Technische Universität München
Lehrstuhl für Informatik V
Univ.-Prof. Dr. M. Bader
M.Sc. S. Seckler, M.Sc. D. Jarema

WS 2016/17
Scientific Computing
09.01.2017/11.01.2017

Worksheet 9
Sample Solutions

Partial Differential Equations

Jacobi Method

An iterative method to solve linear systems of equations $Ax = b$ with $A \in \mathbb{R}^{N\times N}$, $b \in \mathbb{R}^N$ is given by the Jacobi method. Starting from an initial vector $x^{(0)}$, the iteration procedure reads:

$$x_i^{(n+1)} = \frac{1}{A_{ii}}\left(b_i - \sum_{j\neq i} A_{ij}x_j^{(n)}\right), \quad i = 1, ..., N$$

Convergence of the Jacobi Method: Diagonal dominance

A matrix $A \in \mathbb{R}^{N\times N}$ is *diagonally dominant* if the following inequality holds for all $i \in \{1, ..., N\}$:

$$|A_{ii}| \geq \sum_{j\neq i} |A_{ij}| \tag{1}$$

Diagonal dominance is an important property of matrices and helps to show that for example the Jacobi iteration scheme for $Ax = b$ converges:

- If A is strictly diagonally dominant, that is we have a "$>$" in equation (1) for all rows i, then the Jacobi iteration converges.
- Let A be diagonally dominant and have at least one row with "$>$" in equation (1). If A is irreducible, then the Jacobi method converges.

We do not want to dive deeper into mathematics here and, thus, will not completely define what irreducibility means (we may discuss the respective definition during the exercise class). However, it should be noted that a matrix is irreducible if the matrix is tridiagonal and only has non-vanishing main- and subdiagonal entries, that is

$$A_{ij} \neq 0 \quad \forall |i - j| \leq 1.$$

(H) Exercise 1: Convection-Diffusion Systems

Consider the one-dimensional differential equation

$$-\frac{d^2u(x)}{dx^2} + v \cdot \frac{du(x)}{dx} = f(x), \quad x \in (0,1) \tag{2}$$

with velocity $v \in \mathbb{R}$ and Dirichlet boundary conditions $u(0) = c_0, u(1) = c_1$.
This equation models the transport of a quantity u in a fluid when the fluid is assumed to move at constant velocity v.

(a) Set up a *finite difference* scheme using

- the standard second-order discretization of the diffusive term (that is the second-order derivative)
- a symmetric second-order discretization for the convective term (that is the first-order derivative).

Write down the formulation for a single row of the arising linear system of equations

$$\sum_j A_{ij} u_j = b_i$$

with $u_j := u(j \cdot h)$, mesh size h and right-hand side b_i.

(b) Formulate the Jacobi relaxation to solve this system. Under which conditions does the Jacobi method converge?

(c) Replace the second-order discretization of the first-order derivative by a first-order one-sided discretization. Re-formulate the Jacobi relaxation for this case. Under which conditions can we expect convergence now?

Solution:

(a) The second-order stencil for the second-order derivative reads

$$\frac{d^2u}{dx^2} \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}$$

and for the first-order derivative

$$\frac{du}{dx} \approx \frac{u_{i+1} - u_{i-1}}{2h}.$$

Using these approximations and defining $f_i := f(ih)$ yields:

$$\left(-1 + \frac{vh}{2}\right) u_{i+1} + 2u_i + \left(-1 - \frac{vh}{2}\right) u_{i-1} = h^2 f_i$$

At the left or right boundary, the value $u_{i-1} = u(0) = c_0$ or $u_{i+1} = u(1) = c_1$ is already known and can thus be removed from the equation for the first inner point. The equations for the left/ right boundary read:

$$\begin{aligned} \text{Left boundary}: \quad & \left(-1 + \tfrac{vh}{2}\right) u_{i+1} + 2u_i &&= h^2 f_i - \left(-1 - \tfrac{vh}{2}\right) c_0 \\ \text{Right boundary}: \quad & 2u_i + \left(-1 - \tfrac{vh}{2}\right) u_{i-1} &&= h^2 f_i - \left(-1 + \tfrac{vh}{2}\right) c_1 \end{aligned}$$

Our matrix A hence has the following form:

$$A_{kl} = \begin{cases} \left(-1 + \frac{vh}{2}\right) & l = k+1 \\ 2 \quad \text{for} & l = k \\ \left(-1 - \frac{vh}{2}\right) & l = k-1 \end{cases}$$

The system is tridiagonal.

