Optimization for Machine Learning Problems

Yan Yan

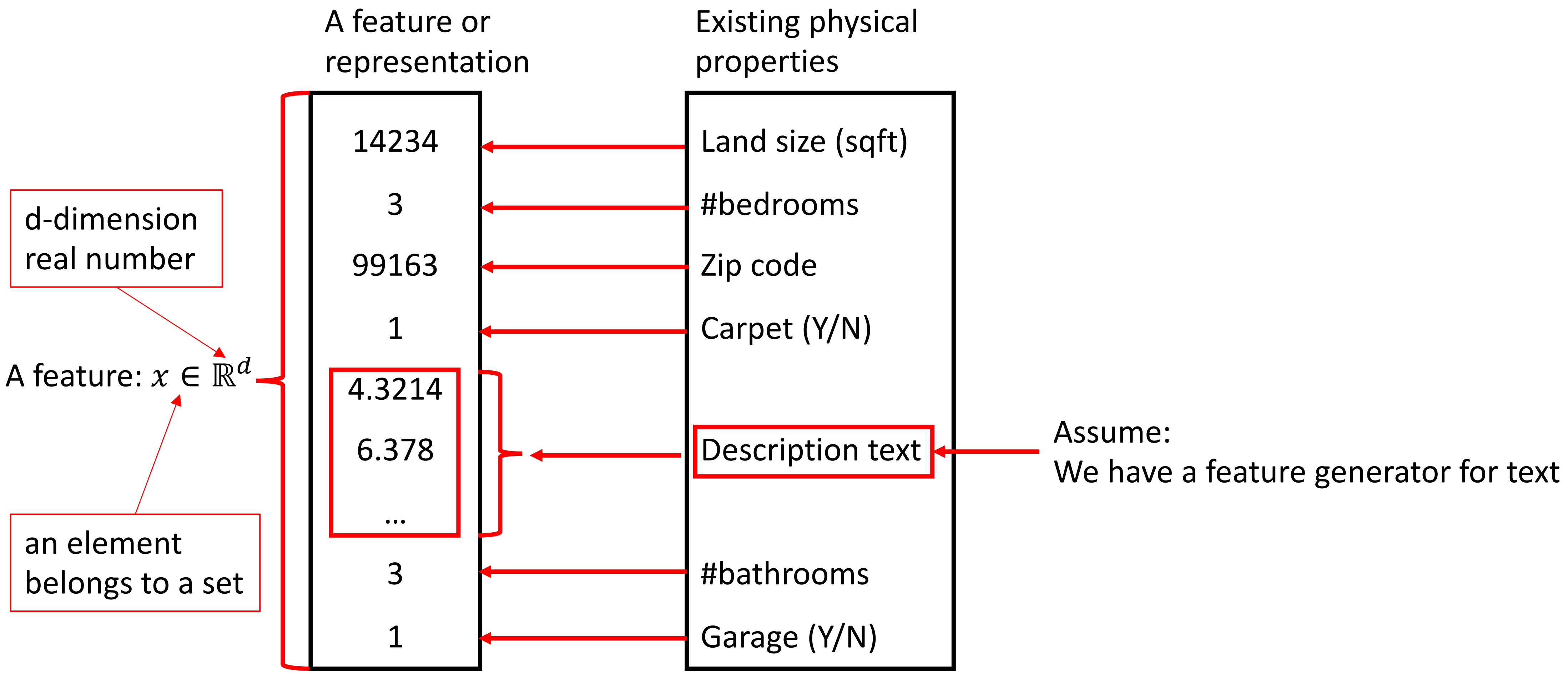
In last class

- Machine learning task details: classification, clustering and regression
    - Terminology
- Their connection to real world applications
    - Why we need these tools
- A showcase: how to construct a learning model
    - What can be used as features
    - Determine model structure
    - Determine model parameters

Today’s class includes

- Why is optimization important in machine learning (for determining model parameters)?
  - Efficiency of optimization algorithms
  - A showcase: the analytical solution VS gradient descent (GD)
- Common optimization algorithms
  - First-order algorithms
  - Second-order algorithms

House price model structure: linear model



Determining model parameters

- An optimization problem (on training set)

Objective function　　　　　　　　n training data

$$min_w f(w) = \frac{1}{n}\sum_{i=1}^{n} l(f(w; x_i), y_i) = \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

- 1-dimensional 1-data showcase with square loss

$$min_w \tfrac{1}{2}(wx - y)^2$$

- Analytical solution?

$$w^* = y/x$$

(first order optimality)

Determining model parameters

- An optimization problem (on training set)

Objective function

n training data

$$min_w f(w) = \frac{1}{n}\sum_{i=1}^{n} l(f(w; x_i), y_i) = \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

- 1-dimensional multi-data square loss

$$min_w \frac{1}{2n}\sum_{i=1}^{n}(wx - y)^2$$

- Analytical solution?

$$w = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}$$

Determining model parameters

- An optimization problem (on training set)

Objective function

n training data

$$min_w f(w) = \frac{1}{n}\sum_{i=1}^{n} l(f(w;x_i),y_i) = \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

- multi-dimensional multi-data square loss

$$min_w \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

- Analytical solution?

Determining model parameters

- An optimization problem (on training set)

Objective function n training data

$$min_w f(w) = \frac{1}{n}\sum_{i=1}^{n} l(f(w; x_i), y_i) = \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

- multi-dimensional multi-data square loss

$$min_w \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

- Analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i{}'wx_i - y_i x_i \ \rightarrow \ 0$$

Determining model parameters

- An optimization problem (on training set)

Objective function

n training data

$$min_w f(w) = \frac{1}{n}\sum_{i=1}^{n} l(f(w;x_i),y_i) = \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

- multi-dimensional multi-data square loss

$$min_w \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

$$X = [x_1, x_2, \dots, x_n] \in \mathbb{R}^{d\times n}$$

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} \in \mathbb{R}^n$$

- Analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i{}'wx_i - y_i x_i \rightarrow 0 \Rightarrow XX'w^* - XY = 0$$

Determining model parameters

- An optimization problem (on training set)

Objective function

n training data

$$min_w f(w) = \frac{1}{n}\sum_{i=1}^{n} l(f(w; x_i), y_i) = \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

- multi-dimensional multi-data square loss

$$X = [x_1, x_2, \dots, x_n] \in \mathbb{R}^{d\times n}$$

$$min_w \frac{1}{2n}\sum_{i=1}^{n}(x_i{}'w - y_i)^2$$

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} \in \mathbb{R}^n$$

- Analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i{}'wx_i - y_i x_i \to 0 \Rightarrow XX'w^* - XY = 0 \Rightarrow w^* = (XX')^{-1}XY$$

Determining model parameters

- An optimization problem (on training set)

Objective function

n training data

$$min_w f(w) = \frac{1}{n}\sum_{i=1}^{n} l(f(w;x_i),y_i) = \frac{1}{2n}\sum_{i=1}^{n}(x_i'w - y_i)^2$$

- multi-dimensional multi-data square loss

$$min_w \frac{1}{2n}\sum_{i=1}^{n}(x_i'w - y_i)^2$$

$$X = [x_1, x_2, \dots, x_n] \in \mathbb{R}^{d\times n}$$

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} \in \mathbb{R}^n$$

- Analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i'wx_i - y_ix_i \rightarrow 0 \Rightarrow XX'w^* - XY = 0 \Rightarrow \boxed{w^* = (XX')^{-1}XY}$$

Q: is this closed form solution a good way in practice? Why?

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i{}' w x_i - y_i x_i \;\rightarrow\; 0 \quad \Rightarrow \quad XX'w^* - XY = 0 \quad \Rightarrow \quad w^* = (XX')^{-1}XY$$

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i{}' w x_i - y_i x_i \ \rightarrow \ 0 \quad \Rightarrow \quad XX'w^* - XY = 0 \quad \Rightarrow \quad w^* = (XX')^{-1}XY$$

- Inverse of a scalar?

$$x\, x^{-1} = 1 \ \rightarrow x^{-1} = 1/x$$

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \ \rightarrow 0 \quad \Rightarrow \quad XX'w^* - XY = 0 \quad \Rightarrow \quad w^* = (XX')^{-1}XY$$

- Inverse of a scalar?

$$x\, x^{-1} = 1 \ \rightarrow x^{-1} = 1/x$$

- Inverse of a matrix?

$$XX^{-1} = I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \ \rightarrow 0 \quad \Rightarrow \quad XX'w^* - XY = 0 \quad \Rightarrow \quad w^* = (XX')^{-1}XY$$

Matrix multiplication	One $n \times m$ matrix & one $m \times p$ matrix	One $n \times p$ matrix	Schoolbook matrix multiplication	$O(nmp)$

Screenshot from https://en.wikipedia.org/wiki/Computational_complexity_of_mathematical_operations#Matrix_algebra

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \rightarrow 0 \quad \Rightarrow \quad XX'w^* - XY = 0 \quad \Rightarrow \quad w^* = (XX')^{-1}XY$$

Matrix multiplication	One $n \times m$ matrix & one $m \times p$ matrix	One $n \times p$ matrix	Schoolbook matrix multiplication	$O(nmp)$

Matrix inversion*	One $n \times n$ matrix	One $n \times n$ matrix	Gauss–Jordan elimination	$O(n^3)$
			Strassen algorithm	$O(n^{2.807})$
			Coppersmith–Winograd algorithm	$O(n^{2.376})$
			Optimized CW-like algorithms	$O(n^{2.373})$

Screenshot from https://en.wikipedia.org/wiki/Computational_complexity_of_mathematical_operations#Matrix_algebra

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \ \rightarrow 0 \ \Rightarrow \ XX'w^* - XY = 0 \ \Rightarrow \ w^* = (XX')^{-1}XY$$

