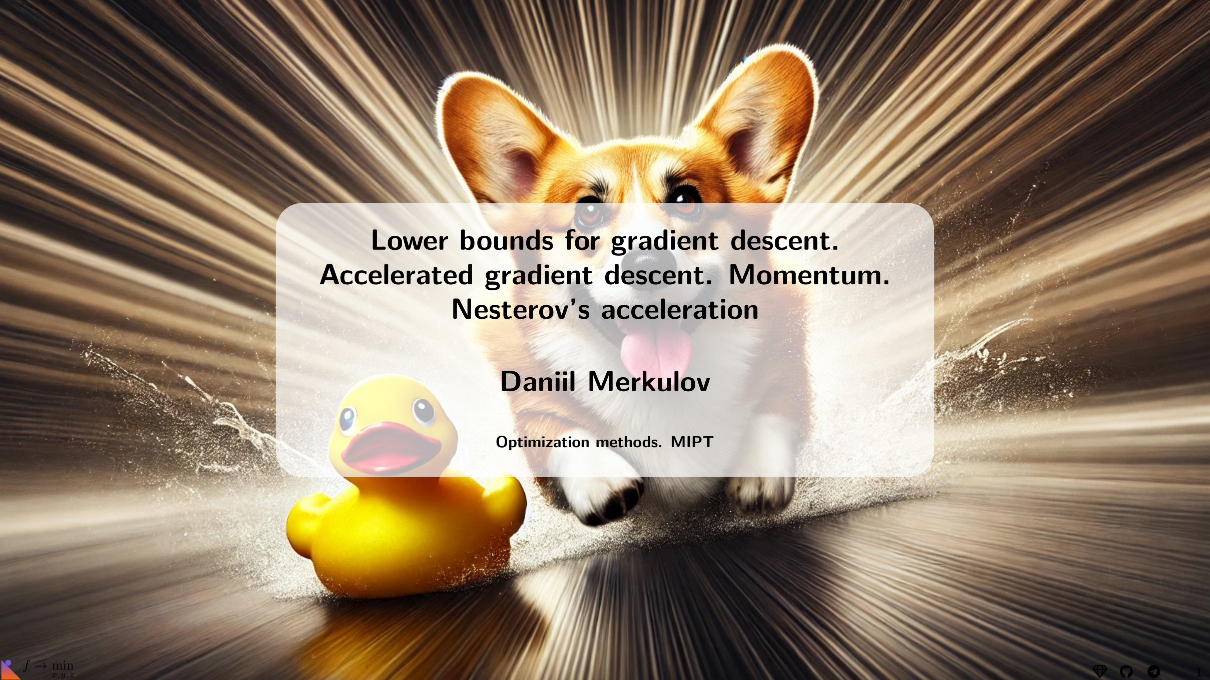


Lower bounds for gradient descent. Accelerated gradient descent. Momentum. Nesterov's acceleration

Daniil Merkulov

Optimization methods. MIPT

Recap of Gradient Descent convergence

Gradient Descent: $\min_{x \in \mathbb{R}^n} f(x)$ $\qquad x^{k+1} = x^k - \alpha^k \nabla f(x^k)$

convex (non-smooth)	smooth (non-convex)	smooth & convex	smooth & strongly convex (or PL)
$f(x^k) - f^* \sim \mathcal{O}\left(\dfrac{1}{\sqrt{k}}\right)$	$\|\nabla f(x^k)\|^2 \sim \mathcal{O}\left(\dfrac{1}{k}\right)$	$f(x^k) - f^* \sim \mathcal{O}\left(\dfrac{1}{k}\right)$	$\|x^k - x^*\|^2 \sim \mathcal{O}\left(\left(1 - \dfrac{\mu}{L}\right)^k\right)$
$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon^2}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\kappa \log \dfrac{1}{\varepsilon}\right)$

Recap of Gradient Descent convergence

Gradient Descent: $\min\limits_{x \in \mathbb{R}^n} f(x) \qquad x^{k+1} = x^k - \alpha^k \nabla f(x^k)$

convex (non-smooth)	smooth (non-convex)	smooth & convex	smooth & strongly convex (or PL)
$f(x^k) - f^* \sim \mathcal{O}\left(\dfrac{1}{\sqrt{k}}\right)$	$\|\nabla f(x^k)\|^2 \sim \mathcal{O}\left(\dfrac{1}{k}\right)$	$f(x^k) - f^* \sim \mathcal{O}\left(\dfrac{1}{k}\right)$	$\|x^k - x^*\|^2 \sim \mathcal{O}\left(\left(1 - \dfrac{\mu}{L}\right)^k\right)$
$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon^2}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\kappa \log \dfrac{1}{\varepsilon}\right)$

For smooth strongly convex we have:

$$
f(x^k) - f^* \leq \left(1 - \frac{\mu}{L}\right)^k (f(x^0) - f^*).
$$

Note also, that for any x

$$
1 - x \leq e^{-x}
$$

Recap of Gradient Descent convergence

Gradient Descent: $\min_{x \in \mathbb{R}^n} f(x)$ $\qquad x^{k+1} = x^k - \alpha^k \nabla f(x^k)$

convex (non-smooth)	smooth (non-convex)	smooth & convex	smooth & strongly convex (or PL)
$f(x^k) - f^* \sim \mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ $k_\varepsilon \sim \mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$	$\|\nabla f(x^k)\|^2 \sim \mathcal{O}\left(\frac{1}{k}\right)$ $k_\varepsilon \sim \mathcal{O}\left(\frac{1}{\varepsilon}\right)$	$f(x^k) - f^* \sim \mathcal{O}\left(\frac{1}{k}\right)$ $k_\varepsilon \sim \mathcal{O}\left(\frac{1}{\varepsilon}\right)$	$\|x^k - x^*\|^2 \sim \mathcal{O}\left(\left(1 - \frac{\mu}{L}\right)^k\right)$ $k_\varepsilon \sim \mathcal{O}\left(\kappa \log \frac{1}{\varepsilon}\right)$

For smooth strongly convex we have:

$$
f(x^k) - f^* \leq \left(1 - \frac{\mu}{L}\right)^k (f(x^0) - f^*).
$$

Note also, that for any x

$$
1 - x \leq e^{-x}
$$

Finally we have

$$
\begin{aligned}
\varepsilon = f(x^{k_\varepsilon}) - f^* &\leq \left(1 - \frac{\mu}{L}\right)^{k_\varepsilon} (f(x^0) - f^*) \\
&\leq \exp\left(-k_\varepsilon \frac{\mu}{L}\right) (f(x^0) - f^*) \\
k_\varepsilon \geq \kappa \log \frac{f(x^0) - f^*}{\varepsilon} &= \mathcal{O}\left(\kappa \log \frac{1}{\varepsilon}\right)
\end{aligned}
$$

Recap of Gradient Descent convergence

Gradient Descent: $\min_{x \in \mathbb{R}^n} f(x)$ $\quad x^{k+1} = x^k - \alpha^k \nabla f(x^k)$

convex (non-smooth)	smooth (non-convex)	smooth & convex	smooth & strongly convex (or PL)
$f(x^k) - f^* \sim \mathcal{O}\left(\dfrac{1}{\sqrt{k}}\right)$	$\\|\nabla f(x^k)\\|^2 \sim \mathcal{O}\left(\dfrac{1}{k}\right)$	$f(x^k) - f^* \sim \mathcal{O}\left(\dfrac{1}{k}\right)$	$\\|x^k - x^*\\|^2 \sim \mathcal{O}\left(\left(1 - \dfrac{\mu}{L}\right)^k\right)$
$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon^2}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\kappa \log \dfrac{1}{\varepsilon}\right)$

For smooth strongly convex we have:

$$
f(x^k) - f^* \leq \left(1 - \frac{\mu}{L}\right)^k (f(x^0) - f^*).
$$

Note also, that for any x

$$
1 - x \leq e^{-x}
$$

Finally we have

$$
\begin{aligned}
\varepsilon = f(x^{k_\varepsilon}) - f^* &\leq \left(1 - \frac{\mu}{L}\right)^{k_\varepsilon} (f(x^0) - f^*) \\
&\leq \exp\left(-k_\varepsilon \frac{\mu}{L}\right) (f(x^0) - f^*) \\
k_\varepsilon \geq \kappa \log \frac{f(x^0) - f^*}{\varepsilon} &= \mathcal{O}\left(\kappa \log \frac{1}{\varepsilon}\right)
\end{aligned}
$$

Question: Can we do faster, than this using the first-order information?

