Numerical Methods for Integration and Differentiation

• Outline:
  • Motivation
  • Revisiting Taylor series
  • Calculating derivatives numerically
  • Calculating integrals numerically
    - Square and Trapezoid methods
    - Recursive Trapezoid and Romberg Algorithm
Motivation

- Taylor series ... important concept in numerical approximation, used in many algorithms, so need to be familiar with it.
- Same with numerical differentiation, e.g., needed for
  - Optimization algorithms
  - Finding roots of (non-linear) equations
  - Numerical integration of differential equations
- Analytical calculations often result in integrals that cannot be computed analytically
  - Numerical integration schemes
Taylor Series

- Given a smooth function \( f(x) \), we can expand it around a point \( c \)

\[
f(x) = f(c) + f'(c)(x-c) + \frac{1}{2!} f''(c)(x-c)^2 + \frac{1}{3!} f'''(x-c)^3 + \ldots
\]

\[
= \sum_{k=0}^{\infty} \frac{1}{k!} f^{(k)}(c)(x-c)^k
\]

- This is the Taylor series of \( f \) at point \( c \). Loosely: “if \( x \) is close to \( c \) the series converges rapidly and slowly (or not at all) if \( x \) is far away from \( c \)”

- Convergence:

\[
E_{n+1} = f(x) - f_n(x) = \sum_{k=n+1}^{\infty} \frac{f^{(k)}(c)(x-c)^k}{k!} = \frac{f^{(n+1)}(\zeta)(x-c)^{n+1}}{(n+1)!}
\]

with \( \zeta \) some value between \( x \) and \( c \)
Some familiar examples

\[ e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots, \quad |x| < \infty \]

\[ \sin(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k+1}}{(2k+1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \ldots, \quad |x| < \infty \]

\[ \cos(x) = \sum_{k=0}^{\infty} (-1)^k \frac{x^{2k}}{(2k)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots, \quad |x| < \infty \]

\[ \frac{1}{1-x} = \sum_{k=0}^{\infty} x^k = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots, \quad |x| < 1 \]

- Computers calculate many functions like this, e.g.

\[ e^x \approx \sum_{k=0}^{N} \frac{x^k}{k!} \quad \text{for some large N.} \]
Example

- Calculate $e^1$ to 6 digit accuracy

- Answer:

$$e = e^1 = 1 + 1 + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \frac{1}{5!} + \ldots$$

$$\frac{1}{2!} = 0.5, \quad \frac{1}{3!} = 0.166667, \quad \frac{1}{4!} = 0.041667, \ldots, \quad \frac{1}{9!} = 0.0000027$$

$$\rightarrow \quad e \approx 1 + 1 + \frac{1}{2!} + \ldots + \frac{1}{9!} = 2.71828$$

next one gives corrections $< 0.000001$
Numerical Differentiation

- Want to have a numerical approximation of $f'(x)$, $f''(x)$, etc.
- Reminder: $f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$
- If we set $n=0$ in Taylor's theorem we have the "Mean-value theorem":
  \[ f(a) - f(b) = (b-a)f'(\xi) \]
  \[ f'(\xi) = \frac{f(b) - f(a)}{b-a} \]
- Can use this to approximate $f'$ if interval $(a,b)$ is small
Finite Difference Schemes

\[ f'(x) \approx \frac{1}{h} (f(x+h) - f(x)) \rightarrow \text{Forward difference (3)} \]

\[ f'(x) \approx \frac{1}{h} (f(x) - f(x-h)) \rightarrow \text{Backward difference (2)} \]

\[ f'(x) \approx \frac{1}{2h} (f(x+h) - f(x-h)) \rightarrow \text{Central difference (1)} \]
What about the errors?

