Optimization

- Goal: find the minimizer of a function

\[ \min_w f(w) \]

For now we assume \( f \) is twice differentiable

- Machine learning algorithm: find the hypothesis that minimizes training error
A function $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function $\iff$ the function $f$ is below any line segment between two points on $f$:

$$\forall x_1, x_2, \forall t \in [0, 1], \quad f(tx_1 + (1-t)x_2) \leq tf(x_1) + (1-t)f(x_2)$$
Convex function

- A function $f : \mathbb{R}^n \to \mathbb{R}$ is a convex function
  $\iff$ the function $f$ is below any line segment between two points on $f$:

  $\forall x_1, x_2, \forall t \in [0, 1], \quad f(tx_1 + (1 - t)x_2) \leq tf(x_1) + (1 - t)f(x_2)$

**Strict convex:** $f(tx_1 + (1 - t)x_2) < tf(x_1) + (1 - t)f(x_2)$
Convex function

- Another equivalent definition for differentiable function:

\[ f(x) \geq f(x_0) + \nabla f(x_0)^T (x - x_0), \quad \forall x, x_0 \]
Convex function:

- (for differentiable function) $\nabla f(w^*) = 0 \iff w^*$ is a global minimum
- If $f$ is twice differentiable $\Rightarrow$
  
  $f$ is convex if and only if $\nabla^2 f(w)$ is positive semi-definite
- Example: linear regression, logistic regression, · · ·
Convex function

- **Strict convex function:**
  - $\nabla f(w^*) = 0 \iff w^*$ is the unique global minimum
  - most algorithms only converge to gradient $= 0$
  - Example: Linear regression when $X^TX$ is invertible
Convex vs Nonconvex

- Convex function:
  \[ \nabla f(w^*) = 0 \iff w^* \text{ is a global minimum} \]
  Example: linear regression, logistic regression, \ldots

- Non-convex function:
  \[ \nabla f(w^*) = 0 \iff w^* \text{ is Global min, local min, or saddle point} \]
  (also called **stationary points**)
  most algorithms only converge to **stationary points**
  Example: neural network, \ldots

![Convex vs Non-Convex Diagram](image)
Gradient descent
Gradient Descent

- Gradient descent: repeatedly do

\[ w^{t+1} \leftarrow w^t - \alpha \nabla f(w^t) \]

\[ \alpha > 0 \text{ is the step size} \]
Gradient Descent

- Gradient descent: repeatedly do

\[ w^{t+1} \leftarrow w^t - \alpha \nabla f(w^t) \]

\( \alpha > 0 \) is the step size

- Generate the sequence \( w^1, w^2, \ldots \)

converge to stationary points ( \( \lim_{t \to \infty} \| \nabla f(w^t) \| = 0 \) )

Step size too large \( \Rightarrow \) diverge; too small \( \Rightarrow \) slow convergence
Gradient Descent

- **Gradient descent**: repeatedly do
  \[ w^{t+1} \leftarrow w^t - \alpha \nabla f(w^t) \]

  \( \alpha > 0 \) is the **step size**

- Generate the sequence \( w^1, w^2, \ldots \)
  converge to stationary points ( \( \lim_{t \to \infty} \| \nabla f(w^t) \| = 0 \) )

- Step size **too large** ⇒ **diverge**; **too small** ⇒ **slow convergence**
Why gradient descent?

- **Successive approximation view**

  At each iteration, form an approximation function of $f(\cdot)$:

  \[ f(w^t + d) \approx g(d) := f(w^t) + \nabla f(w^t)^T d + \frac{1}{2\alpha} \|d\|^2 \]

  Update solution by $w^{t+1} \leftarrow w^t + d^*$

  \[ d^* = \text{arg min}_d g(d) \]

  \[ \nabla g(d^*) = 0 \Rightarrow \nabla f(w^t) + \frac{1}{\alpha} d^* = 0 \Rightarrow d^* = -\alpha \nabla f(w^t) \]

  $d^*$ will decrease $f(\cdot)$ if $\alpha$ (step size) is sufficiently small.
Why gradient descent?