Recap of Gradient Descent convergence

Gradient Descent: $$\min_{x \in \mathbb{R}^n} f(x) \qquad x^{k+1} = x^k - \alpha^k \nabla f(x^k)$$

convex (non-smooth)	smooth (non-convex)	smooth & convex	smooth & strongly convex (or PL)
$f(x^k) - f^* \sim \mathcal{O}\left(\dfrac{1}{\sqrt{k}}\right)$	$\\|\nabla f(x^k)\\|^2 \sim \mathcal{O}\left(\dfrac{1}{k}\right)$	$f(x^k) - f^* \sim \mathcal{O}\left(\dfrac{1}{k}\right)$	$\\|x^k - x^*\\|^2 \sim \mathcal{O}\left(\left(1 - \dfrac{\mu}{L}\right)^k\right)$
$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon^2}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\kappa \log \dfrac{1}{\varepsilon}\right)$

For smooth strongly convex we have:

$$f(x^k) - f^* \leq \left(1 - \frac{\mu}{L}\right)^k (f(x^0) - f^*).$$

Note also, that for any x

$$1 - x \leq e^{-x}$$

Finally we have

$$\begin{aligned}
\varepsilon = f(x^{k_\varepsilon}) - f^* &\leq \left(1 - \frac{\mu}{L}\right)^{k_\varepsilon} (f(x^0) - f^*) \\
&\leq \exp\left(-k_\varepsilon \frac{\mu}{L}\right) (f(x^0) - f^*) \\
k_\varepsilon \geq \kappa \log \frac{f(x^0) - f^*}{\varepsilon} &= \mathcal{O}\left(\kappa \log \frac{1}{\varepsilon}\right)
\end{aligned}$$

Question: Can we do faster, than this using the first-order information? **Yes, we can.**

Lower bounds

Lower bounds

convex (non-smooth)	smooth (non-convex)[1]	smooth & convex[2]	smooth & strongly convex (or PL)
$\mathcal{O}\left(\dfrac{1}{\sqrt{k}}\right)$	$\mathcal{O}\left(\dfrac{1}{k^2}\right)$	$\mathcal{O}\left(\dfrac{1}{k^2}\right)$	$\mathcal{O}\left(\left(1-\sqrt{\dfrac{\mu}{L}}\right)^k\right)$
$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\varepsilon^2}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\sqrt{\varepsilon}}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\dfrac{1}{\sqrt{\varepsilon}}\right)$	$k_\varepsilon \sim \mathcal{O}\left(\sqrt{\kappa}\log\dfrac{1}{\varepsilon}\right)$

[1]Carmon, Duchi, Hinder, Sidford, 2017
[2]Nemirovski, Yudin, 1979

How optimal is $\mathcal{O}\left(\frac{1}{k}\right)$?

- Is it somehow possible to understand, that the obtained convergence is the fastest possible with this class of problem and this class of algorithms?

How optimal is $\mathcal{O}\left(\frac{1}{k}\right)$?

* Is it somehow possible to understand, that the obtained convergence is the fastest possible with this class of problem and this class of algorithms?
* The iteration of gradient descent:

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) \\
&= x^{k-1} - \alpha^{k-1} \nabla f(x^{k-1}) - \alpha^k \nabla f(x^k) \\
&\;\;\vdots \\
&= x^0 - \sum_{i=0}^{k} \alpha^{k-i} \nabla f(x^{k-i})
\end{aligned}
$$

How optimal is $\mathcal{O}\left(\frac{1}{k}\right)$?

- Is it somehow possible to understand, that the obtained convergence is the fastest possible with this class of problem and this class of algorithms?
- The iteration of gradient descent:

$$
\begin{aligned}
x^{k+1} &= x^k - \alpha^k \nabla f(x^k) \\
&= x^{k-1} - \alpha^{k-1} \nabla f(x^{k-1}) - \alpha^k \nabla f(x^k) \\
&\;\;\vdots \\
&= x^0 - \sum_{i=0}^{k} \alpha^{k-i} \nabla f(x^{k-i})
\end{aligned}
$$

- Consider a family of first-order methods, where

$$
\begin{aligned}
x^{k+1} &\in x^0 + \text{span}\left\{\nabla f(x^0), \nabla f(x^1), \ldots, \nabla f(x^k)\right\} & f \text{ - smooth} \\
x^{k+1} &\in x^0 + \text{span}\left\{g_0, g_1, \ldots, g_k\right\}\text{, where } g_i \in \partial f(x^i) & f \text{ - non-smooth}
\end{aligned}
\tag{1}
$$

Non-smooth convex case

Theorem

There exists a function f that is G-Lipschitz and convex such that any method 1 satisfies

$$\min_{i\in[1,k]} f(x^i) - \min_{x\in\mathbb{B}(R)} f(x) \geq \frac{GR}{2(1+\sqrt{k})}$$

for $R > 0$ and $k \leq n$, where n is the dimension of the problem.

Non-smooth convex case

Theorem

There exists a function f that is G-Lipschitz and convex such that any method 1 satisfies

$$\min_{i \in [1,k]} f(x^i) - \min_{x \in \mathbb{B}(R)} f(x) \geq \frac{GR}{2(1+\sqrt{k})}$$

for $R > 0$ and $k \leq n$, where n is the dimension of the problem.

Proof idea: build such a function f that, for any method 1, we have

$$\text{span}\{g_0, g_1, \ldots, g_k\} \subset \text{span}\{e_1, e_2, \ldots, e_i\}$$

where e_i is the i-th standard basis vector. At iteration $k \leq n$, there are at least $n-k$ coordinate of x are 0. This helps us to derive a bound on the error.

Non-smooth case (proof)

Consider the function:

$$f(x) = \beta \max_{i \in [1,k]} x[i] + \frac{\alpha}{2} \|x\|_2^2,$$

where $\alpha, \beta \in \mathbb{R}$ are parameters, and $x[1:k]$ denotes the first k components of x.

Non-smooth case (proof)

Consider the function:

$$f(x) = \beta \max_{i \in [1,k]} x[i] + \frac{\alpha}{2}\|x\|_2^2,$$

where $\alpha, \beta \in \mathbb{R}$ are parameters, and $x[1:k]$ denotes the first k components of x.

Key Properties:

- The function $f(x)$ is α-strongly convex due to the quadratic term $\frac{\alpha}{2}\|x\|_2^2$.

Non-smooth case (proof)

Consider the function:

$$f(x) = \beta \max_{i \in [1,k]} x[i] + \frac{\alpha}{2}\|x\|_2^2,$$

where $\alpha, \beta \in \mathbb{R}$ are parameters, and $x[1:k]$ denotes the first k components of x.

Key Properties:

- The function $f(x)$ is α-strongly convex due to the quadratic term $\frac{\alpha}{2}\|x\|_2^2$.
- The function is non-smooth because the first term introduces a non-differentiable point at the maximum coordinate of x.

Non-smooth case (proof)

Consider the function:

$$f(x) = \beta \max_{i \in [1,k]} x[i] + \frac{\alpha}{2}\|x\|_2^2,$$

where $\alpha, \beta \in \mathbb{R}$ are parameters, and $x[1:k]$ denotes the first k components of x.

Key Properties:

- The function $f(x)$ is α-strongly convex due to the quadratic term $\frac{\alpha}{2}\|x\|_2^2$.
- The function is non-smooth because the first term introduces a non-differentiable point at the maximum coordinate of x.

Non-smooth case (proof)

Consider the function:

$$f(x) = \beta \max_{i \in [1,k]} x[i] + \frac{\alpha}{2}\|x\|_2^2,$$

where $\alpha, \beta \in \mathbb{R}$ are parameters, and $x[1:k]$ denotes the first k components of x.

Key Properties:

- The function $f(x)$ is α-strongly convex due to the quadratic term $\frac{\alpha}{2}\|x\|_2^2$.
- The function is non-smooth because the first term introduces a non-differentiable point at the maximum coordinate of x.

Consider the subdifferential of $f(x)$ at x:

$$\begin{aligned}
\partial f(x) &= \partial\left(\beta \max_{i \in [1,k]} x[i]\right) + \partial\left(\frac{\alpha}{2}\|x\|_2^2\right) \\
&= \beta \partial\left(\max_{i \in [1,k]} x[i]\right) + \alpha x. \\
&= \beta \text{conv}\left\{e_i \mid i : x[i] = \max_j x[j]\right\} + \alpha x.
\end{aligned}$$

Non-smooth case (proof)

Consider the function:

$$f(x) = \beta \max_{i \in [1,k]} x[i] + \frac{\alpha}{2}\|x\|_2^2,$$

where $\alpha, \beta \in \mathbb{R}$ are parameters, and $x[1:k]$ denotes the first k components of x.

Key Properties:

- The function $f(x)$ is α-strongly convex due to the quadratic term $\frac{\alpha}{2}\|x\|_2^2$.
- The function is non-smooth because the first term introduces a non-differentiable point at the maximum coordinate of x.

Consider the subdifferential of $f(x)$ at x:

$$\begin{aligned}
\partial f(x) &= \partial\left(\beta \max_{i \in [1,k]} x[i]\right) + \partial\left(\frac{\alpha}{2}\|x\|_2^2\right) \\
&= \beta \partial\left(\max_{i \in [1,k]} x[i]\right) + \alpha x. \\
&= \beta \text{conv}\left\{e_i \mid i : x[i] = \max_j x[j]\right\} + \alpha x.
\end{aligned}$$

It is easy to see, that if $g \in \partial f(x)$ and $\|x\| \leq R$, then

$$\|g\| \leq \alpha R + \beta$$

Thus, f is $\alpha R + \beta$-Lipschitz on $B(R)$.

Non-smooth case (proof)

Next, we describe the first-order oracle for this function. When queried for a subgradient at a point x, the oracle returns

$$\alpha x + \gamma e_i,$$

where i is the *first* coordinate for with $x[i] = \max_{1 \leq j \leq k} x[j]$.

- We ensure, that $\|x^0\| \leq R$ by starting from $x^0 = 0$.

Non-smooth case (proof)

Next, we describe the first-order oracle for this function. When queried for a subgradient at a point x, the oracle returns

$$\alpha x + \gamma e_i,$$

where i is the *first* coordinate for with $x[i] = \max_{1 \leq j \leq k} x[j]$.

- We ensure, that $\|x^0\| \leq R$ by starting from $x^0 = 0$.
- When the oracle is queried at $x^0 = 0$, it returns e_1. Consequently, x^1 must lie on the line generated by e_1.