(b) The Jacobi iteration in iteration $n+1$ is given by:

$$u_i^{n+1} = \frac{1}{2}\left(h^2 f_i - \left(-1 - \frac{vh}{2}\right) u_{i-1}^n - \left(-1 + \frac{vh}{2}\right) u_{i+1}^n\right)$$

where u_i^n denotes the solution in iteration n.

The condition for the diagonal dominance reads:

$$\left|-1 + \frac{hv}{2}\right| + \left|-1 - \frac{hv}{2}\right| \leq 2 \Leftrightarrow \left|\frac{vh}{2} - 1\right| + \left|1 + \frac{vh}{2}\right| \leq 2$$

The arising conditions for diagonal dominance at the boundaries read:

$$\begin{aligned} \text{Left boundary}: \quad & \left|\tfrac{vh}{2} - 1\right| \leq 2 \\ \text{Right boundary}: \quad & \left|-\tfrac{vh}{2} - 1\right| \leq 2 \quad \Leftrightarrow \quad \left|\tfrac{vh}{2} + 1\right| \leq 2 \end{aligned}$$

The form of the Jacobi iteration does not need to be modified at the boundaries: we only solve the linear system for the *inner* points, as there is no separate equation for the boundary points. We can now think of the different cases (to evaluate the absolute values from the inequalities above):

- For $-\infty < vh \leq -2$:

$$\begin{aligned} \text{Inner:} \quad & -(\tfrac{vh}{2} - 1) - (\tfrac{vh}{2} + 1) \leq 2 \quad \Leftrightarrow \quad vh \geq -2 \\ \text{Left boundary:} \quad & -(\tfrac{vh}{2} - 1) \leq 2 \quad \Leftrightarrow \quad vh \geq -2 \\ \text{Right boundary:} \quad & -(\tfrac{vh}{2} + 1) \leq 2 \quad \Leftrightarrow \quad vh \geq -6 \end{aligned}$$

  where all conditions are solely fulfilled if $vh = -2$. In this case, all lower subdiagonal entries of the matrix A become zero. The matrix is hence reducible and no convergence can be proven.

- For $-2 < vh \leq 2$:

Inner:	$-(\frac{vh}{2}-1)+(\frac{vh}{2}+1)\leq 2$	$\Leftrightarrow$	$2 \leq 2$
Left boundary:	$-(\frac{vh}{2}-1)\leq 2$	$\Leftrightarrow$	$vh \geq -2$
Right boundary:	$(\frac{vh}{2}+1)\leq 2$	$\Leftrightarrow$	$vh \leq 2$

  The equation for the left boundary is even fulfilled with $vh > -2$. If $vh = 2$, all upper subdiagonal entries are zero and the matrix becomes reducible. The convergence of Jacobi can hence be proven for $-2 < vh < 2$.

- For $2 < vh < \infty$:

Inner:	$(\frac{vh}{2}-1)+(\frac{vh}{2}+1)\leq 2$	$\Leftrightarrow$	$vh \leq 2$
Left boundary:	$(\frac{vh}{2}-1)\leq 2$	$\Leftrightarrow$	$vh \leq 6$
Right boundary:	$(\frac{vh}{2}+1)\leq 2$	$\Leftrightarrow$	$vh \leq 2$

  The inequality for the non-boundary equations and the right boundary equation cannot be fulfilled. No convergence can hence be proven for this case.

