6.006 Lecture 20 11/29/16

Topic Continuous Optimization I

Today

Continuous Optimization

Gradient Descent Method

- Convexity & Optimality

So far in class:

Combinatorial optimization:

- optimize over large finite (discrete) collection of objects

e.g. APSP - paths
sorting - orderings of lists

- brute force is possible, just not efficient

Today:

Continuous optimization:

- optimize over continuous (infinite) set of solutions
- usually: some subset of $\mathbb{R}^n$

Isn't everything on a computer finite?

still useful:

- finite, but huge
- scientific computing (solve PDE's)
- solve systems of linear equations
- robotics, control
- machine learning - deep learning
- graph algorithms
- ...

A basic, very general, problem:

Unconstrained Minimization

Given real-valued function

OBJECTIVE FUNCTION $\Rightarrow$ $f: \mathbb{R}^n \to \mathbb{R}$

find $\min_{x \in \mathbb{R}^n} f(x)$ $\Leftarrow$ x is vector in $\mathbb{R}^n$
$x = (x_1 \cdots x_n)$

Why do we call it "unconstrained"?
we let x be any vector in $\mathbb{R}^n$

OPTIMAL SOLUTION $x^* \equiv \operatorname{argmin}_{x \in \mathbb{R}^n} f(x)$

Comments:

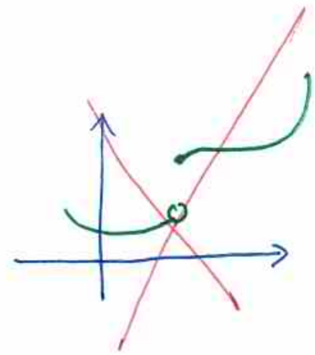
- This is pretty general!

e.g. • to solve $\max_{x \in \mathbb{R}^n} f(x)$, just solve $\min_{x \in \mathbb{R}^n} -f(x)$

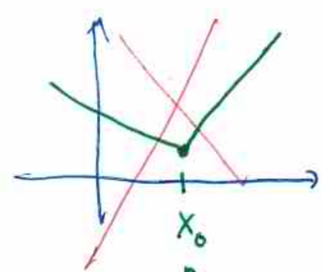
- can encode any continuous optimization task by making f sufficiently complex
  e.g. if want constrained optimization, make $f(x) = \infty$ when x doesn't satisfy constraints

Assumptions on f:

- f is continuous



- f is (infinitely) differentiable



assumptions can be relaxed

How to Solve?

luckily we have a powerful approach coming to the rescue ...

GRADIENT DESCENT

Key Idea: Greedy local search

- start with some initial point x^0
- in each iteration, move a bit in direction that reduces value of f the most

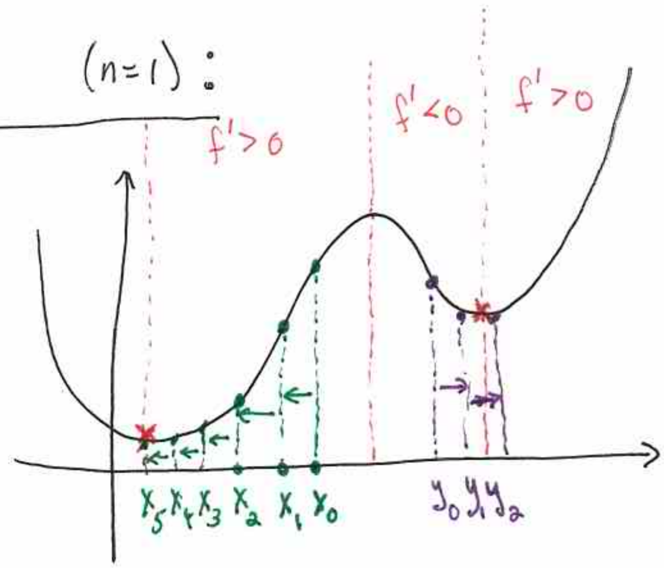
⇐ "local"

⇐ "greedy"

- guarantees $f(x^{i+1}) \leq f(x^i)$

Warmup:

Gradient descent in 1-D $(n=1)$:



Algorithm idea:

$i \leftarrow 0$
$x^0 \leftarrow$ arbitrary point

While $f'(x^i) \not\approx 0$ do

if $f'(x^i) > 0$: move left

if $f'(x^i) < 0$: move right

$i \leftarrow i+1$

if $\underbrace{f'(x^i) \approx 0}_{\text{local optimality condition}}$ then x is $\underbrace{\text{local extremum}}_{\text{no more local improvement possible}}$

minimum 🙂
or
maximum ☹️
⇑
only happens if unlucky

Important note:
different starting points x^0 give different local maxima

Questions:

- how far to move?
- What does "$\approx$" mean?