Non-smooth case (proof)

Next, we describe the first-order oracle for this function. When queried for a subgradient at a point x, the oracle returns

$$\alpha x + \gamma e_i,$$

where i is the *first* coordinate for with $x[i] = \max_{1 \leq j \leq k} x[j]$.

- We ensure, that $\|x^0\| \leq R$ by starting from $x^0 = 0$.
- When the oracle is queried at $x^0 = 0$, it returns e_1. Consequently, x^1 must lie on the line generated by e_1.
- By an induction argument, one shows that for all i, the iterate x^i lies in the linear span of $\{e_1, \ldots, e_i\}$. In particular, for $i \leq k$, the $k+1$-th coordinate of x_i is zero and due to the structure of $f(x)$:

$$f(x^i) \geq 0.$$

Non-smooth case (proof)

- It remains to compute the minimal value of f. Define the point $y \in \mathbb{R}^n$ as

$$y[i] = -\frac{\beta}{\alpha k} \quad \text{for } 1 \leq i \leq k, \qquad y[i] = 0 \quad \text{for } k+1 \leq i \leq n.$$

Non-smooth case (proof)

- It remains to compute the minimal value of f. Define the point $y \in \mathbb{R}^n$ as

$$y[i] = -\frac{\beta}{\alpha k} \quad \text{for } 1 \leq i \leq k, \qquad y[i] = 0 \quad \text{for } k+1 \leq i \leq n.$$

- Note, that $0 \in \partial f(y)$:

$$\begin{aligned} \partial f(y) &= \alpha y + \beta \text{conv}\left\{ e_i \mid i : y[i] = \max_j y[j] \right\} \\ &= \alpha y + \beta \text{conv}\left\{ e_i \mid i : y[i] = 0 \right\} \\ 0 &\in \partial f(y). \end{aligned}$$

Non-smooth case (proof)

- It remains to compute the minimal value of f. Define the point $y \in \mathbb{R}^n$ as

$$y[i] = -\frac{\beta}{\alpha k} \quad \text{for } 1 \leq i \leq k, \qquad y[i] = 0 \quad \text{for } k+1 \leq i \leq n.$$

- Note, that $0 \in \partial f(y)$:

$$\begin{aligned} \partial f(y) &= \alpha y + \beta \text{conv}\left\{ e_i \mid i : y[i] = \max_j y[j] \right\} \\ &= \alpha y + \beta \text{conv}\left\{ e_i \mid i : y[i] = 0 \right\} \\ 0 &\in \partial f(y). \end{aligned}$$

- It follows that the minimum value of $f = f(y) = f(x^*)$ is

$$f(y) = -\frac{\beta^2}{\alpha k} + \frac{\alpha}{2} \cdot \frac{\beta^2}{\alpha^2 k} = -\frac{\beta^2}{2\alpha k}.$$

Non-smooth case (proof)

- It remains to compute the minimal value of f. Define the point $y \in \mathbb{R}^n$ as

$$y[i] = -\frac{\beta}{\alpha k} \quad \text{for } 1 \leq i \leq k, \qquad y[i] = 0 \quad \text{for } k+1 \leq i \leq n.$$

- Note, that $0 \in \partial f(y)$:

$$\begin{aligned}
\partial f(y) &= \alpha y + \beta \text{conv}\left\{ e_i \mid i : y[i] = \max_j y[j] \right\} \\
&= \alpha y + \beta \text{conv}\left\{ e_i \mid i : y[i] = 0 \right\} \\
0 &\in \partial f(y).
\end{aligned}$$

- It follows that the minimum value of $f = f(y) = f(x^*)$ is

$$f(y) = -\frac{\beta^2}{\alpha k} + \frac{\alpha}{2} \cdot \frac{\beta^2}{\alpha^2 k} = -\frac{\beta^2}{2\alpha k}.$$

- Now we have:

$$f(x^i) - f(x^*) \geq 0 - \left(-\frac{\beta^2}{2\alpha k} \right) \geq \frac{\beta^2}{2\alpha k}.$$

Non-smooth case (proof)

We have: $f(x^i) - f(x^*) \geq \frac{\beta^2}{2\alpha k}$, while we need to prove that $\min\limits_{i \in [1,k]} f(x^i) - f(x^*) \geq \frac{GR}{2(1+\sqrt{k})}$.

Non-smooth case (proof)

We have: $f(x^i) - f(x^*) \geq \frac{\beta^2}{2\alpha k}$, while we need to prove that $\min\limits_{i \in [1,k]} f(x^i) - f(x^*) \geq \frac{GR}{2(1+\sqrt{k})}$.

Convex case

$$
\alpha = \frac{G}{R}\frac{1}{1+\sqrt{k}} \qquad \beta = \frac{\sqrt{k}}{1+\sqrt{k}}
$$

$$
\frac{\beta^2}{2\alpha} = \frac{GRk}{2(1+\sqrt{k})}
$$

Note, in particular, that $\|y\|_2^2 = \frac{\beta^2}{\alpha^2 k} = R^2$ with these parameters

$$
\min_{i \in [1,k]} f(x^i) - f(x^*) \geq \frac{\beta^2}{2\alpha k} = \frac{GR}{2(1+\sqrt{k})}
$$

Non-smooth case (proof)

We have: $f(x^i) - f(x^*) \geq \frac{\beta^2}{2\alpha k}$, while we need to prove that $\min\limits_{i \in [1,k]} f(x^i) - f(x^*) \geq \frac{GR}{2(1+\sqrt{k})}$.

Convex case

$$\alpha = \frac{G}{R}\frac{1}{1+\sqrt{k}} \quad \beta = \frac{\sqrt{k}}{1+\sqrt{k}}$$

$$\frac{\beta^2}{2\alpha} = \frac{GRk}{2(1+\sqrt{k})}$$

Note, in particular, that $\|y\|_2^2 = \frac{\beta^2}{\alpha^2 k} = R^2$ with these parameters

$$\min_{i \in [1,k]} f(x^i) - f(x^*) \geq \frac{\beta^2}{2\alpha k} = \frac{GR}{2(1+\sqrt{k})}$$

Strongly convex case

$$\alpha = \frac{G}{2R} \quad \beta = \frac{G}{2}$$

Note, in particular, that $\|y\|_2^2 = \frac{\beta^2}{\alpha^2 k} = \frac{G^2}{4\alpha^2 k} = R^2$ with these parameters

$$\min_{i \in [1,k]} f(x^i) - f(x^*) \geq \frac{G^2}{8\alpha k}$$

Smooth case

Theorem

There exists a function f that is L-smooth and convex such that any method 1 satisfies

$$\min_{i \in [1,k]} f(x^i) - f^* \geq \frac{3L\|x^0 - x^*\|_2^2}{32(1+k)^2}$$

Smooth case

Theorem

There exists a function f that is L-smooth and convex such that any method 1 satisfies

$$\min_{i \in [1,k]} f(x^i) - f^* \geq \frac{3L\|x^0 - x^*\|_2^2}{32(1+k)^2}$$

- No matter what gradient method you provide, there is always a function f that, when you apply your gradient method on minimizing such f, the convergence rate is lower bounded as $\mathcal{O}\left(\frac{1}{k^2}\right)$.

Smooth case

Theorem

There exists a function f that is L-smooth and convex such that any method 1 satisfies

$$\min_{i \in [1,k]} f(x^i) - f^* \geq \frac{3L\|x^0 - x^*\|_2^2}{32(1+k)^2}$$

- No matter what gradient method you provide, there is always a function f that, when you apply your gradient method on minimizing such f, the convergence rate is lower bounded as $\mathcal{O}\left(\frac{1}{k^2}\right)$.
- The key to the proof is to explicitly build a special function f.

Nesterov's worst function

- Let $n = 2k + 1$ and $A \in \mathbb{R}^{n \times n}$.

$$
\begin{bmatrix}
2 & -1 & 0 & 0 & \cdots & 0 \\
-1 & 2 & -1 & 0 & \cdots & 0 \\
0 & -1 & 2 & -1 & \cdots & 0 \\
0 & 0 & -1 & 2 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & \cdots & 2
\end{bmatrix}
$$

Nesterov's worst function

- Let $n = 2k + 1$ and $A \in \mathbb{R}^{n \times n}$.

$$
\begin{bmatrix}
2 & -1 & 0 & 0 & \cdots & 0 \\
-1 & 2 & -1 & 0 & \cdots & 0 \\
0 & -1 & 2 & -1 & \cdots & 0 \\
0 & 0 & -1 & 2 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & \cdots & 2
\end{bmatrix}
$$

- Notice, that

$$
x^T A x = x[1]^2 + x[n]^2 + \sum_{i=1}^{n-1} (x[i] - x[i+1])^2,
$$

and, from this expression, it's simple to check $0 \preceq A \preceq 4I$.

Nesterov's worst function

- Let $n = 2k + 1$ and $A \in \mathbb{R}^{n \times n}$.

$$
\begin{bmatrix}
2 & -1 & 0 & 0 & \cdots & 0 \\
-1 & 2 & -1 & 0 & \cdots & 0 \\
0 & -1 & 2 & -1 & \cdots & 0 \\
0 & 0 & -1 & 2 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & \cdots & 2
\end{bmatrix}
$$

- Notice, that

$$
x^T A x = x[1]^2 + x[n]^2 + \sum_{i=1}^{n-1} (x[i] - x[i+1])^2,
$$

and, from this expression, it's simple to check $0 \preceq A \preceq 4I$.

