# CS257 Linear and Convex Optimization

## Lecture 10

Bo Jiang

John Hopcroft Center for Computer Science
Shanghai Jiao Tong University

November 9, 2020

# Recap

Strong convexity. $f$ is $m$-strongly convex if

- $f(x) - \frac{m}{2}\|\boldsymbol{x}\|^2$ is convex
- first-order condition

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

- second-order condition

$$\nabla^2 f(\boldsymbol{x}) \succeq m\boldsymbol{I} \iff \lambda_{\min}(\nabla^2 f(\boldsymbol{x})) \geq m$$

Convergence. For $m$-strongly convex and $L$-smooth $f$ with minimum $\boldsymbol{x}^*$, gradient descent with constant step size $t \in (0, \frac{1}{L}]$ satisfies

$$f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*) \leq \frac{L(1-mt)^k}{m}[f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*)]$$

Condition number. For $\boldsymbol{Q} \succ \boldsymbol{O}$,

$$\kappa(\boldsymbol{Q}) = \frac{\lambda_{\max}(\boldsymbol{Q})}{\lambda_{\min}(\boldsymbol{Q})}$$

Well-/Ill-conditioned if $\kappa(\boldsymbol{Q})$ is small/large $\implies$ fast/slow convergence.

# Today

- exact line search
- backtracking line search
- Newton's method

# Step Size

Gradient descent

$$\boxed{\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - t_k \nabla f(\boldsymbol{x}_k)}$$

- constant step size: $t_k = t$ for all $k$
- exact line search: optimal $t_k$ for each step

$$t_k = \arg\min_s f(\boldsymbol{x}_k - s\nabla f(\boldsymbol{x}_k))$$

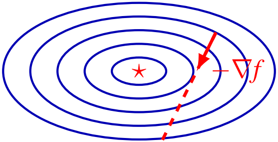
- backtracking line search (Armijo's rule): $t_k$ satisfies

$$f(\boldsymbol{x}_k) - f(\boldsymbol{x}_k - t_k \nabla f(\boldsymbol{x}_k)) \geq \alpha t_k \|\nabla f(\boldsymbol{x}_k)\|_2^2$$

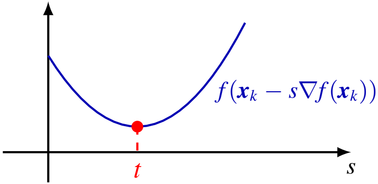
for some given $\alpha \in (0, 1)$.

# Exact Line Search

initialization $\boldsymbol{x} \leftarrow \boldsymbol{x}_0 \in \mathbb{R}^n$
**while** $\|\nabla f(\boldsymbol{x})\| > \delta$ **do**
  $t \leftarrow \arg\min_s f(\boldsymbol{x} - s\nabla f(\boldsymbol{x}))$
  $\boldsymbol{x} \leftarrow \boldsymbol{x} - t\nabla f(\boldsymbol{x})$
**end while**
**return** $\boldsymbol{x}$




level curves of $f(x_1, x_2) = \frac{x_1^2}{4} + x_2^2$




Note. Often impractical; used only if the inner minimization is cheap.

# Exact Line Search for Quadratic Functions

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{b}^T\boldsymbol{x}, \quad \boldsymbol{Q} \succ \boldsymbol{O}$$

- gradient at $\boldsymbol{x}_k$ is $\boldsymbol{g}_k = \nabla f(\boldsymbol{x}_k) = \boldsymbol{Q}\boldsymbol{x}_k + \boldsymbol{b}$
- second-order Taylor expansion is exact for quadratic functions,

$$\begin{aligned} h(t) &= f(\boldsymbol{x}_k - t\boldsymbol{g}_k) \\ &= f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T(-t\boldsymbol{g}_k) + \frac{1}{2}(-t\boldsymbol{g}_k)^T \nabla^2 f(\boldsymbol{x}_k)(-t\boldsymbol{g}_k) \\ &= \left(\frac{1}{2}\boldsymbol{g}_k^T\boldsymbol{Q}\boldsymbol{g}_k\right) t^2 - \boldsymbol{g}_k^T\boldsymbol{g}_k t + f(\boldsymbol{x}_k) \end{aligned}$$

