Norwegian University of Science and Technology
Department of Mathematical Sciences

TMA4135
Matematikk 4D
Autumn 2018

Exercise set 8

The theory and the codes are taken from the Jupyter notebook on *Polynomial Interpolation* an *Numerical Quadrature.*

Exercises supposed to be done by hand are marked with an (H).

Exercises in which you are supposed to use/modify code in the Jupyter notebook is marked with a (J).

For Jupyter-exercises, hand in a screen-dump of the relevant cell with output.

1 **a)** (H) Given the points

x_i	-2	1	6
y_i	1	2	3

Set up the table of divided differences, and write down the second order interpolation polynomial in the Newton form.

b) (H) Find the interpolation polynomial interpolating the points from **a)** and one extra point, $x_3 = -3/4$ and $y_3 = 3/2$.

Inverse interpolation This is a strategy that can be used to find approximations to solutions of an equation $f(x) = 0$.

Given a sufficiently continuous function $f(x)$, with a root r in some interval $[a, b]$. Choose this interval sufficiently small to ensure f to be strict monotonically increasing or decreasing at $[a, b]$. Under these conditions f is invertible on the interval, in the sense that

$$y = f(x) \qquad \Rightarrow \qquad x = f^{-1}(y).$$

In particular:

$$f(r) = 0 \qquad \Rightarrow \qquad r = f^{-1}(0).$$

The strategy is then: Choose $n + 1$ distinct data points (x_i, y_i) in the interval $[a, b]$ and find the interpolation polynomial $p_n(y)$ satisfying

$$p_n(y_i) = x_i.$$

In this case, $p_n(y) \approx f^{-1}(y)$, and $r \approx p_n(0)$.

c) (H) Let $f(x) = x^2-3$, and $[a, b] = [1, 3]$. As nodes, choose $x_0 = 1, x_1 = 2, x_2 = 3$ and use the idea outlined above to find an approximation to the solution of $f(x) = 0$. How close to the exact solution is the approximation?

To get a better approximation, add the node $x_3 = 3/2$. Will this provide a better result?

Hint: Use the results from **a)**.

d) (J) Repeat the example above, but now with $n+1$ uniformly distributed nodes over the interval $[1, 3]$. Use the functions `divdiff` and `newton_interpolation`. Choose $n = 2$ (to control your hand calculations), $4, 8$ and 16. Find the approximation in each case, as well as the error.

2 (H, optional, but part of the curriculum).

We shall derive the Chebyshev polynomials and their zeros

a) Defining $\kappa := \cos^{-1}(x)$ for $x \in [-1, 1]$, show that $T_n(x) = \cos(n\kappa)$ and $U_n(x) = \sin(n\kappa)$ solve the Chebychev differential equation

$$(1 - x^2)\frac{d^2y}{dx^2} - x\frac{dy}{dx} + n^2 y = 0$$

for $n = 0, 1, 2, 3, 4, \ldots$. The $T_n(x)$ and $U_n(x)$ are called Chebyshev polynomials of the 1st and 2nd kind of degree n, respectively.

b) Show that $T_n(x) = \cos(n\kappa)$ satisfies the recurrence relation

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x).$$

That is, the Chebyshev polynomials $T_0(x)$ and $T_1(x)$ imply all other polynomials by means of this recurrence formula. (Hint: use the trigonometric identities on $\cos((n \pm 1)\kappa)$.)

c) Write down the first 7 Chebyshev polynomials as polynomials in x.

d) Derive the formula for the roots of $T_{n+1}(x)$. Consider the zeros $\{x_i^{\text{cheb}}\}_{i=0}^{n}$ for $n = 3, 4, 5$ and deduce the corresponding points on the unit circle in the upper half plane. Graph these three cases.

e) Show that for Chebyshev polynomials written on the form

$$T_{n+1}(x) = c_{n+1}x^{n+1} + c_n x^n + \cdots + c_1 x + c_0$$

the leading coefficient satisfies $c_{n+1} = 2^n$. Use this to prove that the polynomial

$$\omega_{\text{cheb}}(x) = \prod_{i=0}^{n}(x - x_i^{\text{cheb}})$$

satisfies

$$|\omega_{\text{cheb}}(x)| \leq \frac{1}{2^n} \text{ for all } x \in [-1, 1].$$

f) Compute the integral

$$\int_{-1}^{1} \frac{T_n(x)T_m(x)}{\sqrt{1 - x^2}} dx.$$

One says that the Chebyshev polynomials form an orthogonal set on the interval $[-1, 1]$ with a so-called weight function $1/\sqrt{1 - x^2}$ (Hint: recall the computation of the integral $\int_0^\pi \cos(n\lambda)\cos(m\lambda)d\lambda$ for integers m, n.)

