October 18, 2020

MAT3110 – Introduction to numerical analysis

Mandatory assignment 2 of 2

Submission deadline

Thursday 29 October 2020 at 14:30 in Canvas (canvas.uio.no).

Instructions

Bachelor students can choose between scanning handwritten notes or use LaTeX. Master students are required to use LaTeX. The assignment must be submitted as a single PDF file. Scanned pages must be clearly legible; please use either the "Color" or "Grayscale" settings, not "Black and white". The submission must contain your name, course and assignment number.

It is expected that you give a clear presentation with all necessary explanations. Remember to include all relevant plots and figures. Students who fail the assignment, but have shown that they have made a genuine attempt at solving the exercises, are given a second attempt at revising their answers. All aids, including collaboration, are allowed, but the submission must be written by you and reflect your understanding of the subject. If we doubt that you have understood the content you have handed in, we may request that you give an oral account.

In exercises where you are asked to write a computer program, you need to hand in the code along with the rest of the assignment.

Application for postponed delivery

If you need to apply for a postponement of the submission deadline due to illness or other reasons, you **have to** contact the student administration at the Department of Mathematics (e-mail: studieinfo@math.uio.no) well before the deadline.

All mandatory assignments in this course must be approved in the same semester, before you are allowed to take the final examination.

Complete guidelines for delivery of mandatory assignments:

www.uio.no/english/studies/examinations/compulsory-activities/mn-math-mandatory.html

Grading: To get your submission approved, you need to show that you have tried to solve every exercise, and have solved about 2/3 of the problems correctly. Each of the three problems 1, 2, 3 are weighted by 1/3 each.

In this assignment we will consider the *ordinary differential equation*[*]

$$\begin{aligned} y''(x) &= f(y(x)) \qquad \text{for } x \in (0,1) \\ y(0) &= \alpha,\ y(1) = \beta. \end{aligned} \tag{1}$$

Here, $f \in C(\mathbb{R})$ is a given continuous function, $\alpha, \beta \in \mathbb{R}$ are given "boundary data", and $y : [0,1] \to \mathbb{R}$ is the unknown function. A function $y = y(x)$ is a *solution of* (1) if $y''(x) = f(y(x))$ at all points $x \in (0,1)$, and $y(0) = \alpha$, $y(1) = \beta$.

If f is a linear or constant function then you have learnt how to solve (1) exactly. If f is *nonlinear* then a solution still exists, but in general it does not have a closed form expression. Our goal will be to numerically approximate this solution.

Problem 1. Let $N \in \mathbb{N}$. We partition the domain $x \in [0,1]$ into N intervals with endpoints

$$x_i = ih \quad (i = 0, 1, \ldots, N) \qquad \text{where } h = \tfrac{1}{N}.$$

At each "grid point" x_i we approximate $z_i \approx y(x_i)$.

(a) Explain why the set of equations

$$\begin{aligned} z_{i-1} - 2z_i + z_{i+1} &= h^2 f(z_i) \qquad \text{for } i = 1, 2, \ldots, N-1 \\ z_0 &= \alpha,\ z_N = \beta \end{aligned} \tag{2}$$

gives an approximation of (1).

(b) Denote the unknown by $z = \begin{pmatrix} z_0 & z_1 & \ldots & z_N \end{pmatrix}^\mathsf{T}$. Find a tridiagonal[†] matrix $A \in \mathbb{R}^{(N+1)\times(N+1)}$ and a vector $b(z) \in \mathbb{R}^{N+1}$ such that (2) can be equivalently written as

$$Az = b(z). \tag{3}$$

(Note that the vector $b(z)$ depends on z. Make sure that you include the boundary conditions from (2) in (3).)

(c) A matrix $A \in \mathbb{R}^{n\times n}$ is *diagonally dominant* if

(i) none of the rows in A contains only zeros
(ii) $|A_{i,i}| \geqslant \sum_{j\neq i} |A_{i,j}|$ for all $i = 1, \ldots, n$
(iii) there is at least one index i such that $|A_{i,i}| > \sum_{j\neq i} |A_{i,j}|$.