- Define the following L-smooth convex function

$$
f(x) = \frac{L}{8} x^T A x - \frac{L}{4} \langle x, e_1 \rangle.
$$

Nesterov's worst function

- Let $n = 2k+1$ and $A \in \mathbb{R}^{n \times n}$.

$$
\begin{bmatrix}
2 & -1 & 0 & 0 & \cdots & 0 \\
-1 & 2 & -1 & 0 & \cdots & 0 \\
0 & -1 & 2 & -1 & \cdots & 0 \\
0 & 0 & -1 & 2 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & \cdots & 2
\end{bmatrix}
$$

- Notice, that

$$
x^T A x = x[1]^2 + x[n]^2 + \sum_{i=1}^{n-1} (x[i] - x[i+1])^2,
$$

and, from this expression, it's simple to check $0 \preceq A \preceq 4I$.

- Define the following L-smooth convex function

$$
f(x) = \frac{L}{8} x^T A x - \frac{L}{4} \langle x, e_1 \rangle.
$$

- The optimal solution x^* satisfies $Ax^* = e_1$, and solving this system of equations gives

$$
x^*[i] = 1 - \frac{i}{n+1},
$$

Nesterov's worst function

- Let $n = 2k+1$ and $A \in \mathbb{R}^{n \times n}$.

$$\begin{bmatrix} 2 & -1 & 0 & 0 & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & -1 & \cdots & 0 \\ 0 & 0 & -1 & 2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 2 \end{bmatrix}$$

- Notice, that

$$x^T A x = x[1]^2 + x[n]^2 + \sum_{i=1}^{n-1} (x[i] - x[i+1])^2,$$

  and, from this expression, it's simple to check $0 \preceq A \preceq 4I$.
- Define the following L-smooth convex function

$$f(x) = \frac{L}{8} x^T A x - \frac{L}{4} \langle x, e_1 \rangle.$$

- The optimal solution x^* satisfies $Ax^* = e_1$, and solving this system of equations gives

$$x^*[i] = 1 - \frac{i}{n+1},$$

- And the objective value is

$$\begin{aligned} f(x^*) &= \frac{L}{8} x^{*T} A x^* - \frac{L}{4} \langle x^*, e_1 \rangle \\ &= -\frac{L}{8} \langle x^*, e_1 \rangle = -\frac{L}{8} \left(1 - \frac{1}{n+1}\right). \end{aligned}$$

Smooth case (proof)

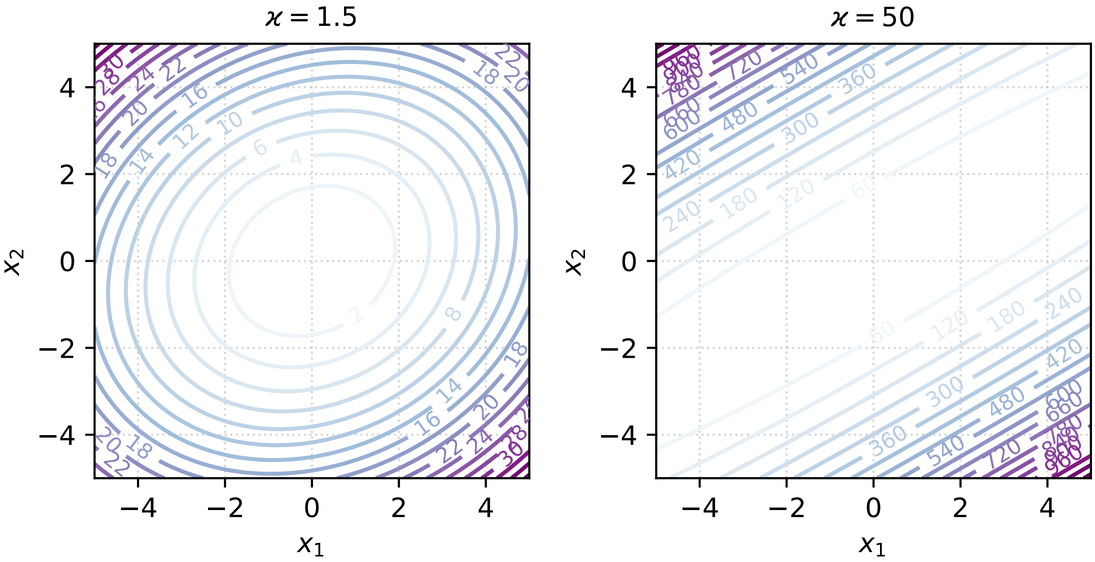
TBD

Smooth case (proof)

TBD

Acceleration for quadratics

Condition number



Condition number and convergence speed

Even with the optimal parameter choice, the error at the next step satisfies

$$\|x_{k+1} - x^*\|_2 \leq q\|x_k - x^*\|_2, \quad \rightarrow \quad \|x_k - x^*\|_2 \leq q^k\|x_0 - x^*\|_2,$$

where

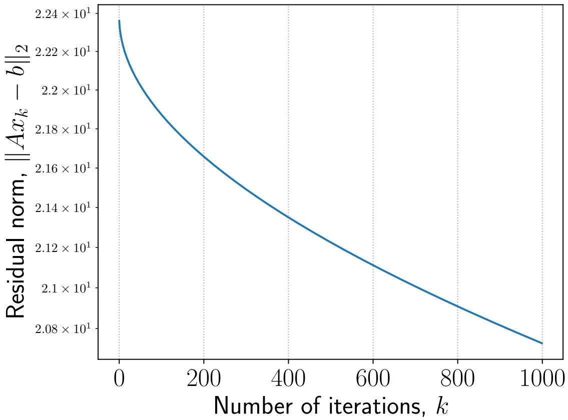
$$q = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} = \frac{\kappa - 1}{\kappa + 1},$$

$$\kappa = \frac{\lambda_{\max}}{\lambda_{\min}} \quad \text{for} \quad A \in \mathbb{S}^n_{++}$$

is the condition number of A.

Let us do some demo...

Demo

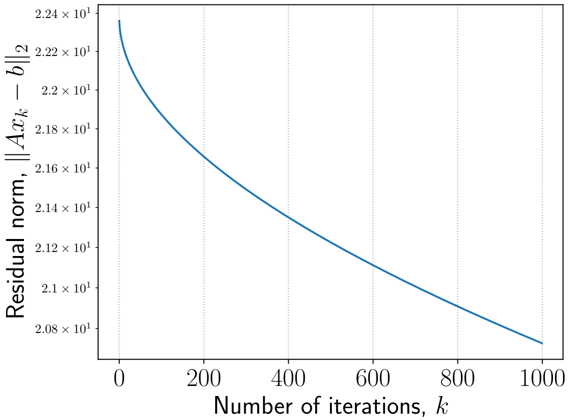


- Thus, for **ill-conditioned** matrices the error of the gradient descent method decays very slowly

Consider non-hermitian matrix A

Possible cases of gradient descent behaviour:

Demo

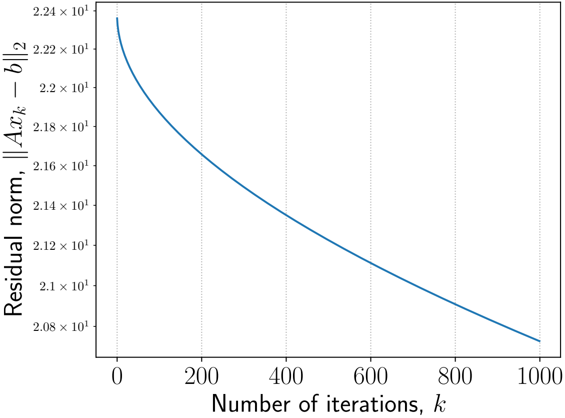


- Thus, for **ill-conditioned** matrices the error of the gradient descent method decays very slowly
- This is another reason why **condition number** is so important:

Consider non-hermitian matrix A

Possible cases of gradient descent behaviour:

Demo

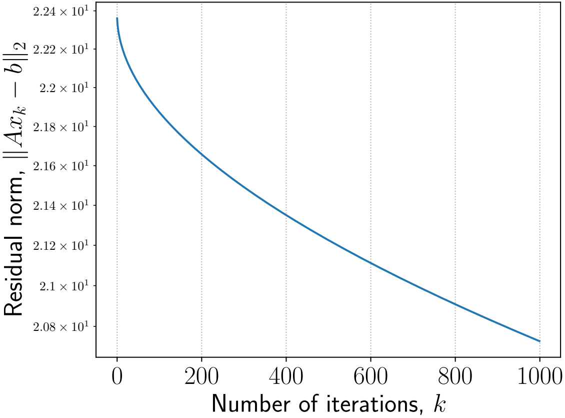


- Thus, for **ill-conditioned** matrices the error of the gradient descent method decays very slowly
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.

Consider non-hermitian matrix A

Possible cases of gradient descent behaviour:

Demo



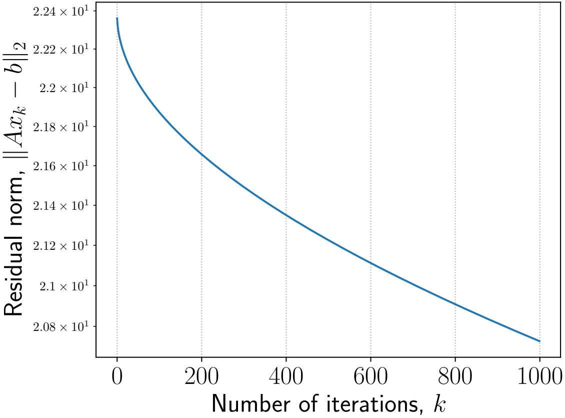
- Thus, for **ill-conditioned** matrices the error of the gradient descent method decays very slowly
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.

