Found Comput Math
DOI 10.1007/s10208-013-9150-3

FOUNDATIONS OF COMPUTATIONAL MATHEMATICS
The Journal of the Society for the Foundations of Computational Mathematics

# Adaptive Restart for Accelerated Gradient Schemes

**Brendan O'Donoghue · Emmanuel Candès**

Received: 2 January 2013 / Accepted: 25 February 2013


**Abstract** In this paper we introduce a simple heuristic adaptive restart technique that can dramatically improve the convergence rate of accelerated gradient schemes. The analysis of the technique relies on the observation that these schemes exhibit two modes of behavior depending on how much momentum is applied at each iteration. In what we refer to as the 'high momentum' regime the iterates generated by an accelerated gradient scheme exhibit a periodic behavior, where the period is proportional to the square root of the local condition number of the objective function. Separately, it is known that the optimal restart interval is proportional to this same quantity. This suggests a restart technique whereby we reset the momentum whenever we observe periodic behavior. We provide a heuristic analysis that suggests that in many cases adaptively restarting allows us to recover the optimal rate of convergence with no prior knowledge of function parameters.

**Keywords** Convex optimization · First order methods · Accelerated gradient schemes

**Mathematics Subject Classification** 80M50 · 90C06 · 90C25

Communicated by Felipe Cucker.

B. O'Donoghue (✉) · E. Candès
Stanford University, 450 Serra Mall, Stanford, CA 94305, USA
e-mail: bodonoghue85@gmail.com

E. Candès
e-mail: candes@stanford.edu

Published online: 03 July 2013

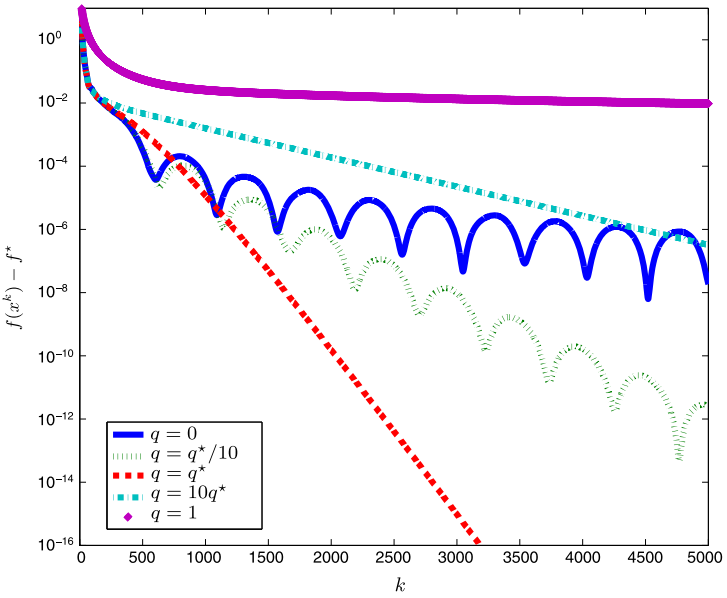


**Fig. 1** Convergence of Algorithm 1 with different estimates of $q$

## 1 Introduction

Accelerated gradient schemes were first proposed by Yurii Nesterov in 1983 [18]. He demonstrated a simple modification to gradient descent that could obtain provably optimal performance for the complexity class of first-order algorithms applied to minimize smooth convex functions. The method, and its successors, are often referred to as 'fast', 'accelerated', or 'optimal' methods. In recent years there has been a resurgence of interest in first-order optimization methods [1, 3, 14, 20, 24], driven primarily by the need to solve very large problem instances unsuited to second-order methods.

Accelerated gradient schemes can be thought of as *momentum* methods, in that the step taken at each iteration depends on the previous iterations, where the momentum grows from one iteration to the next. When we refer to *restarting* the algorithm we mean starting the algorithm again, taking the current iteration as the new starting point. This erases the memory of previous iterations and resets the momentum back to zero.

Unlike gradient descent, accelerated methods are not guaranteed to be monotone in the objective function value. A common observation when running an accelerated method is the appearance of ripples or bumps in the trace of the objective value. These are seemingly regular increases in the objective, see Fig. 1 for an example. In this paper we demonstrate that this behavior can occur when the momentum has exceeded a critical value (the optimal momentum value derived by Nesterov in [19]) and that the period of these ripples is proportional to the square-root of the (local) condition number of the function. Separately, we re-derive the previously known result that the optimal restart interval is also proportional to the square root of the condition number.

Combining these provides a justification for the use of a restart technique whereby we restart whenever we observe this rippling behavior. The analysis also suggests that if the function is locally well-conditioned we may be able to use restarting to obtain a linear convergence rate inside the well-conditioned region.

*Smooth Unconstrained Optimization* We wish to minimize a smooth convex function [4] of a variable $x \in \mathbf{R}^n$, i.e.,

$$\text{minimize} \quad f(x) \tag{1}$$

where $f : \mathbf{R}^n \to \mathbf{R}$ has a Lipschitz continuous gradient with constant $L$, i.e.,

$$\left\| \nabla f(x) - \nabla f(y) \right\|_2 \leq L \|x - y\|_2, \quad \forall x, y \in \mathbf{R}^n.$$

We shall denote by $f^\star$ the optimal value of the above optimization problem, if a minimizer exists then we shall write it as $x^\star$. Further, a continuously differentiable function $f$ is said to be strongly convex with strong convexity parameter $\mu > 0$ if

$$f(x) \geq f(y) + \nabla f(y)^{\mathrm{T}}(x - y) + (\mu/2)\|x - y\|_2^2, \quad \forall x, y \in \mathbf{R}^n.$$

The *condition number* of a smooth, strongly convex function is given by $L/\mu$.

