# Fys4150
# Project 5

Peter Killingstad and Karl Jacobsen

https://github.com/kaaja/fys4150

November 24, 2017

## Note to instructurs reagarding Github repository

If the above Github-link does not work, it is eighter because you have not yet accepted our invite to the repository, or you have not yet provided us with an e-mail addres available at Github so that we can invite you. The Github user you will be invited from is "kaaja". If the latter applies to you, please send us an e-mail with an e-mailadress available in Github or your Github username so that we can send you an invite. Our e-mailadresses: peter.killingstad@hotmail.com, karljaco@gmail.com.

## 1 Analytical solution 1D

The one-dimensional problem is written as

$$\frac{\partial^2 u(x,t)}{\partial x^2} = \frac{\partial u(x,t)}{\partial t},\ t > 0,\ x \in (0, L) \tag{1a}$$

$$u(x,0) = 0,\ 0 < x < 1, \tag{1b}$$
$$u(0,t) = 0,\ t > 0 \tag{1c}$$
$$u(L,t) = 1,\ t > 0, \tag{1d}$$

where the length has been scaled by the lenght of the $x$-domain, $L$.

The problem consists of non-homogeneous boundary conditions, and it is not trivial to find a closed form solution. The problem can be seen as a physical problem such as a temperature gradient in a rod or flow between two infinite flat plates, where the fluid is initially at rest and the plate at $x = 1$ is given a sudden movement.

From physical observations we know that as time goes, i.e. when $t \to \infty$, the problem coinsides whith its surroundings and becomes steady.It is reasonable to introduce a solution consisting of the sum of two parts, a steady-state solution, and a transient solution that depends on the initial conditions:

$$u(x,t) = U(x) + V(x,t) \tag{2}$$

where $U(x)$ is the steady-state solution and $V(x,t)$ is the transient solution.

Since the steady-state problem is not changing in time, i.e. $\partial_t = 0$, it is written in a compact form as:

$$U_{xx} = 0\ ,\ x \in [0,1] \tag{3a}$$

$$U(0) = 0 \tag{3b}$$
$$U(L) = 1 \tag{3c}$$

For the transient part of the problem we have the problem written in compact form as:

$$V_t = V_{xx} \ , \ t > 0 \ , \ x \in (0,1) \tag{4a}$$

$$V(x,0) = -U(x) \ , \ 0 < x < 1 \tag{4b}$$
$$V(0,t) = 0 \ , \ t > 0 \tag{4c}$$
$$V(1,t) = 0 \ , \ t > 0 \tag{4d}$$

The steady state solution is solved by integrating (3a) twice

$$\begin{aligned} \int \frac{\partial^2 u(x,t)}{\partial x^2} \, dx &= A \\ \int \frac{\partial u(x,t)}{\partial x} \, dx &= \int A dx + B \end{aligned}$$

and we end up with the general solution $U(x) = Ax + B$. Applying the boundary conditions at $x = 0$ and $x = 1$ we get that

$$U(x) = x \tag{5}$$

The transient problem (4a) has homogeneous boundary conditions, and we can solve it by seperation of variables. We start by making the anzats that $V(x,t) = T(t)X(x)$ so that we can rewrite (4a) as:

$$XT' = X''T \tag{6}$$

Reordering the equation we get the following:

$$\frac{T'}{T} = \frac{X''}{X} = k, \tag{7}$$

where $k$ is an unknown constant. The reason we can insert $k$ into the above equation, is that the sides depends on different varirables, and so equality between the sides is only acheived if both sides equals a constant. We have homogenuous BCs, and for this to be achieved when we solve the $X$-part of the above equaion, we must have $k < 0$. As can be seen later, with $k \geq 0$ we would not be able to satisfy the homogenupus BCs. For computationaly ease, we set $k = -\lambda^2$, where $\lambda > 0$.

Starting with the $t$ dependent part, we get

$$\frac{T'}{T} = -\lambda^2 \tag{8}$$
$$T' = -\lambda^2 T \tag{9}$$

The ODE can be solved by separation of variables

$$\frac{1}{T}\frac{dT}{dt} = -\lambda^2 \tag{10a}$$
$$\Rightarrow \frac{1}{T} dT = -\lambda^2 dt \tag{10b}$$
$$\int \frac{1}{T} dT = -\int \lambda^2 dt \tag{10c}$$
$$\Rightarrow \ln(T) = -\lambda^2 + A \tag{10d}$$

The $t$ dependent term is then given by

$$T = Ae^{-\lambda^2 t} \tag{11}$$

We see that our assumption $\lambda > 0$ makes sense also for the $T$-solution, since we do not get blow-up as $t \to \infty$.

For the $x$ dependent term we get

$$X'' = -\lambda^2 X \tag{12a}$$
$$\Rightarrow X'' + \lambda^2 X = 0 \tag{12b}$$

We use an anzats $X = e^{\alpha x}$:

$$\alpha^2 e^{\alpha x} + \lambda^2 e^{\alpha x} = 0 \tag{13a}$$
$$\Rightarrow \alpha^2 = -\lambda^2 \tag{13b}$$
$$\Rightarrow \alpha = \pm\sqrt{-\lambda^2} \tag{13c}$$

Since we have $\lambda > 0$, we get our wanted trigonometric solution

$$X = Be^{-i\lambda x} + Ce^{i\lambda x}, \tag{14}$$

which can be written as

$$X(x) = B\cos(\lambda x) + C\sin(\lambda x) \tag{15}$$

,
where the constants have changed.

Applying the boundary conditions we get that

$$X(0) = B\cos(0) + C\sin(0) = 0 \tag{16}$$
$$X(1) = B\cos(\lambda) + C\sin(\lambda) = 0 \tag{17}$$

From the first boundary condition wher $x = 0$ we must have that $B = 0$. We are lookong for non-trivial solutions of the problem, so for $x = 1$ we must find the values that gives $C\sin(\lambda) = 0$. We know that $\sin(n\pi) = 0$ *for* $n = 1, 2, \ldots$, which lets us determine that $\lambda = n\pi$.

Summing up, we have the following particular solutions for the transient part:

$$V_n(x,t) = A_n e^{-(n\pi)^2 t}\sin(n\pi x), n = 1, 2, ..., \infty \tag{18a}$$

Since all particular solutions satisfy the PDE, and the BCs are homogenuous, all linear combinations of the particular solutions will also satisfy our problem, giving the general soultion of the transient problem

$$\sum_{n=1}^{\infty} a_n e^{-(n\pi)^2 t}\sin(n\pi x) \tag{19}$$

The coefficients $a_n$ is found by applying the initial condidtion (IC)

$$-U(x) = \sum_{n=1}^{\infty} a_n \sin(n\pi x) \tag{20}$$

To obtain an expression for the coefficients $a_n$ we use that the functions $\sin(n\pi x)$ is orthogonal to each other in the sense that

$$\int_0^1 \sin(n\pi x)\sin(m\pi x)dx = \begin{cases} 0 & \text{if } m \neq n \\ 1/2 & \text{if } m = n \end{cases} \tag{21}$$

Using the above statement, multiplying 20 by $\sin(m\pi x)$ and then integrating over $x$ over the domain, we get

$$
\begin{aligned}
-\int_0^1 U(x)\sin(m\pi x)\,dx &= a_n\int_0^1\sum_{n=1}^{\infty}\sin(m\pi x)\sin(n\pi x)\,dx = a_m\int_0^1\sin^2(m\pi x)\,dx \\
&\Rightarrow \quad -\int_0^1 U(x)\sin(m\pi x) = \frac{a_m}{2} \\
&\Rightarrow \quad a_m = -2\int_0^1 U(x)\sin(m\pi x)\,dx \qquad (22)
\end{aligned}
$$

Solving for $a_n$, applying the steady state solution (5), yields

$$
\begin{aligned}
a_n = -2\int_0^1 x\sin(m\pi x) &= \Big[\frac{2}{(m\pi)^2}\sin(m\pi x) + \frac{2x}{m\pi}\cos(m\pi x)\Big]_0^1 \\
&\Rightarrow \quad a_n = \frac{2}{m\pi}(-1)^m \qquad (23)
\end{aligned}
$$

Putting the coefficient from (23) into the fundamental solutions (19) we end up with the following solution for the transient problem:

$$
V(x,t) = 2\sum_{n=1}^{\infty}\frac{(-1)^k}{n\pi}e^{-(n\pi)^2 t}\sin(n\pi x) \qquad (24)
$$

Now we get an expression for the whole problem by putting combining the steady-state solution (5) and the transient solution (24) such that we obtain

$$
u(x,t) = U(x) + V(x,t) = x + 2\sum_{n=1}^{\infty}e^{-(n\pi)^2 t}\frac{(-1)^k}{n\pi}\sin(n\pi x) \qquad (25)
$$

# 2 Analytical solution 2D

In the two-dimensional case the differential equation becomes

$$
\begin{aligned}
\frac{\partial^2 u(x,y,t)}{\partial x^2} + \frac{\partial^2 u(x,y,t)}{\partial y^2} &= \frac{\partial u(x,y,t)}{\partial t}\ ,\ t>0\ ,\ x,y\in(0,1) && (26a)\\
u(x,y,0) &= 0\ ,\ x,y\in(0,1) && (26b)\\
u(0,y,t) &= 0\ ,\ t>0 && (26c)\\
u(1,y,t) &= 0\ ,\ t>0 && (26d)\\
u(x,0,t) &= 0\ ,\ t>0 && (26e)\\
u(x,1,t) &= 1\ ,\ t>0 && (26f)
\end{aligned}
$$

The boundary conditions is extended in such a way that the physical problem can be that of a flow whitin a box with infinite plates in the streamwise direction ($z$), where the fluid is initially at rest and the plate at $y = 1$ is given a sudden movement. We have "no slip" boundary conditions i.e. the fluid has zero movement on all boundaries except at $y = 1$.

To obtain a closed form solution for the two-dimensional, we use the same argument as for the one-dimensional problem, i.e. we split the solution into a steady-state solution and a transient solution.

We end up with a solution defined as:

$$
u(x,y,t) = U(x,y) + V(x,y,t) \qquad (27)
$$

where $U(x,y)$ is the steady-state solution and $V(x,y,t)$ is the transient solution.

