# Unconstrained Optimization Algorithms

Yinyu Ye
Department of Management Science and Engineering
Stanford University
Stanford, CA 94305, U.S.A.

http://www.stanford.edu/~yyye

Chapter 8.1-8.2, 8.4-8.5

# Taylor Expansion and Lipschitz Functions

$$\min f(\mathbf{x}) \quad \text{s.t.} \quad \mathbf{x} \in \mathbf{R}^n.$$

and we look for a KKT solution, that is, a solution $\mathbf{x}$ such that $\nabla f(\mathbf{x}) = \mathbf{0}$.

(If the function is strictly convex, then the solution is unique.)

**Taylor's (or the mean-value) theorem**: for some $\boldsymbol{\xi}$ between $\mathbf{x}$ and $\mathbf{y}$

First-Order Expansion: $f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\boldsymbol{\xi})^T(\mathbf{y} - \mathbf{x})$

Second-Order Expansion: $f(\mathbf{y}) = f(\mathbf{x}) + \nabla f(\mathbf{x})^T(y\text{-}x) + \frac{1}{2}(\mathbf{x}\text{-}\mathbf{y})^T \nabla^2 f(\boldsymbol{\xi})(\mathbf{y} - \mathbf{x})$

**Lipschitz Functions**: exists some β ≥ 0 such that

First-Order β-Lipschitz: for all $\mathbf{x}$ and $\mathbf{y}$ in the domain of $f$

$$| f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^T(\mathbf{y}\text{-}\mathbf{x}) | \leq \frac{\beta}{2} \|\mathbf{y} - \mathbf{x}\|^2$$

Second-Order β-Lipschitz: for all $\mathbf{x}$ and $\mathbf{y}$ in the domain of $f$

$$| f(\mathbf{y}) - f(\mathbf{x}) - \nabla f(\mathbf{x})^T(\mathbf{y}\text{-}\mathbf{x}) - \frac{1}{2}(\mathbf{x}\text{-}\mathbf{y})^T \nabla^2 f(\mathbf{x})(\mathbf{y} - \mathbf{x}) | \leq \frac{\beta}{3} \|\mathbf{y} - \mathbf{x}\|^3$$

# Solution Convergence and Convergence Speed

The key is to guarantee that the sequence of iterates generated by the algorithm converges to an element of the solution set of the problem. Let $\{\mathbf{x}^k\}$ be a sequence of iterates. Then we like $\{\mathbf{x}^k\}$ converges to $\mathbf{x}^*$ where $\mathbf{x}^*$ is an exact solution.

More precisely, for any small real number $\varepsilon > 0$ there exists a positive integer $K$ such that

$$\|\mathbf{x}^k - \mathbf{x}^*\| \le \varepsilon, \quad \text{for all } k \ge K.$$

or

$$\|\nabla f(\mathbf{x}^k)\| \le \varepsilon, \quad \text{for all } k \ge K.$$

or

$$f(\mathbf{x}^k) - f(\mathbf{x}^*) \le \varepsilon, \quad \text{for all } k \ge K.$$

The convergence speed of an iterative algorithm would be the total number of iterations needed to meet the $\varepsilon$-accuracy condition:

$O(1/\varepsilon^2)$, $O(1/\varepsilon)$, $O(1/\sqrt{\varepsilon})$, $O(\log(1/\varepsilon))$, $O(\log[\log(1/\varepsilon)])$

arithmetic linear/geometric quadratic

# Zero-Order: the Golden-Section Method I

Assume that the one variable function $f(x)$ is unimodal in interval $[a \ \ b]$, that is, for any point $x$ in interval $[a' \ b']$ such that $a \le a' < b' \le b$, we have that $f(x) \le \max\{ f(a'), f(b') \}$. How do we find $x^*$ (within an error tolerance $\epsilon$)? Again, without loss of generality, let $a = 0$ and $b = 1$.

## Golden-Section Method (0-Order Method)

1. Initialization: let $x_l = 0$, $x_r = 1$, and choose

   constant $0 < r < 0.5;$

2. Let two other points $x'_l = r(x_r - x_l)$ and

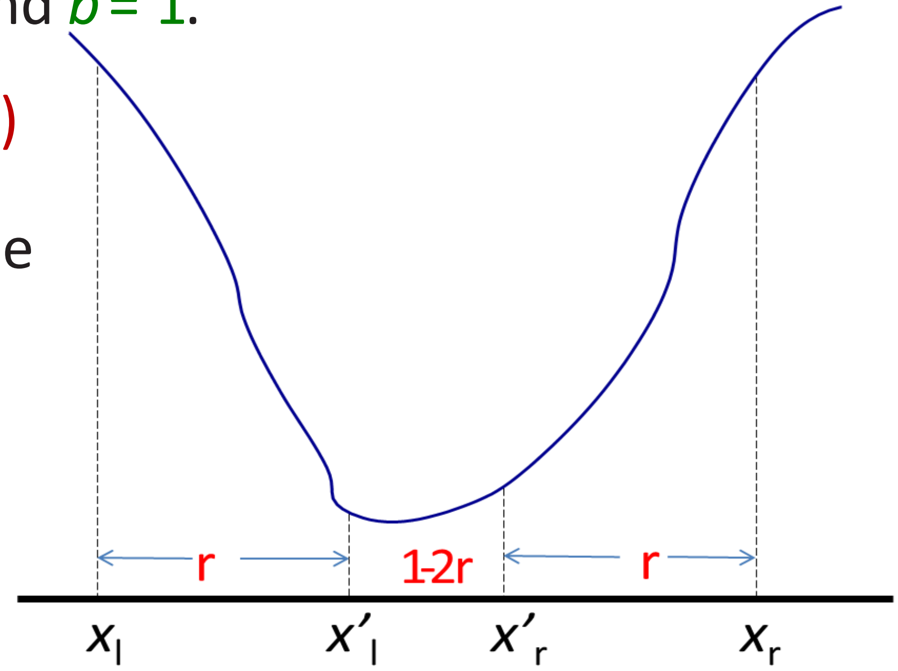
   $x'_r = (1 - r)(x_r - x_l)$ .

