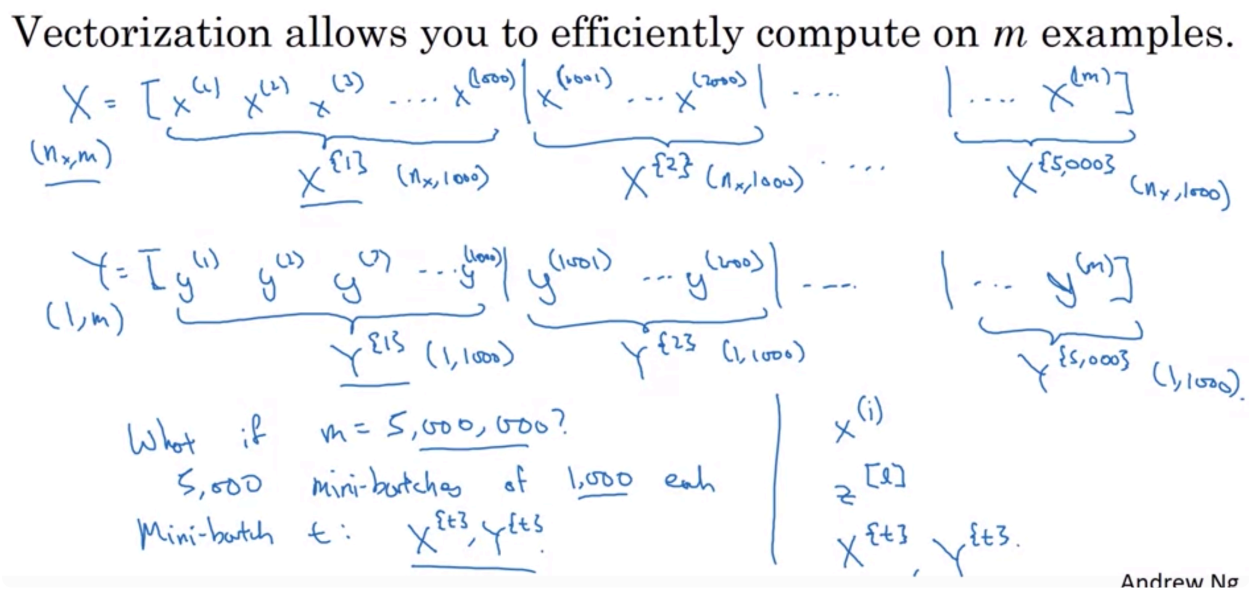
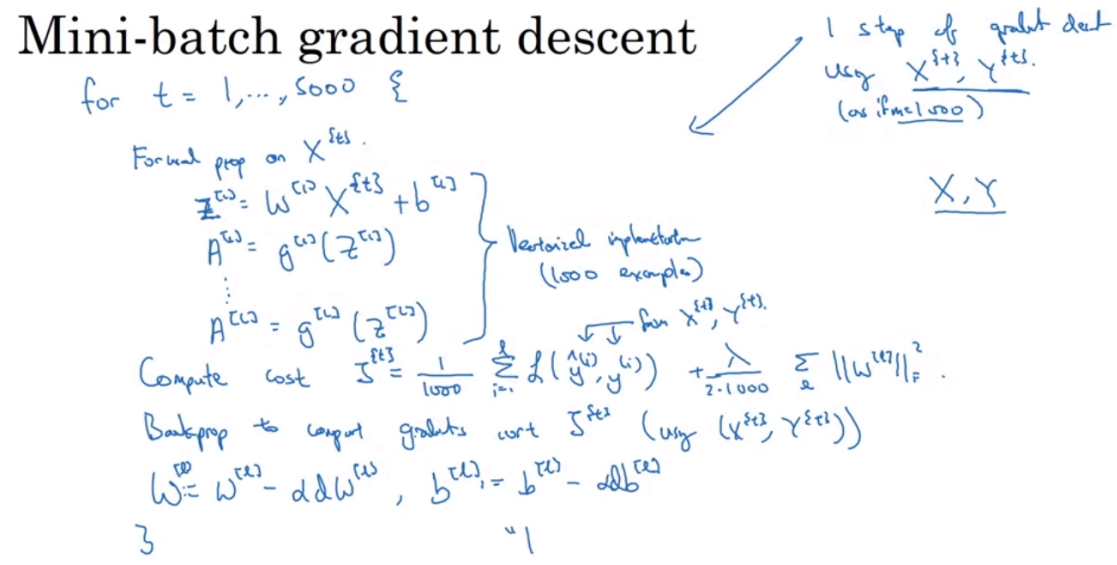
Batch



