Formal Optimization Problem Statements
3. First-order methods
4. Stochastic vs Batch Opti. methods
5. Theoretical analysis of SGD
6. Project proposals

Context

- Consider Empirical risk minimization Problems (recalled here-under):

$$\min_{w} F(w) := \frac{1}{n} \sum_{i=1}^{n} f_i(w)$$

- **Note**: later discussion will focus on the performance of such algorithms when considering the *true* measure of interest, the *expected risk*.
- We introduce two categories of opti. methods for machine learning:
    1. Stochastic: the prototypical method is the so-called Stochastic Gradient Descent (SGD)
    2. Batch: include Gradient Descent (GD) (often referred in ML community to as batch gradient or full gradient) method, the AGD, conjugate gradient, quasi-Newton and inexact Newton methods.

Stochastic Gradient Descent - Algorithm

Initialization

Set $w_0 \in \mathcal{W}$.

Iteration $(k \geqslant 0)$:

1. Choose $i_k \sim \mathcal{U}(\{1, \cdots, n\})$ (uniform dist.)
2. Choose a step size $\alpha_k > 0$
3. Set

$$w_{k+1} \leftarrow w_k - \alpha_k \nabla f_{i_k}(w_k) \tag{12}$$

Remarks:

- The index i_k (corresponding to the seed $\xi_{[i_k]}$, i.e., the sample pair (x_{i_k}, y_{i_k})) is chosen *randomly* from $\{1, \cdots, n\}$.
- $f_{i_k}(w_k) := f(w_k; \xi_{[i_k]})$
- if $f_{i_k}(w)$ is not differentiable → use a subgradient

Stochastic Gradient Descent - Insights

- Each iteration is very cheap; the computation of the gradient $\nabla f_{i_k}(w_k)$ corresponding to one sample.
- Unlike GD (deterministic process:)): $\{w_k\}_{k=0}^{\infty}$ is a stochastic process (driven by the random sequence $\{i_k\}_{k=0}^{\infty}$)
- $-\nabla f_{i_k}(w_k)$ might not be one descent from w_k; does not necessarily yield to a negative directional derivative for F from w_k
- **But**: if it is a descent direction in *expectation*
  → sequence $\{w_k\}_{k=0}^{\infty}$ can be guided toward a minimizer of F.

Stochastic Gradient Descent - Challenges

- the step size (a.k.a. learning rate) is difficult to tune since we cannot rely on monotonicity (and the method is very sensitive to this choice),
- it is difficult to obtain high accuracy solutions (the methods oscillates when getting close to a locally optimal solution) – in most practical problems, this is not an issue because it is not necessary to have high accuracy solutions (because the data is very noisy anyway).

In the next section,

- we study the convergence of SGD; we will make "appear" these challenges and we will discuss leads to mitigate them.
- The step size sequence $\{\alpha_k\}_{k=1}^{\infty}$ will be central, in particular the use of a *diminishing* step size sequence .

Batch approaches

- For the ML community: a *batch* approach is natural and well-known idea,
- The simplest method for this class of methods: the GD (also referred to as batch gradient, or full gradient)
- For empirical risk min, iterations of GD:

$$w_{k+1} \leftarrow w_k - \frac{\alpha_k}{n} \sum_{i=1}^{n} \nabla f_i(w_k) \tag{13}$$

- Even if this more expensive than SGD (roughly *n* times more expensive than SGD): one may expect that a better "step" is obtained by using all the samples...
- **Advantage**: the structure of the empirical risk → benefit from parallelization !

Batch approaches

- Stochastic and batch approaches offer different trade-offs in terms of:
  1. per-iteration costs (**computational costs**),
  2. per-iteration improvements (**rates of convergence**)

  in min. *empirical risk*
- We will look deeper into this trade off in the remaining slides of this section.
- To ease the discussion: we will consider the GD (full gradient) as the **batch** approach.
- Different aspects or motivations will be considered to compare them.
- Moreover, a deeper look into their abilities to guarantee improvement in the underlying *expected risk* R.

GD vs SGD: intuitive motivations

- **Intuitively**: SGD employs information more efficiently
- **Reasons**: given a training set $\mathcal{S}$ which is ten copies of a set $\mathcal{S}_{\text{sub}}$
  1. A minimizer of empirical risk for $\mathcal{S}$ is given by mnimizer for $\mathcal{S}_{\text{sub}}$
  2. Min. R_n over $\mathcal{S}$ with GD: each iteration *10* times more expensive than if one had only one copy of $\mathcal{S}_{\text{sub}}$,
  3. SGD performs the same computations in both scenarios: choosing elements from $\mathcal{S}_{\text{sub}}$ with the same probabilities.
- **Ok... in reality**: training sets usually are not like that...
- **but**: in many large-scale applications, data does involve a good deal of (approximate) redundancy.

  → using all the data is inefficient.

GD vs SGD: theoretical motivations

- One can also cite theoretical arguments for a preference for SGD over a batch approach.
- Let us now give a *preview* of these arguments, which are studied in more depth and further detail later.
- Need to summarize this now before we speak about "practical" motivations.
- The rates of convergence summarized

GD vs SGD: theoretical motivations

For smooth and μ-strongly convex functions

- Fixed step size: $\alpha_k = \alpha$ for all k, small enough:

$$\mathbb{E}[F(w_k) - F^\star] \leqslant C + O(\rho^k)$$

for some constant C and $\rho < 1$ that is a function on the conditioning $\frac{L}{\mu}$ of F, the second moment of $\nabla f_i(w)$ and the choice for step size.

