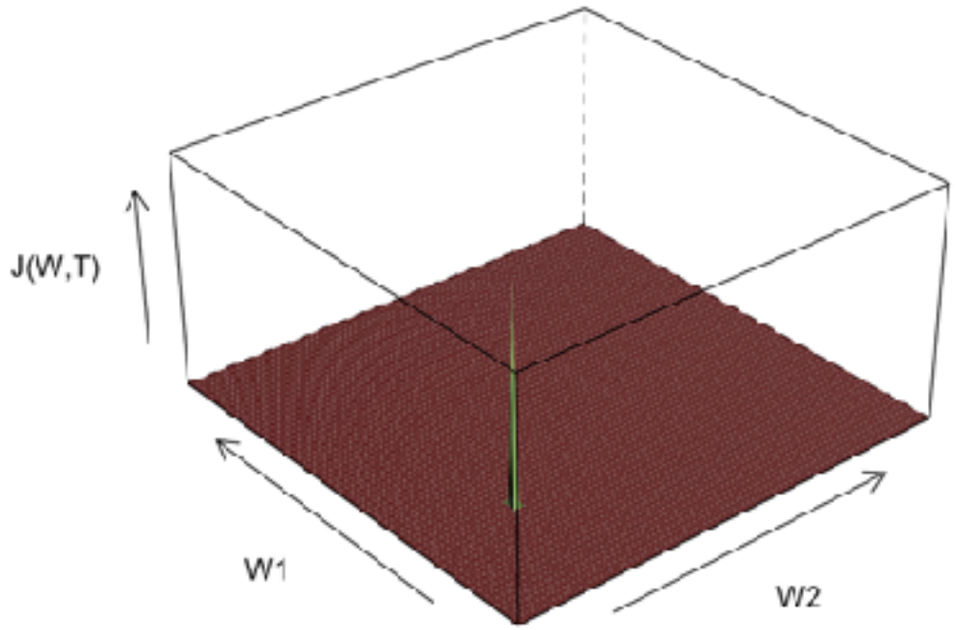
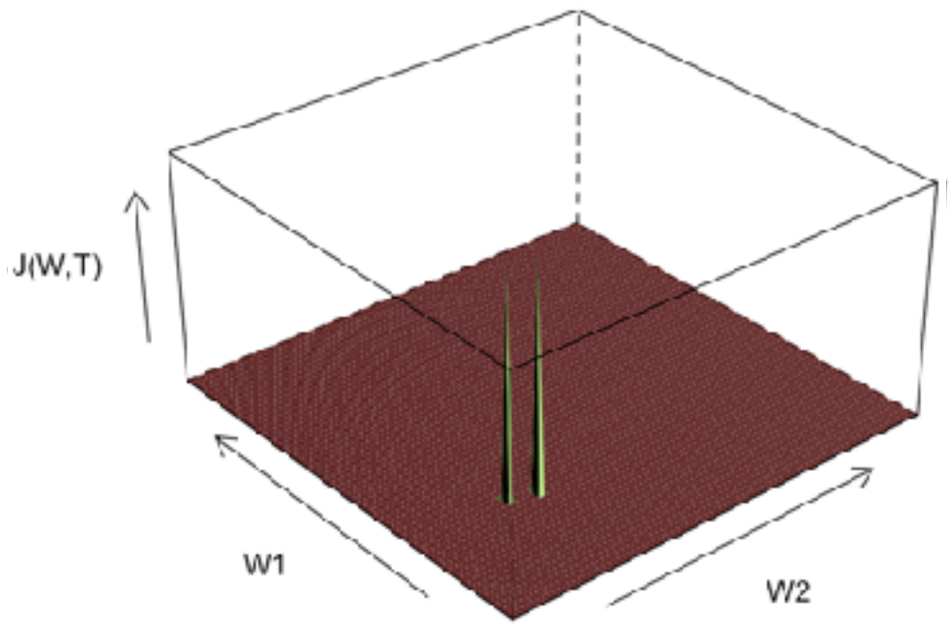
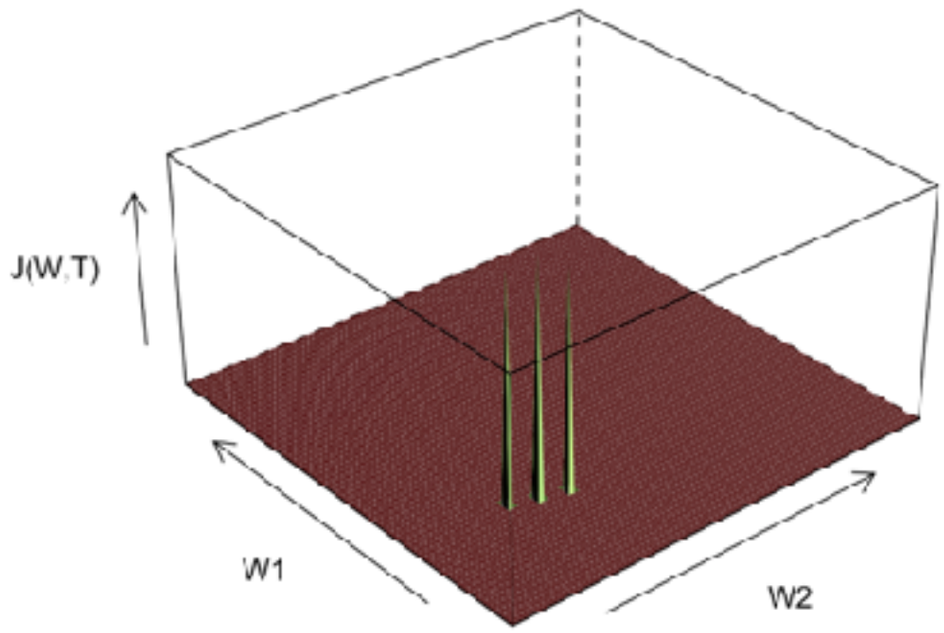
Gradient Descent Updater Strategies

