Worst-Case Complexity Guarantees and Nonconvex Smooth Optimization

Frank E. Curtis, Lehigh University

"Beyond Convexity" Workshop, Oaxaca, Mexico

26 October 2017

Outline

Outline

Unconstrained nonconvex optimization

Given $f : \mathbb{R}^n \to \mathbb{R}$, consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x).$$

What type of method to use?

- First-order or second-order?
- One motivation for second-order: improved complexity bounds.

Unconstrained nonconvex optimization

Given $f : \mathbb{R}^n \to \mathbb{R}$, consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x).$$

What type of method to use?

- First-order or second-order?
- One motivation for second-order: improved complexity bounds.

Main message of this talk:

- For nonconvex optimization...
- ...complexity bounds should be taken with a grain of salt (for now).

Unconstrained nonconvex optimization

Given $f : \mathbb{R}^n \to \mathbb{R}$, consider the unconstrained optimization problem

$$\min_{x \in \mathbb{R}^n} f(x).$$

What type of method to use?

- First-order or second-order?
- One motivation for second-order: improved complexity bounds.

Main message of this talk:

- For nonconvex optimization...
- ...complexity bounds should be taken with a grain of salt (for now).

Parting words:

- There are other, better motivations for second-order methods.

Methods of interest

Trust region methods

- Decades of algorithmic development
- Levenberg (1944); Marquardt (1963); Powell (1970); many more!

Cubic regularization methods

- Relatively recent algorithmic development; fewer variants
- Griewank (1981); Nesterov & Polyak (2006); Cartis, Gould, & Toint (2011)

Methods of interest

Trust region methods

- Decades of algorithmic development
- Levenberg (1944); Marquardt (1963); Powell (1970); many more!

Cubic regularization methods

- Relatively recent algorithmic development; fewer variants
- Griewank (1981); Nesterov & Polyak (2006); Cartis, Gould, & Toint (2011)

Theoretical guarantees to assess a nonconvex optimization algorithm:

- Global convergence, i.e., $\nabla f(x_k) \to 0$ and maybe $\min(\text{eig}(\nabla^2 f(x_k))) \geq 0$
- Local convergence rate, i.e., $\|\nabla f(x_{k+1})\|_2 / \|\nabla f(x_k)\|_2 \to 0$ (or more)
- Worst-case complexity, i.e., upper bound on number of iterations[1] to achieve

$$\|\nabla f(x_k)\|_2 \leq \epsilon \quad \text{and perhaps} \quad \min(\text{eig}(\nabla^2 f(x_k))) \geq -\epsilon \quad \text{for some} \quad \epsilon > 0$$

[1] ...or function evaluations, subproblem solves, etc.

Methods of interest

Trust region methods

- Decades of algorithmic development
- Levenberg (1944); Marquardt (1963); Powell (1970); many more!
- Global convergence, local quadratic rate when $\nabla^2 f(x_*) \succ 0$
- $\mathcal{O}(\epsilon^{-2})$ complexity to first-order ϵ-criticality, $\mathcal{O}(\epsilon^{-3})$ to second-order

Cubic regularization methods

- Relatively recent algorithmic development; fewer variants
- Griewank (1981); Nesterov & Polyak (2006); Cartis, Gould, & Toint (2011)
- Global convergence, local quadratic rate when $\nabla^2 f(x_*) \succ 0$
- $\mathcal{O}(\epsilon^{-3/2})$ complexity to first-order ϵ-criticality, $\mathcal{O}(\epsilon^{-3})$ to second-order

Theoretical guarantees to assess a nonconvex optimization algorithm:

- Global convergence, i.e., $\nabla f(x_k) \to 0$ and maybe $\min(\text{eig}(\nabla^2 f(x_k))) \geq 0$
- Local convergence rate, i.e., $\|\nabla f(x_{k+1})\|_2 / \|\nabla f(x_k)\|_2 \to 0$ (or more)
- Worst-case complexity, i.e., upper bound on number of iterations[1] to achieve

$$\|\nabla f(x_k)\|_2 \leq \epsilon \quad \text{and perhaps} \quad \min(\text{eig}(\nabla^2 f(x_k))) \geq -\epsilon \quad \text{for some} \quad \epsilon > 0$$

[1] ...or function evaluations, subproblem solves, etc.

