AI505

Optimization

Derivatives and Gradients

Marco Chiarandini

Department of Mathematics & Computer Science
University of Southern Denmark

Outline

1. Derivaties

2. Numerical Differentiation

3. Automatic Differentiation

Definitions

- $[a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}$ closed interval
  $(a, b) = \{x \in \mathbb{R} \mid a < x < b\}$ open interval

- column vectors and matrices
  scalar product: $\boldsymbol{y}^T \boldsymbol{x} = \sum_{i=1}^{n} y_i x_i$

- $A\boldsymbol{x}$ column vector combination of the columns of A;
  $\boldsymbol{u}^T A$ row vector combination of the rows of A

- **linear combination**

$$\boldsymbol{v}_1, \boldsymbol{v}_2 \ldots, \boldsymbol{v}_k \in \mathbb{R}^n \qquad \boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_k]^T \in \mathbb{R}^k$$

$$\boldsymbol{x} = \lambda_1 \boldsymbol{v}_1 + \cdots + \lambda_k \boldsymbol{v}_k = \sum_{i=1}^{k} \lambda_i \boldsymbol{v}_i$$

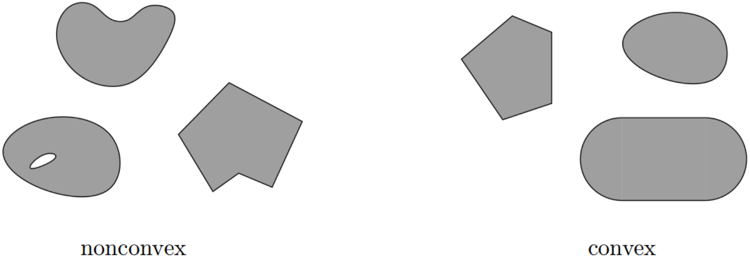
moreover:

$\boldsymbol{\lambda} \geq 0$ **conic combination**

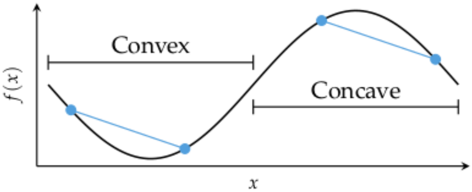
$\boldsymbol{\lambda}^T 1 = 1$ **affine combination** $\left(\sum_{i=1}^{k} \lambda_i = 1\right)$

Definitions

- **convex set**: if $\boldsymbol{x}, \boldsymbol{y} \in S$ and $0 \leq \lambda \leq 1$ then $\lambda \boldsymbol{x} + (1-\lambda)\boldsymbol{y} \in S$



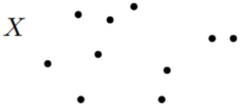
- **convex function** if its epigraph $\{(x, y) \in \mathbb{R}^2 : y \geq f(x)\}$ is a convex set or if $f : \mathbb{R}^n \to \mathbb{R}$ and
  if $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \alpha \in [0, 1]$ it holds that
  $f(\alpha \boldsymbol{x} + (1-\alpha)\boldsymbol{y}) \leq \alpha f(\boldsymbol{x}) + (1-\alpha) f(\boldsymbol{y})$

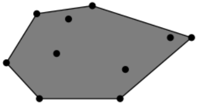


Definitions

- For a set of points $S \subseteq \mathbb{R}^n$

$$\begin{array}{rl} \mathrm{lin}(S) & \text{linear hull (span)} \\ \mathrm{cone}(S) & \text{conic hull} \\ \mathrm{aff}(S) & \text{affine hull} \\ \mathrm{conv}(S) & \text{convex hull} \end{array}$$





the convex hull of X

$$\boxed{\mathrm{conv}(X) = \{\lambda_1 \boldsymbol{x}_1 + \lambda_2 \boldsymbol{x}_2 + \ldots + \lambda_n \boldsymbol{x}_n \ \mid\ \boldsymbol{x}_i \in X,\ \lambda_1, \ldots, \lambda_n \geq 0 \text{ and } \textstyle\sum_i \lambda_i = 1\}}$$

Norms

Def. A **norm** is a function that assigns a length to a vector.