(c) We will only consider the discretization

$$\frac{du}{dx} \approx \frac{u_{i+1}-u_i}{h}.$$

The system of equation in this case evolves to

Inner:	$(-1+hv)u_{i+1}+(2-hv)u_i+(-1)u_{i-1}$	$=$	$h^2 f_i$
Left boundary:	$(-1+hv)u_{i+1}+(2-hv)u_i$	$=$	$h^2 f_i-(-1)c_0$
Right boundary:	$(2-hv)u_i+(-1)u_{i-1}$	$=$	$h^2 f_i-(-1+hv)c_1$

The Jacobi relaxation for this linear system of equations (for the inner points) is given by:

$$u_i^{n+1} = \frac{1}{(2-hv)}\left(h^2 f_i-(-1+hv)u_{i+1}^n-(-1)u_{i-1}^n\right)$$

The conditions for diagonal dominance in this case read:

Inner:	$\lvert hv-1\rvert+1 \leq \lvert vh-2\rvert$
Left boundary:	$\lvert hv-1\rvert \leq \lvert vh-2\rvert$
Right boundary:	$1 \leq \lvert vh-2\rvert$

We can again consider different cases for v:

- $-\infty < vh \leq 1$:

Inner:	$-(hv-1)+1\leq -(vh-2)$	$\Leftrightarrow$	$0 \leq 0$
Left boundary:	$-(hv-1)\leq -(vh-2)$	$\Leftrightarrow$	$0 \leq 1$
Right boundary:	$1\leq -(vh-2)$	$\Leftrightarrow$	$vh \leq 1$

All inequalities are fulfilled and the inequality for the left boundary is even fulfilled for "$<$". We thus obtain convergence of the Jacobi relaxation over a very wide range of vh, that is for $vh \in (-\infty, 1)$.

- $1 < vh \leq 2$:

$$\begin{array}{lll} \text{Inner:} & (hv-1)+1 \leq -(vh-2) & \Leftrightarrow \quad vh \leq 1 \\ \text{Left boundary:} & (hv-1) \leq -(vh-2) & \Leftrightarrow \quad vh \leq \frac{3}{2} \\ \text{Right boundary:} & 1 \leq -(vh-2) & \Leftrightarrow \quad vh \leq 1 \end{array}$$

The conditions for the points in the inner part of the domain and the point near the right boundary cannot be fulfilled. We can hence not conclude convergence.

- $2 < vh < \infty$:

$$\begin{array}{lll} \text{Inner:} & (hv-1)+1 \leq (vh-2) & \Leftrightarrow \quad 0 \leq -2 \\ \text{Left boundary:} & (hv-1) \leq (vh-2) & \Leftrightarrow \quad 0 \leq -1 \\ \text{Right boundary:} & 1 \leq (vh-2) & \Leftrightarrow \quad vh \geq 3 \end{array}$$

The conditions for points in the inner part of the domain and the left boundary cannot be fulfilled. We can hence not conclude convergence.

Remarks:

- We see that the first-order discretization of the convective term yields a larger range of velocities v for which the Jacobi method must converge. If we have to deal with velocities $v > 0$, we can apply the one-sided difference

$$\frac{du}{dx} \approx \frac{u_i - u_{i-1}}{h}$$

instead of the one from above and obtain similar results.

- From the worksheet, we only know the direction "If A is irreducible and diagonally dominant, then we have convergence". This does not imply that the Jacobi method will definitely not converge for all other cases. More analysis is required for those cases. One may for example analyze the iteration matrix of the Jacobi method and check whether its eigenvalues are within the range (-1,1).

- The conditions that we derived for the Jacobi iteration to converge can also be considered from a more general point of view: assume that the signs of the subdiagonal entries in a single row of our tridiagonal matrix A and the sign of the corresponding diagonal entry are different (for example, $A_{ii} = 2$, $A_{ii-1} = -1$, $A_{ii+1} = -1$). If both values $x_{i-1}^{(n)}$, $x_{i+1}^{(n)}$ are greater/ smaller than zero, then the value $x_i^{(n+1)}$ is greater/ smaller than zero as well (if we don't take the right-hand side b_i into consideration at this point). In this context, the method preserves positivity.