Mini-Batch

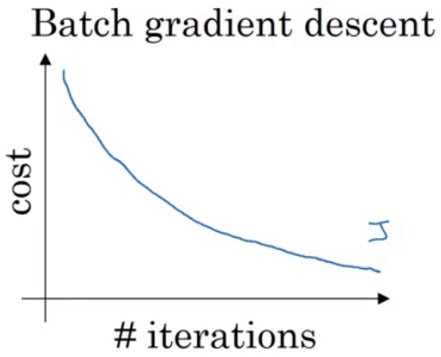
1 epoch - single pass through training set

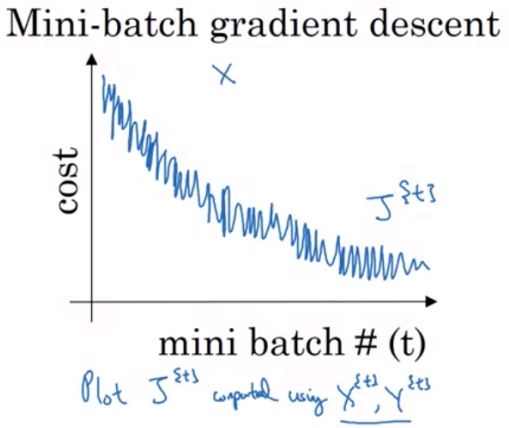


Large dataset - mini batch way faster

Understanding Mini-batch

- May not decrease on every mini-batch, unlike batch
- Why? Some batch may be harder (higher costs)

