Nesterov's Acceleration

Raghav Somani

March 30, 2018

This article contains a summary and survey of the Nesterov's accelerated gradient descent method and some insightful implications that can be derived from it.

The oracle in consideration is the first order deterministic oracle where each query is a point $\mathbf{x} \in \mathbb{R}^d$ in the space, and the oracle outputs a tuple of vectors $(f(\mathbf{x}), g(\mathbf{x}))$, where $g(\mathbf{x}) \in \partial f(\mathbf{x})$ for a continuous function $f : \mathbb{R}^d \to \mathbb{R}$. The dynamics of acceleration can be understood precisely if we consider the right and "easy" model - convex quadratic. Therefore we first consider our optimization function to be the loss function of the naive linear regression problem.

$$f(\mathbf{x}) = \frac{1}{2n} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 \tag{0.0.1}$$

where $\mathbf{A} \in \mathbb{R}^{n \times d}$ and $\mathbf{b} \in \mathbb{R}^n$. The goal is to converge to the optimal solution to the convex problem $\mathbf{x}^*$. We assume that the function f is strongly convex, i.e., $\nabla^2 f(\mathbf{x}) \succ 0$.

$$\begin{aligned} f(\mathbf{x}) &= \frac{1}{2n}\left[\mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} - 2\mathbf{x}^T\mathbf{A}^T\mathbf{b} + \|\mathbf{b}\|_2^2\right] \\ \nabla f(\mathbf{x}) &= \frac{1}{2n}\left[2\mathbf{A}^T\mathbf{A}\mathbf{x} - 2\mathbf{A}^T\mathbf{b}\right] \\ \nabla^2 f(\mathbf{x}) &= \frac{1}{n}\mathbf{A}^T\mathbf{A} \end{aligned}$$

Let us denote the Hessian of the function f as $\mathbf{\Sigma}$. The assumption therefore requires the least eigenvalue of the matrix $\mathbf{\Sigma}$ to be positive. Since every symmetric matrix has an eigenvalue decomposition, therefore let the same for $\mathbf{\Sigma}$ be

$$\mathbf{\Sigma} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T \tag{0.0.2}$$

where $\mathbf{\Lambda} = diag(\lambda_1, \lambda_2, \ldots, \lambda_d)$ is the eigenvalues matrix with $\lambda_1 > \lambda_2 > \cdots > \lambda_d$ being the eigenvalues of $\mathbf{\Sigma}$. The matrix $\mathbf{Q}$ contains the eigenvectors of $\mathbf{\Sigma}$ and is an orthogonal matrix.
To appreciate acceleration, we first analyze the naive gradient descent method for the convex quadratic optimization problem (0.0.1).

1 Gradient Descent

Gradient Descent is a greedy, cheap and a locally optimal way to decrease a convex function's value is to iteratively move in a negative sub-gradient direction. Algorithmically, we start at an initial iterate $\mathbf{x}_0$ and the $(t+1)^{th}$ update rule is written as

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \eta_t \mathbf{g}_t \tag{1.0.1}$$

where $\mathbf{g}_t \in \partial f(\mathbf{x}_t)$ and η_t is the step-size. Notice that we can write $f(\mathbf{x})$ as

$$\begin{aligned} f(\mathbf{x}) &= \frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}\mathbf{x} - \frac{1}{n}\mathbf{b}^T\mathbf{A}\mathbf{x} + \frac{1}{2n}\|\mathbf{b}\|_2^2 \\ \nabla f(\mathbf{x}) &= \mathbf{\Sigma}\mathbf{x} - \frac{1}{n}\mathbf{A}^T\mathbf{b} \end{aligned}$$

Define $\mathbf{y} := \frac{1}{n}\mathbf{A}^T\mathbf{b}$, so we have $\nabla f(\mathbf{x}) = \mathbf{\Sigma}\mathbf{x} - \mathbf{y}$. By equating the gradient to $\mathbf{0}$, we have a closed form solution of the problem, $\mathbf{x}^* = \mathbf{\Sigma}^{-1}\mathbf{y}$. The inverse of $\mathbf{\Sigma}$ exists because of the fact that it is positive definite.