## 2 Accelerated Methods

Accelerated first-order methods to solve (1) were first developed by Nesterov [18], this scheme is from [19]:

**Algorithm 1** Accelerated scheme I

**Require:** $x^0 \in \mathbf{R}^n$, $y^0 = x^0$, $\theta_0 = 1$ and $q \in [0, 1]$

1: **for** $k = 0, 1, \ldots$ **do**
2: $\quad x^{k+1} = y^k - t_k \nabla f(y^k)$
3: $\quad \theta_{k+1}$ solves $\theta_{k+1}^2 = (1 - \theta_{k+1})\theta_k^2 + q\theta_{k+1}$
4: $\quad \beta_{k+1} = \theta_k(1 - \theta_k)/(\theta_k^2 + \theta_{k+1})$
5: $\quad y^{k+1} = x^{k+1} + \beta_{k+1}(x^{k+1} - x^k)$
6: **end for**

There are many variants of the above scheme, see, e.g., [1, 2, 14, 20, 24]. Note that by setting $q = 1$ in the above scheme we recover gradient descent. For a smooth convex function the above scheme converges for any $t_k \leq 1/L$; setting $t_k = 1/L$ and $q = 0$ obtains a guaranteed convergence rate of

$$f\left(x^k\right) - f^\star \leq \frac{4L\|x^0 - x^\star\|^2}{(k+2)^2}, \tag{2}$$

assuming a minimizer exists. If the function is also strongly convex with strong convexity parameter $\mu$, then a choice of $q = \mu/L$ (the reciprocal of the condition number) will achieve

$$f(x^k) - f^\star \leq L\left(1 - \sqrt{\frac{\mu}{L}}\right)^k \|x^0 - x^\star\|^2. \tag{3}$$

This is often referred to as *linear convergence*. With this convergence rate we can achieve an accuracy of $\epsilon$ in

$$\mathcal{O}\left(\sqrt{\frac{L}{\mu}} \log \frac{1}{\epsilon}\right) \tag{4}$$

iterations.

In the case of a strongly convex function the following simpler scheme obtains the same guaranteed rate of convergence [19]:

**Algorithm 2** Accelerated scheme II

**Require:** $x^0 \in \mathbf{R}^n$, $y^0 = x^0$

1: **for** $k = 0, 1, \ldots$ **do**
2: $\quad x^{k+1} = y^k - (1/L)\nabla f(y^k)$
3: $\quad y^{k+1} = x^{k+1} + \beta^\star(x^{k+1} - x^k)$
4: **end for**

where we set

$$\beta^\star = \frac{1 - \sqrt{\mu/L}}{1 + \sqrt{\mu/L}}. \tag{5}$$

Note that in Algorithm 1, using the optimal choice $q = \mu/L$, we have $\beta_k \uparrow \beta^\star$. If $\beta_k$ is interpreted as a momentum term then $\beta^\star$ is the maximum amount of momentum we should apply; when we have a value of $\beta$ higher than $\beta^\star$ we are in the 'high momentum' regime. We shall return to this point later.

The convergence rates of Algorithms 1 and 2 are optimal in the sense of the lower complexity bounds derived by Nemirovski and Yudin in [17]. However, this convergence is only guaranteed when the function parameters $\mu$ and $L$ are known in advance.

### 2.1 Robustness

A natural question to ask is how robust are accelerated methods to errors in the estimates of the Lipschitz constant $L$ and strong convexity parameter $\mu$? For the case of an unknown Lipschitz constant we can estimate the optimal step-size by the use of backtracking; see, e.g., [2, 3, 24]. Estimating the strong convexity parameter is much more challenging.

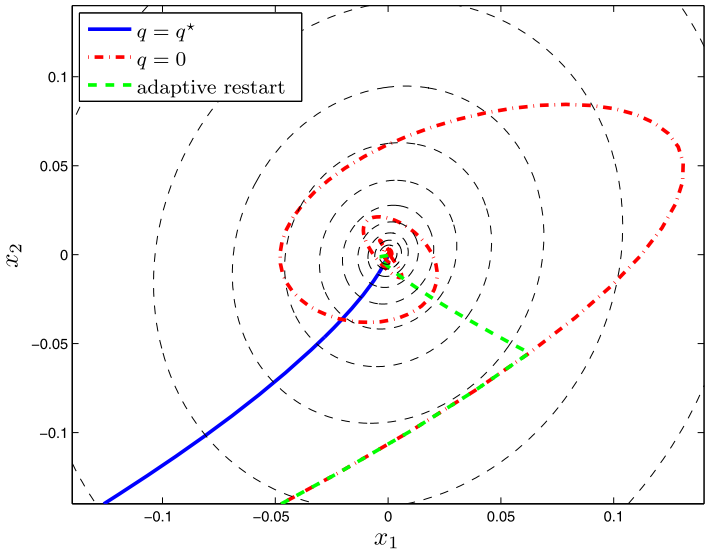


**Fig. 2** Sequence trajectories under Algorithm 1 and with adaptive restart

*Estimating the Strong Convexity Parameter* In [20] Nesterov demonstrated a method to bound $\mu$, similar to the backtracking schemes to estimate $L$. His scheme achieves a convergence rate quite a bit slower than Algorithm 1 when $\mu$ is known. In practice, we often assume (or guess) that $\mu$ is zero, which corresponds to setting $q = 0$ in Algorithm 1. Indeed many discussions of accelerated algorithms do not even include a $q$ term, e.g., the original algorithm in [18]. However, this choice can dramatically slow down the convergence of the iterates. Figure 1 shows Algorithm 1 applied to minimize a positive definite quadratic function in $n = 200$ dimensions, with the optimal choice of $q$ being $q^\star = \mu/L = 4.1 \times 10^{-5}$ (a condition number of about $2.4 \times 10^4$), and step-size $t = 1/L$. Each trace is the progress of the algorithm with a different choice of $q$ (hence a different estimate of $\mu$).

We observe that slightly over or underestimating the optimal value of $q$ for the function can have a severe detrimental effect on the rate of convergence of the algorithm. We also note the clear difference in behavior between the cases where we underestimate and where we overestimate $q^\star$; in the latter we observe monotonic convergence but in the former we notice the appearance of regular ripples or bumps in the traces.

