Semester Project in TMA4212

Thorvald M. Ballestad
Jonas Bueie
Knut Andre Grytting Prestsveen
Herman Sletmoen

April 27, 2021

Contents

1 Poisson equation in one dimension

In this section, we will solve the one-dimensional Poisson equation

$$\frac{\partial^2 u}{\partial x^2} = f(x) \qquad (0 < x < 1) \tag{1}$$

subject to a source term $f(x)$ and different combinations of Dirichlet and Neumann boundary conditions at $x = 0$ and $x = 1$. First, we will solve it with finite difference methods of first and second order on a uniform grid. Finally, we solve it on a non-uniform grid and investigate how adaptive mesh refinement (AMR) can be used to obtain accurate solutions by distributing fewer points more cleverly along the grid.

1.1 Analytical solution

One way to express the analytical solution is to simply integrate equation (1) twice to get

$$\begin{aligned} u(x) &= C_1 + \int^x \mathrm{d}x' u_x(x') \\ &= C_1 + \int^x \mathrm{d}x' \left(C_2 + \int^{x'} \mathrm{d}x'' u_{xx}(x'') \right) \\ &= C_1 + C_2 x + \int^x \mathrm{d}x' \int^{x'} \mathrm{d}x'' f(x''), \end{aligned} \tag{2}$$

where the constants C_1 and C_2 are determined from two boundary conditions and the integrals can be done from any lower limit. Note that this is equivalent to saying that the solution is a sum of the solution to the homogenuous equation $u_{xx} = 0$ and a solution to the inhomogenuous equation $u_{xx} = f(x)$.

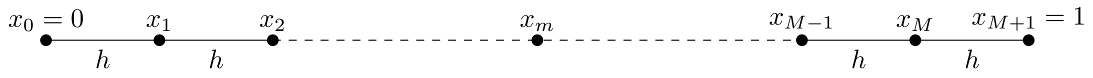
Note that if *two* Neumann boundary conditions $u_x(0) = a$ and $u_x(1) = b$ are imposed, then the solution $u(x)$ is unique only up to a constant. If $u(x)$ is a solution to the boundary value problem defined by equation (1) with $u_x(0) = a$ and $u_x(1) = b$, then also $(u + C)_{xx} = u_{xx}$ in the interior and $(u + C)_x(0) = u_x(0) = a$ on the left boundary, and similarly on the right boundary $x = 1$. It can also be seen by observing that C_1 is undetermined when 2 is differentiated.

1.2 Numerical solution on a uniform grid

First, consider the boundary value problem defined by equation (1), subject to the boundary conditions

$$u(0) = a \quad \text{or} \quad u_x(0) = a \qquad \text{and} \qquad u(0) = b \quad \text{or} \quad u_x(1) = b.$$

To solve the equation numerically, we divide the interval $[0, 1]$ into the uniform grid



of $M + 2$ points and step length h. We approximate the second derivative at interior points with the central difference

$$\frac{\partial u}{\partial x}(x_m) = \frac{u_{m-1} - 2u_m + u_{m+1}}{h^2} + \mathcal{O}(h^2) \qquad (1 \leq m \leq M - 1).$$

To handle the Dirichlet boundary condition $u(0) = a$ at the left edge or $u(1) = b$ at the right edge, we insert the trivial equation

$$1 \cdot u_0 = a \qquad \text{or} \qquad 1 \cdot u_{M+1} = b.$$

To handle the Neumann boundary condition $u_x(0) = a$ at the left edge or $u_x(1) = b$ at the right edge to second order, we approximate the first derivative to second order with forward or backward differences to get

$$u_x(0) = \frac{-\frac{3}{2}u_0 - 2u_1 - \frac{1}{2}u_2}{h} + \mathcal{O}(h^2) = b \qquad \text{or} \qquad u_x(1) = \frac{\frac{1}{2}u_{M-1} - 2u_M + \frac{3}{2}u_{M+1}}{h} + \mathcal{O}(h^2) = b.$$

Writing all these equations in $(M+2) \times (M+2)$-matrix form $AU = b$, we obtain for example with $u(0) = a$ and $u_x(1) = b$

$$\begin{bmatrix} 1 & & & & \\ +1/h^2 & -2/h^2 & +1/h^2 & & \\ & \ddots & \ddots & \ddots & \\ & & +1/h^2 & -2/h^2 & +1/h^2 \\ & & +1/2h & -2/h & +3/2h \end{bmatrix} \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_M \\ U_{M+1} \end{bmatrix} = \begin{bmatrix} a \\ f(x_1) \\ \vdots \\ f(x_M) \\ b \end{bmatrix}, \tag{3}$$

where the first and last rows of the matrix generally vary depending on the boundary conditions.

Note that if the numerical solution is subject to two Neumann boundary conditions, the matrix becomes singular and the solution non-unique. In this case, we impose the additional constraint $U_0 = 0$ by setting all entries in the first column of A to zero. To handle the singular matrix, we instead find the least-squares solution to the matrix equation. [5]

We now apply our method to the boundary value problem with the source function

$$f(x) = x + \cos(2\pi x).$$

Inserting it into equation (2) and doing the integrals, we get the exact solution

$$u(x) = C_1 + C_2 x + \frac{1}{3!}x^3 - \frac{1}{4\pi^2}\cos(2\pi x)).$$

In figure 1, we present numerical solutions for three different combinations of boundary conditions.

Our approach to handling the boundary conditions is not the only possible approach. The system of equations is equivalent if we remove the first row and column of A and the first entries in U and b, but simultaneously modify the entry $f(x_1) \to f(x_1) - a/h^2$. This approach is done in [6], for example, and is more consistent with treating U_0 as a known variable, since its precise value is defined by the Dirichlet boundary condition. However, our approach of inserting a trivial equation $1 \cdot U_0 = a$ keeps the matrix dimensions independent of boundary conditions and makes it easier to reason with how the discretized differential operator represented by A operates on the grid point U_0 in the same way it operates on all other grid points.

Neumann boundary conditions can also be handled differently. Instead of approximating the second derivative only on actual grid points, we could approximate it with a fictuous point x_{-1} and a central difference $u_x(0) \approx (U_1 - U_{-1})/(2h)$. Then we could use this together with the central difference $(U_{-1} - 2U_0 + U_1)/h^2 = f(x_0)$ to eliminate U_{-1}. Eliminating U_{-1}, the first equation becomes $(U_1 - U_0)/h = a + hf(x_0)/2$, so the boundary condition could be handled by setting the first row to $[-1/h, +1/h, 0, \dots]$ and modifying the first entry in b to $a \to a + hf(x_0)/2$. This would also be second order, and would allow us to use the same stencil also at x_0, but we would then have to pay the price of modifying the right side of the matrix equation in an unnatural way. This approach is also done in [6].

1.3 Adaptive numerical solution on a non-uniform grid

We will now demonstrate how the numerical solution can be generalized to a non-uniform grid with $x_i - x_{i-1} \neq$ const. Then we will attempt to make the numerical solution as good as possible using as few grid points as possible, by placing points tighter where the solution varies rapidly.

To derive a nonuniform stencil for the second derivative at x_m, we proceed similarly to the uniform stencil. First approximate one derivative by stepping halfway left and right, landing at $x_{m-1/2}$ and $x_{m+1/2}$. Then we approximate another derivative by stepping halfway to the sides again, landing at x_{m-1}, x_m and x_{m+1}. This yields

$$u''_m \approx \frac{u'_{m+1/2} - u'_{m-1/2}}{x_{m+1/2} - x_{m-1/2}} \approx \frac{2}{x_{m+1} - x_{m-1}} \left(\frac{u_{m+1} - u_m}{x_{m+1} - x_m} - \frac{u_m - u_{m-1}}{x_m - x_{m-1}} \right).$$

Assuming Dirichlet boundary conditions, the nonzero entries of A (indexed from zero) becomes

$$A_{0,0} = A_{M+1,M+1} = 1 \qquad A_{m,m-1} = \frac{2}{x_{m+1} - x_{m-1}} \frac{1}{x_m - x_{m-1}}$$

$$A_{m,m} = \frac{-2}{x_{m+1} - x_{m-1}} \left(\frac{1}{x_m - x_{m-1}} + \frac{1}{x_{m+1} - x_m} \right) \qquad A_{m,m+1} = \frac{2}{x_{m+1} - x_{m-1}} \frac{1}{x_{m+1} - x_m}.$$

The job is then once again to solve the system $AU = b$. Note that the stencil reduces to the one in equation (3) when $x_m - x_{m-1} = x_{m+1} - x_m = h$, as it should.

To do adaptive mesh refinement, we will

1. Start with a coarse uniform grid, such as $[x_0, x_1] = [0, 1]$.
2. Wisely choose *one* grid interval $[x_m, x_{m+1}]$ based on some strategy.
3. Split the interval in half by inserting a new point at $(x_m + x_{m+1})/2$.
4. Repeat step 2 and 3 until the grid has the desired resolution.

We will compare three different strategies for selecting the grid interval:

1. **Error strategy:** Select the interval $[x_m, x_{m+1}]$ with the largest error

   $$\int_{x_m}^{x_{m+1}} \mathrm{d}x |u(x) - U(x)|, \qquad \text{where } U(x) = U_m + \frac{x - x_m}{x_{m+1} - x_m} (U_{m+1} - U_m)$$

   is a linearly interpolated numerical solution on the *current* grid and $u(x)$ is the exact solution. This strategy requires knowledge of the exact solution $u(x)$ and solving the system numerically before each splitting.

2. **Truncation error strategy:** Select the interval $[x_m, x_{m+1}]$ with the largest absolute truncation error

   $$\left| \frac{2}{x_{m+1} - x_m} \left(\frac{u_{m+1} - u_{m+1/2}}{x_{m+1} - x_{m+1/2}} - \frac{u_{m+1/2} - u_m}{x_{m+1/2} - x_m} \right) - f(x_m) \right|,$$

   upon insertion of a middle point $x_{m+1/2} = (x_m + x_{m+1})/2$, where $u(x)$ is the exact solution. This strategy also requires knowledge of the exact solution $u(x)$, but does not rely on intermediate computations of the numerical solution.

3. **Source strategy:** Select the interval $[x_m, x_{m+1}]$ with the largest "absolute source"

   $$\int_{x_m}^{x_{m+1}} \mathrm{d}x |f(x)|.$$

   In physical applications, $f(x)$ is typically mass density or charge density. The idea is to refine intervals on which there is much mass or charge, as the solution is expected to vary faster there. This splitting strategy requires neither knowlege of the exact solution or the numerical solution, only on the source function $f(x)$, as is typically the case in practice.

In figure 2, we demonstrate how the initial grid $[0, 0.5, 1]$ and the numerical solution $U(x)$ evolves through adaptive refinement with the error strategy. Observe how the refinement concentrates on resolving critical areas of the solution near the peak and the inflection points.

In figure 3, we compare the convergence of the three adaptive refinement strategies to the $\mathcal{O}(h^2)$-convergence of uniform refinement on the same problem. The error strategy requires knowledge of the exact solution and intermediate computations, but in return it is the most effective strategy. The source strategy requires neither, but is also the least effective strategy. We can say that the more knowledge of the exact solution and intermediate computations, the greater the accuracy.

Note that the errors are not strictly decreasing with each refinement $M \to M + 2$. In particular, the error from the error strategy exhibits an oscillating pattern for $M \geq 32$. This is a weakness of refining only *exactly* two symmetric intervals for each refinement. An alternative method is to refine *multiple* intervals at every refinement step using a criterion that splits not only the interval with the largest error, but all intervals with error above some reference error. This procedure removes our control over the exact number of intervals, but in return gives us control over the maximal acceptable error on any interval. In section 6.4.2, we will improve our AMR strategy exactly in this way. The effect is that the oscillating pattern is eliminated and that the error decreases strictly with each refinement step. This will be equivalent to jumping directly from one local minimum in the oscillation to the next.

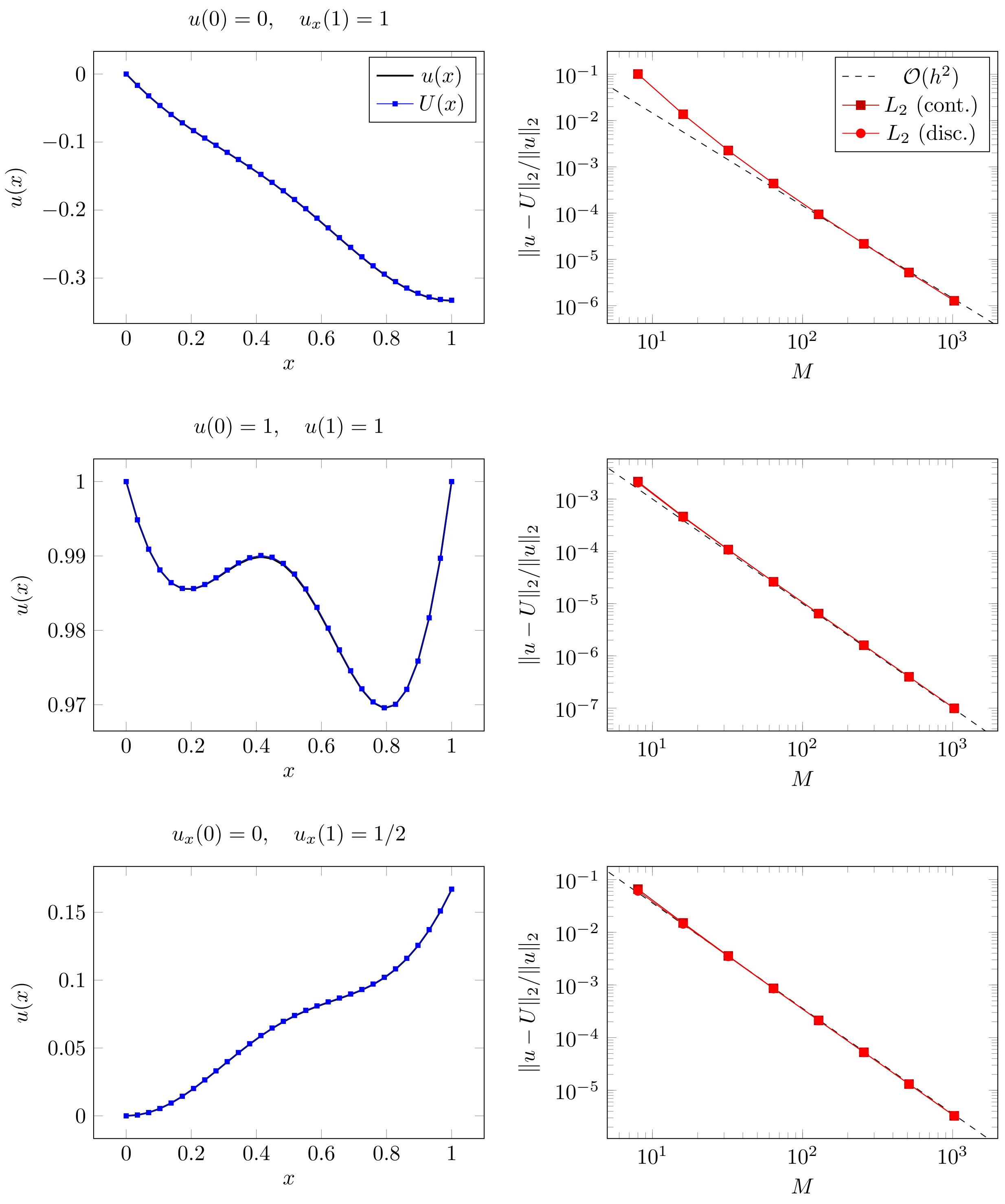


Figure 1: The left plots show analytical solutions $u(x)$ and numerical solutions $U(x)$ with $M = 30$ grid points for $u_{xx} = x + \cos 2\pi x$ subject to three different boundary conditions. The right plots show convergence plots corresponding to the same boundary conditions, where the error is measured with both the a continuous and discrete L_2-norm.

