4 Nesterov's fast gradient method

Is the gradient method optimal? Or is there another algorithm that can achieve faster rate of convergence?

4.1 Nesterov's fast gradient method

We will see that a simple (yet nontrivial!) modification of the gradient method allows us to boost the convergence rate from $O(1/k)$ to $O(1/k^2)$ for L-smooth functions. The algorithm is as follows:

Start with $x_0 \in \mathbb{R}^n, \theta_0 = 1, v_0 = x_0$ and iterate for $k = 0, 1, \ldots$:

$$\begin{cases} \text{If } k \geq 1\text{: choose } \theta_k \in (0,1) \text{ so that } \frac{(1-\theta_k)t_k}{\theta_k^2} \leq \frac{t_{k-1}}{\theta_{k-1}^2} \\ y = (1-\theta_k)x_k + \theta_k v_k \\ x_{k+1} = y - t_k \nabla f(y) \\ v_{k+1} = x_k + \frac{1}{\theta_k}(x_{k+1} - x_k) \end{cases} \tag{1}$$

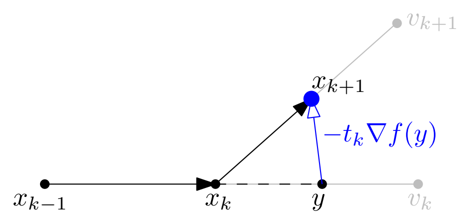

Figure 1: Iteration rule for the fast gradient method. y is defined as an extrapolation of x_k along the direction $x_k - x_{k-1}$, namely $y = x_k + \beta_k(x_k - x_{k-1})$. We evaluate the gradient of f at y and the new iterate is defined as $y - t_k \nabla f(y)$. We also show in this figure the iterates v_k. We show them in light gray because they are not "essential" for the algorithm (i.e., they can be eliminated). The only point to note here is that y is a θ-combination of x_k and v_k; and v_{k+1} is defined in such a way that x_{k+1} is a θ-combination (with the same θ) of x_k and v_{k+1}. It is easy to see from the picture that $v_{k+1} - v_k$ must be proportional to $\nabla f(y)$.

Some comments on the algorithm:

- The condition on θ_k looks complicated; it comes from the analysis of the sequences $\{x_k, v_k\}$. We will comment on the choice of θ_k later.
- The iterates v_k can be eliminated. In this case, the algorithm has only two steps per iteration: $y = x_k + \beta_k(x_k - x_{k-1})$ where $\beta_k = \theta_k(\theta_{k-1}^{-1} - 1)$ and $x_{k+1} = y - t_k \nabla f(y)$. See Figure 4.1 for an illustration.
- Algorithm (1) is very similar to a standard gradient method: the "only" difference is that the gradient is taken at a point y that is an extrapolation of x_k along the direction $x_k - x_{k-1}$.
- The defining property of v_{k+1} (last line of (1)) is that $x_{k+1} = (1-\theta_k)x_k + \theta_k v_{k+1}$. See also comment in Figure 4.1.

We now comment on the θ_k's:

- One can always find $\theta_k \in (0,1)$ such that the condition in the first line of the algorithm is always satisfied. In fact one can find a θ_k such that we have equality. This is given by $\theta_k = \frac{-a+\sqrt{a^2+4}}{2}$ where $a^2 = \theta_{k-1}^2 t_k/t_{k-1}$.
- When $t_k = t$ is fixed, one can check that the sequence $\theta_k = \frac{2}{k+2}$ satisfies the desired inequality $\frac{1-\theta_k}{\theta_k^2} \le \frac{1}{\theta_{k-1}^2}$ (but it does not satisfy equality)

We are now ready to prove convergence of the algorithm:

Theorem 4.1 (Nesterov)**.** *Let f be convex with L-Lipschitz continuous gradient. The iterations of (1) with constant step size $t_k = t \in (0, 1/L]$ and with $\theta_k = \frac{2}{k+2}$ satisfy*

$$f(x_k) - f^* \le \frac{2}{(k+1)^2 t}\|x_0 - x^*\|_2^2$$

for all $k \ge 1$.