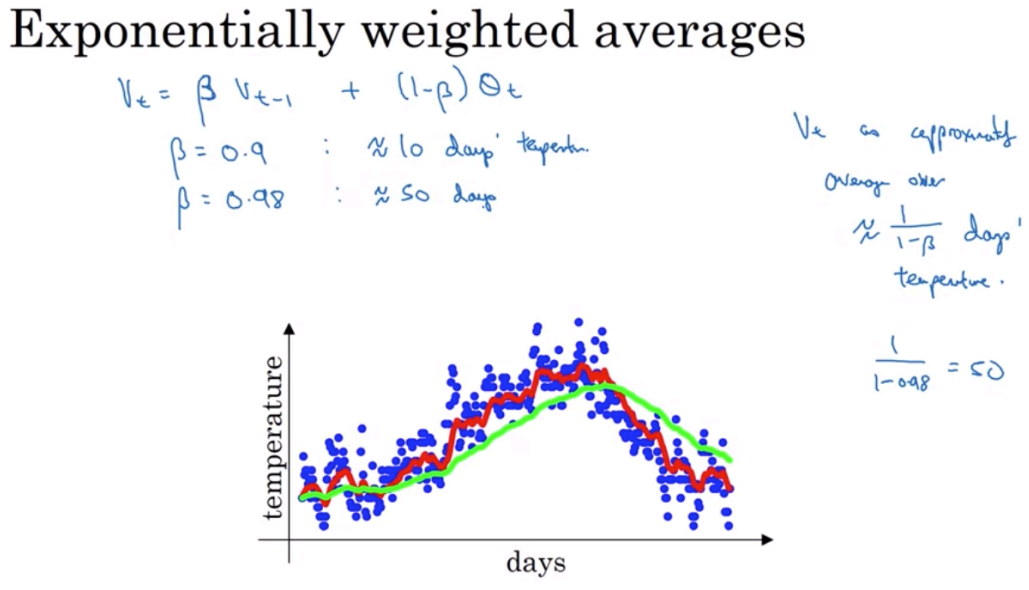




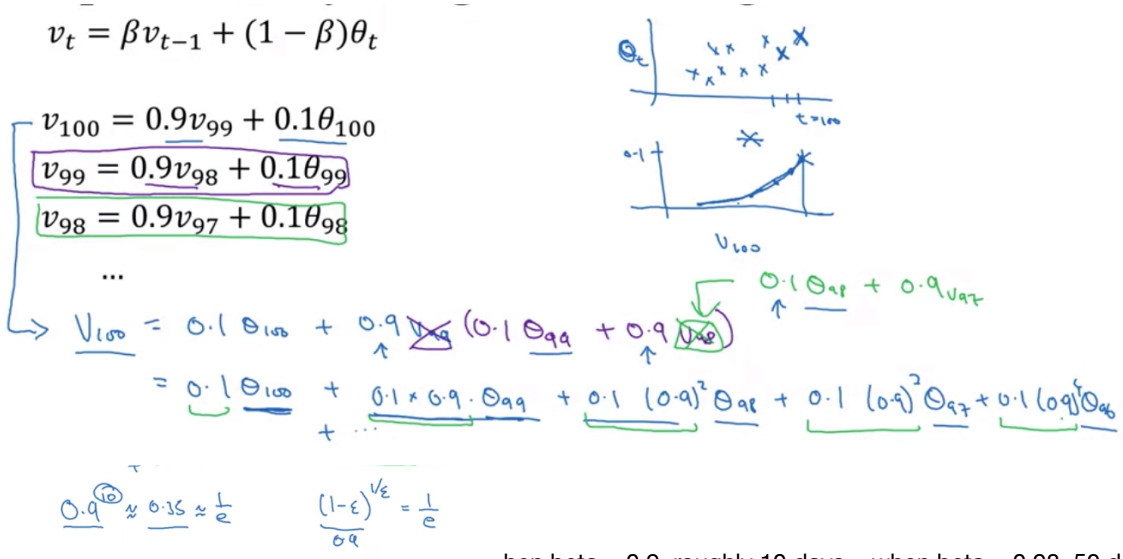
Stochastic: loss speedup through vectorization
Batch: too long per iteration

- Small-training set (<2000) use batch
- Typical mini-batch: 64,128,256,512
- Make sure mini batch fit in CPU/GPU men
- Generally Try a few and pick the most efficient.

Exponentially weighted average



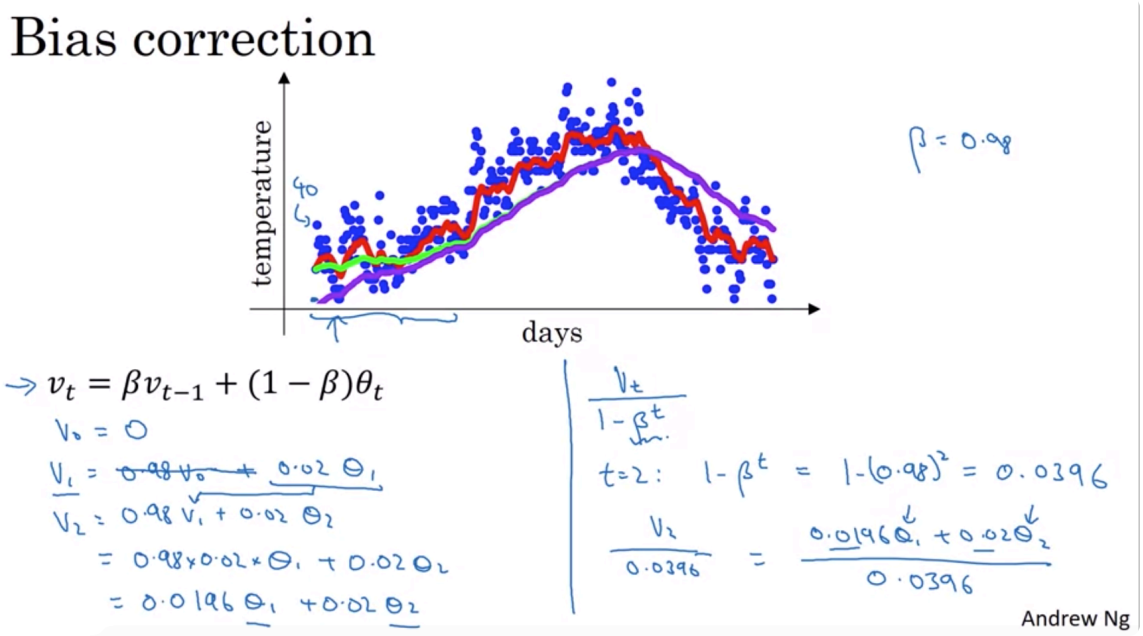
Understanding Exponentially weighted average



