Gradient and Optimization

Digression

DERIVATIVES AND GRADIENT

Derivatives

- Some derivation rules:

$$\left(w^2\right)' = 2w$$

$$\left(w^a\right)' = aw^{a-1}$$

$$\left(e^w\right)' = e^w$$

$$\left(\ln(w)\right)' = \frac{1}{w}$$

$$\left(f(w)^2\right)' = 2f(w) \cdot f'(w)$$

$$\left(f(w)^a\right)' = a\left[f(w)\right]^{a-1} \cdot f'(w)$$

$$\left(e^{f(w)}\right)' = e^{f(w)} \cdot f'(w)$$

$$\left(\ln(f(w))\right)' = \frac{1}{f(w)} \cdot f'(w)$$

$$c' = 0$$

$$(cw)' = c$$

$$\left(f(w) + g(w)\right)' = f'(w) + g'(w)$$

- If we are supplied a value for w, **say 5**, then the above become numbers.
  - We say, we obtain the derivative "**at point 5**"

Partial Derivatives

- Suppose now we have a function of multiple variables, e.g.

$$f(w_1, w_2, w_3) = (w_1 w_2 w_3)^2$$

- It can also written as $f(\mathbf{w})$, where $\mathbf{w}$ is $[w_1, w_2, w_3]$
- This function has three partial derivatives:
  - f'_{w1} obtained by considering only w_1 variable and w_2, w_3 constant
  - f'_{w2} obtained by considering only w_2 variable and w_1, w_3 constant
  - f'_{w3} obtained by considering only w_3 variable and w_1, w_2 constant

$$f'_{w_1}(w_1, w_2, w_3) = 2(w_1 w_2 w_3) w_2 w_3 = 2 w_1 w_2^2 w_3^2$$

$$f'_{w_2}(w_1, w_2, w_3) = 2(w_1 w_2 w_3) w_1 w_3 = 2 w_1^2 w_2 w_3^2$$

$$f'_{w_3}(w_1, w_2, w_3) = 2(w_1 w_2 w_3) w_1 w_2 = 2 w_1^2 w_2^2 w_3$$

Derivation rules are the same as those for a single variable.

Other Notation

f'_{w_1} also denoted by $\dfrac{\partial f}{\partial w_1}$

f'_{w_2} also denoted by $\dfrac{\partial f}{\partial w_2}$

f'_{w_3} also denoted by $\dfrac{\partial f}{\partial w_3}$

Gradient

- The gradient is the vector of partial derivatives.

$$\nabla_f(\mathbf{w}) = \begin{bmatrix} 2w_1 w_2^2 w_3^2, & 2w_1^2 w_2 w_3^2, & 2w_1^2 w_2^2 w_3 \end{bmatrix}$$

- Now suppose we want to compute the gradient **at a point**, say **w**=(1,2,3), i.e. w_1=1, w_2=2, w_3=3

$$\nabla_f(\mathbf{w}) = \nabla_f([1,2,3]) = \begin{bmatrix} 2\cdot 1\cdot 2^2\cdot 3^2, & 2\cdot 1^2\cdot 2\cdot 3^2, & 2\cdot 1^2\cdot 2^2\cdot 3 \end{bmatrix} = \begin{bmatrix} 81, & 36, & 24 \end{bmatrix}$$

OPTIMIZATION

Minimization Problem

$$\min_{\mathbf{w}} f(\mathbf{w})$$

Iterative Method

- Start at some $\mathbf{w}_0$; take a step down the **steepest slope**
- Fixed step size:

$$\mathbf{w} \leftarrow \mathbf{w} - \kappa\mathbf{v}$$

- $\mathbf{v}$ is a **vector** in the direction of the **steepest slope**.
  - Steepest slope at some point? – The gradient vector at that point.
  - E.g. $f(w_1, w_2, w_3) = (w_1 w_2 w_3)^2$

$$\nabla_f(\mathbf{w}) = \left[2w_1 w_2^2 w_3^2, \quad 2w_1^2 w_2 w_3^2, \quad 2w_1^2 w_2^2 w_3\right]$$