*Interpretation* The optimal momentum depends on the condition number of the function; specifically, higher momentum is required when the function has a higher condition number. Underestimating the amount of momentum required leads to slower convergence. However, we are more often in the other regime, that of overestimated momentum, because generally $q = 0$, in which case $\beta_k \uparrow 1$; this corresponds to high momentum and rippling behavior, as we see in Fig. 1. This can be visually understood in Fig. 2, which shows the trajectories of sequences generated by Algorithm 1 minimizing a positive definite quadratic in two dimensions, under $q = q^\star$, the

optimal choice of $q$, and $q = 0$. The high momentum causes the trajectory to overshoot the minimum and oscillate around it. This causes a rippling in the objective function values along the trajectory. In the sequel we shall demonstrate that the period of these ripples is proportional to the square root of the (local) condition number of the function.

Lastly we mention that the condition number is a *global* parameter; the sequence generated by an accelerated scheme may enter regions that are locally better conditioned, say, near the optimum. In these cases the choice of $q = q^\star$ is appropriate outside of this region, but once we enter it we expect the rippling behavior associated with high momentum to emerge, despite the optimal choice of $q$.

## 3 Restarting

### 3.1 Fixed Restart

For strongly convex functions an alternative to choosing the optimal value of $q$ in Algorithm 1 is to use restarting, [3, 11, 15, 16, 20]. One example of a fixed restart scheme is as follows:

**Algorithm 3** Fixed restarting

**Require:** $x^0 \in \mathbf{R}^n$, $y^0 = x^0$, $\theta_0 = 1$

1: **for** $j = 0, 1, \ldots$ **do**
2: carry out Algorithm 1 with $q = 0$ for $k$ steps
3: set $x^0 = x^k$, $y^0 = x^k$ and $\theta_0 = 1$.
4: **end for**

We restart the algorithm every $k$ iterations, taking as our starting point the last point produced by the algorithm, and reset the momentum back to zero.

*Optimal Fixed Restart Interval* Fixed restart intervals have been examined and upper bounds on the optimal restart interval have been derived by several authors; see, e.g., [16, §11.4], [11, 13, 20]. We re-derive an upper bound here.

If we restart every $k$ iterations we have, at outer iteration $j$, inner loop iteration $k$ (just before a restart),

$$\begin{aligned} f\big(x^{(j+1,0)}\big) - f^\star = f\big(x^{(j,k)}\big) - f^\star &\leq 4L\big\|x^{(j,0)} - x^\star\big\|/k^2 \\ &\leq \big(8L/\mu k^2\big)\big(f\big(x^{(j,0)}\big) - f^\star\big), \end{aligned}$$

where the first inequality is the convergence guarantee of Algorithm 1, and the second comes from the strong convexity of $f$. So after $jk$ steps we have

$$f\big(x^{(j,0)}\big) - f^\star \leq \big(8L/\mu k^2\big)^j \big(f\big(x^{(0,0)}\big) - f^\star\big).$$

We wish to minimize this quantity over $j$ and $k$ jointly, subject to having $jk = c$ total iterations. A simple calculation yields

$$k^\star = e\sqrt{8L/\mu}. \tag{6}$$

Using this as our restart interval we obtain an accuracy of $\epsilon$ in less than $\mathcal{O}(\sqrt{L/\mu}\log(1/\epsilon))$ iterations, i.e., the optimal linear convergence rate as in (4).

The drawbacks in using fixed restarts are two-fold, firstly it depends on unknown parameters $L$ and $\mu$, and secondly it is a conservative estimate based on global parameters and may be inappropriate in better conditioned regions.

### 3.2 Adaptive Restart

The above analysis suggests that an *adaptive* restart technique may be useful when using Algorithm 1. In particular we want a scheme that makes some computationally cheap observation and decides whether or not to restart based on that observation. In this paper we suggest two schemes that perform well in practice and provide a heuristic analysis that suggests improved convergence when these schemes are used.

- **Function scheme:** restart whenever

$$f\left(x^k\right) > f\left(x^{k-1}\right).$$

- **Gradient scheme:** restart whenever

$$\nabla f\left(y^{k-1}\right)^{\mathrm{T}}\left(x^k - x^{k-1}\right) > 0.$$

Empirically we observe that these two schemes perform similarly well. The gradient scheme has two advantages over the function scheme. Firstly all quantities involved in the gradient scheme are already calculated in accelerated schemes, so no extra computation is required. Secondly near to the optimum the gradient scheme may be more numerically stable, since $\nabla f(y^{k-1})^{\mathrm{T}} x^k$ will tend to zero as we get close to the optimum, whereas $f(x^k)$ will tend to $f^\star$, leading to possible cancellation errors when evaluating $f(x^k) - f(x^{k-1})$.

We can give rough justifications for each scheme. The function scheme restarts at the bottom of the troughs as in Fig. 1, thereby avoiding the wasted iterations where we are moving away from the optimum. The gradient scheme restarts whenever the momentum term and the negative gradient are making an obtuse angle. In other words we restart when the momentum seems to be taking us in a bad direction, as measured by the negative gradient at that point.

Figure 3 shows the effect of different restart intervals on minimizing a positive definite quadratic function in $n = 500$ dimensions. In this particular case the upper bound on the optimal restart interval is every 700 iterations. We note that when this interval is used the convergence is better than when no restart is used, however, not as good as using the optimal choice of $q$. We also note that restarting every 400 iterations performs about as well as restarting every 700 iterations, suggesting that the optimal restart interval is somewhat lower than 700. We have also plotted the performance of the two adaptive restart schemes. The performance is on the same order as the

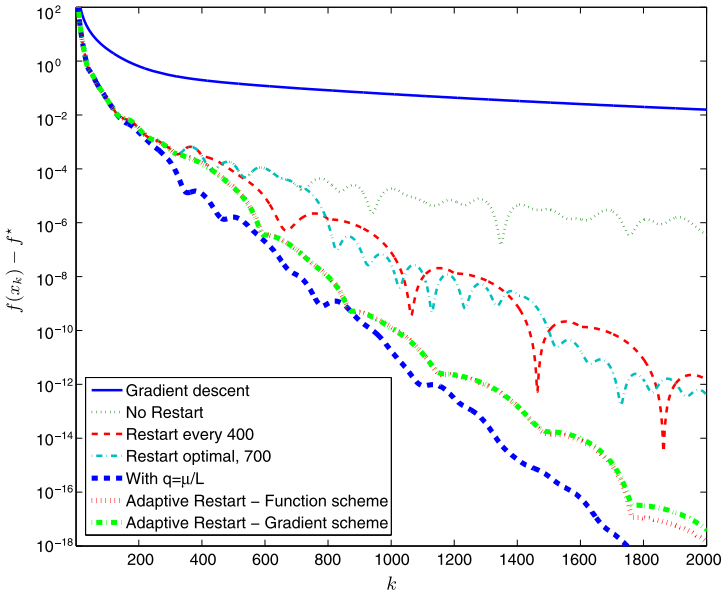


**Fig. 3** Comparison of fixed and adaptive restart intervals

algorithm with the optimal $q$ and much better than using the fixed restart interval. Figure 2 demonstrates the function restart scheme trajectories for a two dimensional example, restarting resets the momentum and prevents the characteristic spiraling behavior.