- minimizing $h(t)$ yields best step size

$$t_k = \frac{\boldsymbol{g}_k^T\boldsymbol{g}_k}{\boldsymbol{g}_k^T\boldsymbol{Q}\boldsymbol{g}_k}$$

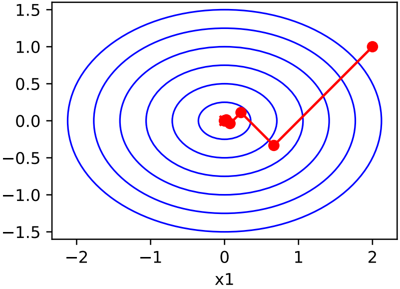
- update step

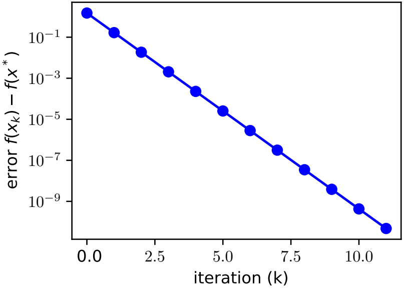
$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - t_k\boldsymbol{g}_k = \boldsymbol{x}_k - \frac{\boldsymbol{g}_k^T\boldsymbol{g}_k}{\boldsymbol{g}_k^T\boldsymbol{Q}\boldsymbol{g}_k}\boldsymbol{g}_k$$

# Example

$$f(x_1, x_2) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} = \frac{\gamma}{2}x_1^2 + \frac{1}{2}x_2^2, \quad \boldsymbol{Q} = \text{diag}\{\gamma, 1\}$$

Well-conditioned. $\gamma = 0.5$, $\boldsymbol{x}_0 = (2, 1)^T$




Fast convergence.

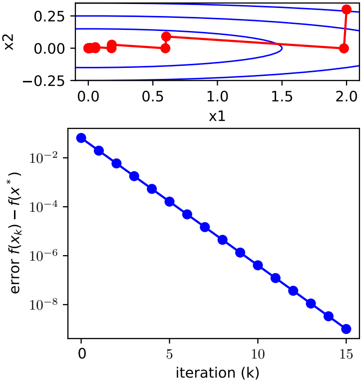
Note. Successive gradient directions are always orthogonal, as

$$0 = h'(t_k) = -\nabla f(\boldsymbol{x}_k - t_k \nabla f(\boldsymbol{x}_k))^T \nabla f(\boldsymbol{x}_k) = -\nabla f(\boldsymbol{x}_{k+1})^T \nabla f(\boldsymbol{x}_k)$$

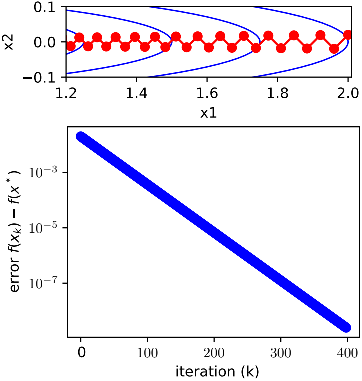
# Example (cont'd)

$$f(x_1, x_2) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} = \frac{\gamma}{2}x_1^2 + \frac{1}{2}x_2^2, \quad \boldsymbol{Q} = \text{diag}\{\gamma, 1\}$$

Ill-conditioned. $\gamma = 0.01$, convergence rate depends on initial point




$\boldsymbol{x}_0 = (2, 0.3)$, fast convergence




$\boldsymbol{x}_0 = (2, 0.02)$, slow convergence

# Convergence Analysis

Theorem. If $f$ is $m$-strongly convex and $L$-smooth, and $\boldsymbol{x}^*$ is a minimum of $f$, then the sequence $\{\boldsymbol{x}_k\}$ produced by gradient descent with exact line search satisfies

$$f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*) \leq \left(1 - \frac{m}{L}\right)^k [f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*)]$$

Notes.