Matrix multiplication	One $n \times m$ matrix & one $m \times p$ matrix	One $n \times p$ matrix	Schoolbook matrix multiplication	$O(nmp)$

Not every matrix has inversion

Matrix inversion*	One $n \times n$ matrix	One $n \times n$ matrix	Gauss–Jordan elimination	$O(n^3)$
			Strassen algorithm	$O(n^{2.807})$
			Coppersmith–Winograd algorithm	$O(n^{2.376})$
			Optimized CW-like algorithms	$O(n^{2.373})$

Screenshot from https://en.wikipedia.org/wiki/Computational_complexity_of_mathematical_operations#Matrix_algebra

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \rightarrow 0 \quad \Rightarrow \quad XX'w^* - XY = 0 \quad \Rightarrow \quad w^* = (XX')^{-1}XY$$

- Matrix multiplication:

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \rightarrow 0 \Rightarrow XX'w^* - XY = 0 \Rightarrow w^* = (XX')^{-1}XY$$

- Matrix multiplication:

$$\boxed{XX'\text{: d×n×d}} \quad \boxed{XY\text{: d×n}} \quad \boxed{(XX')^{-1}XY: d \times d \times n} \rightarrow O(d^2 n)$$

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \ \rightarrow 0 \quad \Rightarrow \quad XX'w^* - XY = 0 \quad \Rightarrow \quad w^* = (XX')^{-1}XY$$

- Matrix multiplication:

$$\boxed{XX': \mathrm{d}\times\mathrm{n}\times\mathrm{d}} \quad \boxed{XY: \mathrm{d}\times\mathrm{n}} \quad \boxed{(XX')^{-1}XY: d\times d\times n} \quad \rightarrow O(d^2 n)$$

- Inverse of a matrix:

$$(XX')^{-1}:\ O(d^{2.373})$$

Determining model parameters

- Computational complexity for the analytical solution?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \ \rightarrow\ 0 \ \Rightarrow \ XX'w^* - XY = 0 \ \Rightarrow \ w^* = (XX')^{-1}XY$$

- Matrix multiplication:

$$\boxed{XX'\colon \mathrm{d}\times\mathrm{n}\times\mathrm{d}} \quad \boxed{XY\colon \mathrm{d}\times\mathrm{n}} \quad \boxed{(XX')^{-1}XY\colon d\times d\times n} \ \rightarrow O(d^2 n)$$

- Inverse of a matrix:

$$(XX')^{-1}\colon\ O(d^{2.373})$$

- Total complexity

$$O(d^2 n + d^{2.373})$$

Determining model parameters

- Can Gradient Descent (GD) do better?

Determining model parameters

- Can Gradient Descent (GD) do better?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \;\rightarrow\; O(dn)$$

$$w_{t+1} = w_t - \alpha_t \nabla_w f(w_t) \;\rightarrow\; O(d) \longrightarrow \text{An iterative algorithm}$$

Determining model parameters

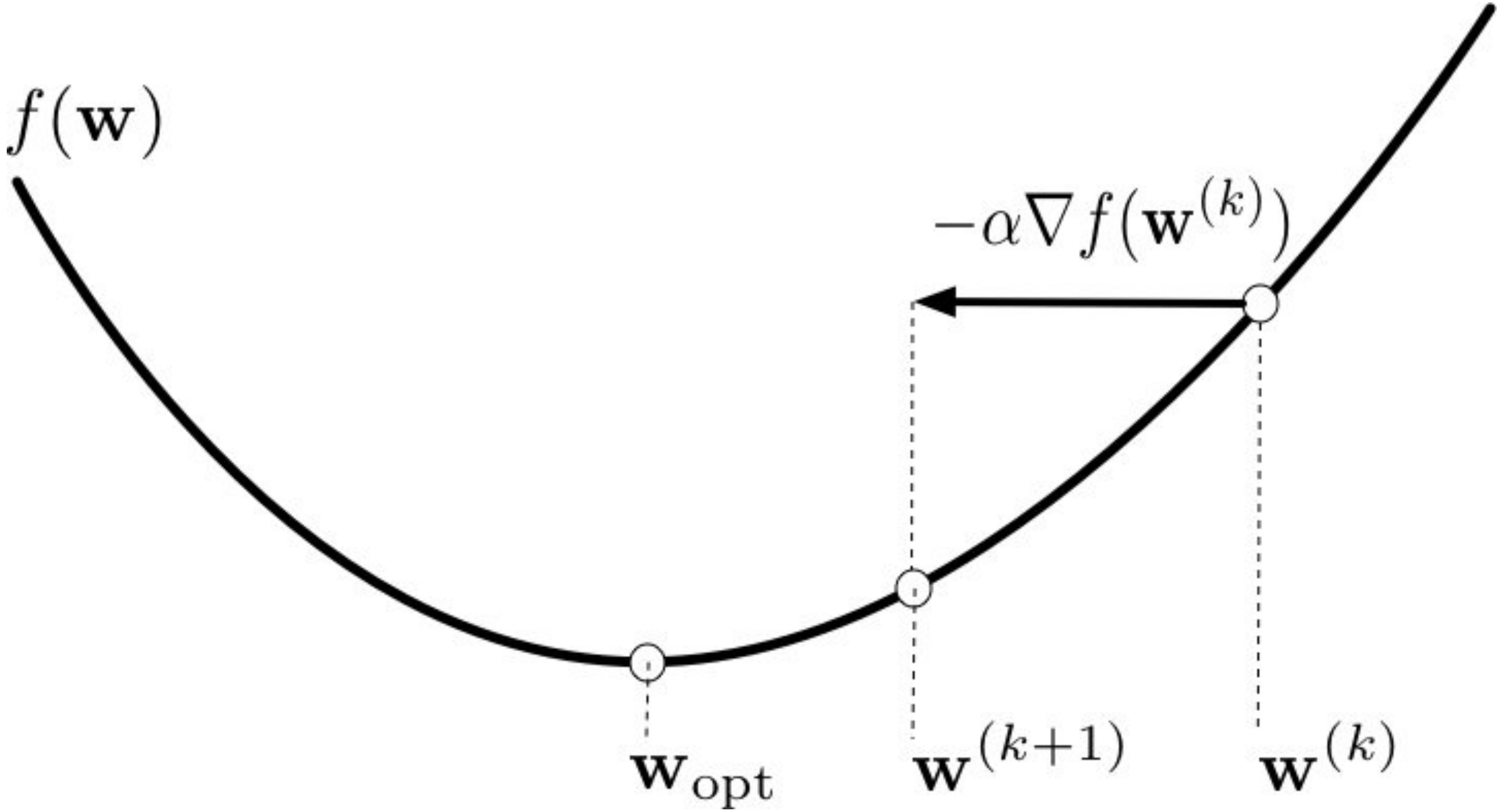
- Can Gradient Descent (GD) do better?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \;\rightarrow\; O(dn)$$

$$w_{t+1} = w_t - \alpha_t \nabla_w f(w_t) \;\rightarrow\; O(d)$$

An iterative algorithm

Step size (learning rate): usually pre-defined


$f(\mathbf{w})$
$-\alpha \nabla f(\mathbf{w}^{(k)})$
$\mathbf{w}_{\mathrm{opt}}$
$\mathbf{w}^{(k+1)}$
$\mathbf{w}^{(k)}$

[Jung2018]

Determining model parameters

- Can Gradient Descent (GD) do better?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i{}' w x_i - y_i x_i \;\rightarrow\; O(dn)$$

$$w_{t+1} = w_t - \alpha_t \nabla_w f(w_t) \;\rightarrow\; O(d) \longrightarrow \text{An iterative algorithm}$$

- Suppose run GD for T iterations

Determining model parameters

- Can Gradient Descent (GD) do better?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \;\rightarrow\; O(dn)$$

$$w_{t+1} = w_t - \alpha_t \nabla_w f(w_t) \;\rightarrow\; O(d)$$ → An iterative algorithm

- Suppose run GD for T iterations
- Total complexity

$$O(dnT)$$

Determining model parameters

- Can Gradient Descent (GD) do better?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \;\rightarrow\; O(dn)$$

$$w_{t+1} = w_t - \alpha_t \nabla_w f(w_t) \;\rightarrow\; O(d) \longrightarrow \text{An iterative algorithm}$$

- Suppose run GD for T iterations
- Total complexity

$O(dnT)$ VS. $O(d^2 n + d^{2.373})$ for the closed form solution

Determining model parameters

- Can Gradient Descent (GD) do better?

$$\nabla_w f(w) = \frac{1}{n}\sum_{i=1}^{n} x_i' w x_i - y_i x_i \ \rightarrow O(dn)$$

$$w_{t+1} = w_t - \alpha_t \nabla_w f(w_t) \ \rightarrow O(d)$$ → An iterative algorithm

