NTNU

Norwegian University of Science and Technology

TMA4125 Matematikk 4N

Organisation & Preliminaries

Ronny Bergmann

Institute of Mathematical Sciences, NTNU.

January 12, 2023

Team

Lecturer.
Lecturer: Ronny Bergmann (office hour: Fridays 16.15–17-15)
Sentralbygg 2, 1034, ronny.bergmann@ntnu.no

About me. german, PhD from Lübeck, Førsteamanuensis since 2021.

Research topics. Numerical Mathematics, Optimization, Riemannian manifolds, Mathematical Image Processing

Teaching Assistants.

- Øyvind S. Auestad oyvinau@ntnu.no
- Alessandro Contri alessandro.contri@ntnu.no
- Håkon Noren hakon.noren@ntnu.no

Resources

- the wiki for all information
  https://wiki.math.ntnu.no/tma4125/2023v/
- the course description
  https://www.ntnu.edu/studies/courses/TMA4125/
- the forum for feedback and questions
  https://mattelab2023v.math.ntnu.no/c/tma4125/
- the blackboard group (mainly) for announcements
  https://ntnu.blackboard.com/
- a JupyterHub http://tma4125.apps.stack.it.ntnu.no
- ovsys2 to hand in exercises https://ovsys.math.ntnu.no/

Lecture. Thursday & Friday 14.15-16.00 here in F1

Exercises. Once a week, starting next week
For the exam you have to pass 8 of the 12 exercise sheets.

Math 4N: Let‘s get started!

...with three short questions.

https://www.menti.com/ye259f1zcy   menti.com Code 7125 2558

TMA4125 Matematikk 4N – What is it about?

In a nutshell, the lecture is about

We want to look at analytical amd numerical techniques for solving ordinary differential equations (ODEs) as well as partial differential equations (PDEs).

We further introduce required numerical concepts in general.

Key Topics.

- Numerical Differentiation, boundary value problems
- Polynomial & Spline Interpolation, Numerical Integration
- Rounding Errors, Fix point iterations, Newton's Method
- periodic functions & Fourier series, DFT
- numerics for / solving PDEs, wave & heat equation
- Laplace transform and numerics for / solving ODEs

Preliminaries

Real vector spaces

A real vector space is a set V together with operations $+$ (addition) and $\cdot$ (multiplication with a scalar) that satisfy

1. $x + y \in V$ for all $x, y \in V$
2. $x + y = y + x$ for all $x, y \in V$
3. $x + (y + z) = (x + y) + z$ for all $x, y, z \in V$
4. There exists some element $0 \in V$ such that $x + 0 = x$ for all $x \in V$
5. For all $x \in V$, there exists some element $(-x) \in V$ s. t. $x + (-x) = 0$
6. $\alpha \cdot x \in V$ for all $x \in V$ and $\alpha \in \mathbb{R}$
7. $\alpha \cdot (\beta \cdot x) = (\alpha\beta) \cdot x$ for all $x \in V$ and $\alpha, \beta \in \mathbb{R}$
8. $1 \cdot x = x$ for all $x \in V$
9. $\alpha \cdot (x + y) = \alpha \cdot x + \alpha \cdot y$ for all $x, y \in V$ and $\alpha \in \mathbb{R}$
10. $(\alpha + \beta) \cdot x = \alpha \cdot x + \beta \cdot x$ for all $x \in V$ and $\alpha, \beta \in \mathbb{R}$

NTNU

Complex vector spaces

A complex vector space is defined in exactly the same way as a real vector space, just replacing $\mathbb{R}$ with $\mathbb{C}$ in the definition from the last slide, so that the scalars $\alpha, \beta \in \mathbb{C}$ are now allowed to be complex numbers.

Examples of vector spaces

The following examples we will see throughout the course

- The set $\mathbb{R}^m$ of real vectors with m components
- The set $\mathbb{R}^{m\times n}$ of real-valued $m \times n$ matrices
- The set $\mathbb{P}_n$ of polynomials of degree n or less
- The set $C^m[a,b]$ of all functions with continuous first m derivatives on the interval $[a,b]$.
  we write $C[a,b]$ for $C^0[a,b]$, i.e. the set of all continuous functions.

Note that $C^n[a,b] \subset C^m[a,b]$ for $n > m$. Further, $\mathbb{P}_n \subset C^\infty[\mathbb{R}]$.

Exercise. Let's verify that $\mathbb{P}_n$ and $C^m[a,b]$ are actually vector spaces.

