Concise Lecture Notes on Optimization Methods for Machine Learning and Data Science

These lecture notes are displayed in the form of slides for teaching convenience.

Presentation outline

The problems typically addressed in ML/DS are of the form

$$\min_{x \in \mathbb{R}^n} f(x) + g(x)$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is smooth (∇f is at least Lipschitz continuous) and $g : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is convex (and proper and closed) typically non-smooth.

Examples:

1. Unconstrained optimization ($g = 0$).
2. Structured regularization, where g is a regularizer like the ℓ_1 one ($g(x) = \lambda\|x\|_1$, $\lambda > 0$).
3. Convex constrained smooth optimization, such as

$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & x \in C, \end{array}$$

with $C \neq \emptyset$ closed and convex, can be formulated with $g = \delta_C$ (indicator function of C)

$$\delta_C(x) \; = \; \begin{cases} 0 & \text{if } x \in C \\ \infty & \text{if } x \notin C \end{cases}$$

However only simple constraints are handled well in such a way.

A data set for analysis involving optimization is typically of the form

$$D = \{(a_j, y_j), j = 1, \ldots, N\}$$

where the a_j's vectors are features or attributes

the y_j's vectors are labels or observation or responses.

The analysis consists of finding a prediction function ϕ such that

$$\phi(a_j) \simeq y_j, \quad j = 1, \ldots, N$$

in some optimal sense.

Now

1. The process of finding ϕ is called learning or training.

2. When the y_j's are reals, one has a regression problem.

3. When the y_j's lie in a finite set $\{1, \ldots, M\}$, one has a classification problem.

   $M = 2$ leads to binary classification.

4. The labels may be null. In that case, one may want to group the a_j's in clusters (clusterization) or identify a low-dimensional subspace (or a collection of) where the a_j's lie (subspace identification).

   The labels may have to be learned while learning ϕ.

5. Data is assumed to be clean for optimization, but still:

   i) (a_j, y_j)'s could be noisy or corrupted.
   ii) Some a_j or y_j's could be missing.
   iii) Data could arrive in streaming fashion (ϕ must be learned online).

   Thus, ϕ has to be robust to changes in the data set.

Such data analysis is often referred to as machine learning or data mining.

predictive or supervised learning (when labels exist)

unsupervised learning (when labels are null): extract interesting information from data

6. ϕ is used to make predictions about future data items, like predicting that the label y associated with a would be $\phi(a)$.

   But it could also be used for feature selection where one has to learn what small fraction of the a_j's reliably predict the labels y_j's.

Often, ϕ is parameterized $\phi(x) = \phi(a;x)$ and the parameters x have to be found such that $\phi(a_j;x) \simeq y_j,\ j = 1, \ldots, N$.

The correct/accurate match of the data is typically quantified by a loss function $\ell(a,y;x)$ and thus learning can be formulated as

$$\min_x \sum_{j=1}^{N} \ell(a_j, y_j; x) \qquad \text{call this function } L_D(x) \equiv L(x)$$

To avoid overfitting such a model to D (remember that D could just be a sample drawn from a larger set), one often regularizes the above minimization by adding a regularizer

$$\begin{tikzcd}
& \min_x \sum_{j=1}^{N} \ell(a_j, y_j; x) + g(x) \arrow[dl] \arrow[dr] & L(x) \\
\lambda \|x\|_1 & & \frac{\lambda}{2}\|x\|_2^2
\end{tikzcd}$$

so that ϕ is not so sensitive to changes in D.

Classical examples

Example (1 Least squares and variants)

Consider $\phi(a;x) = \phi(a;\omega,b) = a^\top\omega + b$ with $x = (\omega, b)$

Linear regression is when $L(x) = \sum_{j=1}^{N}(a_j^\top\omega + b - y_j)^2$

Such least-squares approach gives a maximum likelihood solution when $y = (a^\top\omega + b) + \epsilon$ and ϵ is normally distributed with variance σ^2

$$Pr[y \mid a, w, b] \sim N(y \mid a^\top\omega + b, \sigma^2)$$

and the (a_j, y_j)'s and the a_j's are independent.

$$(w, b) \in \operatorname{argmax}_{w,b} \log Pr[D \mid \omega, b]$$

Ridge regression is when

$$L(x) = \frac{1}{N}\sum_{j=1}^{N}(a_j^\top \omega - y_j)^2 + \underbrace{\lambda\|\omega\|^2}_{g(x)}$$

Here $x = \omega$, since the regularizer does not depend on b and w.l.o.g. one can remove b from the model. Why?

There is also a statistical argument for the ridge (reason why L was divided by N).

The term $\lambda\|x\|_2^2$ makes x less sensitive to perturbation in the data.

The LASSO related formulation $L(x) = \sum_{j=1}^{N}(a_j^\top \omega - y_j)^2 + \lambda\|\omega\|_1$ tends to yield solutions x that are sparse, promoting feature selection.
More later in the course.

Example (2 Support vector machines and Logistic regression)

In SVM one does binary classification ($y \in \{-1, 1\}$) by determining a separating hyperplane $\omega^\top a - b$, i.e., by determining (ω, b) such that

$$\begin{cases} \omega^\top a_j - b > 0 & \text{when } y_j = 1 \\ \omega^\top a_j - b \leq 0 & \text{when } y_j = -1 \end{cases} \forall j = 1, \ldots, N$$

<u>**using**</u> the hinge loss function

$$\begin{aligned} \ell_H(a, y; \omega, b) &= \max\{0, 1 - y(\omega^\top a - b)\} \\ &= \begin{cases} 0 & \text{if } y(\omega^\top a - b) \geq 1 \\ 1 - y(\omega^\top a - b) & \text{otherwise} \end{cases} \end{aligned}$$

Example (2 Support vector machines and Logistic regression (Cont.))

In fact, seeking a separating hyperplane $x = (\omega_*, b_*)$ such that

$$\begin{cases} \omega^\top a_j - b \geq 1 & \text{when } y_j = 1 \\ \omega^\top a_j - b \leq -1 & \text{when } y_j = -1 \end{cases} \forall j = 1, \ldots, N \quad \textcircled{S}$$

can be done by

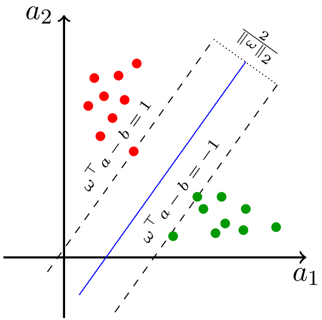
$$\min_{\omega, b} \frac{1}{N} \sum_{j=1}^{N} \ell_H(a_j, y_j; \omega, b) = L(\omega, b) \quad (**)$$

as $L(\omega_*, b_*) = 0$. When no pair (ω, b) exists such that $L(\omega, b) = 0$, the solution of $(**)$ will be the one closest to satisfying $\textcircled{S}$ in some sense.

Note that $\textcircled{S}$ is what we want to do w.l.o.g.. Why?

Example (2 Support vector machines and Logistic regression (Cont.))

A regularizer $\frac{\lambda}{2}\|\omega\|_2^2$ is often added to $L(\omega, b)$ to obtain a maximum-margin separating hyperplane, which is more robust:



Maximizing $2/\|\omega\|_2$ is then the same as minimizing $\|\omega\|_2^2$. EXERCISE: Prove that the distance between the hyperplanes is really $\frac{2}{\|\omega\|_2}$.

In SVM, the hinge loss is a *convex and continuous* replacement for

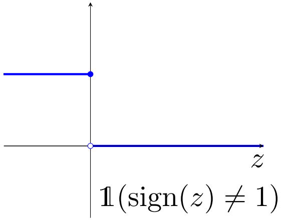
$$\ell(a, y; \omega, b) = \mathbb{1}(h(a; \omega, b) \neq y)$$

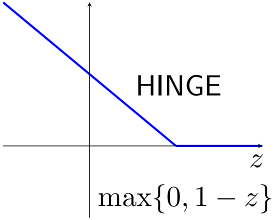
(with $\mathbb{1}(\text{condition}) = 1$ if condition is true and 0 otherwise), where

$$h(a; \omega, b) = \underbrace{2 \times \mathbb{1}(\omega^\top a - b > 0) - 1}_{\text{sign}(\omega^\top a - b)}.$$

which is *nonconvex and discontinuous*.

$\boxed{y = 1}$





In the pictures z plays the role of $\omega^\top a - b$.

There is a statistical interesting interpretation of such optimal linear classifier when using the above loss (as the so-called Bayes function).

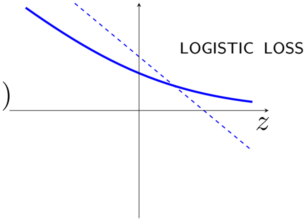
Another replacement is the smooth convex logistic loss

$$\ell_L(a, y; \omega, b) \;=\; \log(1 + e^{-y(\omega^\top a - b)})$$

leading to logistic regression (convex objective function)

$$\min_{\omega, b} \frac{1}{N} \sum_{j=1}^{N} \ell_L(a_j, y_j; \omega, b) + \frac{\lambda}{2} \|\omega\|_2^2$$

$\boxed{y = 1}$ $\quad \log(1 + e^{-z})$



Logistic regression can also be interpreted as maximizing a log -likelihood function of the odds of belonging to one class or the other.

similar to $\textcircled{S}$

Why?

$$\begin{cases} P(a_j;\omega,b) \simeq 1 & \text{when } y_j = +1 \\ \\ P(a_j;\omega,b) \simeq 0 & \text{when } y_j = -1 \end{cases}$$

$$P(a;\omega,b) = \left(1 + e^{-(\omega^\top a - b)}\right)^{-1}$$

Returning to SVM:

Using a hyperplane to separate the $+$ and $-$ cases may lead to a useless classifier.

Often a mapping Ψ is applied first on the data vectors:

$$\Psi(a_j), \quad j = 1, \ldots, N.$$

The system $\textcircled{\small S}$

$$\left\{ \begin{array}{ll} \omega^\top a_j - b \geq 1 & \text{when } y_j = 1 \\ \omega^\top a_j - b \leq -1 & \text{when } y_j = -1 \end{array} \right. \forall j = 1, \ldots, N$$

becomes

$$\left\{ \begin{array}{ll} \omega^\top \Psi(a_j) - b \geq 1 & \text{when } y_j = 1 \\ \omega^\top \Psi(a_j) - b \leq -1 & \text{when } y_j = -1 \end{array} \right. \forall j = 1, \ldots, N$$

and the SVM optimization problem is

$$\min_{\omega, b} \frac{1}{N} \sum_{j=1}^{N} \max\{0, 1 - y_j(\omega^\top \Psi(a_j) - b)\} + \frac{\lambda}{2} \|\omega\|_2^2$$

It can be easily seen that the SVM optimization problem can be written as a convex Quadratic Program (QP):

$$
\begin{aligned}
\min_{\omega,b,c} \quad & \frac{e^\top c}{N} + \frac{\lambda}{2}\|\omega\|_2^2 \\
\text{s.t.} \quad & y_j(\omega^\top \Psi(a_j) - b) + c_j \geq 1, \qquad c_j \geq 0, \qquad j = 1, \ldots, N
\end{aligned}
$$

where

$$
e = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \in \mathbb{R}^N \quad \text{and} \quad c = \begin{pmatrix} c_1 \\ \vdots \\ c_N \end{pmatrix}
$$

NOTE: In turns out that the *dual* of this QP is the following convex QP

$$\begin{array}{ll} \max\limits_{\alpha} & e^\top \alpha - \dfrac{1}{2}\alpha^\top Q \alpha \\ \text{s.t.} & 0 \leq \alpha \leq \dfrac{1}{N} e \\ & y^\top \alpha \ = \ 0 \end{array} \qquad \alpha = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix} \quad y = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}$$

with $Q_{k\ell} = (1/\lambda) y_k y_\ell \Psi(a_k)^\top \Psi(a_\ell)$.

The "kernel trick" consists of replacing $\Psi(a_k)^\top \Psi(a_j)$ by $K(a_k, a_j)$, without requiring the knowledge of Ψ.

A popular choice is the Gaussian kernel

$$K(a_k, a_j) \ = \ e^{-\|a_k - a_j\|^2 / 2\sigma}$$

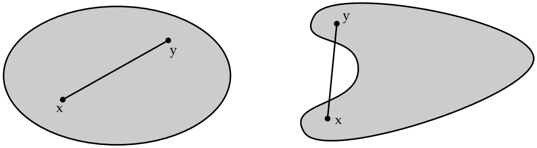
with $\sigma > 0$.

Presentation outline

Convexity is a key concept in Optimization

A set C is convex if

$$\alpha x + (1-\alpha) y \in C, \quad \forall x, y \in C, \alpha \in [0,1]$$



convex set nonconvex set

$\alpha x + (1-\alpha) y, \alpha \in [0,1]$ is a convex combination of x and y.

$\sum_{i=1}^{n} \alpha_i x_i$ with $\sum_{i=1}^{n} \alpha_i = 1$, $\alpha_i \geq 0, \forall i$ is a convex combination of $x_1, \ldots, x_n$.

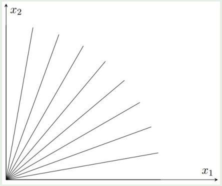
Examples

- $\mathbb{R}^n$
- $\emptyset$ (by convention)
- a subspace
- a polyhedral set $\{x \in \mathbb{R}^n : A_1 x = b_1, A_2 x \geq b_2\}$
  - a hyperplane $\{x \in \mathbb{R}^n : a^\top x = b\}$
  - a halfspace $\{x \in \mathbb{R}^n : a^\top x \geq b\}$
  - a system of linear equations $\{x \in \mathbb{R}^n : Ax = b\}$
  - a polytope (bounded polyhedral set)
- a convex cone: K is a cone if $\alpha x \in K, \forall \alpha > 0, x \in K$

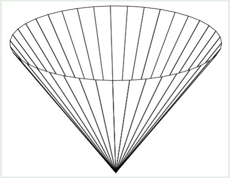
Examples

Here are three important cones in optimization:

1. $\mathbb{R}^n_+$ the cone of linear programming

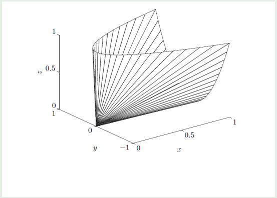


2. $\left\{x \in \mathbb{R} : x_n \geq \sqrt{x_1^2 + \ldots + x_{n-1}^2}\right\}$ the ice cream cone (Lorentz or Minkowski) that appears in second-order cone programming



Examples

3. $\{A \in \mathbb{R}^{n\times n} : A \succeq 0\}$, A is a symmetric and positive semidefinite (eigenvalues ≥ 0), the semidefinite cone programming (SDP) cone



$$\begin{bmatrix} x & y \\ y & z \end{bmatrix}, \quad x, z \geq 0,\ xz \geq y^2$$

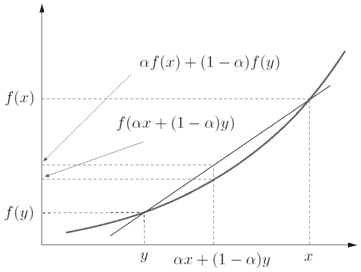
Operations preserving convexity

- Intersection $C_1 \cap C_2$
- Set sum $C_1 + C_2 = \{x_1 + x_2 : x_1 \in C_1, x_2 \in C_2\}$
- Affine transformation $f(C) = \{Ax + b : x \in C\}$ with $f(x) = Ax + b$. Particular cases:
    - scaling
    - translation
    - projection $\{x_1 : \exists x_2 : (x_1, x_2) \in C\}$
- Cartisian product $C_1 \times C_2 = \{(x_1, x_2) : x_1 \in C_1, x_2 \in C_2\}$

The proofs are left as EXERCISES.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if

$$f(\alpha x + (1-\alpha)y) \ \leq \ \alpha f(x) + (1-\alpha) f(y), \forall \alpha \in [0,1], x, y \in \mathbb{R}^n$$



Examples

- All norms: in particular p–norms, $\|x\|_p, 1 \leq p \leq \infty$.
- Affine functions: $f(x) = a^\top x + b$.

A convex function f could be of extended type $f : \mathbb{R}^n \to [-\infty, \infty]$.

An example is the indicator of a (convex) set C (seen before):

$$\delta_C(x) = \begin{cases} 0 & \text{if } x \in C \\ \infty & \text{if } x \notin C \end{cases}$$

Another example is $(n = 1)$

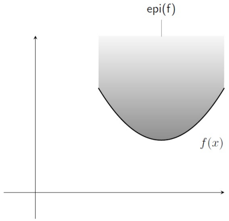
$$f(x) = -\log(x) \text{ if } x > 0, \quad f(x) = \infty \text{ if } x \leq 0.$$

We thus restrict our attention to proper convex functions where $-\infty$ is never attained and $\exists x \in \mathbb{R}^n : f(x) < \infty$ ($\implies \mathrm{dom}(f) \neq \emptyset$).

Also, the definition of a convex function can be restricted to a convex set or to the domain of the function (assumed convex). Hopefully each example will reveal whether we are assuming convexity over a set, the domain, or the whole space.

In many examples it is difficult to apply directly the definition to verify convexity. Hence we use necessary and sufficient conditions:

1. (GEOMETRY) f is convex iff its epigraph $\text{epi}(f) = \{(x, y) : y \geq f(x)\}$ is convex in $\mathbb{R}^{n+1}$



2. (REDUCTION TO THE SCALAR CASE) f is convex iff $f(x + \alpha v)$ is convex $\forall x, v$.

   With this characterization it is easy to verify that

$$f(X) = -\log(\det(X))(= -\sum_{i=1}^{n} \log(\lambda_i(X)))$$

   is convex in the space of matrices $n \times n$ and of domain

$$\{X \in \mathbb{R}^{n\times n} : X = X^\top, \quad \underbrace{\lambda_i(X)}_{i\text{–th eigenvalue of } X} \quad > 0, i = 1, \dots, n\}$$

3. (CONTINUOUS DIFFERENTIABILITY) Let f be continuous differentiable in $\mathbb{R}^n$.

   f is convex iff $(\nabla f(y) - \nabla f(x))^\top (y - x) \geq 0 \ \forall x, y \in \mathbb{R}^n$

4. (TWICE CONT. DIFF.) Let f be twice cont. differentiable in $\mathbb{R}^n$. f is convex iff $d^\top \nabla^2 f(x) d \geq 0, \forall x, d$.

EXERCISE: Adapt 1–4 to the case where f is defined over a convex set C. In 3–4, you have to consider the convex cone of feasible directions: $A_C(x) = \{d \in \mathbb{R}^n : x + \alpha d \in C$ for some $\alpha > 0\}$.

It is easy to see that $f(x) = b^\top x + \frac{1}{2} x^\top A x$, with A symmetric and $\underbrace{\text{positive semi-definite}}_{\text{PSD}}$ (eigenvalues ≥ 0) is convex using 4: $\nabla^2 f(x) = A$ and $d^\top A d \geq 0, \forall d$.

Operations preserving convexity:

- Positive weighted sum $\sum_{i=1}^{p} \alpha_i f_i$, $\alpha_i > 0$, and f_i convex $\forall i$.
- Composition by affine transformation: $f(Ax + b)$.
- Pointwise maximum $\max_{1 \leq i \leq p} f_i(x)$, f_i convex $\forall i$.
- Composition by nondecreasing convex function

  $g(f)$, f, g convex and g nondecreasing
- Minimum over a closed convex set $g(x) = \inf_{y \in C} f(x, y)$

The proofs are left as EXERCISES.

One can now list more examples of convex functions relevant for this course and other data science contexts.

- $\|Ax - b\|_p$ and $\|Ax - b\|_2^2$
- $\sum_{i=1}^{p} e^{g_i(x)}$ with g_i convex
- $-\log(\det(X))$ for X symmetric PD (seen before)
- $-\sum_{i=1}^{p} \log(b_i - a_i^\top x)$
- distance to a set C (convex and closed)

$$d_C(x) \;=\; \min_{y \in C} \|x - y\| \quad \|\cdot\| = \|\cdot\|_2 \text{ by default}$$

- largest singular value of a matrix
- largest eigenvalue of a symmetric matrix

The proofs are left as EXERCISES.

Why is convexity relevant in Optimization?