- Diminishing step size: $\alpha_k = \frac{\beta}{\gamma + k}$ for appropriate β and $\gamma > 0$:

$$\mathbb{E}[F(w_k) - F^\star] \leqslant O(\frac{1}{k})$$

For convex and general non-convex functions, we will see later.

GD vs SGD: practical motivation

In (very) brief, for the smooth and μ-strongly convex case:

- GD has linear convergence rate $O((1-\frac{\mu}{L})^k)$, while
- SGD (diminishing step size) has sublinear convergence rate $O(\frac{1}{k})$.

Why choose SGD ? $\rightarrow$ to achieve an ϵ-accuracy error:

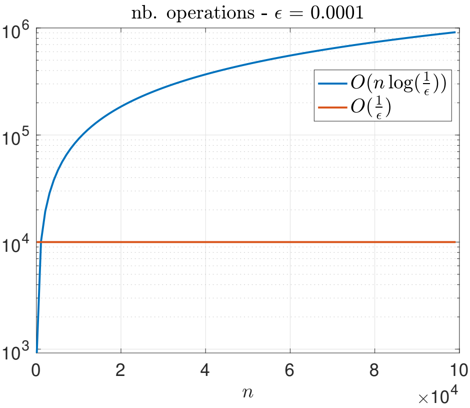
- GD requires $O(\log(\frac{1}{\epsilon}))$ iterations,
- SGD requires $O(\frac{1}{\epsilon})$ iterations.

Hence, for a empirical risk min. problem:

- GD requires $O(n\log(\frac{1}{\epsilon}))$ operations,
- SGD requires $O(\frac{1}{\epsilon})$ operations.

If n is large and ϵ is not too small, SGD is superior to GD!!

GD vs SGD: practical motivation



Example: $n = 10^6$, $\epsilon = 10^{-4}$: $n \log(\frac{1}{\epsilon}) = 4.10^6$ vs $\frac{1}{\epsilon} = 10^4$.

GD vs SGD: showcase

Setup: Given $A \in \mathbb{R}^{d \times n}$, a set of samples $y_i \in \mathbb{R}^d$ with $1 \leqslant i \leqslant n$, we want to solve:

$$\min_{w \in \mathbb{R}^d} F(w) := \frac{1}{n} \sum_{i=1}^{n} \|Aw - y_i\|_2^2$$

with:

- $d = 2$ (to ease visualisations) and $n = 8$,
- $w_i^\star$ the minimzer of each quadratic, and $w^\star$ the global minimizer of $F(w)$.

We compare four methods: GD, AGD, SGD with constant step size and SGD with diminishing step size (SGD-DS).

Demo:

- Code ▸ Link,
- Online - Code ocean ▸ Link

GD vs SGD: showcase results

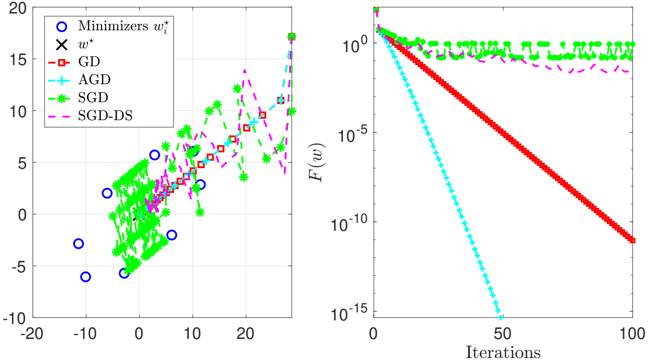


Figure: Benchmark for tackling sum of quadratic forms

Remark: not a fair comparison: SGD and SGD-DS are 8 times cheaper.

GD vs SGD: showcase results

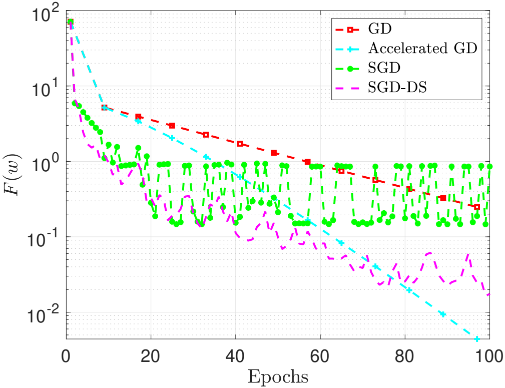


Figure: Benchmark for tackling sum of quadratic forms

Remark: 1 Epoch = 1 gradient computation of a single f_i.

GD vs SGD: showcase results

Observation for SGD: fast initial improvement achieved, followed by a drastic slow down.
Why ?: the notion of *region of confusion*.

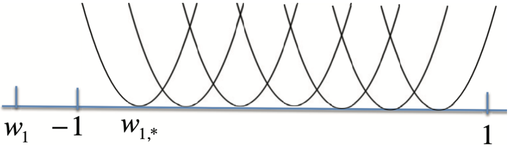


Figure: illustration to motivate the fast initial behavior of the SGD method for min. empirical risk, where each f_i is a convex quadratic.