1.1 Decomposition of parameter error

The gradient at a point $\mathbf{x}_t$ can be written as $\mathbf{g}_t = \mathbf{\Sigma}(\mathbf{x}_t - \mathbf{x}^*)$. Plugging this expression of the gradient in the gradient descent step equation (1.0.1), we get

$$\begin{aligned} \mathbf{x}^{t+1} &= \mathbf{x}^t - \eta_t \mathbf{\Sigma}(\mathbf{x}_t - \mathbf{x}^*) \\ \mathbf{x}^{t+1} - \mathbf{x}^* &= \mathbf{x}^t - \mathbf{x}^* - \eta_t \mathbf{\Sigma}(\mathbf{x}_t - \mathbf{x}^*) \end{aligned} \tag{1.1.1}$$

The vector $\mathbf{x}_t - \mathbf{x}^*$ denotes the position of $\mathbf{x}_t$ from $\mathbf{x}^*$ in the canonical basis. Let us view the same dynamics from the eigenbasis of $\mathbf{\Sigma}$. For this, we do a basis transformation and define $\mathbf{z}_t = \mathbf{Q}^T(\mathbf{x}^t - \mathbf{x}^*)\ \forall\ t \geq 0$. Multiplying Equation (1.1.1) with $\mathbf{Q}^T$ from the left, we obtain

$$\mathbf{Q}^T(\mathbf{x}^{t+1} - \mathbf{x}^*) = \mathbf{Q}^T(\mathbf{x}^t - \mathbf{x}^*) - \eta_t \mathbf{Q}^T \mathbf{\Sigma}(\mathbf{x}_t - \mathbf{x}^*) \tag{1.1.2}$$

Using the eigen-decomposition of $\mathbf{\Sigma}$ as in (0.0.2) and the definition of $\mathbf{z}_t$, we can re-write (1.1.2) as

$$\begin{aligned} \mathbf{z}_{t+1} &= \mathbf{z}_t - \eta_t \mathbf{Q}^T \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T (\mathbf{x}_t - \mathbf{x}^*) \\ &= \mathbf{z}_t - \eta_t \mathbf{\Lambda} \mathbf{z}_t \end{aligned} \tag{1.1.3}$$

Each component of the vector $\mathbf{z}_t$ denotes the "closeness" of the iterate $\mathbf{x}_t$ in each eigenvector direction, and from (1.1.3) we have a closed form expression of the same

$$\begin{aligned} z_{t+1}^{(i)} &= z_t^{(i)} - \eta_t \lambda_i z_t^{(i)} \\ &= (1 - \eta_t \lambda_i) z_t^{(i)} \\ &= (1 - \eta \lambda_i)^{t+1} z_t^{(0)} \qquad (\text{for } \eta_t = \eta\ \forall\ t \geq 0) \end{aligned} \tag{1.1.4}$$

This shows that for $\eta_t \leq \frac{1}{\lambda_1}$, along each eigenvector direction, gradient descent converges geometrically with a factor that is linear in the corresponding eigenvalue. Therefore for eigen-directions with high eigenvalues, we have a good geometric progress, and for eigen-directions with low eigenvalues, we have poorer geometric progress. This proportionate dependence of eigenvalues on the geometric contraction level of progress is the reason why gradient descent are hugely dependent on the underlying geometry of the optimization landscape. The same phenomenon can be seen in the function values. Let us see how.