Consider non-hermitian matrix A

Possible cases of gradient descent behaviour:

- convergence

Demo



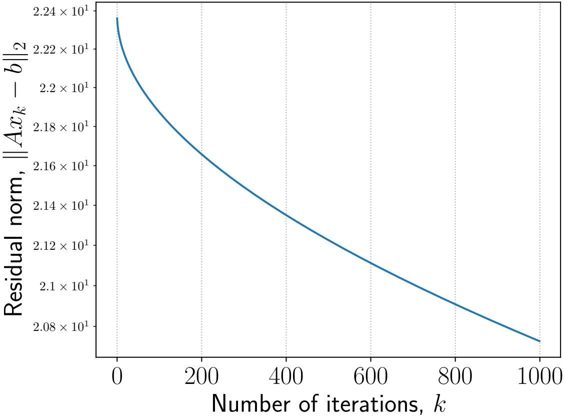
- Thus, for **ill-conditioned** matrices the error of the gradient descent method decays very slowly
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.

Consider non-hermitian matrix A

Possible cases of gradient descent behaviour:

- convergence
- divergence

Demo



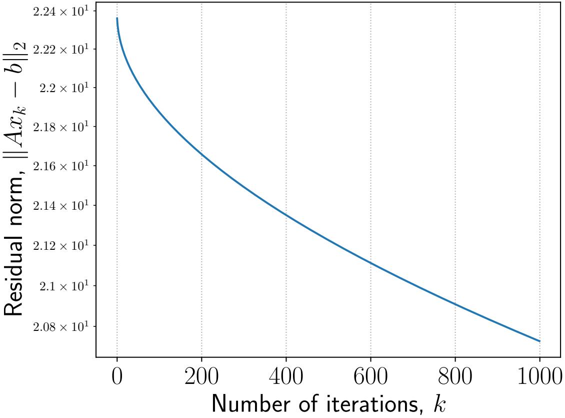
- Thus, for **ill-conditioned** matrices the error of the gradient descent method decays very slowly
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.

Consider non-hermitian matrix A

Possible cases of gradient descent behaviour:

- convergence
- divergence
- almost stable trajectory

Demo



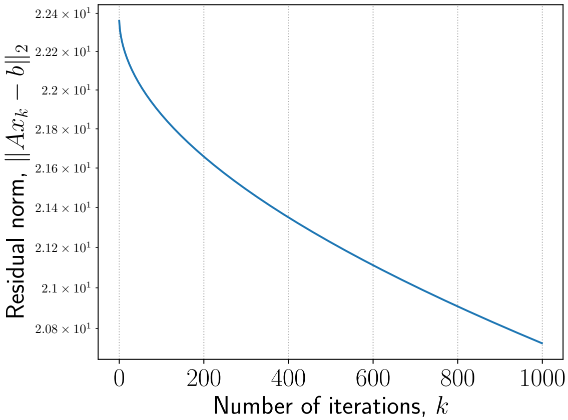
- Thus, for **ill-conditioned** matrices the error of the gradient descent method decays very slowly
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.

Consider non-hermitian matrix A

Possible cases of gradient descent behaviour:

- convergence
- divergence
- almost stable trajectory

Demo



- Thus, for **ill-conditioned** matrices the error of the gradient descent method decays very slowly
- This is another reason why **condition number** is so important:
- Besides the bound on the error in the solution, it also gives an estimate of the number of iterations for the iterative methods.

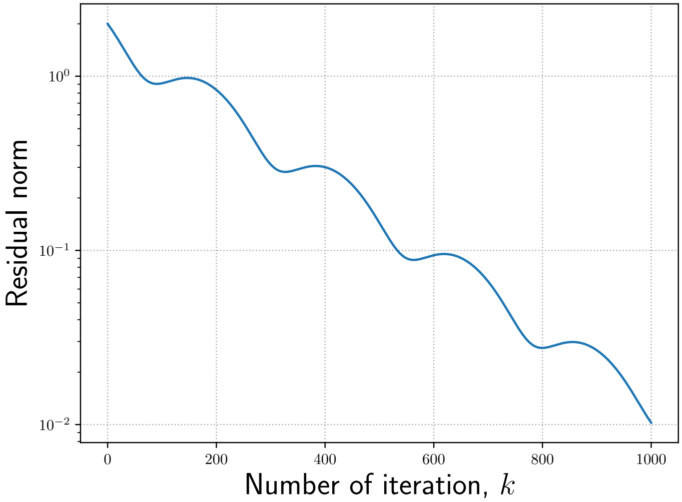
Consider non-hermitian matrix A

Possible cases of gradient descent behaviour:

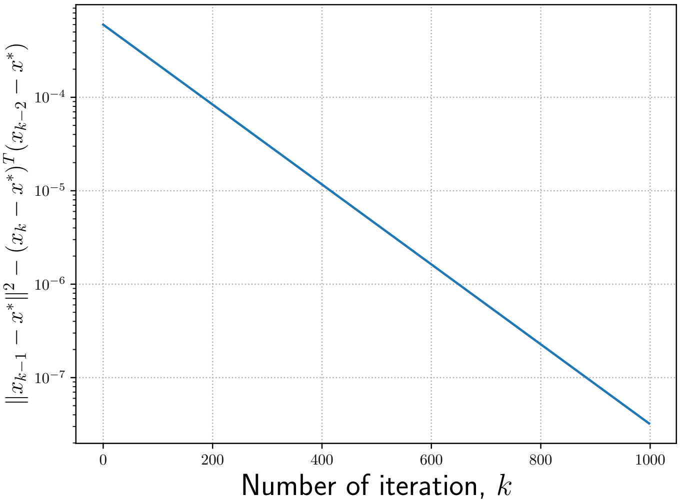
- convergence
- divergence
- almost stable trajectory

Q: how can we identify our case **before** running iterative method?

Spectrum directly affects the convergence

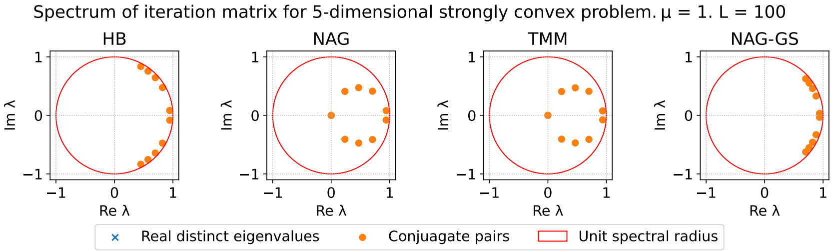


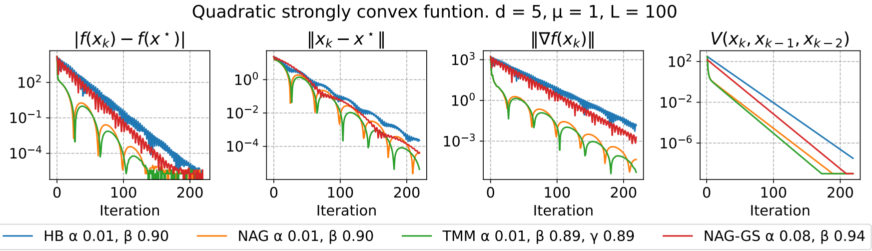
One can still formulate a Lyapunov function [3]



[3] Another approach to build Lyapunov functions for the first order methods in the quadratic case. D. M. Merkulov, I. V. Oseledets

Relation of the method matrix spectrum for the quadratic problem and convergence of methods[4]





[4] Another Approach to Build Lyapunov Functions for the First Order Methods in the Quadratic Case

Attempt 1: Exact line search aka steepest descent

$$
\alpha_k = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))
$$

More theoretical than practical approach. It also allows you to analyze the convergence, but often exact line search can be difficult if the function calculation takes too long or costs a lot. An interesting theoretical property of this method is that each following iteration is orthogonal to the previous one:

$$
\alpha_k = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))
$$

Attempt 1: Exact line search aka steepest descent

$$
\alpha_k = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))
$$

More theoretical than practical approach. It also allows you to analyze the convergence, but often exact line search can be difficult if the function calculation takes too long or costs a lot. An interesting theoretical property of this method is that each following iteration is orthogonal to the previous one:

$$
\alpha_k = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))
$$

Optimality conditions:

Attempt 1: Exact line search aka steepest descent

$$\alpha_k = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_{k+1}) = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))$$

More theoretical than practical approach. It also allows you to analyze the convergence, but often exact line search can be difficult if the function calculation takes too long or costs a lot. An interesting theoretical property of this method is that each following iteration is orthogonal to the previous one:

$$\alpha_k = \text{arg}\min_{\alpha \in \mathbb{R}^+} f(x_k - \alpha \nabla f(x_k))$$

Optimality conditions:

$$\nabla f(x_{k+1})^\top \nabla f(x_k) = 0$$

The convergence rate is the same as for the gradient descent!