(H) Exercise 2: Runge-Kutta for the Heat Equation

The following partial differential equation describes the distribution of the temperature T in a stick:

$$\frac{\partial T}{\partial t} = D\frac{\partial^2 T}{\partial x^2}, \quad x \in (0,1)$$

The constant D is the thermal diffusivity and describes how fast the temperature can diffuse within the stick. We further assume that the temperature of the stick at the outer ends is known, that is $T(t, x = 0) = T_0$, $T(t, x = 1) = T_N$.

(a) Review from the lecture: Apply the symmetric finite difference approximation for the second-order spatial derivative similar to exercise 1 (a). Which kind of differential equations remains?

(b) Discretize the new differential equations from (a) using the method of Heun.

(c) Write a python script that solves the discrete problem with $D = 1$, a time step $\tau = 0.001$, an initial temperature distribution $T(t = 0, x) = 0$ in the inner part of the stick and temperature values $T_0 = 0$, $T_1 = 1$. Use a mesh sizes $h = 1/20$ for the spatial discretization and plot the result after 1, 10, 100 and 1000 time steps.

Solution:

(a) We introduce $N + 1$ grid points along the stick and a mesh size $h := \frac{1}{N}$. At each point $x_i := i \cdot h$, we define a temperature value $T_i(t) := T(t, x_i)$. Evaluating the partial differential equation in each point x_i and using the symmetric second-order approximation for the second-order derivative yields

$$\frac{dT_i(t)}{dt} = D \cdot \frac{T_{i+1}(t) - 2T_i(t) + T_{i-1}(t)}{h^2}, \quad i = 1, ..., N-1 \tag{3}$$

with T_0 and T_N prescribed at the outer boundary points as assumed in this exercise. We hence obtain a *system of ordinary differential equations* for the local temperature values $T_i(t)$ in the inner part of the domain.

(b) We introduce a matrix $A \in \mathbb{R}^{N-1 \times N-1}$ and a vector $b \in \mathbb{R}^{N-1}$:

$$A = \frac{D}{h^2}\begin{pmatrix} -2 & 1 & 0 & \dots & \dots & \dots & 0 \\ 1 & -2 & 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & -2 & 1 & 0 & \dots & 0 \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ 0 & \dots & \dots & 0 & 1 & -2 & 1 \\ 0 & \dots & \dots & \dots & 0 & 1 & -2 \end{pmatrix}, \quad b = \frac{D}{h^2}\begin{pmatrix} T_0(t) \\ 0 \\ \vdots \\ 0 \\ T_N(t) \end{pmatrix} = \frac{D}{h^2}\begin{pmatrix} T_0 \\ 0 \\ \vdots \\ 0 \\ T_N \end{pmatrix}$$

Using A and b, we can write the system of ODEs from equation (3) as follows:

$$\frac{d}{dt}\begin{pmatrix} T_1(t) \\ \vdots \\ T_{N-1}(t) \end{pmatrix} = A \cdot \begin{pmatrix} T_1(t) \\ \vdots \\ T_{N-1}(t) \end{pmatrix} + b$$

Let's define a time step τ and the vector $\vec{T}^n := (T_1(n\tau), ..., T_{N-1}(n\tau))^\top$. Plugging in the method of Heun (see for example the slides 05_ode_numerics.pdf) for the system of ordinary differential equations from above yields:

$$\begin{aligned}
\vec{T}_I^n &= \vec{T}^n \\
\vec{T}_{II}^n &= \vec{T}^n + \tau\left(A\vec{T}_I^n + b\right) \\
&= \vec{T}^n + \tau\left(A\vec{T}^n + b\right) \\
\vec{T}^{n+1} &= \vec{T}^n + \frac{\tau}{2}\left(A\vec{T}_I^n + b + A\vec{T}_{II}^n + b\right) \\
&= \vec{T}^n + \frac{\tau}{2}\left(A\vec{T}^n + b + A\left(\vec{T}^n + \tau\left(A\vec{T}^n + b\right)\right) + b\right) \\
&= \dots \\
&= \vec{T}^n + \tau\left(A\vec{T}^n + b\right) + \frac{\tau^2}{2}A\left(A\vec{T}^n + b\right)
\end{aligned}$$

