University of California, Berkeley Lecturer: Benjamin Recht

CS227C/STAT260 Convex Optimization and Approximation: Optimization for Modern Data Analysis

January 29, 2015 Notes: Ashia Wilson and Benjamin Recht

Lecture 4: Momentum

We saw last lecture that the gradient method achieved a linear rate of convergence for strongly convex functions. Namely, if the gradients were L-Lipschitz, and f was strongly convex with parameter m, then

$$O\left(\frac{L}{m}\log(1/\epsilon)\right)$$

iterates sufficed to converge to an accuracy of ϵ. For weakly convex functions,

$$O\left(\frac{L}{\epsilon}\right)$$

iterations suffice. The question remains: is this the best we can do? In this lecture we describe a minor modification to the gradient method that results in a major reduction in iteration complexity.

1 Appealing to differential equations for motivation

We described in the last lecture that the gradient method could be overly greedy, always taking the steepest descent direction. When the contours of f are very narrow and elongated, this strategy results in oscillation.

One possible way to avert this situation is to add *momentum* to the iterate. That is, in the spirit of the Wolfe conditions, if the direction we were moving was a good one, we might consider moving along this direction for more than one step.

The intuition for these momentum methods comes from looking at our algorithm as a differential equation. The gradient method is akin to moving down a potential well where the potential is defined by the gradient of f:

$$\frac{dx}{dt} = -\nabla f(x)$$

This differential equation has fixed points precisely when $\nabla f(x) = 0$.

Of course, there are other differential equations whose stationary points are precisely those where the gradient vanishes. In particular, the second order differential equation:

$$\mu\frac{d^2x}{dt^2} = -\nabla f(x) - b\frac{dx}{dt}$$

also converges to a point where $\nabla f(x) = 0$. This corresponds to the Euler-Lagrange equations associated with an object in a potential well in the presence of friction or viscosity. If $b = 0$, then the system may always gain acceleration. Moreover, the trajectories will tend to continue to move in the direction they were moving before (heavier objects move down hill faster than light objects in the presence of friction).

If we approximate this ODE using simple finite differences, we have

$$\mu \frac{x(t+\Delta t) - 2x(t) + x(t-\Delta t)}{\Delta t^2} \approx -\nabla f(x(t)) - b\frac{x(t) - x(t-\Delta t)}{\Delta t}$$

Rearranging the terms in this expression gives the finite difference equation:

$$x(t+\Delta t) = x(t) - \frac{\Delta t^2}{\mu}\nabla f(x(t)) + \left(1 - \frac{b}{\mu}\Delta t\right)(x(t) - x(t-\Delta t))$$

which is precisely the momentum method we are interested in.

2 Momentum methods

The method we derived in the previous section is known as *the heavy ball method*, as it appeals to physical intuition. Rewriting things in a more pleasant way we have

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

where α and β are positive constants to be determined. Define

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

With this identification, we can rewrite the iteration in terms of two sequences:

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

A similar algorithm is known as *Nesterov's accelerated* method or *Nesterov's optimal method*:

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

This method only differs in their discretization of the ODE. In the two state version, Nesterov's method becomes

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

Another popular way to write Nesterov's accelerated method is by the iterations

$$\begin{aligned} z_k &= x_k + \beta_k(x_k - x_{k-1}) \\ x_{k+1} &= z_k - \alpha_k\nabla f(z_k). \end{aligned}$$

All three of these formulations are equivalent to one another.

It turns out that Nesterov's method converges more rapidly than the Heavy Ball method for general convex functions. This is peculiar, because as you will see in next week's homework, they achieve the same rate of convergence when f is a quadratic. Nonetheless, we will focus our attention on Nesterov's method, even though the two approaches look very similar to one another.

3 Analysis of the Nesterov's method for convex quadratic functions

Later in the semester, we will analyze Nesterov's method for general convex f. But, unfortunately, the proofs are still a bit unintuitive or require some more advanced concepts that we have not yet covered. So we will focus our attention on quadratics for now. But one must be careful! Just because one proves that an algorithm converges for all strongly convex quadratics, it does not mean that the method necessarily converges for non-quadratic functions. A simple one dimensional example will be studied in the homework.

