A Lyapunov analysis for accelerated gradient methods: From deterministic to stochastic case

M. Laborde
joint work with Adam Oberman

McGill University

High Dimensional Hamilton-Jacobi PDEs
Workshop II: PDE and Inverse Problem Methods in Machine Learning,
IPAM, 20 April 2020

Motivation: Machine learning

FIGURE 1. Illustration of the classification map $f(x) : \mathcal{X} \to \mathcal{Y}$ on the ImageNet dataset.

- Label dataset: $S_m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$
- Hypothesis class of functions mapping data x to labels y:

$$\mathcal{H} = \{f(x, w) \; : \; w \in \mathbb{R}^n\}$$

- Loss function defined on labels: $\mathcal{L}(f(x_i, w), y_i)$
- Empirical Loss Minimization problem:

$$\min_{w \in \mathbb{R}^n} f(w) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f(x_i, w), y_i).$$

Optimization for machine learning: SGD

- Why is stochastic gradient important for machine learning? Evaluate full Loss (and its gradient) on all m data points can be too costly.

Optimization for machine learning: SGD

- Why is stochastic gradient important for machine learning? Evaluate full Loss (and its gradient) on all m data points can be too costly.
- Define random minibatch $I \subset \{1, \dots, m\}$,

$$f_I(w) = \frac{1}{|I|} \sum_{i \in I} \mathcal{L}(f(x_i, w), y_i).$$

Optimization for machine learning: SGD

- Why is stochastic gradient important for machine learning? Evaluate full Loss (and its gradient) on all m data points can be too costly.
- Define random minibatch $I \subset \{1, \dots, m\}$,

$$f_I(w) = \frac{1}{|I|} \sum_{i \in I} \mathcal{L}(f(x_i, w), y_i).$$

- Stochastic Gradient Descent corresponds to

$$w^{k+1} = w^k - h_k \nabla_w f_{I_k}(w^k), \qquad I_k \text{random and } h_k \text{ learning rate}$$

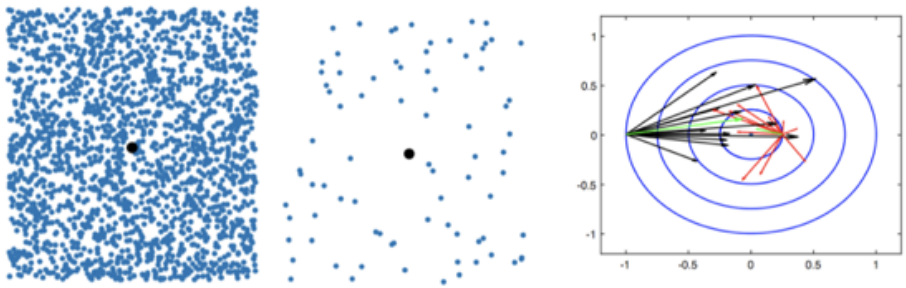


FIGURE 2. Full data set and a minibatch for data sampled uniformly in the square. Component gradients (black) and the minibatch gradient (green). Closer to the minimum, the relative error in the minibatch gradient is larger.

Stochastic Approximation

- The minibatch stochastic gradient is harder to analyze (future work)
- Instead make the standart Stochastic Approximation assumption: Let $\hat{g}$ be the stochastic gradient of f

$$\hat{g}(x, \xi) = \nabla f(x) + e(x, \xi),$$

where the noise e satisfies

$$\mathbb{E}[e] = 0 \qquad \text{and} \qquad \text{Var}(e) = \sigma^2.$$

- In particular,

$$\mathbb{E}[\hat{g}(x, \xi)] = \nabla f(x) \qquad \text{and} \qquad \mathbb{E}[|\hat{g}(x, \xi)|^2] = |\nabla f(x)|^2 + \sigma^2.$$

Strong convexity and L-smoothness

Let f be proper convex function, $x_* = \mathrm{argmin_x f(x)}$ and $f^* = f(x_*)$.

- f is L-smooth: $\forall x, y \in \mathbb{R}^n$

$$f(x) - f(y) + \nabla f(y) \cdot (y - x) \leqslant \frac{L}{2}|x - y|^2$$

- f is μ-strongly convex: $\forall x, y \in \mathbb{R}^n$

$$f(x) - f(y) + \nabla f(y) \cdot (y - x) \geqslant \frac{\mu}{2}|x - y|^2$$

$C_f := \frac{L}{\mu}$: condition number of f.

f quadratic: the constants are the smallest and largest eigenvalues of the Hessian of f.

1. Background on Gradient Descent and Stochastic Gradient Descent
   - Gradient Descent and Stochastic Gradient Descent
   - Accelerated gradient descent
2. Accelerated Stochastic algorithm
   - Convex Case
   - Strongly convex Case
3. Proofs of the results
   - Convex case
   - Strongly convex case
4. Future work

Table of contents

Gradient descent

Goal: minimize convex function f with gradient oracle.

- Continuous interpretation:

$$\dot{x} = -\nabla f(x)$$

- Algorithm:

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

- Convergence: Lyapunov analysis, $h = \frac{1}{L}$,

$$f(x_k) - f^* \leqslant \frac{L}{2k}|x_0 - x^*|^2 \qquad \text{(convex)}$$

and

$$f(x_k) - f^* \leqslant \left(1 - \frac{\mu}{L}\right)^k (f(x_0) - f^*) \qquad \text{(strongly convex)}$$

Stochastic Gradient descent

Goal: minimize convex function f with stochastic gradient oracle.

Algorithm: Stochastic Gradient descent

- Variable learning rate: e.g: $h_k = \frac{c}{k+k_0}$
- stochastic variance: σ^2

$$x_{k+1} = x_k - h_k \hat{g}(x_k),$$

where $\hat{g}(x) = \nabla f(x) + e(x, \xi)$ with

$$\mathbb{E}[e] = 0 \qquad \text{and} \qquad \text{Var}(e) = \sigma^2.$$

Convergence

From Bottou, Curtis, Nocedal (2016),

$$\mathbb{E}[f(x_k) - f^*] \leqslant C_1(1 - \mu h)^k + C_2 h \sigma^2$$

Need: $h \searrow 0$ or $\sigma^2 \searrow 0$.

Convergence

From Bottou, Curtis, Nocedal (2016),

$$\mathbb{E}[f(x_k) - f^*] \leqslant C_1(1 - \mu h)^k + C_2 h \sigma^2$$

Need: $h \searrow 0$ or $\sigma^2 \searrow 0$.

- Noise reduction techniques:
  - Variance reduction: SVRG (Jonhson, Zhang, 2013), SAG (Schmidt, Le Roux, Bach, 2013), SAGA (Defazio, Bach, Lacoste-Julien, 2014),...

Does not seem to help in deep learning.

Convergence

From Bottou, Curtis, Nocedal (2016),

$$\mathbb{E}[f(x_k) - f^*] \leqslant C_1(1 - \mu h)^k + C_2 h\sigma^2$$

Need: $h \searrow 0$ or $\sigma^2 \searrow 0$.

- Noise reduction techniques:
  - Variance reduction: SVRG (Jonhson, Zhang, 2013), SAG (Schmidt, Le Roux, Bach, 2013), SAGA (Defazio, Bach, Lacoste-Julien, 2014),...