(c) See ws9_ex2.py. From the latter formula, we see that we need at least two vectors: one vector to store the temperature $\vec{T}^n$ for one time step and another vector to store $A\vec{T}^n + b$. All the rows of the matrix A—except for the very first and last row—are identical. We hence do not need to store the matrix explicitly, but can apply the respective matrix entries directly! Since the storage of matrices may require huge amounts of memory, matrix-free computations should be preferred in most cases. The latter vector is denoted as *rhsvec* in the python script. After the initialization of all parameters and the vectors, we do the following steps per time step:

- Compute $A\vec{T}^n + b$. We split this computation into the evaluation close the boundaries and the inner part of the computational domain. Alternatively, one could enlarge $\vec{T}^n$ by two entries (one at the very beginning and one entry at the very end) which contain the boundary values. Then, we can apply the inner stencil for all entries with one loop and do not have to use the separate boundary rules.
- Compute the new temperature values. Similar to the case of the vector *rhsvec*, we have modified matrix-vector products for the points close to the boundary.

We finally obtain (after 1000 time steps) an approx. linear distribution of the temperature in the stick.

(I) Exercise 3: Discretization of the Laplace Equation

Consider the Laplace equation

$$\Delta u = 0 \quad \text{in} \quad (0;1)^2, \tag{4}$$

with Neumann boundary conditions

$$\frac{\partial u}{\partial \vec{n}} = 0 \quad \text{at} \quad \{0\} \times [0;1] \cup \{1\} \times [0;1] \cup [0;1] \times \{0\} \cup [0;1] \times \{1\}$$

and the following finite difference discretization on a square grid, see Figure 1.

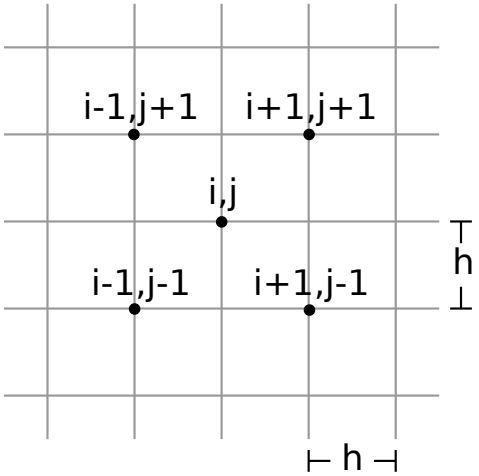

Figure 1: Finite difference discretization on a square grid.

$$\Delta u(x_{i,j}) \approx \frac{u_{i-1,j-1} + u_{i-1,j+1} - 4u_{i,j} + u_{i+1,j-1} + u_{i+1,j+1}}{2h^2} \tag{5}$$

at all inner grid points $x_{i,j}$. For the first order derivative on the boundary nodes use the central difference discretization:

$$\begin{aligned} u_{x_1}(x_{i,j}) &\approx \frac{u_{i+1,j} - u_{i-1,j}}{2h} \\ u_{x_2}(x_{i,j}) &\approx \frac{u_{i,j+1} - u_{i,j-1}}{2h} \end{aligned} \tag{6}$$

(a) Is this discretization consistent?

(b) Does the discretization scheme lead to a unique physical solution? Hint: try to find an oscillating pattern in the sample grid in Figure 2, which satisfies the boundary condition and is annihilated by the discrete operator.