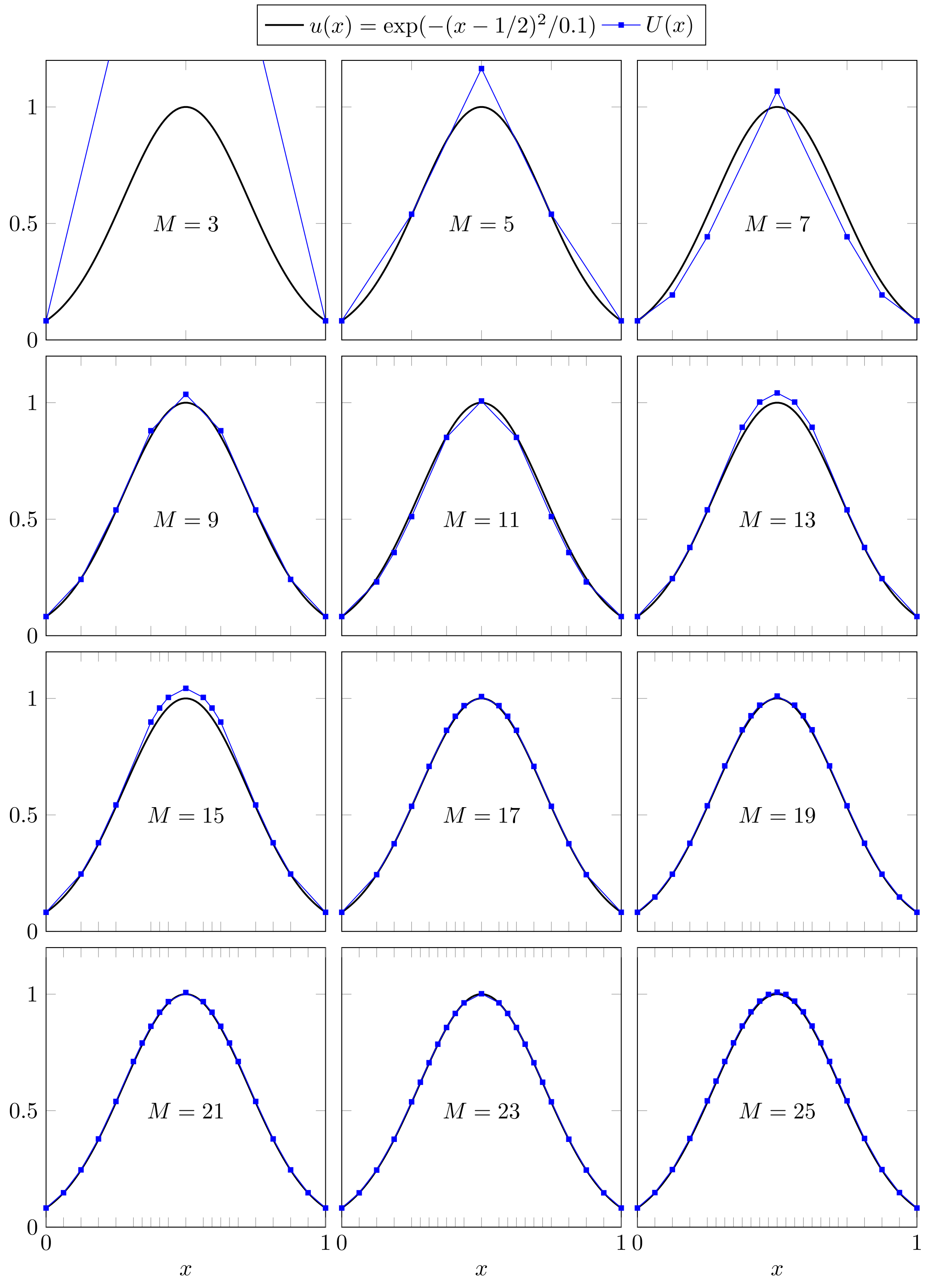


Figure 2: During adaptive mesh refinement (AMR) with the error strategy (strategy number 1), the interval with the largest error $\int \mathrm{d}x|u(x) - U(x)|$ is split in half. Here, $u(x) = \exp(-(x-1/2)^2/0.1)$ is the symmetric solution to whichever Poisson equation has $f(x) = u_{xx}$ on $x \in [0, 1]$. Symmetry is imposed numerically by also adding the point $1 - x$ to the grid whenever a point $x \neq 1/2$ is added.

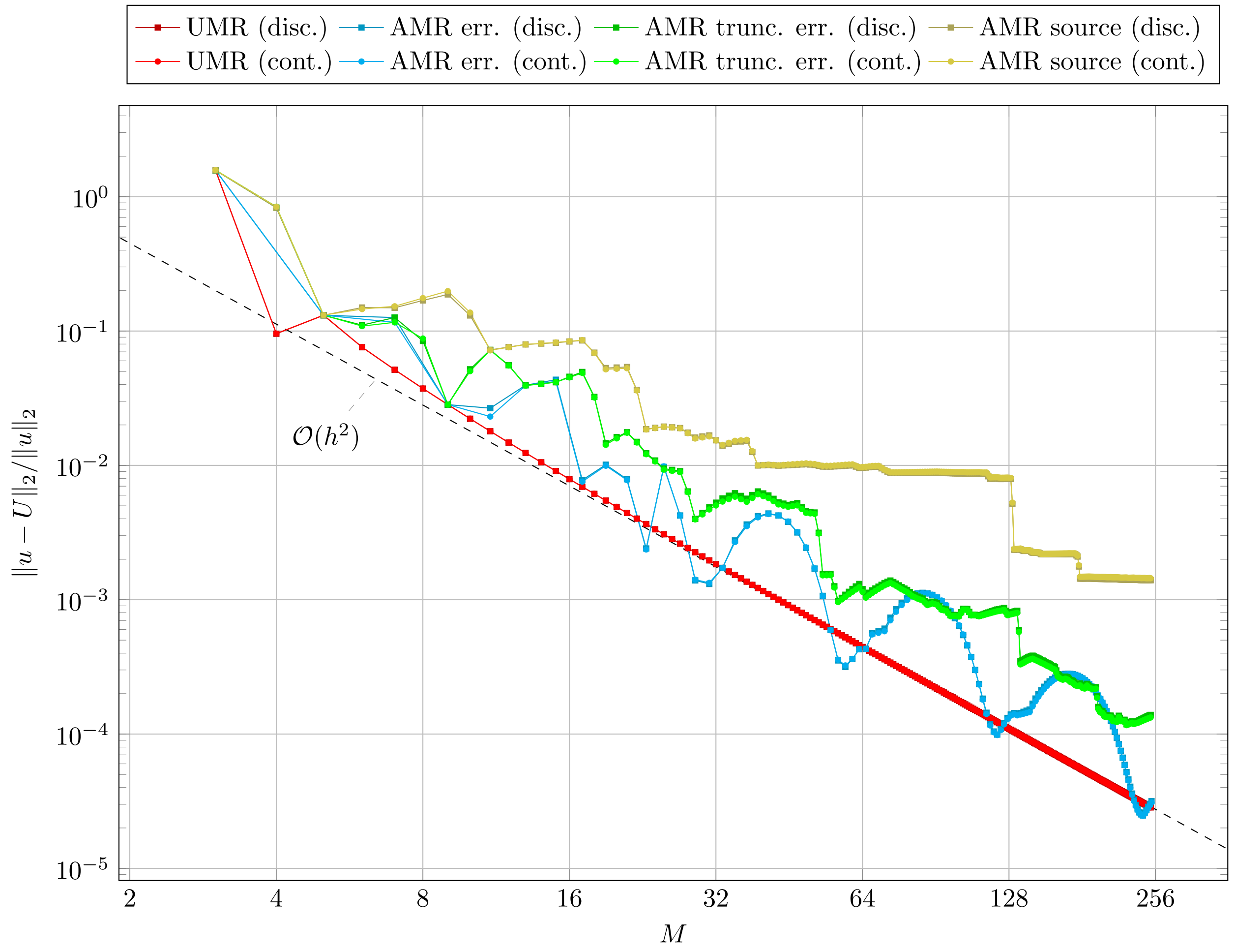


Figure 3: Comparison between the convergence of the numerical solution $U(x)$ with uniform mesh refinement (UMR) and adaptive mesh refinement (AMR) on the problem $u_{xx} = f(x)$ on $x \in [0,1]$ with analytical solution $u(x) = \exp(-(x-1/2)^2/0.1)$. The adaptive refinement is done using three different strategies that subdivide the interval with the largest absolute error $|u-U|$, largest truncation error $Lu - f(x)$ (where $L \approx \partial^2/\partial x^2$ is the discretized differentiation operator) or largest amount of source $\int \mathrm{d}x|f(x)|$. Errors $\|u-U\|_2$ are measured with the continuous and discrete L_2-norm.

2 Heat equation in one dimension

In this section, we consider the one-dimensional heat equation for $u = u(x,t)$,

$$u_t = u_{xx}, \quad u(x,0) = f(x), \quad x \in [0,1] := \Omega,$$

with either Neumann or Dirchlet boundary conditions, and solve it numerically using both the Backward Euler method and Crank-Nicolson method. These are $\mathcal{O}(k+h^2)$ and $\mathcal{O}(k^2+h^2)$ methods respectively [6], and we will analyze and compare their convergence using mesh refinement as we did in section 1. We will do refinement of the grids in both the x-direction and the t-direction, however we will here restrict our attention to uniform grids only.

2.1 Numerical solution method

To solve the heat equation numerically we first perform semi-discretization, i.e. we do spatial discretization and keep the time continous. As in section 1, we divide the interval Ω into $M+2$ equidistant nodes with separation $h = 1/(M+1)$, so that we get a uniform grid with M internal nodes and two boundary nodes. We then express the spatial derivative using the central finite difference to get

$$u_t(x_m,t) = \frac{1}{h^2}\delta_x^2 u(x_m,t) + \mathcal{O}(h^2), \quad m = 0,...,M+1.$$

We now introduce the single variate functions $v_m(t)$ as the approximation to $u(x_m,t)$, at each node x_m, turning the PDE into a set of ODEs

$$\frac{dv_m(t)}{dt} = \frac{1}{h^2}\delta_x^2 v_m(t), \quad v_m(0) = f(x_m).$$

The problem is then generally solved by imposing the boundary conditions, and numerically integrating the equations in time, using for instance one of the many standard schemes for ODEs such as Euler's method. For the sake of convenience we employ the θ-method, which for general ODEs $y' = g(y,t)$ is given as

$$y^{n+1} = y^n + \Delta t\left((1-\theta)g(y^n,t_n) + \theta g(y^{n+1},t_{n+1})\right),$$

and the value of θ determines the specific numerical scheme

$$\begin{aligned} \text{Forward Euler} \quad & \theta = 0 \\ \text{Backward Euler} \quad & \theta = 1 \\ \text{Crank-Nicolson} \quad & \theta = \frac{1}{2}. \end{aligned}$$

We use a constant time step $k = 1/(N-1)$, where N denotes the number of time steps, and the final uniform grid is illustrated in figure4. This gives the approximate solution of $v_m(t)$ at $t_n = nk$, where $n = 0,\dots N-1$, and we denote the fully discretized approximation of $u(x_m,t_n)$ as U_m^n. After organizing the terms, the θ-method for the 1D heat equation is then written out as

$$(1-\theta r\delta_x^2)U_m^{n+1} = \left(1 + (1-\theta)r\delta_x^2\right)U_m^n, \tag{4}$$

where we have defined $r = k/h^2$. In the following we will as mentioned consider the Backward-Euler and Crank-Nicolson methods.