It should be noted that the conjugate gradient method [12, 21] outperforms fast gradient schemes when minimizing a quadratic, both in theory and practice. See [21, eq. 5.36] and compare with the convergence rate in (3). We use quadratics here simply to illustrate the technique.

## 4 Analysis

In this section we consider applying an accelerated scheme to minimizing a positive definite quadratic function. We shall see that once the momentum is larger than a critical value we observe periodicity in the iterates. We use this periodicity to recover linear convergence when using adaptive restarting. The analysis presented in this section is similar in spirit to the analysis of the heavy ball method in [22, §3.2].

### 4.1 Minimizing a Quadratic

Consider minimizing a strongly convex quadratic. Without loss of generality we can assume that $f$ has the following form:

$$f(x) = (1/2)x^{\mathrm{T}}Ax$$

where $A \in \mathbf{R}^{n\times n}$ is positive definite and symmetric. In this case $x^\star = 0$ and $f^\star = 0$. We have strong convexity parameter $\mu = \lambda_{\min} > 0$ and $L = \lambda_{\max}$, where $\lambda_{\min}$ and $\lambda_{\max}$ are the minimum and maximum eigenvalues of $A$, respectively.

### 4.2 The Algorithm as a Linear Dynamical System

We apply an accelerated scheme to minimize $f$ with a fixed step-size $t = 1/L$. Given quantities $x^0$ and $y^0 = x^0$, Algorithm 1 is carried out as follows:

$$\begin{aligned} x^{k+1} &= y^k - (1/L)Ay^k, \\ y^{k+1} &= x^{k+1} + \beta_k\big(x^{k+1} - x^k\big). \end{aligned}$$

For the rest of the analysis we shall take $\beta_k$ to be constant and equal to some $\beta$ for all $k$. By making this approximation we can show that there are two regimes of behavior for the system, depending on the value of $\beta$. Consider the eigenvector decomposition of $A = V\Lambda V^{\mathrm{T}}$. Denote by $w^k = V^{\mathrm{T}}x^k$, $v^k = V^{\mathrm{T}}y^k$. In this basis the update equations can be written

$$\begin{aligned} w^{k+1} &= v^k - (1/L)\Lambda v^k, \\ v^{k+1} &= w^{k+1} + \beta\big(w^{k+1} - w^k\big). \end{aligned}$$

These are $n$ independently evolving dynamical systems. The $i$th system evolves according to

$$\begin{aligned} w_i^{k+1} &= v_i^k - (\lambda_i/L)v_i^k, \\ v_i^{k+1} &= w_i^{k+1} + \beta\big(w_i^{k+1} - w_i^k\big), \end{aligned}$$

where $\lambda_i$ is the $i$th eigenvalue of $A$. Eliminating the sequence $v_i^{(k)}$ from the above we obtain the following recurrence relation for the evolution of $w_i$:

$$w_i^{k+2} = (1+\beta)(1-\lambda_i/L)w_i^{k+1} - \beta(1-\lambda_i/L)w_i^k, \quad k = 0, 1, \ldots,$$

where $w_i^0$ is known and $w_i^1 = w_i^0(1-\lambda_i/L)$, i.e., a gradient step from $w_i^0$.

The update equation for $v_i$ is identical, differing only in the initial conditions,

$$v_i^{k+2} = (1+\beta)(1-\lambda_i/L)v_i^{k+1} - \beta(1-\lambda_i/L)v_i^k, \quad k = 0, 1, \ldots,$$

where $v_i^0 = w_i^0$ and $v_i^1 = ((1+\beta)(1-\lambda_i/L) - \beta)v_i^0$.

### 4.3 Convergence Properties

The behavior of this system is determined by the characteristic polynomial of the recurrence relation,

$$r^2 - (1+\beta)(1-\lambda_i/L)r + \beta(1-\lambda_i/L). \tag{7}$$

Let $\beta_i^\star$ be the critical value of $\beta$ for which this polynomial has repeated roots, i.e.,

$$\beta_i^\star := \frac{1-\sqrt{\lambda_i/L}}{1+\sqrt{\lambda_i/L}}.$$

If $\beta \leq \beta_i^\star$ then the polynomial (7) has two real roots, $r_1$ and $r_2$, and the system evolves according to [8]

$$w_i^k = c_1 r_1^k + c_2 r_2^k. \tag{8}$$

When $\beta = \beta_i^\star$ the roots coincide at the point $r^\star = (1+\beta)(1-\lambda_i/L)/2 = (1-\sqrt{\lambda_i/L})$; this corresponds to critical damping. We have the fastest monotone convergence at rate $\propto (1-\sqrt{\lambda_i/L})^k$. Note that if $\lambda_i = \mu$ then $\beta_i^\star$ is the optimal choice of $\beta$ as given by (5) and the convergence rate is the optimal rate, as given by (3). This occurs as typically the mode corresponding to the smallest eigenvalue dominates the convergence of the entire system.

If $\beta < \beta_i^\star$ we are in the low momentum regime, and we say the system is over-damped. The convergence rate is dominated by the larger root, which is greater than $r^\star$, i.e., the system exhibits slow monotone convergence.