Exercise: Vector spaces of Polynomials $\mathbb{P}_n$

What are "$+$" and "$\cdot$" for polynomials $f, g \in \mathbb{P}_n$?

$$(f+g)(x) =$$

$$(\alpha \cdot f)(x) =$$

Exercise: Vector spaces of smooth functions $C^m[a,b]$

What are "$+$" and "$\cdot$" for $f, g \in C[a,b]$?

$$(f+g)(x) =$$

$$(\alpha \cdot f)(x) =$$

Norms

Let V be a vector space. A norm $\|\cdot\|$ is a function such that the following properties hold

1. $\|x\| \geq 0$ for all $x \in V$
2. $\|x\| = 0$ if and only if $x = 0$
3. $\|\alpha \cdot x\| = |\alpha|\|x\|$ for all $x \in V$ and $\alpha \in \mathbb{R}$
4. $\|x + y\| \leq \|x\| + \|y\|$ for all $x, y \in V$ (triangle inequality)

Remarks.

- the norm $\|\cdot\|$ of $x \in V$ is essentially a measure of the size of x
- $\|x - y\|$, $x, y \in V$, is a measure for the distance between x and y or: a way of saying how similar they are.
- usually: different meaningful norms for one vector space V

Norms for $\mathbb{R}^n$

On $\mathbb{R}^n$, $n \in \mathbb{N}$ we denote elements by $\boldsymbol{x} = (x_1, \dots, x_n)^{\mathrm{T}} \in \mathbb{R}^n$.

Then we have the following norms

- The maximum norm $\|\boldsymbol{x}\|_\infty = \max_{i=1,2,\dots,n} |x_i|$
- The Euclidean norm $\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$
- more generally the ℓ^p norm $\|\boldsymbol{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}, \quad 1 \leq p \leq \infty$

Note. For $n = 1$ all these are equal and we just use $\|x\| = |x|$.

Norm of functions $C[a,b]$

On $C[a,b]$, $a < b$, we can similarly define for a function $f \in C[a,b]$ the following norms

- The maximum-norm $\|f\|_\infty = \max_{x \in [a,b]} |f(x)|$
- The L^2-norm $\|f\|_2 = \sqrt{\int_a^b f(x)^2 \, \mathrm{d}x}$
- more generally the L^p-norm $\|f\|_p = \left(\int_a^b |f(x)|^p \, \mathrm{d}x \right)^{1/p}, \quad 1 \le p \le \infty$

For two functions $f, g \in C[a,b]$

- $\|f - g\|_\infty$ measures the maximal pointwise difference
- $\|f - g\|_2$ measures the average (quadratic) difference

Exercise. Check that $\|f\|_2$ is a norm.

Examples of norms

For the $\mathbb{R}^n$ let's look at a small Python code example.

Let $f(x) = sin(x)$ on $[0, 2\pi]$ be given. Then $f \in C[0, 2\pi]$.

We obtain

$$\|f\|_2 = \sqrt{\int_0^{2\pi} sin^2(x)\,\mathrm{d}x} = \sqrt{\pi} \approx 1.7725$$

$$\|f\|_\infty = \max_{x\in[0,2\pi]} |\sin(x)| = 1$$

Convergence of a sequence

Definition. (Convergence of a sequence) Let $\{x_k\}_{k=0}^{\infty}$ be an infinite sequence of real numbers. The sequence converges to x, if, for any $\varepsilon > 0$ there exist a positive integer $N(\varepsilon)$ such that $|x_k - x| < \varepsilon$ whenever $k > N(\varepsilon)$.

Common notations

$$\lim_{k\to\infty} x_k = x \qquad \text{or} \qquad x_k \to x \text{ as } k \to \infty.$$

Example. The sequence

$$x_k = \left(1 + \frac{1}{k}\right)^k \text{ converges to } \lim_{k\to\infty} x_k = e.$$

Let's use Python to get an intuition when we are unsure.

Convergence of an Iterative Process

Let X be the exact solution of a problem, and X_k a numerical approximation achieved by some iterative process $X_{k+1} = G(X_k)$.

We say X_k converges to X if

$$\lim_{k\to\infty} \|X - X_k\| = 0.$$

We measure the error as $e_k := \|X - X_k\|$
In practice: Choose a (problem dependent) appropriate norm.

The order of convergence is p if there exist a positive constant C such that

$$e_{k+1} \leq C e_k^p$$

Notation. We speak of linear ($p = 1$), quadratic ($p = 2$) and cubic ($p = 3$) convergence.

Numerical Verification

Given. The errors during our iterations e_k, $k = 1, 2, \dots$

Assumptions. $e_{k+1} = C_k e_k^p$, $C_k \approx C_{k+1} \approx C$ does not change (much).

Goal. Compute the convergence order p.