    Does not seem to help in deep learning.
  - Strong Growth Condition: Assume

    $$\mathbb{E}[|\hat{g}(x, \xi)|^2] \leqslant \rho|\nabla f(x)|^2$$

    Strong assumption: can exactly fit the data!

Convergence

From Bottou, Curtis, Nocedal (2016),

$$\mathbb{E}[f(x_k) - f^*] \leqslant C_1(1 - \mu h)^k + C_2 h\sigma^2$$

Need: $h \searrow 0$ or $\sigma^2 \searrow 0$.

- Noise reduction techniques:
  - Variance reduction: SVRG (Jonhson, Zhang, 2013), SAG (Schmidt, Le Roux, Bach, 2013), SAGA (Defazio, Bach, Lacoste-Julien, 2014),...

    Does not seem to help in deep learning.
  - Strong Growth Condition: Assume

    $$\mathbb{E}[|\hat{g}(x, \xi)|^2] \leqslant \rho |\nabla f(x)|^2$$

    Strong assumption: can exactly fit the data!
- Variable learning rate:
  - Convergence in average: Polyak, Juditsky (1992), Rakhlin, Shamir, Sridharan (2011), Lacoste-Julien, Schmidt, Bach (2012), Shamir, Zhang (2013),...

    Optimal rates: $\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ (convex), $\mathcal{O}\left(\frac{1}{k}\right)$ (strongly convex)

Variable learning rate: Convergence for the last iterate

Define G^2 bound on stochastic gradient: $\mathbb{E}[\widehat{g}^2] \leq G^2$

Remark: $G^2 = L^2 D^2 + \sigma^2$ where D is the diameter of the domain

- Strongly convex case: Learning rate $h_k = \mathcal{O}\left(\frac{1}{k}\right)$
  - Previous results: Nemirovski, Juditsky, Lan, and Shapiro (2009), Shamir, Zhang (2013), Jain, Kakade, Netrapalli, Sidford (2018): optimal rate $\mathcal{O}\left(\frac{1}{k}\right)$ with constants depending on G^2, D and μ.

Variable learning rate: Convergence for the last iterate

Define G^2 bound on stochastic gradient: $\mathbb{E}[\widehat{g}^2] \leq G^2$

Remark: $G^2 = L^2 D^2 + \sigma^2$ where D is the diameter of the domain

- Strongly convex case: Learning rate $h_k = \mathcal{O}\left(\frac{1}{k}\right)$
  - Previous results: Nemirovski, Juditsky, Lan, and Shapiro (2009), Shamir, Zhang (2013), Jain, Kakade, Netrapalli, Sidford (2018): optimal rate $\mathcal{O}\left(\frac{1}{k}\right)$ with constants depending on G^2, D and μ.
  - Our result: $\mathcal{O}\left(\frac{1}{k}\right)$ rate with constants independent of the L-smoothness bound of the gradient (depends only on σ^2 and μ).

Variable learning rate: Convergence for the last iterate

Define G^2 bound on stochastic gradient: $\mathbb{E}[\widehat{g}^2] \leq G^2$

Remark: $G^2 = L^2 D^2 + \sigma^2$ where D is the diameter of the domain

- Strongly convex case: Learning rate $h_k = \mathcal{O}\left(\frac{1}{k}\right)$
  - Previous results: Nemirovski, Juditsky, Lan, and Shapiro (2009), Shamir, Zhang (2013), Jain, Kakade, Netrapalli, Sidford (2018): optimal rate $\mathcal{O}\left(\frac{1}{k}\right)$ with constants depending on G^2, D and μ.
  - Our result: $\mathcal{O}\left(\frac{1}{k}\right)$ rate with constants independent of the L-smoothness bound of the gradient (depends only on σ^2 and μ).
- Convex case:
  - Shamir, Zhang (2013): optimal rate order: $\frac{\log(k)}{\sqrt{k}}$ with a rate constant that depends on G^2 and D. Learning rate: $h_k = \mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$.
  - Jain, Nagaraj, Netrapalli (2019) remove the log factor assuming that the number of iterations is decided in advance.

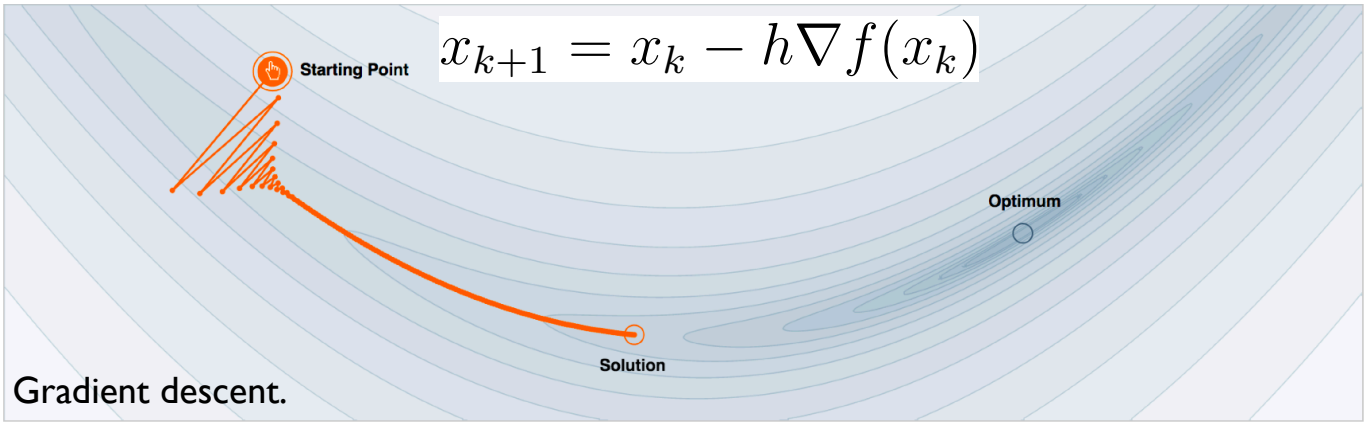
Variable learning rate: Convergence for the last iterate

Define G^2 bound on stochastic gradient: $\mathbb{E}[\widehat{g}^2] \leq G^2$

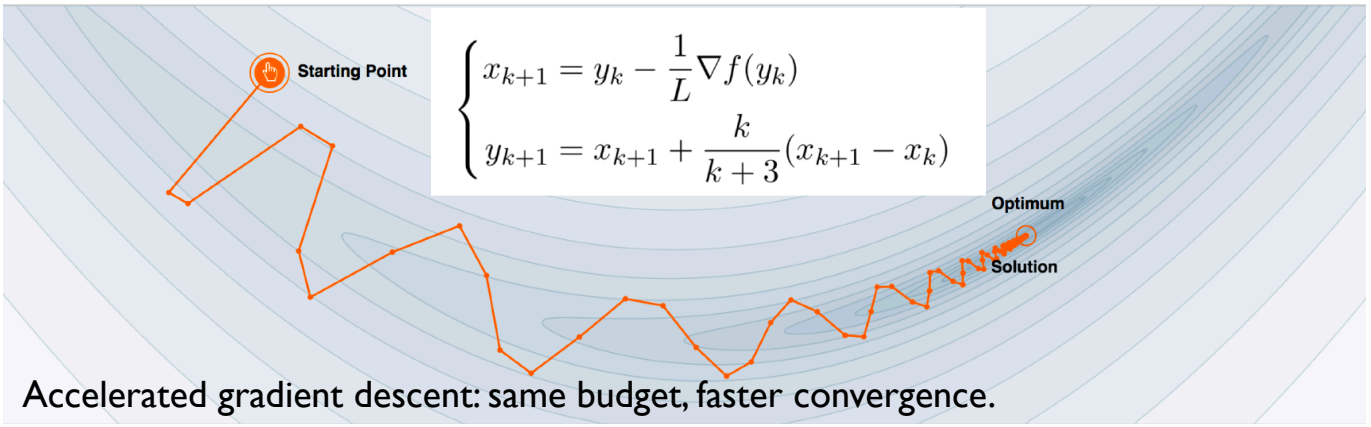
Remark: $G^2 = L^2 D^2 + \sigma^2$ where D is the diameter of the domain