Let us consider an optimization problem of the form

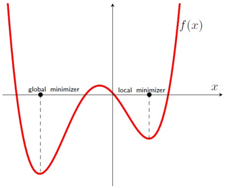
$$\begin{array}{ll} \min & f(x) \\ \text{s.t.} & x \in \Omega \end{array}$$

A point x_* is a local (strict) minimizer if $\exists N$ neighborhood of x_* such that

$$f(x_*) \underset{(<)}{\leq} f(x), \quad \forall x \in (N \cap \Omega) \setminus \{x_*\}$$

x_* is said a global (strict) minimizer if

$$f(x_*) \underset{(<)}{\leq} f(x), \quad \forall x \in \Omega \setminus \{x_*\}$$



Then we have

Theorem

If f is convex over C, then every local minimum is global.

Proof.

If x a local minimizer is not global, $\exists z : f(z) < f(x)$. Then for $\alpha \in (0,1)$:

$$\begin{aligned} f(\alpha z + (1-\alpha)x) &\leq \alpha f(z) + (1-\alpha) f(x) \\ &< f(x) \end{aligned}$$

which when $\alpha \to 0$ contradicts the fact that x is a local minimizer. □

Moreover, if f is strictly convex on C ("$<$" in the definition) and $\exists$ a local minimizer then $\exists$ a unique global minimizer. Why?

The set of minimizers of a convex function is convex. Why?

Convexity or strict convexity does not guarantee the existence of minimizers (take e^x in $\mathbb{R}$).

In the general, possible nonconvex case existence of minimizers is guaranteed by the Weierstrass Theorem: A continuous function has a minimizer (and a maximizer) in a compact set Ω (in $\mathbb{R}^n$ closed and bounded).

One can trade boundedness of Ω by uniform convexity of f in Ω convex.

A function f is uniformly convex (with constant $\mu_f > 0$ called the modulus) if

$$f(\alpha x + (1-\alpha)y) \;\leq\; \alpha f(x) + (1-\alpha) f(y) - \frac{\mu_f}{2}\alpha(1-\alpha)\|x-y\|^2$$

The smooth characterizations of uniform convexity are:

$$\begin{aligned}(\nabla f(y) - \nabla f(x))^\top (y-x) \;&\geq\; \mu_f \|x-y\|^2, \quad \forall x, y\\ d^\top \nabla^2 f(x) d \;&\geq\; \mu_f \|d\|^2, \quad \forall x, d\end{aligned}$$

Hence a quadratic $q(x) = b^\top x + \frac{1}{2} x^\top A x$ with A symmetric and PD is μ_q–uniformly convex with $\mu_q = \lambda_{\min}(A)$.

Uniform convexity can be restricted to a convex set C (work out the details!).

e^x is not uniformly convex for any $\mu > 0$, and it does not attain a minimizer in $\mathbb{R}$ (which is closed and convex).

But $b^\top x + \frac{1}{2}x^\top Ax$ does (with A symmetric and PD).

In fact, one has

Theorem

If f is μ_f–uniformly convex ($\mu_f > 0$) and continuous (NOTE: It is enough to be proper and closed) in C closed and convex, then it has a single minimizer in C.

We will see a proof of a simplified version of this result in a minute. For the moment we remark that:

Theorem

If f is convex, then it is continuous at any point in $\mathrm{int}(\mathrm{dom}(f))$.

Proof.

See any textbook in Convex Analysis (or Beck 2017). $\square$

Many convex functions seen in this course are smooth, meaning at least continuous differentiable (C^1).

We pause to present a key inequality for C^1 functions used a lot in this course. The fundamental theorem of calculus yields

$$f(y) - f(x) \;=\; \int_0^1 \nabla f(x + \xi(y-x))^\top (y-x) d\xi.$$

If in addition ∇f is Lipschitz continuous with constant $L_{\nabla f}$, meaning

$$\|\nabla f(x) - \nabla f(y)\| \;\leq\; L_{\nabla f}\|x-y\|, \quad \forall x, y$$

then

$$
\begin{aligned}
f(y) - f(x) - \underbrace{\nabla f(x)^\top (y-x)}_{\int_0^1 \nabla f(x)^\top (y-x) d\xi} &= \int_0^1 (\nabla f(x + \xi(y-x)) - \nabla f(x))^\top (y-x) d\xi \\
&\le \int_0^1 \|\nabla f(x + \xi(y-x)) - \nabla f(x))\| \|y-x\| d\xi \\
&\le \int_0^1 \xi d\xi \times L_{\nabla f} \|y-x\| \times \|y-x\| \\
&= \frac{L_{\nabla f}}{2} \|y-x\|^2,
\end{aligned}
$$

and

$$
f(x) - f(y) - \nabla f(x)^\top (y-x) \ \le \ \frac{L_{\nabla f}}{2} \|y-x\|^2, \quad \forall x, y
$$

On the other hand if f is convex and C^1 then (Why?)

$$f(y) \;\geq\; f(x) + \nabla f(x)^\top (y - x) \quad \forall x, y$$

In the C^1 case, μ_f–uniform convexity is equivalent to

$$f(y) \;\geq\; f(x) + \nabla f(x)^\top (y - x) + \frac{\mu_f}{2}\|y - x\|^2 \quad \forall x, y$$

In such a case we will say that f is μ_f–strongly convex. The proof of this equivalence is not requested.

The condition number of a smooth strongly convex function f is defined by

$$\kappa_f \;=\; \frac{L_{\nabla f}}{\mu_f},$$

where $L_{\nabla f}$ is the Lips. constant of the gradient and μ_f is the strong convexity constant.

We will finally prove now a simplified version of the existence result (the proof is enough to gain intuition for the general case), that will be used later in the course.

Theorem

Let f be continuous differentiable and μ_f–strongly convex. Then f has a unique minimizer x_*.

Proof.

The idea is to first show that the level set

$$L(x_0) \ = \ \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$$

is closed and bounded (compact) for any x_0. The closeness comes from the continuity of f. Suppose it is not bounded.

Proof (Cont.)

Then there exist a sequence $\{x_l\}$ such that

$$\|x_l\| \to \infty \quad \text{and} \quad f(x_l) \leq f(x_0).$$

From strong convexity

$$f(x_l) \;\geq\; f(x_0) + \nabla f(x_0)^\top (x_l - x_0) + \frac{\mu_f}{2}\|x_l - x_0\|^2$$

$$\Downarrow$$

$$\begin{aligned} \frac{\mu_f}{2}\|x_l - x_0\|^2 \;&\leq\; f(x_l) - f(x_0) - \nabla f(x_0)^\top (x_l - x_0) \\ &\leq\; \|\nabla f(x_0)\| \|(x_l - x_0)\| \qquad \text{since } f(x_l) \leq f(x_0) \end{aligned}$$

$$\Downarrow$$

$$\|x_l - x_0\| \;\leq\; \frac{2}{\mu_f}\|\nabla f(x_0)\|$$

Proof (Cont.)

and this contradicts the unboundedness of $\{x_l\}$.

By the Weierstrass Theorem, f has a minimizer, say x_*, in $L(x_0)$. It remains to show that is unique.

Suppose that there are two distinct minimizers x_*^1 and x_*^2 (with the same objective function value f_*). Then a contradiction is easily reached.

$$\begin{aligned} f\left(\frac{1}{2}x_*^1 + \frac{1}{2}x_*^2\right) &\leq \frac{1}{2}f(x_*^1) + \frac{1}{2}f(x_*^2) - \underbrace{\frac{\mu_f}{8}\|x_*^1 - x_*^2\|}_{\neq 0} \\ &< f_* \end{aligned}$$

□

A number of convex functions in this course are nonsmooth, and it is time now to define tools to deal with nondifferentiability.

Let f a be convex function, possibly of value extended to $[-\infty, +\infty]$.

The vector v is a subgradient of f at x if

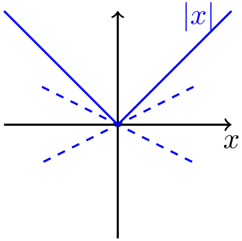
$$f(x+d) \;\geq\; f(x) + v^\top d, \quad \forall d \in \mathbb{R}^n.$$

The set of all subgradients is called subdifferential

$$\partial f(x) \;=\; \{v \in \mathbb{R}^n : v \text{ is subgradient of } f \text{ at } x\}$$

Let us see two examples:

$f(x) = |x|$. In this case $\partial f(0) = [-1, 1]$



$\delta_C(x) = \begin{cases} 0 & \text{if } x \in C \\ \infty & \text{if } x \notin C \end{cases}$. In this case, for any $x \in C$, $v \in \partial \delta_C(x)$ iff

$$\delta_C(z) \;\geq\; \delta_C(x) + v^\top (z - x) \quad \forall z \in C$$

$$\Updownarrow$$

$$v^\top (z - x) \;\leq\; 0 \quad \forall z \in C$$

$$\Updownarrow$$

$$v \in N_C(x)$$

where N_C is the cone normal to C at x.

What are the features of the subdifferential?

Theorem

If f is convex and proper then $\partial f(x)$ is closed and convex for all $x \in \mathrm{dom}(f)$.

Proof.

A simple consequence of $\partial f(x)$ being the intersection of half-spaces (which are closed and convex). □

Theorem

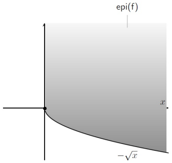
If f is convex and proper then $\underbrace{\partial f(x) \neq \emptyset}_{f \text{ is subdifferentiable at } x}$ *and $\partial f(x)$ is bounded for all $x \in \mathrm{int}(\mathrm{dom}(f))$.*

Proof.

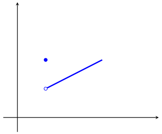
See Beck 2017. □

To better understand the last result, consider $f(x) = -\sqrt{x}$, $\mathrm{dom}(f) = \mathbb{R}_0^+$. Note that f is also closed in the sense that $\mathrm{epi}(f)$ is closed.

However, $\partial f(0)$ is the empty set!



Also, convex functions are not necessarily continuous at boundary points of their domains, as we see from



and this can even happen when f is closed (but for $n > 1$).

The subdifferential characterizes optimality for convex functions:

Theorem

x_* is a (global) minimum of f convex iff $0 \in \partial f(x_*)$

Proof.

If x_* is a minimizer,

$$f(x_* + d) \geq f(x_*) \geq f(x_*) + 0^\top d \quad \forall d,$$

showing that

$$0 \in \partial f(x).$$

If $0 \in \partial f(x_*)$,

$$f(x_* + d) \geq f(x_*) \quad \forall d.$$

□

Calculus rules for ∂f (for simplicity all convex functions are assumed real value with domain $\mathbb{R}^n$; proofs are left as EXERCISES).

Continuous differentiability $\partial f(x) = \{\nabla f(x)\}$

Positive weighted sum $\partial(\sum_{i=1}^{p} \alpha_i f_i)(x) = \sum_{i=1}^{p} \alpha_i \partial f_i(x)$

Composition by affine transformation $g = f(Ax + b)$

$$\partial g(x) \; = \; A^\top \partial f(Ax + b)$$

Pointwise maximum $g(x) = \max_{1 \le i \le p} f_i(x)$

$$\partial g(x) = \underbrace{\text{conv}}_{\text{convex hull}} \bigcup_{i \in I(x)} \partial f_i(x)$$

where $I(x) = \{i : f_i(x) = g(x)\}$. Hence, in a weak sense, any element of $\partial f_i(x)$, $i \in I(x)$, is in $\partial g(x)$.

Composition by nondecreasing convex function

$$\begin{tikzcd} & h \; = \; g & (f) \\ \text{convex nondecreasing} \arrow[from=1-2] & & \text{convex} \arrow[from=1-3] \end{tikzcd}$$

Let us assume that $g : \mathbb{R} \to \mathbb{R}$ is continuous differentiable. Then

$$\partial h(x) \; = \; g'(f(x)) \partial f(x)$$

Distance to a closed convex set $d_C(x) = \min_{y \in C} \|x - y\|$

$$\partial d_C(x) \;=\; \left\{ \frac{x - P_C(x)}{\underbrace{d_C(x)}_{=\|x - P_C(x)\|}} \right\} \quad \text{for} \quad x \notin C$$

where $P_C(x)$ is the orthogonal projection of x onto C.

If $x \in C$, $\partial d_C(x) = \underbrace{N_C(x)}_{\text{normal cone}} \cap \underbrace{B(0;1)}_{\{v : \|v\| \le 1\}}$.

In particular, one has $0 \in \partial d_C(x)$.

Examples

- $\partial f(x) = \frac{1}{2}\|Ax-b\|_2^2 + \lambda\|x\|_1$

$$\begin{aligned}\partial f(x) &= A^\top Ax - A^\top b + \lambda\partial_{\|\cdot\|}(x)\\ &= A^\top Ax - A^\top b + \lambda\left\{\sum_{x_i\neq 0}\mathrm{sign}(x_i)e_i + \sum_{x_i=0}[-e_i, e_i]\right\}\end{aligned}$$

  with $I = [e_1 \dots e_n]$.

- $f(x) \ = \ \sum_{i=1}^p |a_i^\top x - b_i|$

$$\partial f(x) \ = \ \sum_{i=1}^p \partial_{|\cdot|}(a_i^\top x - b_i)a_i$$

- $f(x) = \|x\|_2$

$$\partial f(x) = \begin{cases} x/\|x\|_2, & x \neq 0\\ B(0;1), & x = 0\end{cases}$$

NOTE: $\nabla f(x_*) = 0$ when f is nonconvex and continuous differentiable is still a necessary condition, but not longer sufficient.

We end this background chapter with some notions of rates of convergent sequences. We will be interested in knowing the speed of convergence of sequences such as

$$\begin{array}{ll} f(x_k) - f(x_*) & \text{optimality gap (when } x_* \text{ is a minimizer)} \\ \left.\begin{array}{c} \|\nabla f(x_k)\| \\ d_{\partial f(x_k)}(0) \end{array}\right\} & \begin{array}{l} \text{stationary or criticality (smooth and} \\ \text{nonsmooth case, respectively)} \end{array} \\ \|x_k - x_*\| & \text{absolute error in the iterates (when } x_k \to x_*) \end{array}$$

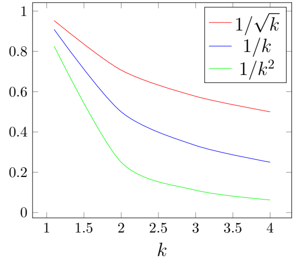
Let $\{\omega_k\} \subset \mathbb{R}^n$ be a sequence converging to ω_*. There are four major types of rates of convergence of interest to us

- SUBLINEAR $\lim_{k\to\infty} \frac{\|\omega_{k+1}-\omega_*\|}{\|\omega_k-\omega_*\|} = 1$

  When $n = 1$ and $\omega_* = 0$, at least three examples will be seen in this course:

$$\frac{1}{\sqrt{k}}, \quad \frac{1}{k}, \quad \frac{1}{k^2}$$

  Sublinear is a slow rate but $\frac{1}{k^2}$ is much faster than $\frac{1}{\sqrt{k}}$.



- LINEAR (also known as geometric or exponential convergence):
  $\exists r \in (0,1) \quad \|\omega_{k+1} - \omega_*\| \leq r\|\omega_k - \omega_*\| \quad \forall k$

  When $n = 1$ and $\omega_* = 0$, an example is $(\frac{1}{2})^k$.

  First-order methods (like the gradient or steepest descent method) exhibit sublinear or linear rates. Second-order methods achieve a sublinear rate (quasi-Newton) or a quadratic rate (Newton).

- SUPERLINEAR $\exists \{\eta_k\}_{\eta_k \to 0} \quad \|\omega_{k+1} - \omega_*\| \leq \eta_k \|\omega_k - \omega_*\| \quad \forall k$

  The example for $n = 1$ and $\omega_* = 0$ is $\frac{1}{k!}$

- QUADRATIC $\exists M > 0, \|\omega_{k+1} - \omega_*\| \leq M\|\omega_k - \omega_*\|^2 \quad \forall k$

  Take $10^{(1/2)^k}$ as the example when $n = 1$ and $\omega_* = 0$.

Quadratic $\Longrightarrow$ Superlinear $\Longrightarrow$ Linear $\Longrightarrow$ Sublinear   Why?

Somehow we have presented a local version of these rates since (by assuming that $\omega_k \to \omega_*$) we supposed that ω_0 is sufficiently close to ω_*.

These rates are called global when no assumption is made about ω_0.

Also the version presented is what is known as the "q–rates".

See, from $\omega_{k+1} \leq \frac{1}{2}\omega_k$ $(\omega_* = 0, \omega_k > 0 \ \forall k)$ one has

$$\omega_k \ \leq \ \left(\frac{1}{2}\right)^k \omega_0.$$

However not all sequences satisfying this rate are q–linear.

A trivial example is

$$\omega_k = \begin{cases} (\frac{1}{2})^k & \text{when } k \text{ is even} \\ 0 & \text{when } k \text{ is odd} \end{cases}$$

Such $\{\omega_k\}$ converges r–linearly to 0.

In general a sequence $\{\omega_k\} \subset \mathbb{R}^n, \omega \to \omega_*$, has a r–linear rate if $\|\omega_k - \omega_*\|$ is bounded by a sequence in $\mathbb{R}$ that converges q–linearly to zero.

Presentation outline

For the minimization of a smooth (cont. diff.) function f in $\mathbb{R}^n$, the steepest descent or gradient method is

$$x_{k+1} \;=\; x_k - \alpha_k \nabla f(x_k)$$

where $\alpha_k > 0$ is the step size.

The negative gradient $-\nabla f(x)$ is a descent direction

$$\downarrow$$

$$d \neq 0 : f'(x;d) \;<\; 0$$

$$\Downarrow$$

$$\exists \bar{\epsilon} > 0 : f(x + \epsilon d) \;<\; f(x), \quad \forall \epsilon \in (0, \bar{\epsilon}]$$

In fact

$$f'(x; -\nabla f(x)) \;=\; \nabla f(x)^\top (-\nabla f(x)) \;=\; -\|\nabla f(x)\|^2 \;<\; 0$$

The exact line search strategy consists of choosing

$$\alpha_k \;=\; \operatorname{argmin}_{\alpha>0} f(x_k - \alpha\nabla f(x_k)).$$

Other strategies will be covered in the next chapter.

When f is not differentiable, $-\nabla f(x_k)$ may not exist. We thus assume that f is convex and $\mathrm{int}(\mathrm{dom}(f)) = \mathbb{R}^n$, and consider a generalization called the subgradient method

$$x_{k+1} \;=\; x_k - \alpha_k g_k, \quad g_k \in \partial f(x_k),$$

where the subgradient g_k is in the subdifferential $\partial f(x_k)$.

Importantly, $-g_k$ might not be a descent direction!

Example

$f(x_1, x_2) = |x_1| + 2|x_2|$

$$\partial f(1,0) \; = \; \{(1,x) : |x| \leq 2\}$$

$$(1,2) \in \partial f(1,0)$$

$$-(1,2) \text{ is not descent : } f'((1,0);-(1,2)) \; = \; g'_+(0) \; = \; 3 > 0$$

with

$$g(\alpha) = f((1,0) - \alpha(1,2)) \; = \; |1-\alpha| + 4\alpha \; = \begin{cases} 1 + 3\alpha, & \alpha \in [0,1] \\ 5\alpha - 1, & \alpha \geq 1 \end{cases}$$

Also, the gradient method may fail for a nonsmooth convex function (Wolf's example): using exact line searches $\{x_k\}$ is such that $\nabla f(x_k)$ exists, $f(x_{k+1}) < f(x_k) \forall k$ but $x_k \to x_*$ such that x_* is non-optimal. Wolf's example is:

$$f(x_1, x_2) = \begin{cases} \sqrt{x_1^2 + x_2^2}, & |x_2| < x_1 \\ \frac{x_1 + \gamma |x_2|}{\sqrt{1+\gamma}}, & \text{otherwise} \end{cases}$$

with $\gamma > 1$. Because the price of generalization is minor, we consider instead the projected subgradient method for

$$\min f(x) \text{ s.t. } \quad x \in C \qquad \text{where } C \neq \emptyset \text{ is convex and closed}$$

given by

$$x_{k+1} = P_C(x_k - \alpha_k g_k), \ g_k \in \partial f(x_k).$$

Notes:

1. f is subdifferentiable over C ($\partial f(x) \neq \emptyset, \forall x \in C$) as it is convex and $C \subseteq \mathrm{int}(\mathrm{dom}(f))$ (see Chapter 2).
2. The orthogonal projection over C is Lips. continuous with constant 1 (nonexpensive): $\|P_C(x) - P_C(y)\| \leq \|x - y\|, \forall x, y$. Why?
3. If $g_k = 0$ for some k, then x_k is a minimizer and $x_i = x_k \forall i \geq k$.