Formula.

$$p \approx \frac{\log(e_{k+2}/e_{k+1})}{\log(e_{k+1}/e_k)}$$

Example. Newton's method to find x^* with $f(x^*) = 0$ is given by

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

and has quadratic convergence "sufficiently close to the solution"
(see `preliminaries.py` and its function `occ-iterations`).

Interlude: The big $\mathcal{O}$-notation

Let f and g be some real valued function and $a \in \mathbb{R}$. We say that

$$f(x) = \mathcal{O}(g(x)) \qquad \text{as } x \to a$$

if there exist $\delta > 0$ and $M > 0$ such that

$$|f(x)| \leq M|g(x)| \quad \text{when } 0 < |x - a| < \delta$$

That is: locally around a the function f can be bounded (up to a constant, M) by the function g.

Convergence of h-dependent approximations

For a problem with exact solution X we consider the numerical solutions $X(h)$, which depend on a parameter h.
We define the (h-dependent) error

$$e(h) = \|X - X(h)\|.$$

The $X(h)$ are said to converge to X if $e(h) \to 0$ as $h \to 0$.

The order of approximation is p if there exists a constant $M > 0$ such that

$$e(h) \leq Mh^p.$$

Short Notation. We write this in short as $e(h) = \mathcal{O}(h^p)$

Numerical Verification

Given. Errors $e(h_k)$ for sufficiently small h_k,
For example with some initial small H, take $h_k = H/2^k$

Assumption. $e(h) = Ch^p$ (or C does again not change much),

Goal. Compute the order of approximation p.

Formula. Since $e(h_k) \approx Ch_k^p$ and $e(h_{k+1}) \approx Ch_{k+1}^p$ we get similarly to before

$$p \approx \frac{\log(e(h_{k+1})/e(h_k))}{\log(h_{k+1}/h_k)}$$

Visual approach. A log-log-plot of $e(h)$
$\Rightarrow p$ is the slope of the line si called convergence plot.

See `preliminaries.py` and its function `occ-h` for an example invesigating the trapezoidal rule.

Taylor Polynomial

In numerics we truncate the Taylor expansion after m summands:

For any $f \in C^{m+1}[a, b]$ fix some $x_0 \in [a, b]$.
We obtain the Taylor polynomial (and a Remainder) of f (at x_0):

$$f(x) = \sum_{k=0}^{m} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k + R_{m+1}(x_0)$$

where the remainder term is given by

$$R_{m+1}(x_0) = \frac{f^{m+1}(\xi)}{(m+1)!}(x - x_0)^{m+1},$$

for some unknown ξ between x_0 and x.

We often write this using the big $\mathcal{O}$-notation with $h = x - x_0$ as

$$f(x_0 + h) = \sum_{k=0}^{m} \frac{f^{(k)}(x_0)}{k!} h^k + \mathcal{O}(h^{m+1})$$

Some other useful results

Theorem 1. Let $f \in C[a,b]$ and let u be a number between $f(a)$ and $f(b)$. Then there exists at least one $\xi \in (a,b)$ such that $f(\xi) = u$

Theorem 2. (Rolle's theorem) Let $f \in C^1[a,b]$ and $f(a) = f(b) = 0$. Then there exists at least one $\xi \in (a,b)$ such that $f`(\xi) = 0$.

Theorem 3. (Mean value theorem) Let $f \in C^1[a,b]$. Then there exists at least one $\xi \in (a,b)$ such that

$$f`(\xi) = \frac{f(b) - f(a)}{b - a}.$$

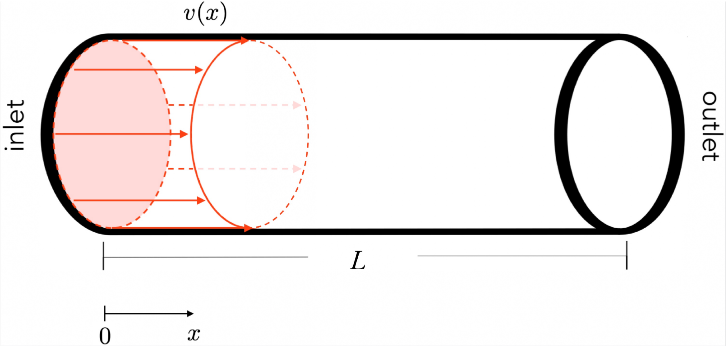
Theorem 4. (Mean value theorem for integrals) Let $f \in C^1[a,b]$ and g an integrable function that does not change sign on $[a,b]$ Then there exists at least one $\xi \in (a,b)$ such that

$$\int_a^b f(x)g(x)\,\mathrm{d}x = f(\xi)\int_a^b g(x)\,\mathrm{d}x.$$

NTNU

A first ODE example and Boundary Value Problems

Motivation: A chemical reaction



Goal. Obtain the concentration $u(x), 0 \leq x \leq L$, of a chemical A in a cylinder during a chemical reaction, for example

- A first-order reaction $A \to B + C$, e. g. $\mathrm{CH_4} \to \mathrm{C} + 2\mathrm{H_2}$
- A second-order reaction $2A \to B + C$ e. g. $2\mathrm{NO_2} \to 2\mathrm{NO} + \mathrm{O_2}$

in a steady state – pr in other words the eqilibrium.