- Strongly convex case: Learning rate $h_k = \mathcal{O}\left(\frac{1}{k}\right)$
  - Previous results: Nemirovski, Juditsky, Lan, and Shapiro (2009), Shamir, Zhang (2013), Jain, Kakade, Netrapalli, Sidford (2018): optimal rate $\mathcal{O}\left(\frac{1}{k}\right)$ with constants depending on G^2, D and μ.
  - Our result: $\mathcal{O}\left(\frac{1}{k}\right)$ rate with constants independent of the L-smoothness bound of the gradient (depends only on σ^2 and μ).
- Convex case:
  - Shamir, Zhang (2013): optimal rate order: $\frac{\log(k)}{\sqrt{k}}$ with a rate constant that depends on G^2 and D. Learning rate: $h_k = \mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$.
  - Jain, Nagaraj, Netrapalli (2019) remove the log factor assuming that the number of iterations is decided in advance.
  - Our result: $\mathcal{O}(\log(k)/\sqrt{k})$ rate for the last iterate, with a constant which depends on σ^2, but independent of the L-smoothness. New learning rate schedule

Nesterov's accelerated gradient descent



Gradient descent.



Accelerated gradient descent: same budget, faster convergence.

Momentum term remembers old gradients, overshoots instead of getting stuck.

Remark: two main A-GD algorithms, correspond to convex and strongly convex case. We focus on convex case, to simplify presentation. Strongly convex case is also covered.

ODE interpretation of Nesterov's method

- Nesterov's method for a convex, L-smooth function, f, [Nesterov, 2013]

$$\begin{cases} x_{k+1} = y_k - \frac{1}{L}\nabla f(y_k) \\ y_{k+1} = x_{k+1} + \frac{k}{k+3}(x_{k+1} - x_k) \end{cases} \tag{C-Nest}$$

ODE interpretation of Nesterov's method

- Nesterov's method for a convex, L-smooth function, f, [Nesterov, 2013]

$$\begin{cases} x_{k+1} = y_k - \frac{1}{L}\nabla f(y_k) \\ y_{k+1} = x_{k+1} + \frac{k}{k+3}(x_{k+1} - x_k) \end{cases} \tag{C-Nest}$$

- Connection between (C-Nest) and

$$\ddot{x} + \frac{3}{t}\dot{x} + \nabla f(x) = 0, \tag{A-ODE}$$

[Su, Boyd, Candés, 2014].

ODE interpretation of Nesterov's method

- Nesterov's method for a convex, L-smooth function, f, [Nesterov, 2013]

$$\begin{cases} x_{k+1} = y_k - \frac{1}{L}\nabla f(y_k) \\ y_{k+1} = x_{k+1} + \frac{k}{k+3}(x_{k+1} - x_k) \end{cases} \tag{C-Nest}$$

- Connection between (C-Nest) and

$$\ddot{x} + \frac{3}{t}\dot{x} + \nabla f(x) = 0, \tag{A-ODE}$$

  [Su, Boyd, Candés, 2014].
- (A-ODE) can be written as the first order system

$$\begin{cases} \dot{x} = \frac{2}{t}(v - x) \\ \dot{v} = -\frac{t}{2}\nabla f(x). \end{cases} \tag{S-A-ODE}$$

  Connection: finite differences in time, and evaluate gradient at y convex combination between x and v.

Convergence of Nesterov's algorithm

- Lyapunov analysis: Su, Boyd, Candès (2014), Wibisono, Wilson, Jordan (2016), Wilson, Recht, Jordan (2016), Wilson, Mackey, Wibisono (2019),...

- Define

$$E^{ac,c}(t,x,v) = t^2(f(x) - f^*) + 2|v - x^*|^2$$

Then,

$$f(x) - f^* \leqslant \frac{2}{t^2}|v_0 - x^*|^2,$$

and for $h = \frac{1}{\sqrt{L}}$, $t_k = \frac{k+2}{\sqrt{L}}$,

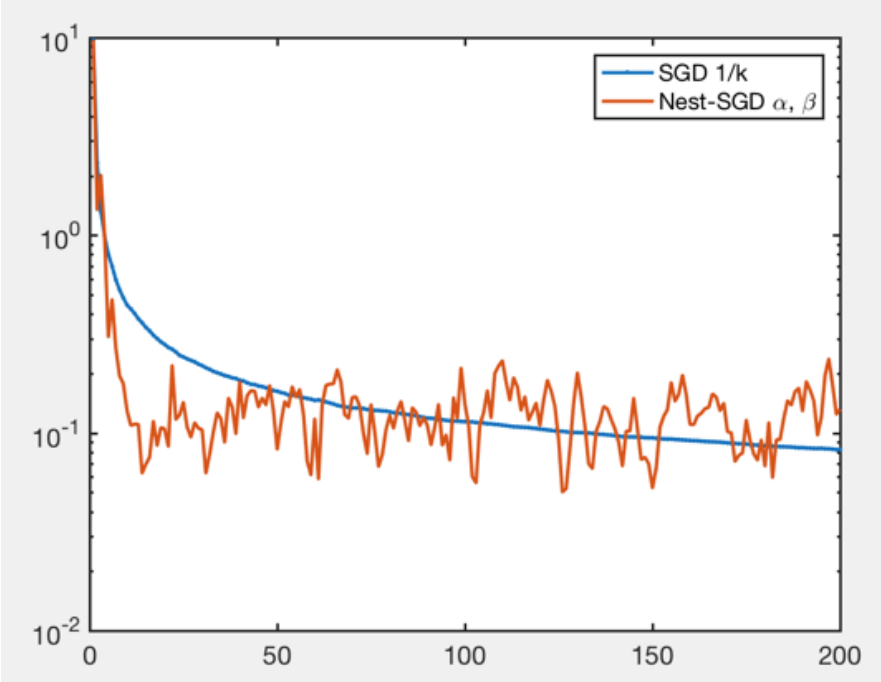
$$f(x_k) - f^* \leqslant \frac{2L}{(k+1)^2}|v_0 - x^*|^2$$

Heuristic accelerated SGD

Use standard Nesterov algorithm for Neural Network, with

- Stochastic gradient,
- constant learning rate $\alpha = \frac{1}{L}$,
- momentum $\beta = 0.9$

Scheduled learning rate for SGD, converges at $1/k$ rate.