The steady-solution is then written in a compact form as:

$$U_{xx} + U_{yy} = 0 \ , \ x,y \in [0,1] \tag{28a}$$

$$\begin{aligned} U(0,y) &= 0 && (28b)\\ U(1,y) &= 0 && (28c)\\ U(x,0) &= 0 && (28d)\\ U(x,1) &= 1 && (28e) \end{aligned}$$

For the transient problem we have the propblem defined as:

$$V_t = V_{xx} + V_{yy} \ , \ t > 0 \ , \ x \in [0,1] \tag{29a}$$

$$\begin{aligned} V(x,0) &= -U(x,y) \ , \ x,y \in [0,1] && (29b)\\ V(0,y,t) &= 0 \ , \ t > 0 && (29c)\\ V(1,y,t) &= 0 \ , \ t > 0 && (29d)\\ V(x,0,t) &= 0 \ , \ t > 0 && (29e)\\ V(x,1,t) &= 0 \ , \ t > 0 && (29f) \end{aligned}$$

For the steady-state problem we have a 2D Laplacian equation, which we can solve by seperation of variables. We use the anzats that $U(x,y) = X(x)Y(y)$, so that we end up with:

$$\begin{aligned} YX'' + XY'' &= 0\\ \frac{X''}{X} = -\frac{Y''}{Y} &= -\beta^2 \end{aligned}$$

Due to the independence of the terms on each side of the equation, we can solve the equations separately, each equation becoming

$$\frac{X''}{X} = -\beta^2$$

$$\frac{Y''}{Y} = \beta^2$$

The sign on the right hand side in both of the above equations is determined by the fact that in the $x$ dependent term we have homogeneous boundaries, while in the $y$ dependent term we have non-homogeneous. By the same reasoning as in the 1D-case, homogenupus BCs in the $x$-direction, we get $\beta > 0$.

For the $x$ dependent term we get

$$\begin{aligned} X'' &= -\beta^2 X\\ \Rightarrow X'' + \beta^2 X &= 0, \end{aligned}$$

which has a similar solution as that of the transient $x$ dependent term in the one-dimensional case (14). With our choise of $\beta$, we get the wanted trigonometric form

$$X(x) = A\cos(\beta x) + B\sin(\beta x).$$

Applying the boundary conditions, we end up with the term

$$X_n(x) = B_n \sin(n\pi x), \tag{30}$$

were $X_n(x)$ is the family of particular solutions.

The $y$ dependent term can be rewritten as

$$y'' = \beta^2 Y \tag{31a}$$
$$\Rightarrow Y'' - \beta^2 Y = 0 \tag{31b}$$

We are again using an anzats so that $Y = e^{\gamma y}$. This leaves us with the term

$$\gamma^2 e^{\gamma y} - \beta^2 e^{\gamma y} = 0 \tag{32a}$$
$$\Rightarrow \gamma^2 = \beta^2 \tag{32b}$$
$$\Rightarrow \gamma = \pm\beta, \tag{32c}$$

and we end up with the following expression

$$Y_n(y) = C_n e^{-n\pi y} + D_n e^{n\pi y} \tag{33}$$
$$\Rightarrow C_n \cosh(n\pi y) + D_n \sinh(n\pi y) \tag{34}$$

for the family of particular solutions.

The fundamental solutions of the steady-state is the given by

$$U(x,y) = \sum_{n=1}^{\infty} X(x)Y(y) = \sum_{n=1}^{\infty} B_n \sin(n\pi x)\Big(C_n \cosh(n\pi y) + D_n \sinh(n\pi y)\Big) \tag{35}$$

Applying the the boundary of $U(x,0)$ to the equation above we get that $C_n$ must be zero. We are then left with the expression

$$U(x,y) = \sum_{n=1}^{\infty} c_n \sin(n\pi x)\sinh(n\pi y) \tag{36}$$

Applying the last boundary $U(x,1) = 1$ we must have that

$$1 = \sum_{n=1}^{\infty} c_n \sin(n\pi x)\sinh(n\pi) \tag{37}$$

setting $d_n = c_n \sinh(n\pi)$ we get that

$$1 = \sum_{n=1}^{\infty} d_n \sin(n\pi x) \tag{38}$$

Using again that the functions $\sin(n\pi x)$ is orthogonal (21), as in the 1D case, multiplying the above equation with $\sin(m\pi x)$ and then integrating over $x$ over the domain, we get that

$$d_n = 2\int_0^1 \sin(m\pi x)dx = \frac{2}{m\pi}\Big(1 - (-1)^m\Big) \tag{39a}$$
$$\Rightarrow c_n = \frac{2}{m\pi \sinh(m\pi)}\Big(1 - (-1)^m\Big). \tag{39b}$$

Putting the coefficients $c_n$ back into the expression for the fundamental solutions (36), we end up with the following expression for the steady-state

$$U(x,y) = \frac{2}{\pi}\sum_{m=1}^{\infty} \frac{\sin(m\pi x)\sinh(m\pi y)}{\pi \sinh(m\pi)}\Big(1 - (-1)^m\Big) \tag{40a}$$
$$\Rightarrow U(x,y) = \frac{4}{\pi}\sum_{m=1}^{\infty} \frac{\sin((2m-1)\pi x)\sinh((2m-1)\pi y)}{\pi \sinh((2m-1)\pi)}. \tag{40b}$$

For the transient problem we once again use the anzats that $V(x,y,t) = X(x)Y(y)T(t)$ in order to write the problem as

$$XYT' = YTX'' + XTY'' \tag{41a}$$
$$\Rightarrow \frac{T'}{T} = \frac{X''}{X} + \frac{Y''}{Y} = -k^2. \tag{41b}$$

Starting by solving the $t$ dependent term, which is solved in the same way as for the one-dimensional term (10a), we end up with the expression

$$T = Ae^{-k^2 t}. \tag{42}$$

For the $x$ dependent part we have

$$\frac{X''}{X} = -\frac{Y''}{Y} - k^2 \tag{43a}$$
$$\Rightarrow \frac{X''}{X} = -\gamma^2, \tag{43b}$$

where

$$-\gamma^2 = \frac{Y''}{Y} - k^2. \tag{44}$$

(43a) has the same solution form as the $x$ dependent transient one-dimensional term (14), as well as the steady state two-dimensional term (30), so we end up with

$$X(x) = B\cos(\gamma x) + C\sin(\gamma x) \tag{45}$$

For the above equation to hold on the boundaries, we get $B = 0$ and $\gamma = n\pi, n = 1, 2, ..., \infty$, so we end up with

$$X_n(x) = c_n \sin(n\pi x) \tag{46}$$

We are left with the $y$ dependent term, which is written as

$$\frac{Y''}{Y} = k^2 - \gamma^2 \tag{47a}$$

Solving for $Y$ we again use the anzats $Y = e^{\alpha y}$ and obtain the term

$$\alpha^2 e^{\alpha y} + (\gamma^2 - k^2)e^{\alpha y} = 0 \tag{48a}$$
$$\alpha^2 + \gamma^2 - k^2 = 0 \tag{48b}$$
$$\Rightarrow \alpha = \pm\sqrt{k^2 - \gamma^2} \tag{48c}$$

We are looking for a solution on the form that satisfies the homogenupus BCs for the transient term, resulting in

$$Y = D\cos(\sqrt{k^2 - \gamma^2}y) + E\sin(\sqrt{k^2 - \gamma^2}y) \tag{49}$$

For the above equation to satisfy the boundary conditions $Y(0) = 0$ and $Y(1) = 0$, we get that $D = 0$ and that $\sqrt{k^2 - \gamma^2} = m\pi,\ m = 1, 2, ...\infty$. The $Y$ term is then given by

$$Y_m(y) = E_m \sin(m\pi y) \tag{50}$$

We can now calculate the constant $k^2$, $k^2 = \gamma^2 + (m\pi)^2 = (n\pi)^2 + (m\pi)^2$. Putting this back into the $t$ dependent transient term (42) we have that

$$T = A_{nm}e^{-\pi^2(m^2+n^2)t}. \tag{51}$$

Combining the $x, y$ and $t$ dependent terms ((46), (50) and (51) respectively), we end up with the fundamental solution

$$V(x,y,t) = \sum_{n=1}^{\infty}\sum_{m=1}^{\infty} A_{mn}\sin(n\pi x)\sin(m\pi y)e^{-\pi^2(m^2+n^2)t}. \tag{52}$$

To find the coefficients $A_{nm}$, we use the initial condition

$$-U(x,y) = \sum_{n=1}^{\infty}\sum_{m=1}^{\infty} A_{mn}\sin(n\pi x)\sin(m\pi y) \tag{53}$$

We can solve for the coefficient by again using the orthogonality from (21), multiplying the equation above by $\sin(p\pi x)\sin(q\pi y)$, and integrating over the $x$ and $y$ domain to get

$$\int_0^1\int_0^1 -U(x,y)\sin(p\pi x)\sin(q\pi y)dxdy = \frac{A_{pq}}{4} \tag{54a}$$

$$\Rightarrow A_{pq} = -4\int_0^1\int_0^1 U(x,y)\sin(p\pi x)\sin(qn\pi y)\ dxdy. \tag{54b}$$

Finally we can combine the steady-state solution and the transient solution in order to get the analytical expression

$$\begin{aligned} u(x,y,t) &= U(x,y) + V(x,y,t) \\ &= U(x,y) + \sum_{n=1}^{\infty}\sum_{m=1}^{\infty} A_{mn}\sin(n\pi x)\sin(m\pi y)e^{-\pi^2(m^2+n^2)t}, \end{aligned} \tag{55}$$

where

$$U(x,y) = \frac{4}{\pi}\sum_{m=1}^{\infty}\frac{\sin((2m-1)\pi x)\sinh((2m-1)\pi y)}{\pi\sinh((2m-1)\pi)} \tag{56}$$

and

$$A_{mn} = -4\int_0^1\int_0^1 U(x,y)\sin(m\pi x)\sin(n\pi y)dxdy. \tag{57}$$

# 3 5a

## 3.1 Derivation of schemes with truncation errors

All the schemes will be derived from Taylor series expansions, and the truncation error will be related to the remainder in the Taylor-series expansions. This remainder is the error we get when truncating the series by leaving out the remainder.

## 3.2 Forward Euler

For the time derivative, we expand $u(x,t+\Delta t)$ around $t$

$$u(x,t+\Delta t) = u(x,t) + u_t(x,t)\Delta t + \mathcal{O}(\Delta t^2) \tag{58a}$$

$$\rightarrow u_t(x,t) = \frac{u(x,t+\Delta t) - u(x,t)}{\Delta t} + \mathcal{O}(\Delta t) \tag{58b}$$

The space derivative, which is a 2nd derivative, we derive by combining two Taylor series'

$$u(x+\Delta x,t) = u(x,t) + u_x(x,t)\Delta x + \frac{u_{xx}(x,t)\Delta x^2}{2} + \frac{u_{xxx}(x,t)\Delta x^3}{6} + \mathcal{O}(\Delta x^4) \tag{59a}$$

$$u(x-\Delta x,t) = u(x,t) - u_x(x,t)\Delta x + \frac{u_{xx}(x,t)\Delta x^2}{2} - \frac{u_{xxx}(x,t)\Delta x^3}{6} + \mathcal{O}(\Delta x^4) \tag{59b}$$

Now we add (59a) and (59b) and solve for $u_{xx}(x,t)$

$$\begin{aligned}
\Big(u(x+\Delta x,t)+u(x-\Delta x,t)\Big) &= \Big(u(x,t)+u(x,t)\Big) \\
&+ \Big(u_x(x,t)\Delta x + (-u_x(x,t)\Delta x)\Big) \\
&+ \Big(\frac{u_{xx}(x,t)\Delta x^2}{2} + \frac{u_{xx}(x,t)\Delta x^2}{2}\Big) \qquad (60a)\\
&+ \Big(\frac{u_{xxx}(x,t)\Delta x^3}{6} + (-\frac{u_{xxx}(x,t)\Delta x^3}{6})\Big) \\
&+ \Big(\mathcal{O}(\Delta x^4) + \mathcal{O}(\Delta x^4)\Big) \\
&= 2u(x,t) + u_{xx}(x,t)\Delta x^2 + \mathcal{O}(\Delta x^4) \qquad (60b)\\
\rightarrow u_{xx}(x,t) &= \frac{u(x-\Delta x,t) - 2u(x,t) + u(x+\Delta x,t)}{\Delta x^2} + \mathcal{O}(\Delta x^2) \qquad (60c)
\end{aligned}$$

Combining (58b) and (60c) we get the Forward Euler scheme

$$u_t(x,t) = u_{xx}(x,t) \qquad (61a)$$

$$\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} + \mathcal{O}(\Delta t) = \frac{u(x-\Delta x,t) - 2u(x,t) + u(x+\Delta x,t)}{\Delta x^2} + \mathcal{O}(\Delta x^2) \qquad (61b)$$

From (61b) we see that the scheme has a truncation error that goes like $\mathcal{O}(\Delta t)$ in time and $\mathcal{O}(\Delta x^2)$ in space.