Remark: the solution of the continuous problem is unique only up to a constant function, but oscillations cannot occur in the exact solution.

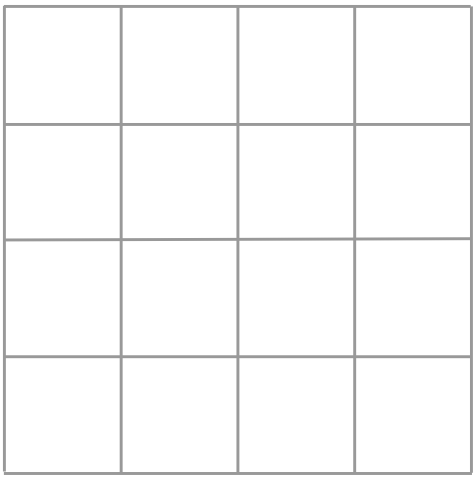

Figure 2: Sample computational grid to try different patterns.

Solution:

(a) Taylor-expansion of u with respect to $x_{i,j}$:

$$\begin{aligned}
u(x_{i+1,j+1}) &= u(x_{i,j}) + h \cdot (u_{x_1}(x_{i,j}) + u_{x_2}(x_{i,j})) + \\
&\quad \frac{h^2}{2}(u_{x_1x_1}(x_{i,j}) + 2u_{x_1x_2}(x_{i,j}) + u_{x_2x_2}(x_{i,j})) + O(h^3), \\
u(x_{i-1,j+1}) &= u(x_{i,j}) + h \cdot (-u_{x_1}(x_{i,j}) + u_{x_2}(x_{i,j})) + \\
&\quad \frac{h^2}{2}(u_{x_1x_1}(x_{i,j}) - 2u_{x_1x_2}(x_{i,j}) + u_{x_2x_2}(x_{i,j})) + O(h^3), \\
u(x_{i+1,j-1}) &= u(x_{i,j}) + h \cdot (u_{x_1}(x_{i,j}) - u_{x_2}(x_{i,j})) + \\
&\quad \frac{h^2}{2}(u_{x_1x_1}(x_{i,j}) - 2u_{x_1x_2}(x_{i,j}) + u_{x_2x_2}(x_{i,j})) + O(h^3), \\
u(x_{i-1,j-1}) &= u(x_{i,j}) + h \cdot (-u_{x_1}(x_{i,j}) - u_{x_2}(x_{i,j})) + \\
&\quad \frac{h^2}{2}(u_{x_1x_1}(x_{i,j}) + 2u_{x_1x_2}(x_{i,j}) + u_{x_2x_2}(x_{i,j})) + O(h^3).
\end{aligned}$$

$$\begin{aligned}
\Rightarrow \quad & u(x_{i+1,j+1}) + u(x_{i-1,j+1}) + u(x_{i+1,j-1}) + u(x_{i-1,j-1}) = \\
& 4(u(x_{i,j}) + \frac{h^2}{2}(u_{x_1x_1}(x_{i,j}) + u_{x_2x_2}(x_{i,j}))) + O(h^3).
\end{aligned}$$

$$\Rightarrow \quad \Delta u(x_{i,j}) - \frac{u(x_{i-1,j-1}) + u(x_{i-1,j+1}) - 4u(x_{i,j}) + u(x_{i+1,j-1}) + u(x_{i+1,j+1})}{2h^2} =$$

$O(h) \to 0$ for $h \to 0$.

And for the Neumann boundary condition:

$$\begin{aligned}
u(x_{i+1,j}) &= u(x_{i,j}) + h \cdot u_{x_1}(x_{i,j}) + \frac{h^2}{2} \cdot u_{x_1x_1}(x_{i,j}) + O(h^3) \\
u(x_{i-1,j}) &= u(x_{i,j}) - h \cdot u_{x_1}(x_{i,j}) + \frac{h^2}{2} \cdot u_{x_1x_1}(x_{i,j}) + O(h^3)
\end{aligned}$$

$$\Rightarrow \qquad u_{x_1}(x_{i,j}) - \frac{u(x_{i+1,j}) - u(x_{i-1,j+1})}{2h} =$$
$$O(h^2) \to 0 \text{ for } h \to 0.$$

$\Rightarrow$ The discretization is consistent.

