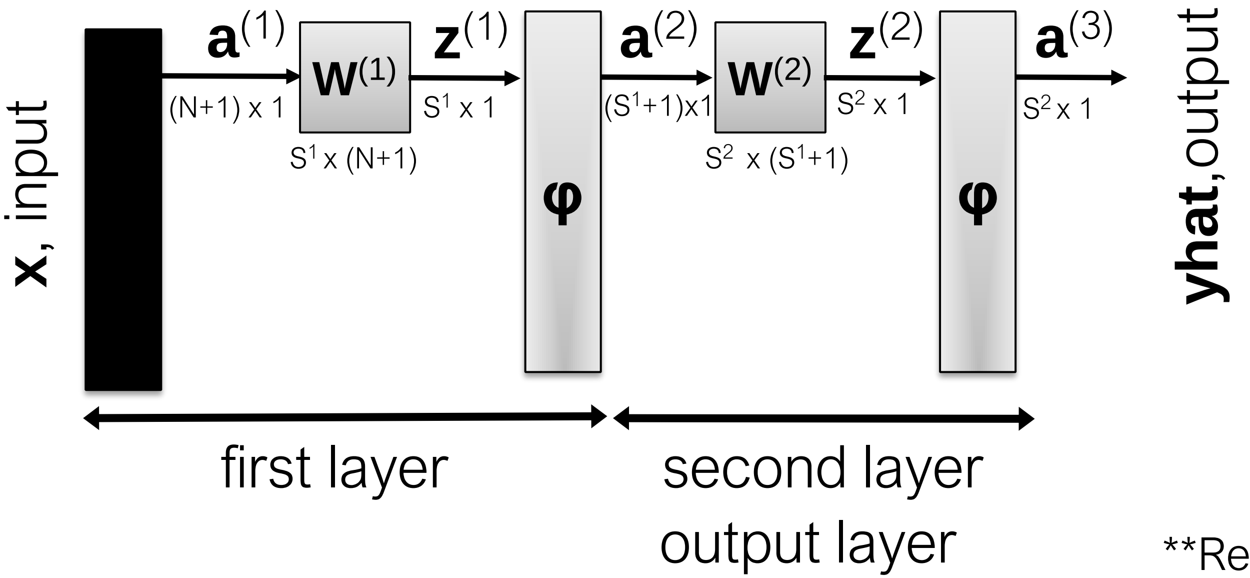
Review: Back propagation

- Steps:
  - propagate weights forward
  - calculate gradient at final layer
  - back propagate gradient for each layer
    - via recurrence relation

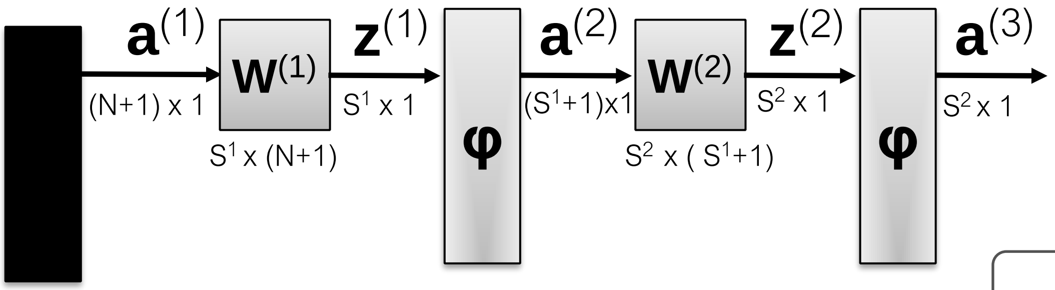


$$J(\mathbf{W}) = \|\mathbf{Y} - \overset{\triangle}{\mathbf{Y}}\|^2$$

$$w_{i,j}^{(l)} \leftarrow w_{i,j}^{(l)} - \eta \frac{\partial J(\mathbf{W})}{\partial w_{i,j}^{(l)}}$$