Theory vs. practice

Researchers have been gravitating to adopt and build on cubic regularization:

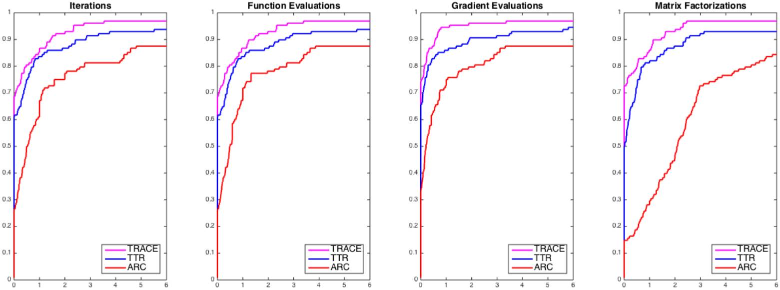
- Agarwal, Allen-Zhu, Bullins, Hazan, Ma (2017)
- Carmon, Duchi (2017)
- Kohler, Lucchi (2017)
- Peng, Roosta-Khorasan, Mahoney (2017)

However, there remains a large gap between theory and practice!

Little evidence that cubic regularization methods offer improved performance:

- Trust region (TR) methods remain the state-of-the-art
- TR-like methods can achieve the same complexity guarantees

Trust region methods with optimal complexity



Closer look

Let's understand these complexity bounds, then look closely at them.

I'll refer to three algorithms:

- TTR: "Traditional" Trust Region algorithm
- ARC: Adaptive Regularisation algorithm using Cubics
  - Cartis, Gould, & Toint (2011)
- TRACE: Trust Region Algorithm with Contractions and Expansions
  - Curtis, Robinson, & Samadi (2017)

Outline

Algorithm basics

TTR
1: Solve to compute s_k:
$\min_{s\in\mathbb{R}^n} \; q_k(s) := f_k + g_k^T s + \frac{1}{2}s^T H_k s$
s.t. $\|s\|_2 \leq \delta_k$ (dual: λ_k)
2: Compute ratio:
$\rho_k^q \leftarrow \frac{f_k - f(x_k+s_k)}{f_k - q_k(s_k)}$
3: Update radius:
$\rho_k^q \geq \eta$: accept and $\delta_k \nearrow$
$\rho_k^q < \eta$: reject and $\delta_k \searrow$

ARC
1: Solve to compute s_k:
$\min_{s\in\mathbb{R}^n} \; c_k(s) := f_k + g_k^T s + \frac{1}{2}s^T H_k s + \frac{1}{3}\sigma_k \|s\|_2^3$
2: Compute ratio:
$\rho_k^c \leftarrow \frac{f_k - f(x_k+s_k)}{f_k - c_k(s_k)}$
3: Update regularization:
$\rho_k^c \geq \eta$: accept and $\sigma_k \searrow$
$\rho_k^c < \eta$: reject and $\sigma_k \nearrow$

Algorithm basics: Subproblem solution correspondence

TTR

1: Solve to compute s_k:

$$\min_{s \in \mathbb{R}^n} \; q_k(s) := f_k + g_k^T s + \tfrac{1}{2} s^T H_k s \quad \text{s.t. } \|s\|_2 \leq \delta_k \;\; (\text{dual: } \lambda_k)$$

2: Compute ratio:

$$\rho_k^q \leftarrow \frac{f_k - f(x_k + s_k)}{f_k - q_k(s_k)}$$

3: Update radius:

$\rho_k^q \geq \eta$: accept and $\delta_k \nearrow$

$\rho_k^q < \eta$: reject and $\delta_k \searrow$

$$\sigma_k = \frac{\lambda_k}{\delta_k} \longrightarrow$$

$$\longleftarrow \delta_k = \|s_k\|_2$$

ARC

1: Solve to compute s_k:

$$\min_{s \in \mathbb{R}^n} \; c_k(s) := f_k + g_k^T s + \tfrac{1}{2} s^T H_k s + \tfrac{1}{3} \sigma_k \|s\|_2^3$$

2: Compute ratio:

$$\rho_k^c \leftarrow \frac{f_k - f(x_k + s_k)}{f_k - c_k(s_k)}$$

3: Update regularization:

$\rho_k^c \geq \eta$: accept and $\sigma_k \searrow$

$\rho_k^c < \eta$: reject and $\sigma_k \nearrow$

Discussion

What are the similarities?

