Updated on April 26, 2022

Exercises: Numerical Calculus

Problems marked with ✎ are to be done by hand; those marked with 💻 are to be solved using a computer.

1. (Deriving the third-order forward difference formula) ✎ Find the third-order forward difference approximation to $f'(x)$, which can be written as

$$D_h^{[3\mathrm{f}]}\{f\}(x) \approx c_1 f(x) + c_2 f(x+h) + c_3 f(x+2h) + c_4 f(x+3h).$$

You may use any one of the approaches presented in lecture[1] or follow the directions found in **LM** 14.1–5.

2. (Another derivation exercise; **LM** 14.1–12)

   (a) ✎ Use the second-order centered difference formula for the first derivative and Richardson extrapolation to obtain a fourth-order centered difference formula.

   (b) 💻 Verify that the formula obtained in part (a) is fourth-order accurate by modifying the script `diff1` on p. 1767 of **LM**.

   (c) ✎ 💻 Repeat the previous parts for the second derivative.

Hint. The second-order centered difference formula for $f''(x)$, with the leading error term, is given by

$$\frac{f(x+h) - 2f(x) + f(x-h)}{h^2} = f''(x) + \frac{1}{12} f''''(x) h^2 + O(h^4).$$

See Lecture 35 or **LM** p. 1766-7. You may use it without derivation for part (c).

3. (Approximating π again; **LM** 14.1–17) Archimedes' algorithm for approximating π calculates the perimeter of the inscribed polygon with n sides, $p_n = n\sin(\pi/n)$, and the circumscribed polygon, $P_n = n\tan(\pi/n)$; see **LM** Section 11.4.1.1. Let $h = 1/n$.

   (a) ✎ Use the Taylor series expansions for $\sin(\pi h)/h$ and $\tan(\pi h)/h$ to show that

$$\begin{aligned} p_n &= \pi + a_1 h^2 + a_2 h^4 + \cdots \\ P_n &= \pi + b_1 h^2 + b_2 h^4 + \cdots , \end{aligned}$$

   where you are to calculate the four coefficients a_1, a_2, b_1, b_2 explicitly.

   (b) ✎ A "better" approximation to π is obtained by averaging the two:

$$\mathfrak{B}_n \equiv \frac{1}{2}(p_n + P_n) = \pi + c_1 h^2 + c_2 h^4 + \cdots ,$$

   Calculate these two coefficients c_1, c_2 explicitly.

[1] Interpolation-based, series-based, or Richardson extrapolation.

(c) Using Richardson extrapolation, find an "even better" approximation, $\mathfrak{R}_n$, to π which is fourth-order accurate, that is, it must satisfy

$$\mathfrak{R}_n = \pi + d_1 h^4 + \cdots ,$$

where you are also to calculate the coefficient d_1 explicitly. (The answer for this part is not unique.)

(d) Archimedes approximated π by letting $n = 96$. Calculate $p_n, P_n, \mathfrak{B}_n$, and $\mathfrak{R}_n$ for $n = 48, 96, 192$. Also print out the error in each.

(e) (Optional) Watch this video by Veratasium [2] which explains a new approximation algorithm suggested by Sir Isaac Newton a couple millennia later, which is based on quadrature (numerical integration, if you like) and the binomial theorem he invented. Write a MATLAB program which implements Newton's idea presented in the video and see how quickly it converges, that is, how many terms of the series is needed to approximate π to full precision on MATLAB?

4. (Variation of Euler spiral; **LM** 14.2–3(b)) Modifying the script[3] generating the Euler spiral, plot the curve

$$x(w) = \int_0^w \cos\left(\frac{1}{4}z^3 - 5.2z\right)\, dz \quad \text{and} \quad y(w) = \int_0^w \sin\left(\frac{1}{4}z^3 - 5.2z\right)\, dz$$

for $w \in [-S, +S]$; use S of your own choice. Use the symmetry to complete the curve.

5. (Smoothness and accuracy of quadrature methods; **LM** 14.2–6) If $f(x)$ is a "smooth" function, the errors in the composite trapezoidal and midpoint methods are $O(h^2)$, and the error in the composite Simpson's method is $O(h^4)$. But what if the function is "not smooth enough"?

