ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Models, Algorithms and Data (MAD): Introduction to Computing

Spring semester 2019

Prof. Dr. Jens Honore Walther
Dr. Julija Zavadlav
ETH Zentrum, CLT
CH-8092 Zürich

Set 9

Issued: 26.04.2019; Due: 05.05.2019

In this exercise, you will practice Richardson extrapolation for improving the accuracy of function evaluation and implement Romberg integration in an engineering problem.

Question 1: Finite differences with Richardson extrapolation

This is a pen and paper exercise.

a) A finite difference approximation (i.e., a numerical approximation) of the first derivative of a function $f(x)$ at $x = 0$ is

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} = G_0(h),$$

and the n-th application of Richardson extrapolation is given by the formula

$$G_n(h) = \frac{1}{2^n - 1}(2^n G_{n-1}(h/2) - G_{n-1}(h)).$$

Let $f(x) = x + e^x$. Set $h = 0.4$ and compute the Richardson extrapolation up to $G_2(h)$. Keep 5 decimal points throughout the calculations.

b) Since the exact value is known ($f'(0) = 2$), you can compute the error $E_n(h) = |G_n(h) - 2|$ for each term in a). Is the accuracy improved over the iterations?

Question 2: Pseudocode for Romberg integration

Write a pseudocode for Romberg integration. Write your own code from scratch or use the skeleton pseudocode below.

Algorithm 1 Romberg integration

Input:

function $f(x)$
interval boundaries a, b
number of iterations K

Output:

$I_K^1 = \texttt{integral}[K, 0]$ approximation to the integral $\int_a^b f(x)\ \mathrm{d}x$

Steps:

$\texttt{maxNumIntervals} \leftarrow 2^K$

// Precompute and store function evaluations
$\texttt{hmin} \leftarrow (b-a)/\texttt{maxNumIntervals}$
for $i \leftarrow 0, \ldots, \texttt{maxNumIntervals}$ **do**
.
end for

// Compute level 0 integrals
for $r \leftarrow 0, \ldots, K$ **do** // refinement
$\texttt{numIntervals} \leftarrow 2^r$
$\texttt{step} \leftarrow 2^{K-r}$ // step between two function evaluations for this refinement
.
.
.
.
// composite trapezoidal rule:
.
end for

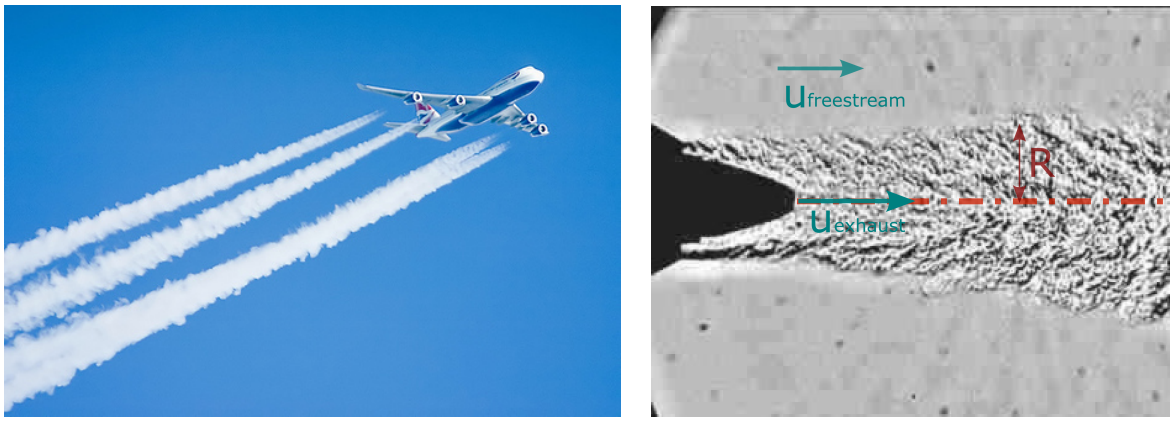
//Advance to higher precision according to Romberg
for $l \leftarrow 1, \ldots, K$ **do** //level
.
.
.
end for

Question 3: Romberg integration

Staring absentmindedly out the window when flying back from your vacation in Hawaii, you suddenly notice the shimmering stream trailing out from the turbofan engine. You begin to wonder why the stream is shaped that particular way, and whether it may be possible to predict the velocity profile across the cross section of the contrail (the white stream in Fig. 1). Using the in-flight Wi-Fi, you quickly figure out that such flows are referred to as *free shear flows*, and there indeed exists a formula for predicting the cross-sectional velocity profile. You decide to start with the formula for a simplified representation called a *planar mixing layer*:

$$u(r) = u_{\text{exhaust}} + (u_{\text{freestream}} - u_{\text{exhaust}}) \cdot \operatorname{erf}\left(\frac{2r}{R}\right),$$

where r is the radius from the center of the jet, and R is the distance to the outer edge of the jet (Fig. 1b), $u_{\text{freestream}} = 800$ km/h and $u_{\text{exhaust}} = 1200$ km/h are the *freestream* and *jet exhaust* velocities, respectively.



(a) (b)

Figure 1: (a) Jet contrails in the wake of a Boeing 747 (source: www.wired.com/2009/05/climate-contrail-connection-contested) (b) *Schlieren* image of the jet exhaust from a single engine (source: supersonic.eng.uci.edu/meaf/jse_round_m09u410.htm)

$\operatorname{erf}(x)$ here stands for the error function:

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

Unfortunately, you quickly figure out that this particular integral has no closed form solution. Recalling your lecture on Romberg numerical integration, you decide to write code that'll help you compute the velocity profile as a function of the radius from the center, and then plot it.

a) First you have to implement Romberg integration. Recall the two most important formulas from the lectures:

$$I_k^n = \frac{4^k I_{k-1}^{2n} - I_{k-1}^n}{4^k - 1},$$

$$I_0^n = \frac{h}{2}\left[f(a) + f(b) + 2\sum_{j=1}^{n-1} f(a + j\,h)\right], \; h = \frac{b-a}{n}.$$

By looking at the second formula, you can notice that I_0^{2m} requires function evaluations at the same points as I_0^m (plus some extra points) for any m. As the function evaluations may be in general very time-consuming, you have to minimize the redundancy in your evaluations and reuse the values computed for the coarser approximation in the finer approximation (or vice-versa). Think of an optimal solution and implement it.

To test the code, compute the approximation I_4^1 for the integral of $f(t) \equiv (2/\sqrt{\pi})\, e^{-t^2}$ on the interval $[0, 0.5]$. Output all the I_k^n calculated on the way together with associated errors (exact value: $\mathrm{erf}(0.5) = 0.5204998778130467$) and observe the increasing accuracy as you advance to higher order approximations.

b) Plot the velocity profile along the radial direction of the jet, i.e., $u(r)$ with $r/R \in [0, 1]$ on the X axis. Comment on whether the profile you get makes sense physically, especially at $r/R = 0$ and $r/R = 1$.