3. Update the triple points

$\{x_l = x_l, x_r = x'_r, x'_r = x'_l\}$ if $f(x'_l) < f(x'_r)$;

otherwise update the triple points $\{x_l = x'_l, x'_l = x'_r, x_r = x_r\}$;

4. Return to Step 2.

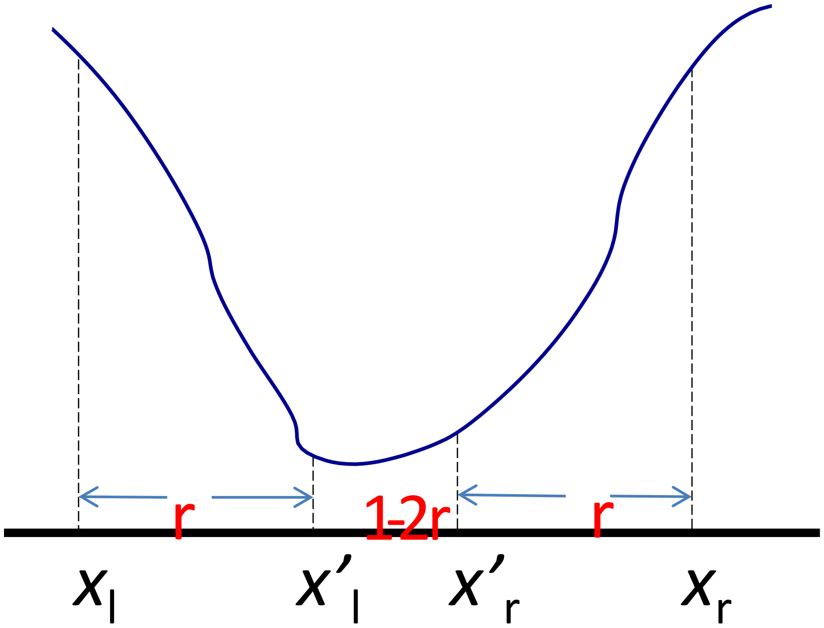


one variable minimization:

$\min \ f(x) \quad \text{s.t.} \quad a \le x \le b$

where the function is unimodal

# Zero-Order: the Golden-Section Method II



one variable minimization:

min f(x) s.t. $a \le x \le b$

where the function is unimodal

In either case of Step 3, the length of the new interval after one bisection step is $(1 - r)$. If we set $(1 - 2r)/(1 - r) = r$, then only one point needs to be recomputed; which leads to $r = 0.382$.

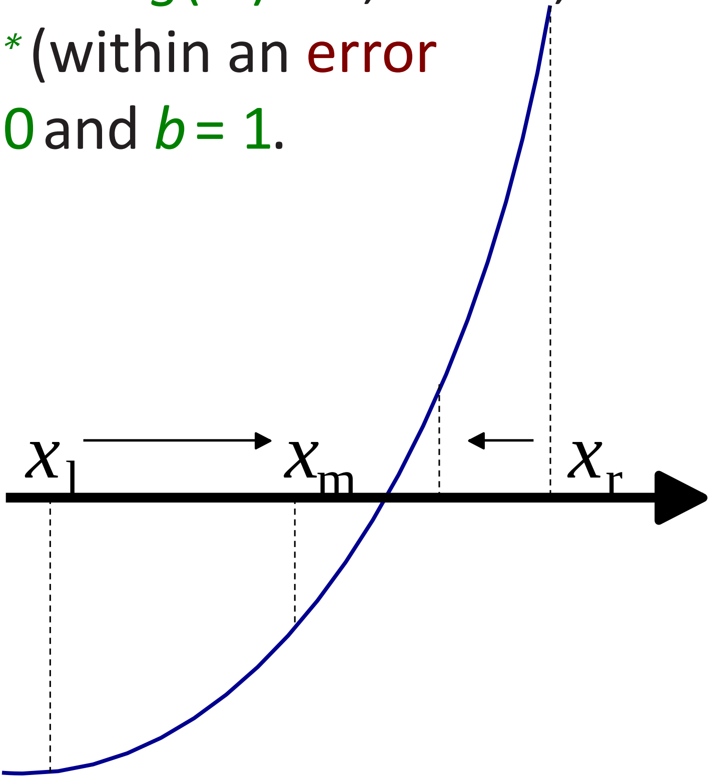
Then, the length of the containing interval is shrinking at rate 0.618 each step, so that the trial points converges to the exact minimizer $x^*$. Precisely, let $x^k$ = the kept middle point of the kth step of the method. Then

$$| x^k - x^* | \le (0.618)^k.$$

Thus, it is an $O(\log(1/\varepsilon))$ zero-order algorithm.

# First-Order: The Bisection Method I

For a one variable problem, an KKT point is the root of $g(x) := f'(x) = 0$. Assume we know an interval $[a \;\; b]$ such that $a < b$, and $g(a) \cdot g(b) < 0$. Then we know there exists an $x^*$, $a < x^* < b$, such that $g(x^*) = 0$; that is, interval $[a, b]$ contains a root. How do we find $x^*$ (within an error tolerance $\epsilon$ )? Without loss of generality, let $a = 0$ and $b = 1$.



