Characterizing the Worst-Case Performance of Algorithms for Nonconvex Optimization

Frank E. Curtis, Lehigh University

joint work with

Daniel P. Robinson, Johns Hopkins University

presented at

Rutgers University

29 March 2018

Outline

Outline

Motivation

Contemporary Analyses

Partitioning the Search Space

Behavior of Common Methods

Summary & Perspectives

Problem

Consider the problem to minimize an objective function $f : \mathbb{R}^n \to \mathbb{R}$:

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

Various iterative algorithms have been proposed of the form

$$x_{k+1} \leftarrow x_k + s_k \;\text{ for all }\; k \in \mathbb{N},$$

where $\{x_k\}$ is the iterate sequence and $\{s_k\}$ is the step sequence.

Problem

Consider the problem to minimize an objective function $f : \mathbb{R}^n \to \mathbb{R}$:

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

Various iterative algorithms have been proposed of the form

$$x_{k+1} \leftarrow x_k + s_k \;\; \text{for all} \;\; k \in \mathbb{N},$$

where $\{x_k\}$ is the iterate sequence and $\{s_k\}$ is the step sequence.

Question: What is the best algorithm for solving *my* problem?

History

Nonlinear optimization algorithm design has had parallel developments:

convexity

Rockafellar
Fenchel
Nemirovski
Nesterov

subgradient inequality

convergence, complexity guarantees

smoothness

Powell
Fletcher
Goldfarb
Nocedal

sufficient decrease

convergence, fast local convergence

Worlds are finally colliding!

Worst-case complexity for convex optimization

Worst-case complexity: Upper limit on the resources an algorithm will require to (approximately) solve a given problem

Worst-case complexity for convex optimization

Worst-case complexity: Upper limit on the resources an algorithm will require to (approximately) solve a given problem

. . . for convex optimization: Bound on the number of iterations (or function or derivative evaluations) until

$$\begin{aligned} &\|x_k - x_*\| \leq \epsilon_x \\ \text{or} \;\; &f(x_k) - f(x_*) \leq \epsilon_f, \end{aligned}$$

where x_* is some global minimizer of f.

Worst-case complexity for convex optimization

Worst-case complexity: Upper limit on the resources an algorithm will require to (approximately) solve a given problem

. . . for convex optimization: Bound on the number of iterations (or function or derivative evaluations) until

$$\begin{aligned} \|x_k - x_*\| &\leq \epsilon_x \\ \text{or} \;\; f(x_k) - f(x_*) &\leq \epsilon_f, \end{aligned}$$

where x_* is some global minimizer of f.

Fact(?): Convex setting: better complexity *often* implies better performance.

(Really, need to consider work complexity, conditioning, structure, etc.)

Worst-case complexity for nonconvex optimization

. . . for *non*convex optimization: Here is how we do it now:

Since one generally cannot guarantee that $\{x_k\}$ converges to a minimizer, one asks for an upper bound on the number of iterations until

$$\begin{aligned} & \|\nabla f(x_k)\| \leq \epsilon_g && \text{(first-order stationarity)} \\ \text{and maybe also } & \lambda(\nabla^2 f(x_k)) \geq -\epsilon_H && \text{(second-order stationarity)} \end{aligned}$$

Worst-case complexity for nonconvex optimization

. . . for *non*convex optimization: Here is how we do it now:

Since one generally cannot guarantee that $\{x_k\}$ converges to a minimizer, one asks for an upper bound on the number of iterations until

$$\begin{aligned} & \|\nabla f(x_k)\| \leq \epsilon_g & \text{(first-order stationarity)} \\ \text{and maybe also} \quad & \lambda(\nabla^2 f(x_k)) \geq -\epsilon_H & \text{(second-order stationarity)} \end{aligned}$$

For example, it is known that for first-order stationarity we have the bounds. . .

Algorithm	until $\|\nabla f(x_k)\|_2 \leq \epsilon_g$
Gradient descent	$\mathcal{O}(\epsilon_g^{-2})$
Newton / trust region	$\mathcal{O}(\epsilon_g^{-2})$
Cubic regularization	$\mathcal{O}(\epsilon_g^{-3/2})$

Self-examination

But...

