# Gradient Methods III: Convergence Speed

Ruoyu Sun

# Questions for Last Time

- **Q1:** You use GD with constant stepsize to solve an unconstrained differentiable optimization problem, but it doesn't converge.
  1) What are the possible reasons?
  (i)
  (ii)
  2) How to make it work?

- **Q2** You use GD with diminishing stepsize to solve a problem. Recall the conditions $\sum_r \alpha_r = \infty$ and $\alpha_r \to 0$.

  This time, you set a lower bound $0.01$; e.g., $\alpha_r = 1/r + 0.01$.

  Will this algorithm converge?

- **Q3** Use GD to solve $\min_x x^6$. How to ensure your algorithm will convergence to stationary points?

# Questions for Last Time

- **Q1:** You use GD with constant stepsize to solve an unconstrained differentiable optimization problem, but it doesn't converge.
  1) What are the possible reasons?
  **Answer:** (i) This is a maximization problem.
  (ii) The objective is not $L$-smooth for any $L$.
  (iii) The stepsize is set to too large.

  2) How to make it work?
  **Answer:** For (i), switch to gradient ascent. For (ii), try line search rules. For (iii), tune down stepsize.

- **Q2** You use GD with diminishing stepsize to minimize $f(\mathbf{x})$. Recall the conditions $\sum_r \alpha_r = \infty$ and $\alpha_r \to 0$.

  This time, you set a lower bound $0.01$ and use $\alpha_r = 1/r + 0.01$.

  Will this algorithm converge?

  **Answer**: Depending on the Lipschitz-smoothness. If $f$ is $L$-smooth with $L < 2/0.01 = 200$, then the algorithm converges to stationary points.

- **Q3** Use GD to solve $\min_x x^6$. How to ensure your algorithm will convergence to stationary points?
  **Answer**: Use line search, such as Armijor rule.

# Today

- Convergence Rate Analysis of GD
- After today's course, you will be able to
  - Describe the convergence rate of GD for strongly convex, convex and nonconvex problems
  - Analyze the convergence rate for quadratic problems

    Optional: Analyze the convergence rate of strongly convex problems
  - Explain why preprocessing data is useful
- **Advanced**: Analyzing a problem starting from simple cases.

# Outline

# What is Convergence Rate Analysis

**Goal:** find an "$\epsilon$-approximate solution":

- For convex functions: $\epsilon$-optimal solution in $\{\mathbf{x}_\epsilon \mid f(\mathbf{x}_\epsilon) - f^* \leq \epsilon\}$
- For nonconvex functions: consider $\epsilon$-stationary solution in $\{\mathbf{x}_\epsilon \mid \|\nabla f(\mathbf{x}_\epsilon)\| \leq \epsilon\}$
- **Convergence Rate**:
  1. Measures the number of iterations required to get an $\epsilon$-approximate solution'
  2. Important measure for evaluating algorithms in big data applications
- Question: What determines the convergence rate?

# Convergence v.s. Convergence Speed Analysis

**Convergence** (convergence to stat-pt or global-min)

1. Sanity check
2. Minimal requirement of any reasonable algorithm

**Convergence speed.**

- **Asymptotic convergence rate**: local analysis, assuming already close to a solution
  1. characterize algorithm behavior when $\#$ of iterations go to infinity
  2. Linear rate/Superlinear rate/Sublinear rate
- **Non-asymptotic convergence rate (Iteration complexity)**
  1. Describes global behavior of the algorithm
  2. Focus of today

# Illustration


Figure: What Determines Convergence Rate?

- How steep the mountain is (environment; **formulation**)
- Route you pick; your speed (your choice; **algorithm**)

# Toy Example in Regression: Data

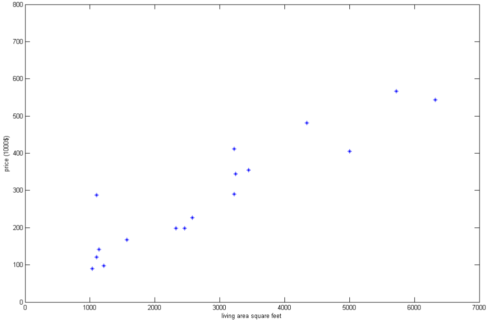
- Let's take a look at a simple example in predicting the house price giving the living areas

| Living Area ($\mathrm{feet}^2$) | Price (1000$) |
|---|---|
| 5719 | 567 |
| 3241 | 345 |
| 1139 | 141 |
| 1572 | 167 |
| 1101 | 287 |
| 2576 | 227 |
| $\vdots$ | $\vdots$ |