When beta = 0.9, roughly 10 days, when beta = 0.98, 50 days

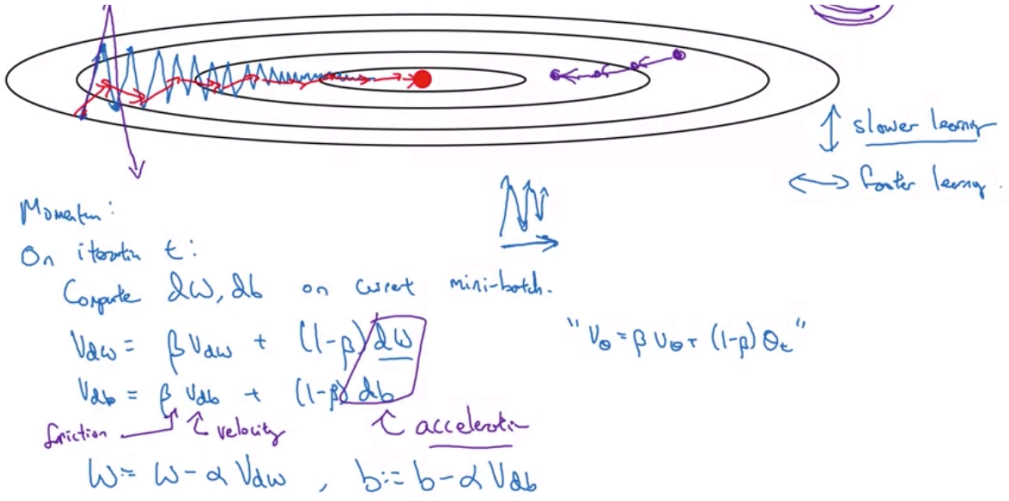
Bias Correction

The purple line deviate from ideal (green) initially, just set v_t/(1-beta_t)



Momentum

v_db is velocity



Implementation details

$v_{dW} = 0$, $v_{db} = 0$

On iteration t:

Compute dW, db on the current mini-batch

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

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

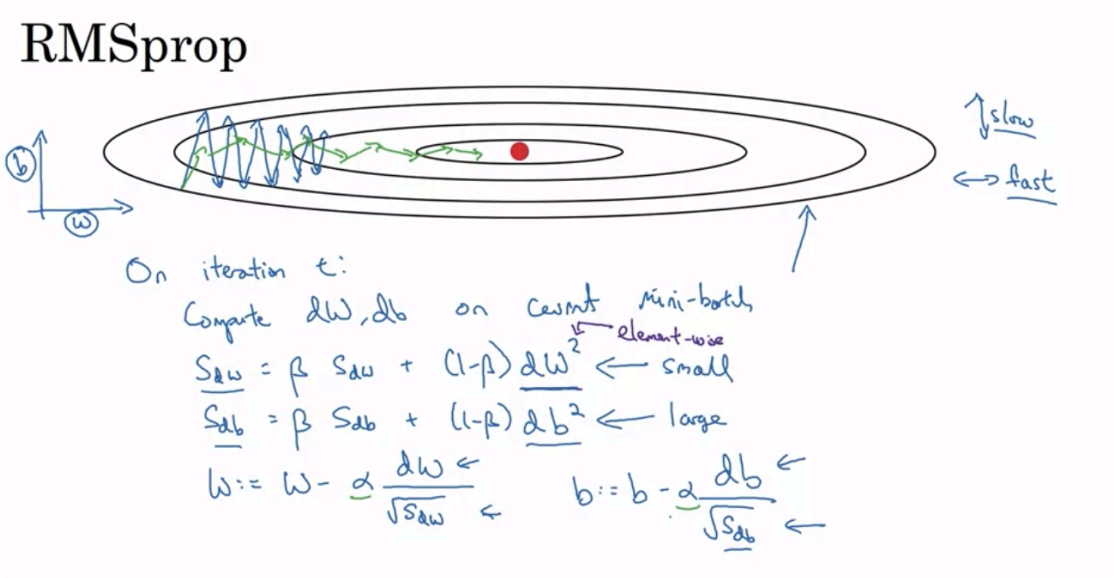
$W = W - \alpha v_{dW}$, $b = b - \alpha v_{db}$