1.2 Decomposition of function error

Since $f(\mathbf{x})$ is a quadratic convex function, the Taylor expansion of $f(\mathbf{x})$ up to the second term is exact representation of the function value. Therefore let us look at the Taylor series expansion of $f(\mathbf{x})$ at the optimum point $\mathbf{x}^*$. We use the fact that the gradient of f at $\mathbf{x}^*$ is $\mathbf{0}$ and therefore the sub-optimality gap of a point $\mathbf{x}$ comprises of just one term.

$$\begin{aligned} f(\mathbf{x}) - f(\mathbf{x}^*) &= \frac{1}{2}(\mathbf{x} - \mathbf{x}^*)^T \nabla^2 f(\mathbf{x}^*)(\mathbf{x} - \mathbf{x}^*) \\ &= \frac{1}{2}(\mathbf{x} - \mathbf{x}^*)^T \mathbf{\Sigma}(\mathbf{x} - \mathbf{x}^*) \end{aligned} \tag{1.2.1}$$

We can break the sub-optimality gap in (1.2.1) along each eigen-vector direction and analyze its convergence separately along each direction. Using the eigen-basis transformation $\mathbf{z}_t = \mathbf{Q}^T(\mathbf{x}_t - \mathbf{x}^*)$, we can re-write (1.1.4) as

$$\begin{aligned} f(\mathbf{x}_t) - f(\mathbf{x}^*) &= \frac{1}{2}(\mathbf{x}^T - \mathbf{x}^*)^T \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T (\mathbf{x}^T - \mathbf{x}^*) \\ &= \mathbf{z}_t^T \mathbf{\Lambda} \mathbf{z}_t \\ &= \sum_{i=1}^{d} \lambda_i (z_t^{(i)})^2 \\ &= \sum_{i=1}^{d} \lambda_i (1 - \eta \lambda_i)^{2t} (z_0^{(i)})^2 \end{aligned} \tag{1.2.2}$$

As we see, again the sub-optimality gap depends aggressively on the magnitude of the eigenvalue along each eigen-direction.

As we shall see, the dependence of eigenvalues on the convergence of iterates along each eigen-direction will be less severe when we analyze the same optimization problem but using the Nesterov style update.

2 Nesterov's Accelerated Gradient Descent

Nesterov's accelerated gradient descent on the other hand is not locally greedy, cheap algorithm which achieves convergence rates that match the lower bound iteration complexity for convex objectives. Algorithmically, we start with two iterates $\mathbf{x}_0$ and $\mathbf{y}_0$ such that $\mathbf{x}_0 = \mathbf{y}_0$ and the $(t+1)^{th}$ update rule can be written as

$$\mathbf{x}_{t+1} = \mathbf{y}_t - \alpha \nabla f(\mathbf{y}_t) \tag{2.0.1a}$$

$$\begin{aligned}\mathbf{y}_{t+1} &= \mathbf{x}_{t+1} + \beta(\mathbf{x}_{t+1} - \mathbf{x}_t) \\ &= (1+\beta)\mathbf{x}_{t+1} - \beta \mathbf{x}_t\end{aligned} \tag{2.0.1b}$$

For our linear regression problem, the updates rules take the form

$$\mathbf{x}_{t+1} = \mathbf{y}_t - \alpha \mathbf{\Sigma}(\mathbf{y}_t - \mathbf{x}^*) \tag{2.0.2a}$$

$$\mathbf{y}_{t+1} = (1+\beta)\mathbf{x}_{t+1} - \beta \mathbf{x}_t \tag{2.0.2b}$$

Or if we subtract $\mathbf{x}^*$ on both sides of both the update equations, we get

$$\begin{aligned}\mathbf{x}_{t+1} - \mathbf{x}^* &= \mathbf{y}_t - \mathbf{x}^* - \alpha \mathbf{\Sigma}(\mathbf{y}_t - \mathbf{x}^*) \\ &= (\mathbf{I} - \alpha \mathbf{\Sigma})(\mathbf{y}_t - \mathbf{x}^*)\end{aligned} \tag{2.0.2c}$$

$$\begin{aligned}\mathbf{y}_{t+1} - \mathbf{x}^* &= (1+\beta)(\mathbf{x}_{t+1} - \mathbf{x}^*) - \beta(\mathbf{x}_t - \mathbf{x}^*) \\ &= (1+\beta)(\mathbf{I} - \alpha \mathbf{\Sigma})(\mathbf{y}_t - \mathbf{x}^*) - \beta(\mathbf{x}_t - \mathbf{x}^*)\end{aligned} \tag{2.0.2d}$$