A fundamental inequality for projected subgradient is

$$\boxed{\|x_{k+1} - x_*\|^2 \ \leq \ \|x_k - x_*\|^2 - 2\alpha_k(f(x_k) - f_*) + \alpha_k^2\|g_k\|^2,}$$

$\forall x_* \in X_*$ (the set of minimizers of f in C assumed $\neq \emptyset$, which is necessarily closed). Why?

f_* is the optimal value.

Proof.

$$\begin{aligned} \|x_{k+1} - x_*\|^2 &= \|P_C(x_k - \alpha_k g_k) - P_C(x_*)\|^2 \\ &\leq \|x_k - \alpha_k g_k - x_*\|^2 \\ &= \|x_k - x_*\|^2 - 2\alpha_k g_k^\top (x_k - x_*) + \alpha_k^2 \|g_k\|^2 \\ &\overset{\text{subgradient inequality}}{\leq} \|x_k - x_*\|^2 - 2\alpha_k (f(x_k) - f_*) + \alpha_k^2 \|g_k\|^2 \end{aligned}$$

□

A natural choice for α_k is the minimizer of the RHS of the fundamental inequality for $\alpha \geq 0$

$$\alpha_k \ = \ \frac{f(x_k) - f_*}{\|g_k\|^2}$$

This results in the Polyak's stepsize rule

$$\alpha_k = \begin{cases} \frac{f(x_k) - f_*}{\|g_k\|^2} & \text{if } g_k \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

One will assume that

$$\boxed{\|g\| \;\leq\; L_{\partial f} \quad \forall g \in \partial f(x) \; \forall x \in C}$$

which actually implies that

$$|f(x) - f(y)| \;\leq\; L_{\partial f} \|x - y\| \quad \forall x, y \in C$$

Why?

Rate of convergence of Polyak's stepsize rule:

$$\boxed{f_{\mathsf{best}}^k - f_* \;\leq\; \frac{L_{\partial f} d_{X_*}(x_0)}{\sqrt{k+1}}, \quad \forall k \geq 0}$$

where $f_{\mathsf{best}}^k = \min_{0 \leq i \leq k} f(x_i)$ (and $d_{X_*}(x_0) < \infty$ since X_* is closed)

Proof.

Plugging the stepsize rule in the fundamental inequality:

$$\begin{aligned}\|x_{k+1} - x_*\|^2 &\leq \|x_k - x_*\|^2 - \frac{(f(x_k) - f_*)^2}{\|g_k\|^2} \\ &\leq \|x_k - x_*\|^2 - \frac{(f(x_k) - f_*)^2}{L_{\partial f}^2}\end{aligned}$$

Thus

$$\begin{aligned}\frac{1}{L_{\partial f}^2} \sum_{i=0}^{k} (f(x_i) - f_*)^2 &\leq \|x_0 - x_*\| - \|x_{k+1} - x_*\|^2 \\ &\leq \|x_0 - x_*\|^2 \\ &\leq d_{X_*}^2(x_0)\end{aligned}$$

and

$$(k+1)(f_{\text{best}}^k - f_*)^2 \leq L_{\partial f}^2 d_{X_*}^2(x_0).$$

Consequences of the proof:

- $\|x_{k+1} - x_*\| \leq \|x_k - x_*\|, \quad \forall k,\ \forall x_* \in X_*$
- $\lim_{k\to\infty} f(x_k) = f_*$

Property 1) is Fejér monotonicity of $\{x_k\}$ w.r.t X_* which actually implies that $\{x_k\}$ does converge to a point in X_*. EXERCISE (see Beck 2017)

The worst case complexity (WCC) is $\mathcal{O}(\epsilon^{-2})$ in the sense that $\mathcal{O}(\epsilon^{-2})$ iterations are required to obtain a x_k such that $f^k_{\text{best}} - f_* \leq \epsilon$. Why?

The limit $f(x_k) \to f_*$ holds for choices of α_k such that $(\sum_{i=0}^k \alpha_i^2)/(\sum_{i=0}^k \alpha_i) \xrightarrow[k\to\infty]{} 0$. EXERCISE (see Beck 2017). An example is $\alpha_k = \frac{1}{\sqrt{k+1}}$.
Variations of this also achieve the $1/\sqrt{k}$ rate (see Beck 2017).

EXERCISE

Apply the subgradient method to $f(x_1, x_2) = |x_1 + 2x_2| + |3x_1 + 4x_2|$.

Applications of the subgradient method

Convex feasiblity problem

Finding x in: $S = \bigcap_{i=1}^{m} S_i$ when S_i is convex and closed $\forall i$

$$\iff \min_{x \in \mathbb{R}^n} f(x) \text{ where } f(x) = \max_{1 \le i \le m} d_{S_i}(x)$$

$$\text{in which case } f_* = 0 \text{ and } X_* = S$$

Given that the S_i's are closed and convex such f is Lipschitz continuous with constant $L_f = 1$ (or nonexpensive)

$$|f(x) - f(y)| \le \|x - y\| \quad \forall x, y$$

Proof.

EXERCISE (see Beck 2017) □

Now, all we need is a weak subgradient computation, meaning the computation of a $g \in \partial f(x)$, not a strong computation (all $\partial f(x)$).

How to compute a g_k in $\partial f(x_k)$ for the f above?

If $x_k \in S$, then $g_k = 0$ and $x_{k+1} = x_k$.

If not, compute $i_k \in \operatorname{argmax}_{1 \leq i \leq m} d_{S_i}(x_k)$

$$g_k \;=\; \frac{x_k - P_{S_{i_k}}(x_k)}{d_{S_{i_k}}(x_k)} \quad \text{(see Chapter 2)}$$

Then

$$\begin{aligned} x_{k+1} \quad &= \quad x_k - \alpha_k g_k \\ &\overset{\text{Polyak}}{=} \quad x_k - \left(\frac{d_{S_{i_k}}(x_k) - f_*}{\|g_k\|^2} \right) \frac{x_k - P_{S_{i_k}}(x_k)}{d_{S_{i_k}}(x_k)} \\ &\overset{f_*=0, \|g_k\|=1}{=} \quad P_{S_{i_k}}(x_k) \end{aligned}$$

One has the greedy projection algorithm:

$$x_{k+1} \;=\; P_{S_{i_k}}(x_k)$$

where $i_k \in \operatorname{argmax}_{1 \leq i \leq m} d_{S_i}(x_k)$ for which the rate of convergence is (of the order) of $1/\sqrt{k}$. Why?

When $m = 2$ one has the alternating projection method

$$x_{k+1} \;=\; P_{S_2}\left(P_{S_1}(x_k)\right).$$

Solution of linear feasibility problems

EXERCISE: State the alternating projection method when

$$S_1 \;=\; \{x \in \mathbb{R}^n : Ax = b\} \quad S_2 \;=\; \{x \in \mathbb{R}^n : x \geq 0\}$$

and then, alternatively, the greedy one when

$$\begin{aligned} S_i \;&=\; \{x \in \mathbb{R}^n : a_i^\top x = b_i, i = 1, \ldots, m\}, (a_i \text{ is the } i\text{-th row of } A) \\ S_{m+1} \;&=\; \{x \in \mathbb{R}^n : x \geq 0\}. \end{aligned}$$

Implement both for $A = \begin{pmatrix} 0 & 6 & -7 & 1 \\ -1 & 2 & 10 & -1 \end{pmatrix}$, $b = \begin{pmatrix} 0 \\ 10 \end{pmatrix}$, and plot $f(x_k)$, for $k = 1, \ldots, 20$, in both cases.

In which cases would you then consider the alternating one? ...

Matrix Completion (more later in the course)

Find a positive semidefinite matrix $M \in \mathbb{R}^{n\times n}$ such that $M_{ij} = A_{ij}, (i,j) \in \Omega$ (for a certain given set of points Ω).

The problem can be formulated as the intersection of two convex sets.

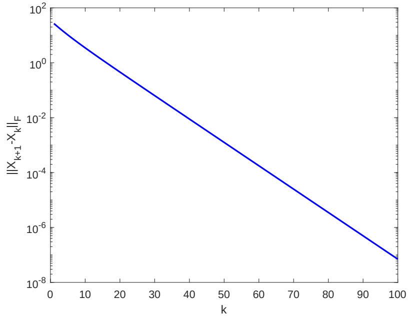
Projections (preferably orthogonal) must be defined over the two sets.

For measuring distance one can use the Frobenius norm

$$\|M\|_F \;=\; \sqrt{\operatorname{tr}(M^\top M)}.$$

Formulate and implement an algorithm for finding such matrix.

$A \in \mathbb{R}^{100\times 100}$ is a PSD matrix, where 20% of its entries are missing. We use the alternating projection method:



So, we see that the distance between Ω and the SDP cone goes to zero. It looks like that it is going to zero at a linear rate, faster than predicted by the theory on the optimality gap... In fact the rate is known to be linear for the alternating projection method when the two sets are closed and convex (Gubin, Polyak, and Raik 1967, Bauschke and Borwein 1993).

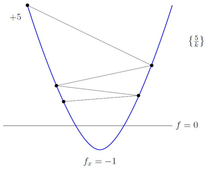
Presentation outline

As we have seen before the gradient method, for continuous differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, is defined by

$$x_{k+1} \ = \ x_k - \alpha_k \nabla f(x_k)$$

where $\alpha_k > 0$ is the step size.

A choice of α_k that only ensures a simple decrease on f might not guarantee convergence to a stationary point



Thus, one has to ensure some form of sufficient decrease

$$f(x_k - \alpha_k \nabla f(x_k)) \ \leq \ f(x_k) - c\alpha_k \|\nabla f(x_k)\|_2^2$$

with $c \in (0, 1)$.

Note that when $x_{k+1} = x_k - \alpha_k p_k$ with p_k a descent direction $(f'(x_k; p_k) = -\nabla f(x_k)^\top p_k < 0)$ sufficient decrease reads like

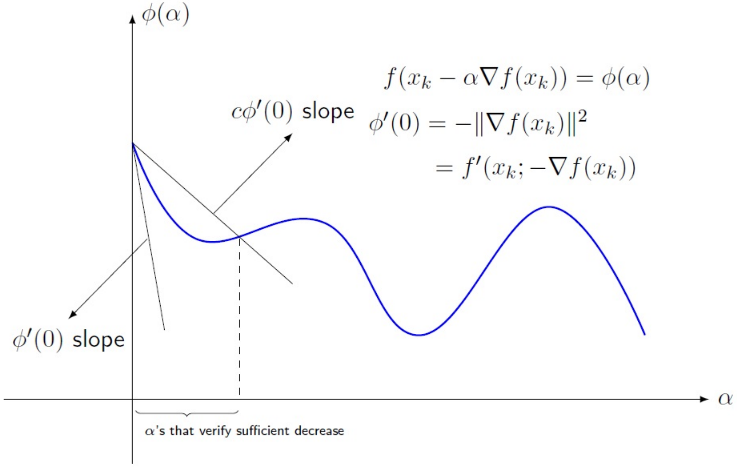
$$f(x_k - \alpha_k p_k) \;\leq\; f(x_k) - c\alpha_k \nabla f(x_k)^\top p_k$$

Such a sufficient decrease condition is typically imposed in Newton or quasi–Newton type methods.

Sufficient decrease guaranteed by a backtracking procedure:

```
Choose $c, \beta \in (0,1)$ and $s > 0$. Set $\bar{\alpha} = s$.
**while** $f(x_k - \bar{\alpha}\nabla f(x_k)) \;>\; f(x_k) - c\bar{\alpha}\|\nabla f(x_k)\|^2$ **do**
    $\bar{\alpha} := \beta \times \bar{\alpha}$
**end while**
$\alpha_k := \bar{\alpha}$
```

As long as f is bounded below, and thus bounded below in $\{x_k - \alpha\nabla f(x_k) : \alpha \geq 0\}$, this procedure may end in a finite number of steps, recalling, of course, that $-\nabla f(x_k)$ is a descent direction:


$\phi(\alpha)$
$f(x_k - \alpha\nabla f(x_k)) = \phi(\alpha)$
$c\phi'(0)$ slope
$\phi'(0) = -\|\nabla f(x_k)\|^2$
$= f'(x_k; -\nabla f(x_k))$
$\phi'(0)$ slope
α
α's that verify sufficient decrease

Let us assume now that ∇f is Lipschitz continuous with constant $L_{\nabla f} > 0$. Note that f may be nonconvex.

As we know from Chapter 2,

$$f(\overbrace{x_k - \alpha\nabla f(x_k)}^{y_k}) - f(x_k) \;\leq\; \nabla f(x_k)(\underbrace{-\alpha\nabla f(x_k)}_{y_k - x_k}) + \frac{L_{\nabla f}}{2}\|\underbrace{-\alpha\nabla f(x_k)}_{y_k - x_k}\|^2$$

giving rise to

$$f(x_k) - f(x_k - \alpha\nabla f(x_k)) \;\geq\; \alpha\left(1 - \frac{\alpha}{2}L_{\nabla f}\right)\|\nabla f(x_k)\|^2.$$

Besides, if $\beta\bar{\alpha}$ does not satisfy sufficient decrease

$$f(x_k) - f(x_k - (\beta\bar{\alpha})\nabla f(x_k)) \;<\; c(\beta\bar{\alpha})\|\nabla f(x_k)\|^2,$$

and this inequality together with the previous one with $\alpha = \beta\bar{\alpha}$ yield

$$\beta\bar{\alpha}\left(1 - \frac{\beta\bar{\alpha}}{2} L_{\nabla f}\right) \; < \; c(\beta\bar{\alpha})$$

thus

$$\bar{\alpha} \; > \; \frac{2(1-c)}{\beta L_{\nabla f}}$$

Hence

$$\boxed{f(x_k) - f(\underbrace{x_{k+1}}_{x_k - \alpha_k \nabla f(x_k)}) \; \geq \; M\|\nabla f(x_k)\|^2}$$

with $M = c \min\left\{s, \frac{2(1-c)}{\beta L_{\nabla f}}\right\}$.

Such an inequality is the key to analyze complexity and convergence for the gradient method with step size satisfying sufficient decrease.

In fact, summing this inequality from 0 to $k-1$ and noting the telescoping sum

$$f(x_0) - f(x_k) \;\geq\; M \sum_{i=0}^{k-1} \|\nabla f(x_i)\|^2$$

and, assuming a lower bound f_{low} on f, we reach the rate of convergence for such method

$$\boxed{\min_{0 \leq i \leq k-1} \|\nabla f(x_i)\| \;\leq\; \sqrt{\frac{f(x_0) - f_{low}}{M}} \frac{1}{\sqrt{k}}, \quad \forall k \geq 0}$$

as in the subgradient method for convex functions.

As a consequence of this proof the series $\sum_{i=0}^{\infty} \|\nabla f(x_i)\|^2$ is summable and therefore the gradient goes to zero

$$\lim_{k \to \infty} \nabla f(x_k) \;=\; 0$$

Moreover, the WCC is $\mathcal{O}(\epsilon^{-2})$ in the sense that $\mathcal{O}(\epsilon^{-2})$ iterations are required to obtain a x_* such that $\|\nabla f(x_*)\| \leq \epsilon$.

The sublinear rate $1/\sqrt{k}$ of the gradient method (with sufficient decrease) is, of course, slow and in the nonconvex case other line search methods based on Newton or quasi-Newton directions are much faster (quadratic or superlinear rates respectively) but require second-order information.

First-order methods (such as the gradient method) find room for application in problems where second-order information is prohibited such as those handling a large amount of data per iteration.

In those situations the function f is typically convex, an assumption made for the rest of this chapter.

The convex case will be treated in the more general scenario (see Chapter 1) where the problem is

$$\min_{x \in \mathbb{R}^n} F(x) \equiv f(x) + g(x)$$

still covering smooth unconstrained optimization ($g = 0$), but then addressing structured regularization ($g(x) = \lambda \|x\|_1$ for instance) and simple convex constraints ($g(x) = \delta_C$, where $C \neq \emptyset$ is closed and convex).

In addition, to later deal efficiently with the inclusion of g, we will cover gradient methods in their proximal variant.

And because of the features of most optimization data problems requiring regularization, proximal gradient methods will be analyzed only when f is convex or strongly convex.

It is simple to see that the gradient method can be expressed as

$$x_{k+1} \; = \; \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \left\{ f(x_k) + \nabla f(x_k)^\top (x - x_k) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}$$

in other words, as the minimizer of the sum of the linearization of f around x_k with a quadratic proximal term.

So, when dealing with the minimization of $F = f + g$, it is natural to consider

$$x_{k+1} \; = \; \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \left\{ f(x_k) + \nabla f(x_k)^\top (x - x_k) + g(x) + \frac{1}{2\alpha_k} \|x - x_k\|^2 \right\}$$

Note that

$$\begin{aligned} & \frac{1}{2} \|x - (x_k - \alpha_k \nabla f(x_k))\|^2 \\ = \; & \frac{1}{2} \|x - x_k\|^2 + (x - x_k)^\top \left(\alpha_k \nabla f(x_k) \right) + (\text{constant in } x) \end{aligned}$$

Hence, the proximal subproblem (after multiplying its objective by α_k) can be written as

$$x_{k+1} \; = \; \operatorname*{argmin}_{x \in \mathbb{R}^n} \left\{ \alpha_k g(x) + \frac{1}{2} \| x - (x_k - \alpha_k \nabla f(x_k)) \|^2 \right\}$$

We now introduce the proximal-operator (prox-operator):

$$\text{prox}_{\alpha g}(x) \; = \; \text{argmin}_{u \in \mathbb{R}^n} \left\{ \alpha g(u) + \frac{1}{2} \| u - x \|^2 \right\}$$

(for fixed α and x). The proximal gradient method can thus be written as

$$\boxed{x_{k+1} \; = \; \text{prox}_{\alpha_k g}(x_k - \alpha_k \nabla f(x_k))}$$

where $\alpha_k > 0$ is the stepsize.

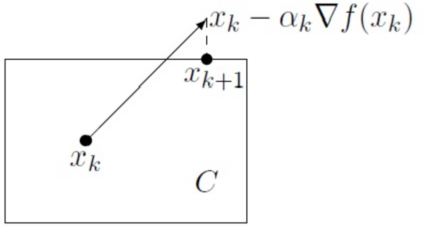
We now pause to cover examples and properties of the prox-operator. Let us start by three important examples:

1. $g(x) = 0, \forall x$. In this case, $\mathrm{prox}_{\alpha g}(x) = x$. This shows of course that the proximal gradient method reduces to the gradient one when there is no regularization.
2. $g(x) = \delta_C(x)$ with C closed and convex. Here

$$\begin{aligned} \mathrm{prox}_{\alpha g}(x) &= \operatorname*{argmin}_{u \in \mathbb{R}^n} \left\{ \alpha \delta_C(u) + \frac{1}{2}\|u - x\|^2 \right\} \\ &= \operatorname*{argmin}_{u \in C} \left\{ \frac{1}{2}\|u - x\|^2 \right\} \\ &= P_C(x) \end{aligned}$$

which is simply the projection of x onto C.

The proximal gradient method is then the projected gradient one.

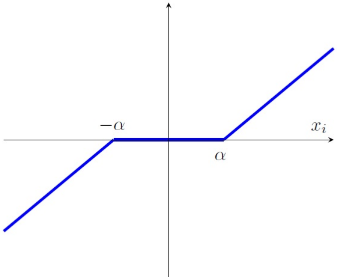


3. $h(x) = \alpha\|x\|_1$. One can then see that the minimization in the definition of the prox-operator separates in its n components, being the i-th one

$$\left(\text{prox}_{\alpha\|\cdot\|}(x)\right)_i \; = \; \text{argmin}_{u_i} \left\{ \alpha|u_i| + \frac{1}{2}(u_i - x_i)^2 \right\}$$

and thus

$$\left(\mathrm{prox}_{\alpha\|\cdot\|}(x)\right)_i = \begin{cases} x_i - \alpha & \text{if } x_i \geq \alpha \\ 0 & \text{if } x_i \in (-\alpha, \alpha) \\ x_i + \alpha & \text{if } x_i \leq \alpha \end{cases}$$

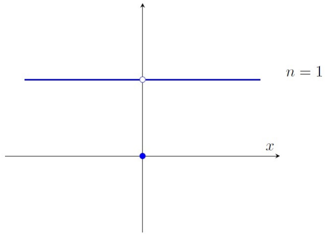


This is called the soft-thresholding operation.