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

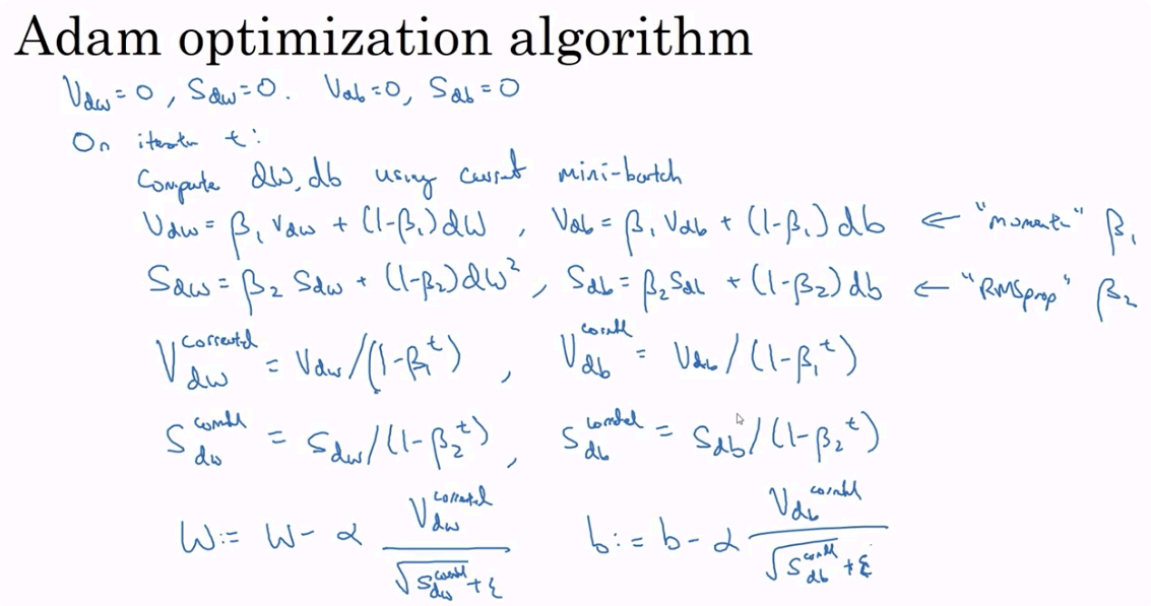
average over last ~10 gradients

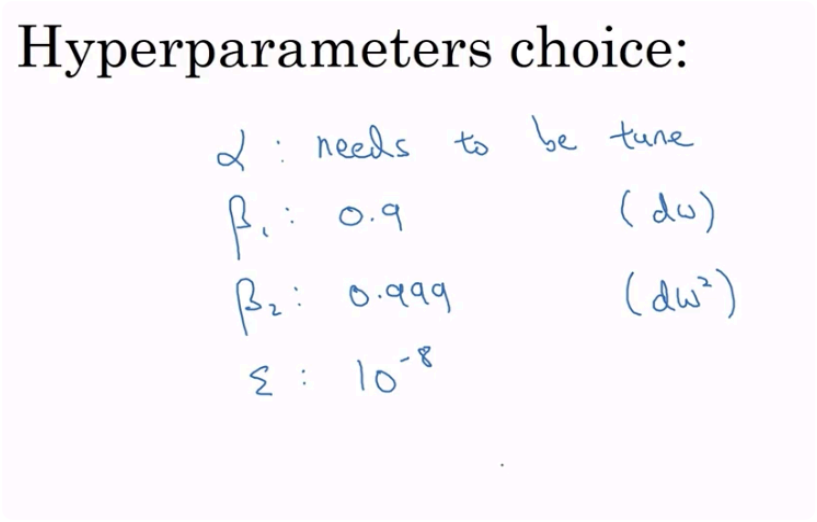
RMSProp

- Can use larger learning rate without diverging
- Root mean square

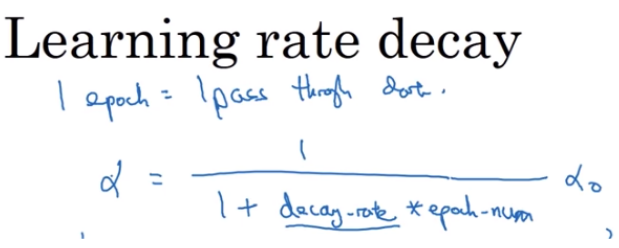


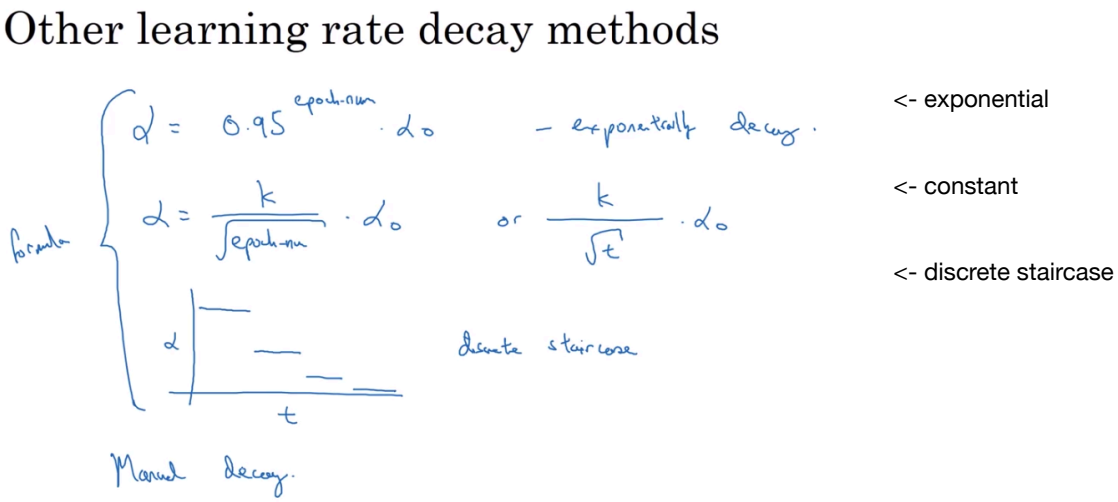
Adam





Learning Rate Decay. (Not usually high on the list for optimizing)





<- exponential

<- constant

<- discrete staircase

Local Optima

- Very less likely to stuck in local optima
- More saddle points.
- **Plateaus** makes learning slow