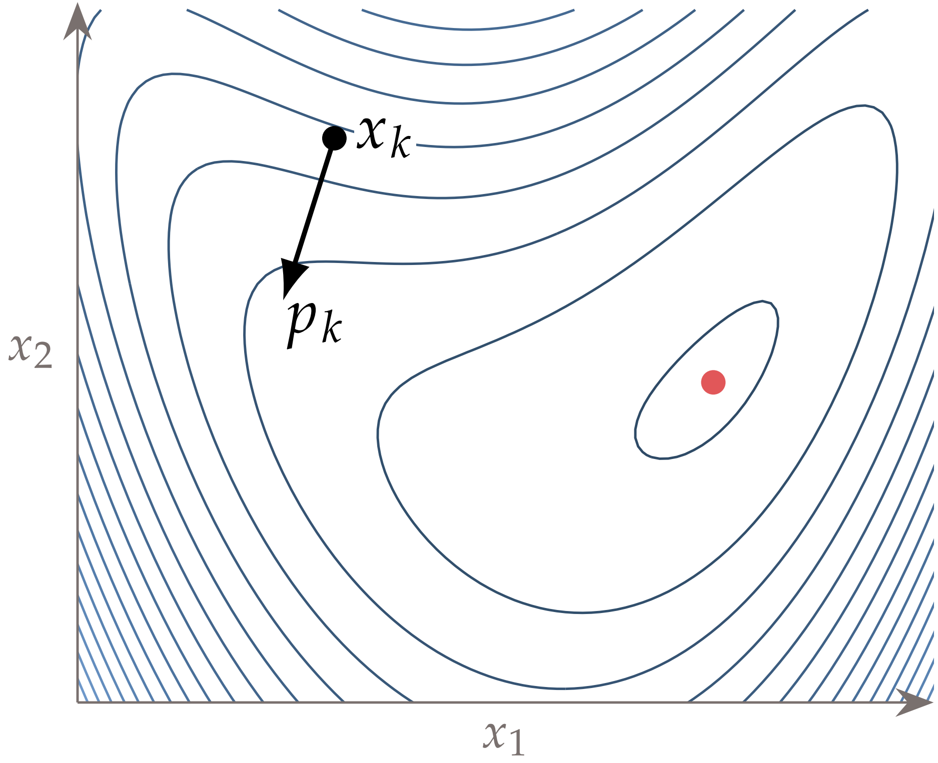
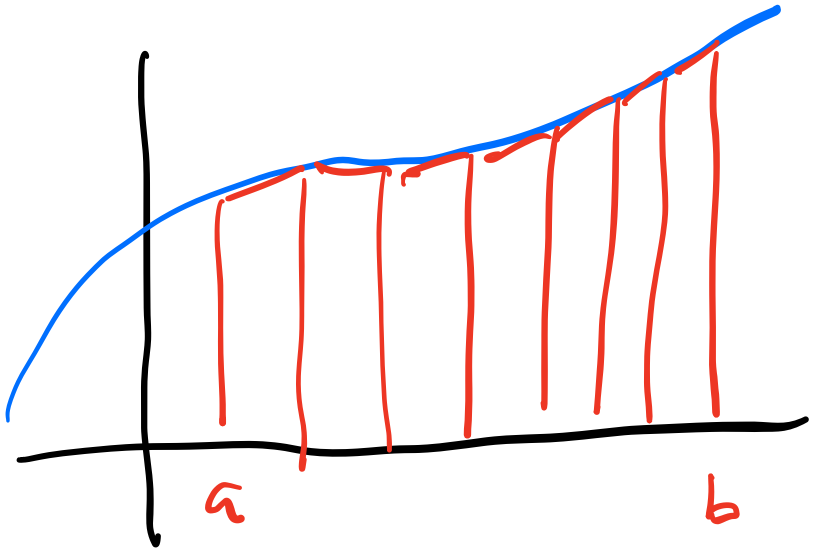
Numerical Differentiation