**Recall from Flipped Assignment!

Review: Back Propagation Summary



1. Forward propagate to get **Z**, **A**
2. Get final layer gradient
3. Back propagate sensitivities
4. Update each $\mathbf{W}^{(l)}$

$$\mathbf{V}^{(2)} = -2(\mathbf{Y} - \mathbf{A}^{(3)}) * \mathbf{A}^{(3)} * (1 - \mathbf{A}^{(3)})$$

$$\nabla^{(2)} = \mathbf{V}^{(2)} \cdot [\mathbf{A}^{(2)}]^T$$

$$\mathbf{V}^{(1)} = \mathbf{A}^{(2)} * (1 - \mathbf{A}^{(2)}) * [\mathbf{W}^{(2)}]^T \cdot \mathbf{V}^{(2)}$$

$$\nabla^{(1)} = \mathbf{V}^{(1)} \cdot [\mathbf{A}^{(1)}]^T$$

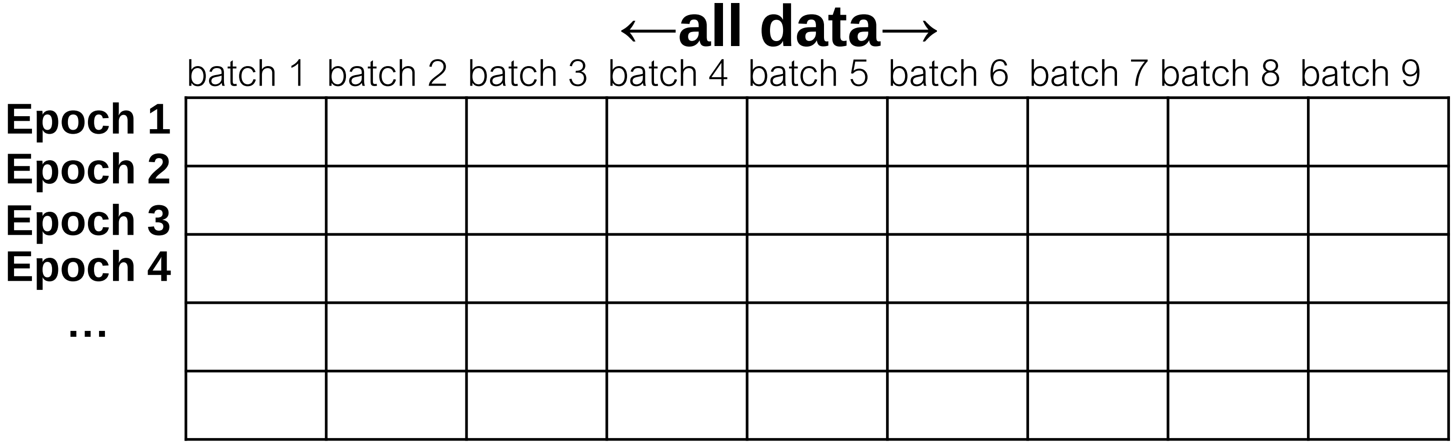
$$\mathbf{W}^{(l)} \leftarrow \mathbf{W}^{(l)} - \eta \nabla^{(l)}$$

Where is the problem of **vanishing gradients** introduced?