A function f is a norm if:

1. $f(\boldsymbol{x}) = 0$ if and only if $\boldsymbol{x}$ is the zero vector
2. $f(a\boldsymbol{x}) = |a| f(\boldsymbol{x})$, such that lengths scale
3. $f(\boldsymbol{x} + \boldsymbol{y}) \leq f(\boldsymbol{x}) + f(\boldsymbol{y})$, also known as trinagle inequality

L_p norms are commonly used set of norms paramterized by a scalar $p \geq 1$:

$$\|x\|_p = \lim_{\rho \to p} \left(|x_1|^\rho + |x_2|^\rho + \ldots + |x_n|^\rho\right)^{\frac{1}{\rho}}$$

L_∞ is also called the max norm, Chebyshev distance or chessboard distance.

Outline

1. Derivaties

2. Numerical Differentiation

3. Automatic Differentiation

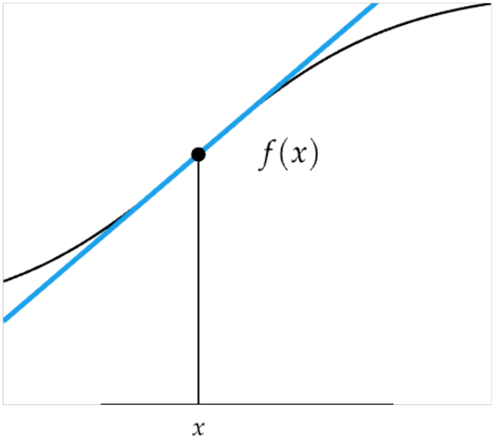
Derivaties

- Derivatives tell us which direction to search for a solution

- Slope of Tangent Line

$$f'(x) := \frac{\mathrm{d}f(x)}{\mathrm{d}x}$$

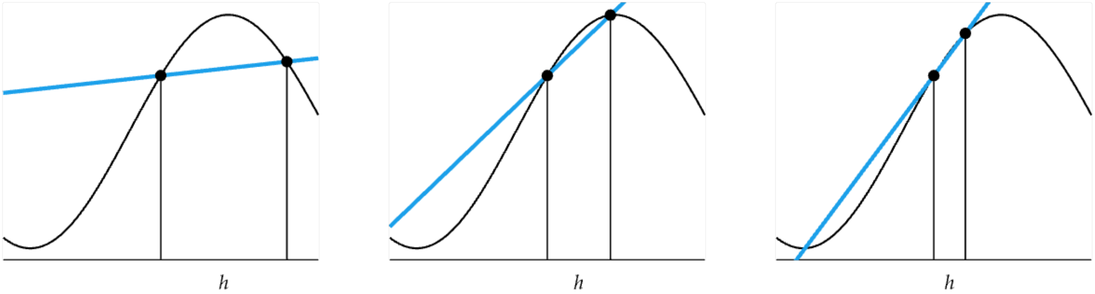
(Leibniz notation)



Derivatives

$$f(x+\Delta x) \approx f(x) + f'(x)\Delta x$$

$$f'(x) = \frac{\Delta x}{\Delta x}$$



Symbolic Differentiation

$$f'(x) \equiv \underbrace{\lim_{h\to 0} \frac{f(x+h) - f(x)}{h}}_{\text{forward difference}} = \underbrace{\lim_{h\to 0} \frac{f(x+h/2) - f(x-h/2)}{h}}_{\text{central difference}} = \underbrace{\lim_{h\to 0} \frac{f(x) - f(x-h)}{h}}_{\text{backward difference}}$$

Symbolic Differentiation

```python
import sympy as sp

# Define the variable
x = sp.symbols('x')

# Define the function
f = x**2 + x/2 - sp.sin(x)/x

# Compute the derivative
df_dx = sp.diff(f, x)

# Display the result
print("The symbolic derivative of f is:")
print(df_dx)
```

derivative.py

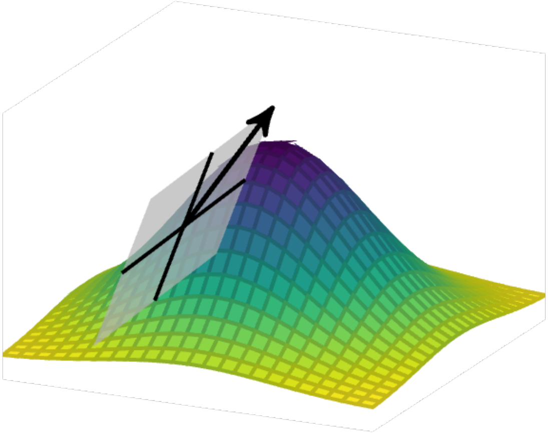
Derivatives in Multiple Dimensions

- Gradient Vector

$$\nabla f(\boldsymbol{x}) = \left[\frac{\partial f(\boldsymbol{x})}{\partial x_1}, \frac{\partial f(\boldsymbol{x})}{\partial x_2}, \ldots, \frac{\partial f(\boldsymbol{x})}{\partial x_n} \right]$$