A fourth example is hard-thresholding,

$$g(x) = \|x\|_0 = |\{i \in \{1, \ldots, n\} : x_i \neq 0\}|,$$

the number of nonzero components of x. Although g is not convex,



the prox-operator is well defined and separates into n components:

$$\left(\text{prox}_{\alpha\|\cdot\|}(x)\right)_i = \begin{cases} x_i & \text{if } |x_i| \geq \sqrt{2\alpha}, \\ 0 & \text{otherwise,} \end{cases} \qquad \text{Why?}$$

$i = 1, \ldots, n$.

EXERCISE: Using $\operatorname{prox}_h(x) = \operatorname*{argmin}_{u \in \mathbb{R}^n} \left\{ h(u) + \frac{1}{2}\|u - x\|^2 \right\}$ calculate the prox-operator when

$$h(x) \;=\; a^\top x + b \quad \text{(affine)}$$

$$h(x) \;=\; \|x\|_2$$

$$h(x) \;=\; a + b^\top x + \frac{1}{2} x^\top A x \quad A \text{ sym. and PSD (convex quadratic)}$$

There are calculus rules for prox-operators, the most relevant being for composition with affine functions and with the Euclidean norm.

Regarding the matrix properties of the prox-operator, the very first one is that is well defined as long as h in

$$\operatorname{prox}_h(x) \;=\; \underset{u\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ h(u) + \frac{1}{2}\|u - x\|^2 \right\}$$

is a proper convex function (see Chapter 2). Thus we assume that g in $F = f + g$ is proper and convex.

Moreover, from the first-order conditions, one has

$$0 \in \partial h(\operatorname{prox}_h(x)) + \operatorname{prox}_h(x) - x \quad (*)$$

$$\iff \; x - \operatorname{prox}_h(x) \in \partial h(\operatorname{prox}_h(x))$$

Note also (not used later) that the prox-operator is nonexpensive (Lipschitz continuous with constant 1):

$$\| \operatorname{proj}_h(x) - \operatorname{proj}_h(y)\| \ \leq \ \|x - y\|$$

The proof is left as an EXERCISE. Hint:

a) First use $(*)$ for x and y.

b) Then use the fact that the subdifferential of h is monotone:

$$a \in \partial h(u), b \in \partial h(v) \Longrightarrow (a-b)^\top (u-v) \ \geq \ 0$$

c) Then rearrange and apply the Cauchy-Schwartz inequality.

We will analyze the proximal gradient method for $F = f + g$ when f is smooth (∇f Lipschitz continuous with constant $L_{\nabla f}$). Then

$$f(y) \ \leq \ f(x) + \nabla f(x)^\top (y - x) + \frac{L_{\nabla f}}{2} \|y - x\|^2, \quad \forall x, y. \qquad (B1)$$

As we said before, given the presence of g, the cases of interest are when f is convex or strongly convex. Then

$$f(y) \ \geq \ f(x) + \nabla f(x)^\top (y - x) + \frac{\mu_f}{2} \|y - x\|_2^2, \quad \forall x, y. \qquad (B2)$$

($\mu_f = 0$ convex; $\mu_f > 0$ strongly convex).

It will be very convenient to write the method as

$$\begin{aligned} x_{k+1} &= \operatorname{prox}_{\alpha_k g}(x_k - \alpha_k \nabla f(x_k)) \\ &= x_k - \alpha_k G_{\alpha_k}(x_k) \end{aligned}$$

$G_\alpha(x)$ is called the gradient mapping but it is not a gradient or a subgradient of $F = f + g$.

Moreover from $(*)$ and omitting the subscript k

$$(x - \alpha \nabla f(x)) - (x - \alpha G_\alpha(x)) \in \partial(\alpha g)\,(x - \alpha G_\alpha(x))$$

$$\Downarrow$$

$$\boxed{G_\alpha(x) - \nabla f(x) \in \partial g(x - \alpha G_\alpha(x)) \qquad (**)}$$

(Remark: From here one has that $G_\alpha(x) = 0 \iff x$ minimizes $F = f + g$.)

Our stepsize rule will be simply $\alpha = \frac{1}{L_{\nabla f}}$ but the derivation to come holds with $\alpha \in (0, \frac{1}{L_{\nabla f}}]$. From $(B1)$

$$f(y) \ \leq \ f(x) + \nabla f(x)^\top (y - x) + \frac{L_{\nabla f}}{2} \|y - x\|^2, \quad \forall x, y,$$

with $y = x - \alpha G_\alpha(x)$, one has then

$$f(x - \alpha G_\alpha(x)) \ \leq \ f(x) - \alpha \nabla f(x)^\top G_\alpha(x) + \frac{\alpha}{2} \|G_\alpha(x)\|^2 \quad (\mathsf{B1}_\alpha)$$

(Here we used $\alpha \leq 1/L_{\nabla f}$.)

We are now ready to prove a key inequality measuring the decrease along $G_\alpha(x)$. First we add $g(x - \alpha G_\alpha(x))$ to both sides of (B1$_\alpha$)

$$F(x-\alpha G_\alpha(x)) \;\le\; f(x)-\alpha \nabla f(x)^\top G_\alpha(x)+\frac{\alpha}{2}\|G_\alpha(x)\|^2+g(x-\alpha G_\alpha(x))$$

For any z, one has from $(B2)$

$$f(y) \;\ge\; f(x)+\nabla f(x)^\top (y-x)+\frac{\mu_f}{2}\|y-x\|^2$$

with y replaced by z

$$f(z) \;\ge\; f(x)+\nabla f(x)^\top (z-x)+\frac{\mu_f}{2}\|z-x\|^2$$

and using it above

$$\begin{aligned}
F(x - \alpha G_\alpha(x)) \;\le\; & f(z) + g(z) - \nabla f(x)^\top (z - x) - \frac{\mu_f}{2}\|z - x\|^2 \\
& - \alpha \nabla f(x)^\top G_\alpha(x) + \frac{\alpha}{2}\|G_\alpha(x)\|^2 + g(x - \alpha G_\alpha(x)) - g(z) \\
= \; & F(z) - \nabla f(x)^\top (z - (x - \alpha G_\alpha(x))) \\
& + G_\alpha(x)^\top (z - (x - \alpha G_\alpha(x))) + G_\alpha(x)^\top (x - z) \\
& - \alpha \|G_\alpha(x)\|^2 - \frac{\mu_f}{2}\|z - x\|^2 + \frac{\alpha}{2}\|G_\alpha(x)\|^2 \\
& + g(x - \alpha G_\alpha(x)) - g(z)
\end{aligned}$$

From $(**)$

$$g(x - \alpha G_\alpha(x)) - g(z) \;\le\; (G_\alpha(x) - \nabla f(x))^\top (x - \alpha G_\alpha(x) - z).$$

Hence the desired inequality

$$\boxed{\begin{gathered} F(x-\alpha G_\alpha(x)) \ \leq \ F(z)+G_\alpha(x)^\top(x-z)-\frac{\alpha}{2}\|G_\alpha(x)\|^2-\frac{\mu_f}{2}\|z-x\|^2 \\ (***) \end{gathered}}$$

from which we can now extract rates of convergence for the convex ($\mu_f = 0$) and strongly convex ($\mu_f > 0$) cases.

First, we point out that setting $z = x$ shows us that the method is indeed descent

$$F(x - \alpha G_\alpha(x)) \ \leq \ F(x) - \frac{\alpha}{2}\|G_\alpha(x)\|^2.$$

Setting $z = x_*$ (a global minimizer of F)

$$\begin{aligned} F(x - \alpha G_\alpha(x)) - F_* &\le G_\alpha(x)^\top (x - x_*) - \frac{\alpha}{2}\|G_\alpha(x)\|^2 - \frac{\mu_f}{2}\|x - x_*\|^2 \\ &= \frac{1}{2\alpha}\left(\|x - x_*\|^2 - \|x - x_* - \alpha G_\alpha(x)\|^2\right) \\ &\quad - \frac{\mu_f}{2}\|x - x_*\|^2 \\ &= \frac{1}{2\alpha}\left((1 - \mu_f \alpha)\|x - x_*\|^2 - \|(x - \alpha G_\alpha(x)) - x_*\|^2\right) \end{aligned}$$

and using again the indices k

$$\boxed{F(x_{k+1}) - F_* \le \frac{1}{2\alpha_k}\left((1 - \mu_f \alpha_k)\|x_k - x_*\|^2 - \|x_{k+1} - x_*\|^2\right) \quad (B3)}$$

We can now derive a global rate.

In the convex case ($\mu_f = 0$), summing from 0 to k (with telescoping cancellation), and using $\alpha_k = 1/L_{\nabla f}$,

$$\begin{aligned} \sum_{i=1}^{k}(F(x_i) - F_*) &\leq \frac{L_{\nabla f}}{2} \sum_{i=1}^{k} \left(\|x_{i-1} - x_*\|^2 - \|x_i - x_*\|^2\right) \\ &= \frac{L_{\nabla f}}{2} \left(\|x_0 - x_*\|^2 - \|x_k - x_*\|^2\right) \\ &\leq \frac{L_{\nabla f}}{2} \|x_0 - x_*\|^2 \end{aligned}$$

and because $F(x_i)$ is nondecreasing we arrive finally at the rate of convergence of the proximal gradient method (f convex in $F = f + g$)

$$\boxed{F(x_k) - F_* \leq \left(\frac{L_{\nabla f}}{2}\|x_0 - x_*\|^2\right)\frac{1}{k} \qquad \forall k \geq 0.}$$

This sublinear rate $(1/k)$ is better than the previous ones $(1/\sqrt{k})$ found before for the subgradient method (for minimizing f convex) and for the gradient method (for minimizing f nonconvex).

The WCC bound to reach $F(x_k) - F_* \leq \epsilon$ is then also better: $\mathcal{O}(\epsilon^{-1})$.

This applies also to $g = 0$, i.e., to the gradient method when f is convex.

In the strongly convex case ($\mu_f > 0$), $(B3)$ gives also ($F(x_{k+1}) \geq F_*$ and $\alpha = \frac{1}{L_{\nabla f}}$)

$$0 \;\leq\; \frac{L_{\nabla f}}{2}\left((1-\mu_f/L_{\nabla f})\|x_k - x_*\|^2 - \|x_{k+1} - x_*\|^2\right)$$

$$\Updownarrow$$

$$\|x_{k+1} - x_*\|^2 \;\leq\; \left(1 - \frac{\mu_f}{L_{\nabla f}}\right)\|x_k - x_*\|^2$$

from which we conclude right away that $\|x_{k+1} - x_*\| \leq \|x_k - x_*\|$, i.e., the distance to the optimal set $X_* = \{x_*\}$ does not increase.

Moreover, we obtain the following rate of convergence for the proximal gradient method (f strongly convex in $F = f + g$)

$$\boxed{\|x_k - x_*\| \;\leq\; \left(\underbrace{1 - \frac{\mu_f}{L_{\nabla f}}}_{<1}\right)^k \|x_0 - x_*\|^2 \qquad \forall k.}$$

This rate is now linear and translates in a WCC bound of $\mathcal{O}(-\log(\epsilon))$ to reach $\|x_k - x_*\| \leq \epsilon$, more precisely

$$\mathcal{O}\left(\kappa_f(-\log(\epsilon))\right) \text{ where } \kappa_f = \frac{L_{\nabla f}}{\mu_f} \text{ is the condition number of } f$$

One can show that $F(x_k) - F_*$ also decreases linearly. Why?

Again this applies to $g = 0$, i.e., to the gradient method when f is strongly convex.

NOTES

1. When $f = 0$, the proximal gradient method for a fixed step size is called the proximal point method, which is not practical as it requires the minimization of g itself at each iteration.

2. An alternative to the stepsize rule $\alpha_k = 1/L_{\nabla f}$ when $L_{\nabla f}$ is unknown or hard to estimate is to do a backtracking scheme or a sufficient decrease condition.

   When $g = 0$ this can be done as in the nonconvex case.

   In the general case $g \neq 0$, one possibility is to impose

$$f(T_\alpha(x_k)) \ \leq \ f(x_k) + \nabla f(x_k)^\top (T_\alpha(x_k) - x_k) + \frac{1}{2\alpha} \|T_\alpha(x_k) - x_k\|^2$$

   with $T_\alpha(x_k) = \operatorname{prox}_{\alpha g}(x_k - \alpha \nabla f(x_k))$.

Note that this condition is equivalent to $(\mathsf{B}1_\alpha)$:

$$f(T_\alpha(x_k)) \ \leq \ f(x_k) - \alpha \nabla f(x_k)^\top G_\alpha(x_k) + \frac{\alpha}{2}\|G_\alpha(x_k)\|^2.$$

The backtracking starts at $\bar{\alpha} = s$ and stops at the first $i \in \{0, 1, \ldots\}$ such that $\beta^i \bar{\alpha}$ ($\beta \in (0,1)$) satisfies the above condition.

Since such condition is satisfied for $\alpha = 1/L_{\nabla f}$ it is easy to show that α_k is bounded from below, and the rates of convergence given before for the convex and strongly convex cases are still valid.

3. The proximal gradient method can also be applied, of course, to the case where f in $F = f + g$ is nonconvex.

   The obtained rate of convergence (now for the gradient mapping $\min_{0\leq i\leq k} \|G_{\alpha_i}(x_i)\|$) is $1/\sqrt{k}$, as expected.

   However, not only such an algorithm setting finds less application, but the analysis is lengthier, as one has to prove much more from the prox-gradient mapping as for instance that is monotone:

$$\|G_{\alpha'}(x)\| \;\geq\; \|G_{\alpha''}(x)\| \qquad \forall 0 < \alpha' \leq \alpha''.$$

4. When $g = \lambda\|\cdot\|$ with $\lambda > 0$ the proximal gradient method is known as ISTA ($\underbrace{\text{iterative shrinkage-thresholding algorithm}}_{\text{more on this in the next chapter}}$).

EXERCISE

Apply the proximal gradient method to solve

1. Convex QPs with simple bounds

$$\begin{aligned} \min \quad & b^\top x + \frac{1}{2} x^\top A x \\ \text{s.t.} \quad & 0 \leq x \leq e \; = \; \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}. \end{aligned}$$

2. LASSO or ℓ_1–regularized least squares

$$\min \|Ax - b\|^2 + \lambda \|x\|_1 \quad (\lambda > 0),$$

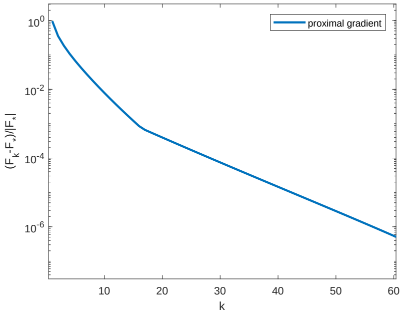
   for which the method is then ISTA.

In both cases, use $\alpha_k = 1/L_{\nabla f}$, after calculating $L_{\nabla f}$.

Plot $(F(x_k) - F_*)/|F_*|$ over the iteration counter k.

Let $A \in \mathbb{R}^{3000\times 3000}$ and $b \in \mathbb{R}^{3000}$ be randomly generated (A PD).

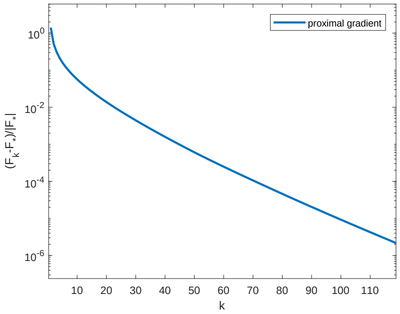
The performance of proximal gradient (PG) or ISTA method for solving the above-mentioned QP problem with $L_{\nabla f} = 1/\lambda_{\max}(A^\top A)$ is the following:



The rate looks sublinear (although the theory says linear).

Let $A \in \mathbb{R}^{5000\times 3000}$ and $b \in \mathbb{R}^{5000}$ be randomly generated.

The performance of proximal gradient (PG) or ISTA method for solving the above-mentioned LASSO problem with $\lambda = 1$ and $L_{\nabla f} = 1/\lambda_{\max}(A^\top A)$ is the following:



The rate looks sublinear (although the theory says linear).

Presentation outline

Using momentum and still one gradient evaluation per iteration, it is possible to derive methods that converge at a rate of $1/k^2$ in the convex case (thus faster than $1/k$).

The momentum idea is to continue to use the direction of the previous step $x_k - x_{k-1}$ (as it brings previous gradient information) combining it linearly with the current gradient $\nabla f(x_k)$.

If for a moment we consider $g = 0$ in $F = f + g$, such an idea is expressed as

$$x_{k+1} \;=\; x_k - \alpha_k \nabla f(x_k) + \underbrace{\beta_k (x_k - x_{k-1})}_{\text{momentum term}}.$$

By applying this formula recursively, one can see that x_{k+1} is written in terms of $\nabla f(x_0), \ldots, \nabla f(x_k)$ in a richer, more parameterized way, when compared to the case $\beta_k = 0$ (gradient method).

When f is a strongly quadratic function and α_k and β_k are held fixed in a certain way, such approach was first proposed by Polyak 1964 (the Heavy Ball method), yielding a better constant in the WCC bound when compared to the gradient method: $\sqrt{\frac{L_{\nabla f}}{\mu_f}}$ instead of $\frac{L_{\nabla f}}{\mu_f}$.

The CG (conjugate gradients) method to solve $Ax = b$, where A is symmetric and PD, invented even before by Hestenes and Stiefel in 1952, makes also use of momentum.

Nonlinear conjugate gradients (extension from strongly convex quadratics to general smooth functions) also make use of momentum.

A key contribution was then made by Nesterov 1983 who introduced an accelerated gradient method for the minimization of a smooth function f with a $1/k^2$ convergence rate.

Each iteration is slightly more complex

$$x_{k+1} \;=\; x_k - \alpha_k \nabla f(x_k + \underbrace{\beta_k(x_k - x_{k-1})}_{}) + \underbrace{\beta_k(x_k - x_{k-1})}_{}$$

momentum in two places

One can rewrite this scheme as (with y_k auxiliary)

$$\begin{aligned} x_{k+1} &= y_k - \alpha_k \nabla f(y_k) \\ y_{k+1} &= x_{k+1} + \beta_{k+1}(x_{k+1} - x_k), \quad \forall k \end{aligned}$$

with x_0 arbitrary and $y_0 = x_0$.

His accelerated gradient (AG) method is when

$$\beta_{k+1} = \frac{t_k - 1}{t_k + 1} \quad \text{with } t_0 = 1 \text{ and } t_{k+1} = \frac{1}{2}(1 + \sqrt{1 + 4t_k^2}).$$

FISTA ("fast ISTA") is the generalization to the composite case $F = f + g$, proposed by Beck and Teboulle 2009 essentially making use of the prox-operator:

$$\begin{aligned} x_{k+1} &= \operatorname{prox}_{\alpha_k g}(y_k - \alpha_k \nabla f(y_k)) \\ t_{k+1} &= \frac{1}{2}(1 + \sqrt{1 + 4t_k^2}) \qquad\qquad \text{FISTA} \\ y_{k+1} &= x_{k+1} + \frac{t_k - 1}{t_k + 1}(x_{k+1} - x_k) \end{aligned}$$

where $\alpha_k > 0$ is the stepsize and $t_0 = 1$, $y_0 = x_0$.

Important facts regarding the sequence $\{t_k\}$:

1. $t_k \geq \frac{k+2}{2}, \quad \forall k \geq 0$ (proof by induction)
2. $t_{k+1}^2 - t_{k+1} = t_k^2, \quad \forall k \geq 0$

NOTE: The rate of convergence holds when $t_{k+1}^2 - t_{k+1} \leq t_k^2$.

Since $t_k = \frac{k+2}{2}$ satisfies this last inequality, AG and FISTA are sometimes presented in this simplified form.

Let us now prove the global rate.