- Is this the best way to ***characterize*** our algorithms?
- Is this the best way to ***represent*** our algorithms?

Self-examination

But...

- Is this the best way to ***characterize*** our algorithms?
- Is this the best way to ***represent*** our algorithms?

People listen! Cubic regularization...

- Griewank (1981)
- Nesterov & Polyak (2006)
- Weiser, Deuflhard, Erdmann (2007)
- Cartis, Gould, Toint (2011), the `ARC` method

...is a framework to which researchers have been attracted...

- 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!

(Trust region methods arguably perform better in general.)

Example: Matrix factorization

Symmetric low-rank matrix factorization problem:

$$\min_{X \in \mathbb{R}^{d \times r}} \ \tfrac{1}{2}\|XX^T - M\|_F^2,$$

where $M \in \mathbb{R}^{d \times d}$ with $\text{rank}(M) = r$.

- Nonconvex, but...
- Global minimum value is known (it's zero)
- All local minima are global minima

Jin, Ge, Netrapalli, Kakade, Jordan (2017)

Example: Dictionary learning

Learning a representation of input data in the form of linear combinations of some (unknown) basic elements, called *atoms*, which compose a *dictionary*:

$$\min_{X \in \mathcal{X}, Y \in \mathbb{R}^{n \times n}} \|Z - XY\|^2 + \phi(Y)$$
$$\text{s.t. } \mathcal{X} := \{X \in \mathbb{R}^{d \times n} : \|X_i\|_2 \leq 1 \text{ for all } i \in \{1, \dots, n\}\},$$

where $Z \in \mathbb{R}^{d \times n}$ is a given input.

Nonconvex, but, under some conditions, all saddle points can be “escaped”.

Example: Dictionary learning

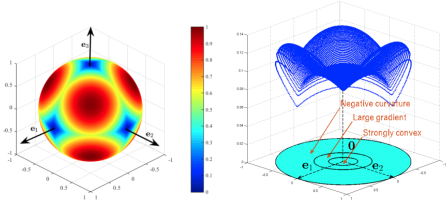
Learning a representation of input data in the form of linear combinations of some (unknown) basic elements, called *atoms*, which compose a *dictionary*:

$$\min_{X \in \mathcal{X}, Y \in \mathbb{R}^{n \times n}} \|Z - XY\|^2 + \phi(Y)$$

$$\text{s.t. } \mathcal{X} := \{X \in \mathbb{R}^{d \times n} : \|X_i\|_2 \leq 1 \text{ for all } i \in \{1, \dots, n\}\},$$

where $Z \in \mathbb{R}^{d \times n}$ is a given input.

Nonconvex, but, under some conditions, all saddle points can be "escaped".



Sun, Qu, Wright (2016)

Other examples

- Phase retrieval
- Orthogonal tensor decomposition
- Deep linear learning
- ...

Pedogogical example

But if we're talking about nonconvex optimization, we also could have. . .

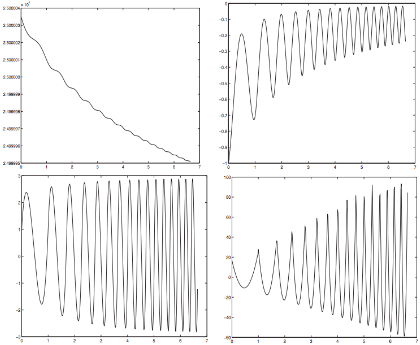


FIG. 2.1. *The function* $f^{(1)}$ *(top left) and its derivatives of order one (top right), two (bottom left), and three (bottom right) on the first* 16 *intervals.*

What *real* problem exhibits this behavior? (I don't know!)

More on this example later. . .

Purpose of this talk

Our goal: A ***complementary*** approach to characterize algorithms.

- global convergence
- worst-case complexity, contemporary type + our approach
- local convergence rate

Purpose of this talk

Our goal: A ***complementary*** approach to characterize algorithms.

- global convergence
- worst-case complexity, contemporary type + our approach
- local convergence rate

We're admitting: Our approach ***does not always*** give the complete picture.