- **Successive approximation view**

  At each iteration, form an approximation function of $f(\cdot)$:

  $$f(w^t + d) \approx g(d) := f(w^t) + \nabla f(w^t)^T d + \frac{1}{2\alpha} \|d\|^2$$

  Update solution by $w^{t+1} \leftarrow w^t + d^*$

  $$d^* = \arg\min_d g(d)$$

  $$\nabla g(d^*) = 0 \Rightarrow \nabla f(w^t) + \frac{1}{\alpha} d^* = 0 \Rightarrow d^* = -\alpha \nabla f(w^t)$$

- $d^*$ will decrease $f(\cdot)$ if $\alpha$ (step size) is sufficiently small
Illustration of gradient descent
Form a quadratic approximation

\[ f(w^t + d) \approx g(d) = f(w^t) + \nabla f(w^t)^T d + \frac{1}{2\alpha} \|d\|^2 \]
Illustration of gradient descent

Minimize $g(d)$:

$$\nabla g(d^*) = 0 \Rightarrow \nabla f(w^t) + \frac{1}{\alpha} d^* = 0 \Rightarrow d^* = -\alpha \nabla f(w^t)$$
Illustration of gradient descent

Update

\[ w^{t+1} = w^t + d^* = w^t - \alpha \nabla f(w^t) \]
Form another quadratic approximation

\[ f(w^{t+1} + d) \approx g(d) = f(w^{t+1}) + \nabla f(w^{t+1})^T d + \frac{1}{2\alpha} \|d\|^2 \]

\[ d^* = -\alpha \nabla f(w^{t+1}) \]
Illustration of gradient descent

Update

\[ w^{t+2} = w^{t+1} + d^* = w^{t+1} - \alpha \nabla f(w^{t+1}) \]
When will it diverge?

Can diverge \((f(w^t) < f(w^{t+1}))\) if \(g\) is not an upperbound of \(f\)

\[ f(w^t) < f(w^{t+1}), \text{ diverge because } g\text{'s curvature is too small} \]
When will it converge?

Always converge \((f(w^t) > f(w^{t+1}))\) when \(g\) is an upperbound of \(f\).

\[ f(w^t) > f(w^{t+1}), \text{ converge when } g\text{'s curvature is large enough } \]
Convergence

- Let $L$ be the Lipchitz constant
  \[
  (\nabla^2 f(x) \leq L I \text{ for all } x)
  \]
- **Theorem:** gradient descent converges if $\alpha < \frac{1}{L}$
Let $L$ be the Lipchitz constant

\[(\nabla^2 f(x) \leq LI \text{ for all } x)\]

**Theorem:** gradient descent converges if $\alpha < \frac{1}{L}$

**Why?**

When $\alpha < \frac{1}{L}$, for any $d$,

\[
g(d) = f(w^t) + \nabla f(w^t)^T d + \frac{1}{2\alpha} \|d\|^2
\]

\[
> f(w^t) + \nabla f(w^t)^T d + \frac{L}{2} \|d\|^2
\]

\[
\geq f(w^t + d)
\]
Convergence

• Let $L$ be the Lipchitz constant
  $$(\nabla^2 f(x) \leq L I \text{ for all } x)$$
• **Theorem:** gradient descent converges if $\alpha < \frac{1}{L}$
• Why?
  • When $\alpha < 1/L$, for any $d$,
    $$g(d) = f(w^t) + \nabla f(w^t)^T d + \frac{1}{2\alpha} \|d\|^2$$
    $$> f(w^t) + \nabla f(w^t)^T d + \frac{L}{2} \|d\|^2$$
    $$\geq f(w^t + d)$$

• So, $f(w^t + d^*) < g(d^*) \leq g(0) = f(w^t)$
Let $L$ be the Lipchitz constant

$$(\nabla^2 f(x) \leq L I \text{ for all } x)$$

**Theorem:** gradient descent converges if $\alpha < \frac{1}{L}$

Why?