- Hessian Matrix

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



Directional derivative

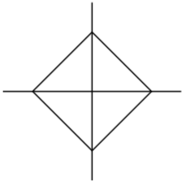
The **directional derivative** $\nabla_{\boldsymbol{s}} f(\boldsymbol{x})$ of a multivariate function f is the instantaneous rate of change of $f(\boldsymbol{x})$ as $\boldsymbol{x}$ is moved with velocity $\boldsymbol{s}$.

$$\nabla_{\mathbf{s}} f(\mathbf{x}) \equiv \underbrace{\lim_{h \to 0} \frac{f(\mathbf{x} + h\mathbf{s}) - f(\mathbf{x})}{h}}_{\text{forward difference}} = \underbrace{\lim_{h \to 0} \frac{f(\mathbf{x} + h\mathbf{s}/2) - f(\mathbf{x} - h\mathbf{s}/2)}{h}}_{\text{central difference}} = \underbrace{\lim_{h \to 0} \frac{f(\mathbf{x}) - f(\mathbf{x} - h\mathbf{s})}{h}}_{\text{backward difference}}$$

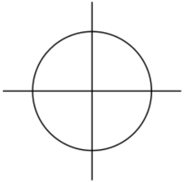
To compute $\nabla_{\boldsymbol{s}} f(\boldsymbol{x})$:

- compute $\nabla_{\boldsymbol{s}} f(\boldsymbol{x}) = \nabla f(\boldsymbol{x})^T s$
- $g(\alpha) := f(\boldsymbol{x} + \alpha \boldsymbol{s})$ and then compute $g'(0)$

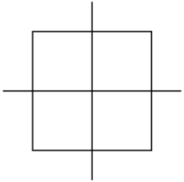
L_1: $\|\mathbf{x}\|_1 = |x_1| + |x_2| + \cdots + |x_n|$



L_2: $\|\mathbf{x}\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$



L_∞: $\|\mathbf{x}\|_\infty = \max(|x_1|, |x_2|, \cdots, |x_n|)$



Matrix Calculus

Common gradient:

$$\nabla_{\boldsymbol{x}} \boldsymbol{b}^T \boldsymbol{x} = ?$$

$$\boldsymbol{b}^T \boldsymbol{x} = [b_1 x_1 + b_2 x_2 + \ldots + b_n x_n]$$

$$\frac{\partial \boldsymbol{b}^T \boldsymbol{x}}{\partial x_i} = b_i$$

$$\nabla_{\boldsymbol{x}} \boldsymbol{b}^T \boldsymbol{x} = \nabla_{\boldsymbol{x}} \boldsymbol{x}^T \boldsymbol{b} = \boldsymbol{b}$$

Matrix Calculus

Common gradient:

$$\nabla_{\boldsymbol{x}} \boldsymbol{x}^T A \boldsymbol{x} = ?$$

$$\boldsymbol{x}^T A \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^T \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}^T \begin{bmatrix} x_1 a_{11} + x_2 a_{12} + \dots + x_n a_{1n} \\ x_1 a_{21} + x_2 a_{22} + \dots + x_n a_{2n} \\ \vdots \\ x_1 a_{n1} + x_2 a_{n2} + \dots + x_n a_{nn} \end{bmatrix}$$

$$= \begin{matrix} x_1^2 a_{11} + x_1 x_2 a_{12} + \dots + x_1 x_n a_{1n} + \\ x_1 x_2 a_{21} + x_2^2 a_{22} + \dots + x_2 x_n a_{2n} + \\ \vdots \\ x_1 x_n a_{n1} + x_2 x_n a_{n2} + \dots + x_n^2 a_{nn} \end{matrix}$$

$$\frac{\partial}{\partial x_i}\mathbf{x}^T A\mathbf{x} = \sum_{j=1}^{n} x_j\left(a_{ij} + a_{ji}\right)$$