- From Taylor:

\[
\begin{align*}
  f(x+h) &= f(x) + hf'(x) + \frac{1}{2} h^2 f''(x) + \frac{1}{6} h^3 f'''(x) + O(h^4) \\
  f(x-h) &= f(x) - hf'(x) + \frac{1}{2} h^2 f''(x) - \frac{1}{6} h^3 f'''(x) + O(h^4)
\end{align*}
\]

\[
\begin{align*}
  \frac{1}{h} (f(x+h) - f(x)) &= f'(x) + \frac{1}{2} h f''(x) + \ldots \\
  \frac{1}{h} (f(x) - f(x-h)) &= f'(x) - \frac{1}{2} h f''(x) + \ldots
\end{align*}
\]

... both first order in h (error prop. to h)

\[
\begin{align*}
  \frac{1}{2h} (f(x+h) - f(x-h)) &= f'(x) + \frac{1}{6} h^2 f'''(x) + \ldots
\end{align*}
\]

... second order in h (i.e. more accurate)
How to calculate \( f'' \)?

- Could just calculate the derivative of the first derivative ...
- Better:

\[
\begin{align*}
  f(x + h) &= f(x) + hf'(x) + \frac{1}{2}h^2 f''(x) + \frac{1}{6}h^3 f'''(x) + \frac{1}{24}h^4 f^{(4)}(x) + ... \\
  f(x - h) &= f(x) - hf'(x) + \frac{1}{2}h^2 f''(x) - \frac{1}{6}h^3 f'''(x) + \frac{1}{24}h^4 f^{(4)}(x) + ...
\end{align*}
\]

\[\longrightarrow \quad f''(x) + \frac{1}{12}h^2 f^{(4)} = \frac{1}{h^2} \left( f(x + h) - 2f(x) + f(x - h) \right)\]

\[f''(x) \approx \frac{1}{h^2} \left( f(x + h) - 2f(x) + f(x - h) \right)\]

... second order in \( h \)
Determining Truncation Errors

- Why bother? We have fast computers and can use small values of h, so no problem?
  - Problems with number representations in computers (more later)
  - Often these approximations are iterated in numerical schemes, so errors accumulate fast
- Truncation error calculations might be quite confusing, but idea is simple:
  - Compare finite difference approximation to full Taylor approximation, difference between both gives error terms and order of the method.
Multivariate Taylor

- A few times later in the module we will have to deal with expansions of scalar functions of multiple variables $f(x,y,z,...)$
- To write this in nice form, introduce multi-indices
  \[\alpha \in \mathbb{N}^n, \alpha = (\alpha_1, \ldots, \alpha_n), |\alpha| = \sum_{j=1}^{n} \alpha_j\]
  \[\alpha! = \alpha_1! \cdot \alpha_2! \cdot \ldots \cdot \alpha_n! \quad \vec{x}^\alpha = x_1^{\alpha_1} \cdot x_2^{\alpha_2} \cdot \ldots \cdot x_n^{\alpha_n}\]
  \[D^\alpha f = \frac{\partial^{|\alpha|} f}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \ldots \partial x_n^{\alpha_n}}\]
- Then:
  \[f(\vec{x} + \vec{h}) = \lim_{k \to \infty} \sum_{|\alpha| \leq k} \frac{1}{|\alpha|!} D^\alpha f(\vec{x}) \vec{h}^\alpha\]
Example

- Taylor
  \[ f(\vec{x}+\vec{h}) = \lim_{k \to \infty} \sum_{|\alpha| \leq k} \frac{1}{|\alpha|!} D^\alpha f(\vec{x}) \vec{h}^\alpha \]

- Expand \( f(x,y) \) up to second order

\[
\begin{align*}
  f(x+h_1, y+h_2) &\approx f(x, y) + \partial_x f h_1 + \partial_y f h_2 + \frac{1}{2!} \partial_x^2 f h_1^2 + \partial_x \partial_y h_1 h_2 + \frac{1}{2!} \partial_y^2 h_2^2 \\
\end{align*}
\]

- Later in the module mostly the linear term will matter ...
Numerical Integration

- Given $f(x)$ in the interval $[a,b]$ we want to find an approximation for

\[ I(f) = \int_{a}^{b} f(x) \, dx \]

- Main strategy:
  - Cut $[a,b]$ into smaller sub-intervals
  - In each interval $i$, approximate $f(x)$ by a polynomial $p_i$
  - Integrate the polynomials analytically and sum up their contributions
First Idea