Consider the inhomogenous convex quadratic objective

$$f(x) = \frac{1}{2}x^T Q x - p^T x + r$$

where $LI \succeq Q \succeq mI$. Strongly convex quadratics can be solved in closed form by direct calculation.

$$x_\star = Q^{-1}p$$

is the minimizer of f and

$$\nabla f(x) = Qx - p = Q(x - x_\star)$$

Plugging in these simplications, our algorithm takes the form

$$\begin{aligned} x_{k+1} - x_\star &= x_k - x_\star - \alpha Q(x_k + \beta(x_k - x_{k-1}) - x_\star) + \beta(x_k - x_{k-1}) \\ &= x_k - x_\star - \alpha Q(x_k + \beta((x_k - x_\star) - (x_{k-1} - x_\star)) - x_\star) + \beta((x_k - x_\star) - (x_{k-1} - x_\star)) \,. \end{aligned}$$

In this second equality, we just added and subtracted $x_\star$ a few times. This trick let's us write the algorithm as the simple matrix equation

$$\begin{bmatrix} x_{k+1} - x_\star \\ x_k - x_\star \end{bmatrix} = T \begin{bmatrix} x_k - x_\star \\ x_{k-1} - x_\star \end{bmatrix} \tag{1}$$

Where T is the matrix

$$T := \begin{bmatrix} (1+\beta)(I - \alpha Q) & -\beta(I - \alpha Q) \\ I & 0 \end{bmatrix} \tag{2}$$

This matrix governs the evolution of the algorithm.

We want to show that all trajectories of the algorithm converge to 0, implying that x_k converges to $x_\star$. The following theorem guarantees such convergence.

Definition 1 *The* spectral radius *of a matrix A is equal to $\rho(A) := \max\{|\lambda| \,:\, \lambda \text{ is an eigenvalue of } A\}$.*

Theorem 1 *Suppose $A \in \mathbb{R}^{d\times d}$. Then $\rho(A) < \rho$ if and only if there exists a $P \succ 0$ satisfying $A^T P A - \rho^2 P \prec 0$.*

Proof If $\rho(A) < \rho$, then the matrix

$$P := \sum_{k=0}^{\infty} \rho^{-2k} (A^k)^T (A^k)$$

is well defined, positive definite because the first term in the sum is a multiple of the identity, and satisfies $A^T P A - \rho^2 P = -\rho^2 I_d \prec 0$. Conversely, assume the LMI has a solution $P \succ 0$ and let λ be an eigenvalue of A with corresponding eigenvector v. Then

$$0 > v^T A^T P A v - \rho^2 v^T P v = (|\lambda|^2 - \rho^2) v^T P v$$

But since $v^T P v > 0$, we must have that $|\lambda| < \rho$. ■

Now suppose we are studying the iteration (1). Then, if there exists a $P \succ 0$ satisfying $T^T P T - \rho^2 P \prec 0$, we have

$$\begin{bmatrix} x_{k+1} - x_\star \\ x_k - x_\star \end{bmatrix}^T P \begin{bmatrix} x_{k+1} - x_\star \\ x_k - x_\star \end{bmatrix} < \rho^2 \begin{bmatrix} x_k - x_\star \\ x_{k-1} - x_\star \end{bmatrix}^T P \begin{bmatrix} x_k - x_\star \\ x_{k-1} - x_\star \end{bmatrix} \tag{3}$$

along all trajectories. If $\rho < 1$, then the sequence $\{x_k\}$ converges linearly to $x_\star$. Iterating (3) down to $k = 0$, we see that

$$\begin{bmatrix} x_k - x_\star \\ x_{k-1} - x_\star \end{bmatrix}^T P \begin{bmatrix} x_k - x_\star \\ x_{k-1} - x_\star \end{bmatrix} < \rho^{2k} \begin{bmatrix} x_0 - x_\star \\ x_{-1} - x_\star \end{bmatrix}^T P \begin{bmatrix} x_0 - x_\star \\ x_{-1} - x_\star \end{bmatrix} .$$

Assuming that $x_0 = x_{-1}$, this implies that

$$\|x_k - x_\star\| < \sqrt{2 \operatorname{cond}(P)}\, \rho^k \|x_0 - x_\star\|$$

where $\operatorname{cond}(P)$ is the condition number of P. The function

$$V(x, z) = \begin{bmatrix} x - x_\star \\ z - x_\star \end{bmatrix}^T P \begin{bmatrix} x - x_\star \\ z - x_\star \end{bmatrix}$$

is a Lyapunov function for the algorithm. This function strictly decreases over all trajectories and hence certifies that the algorithm is *stable*, i.e., converges to nominal values. For quadratic f, we are able to construct a quadratic Lyapunov function by doing an elementary eigenvalue analysis. But this proof doesn't generalize to the non-quadratic case, which is why we reserve the study of Nesterov's method on more general convex functions for later in the semester.