- When $\alpha < 1/L$, for any $d$,

$$g(d) = f(w^t) + \nabla f(w^t)^T d + \frac{1}{2\alpha} \|d\|^2$$

$$> f(w^t) + \nabla f(w^t)^T d + \frac{L}{2} \|d\|^2$$

$$\geq f(w^t + d)$$

So, $f(w^t + d^*) < g(d^*) \leq g(0) = f(w^t)$

In formal proof, need to show $f(w^t + d^*)$ is sufficiently smaller than $f(w^t)$
Applying to Logistic regression

### gradient descent for logistic regression

- Initialize the weights $w_0$
- For $t = 1, 2, \cdots$
  - Compute the gradient
    \[
    \nabla f(w) = - \frac{1}{N} \sum_{n=1}^{N} \frac{y_n x_n}{1 + e^{y_n w^T x_n}}
    \]
- Update the weights: $w \leftarrow w - \alpha \nabla f(w)$
- Return the final weights $w$

When to stop?
- Fixed number of iterations, or
- Stop when $\|\nabla f(w)\| < \epsilon$
Applying to Logistic regression

gradient descent for logistic regression

- Initialize the weights $w_0$
- For $t = 1, 2, \ldots$
  - Compute the gradient
    \[
    \nabla f(w) = -\frac{1}{N} \sum_{n=1}^{N} \frac{y_n x_n}{1 + e^{y_n w^T x_n}}
    \]
  - Update the weights: $w \leftarrow w - \alpha \nabla f(w)$
- Return the final weights $w$

When to stop?

- Fixed number of iterations, or
- Stop when $\|\nabla f(w)\| < \epsilon$
Line Search

- In practice, we do not know $L \cdots$
  
  need to tune step size when running gradient descent
Line Search

- In practice, we do not know $L \cdots$ need to tune step size when running gradient descent
- Line Search: Select step size automatically (for gradient descent)
The back-tracking line search:

- Start from some large $\alpha_0$
- Try $\alpha = \alpha_0, \frac{\alpha_0}{2}, \frac{\alpha_0}{4}, \cdots$
  - Stop when $\alpha$ satisfies some sufficient decrease condition

A simple condition:

$$f(w + \alpha d) < f(w)$$

often works in practice but doesn't work in theory

A (provable) sufficient decrease condition:

$$f(w + \alpha d) \leq f(w) + \sigma \alpha \nabla f(w)^T d$$

for a constant $\sigma \in (0, 1)$
The **back-tracking** line search:

- Start from some large \( \alpha_0 \)
- Try \( \alpha = \alpha_0, \frac{\alpha_0}{2}, \frac{\alpha_0}{4}, \cdots \)
  - Stop when \( \alpha \) satisfies some **sufficient decrease condition**
- A simple condition: \( f(w + \alpha d) < f(w) \)
The back-tracking line search:
- Start from some large $\alpha_0$
- Try $\alpha = \alpha_0, \frac{\alpha_0}{2}, \frac{\alpha_0}{4}, \ldots$
  - Stop when $\alpha$ satisfies some sufficient decrease condition
- A simple condition: $f(w + \alpha d) < f(w)$
  - often works in practice but doesn’t work in theory
The **back-tracking** line search:

- Start from some large $\alpha_0$
- Try $\alpha = \alpha_0, \frac{\alpha_0}{2}, \frac{\alpha_0}{4}, \cdots$
  
  Stop when $\alpha$ satisfies some **sufficient decrease condition**
- A simple condition: $f(w + \alpha d) < f(w)$
  
  often works in practice but doesn’t work in theory
- A (provable) sufficient decrease condition:

\[
f(w + \alpha d) \leq f(w) + \sigma \alpha \nabla f(w)^T d
\]

for a constant $\sigma \in (0, 1)$
Line Search

**gradient descent with backtracking line search**

- Initialize the weights $w_0$
- For $t = 1, 2, \cdots$
  - Compute the gradient
    \[
    d = -\nabla f(w)
    \]
  - For $\alpha = \alpha_0, \alpha_0/2, \alpha_0/4, \cdots$
    Break if $f(w + \alpha d) \leq f(w) + \sigma \alpha \nabla f(w)^T d$
  - Update $w \leftarrow w + \alpha d$
- Return the final solution $w$
Stochastic Gradient descent
Large-scale Problems

