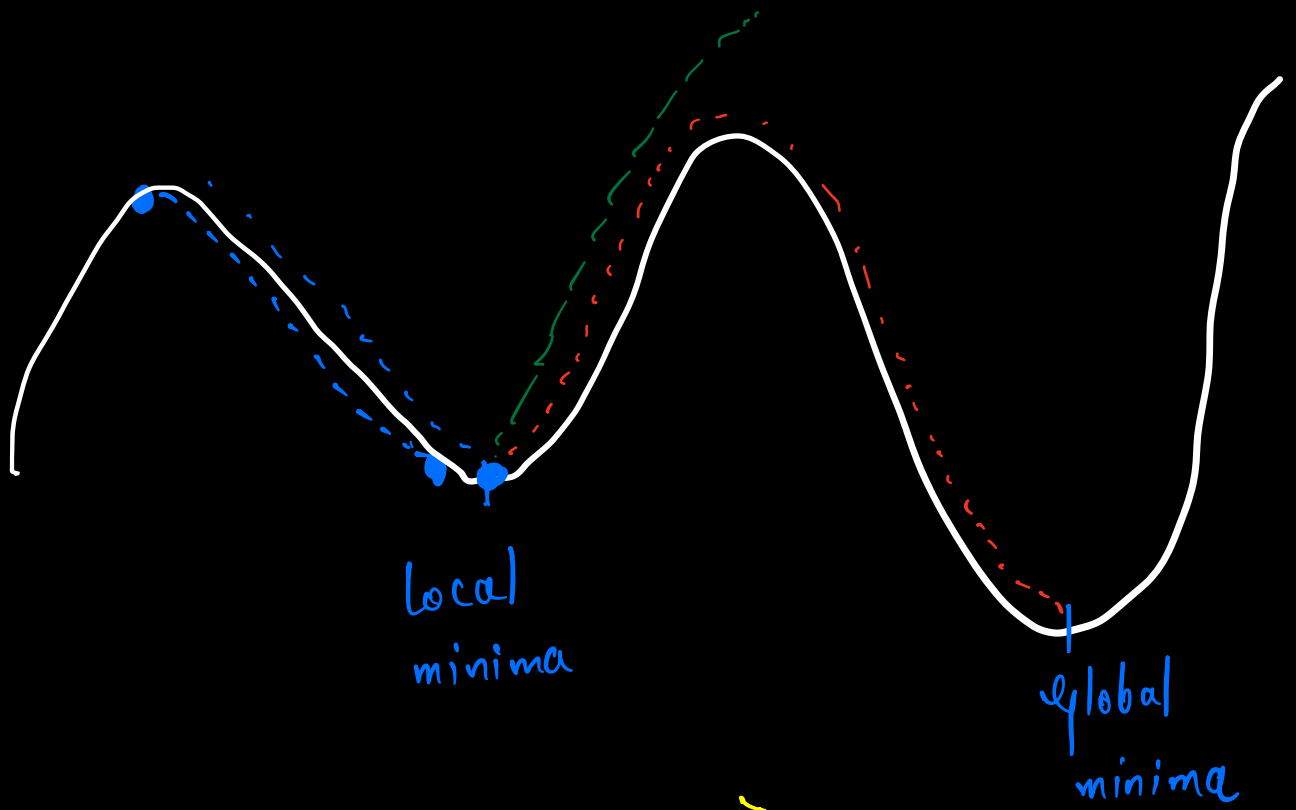
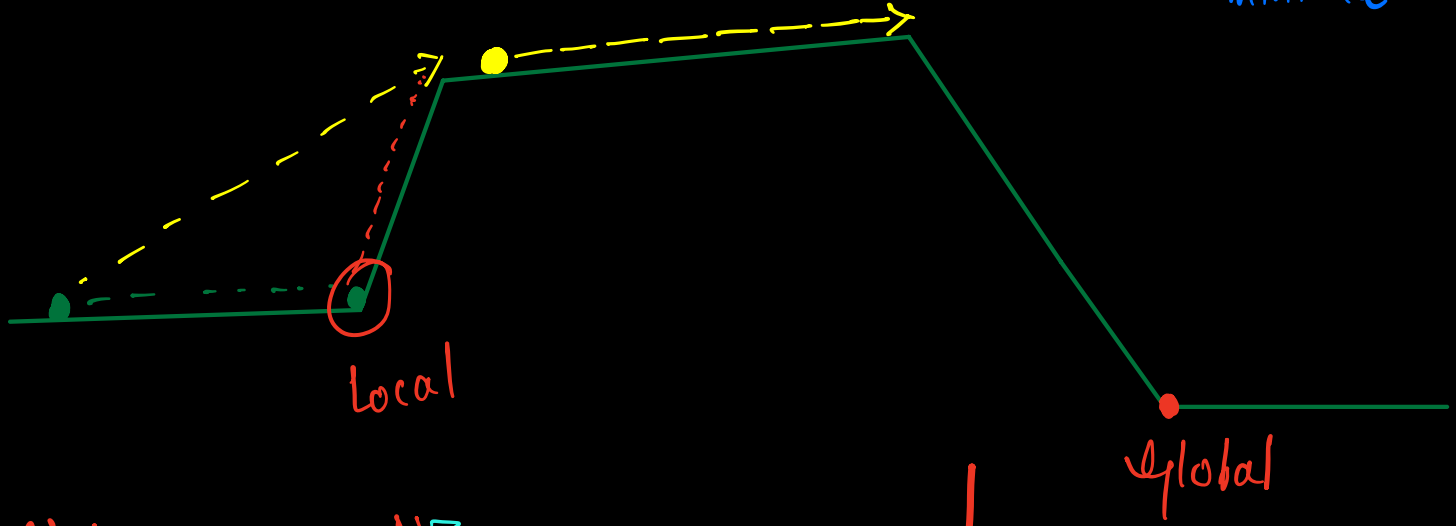
Today's Agenda