Model for the concentration $u(x)$ of A: A BVP

Assumption. We are interested in the steady state $u(x)$, and we know the (prescibed/given) concentrations u_0 (left) and u_L (right) at the boundary.

We obtain a boundary value problem:

Determine a function $u = u(x)$ that fulfills

$$\begin{aligned} \alpha u'' - v u' - \kappa u^n &= 0, \\ u(0) &= u_0, \\ u(L) &= u_L, \end{aligned}$$

where α, v and κ are assumed to be given constant coefficients,

Note. For $n = 1$ the BVP is linear and this problem can be solved analytically (as we will do later in the course)

If

A Numerical Scheme

For this example: Consideer a finite difference scheme.

Steps of a finite difference scheme.

1. Discretize the domain on which the equation is defined
2. on each grid point: approximate the derivatives using neighbouring values
3. Replace the exact solution by this approximation

$\Rightarrow$ We obtain a system of equations

4. Solve the resulting system of equations

Today.
Consider the linear case $n = 1$ but with nonnonconstant coefficients:

$$u'' + p(x)u' + q(x)u = r(x), \qquad a \leq x \leq b, \qquad u(a) = u_a, \quad u(b) = u_b$$

Numerical Differentiation: Finite Differences

Reminder.
The derivative of $f(x)$ is defined as $f'(x) = \lim_{h \to 0} \dfrac{f(x+h) - f(x)}{h}$.

Idea. Instead of the limit, take a small value of h to approximate f'.

The 3 most common approximations are

$$f'(x) \approx \begin{cases} \dfrac{f(x+h) - f(x)}{h}, & \text{Forward difference,} \\ \dfrac{f(x) - f(x-h)}{h}, & \text{Backward difference,} \\ \dfrac{f(x+h) - f(x-h)}{2h}, & \text{Central difference.} \end{cases}$$

and for f'' we iterate

$$f''(x) = (f'(x))' \approx \frac{f'(x+h) - f'(x)}{h} = \frac{\frac{f(x+h)-f(x)}{h} - \frac{f(x)-f(x-h)}{h}}{h} = \frac{f(x+h) - 2f}{h}$$

$\Rightarrow$ Jupyter.

Numerical Differentiation: Error Analysis

! When we approximate $\Rightarrow$ knowledge about the error is crucial!

Using Taylor expansions we get for the forward difference

$$e(x,h) = f'(x) - \frac{f(x+h) - f(x)}{h} = -\frac{1}{2} f''(\xi) h$$

for some $\xi \in (x, x+h)$.
For the central difference we estimate

$$e(x,h) = f'(x) - \frac{f(x+h) - f(x-h)}{2h} = -\frac{1}{6} f'''(\eta) h^2$$

for some $\eta \in (x-h, x+h)$.
For the second order difference we obtain $e(x,h) = \mathcal{O}(h^2)$ as well.

To summarize.

- forward and backward difference have an order 1 of approximation
- the central and second difference have an order 2 of approximation

A Numerical Scheme for the Two Point BVP I

Numerical Scheme for the BVP consists of four steps

$$u'' + p(x)u' + q(x)u = r(x), \qquad a \le x \le b, \qquad u(a) = u_a, \quad u(b) = u_b$$

1. Choose N. Let $h = \frac{b-a}{N}$ and set $x_i = a + ih, i = 0, \dots, N$.
2. For each inner point $x_1, \dots, x_{N-1}$:
   Replace derivatives $u'(x_i)$, $u''(x_i)$ by finite differences:

$$\frac{u(x_i+h)u(x_{i+1}) - 2u(x_i) + u(x_i-h)u(x_{i-1})}{h^2} + p(x_i)\frac{u(x_i+h)u(x_{i+1}) - u(x_i-h)u(x_{i-1})}{2h}$$

   which is the same as above due to the $\mathcal{O}(h^2)$ error term.
3. Rename the unknowns $U_i = u(x_i)$ and ignore the error term

$$\frac{U_{i+1} - 2U_i + U_{i-1}}{h^2} + p(x_i)\frac{U_{i+1} - U_{i-1}}{2h} + q(x_i)U_i = r(x_i), \quad i = 1, \dots, N-1.$$