## Bisection Method

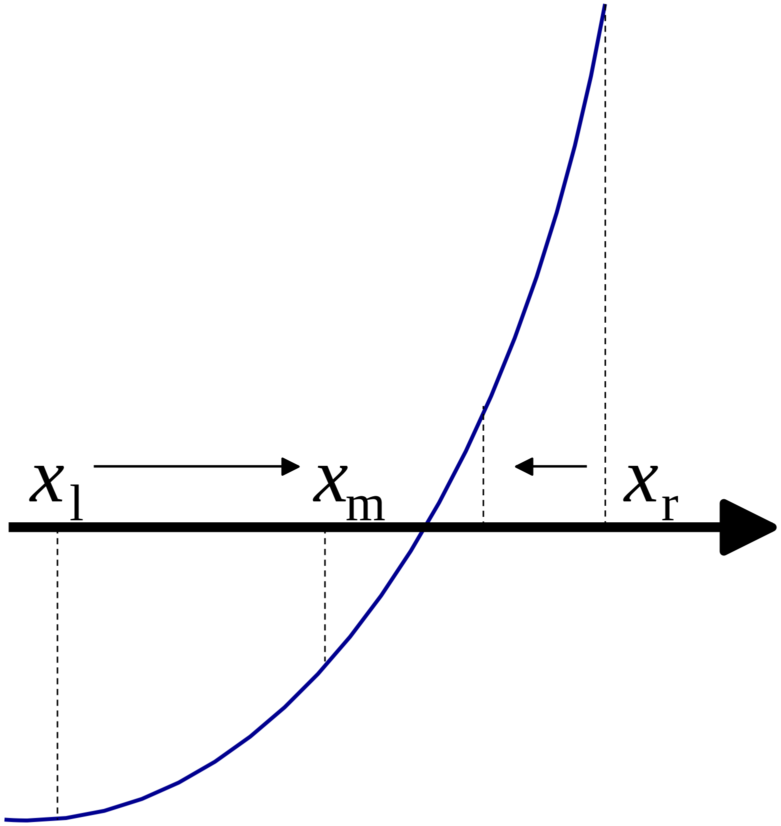
1. Initialization: let $x_l = a$, $\mathbf{x}_r = b$ ;
2. Let $x_m = (x_l + x_r)/2$ and evaluate $g(x_m)$.
3. If $g(x_m) = 0$ or $x_r - x_l < \varepsilon$ stop and

output $x^* = x_m$.

4. Otherwise, if $g(x_l) \cdot g(x_m) > 0$ set $x_l = x_m$; else set $x_r = x_m$; and return to Step 2.

What is the length of the new interval containing a root after one bisection step?

# The Bisection Method II



The length of the containing interval is halved each step, so that the trial points converges to an exact root $x^*$.

Precisely, let $x^k = x_m$ of the kth step of the method. Then

$$| x^k – x^* | \leq 2^{-k}.$$

Thus, it is an $O(\log(1/\varepsilon))$ first order algorithm.

# $2^{nd}$-Order: The Newton Method I

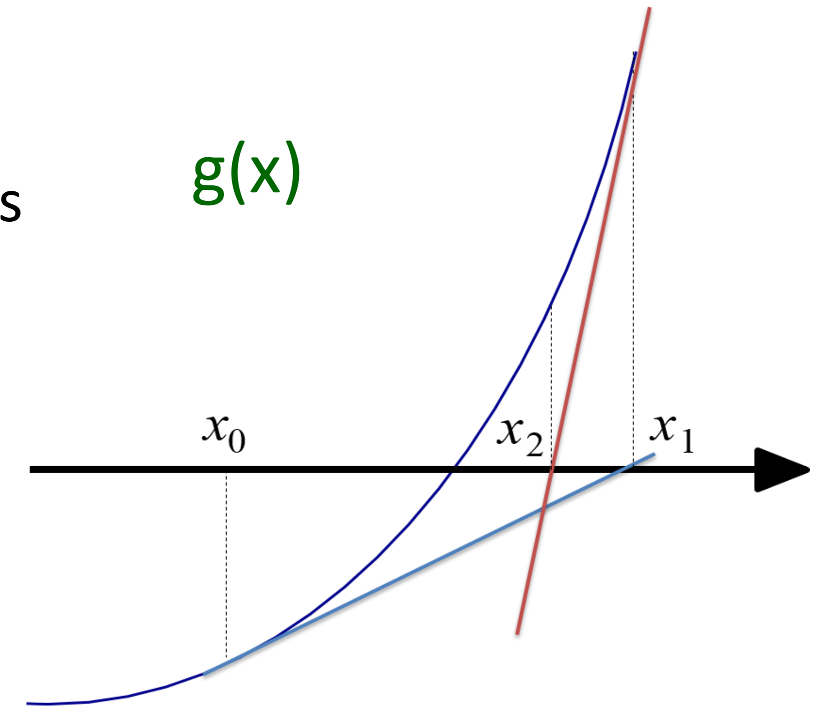
Again, for functions of a single real variable $x$, the KKT solution is the root of $g(x) := f'(x) = 0$.

When $f$ is twice continuously differentiable then $g$ is once continuously differentiable, Newton's method can be a very effective way to solve such equations and hence to locate a root of $g$.