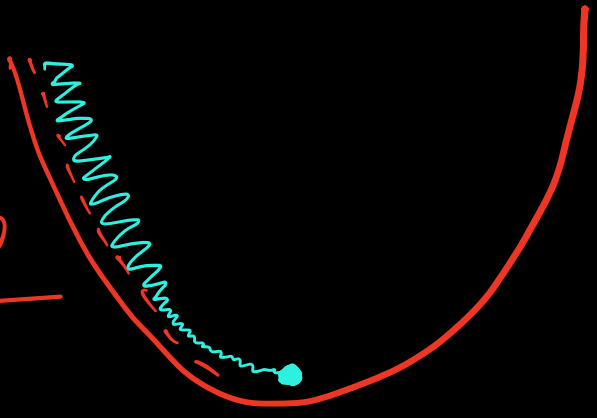
Optimizers





Steps:-

updating w & b



Types

1) Batch Gradient Descent

i) Pass the entire dataset once.

Cons :- Slow, high computational power

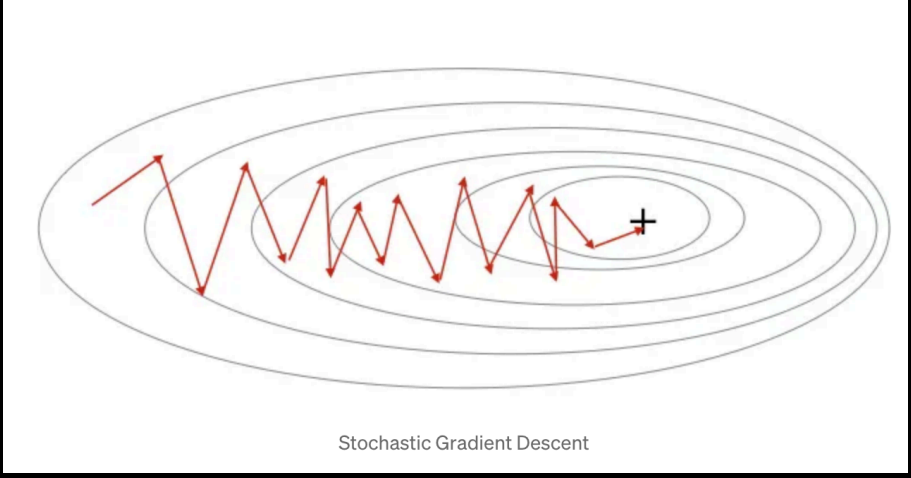
Weight Update :- $(\alpha) \rightarrow$ L·R

$$w_N = w_0 - \underbrace{\eta}_{\text{L·R}} \cdot \frac{1}{N} \sum_{i=0}^{N} \frac{\nabla_w L_i(w)}{\left\{\frac{\partial \text{loss}}{\partial w_{\text{old}}}\right\}}$$

Pro :- Very accurate updates

(ii) Stochastic Gradient Descent

(i) Pass one data point at one time.



