Deep Learning

Dr. Mehran Safayani

safayani@iut.ac.ir

safayani.iut.ac.ir

https://www.aparat.com/mehran.safayani

https://github.com/safayani/deep_learning_course

Department of Electrical and computer engineering, Isfahan university of technology, Isfahan, Iran

Optimization Algorithms

Understanding exponentially weighted averages

Exponential moving average

$v_0 = 0$

$v_1 = 0.9v_0 + 0.1\theta_1$

$v_2 = 0.9v_1 + 0.1\theta_2$

$v_3 = 0.9v_2 + 0.1\theta_3$

.

.

.

$v_t = 0.9v_{t-1} + 0.1\theta_t$

$v_t = \beta v_{t-1} + (1-\beta)\theta_t$

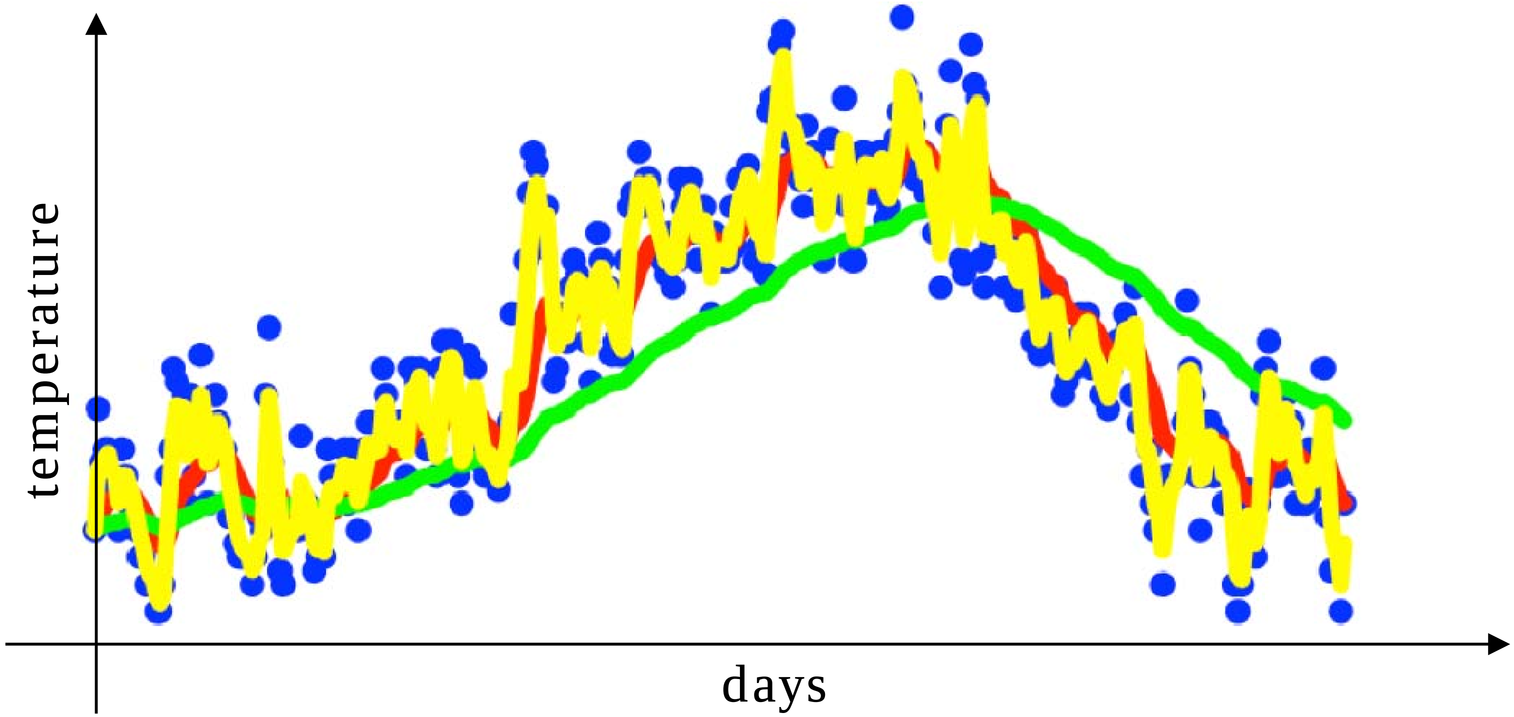
$\beta = 0.9 \approx$ 10 days $\qquad \frac{1}{1-\beta} days$