- $0 \leq 1 - \frac{m}{L} < 1$, so $\boldsymbol{x}_k \to \boldsymbol{x}^*$ and $f(\boldsymbol{x}_k) \to f(\boldsymbol{x}^*)$ exponentially fast
- The number of iterations to reach $f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*) \leq \epsilon$ is $O(\log \frac{1}{\epsilon})$. For $\epsilon = 10^{-p}$, $k = O(p)$, linear in the number of significant digits.
- The convergence rate depends on the condition number $L/m$ and can be slow if $L/m$ is large. When close to $\boldsymbol{x}^*$, we can estimate $L/m$ by $\kappa(\nabla f^2(\boldsymbol{x}^*))$.

# Proof

1. By the quadratic upper bound for $L$-smooth functions,

$$f(\boldsymbol{x}_k - t\nabla f(\boldsymbol{x}_k)) \leq f(\boldsymbol{x}_k) - t\|\nabla f(\boldsymbol{x}_k)\|^2 + \frac{Lt^2}{2}\|\nabla f(\boldsymbol{x}_k)\|^2 \triangleq q(t)$$

2. Minimizing over $t$ in step 1,

$$f(\boldsymbol{x}_{k+1}) = \min_t f(\boldsymbol{x}_k - t\nabla f(\boldsymbol{x}_k)) \leq \min_t q(t) = q(\frac{1}{L}) = f(\boldsymbol{x}_k) - \frac{1}{2L}\|\nabla f(\boldsymbol{x}_k)\|^2$$

3. By $m$-strong convexity,

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T(\boldsymbol{x} - \boldsymbol{x}_k) + \frac{m}{2}\|\boldsymbol{x} - \boldsymbol{x}_k\|^2 \triangleq \hat{f}(\boldsymbol{x})$$

4. Minimizing over $\boldsymbol{x}$ in step 3,

$$f(\boldsymbol{x}^*) = \min_{\boldsymbol{x}} f(\boldsymbol{x}) \geq \min_{\boldsymbol{x}} \hat{f}(\boldsymbol{x}) = \hat{f}(\boldsymbol{x}_k - \frac{1}{m}\nabla f(\boldsymbol{x}_k)) = f(\boldsymbol{x}_k) - \frac{1}{2m}\|\nabla f(\boldsymbol{x}_k)\|^2$$

5. By 4, $\|\nabla f(\boldsymbol{x}_k)\|^2 \geq 2m[f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*)]$. Plugging into 2,

$$f(\boldsymbol{x}_{k+1}) - f(\boldsymbol{x}^*) \leq \left(1 - \frac{m}{L}\right)[f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*)]$$

# Backtracking Line Search

Exact line search is often expensive and not worth it. Suffices to find a good enough step size. One way to do so is to use backtracking line search, aka Armijo's rule.

## Gradient descent with backtracking line search

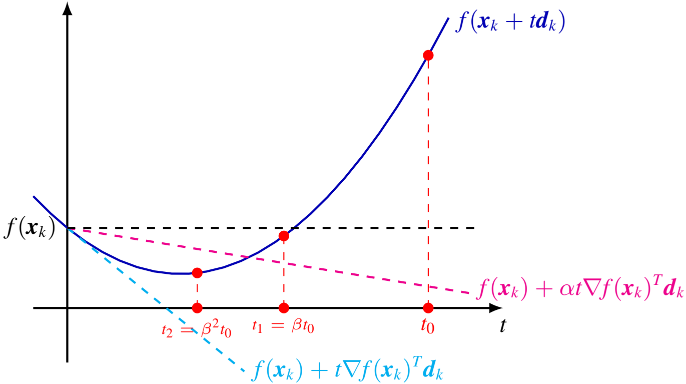
initialization $\boldsymbol{x} \leftarrow \boldsymbol{x}_0 \in \mathbb{R}^n$
**while** $\|\nabla f(\boldsymbol{x})\| > \delta$ **do**
    $t \leftarrow t_0$
    **while** $f(\boldsymbol{x} - t\nabla f(\boldsymbol{x})) > f(\boldsymbol{x}) - \alpha t \|\nabla f(\boldsymbol{x})\|_2^2$ **do**
        $t \leftarrow \beta t$
    **end while**
    $\boldsymbol{x} \leftarrow \boldsymbol{x} - t\nabla f(\boldsymbol{x})$
**end while**
**return** $\boldsymbol{x}$

$\alpha \in (0, 1)$ and $\beta \in (0, 1)$ are constants. Armijo used $\alpha = \beta = 0.5$