GD vs SGD: Region of confusion

Insights:

- At $w_1 << -1$: SGD will move right, that is towards $w^\star$,
- As soon as the current iterate is at "leftmost" quadratic: it is *likely* (not certain) that SGD will move to the right,
- As iterates near $w^\star$: SGD enters a **region of confusion**: in which significant chance that a step will not move towards $w^\star$ → progress slow significantly.
- In next section: employing a sequence of diminishing step sizes ensure convergence by overcoming oscillatory behavior of the algo, see previous showcase for an illustration.

Table of Contents

1. General Introduction
2. Formal Optimization Problem Statements
3. First-order methods
4. Stochastic vs Batch Opti. methods
5. Theoretical analysis of SGD
6. Project proposals

Our Menu

- Provide insights into the behavior of an SGD method by establishing :

  1. its convergence properties,
  2. worst-case iteration complexity bounds.

- We gave a *preview* earlier, but now we prove it.
- F[1] is μ-strongly convex function.
  → possible to establish a *global* rate of convergence to the optimal function value $F^\star$.
- **Between dishes**: a small demo on strongly convex case, and notion of mini-batch SGD
- **Main dish**: analyses of SGD for generic non-convex functions.
- **Desserts**: lower-complexity bound and some comments, (F convex function in Part 3).

[1] F can be either the expected risk or the empirical risk, i.e. $F(w) = R(w) = \mathbb{E}[f(w;\xi)]$ or $F(w) = R_n(w) = \frac{1}{n}\sum_{i=1}^{n} f_i(w)$.

How come we deal with both risks ?

Our analyses apply equally to both objectives; the only difference lies in the way that one picks the stochastic gradient estimates in the method:

- **Way 1**: picking samples uniformly from a finite training set, replacing them in the set for each iteration → sampling from a discrete uniform distribution.
  → SGD here optimizes $F(w) = R_n(w)$.
- **Way 2**: picking samples in each iteration according to the distribution P
  → SGD optimizes $F(w) = R(w)$.

Generalized Stochastic Gradient Descent - Algorithm

Initialization

Set $w_1 \in \mathcal{W}$.

Iteration $(k \geqslant 1)$:

1. Generate a realization of the random variable ξ_k
2. Compute a stochastic vector $\nabla f(w_k; \xi_k)$
3. Choose a step size $\alpha_k > 0$
4. Set

$$w_{k+1} \leftarrow w_k - \alpha_k \nabla f(w_k; \xi_k) \tag{14}$$

Three implicit mechanisms:

1. generation of a realization of a random variable ξ_k ($\{\xi_k\}_{k=1}^{\infty}$ represents a sequence of jointly independent random variables)
2. computation of a stochastic vector,
3. computation of the step size α_k

Two fundamental Lemmas

Convergence results for SGD based on assumption of smoothness of the objective function:

Assumption 1

Function F is differentiable and has L-Lipschitz continuous gradients:

$$\|\nabla F(w) - \nabla F(\bar{w})\| \leqslant L\|w - \bar{w}\| \quad \forall w, \bar{w} \in \mathcal{W}(= \mathbb{R}^d) \tag{15}$$

Meaning: gradient of F does not change arbitrarily quickly w.r.t. w
Consequences (not proved):

1. $$F(w) \leqslant F(\bar{w}) + \langle \nabla F(\bar{w}), w - \bar{w} \rangle + \frac{L}{2}\|w - \bar{w}\|^2 \quad \forall w, \bar{w} \in \mathcal{W}(= \mathbb{R}^d) \tag{16}$$
2. If F cont. twice diff., the Hessian matrix satisfies: $\|\nabla^2 F(w)\| \leqslant L$ for all w.

Two fundamental Lemmas - Important notes

1. later we use $\mathbb{E}_{\xi_k}[.]$: the expected value taken w.r.t. the distribution of the random variable ξ_k given w_k.
   Since $w_{k+1} := w_k - \alpha_k \nabla f(w_k; \xi_k)$ (depends on ξ_k)
   → $\mathbb{E}_{\xi_k}[F(w_{k+1})]$ is a meaningful quantity.
2. We will to introduce the notion of *total expectation* to be able to derive rates of convergence as a function of iteration counter k in the theorems later. (will be more clear during the demos)
   Formally: although the variables of the sequence $\{\xi_k\}_{k=1}^{\infty}$ are statistically independent, it is not the case for sequence $\{w_k\}_{k=1}^{\infty}$:
   Example: w_k is completely determined by the realizations of $\{\xi_1, ..., \xi_{k-1}\}$
   → makes sense to use a *total expectation*: an expected value taken w.r.t. to the joint distribution of all random variables, and defined as:

$$\mathbb{E}[F(w_k)] := \mathbb{E}_{\xi_1} ... \mathbb{E}_{\xi_{k-1}}[F(w_k)] \tag{17}$$

Two fundamental Lemmas

Lemma 1

If Assumption 1 is satisfied, the iterates generated by SGD satisfy:

$$\mathbb{E}_{\xi_k}[F(w_{k+1})] - F(w_k) \leqslant -\alpha_k\langle\nabla F(w_k), \mathbb{E}_{\xi_k}[\nabla f(w_k;\xi_k)]\rangle + \frac{\alpha_k^2 L}{2}\mathbb{E}_{\xi_k}[\|\nabla f(w_k;\xi_k)\|^2] \tag{18}$$

Proof: on the board :).