Proof. We start like we did with the gradient method. We let $x^+ = y - t\nabla f(y)$. Then we have:

$$\begin{aligned} f(x^+) &\le f(y) + \left\langle \nabla f(y), (x^+ - y)\right\rangle + \frac{L}{2}\|x^+ - y\|_2^2 \\ &= f(y) - t\|\nabla f(y)\|_2^2(1 - Lt/2) \\ &\le f(y) - (t/2)\|\nabla f(y)\|_2^2 \end{aligned} \tag{2}$$

where we used that $0 < t \le 1/L$. By convexity of f we also have, for any $z \in \mathbb{R}^n$, $f(y) - f(z) \le \nabla f(y)^T(y - z)$. Combining this with (2), and using the fact that $\nabla f(y) = -\frac{1}{t}(x^+ - y)$ we get

$$\begin{aligned} f(x^+) - f(z) &\le f(y) - f(z) - (t/2)\|\nabla f(y)\|_2^2 \\ &\le \langle \nabla f(y), y - z\rangle - (t/2)\|\nabla f(y)\|_2^2 \\ &= -(t/2)\|\nabla f(y) - (1/t)(y - z)\|_2^2 + \frac{1}{2t}\|y - z\|_2^2 \\ &= \frac{1}{2t}\left[-\|x^+ - z\|_2^2 + \|y - z\|_2^2\right]. \end{aligned} \tag{3}$$

Until now this is the same as for the analysis of the gradient method [In the gradient method we had $y = x_k$, $z = x^*$, then we summed the inequality and the terms on the right-hand side telescoped].

What we will do here is that we will evaluate (3) at the points $z = x^*$ and $z = x$ and consider the convex combination with weights $\{\theta, 1-\theta\}$. Observe that the RHS of (3) is *affine* in z (this is apparent from the second line). Thus we get:

$$f(x^+) - (\theta f(x^*) + (1-\theta)f(x)) \le \frac{1}{2t}\left[\|y - (\theta x^* + (1-\theta)x)\|_2^2 - \|x^+ - (\theta x^* + (1-\theta)x)\|_2^2\right].$$

Now let's recall that $y = (1-\theta)x + \theta v$ (where v stands for v_k and v^+ for v_{k+1}). This implies that the first-term on the RHS of (3) is $\theta^2\|v - x^*\|_2^2$. Also recall that $x^+ = (1-\theta)x + \theta v^+$ and so the second-term on the RHS is (3) is $\theta^2\|v^+ - x^*\|_2^2$. Finally we get [with a slight rewrite of the LHS]

$$f(x_{k+1}) - f(x^*) - (1-\theta_k)(f(x_k) - f(x^*)) \le \frac{\theta_k^2}{2t}\left[\|x^* - v_k\|_2^2 - \|x^* - v_{k+1}\|_2^2\right]. \tag{4}$$

Rearranging to put the iterates $k+1$ on one side of the inequality, and the iterates k on the other side:

$$\frac{t}{\theta_k^2}(f(x_{k+1}) - f(x^*)) + \frac{1}{2}\|x^* - v_{k+1}\|_2^2 \quad \leq \quad \frac{(1-\theta_k)t}{\theta_k^2}(f(x_k) - f(x^*)) + \frac{1}{2}\|x^* - v_k\|_2^2 \tag{5}$$

Now we use the assumption that $(1-\theta_k)/\theta_k^2 \leq 1/(\theta_{k-1})^2$ to get:

$$\frac{t}{\theta_k^2}(f(x_{k+1}) - f(x^*)) + \frac{1}{2}\|x^* - v_{k+1}\|_2^2 \quad \leq \quad \frac{t}{\theta_{k-1}^2}(f(x_k) - f(x^*)) + \frac{1}{2}\|x^* - v_k\|_2^2. \tag{6}$$