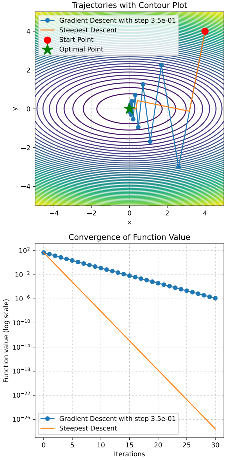


Figure 1: Steepest Descent

Open In Colab ♣

Attempt 2: Chebyshev acceleration

Another way to find α_k is to consider

$$\|x_{k+1} - x^*\| = (I - \alpha_k A)\|x_k - x^*\| = (I - \alpha_k A)(I - \alpha_{k-1} A)\|x_{k-1} - x^*\| = \ldots = p(A)\|x_0 - x^*\|,$$

where $p(A)$ is a **matrix polynomial** (simplest matrix function)

$$p(A) = (I - \alpha_k A) \ldots (I - \alpha_0 A),$$

and $p(0) = I$.

Optimal choice of time steps

The error is written as

$$e_{k+1} = p(A)e_0,$$

and hence

$$\|e_{k+1}\| \leq \|p(A)\|\|e_0\|,$$

where $p(0) = 1$ and $p(A)$ is a **matrix polynomial**.

To get better **error reduction**, we need to minimize

$$\|p(A)\|$$

over all possible polynomials $p(x)$ of degree $k+1$ such that $p(0) = 1$. We will use $\|\cdot\|_2$.

Polynomials least deviating from zeros

Important special case: $A = A^* > 0$.

Then, $A = U\Lambda U^*$,

and

$$\|p(A)\|_2 = \|Up(\Lambda)U^*\|_2 = \|p(\Lambda)\|_2 = \max_i |p(\lambda_i)| \overset{!}{\leq} \max_{\lambda_{\min} \leq \lambda \leq \lambda_{\max}} |p(\lambda)|.$$

The latter inequality is the only approximation. Here we make a **crucial assumption** that we do not want to benefit from the distribution of the spectrum between $\lambda_{\min}$ and $\lambda_{\max}$.

Thus, we need to find a polynomial $p(\lambda)$ such that $p(0) = 1$, and which has the least possible deviation from 0 on $[\lambda_{\min}, \lambda_{\max}]$.

Polynomials least deviating from zeros (2)

We can do the affine transformation of the interval $[\lambda_{\min}, \lambda_{\max}]$ to the interval $[-1, 1]$:

$$\xi = \frac{\lambda_{\max} + \lambda_{\min} - (\lambda_{\min} - \lambda_{\max})x}{2}, \quad x \in [-1, 1].$$

The problem is then reduced to the problem of finding the **polynomial least deviating from zero** on an interval $[-1, 1]$.

Exact solution: Chebyshev polynomials

The exact solution to this problem is given by the famous **Chebyshev polynomials** of the form

$$T_n(x) = \cos(n \arccos x)$$

What do you need to know about Chebyshev polynomials

1. This is a polynomial!

What do you need to know about Chebyshev polynomials

1. This is a polynomial!
2. We can express T_n from T_{n-1} and T_{n-2}:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x$$

What do you need to know about Chebyshev polynomials

1. This is a polynomial!
2. We can express T_n from T_{n-1} and T_{n-2}:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x$$

3. $|T_n(x)| \leq 1$ on $x \in [-1, 1]$.

What do you need to know about Chebyshev polynomials

1. This is a polynomial!
2. We can express T_n from T_{n-1} and T_{n-2}:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x$$

3. $|T_n(x)| \leq 1$ on $x \in [-1, 1]$.
4. It has $(n+1)$ **alternation points**, where the maximal absolute value is achieved (this is the sufficient and necessary condition for the **optimality**) (Chebyshev alternance theorem, no proof here).

What do you need to know about Chebyshev polynomials

1. This is a polynomial!
2. We can express T_n from T_{n-1} and T_{n-2}:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x$$

3. $|T_n(x)| \leq 1$ on $x \in [-1, 1]$.
4. It has $(n+1)$ **alternation points**, where the maximal absolute value is achieved (this is the sufficient and necessary condition for the **optimality**) (Chebyshev alternance theorem, no proof here).
5. The **roots** are just

$$n \arccos x_k = \frac{\pi}{2} + \pi k, \quad \rightarrow \quad x_k = \cos \frac{\pi(2k+1)}{2n}, \; k = 0, \ldots, n-1$$

We can plot them...

Convergence of the Chebyshev-accelerated gradient descent

Note that $p(x) = (1 - \tau_n x) \dots (1 - \tau_0 x)$, hence roots of $p(x)$ are $1/\tau_i$ and that we additionally need to map back from $[-1, 1]$ to $[\lambda_{\min}, \lambda_{\max}]$. This results into

$$\tau_i = \frac{2}{\lambda_{\max} + \lambda_{\min} - (\lambda_{\max} - \lambda_{\min}) x_i}, \quad x_i = \cos \frac{\pi(2i + 1)}{2n} \quad i = 0, \dots, n - 1$$

The convergence (we only give the result without the proof) is now given by

$$e_{k+1} \leq C q^k e_0, \quad q = \frac{\sqrt{\text{cond}(A)} - 1}{\sqrt{\text{cond}(A)} + 1},$$

which is better than in the gradient descent.

Convergence of the Chebyshev-accelerated gradient descent

Note that $p(x) = (1 - \tau_n x) \dots (1 - \tau_0 x)$, hence roots of $p(x)$ are $1/\tau_i$ and that we additionally need to map back from $[-1, 1]$ to $[\lambda_{\min}, \lambda_{\max}]$. This results into

$$\tau_i = \frac{2}{\lambda_{\max} + \lambda_{\min} - (\lambda_{\max} - \lambda_{\min})x_i}, \quad x_i = \cos \frac{\pi(2i+1)}{2n} \quad i = 0, \dots, n-1$$

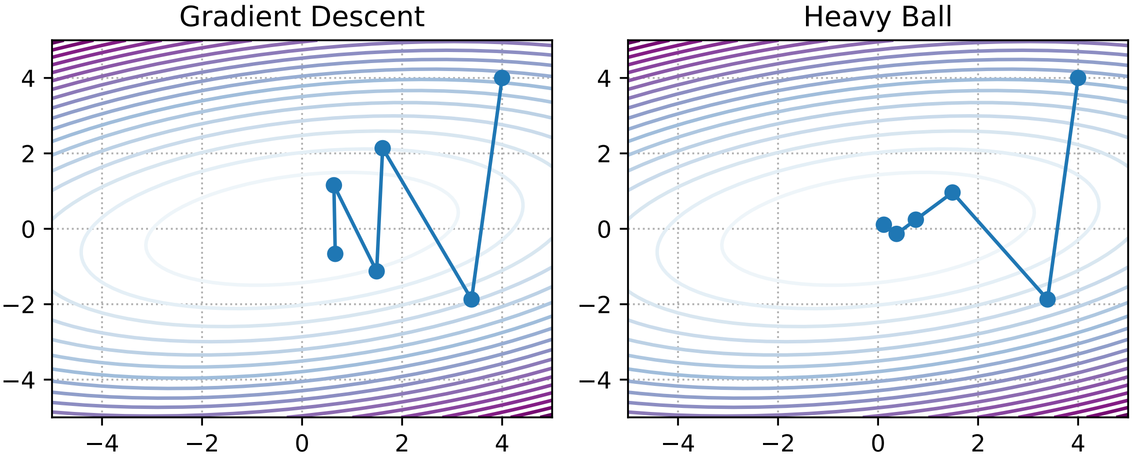
The convergence (we only give the result without the proof) is now given by

$$e_{k+1} \leq C q^k e_0, \quad q = \frac{\sqrt{\mathrm{cond}(A)} - 1}{\sqrt{\mathrm{cond}(A)} + 1},$$

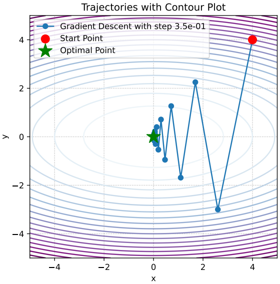
which is better than in the gradient descent.

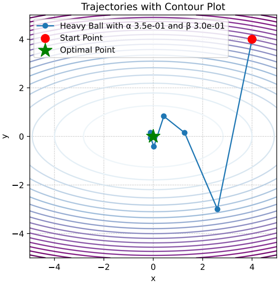
Heavy ball

Oscillations and acceleration



Polyak Heavy ball method

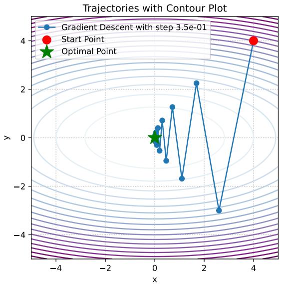


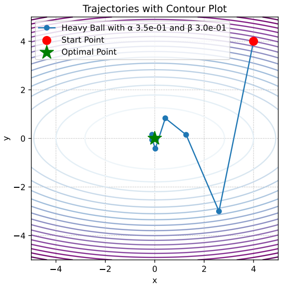


Let's introduce the idea of momentum, proposed by Polyak in 1964. Recall that the momentum update is

$$
x^{k+1} = x^k - \alpha \nabla f(x^k) + \beta (x^k - x^{k-1}).
$$

Polyak Heavy ball method





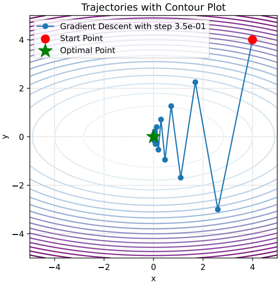
Let's introduce the idea of momentum, proposed by Polyak in 1964. Recall that the momentum update is

