09. Calculating derivatives numerically

Summary of last lecture

- Solving equations in higher dimensions
- Vector functions $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$

- Newton method
- Jacobian matrix (first partial derivatives)

- We need to be able to calculate:
  - (Partial) derivatives
  - Solution of a linear system

Goals for today

- Calculating derivatives numerically
- Finite differences (approximate)
- Algorithmic differentiation (“exact”)

Definition of derivative

- Let's recall the **definition** of the **derivative** of $f : \mathbb{R} \to \mathbb{R}$:

$$f'(a) := \lim_{h \to 0} \left[\frac{f(a+h) - f(a)}{h} \right]$$

Definition of derivative

- Let's recall the **definition** of the **derivative** of $f : \mathbb{R} \to \mathbb{R}$:

$$f'(a) := \lim_{h \to 0} \left[\frac{f(a+h) - f(a)}{h} \right]$$

- Geometrically: Slope of a secant line starting at $(a, f(a))$
- Limit as we move the other end closer to a

Definition of derivative

- Let's recall the **definition** of the **derivative** of $f : \mathbb{R} \to \mathbb{R}$:

$$f'(a) := \lim_{h \to 0} \left[\frac{f(a+h) - f(a)}{h} \right]$$

- Geometrically: Slope of a secant line starting at $(a, f(a))$
- Limit as we move the other end closer to a

- How could we calculate this *numerically*?

Definition of derivative

- Let's recall the **definition** of the **derivative** of $f : \mathbb{R} \to \mathbb{R}$:

$$f'(a) := \lim_{h \to 0} \left[\frac{f(a+h) - f(a)}{h} \right]$$

- Geometrically: Slope of a secant line starting at $(a, f(a))$
- Limit as we move the other end closer to a

- How could we calculate this *numerically*?
- The difficulty is the limit

Collaboration I

Calculating derivatives numerically

1. What could we do with the limit on the computer?
2. Can you estimate the error when you do so?

Finite differences

- The simplest numerical method is: just **ignore the limit**!:

$$f'(a) \simeq \frac{f(a+h) - f(a)}{h}$$

- Calculate for some fixed, small value of h

Finite differences

- The simplest numerical method is: just **ignore the limit**!:

$$f'(a) \simeq \frac{f(a+h) - f(a)}{h}$$

- Calculate for some fixed, small value of h

- Known as a **finite difference quotient**
- Here, “finite” is as opposed to “infinitesimal” in the definition of a derivative

Finite differences II

- Immediate questions arise:

Finite differences II

- Immediate questions arise:

- Is the approximation any good?
- How large is the error compared to the true value of $f'(a)$?
- How can we get a better approximation?

Estimating the error

- Let's rearrange the equation:

$$f(a+h) \simeq f(a) + hf'(a)$$

Estimating the error

- Let's rearrange the equation:

$$f(a+h) \simeq f(a) + hf'(a)$$

- As usual, $f(a+h)$ suggests a **Taylor expansion**:

$$f(a+h) = f(a) + hf'(a) + R_1(h)$$

Estimating the error

- Let's rearrange the equation:

$$f(a+h) \simeq f(a) + hf'(a)$$

- As usual, $f(a+h)$ suggests a **Taylor expansion**:

$$f(a+h) = f(a) + hf'(a) + R_1(h)$$

- Lagrange remainder: $R_1(h) = \frac{1}{2}h^2 f''(\xi)$

  for some $\xi \in [a, a+h]$

Estimating the error II

- Truncation error $R_1(h) = \frac{1}{2}h^2 f''(\xi)$
- So $|R_1(h)| \leq Mh^2$ if f'' is **bounded** on $[a, a+h]$

Estimating the error II

- Truncation error $R_1(h) = \frac{1}{2}h^2 f''(\xi)$
- So $|R_1(h)| \leq Mh^2$ if f'' is **bounded** on $[a, a+h]$
- Then $f(a+h) = f(a) + hf'(a) + \mathcal{O}(h^2)$