A closer look:

Taylor Expansion for $x' = x + \varepsilon$

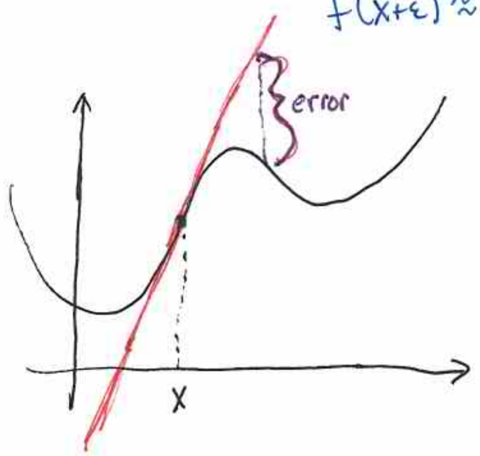
$$f(x') \approx f(x+\varepsilon) = \underbrace{f(x) + f'(x)\cdot\varepsilon}_{\text{use this to approximate } f \text{ "locally" as a linear fctn}} + \underbrace{\frac{\varepsilon^2}{2!} f''(x) + \frac{\varepsilon^3}{3!} f'''(x) + \dots}_{\text{error of pretending } f \text{ is linear}}$$

assuming ε small $\approx \varepsilon^2 f''(x)$

$\Rightarrow$ will use small ε + assume

$$f(x+\varepsilon) \approx f(x) + f'(x)\cdot\varepsilon$$

so treat f as a linear function!!



tangent of f at x

What should the step size be?

- needs to be small enough such that approximation is still "good"
- needs to be large enough such that we make some progress?

Step size/direction

- move in direction of $-f'(x)$
- move by amount $-\eta f'(x)$

why should step size depend on f'? we will see later that this works really well for analysis!

η ← stepsize / learning rate

η depends on f

Claim good choice of η is $\eta = \frac{1}{f''(x)}$

why?

good thing to know otherwise you aren't necessarily getting better

⟹ Make sure the improvement is bigger than the error!

improvement:

ε from before, now set to $-\eta f'(x^i)$

$$f(x^{i+1}) = f(x^i \underbrace{- \eta f'(x^i)}_{\varepsilon})$$

$$\approx f(x^i) - \underbrace{(\eta f'(x^i))}_{\varepsilon} f'(x^i) + \text{Error term}$$

$$\approx f(x^i) - \underbrace{\eta f'(x^i)^2}_{\text{improvement / progress}}$$

error: bound by twice next term in Taylor expansion

$$\cancel{2} \cdot \frac{\varepsilon^2}{\cancel{2}!} f''(x^i) = \underbrace{\eta^2 f'(x^i)^2 f''(x^i)}_{\text{approx error}}$$

more exact calculations $\Rightarrow \eta \approx \frac{1}{2f''(x)}$

So need $\cancel{\eta} \cancel{f'(x^i)^2} \gtrsim \eta^{\cancel{2}} \cancel{f'(x^i)^2} f''(x^i) \Rightarrow \eta \approx 1/f''(x^i)$

$$\boxed{\text{So progress in each step: } \approx \eta (f'(x)^2) \approx \frac{f'(x)^2}{f''(x)}}$$

Commercial break:

Learning (Deep and Shallow ...)

Given a bunch of points

$(x_1, y_1)\ (x_2, y_2)\ \ldots$

e.g. x_i = picture $y_i = 1$ if it is a table, 0 o.w

x_i = vector $y_i = Ax$ ← unknown matrix

x_i = patient information y_i = probability patient dies

Goal we want to "learn" a function, which given x_i computes a good prediction to y_i

In general, learning is not really possible

can't learn a random function!