We will analyze the stability of the Forward Euler scheme (61b) by applying Neuman stability analysis. From the analytical solution of the problem, we know that the particular solutions are on the form $u = e^{-(k\pi)^2 t}e^{ik\pi x}$, where $k$ is an integer greater than one. We observe that the solutions are stable in $t$, meaning that the solutions do not blow up as $t$ increaes. Based on the analytical particular solution, we make the numerical ansatz

$$u = a_k^n e^{ik\pi x_j} \qquad (62)$$

For the numerical ansatz (62) to reproduce the characteristics of the analytical particular solution, with stability in $t$, we observe that $|a_k^n| < 1$ is necessary. We now plug in the ansatz (62) into the (61b) and derive an equation for $|a_k^n|$:

$$\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} = \frac{u(x-\Delta x,t) - 2u(x,t) + u(x+\Delta x,t)}{\Delta x^2} \qquad (63a)$$

$$\frac{a_k^{n+1}e^{ik\pi(j+1)\Delta x} - a_k^n e^{ik\pi j\Delta x}}{\Delta t} = \frac{a_k^n e^{ik\pi(j-1)\Delta x} - 2a_k^n e^{ik\pi j\Delta x} + a_k^n e^{ik\pi(j+1)\Delta x}}{\Delta x^2} \qquad (63b)$$

$$a_k^n e^{ik\pi j\Delta x}\ \frac{a_k - 1}{\Delta t} = a_k^n e^{ik\pi j\Delta x}\ \frac{e^{-ik\pi\Delta x} - 2 + e^{ik\pi\Delta x}}{\Delta x^2} \qquad (63c)$$

$$\frac{a_k - 1}{\Delta t} = \frac{e^{-ik\pi\Delta x} - 2 + e^{ik\pi\Delta x}}{\Delta x^2} \qquad (63d)$$

$$\begin{aligned}
a_k &= 1 + \frac{\Delta t}{\Delta x^2}(e^{-ik\pi\Delta x} - 2 + e^{ik\pi\Delta x}) && (63e)\\
&= 1 + \frac{\Delta t}{\Delta x^2}\Big(2\cos(k\pi\Delta x) - 2\Big) && (63f)\\
&= 1 + 2\frac{\Delta t}{\Delta x^2}\Big(\cos(k\pi\Delta x) - 1\Big) && (63g)\\
&= 1 + 2\frac{\Delta t}{\Delta x^2}\Big(-2\sin^2(\frac{k\pi\Delta x}{2})\Big) && (63h)\\
&= 1 - 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2}) && (63i)\\
|a_k| &= |1 - 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})| && (63j)
\end{aligned}$$

From (63j) we get

$$|a_k| < 1 \text{ if } ||1 - 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})|| < 1 \tag{64a}$$

$$\rightarrow |1 - 4\frac{\Delta t}{\Delta x^2}| < 1 \rightarrow |a_k| < 1 \text{ (Since } \sin^2(k\pi\Delta x/2)_{max} = 1) \tag{64b}$$

$$\rightarrow 1 - 4\frac{\Delta t}{\Delta x^2} > -1 \tag{64c}$$

$$\rightarrow \frac{\Delta t}{\Delta x^2} < \frac{1}{2} \tag{64d}$$

(64d) gives that the Forward Euler scheme is conditionally stable, and the condition that ensures stability.

### 3.3 Backward Euler

Here we will do the same as we did for Forward Euler above: Derive the scheme, including truncation errors, and analyze stability.

The only change compared to Forward Euler, is the time discretization, which now becomes

$$u(x, t - \Delta t) = u(x, t) + u_t(x, t)\Delta t - \mathcal{O}(\Delta t^2) \tag{65a}$$

$$\rightarrow u_t(x, t) = \frac{u(x, t) - u(x, t - \Delta t)}{\Delta t} + \mathcal{O}(\Delta t) \tag{65b}$$

The space discretization is the same as for Forward Euler, (60c). Combining the space discretization (60c) and (65b) gives

$$\frac{u(x, t) - u(x, t - \Delta t)}{\Delta t} + \mathcal{O}(\Delta t) = \frac{u(x - \Delta x, t) - 2u(x, t) + u(x + \Delta x, t)}{\Delta x^2} + \mathcal{O}(\Delta x^2) \tag{66a}$$

We note that the truncation errors have the same asymptoptic behavior as for the Forward Euler scheme.

To analyze the stability of the Backward Euler scheme, we apply the same method as we did for Forward Euler and insert the ansatz (62) into (66a) to get

$$\frac{u(x, t) - u(x, t - \Delta t)}{\Delta t} = \frac{u(x - \Delta x, t) - 2u(x, t) + u(x + \Delta x, t)}{\Delta x^2} \tag{67a}$$

$$\frac{a_k^n e^{ik\pi j\Delta x} - a_k^{n-1} e^{ik\pi j\Delta x}}{\Delta t} = \frac{a_k^n e^{ik\pi(j-1)\Delta x} - 2a_k^n e^{ik\pi j\Delta x} + a_k^n e^{ik\pi(1+j)\Delta x}}{\Delta x^2} \tag{67b}$$

$$a_k^n e^{ik\pi j\Delta x}\ \frac{1 - a_k^{-1}}{\Delta t} = a_k^n e^{ik\pi j\Delta x}\ \frac{e^{-ik\pi\Delta x} - 2 + e^{ik\pi\Delta x}}{\Delta x^2} \tag{67c}$$

$$\frac{1 - a_k^{-1}}{\Delta t} = \frac{e^{-ik\pi\Delta x} - 2 + e^{ik\pi\Delta x}}{\Delta x^2} \tag{67d}$$

$$a_k^{-1} = 1 - \frac{\Delta t}{\Delta x^2}(e^{-ik\pi\Delta x} - 2 + e^{ik\pi\Delta x}) \tag{67e}$$

$$a_k = \frac{1}{1 - \frac{\Delta t}{\Delta x^2}(e^{-ik\pi\Delta x} - 2 + e^{ik\pi\Delta x})} \tag{67f}$$

$$\overset{(63)}{=} \frac{1}{1 + 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})} \tag{67g}$$

$$|a_k| = \left|\frac{1}{1 + 4\frac{\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})}\right| < 1. \tag{67h}$$

(67i)

From (67h) we see that, in contrast to the Forward Euler scheme, the Backward Euler scheme is unconditionally stable.

(66a) reveals another difference between the Backward Euler scheme and the Forward Euler scheme: (66a) is implicit in $u(x,t)$, meaning that we cannot solve (66a) directly for $u(x,t)$, as we did in the Forward Euler scheme. However, we can find $u(x,t)$ from (66a) by recognizing that (66a) can be rewritten as a linear system:

$$\frac{u(x,t)-u(x,t-\Delta t)}{\Delta t} = \frac{u(x-\Delta x,t)-2u(x,t)+u(x+\Delta x,t)}{\Delta x^2} \tag{68a}$$

$$\frac{u_i^n-u_i^{n-1}}{\Delta t} = \frac{u_{i-1}^n-2u_i^n+u_{i+1}^n}{\Delta x^2} \tag{68b}$$

$$\frac{\Delta x^2}{\Delta t}(u_i^n-u_i^{n-1}) = u_{i-1}^n-2u_i^n+u_{i+1}^n \tag{68c}$$

$$-\Big(u_{i-1}^n-(2+\frac{\Delta x^2}{\Delta t})u_i^n+u_{i+1}^n\Big) = \frac{\Delta x^2}{\Delta t}u_i^{n-1} \tag{68d}$$

$$\Big(-u_{i-1}^n+(2+\frac{\Delta x^2}{\Delta t})u_i^n-u_{i+1}^n\Big) = \frac{\Delta x^2}{\Delta t}u_i^{n-1} \tag{68e}$$

$$\underbrace{\begin{bmatrix} 2+\frac{\Delta x^2}{\Delta t} & -1 & \cdots & 0 \\ -1 & 2+\frac{\Delta x^2}{\Delta t} & -1 & \vdots \\ \vdots & & \ddots & \vdots \\ 0 & \cdots & -1 & 2+\frac{\Delta x^2}{\Delta t} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} u_1^n \\ u_2^n \\ \vdots \\ u_N^n \end{bmatrix}}_{\mathbf{U}} = \underbrace{\frac{\Delta x^2}{\Delta t}\begin{bmatrix} u_0^{n-1} \\ u_1^{n-1} \\ \vdots \\ u_N^{n-1} \end{bmatrix}}_{\tilde{\mathbf{b}}} \tag{68f}$$

We see from (68f) that solving Backward Euler corresponds to solving a linear system $AU=\tilde{b}$, where $A$ is a trdiagonal matrix.