 Romeo Kienzler

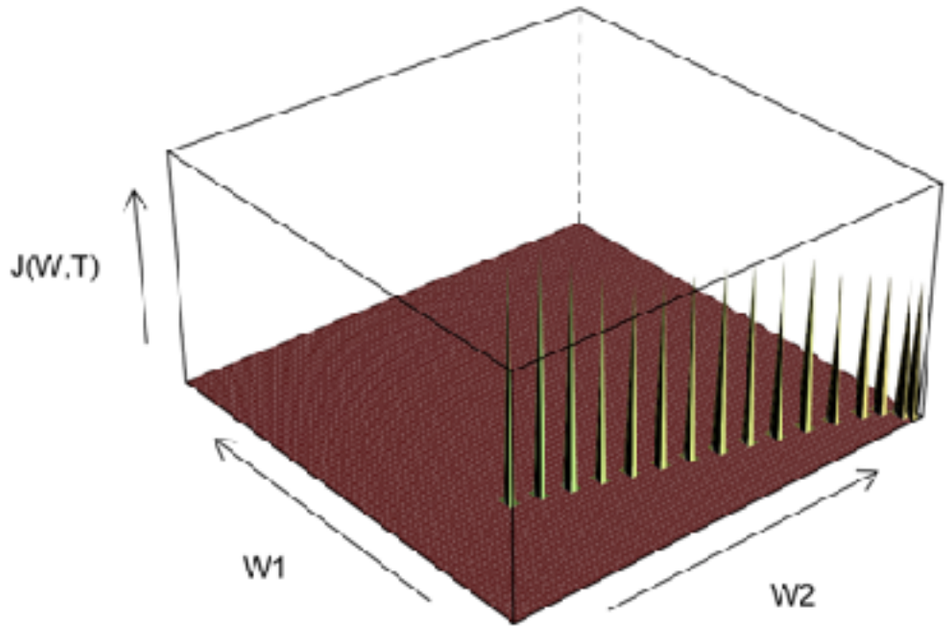




We then could climb down the ladder



..one step by another…



..until we reached an optimal value - this method is called gradient descent - and with all it’s variations it’s the de-facto standard in neural network training.

Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, X, Y)$$

Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, X, Y)$$

Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, X, Y)$$

Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, X, Y)$$

Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, X, Y)$$

Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, X, Y)$$

Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, X, Y)$$

Stochastic Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, x^{(i)}, y^{(i)})$$

Stochastic Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, x^{(i)}, y^{(i)})$$

(Mini) Batch Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, x^{(i..i+n)}, y^{(i..i+n)})$$

(Mini) Batch Gradient Descent

$$\theta_{t+1} = \theta_t - \eta \Delta_\theta J(\theta_t, x^{(i..i+n)}, y^{(i..i+n)})$$

Momentum

$$\nu_t = \gamma \nu_{t-1} - \eta \Delta_\theta J(\theta_t, X, Y)$$
$$\theta_{t+1} = \theta_t - \nu_t$$

Momentum

$$\nu_t = \gamma \nu_{t-1} - \eta \Delta_\theta J(\theta_t, X, Y)$$
$$\theta_{t+1} = \theta_t - \nu_t$$

Momentum

$$\nu_t = \gamma \nu_{t-1} - \eta \Delta_\theta J(\theta_t, X, Y)$$
$$\theta_{t+1} = \theta_t - \nu_t$$

Nesterov accelerated gradient

$$\nu_t = \gamma \nu_{t-1} - \eta \Delta_\theta J(\theta_t - \gamma \nu_{t-1}, X, Y)$$
$$\theta = \theta - \nu_t$$

Nesterov accelerated gradient

$$\nu_t = \gamma \nu_{t-1} - \eta \Delta_\theta J(\theta_t - \gamma \nu_{t-1}, X, Y)$$
$$\theta = \theta - \nu_t$$

Adagrad

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii}+\epsilon}} \Delta_\theta J(\theta_{t,i}, X, Y)$$

Adagrad

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \Delta_\theta J(\theta_{t,i}, X, Y)$$

Adagrad

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii}+\epsilon}} \Delta_\theta J(\theta_{t,i}, X, Y)$$

Adagrad

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii}+\epsilon}} \Delta_\theta J(\theta_{t,i}, X, Y)$$

Adagrad

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \Delta_\theta J(\theta_{t,i}, X, Y)$$

Adagrad

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \Delta_\theta J(\theta_{t,i}, X, Y)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \Delta_\theta J(\theta_t, X, Y)$$

Adadelta

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[\Delta\theta^2]_t + \epsilon}} \Delta_\theta J(\theta_t, X, Y)$$

Adadelta

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[\Delta\theta^2]_t + \epsilon}} \Delta_\theta J(\theta_t, X, Y)$$

RMSProp

 Romeo Kienzler

RMSProp

Adam

AdaMax

RMSProp

Adam

AdaMax

RMSProp

Adam

Nadam

Summary

Activation Functions

 Romeo Kienzler