Stochastic Gradient Descent

Con:- Noisy Updates → {Time}

Weight Update

$$W_N = W_0 - \eta \nabla_w L(w) \rightarrow \frac{\partial \text{loss}}{\partial w_0}$$

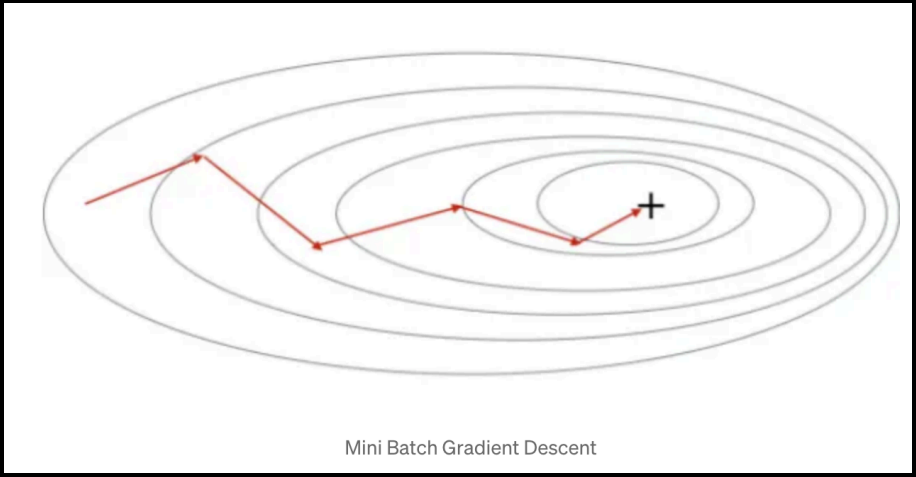
L-R → WRT Time, it should decrease

{Dynamic LR} → Epochs

- 0001
- 00001
- 000001
- 0000001

- 1
- 01
- 001
- 0001

Mini Batch Gradient Descent



Mini Batch Gradient Descent

OOM

Batch → group of datapoints

2^n → 8, 16, 32, 64, 128, 256 VRAM / GPU

Pro:- Smooth convergence, lesser noise

more stable than SGD.

Con:- Needs Tuning of batch size

Weight Update Rule:-

$$w_{new} = w_{old} - \eta \cdot \frac{1}{m} \sum_{n=1}^{m} \nabla_w L(w)$$

m = mini batch size, $m \in N$

1000 data points

batch size = 100

Total batches = 10

Data Representation

1) Batch

2) Iteration
one update of the model using batch

SGD :- 1000 datapoints
1000 iterations

Mini :- Batch size = 100
Total iteration 10

Epochs

Full cycle of the entire dataset.

Model → all training samples

1 Epoch → entire dataset

10 Epochs → entire dataset (10 X)

1000 → DP

100 → BS

Total iterations for 1 epoch = 10

10 epochs = 10 X 10

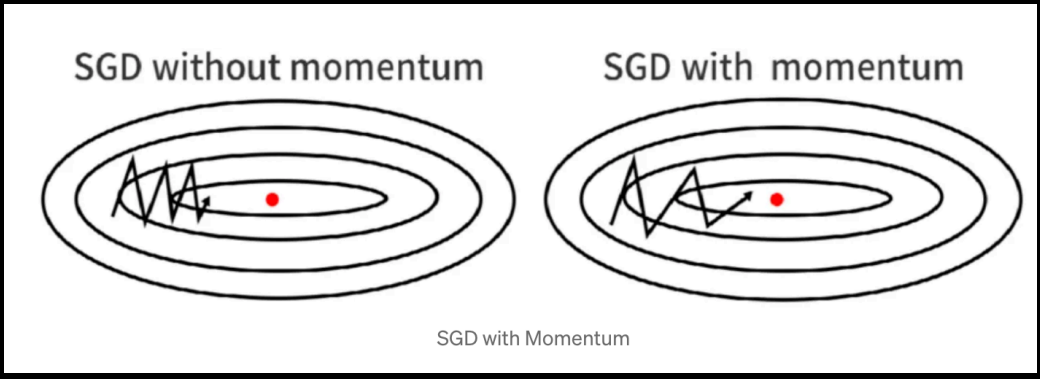
= 100

4) Step → DL Frameworks (Keras, Pytorch)