- Works well in practice: fast initial drop, then noise takes over.
- What could you prove? Vanilla SGD converges at rate $1/k$. Have lower bounds. Only room for improvement is the rate constant.

Our goal: better understand this algorithm to build an effective stochastic version

Accelerated SGD?

- Go back to ODE for Nesterov (there is more than one)
- Redo convergence rate, first in continuous, then in discrete time (Lyapunov analysis).
- Find the dissipation of the Lyapunov analysis for stochastic version
- Prove the convergence rate
- Use proof to tune the learning rate schedule, for optimal convergence rate
- Determine if we get a practical algorithm.
- There are two algorithms: convex version, and strongly convex version.

Table of contents

Another ODE for Nesterov's Method

Our starting point is a perturbation of (S-A-ODE)

$$\begin{cases} \dot{x} = \frac{2}{t}(v - x) - \frac{1}{\sqrt{L}}\nabla f(x) \\ \dot{v} = -\frac{t}{2}\nabla f(x), \end{cases} \tag{1st-ODE}$$

The system (1st-ODE) is equivalent to the following ODE

$$\ddot{x} + \frac{3}{t}\dot{x} + \nabla f(x) = -\frac{1}{\sqrt{L}}\left(D^2 f(x)\cdot \dot{x} + \frac{1}{t}\nabla f(x)\right) \tag{H-ODE}$$

which has an additional Hessian damping term with coefficient $1/\sqrt{L}$.

Another ODE for Nesterov's Method

Our starting point is a perturbation of (S-A-ODE)

$$\begin{cases} \dot{x} = \frac{2}{t}(v - x) - \frac{1}{\sqrt{L}}\nabla f(x) \\ \dot{v} = -\frac{t}{2}\nabla f(x), \end{cases} \tag{1st-ODE}$$

The system (1st-ODE) is equivalent to the following ODE

$$\ddot{x} + \frac{3}{t}\dot{x} + \nabla f(x) = -\frac{1}{\sqrt{L}}\left(D^2 f(x)\cdot \dot{x} + \frac{1}{t}\nabla f(x)\right) \tag{H-ODE}$$

which has an additional Hessian damping term with coefficient $1/\sqrt{L}$.

- 2nd order ODE with Hessian damping: Alvarez, Attouch, Bolte, Redont (2002), Attouch, Peyrouquet, Redont (2016)

Another ODE for Nesterov's Method

Our starting point is a perturbation of (S-A-ODE)

$$\begin{cases} \dot{x} = \frac{2}{t}(v - x) - \frac{1}{\sqrt{L}}\nabla f(x) \\ \dot{v} = -\frac{t}{2}\nabla f(x), \end{cases} \tag{1st-ODE}$$

The system (1st-ODE) is equivalent to the following ODE

$$\ddot{x} + \frac{3}{t}\dot{x} + \nabla f(x) = -\frac{1}{\sqrt{L}}\left(D^2 f(x)\cdot \dot{x} + \frac{1}{t}\nabla f(x)\right) \tag{H-ODE}$$

which has an additional Hessian damping term with coefficient $1/\sqrt{L}$.

- 2nd order ODE with Hessian damping: Alvarez, Attouch, Bolte, Redont (2002), Attouch, Peyrouquet, Redont (2016)
- Shi, Du, Jordan, Su (2018) introduced a family of high resolution second order ODEs which also lead to Nesterov's method: (H-ODE), special case $s = \frac{1}{\sqrt{L}}$

Advantages of (1st-ODE)

- First order system (no Hessian)

- The system (1st-ODE) can be discretized to recover Nesterov's method using an explicit discretization with a constant time step $h = \frac{1}{\sqrt{L}}$

- Decrease the same Lyapunov function as (S-A-ODE) but faster (gap).

$$\Rightarrow \text{Faster convergence for } h < \frac{1}{\sqrt{L}}$$

From (1st-ODE) to Nesterov

Define the learning rate $h_k > 0$ and a discretization of the total time t_k.
The time discretization of (1st-ODE) with gradients evaluated at

$$y_k = \left(1 - \frac{2h_k}{t_k}\right) x_k + \frac{2h_k}{t_k} v_k.$$

is given by

$$\begin{cases} x_{k+1} - x_k = \dfrac{2h_k}{t_k}(v_k - x_k) - \dfrac{h_k}{\sqrt{L}} \nabla f(y_k), \\ v_{k+1} - v_k = -\dfrac{h_k t_k}{2} \nabla f(y_k), \end{cases} \tag{FE-C}$$

From (1st-ODE) to Nesterov

Define the learning rate $h_k > 0$ and a discretization of the total time t_k.
The time discretization of (1st-ODE) with gradients evaluated at

$$y_k = \left(1 - \frac{2h_k}{t_k}\right) x_k + \frac{2h_k}{t_k} v_k.$$

is given by

$$\begin{cases} x_{k+1} - x_k = \dfrac{2h_k}{t_k}(v_k - x_k) - \dfrac{h_k}{\sqrt{L}} \nabla f(y_k), \\ v_{k+1} - v_k = -\dfrac{h_k t_k}{2} \nabla f(y_k), \end{cases} \tag{FE-C}$$

Proposition

The discretization of (1st-ODE) *given by* (FE-C) *with* $h_k = h = \frac{1}{\sqrt{L}}$ *and* $t_k = h(k+2)$ *is equivalent to the standard Nesterov's method* (C-Nest).

Fix learning rate, get Nesterov's algorithm

Stochastic gradient version

$$\begin{cases} x_{k+1} - x_k = \dfrac{2h_k}{t_k}(v_k - x_k) - \dfrac{h_k}{\sqrt{L}}(\nabla f(y_k) + e_k), \\ v_{k+1} - v_k = -\dfrac{h_k t_k}{2}(\nabla f(y_k) + e_k), \end{cases} \tag{1}$$

with

$$h_k = \frac{c}{k^\alpha} \leqslant \frac{1}{\sqrt{L}}, \qquad t_k = \sum_{i=1}^{k} h_i, \qquad \text{and} \qquad \alpha \in \left[\frac{3}{4}, 1\right)$$

Same algorithm as before, but now

- Variable learning rate
- Stochastic gradient
- Optimal exponent for learning rate: $\alpha = \frac{3}{4}$
- Determined by convergence rate analysis, below

Convergence result (convex case)

Define the continuous time Lyapunov function

$$E^{ac,c}(t,x,v) := t^2(f(x)-f^*)+2|v-x^*|^2$$

Define the discrete time Lyapunov function E_k^c by

$$E_k^{ac,c} = E^{ac,c}(t_{k-1},x_k,v_k)$$

Proposition (L., Oberman, 2020)

Assume $h_k := \frac{c}{k^\alpha} \leq \frac{1}{\sqrt{L}}$ *and* $t_k = \sum_{i=1}^k h_i$, *then for* $\alpha = \frac{3}{4}$,

$$\mathbb{E}[f(x_k)] - f^* \leqslant \frac{\frac{1}{16c^2}E_0 + c^2\sigma^2(1+\log(k))}{(k^{1/4}-1)^2}$$

Comparison with previous results

	Shamir, Zhang (2013)	Acc. SGD
h_k	$\frac{c^2}{\sqrt{k}}$	$\frac{c}{k^{3/4}}$
Rate	$\left(\frac{D^2}{c^2} + c^2 G^2\right) \frac{(2+\log(k))}{\sqrt{k}}$	$\frac{\frac{E_0}{16c^2} + c^2\sigma^2(1+\log(k))}{(k^{1/4}-1)^2}$

Table: Convergence rate $\mathbb{E}[f(x_k) - f^*]$ after k steps. G^2 is a bound on $\mathbb{E}[\hat{g}^2]$. E_0 is the initial value of the Lyapunov function.

