Optimization Methods for Deep Learning

Gradient descent for non-convex optimization

Descent Lemma: Let $f : \mathbb{R}^d \to \mathbb{R}$ be twice differentiable, and $\|\nabla^2 f\|_2 \leq \beta$. Then setting the learning rate $\eta = 1/\beta$, and applying gradient descent, $x_{t+1} = x_t - \eta \nabla f(x_t)$, we have:

$$f(x_t) - f(x_{t+1}) \geq \frac{1}{2\beta} \|\nabla f(x_t)\|_2^2.$$

Converging to stationary points

Theorem: In $T = O(\frac{\beta}{\epsilon^2})$ iterations, we have $\|\nabla f(x)\|_2 \leq \epsilon$.

Gradient Descent for Quadratic Functions

optimal $x^* = 0$ / init: x_0

Problem: $\min_x \frac{1}{2} x^\top A x$ with $A \in \mathbb{R}^{d \times d}$ being positive-definite. $\succ 0$

Theorem: Let $\lambda_{\max}$ and $\lambda_{\min}$ be the largest and the smallest eigenvalues of A. If we set $\eta \leq \frac{1}{\lambda_{\max}}$, we have

$$\|x_t\|_2 \leq \left(1 - \eta\lambda_{\min}\right)^t \|x_0\|_2$$

$$\begin{aligned} \|x_{t+1}\|_2 &= \|x_t - \eta A x_t\|_2 \\ &= \|(I - \eta A) x_t\|_2 \\ &\leq \|I - \eta A\|_2 \cdot \|x_t\|_2 \\ &\leq (1 - \eta\lambda_{\min}) \cdot \|x_t\|_2 \\ &\leq (1 - \eta\lambda_{\min})^{t+1} \|x_0\|_2 \end{aligned}$$

linear convergence

If want $\|x_T\|_2 \leq \varepsilon$

we set $\eta = \frac{1}{\lambda_{\max}}$

$\Rightarrow$ need $O\left(\frac{\lambda_{\max}}{\lambda_{\min}} \log\left(\frac{1}{\varepsilon}\right)\right)$

call $\kappa \triangleq \frac{\lambda_{\max}}{\lambda_{\min}}$ condition number

can be generalized to strongly convex function

Momentum: Heavy-Ball Method (Polyak '64)

$v_0 = 0$
$v_1 = -\nabla f(x_0)$

Problem: $\min_x f(x)$

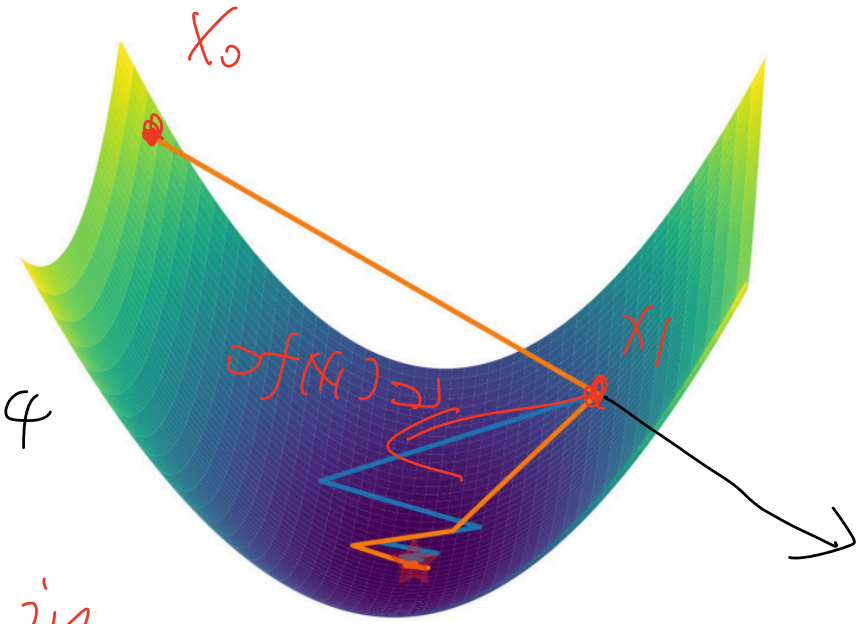
Method:
$$v_{t+1} = -\nabla f(x_t) + \beta v_t$$
$$x_{t+1} = x_t + \eta v_{t+1}$$

