Acceleration of GD via Momentum

Axel Böhm

November 15, 2021

Smooth convex functions: less than $\mathcal{O}(\epsilon^{-1})$ steps?

Given L and $D = \|x_0 - x^*\|$ we know that **gradient descent**

- converges with $\mathcal{O}(1/k)$
- cannot go faster ("lower bound")

> Maybe GD is not the best possible algorithm?

After all, it is arguably the simplest possible method using the gradient.

Smooth convex functions: less than $\mathcal{O}(\epsilon^{-1})$ steps?

So let's look at the following classes of methods:

First-order method:

- Access to data only via an **oracle** returning f and ∇f at given points.
- Clearly, GD is a first order method.

Q: What is the best first-order method for smooth convex functions.

best: smallest upper bound on the number of oracle calls *in the worst case*.

- Nemirovski and Yudin 1979 proved that

every first-order method needs at least $\Omega(1/\sqrt{\epsilon})$ iterations
to find a point $\bar{x}$ with $f(\bar{x}) - f^* \leq \epsilon$.

$\Rightarrow$ no method can be faster than $\mathcal{O}(1/k^2)$

Acceleration to $\mathcal{O}(1/\sqrt{\epsilon})$ steps

- Nesterov 1983 proposed a method that needs only $\mathcal{O}(1/\sqrt{\epsilon})$ iterations (and is therefore the *best one*).
- Known as **Nesterov's accelerated gradient** method.
- By now multiple similar algorithms with same complexity exist.
- Proofs are generally not really instructive (some are computer assisted).

Nesterov's accelerated gradient method

Algorithm Nesterov's accelerated gradient method (NAG)

1: **for** $k = 0, 1, \ldots$ **do**
2: $\quad x_{k+1} = y_k - \frac{1}{L}\nabla f(y_k)$
3: $\quad z_{k+1} = z_k - \frac{k+1}{2L}\nabla f(y_k)$
4: $\quad y_{k+1} = \frac{k+1}{k+3}x_{k+1} + \frac{2}{k+3}z_{k+1}$

- perform "**smooth step**" from y_k to x_{k+1}
- perform **aggressive step** from z_k to z_{k+1}
- form **weighted average** of the two
  compensate for the aggressive step by giving less weight

Nesterov's algorithm as a momentum method

A different way to write the method is via momentum

$$y_k = x_k + \beta_k(x_k - x_{k-1})$$
$$x_{k+1} = y_k - \frac{1}{L}\nabla f(y_k).$$

- differs from GD on in momentum/inertia term $\beta_k(x_k - x_{k-1})$
- has to chosen carefully $\beta_k = \frac{k-1}{k+2}$
- coefficient approaches $\frac{k-1}{k+2} \approx 1 - \frac{3}{k}$

Nesterov's accelerated gradient method: convergence

Minimum is obtained for x^*.

Theorem

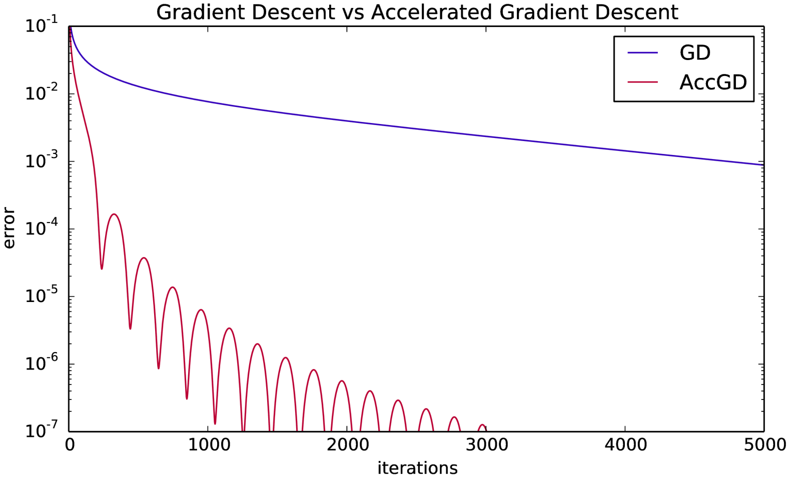
Let $f : R^d \to \mathbb{R}$ be convex and L-smooth, then ***NAG*** *yields*

$$f(x_k) - f(x^*) \leq \frac{2L\|x_0 - x^*\|^2}{\boldsymbol{k}(\boldsymbol{k}+1)}$$