- Let's build a linear regression model and use GD to solve the problem

# Toy Example in Regression: Optimization Problem

- Data points $x_1, \ldots, x_n$ (total of $n = 17$ data points)
- Construct the regression model: variables $w_1$ (slope), $w_0$ (intercept); $P = 2$
- Data matrix $X \in \mathbb{R}^{2\times 17}$ $(P \times n)$; first row the area data, second row all $1$
- True output $y \in \mathbb{R}^{17\times 1}$ $(n \times 1)$: housing price.
- Solve the following problem $\min \frac{1}{2}\|X^T w - y\|^2$



# Toy Example in Regression: First Try

- **First Try**: Scale the data so that all the areas are multiplied by $0.01$, and all the prices are multiplied by $0.1$ – scaling the data
- Run GD for $1000$ iterations; initial $w = (10, 50)$

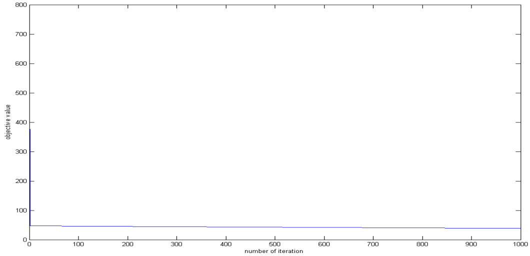



Figure: The decrease of objective function

Error seems quite large.

# First try: fitting

- Final fitting: far from optimum.
- So the issue is “not converged yet”

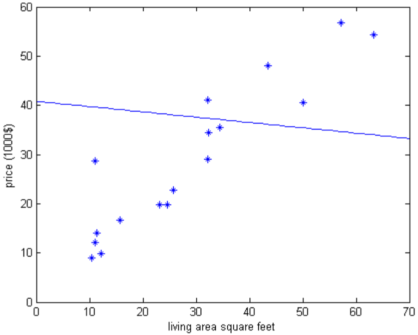



Figure: The final fitting.

Eigenvalues of $XX^T$ ($P \times P$, i.e., $2 \times 2$) are $0.0004$ and $1.856$.
Hessian is ill-conditioned!

# Toy Example in Regression: Second Try

- **Second Try**: Scale the data so that all the areas are multiplied by $0.0001$, and all the prices are multiplied by $0.1$ – scaling the data
- Run GD for $1000$ iterations; initial $(10, 50)$

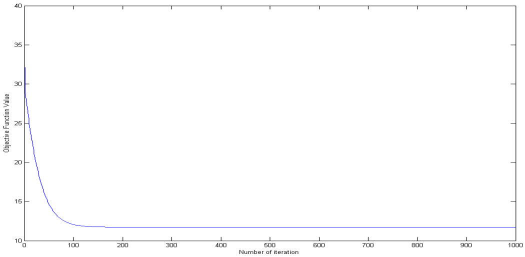



Figure: The decrease of objective function

Why does it work now?

Eigenvalues of $XX^T$ are $0.4$ and $18.45$. Hessian is well-conditioned!

# Second try: fitting

- Final fitting:

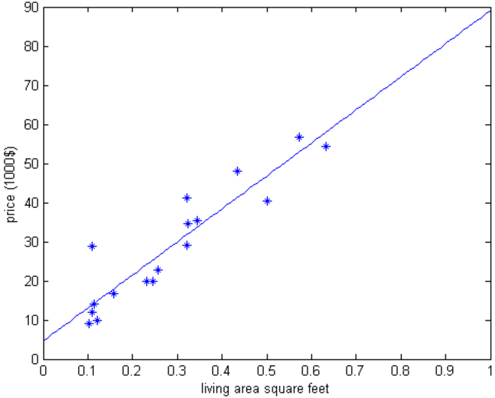



Figure: The final fitting.