Given a starting point $x^0$, the iterative process of the Newton method for finding the root is

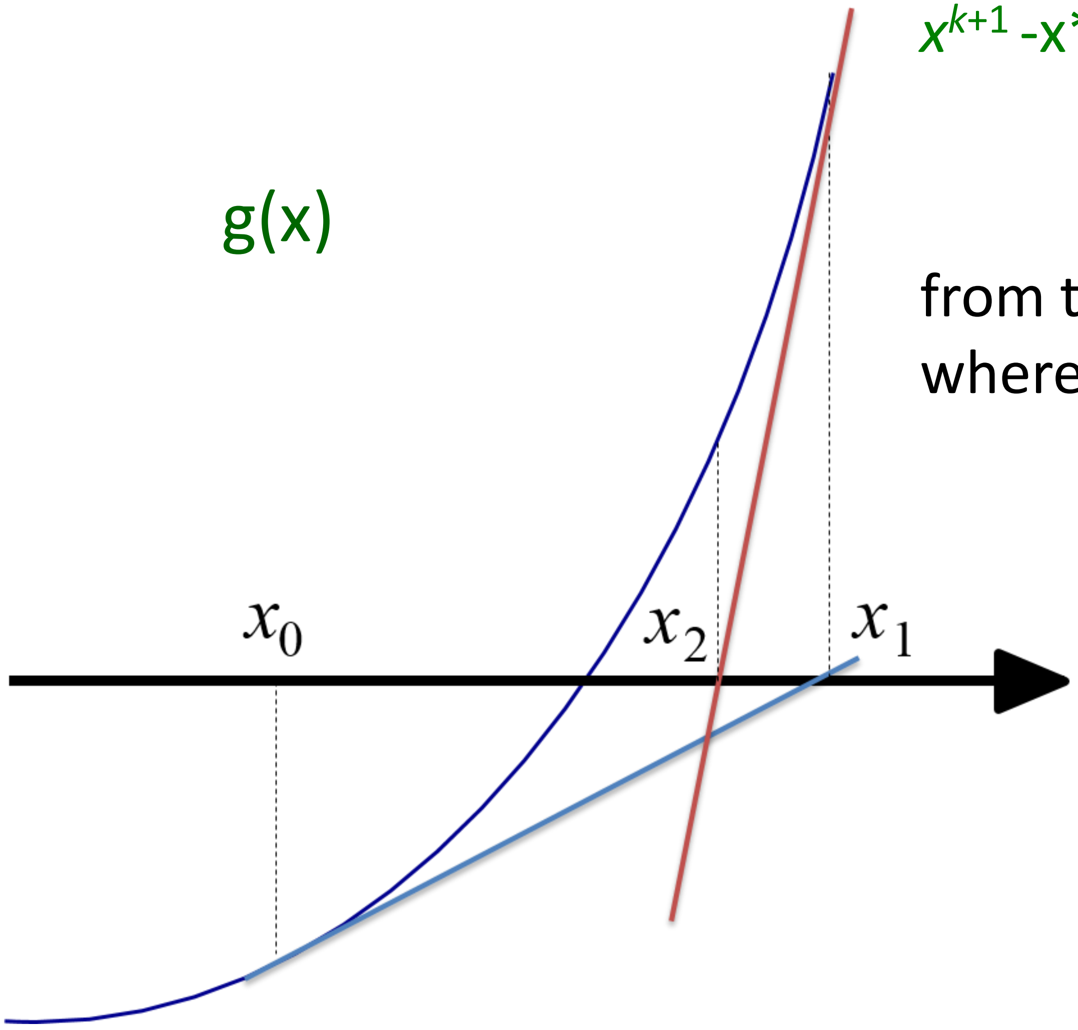
$$x^{k+1} = x^k - g(x^k)/g'(x^k)= x^k – f'(x^k)/f''(x^k)$$

where the iteration formula is well defined provided that $g'(x^k)=f''(x^k) \neq 0$ at each step.



This condition would hold if the starting point $x^0$ is sufficiently close to the root $x^*$ and $g'(x^*)=f''(x^*) \neq 0$.

# Convergence of the Newton Method



$$x^{k+1} = x^k - g(x^k)/g'(x^k) \rightarrow$$

$$x^{k+1} - x^* = x^k - x^* - g(x^k)/g'(x^k)$$

$$= [g(x^*) - g(x^k) - g'(x^k)(x^* - x^k)]/g'(x^k)$$

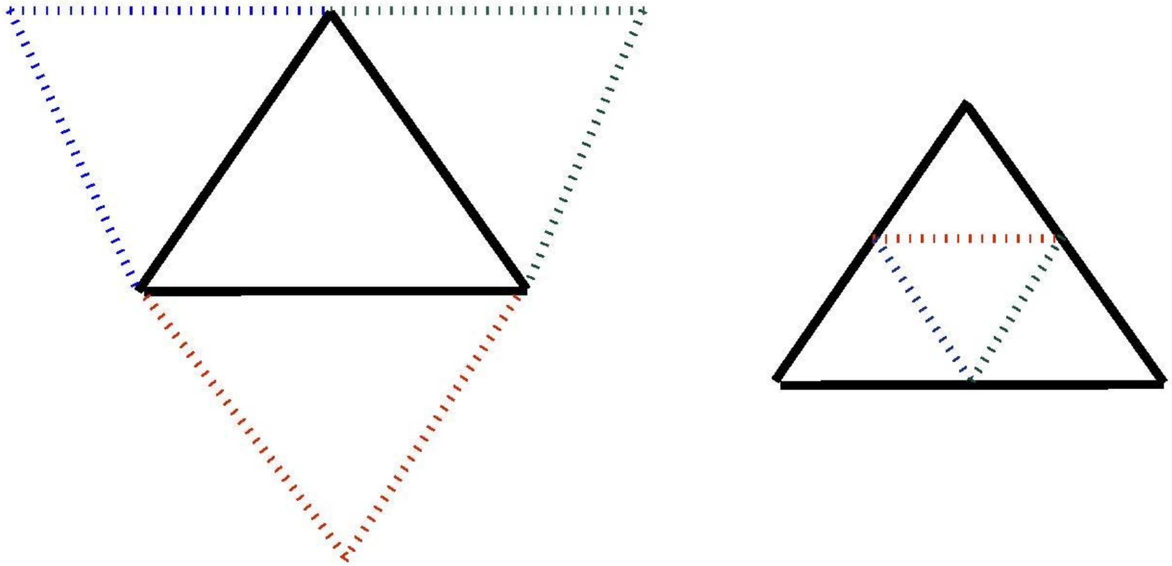
$$= \frac{1}{2}[g''(\xi)/g'(x^k)](x^k - x^*)^2$$