- Machine learning: usually minimizing the training loss

\[
\min_w \left\{ \frac{1}{N} \sum_{n=1}^{N} \ell(w^T x_n, y_n) \right\} := f(w) \text{ (linear model)}
\]

\[
\min_w \left\{ \frac{1}{N} \sum_{n=1}^{N} \ell(h_w(x_n), y_n) \right\} := f(w) \text{ (general hypothesis)}
\]

\(\ell\): loss function (e.g., \(\ell(a, b) = (a - b)^2\))

- Gradient descent:

\[
\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla f(\mathbf{w})
\]

Main computation
Large-scale Problems

- Machine learning: usually minimizing the training loss

\[
\min_w \left\{ \frac{1}{N} \sum_{n=1}^N \ell(w^T x_n, y_n) \right\} := f(w) \text{ (linear model)}
\]

\[
\min_w \left\{ \frac{1}{N} \sum_{n=1}^N \ell(h_w(x_n), y_n) \right\} := f(w) \text{ (general hypothesis)}
\]

\( \ell \): loss function (e.g., \( \ell(a, b) = (a - b)^2 \))

- Gradient descent:

\[
w \leftarrow w - \eta \nabla f(w)
\]

Main computation

- In general, \( f(w) = \frac{1}{N} \sum_{n=1}^N f_n(w) \),
  each \( f_n(w) \) only depends on \((x_n, y_n)\)
Stochastic gradient

- Gradient:

\[ \nabla f(w) = \frac{1}{N} \sum_{n=1}^{N} \nabla f_n(w) \]

- Each gradient computation needs to go through all training samples slow when millions of samples

- Faster way to compute “approximate gradient”? 

- Use stochastic sampling:

  Sample a small subset \( B \subseteq \{1, \cdots, N\} \)

  Estimated gradient 

  \[ \nabla f(w) \approx \frac{1}{|B|} \sum_{n \in B} \nabla f_n(w) \]

  \( |B| \): batch size
Stochastic gradient

- **Gradient:**
  \[ \nabla f(w) = \frac{1}{N} \sum_{n=1}^{N} \nabla f_n(w) \]

- Each gradient computation needs to go through all training samples
  slow when millions of samples

- Faster way to compute “approximate gradient”? 

- Use **stochastic sampling:**
  - Sample a small subset \( B \subseteq \{1, \cdots, N\} \)
  - Estimated gradient
    \[ \nabla f(w) \approx \frac{1}{|B|} \sum_{n \in B} \nabla f_n(w) \]

  \(|B|\): batch size
Stochastic Gradient Descent (SGD)

- Input: training data \( \{x_n, y_n\}_{n=1}^{N} \)
- Initialize \( w \) (zero or random)
- For \( t = 1, 2, \cdots \)
  - Sample a small batch \( B \subseteq \{1, \cdots, N\} \)
  - Update parameter

\[
    w \leftarrow w - \eta_t \frac{1}{|B|} \sum_{n \in B} \nabla f_n(w)
\]
Stochastic gradient descent

**Stochastic Gradient Descent (SGD)**

- **Input:** training data \( \{x_n, y_n\}_{n=1}^N \)
- **Initialize** \( w \) (zero or random)
- **For** \( t = 1, 2, \ldots \)
  - Sample a small batch \( B \subseteq \{1, \ldots, N\} \)
  - Update parameter

\[
    w \leftarrow w - \eta^t \frac{1}{|B|} \sum_{n \in B} \nabla f_n(w)
\]