Meaning: Regardless on how SGD arrived in w_k, the expected decrease in the obj. fun. yielded by the k-step is bounded above by:

1. the expected directional derivative of F at w_k along $-\nabla f(w_k;\xi_k)$,
2. the "second-moment" of $\nabla f(w_k;\xi_k)$ (more accurately, the trace of covariance matrix centered at zero).

Q: What happens if $\nabla f(w_k;\xi_k)$ is an unbiased estimator of $\nabla F(w_k)$?

Two fundamental Lemmas

As we will see, SGD is guaranteed to converge as soon as the RHS of (18) is bounded above by a *deterministic* quantity that asymptotically ensures sufficient decrease in F.

Assumption 2

1. $\{w_k\}_{k=1}^{\infty} \in \mathcal{W}$ s.t. set $\mathcal{W}$ is opened and $\min_k(F(w_k)) = F_{inf} > -\infty$ (bounded below).
2. There exists $\mu_G \geqslant \mu > 0$ s.t. for all k:
   - $\langle \nabla F(w_k), \mathbb{E}_{\xi_k}[\nabla f(w_k;\xi_k)] \rangle \geqslant \mu \|\nabla F(w_k)\|^2$ (small angle)
   - $\|\mathbb{E}_{\xi_k}[\nabla f(w_k;\xi_k)]\| \leqslant \mu_G \|\nabla F(w_k)\|$ (bounded norm of stoch. vectors, $\approx$ norm of gradient)
3. There exists $M, M_v \geqslant 0$ s.t.:

$$\begin{aligned} \mathcal{V}_{\xi_k}[\nabla f(w_k;\xi_k)] &= \mathbb{E}_{\xi_k}[\|\nabla f(w_k;\xi_k)\|^2] - \|\mathbb{E}_{\xi_k}[\nabla f(w_k;\xi_k)]\|^2 \\ &\leqslant M + M_v \|\nabla F(w_k)\|^2 \end{aligned} \tag{19}$$

Two fundamental Lemmas

Insights on Assumption 2:

- **Q: What happens if $\nabla f(w_k; \xi_k)$ is an unbiased estimator of $\nabla F(w_k)$?**

  A: Assumption 2 holds directly with $\mu = \mu_G = 1$
- Warning ! : μ is not a strongly convexity parameter here.
- Useful combination: Combining inequalities of Assumption 2, we have:

$$\mathbb{E}_{\xi_k}[\|\nabla f(w_k; \xi_k)\|^2] \leqslant M + M_G \|\nabla F(w_k)\|^2 \tag{20}$$

with $M_G := M_v + \mu_G^2 \geqslant \mu^2 > 0$.

Two fundamental Lemmas

Lemma 2

If Assumptions 1 and 2 are satisfied, the iterates generated by SGD satisfy:

$$\mathbb{E}_{\xi_k}[F(w_{k+1})] - F(w_k) \leqslant -(\mu - \frac{1}{2}\alpha_k L M_G)\alpha_k \|\nabla F(w_k)\|^2 + \frac{\alpha_k^2 L M}{2} \tag{21}$$

Proof: Simply combine Lemma 1 et equation (20).

Insights:

1. RHS of (21) is a pure deterministic quantity,
2. first-term of RHS of (21) decreases $\sim$ norm of the gradient,
3. second-term of RHS of (21) could be large to allow F to increase,
4. **Challenge**: how to balance efficiently those two terms ?
5. Q: is convexity required here ?

SGD for strongly convex functions

Why strongly convex functions ?

1. Strongly convex case is important since minimizer is unique, we may have convergence rates w.r.t. distance of w_k to the minimizer $w^\star$ with $F^\star = F(w^\star)$
2. covers important ML models, such as : "logistic regression" with regularization (tip: convex model $+$ strongly convex reg. $=$ strongly convex model)
3. there exists a variety of situations in which the objective function is not globally (strongly) convex, but is so in the neighborhood of local minimizers, meaning that our results can represent the behavior of the algorithm in such regions of the search space.

SGD for strongly convex functions

Assumption 3

Given $0 < c \leqslant L < \infty$, a real valued differentiable Function F is c-strongly convex function if

$$F(w) \geqslant F(\bar{w}) + \langle \nabla F(\bar{w}), w - \bar{w} \rangle + \frac{c}{2}\|w - \bar{w}\|^2 \quad \forall w, \bar{w} \in \mathcal{W}(= \mathbb{R}^d) \tag{22}$$

Meaning: function not too flat, possible to lower-bound F by a quadratic
Useful inequality:

$$2c(F(w) - F^\star) \leqslant \|\nabla F(w)\|^2 \quad \forall w \in \mathbb{R}^d \tag{23}$$

Tip: compute the minimizer of RHS of (22).

