CME 216, ME 343 - Spring 2020

Eric Darve, ICME

Stanford University

There are several schemes that were designed to improve convergence.

One of the simplest trick is to use the so-called momentum.

One way to explain momentum is to go back to ordinary differential equations.

Assume we consider the following ODE:

$$\frac{dx}{dt} = -ax + f(t)$$

$a > 0$

To solve this ODE, introduce $y(t)$ with

$$x(t) = y(t)e^{-at}$$

$$x'(t) = (y'(t) - ay(t))e^{-at}$$

Differential calculus:

$$x' + ax = (y' - ay)e^{-at} + aye^{-at} = y'e^{-at}$$

From the ODE:

$$y'e^{-at} = f(t)$$

$$y(t) = \int e^{as} f(s) ds$$

$$x(t) = x_0 e^{-at} + \int_{s=0}^{t} e^{-a(t-s)} f(s) ds$$

For long times t:

$$x(t) \approx \int^{t} e^{-a(t-s)} f(s) ds$$

- when a large: only values of $f(s)$ with $s \approx t$ have a significant weight
- when a is small: x is close to $\approx \int_{t-\tau}^{t} f(s) ds$; $x(t)$ varies slowly.

Discretize in time:

$$\frac{dx}{dt} = -ax + f(t)$$

$\Rightarrow$

$$\frac{dx}{dt} \approx \frac{x_{n+1} - x_n}{\Delta t}$$

Update equation:

$$x_{n+1} = (1 - a\Delta t)x_n + \Delta t\ f_n$$

General form:

$$x_{n+1} = \beta x_n + f_n, \quad -1 < \beta < 1$$

$$x_{n+1} = \beta x_n + f_n$$

Using the same strategy we used to solve the ODE we can find the general solution:

$$x_n = x_0 \beta^n + \sum_{k=0}^{n-1} \beta^k f_{n-1-k}$$

As before:

- when $\beta \approx 1$: x_n varies slowly and we sum f_k over a large interval
- when $|\beta|$ small: only the value of f_{n-1} has a large weight and x_n varies more rapidly.

This strategy can be applied to integrate the gradient.

In the momentum method we use the following equation

$$m \leftarrow \beta m - \alpha \nabla_W L$$

$$\Delta W = m$$

For a large number of steps n:

$$m \approx -\alpha \sum_{k=0}^{n-1} \beta^{n-1-k} \nabla_W L_k$$

Take $\beta = 0.9$. We can get a huge boost.

Towards the end of the convergence when the gradient is nearly constant:

$$m \approx -\alpha \sum_{k=0}^{n-1} \beta^{n-1-k} \nabla_W L_k = -\alpha \frac{\nabla_W L_n}{1-\beta}$$

$$m \approx -10\alpha \nabla_W L_n$$

$$m \approx -10\alpha\nabla_W L_n$$

$$\Delta W = m$$

It's like converging 10 times faster to the solution!

```python
def sgd_momentum(W, m, lr, beta, batch_size):
    m = beta * m + lr * W.grad / batch_size
    W -= m
```