Recall that the gradient descent bound was

$$f(x_k) - f(x^*) \leq \frac{L\|x_0 - x^*\|^2}{2k}.$$

$\mathcal{O}(1/k^2)$ vs $\mathcal{O}(1/k)$ in practice



Proof idea

Potential function Φ that decreases along trajectory (standard technique). Out of the blue: Use

$$\Phi(k) := k(k+1)(f(x_k) - f^*) + 2L\|z_k - x^*\|^2.$$

Then show that

$$\Phi(k+1) \leq \Phi(k).$$

Results in

$$\Phi(k+1) \leq \Phi(k) \leq \cdots \leq \Phi(0)$$

and therefore

$$k(k+1)(f(x_k) - f^*) \leq 2L\|z_0 - x^*\|^2.$$

Why momentum?

- GD has problems with **ravines**, i.e. areas where the surface curves much more steeply in one dimension than in another.
- Results in zig-zagging.

Figure: no momentum

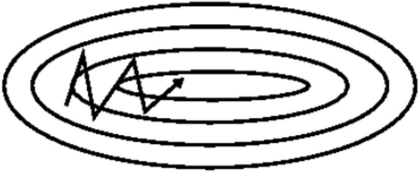
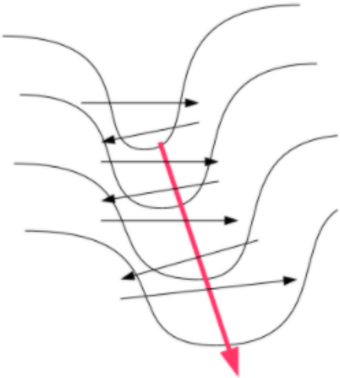
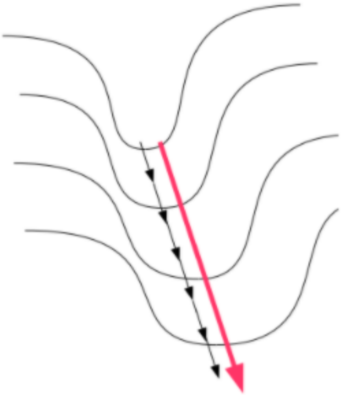


Figure: with momentum

Momentum and ravines



SGD bounces back and forth from one side of the valley to the other



Using Momentum the zig-zag cancels out, while the direction along the valley is reinforced

Momentum in terms of velocity

Consider a ball rolling down a slope. Its **velocity** is

$$
\begin{aligned}
v_k &= \beta v_{k-1} + \alpha \nabla f(x_k) \\
x_{k+1} &= x_k - v_k
\end{aligned}
$$

- a fraction β of the **previous velocity** (friction)
- plus, steepness of the **slope**

In terms of iterates:

$$
\begin{aligned}
x_{k+1} &= x_k - v_k \\
&= x_k - \alpha \nabla f(x_k) - \beta v_{k-1} \\
&= x_k - \alpha \nabla f(x_k) + \beta (x_k - x_{k-1})
\end{aligned}
$$

Heavy ball: Polyak 1964

We derived

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1}),$$

while Nesterov's method was

$$y_k = x_k + \beta_k(x_k - x_{k-1})$$
$$x_{k+1} = y_k - \frac{1}{L} \nabla f(y_k).$$

However, **Polyak's** momentum provides no speedup over $\mathcal{O}(1/k)$ (for smooth convex function).

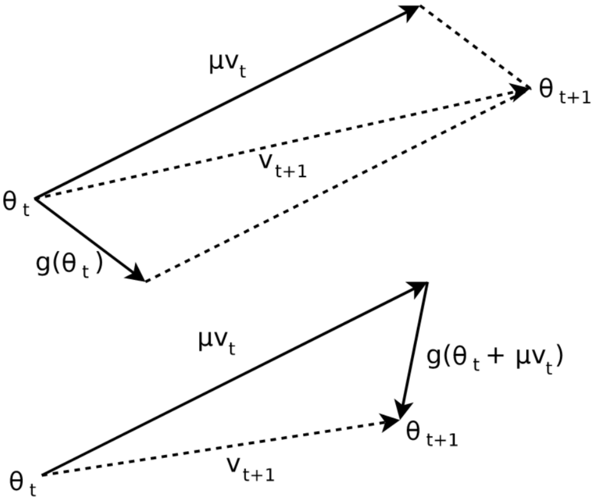
What's the difference?