Inequality above tells us that the quantity $V_k = \frac{t}{\theta_{k-1}^2}(f(x_k) - f(x^*)) + \frac{1}{2}\|x^* - v_k\|_2^2$ is nonincreasing with k. Thus we have $V_k \leq V_{k-1} \leq \cdots \leq V_1$ which gives

$$\begin{aligned}
\frac{t}{\theta_{k-1}^2}(f(x_k) - f(x^*)) + \frac{1}{2}\|x^* - v_k\|_2^2 \quad &\leq \quad \frac{t}{\theta_0^2}(f(x_1) - f(x^*)) + \frac{1}{2}\|x^* - v_1\|_2^2 \\
&\leq \quad \frac{(1-\theta_0)t}{\theta_0^2}(f(x_0) - f(x^*)) + \frac{1}{2}\|x^* - v_0\|_2^2 \\
&= \quad \frac{1}{2}\|x^* - x_0\|_2^2
\end{aligned}$$

where the second line follows from (5) with $k = 0$, and the last line uses $\theta_0 = 1$ and $v_0 = x_0$. Thus we get $f(x_k) - f^* \leq \frac{\theta_{k-1}^2}{2t}\|x^* - x_0\|_2^2$, and with $\theta_{k-1} = \frac{2}{k+1}$ we get the desired rate. $\square$

Some remarks on the algorithm:

Descent The fast gradient method is not a descent method, i.e., it is possible that $f(x_{k+1}) > f(x_k)$ (unlike the gradient method). The convergence analysis proves however that a certain combination of $f(x_k) - f^*$ and $\|x^* - v_k\|_2^2$ decreases with k (cf. Equation (6)).

Backtracking line search One can also prove convergence of the algorithm with a backtracking line search, rather than a constant line search. The only requirement on the step size t_k is that inequality (2) is satisfied; this is the only thing needed in the convergence proof. The scheme works as follows: Starting with $t_k = \hat{t} > 0$, keep updating $t_k = \beta t_k$ with $\beta \in (0,1)$ until condition (2) is satisfied. (Note that the latter condition can be more succintly written as $f(x_{k+1}) \leq f(y) - \frac{t_k}{2}\|\nabla f(y)\|_2^2$.) Also note that each time t_k is updated, one has to recompute θ_k, y, and x_{k+1}. In all, the line search at iteration k proceeds as follows:

Start with $t_k = \hat{t}$, and compute associated θ_k, y, x_{k+1}
While $f(x_{k+1}) > f(y) - \frac{t_k}{2}\|\nabla f(y)\|_2^2$
  Update $t_k = \beta t_k$
  Compute θ_k such that $\frac{1-\theta_k}{\theta_k^2} t_k \leq \frac{t_{k-1}}{\theta_{k-1}^2}$
  Compute $y = (1-\theta_k)x_k + \theta_k v_k$
  Compute $x_{k+1} = y - t_k \nabla f(y)$

Illustration Consider the function $f(x) = \sum_{i=1}^N \log\left(1 + e^{a_i^T x + b_i}\right)$ which we considered in the previous lecture. The plot below compares the standard gradient method with the fast gradient method, and we observe that the latter converges faster.