SGD for strongly convex functions - fixed α_k

Fixed step size (but not too large:))

Theorem 2

Under assumptions 1, 2 and 3, suppose SGD is run with a fixed step size $\alpha_k = \alpha$, for all k satisfying:

$$0 < \alpha \leqslant \frac{\mu}{LM_G} \tag{24}$$

Then the expected optimality gap satisfies:

$$\begin{aligned}\mathbb{E}[F(w_k) - F^\star] &\leqslant \frac{\alpha LM}{2c\mu} + (1-\alpha c\mu)^{k-1}(F(w_1) - F^\star - \frac{\alpha LM}{2c\mu}) \\ &\xrightarrow{k\to\infty} \frac{\alpha LM}{2c\mu}\end{aligned} \tag{25}$$

Proof: on the board (start with Lemma 2, then use limit on α, then use (23)).

SGD for strongly convex functions - fixed α_k

Remarks:

- if $\mathbb{E}[\nabla f(w_k;\xi_k)] = \nabla F(w_k)$, then $\mu = \mu_G = 1$. Given $M_G := M_v + \mu_G^2 \geqslant \mu^2$, we may assume $M_v = 0$ and then $M_G = 1$. $\rightarrow \alpha \leqslant \frac{1}{L}$ (a classical step size choice)
- If there is no noise, i.e. $M = 0$, then one could obtain linear conv. rate to the optimal value.
- if $M > 0$: expected objective values conv. lin. to a neighborhood of the opti. value, after the noise in the gradient estimate prevents further progress.
- Select a smaller stepsize worsens the contraction rate ($(1-\alpha c\mu) \nearrow$), but allows to arrive closer to opt. values ($\frac{\alpha L M}{2c\mu} \searrow$)

$\rightarrow$ let us consider varying step sizes (deterministic).

SGD for strongly convex functions - diminishing α_k

Theorem 3

Under assumptions 1, 2 and 3, suppose SGD is run with step sizes α_k, for all k satisfying:

$$\alpha_k = \frac{\beta}{\gamma + k}, \quad \text{for } \beta > \frac{1}{c\mu} \quad \text{and } \gamma > 0 \text{ s.t. } \alpha_1 \leqslant \frac{\mu}{LM_G} \tag{26}$$

Then the expected optimality gap satisfies:

$$\mathbb{E}[F(w_k) - F^\star] \leqslant \frac{\nu}{\gamma + k} \sim O(\frac{1}{k}) \tag{27}$$

where $\nu = \max\{\frac{\beta^2 LM}{2(\beta c\mu - 1)}, (\gamma + 1)(F(w_1) - F^\star)\}$

Proof: on the board.

SGD for strongly convex functions - diminishing α_k

Notes:

1. Role of Strong Convexity: crucial role played by the strong convexity parameter $c > 0$, the positivity of which is needed to argue the contraction of the expected optimality gap.
2. Role of the Initial Point: determines the initial optimality gap: $(F(w_1) - F^\star)$. For diminishing step size SGD, for instance, the gap appears prominently in the second term defining ν.

SGD for strongly convex functions - demo

- **Test case**: Let us consider our first example; "Maximum likelihood estimator for logistic regression" but with a regularization (differentiable)
- For simplicity, we drop variable w_0
- **Data generation**: n data points $\xi_{[i]} = (x_i, y_i)$ where $x_i \in \mathbb{R}^{100}$ (input feature vector) and $y_i \in \{-1, 1\}$ (class label).
  1. x_i independently generated from a Gaussian distribution with zero mean and symmetric covariance, in particular:

     $$P(x_i) = \mathcal{N}(0, 20I) \tag{28}$$

  2. For each x_i, the class label y_i was generated by a logistic regression model, as follows:

     $$P(y_i|x_i; w) = \frac{1}{1 + e^{-y_i\langle w, x_i\rangle}} \tag{29}$$

     where we selected $w = \frac{1}{2}[1, ..., 1]^T \in \mathbb{R}^{100}$

SGD for strongly convex functions - demo

- **Problem**: Pretending not to know the value of w that generated the data, our objective is to fit a regularized logistic regression model to the generated data, i.e. to minimize the following objective function:

$$F(w) = \frac{1}{n}\sum_{i=1}^{n} \log\left(1 + e^{-y_i\langle w, x_i\rangle}\right) + \frac{\lambda}{2}\|w\|^2 \tag{30}$$

  For our numerical experiments: $\lambda = 0.1$.

- **Remark**: it can be shown that $F(w)$ from (30) has Lipschitz-continuous gradients and is λ-strongly convex, therefore it has a unique global minimum. However, no closed-from solution for finding the global minimum exists, therefore one needs to resort to numeric optimization instead.

- ▸ Demo - Colab

SGD for strongly convex functions - demo

Instance: $n = 500$, $\alpha = \frac{1}{L}$ for GD, and $\alpha = \frac{\mu}{LM_G}$ SGD (FS) and $\alpha_k = \frac{\beta}{\gamma+k}$ with $\beta = \frac{1}{c\mu} + 1$ and $\gamma = 500$ for SGD (DS).