- Suppose run GD for T iterations
- Total complexity

$O(dnT)$ VS. $O(d^2 n + d^{2.373})$ for the closed form solution

$T < d$: then GD has less computational complexity

Determining model parameters

- When to terminate GD (determining T)?
  - An approximated solution

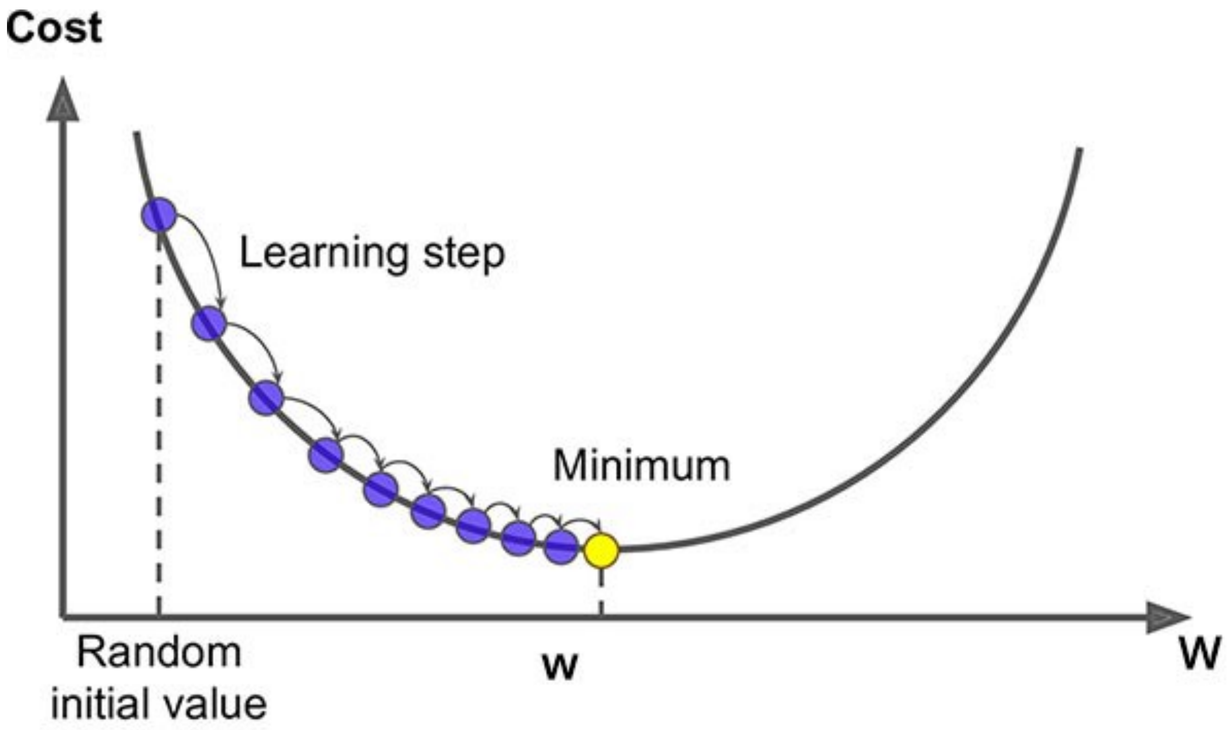


Image from https://morioh.com/p/15c995420be6

Determining model parameters

- When to terminate GD (determining T)?
  - An approximated solution: difficult to set up step size exactly to minimal solution(s)

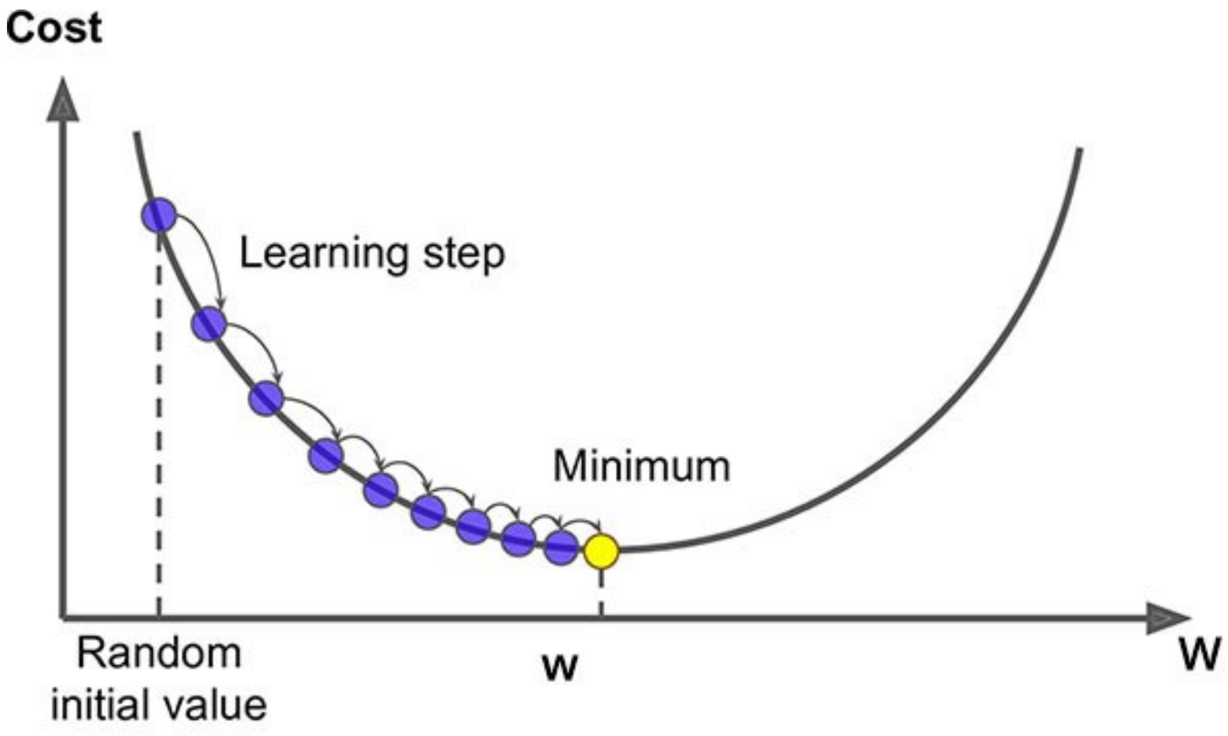


Image from https://morioh.com/p/15c995420be6

Determining model parameters

- When to terminate GD (determining T)?
  - How to measure the approximation error?

$$f(w_T) - f(w_*) \leq \epsilon$$

Determining model parameters

- When to terminate GD (determining T)?
  - How to measure the approximation error?

$$f(w_T) - f(w_*) \leq \epsilon$$

w_*: optimal solution

$f(w_*) = min_w f(w)$

Determining model parameters

- When to terminate GD (determining T)?
  - How to measure the approximation error?

$$f(w_T) - f(w_*) \leq \epsilon$$

w_*: optimal solution
$f(w_*) = min_w f(w)$

No need to know this solution
Only for convergence analysis

Determining model parameters

- When to terminate GD (determining T)?
  - How to measure the approximation error?

$$f(w_T) - f(w_*) \leq \epsilon$$

Level of accuracy:
ϵ-accurate solution

w_*: optimal solution
$f(w_*) = min_w f(w)$

No need to know this solution
Only for convergence analysis

Determining model parameters

- When to terminate GD (determining T)?
  - How to measure the approximation error?

$$f(w_T) - f(w_*) \le \epsilon$$

Level of accuracy:
ϵ-accurate solution

w_*: optimal solution
$f(w_*) = min_w f(w)$

No need to know this solution
Only for convergence analysis

  - Convergence of GD?
    - When $T \to \infty$, then $\epsilon \to 0$
    - But: when to terminate GD?

Determining model parameters

- When to terminate GD (determining T)?
  - Provable convergence: rate of convergence
    - If ϵ is a function of T: $\epsilon(\mathrm{T})$
    - Larger T $\rightarrow$ smaller $\epsilon(\mathrm{T})$

Determining model parameters

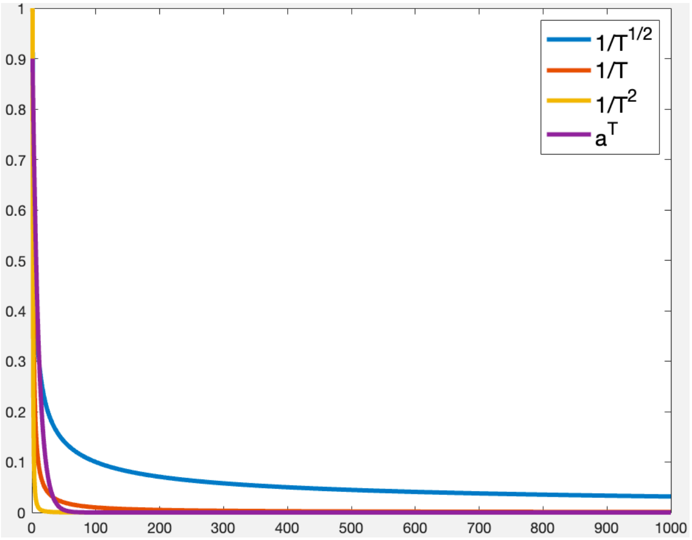
- When to terminate GD (determining T)?
  - Provable convergence: rate of convergence
    - If ϵ is a function of T: $\epsilon(\mathrm{T})$
    - Larger T $\rightarrow$ smaller $\epsilon(\mathrm{T})$
    - If determining $\epsilon(\mathrm{T})$ theoretically:

      we know roughly when to terminate optimization algorithm

Determining model parameters

- When to terminate GD (determining T)?
  - Provable convergence: rate of convergence
    - If ϵ is a function of T: $\epsilon(\mathrm{T})$
    - Larger T $\rightarrow$ smaller $\epsilon(\mathrm{T})$
    - If determining $\epsilon(\mathrm{T})$ theoretically:

      we know roughly when to terminate optimization algorithm



$$\epsilon(\mathrm{T}) = \frac{1}{\sqrt{T}}$$

$$\epsilon(\mathrm{T}) = \frac{1}{T}$$

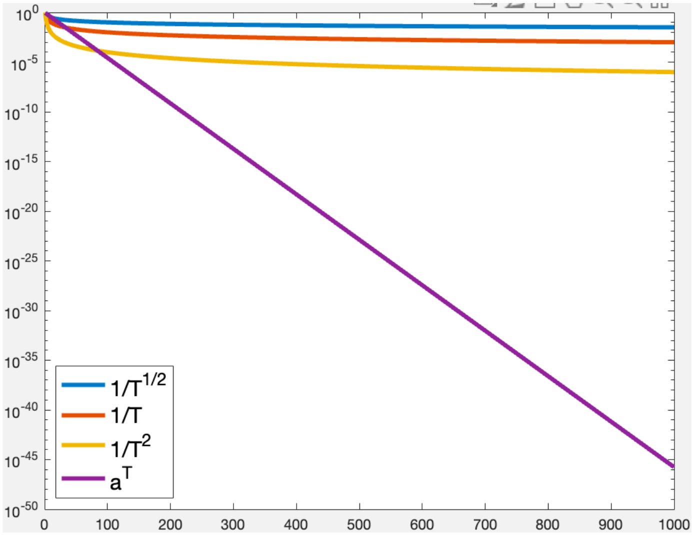
$$\epsilon(\mathrm{T}) = \frac{1}{T^2}$$