For quadratic optimization

$\sqrt{\kappa} \log\left(\frac{1}{\varepsilon}\right)$

vs. gradient $\kappa \cdot \log\left(\frac{1}{\varepsilon}\right)$

$\kappa : 10^6 \Rightarrow$ improve $10^6 \to 10^4$



However, does not hold in general strongly convex function

Momentum: Nesterov Acceleration (Nesterov '89)

look ahead

Problem: $\min_x f(x)$

Method:

$$v_{t+1} = -\nabla f(x_t + \beta v_t) + \beta v_t$$

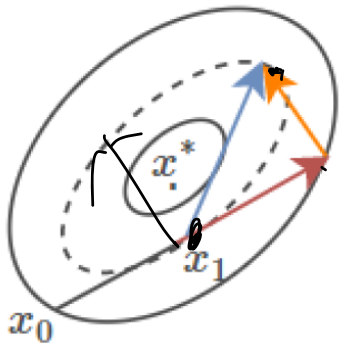
$$x_{t+1} = x_t + \eta v_{t+1}$$

For general strongly convex functions

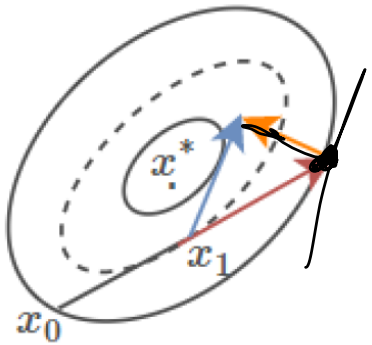
$\sqrt{\kappa} \cdot \log\left(\frac{1}{\epsilon}\right)$

tight

Polyak's Momentum



Nesterov Momentum



$x \in \mathbb{R}^d$ $O(d^2)$ inverse $O(d^3)$

Newton's Method

$d \times d$

Newton's Method: $x_{t+1} = x_t - \eta(\nabla^2 f(x_t))^{-1} \nabla f(x_t)$

GD: $x_{t+1} = x_t - \eta \nabla f(x_t)$

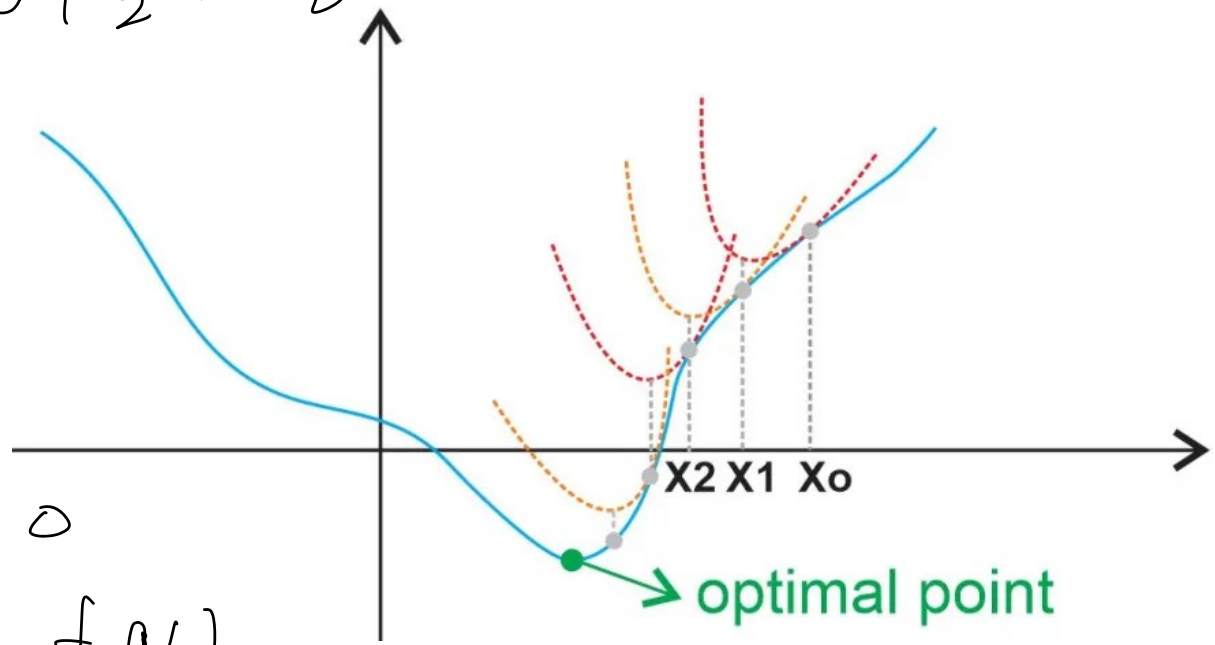
$f(x+\delta) \approx f(x) + \delta^T \nabla f(x) + \frac{1}{2}\|\delta\|_2^2$