Interpreting Improved rate: remove dependence on L-smoothness from the rate

Comparison with previous results

	Shamir, Zhang (2013)	Acc. SGD
h_k	$\frac{c^2}{\sqrt{k}}$	$\frac{c}{k^{3/4}}$
Rate	$\left(\frac{D^2}{c^2} + c^2 G^2\right) \frac{(2+\log(k))}{\sqrt{k}}$	$\frac{\frac{E_0}{16c^2} + c^2\sigma^2(1+\log(k))}{(k^{1/4}-1)^2}$

Table: Convergence rate $\mathbb{E}[f(x_k) - f^*]$ after k steps. G^2 is a bound on $\mathbb{E}[\hat{g}^2]$. E_0 is the initial value of the Lyapunov function.

Interpreting Improved rate: remove dependence on L-smoothness from the rate

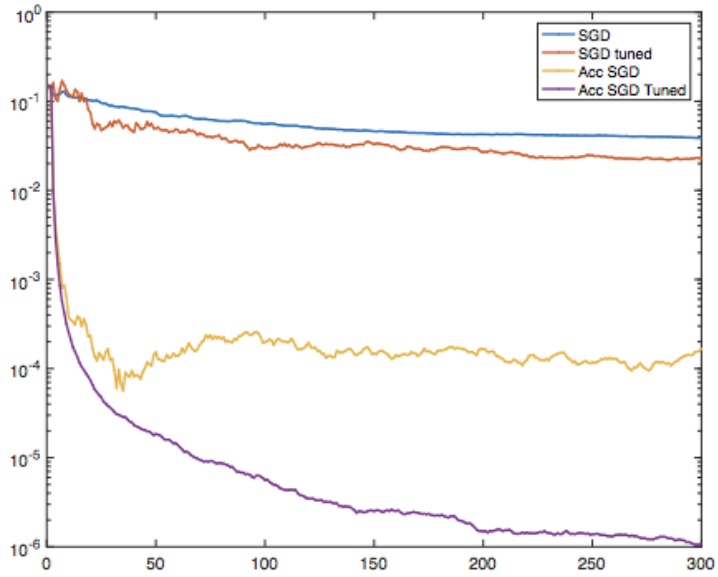
This is a nice theoretical result, and nice example of continuous time method. In order for it to be of practical use, desire

- practical algorithm, faster in practice (saw this for simple examples)
- remains faster when theory no longer applies (work in progress/future work)
  - e.g. nonconvex examples
  - e.g. training DNNs

Acc-SGD results: synthetic noise

- Simple quadratic example, synthetic noise
- Tuned version just means optimize the learning rate

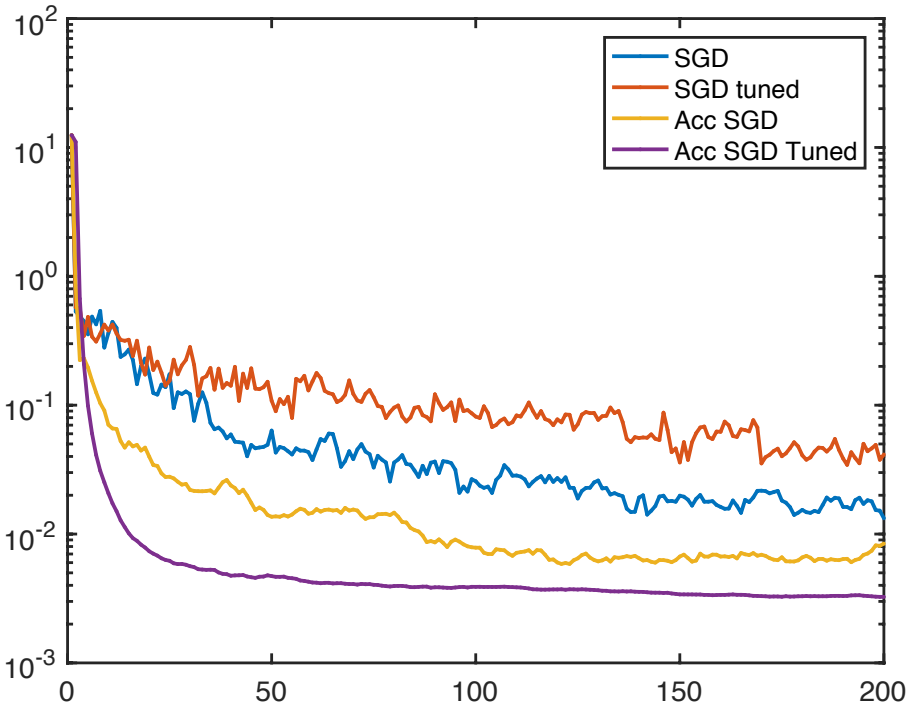
- SGD (with $1/k$ schedule):
  - guess/tune learning rate
  - Results at $k = 300$: 10^{-1}, $5 \cdot 10^{-2}$
- A-SGD ($\alpha = \frac{3}{4}$):
  - guess/tune parameter, c
  - Results at $k = 300$: 10^{-4}, 10^{-6}



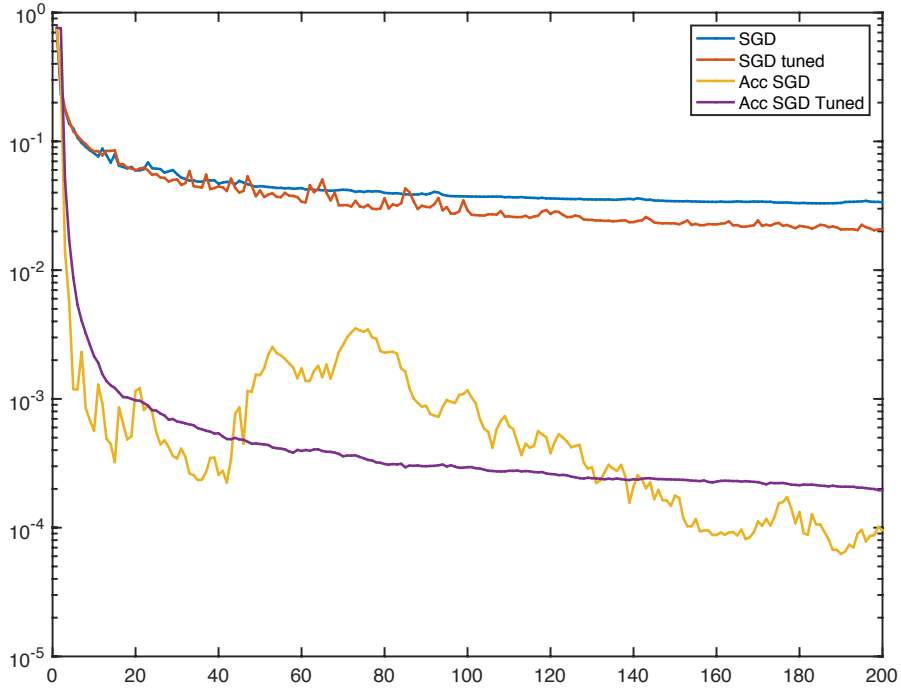
Quadratic function, C = 300

Acc-SGD results: minibatch SGD (1)

