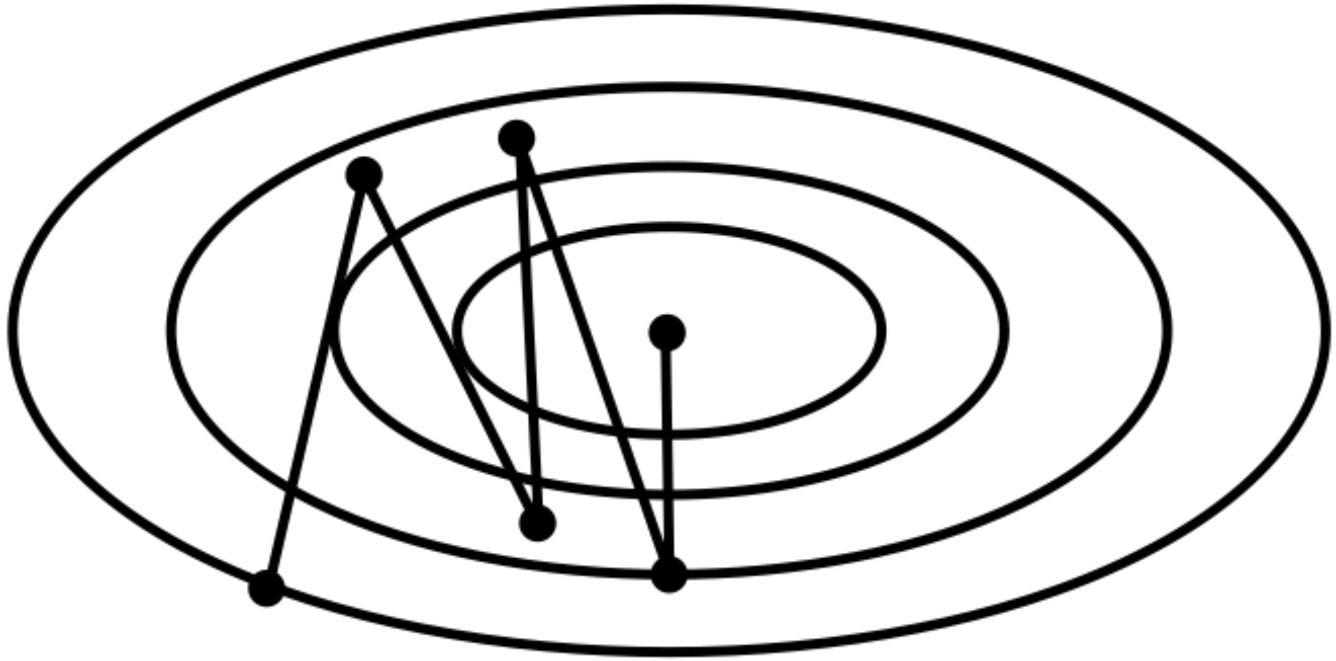
Gradient descent extensions

Difficult function



We don’t want our gradient descent to oscillate like this, esp because it may fail to converge. This is bad!

Mini-batch gradient descent

```
$w^0$ — initialization
while True:
    $i_1, \dots, i_m$ = random indices between 1 and $\ell$
    $g_t = \frac{1}{m} \sum_{j=1}^{m} \nabla L\left(w^{t-1}; x_{i_j}; y_{i_j}\right)$
    $\boldsymbol{w^t = w^{t-1} - \eta_t g_t}$
    if $\|w^t - w^{t-1}\| < \epsilon$ then break
```

