Gradient Descent w/ Momentum

Almost always works faster than gradient descent.

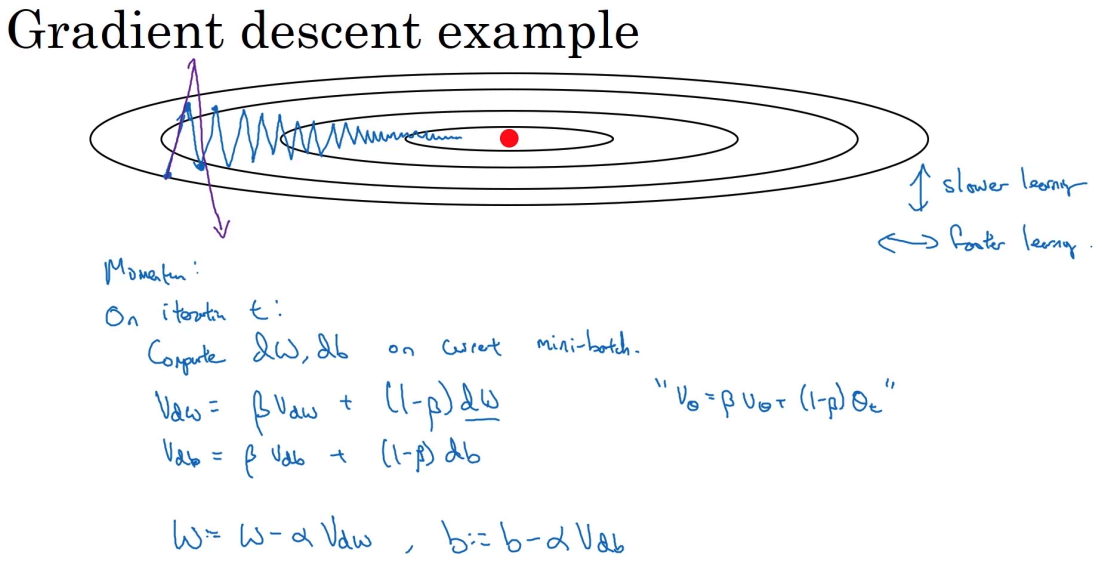
Basic idea: compute an EWA of gradients and use that gradient to update weights instead.'

I.e smooth out gradient descent

To prevent oscillations, we need the vertical learning to be slower and we need the horizontal learning to be faster.

To do this, for every iteration

- Just compute the gradients dW and db on current minibatch
- Simply get the EWA V_dW = beta V_dW + (1-beta) dW
- And same for the EWA V_db
- Then do the updates W = W - lr * V_dW and b = b - alpha * V_db



Eg: averaging over these gradients will cause vertical component to be almost zero and horizontal to be much more.

Now two hyperparameters: alpha (lr) and beta (most common in 0.9, average over last 10)

Implementation details

On iteration t:

Compute dW, db on the current mini-batch

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

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

$W = W - \alpha v_{dW}, \quad b = b - \alpha v_{db}$

Hyperparameters: α, β $\qquad\qquad$ $\beta = 0.9$