$$\nabla_x \mathbf{x}^T A\mathbf{x} = \begin{bmatrix} \sum_{j=1}^{n} x_j\left(a_{1j} + a_{j1}\right) \\ \sum_{j=1}^{n} x_j\left(a_{2j} + a_{j2}\right) \\ \vdots \\ \sum_{j=1}^{n} x_j\left(a_{nj} + a_{jn}\right) \end{bmatrix} = \begin{bmatrix} a_{11} + a_{11} & a_{12} + a_{21} & \dots & a_{1n} + a_{n1} \\ a_{21} + a_{12} & a_{22} + a_{22} & \dots & a_{2n} + a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} + a_{1n} & a_{n2} + a_{2n} & \dots & a_{nn} + a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \left(A + A^T\right)\mathbf{x}$$

Smoothness

Def. The **smoothness** of a function is a property measured by the number of continuous derivatives (differentiability class) it has over its domain.

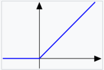
A function of **class** C^k is a function of smoothness at least k; that is, a function of class C^k is a function that has a kth derivative that is continuous in its domain.

The term **smooth function** refers to a C^∞-function. However, it may also mean "sufficiently differentiable" for the problem under consideration.

Smoothness

- Let U be an open set on the real line and a function f defined on U with real values. Let k be a non-negative integer.
- The function f is said to be of **differentiability class** C^k if the derivatives $f', f'', \ldots, f^{(k)}$ exist and are continuous on U.
- If f is k-differentiable on U, then it is at least in the class C^{k-1} since $f', f'', \ldots, f^{(k-1)}$ are continuous on U.
- The function f is said to be **infinitely differentiable**, **smooth**, or of **class** C^∞, if it has derivatives of all orders (continous) on U.
- The function f is said to be of **class** C^ω, or **analytic**, if f is smooth and its Taylor series expansion around any point in its domain converges to the function in some neighborhood of the point.
- There exist functions that are smooth but not analytic; C^ω is thus strictly contained in C^∞. Bump functions are examples of functions with this property.

Example: continuous (C^0) but not differentiable [edit]



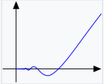
The C^0 function $f(x) = x$ for $x \geq 0$ and 0 otherwise.

The function

$$f(x) = \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{if } x < 0 \end{cases}$$

is continuous, but not differentiable at $x = 0$, so it is of class C^0, but not of class C^1.

Example: finitely-times differentiable (C^k) [edit]



The function $g(x) = x^2 \sin(1/x)$ for $x > 0$.

For each even integer k, the function

$$f(x) = |x|^{k+1}$$

is continuous and k times differentiable at all x. At $x = 0$, however, f is not $(k + 1)$ times differentiable, so f is of class C^k, but not of class C^j where $j > k$.

Example: differentiable but not continuously differentiable (not C^1) [edit]

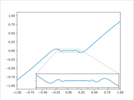
The function

$$g(x) = \begin{cases} x^2 \sin\left(\frac{1}{x}\right) & \text{if } x \neq 0, \\ 0 & \text{if } x = 0 \end{cases}$$

is differentiable, with derivative

$$g'(x) = \begin{cases} -\cos\left(\frac{1}{x}\right) + 2x \sin\left(\frac{1}{x}\right) & \text{if } x \neq 0, \\ 0 & \text{if } x = 0. \end{cases}$$

Because $\cos(1/x)$ oscillates as $x \to 0$, $g'(x)$ is not continuous at zero. Therefore, $g(x)$ is differentiable but not of class C^1.



The function $f : \mathbb{R} \to \mathbb{R}$ with $f(x) = x^2 \sin\left(\frac{1}{x}\right)$ for $x \neq 0$ and $f(0) = 0$ is differentiable. However, this function is not continuously differentiable.

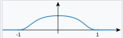
Example: differentiable but not Lipschitz continuous [edit]

The function

$$h(x) = \begin{cases} x^{4/3} \sin\left(\frac{1}{x}\right) & \text{if } x \neq 0, \\ 0 & \text{if } x = 0 \end{cases}$$

is differentiable but its derivative is unbounded on a compact set. Therefore, h is an example of a function that is differentiable but not locally Lipschitz continuous.

Example: analytic (C^ω) [edit]



A smooth function that is not analytic.

