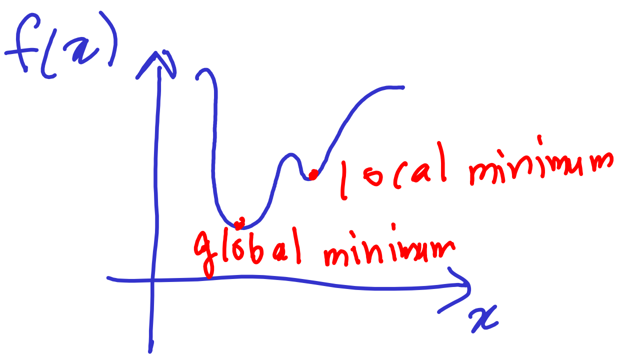
Learning to Optimize – Li, Malik

1. Optimization:

Restrict to continuous parameter space



Objective: find global minimum

Barriers: f can depend on 10s, 100s, millions, billions of parameters

⇒ cannot visualize structure of f

Solution: Iterative algorithms

start at initial solution $x^{(0)}$ and update

$$x^{(0)} \to x^{(1)} \to \dots \to x^{(t)} \to \dots x^{(T)}$$

Final

where (hopefully) $x^{(T)}$ corresponds to global minimum of $f(x)$

Task: Suggest appropriate corrections $\Delta x^{(t)}$:

$$x^{(t+1)} = x^{(t)} + \underbrace{\Delta x^{(t)}}_{\text{correction to } x^{(t)}}$$

such that:

* within reasonable time, T, we get close to global minimum

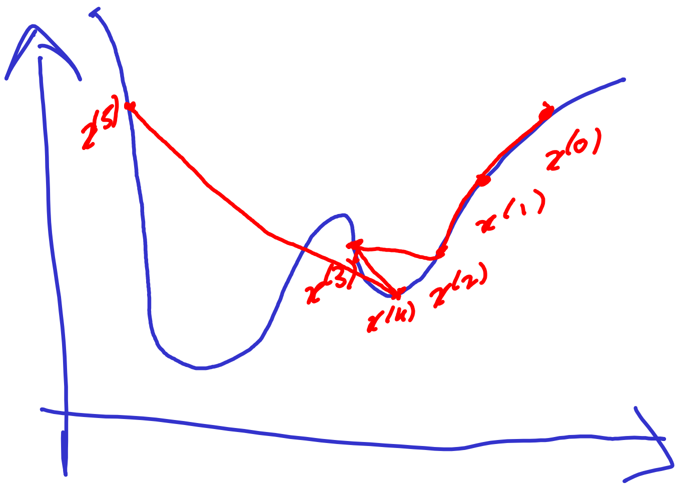
Some questions:

1. what is reasonable T? As short as possible
2. Is this task even possible? Given infinite time, YES. But realistically, have to be happy with finding local minima (in most cases).
3. what should the updates depend on? ⊛ IMPORTANT ⊛

Updates:

Each optimization algorithm corresponds to a rule for deciding $\Delta x^{(t)}$

NEXT PICASSO?



Updates should capture local and global geometry of f.

Generally, know nothing about global geometry (but sometimes we do, e.g. convex functions) but can measure local geometry.

Recall the Taylor expansion:

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2} f''(x_0)(x - x_0)^2 + \dots$$

for reasonable (smooth) functions.

It represents a series of approximation of $f(x)$ when x is close to x_0

Approx 1: $f(x) \approx f(x_0)$ constant

Approx 2: $f(x) \approx f(x_0) + f'(x_0)(x - x_0)$ straight line

Approx 3: $f(x) \approx f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2}(x - x_0)^2$

quadratic

Note how derivatives are required to understand the behavior of $f(x)$ in a neighborhood of x_0.

For multidimensional functions, the analog is:
(dim=3 below)

$$
\begin{aligned}
f(x,y,z) = f(x_0,y_0,z_0) &+ \frac{\partial f}{\partial x}(x_0,y_0,z_0)(x - x_0) \\
&+ \frac{\partial f}{\partial y}(x_0,y_0,z_0)(y - y_0) \\
&+ \frac{\partial f}{\partial z}(x_0,y_0,z_0)(z - z_0) \\
&+ \frac{1}{2}\frac{\partial^2 f}{\partial x^2}(x_0,y_0,z_0)(x - x_0)^2 + \frac{1}{2}\frac{\partial^2 f}{\partial y^2}(x_0,y_0,z_0)(y - y_0)^2 \\
&+ \frac{1}{2}\frac{\partial^2 f}{\partial z^2}(x_0,y_0,z_0)(z - z_0)^2
\end{aligned}
$$

derivs always evaluated at (x_0,y_0,z_0)

$$
\left\{ + \frac{\partial^2 f}{\partial x \partial y}(x - x_0)(y - y_0) + \frac{\partial^2 f}{\partial y \partial z}(y - y_0)(z - z_0) + \frac{\partial^2 f}{\partial x \partial z}(x - x_0)(z - z_0) \right.
$$

$+$ higher order terms

<u>Summary</u>: higher derivatives capture more local information about f.