ME EN 275
Andrew Ning
aning@byu.edu

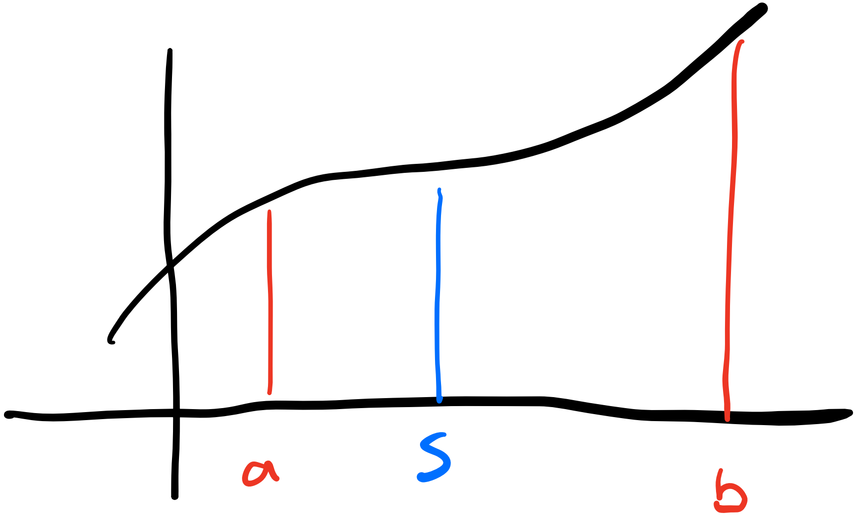
Integration Review



np.trapz (numpy v1) or np.trapezoid (numpy v2)