# Data Preprocessing: Normalization

- **ill-conditioning**: the condition number is very large
- Can arise when the variables are scaled incoherently
  Example:
  1. feature 1: area, unit $m^2$, nominal value $120$
  2. featuer 2: age, unit "day", nominal value $6000$
  3. feature 3: $\#$ of bedrooms, nominal value $3$
- "Data preprocessing": preprocess the data so that the data matrix has a better property (e.g. better condition number)
  - Suppose $X = [x[1], x[2], \ldots, x[n]] \in \mathbb{R}^{d \times n}$
  - Each column represents a sample
  - Each row represents a feature
  - **Normalization**: Scale each row so that each row has unit norm
- **Claim:**(informal) Normalizing data can reduce condition number (in most cases), which makes GD faster.

# Data Preprocessing: Centering

**Example**: Predict which university to get in:

- feature 1: score, $X_1^T = [630, 650, 640, 660]$
- After normalization: $(0.488, 0.503, 0.496, 0.511)$; quite close

**Centering**: substract each entry by the mean of all data

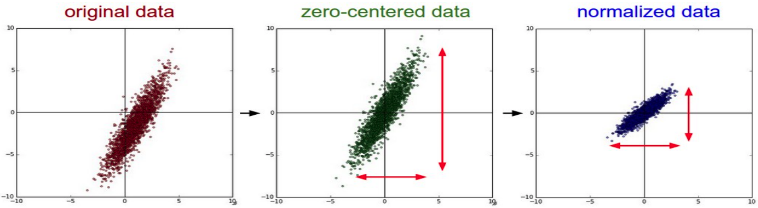
E.g. $X_1^T = [630, 650, 640, 660]$

- After centering: $-15, -5, 5, 15$.
- Further after normalization: $-0.67, -0.22, 0.22, 0.67$

Claim (informal): Centering can also reduce condition number (in most cases), which makes GD faster.

Exercise: compare matrix with entries in $Unif[0, 1]$ and matrix with entries in $Unif[-0.5, 0.5]$

# Data preprocessing: Figure Illustration




Common data preprocessing pipeline. Left: Original toy, 2-dimensional input data. Middle: The data is zero-centered by subtracting the mean in each dimension. The data cloud is now centered around the origin. Right: Each dimension is additionally scaled by its standard deviation. The red lines indicate the extent of the data - they are of unequal length in the middle, but of equal length on the right.

Figure: Data preprocessing pipeline. Source: Stanford CS231n course notes.

# Story

- **Story**: When I learned Torch, 1st "trick": **scale data properly**.

  It looks magical first; one day, I realize it should be explainable by optimization theory

  - Step 1: Scaling data is reducing condition number
  - Step 2: Condition number is critical for convergence speed

- **Quora**: When does logistic regression not converge?

  **Top Answer**: Common reasons:

  1. Code/Optimization method has bug.

  2. Learning rate too large

  3. Feature not normalized (values of different features have totally different scale, just ran into today).

# Outline

# Analysis of 2-dim Example

- Start from simple case $\min_{x_1,x_2} \frac{1}{2}(Lx_1^2 + x_2^2)$.
- $\nabla f(x) = \begin{pmatrix} Lx_1 \\ x_2 \end{pmatrix}$, $\nabla^2 f(x) = \begin{pmatrix} L & 0 \\ 0 & 1 \end{pmatrix}$, eigenvalues $L, 1$
- GD: $x^+ = x - \alpha \begin{pmatrix} Lx_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} (1-\alpha L)x_1 \\ (1-\alpha)x_2 \end{pmatrix}$
- Pick stepsize $\alpha = 1/L$, then

$$x_1^+ = 0$$
$$x_2^+ = (1 - \frac{1}{L})x_2$$

- Convergence speed depends on $|1 - \frac{1}{L}|$

# Analysis of Quadratic Problems: Diagonal Case

- Next, consider $\min_{\mathbf{x}\in\mathbb{R}^n} \frac{1}{2}\mathbf{x}^T Q\mathbf{x}$.
  For simplicity, assume $Q$ is symmetric positive-definite.