$\beta = 0.98 \approx$ 50 days $\qquad \frac{1}{1-0.98} = 50$

$\beta = 0.5 \approx$ 2 days

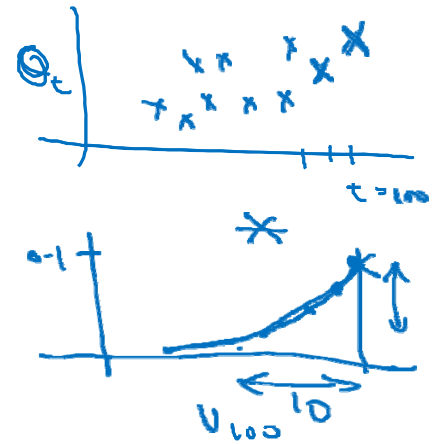
Exponentially weighted averages

- $v_t = \beta v_{t-1} + (1-\beta)\theta_t$



Exponentially weighted averages

- $V_t = \beta V_{t-1} + (1-\beta)\theta_t$
- $V_{100} = 0.9V_{99} + 0.1\theta_{100}$
- $V_{99} \;\; = 0.9V_{98} + 0.1\theta_{99}$
- $V_{98} = 0.9V_{97} + (1-\beta)\theta_t$



$\approx \frac{1}{1-\beta}$

$\varepsilon = 1-\beta$

……..

- $V_{100} = 0.1\theta_{100} + 0.9(0.9V_{98} + 0.1\theta_{99}) = \mathbf{0.1}\boldsymbol{\theta}_{\mathbf{100}} + \mathbf{0.1}\times\,\mathbf{0.9}\boldsymbol{\theta}_{\mathbf{99}} + \mathbf{0.1}\times\,(\mathbf{0.9})^{\mathbf{2}}\boldsymbol{\theta}_{\mathbf{98}} + \mathbf{0.1}\times\,(\mathbf{0.9})^{\mathbf{3}}\boldsymbol{\theta}_{\mathbf{97}} + \mathbf{0.1}\times\,(\mathbf{0.9})^{\mathbf{4}}\boldsymbol{\theta}_{\mathbf{96}}$

$$\boxed{\begin{gathered}(\mathbf{0.9})^{\mathbf{10}} \cong \mathbf{0.35} = \frac{\mathbf{1}}{\boldsymbol{e}} \\ \boldsymbol{\varepsilon} = \mathbf{1} - \boldsymbol{\beta}\end{gathered}}$$

$$\boxed{\begin{gathered}(1-\varepsilon)^{1/\varepsilon} = \frac{1}{e} \\ ((1-0.1)^{1/0.1} = \frac{1}{e} \\ \boldsymbol{\beta} = \mathbf{0.98} \\ \mathbf{0.98} = ((1-0.02)^{1/0.02} \cong e \cong 0.35\end{gathered}}$$

Implementing exponentially weighted averages

$V_\theta = 0$

$V_\theta = V_\theta + (1-\beta)\theta_1$

$V_\theta = V_\theta + (1-\beta)\theta_2$

$V_\theta = 0$

Repeat {

  Get Next θ_t

$V_\theta = V_\theta + (1-\beta)\theta_t$

}

Bias correction in exponentially weighted average

- **Bias correction**:
- $V_t = \beta V_{t-1} + (1-\beta)\theta_t$

  $V_0 = 0$

  $V_1 = 0.98V_0 + 0.02\theta_1$

  $V_2 = 0.98V_1 + 0.02\theta_2$

  $= 0.98 \times 0.02\theta_1 + 0.02\theta_2$

  $= 0.0196\,\theta_1 + 0.02\theta_2$

$$\boxed{V_t = \frac{V_t}{1-\beta^t}}$$

t=2

$1-\beta^t = 1-(0.98)^2 = 0.0396$

$\tilde{V}_2 = \frac{V_2}{0.0396} = \frac{0.0196\theta_1 + 0.02\theta_2}{0.0396}$

$$\boxed{V_t = \frac{V_t}{1-\beta^t}} \Longrightarrow$$

t بزرگ شود $\longrightarrow$ $\boxed{\beta^t = 0}$

Optimization Algorithms

Gradient descent with momentum

- Momentum:

On iteration t:

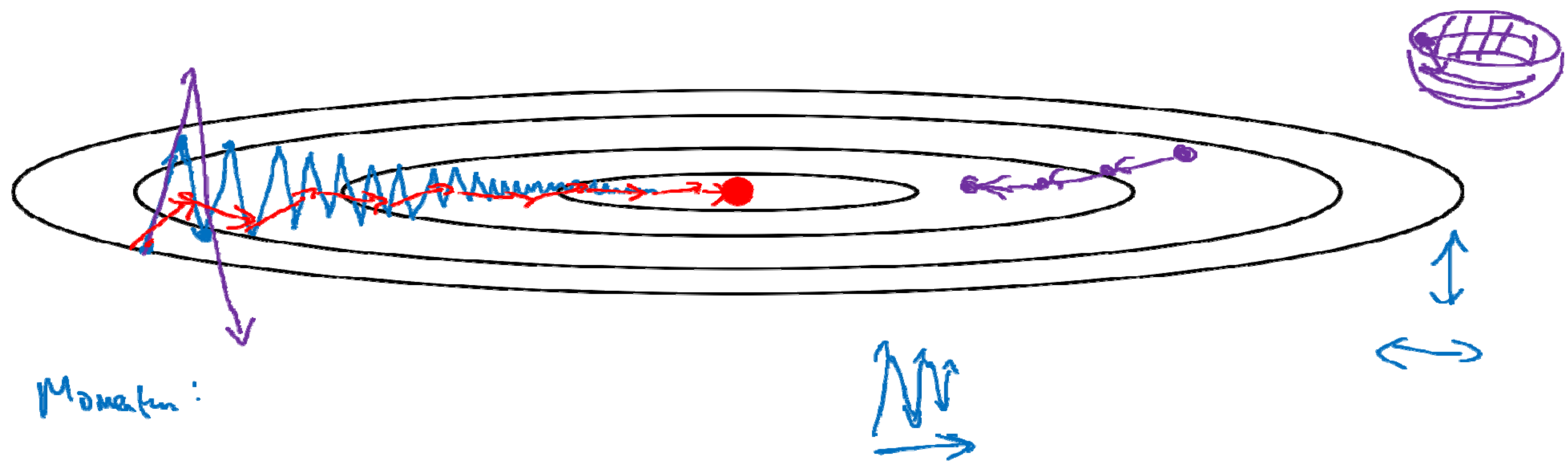
Compute dw, db on the current mini-batch

$$v_{dW} = \beta v_{dW} + (1-\beta) dw$$

$$v_{db} = \beta v_{db} + (1-\beta) db$$

$$w = w - \alpha v_{dW}, \; b = b - \alpha v_{db}$$

$$\boxed{V_t = \beta V_{t-1} + (1-\beta)\theta_t}$$



Momentum:

Gradient descent with momentum

Implementation details

$v_{dw} = 0 \quad , v_{db} = 0$

On Iteration t:

$v_{\mathrm{dw}} = \beta v_{\mathrm{dw}} + (1-\beta)\mathrm{dw} \equiv v_{\mathrm{dw}} = \beta v_{\mathrm{dw}} + \mathrm{dw}$

$v_{\mathrm{db}} = \beta v_{\mathrm{db}} + \mathrm{db}$

w=w- αv_{dw}

b = b- αv_{db}

$$\alpha[\, \beta v_{\mathrm{dw}} + (1-\beta)\mathrm{dw}] = \underbrace{(1-\beta)\alpha}_{\acute{\alpha}}[\underbrace{\frac{\beta}{1-\beta}}_{\acute{\beta}} v_{dw} + \mathrm{dw}] = \acute{\alpha}\,[\,\acute{\beta} v_{\mathrm{dw}} + \mathrm{dw}]$$

Optimization Algorithms

RMSprop(Root Mean Square Propagation)

- RMS Prop:

On iteration t:

Compute *dw*, *db* on current mini-batch

$$S_{dw} = \beta_2 S_{dw} + (1-\beta_2)dw^2$$

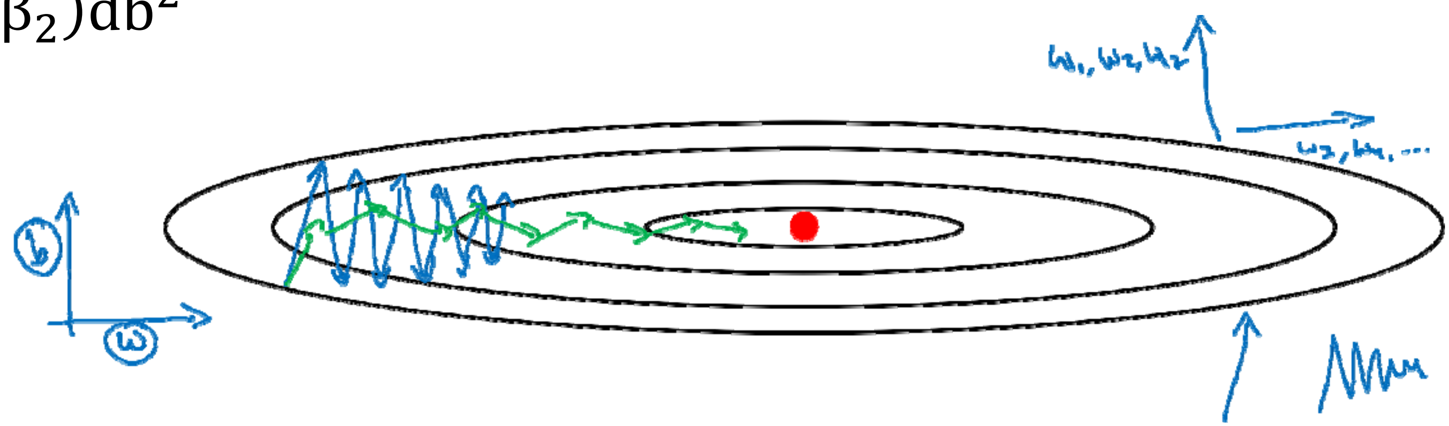
Element wise

$$S_{db} = \beta_2 S_{db} + (1-\beta_2)db^2$$

$$w = w - \alpha \frac{dw}{\sqrt{S_{dw}+\varepsilon}}$$

$$b = b - \alpha \frac{db}{\sqrt{S_{db}+\varepsilon}}$$

$$\boxed{\varepsilon = 10^{-8}}$$



Optimization Algorithms

Adam: Adaptive momentum estimation

- Adam:

$v_{dw} = 0, S_{db} = 0 , v_{db} = 0 , S_{db} = 0$

On iteration t:

Compute *dw, db* using current mini-batch

momentum $v_{\mathrm{dw}} = \beta_1 v_{\mathrm{dw}} + (1-\beta_1)\mathrm{dw} \;,\; v_{\mathrm{db}} = \beta_1 v_{\mathrm{db}} + (1-\beta_1)\mathrm{db}$

RMS Prop $\mathrm{S_{dw}} = \beta_2 \mathrm{S_{dw}} + (1-\beta_2)\mathrm{dw}^2 \;,\; \mathrm{S_{db}} = \beta_2 \mathrm{S_{db}} + (1-\beta_2)\mathrm{db}^2$

$$v_{dw}^{corrected} = \frac{v_{\mathrm{dw}}}{1-\beta_1^t} \;, \quad v_{db}^{corrected} = \frac{v_{\mathrm{db}}}{1-\beta_1^t}$$

$$S_{dw}^{corrected} = \frac{\mathrm{S_{dw}}}{1-\beta_2^t} , \quad S_{db}^{corrected} = \frac{\mathrm{S_{db}}}{1-\beta_2^t}$$

$$\mathrm{w=w-}\ \alpha \frac{v_{dw}^{corrected}}{\sqrt{S_{dw}^{corrected}+\varepsilon}} \quad , \mathrm{b=b-}\ \alpha \frac{v_{db}^{corrected}}{\sqrt{S_{db}^{corrected}+\varepsilon}}$$