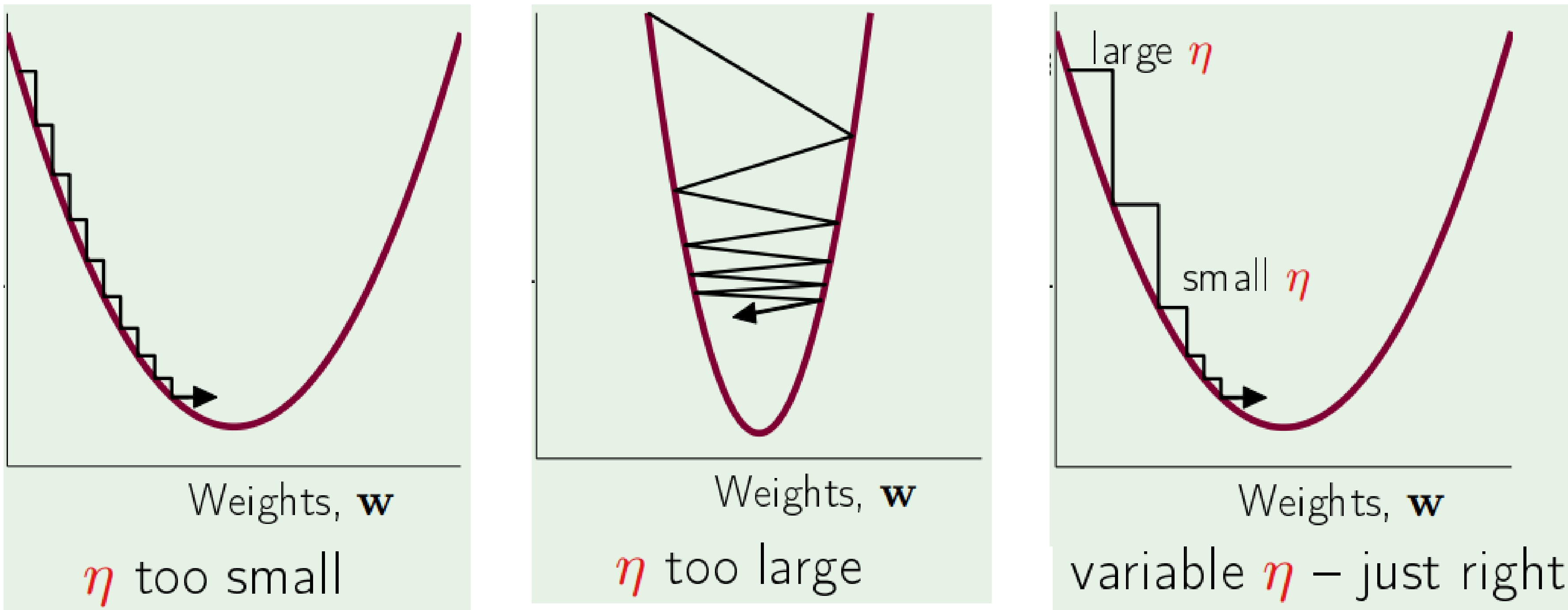
At point **w**=(1,2,3), i.e. w_1=1, w_2=2, w_3=3, we have

$$\nabla_f(\mathbf{w}) = \nabla_f([1,2,3]) = \left[2\cdot 1\cdot 2^2\cdot 3^2, \quad 2\cdot 1^2\cdot 2\cdot 3^2, \quad 2\cdot 1^2\cdot 2^2\cdot 3\right]$$
$$= \left[81, \quad 36, \quad 24\right]$$

$\mathbf{w} \leftarrow [1,2,3] - 0.0001 * [81,36,24]$

Step Size (kappa, often denoted eta)

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \mathbf{v}$$



Figures taken from Y. A. Mostafa (California Institute of Technology)