   (a) Show numerically that the errors in the composite trapezoidal method, midpoint method, and Simpson's method are all $O(h^{3/2})$ when calculating

   $$I_0 = \int_0^1 \sqrt{x}\, dx.$$

   Generate the table with headers

   ```
   h          err-trap          err-mid          err-simp
   ```

   and show that the errors decrease by a factor of approximately $2^{3/2} \approx 2.8$ when h is halved.

   (b) Repeat the previous part for

   $$I_1 = \int_0^1 x^{3/2}\, dx \quad \text{and} \quad I_2 = \int_0^1 x^{5/2}\, dx.$$

6. (Extrapolation for composite methods; **LM** 14.2–11(a)) Use Richardson extrapolation to derive the composite Simpson's method from the composite trapezoidal method.

[2] If the link above does not work, use `https://youtu.be/gMlf1ELvRzc`.

[3] See the live script posted on Week 15 supplementary resources page. The code in the live script is a solution to **LM** 14.2–3(a).

Hint. Apply the trapezoidal method to the interval $[a, b]$ with subintervals of length h, which we denote by I_h, and then with subintervals of length $h/2$, which we denote by $I_{h/2}$. Take the appropriate linear combination to obtain the composite Simpson's method.

7. (Optimal step size and Jacobian) In lecture, the optimal h for the second-order centered difference formula was shown to be about $\boxed{\text{eps}}^{1/3}$. At this optimal h, the leading error is $O(\boxed{\text{eps}}^{2/3})$. (Why?)

   (a) Determine the optimal h for the first-order forward difference formula by following a similar argument. Also determine the leading error at this optimal h.

   (b) Generalize the argument to determine the optimal h for an m-th order accurate method, where m is any positive integer. Also determine the leading error at this optimal h.

   (c) Complete the following program approximating the Jacobian of $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ using the first-order forward difference using the optimal step size determined in the previous parts.

```
function J = jacfd(f, x0)
% JACFD Approximation of a Jacobian by 1st-order forward difference
% Input:
%   f       function to be differentiated
%           which takes (n-by-1) column vector as an input
%           and produces (m-by-1) column vector as an output
%   x0      evaluation point
% Output:
%   J       approximate Jacobian (m-by-n)

    h = [.....];    % optimal step size






end
```

   (d) Hint. (Tip for vectorization) Recall that

$$\mathbf{J}(\mathbf{x}) = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \dfrac{\partial f_1}{\partial x_2} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \dfrac{\partial f_2}{\partial x_1} & \dfrac{\partial f_2}{\partial x_2} & \cdots & \dfrac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial f_m}{\partial x_1} & \dfrac{\partial f_m}{\partial x_2} & \cdots & \dfrac{\partial f_m}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{m \times n} \tag{1}$$

The jth column of $\mathbf{J}$ consists of all partial derivatives with respect to x_j:

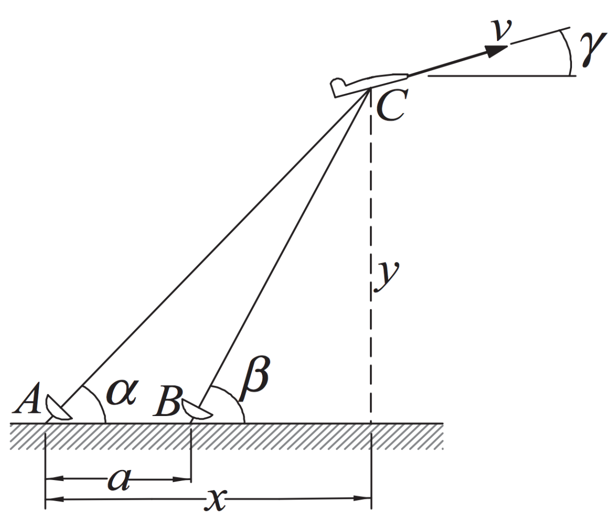
$$\mathbf{J}(\mathbf{x})\mathbf{e}_j = \begin{bmatrix} \frac{\partial f_1}{\partial x_j} \\ \frac{\partial f_2}{\partial x_j} \\ \vdots \\ \frac{\partial f_m}{\partial x_j} \end{bmatrix}$$

This column vector can be approximated by a finite difference formula involving a perturbation only in x_j:

$$\mathbf{J}(\mathbf{x})\mathbf{e}_j \approx \frac{\mathbf{f}(\mathbf{x} + h\mathbf{e}_j) - \mathbf{f}(\mathbf{x})}{h}, \quad j = 1, \ldots, n,$$

where h is optimally chosen according to the previous parts.