**Extreme case:** \( |B| = 1 \Rightarrow \text{Sample one training data at a time} \)
Logistic Regression by SGD

- Logistic regression:

\[
\min_w \frac{1}{N} \sum_{n=1}^{N} \log(1 + e^{-y_n w^T x_n})
\]

SGD for Logistic Regression

- Input: training data \( \{x_n, y_n\}_{n=1}^{N} \)
- Initialize \( w \) (zero or random)
- For \( t = 1, 2, \cdots \)
  - Sample a batch \( B \subseteq \{1, \cdots, N\} \)
  - Update parameter

\[
w \leftarrow w - \eta^t \frac{1}{|B|} \sum_{i \in B} \frac{-y_n x_n}{1 + e^{y_n w^T x_n}} \nabla f_n(w)
\]
Why SGD works?

- Stochastic gradient is an unbiased estimator of full gradient:

\[
E\left[ \frac{1}{|B|} \sum_{n \in B} \nabla f_n(w) \right] = \frac{1}{N} \sum_{n=1}^{N} \nabla f_n(w) = \nabla f(w)
\]
Why SGD works?

- Stochastic gradient is an **unbiased estimator** of full gradient:

\[
E\left[ \frac{1}{|B|} \sum_{n \in B} \nabla f_n(w) \right] = \frac{1}{N} \sum_{n=1}^{N} \nabla f_n(w)
\]

\[= \nabla f(w)\]

- Each iteration updated by

\text{gradient} + \text{zero-mean noise}
Stochastic gradient descent

- In gradient descent, $\eta$ (step size) is a fixed constant
- Can we use fixed step size for SGD?
In gradient descent, $\eta$ (step size) is a fixed constant.

Can we use fixed step size for SGD?

SGD with fixed step size cannot converge to global/local minimizers.
In gradient descent, $\eta$ (step size) is a fixed constant

Can we use fixed step size for SGD?

SGD with fixed step size cannot converge to global/local minimizers

If $w^*$ is the minimizer, $\nabla f(w^*) = \frac{1}{N} \sum_{n=1}^{N} \nabla f_n(w^*) = 0$, but $\frac{1}{|B|} \sum_{n \in B} \nabla f_n(w^*) \neq 0$ if $B$ is a subset

Even if we got minimizer, SGD will move away from it.
Stochastic gradient descent

- In gradient descent, \( \eta \) (step size) is a fixed constant
- Can we use fixed step size for SGD?
- SGD with fixed step size cannot converge to global/local minimizers
- If \( \mathbf{w}^* \) is the minimizer, \( \nabla f(\mathbf{w}^*) = \frac{1}{N} \sum_{n=1}^{N} \nabla f_n(\mathbf{w}^*) = 0 \),
  
  \[
  \frac{1}{|B|} \sum_{n \in B} \nabla f_n(\mathbf{w}^*) \neq 0 \quad \text{if } B \text{ is a subset}
  \]
Stochastic gradient descent

- In gradient descent, $\eta$ (step size) is a fixed constant
- Can we use fixed step size for SGD?
- SGD with fixed step size cannot converge to global/local minimizers
- If $\mathbf{w}^*$ is the minimizer, $\nabla f(\mathbf{w}^*) = \frac{1}{N} \sum_{n=1}^{N} \nabla f_n(\mathbf{w}^*) = 0$,

  but $\frac{1}{|B|} \sum_{n \in B} \nabla f_n(\mathbf{w}^*) \neq 0$ if $B$ is a subset

(Even if we got minimizer, SGD will move away from it)
Stochastic gradient descent, step size

- To make SGD converge:
  - Step size should decrease to 0
  \[ \eta^t \rightarrow 0 \]
  - Usually with polynomial rate: \( \eta^t \approx t^{-a} \) with constant \( a \)
Stochastic gradient descent vs Gradient descent

Stochastic gradient descent:

- **pros:**
  - cheaper computation per iteration
  - faster convergence in the beginning
- **cons:**
  - less stable, slower final convergence
  - hard to tune step size