scipy.integrate.quad

Cumulative Integration



$$I(s) = \int_a^{s\ \cancel{b}} f(x)\,dx$$

vary s from a to b

$$f(x) = \sin(x)$$

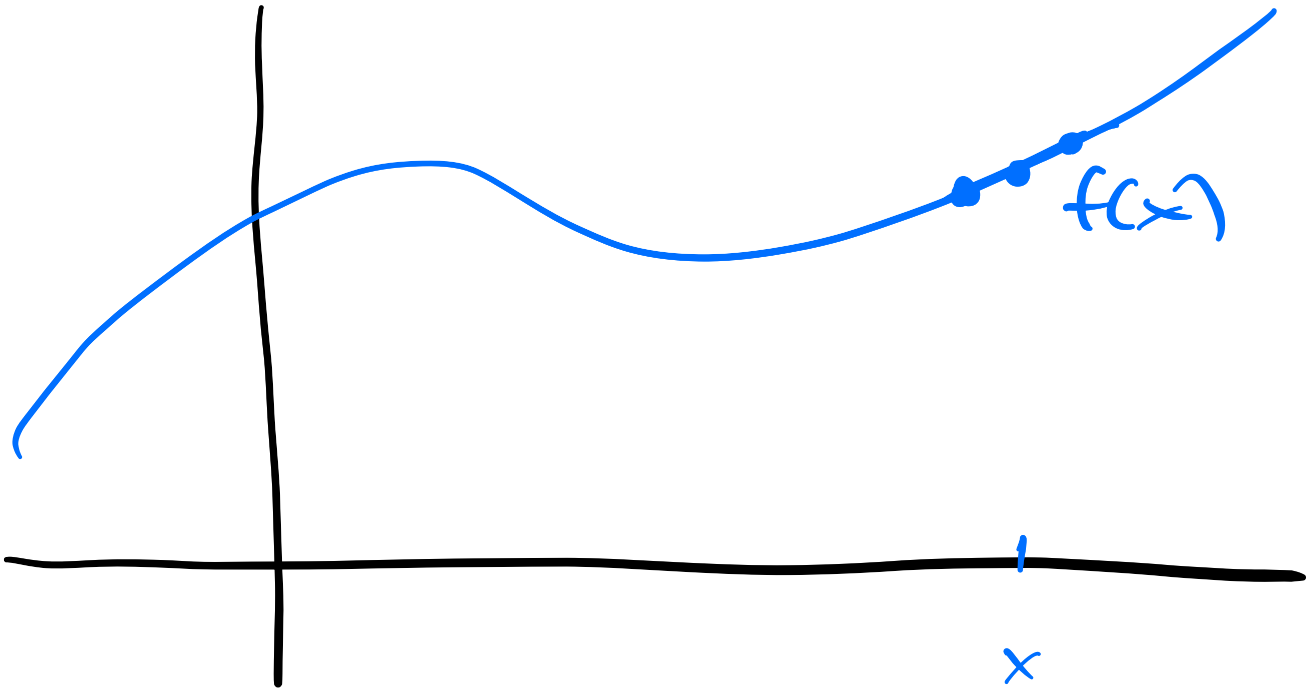
$$x = 0 \ldots \pi$$

scipy.integrate.cumulative_trapezoid

Motivation

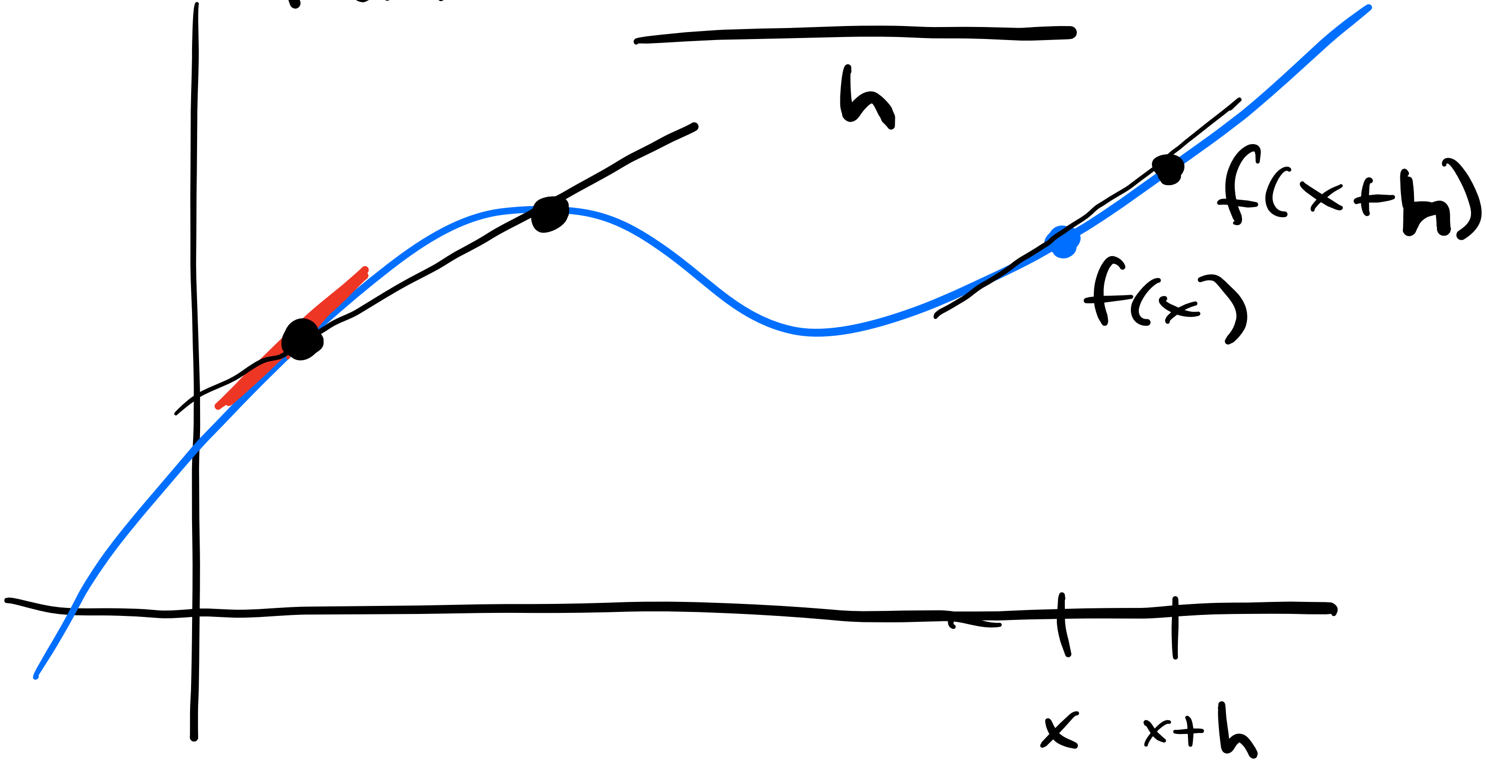
structures
fluids
heat transfer
robotics
materials
optimization
machine learning

How would you estimate the derivative numerically?