- algorithmic frameworks are almost identical
- one-to-one correspondence (except $\lambda_k = 0$) between subproblem solutions

What are the key differences?

- step acceptance criteria
- trust region vs. regularization coefficient updates

Discussion

What are the similarities?

- algorithmic frameworks are almost identical
- one-to-one correspondence (except $\lambda_k = 0$) between subproblem solutions

What are the key differences?

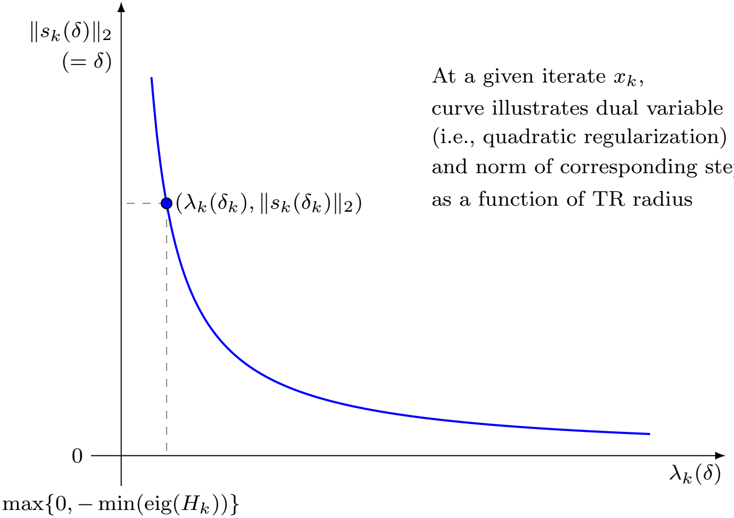
- step acceptance criteria
- trust region vs. regularization coefficient updates

Recall that a solution s_k of the TR subproblem is also a solution of

$$\min_{s \in \mathbb{R}^n} \; f_k + g_k^T s + \tfrac{1}{2} s^T (H_k + \lambda_k I) s,$$

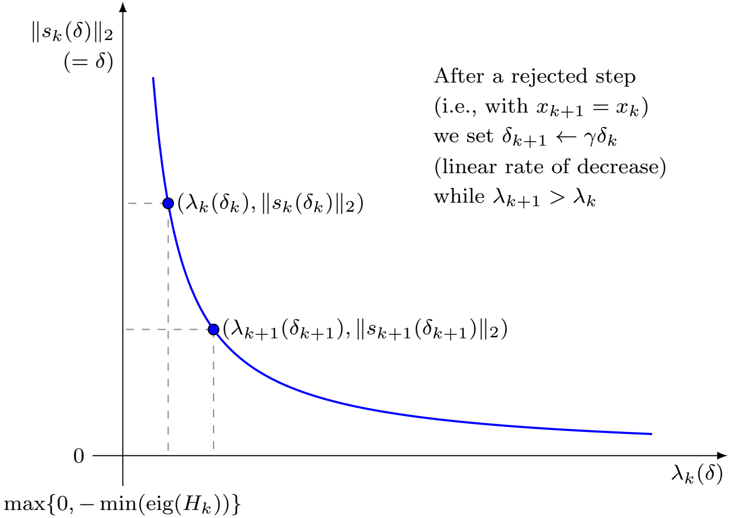
so the dual variable λ_k can be viewed as a quadratic regularization coefficient.

Regularization/stepsize trade-off: TTR



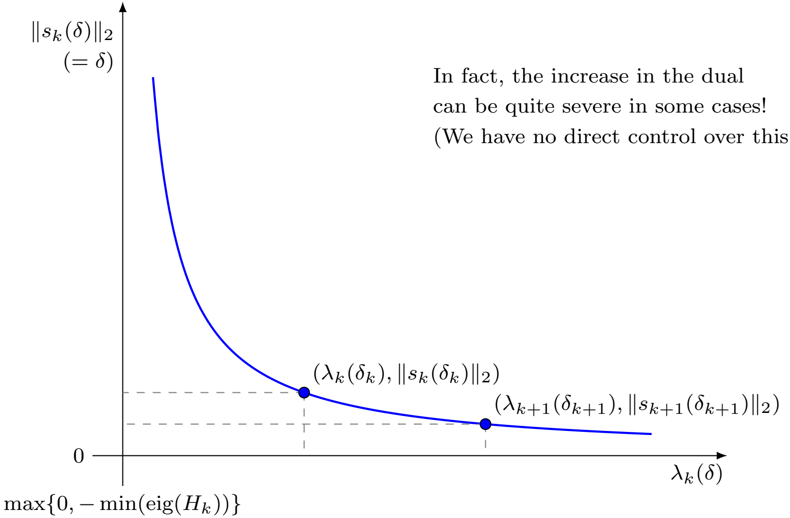
At a given iterate x_k,
curve illustrates dual variable
(i.e., quadratic regularization)
and norm of corresponding step
as a function of TR radius