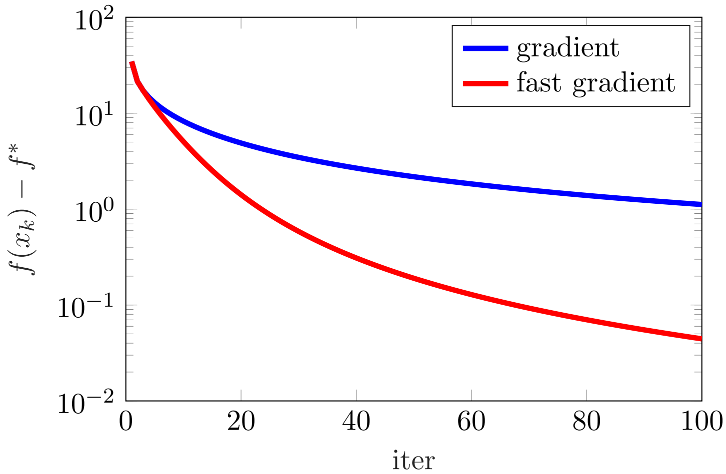

Figure 2: Fast gradient method for logistic regression

Strongly convex case We have seen in Lecture 3 that when the function f is m-strongly convex, the gradient method with step size $t = 2/(m+L)$ converges at a linear rate $\approx (1 - \frac{1}{\kappa})^{2k}$ where $\kappa = \frac{L}{m} \geq 1$ is the condition number. What about the fast gradient method? If we know the strong convexity parameter $m > 0$, algorithm (1) can be slightly modified to incorporate this knowledge. We do not give the general algorithm (as we did in Equation (1)), but only an important special case, where $t_k = 1/L$ and a specific choice of θ_k. The algorithm reads:

$$\begin{cases} y = x_k + \frac{1-\sqrt{m/L}}{1+\sqrt{m/L}}(x_k - x_{k-1}) \\ x_{k+1} = y - (1/L)\nabla f(y). \end{cases} \tag{7}$$

One can prove that if f is m-strongly convex and ∇f is L-Lipschitz, then the convergence rate of (7) is $\approx (1 - \sqrt{1/\kappa})^{2k}$. This means that we reach accuracy ϵ in at most $O(\sqrt{\frac{L}{m}}\log(1/\epsilon))$ iterations. This can be much smaller than the $O(\frac{L}{m}\log(1/\epsilon))$ iterations of the gradient method [cf. Lecture 3].

One drawback of the algorithm (7) is that it relies on the knowledge of m which can sometimes be difficult to estimate. (Note that the gradient method does not require knowledge of m. In lecture 3 we assumed $t_k = 2/(m+L)$ but one can easily see that $t_k = 1/L$ also gives a linear convergence rate of the form $(1 - 1/\kappa)^k$.) Several improvements and adaptations that avoid knowledge of m have been proposed recently in the literature, see e.g., [OC15, Section 2.1].

4.2 Lower complexity bounds

It turns that $O(1/k^2)$ is the best rate one can get for minimization of L-smooth convex functions, assuming we only have access to gradients of f.

A *first-order* algorithm is one that has access to function values $f(x)$ and gradients $\nabla f(x)$. The complexity of such an algorithm is the number of queries it makes. We consider here algorithms that satisfy the following assumption: the k'th iterate/query point x_k of the algorithm satisfies:

$$x_k \in x_0 + \operatorname{span}\left\{\nabla f(x_0), \nabla f(x_1), \ldots, \nabla f(x_{k-1})\right\}. \tag{8}$$

Clearly the gradient and fast gradient methods satisfy this assumption.

Define $\mathcal{F}_L = \{f : \mathbb{R}^n \to \mathbb{R}$ convex with L-Lipschitz gradient$\}$. We want to understand how well can first-order algorithms behave on functions in $\mathcal{F}_L$. The next theorem, due to Nesterov, shows that $O(1/k^2)$ is the best rate one can hope for.

Theorem 4.2 (Nesterov)**.** *Fix $L > 0$ and an integer $k \geq 1$. For any algorithm satisfying (8), there is a function $f \in \mathcal{F}_L$ on $n = 2k+1$ variables such that after k steps of the algorithm*

$$f(x_k) - f^* \geq \frac{3}{32} \frac{L\|x_0 - x^*\|_2^2}{(k+1)^2} \tag{9}$$

and

$$\|x_k - x^*\|_2^2 \geq \frac{1}{8}\|x_0 - x^*\|_2^2. \tag{10}$$

Proof. Let $n = 2k+1$ and consider the function $f : \mathbb{R}^n \to \mathbb{R}$ as follows

$$f(x) = \frac{L}{8}\left(x_n^2 + \sum_{i=1}^{n-1}(x_{i+1} - x_i)^2 + x_1^2 - 2x_1\right). \tag{11}$$