To impose Dirchlet boundary conditions, $u(0,t) = \sigma$, $u(1,t) = \beta$, we substitute $U_0^{n+1} = \sigma$ and $U_{M+1}^{n+1} = \beta$ in equation (4) for $m = 1$ and $m = M$ to obtain

$$\begin{aligned} (1+2r\theta)U_1^{n+1} - r\theta U_2^{n+1} &= \left(1-2r(1-\theta)\right)U_1^n + r(1-\theta)U_2^n + r\sigma \quad (\text{for } m=1) \\ (1+2r\theta)U_M^{n+1} - r\theta U_{M-1}^{n+1} &= \left(1-2r(1-\theta)\right)U_M^n + r(1-\theta)U_{M-1}^n + r\beta \quad (\text{for } m=M), \end{aligned}$$

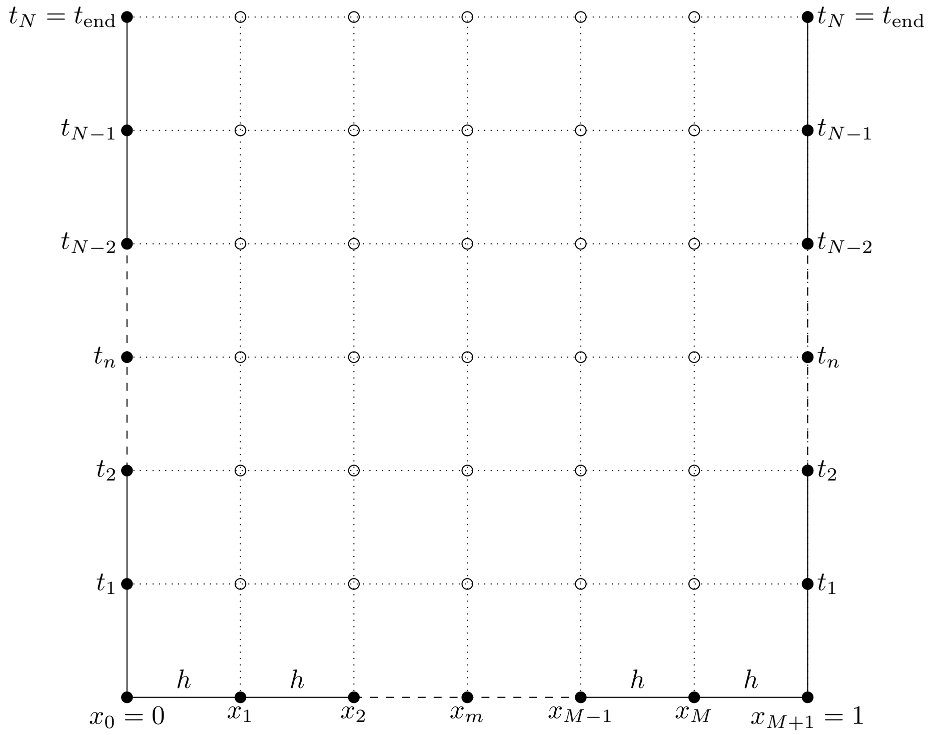


Figure 4: Discrete uniform grid

which we combine with equation 4 for the remaining spatial nodes to write the system of equations in matrix form

$$(I - \theta r A)U^{n+1} = (I + (1-\theta) r A)U^n + \rho, \tag{5}$$

with

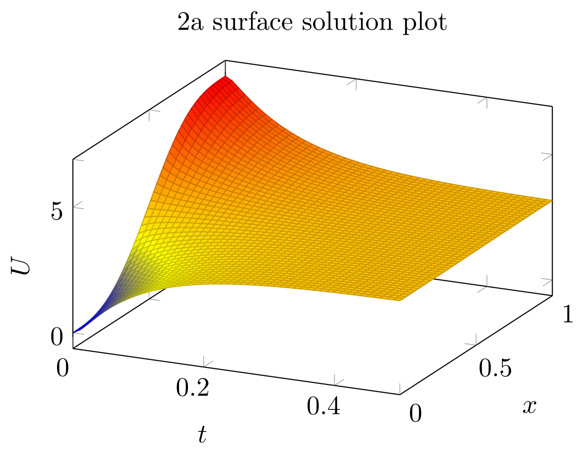
$$A = \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix} \quad \text{and} \quad \rho = \begin{bmatrix} r\sigma \\ 0 \\ \vdots \\ 0 \\ r\beta \end{bmatrix}. \tag{6}$$

For Neumann boundary conditions, $u_x(0,t) = \sigma$, $u_x(1,t) = \beta$, we introduce fictitious nodes at $m = -1$ and $m = M + 2$, and approximate the first derivatives at the boundaries by

$$\frac{U_1 - U_{-1}}{2h} = \sigma \quad \text{and} \quad \frac{U_{M+2} - U_M}{2h} = \beta.$$

We then use these expressions to eliminate the fictitious nodes from equation (4) for $m = 0$ and $m = M + 1$ to get

$$(1 + 2r\theta)U_0^{n+1} - 2r\theta U_1^{n+1} = \big(1 - 2r(1-\theta)\big)\, U_0^n + 2r(1-\theta)U_1^n - 2hr\sigma \quad (\text{for } m = 0)$$
$$(1 + 2r\theta)U_{M+1}^{n+1} - 2r\theta U_M^{n+1} = \big(1 - 2r(1-\theta)\big)\, U_{M+1}^n + 2r(1-\theta)U_M^n + 2rh\beta \quad (\text{for } m = M + 1).$$



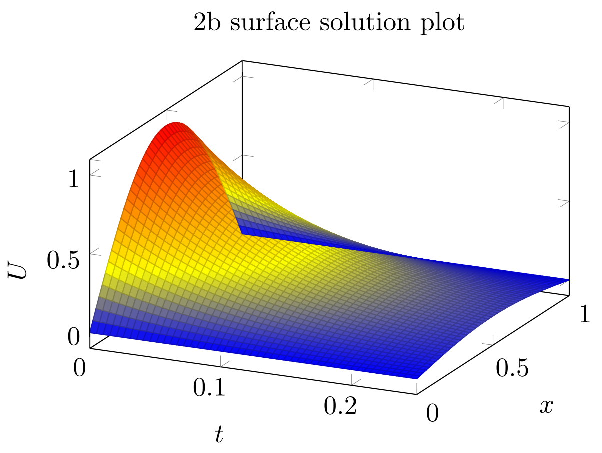


Figure 5: I am a surface plot, Hooray :)

Now we can write the system of equations on the same matrix form (5), but with

$$A = \begin{bmatrix} -2 & 2 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 2 & -2 \end{bmatrix} \quad \text{and} \quad \rho = \begin{bmatrix} -2rh\sigma \\ 0 \\ \vdots \\ 0 \\ 2rh\beta \end{bmatrix}. \tag{7}$$

Note that with Dirchlet conditions at both boundaries we only solve the equations for the internal spatial nodes $x_1 \dots x_M$ so that A in (6) is an $M \times M$ matrix. With Neumann conditions however we still need to solve for the boundary nodes, and A is in (7) an $(M+2) \times (M+2)$ matrix. In both cases though, all quantities on the right hand sides in (5) are known, i.e. the equations are on the form $A\vec{x} = \vec{b}$, and the known $\vec{b}$ is just written via a matrix-vector product for notational convenience. To solve the problem we now solve this system of equations at each time step, and since matrices in both cases are tridiagonal, we represent them as sparse matrices and use a solver for sparse systems to save both memory and time.

With the numerical schemes in hand we now solve the heat equation with the following Neumann boundary conditions and initial condition,

$$u_x(0,t) = u_x(1,t) = 0, \quad u(x,0) = 2\pi x - \sin(2\pi x). \tag{8}$$

The computed solutions for $t \in [0, 0.5]$ is plotted in figure 5, and qualitatively the solution behaves in accordance with what we expect for the heat equation. To quantify and compare the accuracy of the numerical schemes we will now proceed to analyze convergence using mesh refinement, similar to what we did in section 1.

2.2 Convergence and mesh refinement

As in section 1 we now analyze the convergence of the numerical solution methods by doing mesh refinement of the spatial grid x_m. For (8), the the analytical solution is not available in closed form, so in order to analyze convergence we compute a reference solution using a sufficiently high M, which we use in place of the analytical solution when computing the error. When doing mesh refinement of the spatial grid, we vary the number of spatial grid points M, and compute the numerical solution at the same point in time $t = t_{end}$. We keep the number of time steps N fixed, so that the error from the time discretization does not vary, and compute the L_2 discrete and $l2$ continous relative errors with respect to the reference solution. The resulting convergence rates from the refinement is plotted in figure6.

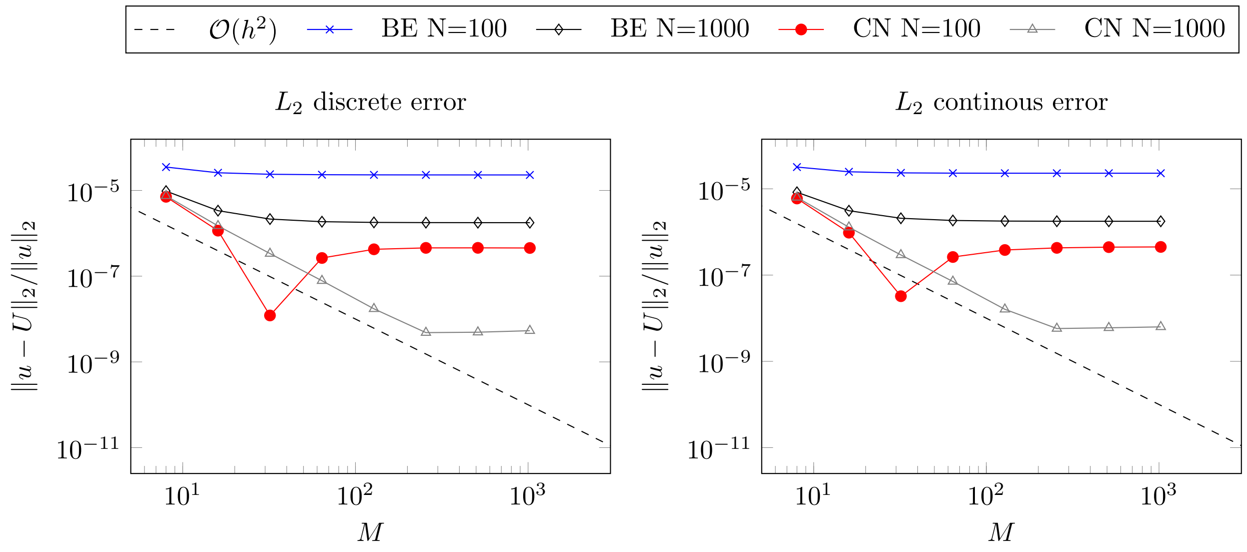


Figure 6: Convergence plot, h refinement (2a)

From figure6 we see that **comment about results**

In order to analyze the convergence further, we also consider the heat equation with a set of boundary and initial conditions for which the analytical solution is known. Specifically we consider

$$u(0,t) = u(1,t) = 0, \quad u(x,0) = \sin(\pi x), \tag{9}$$

on the same domain $x \in [0,1] := \Omega$ and $t > 0$. Note that we now have Dirchlet boundary conditions, and we are now able compute the analytical solution, which we do using the method of sepparation of variables. **Skal skrive sepparasjon her da**

$$u(x,t) = \sin(\pi x)e^{-\pi^2 t}. \tag{10}$$

We now do the same spatial refinement for equation (9), however now we compute the errors with respect to the analytical solution. The resulting convergence plots are shown in figure 7, and here the difference between Backward Euler and Crank-Nicolson becomes more apparent. Both methods are second order in the spatial step h, and are unconditionally stable [**find some source probably brynjulf**], unlike e.g. the explicit Forward Euler method ($\theta = 0$). Crank-Nicolson is however one order higher in the time step k, so that the total error is lower for Crank-Nicolson than Backward-Euler when M and N are the same. This is seen when refining the spatial step h; at large h (low M), the error is dominated by the spatial error, and as we decrease h (increase M) we should see the expected second order convergence in the spatial step. At some point, when the spatial error has become sufficiently small, the total error will be dominated by the error in the time step k, and we expect this to happen earlier for the Backward-Euler method. In figure 7 we see exactly this. The error of the Crank-Nicolson solution with $N = 10000$ exhibits second order convergence throughout the entire refinement, but when lowering the number of time steps to $N = 1000$, the time step error starts to dominate towards the end of the refinement and the error curve flattens. For the solution computed with the Backward-Euler method we see the flattening happening much earlier, which is due to this method being less accurate in time.

Now we proceed to do refinement in the t-direction. As mentioned, Crank-Nicolson

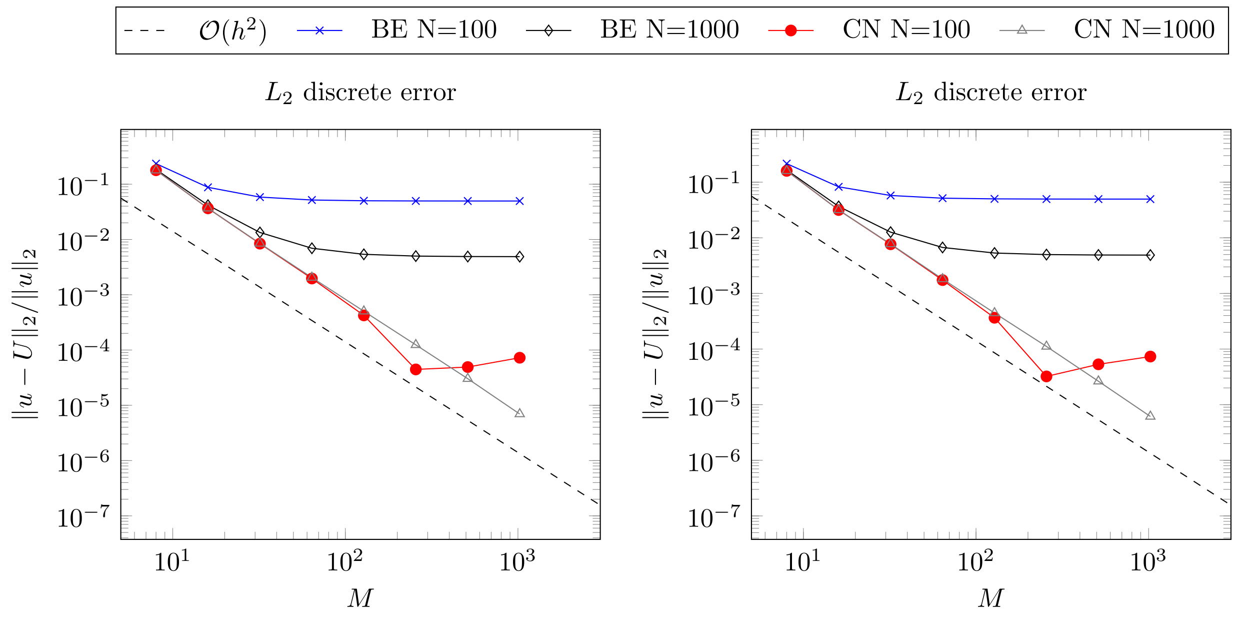


Figure 7: Convergence plot, h refinement (2b)

3 Invicid Burgers' equation

In this section we turn to solve the inviscid Burgers' equation with given Dirchlet boundary conditions and initial condition

$$u_t = -uu_x, \quad u(0,t) = u(1,t) = 0, \quad u(x,0) = exp(-400(x-1/2)^2). \tag{11}$$

This equation exhibits breaking; after some point in time t_b the solution breaks, and the unique solution does not exist, leading to the formation of a *shock wave*.[**burgers**] The time t_b before this can happen is given by

$$t_b = \frac{-1}{minf'(x)}, \tag{12}$$

where $f(x)$ is the given initial condition $u(x,0) = f(x)$.[**burgers**]

Numerical solution method

To solve (11) numerically we perform semidiscretization in the same way as we did for the heat equation in section 2, also on a uniform spatial grid as described in section 1. The resulting system of ODEs is

$$\frac{\partial v_m}{\partial t} = -v_m \frac{1}{2h}(v_{m+1} - v_{m-1}).$$

We impose the Dirchlet boundary conditions and integrate the ODEs using *solve_ivp* from the SciPy library, with the default explicit Runge-Kutta method of order 4(5)[**solveˆivp**].
Comment/question: Is it fine to use scipy.integrate.solve_ivp, or should it be all home cooking?