Let us start by recall our assumptions on $F = f + g$: f is a (proper) convex function with Lipschitz continuous gradient (with constant $L_{\nabla f}$) and g is a (proper) closed convex function; F has a minimum x_*.

This time we need a more general form of $(***)$

$$F(T_\alpha(y)) - F(x) \ \leq \ \frac{1}{2\alpha}\left(\|x-y\|^2 - \|x - T_\alpha(y)\|^2\right) \quad (4*)$$

where $T_\alpha(y) = \mathrm{prox}_{\alpha g}(y - \alpha \nabla f(y))$
This is essentially the fundamental prox-grad inequality (Theorem 10.16 of Beck 2017 when f is convex).

We omit the proof in part because $(4*)$ is essentially a generalization of $(***)$. This latter one is ($z = x$, $\mu_f = 0$)

$$\begin{aligned} F(\underbrace{x - \alpha G_\alpha(x)}_{T_\alpha(x)}) - F(x) \ &\leq \ G_\alpha(x)^\top (x-y) - \frac{\alpha}{2}\|G_\alpha(x)\|^2 \\ &= \ \frac{1}{2\alpha}\left(\|x-y\|^2 - \|x - T_\alpha(x)\|^2\right) \end{aligned}$$

So $(4*)$ generalizes $(***)$ in the sense of having $T_\alpha(y)$ instead of $T_\alpha(x)$, and this is to accommodate the way into the prox operator is applied in FISTA.

We consider a fixed stepsize $\alpha_k = 1/L_{\nabla f}$.

The proof is known to be quite technical. First one applies the fundamental prox-grad inequality (4*) with $x = t_k^{-1}x_* + (1-t_k^{-1})x_k$ and $y = y_k$

$$\begin{aligned} & F(t_k^{-1}x_* + (1-t_k^{-1})x_k) - F(x_{k+1}) \\ & \quad \geq \; \frac{L_{\nabla f}}{2}\|x_{k+1} - (\underbrace{t_k^{-1}x_* + (1-t_k^{-1})x_k}_{\longrightarrow})\|^2 - \frac{L_{\nabla f}}{2}\|y_k - \underset{\uparrow}{\quad}\|^2 \\ & \quad = \; \frac{L_{\nabla f}}{2t_k^2}\|t_k x_{k+1} - (\underbrace{x_* + (t_k-1)x_k}_{\longrightarrow})\|^2 - \frac{L_{\nabla f}}{2t_k^2}\|t_k y_k - \underset{\uparrow}{\quad}\|^2 \end{aligned}$$

Using the convexity of F and $\delta_k = F(x_k) - F_*$

$$\begin{aligned} & F(t_k^{-1}x_* + (1 - t_k^{-1})x_k) - F(x_{k+1}) \\ \leq \; & (1 - t_k^{-1})(F(x_k) - F_*) - (F(x_{k+1}) - F_*) \\ = \; & (1 - t_k^{-1})\delta_k - \delta_{k+1} \end{aligned}$$

It is simple to see from the expression for y_k that

$$\|t_k y_k - (x_* + (t_k - 1)x_k)\|^2 \; = \; \| \underbrace{t_{k-1}x_k - (x_* + (t_{k-1} - 1)x_{k-1})}_{u_k} \|^2$$

Putting it altogether,

$$(1 - t_k^{-1})\delta_k - \delta_{k+1} \; \geq \; \frac{L_{\nabla f}}{2t_k^2}\|u_{k+1}\|^2 - \frac{L_{\nabla f}}{2t_k^2}\|u_k\|^2$$

From $t_k^2 - t_k = t_{k-1}^2$,

$$\|u_{k+1}\|^2 + \frac{2}{L_{\nabla f}} t_k^2 \delta_{k+1} \; \leq \; \|u_k\|^2 + \frac{2}{L_{\nabla f}} t_{k-1}^2 \delta_k.$$

By summing both sides of this inequality from 1 to $k-1$

$$\begin{aligned}\|u_k\|^2 + \frac{2}{L_{\nabla f}} t_{k-1}^2 \delta_k &\leq \|u_1\|^2 + \frac{2}{L_{\nabla f}} t_0^2 \delta_1 \\ &= \|x_1 - x_*\|^2 + \frac{2}{L_{\nabla f}} (F(x_1) - F_*) \\ &\leq \|x_0 - x_*\|^2 \quad \text{Why?}\end{aligned}$$

... but this is not really necessary ...

Hence,

$$\frac{2}{L_{\nabla f}} t_{k-1}^2 \delta_k \leq \|x_0 - x_*\|^2$$

and from $t_{k-1} \geq (k+1)/2$

$$\boxed{F(x_0) - F_* \leq \left(2L_{\nabla f}\|x_0 - x_*\|^2\right) \frac{1}{(k+1)^2}, \quad \forall k \geq 1}$$

From here we see that both AG and FISTA require $\mathcal{O}(\epsilon^{-0.5})$ iterations to achieve $F(x_k) - F_* < \epsilon$.

This result is also true for a stepsize rule enforcing sufficient decrease (Slide 104) provided:

i) The backtracking starts at $\bar{\alpha} = \alpha_{k-1}$. In this way $\alpha_k \leq \alpha_{k-1}, \forall k \geq 1$ and the stepsize is monotone decreasing (something to be used in the above proof).

ii) The sufficient decrease condition of Slide 104 is invoked at y_k rather than x_k.

Contrasting with the proximal gradient method, and for any of the cases (stepsize fixed at $1/L_{\nabla f}$ or satisfying sufficient decrease), AG/FISTA are not descent methods, meaning that $F(x_{k+1}) < F(x_k)$ might not hold.

When applied to a strongly convex f (f in $F = f + g$), the WCC bound becomes

$$\mathcal{O}\left(\sqrt{\frac{L_{\nabla f}}{\mu_f}}(-\log \epsilon)\right)$$

this with a constant better than the proximal gradient method by a constant factor of $\sqrt{L_{\nabla f}/\mu_f}$, as in the heavy ball method. For a proof in the AG case ($g = 0$) see, e.g., Wright 2017.

In the case of FISTA, this bound is achieved under a modification consisting of restarting the method at every $N = \left\lceil \sqrt{8\kappa_f} - 1 \right\rceil$ iterations with $\kappa_f = \frac{L_{\nabla f}}{\mu_f}$ (see Theorem 10.39 in Beck 2017).

Nesterov's AG method is optimal in the sense of achieving the best convergence rates within all methods of the type

$$x_{k+1} \;=\; x_k + \sum_{j=0}^{k} \xi_j \nabla f(x_j) \quad \text{Ⓖ}$$

for some ξ_j, $j = 0, \ldots, k$. This includes already a number of gradient methods and, according to Nesterov's 1998 lecture notes, it can be further generalized to include nearly all practical gradient methods.

For convex functions the result is as follows:

For any $1 \leq k < \frac{1}{2}(n+1)$ and x_0, there exists a smooth convex function such that for any method of the type Ⓖ,

$$f(x_k) - f_* \;\geq\; \frac{3L\|x_0 - x_*\|^2}{32(k+1)^2}$$

where x_* is a minimizer of f, $f_* = f(x_*)$, and $L = L_{\nabla f}$ is the Lipschitz constant of ∇f.

The function f depends on k (!!). First we define $f_k(x) = \frac{L}{4}(e_1^\top x - \frac{1}{2}x^\top Ax)$ with $e_1 = (1, 0, \ldots, 0)^\top$, $L > 0$, and

$$A = \begin{array}{c} \\ k\left\{\vphantom{\begin{matrix}1\\1\\1\\1\\1\\1\end{matrix}}\right. \\ \\ \end{array} \left(\begin{array}{cc} \overbrace{\begin{matrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & 2 & -1 \\ & & & & -1 & 2 \end{matrix}}^{k} & \Large 0 \\ \Large 0 & \Large 0 \end{array} \right)$$

The proof of the result reasons around

$$(x_k)_i \;=\; \begin{cases} 1-\frac{i}{k+1}, & i=1,\ldots,k \\ 0, & k+1\leq i\leq n \end{cases}$$

and $(f_k)_* = \frac{L}{8}(-1+\frac{1}{k+1})$. The function f is taken as f_{2k+1}.

Along these lines, Nesterov 1998 also provided a strongly convex example f for which any method of type $\textcircled{\text{G}}$ yields

$$\|x_k - x_*\| \;\geq\; \left(\frac{\sqrt{\kappa_f}-1}{\sqrt{\kappa_f}+1}\right)^k \|x_0 - x_*\|$$

$$f(x_k) - f_* \;\geq\; \frac{\mu}{2}\left(\frac{\sqrt{\kappa_f}-1}{\sqrt{\kappa_f}+1}\right)^{2k} \|x_0 - x_*\|^2$$

where $\kappa_f = L_{\nabla f}/\mu_f$, x_* is the minimizer of f and $f_* = f(x_*)$.

The first inequality translates into a WCC bound of $\mathcal{O}(\sqrt{\kappa_f}(-\log\epsilon))$.

EXERCISE: Apply FISTA to the examples at the end of Chapter 4 and compare.

We now elaborate on two problems mentioned before.

Example (3 Feature selection and LASSO)

In many data science/machine learning applications some features are more important than others. Many features are irrelevant for prediction.

The process of feature selection is to select the relevant features among all $a_j, j = 1, \ldots, N$. A simple idea is to solve

$$\min_{\omega} \sum_{j=1}^{N} (a_j^\top \omega - y_j)^2 + \lambda \|\omega\|_0$$

The larger $\lambda > 0$ is the less features will be used. Remember that $\|\omega\|_0 = |\{i : \omega_i \neq 0\}|$ is the support of ω (and not really a norm).

Example (3 Feature selection and LASSO (Cont.))

As in previous relevant examples, this problem can be statistically motivated.

Its solution is difficult as it is NP-hard. Two popular heuristics to compute an approximate solution are:

1. Iterative Hard Thresholding. This amounts to apply the proximal gradient method with $g(\omega) = \lambda\|\omega\|_0$ (and $x = \omega$), using the prox-operator for $\alpha\|\omega\|_0$ given in Chapter 3.
2. Orthogonal Matching Pursuit. A simple idea consisting of iteratively increasing the support of ω in a greedy way from an initial singleton. Each iteration cost a linear least squares problem.

Example (3 Feature selection and LASSO (Cont.))

3. One can also consider an alternative formulation given by

$$\begin{aligned} \min \quad & \sum_{j=1}^{N} (a_j^\top \omega - y_j)^2 \qquad \textcircled{$L0_k$} \\ \text{s.t.} \quad & \|\omega\|_0 \leq k, \end{aligned}$$

where $k \in \mathbb{N}$. Note that this problem is not equivalent (In what sense?) to the $L0_\lambda$ one (because $\|\cdot\|_0$ is not convex).

3.1 Then one can also do Iterative Hard Thresholding on $L0_k$ by $g(\omega) = \delta_C(\omega)$ and $C = \{\omega : \|\omega_0\| \leq k\}$ and the proximal gradient method. EXERCISE: Figure out the prox-operator.

3.2 $L0_\lambda$ can be reformulated as a QP with binary variables, and solved using appropriate software. This can lead to rigorous techniques (in the sense of really solving $L0_\lambda$).

Alternatives to deal with the $\|\cdot\|_0$ norm in $L0_k$ and $L0_\lambda$, are respectively,

$$\begin{array}{lll} \min\limits_{\omega} & \sum\limits_{j=1}^{N} (a_j^\top \omega - y_j)^2 & \quad L1_\delta \\ \text{s.t.} & \|\omega\|_1 \leq \delta & \end{array}$$

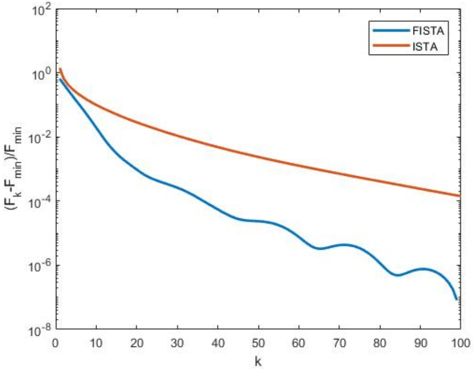
(known as LASSO, least absolute shrinkage and selection operator, Tibshirani 1996), where $\delta > 0$, and

$$\min_{\omega} \sum_{j=1}^{N} (a_j^\top \omega - y_j)^2 + \lambda \|\omega\|_1 \quad L1_\lambda$$

(known as basis pursuit denoising BPDN, Chen, Donoho, and Sanders 1999), where $\lambda > 0$.

Let $A \in \mathbb{R}^{500\times 300}$ and $b \in \mathbb{R}^{500}$ be randomly generated.

The comparison between ISTA and FISTA for solving L1$_\lambda$ with $\lambda = 1$ and $L_{\nabla f} = 1/\lambda_{\max}(A^\top A)$:



EXERCISE $L1_\delta$ and $L1_\lambda$ are equivalent in the sense that for each $\lambda \geq 0$ there is an optimal solution of $L1_\delta$ for some $\delta \geq 0$ that is optimal for $L1_\lambda$ and vice versa.

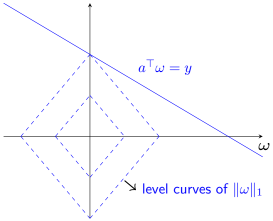
EXERCISE Both problems $L1_\delta$ and $L1_\lambda$ are equivalent to LPs (and thus can be solved by Linear Programming techniques, such as the simplex method or interior-point algorithms).

Problem $L1_\lambda$ can also be derived from statistical assumptions. If we assume that the components of ω_j are independently Laplace distributed

$$Pr(\omega_j \mid \alpha) \;=\; e^{-\alpha|\omega_j|}$$

for some $\alpha > 0$, it can be shown that maximizing likelihood is equivalent to $L1_\lambda$ for some $\lambda > 0$.

Intuitively, one can also see that the ℓ_1-norm promotes sparsity (thus making $L1_\lambda$ a good surrogate model for $L0_\lambda$):



One popular technique to solve $L1_\lambda$ is LARS (Least Angle Regression and Shrinkage, Efron et al. 2004): when λ is very large, the optimal solution of $L1_\lambda$ is $\omega_*(\lambda) = 0$. On the other hand, for $\lambda = 0$ $\omega_*(0)$ is the optimal least squares solution.

It can be showed that $\omega_*(\lambda)$ is piecewise linear. Why?

LARS starts from $\omega = 0$ adding features iteratively (a bit like Orthogonal Matching Pursuit), based on explicit formulas for the changes of the features along the path. To some extent λ is not pre-determined here.

Problem $L1_\lambda$ can be solved efficiently with FISTA, which can be further enhanced by appropriate choices of the stepsize.

We have seen two possibilities: $\alpha_k = 1/L_{\nabla f}$ or α_k satisfying a sufficient decrease condition.

Another possibility is the so-called Barzilai-Borwein (or spectral) stepsize, which is derived for $F = f$ but can be used when $g \neq 0$ (either in ISTA or FISTA): When we have exact Hessian $\nabla^2 f(x_k)$ (surely the case $L1_\lambda$).

$$\nabla^2 f(x_k)(x_k - x_{k-1}) \simeq \nabla f(x_k) - \nabla f(x_{k-1})$$

Using the approximation $\nabla^2 f(x_k) \simeq \frac{1}{\alpha_k} I$ and imposing this relationship in the least-square sense

$$\frac{1}{\alpha_k} \; = \; \underset{\alpha}{\operatorname{argmin}} \| (x_k - x_{k-1})\alpha - (\nabla f(x_k) - \nabla f(x_{k-1})) \|^2$$

resulting in

$$\alpha_k \; = \; \frac{(x_k - x_{k-1})^\top (x_k - x_{k-1})}{(x_k - x_{k-1})^\top (\nabla f(x_k) - \nabla f(x_{k-1}))} \qquad \text{BB}$$

A related expression could be derived if instead one would have used

$$\alpha_k = \underset{\alpha}{\operatorname{argmin}} \|(x_k - x_{k-1}) - (\nabla f(x_k) - \nabla f(x_{k-1}))\alpha\|^2$$

Then schemes are not monotone (neither in $f(x_k)$ nor in $\nabla f(x_k)$) when $g = 0$, i.e., when we simply have

$$x_{k+1} \;=\; x_k - \alpha_k \nabla f(x_k).$$

No convergence is guaranteed for a smooth convex f (although it converges linearly, actually r-linearly, for quadratic functions), again when $g = 0$. The fix is to use a non-monotone line search.

In the case of $g \neq 0$ and in the context of the proximal gradient method, SpaRSA (Wright, Nowak, Figueiredo 2009) is such a non-monotone scheme:

It starts each line search with $\bar{\alpha} = s \in [\alpha_{\min}, \alpha_{\max}]$, with $0 < \alpha_{\min} < \alpha_{\max}$, as close as possible to BB. Then $\alpha_k = \beta^i \bar{\alpha}$, with $\beta \in (0,1)$, and $i \in \{0,1,\ldots\}$, the first that

$$f(T_{\bar{\alpha}}(x_k)) \; > \; \max_{\max\{k-M,0\} \leq i \leq k} f(x_i) - \frac{\sigma}{2\alpha_k} \|T_\alpha(x_k) - x_k\|^2$$

with $\sigma \in (0,1)$ chosen very small $M \in \{0,1,\ldots\}$.

Example (4 Matrix completion (already seen a little in Chapter 3))

One of the typical formulations of matrix completion can be seen as the extension of least-squares from vector to matrix spaces. The data is represented by A_j, $j = 1, \ldots, N$, where A_j is an $n \times p$ matrix.

We then seek an $n \times p$ matrix X by solving

$$\min_X \sum_{j=1}^{m} (\langle A_j, X \rangle - y_j)^2 \qquad \textcircled{\text{MC1}}$$

where $\langle \cdot, \cdot \rangle$ represents the trace inner product $\langle A, B \rangle = \mathrm{tr}(A^\top B)$. The observed A's can come in different types possibly containing one (or a few) nonzero elements.

Example (4 Matrix completion (Cont.))

One would like the recovered X to contain as much information possible in a compressed way. So it is natural to solve

$$\min_X \sum_{j=1}^{m} (\langle A_j, X\rangle - y_j)^2 + \lambda \|X\|_*, \qquad \text{MC2}$$

where $\|\cdot\|_*$ is the so-called nuclear or trace norm.

$$\|X\|_* = \sum_{i=1}^{r} \sigma_i(X),$$

in other words the sum of the r singular values of a matrix.

Example (4 Matrix completion (Cont.))

So MC2 is a regularized form of MC1, promoting solutions that are low rank.

Note that $\|\cdot\|_*$ is non-smooth but convex. A related popular matrix norm is the Frobenius norm

$$\|X\|_F = \sqrt{\operatorname{tr}(X^\top X)} = \sqrt{\sum_{i=1}^{r} \sigma_i(X)^2}.$$

Example (4 Matrix completion (Cont.))

Another typical formulation appears in the design of recommender systems (see the NETFLIX challenge) where one seeks a minimal rank X for which some entries $X_{ij} = A_{ij}$ are known, $(i,j) \in \Omega$:

$$\min_X \operatorname{rank}(x) \quad \text{s.t.} \quad X_{ij} \;=\; A_{ij},\; (i,j) \in \Omega \qquad \text{(MC3)}$$

for which a convex surrogate is

$$\min_X \|X\|_* \quad \text{s.t.} \quad X_{ij} \;=\; A_{ij},\; (i,j) \in \Omega \qquad \text{(MC4)}$$

This problem also offers a valid statistical solution (Candès and Recht 2009).

Example (4 Matrix completion (Cont.))

A formulation related to MC4 is (for some $\lambda > 0$)

$$\min_X \underbrace{\frac{1}{2} \sum_{(i,j)\in\Omega} (X_{ij} - A_{ij})^2}_{f(X)} + \underbrace{\lambda \|X\|_*}_{g(X)} \qquad \textcircled{\text{MC5}}$$

leading to our familiar setting $F = f + g$ where f is smooth and g is convex.

One can apply proximal gradient methods (such as FISTA) to solver MC5. The prox-operator is

$$\operatorname{prox}_{\lambda g}(X) \;=\; \operatorname*{argmin}_{U\in\mathbb{R}^{n\times p}} \left\{ \alpha g(U) + \frac{1}{2}\|U - X\|_F^2 \right\}$$

Example (4 Matrix completion (Cont.))

