Convex Optimization

Mark Schmidt - CMPT 419/726

Motivation: Why Learn about Convex Optimization?

Why learn about optimization?

- Optimization is at the core of many ML algorithms.
- ML is driving a lot of modern research in optimization.

Motivation: Why Learn about Convex Optimization?

Why learn about optimization?

- Optimization is at the core of many ML algorithms.
- ML is driving a lot of modern research in optimization.

Why in particular learn about convex optimization?

- Among only *efficiently-solvable* continuous problems.
- You can do a lot with convex models.

  (least squares, lasso, generlized linear models, SVMs, CRFs)
- Empirically effective non-convex methods are often based methods with good properties for convex objectives.

  (functions are locally convex around minimizers)

Outline

Convexity: Zero-order condition

A real-valued function is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),$$

for all $x, y \in \mathbb{R}^n$ *and all* $0 \leq \theta \leq 1$.

- Function is *below a linear interpolation* from x to y.
- Implies that all local minima are global minima.

(contradiction otherwise)

Convexity: Zero-order condition

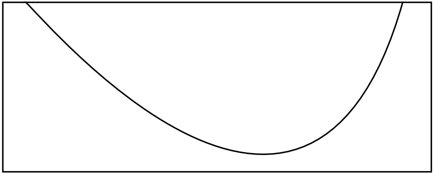
A real-valued function is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),$$

for all $x, y \in \mathbb{R}^n$ *and all* $0 \leq \theta \leq 1$.

- Function is *below a linear interpolation* from x to y.
- Implies that all local minima are global minima.

(contradiction otherwise)



Convexity: Zero-order condition

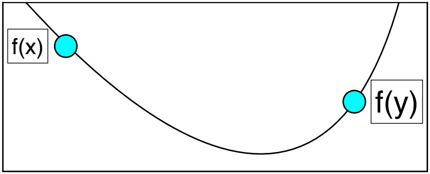
A real-valued function is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),$$

for all x, $y \in \mathbb{R}^n$ *and all* $0 \leq \theta \leq 1$.

- Function is *below a linear interpolation* from x to y.
- Implies that all local minima are global minima.

(contradiction otherwise)



Convexity: Zero-order condition

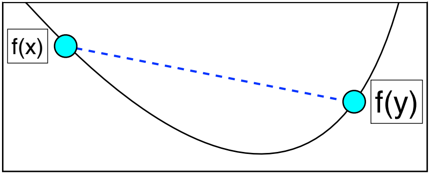
A real-valued function is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),$$

for all $x, y \in \mathbb{R}^n$ *and all* $0 \leq \theta \leq 1$.

- Function is *below a linear interpolation* from x to y.
- Implies that all local minima are global minima.

(contradiction otherwise)



Convexity: Zero-order condition

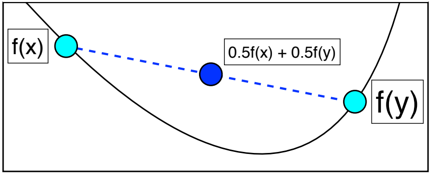
A real-valued function is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),$$

for all $x, y \in \mathbb{R}^n$ *and all* $0 \leq \theta \leq 1$.

- Function is *below a linear interpolation* from x to y.
- Implies that all local minima are global minima.

(contradiction otherwise)



Convexity: Zero-order condition

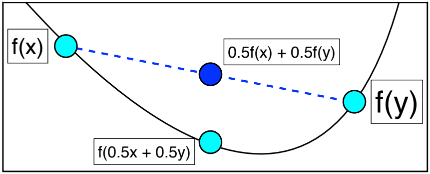
A real-valued function is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),$$

for all $x, y \in \mathbb{R}^n$ *and all* $0 \leq \theta \leq 1$.

- Function is *below a linear interpolation* from x to y.
- Implies that all local minima are global minima.

(contradiction otherwise)



Convexity: Zero-order condition

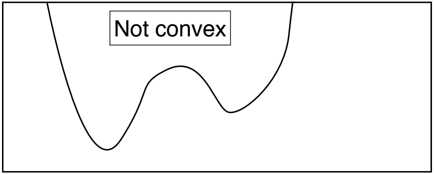
A real-valued function is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),$$

for all $x, y \in \mathbb{R}^n$ *and all* $0 \leq \theta \leq 1$.

- Function is *below a linear interpolation* from x to y.
- Implies that all local minima are global minima.

(contradiction otherwise)



Convexity: Zero-order condition

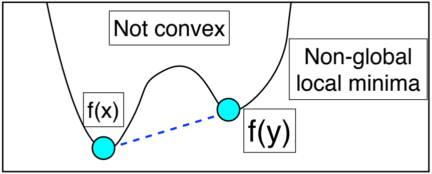
A real-valued function is convex if

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta)f(y),$$

for all $x, y \in \mathbb{R}^n$ *and all* $0 \leq \theta \leq 1$.

- Function is *below a linear interpolation* from x to y.
- Implies that all local minima are global minima.

(contradiction otherwise)



Convexity of Norms

We say that a function f is a **norm** *if:*

1. $f(0) = 0$.
2. $f(\theta x) = |\theta| f(x)$.
3. $f(x + y) \leq f(x) + f(y)$.

Examples:

$$\|x\|_2 = \sqrt{\sum_i x_i^2} = \sqrt{x^T x}$$

$$\|x\|_1 = \sum_i |x_i|$$

$$\|x\|_H = \sqrt{x^T H x}$$

Convexity of Norms

We say that a function f is a **norm** *if:*

1. $f(0) = 0$.
2. $f(\theta x) = |\theta| f(x)$.
3. $f(x + y) \leq f(x) + f(y)$.

Examples:

$$\|x\|_2 = \sqrt{\sum_i x_i^2} = \sqrt{x^T x}$$

$$\|x\|_1 = \sum_i |x_i|$$

$$\|x\|_H = \sqrt{x^T H x}$$

Norms are convex:

$$f(\theta x + (1 - \theta) y) \leq f(\theta x) + f((1 - \theta) y) \qquad (3)$$

$$= \theta f(x) + (1 - \theta) f(y) \qquad (2)$$

Strict Convexity

A real-valued function is strictly convex if

$$f(\theta x + (1-\theta)y) < \theta f(x) + (1-\theta)f(y),$$

for all $x \neq y \in \mathbb{R}^n$ *and all* $0 < \theta < 1$.

- *Strictly below the linear interpolation* from x to y.

Strict Convexity

A real-valued function is strictly convex if

$$f(\theta x + (1-\theta)y) < \theta f(x) + (1-\theta)f(y),$$

for all $x \neq y \in \mathbb{R}^n$ *and all* $0 < \theta < 1$.

- *Strictly below the linear interpolation* from x to y.
- Implies at most one global minimum.

(otherwise, could construct lower global minimum)

Convexity: First-order condition

A real-valued differentiable function is convex iff

$$f(y) \geq f(x) + \nabla f(x)^T (y - x),$$

for all $x, y \in \mathbb{R}^n$.

- The function is globally *above the tangent* at x.

(if $\nabla f(y) = 0$ then y is a a global minimizer)

Convexity: First-order condition

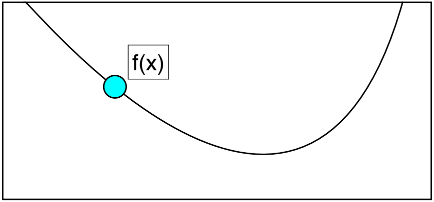
A real-valued differentiable function is convex iff

$$f(y) \geq f(x) + \nabla f(x)^T (y - x),$$

for all x, $y \in \mathbb{R}^n$.

- The function is globally *above the tangent* at x.

(if $\nabla f(y) = 0$ then y is a a global minimizer)



Convexity: First-order condition

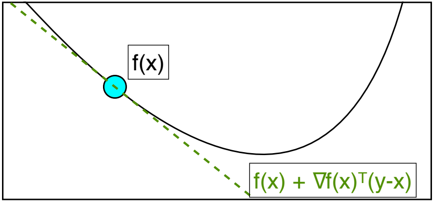
A real-valued differentiable function is convex iff

$$f(y) \geq f(x) + \nabla f(x)^T (y - x),$$

for all $x, y \in \mathbb{R}^n$.

- The function is globally *above the tangent* at x.

(if $\nabla f(y) = 0$ then y is a a global minimizer)



Convexity: First-order condition

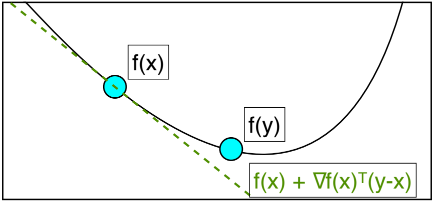
A real-valued differentiable function is convex iff

$$f(y) \geq f(x) + \nabla f(x)^T (y - x),$$

for all $x, y \in \mathbb{R}^n$.

- The function is globally *above the tangent* at x.

(if $\nabla f(y) = 0$ then y is a a global minimizer)



Convexity: Second-order condition

A real-valued twice-differentiable function is convex iff

$$\nabla^2 f(x) \succeq 0$$

for all $x \in \mathbb{R}^n$.

- The function is *flat or curved upwards* in every direction.

Convexity: Second-order condition

A real-valued twice-differentiable function is convex iff

$$\nabla^2 f(x) \succeq 0$$

for all $x \in \mathbb{R}^n$.

- The function is *flat or curved upwards* in every direction.

A real-valued function f is a quadratic if it can be written in the form:

$$f(x) = \frac{1}{2}x^T Ax + b^T x + c.$$

Since $\nabla f(x) = Ax + b$ and $\nabla^2 f(x) = A$, it is convex if $A \succeq 0$.

Examples of Convex Functions

Some simple convex functions:

- $f(x) = c$
- $f(x) = a^T x$
- $f(x) = xa^2 + b$
- $f(x) = \exp(ax)$
- $f(x) = x \log x$ (for $x > 0$)
- $f(x) = ||x||^2$
- $f(x) = \max_i \{x_i\}$

Examples of Convex Functions

Some simple convex functions:

- $f(x) = c$
- $f(x) = a^T x$
- $f(x) = xa^2 + b$
- $f(x) = \exp(ax)$
- $f(x) = x \log x$ (for $x > 0$)
- $f(x) = ||x||^2$
- $f(x) = \max_i\{x_i\}$

Some other notable examples:

- $f(x, y) = \log(e^x + e^y)$
- $f(X) = \log \det X$ (for X positive-definite).
- $f(x, Y) = x^T Y^{-1} x$ (for Y positive-definite)

Operations that Preserve Convexity

1. Non-negative weighted sum:

$$f(x) = \theta_1 f_1(x) + \theta_2 f_2(x).$$

2. Composition with affine mapping:

$$g(x) = f(Ax + b).$$

3. Pointwise maximum:

$$f(x) = \max_i \{f_i(x)\}.$$

Operations that Preserve Convexity

1. Non-negative weighted sum:

$$f(x) = \theta_1 f_1(x) + \theta_2 f_2(x).$$

2. Composition with affine mapping:

$$g(x) = f(Ax + b).$$

3. Pointwise maximum:

$$f(x) = \max_i \{f_i(x)\}.$$

Show that least-residual problems are convex for any ℓ_p-norm:

$$f(x) = ||Ax - b||_p$$

Operations that Preserve Convexity

1. Non-negative weighted sum:

$$f(x) = \theta_1 f_1(x) + \theta_2 f_2(x).$$

2. Composition with affine mapping:

$$g(x) = f(Ax + b).$$

3. Pointwise maximum:

$$f(x) = \max_i \{f_i(x)\}.$$

Show that least-residual problems are convex for any ℓ_p-norm:

$$f(x) = ||Ax - b||_p$$

We know that $\| \cdot \|_p$ is a norm, so it follows from (2).