If $\beta > \beta_i^\star$ then the roots of the polynomial (7) are complex and we are in the high momentum regime. The system is under-damped and exhibits periodicity. In that case the characteristic solution is given by [8]

$$w_i^k = c_i\big(\beta(1-\lambda_i/L)\big)^{k/2}\big(\cos(k\psi_i - \delta_i)\big)$$

where

$$\psi_i = \cos^{-1}\big((1-\lambda_i/L)(1+\beta)/2\sqrt{\beta(1-\lambda_i/L)}\big)$$

and $\delta_i$ and $c_i$ are constants that depend on the initial conditions; in particular for $\beta \approx 1$ we have $\delta_i \approx 0$ and we will ignore it. Similarly,

$$v_i^k = \hat{c}_i\big(\beta(1-\lambda_i/L)\big)^{k/2}\big(\cos(k\psi_i - \hat{\delta}_i)\big)$$

where $\hat{\delta}_i$ and $\hat{c}_i$ are constants, and again $\hat{\delta}_i \approx 0$. For small $\theta$ we know that $\cos^{-1}(\sqrt{1-\theta}) \approx \sqrt{\theta}$, and therefore if $\lambda_i \ll L$, then

$$\psi_i \approx \sqrt{\lambda_i/L}.$$

In particular the frequency of oscillation for the mode corresponding to the smallest eigenvalue $\mu$ is approximately given by $\psi_\mu \approx \sqrt{\mu/L}$.

To summarize, based on the value of $\beta$ we observe the following behavior:

- $\beta > \beta_i^\star$: high momentum, under-damped.
- $\beta < \beta_i^\star$: low momentum, over-damped.
- $\beta = \beta_i^\star$: optimal momentum, critically damped.

### 4.4 Observable Quantities

We do not observe the evolution of the modes, but we can observe the evolution of the function value; which is given by

$$f\left(x^k\right) = \sum_{i=1}^{n} \left(w_i^k\right)^2 \lambda_i$$

and if $\beta > \beta^\star = (1 - \sqrt{\mu/L})/(1 + \sqrt{\mu/L})$ we are in the high momentum regime for all modes and thus

$$f\left(x^k\right) = \sum_{i=1}^{n} \left(w_i^k\right)^2 \lambda_i \approx \sum_{i=1}^{n} \left(w_i^0\right)^2 \lambda_i \beta^k (1 - \lambda_i/L)^k \cos^2(k\psi_i).$$

The function value will quickly be dominated by the smallest eigenvalue and we have

$$f\left(w^k\right) \approx \left(w_\mu^0\right)^2 \mu \beta^k (1 - \mu/L)^k \cos^2\left(k\sqrt{\mu/L}\right), \tag{9}$$

where we have replaced $\psi_\mu$ with $\sqrt{\mu/L}$, and we are using the subscript $\mu$ to denote those quantities corresponding to the smallest eigenvalue.

A similar analysis for the gradient restart scheme yields

$$\nabla f\left(y^k\right)^{\mathrm{T}}\left(x^{k+1} - x^k\right) \approx \mu v_\mu^k \left(w_\mu^{k+1} - w_\mu^k\right) \propto \beta^k (1 - \mu/L)^k \sin\left(2k\sqrt{\mu/L}\right). \tag{10}$$

In other words observing the quantities in (9) or (10) we expect to see oscillations at a frequency proportional to $\sqrt{\mu/L}$, i.e., the frequency of oscillation is telling us something about the condition number of the function.

### 4.5 Convergence with Adaptive Restart

Applying Algorithm 1 with $q = 0$ to minimize a quadratic starts with $\beta_0 = 0$, i.e., the system starts in the low momentum, monotonic regime. Eventually $\beta_k$ becomes larger than $\beta^\star$ and we enter the high momentum, oscillatory regime. It takes about $(3/2)\sqrt{L/\mu}$ iterations for $\beta_k$ to exceed $\beta^\star$. After that the system is under-damped and the iterates obey (9) and (10). Under either adaptive restart scheme, (9) and (10) indicate that we shall observe the restart condition after a further $(\pi/2)\sqrt{L/\mu}$ iterations. We restart and the process begins again, with $\beta_k$ set back to zero. Thus under either scheme we restart approximately every

$$k^\star = \frac{\pi + 3}{2}\sqrt{\frac{L}{\mu}}$$

iterations (cf. the upper bound on optimal fixed restart interval (6)). Following a similar calculation to Sect. 3.1, this restart interval guarantees us an accuracy of $\epsilon$ within $\mathcal{O}(\sqrt{L/\mu}\log(1/\epsilon))$ iterations, i.e., we have recovered the optimal linear convergence rate of (4) via adaptive restarting, with no prior knowledge of $\mu$.

### 4.6 Extension to Smooth Convex Minimization

If the function we are minimizing has a positive definite Hessian at the optimum, then by Taylor's theorem there is a region inside of which

$$f(x) \approx f\left(x^\star\right) + \left(x - x^\star\right)^{\mathrm{T}} \nabla^2 f\left(x^\star\right)\left(x - x^\star\right),$$

and loosely speaking we are minimizing a quadratic. Once we are inside this region we will observe behavior consistent with the analysis above, and we can exploit this behavior to achieve fast convergence by using restarts. Note that the Hessian at the optimum may have smallest eigenvalue $\lambda_{\min} > \mu$, the global strong convexity parameter, and we may be able to achieve faster local convergence than (3) would suggest. This result is similar in spirit to the restart method applied to the non-linear conjugate gradient method, where it is desirable to restart the algorithm once it reaches a region in which the function is well approximated by a quadratic [21, §5.2].

The effect of these restart schemes outside of the quadratic region is unclear. In practice we observe that restarting based on one of the criteria described above is almost always helpful, even far away from the optimum. However, we have observed cases where restarting far from the optimum can slow down the early convergence slightly, until the quadratic region is reached and the algorithm enters the rapid linear convergence phase.

## 5 Numerical Examples

In this section we describe three further numerical examples that demonstrate the improvement of accelerated algorithms under an adaptive restarting technique.