Step = iteration

↳ weight update

SGD with Momentum



SGD = $w_w = w_0 - \eta \nabla_w L(w)$

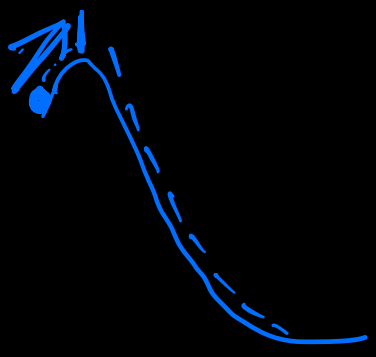
Velocity = velocity * momentum $- \eta \cdot \nabla_w L(w)$

momentum → 0.9

weights = weights + velocity

Step 1 → 90% of step 1

Step 2 → convergence smoother



SGD with Momentum :-

$$v = \beta \times v - \eta * \nabla_w L(w)$$

η → L.R

$\nabla_w L(w)$ → Gradient

(i) Updating the weights

$$W = w + v$$

v = velocity

β = Momentum

Initialing the, $v = 0$

Momentum Co-Efficient

0.9 or 0.99

Ada Grad

Adaptive Optimization Algorithm

(i) Adjust of your L.R

Calculation

Variables :-

η = L.R

g_t = gradient at time step t

G_t = Sum of squares of all past gradients

(1) $G_t = G_{t-1} + g_t^2$ → Accumulated Squared gradients

(2) Update Rule :-

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \cdot g_t$$

where

$\epsilon = 1e^{-8}$

Examples:-

Step 1, $g_1 = 0.5$

Step 2 $g_2 = 0.5$

$G_1 = (0.5)^2 = 0.25$

$G_2 = 0.25 + (0.5)^2$

$= 0.25 + 0.25$

$= 0.5$

Adaptive LR :- $\frac{\eta}{\sqrt{G_t}} = \frac{\eta}{\sqrt{0.5}} < \frac{\eta}{\sqrt{0.25}}$

Pros

1) No LR Tuning

Cons

L.R will keep shrinking forever.

RMSPROP

Root Mean Square Propagation

Definition

g_t : gradient at time step

g_t
g_{t+1}
g_{t+2}
⋮
g_{t+100}

$E[g^2]_t$:- Exponentially decaying average of past squared gradients

β :- decay rate (usually 0.9)

ϵ :- very small term

1. Update the Moving Average of Squared gradients.

$$E[g^2]_t = \beta \cdot E[g^2]_{t-1} + (1-\beta) \cdot g_t^2$$

Moving Average :- Smooth out fluctations

1) Simple Moving Average

2) Exponential Moving Average

$$EMA_t = \alpha \cdot x + (1-\alpha)\, EMA_{t-1}$$

$\alpha \; (0-1)$

↳ gives more weightage to recent values.

$$0.9 = 0.9 \cdot x + (1-0.9)\, EMA_{t-1}$$

$$= 0.9\, x + 0.1\, EMA_{t-1}$$

Weight Update :-

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} \cdot g_t$$

Adagrad vs RMSProp

Gradient History	All past Squared	Recent Squared gradients (decay)
L.R	Keeps shrinking	Stays Adaptive

Cons :-

1) More Complex
2) Tuning of β parameter (decay rate)

Adam

Adaptive Moment Estimation

Combination :- (i) SGD with momentum
(ii) Adaptive LR (RMS Prop)

Maths of Adam

g_t :- gradient at time step t

m_t :- first moment (moment = mean of gradients)

v_t :- Second moment (mean of squared gradients)

β_1 :- decay rate of m_t (0.9)

β_2 :- decay rate of v_t (0.999)

η = L.R

ϵ = small term.

Step four Weight Updates

1) Compute the moment estimates

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$

EWA = EMA

Exponential Weighted Average

Exponential Moving Average

2) Bias Correction (when t is small)

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

This will help in mitigating the bias caused by $m_0 = 0$, $v_0 = 0$

3) Weight Update Rule

$$w_{t+1} = w_t - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

(i) Introducing the momentum

(ii) Scale down the LR if the gradient have high variability