$$\epsilon(\mathrm{T}) = a^T \qquad a = 0.9$$

Determining model parameters

- When to terminate GD (determining T)?
  - Provable convergence: rate of convergence
    - If ϵ is a function of T: $\epsilon(\mathrm{T})$
    - Larger T $\rightarrow$ smaller $\epsilon(\mathrm{T})$
    - If determining $\epsilon(\mathrm{T})$ theoretically:

      we know roughly when to terminate optimization algorithm



$$\epsilon(\mathrm{T}) = \frac{1}{\sqrt{T}}$$

$$\epsilon(\mathrm{T}) = \frac{1}{T}$$

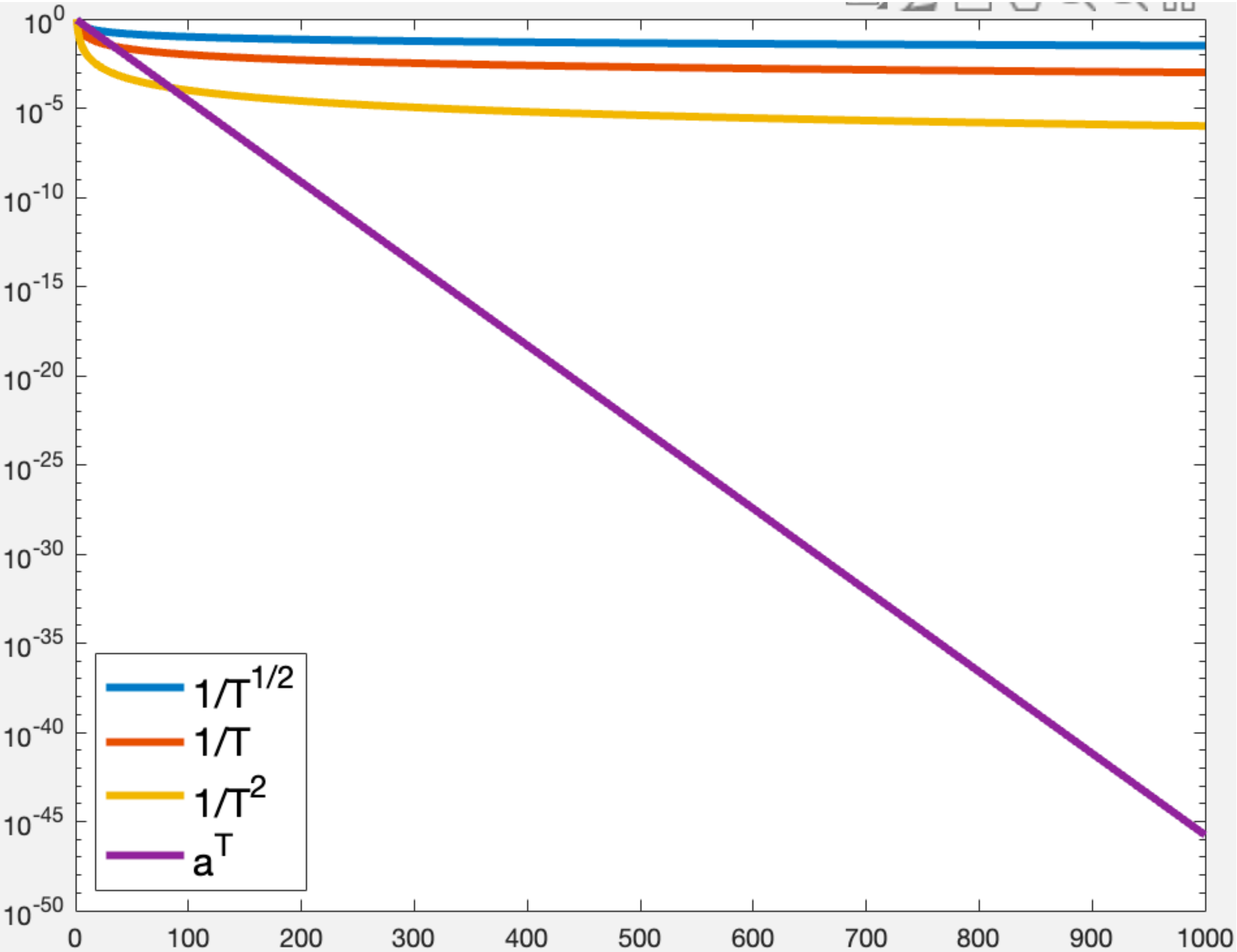
$$\epsilon(\mathrm{T}) = \frac{1}{T^2}$$

$$\epsilon(\mathrm{T}) = a^T \qquad a = 0.9$$

Dete

• Whe

• Pr



10^0
10^-5
10^-10
10^-15
10^-20
10^-25
10^-30
10^-35
10^-40
10^-45
10^-50
0
100
200
300
400
500
600
700
800
900
1000
1/T^1/2
1/T
1/T^2
a^T

Determining model parameters

- When to terminate GD (determining T)?
  - Main factors influencing convergence rate?
    - Step size (learning rate)

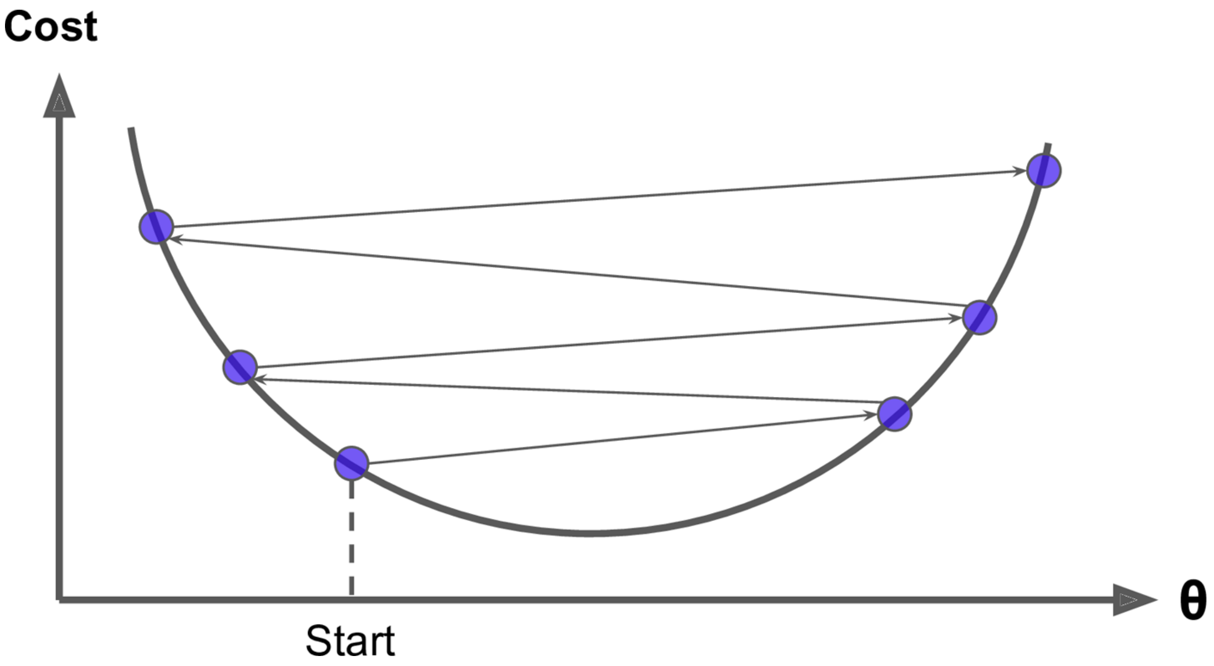


Image from https://www.oreilly.com/library/view/hands-on-machine-learning/9781491962282/ch04.html

Determining model parameters

- When to terminate GD (determining T)?
  - Main factors influencing convergence rate?
    - Step size (learning rate)