**Recall from Flipped Assignment!

Mini-batching

- Numerous instances to find one gradient update
  - **solution**: mini-batch



←**all data**→

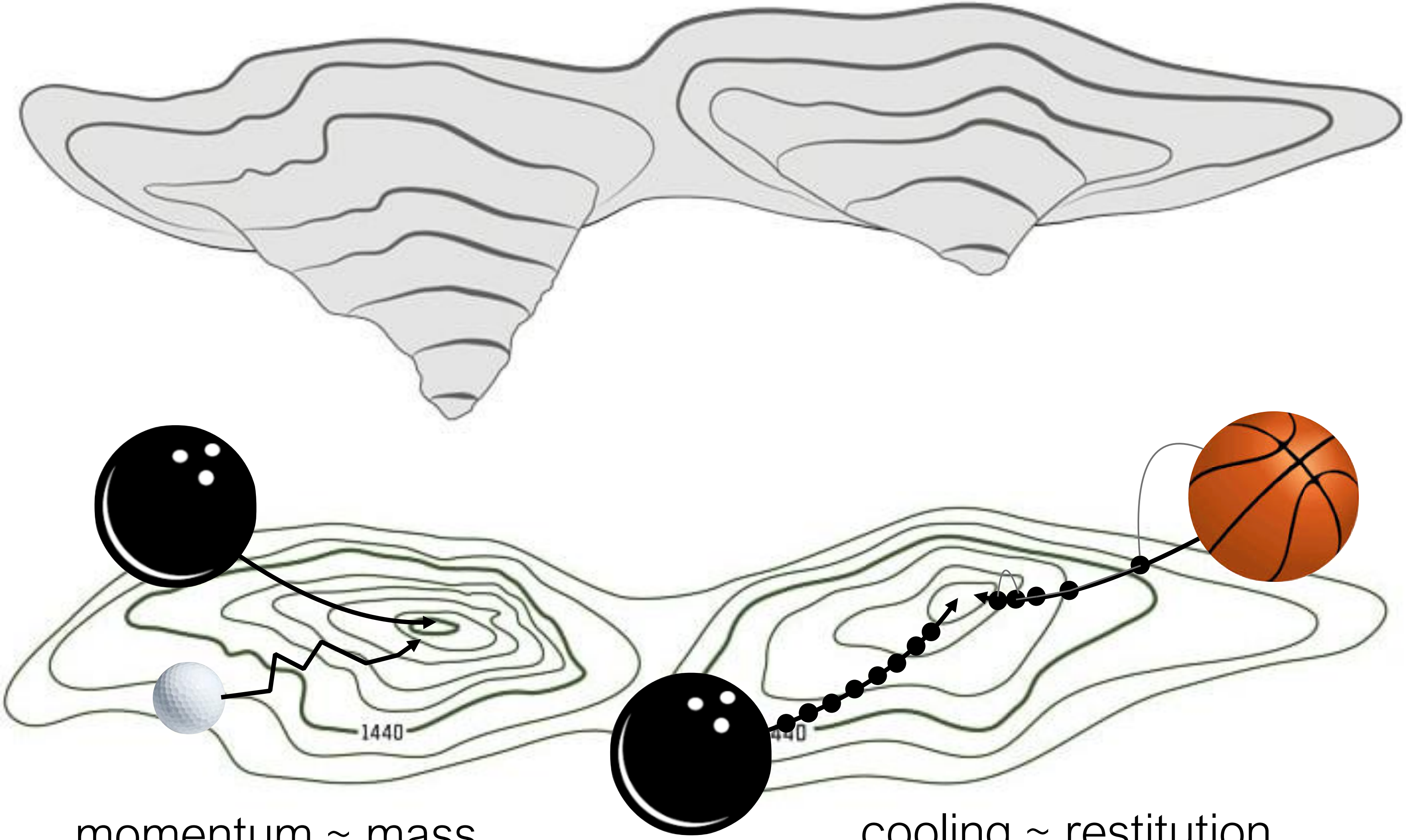
batch 1 batch 2 batch 3 batch 4 batch 5 batch 6 batch 7 batch 8 batch 9

Epoch 1
Epoch 2
Epoch 3
Epoch 4
…

shuffle ordering each epoch and update W's after each batch

- **new problem**: mini-batch gradient updates erratic
  - **solutions**:
    - momentum
    - adaptive learning steps (cooling)