3.1 Time of breaking

Insertion of $u(x,0)$ for f in (12), gives $t_b \approx 0.058$. To get a criterion for when the numerical solution has broken down, we use that the stable solution should be strictly increasing from $x = 0$ to towards the apex,

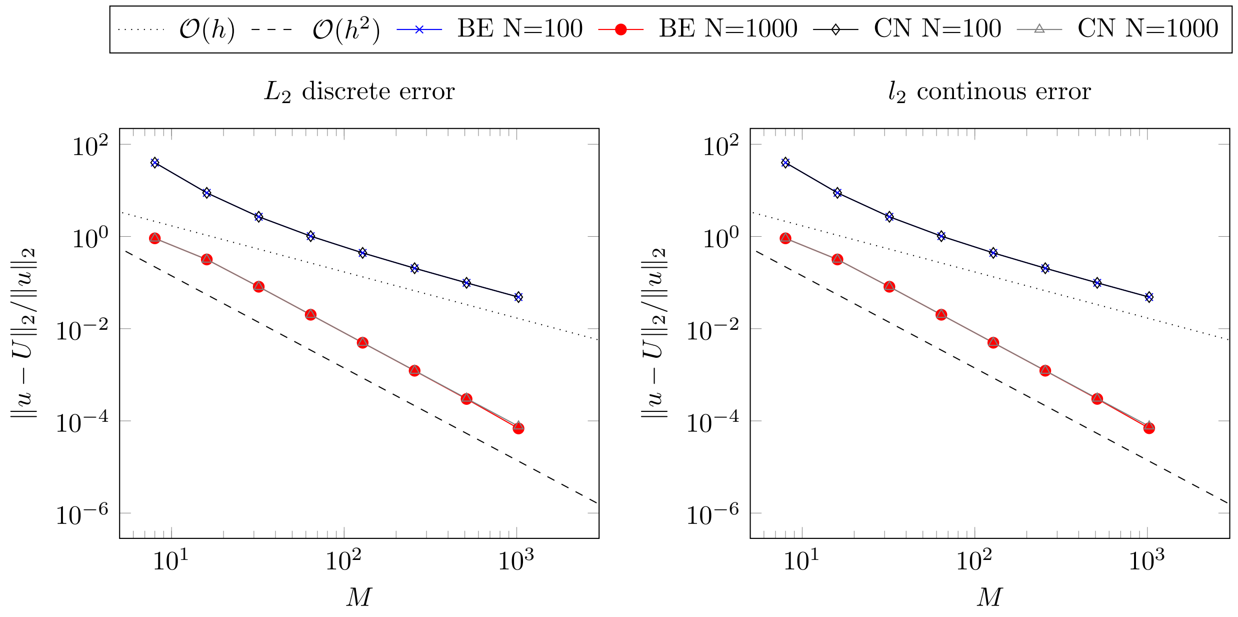


Figure 8: Convergence plot, t refinement (2b)

and then strictly decreasing from the apex towards the right boundary at $x = 1$. When this is no longer the case we say that the solution has broken, and the time for which this happened for our solution was at $t^* \approx 0.055$. Figure 10 shows the solution sampled around the time of breaking.

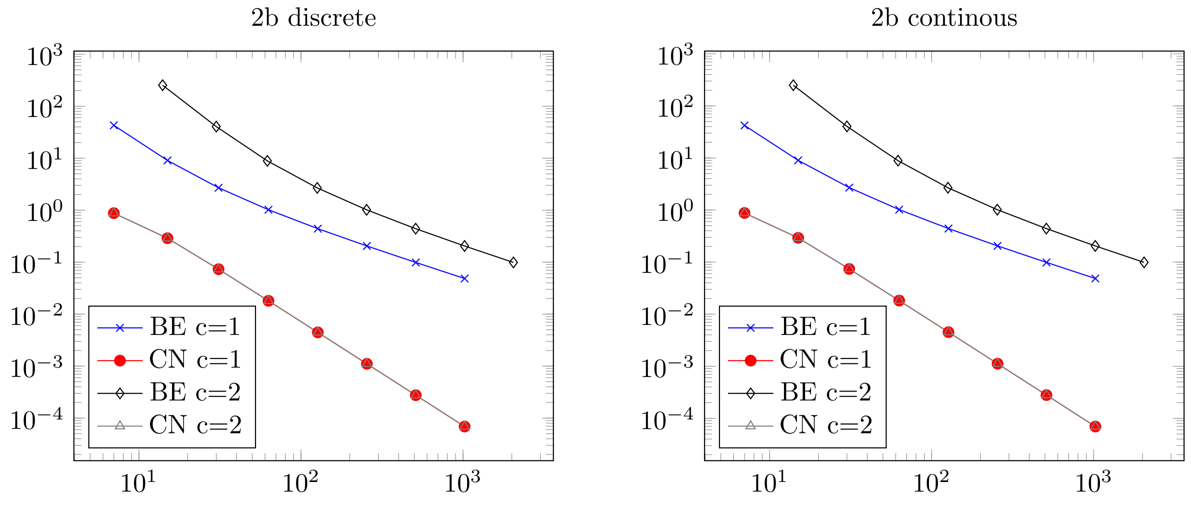


Figure 9: Convergence plot, kch refinement (2b)

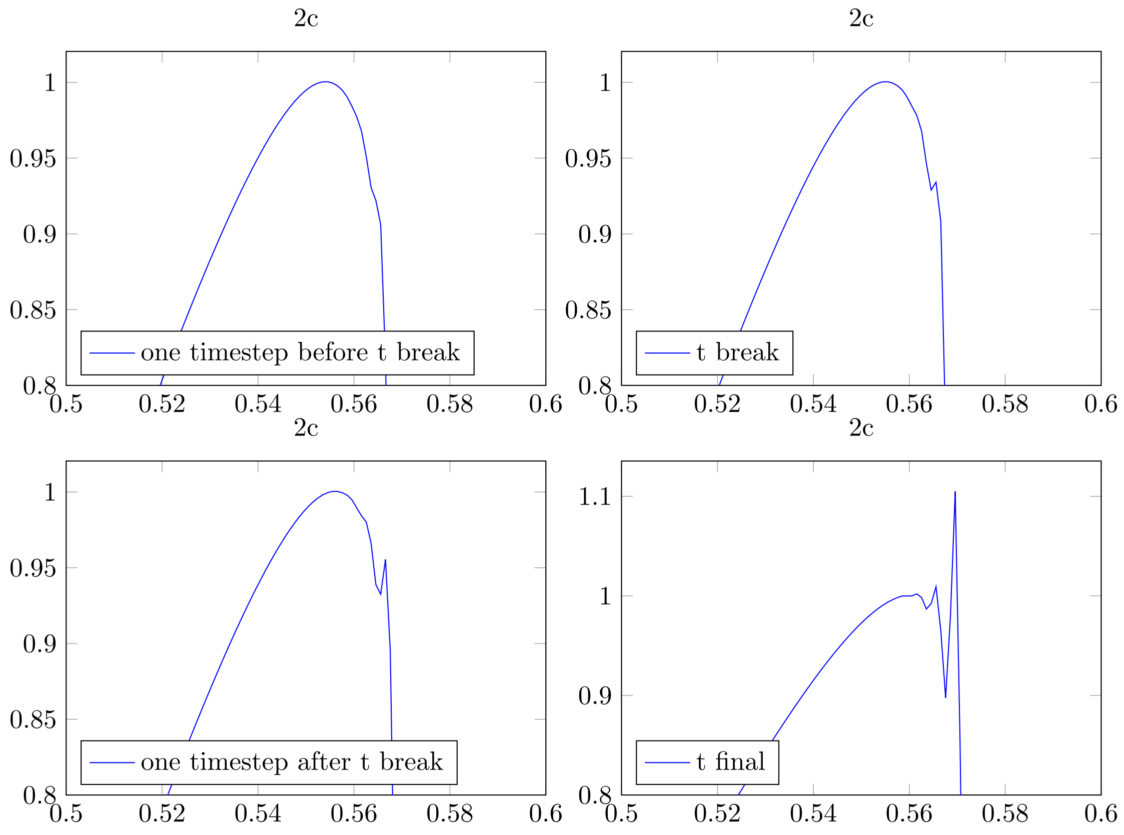


Figure 10: Numerically computed solution to the Invicid Burgers' equation around the time of breaking.

4 Laplace equation in two dimensions

In this section, we will solve the two-dimensional Laplace equation on a quadratic domain

$$u_{\mathrm{xx}} + u_{\mathrm{yy}} = 0,\ (x, y) \in \Omega := [0, 1]^2, \tag{13}$$

with boundary conditions on the edges of Ω

$$\begin{aligned} u(0, y) &= 0, \\ u(1, y) &= 0, \\ u(x, 0) &= 0, \\ u(x, 1) &= \sin(2\pi x). \end{aligned} \tag{14}$$

We will solve this equation numerically using a five point stencil, but first, we solve it analytically to provide a reference solution which can be compared with the numerical one.

4.1 Analytical solution

The solution of equation 13 can be found by separation of variables. First, asssume that we can write

$$u(x, y) = \alpha(x)\beta(y),$$

which implies that

$$u_{\mathrm{xx}} + u_{\mathrm{yy}} = \alpha''(x)\beta(y) + \alpha(x)\beta''(y) = 0,$$

where the prime markers $'$ denote differentiation of the single variable functions $\alpha(x)$ and $\beta(y)$. Rearranging, we get that

$$\frac{\alpha''(x)}{\alpha(x)} = \frac{\beta''(y)}{\beta(y)} = c$$

must be constant, since α and β are functions of indepentent variables. Thus, we have two second order differential equations

$$\begin{aligned} \alpha''(x) - c\alpha(x) &= 0, \\ \beta''(x) - c\beta(x) &= 0, \end{aligned}$$

with boundary conditions

$$\begin{aligned} \alpha(0) = \alpha(1) = \beta(0) &= 0, \\ \alpha(x)\beta(1) &= \sin(2\pi x). \end{aligned}$$

Setting $\beta(1)$ to 1 yields $\alpha(x) = \sin(2\pi x)$, so that $\alpha''(x) = -4\pi^2\alpha(x)$ where $y = 1$, we find that $c = -4\pi^2$. Solving the equation for $\beta(y)$, we find that

$$\beta(y) = b_1 e^{\sqrt{c}y} + b_2 e^{-\sqrt{c}y}.$$

Inserting $c = 4\pi$ and the boundary condtitions $\beta(0) = 0$ and $\beta(1) = 1$, we get

$$\beta(y) = \frac{\sinh(2\pi y)}{\sinh(2\pi)},$$

and finally

$$u(x, y) = \frac{\sin(2\pi x) \cdot \sinh(2\pi y)}{\sinh(2\pi)}.$$

4.2 Numerical solution

We solve the equation numerically by discretizing the domain $\Omega = [0,1]^2$, approximate the equation on that domain using a five point stencil, and solving the approximated system. The domain is discretized with $M+2$ and $N+2$ points in the x and y direction, so that there are M and N internal points in each direction. The total system to be solved is thus $M \times N$ points, as the boundaries are known.

Rewriting Laplace's equation using central differences, we get

$$\begin{aligned}
\partial_x^2 u(x_m, y_n) &= \frac{1}{h^2}[u(x_{m-1}, y_n) + 2u(x_m, y_n) + u(x_{m+1}, y_n)] + \mathcal{O}(h^2) \\
&= \frac{1}{h^2}\delta_x^2 u(x_m, y_n) + \mathcal{O}(h^2), \\
\partial_y^2 u(x_m, y_n) &= \frac{1}{k^2}[u(x_\mathrm{m}, y_{n-1}) + 2u(x_m, y_n) + u(x_\mathrm{m}, y_{n+1})] + \mathcal{O}(k^2) \\
&= \frac{1}{k^2}\delta_y^2 u(x_m, y_n) + \mathcal{O}(k^2),
\end{aligned}$$

where (x_m, y_n) denote the point (m, n) in the grid. Adding these expressions, and naming our approximated solution with the shorthand notation $U_m^n := u(x_m, y_n)$, we find that the Laplace equation can be approximated

$$0 = \partial_x^2 u(x_m, y_n) + \partial_y^2 u(x_m, y_n) \approx \frac{1}{h^2}\delta_x^2 U_m^n + \frac{1}{k^2}\delta_y^2 U_m^n,$$

or, simplifying the notation with the notation visualized in figure 11,

$$\frac{1}{k^2}(U_\text{above} + U_\text{below} - 2U_\text{center}) + \frac{1}{h^2}(U_\text{left} + U_\text{right} - 2U_\text{center}) = 0.$$

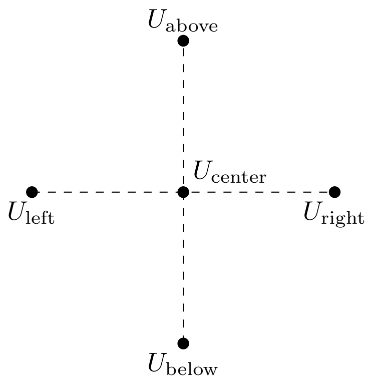


Figure 11: The five-point stencil corresponding to central difference differentiation in both the x- and y-direction.