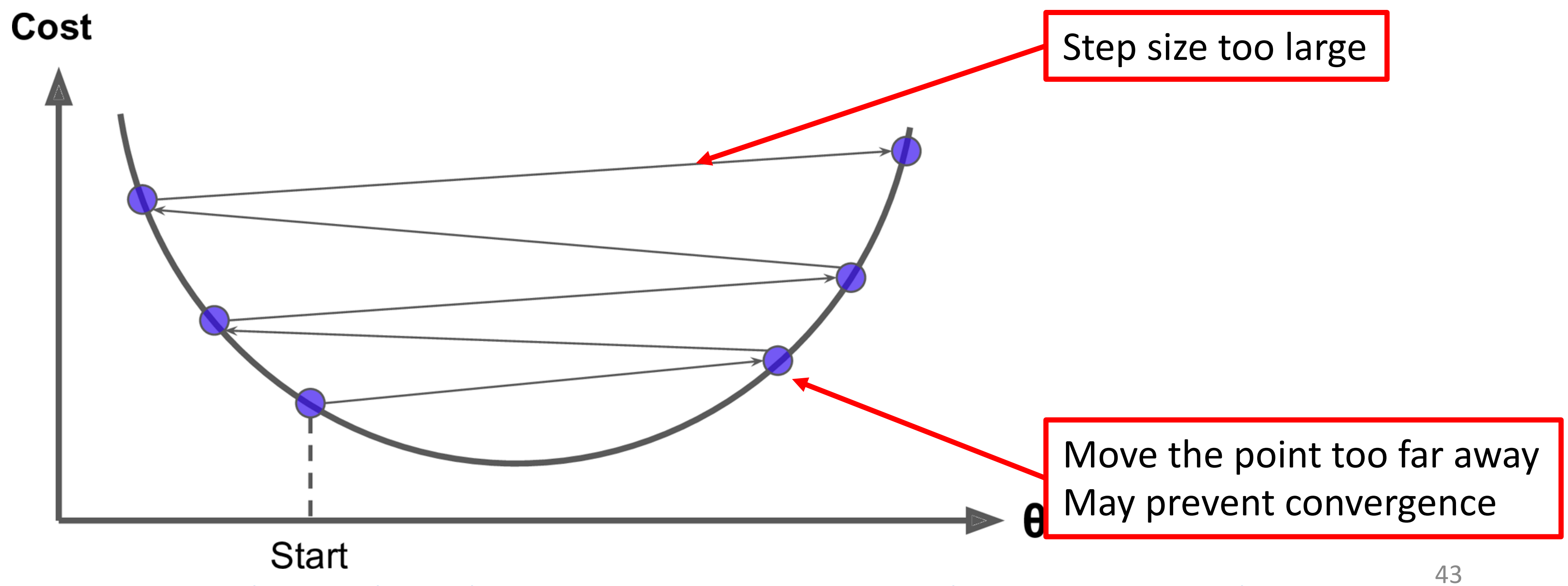


Image from https://www.oreilly.com/library/view/hands-on-machine-learning/9781491962282/ch04.html

Determining model parameters

- When to terminate GD (determining T)?
  - Main factors influencing convergence rate?
    - Step size (learning rate)

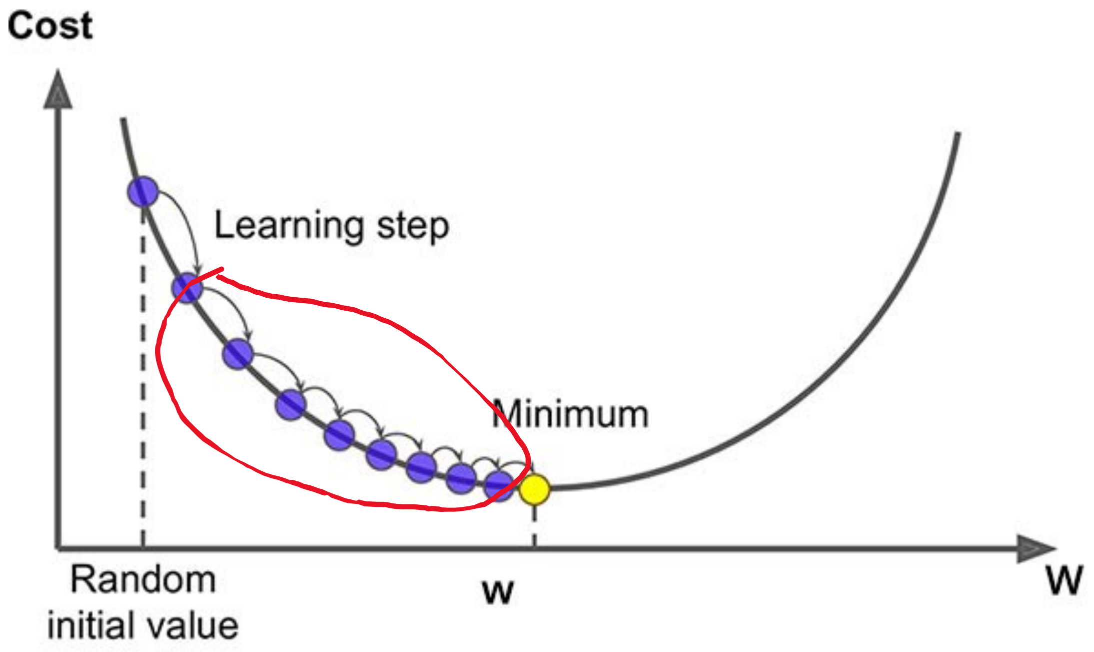


Image from https://morioh.com/p/15c995420be6

Determining model parameters

- When to terminate GD (determining T)?
  - Main factors influencing convergence rate?
    - Step size (learning rate)

Step size too small

May cost too many steps approaching the minimal solution

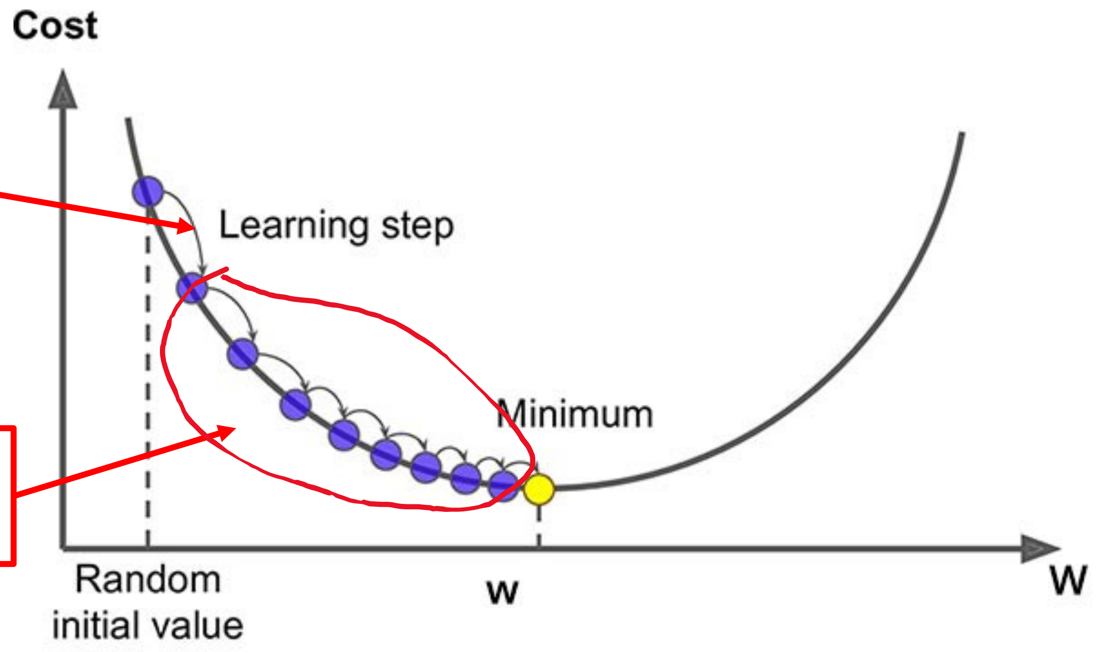


Image from https://morioh.com/p/15c995420be6

Determining model parameters

- When to terminate GD (determining T)?
  - Main factors influencing convergence rate?
    - Step size (learning rate)
    - Conditions of the objective function

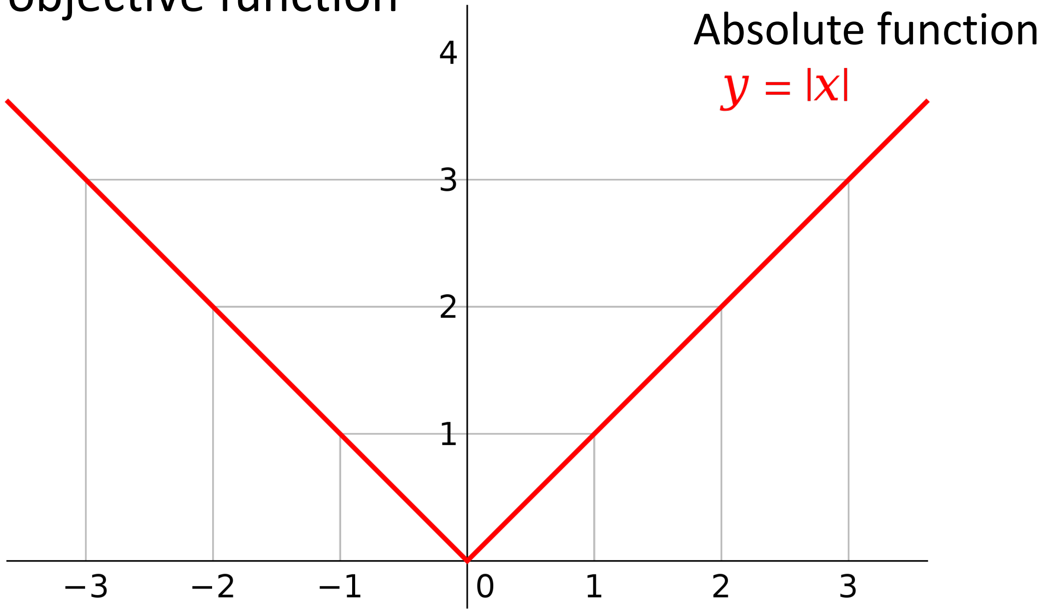


Image from https://en.wikipedia.org/wiki/Absolute_value#/media/File:Absolute_value.svg

Determining model parameters

- When to terminate GD (determining T)?
  - Main factors influencing convergence rate?
    - Step size (learning rate)
    - Conditions of the objective function