Estimating the error II

- Truncation error $R_1(h) = \frac{1}{2}h^2 f''(\xi)$
- So $|R_1(h)| \leq Mh^2$ if f'' is **bounded** on $[a, a+h]$
- Then $f(a+h) = f(a) + hf'(a) + \mathcal{O}(h^2)$

$$f'(a) = \frac{f(a+h) - f(a) + \mathcal{O}(h^2)}{h} \tag{1}$$

$$= \frac{f(a+h) - f(a)}{h} + \mathcal{O}(h) \tag{2}$$

Estimating the error II

- Truncation error $R_1(h) = \frac{1}{2}h^2 f''(\xi)$
- So $|R_1(h)| \leq Mh^2$ if f'' is **bounded** on $[a, a+h]$
- Then $f(a+h) = f(a) + hf'(a) + \mathcal{O}(h^2)$

$$f'(a) = \frac{f(a+h) - f(a) + \mathcal{O}(h^2)}{h} \tag{1}$$

$$= \frac{f(a+h) - f(a)}{h} + \mathcal{O}(h) \tag{2}$$

- The truncation error is $\mathcal{O}(h)$ ("big O of h")
- We say that the method is **order 1**

Collaboration II

Better finite-difference approximations

1. Find an approximation for the derivative to $\mathcal{O}(h^2)$
2. What about an approximation for the *second* derivative?

Higher-order finite difference approximations

- Write out the Taylor series:

$$f(a+h) = f(a) + hf'(a) + \tfrac{1}{2!}h^2 f''(a) + \tfrac{1}{3!}h^3 f'''(a) + \cdots$$

Higher-order finite difference approximations

- Write out the Taylor series:

$$f(a+h) = f(a) + hf'(a) + \tfrac{1}{2!}h^2 f''(a) + \tfrac{1}{3!}h^3 f'''(a) + \cdots$$

$$f(a-h) = f(a) - hf'(a) + \tfrac{1}{2!}h^2 f''(a) - \tfrac{1}{3!}h^3 f'''(a) + \cdots$$

Higher-order finite difference approximations

- Write out the Taylor series:

$$f(a+h) = f(a) + hf'(a) + \tfrac{1}{2!}h^2 f''(a) + \tfrac{1}{3!}h^3 f'''(a) + \cdots$$

$$f(a-h) = f(a) - hf'(a) + \tfrac{1}{2!}h^2 f''(a) - \tfrac{1}{3!}h^3 f'''(a) + \cdots$$

- Subtract the two:
- $f(a+h) - f(a-h) = 2hf'(a) + \mathcal{O}(h^3)$

Higher-order finite difference approximations

- Write out the Taylor series:

$$f(a+h) = f(a) + hf'(a) + \tfrac{1}{2!}h^2 f''(a) + \tfrac{1}{3!}h^3 f'''(a) + \cdots$$

$$f(a-h) = f(a) - hf'(a) + \tfrac{1}{2!}h^2 f''(a) - \tfrac{1}{3!}h^3 f'''(a) + \cdots$$

- Subtract the two:
- $f(a+h) - f(a-h) = 2hf'(a) + \mathcal{O}(h^3)$
- Get **centered difference** approximation:

$$f'(a) = \frac{f(a+h) - f(a-h)}{2h} + \mathcal{O}(h^2)$$

Algorithmic (automatic) differentiation

Derivatives as rules

- Think about differentiating
  $f(x) = \sin(x + (x + 1)(x^2 + 2))$

Derivatives as rules

- Think about differentiating
  $f(x) = \sin(x + (x + 1)(x^2 + 2))$
- We apply **rules** for derivatives of sums, products, compositions

Derivatives as rules

- Think about differentiating
  $f(x) = \sin(x + (x + 1)(x^2 + 2))$
- We apply **rules** for derivatives of sums, products, compositions
- I.e. we apply an **algorithm**
- Can we get the computer to execute that for us?

Derivatives as rules

- Think about differentiating
  $f(x) = \sin(x + (x + 1)(x^2 + 2))$
- We apply **rules** for derivatives of sums, products, compositions
- I.e. we apply an **algorithm**
- Can we get the computer to execute that for us?
- **Forward-mode algorithmic (automatic) differentiation**

Alternative viewpoint: Little-o notation

- Let's rewrite the derivative definition by hiding the limit:

$$\lim_{h \to 0} \left[\frac{f(a+h) - f(a)}{h} - f'(a) \right] = 0$$

Alternative viewpoint: Little-o notation

- Let's rewrite the derivative definition by hiding the limit:

$$\lim_{h \to 0} \left[\frac{f(a+h) - f(a)}{h} - f'(a) \right] = 0$$

- So $f(a+h) = f(a) + hf'(a) + o(h)$

  where $\frac{o(h)}{h} \to 0$ when $h \to 0$

Alternative viewpoint: Little-o notation

- Let's rewrite the derivative definition by hiding the limit:

$$\lim_{h \to 0} \left[\frac{f(a+h) - f(a)}{h} - f'(a) \right] = 0$$

- So $f(a+h) = f(a) + hf'(a) + o(h)$

  where $\frac{o(h)}{h} \to 0$ when $h \to 0$

- $o(h)$: "some function that goes to 0 *faster* than h"
- [Really: The equivalence class of *all* such functions]
- "Little-o notation"