Gradient Descent Algorithm

Initialize $\mathbf{w}=\mathbf{0}$

For t=0,1,2,…do

Compute the gradient $\nabla_f(\mathbf{w})$

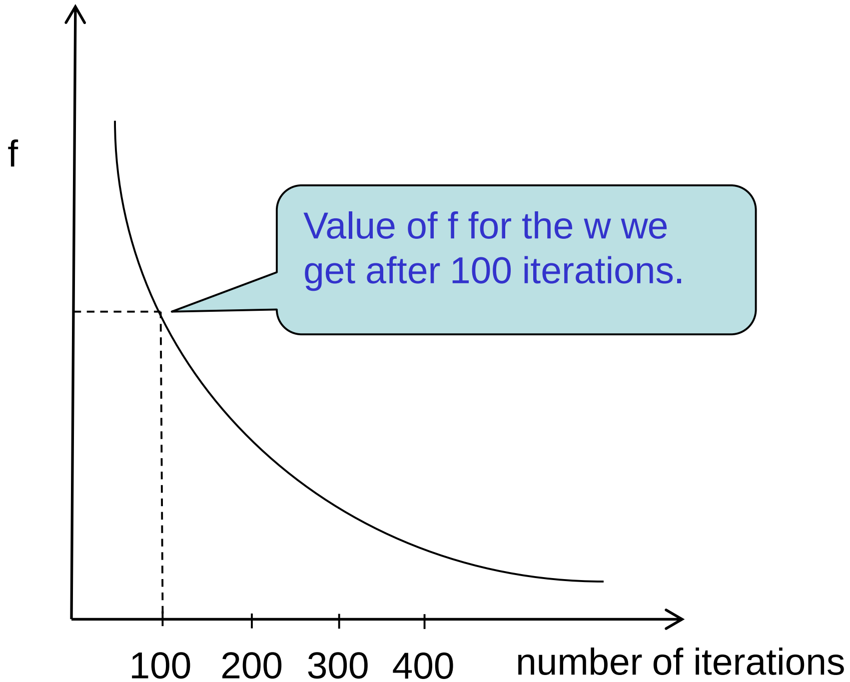
Update the weights $\mathbf{w} \leftarrow \mathbf{w} - \kappa \nabla_f(\mathbf{w})$

Iterate with the next step until **w** doesn't change too much (or for a fixed number of iterations)

Return final **w**.

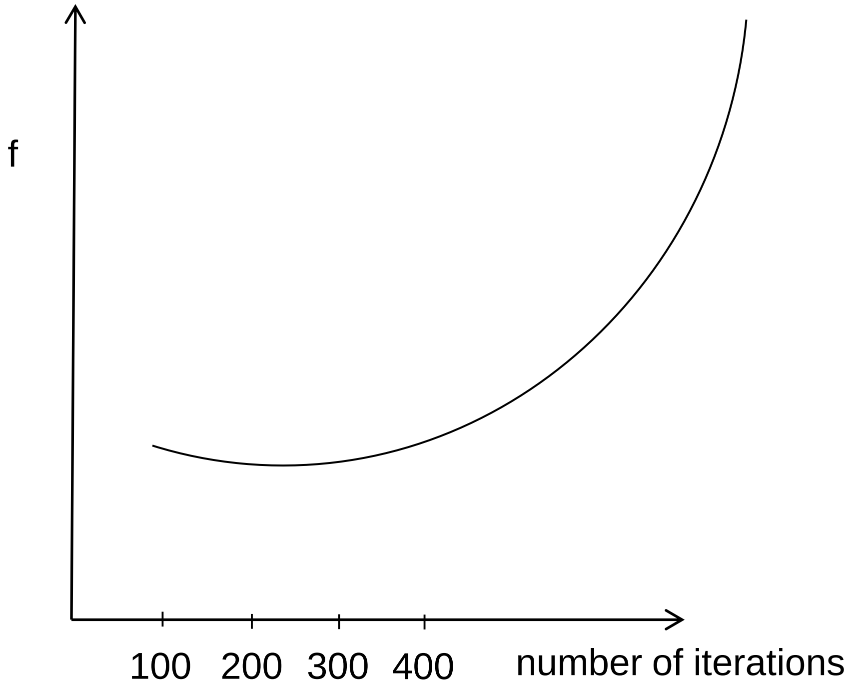
HOW DO WE DETERMINE κ?

f(**w**) during iterations



A picture like this tells us that the gradient descent is working fine.

f(**w**) during iterations



A picture like this tells us that the gradient descent is NOT working fine.

We should use smaller κ

If κ is small enough, a convex f(w) should decrease on every iteration. However, if κ is too small, it well take a long time to converge.

Practically

Try

κ=0.001 κ=0.01 κ=0.1 κ=1

Plot or see f(**w**) for each one. If it decreasing with a reasonable speed, choose that value for κ.