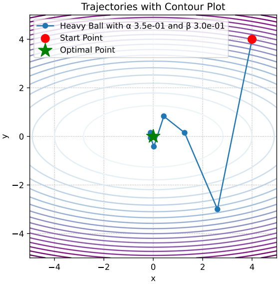
$$
x^{k+1} = x^k - \alpha \nabla f(x^k) + \beta (x^k - x^{k-1}).
$$

Which is in our (quadratics) case is

$$
\hat{x}_{k+1} = \hat{x}_k - \alpha \Lambda \hat{x}_k + \beta (\hat{x}_k - \hat{x}_{k-1}) = (I - \alpha \Lambda + \beta I) \hat{x}_k - \beta \hat{x}_{k-1}
$$

Polyak Heavy ball method





Let's introduce the idea of momentum, proposed by Polyak in 1964. Recall that the momentum update is

$$x^{k+1} = x^k - \alpha \nabla f(x^k) + \beta (x^k - x^{k-1}).$$

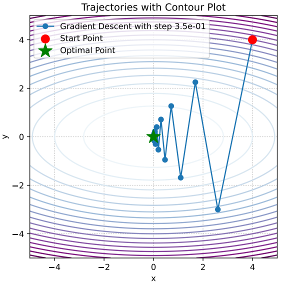
Which is in our (quadratics) case is

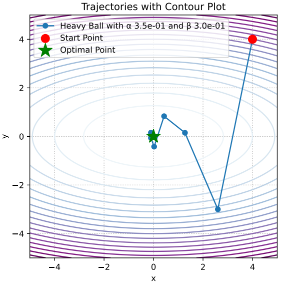
$$\hat{x}_{k+1} = \hat{x}_k - \alpha \Lambda \hat{x}_k + \beta (\hat{x}_k - \hat{x}_{k-1}) = (I - \alpha \Lambda + \beta I)\hat{x}_k - \beta \hat{x}_{k-1}$$

This can be rewritten as follows

$$\begin{aligned} \hat{x}_{k+1} &= (I - \alpha \Lambda + \beta I)\hat{x}_k - \beta \hat{x}_{k-1}, \\ \hat{x}_k &= \hat{x}_k. \end{aligned}$$

Polyak Heavy ball method





Let's introduce the idea of momentum, proposed by Polyak in 1964. Recall that the momentum update is

$$
x^{k+1} = x^k - \alpha \nabla f(x^k) + \beta (x^k - x^{k-1}).
$$

Which is in our (quadratics) case is

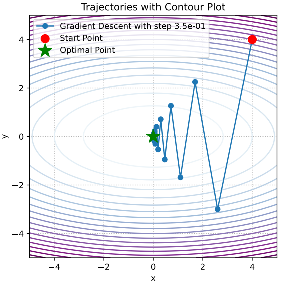
$$
\hat{x}_{k+1} = \hat{x}_k - \alpha \Lambda \hat{x}_k + \beta (\hat{x}_k - \hat{x}_{k-1}) = (I - \alpha \Lambda + \beta I)\hat{x}_k - \beta \hat{x}_{k-1}
$$

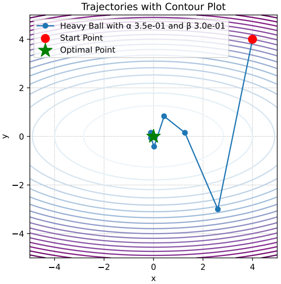
This can be rewritten as follows

$$
\begin{aligned}
\hat{x}_{k+1} &= (I - \alpha \Lambda + \beta I)\hat{x}_k - \beta \hat{x}_{k-1}, \\
\hat{x}_k &= \hat{x}_k.
\end{aligned}
$$

Let's use the following notation $\hat{z}_k = \begin{bmatrix} \hat{x}_{k+1} \\ \hat{x}_k \end{bmatrix}$. Therefore $\hat{z}_{k+1} = M \hat{z}_k$, where the iteration matrix M is:

Polyak Heavy ball method





Let's introduce the idea of momentum, proposed by Polyak in 1964. Recall that the momentum update is

$$x^{k+1} = x^k - \alpha \nabla f(x^k) + \beta (x^k - x^{k-1}).$$

Which is in our (quadratics) case is

$$\hat{x}_{k+1} = \hat{x}_k - \alpha \Lambda \hat{x}_k + \beta(\hat{x}_k - \hat{x}_{k-1}) = (I - \alpha\Lambda + \beta I)\hat{x}_k - \beta \hat{x}_{k-1}$$

This can be rewritten as follows

$$\begin{aligned} \hat{x}_{k+1} &= (I - \alpha\Lambda + \beta I)\hat{x}_k - \beta \hat{x}_{k-1}, \\ \hat{x}_k &= \hat{x}_k. \end{aligned}$$

Let's use the following notation $\hat{z}_k = \begin{bmatrix} \hat{x}_{k+1} \\ \hat{x}_k \end{bmatrix}$. Therefore $\hat{z}_{k+1} = M \hat{z}_k$, where the iteration matrix M is:

$$M = \begin{bmatrix} I - \alpha\Lambda + \beta I & -\beta I \\ I & 0_d \end{bmatrix}.$$

Reduction to a scalar case

Note, that M is $2d \times 2d$ matrix with 4 block-diagonal matrices of size $d \times d$ inside. It means, that we can rearrange the order of coordinates to make M block-diagonal in the following form. Note that in the equation below, the matrix M denotes the same as in the notation above, except for the described permutation of rows and columns. We use this slight abuse of notation for the sake of clarity.

Reduction to a scalar case

Note, that M is $2d \times 2d$ matrix with 4 block-diagonal matrices of size $d \times d$ inside. It means, that we can rearrange the order of coordinates to make M block-diagonal in the following form. Note that in the equation below, the matrix M denotes the same as in the notation above, except for the described permutation of rows and columns. We use this slight abuse of notation for the sake of clarity.

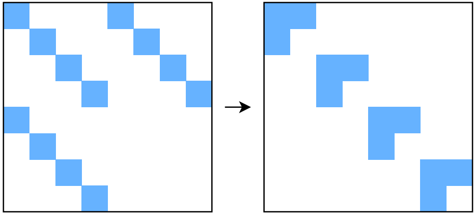


Figure 2: Illustration of matrix M rearrangement

$$\begin{bmatrix} \hat{x}_k^{(1)} \\ \vdots \\ \hat{x}_k^{(d)} \\ \hat{x}_{k-1}^{(1)} \\ \vdots \\ \hat{x}_{k-1}^{(d)} \end{bmatrix} \to \begin{bmatrix} \hat{x}_k^{(1)} \\ \hat{x}_{k-1}^{(1)} \\ \vdots \\ \hat{x}_k^{(d)} \\ \hat{x}_{k-1}^{(d)} \end{bmatrix} \quad M = \begin{bmatrix} M_1 & & & \\ & M_2 & & \\ & & \ldots & \\ & & & M_d \end{bmatrix}$$

where $\hat{x}_k^{(i)}$ is i-th coordinate of vector $\hat{x}_k \in \mathbb{R}^d$ and M_i stands for 2×2 matrix. This rearrangement allows us to study the dynamics of the method independently for each dimension. One may observe, that the asymptotic convergence rate of the $2d$-dimensional vector sequence of $\hat{z}_k$ is defined by the worst convergence rate among its block of coordinates. Thus, it is enough to study the optimization in a one-dimensional case.

Reduction to a scalar case

For i-th coordinate with λ_i as an i-th eigenvalue of matrix W we have:

$$M_i = \begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix}.$$

Reduction to a scalar case

For i-th coordinate with λ_i as an i-th eigenvalue of matrix W we have:

$$M_i = \begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix}.$$

The method will be convergent if $\rho(M) < 1$, and the optimal parameters can be computed by optimizing the spectral radius

$$\alpha^*, \beta^* = \arg\min_{\alpha,\beta} \max_i \rho(M_i) \quad \alpha^* = \frac{4}{(\sqrt{L} + \sqrt{\mu})^2}; \quad \beta^* = \left(\frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}\right)^2.$$

Reduction to a scalar case

For i-th coordinate with λ_i as an i-th eigenvalue of matrix W we have:

$$M_i = \begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix}.$$

The method will be convergent if $\rho(M) < 1$, and the optimal parameters can be computed by optimizing the spectral radius

$$\alpha^*, \beta^* = \arg\min_{\alpha,\beta} \max_i \rho(M_i) \quad \alpha^* = \frac{4}{(\sqrt{L} + \sqrt{\mu})^2}; \quad \beta^* = \left(\frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}\right)^2.$$

It can be shown, that for such parameters the matrix M has complex eigenvalues, which forms a conjugate pair, so the distance to the optimum (in this case, $\|z_k\|$), generally, will not go to zero monotonically.

Heavy ball quadratic convergence

We can explicitly calculate the eigenvalues of M_i:

$$\lambda^M_1, \lambda^M_2 = \lambda \left(\begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix} \right) = \frac{1 + \beta - \alpha\lambda_i \pm \sqrt{(1 + \beta - \alpha\lambda_i)^2 - 4\beta}}{2}.$$

Heavy ball quadratic convergence

We can explicitly calculate the eigenvalues of M_i:

$$\lambda^M_1, \lambda^M_2 = \lambda \left(\begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix} \right) = \frac{1 + \beta - \alpha\lambda_i \pm \sqrt{(1 + \beta - \alpha\lambda_i)^2 - 4\beta}}{2}.$$

When α and β are optimal (α^*, β^*), the eigenvalues are complex-conjugated pair $(1 + \beta - \alpha\lambda_i)^2 - 4\beta \leq 0$, i.e. $\beta \geq (1 - \sqrt{\alpha\lambda_i})^2$.