The exponential function e^x is analytic, and hence falls into the class C^ω (where ω is the smallest transfinite ordinal). The trigonometric functions are also analytic wherever they are defined, because they are linear combinations of complex exponential functions e^{ix} and e^{-ix}.

Example: smooth (C^∞) but not analytic (C^ω) [edit]

The bump function

$$f(x) = \begin{cases} e^{-\frac{1}{1-x^2}} & \text{if } |x| < 1, \\ 0 & \text{otherwise} \end{cases}$$

is smooth, so of class C^∞, but it is not analytic at $x = \pm 1$, and hence is not of class C^ω. The function f is an example of a smooth function with compact support.

Positive Definteness

Def. A symmetric matrix A is **positive definite** if $\boldsymbol{x}^T A\boldsymbol{x}$ is positive for all points other than the origin: $x^T A\boldsymbol{x} > 0$ for all $x \neq 0$.
Def. A symmetric matrix A is **positive semidefinite** if $\boldsymbol{x}^T A\boldsymbol{x}$ is always non-negative: $\boldsymbol{x}^T A\boldsymbol{x} \geq 0$ for all $\boldsymbol{x}$.

If the matrix A is positive definite in the function $f(\boldsymbol{x}) = x^T A\boldsymbol{x}$, then f has a unique global minimum.

Recall that the second order Taylor approximation of a twice-differentiable function f at x_0 is

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}_0) + \nabla f(x_0)^T (x - x_0) + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_0)^T H_0(\boldsymbol{x} - \boldsymbol{x}_0)$$

where H_0 is the Hessian evaluated at $\boldsymbol{x}_0$. If $(\boldsymbol{x} - \boldsymbol{x}_0)^T H_0(\boldsymbol{x} - \boldsymbol{x}_0)$ has a unique global minimum, then the overall approximation has a unique global minimum.

We wish to compute the directional derivative of $f(\mathbf{x}) = x_1 x_2$ at $\mathbf{x} = [1, 0]$ in the direction $\mathbf{s} = [-1, -1]$:

$$
\begin{aligned}
\nabla f(\mathbf{x}) &= \left[\frac{\partial f}{\partial x_1}, \quad \frac{\partial f}{\partial x_2}\right] = [x_2, x_1] \\
\nabla_{\mathbf{s}} f(\mathbf{x}) &= \nabla f(\mathbf{x})^\top \mathbf{s} = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \end{bmatrix} = -1
\end{aligned}
$$

We can also compute the directional derivative as follows:

$$
\begin{aligned}
g(\alpha) &= f(\mathbf{x} + \alpha \mathbf{s}) = (1 - \alpha)(-\alpha) = \alpha^2 - \alpha \\
g'(\alpha) &= 2\alpha - 1 \\
g'(0) &= -1
\end{aligned}
$$

Outline

Numerical Differentiation

Finite Difference Method

- Neighboring points are used to approximate the derivative

$$f'(x) \approx \underbrace{\frac{f(x+h)-f(x)}{h}}_{\text{forward difference}} \approx \underbrace{\frac{f(x+h/2)-f(x-h/2)}{h}}_{\text{central difference}} \approx \underbrace{\frac{f(x)-f(x-h)}{h}}_{\text{backward difference}}$$

- h too small causes numerical cancellation errors (square root or cube root of the machine precision for floating point values: `sys.float_info.epsilon` difference between 1 and closest representable number)

Derivation

from Taylor series expansion:

$$f(x+h) = f(x) + \frac{f'(x)}{1!}h + \frac{f''(x)}{2!}h^2 + \frac{f'''(x)}{3!}h^3 + \cdots$$

We can rearrange and solve for the first derivative:

$$f'(x)h = f(x+h) - f(x) - \frac{f''(x)}{2!}h^2 - \frac{f'''(x)}{3!}h^3 - \cdots$$

$$f'(x) = \frac{f(x+h) - f(x)}{h} - \frac{f''(x)}{2!}h - \frac{f'''(x)}{3!}h^2 - \cdots$$