$\Rightarrow$ using them should lead to better updates $\Delta x^{(t)}$

<u>BUT</u> they are expensive to calculate

Given: $f(x_1, \dots, x_N)$, there are

N first derivatives $- \dfrac{\partial f}{\partial x_i}$

$N + \binom{N}{2} \sim N^2$ second derivatives: $\underbrace{\dfrac{\partial^2 f}{\partial x_i^2}}_{N}$, $\underbrace{\dfrac{\partial^2 f}{\partial x_i \partial x_j}}_{\binom{N}{2}}$ $i \neq j$

$N + \binom{N}{2} + \binom{N}{3} \sim N^3$ third derivs: $\underbrace{\dfrac{\partial^3 f}{\partial x_i^3}}_{N}$, $\underbrace{\dfrac{\partial^3 f}{\partial x_i \partial^2 x_j}}_{\binom{N}{2}}$ $i \neq j$

$\underbrace{\dfrac{\partial^3 f}{\partial x_i \partial x_j \partial x_k}}_{\binom{N}{3}}$ $i \neq j \neq k$

Optimization Algorithm:

Given objective function f and initial guess x^0

While stopping criterion not reached:

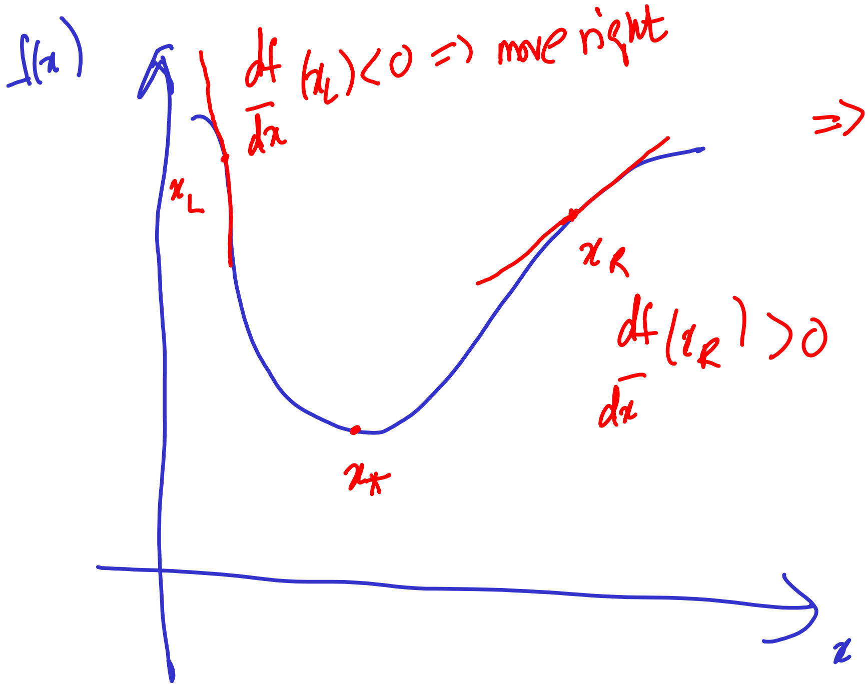
$$x^{(t+1)} = x^{(t)} + \Delta x^{(t)}$$

where $\Delta x^{(t)} = \underline{\pi}\left(f, \{x^{(0)}, \dots, x^{(t)}\}, \text{derivatives at } x^{(0)} \dots x^{(t)}\right)$

↳ core problem

Example: Gradient Descent

$$x^{(t+1)} = x^{(t)} + \underbrace{\left(-\overset{\text{learning rate}}{\eta} \frac{df}{dx}(x^{(t)})\right)}_{\Delta x^{(t)}}$$



$$\Rightarrow \Delta x^{(t)} \propto -\frac{df}{dx}(x^{(t)})$$

Gradient descent with momentum

$$x^{(t+1)} = x^{(t)} + \Delta x^{(t)}$$

$$\text{where} \quad \Delta x^{(t)} = (1-\alpha)\,\Delta x^{(t-1)} + \underbrace{\alpha}_{\in [0,1)} \left(-\eta \frac{df}{dx}(x^{(t)})\right)$$

* use history of moves

if $\alpha = 1$, get gradient descent

otherwise:

$$\Delta x^{(0)} = -\alpha \eta f'(x^{(0)}) \qquad (\Delta x^{(-1)} \equiv 0)$$

$$\begin{aligned}
\Delta x^{(1)} &= (1-\alpha)\,\Delta x^{(0)} - \alpha \eta f'(x^{(1)}) \\
&= -\alpha(1-\alpha)\eta f'(x^{(0)}) - \alpha\eta f'(x^{(1)}) \\
&= -\alpha\eta \left\{ (1-\alpha) f'(x^{(0)}) + f'(x^{(1)}) \right\}
\end{aligned}$$

$$\begin{aligned}
\Delta x^{(2)} &= (1-\alpha)\,\Delta x^{(1)} - \alpha\eta f'(x^{(2)}) \\
&= -\alpha\eta \left\{ (1-\alpha)^2 f'(x^{(0)}) + (1-\alpha) f'(x^{(1)}) \right. \\
&\qquad \left. + f'(x^{(2)}) \right\}
\end{aligned}$$

So, instead of $\Delta x^{(2)} = -\eta f'(x^{(2)})$

we get a weighted sum of f' over all past steps with steps further in history suppressed exponentially by powers of $1-\alpha$ ($1-\alpha \in [0,1)$).

<u>Natural Next step:</u>

Instead of fixing the update rule, can we make it dynamic?

<u>E.g.</u> (completely made up)

if $\dfrac{f'(x^{(t)})}{f(x^{(t)})} < 4$, $\Delta x^{(t)} = +1$

else, $\Delta x^{(t)} = -1$

Any dependencies like this should be learned.