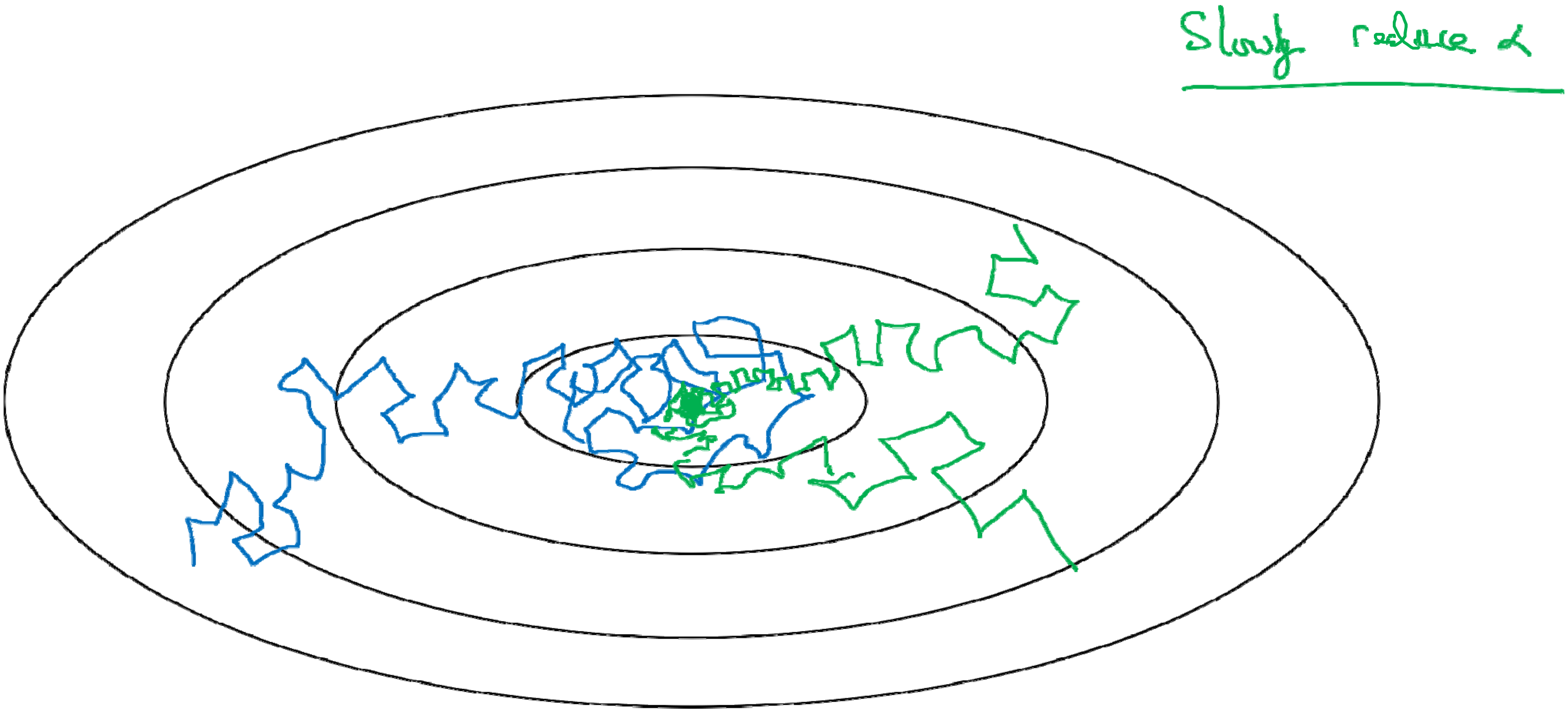
$\alpha =$ needs to be tuned

$\beta_1 = 0.9$

$\beta_2 = 0.999$

$\varepsilon = 10^{-8}$

Optimization Algorithms
Learning rate decay



Optimization Algorithms

Learning rate decay

- $\alpha = \frac{1}{1+(decay\text{-}rate)*(epoch\text{-}num)}\alpha_0$

Epoch	α
1	0.1
2	0.67
3	0.05
4	0.04
...	...

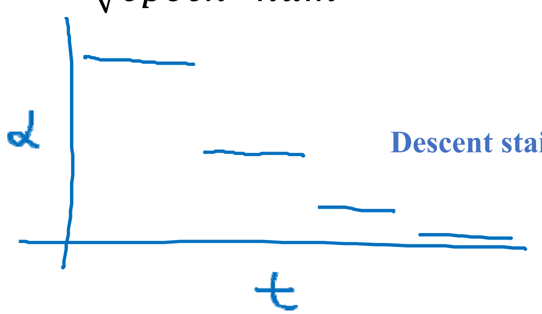
$\alpha_0 = 0.02$

decay-rate =1

- $\alpha = (0.95)^{\#epoch\text{-}num} \times \alpha_0$
- $\alpha = \frac{k}{\sqrt{epoch\text{-}num}} \times \alpha_0$

$\alpha = \frac{}{\sqrt{t}} \times \alpha_0$

#mini-batch



Descent staircase