Values suggested in [BV]: $\alpha \in [0.01, 0.3]$, $\beta \in [0.1, 0.8]$

Note. For general $\boldsymbol{d}$, use condition $f(\boldsymbol{x} + t\boldsymbol{d}) > f(\boldsymbol{x}) + \alpha t \nabla f(\boldsymbol{x})^T \boldsymbol{d}$

# Backtracking Line Search (cont'd)



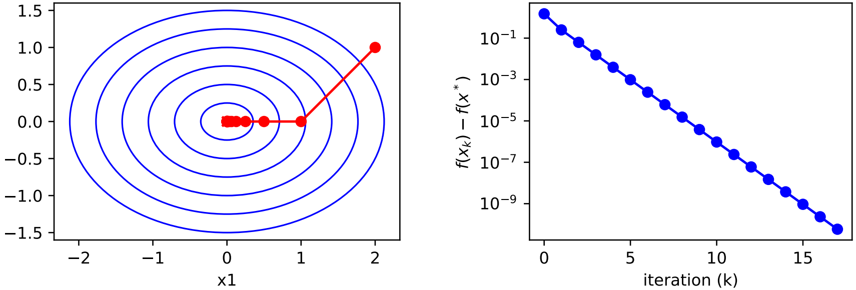

- $\nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}_k < 0$ for descent direction $\boldsymbol{d}_k$
- start from some "large" step size $t_0$ ([BV] uses $t_0 = 1$)
- reduce step size geometrically until decrease is "large enough"

$$\underbrace{f(\boldsymbol{x}_k) - f(\boldsymbol{x}_k + t\boldsymbol{d}_k)}_{\text{actual decrease in function value}} \geq \alpha \times \underbrace{t|\nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}_k|}_{\text{decrease along tangent line}}$$

# Example

$$f(x_1, x_2) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} = \frac{\gamma}{2}x_1^2 + \frac{1}{2}x_2^2, \quad \boldsymbol{Q} = \text{diag}\{\gamma, 1\}$$

Well-conditioned. $\gamma = 0.5$, $\boldsymbol{x}_0 = (2, 1)^T$

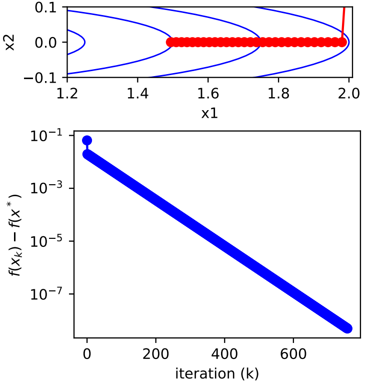



Fast convergence.

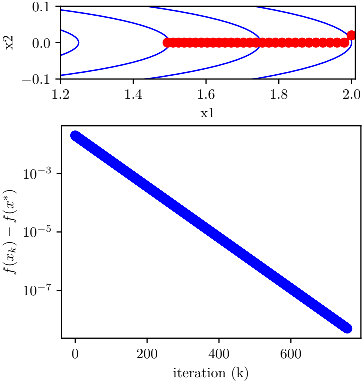
# Example (cont'd)

$$f(x_1, x_2) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} = \frac{\gamma}{2}x_1^2 + \frac{1}{2}x_2^2, \quad \boldsymbol{Q} = \text{diag}\{\gamma, 1\}$$

**Ill-conditioned.** $\gamma = 0.01$




$\boldsymbol{x}_0 = (2, 0.3)$, slow convergence




$\boldsymbol{x}_0 = (2, 0.02)$, slow convergence

# Convergence Analysis

**Theorem.** If $f$ is $m$-strongly convex and $L$-smooth, and $\boldsymbol{x}^*$ is a minimum of $f$, then the sequence $\{\boldsymbol{x}_k\}$ produced by gradient descent with backtracking line search satisfies

$$f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*) \leq c^k [f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*)]$$

where

$$c = 1 - \min\left\{2m\alpha t_0, \frac{4m\beta\alpha(1-\alpha)}{L}\right\}$$

**Notes.**

- $c \in (0,1)$, as
$$\frac{4m\beta\alpha(1-\alpha)}{L} \leq \frac{\beta m}{L} \leq \beta < 1$$
so $\boldsymbol{x}_k \to \boldsymbol{x}^*$ and $f(\boldsymbol{x}_k) \to f(\boldsymbol{x}^*)$ exponentially fast
- Number of iterations to reach $f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*) \leq \epsilon$ is $O(\log \frac{1}{\epsilon})$. For $\epsilon = 10^{-p}$, $k = O(p)$, linear in the number of significant digits.