(b) Consider an oscillating pattern in Figure 3

1	0	1	0	1
0	1	0	1	0
1	0	1	0	1
0	1	0	1	0
1	0	1	0	1

Figure 3: Oscillating pattern annihilated by the discrete operator.

As there is an oscillating pattern, which is annihilated by the operator and fulfilling the boundary conditions, the discretization can yield unphysical oscillating solutions.

(H*) Exercise 4: Flow Modeling

The incompressible Navier-Stokes equations for a two-dimensional flow with the vertical velocity component (y - direction) equal zero takes the form

$$\frac{\partial u}{\partial t} - C\frac{\partial^2 u}{\partial y^2} = g(t) \tag{7}$$

where u is the fluid velocity horizontal component (x - direction), $g(t)$ is the pressure gradient $-\partial p/\partial x$.

(a) Develop a numerical scheme to solve equation (7): use a first-order finite difference for the temporal derivative, a time-implicit, spatial second-order finite difference discretization for the spatial second-order derivative and a time-explicit evaluation of the function $g(t)$. You may further assume an equidistant discretization of the unit interval in space, $y \in [0, 1]$, using $N + 1$ grid points and a resulting mesh size $h := 1/N$. The time step shall be denoted by τ, the discrete velocity values for $u(t, y)$ by $u_i^n := u(n\tau, ih)$, $i = 0, ..., N$, and the discrete representation of the function $g(t)$ by $g^n := g(n\tau)$.

Formulate the relation at one particular time step $n \to n+1$ in form of a linear system of equations

$$\sum_{j=1}^{N} A_{ij} u_j^{n+1} = b_i^n, \quad i = 0, ...N, \tag{8}$$

with matrix $(A_{ij}) \in \mathbb{R}^{N+1 \times N+1}$. The term b_i^n should contain all contributions from the previous time step n. Further assume homogeneous Dirichlet conditions at the outer boundaries, i.e. $u_0^{n+1} = 0$, $u_N^{n+1} = 0$. Give an exact definition of the matrix entries A_{ij} and right hand side entries b_i^n.

(b) Sketch the individual steps of the time stepping scheme to solve the finite difference formulation from (a) in pseudo-code.

Solution:

(a) The first-order time-implicit scheme is given by the implicit Euler method, the second-order spatial discretization corresponds to the default three-point stencil for the 1D Laplacian:

$$\frac{u_i^{n+1} - u_i^n}{\tau} - C \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{h^2} = g^n \tag{9}$$

Splitting the equation from above into information from time step n and $n+1$ results in

$$-\frac{C}{h^2} u_{i-1}^{n+1} + \left(\frac{1}{\tau} + \frac{2C}{h^2} \right) u_i^{n+1} - \frac{C}{h^2} u_{i+1}^{n+1} = g^n + \frac{1}{\tau} u_i^n \tag{10}$$

for all inner points $i = 1, ..., N-1$. The matrix A can thus be defined as

$$A_{ij} := \begin{cases} -\frac{C}{h^2} & |j-i| = 1 \text{ and } i \neq 0, N \\ \frac{1}{\tau} + \frac{2C}{h^2} & \text{if } \quad i = j \text{ and } i \neq 0, N \\ 1 & i = j \text{ and } (i = 0 \vee i = N) \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

The right hand side b_i^n evolves at $b_i^n := g^n + \frac{1}{\tau} u_i^n$ for $i = 1, ..., N-1$ and $b_0^n = 0$, $b_N^n = 0$.

(b) The algorithm reads:

```
for t=0; t<t_end; t+dt
  compute g at t
  assemble b
  compute u = A^{-1} b
end
```