- Approximate $f$ by a constant for each sub-interval $\rightarrow p_i(x) = f(x_i) = f_i$
First Idea

- Each interval \( \int_{x_i}^{x_{i+1}} p_i(x) \, dx = \int_{x_i}^{x_{i+1}} f_i \, dx = f_i(x_{i+1} - x_i) \)

- Adding and using equidistant intervals

\[
\int_{a}^{b} f(x) \, dx \approx \sum_{i=0}^{n-1} f_i(x_{i+1} - x_i)
\]

\[
h = \frac{b-a}{n}, \quad x_i = x_0 + (i-1)h \quad \rightarrow \quad \int_{a}^{b} f(x) \, dx \approx h \sum_{i=0}^{n-1} f_i
\]

- Error (roughly):
  - Approximation error per for f is \( \sim h \)
  - After integration \( \sim h^2 \)
  - Summing up order \( 1/h \) intervals: \( E \sim h \)
  - Maybe we can do better without much more computational effort?
Trapezoid Rule

- On interval \([x_i, x_{i+1}]\) interpolate \(f(x)\) by a linear polynomial that connects end points: \(f(x_i) = p_i(x_i)\), \(f(x_{i+1}) = p_i(x_{i+1})\)

\[
p_i(x) = f(x_i) + (x - x_i) \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}
\]
Can now integrate the p(x)'s

$$\int_{x_i}^{x_{i+1}} p_i(x) \, dx = \left[ f(x_i) x - x_i x \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} + x^2 / 2 \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \right]_{x_i}^{x_{i+1}}$$

... 

$$= \frac{1}{2} (f(x_{i+1}) + f(x_i))(x_{i+1} - x_i)$$

which is just the area of the trapezium ...
Trapezoid (3)

- Adding up all intervals

\[ \int_{a}^{b} f(x) \, dx \approx \sum_{i=0}^{n-1} \frac{1}{2} (f(x_{i+1}) + f(x_i)) (x_{i+1} - x_i) \]

- Using equidistant intervals

\[
h = \frac{b-a}{n}, \quad x_i = x_0 + (i-1)h
\]

\[
\int_{a}^{b} f(x) \, dx \approx \sum_{i=0}^{n-1} \frac{h}{2} (f(x_{i+1}) + f(x_i))
\]

\[
\int_{a}^{b} f(x) \, dx \approx h \left[ \frac{1}{2} f(x_0) + \sum_{i=1}^{n-1} f(x_i) + \frac{1}{2} f(x_n) \right] := T(f ; h)
\]
Error Estimate

- Only approximate functions over the intervals, hence there is a systematic truncation error

\[ E(f;h) = I(f) - T(f;h) = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} [f(x) - p^i(x)] \]

\[ = \sum_{i=0}^{n-1} E_i(f;h) \]

- Error from linear approximation of f per interval \( \propto h^2 \)
- Integration -> \( E_i \propto h^3 \)
- Summing up order of 1/h E's -> \( E \propto h^2 \)
Simpson's Rule

- So far, approximated $f$ by
  - A constant $\rightarrow E \sim h$
  - A linear function $\rightarrow E \sim h^2$
  - Maybe try a quadratic function $\rightarrow$ Simpson's rule
Simpson's Rule (2)

- Won't go into any technical details about this, but for equidistant intervals one has

\[ I \approx S(f; h) = \frac{h}{3} \sum_{i=0}^{n-1} [f(x_{2i}) + 4f(x_{2i+1}) + f(x_{2i+2})] \]

- One can show that Simpson's rule is 4\textsuperscript{th} order, i.e. \( E \propto h^4 \)
Recursive Trapezoid

- A way to control truncation error without unnecessary computations

- Divide $[a,b]$ into $2^n$ sub-intervals and evaluate Trapezoid rule for $h_n=2^{-n}(b-a)$ and $h_{n+1}=h_n/2$

- Apply Trapezoid rule
Recursive Trapezoid (2)

- For $h_n$: \[ T(f; h_n) = h_n \left[ \frac{f(a) + f(b)}{2} + \sum_{i=1}^{2^n-1} f(a + ih_n) \right] \]