- GD: $\mathbf{x}^+ = \mathbf{x} - \alpha\nabla f(\mathbf{x}) = (I - \alpha Q)\mathbf{x}$.

  **Diagonal case**: $Q = \text{diag}(\lambda_1, \ldots, \lambda_n)$, $\lambda_1 \geq \cdots \geq \lambda_n$.

$$\mathbf{x}^+ = \left( I - \alpha \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix} \right) \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{pmatrix} = \begin{pmatrix} (1-\alpha\lambda_1)x_1 \\ (1-\alpha\lambda_2)x_2 \\ \cdots \\ (1-\alpha\lambda_n)x_n \end{pmatrix}$$

  We get independent sequences $x_i^+ = (1-\alpha\lambda_i)x_i, i = 1, \ldots, n.$
  Convergence speed of $\{x_i^r\}_{r=0}^{\infty}$ depends on $|1-\alpha\lambda_i|$.

  **Claim**: $\frac{\|x^r\|}{\|x^0\|} \leq \max_{i\in[n]} |1-\alpha\lambda_i|^r$.

# General Convex Quadratic Problem

- What about general quadratic problem $\min_{\mathbf{x}} \mathbf{x}^T Q\mathbf{x} + 2\mathbf{b}^T\mathbf{x}$?
- $\mathbf{x}^+ = \mathbf{x} - \alpha(Q\mathbf{x} + \mathbf{b})$.

  Suppose $x^*$ is one optimal solution, then it satisfies
  $\mathbf{x}^* = \mathbf{x}^* - \alpha(Q\mathbf{x} + \mathbf{b})$.

  Then $\mathbf{x}^+ - \mathbf{x}^* = (\mathbf{x} - \mathbf{x}^*) - \alpha Q(\mathbf{x} - \mathbf{x}^*) = (I - \alpha Q)(\mathbf{x} - \mathbf{x}^*)$.

  Eigenvalues of $I - \alpha Q$ are $1 - \alpha\lambda_i, i = 1, 2, \ldots, n$.
  Thus $\|\mathbf{x}^+ - \mathbf{x}^*\| \leq \max_i |1 - \alpha\lambda_i| \|\mathbf{x} - \mathbf{x}^*\|$.

  **Claim**: $\frac{\|\mathbf{x}^r - \mathbf{x}^*\|}{\|\mathbf{x}^0 - \mathbf{x}^*\|} \leq \max_{i \in [n]} |1 - \alpha\lambda_i|^r$.

  **Corollary**: If stepsize $\alpha = 1/\lambda_1$, then $\frac{\|\mathbf{x}^r - \mathbf{x}^*\|}{\|\mathbf{x}^0 - \mathbf{x}^*\|} \leq (1 - \frac{\lambda_n}{\lambda_1})^r$.

- **Analysis chain**:
  diagonal $\rightarrow$ pure quadratic $\rightarrow$ quadratic plus linear term.

# Result for Quadratic Case

**Thm 1a** (strongly convex quadratic): Suppose $Q$ is symmetric PD (positive-definite).
Consider solving

$$\min_{\mathbf{w}\in\mathbb{R}^n} f(\mathbf{w}) \triangleq 0.5\mathbf{w}^T Q\mathbf{w} + \mathbf{b}^T\mathbf{w} + \mathbf{c}.$$

Suppose GD with stepsize $1/L$, where $L = \lambda_{\max}(Q)$, generates a sequence $\{\mathbf{w}_r\}$. Then

$$\|w_{r+1} - w^*\|^2 \leq \left(1 - \frac{1}{\kappa}\right)\|w_r - w^*\|^2, \tag{1}$$

where the condition number $\kappa = \lambda_{\max}(Q)/\lambda_{\min}(Q)$ .