Operations that Preserve Convexity

1. Non-negative weighted sum:
$$f(x) = \theta_1 f_1(x) + \theta_2 f_2(x).$$
2. Composition with affine mapping:
$$g(x) = f(Ax + b).$$
3. Pointwise maximum:
$$f(x) = \max_i \{f_i(x)\}.$$

Show that SVMs are convex:
$$f(x) = \frac{1}{2}||x||^2 + C\sum_{i=1}^{n} \max\{0, 1 - b_i a_i^T x\}.$$

Operations that Preserve Convexity

1. Non-negative weighted sum:

$$f(x) = \theta_1 f_1(x) + \theta_2 f_2(x).$$

2. Composition with affine mapping:

$$g(x) = f(Ax + b).$$

3. Pointwise maximum:

$$f(x) = \max_i \{f_i(x)\}.$$

Show that SVMs are convex:

$$f(x) = \frac{1}{2}||x||^2 + C\sum_{i=1}^{n} \max\{0, 1 - b_i a_i^T x\}.$$

The first term has Hessian $I \succ 0$, for the second term use (3) on the two (convex) arguments, then use (1) to put it all together.

Outline

How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

- General function: impossible!

(think about arbitrarily small value at some infinite decimal expansion)

How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

- General function: impossible!

(think about arbitrarily small value at some infinite decimal expansion)

We need to make some assumptions about the function:

- Assume f is Lipschitz-continuous: (can not change too quickly)

$$|f(x) - f(y)| \leq L\|x - y\|.$$

How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

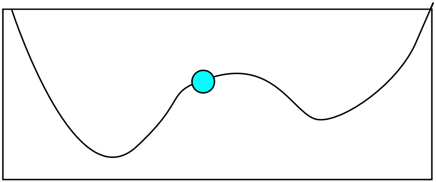
- General function: impossible!

(think about arbitrarily small value at some infinite decimal expansion)

We need to make some assumptions about the function:

- Assume f is Lipschitz-continuous: (can not change too quickly)

$$|f(x) - f(y)| \leq L\|x - y\|.$$



How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

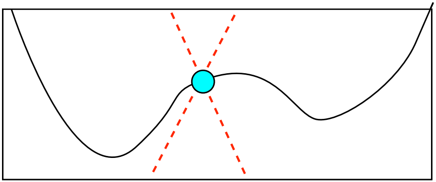
- General function: impossible!

(think about arbitrarily small value at some infinite decimal expansion)

We need to make some assumptions about the function:

- Assume f is Lipschitz-continuous: (can not change too quickly)

$$|f(x) - f(y)| \leq L\|x - y\|.$$



How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

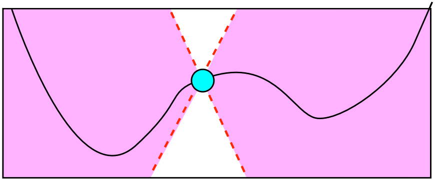
- General function: impossible!

(think about arbitrarily small value at some infinite decimal expansion)

We need to make some assumptions about the function:

- Assume f is Lipschitz-continuous: (can not change too quickly)

$$|f(x) - f(y)| \leq L\|x - y\|.$$



How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

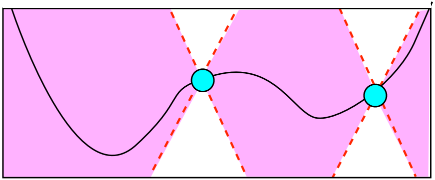
- General function: impossible!

(think about arbitrarily small value at some infinite decimal expansion)

We need to make some assumptions about the function:

- Assume f is Lipschitz-continuous: (can not change too quickly)

$$|f(x) - f(y)| \leq L\|x - y\|.$$



How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

- General function: impossible!

(think about arbitrarily small value at some infinite decimal expansion)

We need to make some assumptions about the function:

- Assume f is Lipschitz-continuous: (can not change too quickly)

$$|f(x) - f(y)| \leq L\|x - y\|.$$

- After t iterations, the error of *any algorithm* is $\Omega(1/t^{1/n})$.

(this is in the worst case, and note that grid-search is nearly optimal)

How hard is real-valued optimization?

How long to find an ϵ-optimal minimizer of a real-valued function?

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

- General function: impossible!

(think about arbitrarily small value at some infinite decimal expansion)

We need to make some assumptions about the function:

- Assume f is Lipschitz-continuous: (can not change too quickly)

$$|f(x) - f(y)| \leq L\|x - y\|.$$

- After t iterations, the error of *any algorithm* is $\Omega(1/t^{1/n})$.

(this is in the worst case, and note that grid-search is nearly optimal)

- Optimization is hard, but assumptions make a big difference.

(we went from impossible to very slow)

ℓ_2-Regularized Logistic Regression

- Consider ℓ_2-regularized logistic regression:

$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))) + \frac{\lambda}{2}\|x\|^2.$$

- Objective f is convex.
- First term is Lipschitz continuous.
- Second term is not Lipschitz continuous.

ℓ_2-Regularized Logistic Regression

- Consider ℓ_2-regularized logistic regression:

$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))) + \frac{\lambda}{2}\|x\|^2.$$

- Objective f is convex.
- First term is Lipschitz continuous.
- Second term is not Lipschitz continuous.
- But we have

$$\mu I \preceq \nabla^2 f(x) \preceq LI.$$

$(L = \frac{1}{4}\|A\|_2^2 + \lambda,\ \mu = \lambda)$

- Gradient is Lipschitz-continuous.
- Function is strongly-convex.

(implies strict convexity, and existence of unique solution)

Properties of Lipschitz-Continuous Gradient

- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

Properties of Lipschitz-Continuous Gradient

- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

- Use that $\nabla^2 f(z) \preceq LI$.

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

- *Global quadratic upper bound on function value.*

Properties of Lipschitz-Continuous Gradient

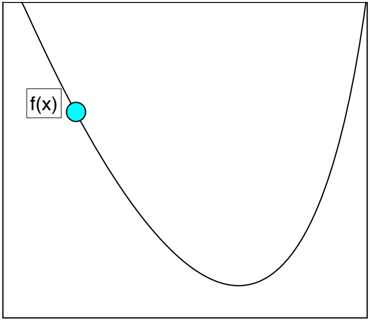
- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x)$$

- Use that $\nabla^2 f(z) \preceq LI$.

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

- *Global quadratic upper bound on function value.*



Properties of Lipschitz-Continuous Gradient

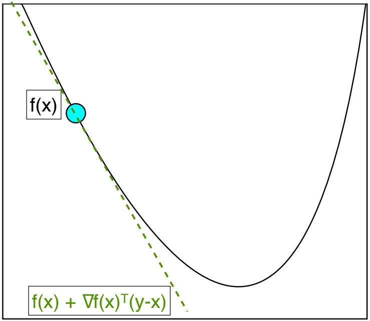
- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

- Use that $\nabla^2 f(z) \preceq LI$.

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

- *Global quadratic upper bound on function value.*



Properties of Lipschitz-Continuous Gradient

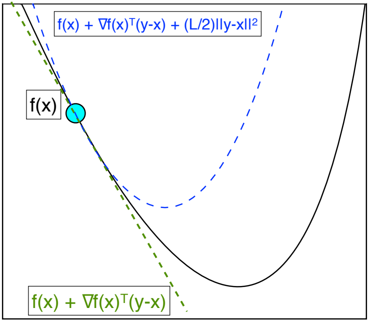
- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

- Use that $\nabla^2 f(z) \preceq LI$.

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

- *Global quadratic upper bound on function value.*



Properties of Lipschitz-Continuous Gradient

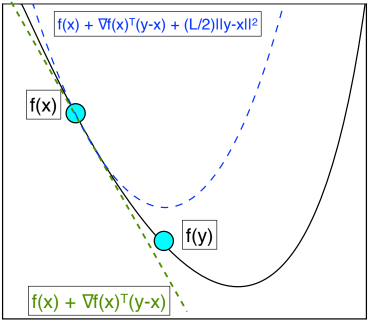
- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

- Use that $\nabla^2 f(z) \preceq LI$.

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

- *Global quadratic upper bound on function value.*



Properties of Lipschitz-Continuous Gradient

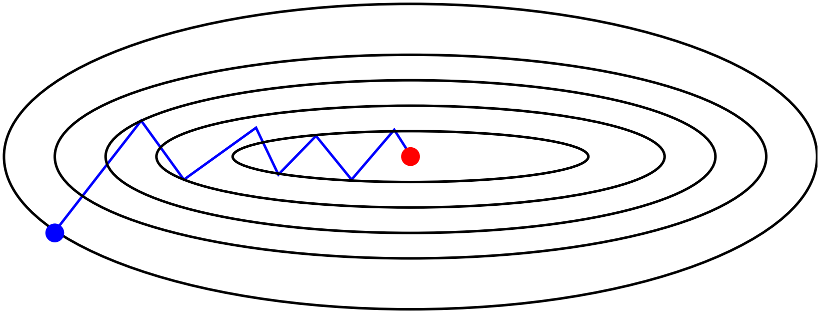
- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

- Use that $\nabla^2 f(z) \preceq LI$.

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

- *Global quadratic upper bound on function value.*



Properties of Lipschitz-Continuous Gradient

- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

- Use that $\nabla^2 f(z) \preceq LI$.

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

- *Global quadratic upper bound on function value.*
- Set x^+ to minimize upper bound in terms of y:

$$x^+ = x - \frac{1}{L}\nabla f(x).$$

(gradient descent with step-size of $1/L$)

- Plugging this value in:

$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2.$$

(decrease of *at least* $\frac{1}{2L}\|\nabla f(x)\|^2$)

Properties of Strong-Convexity

- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x)$$

Properties of Strong-Convexity

- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x)$$

- Use that $\nabla^2 f(z) \succeq \mu I$.

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

- *Global quadratic upper bound on function value.*

Properties of Strong-Convexity

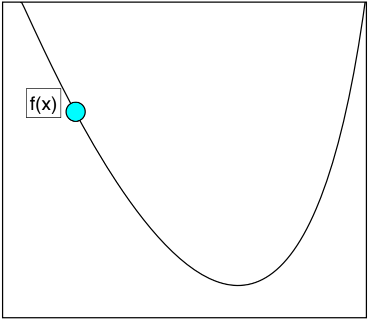
- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T (y - x) + \frac{1}{2}(y - x)^T \nabla^2 f(z)(y - x)$$

- Use that $\nabla^2 f(z) \succeq \mu I$.

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

- *Global quadratic upper bound on function value.*



Properties of Strong-Convexity

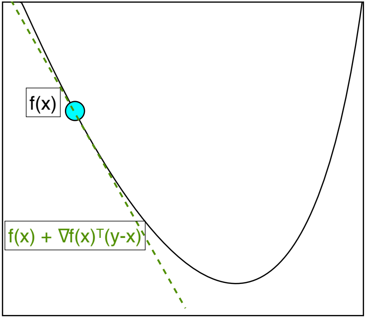
- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

- Use that $\nabla^2 f(z) \succeq \mu I$.