8. (Air plane velocity from radar readings) The radar stations A and B, separated by the distance $a = 500$ m, track a plane C by recording the angles α and β at one-second intervals. Your goal, back at air traffic control, is to determine the speed of the plane.



Let the position of the plane at time t be given by $(x(t), y(t))^{\mathrm{T}}$. The speed at time t is the magnitude of the velocity vector,

$$\left\| \frac{d}{dt} \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} \right\| = \sqrt{x'(t)^2 + y'(t)^2}. \tag{2}$$

The closed forms of the functions $x(t)$ and $y(t)$ are unknown (and may not exist at all), but we can still use numerical methods to estimate $x'(t)$ and $y'(t)$. For example, at $t = 3$, the second order centered difference quotient for $x'(t)$ is

$$x'(3) \approx \frac{x(3+h) - x(3-h)}{2h} = \frac{1}{2}(x(4) - x(2)).$$

In this case $h = 1$ since data comes in from the radar stations at 1 second intervals.

Successive readings for α and β at integer times $t = 7, 8, \ldots, 14$ are stored in the file `plane.dat`. Each row in the array represents a different reading; the columns are the observation time t, the angle α (in degrees), and the angle β (also in degrees), in that order. The Cartesian coordinates of the plane can be calculated from the angles α and β as follows:

$$x(\alpha, \beta) = a\frac{\tan(\beta)}{\tan(\beta) - \tan(\alpha)} \quad \text{and} \quad y(\alpha, \beta) = a\frac{\tan(\beta)\tan(\alpha)}{\tan(\beta) - \tan(\alpha)}. \tag{3}$$

(a) Verify the equations in (3).

(b) Load the data, convert α and β to radians[4], then compute the coordinates $x(t)$ and $y(t)$ at each given t using (3). Approximate $x'(t)$ and $y'(t)$ using the second-order forward difference for $t = 7$, the second-order backward difference for $t = 14$, and the second-order centered difference for $t = 8, 9, \ldots, 13$. Return the values of the speed at each t using (2).

9. (Mechanical vibration using Euler-midpoint method) Suppose that the motion of a certain spring-mass system satisfies the differential equation

$$u'' + u' + \frac{1}{5}u^3 = 3\cos\omega t$$

and the initial conditions

$$u(0) = 2, u'(0) = 0.$$

Write a MATLAB program to plot the trajectory $u(t)$ for $0 \leqslant t \leqslant 100$ using the *Euler-midpoint method*.

Note. The **Euler-midpoint method** is an example of second-order Runge-Kutta methods. It was introduced in Lecture 37 along with Euler-trapezoidal and Heun's methods. The Euler-midpoint method for the IVP $\mathbf{y}' = \mathbf{f}(t, \mathbf{y})$, $\mathbf{y}(t_0) = \mathbf{y}_0$ can be written as

$$\mathbf{y}_{n+1} = \mathbf{y}_n + h\mathbf{f}\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}\mathbf{f}(t_n, \mathbf{y}_n)\right).$$

Confirm for yourself that this agrees with what was shown in lecture.

10. (Lorenz model, butterfly effect, and MATLAB `ode45`) The Lorenz equations are the nonlinear autonomous three-dimensional system

$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= x(\rho - z) - y, \\ \dot{z} &= xy - \beta z \end{aligned}$$

where the dot notation indicates the time-derivative $\frac{d}{dt}$. Using

$$\sigma = 10, \quad \rho = 28, \quad \beta = 8/3,$$

plot the three-dimensional trajectory of the particle initially located at $(x, y, z) = (-8, 8, 27)$ for $0 \leqslant t \leqslant 10$ using `ode45`.

[4] You may ignore this step and use `tand`.