For the nuclear norm $g = \|\cdot\|_*$ EXERCISE

$$\mathrm{prox}_{\lambda\|\cdot\|_*}(X) \;=\; U\Sigma_{\mathrm{prox}}V^\top$$

where $U\Sigma V^\top$ is the SVD of X and

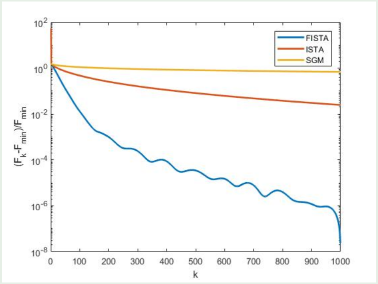
$$\Sigma_{\mathrm{prox}} \;=\; \mathrm{diag}\left(\max\{\sigma_i(X) - \lambda, 0\}\right).$$

What would be the prox-operator for $g(\cdot) = \|\cdot\|_F$?

Example (4 Matrix completion (Cont.))

$A \in \mathbb{R}^{500\times 500}$ is formed from a subset of MovieLens dataset, where A_{ij} represents rating (from 1 to 5) of movie j by user i. Every movie has been rated by just a number of users. Thus, many entries of A are missing.

Here is the performance of FISTA, ISTA, and subgradient method (SGM) for completing A (by setting $\lambda = 1$ in MC5):



Presentation outline

Our data set for analysis

$$D = \{(a_j, y_j), j = 1, ..., N\}$$

of features (a_j) and labels (y_j) will now be seen as a sample taken from an input-output space

$$D \subset \mathcal{A} \times \mathcal{Y}$$

The goal is still to learn a prediction function $\phi : \mathcal{A} \to \mathcal{Y}$ that takes a feature a and predicts a label $\phi(a)$.

We need to assume that a point $(a, y) \in \mathcal{A} \times \mathcal{Y}$ arises with a certain (joint) probability P_Ω.

Our goal is thus to find ϕ that minimizes the expected risk of misclassification

$$R(\phi) \;=\; P_\Omega[\phi(a) \neq y] \;=\; E[\mathbb{1}(\phi(a) \neq y)]$$

where $\mathbb{1}$ is the logical indicator function ($= 1$ if the argument is true, $= 0$ otherwise).

Of course P_Ω is generally unknown, and this is why one takes a sample D, leading to the empiricial risk of misclassification

$$R_N(\phi) \;=\; \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}[\phi(a_j) \neq y_j]$$

as an approximation or estimation of $R(\phi)$.

We have touched upon overfitting in Chapter 1: A ϕ that perfectly minimizes $R_N(\phi)$ may overfit D and perform poorly in $\mathcal{A} \times \mathcal{Y}$.

In fact consider the prediction function of memorization type

$$\phi_N^{mem}(a) \; = \; \begin{cases} y_j & \text{if } a = y_j \text{ for some } j, \\ 1 & \text{otherwise.} \end{cases}$$

One has $R_N(\phi_N^{mem}) = 0$, i.e., ϕ_N^{mem} is optimal for the training set D.

However, ϕ_N^{mem} will certainly perform badly outside D (and besides its evaluation requires storing D).

Can we approximate $\min_\phi R(\phi)$ with a sequence of predictions ϕ_N learned from $D_N = D$ with $N \to \infty$?

In other words, can we obtain

$$\lim_{N\to\infty} R(\phi_N) \;=\; \min_{\phi} R(\phi)$$

Unfortunately there is a bad probability under which this does not happen (the "no free lunch theorem")

$$\forall \epsilon > 0, N \in \mathbb{N} \quad \exists P_\Omega \text{ for } \mathcal{A} \times \mathcal{Y} \quad \text{such that}$$

$$\begin{cases} R(\phi_*) = 0 & \text{for some } \phi_* : \mathcal{A} \to \mathcal{Y} \\ R_N(\phi_N) \geq \frac{1}{2} - \epsilon & \text{for any prediction } \phi_N \text{ learned from } D_N \end{cases}$$

See Devroye, Györfi, and Lugosi 1996

Note that the above R depends on N (as P_Ω also does).

This is indeed a bit cheating. A stronger version says that

$\forall\{\epsilon_N\}$ with $\epsilon_N > 0$ and $\epsilon_N \to 0$ $\quad \exists P_\Omega$ for $\mathcal{A} \times \mathcal{Y}$ $\quad$ such that

$$\left\{ \begin{array}{ll} R(\phi_*) = 0 & \text{for some } \phi_* : \mathcal{A} \to \mathcal{Y} \\ R_N(\phi_N) \geq \epsilon_N & \forall N \end{array} \right.$$

Now P_Ω does not depend on N but on the whole sequence $\{\epsilon_N\}$.

Still, all of this tells us that we either restrict P_Ω (generative approach) or we restrict the class of predictions (discriminative approach).

The discriminative approach corresponds to the parameterization seen in Chapter 1.

To know how to appropriately parameterize the class of prediction functions, avoid overfitting, and learn the optimal parameters, we need to understand better the relationship between the problems

$$\textcircled{1} \quad \boxed{\min_{\phi\in\Phi} R(\phi)} \qquad \text{and} \qquad \boxed{\min_{\phi\in\Phi} R_N(\phi)} \quad \textcircled{2}$$

for a certain class Φ of prediction functions.

We will assume that the minimizers exist:

- ϕ_* is a minimizer of $\textcircled{1}$,
- ϕ_N is a minimizer of $\textcircled{2}$.

(ϕ_* is the Bayes optimal function and $R(\phi_*)$ the Bayes error.)

First, we bound the regret

$$
\begin{aligned}
& R(\phi_N) - R(\phi_*) \\
& \leq \; [R(\phi_N) - R_N(\phi_N)] + \underbrace{[R_N(\phi_N) - R_N(\phi_*)]}_{\leq 0} + [R_N(\phi_*) - R(\phi_*)] \\
& \leq \sup_{\phi \in \Phi} \{R(\phi) - R_N(\phi)\} + [R_N(\phi_*) - R(\phi_*)]
\end{aligned}
$$

To show that the second term goes to zero with N, we apply the Hoeffding's inequality:

Let $Z_1, \ldots, Z_N$ be independent bounded random variables ($Z_i \in [a, b]$). Then for any $t > 0$

$$P_\Omega\left(\frac{1}{N}\sum_{j=1}^{N}(Z_j - E(Z_j)) \geq t\right) \;\leq\; \exp\left(\frac{-2Nt^2}{(b-a)^2}\right)$$

In fact from there we have (EXERCISE)

Let ϕ be a measurable function from $\mathcal{A}$ to $\mathcal{Y}$. Then

$$P_\Omega\left(|R_N(\phi) - R(\phi)| \;\leq\; \sqrt{\frac{1}{2N}\log\left(\frac{1}{\alpha}\right)}\right) \;>\; 1-\alpha$$

for any $\alpha \in (0,1)$.

By taking $N \to \infty$ we thus see that $R_N(\phi_*)$ tends to $R(\phi_*)$ with high probability.

Bounding the first term $\sup_{\phi \in \Phi} |R(\phi) - R_N(\phi)|$ requires understanding the complexity or capacity of the class of prediction functions Φ.

One such measure of complexity is the Vapnik-Chervonenkis (VC) dimension:

Let Φ be a class of functions from $\mathcal{A}$ to $\mathcal{Y}$.

The VC-dimension C_Φ of Φ is the largest number of features $\{a_1, \ldots, a_{C_\Phi}\}$ such that for any label set $\{y_1, \ldots, y_{C_\Phi}\}$, $\exists \phi \in \Phi$ that correctly classifies the features.

A simple example is binary classification with linear predictions. Here $|\mathcal{Y}| = 2$ and ϕ are affine functions $\mathbb{R}^n \to \mathbb{R}$. In the notation of Chapter 1,

$$\phi(a; x) \;=\; \phi(a; w, b) \;=\; a^\top w + b \quad \text{with} \quad x = (w, b)$$

One has $(w, b) \in \mathbb{R}^{n+1}$ and the VC-dimension is $n + 1$ (EXERCISE).

The VC-dimension of the space of polynomials of degree $\leq p$ would then be $\binom{n+p}{n}$, thus higher.

So, the gap between the expected and empirical risks is larger for higher order polynomials since the higher capacity gives room to overfit a training data set.

Using the VC-dimension one has:

Let Φ be a class of functions with VC-dimension C_Φ. Then

$$P_\Omega\left(\sup_{\phi\in\Phi}|R_N(\phi)-R(\phi)| \;\leq\; \mathcal{O}\left(\sqrt{\frac{1}{2N}\log\left(\frac{1}{\alpha}\right)+\frac{C_\Phi}{N}\log\left(\frac{N}{C_\Phi}\right)}\right)\right)$$

$$> \; 1-\alpha \quad \textcircled{\star}$$

for any $\alpha\in(0,1)$.

We see immediately that for a fixed Φ the error

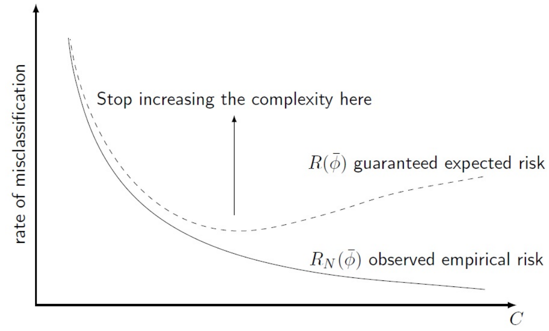
$$\sup_{\phi\in\Phi}|R(\phi)-R_N(\phi)| \xrightarrow[N\to\infty]{} 0$$

with high probability (this result is of type PAC, probably approximately correct).

To learn effectively we need to balance the complexity/capacity of the predictions with the size of the training set.

Standard risk minimization is an approach where one chooses a structure, i.e., a collection of nested prediction function families with increased complexity.

Given the dependence of $\bigcirc\!\!\!\!\!\star$ on C_Φ (of type $C_\Phi \log(1/C_\Phi)$) there is a point after which we should not further increase C_Φ.



The experiments corresponding to this plot is roughly carried out by forming a collection of subsets of a given family Φ (Φ contains several classes of functions)

$$\Phi_C \ = \ \{\phi \in \Phi : C_\Phi(\phi) \leq C\}$$

where C is a hyperparameter. Then $\bar{\phi}$ is the minimizer of $R_N(\phi)$ in Φ_C. See Vapnik 1998 and Vapnik and Chernovenkis 1974/79.

NOTE: This curve behavior is also observed when we replace C by the time or effort taken in the learning/training phase (application of optimization). However, we leave such an analysis for later in the course.

A common practice for choosing among prediction functions in a given class (and thus estimate the gap between expected and empirical risks) is to compare them using cross-validation.

The data split into three disjoint sets: training, validation, and testing

One first searches a small subset of viable candidates by minimizing $R_N(\phi)$ in the chosen prediction class using the training set.

From those candidates, one selects the prediction which as the best performance in the validation set.

The testing set is only used to assess the generalized performance of the prediction chosen.

Recent research has shown that convex models may not provide powerful predictions.

One popular type of such nonconvex models is nonlinear regression, where one minimizes

$$L(x) \;=\; \frac{1}{N}\sum_{j=1}^{N}\Big(r(a_j^\top w) - y_j\Big)^2 \quad (x = w)$$

where remember $D = \{(a_j, y_j), j = 1, \ldots, N\}$ is the data set of features and labels available for training.

In this case there is only one ridge function r (ridge because is only varies along the directions a_j's).

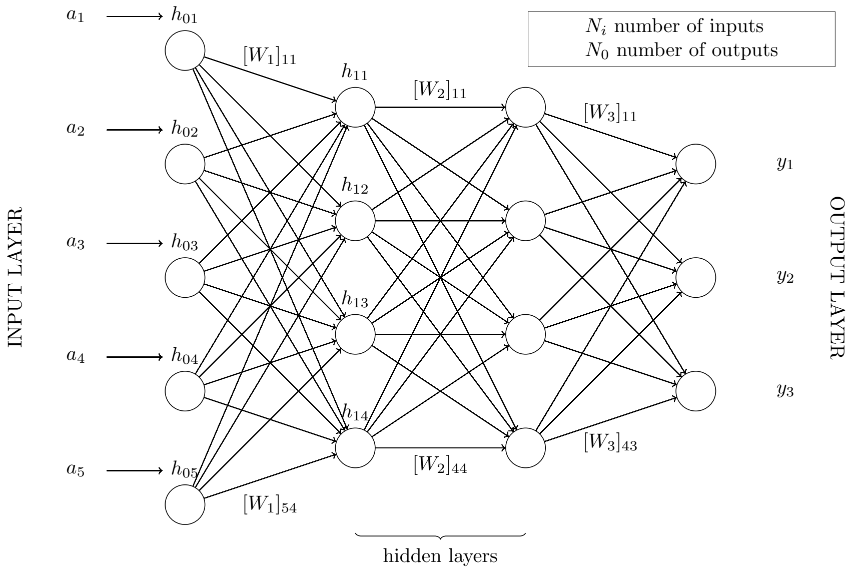
We have essentially a nonlinear least-squares problem.

Advances in nonconvex models have taken place in deep (or hierarchical) learning, when training deep artificial neural networks.

An artificial neural network is a network of connected nodes (known as units, neurons, or perceptrons) connected by edges (known as links).

They are inspired from the brain neural networks.

The nodes are typically arranged by layers between an input and output layer.



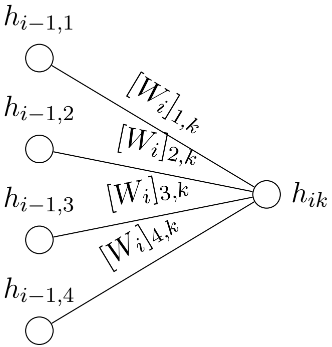
(Example taken from Curtis and Scheinberg 2017.)

The features are feed to the input layer. The subsequent layers are hidden as their values are not observed from the outside (this will only happen in the output layer).

The simple example above depicts a feed forward neural network where edges only exist between nodes in a layer and subsequent layers.

The values are passed from one node in a layer to all nodes in the next layer, after multiplication by connection weights (associated with each edge), followed by adding a term (called the bias term).

At a given node, the value is further modified by the use of an activation function (or gateway). Using the example above one has



$$h_{ik} \; = \; s_{ik} \left(\sum_l [W_i]_{lk} h_{i-1,l} + (w_i)_k \right)$$

activation function weights bias term

Finally, the last layer provides the value of the prediction $\phi(x) = \phi(a; x)$, and the goal is to find the parameters x such that $\phi(a_j; x) \simeq y_j, j = 1, \ldots, N_0$.

The weights and the bias term are the parameters of the overall prediction function.

In the above example, one has

$$W_1 \in \mathbb{R}^{4\times5}, W_2 \in \mathbb{R}^{4\times4}, W_3 \in \mathbb{R}^{3\times4}$$

and

$$w_1 \in \mathbb{R}^{4}, w_2 \in \mathbb{R}^{4}, w_3 \in \mathbb{R}^{3}$$

a total of 59 parameters.

The activation function is responsible to the introduction of nonlinearity in the prediction model (and can also be parameterized).

The idea is to mimic the behavior of a brain neuron (if the combined energy inputed is large enough, the output of the neuron will be large too, otherwise small), leading to the threshold function (τ is the threshold)

$$s(t) \;=\; \begin{cases} 1 & \text{if } t > \tau \\ 0 & \text{otherwise.} \end{cases}$$

However, this function is discontinuous. Smooth versions of the threshold function are the sigmoid and $\tanh$ functions

$$s(t) \;=\; \frac{1}{1+e^{-t}}, \qquad s(t) \;=\; \frac{e^t - e^{-t}}{e^t + e^{-t}}$$

Another popular choice is the ReLU (rectified linear unit) (related to the hinge loss of Chapter 1)

$$s(t) \;=\; \begin{cases} t & \text{if } t > 0 \\ 0 & \text{otherwise,} \end{cases}$$

and its variants (PReLU and LReLU).

At the end one computes $\phi_j(x), j = 1, \ldots, N_0$.

In deep learning, the graph structure is more complex.

In recurrent neural networks, for instance, there are recurrent constructions where at least one node in a layer outputs its value to a node in a previous layer.

This effect introduces memory in the network, making the model even more expressive

Another example is convolutional neural networks that exploit inputs consisting of images (which are inherently connected due to their spatial relationship).

The resulting problems are slightly different from Chapter 1

$$\min_x L(x) \;=\; \textstyle\sum_{j=1}^{N_0} \ell(y_j; x) \;=\; \ell(y_j; \phi_j(x))$$

W's, w's, etc.

and are highly nonconvex (with numerous saddle points and minimizers).

We obtain N_0 output functions

$$\phi_j(y) = \phi_j(a_1, \ldots, a_{N_i}; x), \quad j = 1, \ldots, N_0.$$

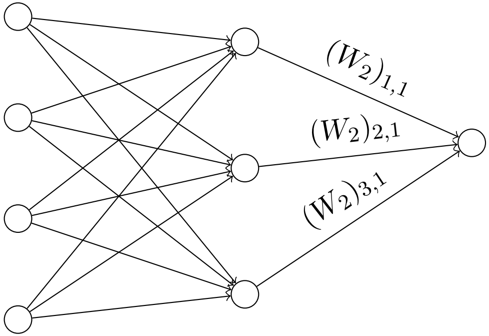
Note that this is a simple extension of the nonlinear regression loss function given before. Why?

For optimization purposes, in neural networks it is possible to (efficiently) compute the gradient of $\phi_j(x)$ for a given x by backpropagation (as in automatic differentiation)

This is done first with a forward pass to compute function values and then by a backward pass to get the derivative values (through the chain rule).

Finally, we remark that neural networks are universal approximations as they approximate any continuous function well on arbitrary compact sets — thus leading to expressiveness.

This can be seen from a feed forward neural network (with only one hidden layer and one output node)



$$(W_1)_\ell = \begin{pmatrix} (W_1)_{11} \\ \vdots \\ (W_1)_{41} \end{pmatrix}$$

One has

$$\psi(a) \;=\; \phi(a; W, w) \;=\; \sum_{l=1}^{N_1} (W_2)_{l_1} s\left((w_1)_l^\top a + (w_1)_l \right)$$

N_1 is the number of nodes in the hidden layer.

The output unit is considered linear.

The following result was derived by Hornick 1991 (remember that N_i is the number of inputs).

Let s be a nonconstant, bounded, and continuous function on a given compact S. Then for any continuous function f on S and any $\epsilon > 0$,

$$\exists N_1 \in \mathbb{N}\; \exists w_1 \in \mathbb{R}^{N_1}, (W_1)_l \in \mathbb{R}^{N_i}, l = 1, \ldots, N_1, (W_2)_1 \in \mathbb{R}^{N_1}$$

such that

$$|\psi(a) - f(a)| \;<\; \epsilon \quad \forall a \in S$$

The functions ψ are thus dense in the space of continuous functions on S.

This result was first proved by Cyberko 1989 for sigmoid activation functions.

Training deep neural networks will depend on the design of the network and on the choice of the starting point (for the optimization method).

Presentation outline

We now turn our attention to optimization methods for large-scale machine learning.

In supervised machine learning, one has access to a training data set $D = \{(a_j, y_j), j \in \{1, \ldots, N\}\}$ of features and labels (which can come all-at-once or incrementally). This data is typically independently drawn.

The goal is to find a prediction function ϕ that maps features to labels, and we will assume that ϕ is parameterized by x, i.e., $\phi(x) = \phi(a; x)$.

Misclassification is quantified by a loss function ℓ.

The expected risk of misclassification of Chapter 6

$$R(\phi) \; = \; P_\Omega[\phi(a) \neq y] \; = \; E(\mathbb{1}(\phi(a) \neq y))$$

is thus written as

$$R(x) \; = \; \int \quad \underbrace{\ell(y; \phi(a; x))}_{\ell(a,y;x) \text{ in Chapter 1}} \quad dP_\Omega(a, y)$$

The empirical risk of misclassification of Chapter 6

$$R_N(\phi) \ = \ \frac{1}{N}\sum_{j=1}^{N} \mathbb{1}[\phi(a_j) \neq y_j]$$

is thus written as

$$R_N(x) \ = \ \frac{1}{N}\sum_{j=1}^{N} \ell(y_j; \phi(a_j; x)).$$

We will focus on the minimization of $R_N(x)$ without any regularization term being added.

For the sake of a simple notation let f be the composition of the loss function ℓ and the prediction function ϕ.