But we believe it ***is*** useful.

Outline

Simple setting

Suppose the gradient $g := \nabla f$ is Lipschitz continuous with constant $L > 0$.

Consider the iteration

$$x_{k+1} \leftarrow x_k - \tfrac{1}{L} g_k \ \text{ for all } \ k \in \mathbb{N}.$$

A contemporary 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

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

$g_k := \nabla f(x_k)$

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

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

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

one finds with $f_{\inf} := \inf_{x \in \mathbb{R}^n} f(x)$ that

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

“Nice” f

But what if f is “nice”?

e.g., satisfying the Polyak-Łojasiewicz condition for $c \in (0, \infty)$, i.e.,

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

Now consider the set

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

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

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

one finds that

$$\begin{aligned}
& & f_k - f_{k+1} &\geq \tfrac{1}{2L} \|g_k\|_2^2 \\
& & &\geq \tfrac{c}{L}(f_k - f_{\inf}) \\
\implies & & (1 - \tfrac{c}{L})(f_k - f_{\inf}) &\geq f_{k+1} - f_{\inf} \\
\implies & & (1 - \tfrac{c}{L})^k (f_0 - f_{\inf}) &\geq f_k - f_{\inf} \\
\implies & & |\mathcal{K}_f(\epsilon_f)| &\leq \log\left(\frac{f_0 - f_{\inf}}{\epsilon_f}\right)\left(\log\left(\frac{L}{L-c}\right)\right)^{-1}.
\end{aligned}$$

For the first step. . .

In the "general nonconvex" analysis. . .

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

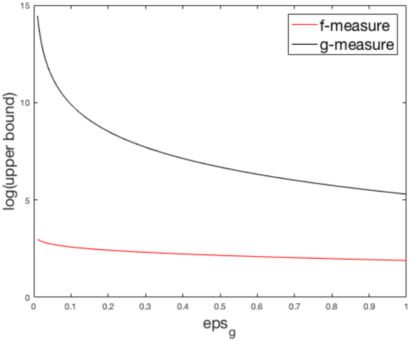
$$\begin{aligned} \text{general nonconvex:} \quad & f_0 - f_1 \geq \tfrac{1}{2L}\epsilon_g^2 \\ \text{PL condition:} \quad & (1 - \tfrac{c}{L})(f_0 - f_{\inf}) \geq f_1 - f_{\inf} \end{aligned}$$

. . . 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 $g(x) = x$.

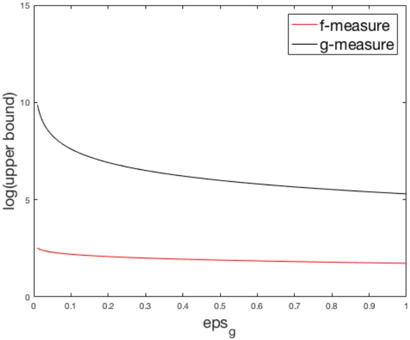
- Let $\epsilon_f = \frac{1}{2}\epsilon_g^2$, meaning that $\mathcal{F}(\epsilon_f) = \mathcal{G}(\epsilon_g)$.
- Let $x_0 = 10$, $c = 1$, and $L = 2$. (Similar pictures for any $L > 1$.)



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.[1]

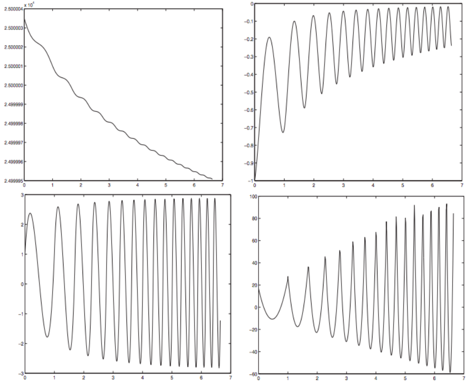


FIG. 2.1. *The function* $f^{(1)}$ *(top left) and its derivatives of order one (top right), two (bottom left), and three (bottom right) on the first* 16 *intervals.*

[1] Cartis, Gould, Toint (2010)

Plea

My plea to optimizers:

Let's not have these be the problems that dictate how we

- characterize our algorithms and
- represent our algorithms to the world!