# Proof

The inner loop terminates with a step size bounded from below.

1. By the quadratic upper bound for $L$-smooth functions,

$$f(\boldsymbol{x}_k - t\nabla f(\boldsymbol{x}_k)) \leq f(\boldsymbol{x}_k) - t(1 - \frac{Lt}{2})\|\nabla f(\boldsymbol{x}_k)\|^2$$

2. The inner loop terminates for sure if

$$-t(1 - \frac{Lt}{2})\|\nabla f(\boldsymbol{x}_k)\|^2 \leq -\alpha t\|\nabla f(\boldsymbol{x}_k)\|^2 \implies t \leq \frac{2(1-\alpha)}{L}$$

3. The step size in backtracking line search satisfies

$$t_k \geq \eta \triangleq \min\left\{t_0, \frac{2\beta(1-\alpha)}{L}\right\}$$

   - $t_k = t_0$ if Armijo's condition is satisfied by $t_0$
   - otherwise, $\frac{t_k}{\beta} > \frac{2(1-\alpha)}{L}$, since the inner loop did not terminate at $\frac{t_k}{\beta}$

# Proof (cont'd)

Now we look at the outer loop

4. By Armijo's condition in the inner loop,

$$f(\boldsymbol{x}_{k+1}) = f(\boldsymbol{x}_k - t_k \nabla f(\boldsymbol{x}_k)) \leq f(\boldsymbol{x}_k) - \alpha t_k \|\nabla f(\boldsymbol{x}_k)\|^2$$

5. By 3 and 4,

$$f(\boldsymbol{x}_{k+1}) - f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*) - \alpha\eta \|\nabla f(\boldsymbol{x}_k)\|^2$$

6. By step 4 of slide 9,

$$\|\nabla f(\boldsymbol{x}_k)\|^2 \geq 2m[f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*)]$$

7. By 5 and 6,

$$f(\boldsymbol{x}_{k+1}) - f(\boldsymbol{x}^*) \leq (1 - 2m\alpha\eta)[f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*)] = c[f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*)]$$

   so

$$f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*) \leq c^k [f(\boldsymbol{x}_0) - f(\boldsymbol{x}^*)]$$

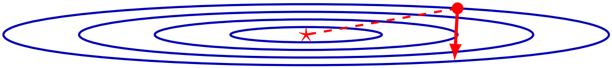
# Better Descent Direction

Gradient descent uses first-order information (i.e. gradient),

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - t_k \nabla f(\boldsymbol{x}_k)$$

Locally $-\nabla f(\boldsymbol{x}_k)$ is the max-rate descending direction, but globally it may not be the "right" direction.

Example. For $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T \boldsymbol{Q}\boldsymbol{x}$ with $\boldsymbol{Q} = \text{diag}\{0.01, 1\}$, optimum is $\boldsymbol{x}^* = \boldsymbol{0}$.



The negative gradient is

$$-\nabla f(\boldsymbol{x}) = -\boldsymbol{Q}\boldsymbol{x} = -(0.01x_1, x_2)^T$$

quite different from the "right" descent direction $\boldsymbol{d} = -\boldsymbol{x}$. Note

$$\boldsymbol{d} = -\boldsymbol{Q}^{-1}\nabla f(\boldsymbol{x}) = -[\nabla^2 f(\boldsymbol{x})]^{-1}\nabla f(\boldsymbol{x})$$

With second-order information (i.e. Hessian), we hope to do better.