We will now construct the matrix A such that we can write our equation as the matrix equation $AU = b$, where U is the flattened solution, and $b = \vec{b}$ contains the boundary conditions of the system, which will be explained in more detail below. Ignoring firstly the above and below nodes of the stencil, we can easily set up a matrix A' in the same way as in Section 1. Note that this is done only in order to clarify the derivation – the matrix A' is merely a "stepping stone" – not a useful result.

$$A'U = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix}\begin{bmatrix} U_1^n \\ U_2^n \\ \vdots \\ U_{M-1}^n \\ U_M^n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{M-1}^n \\ b_M^n \end{bmatrix},$$

Note also that this equation only considers one particular value of y, corresponding to n. The boundary conditions on the right hand side are zero for all internal points, while the values along the edges, that is $n = 1, N$ or $m = 1, M$, are set according to (14).

In order to actually solve our entire system, we must include the nodes above and below the center as well. This can be done by considering a much larger matrix A and a much longer vector U. The latter being a stacked vector containing all M elements $U_1^1, \ldots, U_M^1$, followed by $U_1^2, \ldots U_M^2$ and so on. In this formulation of the problem, the values U_{right} and U_{left} correspond to the neighbouring points in U. The above and below nodes – instead of being above and below U_{center} – are now to the sides, M nodes away, as illustrated in figure 12.

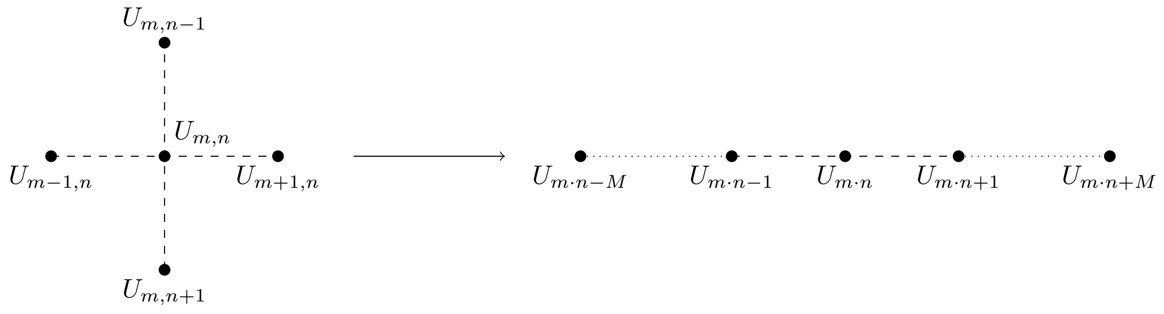


Figure 12: By flattening the five-point stencil, we can write the system of equations, which is then on the form $AU = 0$.

We thus write

$$AU = \begin{bmatrix} \frac{-2}{h^2}+\frac{-2}{k^2} & \frac{1}{h^2} & & \frac{1}{k^2} & & \\ \frac{1}{h^2} & \frac{-2}{h^2}+\frac{-2}{k^2} & \frac{1}{h^2} & & \frac{1}{k^2} & \\ & \frac{1}{h^2} & \frac{-2}{h^2}+\frac{-2}{k^2} & 0 & & \frac{1}{k^2} \\ & \ddots & \ddots & \ddots & & \\ \frac{1}{k^2} & & 0 & \frac{-2}{h^2}+\frac{-2}{k^2} & \frac{1}{h^2} & \\ & \frac{1}{k^2} & & \frac{1}{h^2} & \frac{-2}{h^2}+\frac{-2}{k^2} & \frac{1}{h^2} \\ & & \frac{1}{k^2} & & \frac{1}{h^2} & \frac{-2}{h^2}+\frac{-2}{k^2} \end{bmatrix} \begin{bmatrix} U_1 \\ \vdots \\ U_{\text{m}} \\ \vdots \\ U_{N\times m} \\ \vdots \\ U_{N\times M} \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \\ \vdots \\ b_{N\times m} \\ \vdots \\ b_{N\times M} \end{bmatrix} = b,$$

which can be solved. Note that the matrix is *not* Toeplitz! There are zeros on the upper and lower diagonal, correponding to the nodes that have less than four neighbours, ie. the nodes on the border. These nodes are handled with the boundary conditions from b, whose values are set in points corresponding to the borders of the system, as mentioned above. Thus, the final equation will be on the form $AU = b$, where b and U are flattened matrices, i.e. vectors, of length $N \times M$, while A is a matrix of size $(N \times M)^2$

The large matrix A is also showed in a more managable way in figure 13, where it is plotted as a heatmap. By noticing its recursive structure, one may realize that the matrix can be constructed by a Kronecker sum.

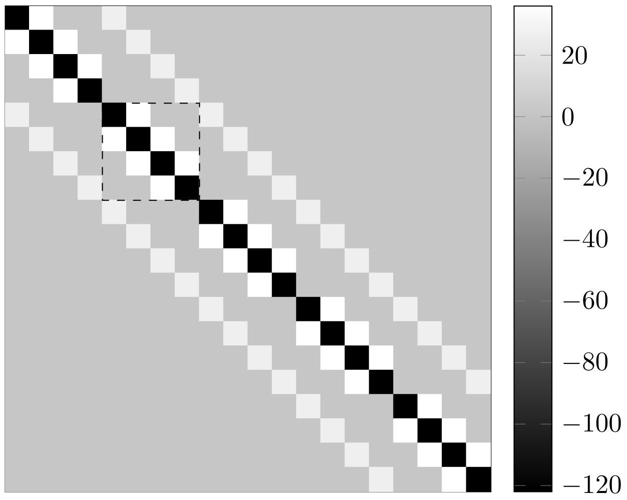


Figure 13: The five point stencil matrix for the case $M = 4$, $N = 5$. Notice the recursive structure.

This procedure is discussed in depth in 7 where the Biharmonic equation is solved using the fast Poisson solver. For now however, we will only take the observation about the Kronecker sum as a convenient way to implement the construction of our matrix. Let K_M be the system matrix of the one dimensional finite central difference scheme of size m introduced as A' in equation (4.2). That is, the $M \times M$ matrix

$$K_M = \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix}.$$

The full matrix A is then compactly written as

$$A = \frac{1}{h^2}K_N \oplus \frac{1}{k^2}K_M = \frac{1}{h^2}K_N \otimes I_M + I_N \otimes \frac{1}{k^2}K_M. \tag{15}$$

This matrix is colorfully illustrated in figure 13.

Using the method described above, we have computed the solution to equation 13, and the results are shown in figure 14. An error analysis showing the error with varying grid resolutions in both the x- and the y-direction is presented in 15.

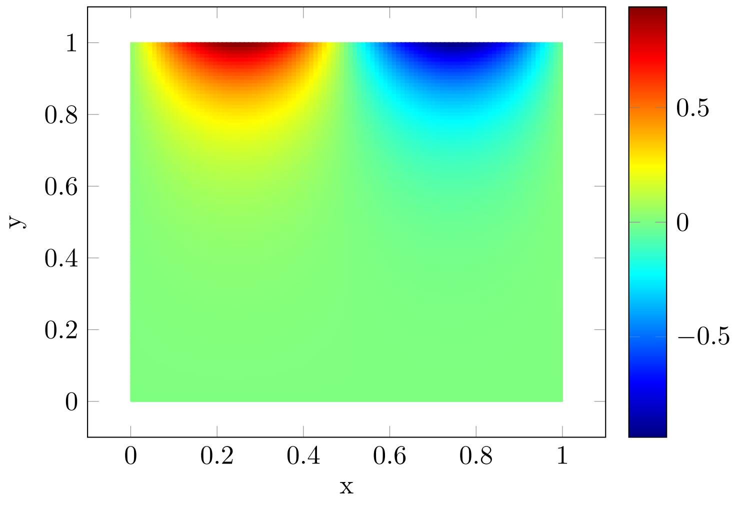


Figure 14: The numerically computed solution to the Laplace equation, using $N = M = 100$.

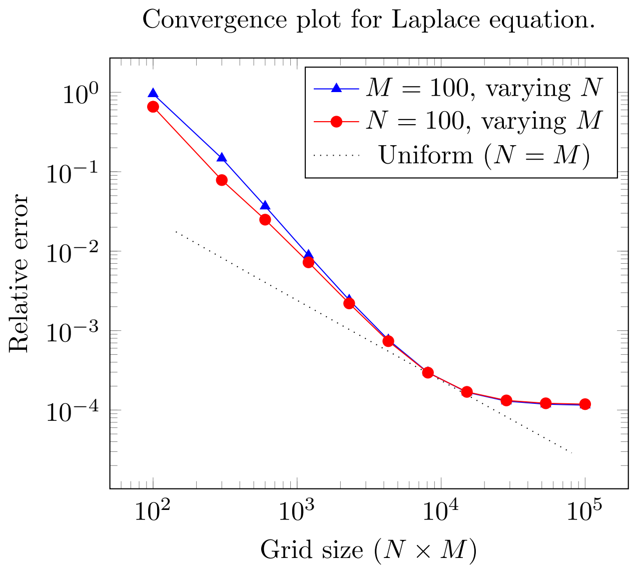


Figure 15: Convergence plot for varying N and M, corresponding to the y- and the x-direction, respectively.

5 Linearized Korteweg-De Vries equation in one dimension

In this section, we will study the one-dimensional linearized Korteweg-De Vries equation

$$\frac{\partial u}{\partial t} + \left(1+\pi^2\right)\frac{\partial u}{\partial x} + \frac{\partial^3 u}{\partial x^3} = 0 \qquad (t \geq 0) \quad (-L/2 \leq x \leq +L/2), \tag{16}$$

where the solution $u = u(x,t)$ is subject to the periodic boundary condition

$$u(x+L,t) = u(x,t).$$

5.1 Analytical solution

Let us solve the Korteweg-De Vries equation with separation of variables, writing one solution as

$$u_n(x,t) = X_n(x)\,T_n(t).$$

For the spatial part of the solution, let us use the periodic ansatz

$$X_n(x) = e^{iq_nx} \quad \text{with wavenumbers} \quad q_n = 2\pi L/n.$$

Now insert $u_n(x,t) = X_n(x)\,T_n(t)$ into equation (16) and divide by $X_n(x)\,T_n(t)$ to get

$$\underbrace{\frac{\dot{T}_n(t)}{T_n(t)}}_{-i\omega_n} + \underbrace{\left(1+\pi^2\right)\frac{X_n'(x)}{X_n(x)} + \frac{X_n'''(x)}{X_n(x)}}_{i\omega_n} = 0.$$

The first term is a function of t only and the remaining terms are a function of x only, so they must be constant. In anticipation of the result, we label the constants $\mp i\omega_n$. The temporal part gives

$$T_n(t) = e^{-i\omega_n t},$$

while inserting our ansatz $X_n(x) = e^{iq_nx}$ into the spatial part gives the dispersion relation

$$\omega_n = (1+\pi^2)q_n - q_n^3.$$

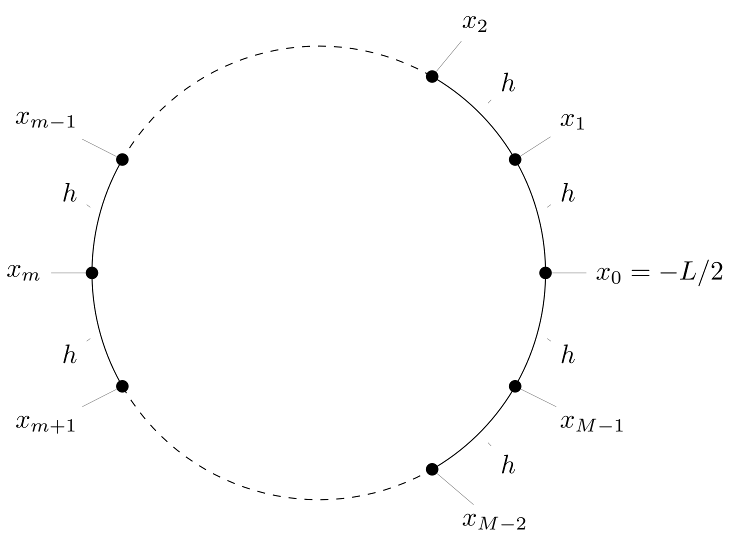
The solution $u_n(x,t)$ is now fully specified. Due to the linearity of equation (16) and the periodic boundary condition, we can superpose multiple solutions $u_n(x,t)$ into a general solution

$$u(x,t) = \sum_{n=-\infty}^{+\infty} c_n \exp\left(i(q_nx - \omega_nt)\right), \tag{17}$$

which is a sum of plane waves propagating at different frequencies.