Just like as we did in the Gradient Descent analysis, we look at these update equations from the eigen-basis by performing a basis transformation. So we define $\mathbf{u}_t := \mathbf{Q}^T(\mathbf{x}_t - \mathbf{x}^*)$ and $\mathbf{v}_t := \mathbf{Q}^T(\mathbf{y}_t - \mathbf{x}^*)\ \forall\ t > 0$ and multiply (2.0.2c) and (2.0.2d) by Q^T from the left thus obtaining

$$\begin{aligned}\mathbf{Q}^T(\mathbf{x}_{t+1} - \mathbf{x}^*) &= \mathbf{Q}^T(\mathbf{y}_t - \mathbf{x}^*) - \alpha \mathbf{Q}^T \mathbf{\Sigma}(\mathbf{y}_t - \mathbf{x}^*) \\ \mathbf{u}_{t+1} &= \mathbf{v}_t - \alpha \mathbf{Q}^T \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T (\mathbf{y}_t - \mathbf{x}^*) \\ &= \mathbf{v}_t - \alpha \mathbf{\Lambda} \mathbf{v}_t \\ &= (\mathbf{I} - \alpha \mathbf{\Lambda})\mathbf{v}_t\end{aligned} \tag{2.0.3a}$$

$$\begin{aligned}\mathbf{Q}^T(\mathbf{y}_{t+1} - \mathbf{x}^*) &= (1+\beta)\mathbf{Q}^T(\mathbf{I} - \alpha \mathbf{\Sigma})(\mathbf{y}_t - \mathbf{x}^*) - \beta \mathbf{Q}^T(\mathbf{x}_t - \mathbf{x}^*) \\ \mathbf{v}_{t+1} &= (1+\beta)(\mathbf{I} - \alpha \mathbf{\Lambda})\mathbf{v}_t - \beta \mathbf{u}_t\end{aligned} \tag{2.0.3b}$$

Now that we have the transformed update equations, we can closely look at the dynamics of these vectors along each eigen-vector direction.

$$u_{t+1}^{(i)} = (1 - \alpha\lambda_i) v_t^{(i)} \qquad \forall\ i = 1, 2, \ldots, d \tag{2.0.4a}$$

$$v_{t+1}^{(i)} = (1+\beta)(1 - \alpha\lambda_i) v_t^{(i)} - \beta u_t^{(i)} \qquad \forall\ i = 1, 2, \ldots, d \tag{2.0.4b}$$

Observing closely, we can find that equations (2.0.4a) and (2.0.4b) form a 2-dimensional vector update. Precisely,

$$\begin{bmatrix} u_{t+1}^{(i)} \\ v_{t+1}^{(i)} \end{bmatrix} = \begin{bmatrix} 0 & (1 - \alpha\lambda_i) \\ -\beta & (1+\beta)(1 - \alpha\lambda_i) \end{bmatrix} \begin{bmatrix} u_t^{(i)} \\ v_t^{(i)} \end{bmatrix} \qquad \forall\ i = 1, 2, \ldots, d \tag{2.0.5}$$

For simplicity, lets define $\mathbf{p}_t^i := \begin{bmatrix} u_t^{(i)} \\ v_t^{(i)} \end{bmatrix}$, and $\mathbf{R}_i(\alpha, \beta) := \begin{bmatrix} 0 & (1 - \alpha\lambda_i) \\ -\beta & (1+\beta)(1 - \alpha\lambda_i) \end{bmatrix}$

The update equation can be re-written as

$$\begin{aligned}\mathbf{p}_{t+1}^i &= \mathbf{R}_i(\alpha, \beta)\mathbf{p}_t^i \\ &= \mathbf{R}_i^{t+1}(\alpha, \beta)\mathbf{p}_0^i\end{aligned} \tag{2.0.6}$$