Regularization/stepsize trade-off: TTR



After a rejected step
(i.e., with $x_{k+1} = x_k$)
we set $\delta_{k+1} \leftarrow \gamma\delta_k$
(linear rate of decrease)
while $\lambda_{k+1} > \lambda_k$

Regularization/stepsize trade-off: TTR



In fact, the increase in the dual
can be quite severe in some cases!
(We have no direct control over this.)

Intuition, please!

Intuitively, what is so important about $\frac{\lambda_k}{\|s_k\|_2} = \frac{\lambda_k}{\delta_k}$?

- Large δ_k implies s_k may not yield objective decrease.
- Small δ_k prohibits long steps.
- Small λ_k suggests the TR is not restricting us too much.
- Large λ_k suggests more objective decrease is possible.

So what is so bad (for complexity's sake) with the following?

$$\frac{\lambda_k}{\delta_k} \approx 0 \quad \text{and} \quad \frac{\lambda_{k+1}}{\delta_{k+1}} \gg 0.$$

It's that we may go from a

- large, but unproductive step to a
- productive, but (too) short step!

ARC magic

So what's the magic of ARC?

- It's not the types of steps you compute (since TR subproblem gives the same).
- It's that a simple update for σ_k gives a good regularization/stepsize balance.

ARC magic

So what's the magic of ARC?

- It's not the types of steps you compute (since TR subproblem gives the same).
- It's that a simple update for σ_k gives a good regularization/stepsize balance.

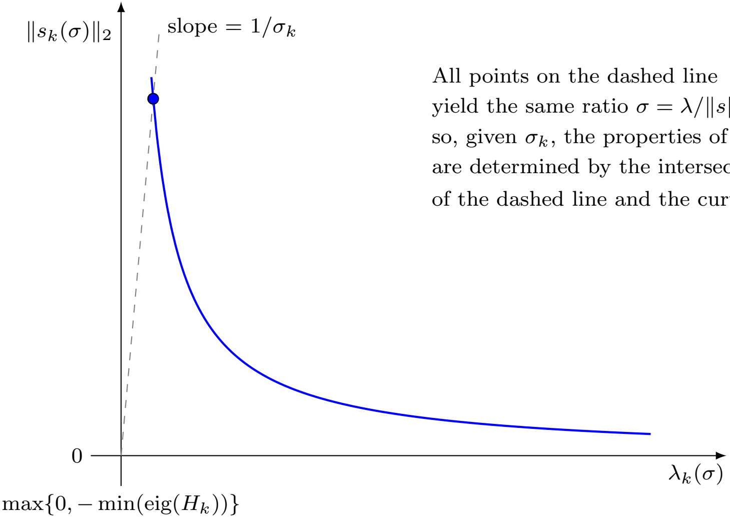
In ARC, restricting $\sigma_k \geq \sigma_{\min}$ for all k and proving that $\sigma_k \leq \sigma_{\max}$ for all k ensures that all accepted steps satisfy

$$f_k - f_{k+1} \geq c_1 \sigma_{\min} \|s_k\|_2^3 \quad \text{and} \quad \|s_k\|_2 \geq \left(\frac{c_2}{\sigma_{\max} + c_3}\right)^2 \|g_{k+1}\|_2^{1/2}.$$

One can also show that, at any point, the number of rejected steps that can occur consecutively is bounded above by a constant (independent of k and ϵ).