5.2 Numerical solution method

To find a numerical solution $U_m^n = U(x_m,t_n) \approx u(x_m,t_n) = u_m^n$ of the Korteweg-De Vries equation, we will discretize it with central differences in space and integrate over time with the Forward Euler method and the Crank-Nicholson method. We will find the solution on the periodic spatial grid



of M points. For the first spatial derivative, we use the central difference

$$\frac{\partial u_m^n}{\partial x} = \frac{u_{m+1}^n - u_{m-1}^n}{2h} + \mathcal{O}(h^2).$$

We repeat the same finite difference three times to approximate the third order spatial derivative as

$$\frac{\partial^3 u_m^n}{\partial x^3} = \frac{u_{m+3}^n - 3u_{m+1}^n + 3u_{m-1}^n - u_{m-3}^n}{8h^3} + \mathcal{O}(h^2).$$

Inserting these approximations into equation (16), we get the intermediate result

$$\frac{\partial u_m^n}{\partial t} = -\left(1+\pi^2\right)\frac{u_{m+1}^n - u_{m-1}^n}{2h} - \frac{u_{m+3}^n - 3u_{m+1}^n + 3u_{m-1}^n - u_{m-3}^n}{8h^3} + \mathcal{O}(h^2) \equiv F(u^n) + \mathcal{O}(h^2).$$

For later convenience, we write the Forward Euler method and Crank-Nicholson method collectively with the θ-method. This gives the final system of difference equations for the numerical solution

$$\frac{U_m^{n+1} - U_m^n}{k} = (1-\theta)F(U^n) + \theta F(U^{n+1}), \tag{18}$$

where the Forward Euler method or the Crank-Nicholson method is obtained by setting $\theta = 0$ or $\theta = 1/2$, respectively. In matrix form, the system can be written

$$(I - \theta k A)\, U^{n+1} = \big(I + (1-\theta)\, kA\big)\, U^n, \tag{19}$$

where $U^n = \begin{bmatrix} U_0^n & \dots & U_{M-1}^n \end{bmatrix}^T$ and $A =$

$$\frac{-(1+\pi^2)}{2h}\begin{bmatrix}
0 & +1 & & & & & & & -1 \\
-1 & 0 & +1 & & & & & & \\
 & -1 & 0 & +1 & & & & & \\
 & & -1 & 0 & +1 & & & & \\
 & & & \ddots & \ddots & \ddots & & & \\
 & & & & -1 & 0 & +1 & & \\
 & & & & & -1 & 0 & +1 & \\
 & & & & & & -1 & 0 & +1 \\
+1 & & & & & & & -1 & 0
\end{bmatrix} - \frac{1}{8h^3}\begin{bmatrix}
0 & -3 & 0 & +1 & & & -1 & 0 & +3 \\
+3 & 0 & -3 & 0 & +1 & & & -1 & 0 \\
0 & +3 & 0 & -3 & 0 & +1 & & & -1 \\
-1 & 0 & +3 & 0 & -3 & 0 & +1 & & \\
 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \\
 & & -1 & 0 & +3 & 0 & -3 & 0 & +1 \\
+1 & & & -1 & 0 & +3 & 0 & -3 & 0 \\
0 & +1 & & & -1 & 0 & +3 & 0 & -3 \\
-3 & 0 & +1 & & & -1 & 0 & +3 & 0
\end{bmatrix}.$$

where we have imposed periodic boundary conditions $U_m^n = U_{m+M}^n$ by simply wrapping the spatial derivative stencils around the matrix. This is equivalent to calculating stencil indices modulo M, consistent with our circular grid.

We then solve the system by preparing U^0 from the initial condition $u(x, 0)$ and solve equation (19) repeatedly to step forward in time. Note that with constant steps h and k in both space and time, the matrices in equation (19) are constant. To save both memory and time, we represent them with sparse matrices. [10] In addition, to efficiently solve the same system with different right hand sides many times, we LU-factorize the sparse matrix for $I - \theta k A$. [9] Note that with the Forward Euler method, $\theta = 0$ and this matrix reduces to the identity, so there is no system to solve – the U^{n+1} is given by simply multiplying the right side.

Next, we test our numerical solution on the problem defined by the initial condition $u(x, 0) = \sin(\pi x)$ on $x \in [-1, +1]$ with $L = 2$. The analytical solution 17 then gets nonzero contributions only from $n = \pm 1$, which gives the analytical solution $u(x, t) = \sin(\pi(x - t))$. As shown in figure 16, the solution represents a sine wave traveling with velocity 1 to the right.

In figure 17, we compare snapshots of the numerical solution at $t = 1$ from the Forward Euler method and the Crank-Nicholson method. Note that the Crank-Nicholson method approaches the exact solution with only $N = 10$ time steps and under hundred spatial grid points M. In contrast, the Forward Euler method seems to become unstable as the spatial resolution is increased, even with $N = 100000$ time steps.

The convergence plot at $t = 1$ in figure 18 supports our suspicions. As we expect from the central finite differences, both methods show second order convergence in space for sufficiently refined grids. But the Forward Euler method diverges as h decreases, although the divergence is delayed by also decreasing k. The Crank-Nicholson method remains stable with much fewer time steps and much finer spatial grids.

5.3 Stability analysis

Motivated by the examples of the Euler method and the Crank-Nicholson method, we perform a Von Neumann analysis of their stability. Just like the exact solution, the numerical solution is subject to periodic boundary conditions in space and can therefore be expanded in a Fourier series [2]

$$U_m^n = U(x_m, t_n) = \sum_l C_l^n \exp(i q_l x_m). \tag{20}$$

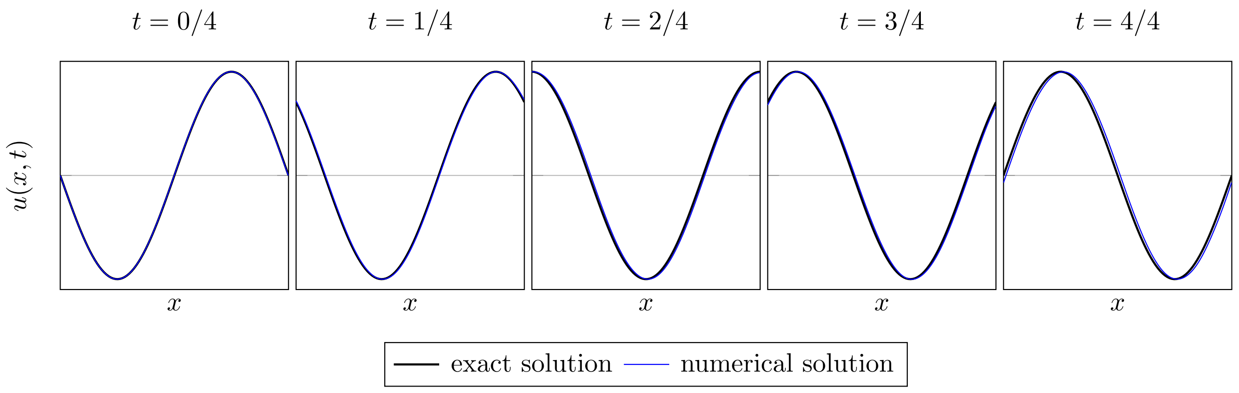


Figure 16: Comparison between the time evolution of the exact solution $u(x, t) = \sin(\pi(x - t))$ and the numerical solution from the Crank-Nicholson method with $h = 1/799$ and $k = 1/99$.

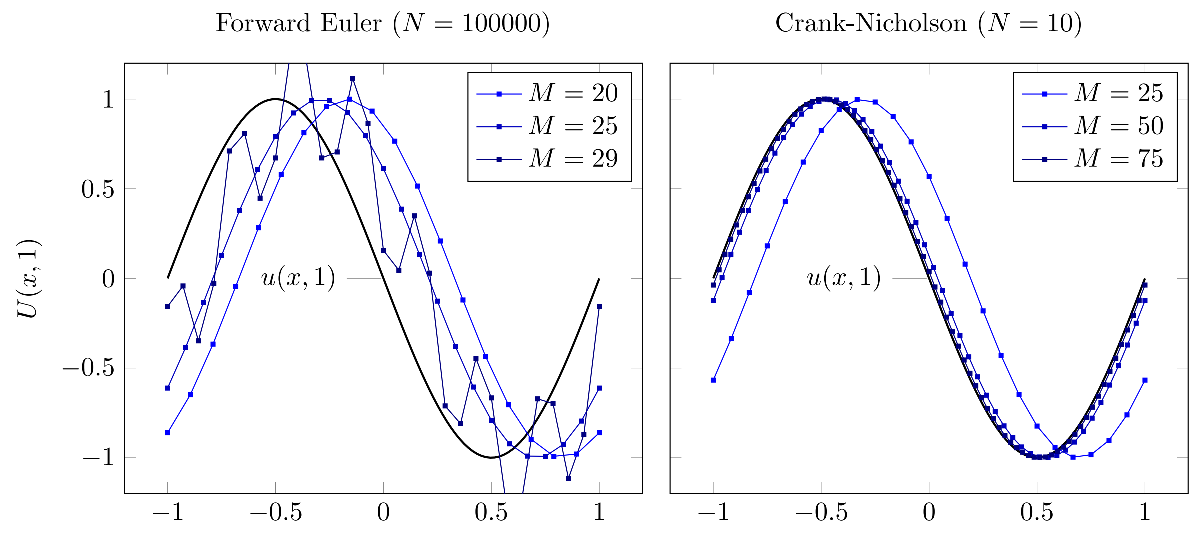


Figure 17: Snapshots of the numerical solution $U(x, 1)$ and the exact solution $u(x, 1)$ for a constant number of time steps N, but varying number of grid points M with the Forward Euler and Crank-Nicholson method. The left plot is meant to demonstrate the downfall of the Euler method and is not supposed to look pretty.

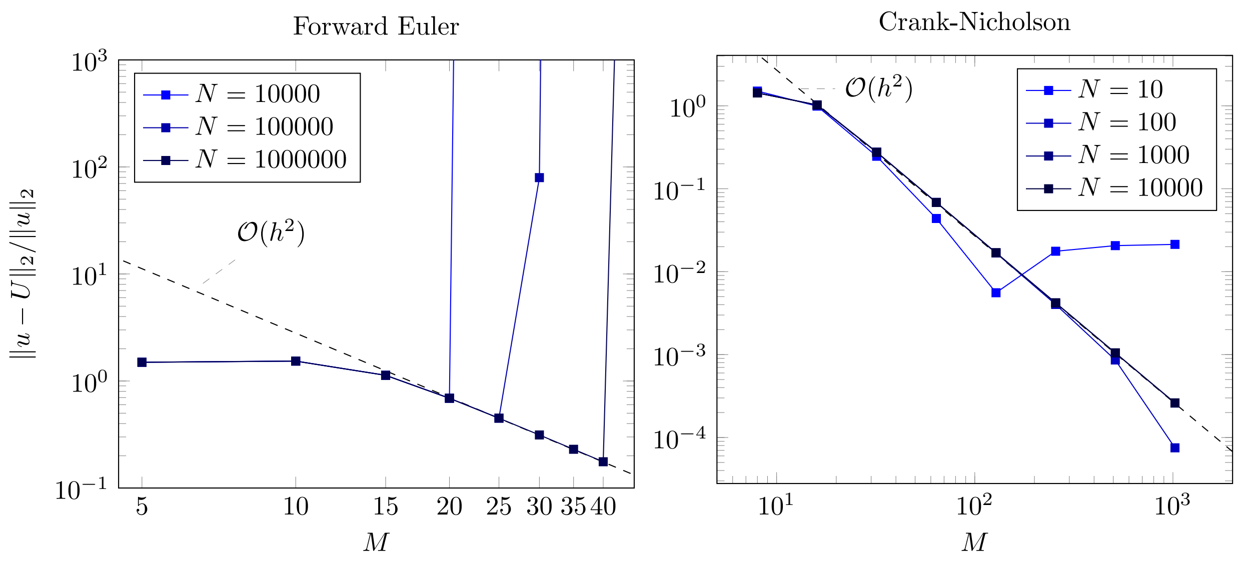


Figure 18: Convergence plots with the discrete L_2 error of the numerical solution $U(x, t)$ for the Forward Euler and Crank-Nicholson method on the problem defined by the exact solution $u(x, t) = \sin(\pi(x - t))$.

Consider now a single Fourier mode $C_l^n \exp(iq_l x_m)$ in this series. Inserting it into equation (18), dividing by $\exp(iq_l x_m)$ and expanding exponentials using Euler's identity $e^{ix} = \cos x + i \sin x$ gives

$$\frac{C_l^{n+1} - C_l^n}{k} = i\left((1-\theta)\,C_l^n + \theta C_l^{n+1}\right) f(q_l), \quad \text{where } f(q_l) = \left(-\left(1+\pi^2\right)\frac{\sin(q_l h)}{h} - \frac{\sin^3(q_l h)}{h^3}\right).$$

Now look at the amplification factor $G_l = C_l^{n+1}/C_l^n$ of Fourier mode l over one time step. With $\theta = 1/2$, the Crank-Nicholson method gives

$$G_l = \frac{1 + ikf(q_l)/2}{1 - ikf(q_l)/2} \quad \Longrightarrow \quad |G_l| = 1. \tag{21}$$

The amplitude of all Fourier modes is thus preserved over time independently of k and h, and we say the Crank-Nicholson method is **unconditionally stable**.

The Euler method has $\theta = 0$ and gives

$$G_l = 1 + ikf(q_l) \quad \Longrightarrow \quad |G_l| = \sqrt{1 + k^2 f(q_l)^2}. \tag{22}$$

Since $\left|\sin(q_l h)\right| \leq 1$ for all q_l, we can bound $f(q_l)$ by

$$\left|f(q_l)\right| \leq \frac{(1+\pi^2)}{h} + \frac{1}{h^3} = \frac{1}{h^3}\left((1+\pi^2)h^2 + 1\right) \leq \frac{1}{h^3}\left((1+\pi^2)L^2 + 1\right).$$

Then $|G_l| = \sqrt{1 + O(k^2/h^6)} > 1$ for all h and k, so each Fourier mode is amplified over time. But the Von Neumann stability criterion $|G_l| \leq 1 + O(k)$ [6] is still attained with $k \leq O(h^6)$, so the Forward Euler method is **conditionally stable**. Only if $k/h^6 << 1$ does it remain stable, which explains the divergence for decreasing h and fixed k we found in figure 18 and why this is delayed by also decreasing k.

Thus, while the Euler method in theory is stable, it is unstable for practical combinations of k and h. The Crank-Nicholson method is far superior, as it remains stable over time and allows both smaller resolution in time and greater resolution in space.

5.4 Time evolution of norm

The stability of the finite difference methods can be even better illustrated by investigating the time evolution of the L_2-norm of the solution. To this end, we will first show that the L_2-norm of the analytical solution is preserved over time. Then we will investigate the time evolution of the norm of numerical solutions.

The L_2-norm of the analytical solution is defined as

$$\|u(x,t)\|_2 = \left(\frac{1}{2}\int_{-L/2}^{+L/2} |u(x,t)|^2 \,\mathrm{d}x\right)^{1/2}.$$

Now insert the solution 17 and use orthogonality of the complex exponentials to get

$$\int_{-L/2}^{+L/2} \mathrm{d}x \left|u(x,t)\right|^2 = \sum_{m,n} c_m c_n^* \exp\left(i(\omega_n - \omega_m)t\right) \underbrace{\int_{-L/2}^{+L/2} \exp\left(i(q_m - q_n)x\right)\mathrm{d}x}_{L\delta_{mn}} = L\sum_m |c_m|^2.$$

The final sum is independent of time, so the L_2-norm is indeed conserved.