- For $h_{n+1}$: \[ T(f; h_{n+1}) = h_{n+1} \left[ \frac{f(a) + f(b)}{2} + \sum_{i=1}^{2^{n+1}-1} f(a + ih_{n+1}) \right] \]

- One can show ...
  \[ T(f; h_{n+1}) = \frac{1}{2} T(f; h_n) + h_{n+1} \sum_{j=0}^{2^n-1} f(a + (2j+1)h_{n+1}) \]

- Advantages
  - Keep computation at level $n$. If not accurate enough -> add another level
  - Don't need to re-evaluate at points we have already evaluated before
Romberg Method

• Idea:
  • Say we have calculated $T(f;h)$, $T(f;h/2)$, $T(f;h/4)$, ...
  • Combine these numbers to get better approximation?

• One can show that

\[
E(f ; h) = I(f) - T(f ; h) = a_2 h^2 + a_4 h^4 + a_6 h^6 + ... \quad (1)
\]

only depends on even powers of $h$

\[
E(f ; h/2) = I(f) - T(f ; h/2) = a_2 (h/2)^2 + a_4 (h/2)^4 + a_6 (h/2)^6 + ... \quad (2)
\]
Re-arranging for $I$ yields:

$$I(f) = T(f; h) + a_2 h^2 + a_4 h^4 + a_6 h^6 + ...$$ (1')

$$I(f) = T(f; h/2) + a_2 (h/2)^2 + a_4 (h/2)^4 + a_6 (h/2)^6 + ...$$ (2')

Multiplying (2') by 4 and subtracting (1'):

$$3 I(f) = 4 T(f; h/2) - T(f; h) + a'_4 h^4 + ...$$

$$I(f) = \frac{4}{3} T(f; h/2) - T(f; h) + a''_4 h^4 + ...$$

$U(h)$

$U(h)$ is fourth order accuracy!

This is called Richardson extrapolation!
Romberg (3)

- Obviously, we can continue with this idea:
  
  \[ I(f) = U(h) + a''_4 h^4 + a''_6 h^6 + ... \]  
  \[ I(f) = U(h/2) + a''_4 (h/2)^4 + a''_6 (h/2)^6 + ... \]  

- To cancel the fourth order term we multiply (4) by \( 2^4 \) and subtract (3)

  \[ V(h) = \frac{2^4 U(h/2) - U(h)}{2^4 - 1} = U(h/2) + \frac{U(h/2) - U(h)}{2^4 - 1} \]

  and \[ I(f) = V(h) + a'''_6 h^6 + ... \]

- And so on ... yields the Romberg Algorithm
Romberg Algorithm

- Set $H = b - a$ and define

$$R(0,0) = T(f; H)$$
$$R(1,0) = T(f; H/2)$$

... 

$$R(1,0) = T(f; H/2^n)$$
(e.g. calculated by recursive Trapezoid)

- Then:

$$R(n,m) = R(n,m-1) + \frac{R(n,m-1) - R(n-1,m-1)}{2^{2m} - 1}$$

- and

$$I(f) = R(m,m) + O(h^{2(m+1)})$$
Romberg Triangle

• Recursive calculation of the $R(n,m)$'s ...
More Ideas …

- “Mesh” should be finer when \( f \) changes a lot
  - \( \rightarrow \) adaptive methods

- Gaussian quadrature:
  - All methods somehow approximate integrals via
    \[
    \int_{a}^{b} f(x) \, dx \approx A_0 f(x_0) + A_1 f(x_1) + \ldots + A_n f(x_n)
    \]
  - Squares \( x_0 = a, A_0 = 1 \)
  - Trapezoid \( x_0 = a, x_1 = b, A_0 = A_1 = 1 \)
  - Could also optimize the \( x_i \) and \( A_i \) to improve precision
  - Gaussian quadrature does this tuning these parameters to get integrals for polynomials of order \( m \) right
Summary

What is important to remember:

- Taylor series and how to use them to estimate truncation errors
- The idea of numerical differentiation
- The idea of basic schemes for numerical integration, say Trapezoid
- More advanced stuff is nice to remember but can be looked up when needed