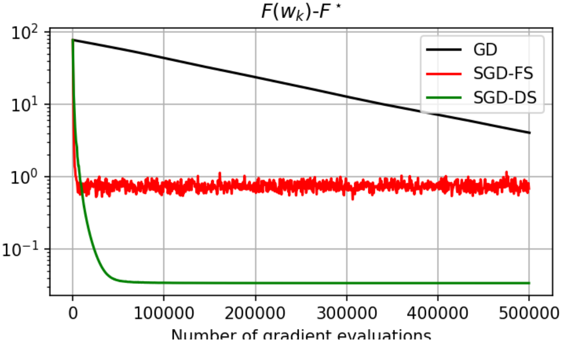


Figure: Benchmark for tackling regularized logistic regression with Maximum likelihood approach (to build $F(w)$)

SGD for strongly convex functions - demo

Instance: $n = 500$, $\alpha = \frac{5}{L}$ for GD, and $\alpha = \frac{5\mu}{LM_G}$ SGD (FS) and $\alpha_k = \frac{\beta}{\gamma+k}$ with $\beta = \frac{1}{c\mu} + 1$ and $\gamma = 500$ for SGD (DS).

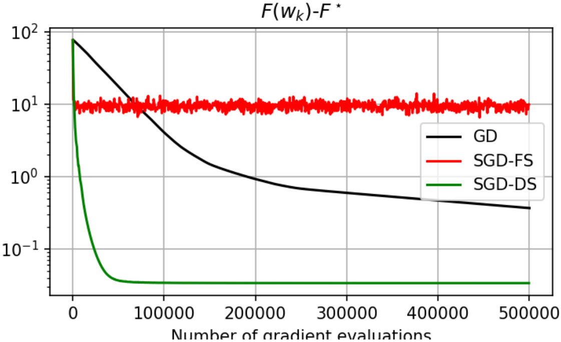


Figure: Benchmark for tackling regularized logistic regression with Maximum likelihood approach (to build $F(w)$)

SGD for strongly convex functions - demo

Instance: $n = 500$, $\alpha = \frac{50}{L}$ for GD, and $\alpha = \frac{\mu}{LM_G}$ SGD (FS) and $\alpha_k = \frac{\beta}{\gamma+k}$ with $\beta = \frac{1}{c\mu} + 1$ and $\gamma = 10$ for SGD (DS).

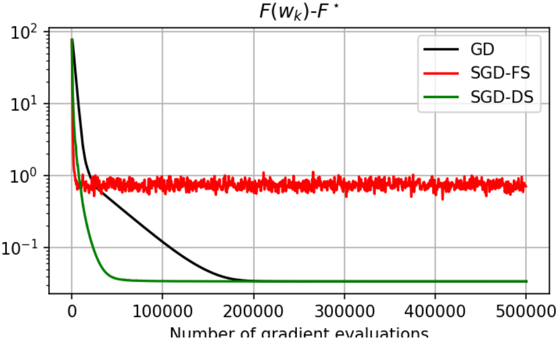


Figure: Benchmark for tackling regularized logistic regression with Maximum likelihood approach (to build $F(w)$)

Mini-Batch Versions of Stochastic Algorithms

Main principle:

- Stochastic algorithms can be easily modified to work on *mini-batches* instead of individual functions.
- a *mini-batch* is a subset of functions f_i of some predetermined fixed size $m < n$.
- For simplicity: consider the case of min. of the *empirical risk* and the algorithm SGD. (rationale can be extended to any stochastic methods !)
- In every stochastic update, instead of choosing a single gradient $\nabla f_{i_k}(w_k)$, minibatch SGD consists of randomly selecting a subset $\mathcal{S}_k$ ($|\mathcal{S}_k| = m$) of the sample indices, the following gradient is used in the update of w_k

$$\nabla_B f(w_k) = \frac{1}{m} \sum_{j \in \mathcal{S}_k} \nabla f_j(w_k) \tag{31}$$

Mini-Batch Versions of Stochastic Algorithms

First remarks:

- if $m = 1$, we go back to the original SGD.
- for $m > 1$: easy to show that $\nabla_B f(w_k)$ is a more reliable estimate of the full gradient that any single gradient.
- **However**: by the time a mini-batch algorithm makes a single update, the original algorithm would have made m updates that, in expectation, move towards the right direction.
- **So**: not obvious whether large mini-batches are advantageous. In fact, (Hinton, 2012) refers to the use of large mini-batches with SGD as "a serious mistake".
- **then why ?**: significant advantage in using mini-batches: computing the m gradients can be easily vectorized or parallelized.

Mini-Batch Versions of Stochastic Algorithms

More theoretical justification: suppose $m \ll n$,

- variance of the direction is reduced by a factor of $\frac{1}{m}$ (more details in ISP - Part 2 lecture).
- **Hence**: constants M and M_v from Assumption 3 are divided by m.
- **Again:** It is natural to ask whether this reduction in the variance pays for the higher per-iteration cost.
- consider a sufficiently small constant stepsize $\alpha > 0$, Theorem 2 for mini-batch SGD leads to:

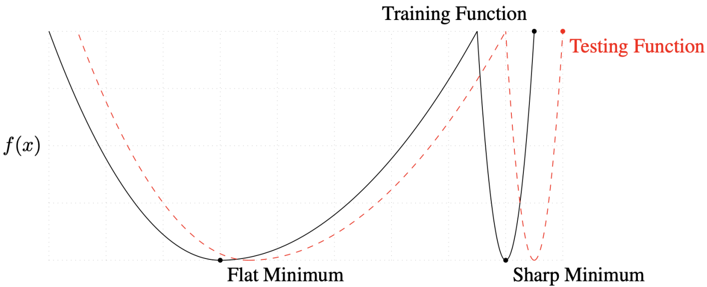
$$\mathbb{E}[F(w_k) - F^\star] \leqslant \frac{\alpha L M}{2c\mu m} + (1 - \alpha c\mu)^{k-1}(F(w_1) - F^\star - \frac{\alpha L M}{2c\mu m})$$

- **But**: using SGD with step size α/m leads to:

$$\mathbb{E}[F(w_k) - F^\star] \leqslant \frac{\alpha L M}{2c\mu m} + (1 - \frac{\alpha c\mu}{m})^{k-1}(F(w_1) - F^\star - \frac{\alpha L M}{2c\mu m})$$

worst contraction rate... in the end, both comparable (except if gpu's used :))

Back to Flat vs Sharp minima



From (Keskar et al., 2017), main observations:

1. large-batch methods tend to converge to sharp minimizers of the training function, tend to generalize less well.
2. small-batch methods converge to flat minimizers

SGD for general objectives

- Many important ML models lead to *nonconvex* optimization problems,
- analyzing SGD in such setting is *much more* challenging, many local minima and other stationary points !
- We present two results as before: for fixed and for diminishing step sizes.

Recall for min. of smooth functions without constraints:

Def 1

A point $w^\star$ is stationary point of F if $\nabla F(w\star) = 0$.

SGD for gen. functions - fixed α_k

Theorem 4 (Proof on the board)

Under assumptions 1 and 2 suppose SGD is run with a fixed step size $\alpha_k = \alpha$, for all k satisfying $0 < \alpha \leqslant \frac{\mu}{LM_G}$. Then the expected sum of squares and averaged-squared gradients of F satisfy for all K iterations done:

$$\mathbb{E}[\sum_{k=1}^{K} \|\nabla F(w_k)\|^2] \leqslant \frac{K\alpha LM}{\mu} + \frac{2(F(w_1) - F_{inf})}{\mu\alpha} \tag{32}$$

and therefore:

$$\begin{aligned}\mathbb{E}[\frac{1}{K}\sum_{k=1}^{K} \| \nabla F(w_k)\|^2] &\leqslant \frac{\alpha LM}{\mu} + \frac{2(F(w_1) - F_{inf})}{\mu\alpha K} \\ &\xrightarrow{K\to\infty} \frac{\alpha LM}{\mu}\end{aligned} \tag{33}$$

SGD for gen. functions - fixed α_k

Insights:

- If $M = 0$, Equation (32) captures a classical result for the full gradient method applied to nonconvex functions:
  the sum of squared gradients remains finite
- **Hence**: series $\sum_{k=1}^{\infty} a_k$ with $a_k = \|\nabla F(w_k)\|$ is convergent, necessarily the sequel $\{\|\nabla F(w_k)\|\}_{k=1}^{\infty}$ converges towards zero.
- Unlike strongly convex case, we cannot bound the expected optimality gap.
- The asymptotic result from Equation (33) illustrates that noise in the gradients ($M > 0$) inhibits further progress.
- The average norm of the gradients can be made arbitrarily small by selecting a small stepsize, but doing so reduces the speed at which the norm of the gradient approaches its limiting distribution.

SGD for gen. functions - diminishing α_k

From now: SGD method is applied to a nonconvex objective with a decreasing sequence of stepsizes satisfying two central conditions:

1. $\sum_{k=1}^{\infty} \alpha_k = \infty$
2. $\sum_{k=1}^{\infty} \alpha_k^2 < \infty$ (finite value)

SGD for gen. functions - diminishing α_k

Theorem 5 (Proof on the board)

Under assumptions 1 and 2 suppose SGD is run with step sizes satisfying the previous conditions. Then, with $A_K := \sum_{k=1}^{K} \alpha_k$,

$$\lim_{K\to\infty} \mathbb{E}[\sum_{k=1}^{K} \alpha_k \|\nabla F(w_k)\|^2] < \infty \tag{34}$$

and therefore:

$$\mathbb{E}[\frac{1}{A_K}\sum_{k=1}^{K} \alpha_k \|\nabla F(w_k)\|^2] \xrightarrow{K\to\infty} 0 \tag{35}$$

SGD for gen. functions - diminishing α_k

Corollary 1

Under assumptions of Theorem 5, we have:

$$\lim_{k\to\infty} \inf \mathbb{E}[\|\nabla F(w_k)\|^2] = 0 \tag{36}$$

Q: Corollary 1 is a direct consequence of Theorem 5, why ?