## 3.4 Crank-Nicolson

Here we Taylor expand $u(x+\Delta x,t+\Delta t)$ and $u(x-\Delta x,t+\Delta t)$ around $t'=t+\Delta t/2$ to get

$$\begin{aligned} u(x+\Delta x,t+\Delta t) &= u(x,t') + \frac{\partial u(x,t')}{\partial x}\Delta x + \frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2} + \frac{\partial^2 u(x,t')}{2\partial x^2}\Delta x^2 + \frac{\partial^3 u(x,t')}{6\partial x^3}\Delta x^3 \\ &\quad + \frac{\partial^2 u(x,t')}{2\partial t^2}\frac{\Delta t^2}{4} + \frac{\partial^2 u(x,t')}{\partial x\partial t}\frac{\Delta t}{2}\Delta x + \mathcal{O}(\Delta t^3) + \mathcal{O}(\Delta x^4) \end{aligned} \tag{69a}$$

$$\begin{aligned} u(x-\Delta x,t+\Delta t) &= u(x,t') - \frac{\partial u(x,t')}{\partial x}\Delta x + \frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2} + \frac{\partial^2 u(x,t')}{2\partial x^2}\Delta x^2 - \frac{\partial^3 u(x,t')}{6\partial x^3}\Delta x^3 \\ &\quad + \frac{\partial^2 u(x,t')}{2\partial t^2}\frac{\Delta t^2}{4} - \frac{\partial^2 u(x,t')}{\partial x\partial t}\frac{\Delta t}{2}\Delta x + \mathcal{O}(\Delta t^3) + \mathcal{O}(\Delta x^4) \end{aligned} \tag{69b}$$

$$\begin{aligned} u(x+\Delta x,t) &= u(x,t') + \frac{\partial u(x,t')}{\partial x}\Delta x - \frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2} + \frac{\partial^2 u(x,t')}{2\partial x^2}\Delta x^2 + \frac{\partial^3 u(x,t')}{6\partial x^3}\Delta x^3 \\ &\quad + \frac{\partial^2 u(x,t')}{2\partial t^2}\frac{\Delta t^2}{4} - \frac{\partial^2 u(x,t')}{\partial x\partial t}\frac{\Delta t}{2}\Delta x + \mathcal{O}(\Delta t^3) + \mathcal{O}(\Delta x^4) \end{aligned} \tag{69c}$$

$$\begin{aligned} u(x-\Delta x,t) &= u(x,t') - \frac{\partial u(x,t')}{\partial x}\Delta x - \frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2} + \frac{\partial^2 u(x,t')}{2\partial x^2}\Delta x^2 - \frac{\partial^3 u(x,t')}{6\partial x^3}\Delta x^3 \\ &\quad + \frac{\partial^2 u(x,t')}{2\partial t^2}\frac{\Delta t^2}{4} + \frac{\partial^2 u(x,t')}{\partial x\partial t}\frac{\Delta t}{2}\Delta x + \mathcal{O}(\Delta t^3) + \mathcal{O}(\Delta x^4) \end{aligned} \tag{69d}$$

$$u(x,t+\Delta t) = u(x,t') + \frac{\partial u(x,t')}{\partial t}\frac{\Delta_t}{2} + \frac{\partial^2 u(x,t')}{2\partial t^2}\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{69e}$$

$$u(x,t) = u(x,t') - \frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2} + \frac{\partial^2 u(x,t')}{2\partial t^2}\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{69f}$$

The above formulae are taken from Hjorth-Jensen's slides [2].

Combining (69e) and (69f) gives the time derivative

$$\begin{aligned}\Big(u(x,t+\Delta t)-u(x,t)\Big) &= \Big(u(x,t')-u(x,t')\Big)+\Big(\frac{\partial u(x,t')}{\partial t}\frac{\Delta_t}{2}-(-\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2})\Big)\\ &\quad+\Big(\frac{\partial^2 u(x,t')}{2\partial t^2}\Delta t^2-\frac{\partial^2 u(x,t')}{2\partial t^2}\Delta t^2\Big)+\Big(\mathcal{O}(\Delta t^3)-\mathcal{O}(\Delta t^3)\Big)\end{aligned} \tag{70a}$$

$$u(x,t+\Delta t)-u(x,t) = \frac{\partial u(x,t')}{\partial t}\Delta t+\mathcal{O}(\Delta t^3) \tag{70b}$$

$$\frac{\partial u(x,t')}{\partial t} = \frac{u(x,t+\Delta t)-u(x,t)}{\Delta t}+\mathcal{O}(\Delta t^2) \tag{70c}$$

Now for the spacial derivative. First we solve (69a) and (69b) for $u_{xx}(x,t')$

$$\begin{aligned}u(x+\Delta x,t+\Delta t)+u(x-\Delta x,t+\Delta t) &= \Big(u(x,t')+u(x,t')\Big)+\Big(\frac{\partial u(x,t')}{\partial x}\Delta x-\frac{\partial u(x,t')}{\partial x}\Delta x\Big)\\ &\quad+\Big(\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2}+\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2}\Big)+\Big(\frac{\partial^2 u(x,t')}{2\partial x^2}\Delta x^2+\frac{\partial^2 u(x,t')}{2\partial x^2}\Delta x^2\Big)\\ &\quad+\Big(\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2}+\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{2}\Big)+\Big(\frac{\partial^3 u(x,t')}{6\partial x^3}\Delta x^3-\frac{\partial^3 u(x,t')}{6\partial x^3}\Delta x^3\Big)\\ &\quad+\Big(\frac{\partial^2 u(x,t')}{2\partial t^2}\frac{\Delta t^2}{4}+\frac{\partial^2 u(x,t')}{2\partial t^2}\frac{\Delta t^2}{4}\Big)+\Big(\frac{\partial^2 u(x,t')}{\partial x\partial t}\frac{\Delta t}{2}\Delta x-\frac{\partial^2 u(x,t')}{\partial x\partial t}\frac{\Delta t}{2}\Delta x\Big)\\ &\quad+\Big(\mathcal{O}(\Delta t^3)+\mathcal{O}(\Delta t^3)\Big)+\mathcal{O}(\Delta x^4)\end{aligned} \tag{71a}$$

$$= 2u(x,t')+\frac{\partial u(x,t')}{\partial t}\Delta t+\frac{\partial^2 u(x,t')}{\partial x^2}\Delta x^2+\frac{\partial^2 u(x,t')}{2\partial t^2}\frac{\Delta t^2}{2}+\mathcal{O}(\Delta t^3)+\mathcal{O}(\Delta x^4) \tag{71b}$$

$$= 2u(x,t')+\frac{\partial u(x,t')}{\partial t}\Delta t+\frac{\partial^2 u(x,t')}{\partial x^2}\Delta x^2+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^4) \tag{71c}$$

$$\begin{aligned}\frac{\partial^2 u(x,t')}{\partial x^2}\Delta x^2 &= u(x+\Delta x,t+\Delta t)+u(x-\Delta x,t+\Delta t)-2u(x,t')\\ &\quad+\frac{\partial u(x,t')}{\partial t}\Delta t+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^4)\end{aligned} \tag{71d}$$

$$\begin{aligned}\frac{\partial^2 u(x,t')}{\partial x^2} &= \frac{u(x-\Delta x,t+\Delta t)-2u(x,t')+u(x+\Delta x,t+\Delta t)}{\Delta x^2}\\ &\quad+\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{\Delta x^2}+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^2)\end{aligned} \tag{71e}$$

Doing the same as in (71e) for (69c) and (69d) we obtain

$$\begin{aligned}\frac{\partial^2 u(x,t')}{\partial x^2} &= \frac{u(x-\Delta x,t)-2u(x,t')+u(x+\Delta x,t)}{\Delta x^2}\\ &\quad-\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{\Delta x^2}+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^2)\end{aligned} \tag{72a}$$

Now we take the mean of (71e) and (72)

$$
\begin{aligned}
u_{xx}(x,t^{'}) &= \frac{1}{2}\Big(\frac{u(x-\Delta x,t+\Delta t)-2u(x,t')+u(x+\Delta x,t+\Delta t)}{\Delta x^2}+\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{\Delta x^2}+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^2)\\
&+\frac{u(x-\Delta x,t)-2u(x,t')+u(x+\Delta x,t)}{\Delta x^2}-\frac{\partial u(x,t')}{\partial t}\frac{\Delta t}{\Delta x^2}+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^2)\Big)\\
&= \frac{1}{2}\Big(\frac{u(x-\Delta x,t+\Delta t)-2u(x,t')+u(x+\Delta x,t+\Delta t)}{\Delta x^2}\\
&+\frac{u(x-\Delta x,t)-2u(x,t')+u(x+\Delta x,t)}{\Delta x^2}\Big)+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^2)\\
&\overset{u(x,t^{'})=\frac{u(x,t)+u(x,t+\Delta t)}{2}}{=} \frac{1}{2}\Big(\frac{u(x-\Delta x,t+\Delta t)-u(x,t)+u(x,t+\Delta t)+u(x+\Delta x,t+\Delta t)}{\Delta x^2}\\
&+\frac{u(x-\Delta x,t)-u(x,t)+u(x,t+\Delta t)+u(x+\Delta x,t)}{\Delta x^2}\Big)+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^2)\\
&= \frac{1}{2}\Big(\frac{u(x-\Delta x,t+\Delta t)-2u(x,t+\Delta t)+u(x+\Delta x,t+\Delta t)}{\Delta x^2}+\frac{u(x-\Delta x,t)-2u(x,t)+u(x+\Delta x,t)}{\Delta x^2}\Big)\\
&+\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta x^2)
\end{aligned}
\tag{73a}
$$

Now combining (70c) and (73a) we get the Crank-Nicolson scheme

$$
\begin{aligned}
\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t}+\mathcal{O}(\Delta t^2) &= \frac{1}{2}\Big(\frac{u(x-\Delta x,t+\Delta t)-2u(x,t+\Delta t)+u(x+\Delta x,t+\Delta t)}{\Delta x^2}\\
&+\frac{u(x-\Delta x,t)-2u(x,t)+u(x+\Delta x,t)}{\Delta x^2}\Big)+\mathcal{O}(\Delta x^2),
\end{aligned}
\tag{74a}
$$

where we have put both $\mathcal{O}(\Delta t^2)$ into a common term.

(74) shows that the Crank-Nicolson scheme is 2nd order in both time and space, implying better convergence properties for the Crank-Nicolson scheme compared to the Backward Euler scheme and the Forward Euler scheme.

We study the stability of the Crank-Nicolson scheme using the same method as for the previous schemes. Insertion of the ansatz (62) into the Crank-Nicolson scheme (74) and solving for $|a_k|$ gives:

$$
\begin{aligned}
\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} &= \frac{1}{2}\Big(\frac{u(x-\Delta x,t+\Delta t)-2u(x,t+\Delta t)+u(x+\Delta x,t+\Delta t)}{\Delta x^2}\\
&+\frac{u(x-\Delta x,t)-2u(x,t)+u(x+\Delta x,t)}{\Delta x^2}\Big)\\
a_k^n e^{ik\pi j\Delta x}\ \frac{a_k-1}{\Delta t} &= \frac{a_k^n e^{ik\pi j\Delta x}}{2}\ \Big(\frac{a_k e^{-ik\pi\Delta x}-2a_k+a_k e^{ik\pi\Delta x}}{\Delta x^2}+\frac{e^{-ik\pi\Delta x}-2+e^{ik\pi\Delta x}}{\Delta x^2}\Big)\\
\frac{a_k-1}{\Delta t} &= \frac{1}{2}\Big(\frac{a_k e^{-ik\pi\Delta x}-2a_k+a_k e^{ik\pi\Delta x}}{\Delta x^2}+\frac{e^{-ik\pi\Delta x}-2+e^{ik\pi\Delta x}}{\Delta x^2}\Big)\\
&= \frac{1+a_k}{2\Delta x^2}(e^{-ik\pi\Delta x}-2+e^{ik\pi\Delta x})\\
&\overset{(63)}{=} -4\frac{1+a_k}{2\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})\\
a_k-1 &= (1+a_k)(-\frac{2\Delta t}{\Delta x^2})\sin^2(\frac{k\pi\Delta x}{2})\\
\Big(1+\frac{2\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})\Big)a_k &= 1-\frac{2\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})
\end{aligned}
\tag{75a}
$$

$$
a_k = \frac{1-\frac{2\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})}{1+\frac{2\Delta t}{\Delta x^2}\sin^2(\frac{k\pi\Delta x}{2})} < 1 \tag{75b}
$$

(75b) shows that the Crank-Nicolson scheme is unconditionally stable.