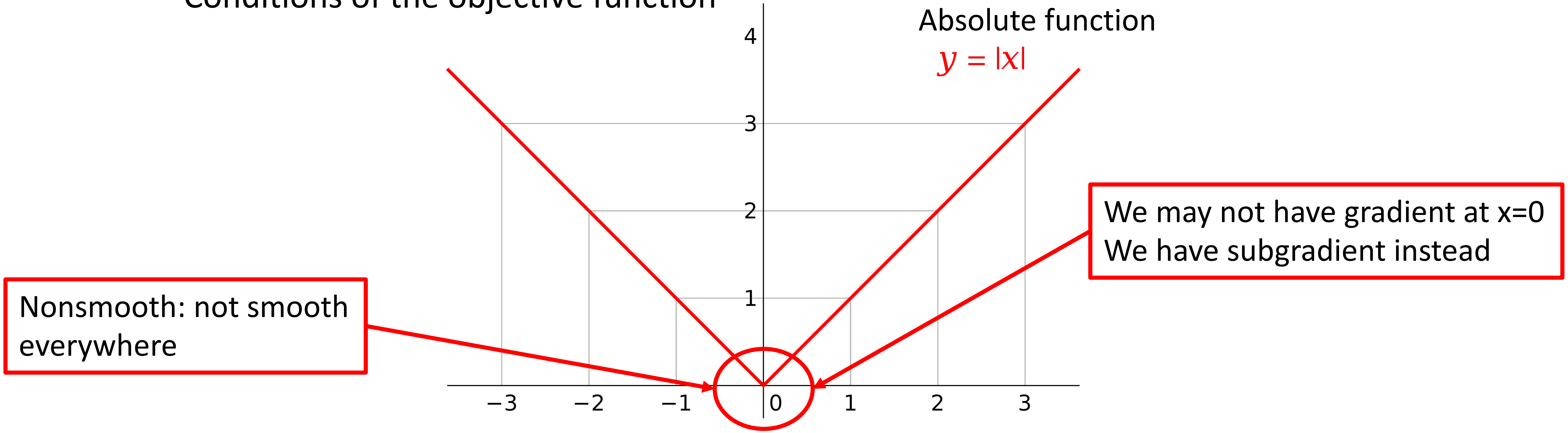


Image from https://en.wikipedia.org/wiki/Absolute_value#/media/File:Absolute_value.svg

Determining model parameters

- When to terminate GD (determining T)?
  - Main factors influencing convergence rate?
    - Step size (learning rate)
    - Conditions of the objective function

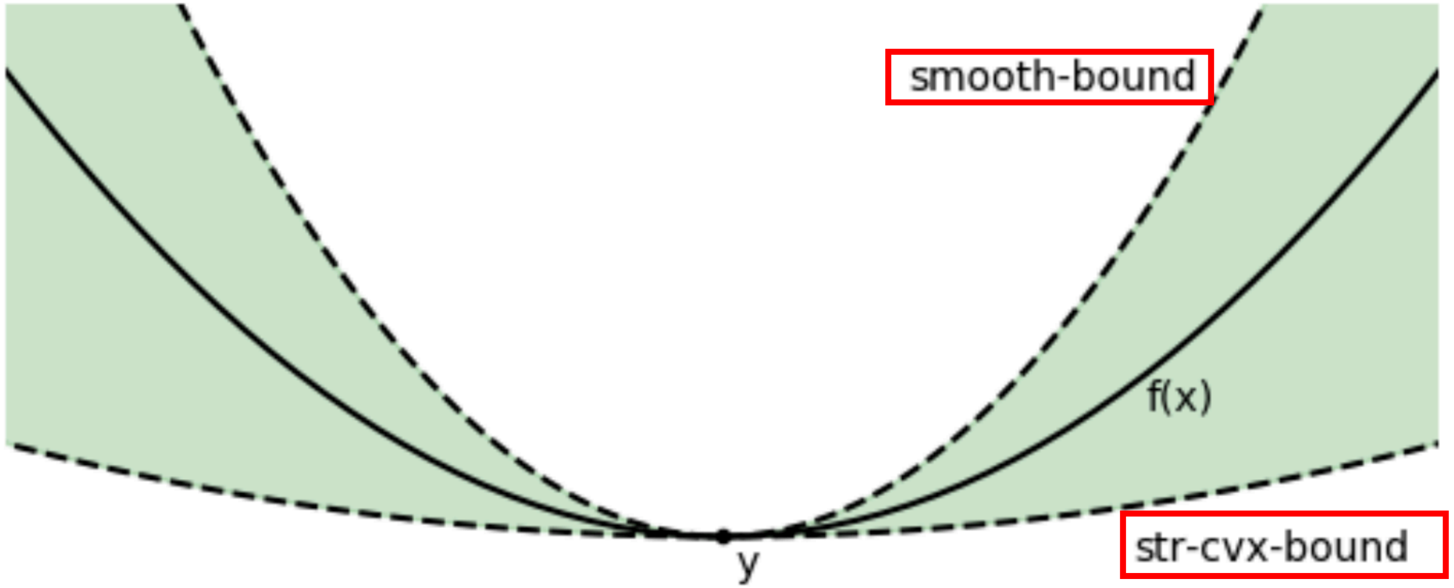


Image from IFT 6085 - Lecture 3 Gradients for smooth and for strongly convex functions
http://mitliagkas.github.io/ift6085-2019/ift-6085-lecture-3-notes.pdf

Determining model parameters

- When to terminate GD (determining T)?
  - Main factors influencing convergence rate?
    - Step size (learning rate)
    - Conditions of the objective function

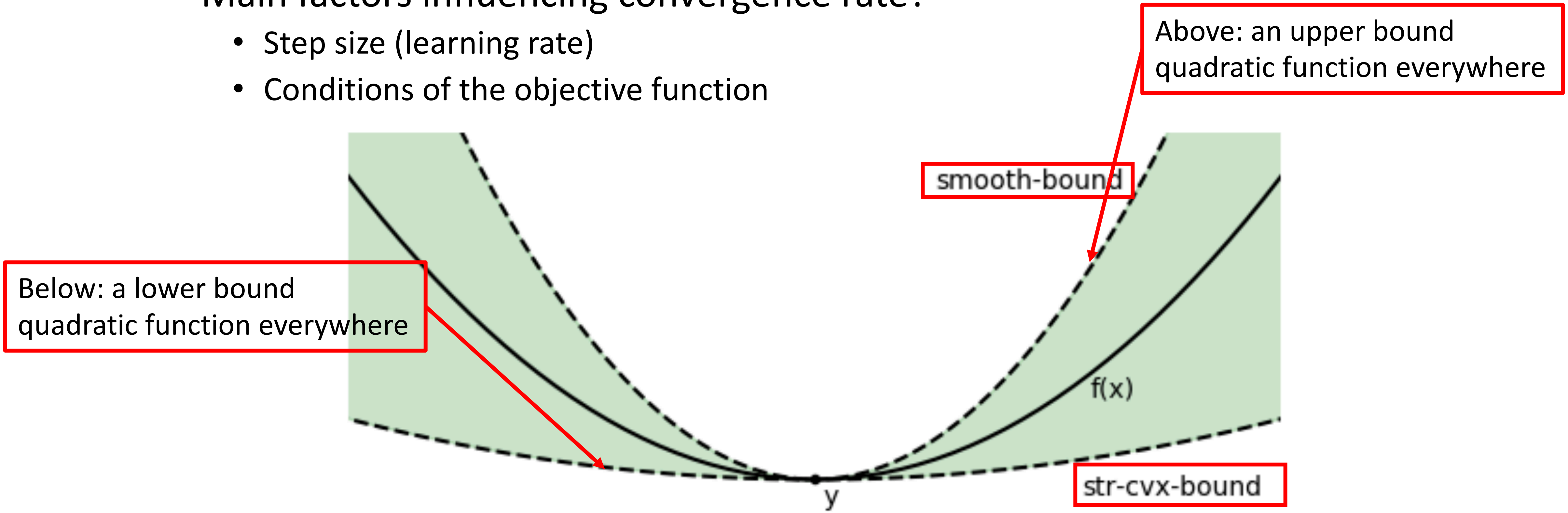


Image from IFT 6085 - Lecture 3 Gradients for smooth and for strongly convex functions
http://mitliagkas.github.io/ift6085-2019/ift-6085-lecture-3-notes.pdf

Determining model parameters

- When to terminate GD (determining T)?
  - Come back to our problem setting:
    Convergence rate for GD?

Determining model parameters

- When to terminate GD (determining T)?
  - Come back to our problem setting:

    Convergence rate for GD?

Theorem 2.1.14 *If* $f \in \mathcal{S}^{1,1}_{\mu,L}(R^n)$ *and* $0 < h \leq \frac{2}{\mu+L}$ *then the gradient method generates a sequence* $\{x_k\}$ *such that*

$$\| x_k - x^* \|^2 \leq \left(1 - \frac{2h\mu L}{\mu + L}\right)^k \| x_0 - x^* \|^2 .$$

If $h = \frac{2}{\mu+L}$ *then*

$$\| x_k - x^* \| \leq \left(\frac{Q_f-1}{Q_f+1}\right)^k \| x_0 - x^* \|,$$

$$f(x_k) - f^* \leq \frac{L}{2}\left(\frac{Q_f-1}{Q_f+1}\right)^{2k} \| x_0 - x^* \|^2,$$

where $Q_f = L/\mu$.

[Nesterov 2003]

Determining model parameters

- When to terminate GD (determining T)?
  - Come back to our problem setting:

    Convergence rate for GD?

Theorem 2.1.14 *If* $f \in \mathcal{S}^{1,1}_{\mu,L}(R^n)$ *and* $0 < h \leq \frac{2}{\mu+L}$ *then the gradient method generates a sequence* $\{x_k\}$ *such that*

$$\| x_k - x^* \|^2 \leq \left(1 - \frac{2h\mu L}{\mu + L}\right)^k \| x_0 - x^* \|^2 .$$

If $h = \frac{2}{\mu+L}$ *then*

$$\| x_k - x^* \| \leq \left(\frac{Q_f - 1}{Q_f + 1}\right)^k \| x_0 - x^* \|,$$

$$f(x_k) - f^* \leq \boxed{\frac{L}{2}\left(\frac{Q_f - 1}{Q_f + 1}\right)^{2k} \| x_0 - x^* \|^2,} \quad \boxed{= \epsilon(\mathrm{k}) = O(a^k)}$$

where $Q_f = L/\mu$.