from the first-order Taylor theorem, where $\xi$ is a value between $x^k$ and $x^*$.

Thus, when $x^k$ is sufficiently close to $x^*$ we would have $|x^{k+1} - x^*| \le c|x^k - x^*|^2$ . for some constant c. Furthermore, if $c|x^k - x^*| \le 1$, $c|x^{k+1} - x^*| \le (c|x^k - x^*|)^2 < 1$, and this error is squared at each iteration, which leads to O(log[log(1/ϵ)]) local convergence speed.

# Zero-Order: The Simplicial and/or Forward-Difference

min $f(\mathbf{x})$
s.t. $\mathbf{x} \in \mathbf{R}^n$.



**Simplicial Method:**

1. Start with a Triangle/Simplex with d+1 corner points and their objective function values.
2. Reflection: Compute other d+1 corner points each of them is an additional corner point of a reflection simplex. If a point is better than its counter point, then the reflection simplex is an improved simplex, and select the most improved simplex and go to Step 1; otherwise go to Step 3.
3. Contraction: Compute the d+1 middle-face points and subdivide the simplex into smaller d+1 simplexes, keep the simplex with the lowest sum of the d+1 function values at corners, and go to Step 1.

**Forward-Difference:** compute numerical partial derivatives (ZeroorderNLP.m)

# Unconstrained Minimization of Differentiable Functions With Multiple Variables

$$\min f(\mathbf{x}) \text{ s.t. } \mathbf{x} \in \mathbf{R}^n.$$

and we look for a KKT solution, that is, a solution $\mathbf{x}$ such that $\nabla f(\mathbf{x}) = \mathbf{0}$.

## General Descent Method

1) Test for convergence: If the $\| \nabla f(\mathbf{x}^k) \| \leq \varepsilon$ termination conditions are satisfied at $\mathbf{x}^k$ is an accepted solution and stop. Otherwise, go to Step 2.

2) Let $\mathbf{d}^k$ be any descent direction. Then Compute a step size, say $\alpha_k$ such that $f(\mathbf{x}^k + \alpha_k \mathbf{d}^k) < f(\mathbf{x}^k)$.

(This may necessitate a one-dimensional or line search).

3) Define the new iterate by setting

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$$

and return to Step 1.

# The Steepest Descent/(sub)Gradient Method: First-Order

First-Order β-Lipschitz $f$: One can choose $\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$ and stepsize to be fixed for all iterations at $\beta^{-1}$, that is,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \beta^{-1} \nabla f(\mathbf{x}^k)$$

Then, the following theorem can be proved.

Theorem 1. Let $f(\mathbf{x})$ be a convex function and admit a minimizer $\mathbf{x}^*$, and it satisfies the first-order β-Lipschitz condition. Then

$$f(\mathbf{x}^k) - f(\mathbf{x}^*) \le 2\beta||\mathbf{x}^0 - \mathbf{x}^*||/k.$$

Note that the algorithm uses a fixed step size and information of the immediate early iterate. This is an $O(\varepsilon^{-1})$ first order algorithm.

Theorem 2. Let $f(\mathbf{x})$ admit a minimizer $\mathbf{x}^*$ and it satisfies the first-order β-Lipschitz condition. Then in at most $2\beta(f(x^0)-f(x^*))/\varepsilon^2$ steps

$$\|\nabla f(\mathbf{x}^k)\| \le \epsilon.$$

# Proof of Theorem 2 for SDM

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \beta^{-1} \nabla f(\mathbf{x}^k)$$

$$f(\mathbf{x}^{k+1}) - f(\mathbf{x}^*) = f(\mathbf{x}^k - \beta^{-1} \nabla f(\mathbf{x}^k)) - f(\mathbf{x}^*)$$

$$\leq f(\mathbf{x}^k) - \beta^{-1} \nabla f(\mathbf{x}^k)^T \nabla f(\mathbf{x}^k) + \frac{\beta}{2} \beta^{-2} \| \nabla f(\mathbf{x}^k) \|^2 - f(\mathbf{x}^*)$$

$$= f(\mathbf{x}^k) - \beta^{-1} \| \nabla f(\mathbf{x}^k) \|^2 + \frac{1}{2} \beta^{-1} \| \nabla f(\mathbf{x}^k) \|^2 - f(\mathbf{x}^*)$$

$$= f(\mathbf{x}^k) - f(\mathbf{x}^*) - \frac{1}{2} \beta^{-1} \| \nabla f(\mathbf{x}^k) \|^2$$

If $\| \nabla f(\mathbf{x}^k) \| > \epsilon$ during the iterative process, then

$$0 \leq f(\mathbf{x}^k) - f(\mathbf{x}^*) < f(\mathbf{x}^0) - f(\mathbf{x}^*) - \frac{k}{2} \beta^{-1} \epsilon^2$$

But $\| \nabla f(\mathbf{x}^k) \| > \epsilon$ cannot hold during the entire process when

$$k \leq 2\beta(f(x^0) - f(x^*))/\varepsilon^2$$

since then the right-hand-side becomes 0 or negative, which is a contradiction.