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

- forward difference has error term $O(h)$, linear error as h approaches zero
- central difference has error term is $O(h^2)$

```python
import sys
import numpy as np

def diff_forward(f, x: float, h: float=np.sqrt(sys.float_info.epsilon)) -> float:
    return (f(x+h) - f(x))/h

def diff_central(f, x: float, h: float=np.cbrt(sys.float_info.epsilon)) -> float:
    return (f(x+h/2) - f(x-h/2))/h

def diff_backward(f, x: float, h: float=np.sqrt(sys.float_info.epsilon)) -> float:
    return (f(x) - f(x-h))/h

# Example usage
def func(x):
    return x**2 + np.sin(x)

x0 = 1.0
print(f"The derivative at x = {x0} is {diff_forward(func, x0)}")
```

finite_diff.py

Numerical Differentiation

Complex step method

Uses one single function evaluation after taking a step in the imaginary direction.

$$f(x+ih) = f(x) + ihf'(x) - h^2\frac{f''(x)}{2!} - ih^3\frac{f'''(x)}{3!} + \cdots$$

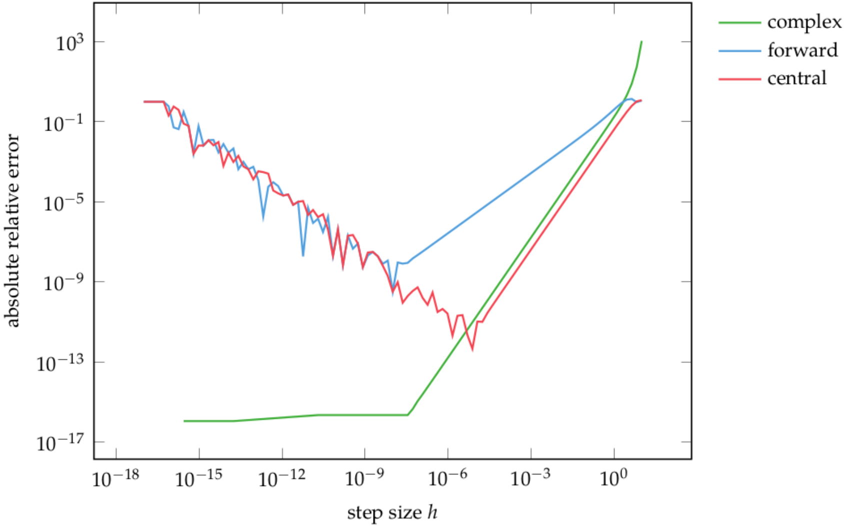
$$\begin{aligned}
\mathrm{Im}(f(x+ih)) &= hf'(x) - h^3\frac{f'''(x)}{3!} + \cdots \\
\Rightarrow f'(x) &= \frac{\mathrm{Im}(f(x+ih))}{h} + h^2\frac{f'''(x)}{3!} - \cdots \\
&= \frac{\mathrm{Im}(f(x+ih))}{h} + O(h^2) \text{ as } h \to 0
\end{aligned}$$

$$\begin{aligned}
\mathrm{Re}(f(x+ih)) &= f(x) - h^2\frac{f''(x)}{2!} + \ldots \\
\Rightarrow f(x) &= \mathrm{Re}(f(x+ih)) + h^2\frac{f''(x)}{2!} - \cdots
\end{aligned}$$

```python
import numpy as np

def diff_complex(f, x: float, h: float=1e-20) -> float:
    return np.imag(f(x + h * 1j)) / h

# Example usage
def func(x):
    return x**2 + np.sin(x)

x0 = 1.0
print(f"The derivative at x = {x0} is {diff_complex(func, x0)}")
```

complex_diff.py

Numerical Differentiation Error Comparison



Outline

Automatic Differentiation

Evaluate a function and compute partial derivatives simultaneously using the **chain rule** of differentiation

$$\frac{\mathrm{d}}{\mathrm{d}x} f(g(x)) = \frac{\mathrm{d}}{\mathrm{d}x} f \circ g(x) = \frac{\mathrm{d}f}{\mathrm{d}g} \frac{\mathrm{d}g}{\mathrm{d}x}$$

- Forward Accumulation is equivalent to expanding a function using the chain rule and computing the derivatives inside-out
- Requires n-passes to compute n-dimensional gradient
- Example:

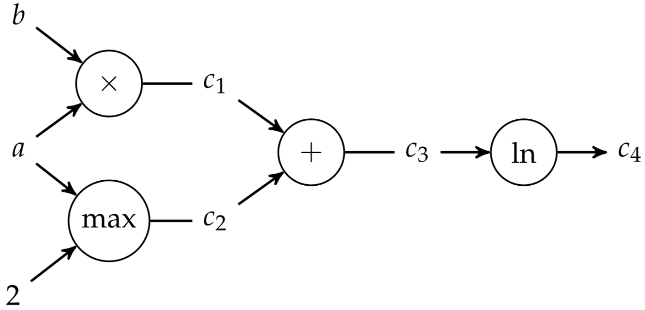
$$f(a,b) = \ln(ab + \max(a,2))$$

$$\begin{aligned}
\frac{\partial f}{\partial a} &= \frac{\partial}{\partial a} \ln(ab + \max(a,2)) \\
&= \frac{1}{ab + \max(a,2)} \frac{\partial}{\partial a}(ab + \max(a,2)) \\
&= \frac{1}{ab + \max(a,2)} \left[\frac{\partial(ab)}{\partial a} + \frac{\partial \max(a,2)}{\partial a} \right] \\
&= \frac{1}{ab + \max(a,2)} \left[\left(b\frac{\partial a}{\partial a} + a\frac{\partial b}{\partial a} \right) + \left((2 > a)\frac{\partial 2}{\partial a} + (2 < a)\frac{\partial a}{\partial a} \right) \right] \\
&= \frac{1}{ab + \max(a,2)} [b + (2 < a)]
\end{aligned}$$

Automatic Differentiation

Computational graph: nodes are are operations and the edges are input-output relations. leaf nodes of a computational graph are input variables or constants, and terminal nodes are values output by the function

Forward accumulation for $f(a, b) = \ln(ab + \max(a, 2))$



$b = 2$

$\dot{b} = 0$

Dual numbers

- Dual numbers can be expressed mathematically by including the abstract quantity ϵ, where ϵ^2 is defined to be 0.
- Like a complex number, a dual number is written $a + b\epsilon$ where a and b are both real values.
- $(a + b\epsilon) + (c + d\epsilon) = (a + c) + (b + d)\epsilon$
  $(a + b\epsilon) \times (c + d\epsilon) = (ac) + (ad + bc)\epsilon$
- by passing a dual number into any smooth function f, we get the evaluation and its derivative. We can show this using the Taylor series:

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k$$

$$f(a+b\epsilon) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(a+b\epsilon-a)^k$$

$$= \sum_{k=0}^{\infty} \frac{f^{(k)}(a)b^k\epsilon^k}{k!}$$

$$= f(a) + bf'(a)\epsilon + \epsilon^2 \sum_{k=2}^{\infty} \frac{f^{(k)}(a)b^k}{k!}\epsilon^{(k-2)}$$

$$= f(a) + bf'(a)\epsilon$$

Automatic Differentiation

- **Reverse accumulation** is performed in single run using two passes over an n-dimensional function (forward and back)
- implemented through two different operation overloading functions (for forward and backward)
- Note: this is central to the backpropagation algorithm used to train neural networks
- Many open-source software implementations are available: eg, Tensorflow

$$\frac{df}{dx} = \frac{df}{dc_4}\frac{dc_4}{dx} = \frac{df}{dc_4}\left(\frac{dc_4}{dc_3}\frac{dc_3}{dx}\right) = \frac{df}{dc_4}\left(\frac{dc_4}{dc_3}\left(\frac{dc_3}{dc_2}\frac{dc_2}{dx} + \frac{dc_3}{dc_1}\frac{dc_1}{dx}\right)\right)$$

$$\frac{df}{dx} = \frac{df}{dc_4}\frac{dc_4}{dx} = \left(\frac{df}{dc_3}\frac{dc_3}{dc_4}\right)\frac{dc_4}{dx} = \left(\left(\frac{df}{dc_2}\frac{dc_2}{dc_3} + \frac{df}{dc_1}\frac{dc_1}{dc_3}\right)\frac{dc_3}{dc_4}\right)\frac{dc_4}{dx}$$

Summary

- Derivatives are useful in optimization because they provide information about how to change a given point in order to improve the objective function
- For multivariate functions, various derivative-based concepts are useful for directing the search for an optimum, including the gradient, the Hessian, and the directional derivative
- One approach to numerical differentiation includes finite difference approximations
- Complex step method can eliminate the effect of subtractive cancellation error when taking small steps
- Analytic differentiation methods include forward and reverse accumulation on computational graphs