# Normalization and Preconditioning

- What is the effect of normalization? Assume $X^T = [X_1, \dots, X_d]$
- Linear regression $\|X_1 w_1 + \cdots + X_d w_d - \mathbf{b}\|^2$. Convergence speed depends on $\kappa(XX^T)$.
- Scale each row $X_i^T$ by $\gamma_i = 1/\|X_i\|, i = 1, \dots, d.$

  Equivalent to: Change matrix $X$ to $\hat{X} = \Gamma X$

  For this $\hat{X}\hat{X}^T = \Gamma X X^T \Gamma^T$, all of its diagonal entries are $1$
- Convergence speed now depends on $\kappa(\hat{X}\hat{X}^T)$
  - e.g. $X = [100, 0; 0, 1]$, what is $\hat{X}$?
- Called Jacobi preconditioning in numerical linear algebra.
  - Does it improve condition number always? Open question; mentioned in R. Sun, Y. Ye, Worst case complexity of Cyclic Coordinate Descent: $O(n^2)$ Gap with Randomized Version. MP 2019.

# Why Data Preprocessing is Useful?

**Claim:**(informal) Normalizing data can reduce condition number (in most cases), which makes GD faster, which makes GD faster.

- **Part I:** In linear regression, normalizing data is called Jacobi preconditioning, which likely improves condition number (but no proof yet)
- **Part II:** In linear regression, smaller condition number implies faster convergence (Thm 1)

**Claim:**(informal) Centering can reduce condition number (in most cases), which makes GD faster, which makes GD faster.

- Part I: centering reduces condition number. No aware of a proof; but likely true for most cases

# Outline

Applying Gradient Descent to Regression

Convergence Rate Analysis for Quadratic Functions

Results for Strongly Convex, Convex, Nonconvex Functions

# Result for Strongly Convex Functions

For a twice-differentiable function $f$, we say it is $L$-smooth and $\mu$-strongly-convex iff

$$LI \succeq \nabla^2 f(\mathbf{w}) \succeq \mu I, \quad \forall \mathbf{w}. \tag{2}$$

**Theorem 1b** (strongly convex): Suppose a twice-differentiable function $f$ is $L$-smooth and $\mu$-strongly convex, and $\mathbf{w}^*$ is a global minimum. Then GD with stepsize $1/L$ generates a sequence $\{\mathbf{w}_r\}$ that satisfies

$$\|\mathbf{w}_{r+1} - \mathbf{w}^*\|^2 \leq \left(1 - \frac{1}{\kappa}\right) \|\mathbf{w}_r - \mathbf{w}^*\|^2, \tag{3}$$

where $\kappa = L/\mu$.

- **Remark 1:** The assumption can be weakend to continuously differentiable function with $L$-Lipschitz gradient and is $\mu$-strongly convex.
- **Remark 2**: Best stepsize is $2/(L+\mu)$, with rate $1 - 2/(\kappa+1)$.

# Proof of Theorem 1b

By the fundamental theorem of calculus, we have

$$\nabla f(\mathbf{w}_r) = \nabla f(\mathbf{w}_r) - \underbrace{\nabla f(\mathbf{w}^*)}_{0} = \int_{t=0}^{1} \nabla^2 f(\mathbf{v}(t))(\mathbf{w}_r - \mathbf{w}^*)dt$$

where $\mathbf{v}(t) = \mathbf{w}_r + t(\mathbf{w}^* - \mathbf{w}_r)$.

Then we have

$$\begin{aligned}
\|\mathbf{w}_{r+1} - \mathbf{w}^*\| &= \|\mathbf{w}_r - \mathbf{w}^* - \alpha \nabla f(\mathbf{w}_r)\| \\
&= \|\mathbf{w}_r - \mathbf{w}^* - \int_{t=0}^{1} \nabla^2 f(\mathbf{v}(t))(\mathbf{w}_r - \mathbf{w}^*)dt\| \\
&= \|\int_{t=0}^{1} [I - \alpha \nabla^2 f(\mathbf{v}(t))](\mathbf{w}_r - \mathbf{w}^*)dt\| \\
&\leq (1 - \frac{\mu}{L})\|\mathbf{w}_r - \mathbf{w}^*\|.
\end{aligned}$$

# Alternative Proof for Strongly Convex Case (reading)

**Proof**: Theorem 2.1.11 in [Nesterov,2013] states that

$$\langle w - w^*, \nabla f(w) \rangle \geq \frac{\mu L}{\mu + L} \|w - w^*\|^2 + \frac{1}{\mu + L} \|\nabla f(w)\|^2. \tag{4}$$