$\delta^* = -\nabla f(x)$

Newton

$f(x+\delta) \approx f(x) + \delta^T \nabla f(x) + \frac{1}{2}\delta^T \nabla^2 f(x)\,\delta$

$\Rightarrow \delta^* = -(\nabla^2 f(x))^{-1} \nabla f(x)$



Theorem: $O(\log\log(\frac{1}{\varepsilon}))$ quadratic convergence

AdaGrad (Duchi et al. '11)

suppose diagonal

Newton Method: $x_{t+1} = x_t - \eta(\nabla^2 f(x_t))^{-1} \nabla f(x_t)$

AdaGrad: separate learning rate for every parameter

diagonal

$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1} \nabla f(x_t), \; (G_t)_{ii} = \sqrt{\sum_{j=1}^{t-1} \left(\nabla f(x_t)_i\right)^2}$$

invertible

in general

pre-conditioner

RMSProp (Hinton et al. '12)

AdaGrad: separate learning rate for every parameter

$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1} \nabla f(x_t), \; (G_t)_{ii} = \sqrt{\sum_{j=1}^{t-1} \left(\nabla f(x_t)_i \right)^2}$$

strictly increasing

≥ 0

RMSProp: exponential weighting of gradient norms

$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1/2} \nabla f(x_t),$$
$$(G_{t+1})_{ii} = \beta(G_t)_{ii} + (1-\beta)(\nabla f(x_t)_i)^2$$

$0 < \beta < 1$

exponential weighting

$\beta^{t+1} \nabla f(x_0)$

AdaDelta (Zeiler '12)

RMSProp:

$$x_{t+1} = x_t - \eta (G_{t+1} + \epsilon I)^{-1/2} \nabla f(x_t),$$
$$(G_{t+1})_{ii} = \beta (G_t)_{ii} + (1-\beta)(\nabla f(x_t)_i)^2$$

AdaDelta:

$$x_{t+1} = x_t - \eta \Delta x_t,$$
$$\Delta x_t = \sqrt{u_t + \epsilon} \cdot (G_{t+1} + \epsilon I)^{-1/2} \nabla f(x_t)$$
$$(G_{t+1})_{ii} = \beta (G_t)_{ii} + (1-\beta)(\nabla f(x_t)_i)^2,$$
$$u_{t+1} = \rho u_t + (1-\rho) \|\Delta x_t\|_2^2$$

Adam (Kingma & Ba '14)

Momentum:

$$v_{t+1} = -\nabla f(x_t) + \beta v_t,\ x_{t+1} = x_t + \eta v_{t+1}$$

RMSProp: exponential weighting of gradient norms

$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1} \nabla f(x_t),$$
$$(G_t)_{ii} = \beta(G_t)_{ii} + (1-\beta)(\nabla f(x_t)_i)^2$$

Adam

$$v_{t+1} = \beta_1 v_t + (1-\beta_1)\nabla f(x_t)$$
$$(G_{t+1})_{ii} = \beta_2(G_t)_{ii} + (1-\beta_2)(\nabla f(x_t)_i)^2$$
$$x_{t+1} = x_t - \eta(G_{t+1} + \epsilon I)^{-1/2} v_{t+1}$$

Default choice nowadays.

Important Techniques in Neural Network Training

Gradient Explosion / Vanishing

- Deeper networks are harder to train:
  - Intuition: gradients are products over layers
  - Hard to control the learning rate

$$f(x, W_1, \ldots, W_{H+1}) = W_{H+1}\,\sigma(W_H \cdots \sigma(W_1 x) \cdots)$$

$$\frac{\partial f}{\partial W_h} = (W_{H+1} A_H \cdots W_{h+1} A_h)^T (A_{h-1} W_{h-1} \cdots W_1 x)$$