Derivative from $f(a + b\epsilon)$

- If f is differentiable at a then
  $f(a + b\epsilon) = f(a) + b\epsilon.f'(a) + o(\epsilon)$

Derivative from $f(a + b\epsilon)$

- If f is differentiable at a then
  $f(a + b\epsilon) = f(a) + b\epsilon.f'(a) + o(\epsilon)$
- The **converse** is also true:

  Suppose $f(a + \epsilon) = c + d\epsilon + o(\epsilon)$
  Then $c = f(a)$ *and* $d = f'(a)$

Derivative from $f(a + b\epsilon)$

- If f is differentiable at a then
  $f(a + b\epsilon) = f(a) + b\epsilon.f'(a) + o(\epsilon)$
- The **converse** is also true:

  Suppose $f(a + \epsilon) = c + d\epsilon + o(\epsilon)$
  Then $c = f(a)$ *and* $d = f'(a)$
- We can *use* this: compute $f(a + 1.\epsilon)$ to first order in ϵ
- Extract the derivative $f'(a)$ as *the coefficient of* ϵ !

Derivative from $f(a + b\epsilon)$

- If f is differentiable at a then
  $f(a + b\epsilon) = f(a) + b\epsilon.f'(a) + o(\epsilon)$
- The **converse** is also true:

  Suppose $f(a + \epsilon) = c + d\epsilon + o(\epsilon)$
  Then $c = f(a)$ *and* $d = f'(a)$
- We can *use* this: compute $f(a + 1.\epsilon)$ to first order in ϵ
- Extract the derivative $f'(a)$ as *the coefficient of* ϵ !

- Equivalent to algebra with $\epsilon^2 = 0$ – an "infinitesimal"
- Or manipulating degree-1 (Taylor) polynomials in ϵ

Differentiation using dual numbers

- Suppose we have a Julia function like `f(x) = x^2 + 2x`
- How can we differentiate f at $a = 3$?

Differentiation using dual numbers

- Suppose we have a Julia function like `f(x) = x^2 + 2x`
- How can we differentiate f at $a = 3$?

- We use $f(a + b\epsilon) = f(a) + b\epsilon.f'(a)$
- So $f'(a)$ is the ϵ component of $f(a + 1.\epsilon)$!

Differentiation using dual numbers

- Suppose we have a Julia function like `f(x) = x^2 + 2x`
- How can we differentiate f at $a = 3$?

- We use $f(a + b\epsilon) = f(a) + b\epsilon.f'(a)$
- So $f'(a)$ is the ϵ component of $f(a + 1.\epsilon)$!

- $a + b\epsilon$ corresponds to `Dual(a, b)`

Differentiation using dual numbers

- Suppose we have a Julia function like `f(x) = x^2 + 2x`
- How can we differentiate f at $a = 3$?

- We use $f(a + b\epsilon) = f(a) + b\epsilon.f'(a)$
- So $f'(a)$ is the ϵ component of $f(a + 1.\epsilon)$!

- $a + b\epsilon$ corresponds to `Dual(a, b)`
- So define `xx = Dual(a, 1)`
- And call `f(xx)`

Differentiation using dual numbers

- Suppose we have a Julia function like `f(x) = x^2 + 2x`
- How can we differentiate f at $a = 3$?

- We use $f(a + b\epsilon) = f(a) + b\epsilon.f'(a)$
- So $f'(a)$ is the ϵ component of $f(a + 1.\epsilon)$!