Outline

Motivation

We want a characterization strategy that

- attempts to capture behavior in *actual practice*
- i.e., is not “bogged down” by pedogogical examples
- can be applied consistently across different classes of functions
- shows more than just the worst of the worst case

Motivation

We want a characterization strategy that

- attempts to capture behavior in *actual practice*
- i.e., is not “bogged down” by pedogogical examples
- can be applied consistently across different classes of functions
- shows more than just the worst of the worst case

Our idea is to

- partition the search space (dependent on f and x_0)
- analyze how an algorithm behaves over different regions
- characterize an algorithm's behavior *by region*

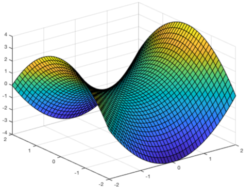
For some functions, there will be holes, but for some of interest there are none!

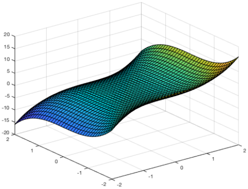
Intuition

Think about an arbitrary point in the search space, i.e.,

$$\mathcal{L} := \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}.$$

- If $\|g(x)\|_2 \gg 0$, then "a lot" of progress can be made.
- If $\min(\text{eig}(\nabla^2 f(x))) \ll 0$, then "a lot" of progress can also be made.





Assumption

Assumption 1

- *f is $\bar{p}$-times continuously differentiable*
- *f is bounded below by $f_{inf} := \inf_{x \in \mathbb{R}^n} f(x)$*
- *for all $p \in \{1, \ldots, \bar{p}\}$, there exists $L_p \in (0, \infty)$ such that*

$$f(x+s) \leq \underbrace{f(x) + \sum_{j=1}^{p} \frac{1}{j!} \nabla^j f(x)[s]^j}_{t_p(x,s)} + \frac{L_p}{p+1} \|s\|_2^{p+1}$$

pth-order term reduction

Definition 2

For each $p \in \{1, \ldots, \bar{p}\}$, define the function

$$m_p(x,s) = \frac{1}{p!}\nabla^p f(x)[s]^p + \frac{r_p}{p+1}\|s\|_2^{p+1}.$$

Letting $s_{m_p}(x) := \arg\min_{s \in \mathbb{R}^n}$, the *reduction in the pth-order term from x* is

$$\Delta m_p(x) = m_p(x,0) - m_p(x, s_{m_p}(x)) \geq 0.$$

*Exact definition of r_p is not complicated, but we'll skip it here

1st-order and 2nd-order term reductions

Theorem 3

For $\bar{p} \geq 2$, the following hold:

$$\Delta m_1(x) = \frac{1}{2r_1}\|\nabla f(x)\|_2^2$$

$$\text{and} \quad \Delta m_2(x) = \frac{1}{6r_2^2}\max\{-\lambda(\nabla^2 f(x_k)), 0\}^3.$$

Regions

We propose to partition the search space, given $(\kappa, f_{\text{ref}}) \in (0,1) \times [f_{\inf}, f(x_0)]$, into

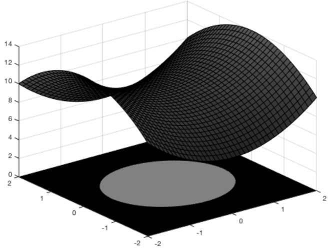
$$\mathcal{R}_1 := \{x \in \mathcal{L} : \Delta m_1(x) \geq \kappa(f(x) - f_{\text{ref}})\},$$

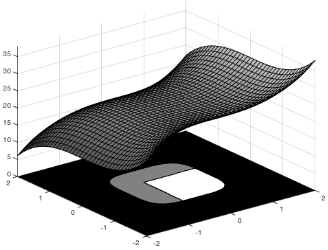
$$\mathcal{R}_p := \{x \in \mathcal{L} : \Delta m_p(x) \geq \kappa(f(x) - f_{\text{ref}})\} \setminus \left(\bigcup_{j=1}^{p-1} \mathcal{R}_j\right) \quad \text{for all} \quad p \in \{2, \dots, \bar{p}\},$$

and $\overline{\mathcal{R}} := \mathcal{L} \setminus \left(\bigcup_{j=1}^{\bar{p}} \mathcal{R}_j\right)$.