### 5.1 Log-Sum-Exp

Here we minimize a smooth convex function that is *not* strongly convex. Consider the following optimization problem:

$$\text{minimize} \quad \rho \log\left(\sum_{i=1}^{m} \exp\left(\left(a_i^{\mathrm{T}} x - b_i\right)/\rho\right)\right)$$

where $x \in \mathbf{R}^n$. The objective function is smooth, but not strongly convex, it grows linearly asymptotically. Thus, the optimal value of $q$ in Algorithm 1 is zero. The quantity $\rho$ controls the smoothness of the function, as $\rho \to 0$, $f(x) \to \max_{i=1,\ldots,m}(a_i^{\mathrm{T}} x - b_i)$. As it is smooth, we expect the region around the optimum to be well approximated by a quadratic (assuming the optimum exists), and thus we expect to eventually enter a region where our restart method will obtain linear convergence without any knowledge of where this region is, the size of the region or the local function parameters within this region. For smaller values of $\rho$ the smoothness of the objective function decreases and thus we expect to take more iterations before we enter the region of linear convergence.

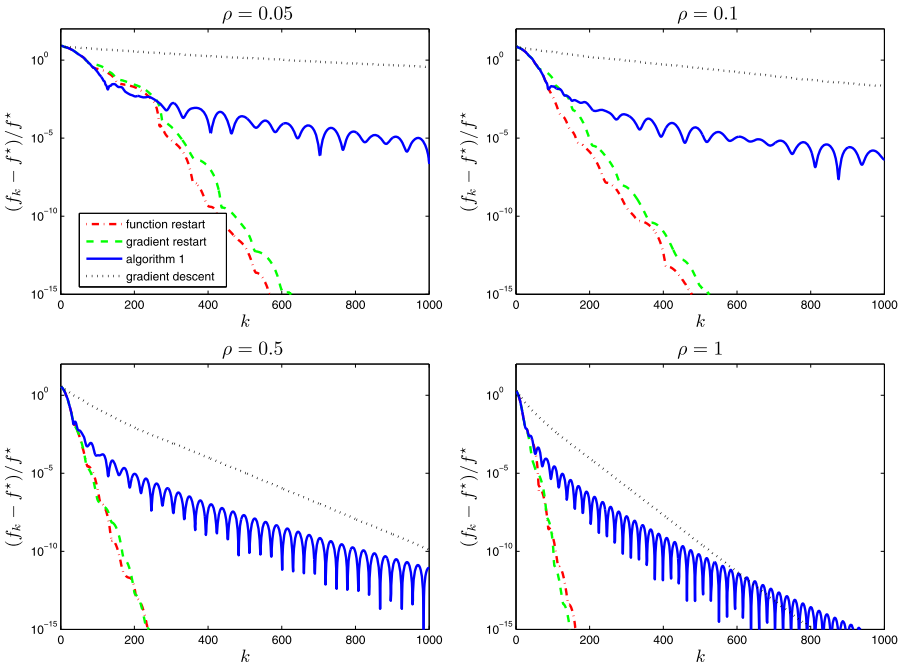


**Fig. 4** Minimizing a smooth but not strongly convex function

As a particular example we took $n = 20$ and $m = 100$; we generated the $a_i$ and $b_i$ randomly. Figure 4 demonstrates the performance of four different schemes for four different values of $\rho$. We selected the step-size for each case using the backtracking scheme described in [3, §5.3]. We note that both restart schemes perform well, eventually beating both gradient descent and the accelerated scheme. Both the function and the gradient schemes eventually enter a region of fast linear convergence. For large $\rho$ we see that even gradient descent performs well: similar to the adaptive restart scheme, it is able to automatically exploit the local strong convexity of the quadratic region around the optimum, see [19, §1.2.3]. Notice also the appearance of the periodic behavior in the trace of Algorithm 1.

## 5.2 Sparse Linear Regression

Consider the following optimization problem:

$$\text{minimize} \quad (1/2)\|Ax - b\|_2^2 + \rho\|x\|_1, \tag{11}$$

over $x \in \mathbf{R}^n$, where $A \in \mathbf{R}^{m\times n}$ and typically $n \gg m$. This is a widely studied problem in the field of compressed sensing, see e.g., [5, 6, 10, 23]. Loosely speaking problem (11) seeks a sparse vector with a small measurement error. The quantity $\rho$ trades off these two competing objectives. The iterative soft-threshold algorithm (ISTA) can be used to solve (11) [7, 9]. ISTA relies on the soft-thresholding operator:

$$\mathcal{T}_\alpha(x) = \text{sign}(x)\max\big(|x| - \alpha, 0\big),$$

where all the operations are applied element-wise. The ISTA algorithm, with constant step-size $t$, is given by

**Algorithm 4** ISTA

**Require:** $x^{(0)} \in \mathbf{R}^n$

1: **for** $k = 0, 1, \ldots$ **do**
2: $\quad x^{k+1} = \mathcal{T}_{\rho t}(x^k - tA^{\mathrm{T}}(Ax^k - b))$.
3: **end for**

The convergence rate of ISTA is guaranteed to be at least $\mathcal{O}(1/k)$, making it analogous to gradient descent. The fast iterative soft thresholding algorithm (FISTA) was developed in [2]; a similar algorithm was also developed by Nesterov in [20]. FISTA essentially applies acceleration to the ISTA algorithm; it is carried out as follows:

**Algorithm 5** FISTA

**Require:** $x^{(0)} \in \mathbf{R}^n$, $y^0 = x^0$ and $\theta_0 = 1$

1: **for** $k = 0, 1, \ldots$ **do**
2: $\quad x^{k+1} = \mathcal{T}_{\rho t}(y^k - tA^{\mathrm{T}}(Ay^k - b))$
3: $\quad \theta_{k+1} = (1 + \sqrt{1 + 4\theta_k^2})/2$
4: $\quad \beta_{k+1} = (\theta_k - 1)/\theta_{k+1}$
5: $\quad y^{k+1} = x^{k+1} + \beta_{k+1}(x^{k+1} - x^k)$.
6: **end for**

For any choice of $t \leq 1/\lambda_{\max}(A^{\mathrm{T}}A)$ FISTA obtains a convergence rate of at least $O(1/k^2)$. The objective in problem (11) is non-smooth, so it does not fit the class of problems we are considering in this paper. However, we are seeking a sparse solution vector $x^\star$, and we note that once the non-zero basis of the solution has been identified we are essentially minimizing a quadratic. Thus we expect that after a certain number of iterations adaptive restarting may provide linear convergence.