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

- *Global quadratic upper bound on function value.*



Properties of Strong-Convexity

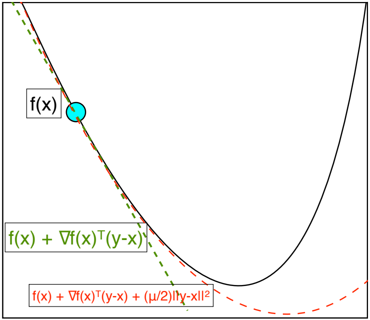
- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y-x) + \frac{1}{2}(y-x)^T\nabla^2 f(z)(y-x)$$

- Use that $\nabla^2 f(z) \succeq \mu I$.

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

- *Global quadratic upper bound on function value.*



Properties of Strong-Convexity

- From Taylor's theorem, for some z we have:

$$f(y) = f(x) + \nabla f(x)^T(y - x) + \frac{1}{2}(y - x)^T\nabla^2 f(z)(y - x)$$

- Use that $\nabla^2 f(z) \succeq \mu I$.

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

- *Global quadratic upper bound on function value.*
- Minimize both sides in terms of y:

$$f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$

- Upper bound on how far we are from the solution.

Linear Convergence of Gradient Descent

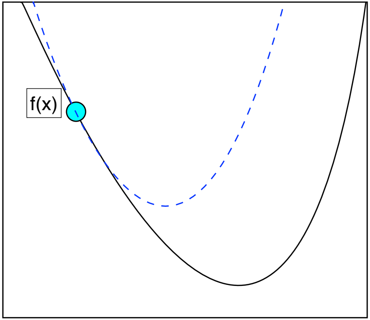
- We have bounds on x^+ and x^*:

$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2, \quad f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$

Linear Convergence of Gradient Descent

- We have bounds on x^+ and x^*:

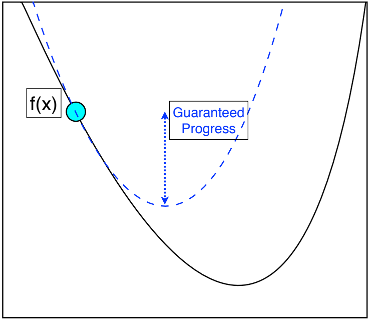
$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2, \quad f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$



Linear Convergence of Gradient Descent

- We have bounds on x^+ and x^*:

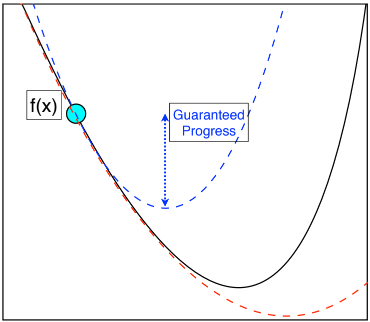
$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2, \quad f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$



Linear Convergence of Gradient Descent

- We have bounds on x^+ and x^*:

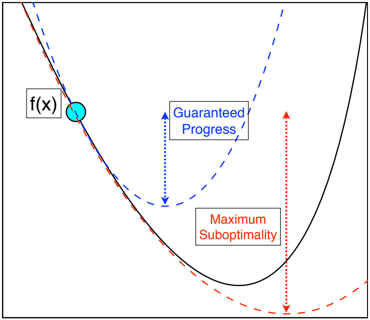
$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2, \quad f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$



Linear Convergence of Gradient Descent

- We have bounds on x^+ and x^*:

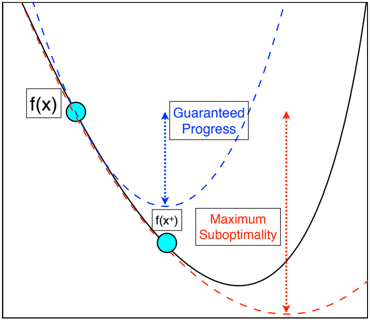
$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2, \quad f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$



Linear Convergence of Gradient Descent

- We have bounds on x^+ and x^*:

$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2, \quad f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$



Linear Convergence of Gradient Descent

- We have bounds on x^+ and x^*:

$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2, \quad f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$

combine them to get

$$f(x^+) \leq f(x) - \frac{\mu}{L}[f(x) - f(x^*)]$$

$$f(x^+) - f(x^*) \leq \left(1 - \frac{\mu}{L}\right)[f(x) - f(x^*)]$$

Linear Convergence of Gradient Descent

- We have bounds on x^+ and x^*:

$$f(x^+) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2, \quad f(x^*) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$

combine them to get

$$f(x^+) \leq f(x) - \frac{\mu}{L}[f(x) - f(x^*)]$$

$$f(x^+) - f(x^*) \leq \left(1 - \frac{\mu}{L}\right)[f(x) - f(x^*)]$$

- This gives a linear convergence rate:

$$f(x^t) - f(x^*) \leq \left(1 - \frac{\mu}{L}\right)^t [f(x^0) - f(x^*)]$$

- Each iteration multiplies the error by a fixed amount.

(very fast if μ/L is not too close to one)

Maximum Likelihood Logistic Regression

- What maximum-likelihood logistic regression?

$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))).$$

Maximum Likelihood Logistic Regression

- What maximum-likelihood logistic regression?

$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))).$$

- We now only have

$$0 \preceq \nabla^2 f(x) \preceq LI.$$

- Convexity only gives a linear upper bound on $f(x^*)$:

$$f(x^*) \leq f(x) + \nabla f(x)^T (x^* - x)$$

Maximum Likelihood Logistic Regression

- What maximum-likelihood logistic regression?

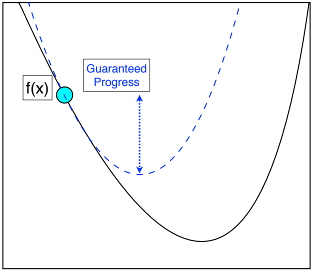
$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))).$$

- We now only have

$$0 \preceq \nabla^2 f(x) \preceq LI.$$

- Convexity only gives a linear upper bound on $f(x^*)$:

$$f(x^*) \leq f(x) + \nabla f(x)^T (x^* - x)$$



Maximum Likelihood Logistic Regression

- What maximum-likelihood logistic regression?

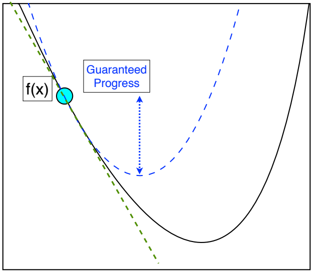
$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))).$$

- We now only have

$$0 \preceq \nabla^2 f(x) \preceq LI.$$

- Convexity only gives a linear upper bound on $f(x^*)$:

$$f(x^*) \leq f(x) + \nabla f(x)^T (x^* - x)$$



Maximum Likelihood Logistic Regression

- What maximum-likelihood logistic regression?

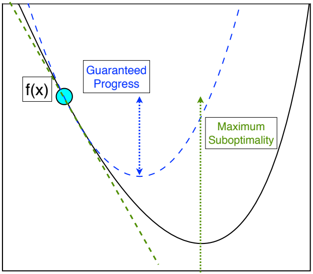
$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))).$$

- We now only have

$$0 \preceq \nabla^2 f(x) \preceq LI.$$

- Convexity only gives a linear upper bound on $f(x^*)$:

$$f(x^*) \leq f(x) + \nabla f(x)^T (x^* - x)$$



Maximum Likelihood Logistic Regression

- Consider maximum-likelihood logistic regression:

$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))).$$

- We now only have

$$0 \preceq \nabla^2 f(x) \preceq LI.$$

- Convexity only gives a linear upper bound on $f(x^*)$:

$$f(x^*) \leq f(x) + \nabla f(x)^T (x^* - x)$$

- If some x^* exists, we have the sublinear convergence rate:

$$f(x^t) - f(x^*) = O(1/t)$$

(compare to slower $\Omega(1/t^{-1/N})$ for general Lipschitz functions)

Maximum Likelihood Logistic Regression

- Consider maximum-likelihood logistic regression:

$$f(x) = \sum_{i=1}^{n} \log(1 + exp(-b_i(x^T a_i))).$$

- We now only have

$$0 \preceq \nabla^2 f(x) \preceq LI.$$

- Convexity only gives a linear upper bound on $f(x^*)$:

$$f(x^*) \leq f(x) + \nabla f(x)^T (x^* - x)$$

- If some x^* exists, we have the sublinear convergence rate:

$$f(x^t) - f(x^*) = O(1/t)$$

(compare to slower $\Omega(1/t^{-1/N})$ for general Lipschitz functions)

- If f is convex, then $f + \lambda\|x\|^2$ is strongly-convex.

Gradient Method: Practical Issues

- In practice, searching for step size (line-search) is usually much faster than $\alpha = 1/L$.

(and doesn't require knowledge of L)

Gradient Method: Practical Issues

- In practice, searching for step size (line-search) is usually much faster than $\alpha = 1/L$.

  (and doesn't require knowledge of L)

- Basic Armijo backtracking line-search:
  1. Start with a large value of α.
  2. Divide α in half until we satisfy (typically value is $\gamma = .0001$)

$$f(x^+) \leq f(x) - \gamma\alpha||\nabla f(x)||^2.$$

Gradient Method: Practical Issues

- In practice, searching for step size (line-search) is usually much faster than $\alpha = 1/L$.

  (and doesn't require knowledge of L)

- Basic Armijo backtracking line-search:
  1. Start with a large value of α.
  2. Divide α in half until we satisfy (typically value is $\gamma = .0001$)

$$f(x^+) \leq f(x) - \gamma\alpha||\nabla f(x)||^2.$$

- Practical methods may use *Wolfe conditions* (so α isn't too small), and/or use *interpolation* to propose trial step sizes.

  (with good interpolation, ≈ 1 evaluation of f per iteration)

Gradient Method: Practical Issues

- In practice, searching for step size (line-search) is usually much faster than $\alpha = 1/L$.

(and doesn't require knowledge of L)

- Basic Armijo backtracking line-search:
  1. Start with a large value of α.
  2. Divide α in half until we satisfy (typically value is $\gamma = .0001$)

$$f(x^+) \leq f(x) - \gamma\alpha||\nabla f(x)||^2.$$

- Practical methods may use *Wolfe conditions* (so α isn't too small), and/or use *interpolation* to propose trial step sizes.

(with good interpolation, ≈ 1 evaluation of f per iteration)

- Also, check your derivative code!

$$\nabla_i f(x) \approx \frac{f(x + \delta e_i) - f(x)}{\delta}$$

- For large-scale problems you can check a random direction d:

$$\nabla f(x)^T d \approx \frac{f(x + \delta d) - f(x)}{\delta}$$

Convex Optimization Zoo

We are going to explore the 'convex optimization zoo':

Algorithm	Assumptions	Rate
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Gradient	Lipshitz Gradient, Strongly-Convex	$O((1-\mu/L)^t)$

Convex Optimization Zoo

We are going to explore the 'convex optimization zoo':

Algorithm	Assumptions	Rate
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Gradient	Lipshitz Gradient, Strongly-Convex	$O((1-\mu/L)^t)$

- Rates are the same if only once-differentiable.
- Line-search doesn't change the worst-case rate.

(strongly-convex slightly improved with $\alpha = 2/(\mu + L)$)

Convex Optimization Zoo

We are going to explore the 'convex optimization zoo':

Algorithm	Assumptions	Rate
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Gradient	Lipshitz Gradient, Strongly-Convex	$O((1-\mu/L)^t)$

- Rates are the same if only once-differentiable.
- Line-search doesn't change the worst-case rate.

  (strongly-convex slightly improved with $\alpha = 2/(\mu + L)$)
- Is this the best algorithm under these assumptions?

Accelerated Gradient Method

- Nesterov's accelerated gradient method:

$$x_{t+1} = y_t - \alpha_t f'(y_t),$$
$$y_{t+1} = x_t + \beta_t(x_{t+1} - x_t),$$

for appropriate α_t, β_t.