*We don't need $f_{\text{ref}} = f_{\inf}$, but, for simplicity, think of it that way here

Illustration





$(\bar{p} = 2)$ $\mathcal{R}_1$: black $\mathcal{R}_2$: gray $\overline{\mathcal{R}}$: white

Functions satisfying Polyak-Łojasiewicz

Theorem 4

A continuously differentiable f with a Lipschitz continuous gradient satisfies the Polyak-Łojasiewicz condition if and only if $\mathcal{R}_1 = \mathcal{L}$ for any $x_0 \in \mathbb{R}^n$.

Hence, if we prove something about the behavior of an algorithm over $\mathcal{R}_1$, then

- we know how it behaves if f satisfies PL *and*
- we know how it behaves *at any point* satisfying the PL inequality.

Functions satisfying a strict-saddle-type property

Theorem 5

If f is twice-continuously differentiable with Lipschitz continuous gradient and Hessian functions such that, at all $x \in \mathcal{L}$ and for some $\zeta \in (0, \infty)$, one has

$$\max\{\|\nabla f(x)\|_2^2, -\lambda_{\min}(\nabla^2 f(x))^3\} \geq \zeta(f(x) - f_{inf}),$$

then $\mathcal{R}_1 \cup \mathcal{R}_2 = \mathcal{L}$.

Outline

Linearly convergent behavior over $\mathcal{R}_p$

Let $s_{w_p}(x)$ be a minimum norm global minimizer of the regularized Taylor model

$$w_p(x,s) = t_p(x,s) + \frac{l_p}{p+1}\|s\|_2^{p+1}$$

Theorem 6

If $\{x_k\}$ is generated by the iteration

$$x_{k+1} \leftarrow x_k + s_{w_p}(x),$$

then, with $\epsilon_f \in (0, f(x_0) - f_{ref})$, the number of iterations in

$$\mathcal{R}_p \cap \{x \in \mathbb{R}^n : f(x) - f_{ref} \geq \epsilon_f\}$$

is bounded above by

$$\left\lceil \log\left(\frac{f(x_0) - f_{ref}}{\epsilon_f}\right)\left(\log\left(\frac{1}{1-\kappa}\right)\right)^{-1}\right\rceil = \mathcal{O}\left(\log\left(\frac{f(x_0) - f_{ref}}{\epsilon_f}\right)\right)$$

Regularized gradient and Newton methods

- Regularized gradient method: Computes s_k by solving

$$\min_{s \in \mathbb{R}^n} \; f(x_k) + \nabla f(x_k)^T s + \frac{l_1}{2}\|s\|_2^2 \quad \Longrightarrow \quad s_k = -\frac{1}{l_1}\nabla f(x_k)$$

- Regularized Newton method: Computes s_k by solving

$$\min_{s \in \mathbb{R}^n} \; f(x_k) + \nabla f(x_k)^T s + \frac{1}{2}s^T \nabla^2 f(x_k) s + \frac{l_2}{3}\|s\|_2^3,$$

also known as cubic regularization (mentioned earlier)

Characterization: Contemporary

Let RG and RN represent regularized gradient and Newton, respectively.

Theorem 7

With $\bar{p} \geq 2$, let

$$\mathcal{K}_1(\epsilon_g) := \{k \in \mathbb{N} : \|\nabla f(x_k)\|_2 > \epsilon_g\}$$
$$\textit{and} \;\; \mathcal{K}_2(\epsilon_H) := \{k \in \mathbb{N} : \lambda_{\min}(\nabla^2 f(x_k)) < -\epsilon_H\}.$$

Then, the cardinalities of $\mathcal{K}_1(\epsilon_g)$ and $\mathcal{K}_2(\epsilon_H)$ are of the order...