Based on the analyzis of the different schemes, we expect the Crank-Nicolson scheme to be the best scheme with respect to convergence and stability.

### 3.5 $\theta$-rule

All the schemes derived above can be derived from a more general scheme, called the $\theta$-rule scheme. To see this, first notice that the Crank-Nicolson scheme (74) is the average of the Forward Euler scheme (61b) and the Backward Euler scheme (66a). If we think of the $\theta$-rule as a weighted average with weights $\theta$ and $1-\theta$ so that $u_t = \theta u_{xx,BE} + (1-\theta)u_{xx,FE}$, we see that the Crank-Nicolson shceme is just a special case of the $\theta$-scheme with equal weights, $\theta = 1/2$. Based on this reasoning, we get the $\theta$-rule

$$(1-\theta)\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} + \theta\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} = \theta\frac{u(x-\Delta x,t+\Delta t)-2u(x,t+\Delta t)+u(x+\Delta x,t+\Delta t)}{\Delta x^2} + (1-\theta)\frac{u(x-\Delta x,t)-2u(x,t)+u(x+\Delta x,t)}{\Delta x^2} \tag{76a}$$

$$\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} = \theta\frac{u(x-\Delta x,t+\Delta t)-2u(x,t+\Delta t)+u(x+\Delta x,t+\Delta t)}{\Delta x^2} + (1-\theta)\frac{u(x-\Delta x,t)-2u(x,t)+u(x+\Delta x,t)}{\Delta x^2} \tag{76b}$$

From the $\theta$-scheme (76b) we see that $\theta = 0,\ 1/2,\ 1$ corresponds to the Forward Euler, the Crank-Nicolson and the Backward Euler scheme respectively.

When implementing the 1D-schemes, we will use the $\theta$-scheme. Using the $\theta$-scheme, we need only write one scheme instead of three.

### 3.6 Implementation of 1D problem

As mentioned in the previous paragraph, we implement the 1D schemes using the $\theta$-scheme. We will now rewrite (76b) to a linear system, and then we will apply the Thomas algorithm to this system.

To ease the notation, we introduce

$$\alpha = \frac{\Delta t}{\Delta x^2}. \tag{77}$$

Insertion of $\alpha$ (77) into the $\theta$-scheme (76b) and introdusing the discretization $u(x+\Delta x, t-\Delta t) = u_{i+1}^{n-1}$ gives

$$\frac{u(x,t+\Delta t)-u(x,t)}{\Delta t} = \theta\frac{u(x-\Delta x,t+\Delta t)-2u(x,t+\Delta t)+u(x+\Delta x,t+\Delta t)}{\Delta x^2} + (1-\theta)\frac{u(x-\Delta x,t)-2u(x,t)+u(x+\Delta x,t)}{\Delta x^2} \tag{78a}$$

$$u(x,t+\Delta t)-u(x,t) = \alpha\theta\Big(u(x-\Delta x,t+\Delta t)-2u(x,t+\Delta t)+u(x+\Delta x,t+\Delta t)\Big) + \alpha(1-\theta)\Big(u(x-\Delta x,t)-2u(x,t)+u(x+\Delta x,t)\Big) \tag{78b}$$

$$u_i^{n+1}-u_i^n = \alpha\theta\Big(u_{i-1}^{n+1}-2u_i^{n+1}+u_{i+1}^{n+1}\Big) + \alpha(1-\theta)\Big(u_{i-1}^n-2u_i^n+u_{i+1}^n\Big) \tag{78c}$$

$$u_i^n-u_i^{n-1} = \alpha\theta\Big(u_{i-1}^n-2u_i^n+u_{i+1}^n\Big) + \alpha(1-\theta)\Big(u_{i-1}^{n-1}-2u_i^{n-1}+u_{i+1}^{n-1}\Big) \tag{78d}$$

$$u_i^n-\alpha\theta\Big(u_{i-1}^n-2u_i^n+u_{i+1}^n\Big) = u_i^{n-1}+\alpha(1-\theta)\Big(u_{i-1}^{n-1}-2u_i^{n-1}+u_{i+1}^{n-1}\Big) \tag{78e}$$

$$-\alpha\theta u_{i-1}^n + (2\alpha\theta+1)u_i^n - \alpha\theta u_{i+1}^n = \alpha(1-\theta)u_{i-1}^{n-1} + \Big(1-2\alpha(1-\theta)\Big)u_i^{n-1} + \alpha(1-\theta)u_{i+1}^{n-1} \tag{78f}$$

$$-2\alpha\theta u_{i-1}^n + 2(2\alpha\theta+1)u_i^n - 2\alpha\theta u_{i+1}^n = 2\alpha(1-\theta)u_{i-1}^{n-1} + 2\Big(1-2\alpha(1-\theta)\Big)u_i^{n-1} + 2\alpha(1-\theta)u_{i+1}^{n-1} \tag{78g}$$

Now we rewrite (78g) as a matrix-vector equation $A_1U^n = A_2U^{n-1}$:

$$\underbrace{\begin{bmatrix} 2(2\alpha\theta+1) & -2\alpha\theta & \cdots & 0 \\ -2\alpha\theta & 2(2\alpha\theta+1) & -2\alpha\theta & \vdots \\ \vdots & & \ddots & -2\alpha\theta \\ 0 & \cdots & -2\alpha\theta & 2(2\alpha\theta+1) \end{bmatrix}}_{\mathbf{A_1}} \underbrace{\begin{bmatrix} u_1^n \\ u_2^n \\ \vdots \\ u_N^n \end{bmatrix}}_{\mathbf{U^n}}$$

$$= \underbrace{\begin{bmatrix} 2(1-2\alpha(1-\theta)) & 2\alpha(1-\theta) & \cdots & 0 \\ 2\alpha(1-\theta) & 2(1-2\alpha(1-\theta)) & 2\alpha(1-\theta) & \vdots \\ \vdots & & \ddots & 2\alpha(1-\theta) \\ 0 & \cdots & 2\alpha(1-\theta) & 2(1-2\alpha(1-\theta)) \end{bmatrix}}_{\mathbf{A_2}} \underbrace{\begin{bmatrix} u_1^{n-1} \\ u_2^{n-1} \\ \vdots \\ u_N^{n-1} \end{bmatrix}}_{\mathbf{U^{n-1}}} \tag{79a}$$

In (79a), the right hand side $\mathbf{A_2U^{n-1}}$ is known, since $\mathbf{U}^{n-1}$ is the previous periods solution. Hence (79a) is a linear system of the known type $AU = b$, which needs to be solved at each time step.

We note that with $\theta = 0$ in (79a), the system reduces to the explicit Forward Euler system, which can be obtained from (61b). $\theta = 1$ gives the Backward Euler system (68f). $\theta = 1/2$ in (79a) gives the system that can be obtained by rewrting the Crank-Nicolson scheme (74) as a linear system.

In principle the above system can be solved by calculating the inverse of $A_1$ and solving $U^n = A_1^{-1}(A_2U^{n-1})$. This is a costly operation and is avoided. Instead of calculating the inverse, one often solves these systems by a elimination method, e.g. Gaussian elimination or by LU. In this case, we note that we are dealing with a tridiagonal matrix. For tridiagonal systems, the Thomas algorithm gives an alterative way of solving the linear system which reduces the number of FLOPS considerably. In our case we are eaven more lucky. Our matrix is symmetric, and the elements are constant, so the standard Thomas algorithm can be further improved. We will now derive the algorithm that we implement.

We start with a tridiagonal symmetric linear system $Au = f$, with $A$ being $4 \times 4$. The following show the forward substitution steps for this sytem

$$\begin{bmatrix} d_1 & e_1 & 0 & 0 & f_1 \\ e_2 & d_2 & e_2 & 0 & f_2 \\ 0 & e_3 & d_3 & e_3 & f_3 \\ 0 & 0 & e_4 & d_4 & f_4 \end{bmatrix} \rightarrow \begin{bmatrix} d_1 & e_1 & 0 & 0 & f_1 \\ 0 & \tilde{d}_2 & e_2 & 0 & \tilde{f}_2 \\ 0 & e_3 & d_3 & e_3 & f_3 \\ 0 & 0 & e_4 & d_4 & f_4 \end{bmatrix} \rightarrow \begin{bmatrix} d_1 & e_1 & 0 & 0 & f_1 \\ 0 & \tilde{d}_2 & e_2 & 0 & \tilde{f}_2 \\ 0 & 0 & \tilde{d}_3 & e_3 & \tilde{f}_3 \\ 0 & 0 & e_4 & d_4 & f_4 \end{bmatrix} \rightarrow \begin{bmatrix} d_1 & e_1 & 0 & 0 & f_1 \\ 0 & \tilde{d}_2 & e_2 & 0 & \tilde{f}_2 \\ 0 & 0 & \tilde{d}_3 & e_3 & \tilde{f}_3 \\ 0 & 0 & e_4 & \tilde{d}_4 & \tilde{f}_4 \end{bmatrix} \tag{80a}$$

From (79a) we have that in our case $e_1 = e_2 = ... = e_N = -2\alpha$ and $d_1 = d_2 = ... = d_N = 2(2\alpha\theta + 1)$. Calculating the first couple of $\tilde{d}$'s we quickly see that we get the following general expression for $\tilde{d}_i$

$$\tilde{d}_i = 2(2\alpha\theta + 1) + \frac{(2\alpha)^2}{\tilde{d}_{i-1}} \text{ for } i > 1. \tag{81}$$

Doing the same for $\tilde{f}$ we get the general expression

$$\tilde{f}_i = f_i + \frac{2\alpha\theta}{\tilde{d}_{i-1}} \tilde{f}_{i-1} \text{ for } i > 1. \tag{82}$$

Having $\tilde{d}$ and $\tilde{f}$ we are ready to do the back substitution step. We have

$$\begin{bmatrix} d_1 & e_1 & 0 & 0 \\ 0 & \tilde{d}_2 & e_2 & 0 \\ 0 & 0 & \tilde{d}_3 & e_3 \\ 0 & 0 & e_4 & \tilde{d}_4 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} f_1 \\ \tilde{f}_2 \\ \tilde{f}_3 \\ \tilde{f}_4 \end{bmatrix} \tag{83a}$$

From (83) we get

$$u_4 = \frac{\tilde{f}_4}{\tilde{d}_4} \tag{84}$$

and

$$u_i = \frac{\tilde{f}_i + 2\alpha\theta u_{i+1}}{\tilde{d}_i} \text{ for } i > 1. \tag{85}$$

The equations (81), (82), (84) and (85) is what we need for our algorithm.

```
for time in times:
   Calculate RHS f (= A_2 U^{n-1})

   // Forward substitution
   for row in rows (Start from 1st row):
      Calculate \tilde{d}
      Calculate \tilde{f}
   end row loop

   // Backward substitution
   For row in rows (Starting from last row)
      Calculate u
   end row loop

   // Update RHS for next time step
   U^{n-1} = U^{n}

end time loop
```

## 3.7 2D schemes

We will derive and apply a Forward Euler (explicit) scheme and a Backward Euler (implicit) scheme for the 2D case.