As in Chapter 6, let us also consider the data set as a sample taken from an input-output space $\mathcal{A} \times \mathcal{Y}$.

For simplicity, we represent a sample or a set of samples by a random seed ξ. There, a set of realizations $\{\xi_{[j]}\}_{j=1}^N$ corresponds to a sample set $\{(a_j, y_j), j = 1, \ldots, N\}$.

The loss incurred by the parameter vector x with respect to the j–th sample is represented as

$$f_j(x) \;=\; f(x; \xi_{[j]}).$$

The empirical risk of misclassification is then written as

$$R_N(x) \;=\; \frac{1}{N} \sum_{j=1}^{N} f_j(x).$$

In an optimization algorithm, k has been denoting in this course the iteration counter. Then, ξ_k will denote the realization sample taken at iteration k.

There are two main classes of algorithms for machine learning: stochastic and batch.

In the first class we find the stochastic gradient (SG) method (Robbins and Monro, 1951):

$$x_{k+1} \;=\; x_k - \alpha_k \nabla f_{j_k}(x_k)$$

where j_k is randomly generated from $\{1, \ldots, N\}$ (and thus there is an underlying $\xi_{[j_k]}$).

Notes:

- Each iteration is very cheap (only one gradient calculation)
- $\{x_k\}$ is no longer a sequence but a stochastic process determined by $\{j_k\}$.
- $\nabla f_{j_k}(x_k)$ might not be descent direction from x_k (although it could be in expectation).

In the second class, we find the gradient, batch gradient, or full gradient method:

$$
\begin{aligned}
x_{k+1} &= x_k - \alpha_k \nabla R_N(x_k) \\
& \left(= x_k - \frac{\alpha_k}{N} \sum_{j=1}^{N} \nabla f_j(x_k) \right)
\end{aligned}
$$

Notes:

- Each iteration is much more expensive but one may expect to take better steps.
- The method can benefit from parallelization in a distributed manner.
- The use of the full gradient opens the door to all type of gradient-based nonlinear optimization.

In Stochastic Optimization, the stochastic gradient is a stochastic approximation (SA) technique whereas the batch gradient method is a sample average approximation (SAA) approach.

There are several reasons why the SG method could be preferable compared to the batch gradient one:

(1) Many large-scale training sets involve redundancy, thus using all the data in every optimization iteration is inefficient (imagine if the training set consists of a series of repeated copies of a subset).

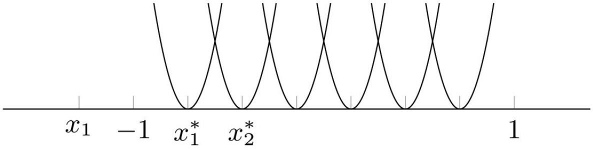
    Also when using the training/validation/testing sets, one could argue against using all the data in the training set to make prediction on the unseen (testing) data.

(2) SG typically makes a very fast initial improvement followed by a slowdown after 1 or 2 epochs (1 epoch $=$ each set of N consecutive accesses):

(Although to obtain such a behavior one needs to run SG for a certain problem using different choices for the step-size $\alpha_k = \alpha$ and identify the best α.)

Bertsekas 2015 came up with the following example to explore the slowdown of SG:

Each f_j in $R_N(x) = \frac{1}{N}\sum_{j=1}^{N} f_j(x)$ is a convex quadratic with zero minimal value and minimizers x_j^* evenly distributed in $[-1, 1]$ such that the minimizer of $R_N(x)$ is $x_* = 0$.



If we start SG with $x_1 \ll -1$, the method will move to the right, but after a while, close to x_*, the method will enter a region of confusion and will move very slowly.

(3) There is also convincing theoretical motivation.

Suppose R_N is strongly convex with minimizer x_* and $R_N^* = R_N(x_*)$.

Then the rate of convergence of the batch method is (see Chapter 4):

$$R_N(x_k) - R_N^* \;=\; \mathcal{O}(r^k), \qquad \text{with } r \in (0,1)$$

corresponding to a WCC bound of

$$\mathcal{O}(\log(1/\epsilon))$$

to reach a gradient of R_N of the size ϵ.

Since the cost is proportional to N (to compute $\nabla R_N(x_k)$) the total work is

$$\mathcal{O}(N \log(1/\epsilon)).$$

On the other hand, the rate of convergence of the SG method (when j_k is uniformly distributed from $\{1, \ldots, N\}$) is (see later in this chapter)

$$E\left[R_N(x_k) - R_N^*\right] \;=\; \mathcal{O}(1/k)$$

corresponding to a total WCC bound of $\mathcal{O}(1/\epsilon)$, without any dependence from N.

In the big data regime, N could be large enough to make $N \log(1/\epsilon)$ worse than $1/\epsilon$.

As we will see also later in this chapter, SG yields the same rate of convergence in the expected risk

$$E\left[R(x_k) - R^*\right] \;=\; \mathcal{O}(1/k), \qquad \text{with } R^* \text{ the minimal value of } R,$$

if we consider $x_{k+1} = x_k - \alpha_k \nabla f(x_k; \xi_k)$ with ξ_k drawn from the joint distribution P_Ω.

In such stochastic optimization setting, the batch method may not be viable as it requires the computation of ∇R.

In turn, the SG method yields the same rate of convergence as for the empirical risk (assuming, of course, N large compared to k).

There is also an argument (Bousquet and Bottou 2008) supporting the idea that the empirical risk does not have to be minimized accurately (thus making SG even more attractive):

Let $\hat{x}_\epsilon$ be an ϵ-accurate solution of $\min R_N(x)$ (which is assumed to have a solution $\hat{x}_*$):

$$R_N(\hat{x}_\epsilon) \;\leq\; R_N(\hat{x}_*) + \epsilon$$

Then, for the expected risk minimization problem $\min R(x)$ (assumed to have a solution x_*), one has

$$\begin{aligned} R(\hat{x}_\epsilon) - R(x_*) \;\leq\; & [R(\hat{x}_\epsilon) - R_N(\hat{x}_\epsilon)] \\ & + \underbrace{[R_N(\hat{x}_\epsilon) - R_N(\hat{x}_*)]}_{\leq \epsilon} \\ & + \underbrace{[R_N(\hat{x}_*) - R(x_*)]}_{\leq R_N(x_*) - R(x_*)} \end{aligned}$$

Hence the first and third term are of (see Chapter 6)

$$\mathcal{O}\left(\sqrt{\frac{\log(1/\alpha) + 2\log(N)(C/\log(C))}{2N}}\right)$$

where $C = C_\Phi$ is the complexity constant of a class Φ of prediction functions assumed fix in this argument.

Our goal is to compute an θ-accurate solution for the expected risk

$$R(\hat{x}_*) \;\leq\; R(x_*) + \theta$$

From the previous derivation we achieve this with $\hat{x}_\epsilon$ if

$$\mathcal{O}\left(\sqrt{\frac{\log(1/\alpha) + 2\log(N)(C/\log(C))}{2N}}\right) \;\leq\; \frac{\theta}{2} \quad \text{and} \quad \epsilon \leq \frac{\theta}{2}$$

From the first inequality above

$$\frac{N}{\log(N)} \;\geq\; \mathcal{O}(\theta^{-2})$$

thus $N \geq \mathcal{O}(\theta^{-2})$.

The batch gradient method has thus a cost of $\mathcal{O}(N) = \mathcal{O}(\theta^{-2})$ per iteration.

The SG, in turn, has a $\mathcal{O}(1)$ cost per iteration.

Hence the SG dominates the batch as long as it reaches an $\frac{\theta}{2}$-accurate solution in fewer than $\mathcal{O}(\theta^{-2})$ iterations.

When R_N is strongly convex, SG complexity bound is $\mathcal{O}(1/\epsilon)$ (see later in this chapter). Since $\epsilon \leq \theta/2$, the cost is $\mathcal{O}(\theta^{-1})$.

When R_N is just convex, SG complexity bound is $\mathcal{O}(\epsilon^{-2})$ thus still $\mathcal{O}(\theta^{-2})$. But, in this case, the batch gradient method converges in $\mathcal{O}(N\epsilon^{-0.5})$ in an acceleration mode thus $\mathcal{O}(\theta^{-2.5})$ and SG is still dominant.

We now analyze the SG method but we will do it (following Bottou, Curtis, and Nocedal 2018) in a general setting of minimizing on obj. function $S : \mathbb{R}^n \to \mathbb{R}$ where $S(x)$ represents

either $E\left[f(x;\xi)\right] = R(x)$ (expected risk)

or $\frac{1}{N}\sum_{j=1}^{N} f_j(x) = R_N(x)$ (empirical risk)

Each iteration of the general stochastic gradient (GSG) method is of the form:

```
k = 0, 1, ...:
    Generate a realization of $\xi_k$.
    Compute a stochastic direction $g(x_k; \xi_k)$.
    Choose a step-size $\alpha_k > 0$.
    Set the new iterate $\boxed{x_{k+1} = x_k - \alpha_k g(x_k; \xi_k).}$
```

We assume that $\{\xi_k\}$ is a sequence of jointly independent random variables.

In the case $S = R_N$, one can pick samples uniformly from $\{1, \ldots, N\}$ and replacing them into the set, corresponding to sampling from a discrete distribution (with uniform weights).

In the case $S = R$, one can pick samples according to the joint distribution P_Ω (in the feature/label setting).

The random variable ξ_k can be seen as a seed for generating the stochastic direction.

The stochastic direction $g(x_k;\xi_k)$ could be

$$\nabla f(x_k;\xi_k) \;=\; \nabla f_{j_k}(x_k) \qquad \text{now called simple or basic SG}$$

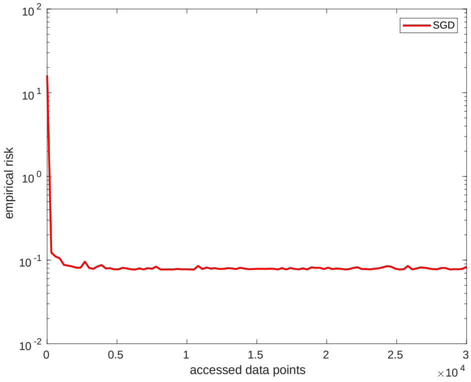
$$\frac{1}{|B_k|}\sum_{j\in B_k}\underbrace{\nabla f(x_k;\xi_{k,j})}_{\nabla_j f(x_k)} \qquad \text{known as mini-batch SG}$$

($|B_k|$ is the mini-batch size)

(The original version of Robbins and Monro covers both as they actually worked with a unbiased estimator of $\nabla S(x_k)$.)

The function S (representing either empirical or expected risk) will be assumed to have Lipschitz continuous gradient ∇S (with constant $L_{\nabla S} > 0$).

Performance of SG on a logistic regression problem with randomly generated 300 samples in dimension 3:



Recall then from Chapter 1 that

$$S(x) - S(\bar{x}) \ \le \ \nabla S(\bar{x})^\top (x - \bar{x}) + \frac{1}{2} L_{\nabla S} \|x - \bar{x}\|^2$$

From this we obtain for the GSG method:

$$\begin{aligned} E_{\xi_k}[S(x_{k+1})] - S(x_k) \ &\le \ -\alpha_k \nabla S(x_k)^\top E_{\xi_k}\left[g(x_k;\xi_k)\right] \\ &\quad + \frac{1}{2}\alpha_k^2 L_{\nabla S} E_{\xi_k}\left[\|g(x_k;\xi_k)\|^2\right] \end{aligned}$$

Note that $E_{\xi_k}[S(x_{k+1})]$ is an expectation w.r.t. ξ_k, and $S(x_{k+1})$ does depend on ξ_k.

Proof.

One takes expectations in the inequality

$$\begin{aligned} S(x_{k+1}) - S(x_k) \ &\le \ \nabla S(x_k)^\top (x_{k+1} - x_k) + \frac{1}{2} L_{\nabla S} \|x_{k+1} - x_k\|^2 \\ &= \ -\alpha_k \nabla S(x_k)^\top g(x_k;\xi_k) + \frac{1}{2} L_{\nabla S} \alpha_k^2 \|g_(x_k;\xi_k)\|^2 \end{aligned}$$

noting that x_k does not depend on ξ_k. □

If $g(x_k;\xi_k)$ is an unbiased estimator of $\nabla S(x_k)$,

$$E_{\xi_k}[g(x_k;\xi_k)] \;=\; \nabla S(x_k),$$

and then

$$E_{\xi_k}[S(x_{k+1})] - S(x_k) \;\leq\; -\alpha_k \|\nabla S(x_k)\|^2 + \frac{1}{2}\alpha_k^2 L_{\nabla S} E_{\xi_k}[\|g(x_k;\xi_k)\|^2]$$

So we need such an assumption but we see that it can be weakened to

$\boxed{\text{A.1}}$ $\exists \mu_G \geq \mu > 0 : \forall k$

$$\begin{aligned} \nabla S(x_k)^\top E_{\xi_k}[g(x_k;\xi_k)] \;&\geq\; \mu \|\nabla S(x_k)\|^2 \\ \|E_{\xi_k}[g(x_k;\xi_k)]\| \;&\leq\; \mu_G \|\nabla S(x_k)\| \end{aligned}$$

We also see that we need to bound the second order moment of $g(x_k;\xi_k)$. We will see that is enough to also impose:

$$\boxed{\textcircled{\text{A.2}}\ \exists M, M_V \geq 0 : \forall k \qquad V_{\xi_k}\left[g(x_k;\xi_k)\right] \ \leq\ M + M_V \|\nabla S(x_k)\|^2}$$

In fact from A.2 and A.1 (second inequality) we obtain

$$(**) \quad \boxed{\begin{array}{rcl} E_{\xi_k}\left[\|g(x_k;\xi_k)\|^2\right] & = & V_{\xi_k}\left[g(x_k;\xi_k)\right] + \|E_{\xi_k}\left[g(x_k;\xi_k)\right]\|^2 \\ & \leq & M + (\underbrace{\mu_V + \mu_G^2}_{=M_G \geq \mu^2 > 0})\|\nabla S(x_k)\|^2 \end{array}}$$

Finally, we will also assume that $\{x_k\}$ is contained in a set where S is bounded (say by S_{low})

Using (*) and A.1 and A.2

$$(\square) \quad \boxed{\begin{aligned} E_{\xi_k}[S(x_{k+1})] - S(x_k) \;\leq\; & -\left(\mu - \tfrac{1}{2}\alpha_k L_{\nabla S} M_G\right) \alpha_k \|\nabla S(x_k)\|^2 \\ & + \tfrac{1}{2}\alpha_k^2 L_{\nabla S} M \end{aligned}}$$

Proof.

First (*) then A.1 (first inequality), then (**). □

NOTES:

1. The upper bound on $E_{\xi_k}[S(x_{k+1})] - S(x_k)$ is deterministic.
2. The random variable w_{k+1} depends only on w_k, ξ_k, and α_k, and not on any past iterates (Markovian behavior).
3. The first term in the upper bound is negative for small α_k, yielding a decrease proportional to $\|\nabla S(x_k)\|^2$. However, the second term could be large enough to allow for an increase...

The analysis of the GSG method is first established for strongly convex obj. functions S (recall from Chapter 1)

$$☆ \quad \exists \mu_S > 0 : S(\bar{x}) \;\geq\; S(x) + \nabla S(x)^\top (\bar{x} - x) + \frac{\mu_S}{2}\|\bar{x} - x\|^2 \quad \forall x, \bar{x}$$

Let x_* be the unique minimizer of S and $S_* = S(x_*) = S_{low}$.

Strong convexity of S implies

$$(\$) \quad \boxed{2\mu_S(S(x) - S_*) \;\leq\; \|\nabla S(x)\|^2 \quad \forall x}$$

The proof is left as an EXERCISE. Hint: Use $\star$ with $\bar{x} = x_*$ and note that $x - \frac{1}{\mu_S}\nabla S(x)$ is the minimizer of the quadratic on the right-hand-side with optimal value $S(x) - \frac{1}{2\mu_S}\|\nabla S(x)\|^2$.

Remember also from Chapter 1 that $L_{\nabla S} \geq \mu_S$.

In the next statements and proofs $E[\cdot]$ will denote the expected value taken w.r.t. the joint distribution of all random variables.

Example: x_k is completely determined by all realizations of $\xi_0, \ldots, \xi_{k-1}$. Hence, the total expectation of $S(x_k)$ is

$$E[S(x_k)] \;=\; E_{\xi_0} E_{\xi_1} \ldots E_{\xi_{k-1}}[S(x_k)]$$

We first state the result for a fixed stepsize. It is only possible to prove convergence for a neighborhood of the optimal value as, despite $\nabla S(x_k) \to 0$, the second term in the RHS of ($\square$) may remain constant.

If $\alpha_k = \bar{\alpha}\ \forall k$ with $0 < \bar{\alpha} \leq \frac{\mu}{L_{\nabla S} M_G}$, then

$$E[S(x_k) - S_*] \ \leq \ \frac{\bar{\alpha} L_{\nabla S} M}{2\mu_S \mu} + (1 - \bar{\alpha}\mu_S\mu)^k \left(S(x_0) - S_* - \frac{\bar{\alpha} L_{\nabla S} M}{2\mu_S \mu} \right)$$

Hence, this upper bound on the expected optimality gap tends (as $k \to \infty$) to $\frac{\bar{\alpha} L_{\nabla S} M}{2\mu_S \mu}$.

Proof.

EXERCISE ... See Curtis et al. 2018. $\square$

One recovers the context of the (deterministic) gradient method when:

1. $g(x_k;\xi_k)$ is an unbiased estimate of $\nabla S(x_k)$ ($\mu_G = 1$) and there is no noise in $g(x_k;\xi_k)$ ($M_V = 0$), thus $M_G = 1$ and $\bar{\alpha} \in (0, 1/L_{\nabla S}]$.
2. There is no noise in $g(x_k;\xi_k)$ ($M = M_V = 0$) or the noise decays with $\|\nabla S(x_k)\|^2$ ($M = 0$), thus obtaining a linear rate of convergence.

When $g(x_k;\xi_k)$ is noisy one still obtains a linear rate until the noise ($M > 0$) presents further progress.

Selecting a smaller step size $\bar{\alpha}$ allows then to arrive closer to the optimal value but worsens the contraction in the linear rate.

This suggests a strategy to reduce $\bar{\alpha}$ when progress stalls (see Curtis et al. 2018).

We now state and prove a sublinear rate of convergence of the expected optimality gap for diminishing stepsizes, i.e., α_k satisfying

$$\sum_{k=0}^{\infty} \alpha_k \; = \; \infty \quad \text{and} \; \sum_{k=0}^{\infty} \alpha_k^2 \; < \; \infty$$

(as in the original work of Robbins and Monro).

If $\alpha_k L_{\nabla S} M_G \leq \alpha_0 L_{\nabla S} M_G \leq \mu$ for all k, then from ($\Box$) and ($\$$)

$$\begin{aligned} E_{\xi_k}[S(x_{k+1})] - S(x_k) \; &\leq \; -\frac{1}{2}\alpha_k \mu \|\nabla S(x_k)\|^2 + \frac{1}{2}\alpha_k^2 L_{\nabla S} M \\ &\leq \; -\alpha_k \mu_S \mu (S(x_k) - S_*) + \frac{1}{2}\alpha_k^2 L_{\nabla S} M \end{aligned}$$

and subtracting S_* and taking total expectations

$$(\times) \quad \boxed{E[S(x_{k+1}) - S_*] \; \leq \; (1 - \alpha_k \mu_S \mu) E[S(x_k) - S_*] + \tfrac{1}{2}\alpha_k^2 L_{\nabla S} M}$$

From this point one can easily prove the main result.

If $\alpha_k = \frac{\beta}{k+\gamma}$ with $\gamma > 0$, $\beta > \frac{1}{\mu_G \mu}$ and $\alpha_0 \leq \frac{\mu}{L_{\nabla S} \mu_G}$, then

$$E[S(x_k) - S_*] \ \leq \ \frac{\nu}{k+\gamma}$$

where $\nu = \max\left\{ \frac{\beta^2 L_{\nabla S} M}{2(\beta \mu_G \mu - 1)}, \gamma(S(x_0) - S_*) \right\}$

The expected optimality gap converges to zero at a sublinear rate of $1/k$.

Proof.