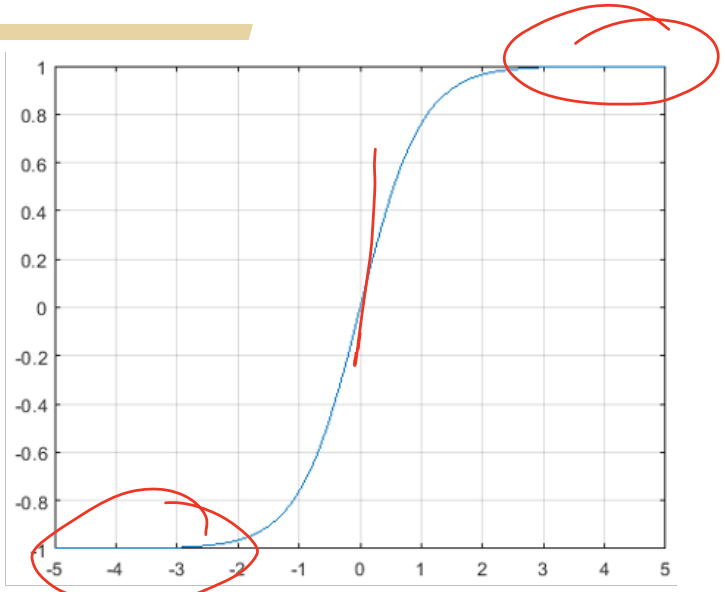
$$A_h = \mathrm{diag}(\sigma'(W_h \sigma(\cdots \sigma(W_1 x) \cdots)))$$