Momentum and Cooling Intuition



momentum ~ mass

cooling ~ restitution

Momentum

$$\mathbf{W}_{k+1} = \mathbf{W}_k - \rho_k$$

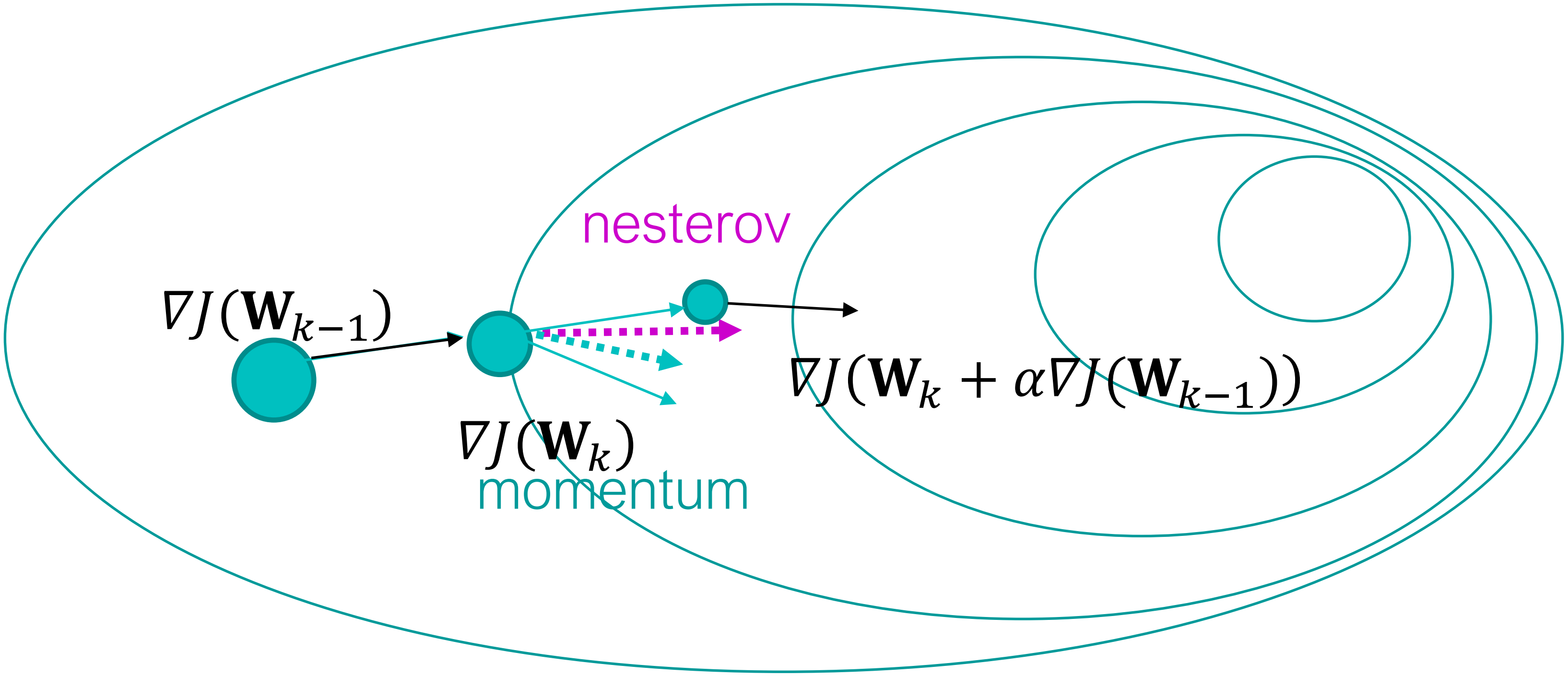
- Momentum

$$\rho_k = \alpha \nabla J(\mathbf{W}_k) + \beta \nabla J(\mathbf{W}_{k-1})$$

- Nesterov’s Accelerated Gradient

$$\rho_k = \beta \nabla J(\mathbf{W}_k + \alpha \nabla J(\mathbf{W}_{k-1})) + \alpha \nabla J(\mathbf{W}_{k-1})$$