- Both types of momentum seem so similar.
- Heavy ball does not care if do momentum or gradient first.
- Nesterov momentum applies **inertia first**, then gradient:

$$
\begin{aligned}
v_k &= \beta v_{k-1} + \alpha \nabla f(x_k + \beta v_{k-1}) \\
x_{k+1} &= x_k - v_k.
\end{aligned}
$$

Provides stabilization if inertia overshoots.

Nesterov vs Polyak momentum.



Momentum for strongly convex functions

For L-smooth μ-strongly convex we know that GD obtains

$$\|x_{k+1} - x^*\|^2 \leq \left(1 - \frac{1}{\kappa}\right) \|x_k - x^*\|^2$$

and

$$f(x_k) - f^* \leq \left(1 - \frac{1}{\kappa}\right)^k \frac{L\|x_0 - x^*\|^2}{2}.$$

Performance depends heavily on the **condition number** $\kappa := L/\mu$:

> Contraction coefficient is $(1 - 1/\kappa)$.

Nesterov **and Polyak** momentum improve this to $(1 - 1/\sqrt{\kappa})$

Momentum for stochastic methods

SGD analysis can be extended to **smooth** functions with rate

$$\mathcal{O}\left(\frac{L}{k} + \frac{\sigma^2}{\sqrt{k}}\right),$$

where $\sigma^2 := \mathbb{E}[\|\nabla f(x) - g(x)\|^2]$ is the variance of the gradient estimator.

This can be improved by momentum (and additional tricks) to

$$\mathcal{O}\left(\frac{L}{k^2} + \frac{\sigma^2}{\sqrt{k}}\right).$$

Improvement only in the "**transient phase**" before noise takes over.

For worst case rates, only the asymptotic phase matters.

Momentum and nonsmoothness

- If f is not differentiable and we have to use subgradients: no way to improve the $\mathcal{O}(1/\sqrt{k})$ rate
- If objective is **structured**: $f + g$ (smooth+nonsmooth)

$$
\begin{aligned}
y_k &= x_k + \beta_k(x_k - x_{k-1}) \\
x_{k+1} &= \operatorname{prox}_{\alpha g}\left(y_k - \alpha \nabla f(y_k)\right).
\end{aligned}
$$

- In particular, also works in the **constrained setting**.

Momentum in the nonconvex world

- In theory: difficult to show benefit of momentum in for nonconvex problems.
  - some statements under additional smoothness assumptions
- Empirical evidence of usefulness is strong.
  - especially in deep learning.
- Theory is mostly limited to escaping of saddle points.

Momentum in DL:

Docs > torch.optim > SGD

SGD

CLASS `torch.optim.SGD(params, lr=<required parameter>, momentum=0, dampening=0, weight_decay=0, nesterov=False)` [SOURCE]

Implements stochastic gradient descent (optionally with momentum).

---

input : γ (lr), θ_0 (params), $f(\theta)$ (objective), λ (weight decay),
μ (momentum), τ (dampening), *nesterov*

---

for $t = 1$ **to** $\ldots$ **do**
  $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$
  if $\lambda \neq 0$
    $g_t \leftarrow g_t + \lambda\theta_{t-1}$
  if $\mu \neq 0$
    if $t > 1$
      $\mathbf{b}_t \leftarrow \mu\mathbf{b}_{t-1} + (1-\tau)g_t$
    else
      $\mathbf{b}_t \leftarrow g_t$
    if *nesterov*
      $g_t \leftarrow g_{t-1} + \mu\mathbf{b}_t$
    else
      $g_t \leftarrow \mathbf{b}_t$
  $\theta_t \leftarrow \theta_{t-1} - \gamma g_t$

---

return θ_t