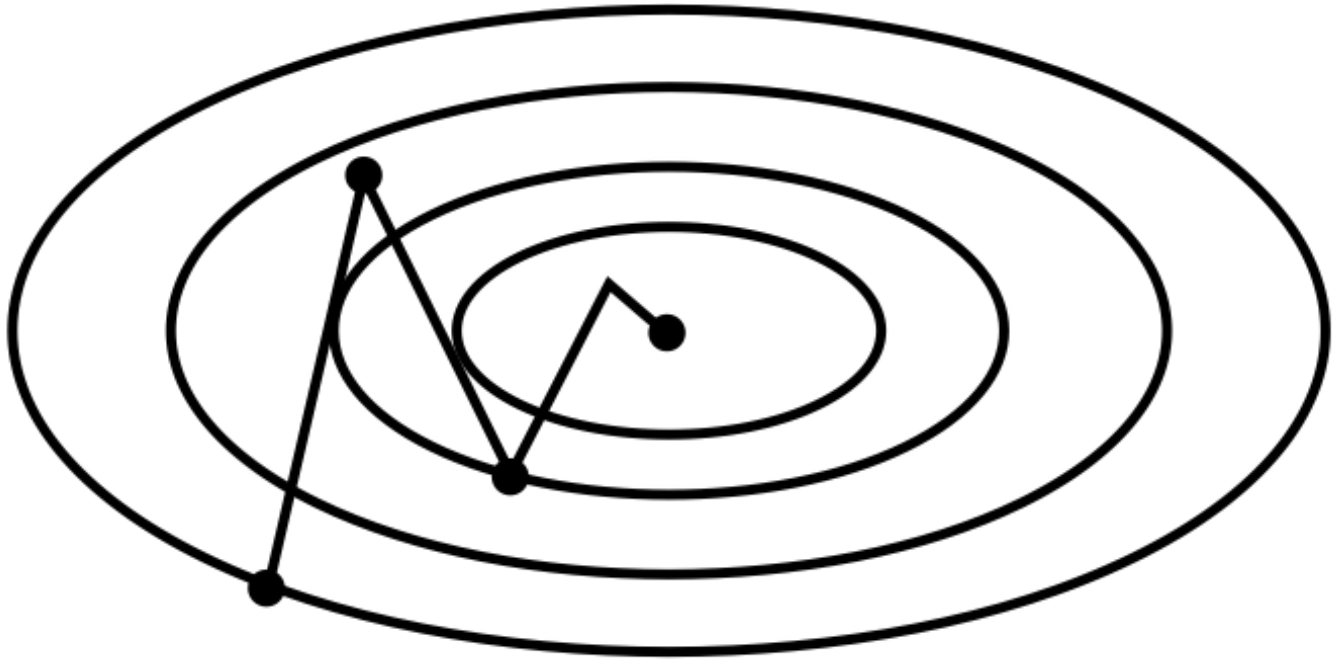
Momentum

$$h_t = \alpha h_{t-1} + \eta_t g_t$$

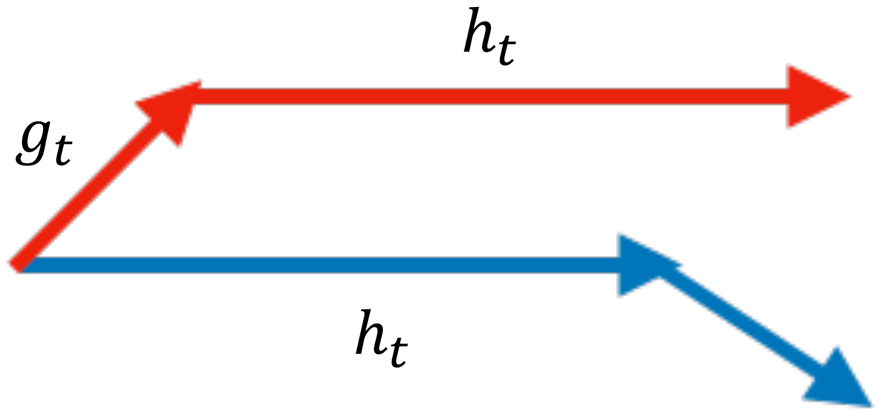
$$w^t = w^{t-1} - h_t$$

- Tends to move in the same direction as on previous steps
- h_t accumulates values along dimensions where gradients have the same sign
- Usually: $\alpha = 0.9$