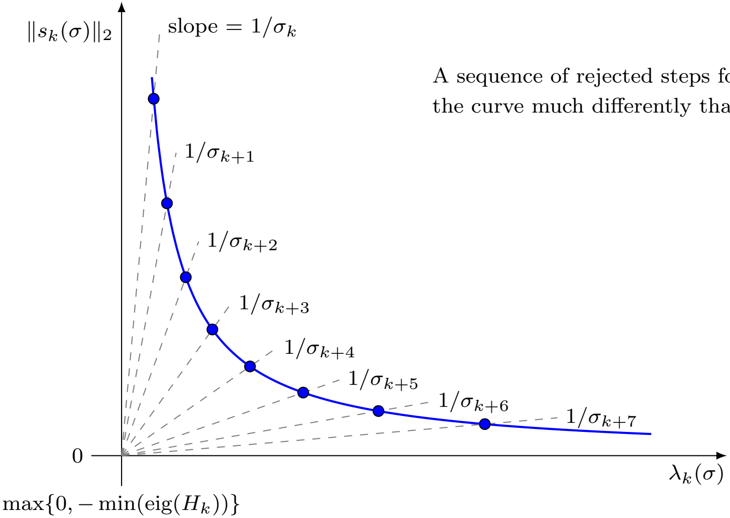
- Important to note that ARC always has the regularization "on."

Regularization/stepsize trade-off: ARC



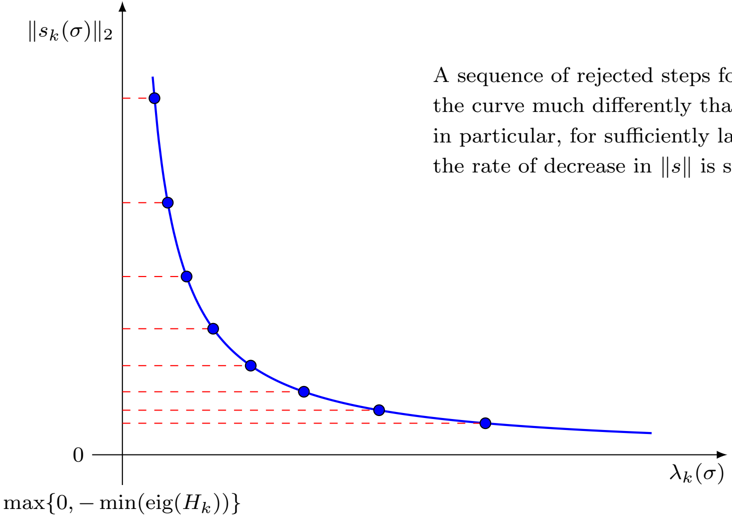
All points on the dashed line yield the same ratio $\sigma = \lambda/\|s\|_2$ so, given σ_k, the properties of s_k are determined by the intersection of the dashed line and the curve

Regularization/stepsize trade-off: ARC



A sequence of rejected steps follow the curve much differently than TTR;

Regularization/stepsize trade-off: ARC



A sequence of rejected steps follow the curve much differently than TTR; in particular, for sufficiently large σ, the rate of decrease in $\|s\|$ is sublinear

From TTR to TRACE

TRACE involves three key modifications of TTR.

1: Different step acceptance ratio

2: New expansion step: May reject step while increasing TR radius

3: New contraction procedure: Explicit or implicit (through update of λ)

With these, we obtain the same complexity properties as ARC.

- We also recently proposed and analyzed a framework that generalizes both
- ...and allows for inexact subproblem solutions (maintaining complexity).

$$\min_{(s,\lambda)\in\mathbb{R}^n\times\mathbb{R}} f_k + g_k^T s + \tfrac{1}{2} s^T (H_k + \lambda I) s$$
$$\text{s.t. } (\sigma_k^L)^2 \|s\|_2^2 \leq \lambda \leq (\sigma_k^U)^2 \|s\|_2^2$$

Outline

Complexity for ARC and TRACE

Suppose that f is twice continuously differentiable...