[Nesterov 2003]

Determining model parameters

- When to terminate GD (determining T)?
  - Come back to our problem setting:
    Convergence rate for GD?

Theorem 2.1.14 *If $f \in \mathcal{S}^{1,1}_{\mu,L}(R^n)$ and $0 < h \leq \frac{2}{\mu+L}$ then the gradient method generates a sequence $\{x_k\}$ such that*

$$\| x_k - x^* \|^2 \leq \left(1 - \frac{2h\mu L}{\mu + L}\right)^k \| x_0 - x^* \|^2 .$$

If $h = \frac{2}{\mu+L}$ then

$$\| x_k - x^* \| \leq \left(\frac{Q_f - 1}{Q_f + 1}\right)^k \| x_0 - x^* \|,$$

$$f(x_k) - f^* \leq \boxed{\frac{L}{2}\left(\frac{Q_f - 1}{Q_f + 1}\right)^{2k} \| x_0 - x^* \|^2,} \quad \boxed{= \epsilon(\mathrm{k}) = O(a^k)} \quad \boxed{0 < a < 1}$$

$$\boxed{k = O(\, log_{1/a}(1/\epsilon))}$$

where $Q_f = L/\mu$.

[Nesterov 2003]

Determining model parameters

- Now we can answer the question:

  Can Gradient Descent (GD) do better?

GD	vs	Analytical solution
$O(dnT)$		$O(d^2n + d^{2.373})$

Determining model parameters

- Now we can answer the question:

  Can Gradient Descent (GD) do better?

$$\boxed{\begin{array}{c}\text{GD}\\ O(dnT)\end{array}} \quad \text{VS} \quad \boxed{\begin{array}{c}\text{Analytical solution}\\ O(d^2n + d^{2.373})\end{array}}$$

$$\downarrow$$

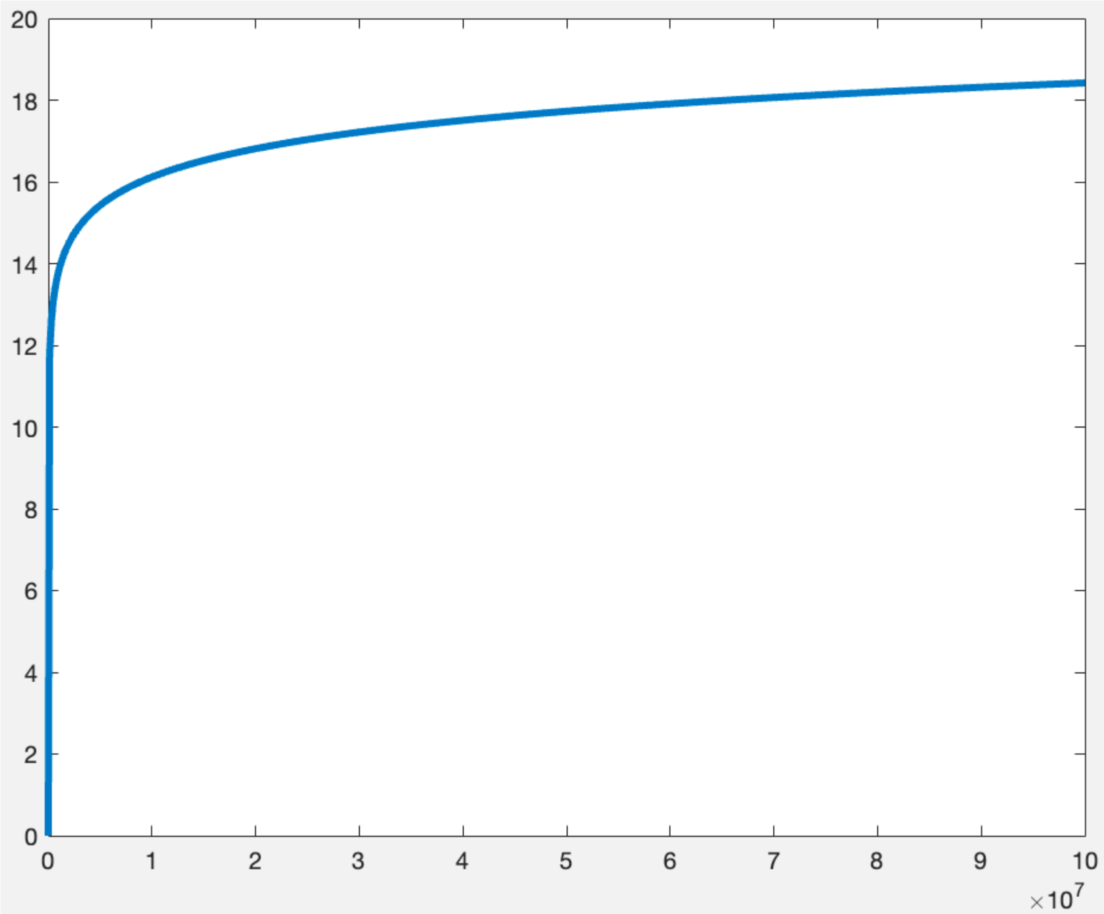
$$\boxed{O(dn{\times}log_{1/a}(1/\epsilon))}$$

Determining model parameters

- Is log term $log_{1/a}(1/\epsilon)$ large?

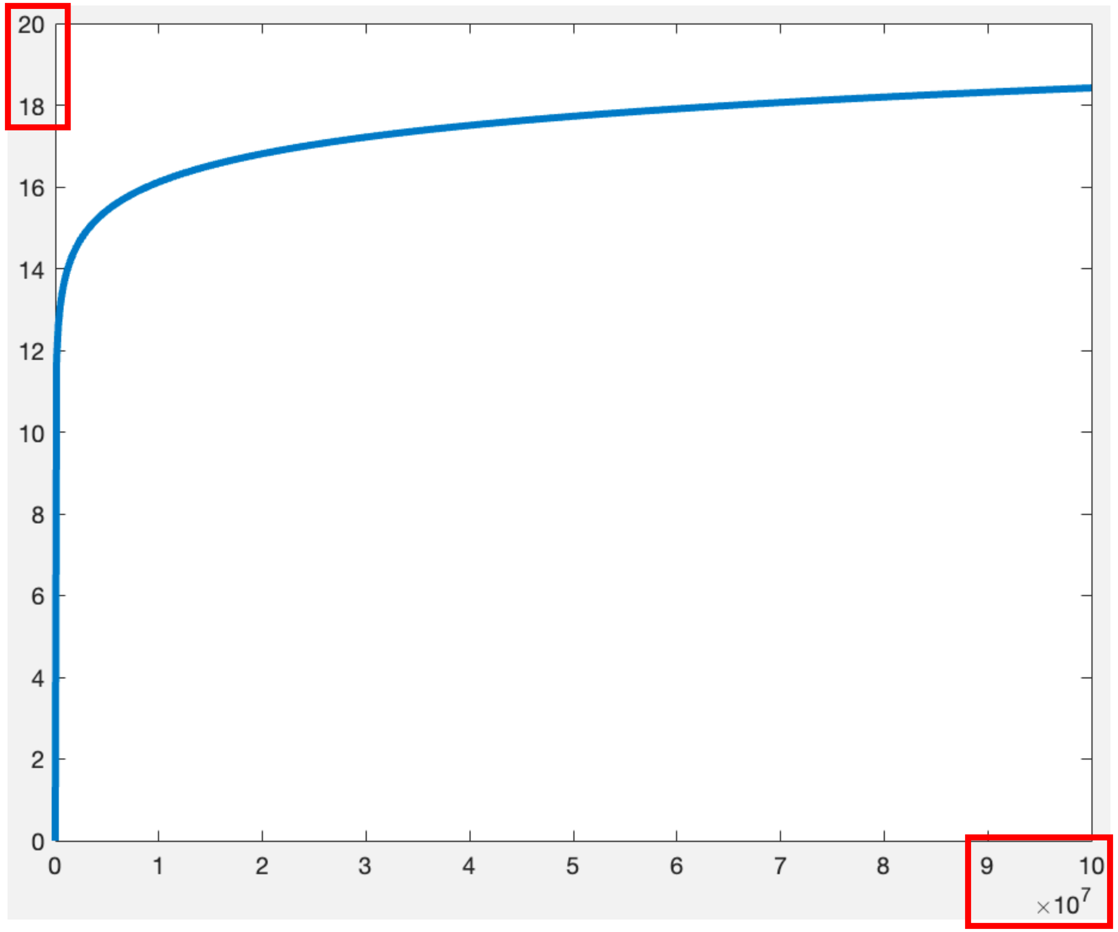
Determining model parameters

- Is log term $log_{1/a}(1/\epsilon)$ large?