Momentum



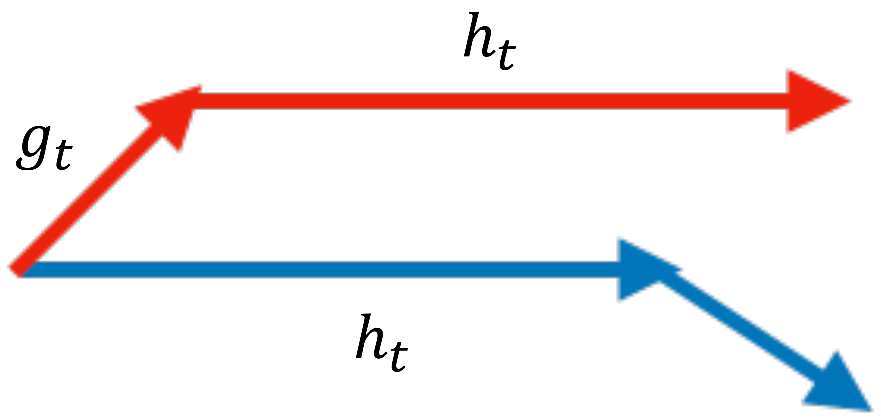
Nesterov momentum



Nesterov momentum

$$h_t = \alpha h_{t-1} + \eta_t \nabla L(w^{t-1} - \alpha h_{t-1})$$

$$w^t = w^{t-1} - h_t$$



This one actually tries to use h_t-1 into the gradient loss function, making it stronger to monmentum.

AdaGrad

Adaptive Gradient Algorihtm - the learning rate 'adapts' and slows down as time goes on.

$$G_j^t = G_j^{t-1} + g_{tj}^2$$

$$w_j^t = w_j^{t-1} - \boxed{\frac{\eta_t}{\sqrt{G_j^t + \epsilon}}} g_{tj}$$

- it CHOOSES its own learning rate for each parameter. This is good.

- g_{tj} — gradient with respect to j-th parameter
- Separate learning rates for each dimension
- Suits for sparse data
- Learning rate can be fixed: $\eta_t = 0.01$
- G_j^t always increases, leads to early stops

^ Can be a disadvantage or advantage! Early stopping could help reduce overfitting.

RMSprop

$$G_j^t = \alpha G_j^{t-1} + (1-\alpha) g_{tj}^2$$

$$w_j^t = w_j^{t-1} - \frac{\eta_t}{\sqrt{G_j^t + \epsilon}} g_{tj}$$

- α is about 0.9
- Learning rate adapts to latest gradient steps

RMSProp is like Adagrad, but is less aggressive in early stopping. We have the alpha parameter
here.
RMS prop is good for online settings.

Adam

$$v_j^t = \frac{\beta_2 v_j^{t-1} + (1-\beta_2) g_{tj}^2}{1-\beta_2^t}$$

$$w_j^t = w_j^{t-1} - \frac{\eta_t}{\sqrt{v_j^t + \epsilon}} g_{tj}$$

Adam

$$m_j^t = \frac{\beta_1 m_j^{t-1} + (1-\beta_1) g_{tj}}{1-\beta_1^t}$$

$$v_j^t = \frac{\beta_2 v_j^{t-1} + (1-\beta_2) g_{tj}^2}{1-\beta_2^t}$$

$$w_j^t = w_j^{t-1} - \frac{\eta_t}{\sqrt{v_j^t} + \epsilon} m_j^t$$

Adam actually combines RMSprop and Adagrad, so that it has both properties:
- good for adaptive learning rate - own learning rate for each parameter.
- good for online SGD-like settings.

Combines momentum and individual learning rates

Summary

- Momentum methods smooth gradients and speed up convergence
- Adaptive methods eliminate sensitive learning rate
- Adam combines both approaches