- ... bounded below by $f_{\min}$
- ... with g and H both Lipschitz continuous (constants L_g and L_H)

Combine the inequalities

$$f_k - f_{k+1} \geq \eta \|s_k\|_2^3 \quad \text{and} \quad \|s_k\|_2 \geq (L_H + \sigma_{\max})^{-1/2} \|g_{k+1}\|_2^{1/2}$$

The cardinality of the set $\{k \in \mathbb{N} : \|g_k\|_2 > \epsilon\}$ is bounded above by

$$\left\lfloor \left(\frac{f_0 - f_{\min}}{\eta (L_H + \sigma_{\max})^{-3/2}} \right) \epsilon^{-3/2} \right\rfloor$$

Complexity for ARC and TRACE

Suppose that f is twice continuously differentiable…

- …bounded below by $f_{\min}$
- …with g and H both Lipschitz continuous (constants L_g and L_H)

Combine the inequalities

$$f_k - f_{k+1} \geq \eta \|s_k\|_2^3 \quad \text{and} \quad \|s_k\|_2 \geq (L_H + \sigma_{\max})^{-1/2} \|g_{k+1}\|_2^{1/2}$$

The cardinality of the set $\{k \in \mathbb{N} : \|g_k\|_2 > \epsilon\}$ is bounded above by

$$\left\lfloor \left(\frac{f_0 - f_{\min}}{\eta (L_H + \sigma_{\max})^{-3/2}} \right) \epsilon^{-3/2} \right\rfloor$$

But these bounds are very pessimistic…

Simpler setting

Consider the iteration

$$x_{k+1} \leftarrow x_k - \tfrac{1}{L} g_k.$$

This type of complexity analysis considers the set

$$\mathcal{G}(\epsilon_g) := \{x \in \mathbb{R}^n : \|g(x)\|_2 \leq \epsilon_g\}$$

and aims to find an upper bound on the cardinality of

$$\mathcal{K}_g(\epsilon_g) := \{k \in \mathbb{N} : x_k \notin \mathcal{G}(\epsilon_g)\}.$$

Upper bound on $|\mathcal{K}_g(\epsilon_g)|$

Using $s_k = -\frac{1}{L_g} g_k$ and the upper bound

$$f_{k+1} \leq f_k + g_k^T s_k + \tfrac{1}{2} L_g \|s_k\|_2^2,$$

one finds

$$\begin{aligned} & f_k - f_{k+1} \geq \tfrac{1}{2L_g} \|g_k\|_2^2 \\ \Longrightarrow \quad & (f_0 - f_*) \geq \tfrac{1}{2L_g} |\mathcal{K}_g(\epsilon_g)| \epsilon_g^2 \\ \Longrightarrow \quad & |\mathcal{K}_g(\epsilon_g)| \leq 2L_g (f_0 - f_*) \epsilon_g^{-2}. \end{aligned}$$

“Nice” f

But what if f is “nice”?

…e.g., satisfying the Polyak-Lojasiewicz condition (for $c \in \mathbb{R}_{++}$), i.e.,

$$f(x) - f_* \leq \tfrac{1}{2c}\|g(x)\|_2^2 \;\text{ for all }\; x \in \mathbb{R}^n.$$

Now consider the set

$$\mathcal{F}(\epsilon_f) := \{x \in \mathbb{R}^n : f(x) - f_* \leq \epsilon_f\}$$

and consider an upper bound on the cardinality of

$$\mathcal{K}_f(\epsilon_f) := \{k \in \mathbb{N} : x_k \notin \mathcal{F}(\epsilon_f)\}.$$

Upper bound on $|\mathcal{K}_f(\epsilon_f)|$

Using $s_k = -\frac{1}{L_g} g_k$ and the upper bound

$$f_{k+1} \leq f_k + g_k^T s_k + \tfrac{1}{2} L_g \|s_k\|_2^2,$$

one finds

$$\begin{aligned}
f_k - f_{k+1} &\geq \tfrac{1}{2L_g} \|g_k\|_2^2 \\
&\geq \tfrac{c}{L_g}(f_k - f_*)
\end{aligned}$$

$$\Longrightarrow \quad (1 - \tfrac{c}{L_g})(f_k - f_*) \geq f_{k+1} - f_*$$

$$\Longrightarrow \quad (1 - \tfrac{c}{L_g})^k (f_0 - f_*) \geq f_k - f_*$$

$$\Longrightarrow \quad |\mathcal{K}_f(\epsilon_f)| \leq \log\left(\frac{f_0 - f_*}{\epsilon_f}\right)\left(\log\left(\frac{L_g}{L_g - c}\right)\right)^{-1}.$$

For the first step...

In the "general nonconvex" analysis...