Let also, for $i = 1, \ldots, n$ $V_i = \{x \in \mathbb{R}^n : x_{i+1} = \cdots = x_n = 0\}$. Then we have the following properties about f:

(i) $f \in \mathcal{F}_L$

(ii) The minimum of f is attained at $x^* = \left(\frac{n}{n+1}, \ldots, \frac{2}{n+1}, \frac{1}{n+1}\right)$ and the optimal value is $f^* = -\frac{L}{8}\frac{n}{n+1}$. More generally the minimum of f on the subspace V_i is $-\frac{L}{8}\frac{i}{i+1}$, attained at the point $\left(\frac{i}{i+1}, \ldots, \frac{2}{i+1}, \frac{1}{i+1}, 0, \ldots, 0\right) \in V_i$.

(iii) If $x \in V_i$ for $i < n$, then $\nabla f(x) \in V_{i+1}$.

We leave it to the reader to check these properties.

Assume without loss of generality that the first query point of the algorithm is $x_0 = 0$ (if it is not we simply consider the function $\tilde{f}(x) = f(x - x_0)$). By property (iii) of f, and by assumption (8) on the algorithm this means that the k'th query point x_k of the algorithm must belong to V_k. Thus this means that

$$f(x_k) \geq \min_{x \in V_k} f(x) = -\frac{L}{8}\frac{k}{k+1}.$$

Now using the fact that $n = 2k+1$ and $f^* = -\frac{L}{8}\frac{n}{n+1}$ we get

$$f(x_k) - f^* \geq \frac{L}{8}\left(\frac{2k+1}{2k+2} - \frac{k}{k+1}\right) = \frac{L}{8}\frac{1}{2k+2}.$$

Also note that $\|x_0 - x^*\|_2^2 = \|x^*\|_2^2 = \frac{1}{(n+1)^2}\sum_{i=1}^{n-1} i^2 = \frac{n}{n+1}\frac{2n+1}{6} \leq \frac{n+1}{3}$, thus

$$\frac{f(x_k) - f^*}{\|x_0 - x^*\|_2^2} \geq \frac{L}{8}\frac{1}{2k+2}\frac{3}{2k+2} = \frac{3L}{32}\frac{1}{(k+1)^2}$$

as desired.

We now prove (10). Since $x_k = (?, \ldots, ?, 0, \ldots, 0)$ then $x_k - x^* = \left(?, \ldots, ?, -\frac{n-k}{n+1}, \ldots, -\frac{1}{n+1}\right)$ which implies $\|x_k - x^*\|_2^2 \geq \frac{1}{(n+1)^2}\sum_{i=1}^{n-k} i^2$. Now using the fact that $n = 2k+1$ we get $\|x_k - x^*\|_2^2 \geq \frac{1}{24}(2k+3)$. Combining with $\|x_0 - x^*\|_2^2 \leq \frac{2k+2}{3}$ we get $\|x_k - x^*\|_2^2 \geq \frac{1}{8}\|x_0 - x^*\|_2^2$ as desired. $\square$

Strongly convex functions: Let $\mathcal{F}_{m,L} = \{f : \mathbb{R}^n \to \mathbb{R}$ m-strongly convex and L-smooth$\}$. One can show in a similar way as the proof above, that for any first-order algorithm $\mathcal{A}$ that runs for k iterations, there is a function $f \in \mathcal{F}_{m,L}$ such that the k'th iterate of $\mathcal{A}$ on f satisfies:

$$f(x_k) - f^* \gtrapprox m \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{2k} \|x_0 - x^*\|^2.$$

This means that to reach accuracy ϵ, one needs at least $\approx \sqrt{L/m} \log(1/\epsilon)$ iterations.

References

[OC15] Brendan O'Donoghue and Emmanuel Candès. Adaptive restart for accelerated gradient schemes. *Foundations of computational mathematics*, 15(3):715–732, 2015. 4