Note. These are $N-1$ equations with $N+1$ unknowns $U_0, \dots, U_N$

A Numerical Scheme for the Two Point BVP II

3. (cont.) **But.** We have the boundary conditions:

$$u(a) = u(x_0) = U_0 = u_a \quad \text{and } u(b) = u(x_N) = U_N = u_b.$$

We can write this in Matrix-Vector form $A\mathbf{U} = \mathbf{b}$ with

$$A = \begin{bmatrix} 1 & 0 & & & & & \\ v_1 & d_1 & w_1 & & & & \\ & v_2 & d_2 & w_2 & & & \\ & & v_3 & \ddots & \ddots & & \\ & & & \ddots & \ddots & w_{N-2} & \\ & & & & v_{N-1} & d_{N-1} & w_{N-1} \\ & & & & & 0 & 1 \end{bmatrix} \quad \text{with } \begin{aligned} v_i &= 1 - \frac{h}{2}p(x_i) \\ d_i &= -2 + h^2 q(x_i) \\ w_i &= 1 + \frac{h}{2}p(x_i) \end{aligned}\ ,$$

$\mathbf{b} = [u_a, h^2 r(x_1), \ldots, h^2 r(x_{N-1}), u_b]^{\mathrm{T}}$,

and the vector of unknowns $\mathbf{U} = \left[U_0, \ldots, U_N\right]^{\mathrm{T}}$.

4. Solve the linear system $A\mathbf{U} = \mathbf{b}$ w.r.t. $\mathbf{U}$.

NTNU

An Example

For the equation

$$u'' + 2u' - 3u = 9x, \quad u(0) = u_a = 1, \quad u(1) = u_b = e^{-3} + 2e - 5 \approx 0.486351$$

we know the exact solution $u(x) = e^{-3x} + 2e^{x} - 3x - 2$

$\Rightarrow$ We can take a look at how good our scheme is

to Jupyter!

Summary / Implementation

So for your implementation assume you are given An example like on slide 31.

Then you have to

1. Implement $p(x), q(x)$ (both constant in the example), and $r(x)$
2. Define a, b, u_a, u_b
3. Choose N and define h as well as the $x_i = a + ih$, $i = 0, \ldots, N$
4. Create the matrix A and the right hand side $\boldsymbol{b}$ from slide 30 (if $p(x), q(x)$ are constant A is actually even easier)
5. Solve the linear system $A\boldsymbol{U} = \boldsymbol{b}$

Common Boundary Conditions

We already saw, that we need information on the boundary to obtain a unique solution.

The most common conditions are

Dirichlet condition The solution is known at the boundary, i. e. we know $u(a) = u_a$ and $u(b) = u_b$ as in the example above

Neumann condition The derivative is known at the boundary, i. e. we know $u'(a) = v_a$ and $u'(b) = v_b$

Robin (or mixed) conditions A combination of the previous two

Question. How can we include Neumann conditions in the algorithm we just used?

Treating Neumann Conditions – Idea I

Idea 1. We can use a forward difference

$$u'_a = \frac{u(x_1) - u(x_0)}{h} + \mathcal{O}(h) \qquad \Rightarrow \qquad \frac{U_1 - U_0}{h} = u'_a$$

could replace our first equation.

Disadvantage. Low accuracy, since forward differences are only first order approximations

Challenge. What is the problem with a central difference here? We would get

$$u'_a = \frac{u(x_1) - u(x_{-1})}{h} + \mathcal{O}(h) \tag{1}$$

But we do not have a value at "$U_{-1} = u(x_{-1})$"!

Neumann boundary conditions: A Trick!

Trick. Let's introduce U_{-1} as a fictitious grid point, that is, at $x_{-1} = a - h$.

Idea. We can actually introduce two new formulae, since with U_{-1} we can also consider the BVP at x_0! We get

$$\frac{U_1 - 2U_0 + U_{-1}}{h^2} + p(x_0)\frac{U_1 - U_{-1}}{2h} + q(x_0)U_0 = r(x_0)$$
$$\frac{U_1 - U_{-1}}{2h} = v_a$$

Now the second one can be rephrased to $U_{-1} = U_1 - 2hv_a$. Plugging this into the first we get

$$\frac{2U_1 - 2U_0 - 2hv_a}{h^2} + p(x_0)v_a + q(x_0)U_0 = r(x_0)$$

as our new first equation.

Similarly: Neumann conditions at $x = b$ using a ficticious node U_{N+1}.