...the expected decrease for the first step is much more pessimistic:

$$\text{general nonconvex:} \qquad f_0 - f_1 \geq \frac{1}{2L_g}\epsilon_g$$

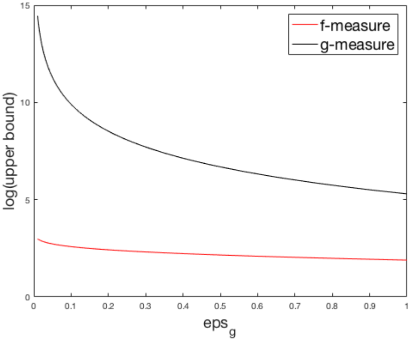
$$\text{PL condition:} \quad (1 - \tfrac{c}{L_g})(f_0 - f_*) \geq f_1 - f_*$$

...and it remains more pessimistic throughout!

Upper bounds on $|\mathcal{K}_f(\epsilon_f)|$ versus $|\mathcal{K}_g(\epsilon_g)|$

Let $f(x) = \frac{1}{2}x^2$, meaning that $g(x) = x$.

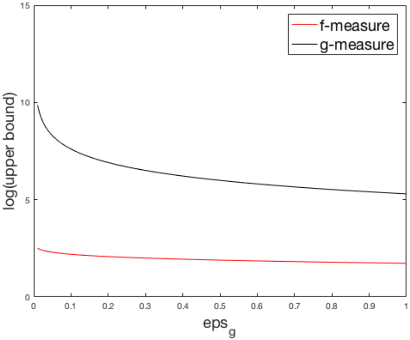
- Let $\epsilon_f = \frac{1}{2}\epsilon_g^2$, meaning that $\mathcal{F}(\epsilon_f) = \mathcal{G}(\epsilon_g)$.
- Let $x_0 = 10$, $c = 1$, and $L = 2$. (Similar pictures for any $L > 1$.)



Upper bounds on $|\mathcal{K}_f(\epsilon_f)|$ versus $|\{k \in \mathbb{N} : \frac{1}{2}\|g_k\|_2^2 > \epsilon_g\}|$

Let $f(x) = \frac{1}{2}x^2$, meaning that $\frac{1}{2}g(x)^2 = \frac{1}{2}x^2$.

- Let $\epsilon_f = \epsilon_g$, meaning that $\mathcal{F}(\epsilon_f) = \mathcal{G}(\epsilon_g)$.
- Let $x_0 = 10$, $c = 1$, and $L = 2$. (Similar pictures for any $L > 1$.)



Bad worst-case!

Worst-case complexity bounds in the general nonconvex case are very pessimistic.

- The analysis immediately admits a large gap when the function is nice.
- The "essentially tight" examples for the worst-case bounds are... weird.[2]

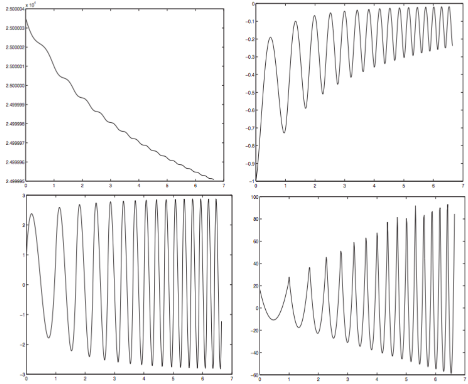


FIG. 2.1. *The function* $f^{(1)}$ *(top left) and its derivatives of order one (top right), two (bottom left), and three (bottom right) on the first* 16 *intervals.*

[2] Cartis, Gould, Toint (2010)

Outline

Summary

We have seen that cubic regularization admits good complexity properties.

- But we have also seen that tweaking trust region methods yields the same.
- And the empirical performance of trust region remains better (I claim!).

These complexity bounds are extremely pessimistic.

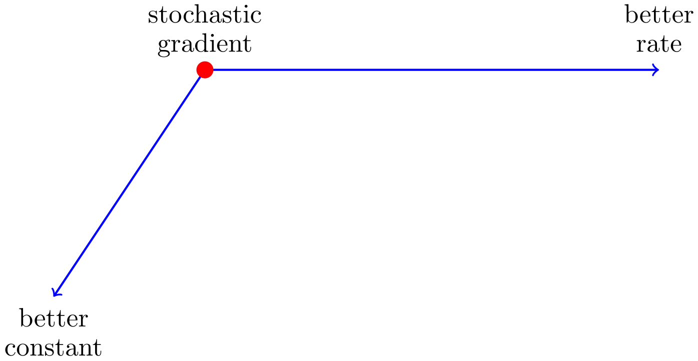
- If your function is "nice", these bounds are way off.
- How can we close the gap between theory and practice?
- Nesterov & Polyak (2006) consider different "phases"...
- ...but can this be generalized (for non-strongly convex)?