### 3.7.1 2D explicit scheme

Compared to the 1D case, we get an extra term for $u_{yy}$ in (61b) for the explicit scheme. $u_{yy}$ is approximated in exactly the same way as $u_{xx}$ (60c), so we get

$$u_{yy}(x,y,t) = \frac{u(x,y-\Delta y,t) - 2u(x,y,t) + u(x,y+\Delta y,t)}{\Delta y^2} + \mathcal{O}(\Delta y^2) \tag{86}$$

Adding (86) to the 1D FE scheme (61b) gives the explicit 2D-scheme

$$\begin{aligned} \frac{u(x,y,t+\Delta t) - u(x,y,t)}{\Delta t} + \mathcal{O}(\Delta t) = &\frac{u(x-\Delta x,y,t) - 2u(x,y,t) + u(x+\Delta x,y,t)}{\Delta x^2} \\ &+ \frac{u(x,y-\Delta y,t) - 2u(x,y,t) + u(x,y+\Delta y,t)}{\Delta y^2} \\ &+ \mathcal{O}(\Delta x^2) + \mathcal{O}(\Delta y^2) \end{aligned} \tag{87a}$$

$$\rightarrow u_{ij}^{n+1} \overset{(77)}{\approx} u_{ij}^n + \alpha(u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n - 4u_{ij}^n), \tag{87b}$$

where we in the last transition assumed $\Delta x = \Delta y$ such that $\alpha$ is given as before.

First we observe that the truncation errors go like $\mathcal{O}(\Delta t)$, $\mathcal{O}(\Delta x^2)$ and $\mathcal{O}(\Delta y^2)$.

We analyze the stability by inserting the ansatz

$$u = a_k^n e^{ik\pi\Delta xj} e^{ik\pi\Delta yl} \tag{88}$$

into (87b) and solve for $|a|$, as before

$$u_{ij}^{n+1} = u_{ij}^n + \alpha(u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n - 4u_{ij}^n) \tag{89a}$$

$$a_k^n e^{i\pi k\Delta xj} e^{i\pi k\Delta jl} a = a_k^n e^{i\pi k\Delta xj} e^{i\pi k\Delta jl} \Big(1 + \alpha(e^{-ik\pi\Delta x} + e^{ik\pi\Delta x}) - 2 + e^{-ik\pi\Delta y} + e^{ik\pi\Delta y}) - 2\Big) \tag{89b}$$

$$a = \Big(1 + \alpha(e^{-ik\pi\Delta x} + e^{ik\pi\Delta x}) - 2 + e^{-ik\pi\Delta y} + e^{ik\pi\Delta y}) - 2\Big) \tag{89c}$$

$$\overset{(63)}{=} 1 - 4\alpha\Big(\sin^2(k\pi\Delta x/2) + \sin^2(k\pi\Delta y/2)\Big) \tag{89d}$$

$$|a| = \Big|1 - 4\alpha\Big(\sin^2(k\pi\Delta x/2) + \sin^2(k\pi\Delta y/2)\Big)\Big| \tag{89e}$$

$$|a| < 1 \rightarrow \Big|1 - 4\alpha\Big(\sin^2(k\pi\Delta x/2) + \sin^2(k\pi\Delta y/2)\Big)\Big| < 1 \tag{89f}$$

$$\Big|1 - 4\alpha\Big(\sin^2(k\pi\Delta x/2) + \sin^2(k\pi\Delta y/2)\Big)\Big| < |1 - 8\alpha| \tag{89g}$$

Again assuming $\Delta x = \Delta y$, we have $\alpha = \Delta t/\Delta x^2 > 0$, so that (89f) and (89g) gives

$$1 - 8\alpha < -1 \tag{90a}$$

$$\alpha < \frac{1}{4}. \tag{90b}$$

(90b) gives that the 2D Forward Euler scheme is conditionally stable. We note that we could probably have applied a simpler ansatz than (88), since it in (87b) was already assumed that $\Delta x = \Delta y$.

The new thing in (87b), compared to the 1D case (61b), is that $uij^{n+1}$ in the 2D case is a matrix and not a vector. The fact that we are dealing with a matrix instead of a vector, does not change anything with regards to the solution strategy. The equation (87b) is still explicit, the only difference in implementation is that instead of one single loop, a double loop over two space dimensions is needed now.

The algorithm for solving 2D Forward Euler follows.

```
Set IC

for time in times:
   Set BC

   for x in X:
      for y in Y:
         Calculate u_{ij}^{n+1} = f(u^{n})
      end y-loop
   end x-loop

   Set u^{n} = u^{n+1}

end time-loop
```

### 3.7.2 2D implicit scheme

We start in the same way as we did for Forward Euler in 2D, and just add an extra term for $u_{yy}$ to the 1D Backward Euler scheme (66a). In addition we will assume $h = \Delta x = \Delta y$.

$$\begin{aligned}\frac{u(x,y,t) - u(x,y,t-\Delta t)}{\Delta t} + \mathcal{O}(\Delta t) &= \frac{u(x-\Delta x,y,t) - 2u(x,y,t) + u(x+\Delta x,y,t)}{\Delta x^2} \\ &+ \frac{u(x,y-\Delta y,t) - 2u(x,y,t) + u(x,y+\Delta y,t)}{\Delta y^2} \\ &+ \mathcal{O}(\Delta x^2) + \mathcal{O}(\Delta y^2)\end{aligned} \tag{91a}$$

$$\begin{aligned}\frac{u_{i,j}^n - u_{i,j}^{n-1}}{\Delta t} &\approx \frac{u_{i-1,j}^n - 2u_{i,j}^n + u_{i+1,j}^n}{\Delta x^2} \\ &+ \frac{u_{i,j-1}^n - 2u_{i,j}^n + u_{i,j+1}^n}{\Delta y^2}\end{aligned} \tag{91b}$$

$$\begin{aligned}\frac{u_{i,j}^n - u_{i,j}^{n-1}}{\Delta t} &\overset{h=\Delta x=\Delta y}{\approx} \frac{u_{i-1,j}^n - 2u_{i,j}^n + u_{i+1,j}^n}{h^2} \\ &+ \frac{u_{i,j-1}^n - 2u_{i,j}^n + u_{i,j+1}^n}{h^2}\end{aligned} \tag{91c}$$

$$u_{i,j}^n - u_{i,j}^{n-1} \overset{(77)}{\approx} \alpha(u_{i-1,j}^n - 4u_{i,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n) \tag{91d}$$

$$u_{i,j}^n = u_{i,j}^{n-1} + \frac{1}{1+4\alpha}(u_{i-1,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n) \tag{91e}$$

First we observe that the truncation errors go like $\mathcal{O}(\Delta t)$, $\mathcal{O}(\Delta x^2)$ and $\mathcal{O}(\Delta y^2)$, so there is no difference in the truncation errors between Forward Euler and Backward Euler, just as for the 1D case.

We study stability of the 2D Backward Euler scheme by inserting the 2D ansatz (88) into (91d)

$$u_{i,j}^n - u_{i,j}^{n-1} = \alpha(u_{i-1,j}^n - 4u_{i,j}^n + u_{i+1,j}^n + u_{i,j-1}^n + u_{i,j+1}^n) \tag{92a}$$

$$a_k - 1 = \alpha a(e^{-ik\pi\Delta x} + e^{ik\pi\Delta x} - 2 + e^{-ik\pi\Delta y} + e^{ik\pi\Delta y} - 2) \tag{92b}$$

$$\overset{(63)}{=} -4\alpha a\Big(\sin^2(k\pi\Delta x/2) + \sin^2(k\pi\Delta y/2)\Big) \tag{92c}$$

$$\overset{\Delta x=\Delta y=h}{=} -8\alpha a \sin^2(kh/2) \tag{92d}$$

$$a = \frac{1}{1+8\alpha\sin^2(kh/2)} < 1 \tag{92e}$$

From (92e) we see that the Backward Euler scheme is unconditionally stable in two dimensions, as it was for one dimension. We note that we probably could have applied a simpler ansatz than (88), since it in (91d) was already assumed that $\Delta x = \Delta y$.

We see that (91e) is an implicit scheme, since the only known is $u_{i,j}^n$. (91e) can be transformed into a linear system of the known type $Au = b$. The procedure for transforming (91e) into a linear system is the same as demonstrated for the 1D case, see Hjorth-Jensen [1] p. 317. However, there is one important difference compared to the 1D linear system: The matrix is not tridiagonal anymore! This means that we cannot apply the Thomas algorithm anymore, and standard methods as LU-decomposion becomes unpractical. However, in our case the matrix is positive definite, implying that iterative schemes converges to the true solution, Hjorth-Jensen [1] p. 322. Iterative methods are often much faster than direct methods like Gaussian elimination and LU-decomposition. Hence we will solve the implicit problem applying iterative methods.

### 3.7.3 Iterative methods

This section follows Olver and Shakiban's [3] chapter on iterative methods closesly. The goal is to present the main idea behind iterative methods, and to present the Jacobi method as part of a general framework.

Firstly, an iterative method is not a direct method, like e.g. Gaussian elemination, that solves the problem as it is stated, typically

$$\mathbf{A}\mathbf{u} = \mathbf{b}. \tag{93}$$

Instead, with iterative methods, one attempts solving (93) by solving another, affine iterative, system:

$$\mathbf{u}^{(k+1)} = \mathbf{T}\mathbf{u}^{(k)} + \mathbf{c}, \tag{94}$$

where $k$ stands for iterations, $T$ is a coefficient matrix of same size as $A$, and $\mathbf{c}$ is a vector. We observe that the right hand side is an affine function of $\mathbf{u}^{(k)}$. For comparison, a linear iterative system is on the form $\mathbf{u}^{(k+1)} = \mathbf{T}\mathbf{u}^{(k)}$, so it is the addition of the vector $\mathbf{c}$ that makes our iterative an affine one, and not a linear one. The affine representation is more genereal compared to the linear one, and allows for more reflexibility in developing iterative schemes.