3 Consider the integral

$$\int_1^3 e^{-x} \mathrm{d}x.$$

a) (H) Find numerical approximations to the integral using Simpson's method over 1 and 2 intervals, that is $S_1(1,3)$ and $S_2(1,3)$. Find an error estimate for $S_2(1,3)$, and compare with the real error.

b) (H) Find the number of intervals m that guarantees that the approximation of the integral by the composite (sammensatt) Simpson's method is less than 10^{-8}.

NB! $h = (b-a)/(2m)$.

c) (J) Find the numerical approximation of the integral by using the function `simpson`, and m from point **b)**. Verify that the error is less than 10^{-8}.

4 The aim of this exercise is to repeat the exercise done on Simpson's method in the note, but now with a Gauss–Legendre quadrature. Which is described at the end of the note.

a) (H) Find the Gauss–Legendre quadrature over the interval $[-1,1]$ with $m=3$.

b) (H) Confirm that the quadrature has degree of precision 5.

c) (H) Transfer the quadrature over to some arbitrary interval $[a,b]$.
Try to express the quadrature more elegantly, e.g., by introducing $c = (b+a)/2$ and $h = (b-a)/2$, and the nodes in terms of c and h.
Use it to find an approximation to $\int_1^3 e^{-x} \mathrm{d}x$.
What is the error?

The error of this Gauss–Legendre quadrature over one interval is given by

$$E(a,b) = \int_a^b f(x)\mathrm{d}x - Q(a,b) = \frac{(b-a)^7}{2016000} f^{(6)}(\eta), \qquad \eta \in (a,b).$$

d) (H) Use this to find an error expression for the composite Gauss–Legendre quadrature

$$Q_m(a,b) = \sum_{k=0}^{m-1} Q(X_k, X_{k+1}),$$

where $Q(X_k, X_{k+1})$ is the basic quadrature over the interval $[X_k, X_{k+1}]$, $X_k = a + kH$, $k = 0, 1, \ldots, m$ and $H = (b-a)/m$.
Based on this expression, find the number of intervals m that guarantees that the error in the composed method is less than 10^{-8} when applied to the problem in point **b)**. Compare with the similar result for Simpson's method.

e) (H) Based on $Q_1(a,b)$ and $Q_2(a,b)$, find an error estimate for $Q_2(a,b)$ (you may assume that $f^{(6)}(x)$ is almost constant over the interval).

f) (J) Write a function `gauss_basis` (similar to `simpson_basis`) implementing the Gauss–Legendre quadrature with error estimate.
Check your function by ensuring that all polynomials of degree 5 or less are exactly integrated.

Use the function to confirm your result from point **c)**. Compare the exact error and the error estimate.

g) (J) Write an adaptive integrator based on the Gauss–Legendre quadrature (you will only need to change one `simpson_basis` with `gauss_basis` in `simpson_adaptive`). Test it on the integrals:

$$i) \qquad \int_1^3 e^{-x}\mathrm{d}x$$

$$ii) \qquad \int_0^5 \frac{1}{1+16x^2}\mathrm{d}x$$

$$iii) \qquad \int_0^2 \left(\frac{1}{(x-0.3)^2+0.01} + \frac{1}{(x-0.9)^2+0.04}\right)\mathrm{d}x$$

Use the tolerances $10^{-3}, 10^{-6}, 10^{-10}$.

Hand-in: Find the error and compare with the tolerance in each case. Comment on the result.

Compare with the similar results you get from `simpson_adaptive` in the note and comment on this.

NB! The value of the third integral is 41.326213804391148551.