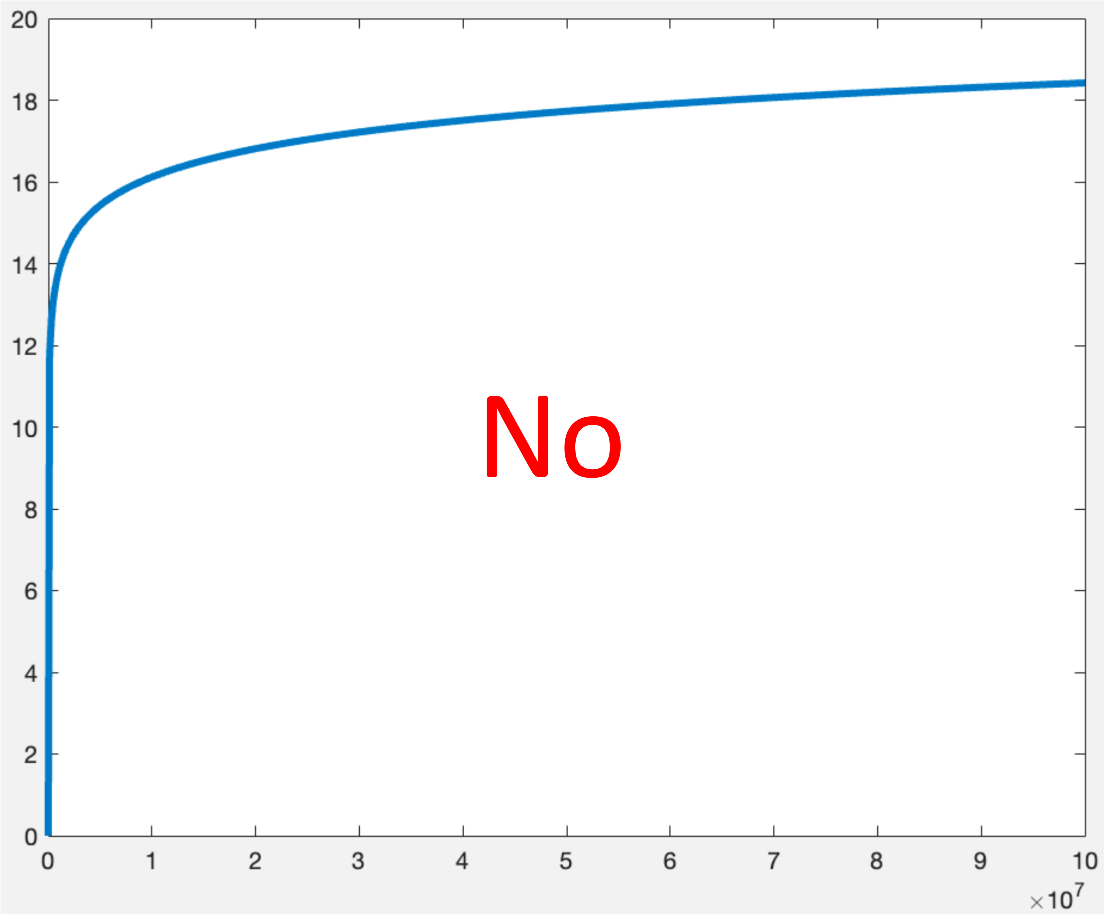
Determining model parameters

- Is log term $log_{1/a}(1/\epsilon)$ large?



Determining model parameters

- Is log term $log_{1/a}(1/\epsilon)$ large?



Determining model parameters

- Is d large?

Screenshot from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

name	source	type	class	training size	testing size	feature
a1a	UCI	classification	2	1,605	30,956	123
a2a	UCI	classification	2	2,265	30,296	123
a3a	UCI	classification	2	3,185	29,376	123
a4a	UCI	classification	2	4,781	27,780	123
a5a	UCI	classification	2	6,414	26,147	123
a6a	UCI	classification	2	11,220	21,341	123
a7a	UCI	classification	2	16,100	16,461	123
a8a	UCI	classification	2	22,696	9,865	123
a9a	UCI	classification	2	32,561	16,281	123
australian	Statlog	classification	2	690		14
avazu	Avazu's Click-through Prediction	classification	2	40,428,967	4,577,464	1,000,000
breast-cancer	UCI	classification	2	683		10
cod-rna	[AVU06a]	classification	2	59,535		8
colon-cancer	[AU99a]	classification	2	62		2,000
covtype.binary	UCI	classification	2	581,012		54
criteo	Criteo's Display Advertising Challenge	classification	2	45,840,617	6,042,135	1,000,000
criteo_tb	Criteo's Terabyte Click Logs	classification	2	4,195,197,692	178,274,637	1,000,000
diabetes	UCI	classification	2	768		8
duke breast-cancer	[MW01a]	classification	2	44		7,129
epsilon	PASCAL Challenge 2008	classification	2	400,000	100,000	2,000
fourclass	[TKH96a]	classification	2	862		2
german.numer	Statlog	classification	2	1,000		24
gisette	NIPS 2003 Feature Selection Challenge [IG05a]	classification	2	6,000	1,000	5,000
heart	Statlog	classification	2	270		13
HIGGS	UCI	classification	2	11,000,000		28
ijcnn1	[DP01a]	classification	2	49,990	91,701	22
ionosphere	UCI	classification	2	351		34
kdd2010 (algebra)	KDD CUP 2010	classification	2	8,407,752	510,302	20,216,830
kdd2010 (bridge to algebra)	KDD CUP 2010	classification	2	19,264,097	748,401	29,890,095
kdd2010 raw version (bridge to algebra)	KDD CUP 2010	classification	2	19,264,097	748,401	1,163,024
kdd2012	KDD CUP 2012	classification	2	149,639,105		54,686,452

Screenshot from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

name	source	type	class	training size	testing size	feature
a1a	UCI	classification	2	1,605	30,956	123
a2a	UCI	classification	2	2,265	30,296	123
a3a	UCI	classification	2	3,185	29,376	123
a4a	UCI	classification	2	4,781	27,780	123
a5a	UCI	classification	2	6,414	26,147	123
a6a	UCI	classification	2	11,220	21,341	123
a7a	UCI	classification	2	16,100	16,461	123
a8a	UCI	classification	2	22,696	9,865	123
a9a	UCI	classification	2	32,561	16,281	123
australian	Statlog	classification	2	690		14
avazu	Avazu's Click-through Prediction	classification	2	40,428,967	4,577,464	1,000,000
breast-cancer	UCI	classification	2	683		10
cod-rna	[AVU06a]	classification	2	59,535		8
colon-cancer	[AU99a]	classification	2	62		2,000
covtype.binary	UCI	classification	2	581,012		54
criteo	Criteo's Display Advertising Challenge	classification	2	45,840,617	6,042,135	1,000,000
criteo_tb	Criteo's Terabyte Click Logs	classification	2	4,195,197,692	178,274,637	1,000,000
diabetes	UCI	classification	2	768		8
duke breast-cancer	[MW01a]	classification	2	44		7,129
epsilon	PASCAL Challenge 2008	classification	2	400,000	100,000	2,000
fourclass	[TKH96a]	classification	2	862		2
german.numer	Statlog	classification	2	1,000		24
gisette	NIPS 2003 Feature Selection Challenge [IG05a]	classification	2	6,000	1,000	5,000
heart	Statlog	classification	2	270		13
HIGGS	UCI	classification	2	11,000,000		28
ijcnn1	[DP01a]	classification	2	49,990	91,701	22
ionosphere	UCI	classification	2	351		34
kdd2010 (algebra)	KDD CUP 2010	classification	2	8,407,752	510,302	20,216,830
kdd2010 (bridge to algebra)	KDD CUP 2010	classification	2	19,264,097	748,401	29,890,095
kdd2010 raw version (bridge to algebra)	KDD CUP 2010	classification	2	19,264,097	748,401	1,163,024
kdd2012	KDD CUP 2012	classification	2	149,639,105		54,686,452

1,000,000

1,000,000
1,000,000

20,216,830
29,890,095
1,164,024
54,686,452

Screenshot from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

Determining model parameters

- Is d large?

  Yes!

Determining model parameters

- Is d large?

  Yes!

GD	vs	Analytical solution
$O(dnT)$		$O(d^2n + d^{2.373})$

Determining model parameters

- Is d large?

  Yes!

GD
$O(dnT)$

VS

Analytical solution
$O(d^2n + d^{2.373})$

$O(dn \times log_{1/a}(1/\epsilon))$

Determining model parameters

- Is d large?

  Yes!

GD
$O(dnT)$

$O(dn \times log_{1/a}(1/\epsilon))$

vs

Analytical solution
$O(d^2n + d^{2.373})$

If:
$n = 10^8$
$d = 10^6$

Determining model parameters

- Is d large?

  Yes!

GD
$O(dnT)$

VS

Analytical solution
$O(d^2n + d^{2.373})$

$O(dn \times log_{1/a}(1/\epsilon))$

If:
$n = 10^8$
$d = 10^6$

$(10^6)^2 \times 10^8 - 10^6 \times 10^8 = 10^{20} - 10^{14}$

Determining model parameters

- Is d large?

  Yes!

GD

$O(dnT)$

vs

Analytical solution

$O(d^2 n + d^{2.373})$

$O(dn \times log_{1/a}(1/\epsilon))$

If:

$n = 10^8$

$d = 10^6$

$(10^6)^2 \times 10^8 - 10^6 \times 10^8 = 10^{20} - 10^{14} \approx 10^{20}$

Determining model parameters

- Is d large?

  Yes!

GD
$O(dnT)$

VS

Analytical solution
$O(d^2n + d^{2.373})$

$O(dn \times log_{1/a}(1/\epsilon))$

If:
$n = 10^8$
$d = 10^6$

$(10^6)^2 \times 10^8 - 10^6 \times 10^8 = 10^{20} - 10^{14} \approx 10^{20}$

$O\left(dn \times log_a\left(\frac{1}{\epsilon}\right)\right) \ll O(d^2n + n^{2.373})$

Other optimization algorithms

- First-order algorithms
  - Gradient descent
  - Momentum methods
  - Stochastic variants
  - Hessian vector products
- Second-order algorithms
  - Newton method
  - Quasi-newton method

Other optimization algorithms

- First-order algorithms (commonly used and researched in machine learning)
  - Gradient descent
  - Momentum methods
  - Stochastic variants
  - Hessian vector products
- Second-order algorithms
  - Newton method
  - Quasi-newton method

References

- Jung, Alexander. (2018). Machine Learning: Basic Principles.
- Nesterov, Yurii. *Introductory lectures on convex optimization: A basic course*. Vol. 87. Springer Science & Business Media, 2003.