- $a + b\epsilon$ corresponds to `Dual(a, b)`
- So define `xx = Dual(a, 1)`
- And call `f(xx)`
- `Dual(a, 1)` represents the identity function $x \mapsto x$ evaluated at $x = a$: the derivative is 1

Decomposing complicated functions

- For a complicated function like $f(y) = y^2 + 2y$, we treat f as a **composition** of known functions
- `f(y) = +( *(y, y), *(2, y) )`

Decomposing complicated functions

- For a complicated function like $f(y) = y^2 + 2y$, we treat f as a **composition** of known functions
- `f(y) = +( *(y, y), *(2, y) )`
- We need to apply rules for derivatives one after another to find $f'(a)$

Sum rule for derivatives

- Let's *derive* some of the rules of derivatives

Sum rule for derivatives

- Let's *derive* some of the rules of derivatives
- Sum of two functions: $(f + g)(x) := f(x) + g(x)$

Sum rule for derivatives

- Let's *derive* some of the rules of derivatives
- Sum of two functions: $(f+g)(x) := f(x) + g(x)$
- We want $(f+g)'(a)$ so calculate $[f+g](a+\epsilon)$:

Sum rule for derivatives

- Let's *derive* some of the rules of derivatives
- Sum of two functions: $(f+g)(x) := f(x) + g(x)$
- We want $(f+g)'(a)$ so calculate $[f+g](a+\epsilon)$:

$$\begin{aligned}[f+g](a+\epsilon) &= f(a+\epsilon) + g(a+\epsilon) \\ &= [f(a) + \epsilon f'(a)] + [g(a) + \epsilon g'(a)] \\ &= [f(a) + g(a)] + [f'(a) + g'(a)]\epsilon\end{aligned}$$

- Hence $(f+g)'(a)$ = (coefficient of ϵ) = $f'(a) + g'(a)$.

Product rule for derivatives

- Product of two functions: $(f \cdot g)(x) := f(x) \cdot g(x)$

  (Here $\cdot$ is normal scalar multiplication)

Product rule for derivatives

- Product of two functions: $(f \cdot g)(x) := f(x) \cdot g(x)$

  (Here $\cdot$ is normal scalar multiplication)

- Calculate $(f \cdot g)(a + \epsilon)$:

$$\begin{aligned}
[f \cdot g](a + \epsilon) &= f(a + \epsilon) \cdot g(a + \epsilon) \\
&= [f(a) + \epsilon f'(a)] \cdot [g(a) + \epsilon g'(a)] \\
&= [f(a) \cdot g(a)] + [f(a)g'(a) + g(a)f'(a)]\epsilon
\end{aligned}$$

- Hence $(f \cdot g)'(a) = f(a)g'(a) + g(a)f'(a)$.

Information used for algorithmic differentiation

- For each simple function in a complicated function we need some information
- How much information do we need for each function?

Information used for algorithmic differentiation

- For each simple function in a complicated function we need some information
- How much information do we need for each function?
- We need exactly *two* pieces of information for each f:
  - the value $f(a)$
  - the derivative $f'(a)$

Information used for algorithmic differentiation

- For each simple function in a complicated function we need some information
- How much information do we need for each function?
- We need exactly *two* pieces of information for each f:
  - the value $f(a)$
  - the derivative $f'(a)$
- So we represent f as the pair $f \rightsquigarrow (f(a), f'(a))$

Representation in Julia

- We need to group together 2 pieces of information

Representation in Julia

- We need to group together 2 pieces of information
- And we want novel **behaviour**, i.e. rules for $+$ and $\times$

Representation in Julia

- We need to group together 2 pieces of information
- And we want novel **behaviour**, i.e. rules for $+$ and $\times$

- So we *define a new type*
- Commonly called a **dual number**

Dual number type in Julia

- We make a new `struct`

```
struct Dual
    value::Float64
    deriv::Float64
end
```

Dual number type in Julia

- We make a new `struct`

```
struct Dual
    value::Float64
    deriv::Float64
end
```

- Recall: Composite type $\equiv$ template for box containing data

Dual number type in Julia

- We make a new `struct`

```
struct Dual
    value::Float64
    deriv::Float64
end
```

- Recall: Composite type $\equiv$ template for box containing data

- **`Dual(c, d)` corresponds directly to** $c + \epsilon d$
- It represents a function f with $f(a) = c$ and $f'(a) = d$
- Note that the point a where we are evaluating is *implicit*

Implementing arithmetic

- To implement arithmetic, import relevant functions:

```
import Base: +, *
```

- Add methods acting on objects of type `Dual`:

```
+(f::Dual, g::Dual) = Dual(f.value + g.value,
                           f.deriv + g.deriv)
```

Higher-dimensional functions: Directional derivative

- What about derivatives of $f : \mathbb{R}^n \to \mathbb{R}$?