- Faster convergence than SGD
- Improved performance on poorly conditioned examples



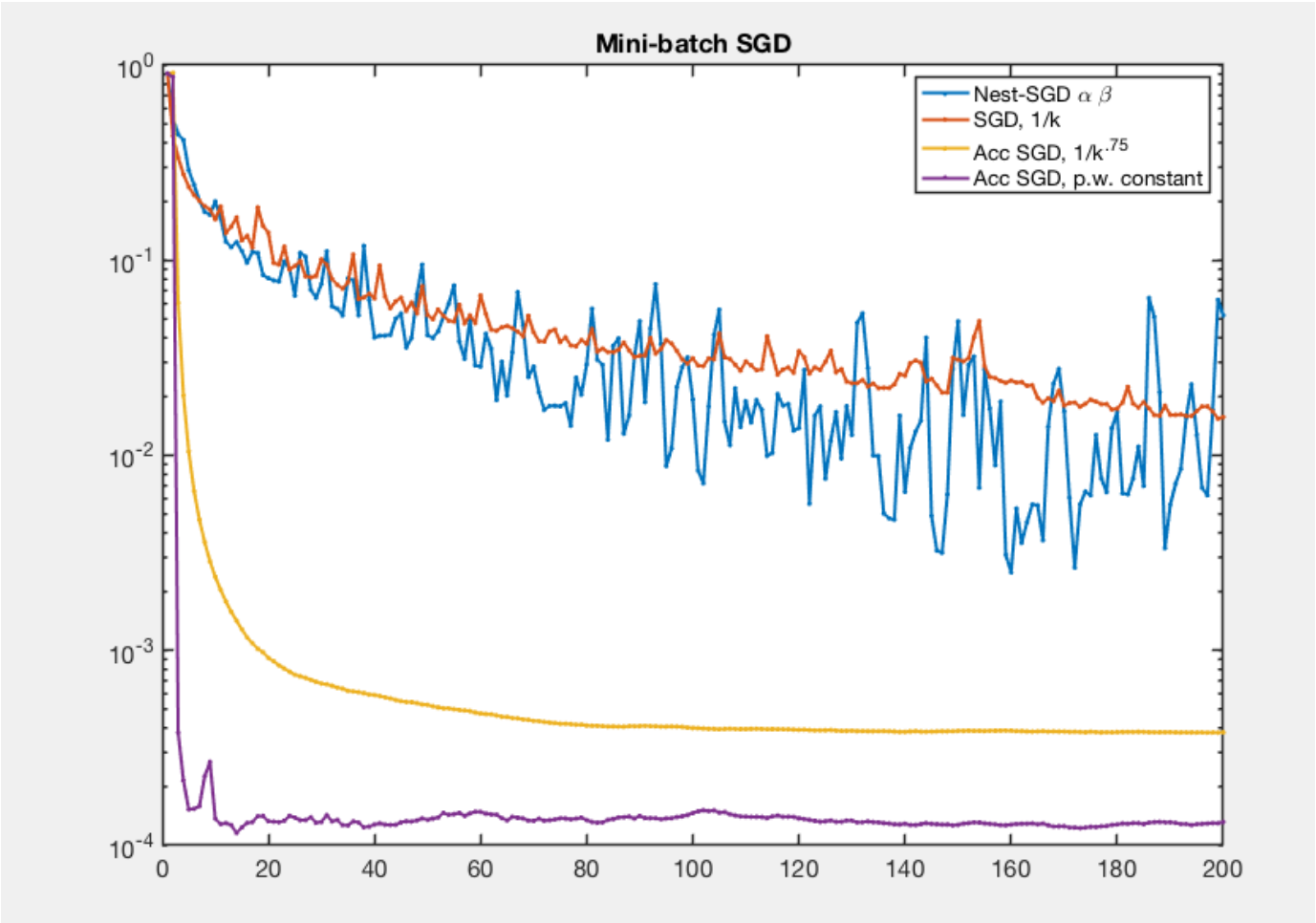
Condition number C = 1
approximately 10 x accuracy



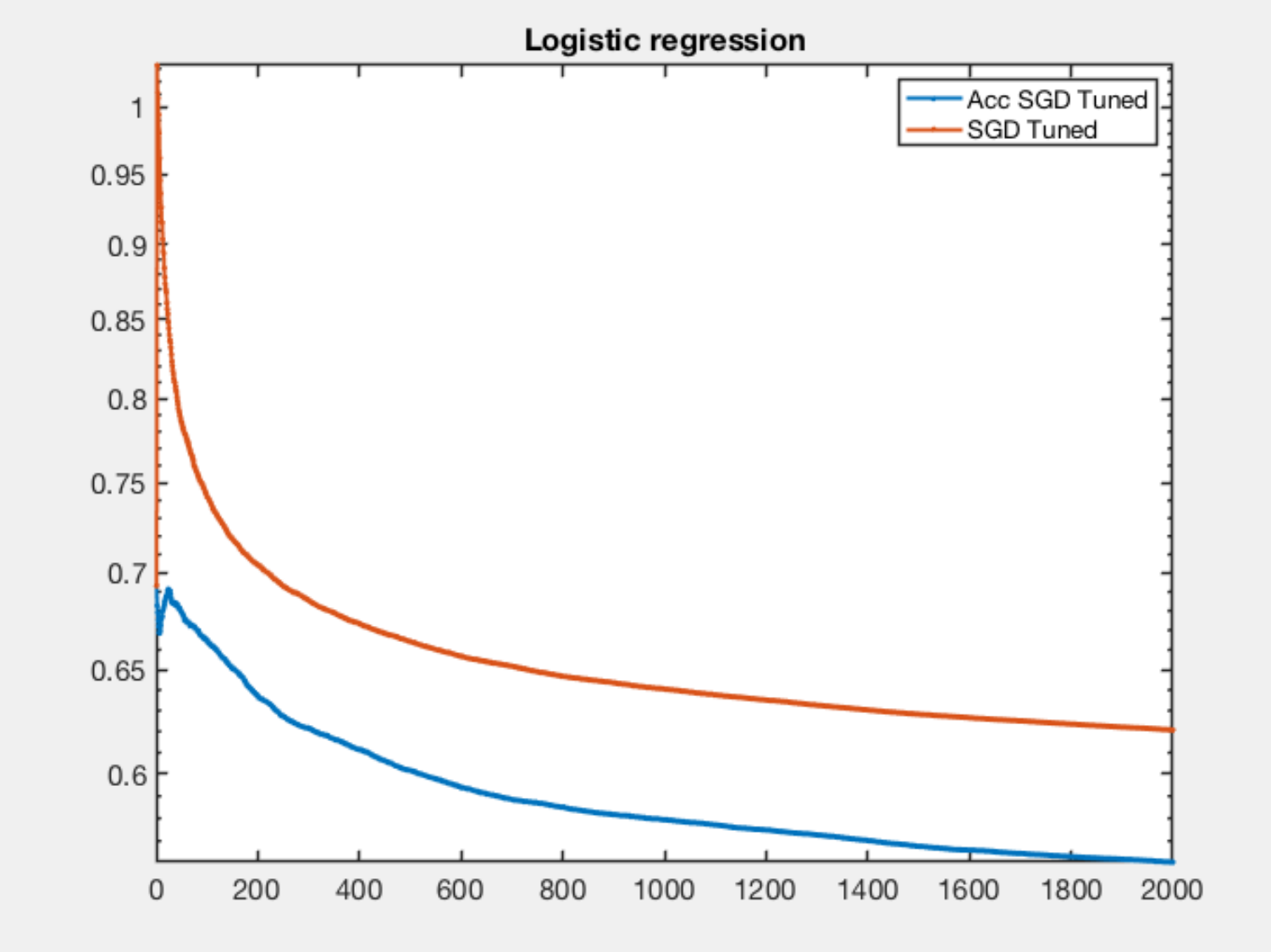
Condition number C = 100
approximately 100 x accuracy