Corollary 2

Under assumptions of Theorem 5, let $k(K)$ be a random index chosen with probabilities proportional to $\{\alpha_k\}_{k=1}^{K}$. Then,

$$\|\nabla F(w_{k(K)})\| \to 0, \tag{37}$$

in probability as $T \to \infty$

SGD for gen. functions - diminishing α_k

Corollary 3

Under assumptions of Theorem 5 and the assumptions of F is twice differentiable, and the mapping $w \to \|\nabla F(w)\|^2$ has Lipschitz-continuous derivatives, we have:

$$\lim_{k \to \infty} \mathbb{E}[\|\nabla F(w_k)\|^2] = 0 \tag{38}$$

Concluding comments on the SGD analysis

A lower bound:

- when only gradient estimates are available through a noisy oracle has been studied, see (Agarwal et al., 2012)
- **Take-home message** when minimizing a strongly convex function, no algorithm that performs k calls to the oracle can guarantee accuracy better than $O(\frac{1}{k})$
- **Ok..** as we have seen, SGD with decreasing step sizes achieves this lower bound up to constant factors.
- This analysis applies for the optimization of both expected risk and empirical risk.

Concluding comments on the SGD analysis

Alternatives with Faster Convergence.:

- (Agarwal et al., 2012) establish lower complexity bounds for optimization algorithms that only access information about the objective function through noisy estimates of $F(w_k)$ and $\nabla F(w_k)$ at each k iteration.
- The bounds apply, e.g., when SGD is employed to minimize the expected risk R using gradient estimates evaluated on samples drawn from the distribution P.
- **However**: an algorithm that optimizes the empirical risk R_n has access to an additional piece of information: it knows when a gradient estimate is evaluated on a training example that has already been visited during previous iterations.
- **Benefit**: gradient aggregation methods (see Part 2) enjoy *linear* rates.

Table of Contents

1. General Introduction
2. Formal Optimization Problem Statements
3. First-order methods
4. Stochastic vs Batch Opti. methods
5. Theoretical analysis of SGD
6. Project proposals

Proposal 1 - join the competition

In this proposal, you have to solve:

$$\min_{W,b} R_n(W,b) := \frac{1}{n}\sum_{i=1}^{n} \|\Phi(Wx_i + b) - y_i\|_2^2 + g(W)$$

where

- $x_i \in \mathbb{R}^{43586}$ is a document represented by a vector of word counts, and $y_i \in \{0,1\}^r$ is the binary vector corresponding to the class it belongs to, where $1 \leqslant i \leqslant 13960$.
- $\Phi(z)$ is a (component-wise) non-linear function, e.g. $\max(0,z)$ or $\frac{1}{1+e^{-z}}$.
- Join the competition ▸ Link
- This proposal is deeply inspired by (Gillis, 2021).

Proposal 2 - join the competition

In this proposal, you have to solve:

$$\min_{W,b} \frac{1}{n} \sum_{i=1}^{n} \left(\log \left(\sum_{j=1}^{10} e^{[Wx_i+b]^{(j)}} \right) - \sum_{j=1}^{10} y_i^{(j)} e^{[Wx_i+b]^{(j)}} \right) + g(W)$$

where:

- $x_i \in \mathbb{R}^{784}$ is a vectorized gray image of a digit between 0 and 9 from (classes from 1 to 10) ▸ MNSIT database, and $y_i \in \{0,1\}^{10}$ is the binary vector corresponding to the class it belongs to, where $1 \leqslant i \leqslant 60000$.
- $g(W)$ is a regularization function, say $g(W) = \frac{\lambda}{2}\|W\|_2^2$.
- Join the competition ▸ Link

Proposal 3 - (deep) paper presentation

Here, you will have to choose, for instance, one of the following papers and : (1) **prepare** a *detailed* and *comprehensive* presentation, and (2) **implement** one of the algorithms presented on a simple case of interest for you.

- ▸ Paper 1: S. Vaswani et al. *Painless Stochastic Gradient: Interpolation, Line-Search, and Convergence Rates*. 2019
- ▸ Paper 2: P. Richtarik et al. *Stochastic reformulations of linear systems: algorithms and convergence theory*. 2018
- ▸ Paper 3: Zeyuan Allen-Zhu. *Katyusha: The first direct acceleration of stochastic gradient methods*. 2017.
- ▸ Paper 4: Gower et al. *Stochastic quasi-gradient methods: variance reduction via Jacobian sketching*. 2017.

You may also select one paper of your choice, to be approved by the instructor (me :), V.Leplat@skoltech.ru).

References

Boyd, Parikh, Chu, Peleato and Eckstein (2010)
Distributed Optimization and Statistical Learning via the Alternating Method of Multipliers
Machine Learning Vol. 3, No. 1 (2010) 1–122

Y.Nesterov. (2018)
Lectures on Convex Optimization.
Springer.

G. E. Hinton. (2012)
A practical guide to training restricted Boltzmann machines.
Neural Networks: Tricks of the Trade, volume 7700 of Lecture Notes in Computer Science pages 599–619. Springer, 2012.

References

N. Gillis (2021)
First-order methods for large-scale optimization
Lectures UMons - Advanced Opt. course

Bottou, Curtis, and Nocedal (2012)
Optimization Methods for Large-Scale Machine Learning.
SIAM REVIEW Vol. 60, No. 2, pp. 223–311, 2018.

G. Papamakarios (2014)
Comparison of Modern Stochastic Optimization Algorithms.
Technical report, University of Edinburgh

N. Keskar et al. (2017)
On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima.
arXiv:1609.04836

References

A. Agarwal et al. (2012)

Information- theoretic lower bounds on the oracle complexity of stochastic convex optimization

IEEE Trans. Inform. Theory, 58, pp. 3235-3249

That's all Folks!