With this theorem in hand, it suffices to find parameters α and β such that T has its eigenvalues to have small modulus as possible.

Proposition 2 *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex quadratic function $f(x) = \frac{1}{2} x^T Q x - p^T x + r$ with $mI \preceq Q \preceq LI$. Let $\kappa := L/m$. Then Nesterov's accelerated method with $\alpha = \frac{1}{L}$ and $\beta = \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}$ converges and $\rho(T) \leq 1 - \kappa^{-1/2}$.*

Proof We will upper bound the spectral radius of T by explicitly computing all of the eigenvalues and upper bounding their magnitudes.

To proceed, write the eigenvalue decomposition of Q as $Q = U_0 \Lambda U_0^T$, where $\Lambda = \operatorname{diag}(\lambda_1, \lambda_2, \ldots, \lambda_d)$. By defining the permutation matrix Π as follows:

$$\Pi_{ij} = \begin{cases} 1 & i \text{ odd}, \ j = (i+1)/2 \\ 1 & i \text{ even}, \ j = d + (i/2) \\ 0 & \text{otherwise} \end{cases} .$$

we have by applying a similarity transformation to the matrix T that

$$\begin{aligned}
&\Pi \begin{bmatrix} U_0 & 0 \\ 0 & U_0 \end{bmatrix}^T \begin{bmatrix} (1+\beta)(I-\alpha Q) & -\beta(I-\alpha Q) \\ I & 0 \end{bmatrix} \begin{bmatrix} U_0 & 0 \\ 0 & U_0 \end{bmatrix} \Pi^T \\
&= \Pi \begin{bmatrix} (1+\beta)(I-\alpha\Lambda) & -\beta(I-\alpha\Lambda) \\ I & 0 \end{bmatrix} \Pi^T \\
&= \begin{bmatrix} T_1 & 0 & \cdots & 0 \\ 0 & T_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & T_d \end{bmatrix},
\end{aligned}$$

where

$$T_i = \begin{bmatrix} (1+\beta)(1-\alpha\lambda_i) & -\beta(1-\alpha\lambda_i) \\ 1 & 0 \end{bmatrix}, \quad i = 1, 2, \ldots, d.$$

The eigenvalues of T are the eigenvalues of T_i, for $i = 1, 2, \ldots, d$. The eigenvalues of T_i are the roots of the following quadratic:

$$\nu_{i,j}{}^2 - (1+\beta)(1-\alpha\lambda_i)\nu_{i,j} + \beta(1-\alpha\lambda_i) = 0, \tag{4}$$

which are given by the formula for the quadratic equation:

$$\begin{aligned}
\nu_{i,1} &= \frac{1}{2}\left[(1+\beta)(1-\alpha\lambda_i) + i\sqrt{4\beta(1-\alpha\lambda) - (1+\beta)^2(1-\alpha\lambda_i)^2}\right], \\
\nu_{i,2} &= \frac{1}{2}\left[(1+\beta)(1-\alpha\lambda_i) - i\sqrt{4\beta(1-\alpha\lambda) - (1+\beta)^2(1-\alpha\lambda_i)^2}\right].
\end{aligned}$$

The two roots are distinct complex numbers when $1-\alpha\lambda_i > 0$ and $(1+\beta)^2(1-\alpha\lambda_i) < 4\beta$, which happens when with our choice of α and β when $m < \lambda < L$. When $\lambda = L$, $\nu_{i,1} = \nu_{i,2} = 0$. When $\lambda = m$, $\nu_{i,1} = \nu_{i,2} = 1 - \sqrt{m/L}$.

It remains to bound the magnitude of the $\nu_{i,j}$ for when $\lambda_i \in (m, L)$.