Algorithm	$\lvert\mathcal{K}_1(\epsilon_g)\rvert$	$\lvert\mathcal{K}_2(\epsilon_H)\rvert$
RG	$\mathcal{O}\left(\frac{l_1(f(x_0)-f_{inf})}{\epsilon_g^2}\right)$	∞
RN	$\mathcal{O}\left(\frac{l_2^{1/2}(f(x_0)-f_{inf})}{\epsilon_g^{3/2}}\right)$	$\mathcal{O}\left(\frac{l_2^2(f(x_0)-f_{inf})}{\epsilon_H^3}\right)$

Characterization: Our approach

Theorem 8

The numbers of iterations in $\mathcal{R}_1$ and $\mathcal{R}_2$ with $f_{ref} = f_{inf}$ are of the order...

Algorithm	$\mathcal{R}_1$	$\mathcal{R}_2$
RG	$\mathcal{O}\left(\log\left(\frac{f(x_0)-f_{inf}}{\epsilon_f}\right)\right)$	∞
RN	$\mathcal{O}\left(\frac{l_2^2(f(x_0)-f_{inf})}{r_1^3}\right) + \mathcal{O}\left(\log\left(\frac{f(x_0)-f_{inf}}{\epsilon_f}\right)\right)$	$\mathcal{O}\left(\log\left(\frac{f(x_0)-f_{inf}}{\epsilon_f}\right)\right)$

There is an initial phase, as seen in Nesterov & Polyak (2006)

Characterization: Our approach

Theorem 8

The numbers of iterations in $\mathcal{R}_1$ and $\mathcal{R}_2$ with $f_{ref} = f_{inf}$ are of the order...

Algorithm	$\mathcal{R}_1$	$\mathcal{R}_2$
RG	$\mathcal{O}\left(\log\left(\frac{f(x_0)-f_{inf}}{\epsilon_f}\right)\right)$	∞
RN	$\mathcal{O}\left(\frac{l_2^2(f(x_0)-f_{inf})}{r_1^3}\right) + \mathcal{O}\left(\log\left(\frac{f(x_0)-f_{inf}}{\epsilon_f}\right)\right)$	$\mathcal{O}\left(\log\left(\frac{f(x_0)-f_{inf}}{\epsilon_f}\right)\right)$

There is an initial phase, as seen in Nesterov & Polyak (2006)

A ∞ can appear, but one could consider probabilistic bounds, too

Trust region method: Gradient-dependent radii

$$\min_{s\in\mathbb{R}^n} \; f(x_k) + \nabla f(x_k)^T s + \tfrac{1}{2} s^T \nabla^2 f(x_k) s \quad \text{s.t.} \quad \|s\|_2 \leq \delta_k$$

- Set $\delta_k \leftarrow \nu_k \|\nabla f(x_k)\|_2$
- Initialize $\nu_0 \in [\underline{\nu}, \overline{\nu}]$
- For some $(\eta, \beta) \in (0,1) \times (0,1)$, if

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{t_2(x_k, 0) - t_2(x_k, s_k)} \geq \eta,$$

then $x_{k+1} \leftarrow x_k + s_k$ and $\nu_{k+1} \in [\underline{\nu}, \overline{\nu}]$; else, $x_{k+1} \leftarrow x_k$ and $\nu_{k+1} \leftarrow \beta \nu_k$.

Trust region method: Gradient-dependent radii

$$\min_{s\in\mathbb{R}^n} f(x_k) + \nabla f(x_k)^T s + \tfrac{1}{2} s^T \nabla^2 f(x_k) s \quad \text{s.t.} \quad \|s\|_2 \leq \delta_k$$

- Set $\delta_k \leftarrow \nu_k \|\nabla f(x_k)\|_2$
- Initialize $\nu_0 \in [\underline{\nu}, \overline{\nu}]$
- For some $(\eta, \beta) \in (0,1) \times (0,1)$, if

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{t_2(x_k, 0) - t_2(x_k, s_k)} \geq \eta,$$

then $x_{k+1} \leftarrow x_k + s_k$ and $\nu_{k+1} \in [\underline{\nu}, \overline{\nu}]$; else, $x_{k+1} \leftarrow x_k$ and $\nu_{k+1} \leftarrow \beta\nu_k$.