# Newton's Method

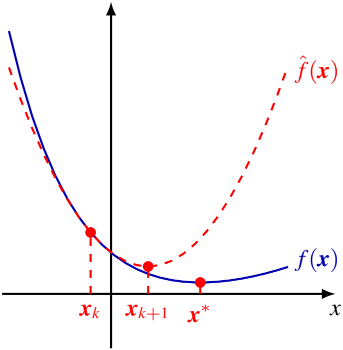
By second-order Taylor expansion,

$$f(\boldsymbol{x}) \approx \hat{f}(\boldsymbol{x}) \triangleq f(\boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k)^T(\boldsymbol{x} - \boldsymbol{x}_k) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_k)^T \nabla^2 f(\boldsymbol{x}_k)(\boldsymbol{x} - \boldsymbol{x}_k)$$

Minimizing quadratic approximation $\hat{f}$,

$$\nabla \hat{f}(\boldsymbol{x}) = \nabla^2 f(\boldsymbol{x}_k)(\boldsymbol{x} - \boldsymbol{x}_k) + \nabla f(\boldsymbol{x}_k) = \boldsymbol{0}$$
$$\Longrightarrow \boldsymbol{x} = \boldsymbol{x}_k - [\nabla^2 f(\boldsymbol{x}_k)]^{-1} \nabla f(\boldsymbol{x}_k)$$

provided $\nabla^2 f(\boldsymbol{x}_k) \succ \boldsymbol{O}$.




Newton step

$$\boxed{\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - [\nabla^2 f(\boldsymbol{x}_k)]^{-1} \nabla f(\boldsymbol{x}_k)}$$

Note. If $f$ is quadratic, then $f = \hat{f}$, and Newton's method gets to the optimum in a single step starting from any $\boldsymbol{x}_0$.

# Newton's Method (cont'd)

```
initialization x ← x_0 ∈ R^n
while ‖∇f(x)‖ > δ do
    x ← x − [∇²f(x)]^{-1} ∇f(x)
end while
return x
```

Note. As in the case of gradient descent, other stopping criteria can be used. [BV] uses $\nabla f(\boldsymbol{x})[\nabla^2 f(\boldsymbol{x})]^{-1}\nabla f(\boldsymbol{x}) > \delta$.

The Newton step is a special case of $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + t_k \boldsymbol{d}_k$ with

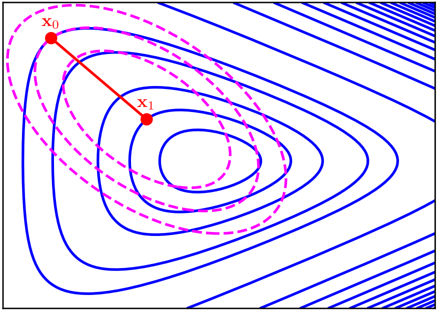
- Newton direction $\boldsymbol{d}_k = -[\nabla^2 f(\boldsymbol{x}_k)]^{-1}\nabla f(\boldsymbol{x}_k)$
- constant step size $t_k = 1$

For $\nabla^2 f(\boldsymbol{x}_k) \succ \boldsymbol{O}$, the Newton direction is a descent direction

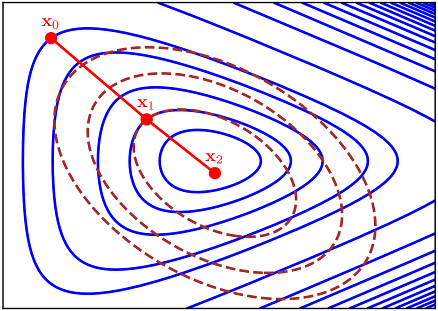
$$\nabla f(\boldsymbol{x}_k)^T \boldsymbol{d}_k = -\nabla f(\boldsymbol{x}_k)^T [\nabla^2 f(\boldsymbol{x}_k)]^{-1} \nabla f(\boldsymbol{x}_k) < 0 \quad \text{if } \nabla f(\boldsymbol{x}_k) \neq \boldsymbol{0}$$

# Newton's Method (cont'd)

The magenta curves are the level curves of the quadratic approximation of $f$ at $\boldsymbol{x}_0$




The brown curves are the level curves of the quadratic approximation of $f$ at $\boldsymbol{x}_1$.



# Example

$$f(x_1, x_2) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$

Newton step at $\boldsymbol{x}_0 = (-2, 1)^T$.