Accelerated Gradient Method

- Nesterov's accelerated gradient method:

$$x_{t+1} = y_t - \alpha_t f'(y_t),$$
$$y_{t+1} = x_t + \beta_t(x_{t+1} - x_t),$$

  for appropriate α_t, β_t.
- Motivation: "to make the math work"

  (but similar to heavy-ball/momentum and conjugate gradient method)

Convex Optimization Zoo

Algorithm	Assumptions	Rate
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Nesterov	Lipshitz Gradient, Convex	$O(1/t^2)$
Gradient	Lipshitz Gradient, Strongly-Convex	$O((1-\mu/L)^t)$
Nesterov	Lipshitz Gradient, Strongly-Convex	$O((1-\sqrt{\mu/L})^t)$

- $O(1/t^2)$ is optimal given only these assumptions.

  (sometimes called the *optimal* gradient method)
- The faster linear convergence rate is close to optimal.
- Also faster in practice, but implementation details matter.

Newton's Method

- The oldest differentiable optimization method is Newton's.

  (also called IRLS for functions of the form $f(Ax)$)

- Modern form uses the update

$$x^+ = x - \alpha d,$$

  where d is a solution to the system

$$\nabla^2 f(x) d = \nabla f(x).$$

(Assumes $\nabla^2 f(x) \succ 0$)

Newton's Method

- The oldest differentiable optimization method is Newton's.

(also called IRLS for functions of the form $f(Ax)$)

- Modern form uses the update

$$x^+ = x - \alpha d,$$

where d is a solution to the system

$$\nabla^2 f(x) d = \nabla f(x).$$

(Assumes $\nabla^2 f(x) \succ 0$)

- Equivalent to minimizing the quadratic approximation:

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha}\|y - x\|^2_{\nabla^2 f(x)}.$$

(recall that $\|x\|^2_H = x^T H x$)

Newton's Method

- The oldest differentiable optimization method is Newton's.

  (also called IRLS for functions of the form $f(Ax)$)

- Modern form uses the update

$$x^+ = x - \alpha d,$$

where d is a solution to the system

$$\nabla^2 f(x) d = \nabla f(x).$$

(Assumes $\nabla^2 f(x) \succ 0$)

- Equivalent to minimizing the quadratic approximation:

$$f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha}\|y - x\|^2_{\nabla^2 f(x)}.$$

(recall that $\|x\|^2_H = x^T H x$)

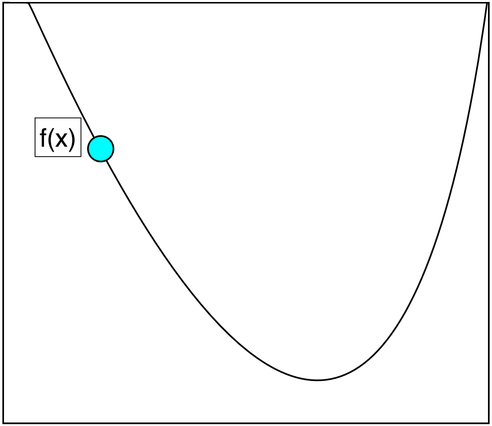
- We can generalize the Armijo condition to

$$f(x^+) \leq f(x) + \gamma\alpha\nabla f'(x)^T d.$$

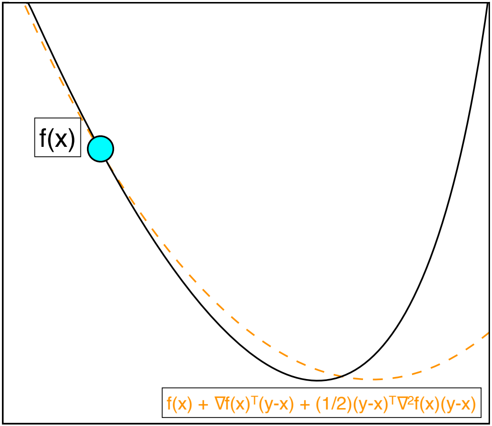
- Has a natural step length of $\alpha = 1$.

  (always accepted when close to a minimizer)

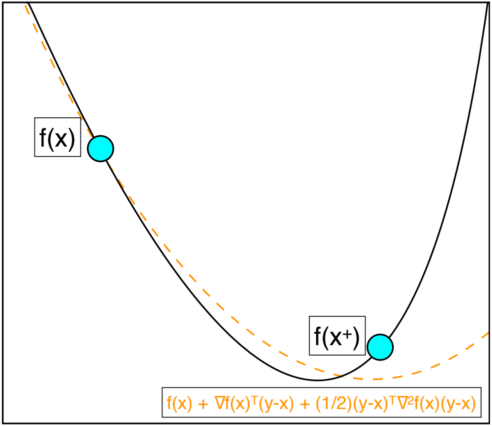
Newton's Method



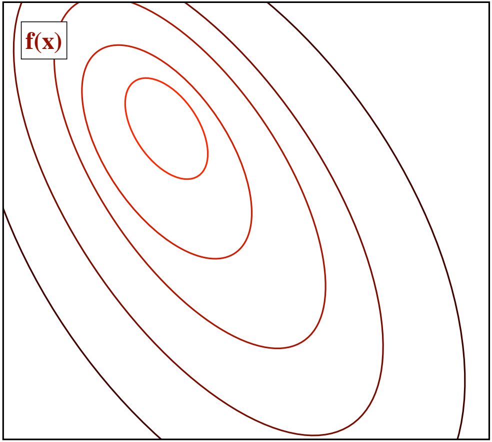
Newton's Method



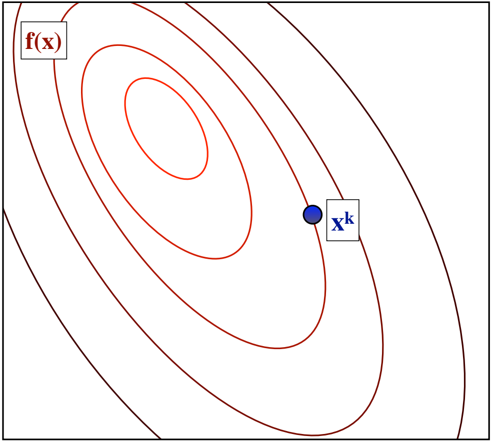
Newton's Method



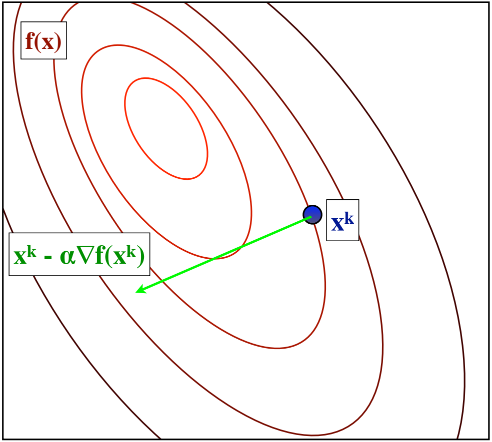
Newton's Method



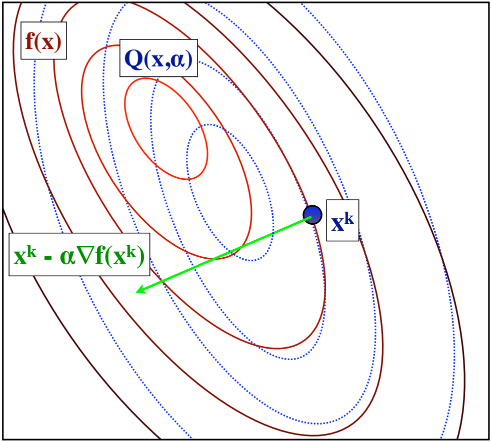
Newton's Method



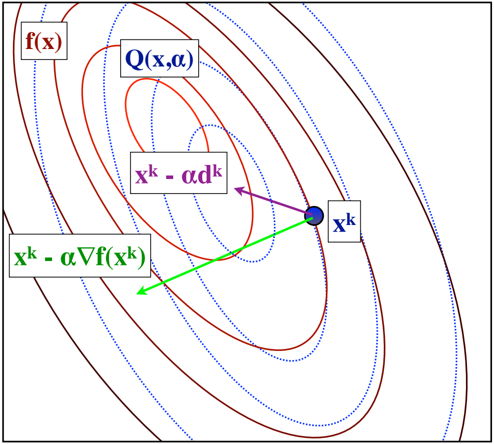
Newton's Method



Newton's Method



Newton's Method



Convergence Rate of Newton's Method

- If $\nabla^2 f(x)$ is Lipschitz-continuous and $\nabla^2 f(x) \succeq \mu$, then close to x^* Newton's method has superlinear convergence:

$$f(x^{t+1}) - f(x^*) \leq \rho_t [f(x^t) - f(x^*)],$$

  with $\lim_{t \to \infty} \rho_t = 0$.
- Converges very fast, use it if you can!
- But requires solving $\nabla^2 f(x) d = \nabla f(x)$.

Convex Optimization Zoo

Algorithm	Assumptions	Rate
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Nesterov	Lipshitz Gradient, Convex	$O(1/t^2)$
Gradient	Lipshitz Gradient, Strongly-Convex	$O((1-\mu/L)^t)$
Nesterov	Lipshitz Gradient, Strongly-Convex	$O((1-\sqrt{\mu/L})^t)$
Newton	Lipschitz Hessian, Strongly-Convex	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

- Here the classical analysis gives a local rate.
- Recent work gives global rates under various assumptions (cubic-regularization/accelerated/self-concordant).

Newton's Method: Practical Issues

There are many practical variants of Newton's method:

- Modify the Hessian to be positive-definite.
- Only compute the Hessian every m iterations.
- Only use the diagonals of the Hessian.
- Quasi-Newton: Update a (diagonal plus low-rank) approximation of the Hessian (BFGS, L-BFGS).

Newton's Method: Practical Issues

There are many practical variants of Newton's method:

- Modify the Hessian to be positive-definite.
- Only compute the Hessian every m iterations.
- Only use the diagonals of the Hessian.
- Quasi-Newton: Update a (diagonal plus low-rank) approximation of the Hessian (BFGS, L-BFGS).
- Hessian-free: Compute d inexactly using Hessian-vector products:
$$\nabla^2 f(x)^T d = \lim_{\delta \to 0} \frac{\nabla f(x + \delta d) - \nabla f(x)}{\delta}$$
- Barzilai-Borwein: Choose a step-size that acts like the Hessian over the last iteration:
$$\alpha = \frac{(x^+ - x)^T(\nabla f(x^+) - \nabla f(x))}{\|\nabla f(x^+) - f(x)\|^2}$$

Another related method is nonlinear conjugate gradient.

Outline

1. Convex Functions
2. Smooth Optimization
3. Non-Smooth Optimization
4. Stochastic Optimization

Motivation: Sparse Regularization

- Consider ℓ_1-regularized optimization problems,

$$\min_x f(x) = g(x) + \lambda \|x\|_1,$$

where g is differentiable.

- For example, ℓ_1-regularized least squares,

$$\min_x \|Ax - b\|^2 + \lambda \|x\|_1$$

- Regularizes and encourages sparsity in x

Motivation: Sparse Regularization

- Consider ℓ_1-regularized optimization problems,

$$\min_x f(x) = g(x) + \lambda \|x\|_1,$$

  where g is differentiable.
- For example, ℓ_1-regularized least squares,

$$\min_x \|Ax - b\|^2 + \lambda \|x\|_1$$

- Regularizes and encourages sparsity in x
- The objective is non-differentiable when any $x_i = 0$.
- How can we solve non-smooth convex optimization problems?

Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y.$$

A vector d is a subgradient of a convex function f at x if

$$f(y) \geq f(x) + d^T (y - x), \forall y.$$

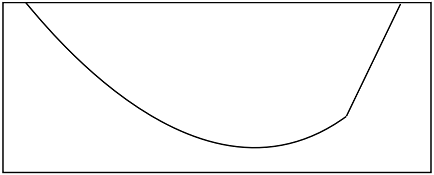
Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T(y - x), \forall x, y.$$

A vector d is a subgradient *of a convex function f at x if*

$$f(y) \geq f(x) + d^T(y - x), \forall y.$$



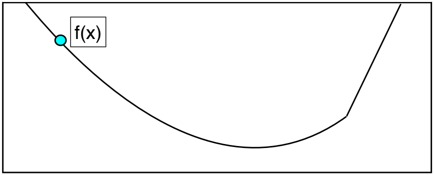
Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y.$$

A vector d is a subgradient *of a convex function f at x if*

$$f(y) \geq f(x) + d^T (y - x), \forall y.$$



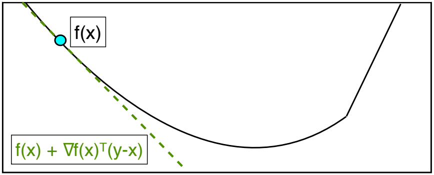
Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y.$$

A vector d is a subgradient *of a convex function f at x if*

$$f(y) \geq f(x) + d^T (y - x), \forall y.$$



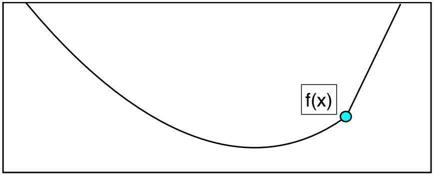
Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y.$$

A vector d is a subgradient *of a convex function f at x if*

$$f(y) \geq f(x) + d^T (y - x), \forall y.$$



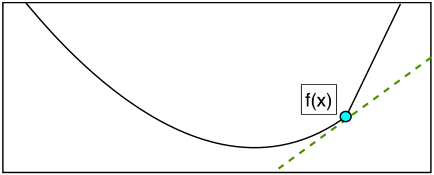
Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y.$$

A vector d is a subgradient *of a convex function f at x if*

$$f(y) \geq f(x) + d^T (y - x), \forall y.$$



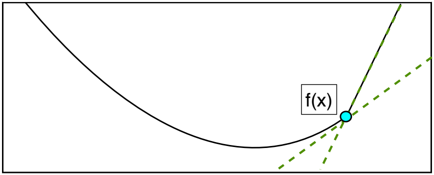
Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T(y-x), \forall x, y.$$

A vector d is a *subgradient* *of a convex function f at x if*

$$f(y) \geq f(x) + d^T(y-x), \forall y.$$



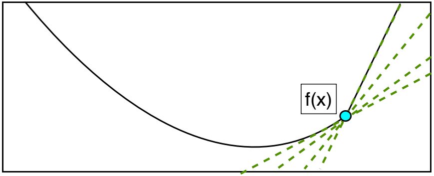
Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y.$$

A vector d is a subgradient *of a convex function f at x if*

$$f(y) \geq f(x) + d^T (y - x), \forall y.$$



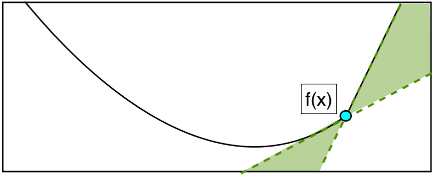
Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y.$$

A vector d is a subgradient of a convex function f at x if

$$f(y) \geq f(x) + d^T (y - x), \forall y.$$



Sub-Gradients and Sub-Differentials

Recall that for *differentiable* convex functions we have

$$f(y) \geq f(x) + \nabla f(x)^T (y - x), \forall x, y.$$

A vector d is a subgradient of a convex function f at x if

$$f(y) \geq f(x) + d^T (y - x), \forall y.$$

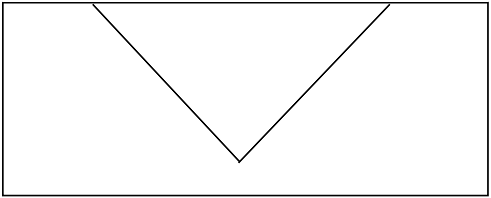
- f is differentiable at x iff $\nabla f(x)$ is the only subgradient.
- At non-differentiable x, we have a set of subgradients.
- Set of subgradients is the sub-differential $\partial f(x)$.
- Note that $0 \in \partial f(x)$ iff x is a global minimum.

Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

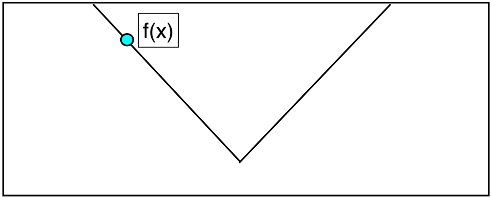


Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

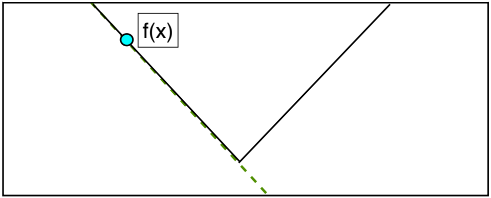


Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

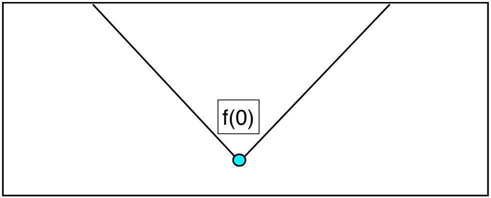


Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

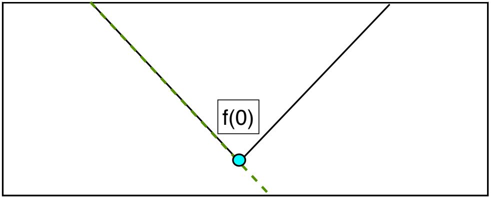


Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

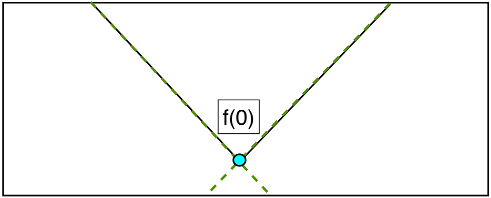


Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

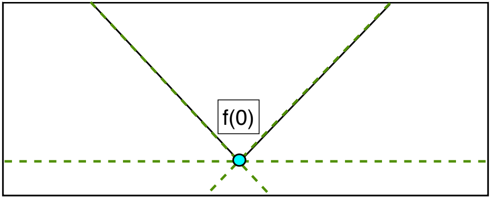


Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

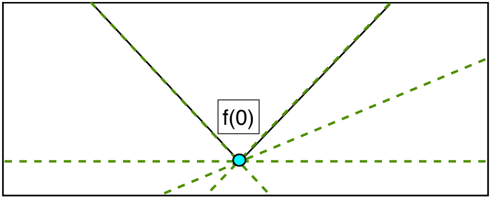


Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

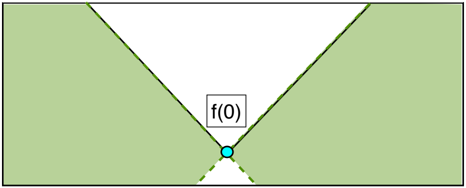


Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)



Sub-Differential of Absolute Value and Max Functions

- The sub-differential of the absolute value function:

$$\partial|x| = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \\ [-1, 1] & x = 0 \end{cases}$$

(sign of the variable if non-zero, anything in $[-1, 1]$ at 0)

- The sub-differential of the maximum of differentiable f_i:

$$\partial \max\{f_1(x), f_2(x)\} = \begin{cases} \nabla f_1(x) & f_1(x) > f_2(x) \\ \nabla f_2(x) & f_2(x) > f_1(x) \\ \theta \nabla f_1(x) + (1-\theta)\nabla f_2(x) & f_1(x) = f_2(x) \end{cases}$$

(any convex combination of the gradients of the argmax)

Sub-gradient method

- The sub-gradient method:

$$x^+ = x - \alpha d,$$

for some $d \in \partial f(x)$.

Sub-gradient method

- The sub-gradient method:

$$x^+ = x - \alpha d,$$

  for some $d \in \partial f(x)$.
- The *steepest descent* step is given by $\arg\min_{d \in \partial f(x)}\{\|d\|\}$.

  (often hard to compute, but easy for ℓ_1-regularization)
- Otherwise, may increase the objective even for small α.
- But $\|x^+ - x^*\| \leq \|x - x^*\|$ for small enough α.
- For convergence, we require $\alpha \to 0$.

Sub-gradient method

- The sub-gradient method:

$$x^+ = x - \alpha d,$$

  for some $d \in \partial f(x)$.
- The *steepest descent* step is given by $\arg\min_{d \in \partial f(x)}\{\|d\|\}$.

  (often hard to compute, but easy for ℓ_1-regularization)
- Otherwise, may increase the objective even for small α.
- But $\|x^+ - x^*\| \leq \|x - x^*\|$ for small enough α.
- For convergence, we require $\alpha \to 0$.
- Many variants average the iterations:

$$\bar{x}^k = \sum_{i=0}^{k-1} w_i x^i.$$

- Many variants average the gradients ('dual averaging'):

$$\bar{d}^k = \sum_{i=0}^{k-1} w_i d^i.$$

Convex Optimization Zoo

Algorithm	Assumptions	Rate
Subgradient	Lipschitz Function, Convex	$O(1/\sqrt{t})$
Subgradient	Lipschitz Function, Strongly	$O(1/t)$
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Nesterov	Lipshitz Gradient, Convex	$O(1/t^2)$
Gradient	Lipshitz Gradient, Strongly	$O((1-\mu/L)^t)$
Nesterov	Lipshitz Gradient, Strongly	$O((1-\sqrt{\mu/L})^t)$
Newton	Lipschitz Hessian, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

Convex Optimization Zoo

Algorithm	Assumptions	Rate
Subgradient	Lipschitz Function, Convex	$O(1/\sqrt{t})$
Subgradient	Lipschitz Function, Strongly	$O(1/t)$
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Nesterov	Lipshitz Gradient, Convex	$O(1/t^2)$
Gradient	Lipshitz Gradient, Strongly	$O((1-\mu/L)^t)$
Nesterov	Lipshitz Gradient, Strongly	$O((1-\sqrt{\mu/L})^t)$
Newton	Lipschitz Hessian, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

- Alternative is cutting-plane/bundle methods:
  - Minimze an approximation based on *all* subgradients $\{d_t\}$.
  - But have the *same rates as the subgradient method*.

(tend to be better in practice)

Convex Optimization Zoo

Algorithm	Assumptions	Rate
Subgradient	Lipschitz Function, Convex	$O(1/\sqrt{t})$
Subgradient	Lipschitz Function, Strongly	$O(1/t)$
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Nesterov	Lipshitz Gradient, Convex	$O(1/t^2)$
Gradient	Lipshitz Gradient, Strongly	$O((1-\mu/L)^t)$
Nesterov	Lipshitz Gradient, Strongly	$O((1-\sqrt{\mu/L})^t)$
Newton	Lipschitz Hessian, Strongly	$O(\prod_{i=1}^{t}\rho_t), \rho_t \to 0$

- Alternative is cutting-plane/bundle methods:
  - Minimze an approximation based on *all* subgradients $\{d_t\}$.
  - But have the *same rates as the subgradient method*.