Complexity properties should probably not (yet) drive algorithm selection.

- Then why second-order?
- Mitigate effects of ill-conditioning, easier to tune, parallel and distributed, etc.

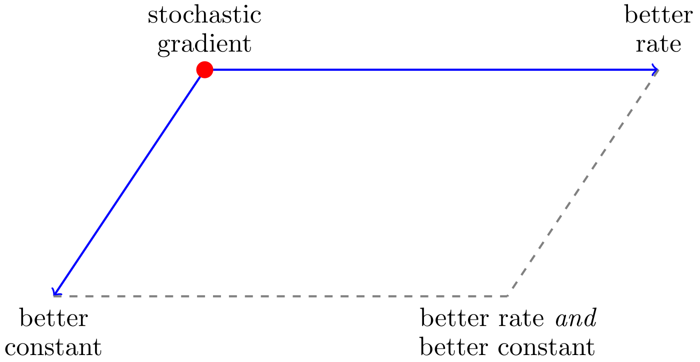
Going back to machine learning (ML) connection...

Optimization for ML: Beyond SG



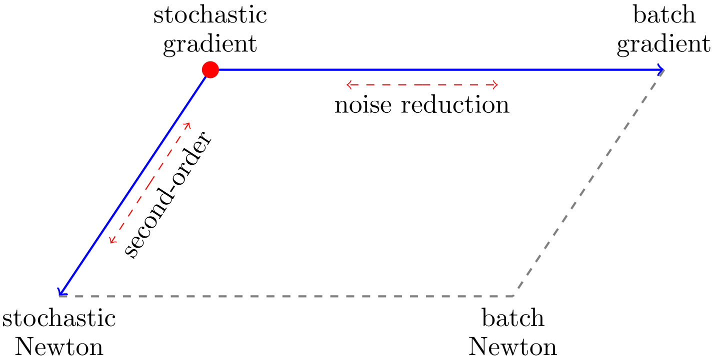
Bottou et al., Optimization Methods for Large-Scale Machine Learning, *SIAM Review* (to appear)

Optimization for ML: Beyond SG



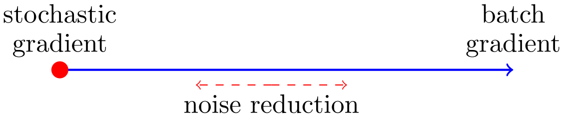
Bottou et al., Optimization Methods for Large-Scale Machine Learning, *SIAM Review* (to appear)

Two-dimensional schematic of methods



Bottou et al., Optimization Methods for Large-Scale Machine Learning, *SIAM Review* (to appear)

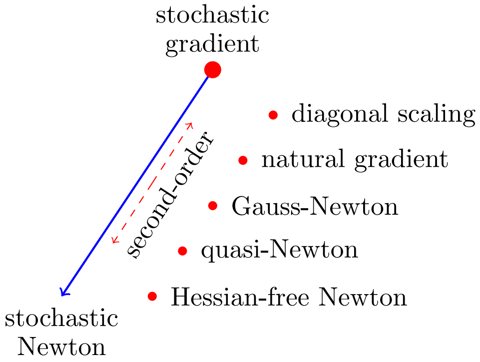
2D schematic: Noise reduction methods



- dynamic sampling
- gradient aggregation
- iterate averaging

Bottou et al., Optimization Methods for Large-Scale Machine Learning, *SIAM Review* (to appear)

2D schematic: Second-order methods



Bottou et al., Optimization Methods for Large-Scale Machine Learning, *SIAM Review* (to appear)

Fairly comparing algorithms

How should we compare optimization algorithms for machine learning?

- Fair comparison would
- . . . not ignore time spent tuning parameters
- . . . demonstrate speed and reliability
- . . . involve many problems (test sets?)
- . . . involve many runs of each algorithm

What about testing accuracy?