Then we have

$$\begin{aligned}
\|w_{r+1} - w^*\|^2 &= \|w_r - \eta \nabla F(w_r) - w^*\|^2 \\
&= \|w_r - w^*\|^2 + \eta^2 \|\nabla F(w_r)\|^2 - 2\eta \langle w_r - w^*, \nabla F(w_r) \rangle \\
&\leq \left(1 - 2\eta \frac{\mu L}{\mu + L}\right) \|w_r - w^*\|^2 + \eta \left(\eta - \frac{2}{\mu + L}\right) \|\nabla F(w_r)\|^2 \\
&\leq \left(1 - 2\eta \frac{\mu L}{\mu + L}\right) \|w_r - w^*\|^2.
\end{aligned}$$

Plugging in $\eta = 1/L$, we obtain the desired result. $\square$

[Nesterov'2013]: Introductory Lectures on Convex Optimization.

**Remark:** This proof looks simple, but requires a non-trivial lemma (4). It can be relaxed a bit: $\langle w - w^*, \nabla f(w) \rangle \geq \frac{\mu}{2} \|w - w^*\|^2 + \frac{1}{2L} \|\nabla f(w)\|^2$. Then using $\eta = 1/L$, can obtain rate $1 - \frac{1}{\kappa}$, which is only slightly worse than Thm 1.

# Result for Convex Functions

**Theorem 2**: Suppose $f$ is a convex function with Lipschitz gradient, i.e.,

$$0 \preceq \nabla^2 f(\mathbf{x}) \preceq LI, \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

Consider $\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x})$. Suppose GD with stepsize $1/L$ generates a sequence $\mathbf{x}^r$, then for any optimal solution $\mathbf{x}^*$, we have

$$e_T \triangleq f(\mathbf{x}^T) - f(\mathbf{x}^*) \leq \frac{2L}{T} \|\mathbf{x}^0 - \mathbf{x}^*\|^2. \qquad .$$

- **Remark 1:** This is called "sublinear rate" $O(1/T)$.

  Error of linear rate: $1, 0.1, 0.01, 0.001, \ldots$

  Error of sublinear rate $O(1/T)$: $1, 1/2, 1/3, \ldots, 1/10, \ldots, 1/100, \ldots$

- **Remark 2:** For most convex problems in practice, they achieve linear rate [Luo,Tseng'92], due to a deeper reason not covered here

# Non-convex Functions

**Theorem 3:** Suppose $F(\mathbf{w})$ is differentiable and $L$-smooth. Consider $GD$ with stepsize $1/L$. Then

$$\min_{0 \le r \le T} \{\|\nabla F(\mathbf{w}^r)\|\} \le \sqrt{\frac{2L(F(\mathbf{w}^0) - F^*)}{T}}.$$

**Corollary 2** Under the same conditions as Theorem 2, when $T \ge \frac{2L(F(\mathbf{w}^0) - F^*)}{\epsilon^2}$, we have $\min_{0 \le r \le T}\{\|\nabla F(\mathbf{w}^r)\|\} \le \epsilon$.

This is sub-linear convergence with rate $O(1/\sqrt{T})$.

# Proof for Nonconvex Case I (reading)

**Descent lemma**: If $F(w)$ is $L$-smooth, then

$$F(v) \leq F(w) + \langle \nabla F(w), v - w \rangle + \frac{L}{2}\|w - v\|^2, \quad \forall w, v. \tag{5}$$

**Proof of Thm. 2:** Step 1 (sufficient descent): GD method makes significant progress in each iteration.

$$\begin{aligned} F(w_{r+1}) - F(w_r) &\leq \langle \nabla F(w_r), w_{r+1} - w_r \rangle + \frac{L}{2}\|w_{r+1} - w_r\|^2 && \text{(6a)} \\ &= -\frac{1}{L}\|\nabla F(w_r)\|^2 + \frac{L}{2}\|w_{r+1} - w_r\|^2 && \text{(6b)} \\ &\leq -\frac{1}{2L}\|\nabla F(w_r)\|^2 && \text{(6c)} \end{aligned}$$

where (6a) is by the $L$-smoothness of $F$ (by the descent lemma); (6b) and (6c) are due to the identity $w_{r+1} - w_r = -\frac{1}{L}\nabla f(w_r)$ .