We now investigate the norm of the numerical solution with the initial gaussian $u(x,0) = \exp\left(-x^2/0.1\right)$. The time evolution illustrated in figure 19 shows how multiple modes are activated. In figure 20, we show

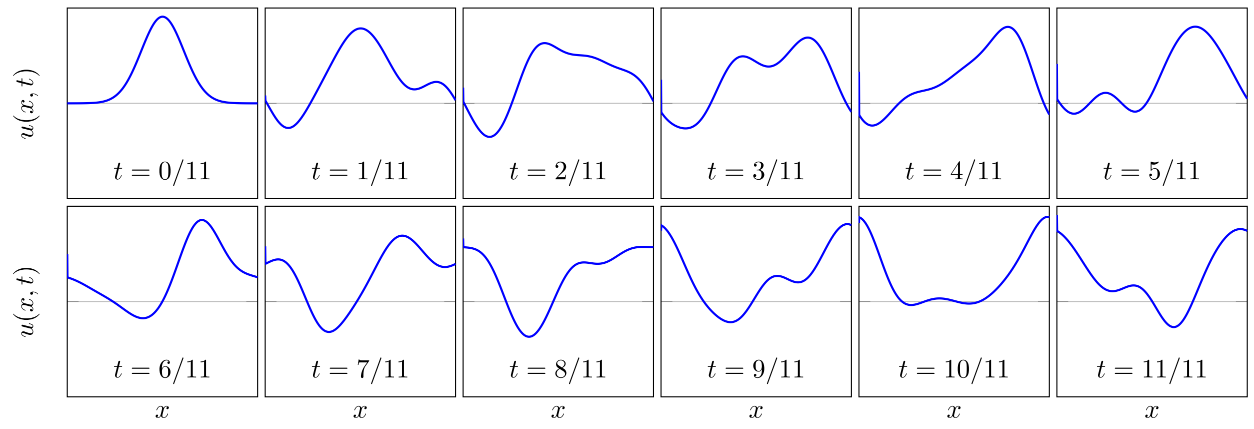


Figure 19: Time evolution of a initial gaussian $u(x, 0) = \exp(-x^2/0.1)$ computed from the Crank-Nicholson method on a grid with $M = 800$ points in space and $N = 100$ points in time.

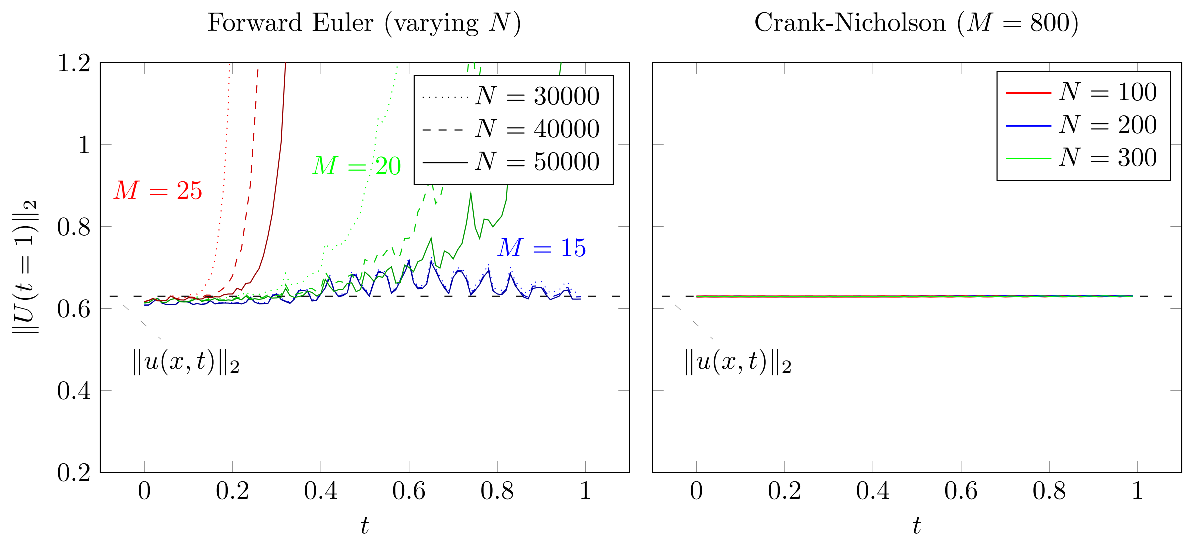


Figure 20: Time evolution of the discrete L_2-norm of the initial gaussian $u(x, 0) = \exp(-x^2/0.1)$ computed from the Euler and the Crank-Nicholson method, on different grids.

how the norm of the numerical solution evolves over time. The Euler method diverges even with tiny time steps, reflecting the amplification factor $G_l > 1$ found in equation (22). In contrast, the Crank-Nicholson metohd is always stable and preserves the norm of the solution, reflecting the amplification factor $G_l = 1$ found in equation (21).

The stability of the Crank-Nicholson method and the property that it preserves the amplitude of Fourier modes make it an optimal method for equations like the Korteweg-De Vries equation, where the analytical solution is known to have a constant norm.

6 Poisson equation in one dimension using finite element method

In this section, we will again solve the Poisson equation

$$-\frac{\partial^2 u}{\partial x^2} = f(x), \quad u(a) = \alpha, \quad u(b) = \beta, \quad (a \leq x \leq b) \tag{23}$$

subject to Dirichlet conditions, but this time using finite elements instead of finite differences.

6.1 Analytical solution

The solution to the Poisson equation is the same as in 2, but with $f(x) \to -f(x)$, so that

$$u(x) = C_1 + C_2 x - \int^x \mathrm{d}x' \int^{x'} \mathrm{d}x'' f(x''). \tag{24}$$

6.2 Weak formulation for the exact solution

To derive a finite element method, we will first derive a **weak formulation** of 23 in a way inspired by [1]. First, we split the solution into two terms

$$u(x) = \hat{u}(x) + r(x), \quad \text{with} \quad \hat{u}(a) = \hat{u}(b) = 0 \quad \text{and} \quad r(x) = \alpha\frac{x-b}{a-b} + \beta\frac{x-a}{b-a}. \tag{25}$$

Note that $u''(x) = \hat{u}''(x)$ and $r(a) = \alpha$ and $r(b) = \beta$. The purpose of this splitting is that $\hat{u}$ solves 23 with homogenuous Dirichlet boundary conditions, while $r(x)$ **lifts** the values at the boundaries to satisfy the inhomogenuous boundary conditions.

Now insert 25 into equation (23), multiply it by an arbitrary **trial function** $v(x)$ and integrate both sides from a to b. We let $v(a) = v(b) = 0$ and use integration by parts on the left, dropping the boundary term $-[u'(x)v(x)]_a^b$. This gives the **weak formulation** of the problem:

$$\text{Find } \hat{u}(x) \text{ such that} \quad \int_a^b \mathrm{d}x\, \hat{u}'(x)v'(x) = \int_a^b \mathrm{d}x\, f(x)v(x) - \int_a^b \mathrm{d}x\, r'(x)v'(x) \quad \text{for all } v(x). \tag{26}$$

The weak formulation 26 is equivalent to the original boundary value problem 23. Any $u(x)$ that solves 23 also solves 26, and reversing the steps we just made shows that the converse is also true.

6.3 Weak formulation for the approximate solution

We have not made any approximations yet. The approximation lies in seeking a solution $U(x) \approx u(x)$ that belongs to a function space different from the one in which the exact solution $u(x)$ belongs. Here, we suppose $U(x)$ lies in the space of piecewise linear functions. We will then repeat the process above to derive a weak formulation for $U(x)$, similarly as for the exact solution.

To see how this works, we first divide the interval $[a, b]$ into the grid

$$a = x_0 < x_1 < \cdots < x_M < x_{M+1} = b \tag{27}$$

and let $U(x)$ be piecewise linear in each **finite element** $[x_i, x_{i+1}]$. Similarly to $u(x)$, we split the approximate solution into

$$U(x) = \hat{U}(x) + R(x), \quad \text{with} \quad \hat{U}(a) = \hat{U}(b) = 0 \quad \text{and} \quad R(x) = \begin{cases} \alpha\frac{x_1 - x}{x_1 - a} & (a \leq x \leq x_1) \\ 0 & (x_1 \leq x \leq x_M) \\ \beta\frac{x - x_M}{b - x_M} & (x_M \leq x \leq b) \end{cases}. \tag{28}$$

Now again insert 29 into 23, multiply by an arbitrary trial function $V(x)$ that vanishes at a and b, integrate from a to b and drop a boundary term. This leads to the weak formulation for the approximate solution:

$$\text{Find } \hat{U}(x) \text{ such that } \quad \int_a^b \mathrm{d}x\, \hat{U}'(x)V'(x) = \int_a^b \mathrm{d}x\, f(x)V(x) - \int_a^b \mathrm{d}x\, R'(x)V'(x) \quad \text{for all } V(x). \tag{29}$$

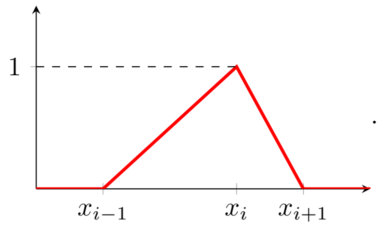
6.4 Numerical solution

To obtain a matrix equation for approximate solution $U(x)$, the next step is to expand

$$\hat{U}(x) = \sum_{i=0}^{M+1} \hat{U}_i \varphi_i(x) \quad \text{and} \quad V(x) = \sum_{i=0}^{M+1} V_i \varphi_i(x) \tag{30}$$

in a basis for the approximate solution function space, namely the piecewise linear functions on the grid 27. The most natural basis for this space are the functions

$$\varphi_i(x) = \begin{cases} (x - x_{i-1})/(x_i - x_{i-1}) & \text{if } x_{i-1} \le x \le x_i \\ (x_{i+1} - x)/(x_{i+1} - x_i) & \text{if } x_i \le x \le x_{i+1} \\ 0 & \text{otherwise} \end{cases} = \tag{31}$$



With this basis, the coefficients $\hat{U}_i = \hat{U}(x_i)$ are simply the values at the grid points, making it straightforward to plot the solution. Inserting this expansion into 29 gives

$$\sum_{i,j} \hat{U}_i V_j \int_a^b \mathrm{d}x\, \varphi_i'(x)\varphi_j'(x) = \sum_j V_j \int_a^b \mathrm{d}x\, \varphi_j(x) f(x)$$
$$- a \sum_j V_j \int_a^b \mathrm{d}x\, \varphi_0'(x)\varphi_j'(x) - b \sum_j V_j \int_a^b \mathrm{d}x\, \varphi_{M+1}'(x)\varphi_j'(x).$$

We can write this as the neat matrix equation $V^T A \hat{U} = V^T F$ by introducing

$$\hat{U} = [\hat{U}_1, \dots, \hat{U}_M]^T, \quad V = [V_1, \dots, V_M]^T, \quad A_{ij} = \int_a^b \mathrm{d}x\, \varphi_i'(x)\varphi_j'(x) \quad \text{and} \quad F_j = \int_a^b \mathrm{d}x\, \varphi_j(x) f(x).$$

for $0 \le i, j \le M+1$. Since this must hold for *any* $V(x)$ and thus V, we must have

$$A\hat{U} = F. \tag{32}$$

This is the matrix equation we will solve to find $U(x)$. After finding $\hat{U}$, we simply sum 30 and add $R(x)$ to find $U(x)$. Note that our particular choice of basis 31 gives the convenient property $U(x_i) = U_i$, so summing is not necessary in practice, and we can instead interpolate U_i between x_i to obtain $U(x)$.

To solve 32, we must first calculate the so-called **stiffness matrix** A and the **load vector** F. The former involves only the known basis functions and gives nonzero entries

$$A_{00} = \frac{1}{x_1 - x_0} \qquad A_{M+1M+1} = \frac{1}{x_{M+1} - x_M}$$
$$A_{ii} = \frac{1}{x_i - x_{i-1}} + \frac{1}{x_{i+1} - x_i} \qquad A_{ii+1} = A_{i+1i} = \frac{1}{x_{i+1} - x_i}.$$

The latter involves integrals over an arbitrary source function $f(x)$ times the basis functions $\varphi_j(x)$. This integral must be approximated numerically and should be split from x_{i-1} to x_i and x_i to x_{i+1} to properly handle the spike in $\varphi_j(x)$ at x_j. We use Gauss-Legendre quadrature to do these integrals. [7]

We now impose $\hat{U}(a) = \hat{U}(b) = 0$ by removing the first and last entries in the matrix equation *after* calculating the entire $(M+2) \times (M+2)$ system described above. This gives an $M \times M$ equation. Then we construct U by appending α and β at the beginning and end of the M-vector $\hat{U}$.

6.4.1 Uniform refinement

We test our method on four problems, shown in figure 20, with uniform elements $x_i - x_{i-1} = (b-a)/(M+1)$.

The approximate solutions resembles the exact solution with few points. For the symmetric Gaussian problems, it is vital to choose an odd number of grid points to capture the spike in the center. In the final problem, the source function diverges at the left boundary, but the numerical integration is still able to find a good numerical solution.

In all but the first problem, errors distribute non-evenly across the elements. Computational resources are wasted by using many points in areas where the solution varies slowly. These resources would be better spent by increasing the grid resolution in the areas where the error is large. This is the motivation for turning to adaptive refinement and non-uniform grids.

6.4.2 Adaptive refinement

Motivated by the uneven error distribution from using uniform elements, we will now do adaptive refinement, similarly to what we did in section 1.3. We start with a uniform grid and successively split those elements on which the error is largest. Contrary to what we did in section 1.3, we will not split only *one* element between each iteration of the numerical solution, but split *all* elements on which the error is greater than some reference error. This leaves us with less control over the number of elements, but in return we will see that the error strictly decreases in each iteration, eliminating the oscillating error in figure 3.

This time, we use two strategies that both involve the exact error:

1. **Average error strategy:** Split the interval $[x_m, x_{m+1}]$ with error
$$\|u(x) - U(x)\|_2 > 0.99 \, \frac{\|u(x) - U(x)\|_2}{N},$$
where N is the number of intervals. The safety factor $0.99 \approx 1$ ensures that intervals are split also when all errors are equal (up to machine precision), so the procedure does not halt unexpectedly.