Geometrically we can think of the update equation (2.0.6) as a linear map applied to consecutive error iterates $\mathbf{p}_t^i$. By analyzing the eigenvalues of the matrix $\mathbf{R}_i(\alpha, \beta)$ we can argue about the decaying nature of the error vector. For the choice of $\alpha = \frac{1}{\lambda_1}$ and $\beta = \frac{\sqrt{\lambda_1} - \sqrt{\lambda_d}}{\sqrt{\lambda_1} + \sqrt{\lambda_d}}$, we get that the eigenvalues of he matrix are of the form $r_i e^{\pm \iota \theta_i}$ where

$$r_i = \left(1 - \frac{\lambda_i}{\lambda_1}\right)\left(\frac{\sqrt{\lambda_1} - \sqrt{\lambda_d}}{\sqrt{\lambda_1} + \sqrt{\lambda_d}}\right) \tag{2.0.7a}$$

$$\theta_i = \tan^{-1}\left(\sqrt{\frac{\lambda_i - \lambda_d}{\lambda_1 - \lambda_i}}\right) \tag{2.0.7b}$$

Therefore the matrix $\mathbf{R}_i^{t+1}(\alpha, \beta)$ has eigenvalues of the form $r_i^{t+1} e^{\pm \iota (t+1)\theta_i}$. Therefore we clearly see that the error vector converges because the eigenvalues decay geometrically.

A geometric illustration of the decaying behavior of the eigenvalues of the matrix $\mathbf{R}_i^{t+1}(\alpha, \beta)$ can be seen in the figure below.

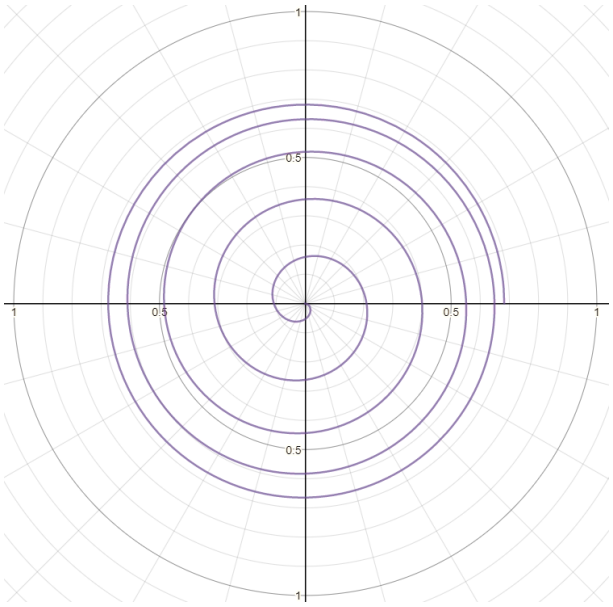

Figure 1: The curve of eigenvalues corresponding to $+\theta_i$, of the operator $\mathbf{R}_i^{20}(\alpha, \beta)$, on the complex plane parameterized by $t = \lambda_i \in [\lambda_d, \lambda_1]$.

From figure 1 and equation (2.0.7a), it is evident that error vectors which are in the direction of higher eigenvalues of $\mathbf{\Sigma}$ converge much faster compared to that in the direction of lower eigenvalues.

From (1.1.4) we have seen that the main reason why gradient descent has a slow convergence is because the decay of the component of the error vector in the direction of the eigenvector with the least corresponding eigenvalue is the slowest due to the small value of λ_d. Analyzing equation (2.0.6) for $i = d$, we have both the eigenvalues of the matrix $\mathbf{R}_d\left(\frac{1}{\lambda_1}, \beta\right)$ equal to $1 - \frac{\sqrt{\lambda_d}}{\sqrt{\lambda_1}}$.

Through these optimal values of α and β, we can see that the error iterates along the least eigenvalue direction converges to $\mathbf{0}$ via a linear operator with eigenvalues $1 - \frac{\sqrt{\lambda_d}}{\sqrt{\lambda_1}}$ instead of $1 - \frac{\lambda_d}{\lambda_1}$ that we saw for the gradient descent case.