It is a fact (which you do not need to prove) that all tridiagonal, diagonally dominant matrices are invertible. (The proof is an elementary, but tedious, proof showing that Gaussian elimination with A works.)

Prove that A from (3) is invertible.

[*]This ODE can be viewed as a model of a piece of string. The ends of the string are fixed at the two points $(0,\alpha)$ and $(1,\beta)$, and $y(x)$ denotes the height above the ground of the string at the position x. The right-hand side f is then a force acting in the vertical direction.

[†]A matrix A is *tridiagonal* if $A_{i,j} = 0$ for $j > i+1$ and $j < i-1$.

(d) Consider now a general linear system

$$Bu = c, \tag{4}$$

where $c \in \mathbb{R}^n$ is given, $u \in \mathbb{R}^n$ is unknown, and where $B \in \mathbb{R}^{n\times n}$ is tridiagonal and diagonally dominant. Explain why solving (4) using Gaussian elimination only requires $O(n)$ floating point operations (flops), as opposed to $O(n^3)$ for a general matrix B.

We now wish to solve (3). Since b depends on z, the equation is nonlinear, and we cannot solve it directly. Instead, we will use an iterative solver which, given an initial guess $z^{(0)} \in \mathbb{R}^{N+1}$, computes approximations $z^{(1)}, z^{(2)}, \cdots \in \mathbb{R}^{N+1}$. To measure the difference between successive iterations (which should be an indication of the error) we will use the L^2-type norm

$$\left\|z^{(k+1)} - z^{(k)}\right\| = \sqrt{h\sum_{i=0}^{N} \left|z_i^{(k+1)} - z_i^{(k)}\right|^2}.$$

We will use two iterative solvers: a fixed point iteration and Newton's method.

Problem 2. Since A is invertible, we can write (3) as

$$z = A^{-1}b(z). \tag{3'}$$

(a) Use (3') to find a fixed point iteration which approximates z.

(b) Using a programming language of your choice, write a function[‡]

```
fixedpointODE(f, alpha, beta, N, tolerance)
```

which takes as input f, α, β, N, and a positive number `tolerance`, runs the fixed point iteration from problem (a) until $\left\|z^{(k)} - z^{(k-1)}\right\| <$ `tolerance`, and returns the final iterate $z^{(k)}$.

(c) Test your code on the two problems

$$\alpha = 0,\ \beta = 1,\ f(z) = -1 \tag{5a}$$

and

$$\alpha = 1,\ \beta = 1,\ f(y) = -y^2, \tag{5b}$$

in both cases using $N = 50$, $z^{(0)} = 0$ and `tolerance` $= 10^{-12}$. Include plots of z as a function of x from both runs. (For (5a) you can compute the exact solution by hand, in case you want to verify your code.)

(d) Estimate how many floating point operations (flops) each iteration requires. You only need a rough estimate, such as $O(N^p)$ for some p.

Problem 3. Alternatively, we can write (3) as

$$g(z) = 0, \qquad \text{where } g(z) = Az - b(z). \tag{3''}$$

[‡]Functions can be passed as variables both in Matlab, Python, and other languages.

(a) Write down Newton's method for solving (3'').

(b) Write a function

```
newtonODE(f, df, alpha, beta, N, tolerance)
```

which takes as input f, its derivative f', as well as α, β, N, and a positive number `tolerance`, runs the Newton iteration from problem (a) until $\left\|z^{(k)} - z^{(k-1)}\right\| <$ `tolerance`, and returns the final iterate $z^{(k)}$.

(c) Test your code in the same way as in problem 2(c).

(d) Estimate how many floating point operations (flops) each iteration requires. You only need a rough estimate, such as $O(N^p)$ for some p.

(e) Based on how many iterations your test runs required, which of the fixed point iteration or Newton's method do you think is faster?