# Proof for Nonconvex Case II (reading)

Step 2: Telescope sum. Take the sum for $r = 0, 1, 2, \dots, T$, we have

$$
\begin{aligned}
& F(w_{T+1}) - F(w_0) \le -\frac{1}{2L} \sum_{r=0}^{T} \|\nabla F(w_r)\|^2. \\
\Longrightarrow & \frac{T}{2L} \min_{0 \le r \le T} \{\|\nabla F(w_r)\|^2\} \le \frac{1}{2L} \sum_{r=0}^{T} \|\nabla F(w_r)\|^2 \le F(w_0) - F(w_{T+1}) \le F(w_0) - F^* \\
\Longrightarrow & \min_{0 \le r \le T} \{\|\nabla F(w_r)\|^2\} \le \frac{2L(F(w_0) - F^*)}{T}.
\end{aligned}
\tag{7}
$$

# Transform Rate to Number of Iterations

**Iteration complexity** of strongly convex functions:

- How many iterations needed for $e(\mathbf{x}^r) = \|\mathbf{x}^r - \mathbf{x}^*\|$ to reach below $\epsilon$?
- We know $e(\mathbf{x}^r)/e(\mathbf{x}^0) \leq \beta^r$, where $\beta = 1 - 1/\kappa$.
  To make sure $e(\mathbf{x}^r)/e(\mathbf{x}^0) \leq \epsilon$, we only need ,

$$\beta^r \leq \epsilon \Leftarrow r \ln \beta \leq \ln \epsilon$$
$$\Leftrightarrow r \ln \frac{1}{\beta} \geq \ln \frac{1}{\epsilon} \Leftrightarrow r \geq \frac{1}{\ln \frac{1}{\beta}} \ln \frac{1}{\epsilon} = \frac{1}{\ln \frac{1}{1-1/\kappa}} \ln \frac{1}{\epsilon}$$

- For large $\kappa$, $\#$ of iterations

$$r \gtrsim \kappa \ln \frac{1}{\epsilon}$$

**Iteration complexity** of convex functions: $r \geq \frac{2LD_0^2}{\epsilon}$ iterations are enough to make sure $f(\mathbf{x}^r) - f^* \leq \epsilon$

**Iteration complexity** of nonconvex functions: $r \geq \frac{2L(F(\mathbf{x}^0)-F^*)}{\epsilon^2}$, we have $\min_{0\leq t \leq r}\{\|\nabla F(\mathbf{x}^t)\|\} \leq \epsilon$.

# Discussion

- Practical lessons: what do we learn from these results?
- If the algorithm fails to converge or converge very slowly, what could be the reasons?
  - Bad stepsize choice; bad condition number
- How to improve algorithm performance?
  - Tune algorithm, e.g., stepsize (can use the help of Hessian info)
  - Modify formulation to get better condition number (data processing; preconditioning methods; etc.)
- How to analyze an algorithm?
  - use simple functions to gain understanding
  - Insights today all come from 2D quadratic functions!
- The main lesson comes from $\kappa$-ite-complexity result of strongly convex case; we do not mention insight from convex and nonconvex problems
  - **General insight**: even for nonconvex problems, local $\kappa(\nabla^2 f(\mathbf{x}^r))$ could be an informative metric for understanding convergence speed (though no rigorous proof)

# Summary

**Convergence rate** of GD with constant stepsize for three cases:

- **Strongly convex**: $O(\kappa \log \frac{1}{\epsilon})$ iterations to achieve $\epsilon$-optimal solution
- **Convex**: $O(\frac{L}{\epsilon})$ iterations to achieve $\epsilon$-optimal solution
- **Nonconvex**: $O(\frac{L}{\sqrt{\epsilon}})$ iterations to achieve $\epsilon$-stationary solution

**Stepsize choice**:

- **Strongly convex**: optimal stepsize $2/(L+\mu)$
- **Convex and nonconvex**: common stepsize $1/L$

A common issue for slow convergence of GD: **ill-conditioning**

**Data preprocessing**: in data analysis, always normalize and/or center data!

- Improve condition number!