Higher-dimensional functions: Directional derivative

- What about derivatives of $f : \mathbb{R}^n \to \mathbb{R}$?

- The **directional derivative** of f in the direction **v** is

$$\partial_{\mathbf{v}} f(\mathbf{a}) := \lim_{\epsilon \to 0} \left[\frac{f(\mathbf{a} + \epsilon \mathbf{v}) - f(\mathbf{a})}{\epsilon} \right]$$

Higher-dimensional functions: Directional derivative

- What about derivatives of $f : \mathbb{R}^n \to \mathbb{R}$?
- The **directional derivative** of f in the direction **v** is

$$\partial_{\mathbf{v}} f(\mathbf{a}) := \lim_{\epsilon \to 0} \left[\frac{f(\mathbf{a} + \epsilon \mathbf{v}) - f(\mathbf{a})}{\epsilon} \right]$$

- Define $g(\epsilon) := f(\mathbf{a} + \epsilon \mathbf{v}) - f(\mathbf{a})$
- $\partial_{\mathbf{v}} f(\mathbf{a})$ is the ϵ coefficient of $g(\epsilon)$!

Directional derivative II

- In 2D: $g(\epsilon) := f(a + v_1\,\epsilon, b + v_2\,\epsilon)$ with the *same* ϵ

Directional derivative II

- In 2D: $g(\epsilon) := f(a + v_1\,\epsilon, b + v_2\,\epsilon)$ with the *same* ϵ
- Taylor:

$$g(\epsilon) = f(a,b) + v_1\,\epsilon\,\frac{\partial f}{\partial x}(a,b) + v_2\,\epsilon\,\frac{\partial f}{\partial y}(a,b)$$

- Coefficient of ϵ is

$$v_1\,\frac{\partial f}{\partial x}(a,b) + v_2\,\frac{\partial f}{\partial y}(a,b) = \nabla f(a,b)\cdot\mathbf{v}$$

Directional derivative II

- In 2D: $g(\epsilon) := f(a + v_1\,\epsilon, b + v_2\,\epsilon)$ with the *same* ϵ
- Taylor:

$$g(\epsilon) = f(a,b) + v_1\,\epsilon\,\frac{\partial f}{\partial x}(a,b) + v_2\,\epsilon\,\frac{\partial f}{\partial y}(a,b)$$

- Coefficient of ϵ is

$$v_1\,\frac{\partial f}{\partial x}(a,b) + v_2\,\frac{\partial f}{\partial y}(a,b) = \nabla f(a,b)\cdot\mathbf{v}$$

- How can we *calculate* this?

Calculating directional derivatives

- Let the vector **v** be `(v₁, v₂)`
- Then

  `f(Dual(a, v₁), Dual(b, v₂))`

  calculates $\partial_{\mathbf{v}} f(\mathbf{a}) = \nabla f(a, b) \cdot \mathbf{v}$!

Calculating directional derivatives

- Let the vector **v** be `(v₁, v₂)`
- Then

  `f(Dual(a, v₁), Dual(b, v₂))`

  calculates $\partial_{\mathbf{v}} f(\mathbf{a}) = \nabla f(a, b) \cdot \mathbf{v}$!

- $\mathbf{v} = (1, 0)$ gives $\partial f / \partial x$
- $\mathbf{v} = (0, 1)$ gives $\partial f / \partial y$

Jacobian

- For $f : \mathbb{R}^2 \to \mathbb{R}^2$, have $f = (f_1, f_2)$ with

$$f_i(\mathbf{a} + \epsilon\mathbf{v}) = f_i(\mathbf{a}) + \epsilon \nabla f_i(\mathbf{a}) \cdot \mathbf{v}$$

- So

$$f(\mathbf{a} + \epsilon\mathbf{v}) = f(\mathbf{a}) + \epsilon \, \mathrm{D}f(\mathbf{a}) \cdot \mathbf{v}$$

- $\mathrm{D}f(\mathbf{a})$ is **Jacobian matrix** of partial derivatives $\frac{\partial f_i}{\partial x_j}$
- Coefficient of ϵ is Jacobian–vector product

Summary

- Finite-difference approximations for derivatives
- Order of truncation error using Taylor expansions

- We can implement rules to calculate derivatives
- By defining a dual number type and overloading operations
- Extends directly to derivatives of higher-dimensional functions