**Remark**: $\nabla f(\mathbf{x}^k)$ can be a sub-gradient vector such as in piece-wise linear minimization

# The Accelerated Steepest Descent Method

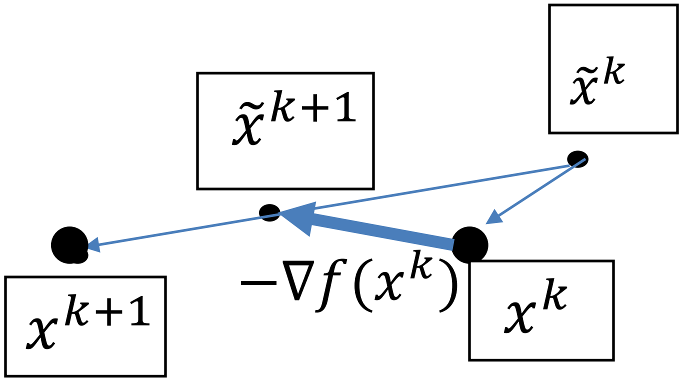
There is an accelerated steepest descent method for minimizing a β-smooth convex function such that

$$f(\mathbf{x}^k) - f(\mathbf{x}^*) \leq 2\beta||\mathbf{x}^0 - \mathbf{x}^*||/k^2.$$

$$\lambda^0 = 0, \lambda^1 = 1, \lambda^{k+1} = \frac{1+\sqrt{1+4(\lambda^k)^2}}{2}, \alpha^k = \frac{1-\lambda^k}{\lambda^{k+1}}$$

$$\tilde{x}^{k+1} = x^k - \frac{1}{\beta}\nabla f(x^k), x^{k+1} = (1-\alpha^k)\tilde{x}^{k+1} + \alpha^k\tilde{x}^k$$

Note that the algorithm uses information of two immediate early iterates, and $\alpha^k < 0$. This is an $O(\varepsilon^{-0.5})$ first order algorithm.



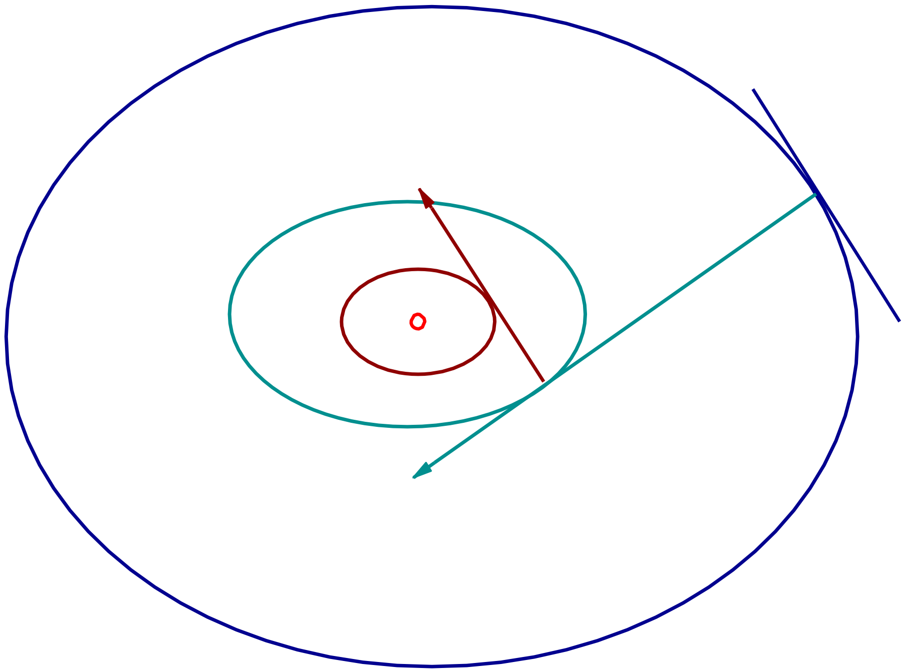
# The Steepest Descent Method with Varying Step-sizes I

Again, the method chooses $\mathbf{d}^k = -\nabla f(\mathbf{x}^k)$ as the search direction at each step and set $\alpha_k = \arg\min f(\mathbf{x}^k + \alpha\mathbf{d}^k)$.

Then the new iterate is defined as $\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$.

QP Example: Let $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + 0.5\mathbf{x}^T Q\mathbf{x}$ where $Q \in \mathbf{R}^{n\times n}$ is symmetric and positive definite. Then, $\nabla f(\mathbf{x}^k) = \mathbf{c} + Q\mathbf{x}^k$, and the step size has a close form formula

$$\alpha_k = -\frac{(\mathbf{c}^T + (\mathbf{x}^k)^{\mathrm{T}}Q)\mathbf{d}^k}{(\mathbf{d}^k)^{\mathrm{T}}Q\mathbf{d}^k} = \frac{(\mathbf{d}^k)^{\mathrm{T}}\mathbf{d}^k}{(\mathbf{d}^k)^{\mathrm{T}}Q\mathbf{d}^k}$$



Initial Interval Finding for Line Search: Start with $[0, \alpha = 1]$; if function value is lower, double $\alpha$ and check again till the value to start increasing.