Theorem 9

# of iterations in $\mathcal{R}_1$ is at most $\mathcal{O}\left(\chi \log\left(\frac{f(x_0) - f_{ref}}{\epsilon_f}\right)\right)$. For $\mathcal{R}_2$, no guarantee.

Trust region method: Gradient- and Hessian-dependent radii

$$\min_{s\in\mathbb{R}^n} \; f(x_k) + \nabla f(x_k)^T s + \tfrac{1}{2} s^T \nabla^2 f(x_k) s \quad \text{s.t.} \quad \|s\|_2 \leq \delta_k$$

- Set
$$\delta_k \leftarrow \nu_k \begin{cases} \|\nabla f(x_k)\|_2 & \|\nabla f(x_k)\|_2^2 \geq |\lambda(\nabla^2 f(x_k))|^3 \\ |\lambda(\nabla^2 f(x_k))| & \text{otherwise} \end{cases}$$
- Initialize $\nu_0 \in [\underline{\nu}, \overline{\nu}]$
- For some $(\eta, \beta) \in (0,1) \times (0,1)$, if
$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{t_2(x_k, 0) - t_2(x_k, s_k)} \geq \eta,$$
then $x_{k+1} \leftarrow x_k + s_k$ and $\nu_{k+1} \in [\underline{\nu}, \overline{\nu}]$; else, $x_{k+1} \leftarrow x_k$ and $\nu_{k+1} \leftarrow \beta \nu_k$.

Trust region method: Gradient- and Hessian-dependent radii

$$\min_{s\in\mathbb{R}^n} \; f(x_k) + \nabla f(x_k)^T s + \tfrac{1}{2} s^T \nabla^2 f(x_k) s \quad \text{s.t.} \quad \|s\|_2 \leq \delta_k$$

- Set

$$\delta_k \leftarrow \nu_k \begin{cases} \|\nabla f(x_k)\|_2 & \|\nabla f(x_k)\|_2^2 \geq |\lambda(\nabla^2 f(x_k))|^3 \\ |\lambda(\nabla^2 f(x_k))| & \text{otherwise} \end{cases}$$

- Initialize $\nu_0 \in [\underline{\nu}, \overline{\nu}]$
- For some $(\eta, \beta) \in (0,1) \times (0,1)$, if

$$\rho_k = \frac{f(x_k) - f(x_k + s_k)}{t_2(x_k, 0) - t_2(x_k, s_k)} \geq \eta,$$

then $x_{k+1} \leftarrow x_k + s_k$ and $\nu_{k+1} \in [\underline{\nu}, \overline{\nu}]$; else, $x_{k+1} \leftarrow x_k$ and $\nu_{k+1} \leftarrow \beta \nu_k$.

Theorem 10

# of iterations in $\mathcal{R}_1$ is at most $\mathcal{O}\left(\chi \log\left(\frac{f(x_0) - f_{ref}}{\epsilon_f}\right)\right)$.

# of iterations in $\mathcal{R}_2$ is at most $\mathcal{O}\left(\chi_2 \log\left(\frac{f(x_0) - f_{ref}}{\epsilon_f}\right)\right)$.

Trust region method: Always good?

What about the classical update?

$$\delta_{k+1} \leftarrow \begin{cases} \geq \delta_k & \text{if } \rho_k \geq \eta \\ < \delta_k & \text{otherwise.} \end{cases}$$

Two challenges:

- Proving a uniform upper bound on number of consecutive rejected steps
- Proving that accepted steps yield sufficient decrease in $\mathcal{R}_1$ and $\mathcal{R}_2$

Outline

Summary & Perspectives

Our goal: A ***complementary*** approach to characterize algorithms.

- global convergence
- worst-case complexity, contemporary type + our approach
- local convergence rate

Our idea is to

- partition the search space (dependent on f and x_0)
- analyze how an algorithm behaves over different regions
- characterize an algorithm's behavior *by region*

For some functions, there are holes, but for others the characterization is complete.

F. E. Curtis and D. P. Robinson, "How to Characterize the Worst-Case Performance of Algorithms for Nonconvex Optimization," Lehigh ISE/COR@L Technical Report 18T-003, submitted for publication, 2018.