In this setting the function restart scheme can be applied unchanged, and it does not require an extra application of the matrix $A$, which is the costly operation in the algorithm. However, in performing FISTA we do not evaluate a gradient so we use the *composite gradient mapping* [20] for the gradient restart scheme, in which we take

$$x^{k+1} = \mathcal{T}_{\lambda t}\big(y^k - tA^{\mathrm{T}}\big(Ay^k - b\big)\big) := y^k - tG\big(y^k\big)$$

to be a generalized gradient step, where $G(y^k)$ is a generalized gradient at $y^k$. In this case the gradient restart scheme amounts to restarting whenever

$$G\big(y^k\big)^{\mathrm{T}}\big(x^{k+1} - x^k\big) > 0, \tag{12}$$

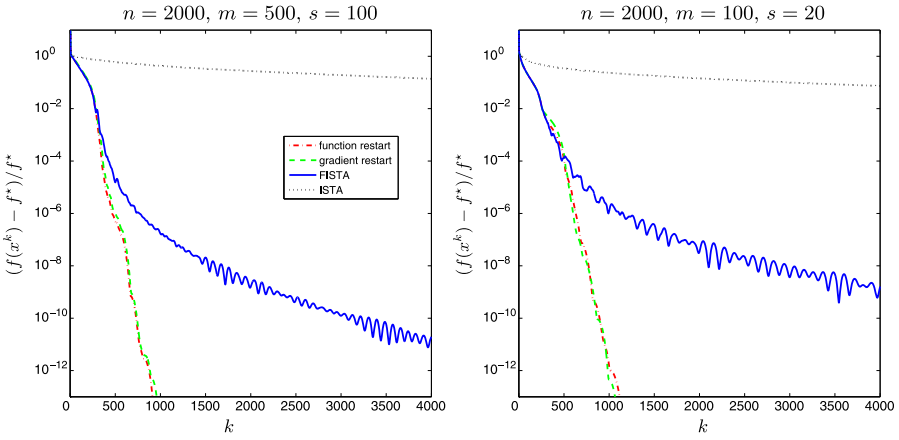


**Fig. 5** Adaptive restarting applied to the FISTA algorithm

or equivalently

$$\left(y^k - x^{k+1}\right)^{\mathrm{T}}\left(x^{k+1} - x^k\right) > 0. \tag{13}$$

We generated data for the numerical instances as follows. Firstly the entries of $A$ were sampled from a standard normal distribution. We then randomly generated a sparse vector $y$ with $n$ entries, only $s$ of which were non-zero. We then set $b = Ay + w$, where the entries in $w$ were IID sampled from $\mathcal{N}(0, 0.1)$. This ensured that the solution vector $x^\star$ is approximately $s$-sparse. We chose $\rho = 1$ and the step-size $t = 1/\lambda_{\max}(A^{\mathrm{T}}A)$. Figure 5 shows the dramatic speedup that adaptive restarting can provide, for two different examples.

### 5.3 Quadratic Programming

Consider the following quadratic program:

$$\begin{array}{ll} \text{minimize} & (1/2)x^{\mathrm{T}}Qx + q^{\mathrm{T}}x \\ \text{subject to} & a \le x \le b, \end{array} \tag{14}$$

over $x \in \mathbf{R}^n$, where $Q \in \mathbf{R}^{n\times n}$ is positive definite and $a, b \in \mathbf{R}^n$ are fixed vectors. The constraint inequalities are to be interpreted element-wise, and we assume that $a < b$. We denote by $\Pi_{\mathcal{C}}(z)$ the projection of a point $z$ onto the constraint set, which amounts to thresholding the entries in $z$.

Projected gradient descent [19] can solve (14); it is carried out as follows:

$$x^{k+1} = \Pi_{\mathcal{C}}\left(x^k - t\left(Qx^k + q\right)\right).$$

Projected gradient descent obtains a guaranteed convergence rate of $\mathcal{O}(1/k)$. Acceleration has been successfully applied to the projected gradient method, [2, 20]:

**Algorithm 6** Accelerated projected gradient

**Require:** $x^0 \in \mathbf{R}^n$, $y^0 = x^0$ and $\theta_0 = 1$

1: **for** $k = 0, 1, \ldots$ **do**
2: $\quad x^{k+1} = \Pi_{\mathcal{C}}(y^k - t(Qy^k + q))$
3: $\quad \theta_{k+1}$ solves $\theta_{k+1}^2 = (1 - \theta_{k+1})\theta_k^2$
4: $\quad \beta_{k+1} = \theta_k(1 - \theta_k)/(\theta_k^2 + \theta_{k+1})$
5: $\quad y^{k+1} = x^{k+1} + \beta_{k+1}(x^{k+1} - x^k)$
6: **end for**

For any choice of $t \leq 1/\lambda_{\max}(Q)$ accelerated projected gradient schemes obtain a convergence rate of at least $\mathcal{O}(1/k^2)$.

The presence of constraints make this a non-smooth optimization problem, however, once the constraints that are active have been identified the problem reduces to minimizing a quadratic on a subset of the variables, and we expect adaptive restarting to increase the rate of convergence. As in the sparse regression example of Sect. 5.2 the function restart remains unchanged. For the gradient scheme we use the *gradient mapping* [19, 2.2.3] as a generalized gradient, in which we take

$$x^{k+1} = \Pi_{\mathcal{C}}\big(y^k - t\big(Qy^k + q\big)\big) = y^k - tG\big(y^k\big)$$

to be a generalized gradient step and $G(y^k)$ to be a generalized gradient at $y^k$. This amounts to restarting based on condition (12) or, equivalently, (13).

For a numerical instance, we took $n = 500$ and generated $Q$ and $q$ randomly; $Q$ had a condition number of $10^7$. We took $b$ to be the vector of all ones, and $a$ to be that of all negative ones. The step-size was set to $t = 1/\lambda_{\max}(Q)$. The solution to this problem had 70 active constraints. Figure 6 shows the performance of projected gradient descent, accelerated projected gradient descent, and the two restart techniques.