Acc-SGD results: minibatch SGD (2)

Robustness to the change of parameters



logistic regression



Strongly convex case: Nesterov's method

- Nesterov's method for a μ-strongly convex, L-smooth function, f, [Nesterov, 2013]

$$\begin{cases} x_{k+1} = y_k - \frac{1}{L}\nabla f(y_k) \\ y_{k+1} = x_{k+1} + \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(x_{k+1} - x_k) \end{cases} \tag{SC-Nest}$$

Strongly convex case: Nesterov's method

- Nesterov's method for a μ-strongly convex, L-smooth function, f, [Nesterov, 2013]

$$\begin{cases} x_{k+1} = y_k - \frac{1}{L}\nabla f(y_k) \\ y_{k+1} = x_{k+1} + \frac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(x_{k+1} - x_k) \end{cases} \tag{SC-Nest}$$

- Connection between (SC-Nest) and Heavy Ball's method

$$\ddot{x} + 2\sqrt{\mu}\dot{x} + \nabla f(x) = 0, \tag{CS-A-ODE}$$

[Polyak, 1964].

Strongly convex case: Nesterov's method

- Nesterov's method for a μ-strongly convex, L-smooth function, f, [Nesterov, 2013]

$$\begin{cases} x_{k+1} = y_k - \dfrac{1}{L}\nabla f(y_k) \\ y_{k+1} = x_{k+1} + \dfrac{\sqrt{L}-\sqrt{\mu}}{\sqrt{L}+\sqrt{\mu}}(x_{k+1} - x_k) \end{cases} \tag{SC-Nest}$$

- Connection between (SC-Nest) and Heavy Ball's method

$$\ddot{x} + 2\sqrt{\mu}\dot{x} + \nabla f(x) = 0, \tag{CS-A-ODE}$$

  [Polyak, 1964].
- (CS-A-ODE) can be written as the first order system

$$\begin{cases} \dot{x} = \sqrt{\mu}(v - x), \\ \dot{v} = \sqrt{\mu}(x - v) - \frac{1}{\sqrt{\mu}}\nabla f(x). \end{cases} \tag{2}$$

Another ODE for Nesterov's Method

Again, Our starting point is a perturbation of (2)

$$\begin{cases} \dot{x} = \sqrt{\mu}(v - x) - \frac{1}{\sqrt{L}}\nabla f(x), \\ \dot{v} = \sqrt{\mu}(x - v) - \frac{1}{\sqrt{\mu}}\nabla f(x), \end{cases} \qquad \text{(1st-ODE-SC)}$$

The system (1st-ODE) is equivalent to the following ODE

$$\ddot{x} + 2\sqrt{\mu}\dot{x} + \nabla f(x) = -\frac{1}{\sqrt{L}}\left(D^2 f(x)\cdot \dot{x} + \sqrt{\mu}\nabla f(x)\right), \qquad \text{(H-ODE-SC)}$$

which has an additional Hessian damping term with coefficient $1/\sqrt{L}$.

- Special case of high resolution second order ODEs (Shi and al., 2018) which also lead to Nesterov's method
- Decrease faster the same Lyapunov function as (2).

Stochastic gradient version

Learning rate: $h_k > 0$. Define

$$\begin{cases} x_{k+1} - x_k = \lambda_k(v_k - x_k) - \frac{h_k}{\sqrt{L}}(\nabla f(y_k) + e_k), \\ v_{k+1} - v_k = \lambda_k(x_k - v_k) - \frac{h_k}{\sqrt{\mu}}(\nabla f(y_k) + e_k), \\ y_k = (1-\lambda_k)x_k + \lambda_k v_k, \qquad \lambda_k = \frac{h_k\sqrt{\mu}}{1+h_k\sqrt{\mu}}. \end{cases} \tag{FE-SC}$$

Same algorithm as before, but now

- Variable learning rate
- Stochastic gradient
- To simplify: replace $\sqrt{\mu}$ by $\frac{\sqrt{\mu}}{1+h_k\sqrt{\mu}}$

Convergence result (strongly convex case)

Define the continuous time Lyapunov function

$$E^{ac,sc}(x,v) := f(x) - f^* + \frac{\mu}{2}|v - x^*|^2$$

Discrete time Lyapunov function $E_k^{ac,sc} := E^{ac,sc}(x_k, v_k)$.

Proposition (L., Oberman, 2020)

Assume $E_0^{ac,sc} \leqslant \frac{1}{\sqrt{L}}$. *If* $h_k := \frac{2\sqrt{\mu}}{\mu k + 4\sigma^2 E_0^{ac,sc}-1}$,

$$\mathbb{E}[f(x_k)] - f^* \leqslant \frac{4\sigma^2}{\mu k + 4\sigma^2 E_0^{ac,sc-1}}.$$

Comparison with previous results

Learning rate: $h_k = \mathcal{O}\left(\frac{1}{k}\right)$

Nemirovski et al. [2009]	Shamir and Zhang [2013]	Jain et al. [2019]	Acc. SGD
$\frac{2C_f G^2}{\mu k}$	$\frac{17G^2(1+\log(k))}{\mu k}$	$\frac{130G^2}{\mu k}$	$\frac{4\sigma^2}{\mu k + 4\sigma^2 E_0^{-1}}$

Convergence rate $\mathbb{E}[f(x_k) - f^*]$ after k steps: G^2 is a bound on $\mathbb{E}[\hat{g}(x)^2]$, and σ^2 variance. E_0 is the initial value of the Lyapunov function

Improvement to the rate constant: independent of L

Table of contents

Proof Outline

- Starting from ODE, and rate-generating Lyapunov function
- Prove rate of decrease of Lyapunov function in time
- Then discretize ODE, constant time, to get a gradient algorithm
- Prove rate in k (consistent with time, e.g $t = hk$),
- Dissipation of the Lyapunov function for stochastic gradients: additional error term
- Now use variable learning rate, to sum errors
- Determine learning rate (parameters) from the sum
- Obtain unified analysis for both stochastic and gradient case