So how is using (94) in place of solving the original system better? It might not be better. But potentially it can be a lot faster. Look at the FLOP count. In (94) the matrix-vector multiplication is $\mathcal{O}(N)$ flops, and the addition is of the same order, giving $\mathcal{O}(N)$ FLOPS per iteration. The total FLOP-count for the iterative method is $\mathcal{O}kN$ FLOPS. In comparison the Gaussian elimination method is always $\mathcal{O}(N^3)$ FLOPS. This shows that if we can construct a $T$ and $\mathbf{c}$ in our affine iterative scheme (94) that converges to the true solution after few iterations, the iterative scheme will be much less demanding with respect to FLOPS compared to direct methods like Gaussian elimination.

We have shown that the potential of the iterative methods with regards to FLOPS is bug. Now we need to analyse condition for when the method actually works. For the iterative method (94) to work, it firstly has to converge:

$$\lim_{k\to\infty} \mathbf{u}^k = \mathbf{u}^*. \tag{95}$$

Taking the limit on both sides of (94) and using (95) we get

$$\lim_{k\to\infty} \mathbf{u}^{k+1} = \lim_{k\to\infty} \mathbf{T}\mathbf{u}^k + \mathbf{c} \tag{96a}$$

$$\mathbf{u}^* = \mathbf{T}\mathbf{u}^* + \mathbf{c}. \tag{96b}$$

(96b) is a fixed point equation, and we have that the solution to the iterative scheme has to converge to a fixed point, $\mathbf{u}^*$.

Secondly, the convergent solution $\mathbf{u}^*$ that is approached as $k \to \infty$ must become equal to the solution $\mathbf{u}$ of the original problem, $\mathbf{A}\mathbf{u} = \mathbf{b}$. If the affine iterative system (94) converges to a fixed point that differs from the true solution, $\mathbf{u}^* \neq \mathbf{u}$, the fixed point splution is of little use for us.

Lets now look at conditions for the 1st condition, convergence to the fixed point $\mathbf{u}^*$, to hold. We study the error with respect to the fixed point

$$\mathbf{e}^{k+1} = \mathbf{u}^{k+1} - \mathbf{u}^* \tag{97a}$$

$$\overset{(94)}{=} \mathbf{T}\mathbf{u}^k + \mathbf{c} - \mathbf{u}^* \tag{97b}$$

$$= \mathbf{T}\mathbf{u}^k + \mathbf{c} - (\mathbf{T}\mathbf{u}^* + \mathbf{c}) \tag{97c}$$

$$= \mathbf{T}(\mathbf{u}^k - \mathbf{u}^* \tag{97d}$$

$$= \mathbf{T}\mathbf{e}^k \tag{97e}$$

$$= \mathbf{T}^k\mathbf{e}^{(0)}. \tag{97f}$$

(97f) shows that the development of the error in convergence towards the fixed point depends on the same matrix as the original iterative problem, $\mathbf{T}$. We have convergence towards the fixed point $\mathbf{u}^*$ if $\mathbf{T}$ is a convergent matrix. $\mathbf{T}$ is a convergent matrix if the spectral radius of $\mathbf{T}$ is less than one, $\rho(T) < 1$. The speed of convergence is inversly related to the spectral radius, implying that we want to construct $T$ with as low spectral radius as possible.

We now have a condition for the first requirement for our iterative scheme: convergence to the fixed point. Now for the 2nd condition: convergence to the true solution. It turns out that convergence to the true solution depeneds on the chosen scheme, represented by the matrix $T$ in our case, and the coefficient matrix of the original problem, $A$. We have that both the Jacobi scheme, to be derived, and the Gauss-Seidel scheme are convergent to the true solution if $A$ is strictly diagonally dominant.

We sum up our results in this section:

The iteration scheme (94) converges to the solution of the original linear system (93) if the following two conditions are satisfied

- The spectral radius of $T$ in (94) is less than one.
- $A$ in (93) is strictly diagonally dominant (In the case of Jacobi and Gauss-Seidel)

Next we will derive the Jacobi-algorithm and relate it to our genereal iterative system (94).

### 3.7.4 Jacobi's algorithm

We begin by a rewrite of the linear system (93), where we split $A$ into one strictly lower triangular matrix, $L$, one strictly upper triangular matrix, $U$, and one diagonal matrix, $D$:

$$A\mathbf{u} = \mathbf{b} \tag{98a}$$
$$(L + D + U)\mathbf{u} = \mathbf{b} \tag{98b}$$
$$\mathbf{u} = D^{-1}\Big( - (L + D)\mathbf{u} + \mathbf{b}\Big) \tag{98c}$$

The above is nothing new. It is still the original system. With the Jacobi algorithm, one sets $\mathbf{u}^{k+1} = \mathbf{u}$ on the left hand side of (98c) and $\mathbf{u} = \mathbf{u}^k$ on the right hand side to get

$$\mathbf{u}^{k+1} = D^{-1}\Big( - (L + D)\mathbf{u}^k + \mathbf{b}\Big) \tag{99a}$$

We see that the Jacobi-scheme (99a) fits our general iterative fixed point scheme (94), with

$$T = -D^{-1}(L + D) \tag{100a}$$
$$\mathbf{c} = D^{-1}\mathbf{b}. \tag{100b}$$

It is perhaps easier to see the workings of the Jacbi algoritm (99a) by considering a single row in the matrix-vector system (99a)

$$u_i^{(k+1)} = -\frac{1}{a_{ii}} \sum_{j=1, i\neq j} a_{ij} u_j^{(k)} + \frac{b_i}{a_{ii}}. \tag{101a}$$

From (101) we see that only non-diagonal terms are used in the iteration on the right hand side, and that all terms are divided by the diagonal terms from the original matrix $A$. (101) is very close to the form of the implemented algorithm.

Using (101) we will in the next section see that it is easy to formulate a Jacobi-solver for the 2D implicit scheme (91e).

### 3.7.5 Jacobi's algorithm applied to the 2D implicit diffusion

Now we will apply Jacobi's algorithm (101) to the 2D implicit scheme (91e). We see that (91e) is already almost set up like a Jacobi-scheme. We get

$$u_{i,j}^{n,(k+1)} = u_{i,j}^{n-1} + \frac{1}{1+4\alpha}(u_{i-1,j}^{n,(k)} + u_{i+1,j}^{n,(k)} + u_{i,j-1}^{n,(k)} + u_{i,j+1}^{n,(k)}) \tag{102a}$$

(102) is our Jacobi-Solver. By setting

$$a_{ii} = -(1+4\alpha) \tag{103a}$$

$$b_i = a_{ii} u_{i,j}^{n-1} \tag{103b}$$

in (102) we see that (102) corresponds to the Jacobi-algorithm (101). We also note that only 4 neighboring pints are needed.

Following Hjorth-Jensen [1] p. 319, we get the following main algorithm for the Jacobi-solver

- 1: Make intial guess for $u_{i,j}$ for all internal points.
- 2: Caclulate iteration $k$ of $u$, $u^k$ using (91e).
- 3: Stop if wanted convergence reached, else continue next iteration.
- 4: Set previous $u$-value equal to new $u$.
- 5: Repeat all steps from step 2.

The above algorithm is used for all time steps, starting with the first internal time step, solving one step at the time before moving to the next step.

## 3.8 2D Physical problem with boundary conditions

We choose to model a laminar flow inside an infinite tunnel with a finite quadratic cross section. We only model a single cross section, the problem being identical for all cross sections because of the homogenity assumption in the flow direction.

There will be no slip boundary conditions on all four walls. The floor and the sides will be fixed, while the roof will be moving in the flow direction. With these boundary conditions we have zero velocity on the floor and the side walls, while we have a non-zero velcocity in the flow direction. The figure below shows our problem.

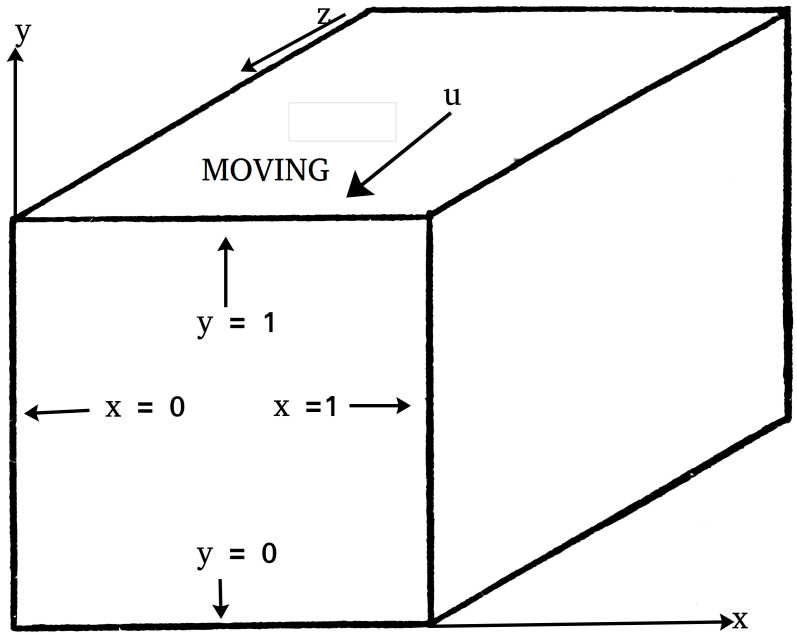


Figure 1: 2D problem sketch

With our choise of coordinate system, positive flow represents flow out of the paper.

We choose $u(x, y = 1) = 1$, and $u(x = 0, y) = u(x = 1, y) = u(x, y = 0) = 0$.

We model the flow with the heat equation. This equation can be derived from the equations of incompressible flow, the Navier-Stokes equations, by assuming laminar flow and a flow profile of type $u = u(x, y)$:

$$\mathbf{u}_t + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{\nabla p}{\rho} + \nabla^2 \mathbf{u} \tag{104a}$$

$$w(x,y)_t + (w\frac{\partial}{\partial z})w(x,y) \overset{\text{Zero pressure gradient}}{=} (\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial x^2})w(x,y) \tag{104b}$$

$$w(x,y)_t = (\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial x^2})w(x,y) \tag{104c}$$

Central assumptions behind the resulting heat equation is that there is no external pressure gradient and that the flow is laminar.