If W magnitude large → gradient explosion

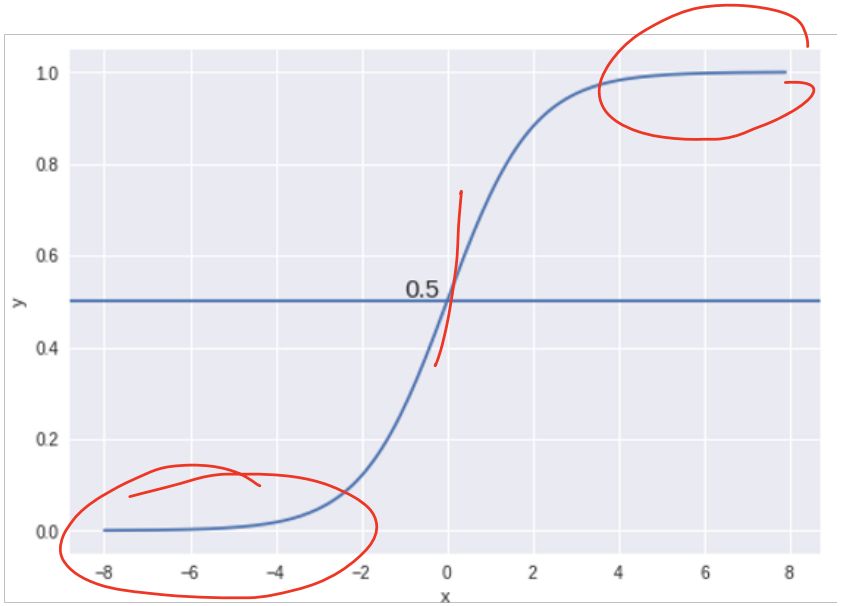
magnitude small → gradient vanishing

A: ↗

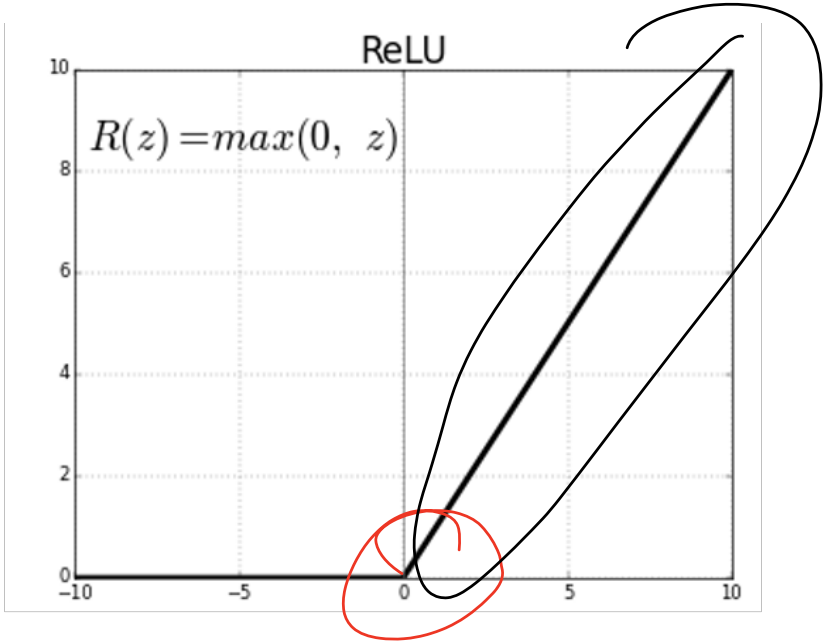
Activation Functions



tanh



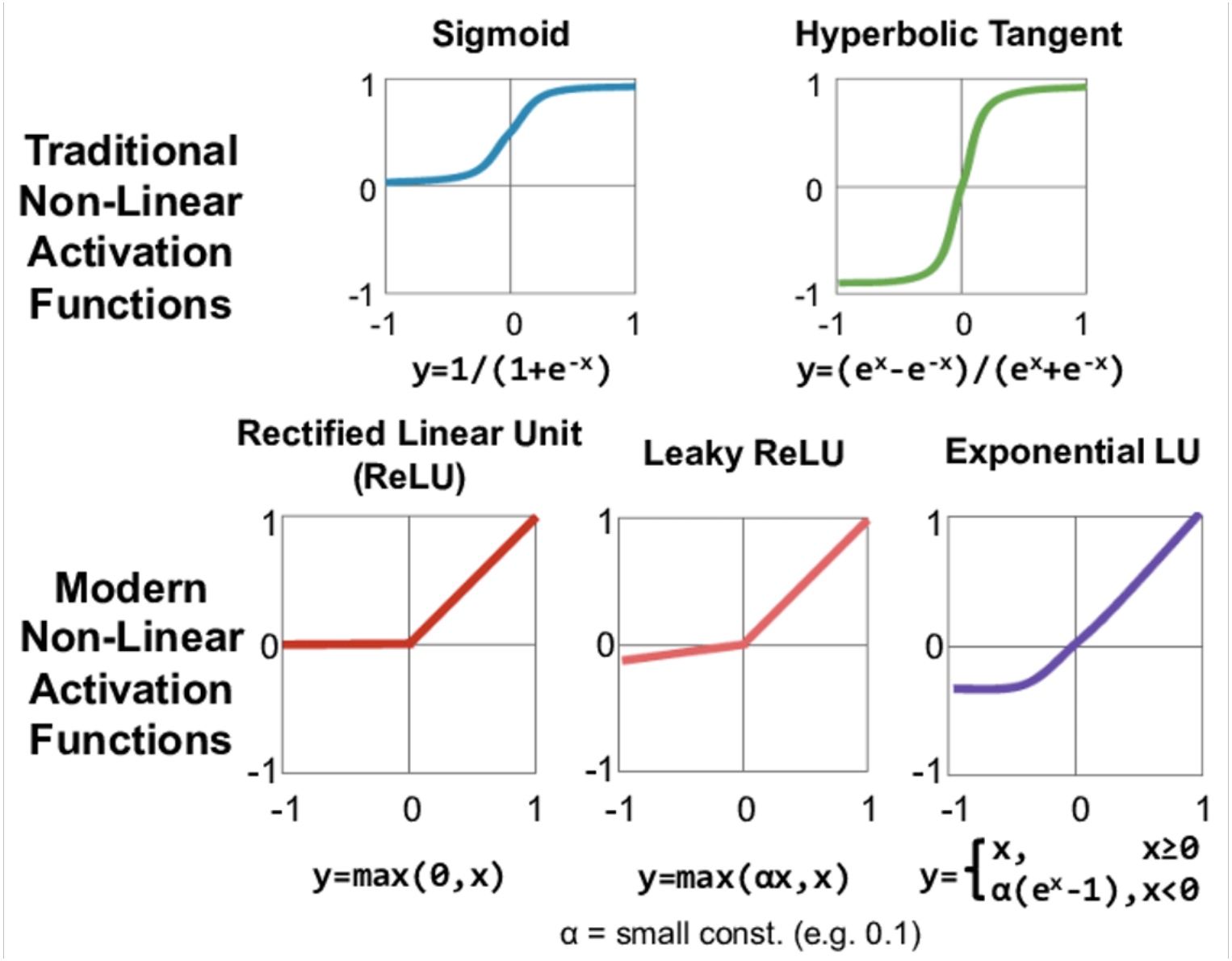
sigmoid



Rectified Linear United

$= 1$

Activation Function



Initialization

- Zero-initialization
- Large initialization
- Small initialization

- Design principles:
  - Zero activation mean

  - Activation variance remains same across layers