Convex case

- Define $t_k = \sum_{i=1}^{k} h_i$, with $h_i = \frac{c}{k^\alpha}$ and $\alpha < 1$,

$$\begin{cases} x_{k+1} - x_k = \dfrac{2h_k}{t_k}(v_k - x_k) - \dfrac{h_k}{\sqrt{L}}(\nabla f(y_k) + e_k), \\ v_{k+1} - v_k = -\dfrac{h_k t_k}{2}(\nabla f(y_k) + e_k), \\ y_k = \left(1 - \dfrac{2h_k}{t_k}\right) x_k + \dfrac{2h_k}{t_k} v_k. \end{cases}$$

- Take the same Lyapunov function as for Nesterov's method:

$$E_k^{ac,c} = t_{k-1}^2 (f(x_k) - f^*) + 2|v_k - x^*|^2.$$

- Result: For $\alpha = \frac{3}{4}$,

$$\mathbb{E}[f(x_k)] - f^* \leqslant \frac{\frac{1}{16c^2} E_0 + c^2\sigma^2(1 + \log(k))}{(k^{1/4} - 1)^2}$$

Dissipation estimate

- Same analysis as in Su, Boyd and Candés but with $\nabla f(y_k) + e_k$ instead of $\nabla f(y_k)$: For $h_k \leqslant \frac{1}{\sqrt{L}}$

$$E_{k+1}^{ac,c} - E_k^{ac,c} \leq h_k \beta_k,$$

where $\beta_k := -t_k \langle 2(v_k - x^*) - \frac{t_k}{\sqrt{L}} \nabla f(y_k), e_k \rangle + 2 h_k t_k^2 \left\langle \nabla f(y_k) + \frac{e_k}{2}, e_k \right\rangle$.

- Expectation: $\mathbb{E}[\beta_k] = h_k t_k^2 \sigma^2$, so

$$\mathbb{E}[E_{k+1}^{ac,c}] - E_k^{ac,c} \leq h_k^2 t_k^2 \sigma^2,$$

and

$$\mathbb{E}[E_k^{ac,c}] \geq t_{k-1}^2 (\mathbb{E}[f(x_k)] - f^*).$$

Proof of the result

- Summing over k,

$$t_{k-1}^2(\mathbb{E}[f(x_k)] - f^*) \leqslant E_0 + \sigma^2 \sum_{i=1}^{k-1} h_i^2 t_i^2$$

- By comparison series-integral

$$t_{k-1}^2 \geqslant 16c^2(k^{1/4} - 1)^2,$$

and

$$\sum_{i=1}^{k-1} h_i^2 t_i^2 \leqslant 16c^4\sigma^2(1 + \log(k))$$

- Conclusion:

$$\mathbb{E}[f(x_k)] - f^* \leqslant \frac{1}{t_{k-1}^2}\left(E_0 + \sigma^2 \sum_{i=1}^{k-1} h_i^2 t_i^2\right) \leqslant \frac{\frac{1}{16c^2}E_0 + c^2\sigma^2(1 + \log(k))}{(k^{1/4} - 1)^2}$$

Strongly convex case

- Define,

$$\begin{cases} x_{k+1} - x_k = \lambda_k(v_k - x_k) - \frac{h_k}{\sqrt{L}}(\nabla f(y_k) + e_k), \\ v_{k+1} - v_k = \lambda_k(x_k - v_k) - \frac{h_k}{\sqrt{\mu}}(\nabla f(y_k) + e_k), \\ y_k = (1 - \lambda_k)x_k + \lambda_k v_k, \qquad \lambda_k = \frac{h_k\sqrt{\mu}}{1+h_k\sqrt{\mu}}. \end{cases}$$

- Take the same Lyapunov function as for Nesterov's method:

$$E^{ac,sc}(x, v) = f(x) - f^* + \frac{\mu}{2}|v - x^*|^2, \qquad E_k^{ac,sc} := E^{ac,sc}(x_k, v_k)$$

- Result: For

$$h_k := \frac{2}{\sqrt{\mu}(k + (\alpha E_0^{ac,sc})^{-1})}, \qquad \alpha := \frac{\mu}{4\sigma^2},$$

then,

$$\mathbb{E}[E_k^{ac,sc}] \leqslant \frac{4\sigma^2}{\mu k + 4\sigma^2 E_0^{ac,sc\,-1}}.$$

Dissipation estimate

- Same analysis as for Heavy ball's method but with $\nabla f(y_k) + e_k$ instead of $\nabla f(y_k)$: For $h_k \leqslant \frac{1}{\sqrt{L}}$

$$E_{k+1}^{ac,sc} \leqslant (1 - h_k\sqrt{\mu})E_k^{ac,sc} + h_k\beta_k,$$

where
$\beta_k := 2h_k \left\langle \nabla f(y_k) + \frac{e_k}{2}, e_k \right\rangle - \left\langle \sqrt{\mu}(x_k - y_k + v_k - x^*) - \frac{1}{\sqrt{L}}\nabla f(y_k), e_k \right\rangle.$

- Expectation: $\mathbb{E}[\beta_k] = h_k\sigma^2$, so

$$\mathbb{E}[E_{k+1}^{ac,sc}] \leqslant (1 - h_k\sqrt{\mu})E_k^{ac,sc} + h_k^2\sigma^2$$

Proof of the result

By induction, show that

$$\mathbb{E}[E_k^{ac,sc}] \leqslant \frac{1}{\alpha(k+\alpha^{-1}E_0^{-1})}, \qquad \alpha = \frac{\mu}{4\sigma^2}.$$

- Initialization, $k = 0$: trivial
- For all $k \geqslant 1$,

$$\mathbb{E}[E_{k+1}^{ac,sc}] \leqslant (1 - h_k\sqrt{\mu})E_k^{ac,sc} + h_k^2\sigma^2$$

and by definition of h_k, α, and using the induction assumption,

$$\begin{aligned}
\mathbb{E}[E_{k+1}^{ac,sc}] &\leqslant \left(1 - \frac{2}{k+\alpha^{-1}E_0^{-1}}\right)\frac{1}{\alpha(k+\alpha^{-1}E_0^{-1})} + \frac{1}{\alpha(k+\alpha^{-1}E_0^{ac,sc\,-1})^2} \\
&\leqslant \frac{1}{\alpha(k+\alpha^{-1}E_0^{ac,sc\,-1})} - \frac{1}{\alpha(k+\alpha^{-1}E_0^{ac,sc\,-1})^2} \\
&\leqslant \frac{1}{\alpha(k+1+\alpha^{-1}E_0^{ac,sc\,-1})},
\end{aligned}$$

which concludes the proof.

Table of contents

Possible extensions and future works

- General abstract Lyapunov analysis: Can be applied to other problems as saddle point, for example.
- Extension to more general error: Deal with variance which depends on x
- Extension to non convex setting: For example assuming Polyak-Lojasiewicz condition
- Numerical simulations: Test on Deep Neural Networks

Thank you for your attention