From the definition of ν

$$\begin{aligned} E(S(x_0) - S_*) &= S(x_0) - S_* \\ &= \gamma \frac{S(x_0) - S_*}{\gamma} \ \leq \ \frac{\nu}{0+\gamma} \end{aligned}$$

and the result holds for $k = 0$.

Proof.

The result is thus proved by induction. It follows from $(\times)$ and $\alpha_k = \frac{\beta}{k+\gamma} \equiv \frac{\beta}{\hat{k}}$

$$\begin{aligned}
E\left[S(x_{k+1}) - S_*\right] &\le \left(1 - \frac{\beta\mu_G\mu}{\hat{k}}\right)\frac{\nu}{\hat{k}} + \frac{\beta^2 L_{\nabla S} M}{2\hat{k}^2} \\
&= \left(\frac{\hat{k} - \beta\mu_G\mu}{\hat{k}^2}\right)\nu + \frac{\beta^2 L_{\nabla S} M}{2\hat{k}^2} \\
&= \left(\frac{\hat{k} - 1}{\hat{k}^2}\right)\nu \underbrace{- \left(\frac{\beta\mu_G\mu - 1}{\hat{k}^2}\right)\nu + \frac{\beta^2 L_{\nabla S} M}{2\hat{k}^2}}_{\le 0 \text{ from the definition of } \nu} \\
&\le \frac{\nu}{\hat{k}}
\end{aligned}$$

The last inequality follows from $\hat{k} - 1 \le \hat{k}$. $\square$

Strong convexity played a major role in contracting the expected optimality gap.

The strong convexity parameter μ_G did not constrain the step size $\bar{\alpha}$ for fixed step sizes but for diminishing stepsizes one can see that β has to be larger than $1/(\mu_G \mu)$, and this was critical to get the $1/k$ rate.

See Curtis et al. 2018 for:

- Role of the initial point.
- Trade-offs of mini-batching.

Many problems in deep learning exhibit a nonconvex function S. One can still prove convergence of the GSG method, but now only towards first order stationarity.

Here are such results for fixed and diminishing stepsizes.

If $\alpha_k = \bar{\alpha} \quad \forall k$ with $0 < \bar{\alpha} \leq \frac{\mu}{LM_G}$, then the expected average-squared gradients of S satisfy

$$E\left[\frac{1}{k}\sum_{k=1}^{k}\|\nabla S(x_k)\|^2\right] \;\leq\; \frac{\bar{\alpha}L_{\nabla S}M}{\mu} + 2\frac{(S(x_0) - S_{low})}{k\mu\bar{\alpha}}$$

and this upper bound tends to $\frac{\bar{\alpha}L_{\nabla S}M}{\mu}$ when $k \to \infty$.

Note that the lower bound S_{low} on $S(x_k)$ appears in the bound.

Remember that $\min\limits_{0 \leq i \leq k-1} \|\nabla f(x_k)\|$ converged to zero at a rate of $1/\sqrt{k}$ for the deterministic gradient method (Chapter 4). When $M = 0$ we somehow recover this result.

Similar conclusions as in the strongly convex case can be drawn here regarding how the choice of $\bar{\alpha}$ impacts the two terms in the lower bound.

If $\sum_{k=0}^{\infty} \alpha_k = \infty$ and $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$, then

$$\lim_{K\to\infty} E\left[\frac{1}{A_K}\sum_{k=0}^{K} \alpha_k \|\nabla S(x_k)\|^2\right] \;=\; 0$$

where $A_K = \sum_{k=0}^{K} \alpha_k$.

In the case of diminishing stepsizes the expected norm of gradient cannot stay bounded away from zero. In fact, a consequence of the above result is that

$$\liminf_{k\to\infty} E\left[\|\nabla S(x_k)\|^2\right] \;=\; 0.$$

It is possible to prove convergence of $\|\nabla S(x_k)\|$ to zero in probability and to have $\lim$ instead of $\liminf$ under stronger assumptions (see Curtis et al. 2018.

Presentation outline

SG suffers from noisy gradient estimates. Besides, the proved rate was only sublinear (for diminishing stepsizes). A fixed stepsize can prevent convergence.

Noise reduction methods have been developed to reduce the error in gradient estimates and/or sequence of iterates.

The three main classes of noise reduction methods are:

Dynamic sampling (increase the mini-batch size)

Gradient aggregation (store and update gradient estimates of previous iterates, define search direction as weighted averages of these estimates)

Both achieve a linear rate for a fixed step size (for the empirical risk, assumed strongly convex).

Iterative averaging (maintain an average of previous iterates)

It remains sublinear, $\mathcal{O}(1/k)$.

We will follow again here the main lines in Bottou, Curtis, and Nocedal 2018.

Recall (valid for both expected risk $S = R$ and empirical risk $S = R_N$):

$$\begin{aligned} E_{\xi_k}\left[S(x_{k+1})\right] - S(x_k) \;&\le -\alpha_k \nabla S(x_k)^\top E_{\xi_k}\left[g(x_k;\xi_k)\right] \\ &\quad + \frac{1}{2}\alpha_k^2 L_{\nabla S} E_{\xi_k}\left[\|g(x_k;\xi_k)\|^2\right] \end{aligned}$$

If $-g(x_k;\xi_k)$ was a descent direction in expectation (first term negative), we see that by decreasing $E_{\xi_k}\left[\|g(x_k;\xi_k)\|^2\right]$ like $\|\nabla S(x_k)\|^2$ would, then we can achieve similar rates as in the full or batch gradient method.

In fact, assuming ∇S Lipschitz continuous, S strongly convex, A.1 and A.2, one does achieve a linear rate if the variances of the stochastic vectors decreases geometrically:

If $\bar{\alpha}_k = \alpha \, \forall k$ with $0 < \bar{\alpha} \leq \left\{ \frac{\mu}{L_{\nabla S} \mu_G^2}, \frac{1}{\mu_S \mu} \right\}$ and if

$$\exists M > 0 \exists \xi \in (0,1) \quad V_{\xi_k}[g(x_k;\xi_k] \ \leq \ M\xi^k \quad \forall k \quad \textcircled{\small V}$$

then

$$E[S(x_k) - S_*] \ \leq \ \omega \rho^k$$

with

$$\omega \ = \ \max \left\{ \frac{\bar{\alpha} L_{\nabla S} M}{\mu_s \mu}, S(x_0) - S_* \right\}$$

and

$$\rho \ = \ \max \left\{ 1 - \frac{\bar{\alpha} \mu_S \mu}{2} \xi \right\} \ < \ 1$$

The proof is as follows:

Using $\textcircled{V}$ and $\alpha_k = \bar{\alpha}$

$$\begin{aligned} E_{\xi_k}\left[S(x_{k+1})\right] - S(x_k) \;&\leq\; -\mu\bar{\alpha}\|\nabla S(x_k)\|^2 \\ &\quad + \frac{1}{2}\bar{\alpha}^2 L_{\nabla S}\left(\mu_G^2\|\nabla S(x_k)\|^2 + M\xi^k\right) \end{aligned}$$

Using the value of $\bar{\alpha}$ and strong convexity

$$\begin{aligned} &= -\left(\mu - \frac{1}{2}\bar{\alpha}L_{\nabla S}\mu_G\right)\bar{\alpha}\|\nabla S(x_k)\|^2 + \frac{1}{2}\bar{\alpha}L_{\nabla S}M\xi^k \\ &\leq -\frac{1}{2}\mu\bar{\alpha}\|\nabla S(x_k)\|^2 + \frac{1}{2}\bar{\alpha}L_{\nabla S}M\xi^k \\ &\leq -\bar{\alpha}\mu_S\mu(S(x_k) - S_*) + \frac{1}{2}\bar{\alpha}L_{\nabla S}M\xi^k \end{aligned}$$

Introducing S_* and taking total expectation yield

$$E[S(x_{k+1}) - S_*] \;\leq\; (1 - \bar{\alpha}\mu_S\mu)E[S(x_k) - S_*] + \frac{1}{2}\bar{\alpha}L_{\nabla S}M\xi^k$$

and the proof can be concluded by induction.

Dynamic Sample Size Methods

Consider the minibatch method with fixed stepsize of the type

$$\boxed{\begin{aligned} g(x_k;\xi_k) \;&=\; \frac{1}{|B_k|}\sum_{j\in B_k}\nabla f(x_k;\xi_{k,j}) \\ &\text{with } |B_k| \;=\; \lceil \tau^k \rceil \;\text{ and }\; \tau > 1 \\ x_{k+1} \;&=\; x_k - \bar{\alpha} g(x_k;\xi_k) \end{aligned}}$$

We now assume that the expectation of each stochastic gradient $\nabla f(x_k;\xi_{k,j})$ is $\nabla S(x_k)$. Hence, $\textcircled{\text{A.1}}$ is satisfied with $\mu = \mu_G = 1$.

If in addition, $\{\xi_{k,j}\}_{j \in B_k}$ are drawn independently according to P_Ω and if $\exists M > 0 \; V_{\xi_k}[\nabla f(x_k;\xi_{k,j})] \leq M$, then

$$\begin{aligned} V_{\xi_k}[g(x_k;\xi_k)] &= \frac{1}{|B_k|} \sum_{j \in B_k} V_{\xi_k}[\nabla f(x_k;\xi_{k,j})] \\ &\leq \frac{M}{|B_k|} \leq M \left\lfloor \frac{1}{\tau} \right\rfloor^k \end{aligned}$$

One can now apply the previous result and conclude that the minibatch with unbiased gradient estimates (and $|B_k| = \lceil \tau^k \rceil, \tau > 1$) drives the optimal value linearly to zero, for strongly convex functions.

We have seen that SG (one gradient evaluation per iteration) has a total effort of $\mathcal{O}(\epsilon^{-1})$ to achieve $E[S(x_k) - S_*]$ in the strongly convex case (rate $1/k$).

One can select τ appropriately, $\tau \in \left(1, \left(1 - \frac{\bar{\alpha}\mu_S\mu}{2}\right)^{-1}\right]$, to make the minibatch achieve a similar bound $\mathcal{O}(\epsilon^{-1})$. (See the details in Curtis et al. 2018, and the discussion before Subsection 5.2.1.)

Prechoosing τ before a run of the algorithm or choosing it dynamically may be difficult in practice.

An alternative is to choose the sample size adaptively according to the optimization process. Here is such an idea (see references in Curtis et al. 2018):

First observe that $g(x_k;\xi_k)$ is a descent direction for S at x_k if, for some $\chi \in [0,1)$,

$$\delta(x_k;\xi_k) \ \equiv \ \|g(x_k;\xi_k) - \nabla S(x_k)\| \ \leq \ \chi \|g(x_k;\xi_k)\|. \quad \otimes$$

In fact, since $\|\nabla S(x_k)\| \geq (1-\chi)\|g(x_k;\xi_k)\|$ one has

$$\begin{aligned} \nabla S(x_k)^\top g(x_k;\xi_k) \ &\geq \ \frac{1}{2}(1-\chi^2)\|g(x_k;\xi_k)\|^2 + \|\nabla S(x_k)\|^2 \\ &\geq \ \frac{1}{2}\left(1-\chi^2 + (1-\chi^2)\right)\|g(x_k;\xi_k)\|^2 \\ &= \ (1-\chi)\|g(x_k;\xi_k)\|^2 \end{aligned}$$

Then, if we assume that $g(x_k;\xi_k)$ is an unbiased gradient estimate

$$E\left[\delta(x_k;\xi_k)^2\right] \;=\; E\left[\|g(x_k;\xi_k)-\nabla S(x_k)\|^2\right] \;=\; V_{\xi_k}[g(x_k;\xi_k)]$$

This quantity can in turn be approximated by

$$V_{\xi_k}[\delta(x_k;\xi_k] \;\simeq\; \frac{(\mathrm{Cov}\,(\{\nabla f(x_k;\xi_{k,j})\}_{j\in B_k}))}{|B_k|} \;\equiv\; \varphi_k$$

An adaptive sampling algorithm would test (see $\otimes$)

$$\varphi_k \;\leq\; \chi^2\|g(x_k;\xi_k)\|^2$$

in conjunction with $|B_k|$ increasing at a geometric rate.

Gradient Aggregation

These methods achieve a lower variance by reusing/revisiting past computed information. They converge linearly for strongly convex empirical risks.

SVRG (stochastic variance reduced gradient) applied to $S = R_N$
Inner cycle:

j_l chosen randomly

$$\tilde{g}_l = \nabla f_{j_l}(\tilde{x}_l) - \left[\underbrace{\nabla f_{j_l}(x_k) - \nabla R_N(x_k)}_{\text{diff}}\right]$$

$$\tilde{x}_l = \tilde{x}_l - \alpha \tilde{g}_l$$

Since the expected value of $\nabla f_{j_l}(x_k)$ is $\nabla R_N(x_k)$, diff is the bias in $\nabla f_{j_l}(x_k)$.

$\tilde{g}_l$ is an unbiased estimator of $\nabla R_N(\tilde{x}_l)$ but with variance expected to be lower than $\nabla f_{j_l}(\tilde{x}_l)$.

Here is the main algorithm:

```
For $k = 0, 1, \ldots$
    Compute batch gradient $\nabla R_N(x_k)$
    Initialize $\tilde{x}_1 = x_k$.
    For $l = 1, \ldots, m$
        $\boxed{\text{inner cycle}}$
    end
    $x_{k+1} = \tilde{x}_{m+1}$ or $x_{k+1} = \sum_{l=1}^{m} \tilde{x}_{l+1}$ or $x_{k+1} = \tilde{x}_{i+1}$ ($i$ chosen
    uniformly in $\{1, \ldots, m\}$)
```

The linear rate is achieved for $S = R_N$ strongly convex choosing m and $\bar{\alpha}$ such that (Johnson and Zhang 2013):

$$\rho \;\equiv\; \frac{1}{1 - 2\bar{\alpha} L_{\nabla S}} \left(\frac{1}{m \mu_S \bar{\alpha}} + 2 L_{\nabla S} \bar{\alpha} \right) \;<\; 1$$

In practice m and $\bar{\alpha}$ are chosen by experimentation.

The result is ($S = R_N$)

$$E[S(x_{k+1}) - S_*] \;\leq\; \rho E[S(x_k) - S_*]$$

SAGA (stochastic average gradient approximation)

Defazio, Bach, Lacoste-Julien 2014

It computes at each iteration an average of stochastic gradients evaluated before.

Gradient computation:

```
Choose $l$ randomly in $\{1, \ldots, N\}$

Compute $\nabla f_l(x_k)$

$g_k = \nabla f_l(x_k) - \nabla f_l(x_{[l]}) + \frac{1}{N} \sum_{j=1}^{N} \nabla f_j(x_{[j]})$
```

Again one has $E[g_k] = \nabla R_N(x_k)$, but with variance expected lower than in SG.

SAGA requires an initialization phase but then the cost per iteration is as in SG.

```
For $j = 1, \ldots, N$
    Compute $\nabla f_j(x_0)$ and store $\nabla f_j(x_{[j]}) = \nabla f_j(x_0)$
end
For $k = 0, 1, \ldots$
    $\boxed{\text{Gradient computation}}$
    Store $\nabla f_l(x_{[l]}) = \nabla f_l(x_k)$
    $x_{k+1} = x_k - \alpha g_k$
```

For a strongly convex $S = R_N$, taking

$$\bar{\alpha} = \frac{1}{2(\mu_S N + L_{\nabla S})},$$

one has

$$E\left(\|x_k - x_*\|^2\right) \leq \left(1 - \frac{\mu_S}{2(\mu_S N + L_{\nabla S}}\right)^{k+1} \left[\|x_0 - x_*\|^2 + \frac{ND}{\mu_S N + L_{\nabla S}}\right]$$

where $D = S(x_0) - \nabla S(x_*)^\top (x_0 - x_*) - S(x_*)$.

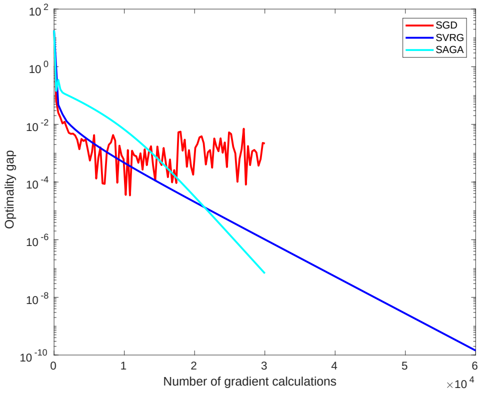
Again the choice of $\bar{\alpha}$ depends on the unknown constants μ_S and $L_{\nabla S}$. (The choice $\bar{\alpha} = 1/(3L_{\nabla S})$ works when μ_S is not known.)

NOTES:

1. Storing N gradients could be prohibitive, but in logistic and least squares regression
$$\nabla_j f(x_k) \;=\; \hat{f}'(y_j^\top x_k) y_j \quad (\hat{f} = f \text{ along } y_j)$$
and the features $\{y_1, \ldots, y_N\}$ are already stored.
2. In the precursor SAG (stochastic average gradient), g_k is biased for $\nabla R_N(x_k)$ but the linear rate is preserved:
$$g_k \;=\; \frac{1}{N}\left(\nabla f_l(x_k) + \nabla f_l(x_{[l]}) + \sum_{j=1}^{N} \nabla f_j(x_{[j]})\right)$$
3. In all these methods (SVRG, SAGA, SAG) the complexity bound is of the form
$$\mathcal{O}\left(\left[N + \frac{L_{\nabla S}}{\mu_S}\right] \log(1/\epsilon)\right)$$

For very large N, gradient aggregation is comparable to batch and it becomes difficult to beat SG. But it may be better when $N >$ (or $\gg$) $\frac{L_{\nabla S}}{\mu_S}$.

Comparison among SG, SVRG, and SAGA algorithms on a logistic regression problem with randomly generated 300 samples in dimension 3:



- Both SVRG and SAGA perform better than SG.
- Linear convergence rate of SVRG and SAGA is observed.

Iterative Averaging Methods

SG generates noisy iterates which may tend to oscillate around minimizers during an optimization run.

A natural idea is then to average the iterates:

$$\boxed{\begin{aligned} x_{k+1} &= x_k - \alpha_k g(x_k;\xi_k) \\ \tilde{x}_{k+1} &= \frac{1}{k+1}\sum_{i=1}^{k+1} x_i \end{aligned}}$$

Note that $\{\tilde{x}_{k+1}\}$ has no effect on $\{x_k\}$.

Choosing a (longer) diminishing stepsize of $\mathcal{O}(1/k^a)$, $a \in (\frac{1}{2}, 1)$, it is possible to attain a rate of $E(\|\tilde{x}_k - x_*\|^2) = \mathcal{O}(1/k)$ for strongly convex functions (which only $E(\|x_k - x_*\|^2) = \mathcal{O}(1/k^a)$)

Iterative averaging has led to various other schemes with longer stepsizes (robust SA, mirror descent SA, primal-dual averaging).

Presentation outline

Presentation outline

References:

Part of the organization of the material of Chapters 1 and 6 is inspired from

- S. J. Wright, *Optimization algorithms for data analysis*, IAS/Park City Mathematics Series, 2017.
- M. Pfetsch and S. Ulbrich, *Optimization Methods for Machine Learning*, Lecture Notes, TU Darmstadt, Winter Term 2016/17.

The presentation of the subgradient methods and the proof of the rate of convergence of FISTA follows

- A. Beck, *First-Order Methods in Optimization*, MPS-SIAM Book Series on Optimization, SIAM, Philadelphia, 2017.

This books was very helpful in other places referenced in our notes.

The proof of the rates of convergence of proximal gradient methods is based on:

- L. Vandenberghe, *EE236C – Optimization Methods for Large-Scale Systems (Spring 2016)*, UCLA, 2016.

The assumptions and proofs related to the rates of convergence of stochastic gradient descent and noisy reduction methods are taken from:

- L. Bottou, F. E. Curtis, and J. Nocedal, *Optimization Methods for Large-Scale Machine Learning*, Related Databases, SIAM Review, 60 (2018) 223–311.

Other references that helped us:

- F. E. Curtis and K. Scheinberg, *Optimization Methods for Supervised Machine Learning: From Linear Models to Deep Learning*, in INFORMS Tutorials in Operations Research, Chapter 5, pages 89–114, 2017.
- K. Scheinberg, *Evolution of randomness in optimization methods for supervised machine learning*, SIAG/OPT Views and News, Vol. 24, Num. 1, 2016.