(Figure from https://medium.com/@ImadPhd/gradient-descent-algorithm-and-its-variants-10f652806a3)
Revisit perceptron Learning Algorithm

- Given a classification data \( \{x_n, y_n\}_{n=1}^{N} \)
- Learning a linear model:

\[
\min_w \frac{1}{N} \sum_{n=1}^{N} \ell(w^T x_n, y_n)
\]

- Consider the loss:

\[
\ell(w^T x_n, y_n) = \max(0, -y_n w^T x_n)
\]

What’s the gradient?
Revisit perceptron Learning Algorithm

\[ \ell(\mathbf{w}^T \mathbf{x}_n, y_n) = \max(0, -y_n \mathbf{w}^T \mathbf{x}_n) \]

Consider two cases:

- **Case I:** \( y_n \mathbf{w}^T \mathbf{x}_n > 0 \) (prediction **correct**)
  - \( \ell(\mathbf{w}^T \mathbf{x}_n, y_n) = 0 \)
  - \( \frac{\partial}{\partial \mathbf{w}} \ell(\mathbf{w}^T \mathbf{x}_n, y_n) = 0 \)

- **Case II:** \( y_n \mathbf{w}^T \mathbf{x}_n < 0 \) (prediction **wrong**)
  - \( \ell(\mathbf{w}^T \mathbf{x}_n, y_n) = -y_n \mathbf{w}^T \mathbf{x}_n \)
  - \( \frac{\partial}{\partial \mathbf{w}} \ell(\mathbf{w}^T \mathbf{x}_n, y_n) = -y_n \mathbf{x}_n \)

**SGD update rule:**
Sample an index 
\( \mathbf{w}_{t+1} \leftarrow \begin{cases} 
\mathbf{w}_t & \text{if } y_n \mathbf{w}^T \mathbf{x}_n \geq 0 \text{ (predict correct)} \\
\mathbf{w}_t + \eta_t y_n \mathbf{x}_n & \text{if } y_n \mathbf{w}^T \mathbf{x}_n < 0 \text{ (predict wrong)} 
\end{cases} \)
Revisit perceptron Learning Algorithm

\[ \ell(w^T x_n, y_n) = \max(0, -y_n w^T x_n) \]

Consider two cases:

- **Case I:** \( y_n w^T x_n > 0 \) (prediction **correct**)
  - \( \ell(w^T x_n, y_n) = 0 \)
  - \( \frac{\partial}{\partial w} \ell(w^T x_n, y_n) = 0 \)

- **Case II:** \( y_n w^T x_n < 0 \) (prediction **wrong**)
  - \( \ell(w^T x_n, y_n) = -y_n w^T x_n \)
  - \( \frac{\partial}{\partial w} \ell(w^T x_n, y_n) = -y_n x_n \)
Revisit perceptron Learning Algorithm

\[ \ell(w^T x_n, y_n) = \max(0, -y_n w^T x_n) \]

Consider two cases:

- **Case I**: \( y_n w^T x_n > 0 \) (prediction correct)
  - \( \ell(w^T x_n, y_n) = 0 \)
  - \( \frac{\partial}{\partial w} \ell(w^T x_n, y_n) = 0 \)

- **Case II**: \( y_n w^T x_n < 0 \) (prediction wrong)
  - \( \ell(w^T x_n, y_n) = -y_n w^T x_n \)
  - \( \frac{\partial}{\partial w} \ell(w^T x_n, y_n) = -y_n x_n \)

**SGD update rule:** Sample an index \( n \)

\[ w^{t+1} \left\{ \begin{array}{ll}
  w^t & \text{if } y_n w^T x_n \geq 0 \text{ (predict correct)} \\
  w^t + \eta^t y_n x_n & \text{if } y_n w^T x_n < 0 \text{ (predict wrong)}
\end{array} \right. \]

Equivalent to Perceptron Learning Algorithm when \( \eta^t = 1 \)
Conclusions

- Gradient descent
- Stochastic gradient descent

Questions?