(tend to be better in practice)

- Bad news: Rates are optimal for black-box methods.

Convex Optimization Zoo

Algorithm	Assumptions	Rate
Subgradient	Lipschitz Function, Convex	$O(1/\sqrt{t})$
Subgradient	Lipschitz Function, Strongly	$O(1/t)$
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Nesterov	Lipshitz Gradient, Convex	$O(1/t^2)$
Gradient	Lipshitz Gradient, Strongly	$O((1-\mu/L)^t)$
Nesterov	Lipshitz Gradient, Strongly	$O((1-\sqrt{\mu/L})^t)$
Newton	Lipschitz Hessian, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

- Alternative is cutting-plane/bundle methods:
  - Minimze an approximation based on *all* subgradients $\{d_t\}$.
  - But have the *same rates as the subgradient method*.

(tend to be better in practice)

- Bad news: Rates are optimal for black-box methods.
- But, we often have more than a black-box.

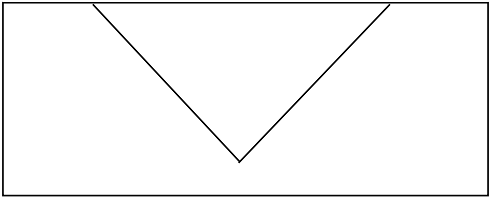
Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.

Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.
- Smooth approximation to the absolute value:

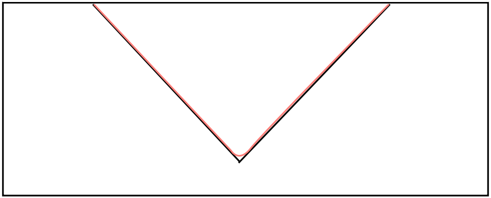
$$|x| \approx \sqrt{x^2 + \nu}.$$



Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.
- Smooth approximation to the absolute value:

$$|x| \approx \sqrt{x^2 + \nu}.$$



Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.
- Smooth approximation to the absolute value:

$$|x| \approx \sqrt{x^2 + \nu}.$$

- Smooth approximation to the max function:

$$\max\{a, b\} \approx \log(\exp(a) + \exp(b))$$

- Smooth approximation to the hinge loss:

$$\max\{0, x\} \approx \begin{cases} 0 & x \geq 1 \\ 1 - x^2 & t < x < 1 \\ (1-t)^2 + 2(1-t)(t-x) & x \leq t \end{cases}$$

Smoothing Approximations of Non-Smooth Functions

- Smoothing: replace non-smooth f with smooth f_ϵ.
- Apply a fast method for smooth optimization.
- Smooth approximation to the absolute value:

$$|x| \approx \sqrt{x^2 + \nu}.$$

- Smooth approximation to the max function:

$$\max\{a, b\} \approx \log(\exp(a) + \exp(b))$$

- Smooth approximation to the hinge loss:

$$\max\{0, x\} \approx \begin{cases} 0 & x \geq 1 \\ 1 - x^2 & t < x < 1 \\ (1-t)^2 + 2(1-t)(t-x) & x \leq t \end{cases}$$

- Generic strategy for constructing ϵ approximation with $O(1/\epsilon)$-Lipschitz gradient: strongly-convex regularization of convex conjugate. (but we won't discuss this in detail)

Convex Optimization Zoo

Algorithm	Assumptions	Rate
Subgradient	Lipschitz Function, Convex	$O(1/\sqrt{t})$
Subgradient	Lipschitz Function, Strongly	$O(1/t)$
Gradient	Smoothed to $1/\epsilon$, Convex	$O(1/\sqrt{t})$
Nesterov	Smoothed to $1/\epsilon$, Convex	$O(1/t)$
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Nesterov	Lipshitz Gradient, Convex	$O(1/t^2)$
Gradient	Lipshitz Gradient, Strongly	$O((1-\mu/L)^t)$
Nesterov	Lipshitz Gradient, Strongly	$O((1-\sqrt{\mu/L})^t)$
Newton	Lipschitz Hessian, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

Convex Optimization Zoo

Algorithm	Assumptions	Rate
Subgradient	Lipschitz Function, Convex	$O(1/\sqrt{t})$
Subgradient	Lipschitz Function, Strongly	$O(1/t)$
Gradient	Smoothed to $1/\epsilon$, Convex	$O(1/\sqrt{t})$
Nesterov	Smoothed to $1/\epsilon$, Convex	$O(1/t)$
Gradient	Lipshitz Gradient, Convex	$O(1/t)$
Nesterov	Lipshitz Gradient, Convex	$O(1/t^2)$
Gradient	Lipshitz Gradient, Strongly	$O((1-\mu/L)^t)$
Nesterov	Lipshitz Gradient, Strongly	$O((1-\sqrt{\mu/L})^t)$
Newton	Lipschitz Hessian, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

- Smoothing is only faster if you use Nesterov's method.
- In practice, faster to slowly decrease smoothing level.
- You can get the $O(1/t)$ rate for $\min_x \max\{f_i(x)\}$ for f_i convex and smooth using Nemirosky's *mirror-prox* method.

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.
- The problem

$$\min_x g(x) + \lambda \|x\|_1,$$

  is equivalent to the problem

$$\min_{x^+ \geq 0, x^- \geq 0} \; g(x^+ - x^-) + \lambda \sum_i (x_i^+ + x_i^-),$$

  or the problems

$$\min_{-y \leq x \leq y} \; g(x) + \lambda \sum_i y_i, \quad \min_{\|x\|_1 \leq \tau} \; g(x) + \lambda\tau$$

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.
- The problem

$$\min_x g(x) + \lambda \|x\|_1,$$

is equivalent to the problem

$$\min_{x^+ \geq 0, x^- \geq 0} \quad g(x^+ - x^-) + \lambda \sum_i (x_i^+ + x_i^-),$$

or the problems

$$\min_{-y \leq x \leq y} \quad g(x) + \lambda \sum_i y_i, \qquad \min_{\|x\|_1 \leq \tau} \quad g(x) + \lambda\tau$$

- These are smooth objective with 'simple' constraints.

$$\min_{x \in \mathcal{C}} f(x).$$

Optimization with Simple Constraints

- Recall: gradient descent minimizes quadratic approximation:

$$x^+ = \mathop{\arg\min}_y \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha}\|y - x\|^2 \right\}.$$

Optimization with Simple Constraints

- Recall: gradient descent minimizes quadratic approximation:

$$x^+ = \operatorname*{arg\,min}_{y} \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha} \|y - x\|^2 \right\}.$$

- Consider minimizing subject to simple constraints:

$$x^+ = \operatorname*{arg\,min}_{y \in \mathcal{C}} \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha} \|y - x\|^2 \right\}.$$

Optimization with Simple Constraints

- Recall: gradient descent minimizes quadratic approximation:

$$x^+ = \operatorname*{arg\,min}_y \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha}\|y - x\|^2 \right\}.$$

- Consider minimizing subject to simple constraints:

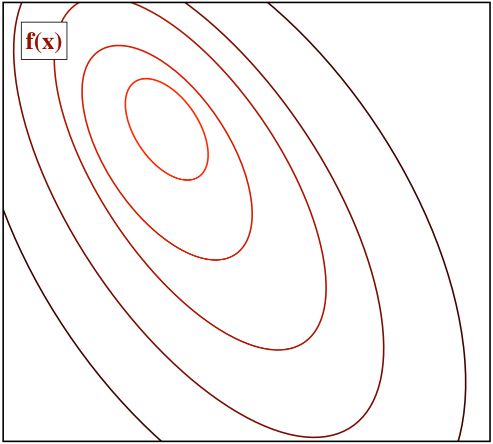
$$x^+ = \operatorname*{arg\,min}_{y \in \mathcal{C}} \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha}\|y - x\|^2 \right\}.$$

- Equivalent to projection of gradient descent:

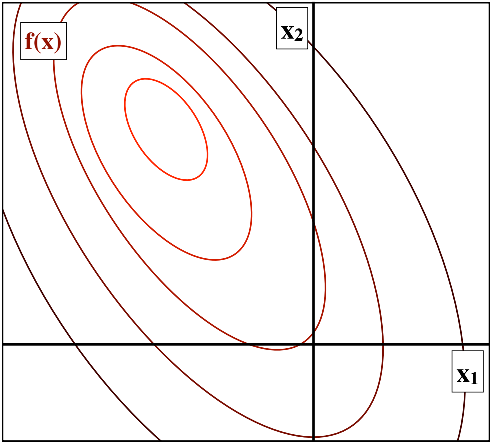
$$x^{GD} = x - \alpha \nabla f(x),$$

$$x^+ = \operatorname*{arg\,min}_{y \in \mathcal{C}} \left\{ \|y - x^{GD}\| \right\},$$

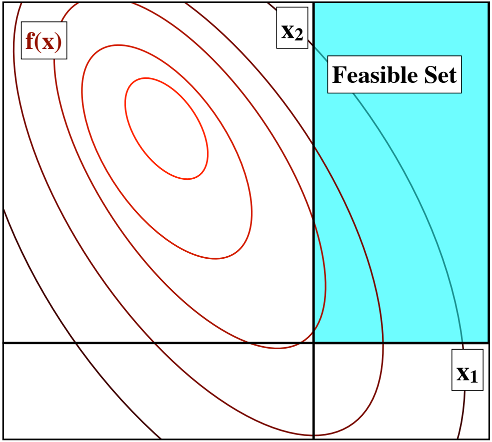
Gradient Projection



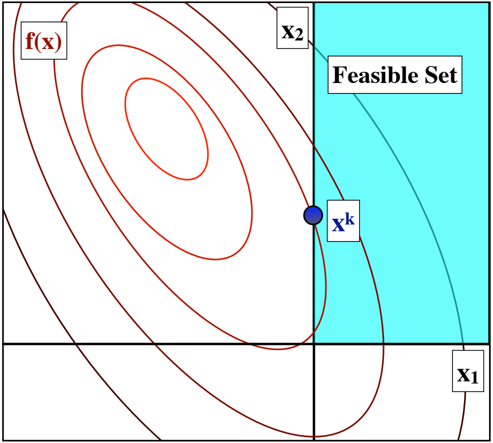
Gradient Projection



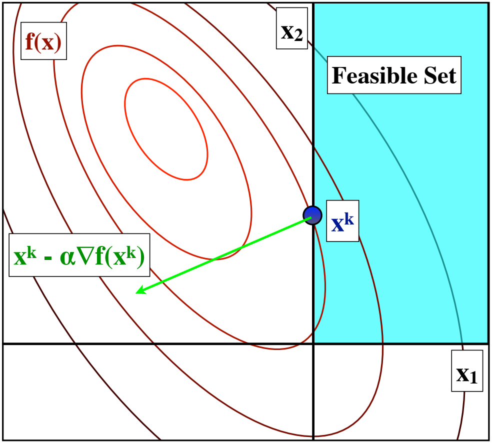
Gradient Projection



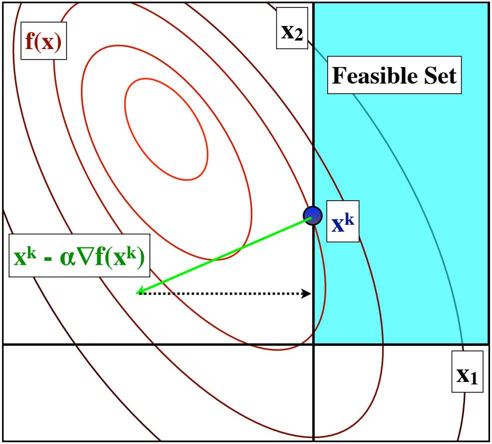
Gradient Projection



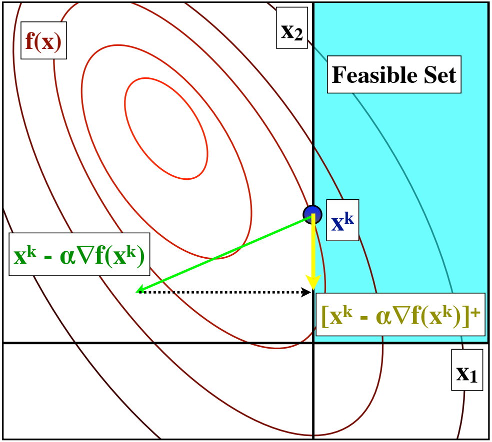
Gradient Projection



Gradient Projection



Gradient Projection



Projection Onto Simple Sets

Projections onto simple sets:

- $\arg\min_{y \geq 0} \|y - x\| = \max\{x, 0\}$
- $\arg\min_{l \leq y \leq u} \|y - x\| = \max\{l, \min\{x, u\}\}$
- $\arg\min_{a^T y = b} \|y - x\| = x + (b - a^T x)a/\|a\|^2$.
- $\arg\min_{a^T y \geq b} \|y - x\| = \begin{cases} x & a^T x \geq b \\ x + (b - a^T x)a/\|a\|^2 & a^T x < b \end{cases}$
- $\arg\min_{\|y\| \leq \tau} \|y - x\| = \tau x/\|x\|$.
- Linear-time algorithm for ℓ_1-norm $\|y\|_1 \leq \tau$.
- Linear-time algorithm for probability simplex $y \geq 0, \sum y = 1$.
- Intersection of simple sets: Dykstra's algorithm.

Convex Optimization Zoo

Algorithm	Assumptions	Rate
P(Subgradient)	Lipschitz Function, Convex	$O(1/\sqrt{t})$
P(Subgradient)	Lipschitz Function, Strongly	$O(1/t)$
P(Nesterov)	Smoothed to $1/\epsilon$, Convex	$O(1/t)$
P(Gradient)	Lipshitz Gradient, Convex	$O(1/t)$
P(Nesterov)	Lipshitz Gradient, Convex	$O(1/t^2)$
P(Gradient)	Lipshitz Gradient, Strongly	$O((1-\mu/L)^t)$
P(Nesterov)	Lipshitz Gradient, Strongly	$O((1-\sqrt{\mu/L})^t)$
P(Newton)	Lipschitz Hessian, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

Convex Optimization Zoo

Algorithm	Assumptions	Rate
P(Subgradient)	Lipschitz Function, Convex	$O(1/\sqrt{t})$
P(Subgradient)	Lipschitz Function, Strongly	$O(1/t)$
P(Nesterov)	Smoothed to $1/\epsilon$, Convex	$O(1/t)$
P(Gradient)	Lipshitz Gradient, Convex	$O(1/t)$
P(Nesterov)	Lipshitz Gradient, Convex	$O(1/t^2)$
P(Gradient)	Lipshitz Gradient, Strongly	$O((1-\mu/L)^t)$
P(Nesterov)	Lipshitz Gradient, Strongly	$O((1-\sqrt{\mu/L})^t)$
P(Newton)	Lipschitz Hessian, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

- Convergence rates are the same for projected versions!
- Can do many of the same tricks (i.e. Armijo line-search, polynomial interpolation, Barzilai-Borwein, quasi-Newton).
- For Newton, you need to project under $\|\cdot\|_{\nabla^2 f(x)}$

  (expensive, but special tricks for the case of simplex or lower/upper bounds)
- You don't need to compute the projection exactly.

Proximal-Gradient Method

- A generalization of projected-gradient is Proximal-gradient.
- The proximal-gradient method addresses problem of the form

$$\min_x f(x) = g(x) + h(x),$$

where g is smooth but h is a general convex function.

Proximal-Gradient Method

- A generalization of projected-gradient is Proximal-gradient.
- The proximal-gradient method addresses problem of the form

$$\min_x f(x) = g(x) + h(x),$$

where g is smooth but h is a general convex function.
- Applies proximity operator of h to gradient descent on g:

$$x^{GD} = x - \alpha \nabla g(x),$$

$$x^+ = \arg\min_y \left\{ \frac{1}{2}\|y - x^{GD}\|^2 + \alpha h(y) \right\},$$

Proximal-Gradient Method

- A generalization of projected-gradient is Proximal-gradient.
- The proximal-gradient method addresses problem of the form

$$\min_x f(x) = g(x) + h(x),$$

  where g is smooth but h is a general convex function.
- Applies proximity operator of h to gradient descent on g:

$$x^{GD} = x - \alpha \nabla g(x),$$

$$x^+ = \operatorname*{arg\,min}_y \left\{ \frac{1}{2}\|y - x^{GD}\|^2 + \alpha h(y) \right\},$$

- If $h(x) = \lambda\|x\|_1$, then

$$\operatorname*{arg\,min}_y \frac{1}{2}\|y - x\|^2 + \alpha\lambda\|y\|_1 = \mathrm{sgn}(x)\max\{0, |x| - \lambda\alpha\}$$

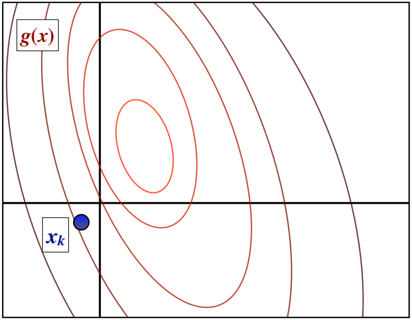
- Convergence rates are still the same as for minimizing g.

Proximal-Gradient Method

- Iterative Soft-Thresholding methods are a special case:

$$h(x) = \lambda \|x\|_1.$$

- In this case $\text{prox}_{\alpha_k}[x]_i$ is $\text{sgn}(x) \max\{0, |x| - \lambda\alpha\}$

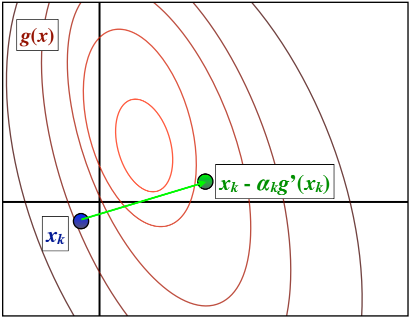


Proximal-Gradient Method

- Iterative Soft-Thresholding methods are a special case:

$$h(x) = \lambda \|x\|_1.$$

- In this case $\text{prox}_{\alpha_k}[x]_i$ is $\text{sgn}(x)\max\{0, |x| - \lambda\alpha\}$

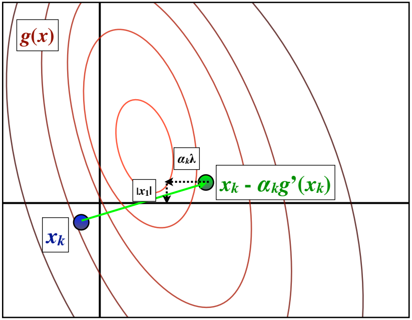


Proximal-Gradient Method

- Iterative Soft-Thresholding methods are a special case:

$$h(x) = \lambda \|x\|_1.$$

- In this case $\text{prox}_{\alpha_k}[x]_i$ is $\text{sgn}(x)\max\{0, |x| - \lambda\alpha\}$

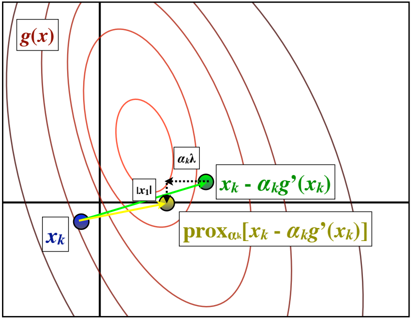


Proximal-Gradient Method

- Iterative Soft-Thresholding methods are a special case:

$$h(x) = \lambda \|x\|_1.$$

- In this case $\text{prox}_{\alpha_k}[x]_i$ is $\text{sgn}(x)\max\{0, |x| - \lambda\alpha\}$



Frank-Wolfe Method

- The projected gradient step

$$x^+ = \mathop{\arg\min}_{y \in \mathcal{C}} \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha} \|y - x\|^2 \right\},$$

  may be hard to compute.
- Frank-Wolfe method simply uses:

$$x^+ = \mathop{\arg\min}_{y \in \mathcal{C}} \left\{ f(x) + \nabla f(x)^T (y - x) \right\},$$

  requires compact $\mathcal{C}$, takes convex combination of x and x^+.

Frank-Wolfe Method

- The projected gradient step

$$x^+ = \operatorname*{arg\,min}_{y \in \mathcal{C}} \left\{ f(x) + \nabla f(x)^T (y - x) + \frac{1}{2\alpha} \|y - x\|^2 \right\},$$

  may be hard to compute.
- Frank-Wolfe method simply uses:

$$x^+ = \operatorname*{arg\,min}_{y \in \mathcal{C}} \left\{ f(x) + \nabla f(x)^T (y - x) \right\},$$

  requires compact $\mathcal{C}$, takes convex combination of x and x^+.
- Iterate can be written as convex combination of vertices of $\mathcal{C}$.
- $O(1/t)$ rate for smooth convex objectives, some linear convergence results for smooth and strongly-convex.

Alternating Direction Method of Multipliers

- Alernating direction method of multipliers (ADMM) solves:

$$\min_{Ax+By=c} g(x) + h(y).$$

Alternating Direction Method of Multipliers

- Alernating direction method of multipliers (ADMM) solves:

$$\min_{Ax+By=c} g(x) + h(y).$$

- Can introduce constraints to convert to this form:

$$\min_{x=y} g(x) + \lambda\|y\|_1.$$

- Alternate between prox-like operators with respect to x and y.
- Useful method for large-scale parallelization.

Dual Methods

- Stronly-convex problems have smooth duals.
- Solve the dual instead of the primal.

Dual Methods

- Stronly-convex problems have smooth duals.
- Solve the dual instead of the primal.
- SVM non-smooth strongly-convex primal:

$$\min_x C \sum_{i=1}^{N} \max\{0, 1 - b_i a_i^T x\} + \frac{1}{2}\|x\|^2.$$

- SVM smooth dual:

$$\min_{0 \leq \alpha \leq C} \frac{1}{2}\alpha^T A A^T \alpha - \sum_{i=1}^{N} \alpha_i$$

- There are many fast methods for bound-constrained problems.

Outline

Stochastic Gradient Method

- Stochastic gradient method uses the iteration

$$x^+ = x - \alpha d,$$

where d is an unbiased estimator of $\nabla f(x)$, so $\mathbb{E}[d] = \nabla f(x)$.

(often using averaging over x or d)

- As in subgradient method, we require $\alpha \to 0$.

(but better in practice with constant step size)

Stochastic Gradient Method

- Stochastic gradient method uses the iteration

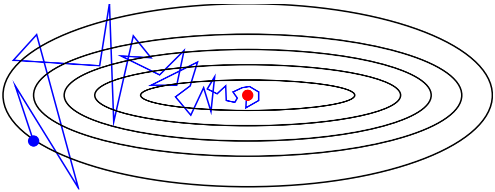
$$x^+ = x - \alpha d,$$

where d is an unbiased estimator of $\nabla f(x)$, so $\mathbb{E}[d] = \nabla f(x)$.

(often using averaging over x or d)

- As in subgradient method, we require $\alpha \to 0$.