Finite Differencing

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

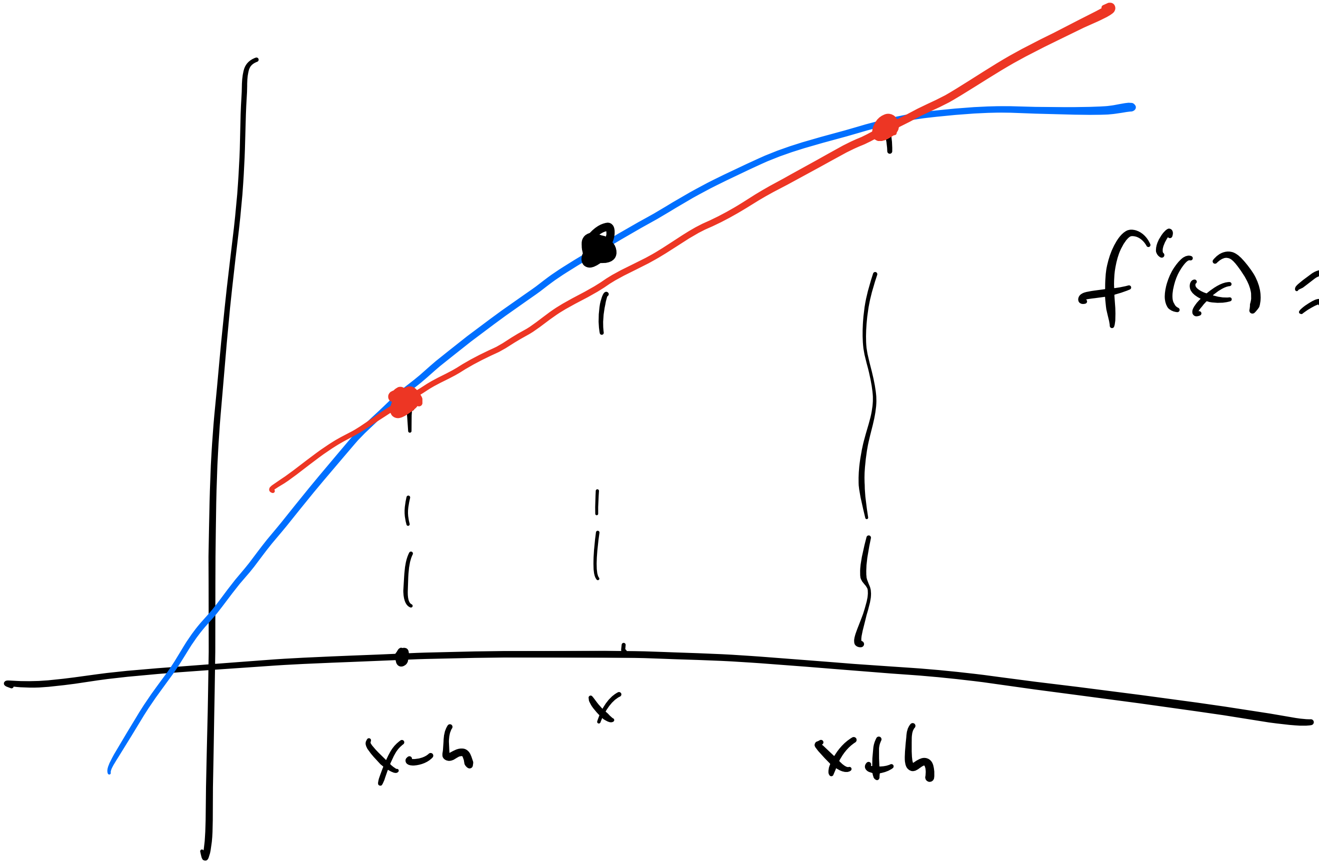


Taylor's series

$$f(x+h) = f(x) + h\frac{df}{dx} + \frac{h^2}{2!}\frac{d^2f}{dx^2} + \cdots$$

$$\frac{df}{dx} = \frac{f(x+h) - f(x)}{h} + O(h)$$

Central Differencing



$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

But small steps leads to roundoff errors

$$\frac{f(x+h)-f(x)}{h}$$

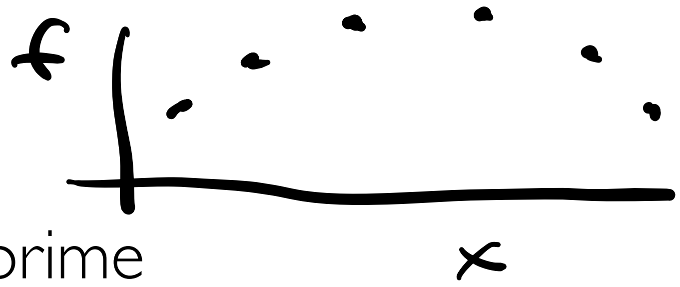
h	$f(x+h)$	Δf	$\mathrm{d}f/\mathrm{d}x$
10^{-1}	4.9562638252880662	0.4584837713419043	4.58483771
10^{-2}	4.5387928890592475	0.0410128351130856	4.10128351
10^{-4}	4.4981854440562818	0.0004053901101200	4.05390110
10^{-6}	4.4977841073787870	0.0000040534326251	4.05343263
10^{-8}	4.4977800944804409	0.0000000405342790	4.05342799
10^{-10}	4.4977800543515052	0.0000000004053433	4.05344203
10^{-12}	4.4977800539502155	0.0000000000040536	4.05453449
10^{-14}	4.4977800539462027	0.0000000000000409	4.17443857
10^{-16}	4.4977800539461619	0.0000000000000000	0.00000000
10^{-18}	4.4977800539461619	0.0000000000000000	0.00000000
Exact	4.4977800539461619		4.05342789

Step Size Dilemma

Functions

data points

np.gradient



scipy.optimize.approx_fprime

function
(limited to fwd)

usually we just write our own

Let's try it

$$y = x^3$$

compute $\dfrac{dy}{dx}$ at $x = 2$

try various step sizes.

Bonus: automatic differentiation