A Generic Acceleration Framework for Stochastic Composite Optimization

Andrei Kulunchakov and Julien Mairal - Inria Grenoble

Goal

Problem: **acceleration** of convex optimization methods for minimization of the **expected risk**.

Developed tool: **generic framework** using iterative construction and minimization of surrogate functions.

Motivation and context

- **Common approach in ML:** approximate the expected risk by a finite sum. But this creates an additional bias of approximation.
- If infinite amount of data is available, one can minimize the **expected risk**, requiring stochastic optimization. Standard fast methods may be unstable or slow in this case.
- Having access only to **approximate gradients**, we substantially accelerate/stabilize these methods by the framework.

Detailed statement of the problem

$$\min_{x\in\mathbb{R}^p}\left\{F(x)=\frac{1}{n}\sum_{i=1}^{n}f_i(x)+\psi(x)\right\} \quad \text{with stochastic terms } f_i(x),$$

which are μ-strongly convex and L-smooth and ψ is convex. When $n=1$, we recover classical composite optimization problems. For each i, the gradient is noisy $\tilde{\nabla} f_i(x)=\nabla f_i(x)+\xi_i$.

Contributions in examples

There are **mild requirements** on the convergence of methods that we want to accelerate: they even **can be biased**!

Examples of acceleration are presented below with $\Delta_0=F(x_0)-F^*$, $\Omega^2=\|x_0-x^*\|^2$ and a targeted accuracy ε.

Deterministic cases:

$$(L/\mu)\log(\Delta_0/\varepsilon) \quad \text{becomes} \quad \sqrt{L/\mu}\log(\Delta_0/\varepsilon),$$
$$(n+L/\mu)\log(\Delta_0/\varepsilon) \quad \text{becomes} \quad \left(n+\sqrt{nL/\mu}\right)\log(\Delta_0/\varepsilon);$$

and **stochastic cases**:

$$(L/\mu)\log(\Delta_0/\varepsilon) \quad \text{biased as } \varepsilon\geq\sigma^2/\mu$$
$$\text{becomes convergent} \quad \sqrt{L/\mu}\log(\Delta_0/\varepsilon)+\sigma^2/\mu\varepsilon, \tag{1}$$

and similarly biased

$$(n+L/\mu)\log(\Delta_0/\varepsilon) \quad \text{becomes convergent} \quad \left(n+\sqrt{nL/\mu}\right)\log(\Delta_0/\varepsilon)+\sigma^2/\mu\varepsilon. \tag{2}$$

Original (deterministic) approach called Catalyst

- Given $\kappa>0$, for each k we build a **surrogate function**
$$h_k(x)\triangleq F(x)+(\kappa/2)\|x-y_{k-1}\|^2.$$
- Enjoying better properties than $F(x)$, it is effectively minimized.
- Iterative minimization of $h_1,h_2,\ldots,h_k$ by some method $\mathcal{M}$, allows to accelerate $\mathcal{M}$ **on minimization of F(x)**.

But, this is only for **deterministic cases**.

Our approach. Stochastic surrogates

We allow **greater flexibility** to surrogate functions h_k:

- h_k is $(\kappa+\mu)$-strongly convex;
- $\mathbb{E}[h_k(x)|\mathcal{F}_{k-1}]\leq F(x)+(\kappa/2)\|x-y_{k-1}\|^2$ for some α_{k-1};
- $\mathcal{M}$ "knows" the exact minimizer x_k^* of h_k and a point x_k such that
$$\mathbb{E}[F(x_k)]\leq\mathbb{E}[h_k^*]+\delta_k \quad \text{with } \delta_k>0.$$

First example of a surrogate

Given g_k as a stochastic realization of $\nabla f(y_{k-1})$, we consider

$$h_k(x):=f(y_{k-1})+g_k^\top(x-y_{k-1})+\frac{\mu+\kappa}{2}\|x-y_{k-1}\|^2+\psi(x),$$

with the exact minimizer $x_k^*=\mathrm{Prox}_{\psi/(\mu+\kappa)}[y_{k-1}-g_k/(\mu+\kappa)]$.

Algorithm [1] (exact minimization)

FOR $k=1,\ldots,K$ **DO**

— using a fixed improvable $\mathcal{M}$, obtain x_k and x_k^*;

— update the extrapolated sequence

$$y_k=x_k^*+\beta_k(x_k^*-x_{k-1})+\frac{(\kappa+\mu)(1-\alpha_k)}{\kappa}(x_k-x_k^*), \tag{3}$$

where α_k,β_k are from standard Nesterov extrapolation technique.

OUTPUT: x_k (final estimate).

As a result, **new accelerated SGD algorithm**, converging as (1).

Algorithm [2] (inexact minimization)

In some situations, the surrogate function h_k is such that x_k^* **is not available**, for example when

$$h_k(x):=F(x)+(\kappa/2)\|x-y_{k-1}\|^2,$$

Then, in Equation (3) we use x_k instead of x_k^*. This results in **new multi-stage algorithms** with the convergence (2).

Advertisement for post-doc positions in Grenoble

in machine learning, optimization, computer vision, and ski.

Basic restart schemes

In order to get converging algorithms out of Alg. [1] and [2], we sometimes address to **restart procedure with mini-batching**:

• at stage k, choose a target accuracy $\varepsilon_k=\varepsilon_{k-1}/2$
• set up a mini-batch of size $b_k=2b_{k-1}$ to sample gradients, so that σ^2 becomes σ^2/b_k at the stage k;
• minimize the objective up to accuracy ε_k using $O(b_k/\tau)$ steps of $\mathcal{M}$ and using previous solution as a "warm start".

Examples of final improvements

We provide practical choices for κ for different algorithms dealing with stochastic perturbations.

Method $\mathcal{M}$	κ	τ	B	q	Final convergence rate
prox-SGD	$L-\mu$	$\frac{1}{2}$	$\frac{1}{L}$	$\frac{\mu}{L}$	$\tilde{O}\left(\sqrt{\frac{L}{\mu}}\log\left(\frac{F_0-F^*}{\varepsilon}\right)+\frac{\sigma^2}{\mu\varepsilon}\right)$
SAGA/MISO/SVRG	$\frac{L}{n}-\mu$	$\frac{1}{n}$	$\frac{1}{L}$	$\frac{\mu n}{L}$	$\tilde{O}\left(\sqrt{n\frac{L}{\mu}}\log\left(\frac{F_0-F^*}{\varepsilon}\right)+\frac{\sigma^2}{\mu\varepsilon}\right)$

We see that we breach the **optimal linear bias** part, while preserving **optimal robustness** to noise.

Experiments on logistic regression ($\sigma^2=0$ on top)

- (left) CIFAR-10 represented by using a two-layer unsupervised convolutional neural network ($n=50000$).
- (center) dataset with gene expressions data and the binary labels ($n=295$);
- (right) Pascal Large Scale Learning Challenge ($n=250000$);