## 6 Summary

In this paper we introduced a simple heuristic adaptive restart technique that can improve the convergence performance of accelerated gradient schemes for smooth convex optimization. We restart the algorithm whenever we observe a certain condition on the objective function value or gradient. We provided a heuristic analysis to show that we can recover the optimal linear rate of convergence in many cases, and near the optimum of a smooth function we can potentially dramatically accelerate the rate of convergence, even if the function is not globally strongly convex. We demonstrated the performance of the scheme on some numerical examples.

**Acknowledgements** We are very grateful to Stephen Boyd for his help and encouragement. We would also like to thank Stephen Wright for his advice and feedback, and Stephen Becker and Michael Grant for useful discussions. E. C. would like to thank the ONR (grant N00014-09-1-0258) and the Broadcom Foundation for their support. We must also thank two anonymous reviewers for their constructive feedback.

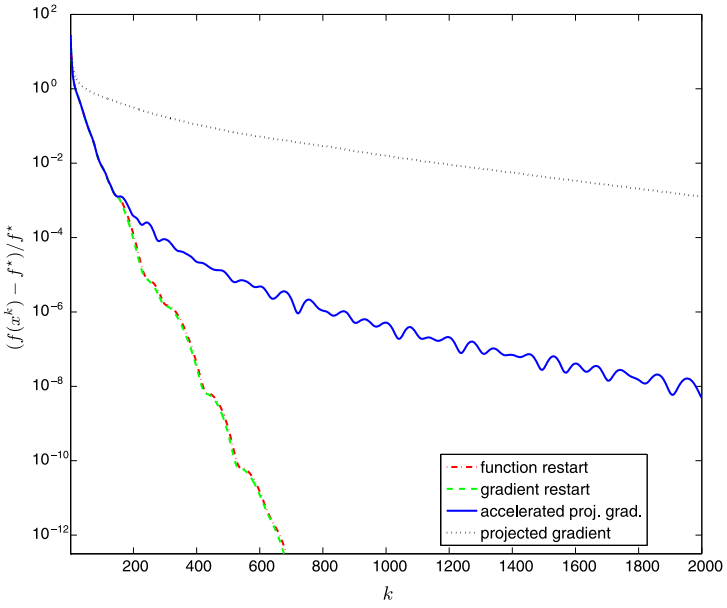


**Fig. 6** Adaptive restarting applied to the accelerated projected gradient algorithm

## References


1. A. Auslender, M. Teboulle, Interior gradient and proximal methods for convex and conic optimization, *SIAM J. Optim.* **16**(3), 697–725 (2006).
2. A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, *SIAM J. Imaging Sci.* **2**, 183–202 (2009).
3. S. Becker, E. Candès, M. Grant, Templates for convex cone problems with applications to sparse signal recovery, *Math. Program. Comput.* **3**(3), 165–218 (2011).
4. S. Boyd, L. Vandenberghe, *Convex Optimization* (Cambridge University Press, Cambridge, 2004).
5. E. Candès, J. Romberg, T. Tao, Stable signal recovery from incomplete and inaccurate measurements, *Commun. Pure Appl. Math.* **59**(8), 1207–1223 (2006).
6. E. Candès, M. Wakin, An introduction to compressive sampling, *IEEE Signal Process. Mag.* **25**(2), 21–30 (2008).
7. A. Chambolle, R. De Vore, N. Lee, B. Lucier, Nonlinear wavelet image processing: variational problems, compression, and noise removal through wavelet shrinkage, *IEEE Trans. Image Process.* **7**(3), 319–335 (1998).
8. A. Chiang, *Fundamental Methods of Mathematical Economics* (McGraw-Hill, New York, 1984).
9. I. Daubechies, M. Defrise, C. De Mol, An iterative thresholding algorithm for linear inverse problems with a sparsity constraint, *Commun. Pure Appl. Math.* **57**(11), 1413–1457 (2004).
10. D. Donoho, Compressed sensing, *IEEE Trans. Inf. Theory* **52**(4), 1289–1306 (2006).
11. M. Gu, L. Lim, C. Wu, PARNES: A rapidly convergent algorithm for accurate recovery of sparse and approximately sparse signals. Technical report (2009). arXiv:0911.0492.
12. M. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. Natl. Bur. Stand.* **49**(6), 409–436 (1952).
13. G. Lan, R. Monteiro, Iteration complexity of first-order penalty methods for convex programming. Manuscript, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, June 2008
14. G. Lan, Z. Lu, R. Monteiro, Primal-dual first-order methods with $o(1/\epsilon)$ iteration-complexity for cone programming, *Math. Program.* 1–29 (2009).

15. J. Liu, L. Yuan, J. Ye, An efficient algorithm for a class of fused lasso problems, in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July (2010), pp. 323–332.
16. A. Nemirovski, Efficient methods in convex programming. *Lecture notes* (1994). http://www2.isye.gatech.edu/~nemirovs/Lect_EMCO.pdf.
17. A. Nemirovski, D. Yudin, *Problem Complexity and Method Efficiency in Optimization*. Wiley-Interscience Series in Discrete Mathematics (Wiley, New York, 1983).
18. Y. Nesterov, A method of solving a convex programming problem with convergence rate $O(1/k^2)$, *Sov. Math. Dokl.* **27**(2), 372–376 (1983).
19. Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course* (Kluwer Academic, Dordrecht, 2004).
20. Y. Nesterov, Gradient methods for minimizing composite objective function. CORE discussion paper (2007). http://www.ecore.be/DPs/dp_1191313936.pdf.
21. J. Nocedal, S. Wright, *Numerical Optimization*. Springer Series in Operations Research (Springer, Berlin, 2000).
22. B. Polyak, *Introduction to Optimization*. Translations Series in Mathematics and Engineering (Optimization Software, Publications Division, New York, 1987).
23. R. Tibshirani, Regression shrinkage and selection via the lasso, *J. R. Stat. Soc. B* **58**(1), 267–288 (1994).
24. P. Tseng, On accelerated proximal gradient methods for convex-concave optimization (2008). http://pages.cs.wisc.edu/~brecht/cs726docs/Tseng.APG.pdf.