Heavy ball quadratic convergence

We can explicitly calculate the eigenvalues of M_i:

$$\lambda^M_1, \lambda^M_2 = \lambda \left(\begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix} \right) = \frac{1 + \beta - \alpha\lambda_i \pm \sqrt{(1 + \beta - \alpha\lambda_i)^2 - 4\beta}}{2}.$$

When α and β are optimal (α^*, β^*), the eigenvalues are complex-conjugated pair $(1 + \beta - \alpha\lambda_i)^2 - 4\beta \leq 0$, i.e. $\beta \geq (1 - \sqrt{\alpha\lambda_i})^2$.

$$\text{Re}(\lambda^M_1) = \frac{L + \mu - 2\lambda_i}{(\sqrt{L} + \sqrt{\mu})^2}; \quad \text{Im}(\lambda^M_1) = \frac{\pm 2\sqrt{(L - \lambda_i)(\lambda_i - \mu)}}{(\sqrt{L} + \sqrt{\mu})^2}; \quad |\lambda^M_1| = \frac{L - \mu}{(\sqrt{L} + \sqrt{\mu})^2}.$$

Heavy ball quadratic convergence

We can explicitly calculate the eigenvalues of M_i:

$$\lambda^M_1, \lambda^M_2 = \lambda \left(\begin{bmatrix} 1 - \alpha\lambda_i + \beta & -\beta \\ 1 & 0 \end{bmatrix} \right) = \frac{1 + \beta - \alpha\lambda_i \pm \sqrt{(1 + \beta - \alpha\lambda_i)^2 - 4\beta}}{2}.$$

When α and β are optimal (α^*, β^*), the eigenvalues are complex-conjugated pair $(1 + \beta - \alpha\lambda_i)^2 - 4\beta \leq 0$, i.e. $\beta \geq (1 - \sqrt{\alpha\lambda_i})^2$.

$$\text{Re}(\lambda^M_1) = \frac{L + \mu - 2\lambda_i}{(\sqrt{L} + \sqrt{\mu})^2}; \quad \text{Im}(\lambda^M_1) = \frac{\pm 2\sqrt{(L - \lambda_i)(\lambda_i - \mu)}}{(\sqrt{L} + \sqrt{\mu})^2}; \quad |\lambda^M_1| = \frac{L - \mu}{(\sqrt{L} + \sqrt{\mu})^2}.$$

And the convergence rate does not depend on the stepsize and equals to $\sqrt{\beta^*}$.

Heavy Ball quadratics convergence

Theorem

Assume that f is quadratic μ-strongly convex L-smooth quadratics, then Heavy Ball method with parameters

$$\alpha = \frac{4}{(\sqrt{L} + \sqrt{\mu})^2}, \beta = \frac{\sqrt{L} - \sqrt{\mu}}{\sqrt{L} + \sqrt{\mu}}$$

converges linearly:

$$\|x_k - x^*\|_2 \leq \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right) \|x_0 - x^*\|$$

Heavy Ball Global Convergence [5]

Theorem

Assume that f is smooth and convex and that

$$\beta \in [0,1), \quad \alpha \in \left(0, \dfrac{2(1-\beta)}{L}\right).$$

Then, the sequence $\{x_k\}$ generated by Heavy-ball iteration satisfies

$$f(\overline{x}_T) - f^\star \leq \begin{cases} \frac{\|x_0 - x^\star\|^2}{2(T+1)}\left(\frac{L\beta}{1-\beta} + \frac{1-\beta}{\alpha}\right), & \text{if } \alpha \in \left(0, \dfrac{1-\beta}{L}\right], \\ \frac{\|x_0 - x^\star\|^2}{2(T+1)(2(1-\beta)-\alpha L)}\left(L\beta + \frac{(1-\beta)^2}{\alpha}\right), & \text{if } \alpha \in \left[\dfrac{1-\beta}{L}, \dfrac{2(1-\beta)}{L}\right), \end{cases}$$

where $\overline{x}_T$ is the Cesaro average of the iterates, i.e.,

$$\overline{x}_T = \frac{1}{T+1}\sum_{k=0}^{T} x_k.$$

[5] Global convergence of the Heavy-ball method for convex optimization, Euhanna Ghadimi et.al.

Heavy Ball Global Convergence [6]

Theorem

Assume that f is smooth and strongly convex and that

$$\alpha \in (0, \frac{2}{L}), \quad 0 \leq \beta < \frac{1}{2}\left(\frac{\mu\alpha}{2} + \sqrt{\frac{\mu^2\alpha^2}{4} + 4(1 - \frac{\alpha L}{2})}\right).$$

where $\alpha_0 \in (0, 1/L]$. Then, the sequence $\{x_k\}$ generated by Heavy-ball iteration converges linearly to a unique optimizer $x^\star$. In particular,

$$f(x_k) - f^\star \leq q^k(f(x_0) - f^\star),$$

where $q \in [0, 1)$.

[6] Global convergence of the Heavy-ball method for convex optimization, Euhanna Ghadimi et.al.

Heavy ball method summary

- Ensures accelerated convergence for strongly convex quadratic problems

Heavy ball method summary

* Ensures accelerated convergence for strongly convex quadratic problems
* Local accelerated convergence was proved in the original paper.

Heavy ball method summary

* Ensures accelerated convergence for strongly convex quadratic problems
* Local accelerated convergence was proved in the original paper.
* Recently was proved, that there is no global accelerated convergence for the method.

Heavy ball method summary

- Ensures accelerated convergence for strongly convex quadratic problems
- Local accelerated convergence was proved in the original paper.
- Recently was proved, that there is no global accelerated convergence for the method.
- Method was not extremely popular until the ML boom

Heavy ball method summary

- Ensures accelerated convergence for strongly convex quadratic problems
- Local accelerated convergence was proved in the original paper.
- Recently was proved, that there is no global accelerated convergence for the method.
- Method was not extremely popular until the ML boom
- Nowadays, it is de-facto standard for practical acceleration of gradient methods, even for the non-convex problems (neural network training)

Nesterov accelerated gradient

The concept of Nesterov Accelerated Gradient method

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

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

$$\begin{cases} y_{k+1} = x_k + \beta (x_k - x_{k-1}) \\ x_{k+1} = y_{k+1} - \alpha \nabla f(y_{k+1}) \end{cases}$$

The concept of Nesterov Accelerated Gradient method

$$x_{k+1} = x_k - \alpha \nabla f(x_k) \qquad x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1}) \qquad \begin{cases} y_{k+1} = x_k + \beta(x_k - x_{k-1}) \\ x_{k+1} = y_{k+1} - \alpha \nabla f(y_{k+1}) \end{cases}$$

Let's define the following notation

$$\begin{aligned} x^+ &= x - \alpha \nabla f(x) && \text{Gradient step} \\ d_k &= \beta_k (x_k - x_{k-1}) && \text{Momentum term} \end{aligned}$$

Then we can write down:

$$\begin{aligned} x_{k+1} &= x_k^+ && \text{Gradient Descent} \\ x_{k+1} &= x_k^+ + d_k && \text{Heavy Ball} \\ x_{k+1} &= (x_k + d_k)^+ && \text{Nesterov accelerated gradient} \end{aligned}$$

NAG convergence for quadratics

General case convergence

Theorem

Let $f : \mathbb{R}^n \to \mathbb{R}$ is convex and L-smooth. The Nesterov Accelerated Gradient Descent (NAG) algorithm is designed to solve the minimization problem starting with an initial point $x_0 = y_0 \in \mathbb{R}^n$ and $\lambda_0 = 0$. The algorithm iterates the following steps:

$$
\begin{aligned}
&\textbf{Gradient update:} && y_{k+1} = x_k - \frac{1}{L}\nabla f(x_k) \\
&\textbf{Extrapolation:} && x_{k+1} = (1-\gamma_k)y_{k+1} + \gamma_k y_k \\
&\textbf{Extrapolation weight:} && \lambda_{k+1} = \frac{1+\sqrt{1+4\lambda_k^2}}{2} \\
&\textbf{Extrapolation weight:} && \gamma_k = \frac{1-\lambda_k}{\lambda_{k+1}}
\end{aligned}
$$

The sequences $\{f(y_k)\}_{k\in\mathbb{N}}$ produced by the algorithm will converge to the optimal value f^* at the rate of $\mathcal{O}\left(\frac{1}{k^2}\right)$, specifically:

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

General case convergence

Theorem

Let $f : \mathbb{R}^n \to \mathbb{R}$ is μ-strongly convex and L-smooth. The Nesterov Accelerated Gradient Descent (NAG) algorithm is designed to solve the minimization problem starting with an initial point $x_0 = y_0 \in \mathbb{R}^n$ and $\lambda_0 = 0$. The algorithm iterates the following steps:

$$
\begin{aligned}
&\textbf{Gradient update:} && y_{k+1} = x_k - \frac{1}{L}\nabla f(x_k) \\
&\textbf{Extrapolation:} && x_{k+1} = (1-\gamma_k)y_{k+1} + \gamma_k y_k \\
&\textbf{Extrapolation weight:} && \gamma_k = \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}
\end{aligned}
$$

The sequences $\{f(y_k)\}_{k\in\mathbb{N}}$ produced by the algorithm will converge to the optimal value f^* linearly:

$$
f(y_k) - f^* \leq \frac{\mu + L}{2}\|x_0 - x^*\|_2^2 \exp\left(-\frac{k}{\sqrt{\kappa}}\right)
$$