$$step\ twice$$

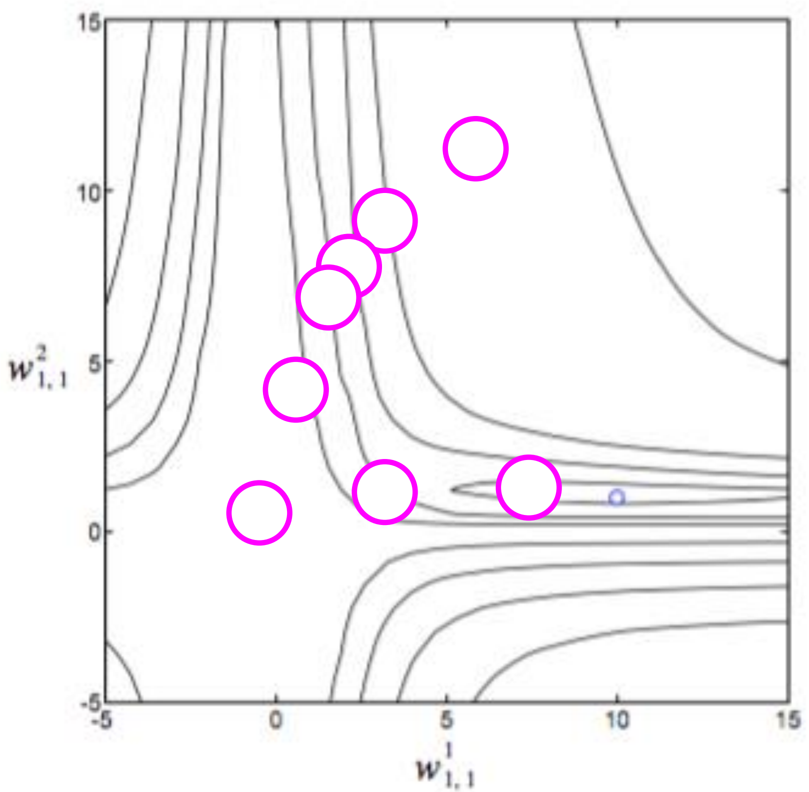

$\nabla J(\mathbf{W}_{k-1})$
nesterov
$\nabla J(\mathbf{W}_k + \alpha \nabla J(\mathbf{W}_{k-1}))$
$\nabla J(\mathbf{W}_k)$
momentum

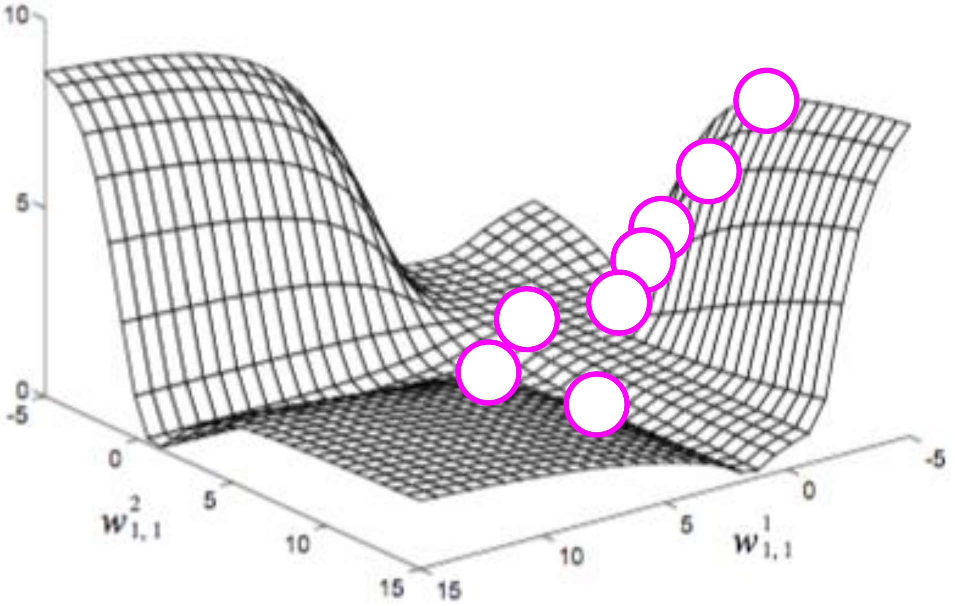
Adaptive Strategy: Cooling

- Fixed Reduction at Each Epoch
- Adjust on Plateau
    - make smaller if when J rapidly changes
    - make bigger when J not changing much

$$\eta_k = \eta_0 \cdot d^{\lfloor \frac{k_{max}}{k} \rfloor}$$ drop by d every k_d epochs

$$\eta_k = \eta_0^{(1+k\cdot d)}$$ drop a little every epoch





Demo

07. MLP Neural Networks.ipynb

optimizations:

mini-batch
momentum
adaptive learning