# The Steepest Descent Method with Varying Step-sizes II

There is another steepest descent method for minimizing a smooth convex function with a smart and varying step-size selection, called the Barzilai-Borwein (BB) method:

Let $\Delta_{\boldsymbol{x}}^{\boldsymbol{k}} = \mathbf{x}^k - \mathbf{x}^{k\text{-}1}$ and $\Delta_{\boldsymbol{g}}^{\boldsymbol{k}} = \nabla f(\mathbf{x}^k) - \nabla f(\mathbf{x}^{k\text{-}1})$;

Set step-size $\alpha^k = \|\Delta_{\boldsymbol{x}}^{\boldsymbol{k}}\|^2/(\Delta_{\boldsymbol{x}}^{\boldsymbol{k}} \bullet \Delta_{\boldsymbol{g}}^{\boldsymbol{k}})$

Then let $\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha^k \nabla f(\mathbf{x}^k)$:

# Convergence of the SDM for General Functions

Theorem Let $f : \mathbf{R}^n \rightarrow R$ be given. For some given point $\mathbf{x}^0 \in \mathbf{R}^n$, let the level set $X^0 = \{\mathbf{x} \in \mathbf{R}^n : f(\mathbf{x}) \leq f(x^0)\}$ be bounded. Assume further that $f$ is continuously differentiable on the convex hull of $X^0$. Let $\{\mathbf{x}^k\}$ be the sequence of points generated by the steepest descent method (with line search) initiated at $\mathbf{x}^0$. Then every accumulation point of $\{\mathbf{x}^k\}$ is a KKT point of $f$ .

Remarks: 1) The steepest descent method initiated at any point of the bounded level set $X^0$ will converge to a stationary point of $f$ . In other words, it is not necessary to start the process in a neighborhood of the (unknown) optimal solution. This property is called global convergence.

2) To solve strictly convex quadratic minimization, the algorithm has a geometric convergence speed O(log(1/ϵ)) where the constant depending on $Q$.

3) Only to a stationary point, not local minimizer:

min $x^2-y^2+y^4$ starting from ($x=1,y=0$).

# Newton's Method: Second-Order

Finding a root vector of $n$ variables from $n$ nonlinear equations

$$\nabla f(\mathbf{x}) = 0.$$

## Newton's Method for Minimization

1) Test for convergence: If the $\| \nabla f(\mathbf{x}^k) \| \le \varepsilon$ termination conditions are satisfied at $\mathbf{x}^k$ is an accepted solution and stop. Otherwise, go to Step 2.

2) Let $\mathbf{d}^k = -(\nabla^2 f(\mathbf{x}^k))^{-1} \nabla f(\mathbf{x}^k)$, and let step size $\alpha_k = 1$;

3) Define the new iterate by setting

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \mathbf{d}^k$$

and return to Step 1.

QP Example: Let $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + 0.5\mathbf{x}^T Q\mathbf{x}$ where $Q \in \mathbf{R}^{n\times n}$ is symmetric and positive definite. Then, $\nabla f(\mathbf{x}^0) = \mathbf{c} + Q\mathbf{x}^0$, $\nabla^2 f(\mathbf{x}^0) = Q$ so that $\mathbf{x}^1 = \mathbf{x}^0 - Q^{-1}(\mathbf{c} + Q\mathbf{x}^0) = -Q^{-1}\mathbf{c}$, the exact KKT point.

# Local Convergence Theorem of Newton's Method

Newton's Method can be applied to any system of nonlinear equations $F(\mathbf{x}) = \mathbf{0}$ where vector functions are continuously differentiable and admit a root vector $\mathbf{x}^*$, and the Jacobian $\nabla F(\mathbf{x})$ is nonsingular everywhere. Then provided that $\|\mathbf{x}^0 - \mathbf{x}^*\|$ is sufficiently small, the iterate sequence generated by Newton's method: $\mathbf{x}^{k+1} = \mathbf{x}^k - \nabla F(\mathbf{x})^{-1}F(\mathbf{x}^k)$, converges quadratically to root $\mathbf{x}^*$.