The magnitude is given by

$$\frac{1}{2}\sqrt{(1+\beta)^2(1-\alpha\lambda_i)^2 + 4\beta(1-\alpha\lambda) - (1+\beta)^2(1-\alpha\lambda_i)^2} = \frac{1}{2}\sqrt{4\beta(1-\alpha\lambda)} = \sqrt{\beta}\sqrt{1-\frac{\lambda_i}{L}}$$

Note that

$$\begin{aligned}
\sqrt{\beta}\sqrt{1-\frac{\lambda_i}{L}} \leq \sqrt{\beta}\sqrt{1-\frac{m}{L}} &= \left(\frac{\sqrt{L}-\sqrt{m}}{\sqrt{L}+\sqrt{m}} \cdot \frac{L-m}{L}\right)^{1/2} \\
&= \left(\frac{\sqrt{L}-\sqrt{m}}{\sqrt{L}+\sqrt{m}} \cdot \frac{(\sqrt{L}-\sqrt{m})(\sqrt{L}+\sqrt{m})}{L}\right)^{1/2} \\
&= \frac{\sqrt{L}-\sqrt{m}}{\sqrt{L}} = 1-\sqrt{m/L}\,.
\end{aligned}$$

which verifies our bound on the spectral radius.

■

Note that if the rate of convergence of the algorithm is given by $(1-\kappa^{-1/2})$, the number of iterations required to reach tolerance ϵ scales as $O(\sqrt{\kappa})$. This is a square root of the number of iterations required by the gradient method.

4 Weakly convex functions

When f is weakly convex, the parameter β is chosen slightly differently. We keep $\alpha = 1/L$ and set

$$\beta_k = \theta_k(\theta_{k-1}^{-1} - 1) \qquad \text{where } \theta_0 = 1,\ \ \theta_k = \frac{1}{2}\left[-\theta_{k-1}^2 + \sqrt{\theta_{k-1}^4 + 4\theta_{k-1}^2}\right].$$

This rather bizarre expression pops out of the proof of Nesterov's method. It is a bit magical, and even Nesterov refers to this as "an algebraic trick" to prove convergence. We will dive into a different derivation after we have discussed Mirror Descent.

In the meantime, let's at least describe the result. The main take away is that Nesterov's method satisfies:

$$f(x_k) - f(x_\star) \leq \tfrac{4L}{(k+2)^2}\|x_0 - x_\star\|^2$$

So, just as was the case for strongly convex f, the number of iterations required to achieve tolerance ϵ scales as $\epsilon^{-1/2}$, or the square root of the number of iterations required for the gradient method.

5 Conjugate Gradient Method

The problem with Nesterov's method as presented is that one is required to know the convexity parameters L and m to compute the appropriate step-sizes. The conjugate gradient method is a simple modification of Nesterov's method that does not require knowledge of these parameters. Consider the momentum iterations

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

This is equivalent to our previous iteration after a change of variables. To choose α_k, we can pick the stepsize to minimize f along the direction y_k (i.e. $\min_{\alpha>0} f(x_k + \alpha y_k)$). For quadratics, if we take derivatives, this gives us

$$\alpha_k = \frac{y_k^T r_k}{y_k^T Q y_k} \qquad \text{where } r_k = Qx_k - p$$

Definition 2 *We say that two non-zero vectors u and v are conjugate (with respect to Q) if*

$$u^T Q v = 0\,.$$

We now pick β_k so that $\langle y_k, Q y_{k-1}\rangle = 0$ (i.e to make y_k and y_{k-1} conjugate).

$$\begin{aligned} \langle y_k, Q y_{k-1}\rangle &= \langle -r_{k-1} + \beta y_{k-1}, Q y_{k-1}\rangle \\ &= \langle -r_{k-1}, Q y_{k-1}\rangle + \beta_k \langle y_{k-1}, Q y_{k-1}\rangle \end{aligned}$$

$$\beta_k = \frac{\langle r_{k-1}, Qy_{k-1} \rangle}{\langle y_{k-1}, Qy_{k-1} \rangle}$$

Conjugacy guarantees that the directions y_k are orthogonal with respect to the inner product

$$\langle u, v \rangle_Q := u^T Q v \,.$$

Consequently, walking along conjugate directions using exact line search yields convergence to $x_\star$ in n steps.

Unfortunately, the conjugate gradient method doesn't admit particularly rigorous analysis when f is not quadratic.