# 4 Results

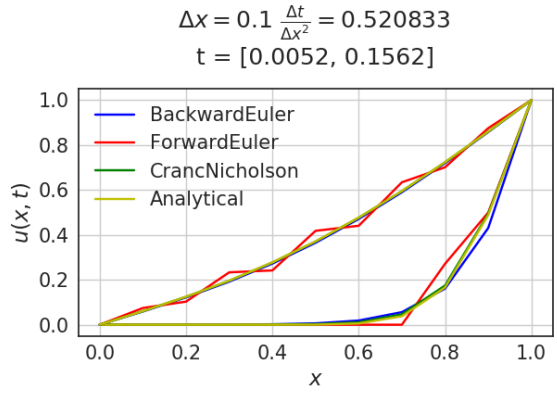


Figure 2: 1D. Coarse mesh. Stability requirement Forward not satisfied.
*Forward Euler is unstable, while the other schemes are stable. Backward Euler and Crank-Nicolson produce results close to the anlytical solution for all time steps. Steady state is approached for Backward Euler and Crank-Nicolson.*

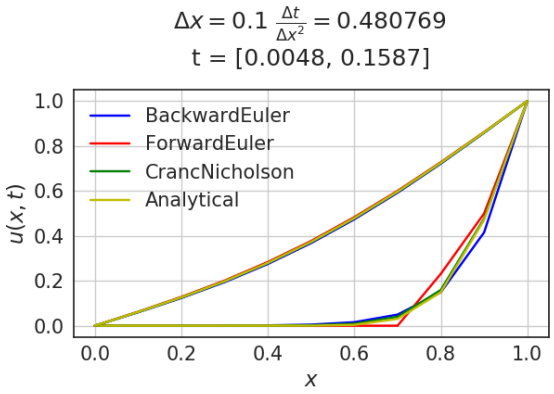


Figure 3: 1D. Coarse mesh. Stability requirement Forward satisfied.
*Forward Euler is now stable. All schemes produce results looking qualitively like the analytical solution. Forward Euler has the largest error..*

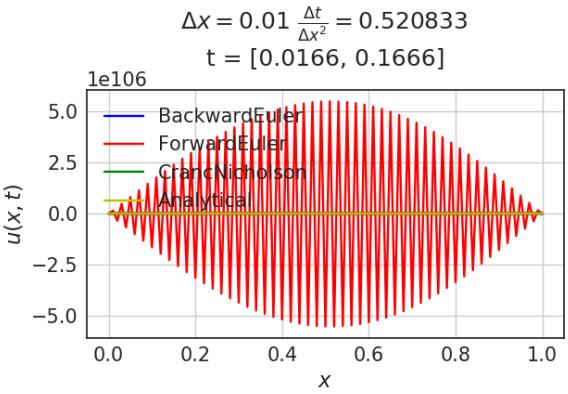


Figure 4: 1D. Fine mesh. Stability requirement Forward Euler not satisfied.
*As for the coarse mesh, Forward Euler is unstable.*

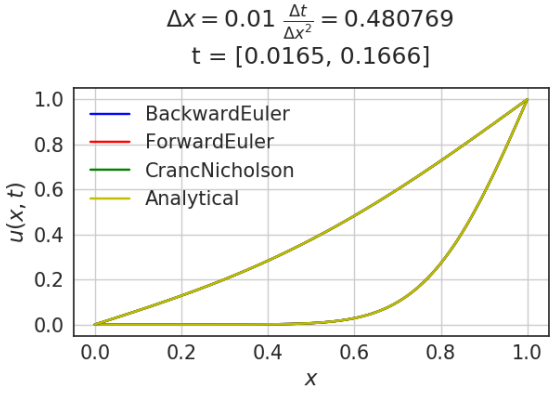


Figure 5: 1D. Fine mesh. Stability requirement Forward satisfied.
*Forward Euler is stable. All methods give solutions close to the analytical solution.*

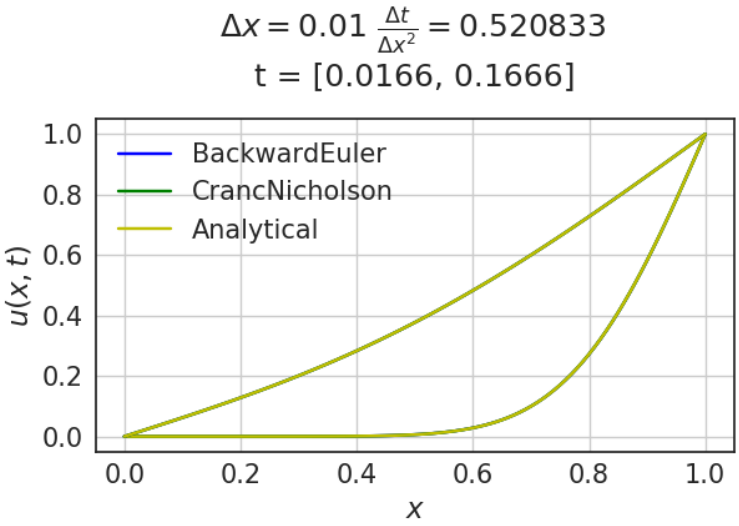


Figure 6: Same as Figure 5 without Forward Euler.
*Backward Euler and Crank-Nicolson produces very good results below also when the stability limit for Forward Euler is not satisfied.*

From the figures above, we see that the Forward Euler scheme is very sensitive to the stability requriement, implying that a very fine time step is needed for acceptable results. From this point of view, the Backward and Crank-Nicolson schemes are prefered, since they need fewer time steps. We also observe that both the Crank-Nicolson schemes seems to produce results closer to the analytical solution also when Forward Euler's stability requirement is satisfied. Hence Backward Euler and Crank Nicolson is prefererred over Forward Euler. From Figure 4 we see that Crank-Nicolson is closer to the analytic solution compared to the Backward Euler scheme. This is not unexpected, given our derivations in the theory section, where we showed that the Crank-Nicolson scheme is one order higher in time than the other schemes when it comes to truncation error.

## 4.1 2D

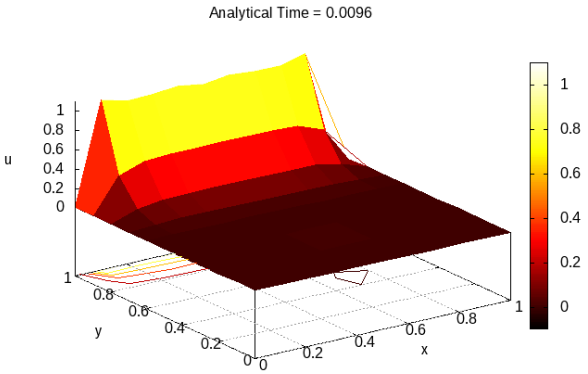


Figure 7: 2D. $\Delta x = 0.1$. $\Delta t = 0.96\cdot$ stability limit. Early time. Analytical Solution.
*Analytical solution seems correct.*

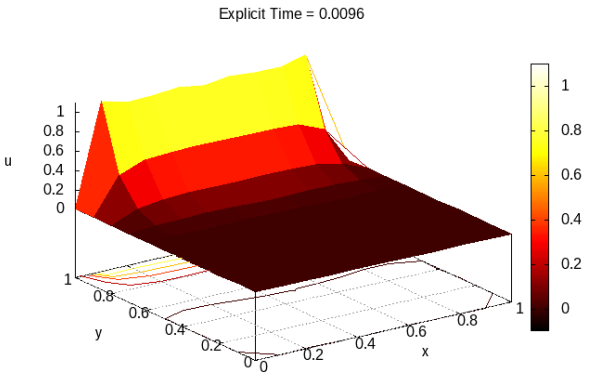


Figure 8: 2D. $\Delta x = 0.1$. $\Delta t = 0.96\cdot$ stability limit. Early time. Explicit scheme.
*Explicit scheme produces results very similar to analytical solution.*

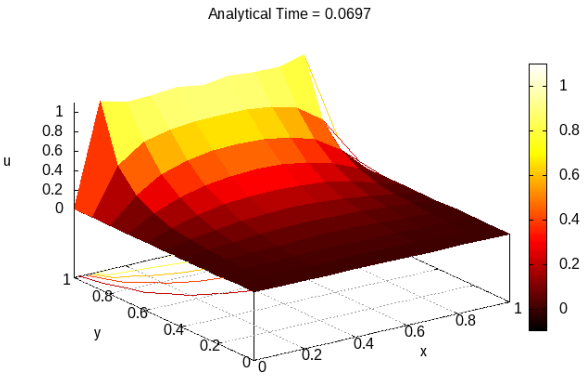


Figure 9: 2D. $\Delta x = 0.1$. $\Delta t = 0.96\cdot$ stability limit. Later time. Analytical Solution.
*Analytical solution seems correct.*

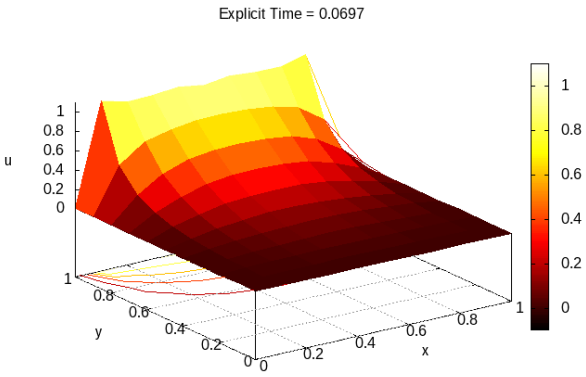


Figure 10: 2D. $\Delta x = 0.1$. $\Delta t = 0.96\cdot$ stability limit. Later time. Explicit scheme.
*Explicit scheme produces results very similar to analytical solution.*

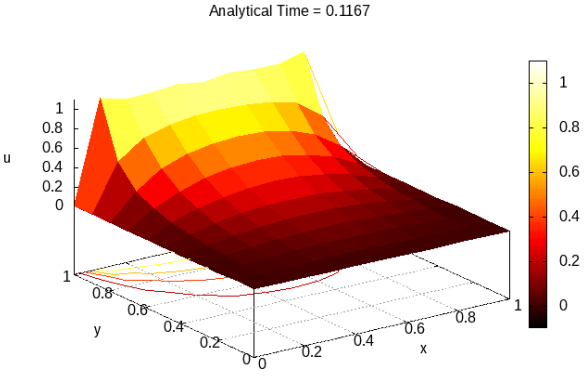


Figure 11: 2D. $\Delta x = 0.1$. $\Delta t = 1.1\cdot$ stability limit. Analytical Solution.

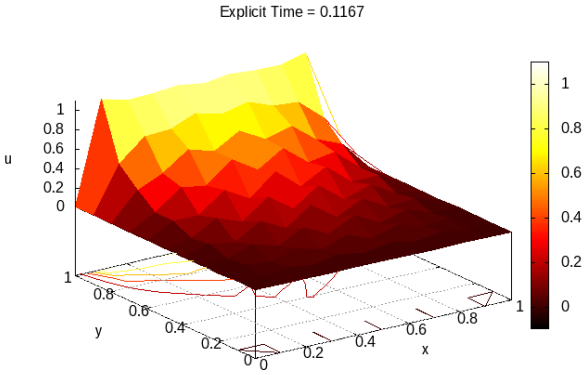


Figure 12: 2D. $\Delta x = 0.1$. $\Delta t = 1.1\cdot$ stability limit. Explicit scheme.
*The explicit scheme is unstable when the stability requirement is not met.*

# 5 Bibliography

[1] Hjorth-Jensen, M.(2015) Computational physics. Lectures fall 2015. `https://github.com/CompPhysics/ComputationalPhysics/tree/master/doc/Lectures`

[2] `https://github.com/CompPhysics/ComputationalPhysics/tree/master/doc/pub/pde`

[3] Olver, P.J and Shakiban, C.(2006) Applied linear algebra. Pearson Prentice Hall.