(but better in practice with constant step size)



Stochastic Gradient Method

- Stochastic gradient method uses the iteration

$$x^+ = x - \alpha d,$$

where d is an unbiased estimator of $\nabla f(x)$, so $\mathbb{E}[d] = \nabla f(x)$.

(often using averaging over x or d)

- As in subgradient method, we require $\alpha \to 0$.

(but better in practice with constant step size)

- For problems of the form

$$\min_x \frac{1}{N} \sum_{i=1}^{N} f_i(x),$$

we take $d = \nabla f_i(x)$ for a random i.

- Iterations require N times fewer gradient evaluations.
- Appealing when N is large, but how fast is it?

Convex Optimization Zoo

Algorithm	Assumptions	Exact	Stochastic
Subgradient	LF, Convex	$O(1/\sqrt{t})$	$O(1/\sqrt{t})$
Subgradient	LF, Strongly	$O(1/t)$	$O(1/t)$
Nesterov	Smoothed, Convex	$O(1/t)$	$O(1/\sqrt{t})$
Gradient	LG, Convex	$O(1/t)$	$O(1/\sqrt{t})$
Nesterov	LG, Convex	$O(1/t^2)$	$O(1/\sqrt{t})$
Gradient	LG, Strongly	$O((1-\mu/L)^t)$	$O(1/t)$
Nesterov	LG, Strongly	$O((1-\sqrt{\mu/L})^t)$	$O(1/t)$
Newton	LG,LH, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$	$O(1/t)$

Convex Optimization Zoo

Algorithm	Assumptions	Exact	Stochastic
Subgradient	LF, Convex	$O(1/\sqrt{t})$	$O(1/\sqrt{t})$
Subgradient	LF, Strongly	$O(1/t)$	$O(1/t)$
Nesterov	Smoothed, Convex	$O(1/t)$	$O(1/\sqrt{t})$
Gradient	LG, Convex	$O(1/t)$	$O(1/\sqrt{t})$
Nesterov	LG, Convex	$O(1/t^2)$	$O(1/\sqrt{t})$
Gradient	LG, Strongly	$O((1-\mu/L)^t)$	$O(1/t)$
Nesterov	LG, Strongly	$O((1-\sqrt{\mu/L})^t)$	$O(1/t)$
Newton	LG,LH, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$	$O(1/t)$

- Good news: for general non-smooth problems, stochastic is as fast as deterministic
- We can solve non-smooth problems N times faster!

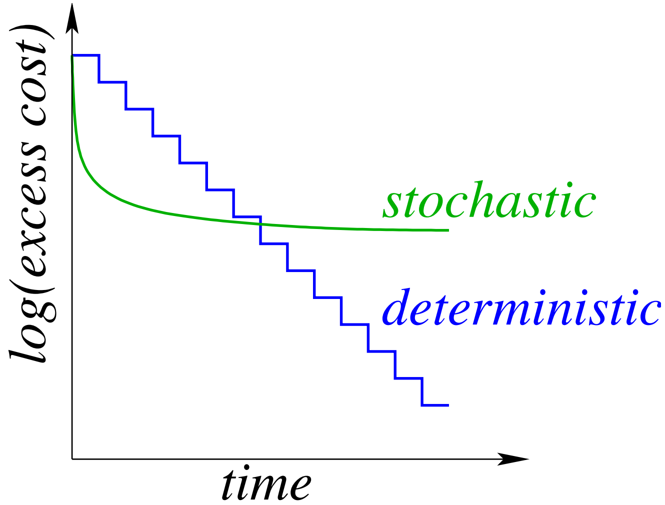
Convex Optimization Zoo

Algorithm	Assumptions	Exact	Stochastic
Subgradient	LF, Convex	$O(1/\sqrt{t})$	$O(1/\sqrt{t})$
Subgradient	LF, Strongly	$O(1/t)$	$O(1/t)$
Nesterov	Smoothed, Convex	$O(1/t)$	$O(1/\sqrt{t})$
Gradient	LG, Convex	$O(1/t)$	$O(1/\sqrt{t})$
Nesterov	LG, Convex	$O(1/t^2)$	$O(1/\sqrt{t})$
Gradient	LG, Strongly	$O((1-\mu/L)^t)$	$O(1/t)$
Nesterov	LG, Strongly	$O((1-\sqrt{\mu/L})^t)$	$O(1/t)$
Newton	LG,LH, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$	$O(1/t)$

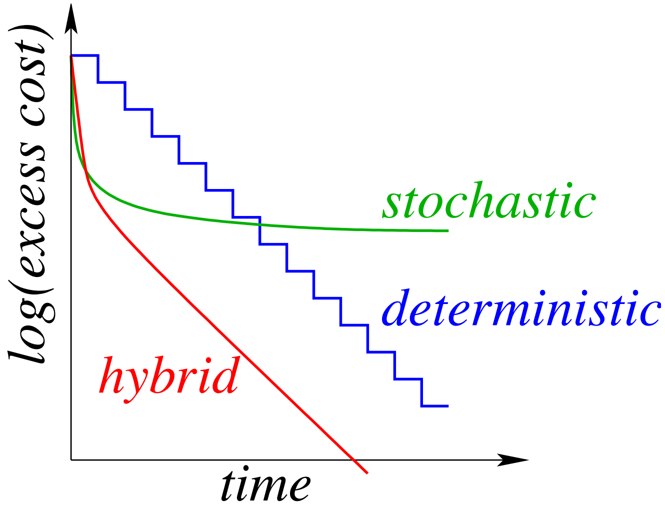
- Good news: for general non-smooth problems, stochastic is as fast as deterministic
- We can solve non-smooth problems N times faster!
- Bad news: smoothness assumptions don't help stochastic methods (most of these rates are optimal).

(recent work shows that $O(1/t)$ for Newton may not require strong convexity)

Motivation for Hybrid Methods for Smooth Problems



Motivation for Hybrid Methods for Smooth Problems



Stochastic Average Gradient Method

- Should we use stochastic methods for smooth problems?
- Problem is that noise doesn't go to 0.
- Solution: make the noise go to zero 'fast enough'.

Stochastic Average Gradient Method

- Should we use stochastic methods for smooth problems?
- Problem is that noise doesn't go to 0.
- Solution: make the noise go to zero 'fast enough'.
- Possible in the case of finite data sets:

$$\min_x \frac{1}{N} \sum_{i=1}^{N} f_i(x),$$

- Stochastic average gradient (SAG) method:

$$x^+ = x - \frac{\alpha}{N} \sum_{i=1}^{N} y_i,$$

on each iteration replace a random y_i with $\nabla f_i(x)$.

Convex Optimization Zoo

Algorithm	Assumptions	Rate	Grads
S(Subgrad)	LF, Convex	$O(1/\sqrt{t})$	1
S(Subgrad)	LF, Strongly	$O(1/t)$	1
SAG	LG, Convex	$O(1/t)$	1
SAG	LG, Strongly	$O((1-\min\{\frac{\mu}{16L_i}, \frac{1}{8N}\})^t)$	1
Nesterov	Smoothed, Convex	$O(1/t)$	N
Gradient	LG, Convex	$O(1/t)$	N
Nesterov	LG, Convex	$O(1/t^2)$	N
Gradient	LG, Strongly	$O((1-\mu/L)^t)$	N
Nesterov	LG, Strongly	$O((1-\sqrt{\mu/L})^t)$	N
Newton	LH, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$	N^2

Convex Optimization Zoo

Algorithm	Assumptions	Rate	Grads
S(Subgrad)	LF, Convex	$O(1/\sqrt{t})$	1
S(Subgrad)	LF, Strongly	$O(1/t)$	1
SAG	LG, Convex	$O(1/t)$	1
SAG	LG, Strongly	$O((1 - \min\{\frac{\mu}{16L_i}, \frac{1}{8N}\})^t)$	1
Nesterov	Smoothed, Convex	$O(1/t)$	N
Gradient	LG, Convex	$O(1/t)$	N
Nesterov	LG, Convex	$O(1/t^2)$	N
Gradient	LG, Strongly	$O((1 - \mu/L)^t)$	N
Nesterov	LG, Strongly	$O((1 - \sqrt{\mu/L})^t)$	N
Newton	LH, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$	N^2

- L_i is the Lipschitz constant over all f_i' ($L_i \geq L$).
- SAG has a similar speed to the gradient method, but only looks at one training example per iteration.
- Recent work gives prox, ADMM, and memory-free variants.

Coordinate Descent Methods

- In coordinate descent methods we only update one variable:

$$x_j^+ = x_j - \alpha d.$$

- We can often cheaply perform a very precise line-search.

Coordinate Descent Methods

- In coordinate descent methods we only update one variable:

$$x_j^+ = x_j - \alpha d.$$

- We can often cheaply perform a very precise line-search.
- The *steepest descent* choice is $j = \arg\min_j \{\nabla_j f(x)\}$.

(but only efficient to calculate in some special cases)

- Choosing a random j has the same convergence rate.
- Faster rate if j sampled according to Lipschitz constants.

Coordinate Descent Methods

- In coordinate descent methods we only update one variable:

$$x_j^+ = x_j - \alpha d.$$

- We can often cheaply perform a very precise line-search.
- The *steepest descent* choice is $j = \arg\min_j \{\nabla_j f(x)\}$.

(but only efficient to calculate in some special cases)

- Choosing a random j has the same convergence rate.
- Faster rate if j sampled according to Lipschitz constants.
- Various extensions:
    - Accelerated version (may lose sparsity of update)
    - Projected coordinate descent (product constraints)
    - Frank-Wolfe coordinate descent (product constraints)
    - Proximal coordinate descent (separable non-smooth term)

(exact step size for ℓ_1-regularized least squares)

Convex Optimization Zoo

Algorithm	Assumptions	Rate
S(Subgrad)	LF, Convex	$O(1/\sqrt{t})$
S(Subgrad)	LF, Strongly	$O(1/t)$
SAG	LG, Convex	$O(1/t)$
SAG	LG, Strongly	$O((1 - \min\{\frac{\mu}{16L_i}, \frac{1}{8N}\})^t)$
CD-Uniform	LP, Convex	$O(1/t)$
CD-Uniform	LP, Strongly	$O((1 - \mu/L_1 P)^t)$
CD-Lipschitz	LP, Strongly	$O((1 - \mu/\sum_i L_i)^t)$
Nesterov	Smoothed, Convex	$O(1/t)$
Gradient	LG, Convex	$O(1/t)$
Nesterov	LG, Convex	$O(1/t^2)$
Gradient	LG, Strongly	$O((1 - \mu/L)^t)$
Nesterov	LG, Strongly	$O((1 - \sqrt{\mu/L})^t)$
Newton	LH, Strongly	$O(\prod_{i=1}^t \rho_t), \rho_t \to 0$

- $L_1 \geq L_2 \geq \ldots L_P$ are Lipschitz constants of the partials $\nabla_i f$ ($L_1 \leq L \leq PL_1$).

References

- A reference to start with for each part:
  - Part 1: *Convex Optimization* (Boyd and Vandenberghe)
  - Part 2: *Introductory Lectures on Convex Optimization* (Nesterov)
  - Part 3: *Convex Optimization Theory* (Bertsekas)
  - Part 4: *Efficient Methods in Convex Programming* (Nemirovski)
- E-mail me for the other references (mark.schmidt@sfu)
- Come talk to me in TASC 9404.
- For tutorial material and code: `http://www.di.ens.fr/~mschmidt/MLSS`
- Come join the MLRG: `http://www.di.ens.fr/~mschmidt/MLRG.html`