- gradient

$$\nabla f(\boldsymbol{x}_0) = e^{-0.1} \begin{pmatrix} e^{x_1+3x_2} + e^{x_1-3x_2} - e^{-x_1} \\ 3e^{x_1+3x_2} - 3e^{x_1-3x_2} \end{pmatrix} \Bigg|_{\boldsymbol{x}=\boldsymbol{x}_0} = \begin{pmatrix} -4.22019458 \\ 7.36051909 \end{pmatrix}$$

- Hessian

$$\nabla^2 f(\boldsymbol{x}_0) = e^{-0.1} \begin{pmatrix} e^{x_1+3x_2} + e^{x_1-3x_2} + e^{-x_1} & 3e^{x_1+3x_2} - 3e^{x_1-3x_2} \\ 3e^{x_1+3x_2} - 3e^{x_1-3x_2} & 9e^{x_1+3x_2} + 9e^{x_1-3x_2} \end{pmatrix} \Bigg|_{\boldsymbol{x}=\boldsymbol{x}_0}$$

$$= \begin{pmatrix} 9.1515943 & 7.36051909 \\ 7.36051909 & 22.19129872 \end{pmatrix}$$

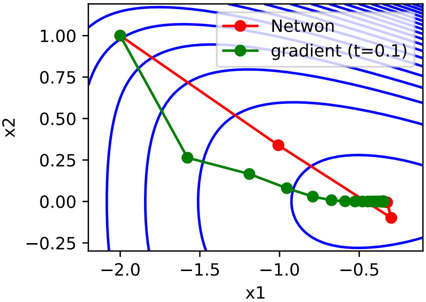
- Newton step

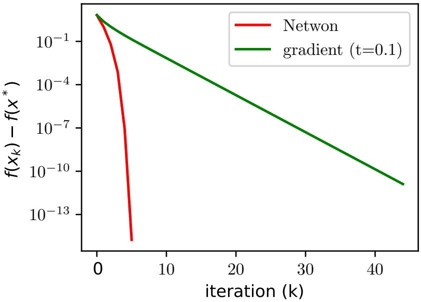
$$\boldsymbol{x}_1 = \boldsymbol{x}_0 - [\nabla^2 f(\boldsymbol{x}_0)]^{-1} \nabla f(\boldsymbol{x}_0) = \begin{pmatrix} -1.00725064 \\ 0.33903509 \end{pmatrix}$$

# Example (cont'd)

$$f(x_1, x_2) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$

Solution using Newton's method and gradient descent with constant step size $0.1$. Initial point $\boldsymbol{x}_0 = (-2, 1)^T$.




- Newton's method takes a more "direct" path
- Newton's method requires much fewer iterations, but each iteration is more expensive

# Connection to Root Finding

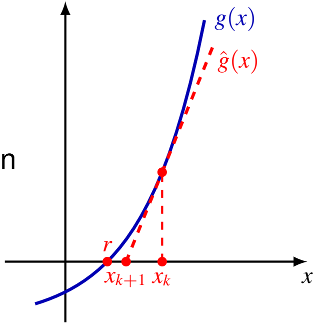
Newton's method is originally an algorithm for solving $g(x) = 0$.

By the first-order Taylor expansion,

$$g(x) \approx \hat{g}(x) \triangleq g(x_k) + g'(x_k)(x - x_k)$$

Use the root of $\hat{g}(x)$ as the next approximation

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x_k)}$$




Example (computing $\sqrt{C}$). $\sqrt{C}$ is a root of $g(x) = x^2 - C$. Newton's method yields

$$x_{k+1} = x_k - \frac{x_k^2 - C}{2x_k} = \frac{1}{2}\left(x_k + \frac{C}{x_k}\right)$$

For $x_0 > 0$, $x_k$ converges to $\sqrt{C}$.

# Connection to Root Finding (cont'd)

Back to the optimization problem,

$$\min_{x} f(x)$$

The optimal solution $x^*$ satisfies

$$f'(x^*) = 0$$

Letting $g = f'$ in Newton's root finding algorithm,

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} = x_k - [f''(x_k)]^{-1} f'(x_k)$$

In $n$-dimension, $f' \to \nabla f, f'' \to \nabla^2 f$. We want to solve

$$\nabla f(\boldsymbol{x}^*) = \mathbf{0}$$

Newton's algorithm becomes

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - [\nabla^2 f(\boldsymbol{x}_k)]^{-1} \nabla f(\boldsymbol{x}_k)$$