What is quadratic convergence: there is a constant $c$ such that

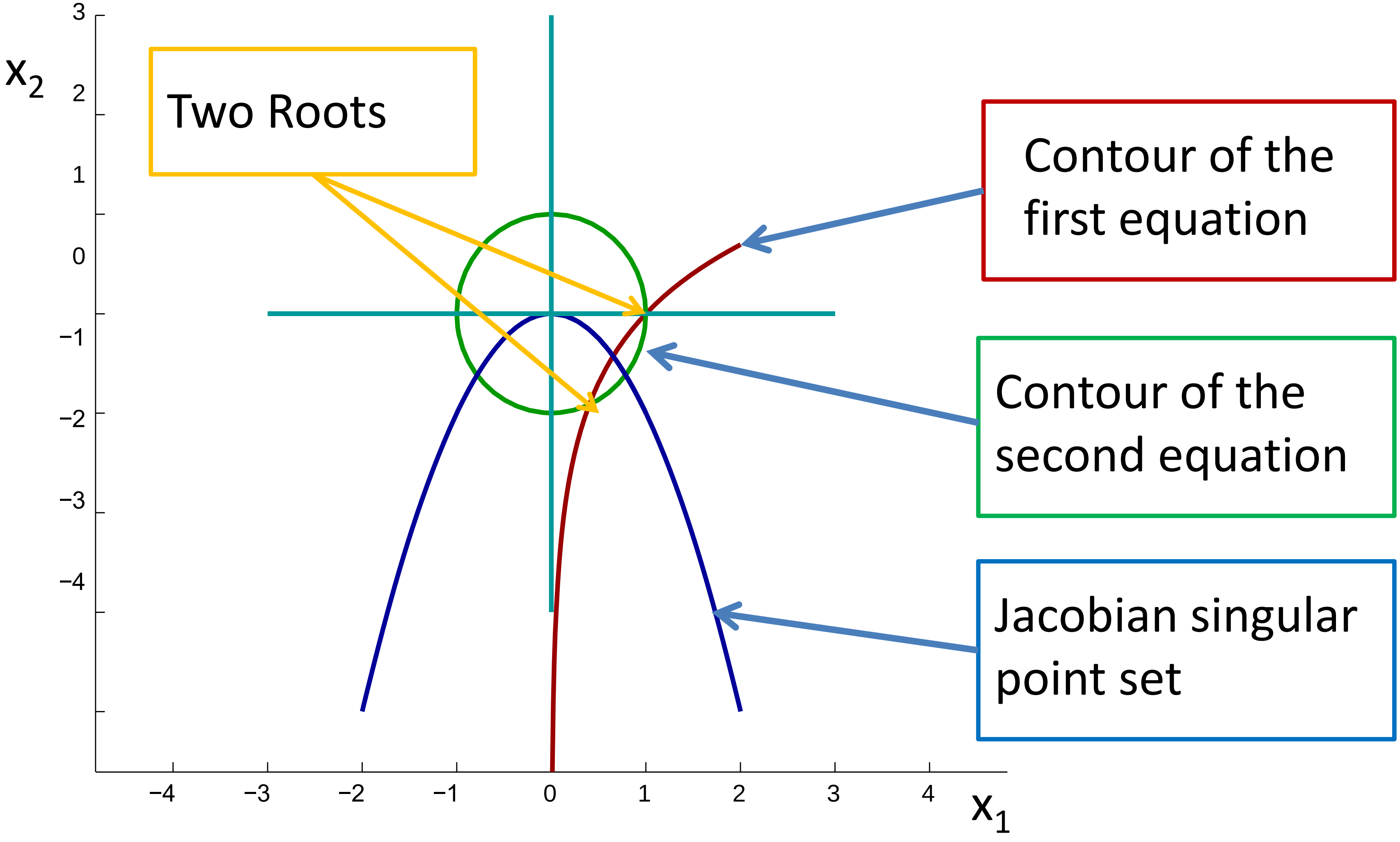
$$\|\mathbf{x}^{k+1} - \mathbf{x}^*\| \leq c \|\mathbf{x}^k - \mathbf{x}^*\|^2$$

System Equation Example:

$$F(x_1, x_2) = \begin{pmatrix} x_2 - \log(x_1) \\ x_1^2 + x_2^2 - 1 \end{pmatrix}$$

$$\nabla F(x_1, x_2) = \begin{pmatrix} -1/x_1 & 1 \\ 2x_1 & 2x_2 \end{pmatrix}$$

# Convergence Illustration of Newton’s Method



Two Roots

Contour of the first equation

Contour of the second equation

Jacobian singular point set

Depending on the starting point, the method may converge to one of the two roots, or it may fail to converge at all.

# Newton’s Method Matlab Code for the Example

$$F(x_1, x_2) = \begin{pmatrix} x_2 - \log(x_1) \\ x_1^2 + x_2^2 - 1 \end{pmatrix}$$

$$\nabla F(x_1, x_2) = \begin{pmatrix} -1/x_1 & 1 \\ 2x_1 & 2x_2 \end{pmatrix}$$

```
%
F=0*x;
F(1)=x(2)-log(x(1));
F(2)=x(1)^2+x(2)^2-1;
J=[-1/x(1) 1; 2*x(1) 2*x(2)];
x=x-J\F;
norm(F)
```

# Newton’s Method May not Converge

Newton’s Method may not converge even when the Jacobian matrix is invertible everywhere: consider to find the root of one-variable function

$$f(x) = \begin{cases} \log(1+x) & \text{if } x \geq 0 \\ -\log(1-x) & \text{otherwise.} \end{cases}$$

$$f'(x) = \begin{cases} \dfrac{1}{1+x} & \text{if } x \geq 0 \\ \dfrac{1}{1-x} & \text{otherwise.} \end{cases}$$

Try starting from x ≥ 4.

# Modified Newton's Method

## Modified Newton's Method for Minimization of f(**x**)

1) Test for convergence: If the $\| \nabla f(\mathbf{x}^k) \| \le \varepsilon$ termination conditions are satisfied at $\mathbf{x}^k$ is an accepted solution and stop. Otherwise, go to Step 2.

2) Let $\mathbf{d}^k = -(\mu I+(1-\mu)\nabla^2 f(\mathbf{x}^k))^{-1}\nabla f(\mathbf{x}^k)$ for a positive constant $0 \le \mu \le 1$, and let step size $\alpha_k = \arg\min f(\mathbf{x}^k + \alpha\mathbf{d}^k)$.

3) Define the new iterate by setting

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha_k \mathbf{d}^k$$

and return to Step 1.

Note that when $\mu$ is large enough so that $\mu I+(1-\mu)\nabla^2 f(\mathbf{x}^k)$ is positive definite, then $\mathbf{d}^k$ becomes a descent direction. By carefully choosing it and step-size, then the algorithm stops in $O(1/\varepsilon^{1.5})$ iterations.

There are also Quasi Newton and Conjugate Gradient methods that are learning the Hessian during the iterative process; see Chapters 9 and 10.