However, we could modify our goal. Here is an example:

Goal' find a linear function

{ could be any class of functions e.g. deep learning networks, human brains, simple network of Boolean logic

which best approximates (x_i, y_i) relationship under L_2 - error

} could be any other error (L_1, information theoretic loss functions)

more specifically:

for hypothesis linear function f_w

the loss of f_w on input is denoted $\mathrm{Loss}(f_w(x_i), y_i)$

- could use $\mathrm{loss}(f_w(x_i), y_i) = (f_w(x_i) - y_i)^2$ ⟸ "L_2-error" "quadratic loss" lots of other names

Total loss of hypothesis on input:

$$\text{Total loss} = \sum_i \text{loss}(f_w(x_i), y_i)$$

But w is a set of parameters (determining a linear function)

how do we find w that minimizes total loss: for given $(x_1, y_1)(x_2, y_2)\ldots$

GRADIENT DESCENT

<u>Deep learning</u>

f_w is not linear

but still, w is a bunch of parameters to a network back to work (now we motivated general case...)

General Case $(n>1)$

Same philosophy: f is locally (multivariate) linear function

What is different? x is now a vector

"grad f" $\nabla f(x)$ is analogue of derivative $= \begin{bmatrix} \frac{\partial f}{\partial x_1}(x) \\ \frac{\partial f}{\partial x_2}(x) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x) \end{bmatrix}$

it is also a vector!

Still have: $f(\underbrace{x+\varepsilon}_{\text{vector in } \mathbb{R}^n}) = \underbrace{f(x) + \nabla f(x)^T \cdot \varepsilon}_{\text{linear approximation}} + \underbrace{\frac{1}{2}\varepsilon^T \nabla^2 f(x)\, \varepsilon + \dots}_{\text{error}}$

Notation:

$\begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}^T = [a_1 \dots a_n]$ "transpose"

Inner Product

$a^T b = \sum_{i=1}^{n} a_i b_i = \langle a,b \rangle$

Analogue of 2^{nd} derivative:

"Hessian" how locally non linear the fctn is

$$\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1}(x) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(x) & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(x) \\ \frac{\partial^2 f}{\partial x_2 \partial x_1}(x) & \ddots & & \vdots \\ \vdots & & & \\ \frac{\partial^2 f}{\partial x_n \partial x_1}(x) & \dots & & \frac{\partial^2 f}{\partial x_n \partial x_n}(x) \end{bmatrix}$$

Gradient Descent Algorithm

- start with x^0
- For $k = 0 \ldots$ do

$$x^{k+1} = x^k - \eta \nabla f(x^k)$$

$\Leftarrow$ Coordinate-wise do this by $x_i^{k+1} = x_i^k - \eta \frac{\partial f}{\partial x_i}(x_i^k)$

$$\boxed{\text{local optimality condition: } \nabla f(x) = 0}$$

$\Leftarrow$ x is either local min or $\underbrace{\text{"saddle point"}}_{\text{not local min}}$

Example1: Linear Function

$$f(x) = c^T x + b$$

vectors!

Note: $\nabla f(x) = (c_1 \ldots c_n) = c$
$\nabla^2 f(x) = (0 \cdots 0)$

so no bound on step size η

what if we move in direction d?

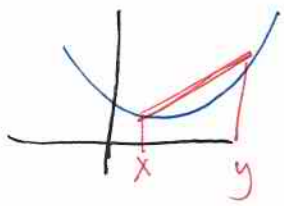
$$f(x+d) = c^T(x+d) + b = c^T x + c^T d + b$$
$$= f(x) + c^T d$$
$$= f(x) + \|c\| \cdot \|d\| \cdot \cos\theta$$

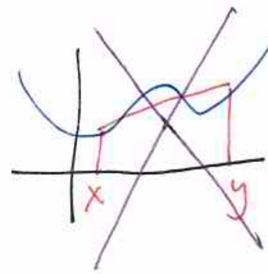
So, best choice of d: let $d = -c = -\nabla f(x)$ $\quad \theta = 180^\circ$, $\cos\theta = -1$

best bang for the buck: $f(x+d) = f(x) - \|c\|^2$

When does the local min equal the global min?

Important case: Convexity





f is "convex" iff

$\forall \; 0 \le \lambda \le 1, \; x, y \in \mathbb{R}^n \qquad f(\lambda x + (1-\lambda) y) \le \lambda f(x) + (1-\lambda) f(y)$

every chord is above the fctn.

$\Updownarrow$

$\forall x, y \quad f(x+y) \ge f(x) + \nabla f(x)^T (y-x)$

Key fact: convex $\Rightarrow$ every local min is a global min