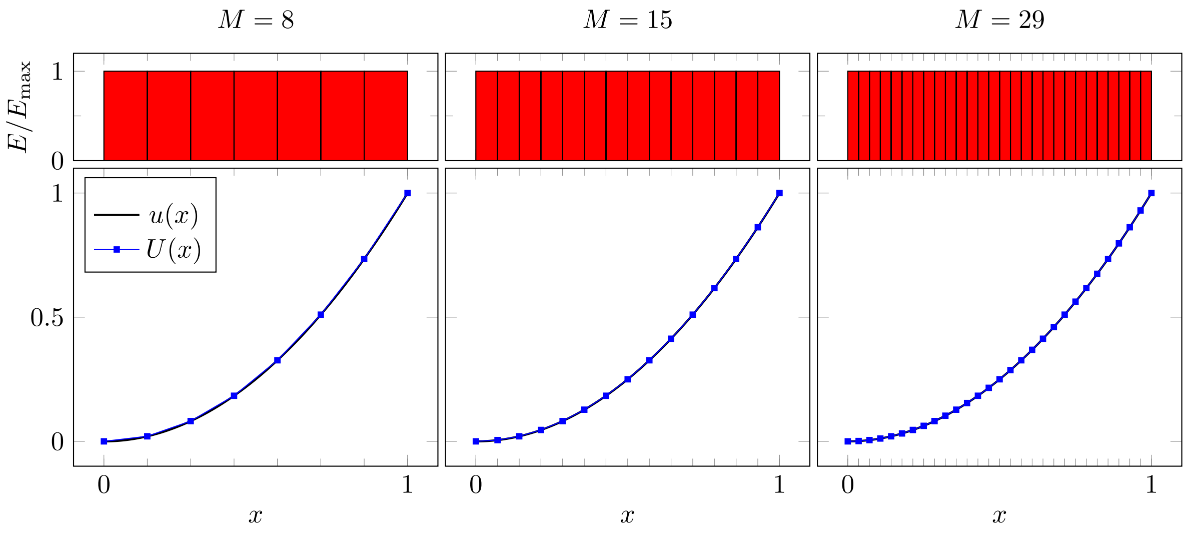
2. **Maximum error strategy:** Split the interval $[x_m, x_{m+1}]$ with error
$$\|u(x) - U(x)\|_2 > 0.70 \, \max \|u(x) - U(x)\|_2,$$
where N is the number of intervals.

In figure 20, we show how the errors distribute on the same four problems as in figure 20 using the average error strategy.

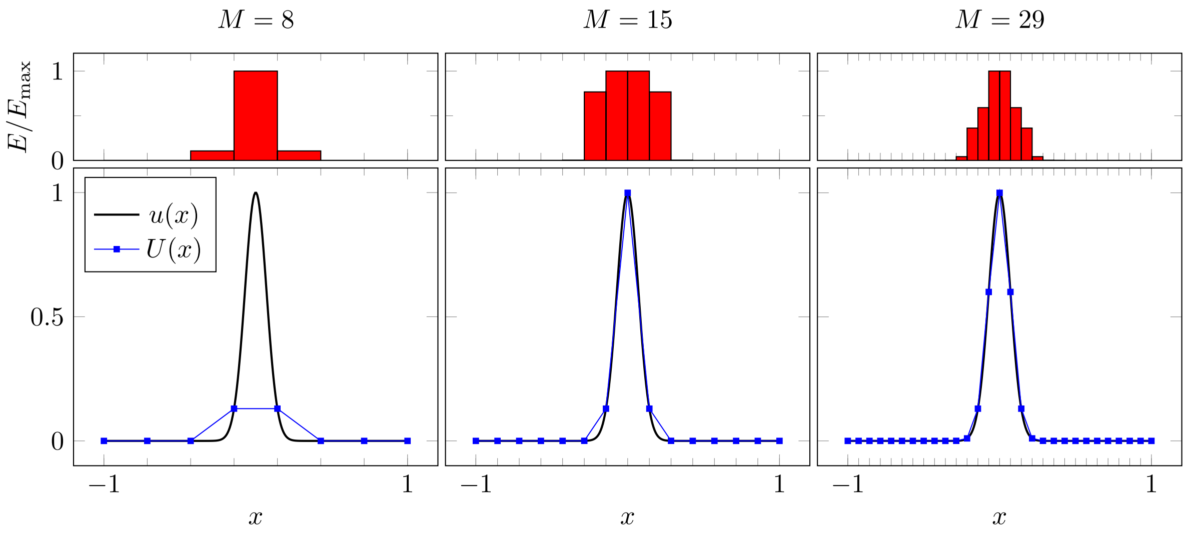
Observe how only elements with large error are refined, while others are left untouched. In the symmetric Gaussian problems, the refinement ensure that the middle element is split immediately if we do not start with a grid point at the peak. In the final problem, we see that it is almost only elements close to the left boundary where the source diverges that needs to be refined.

As discussed above, we see that the errors in the first problem distribute evenly on the initial uniform grid. This shows the importance of the safety factor 0.99 in the average error strategy. Without it, precision issues would make some elements skip the refinement criterion.

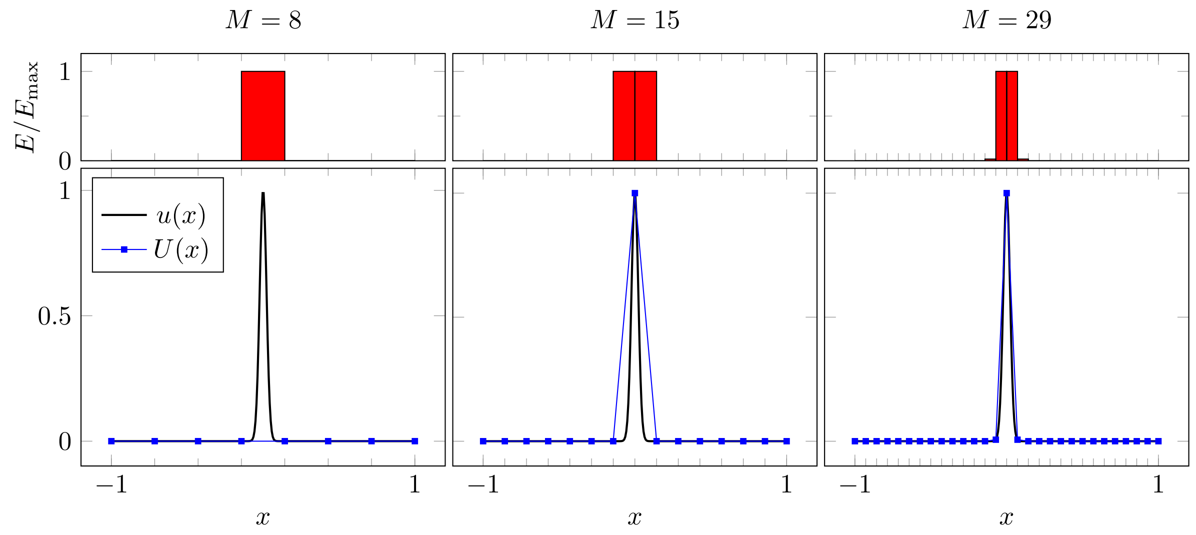
$$f(x) = -2, \quad u(0) = 0, \quad u(1) = 1$$



$$f(x) = -(40000x^2 - 200)\exp(-100x^2), \quad u(-1) = e^{-100}, \quad u(1) = e^{-100}$$



$$f(x) = -(4000000x^2 - 2000)\exp(-1000x^2), \quad u(-1) = e^{-1000}, \quad u(1) = e^{-1000}$$



$$f(x) = 2x^{-4/3}/9, \quad u(0) = 0, \quad u(1) = 1$$

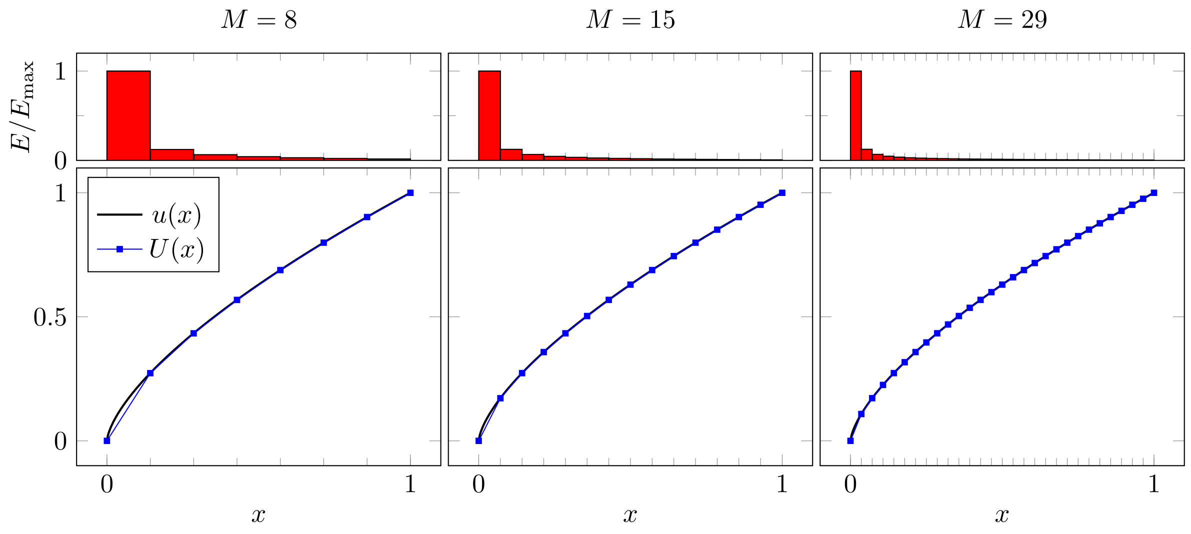


Figure 20: Uniform mesh refinement on four problems with the average error strategy, showing the evolution of an initial mesh whose elements are split in half over three iterations, along with the distribution of L_2-errors across the elements.

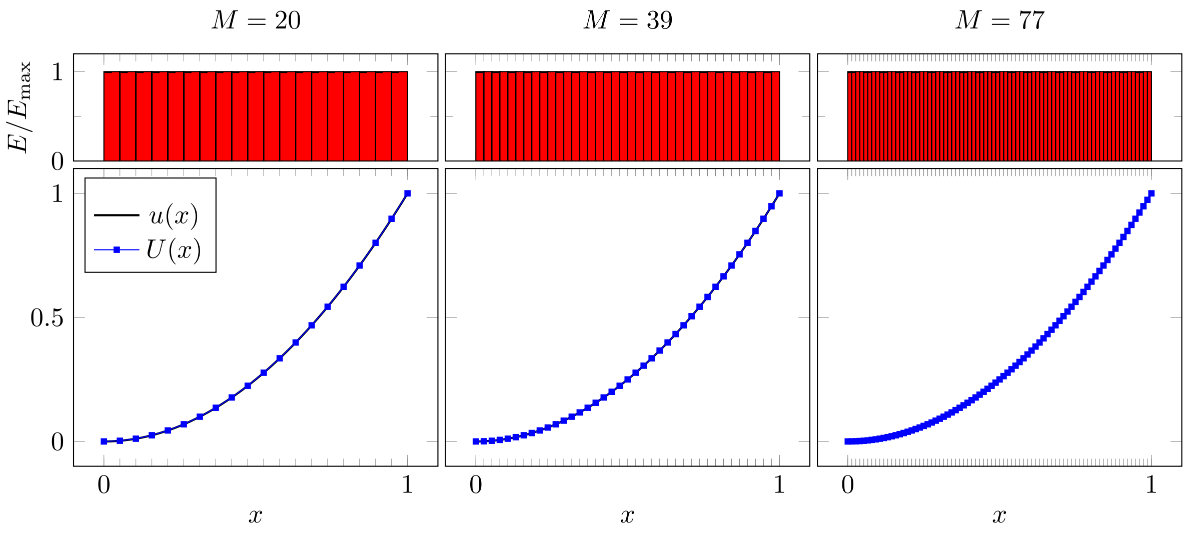
6.4.3 Comparison of convergence

Finally, in figure 21, we compare the convergence of uniform and adaptive refinement strategies. With uniform refinement, our finite element method yields second order convergence for the three first problems. In the fourth problem, the source $f(x) = 2x^{-4/3}/9$ diverges at the left boundary $x = 0$, so the integrals over it become inaccurate. This can explain the lower order convergence.

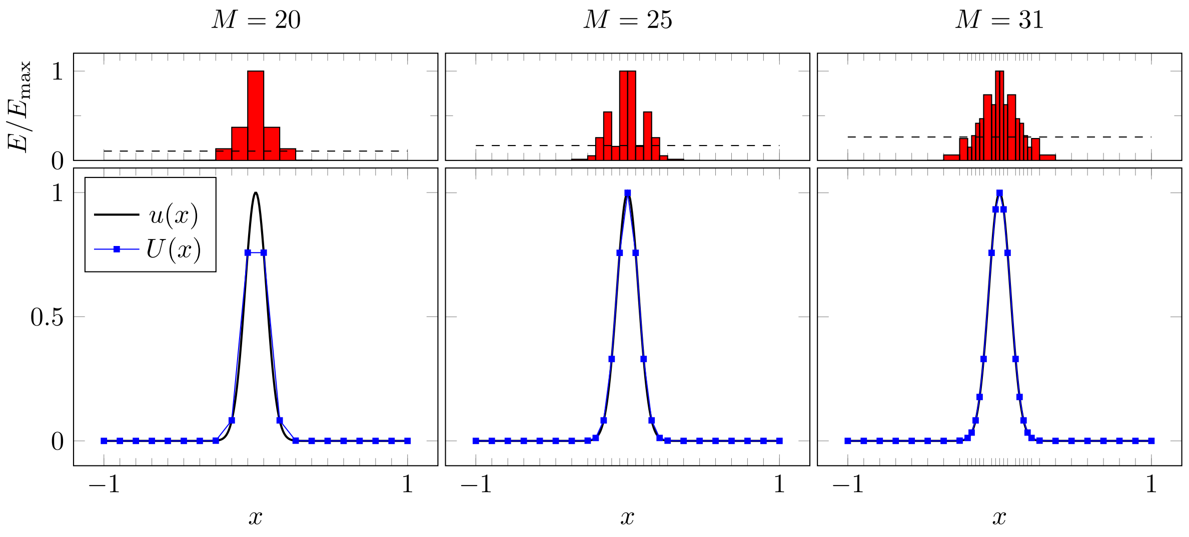
In all problems, adaptive refinement yields greater or equal accuracy for a given number of elements compared to uniform refinement. This is in contrast to what was the case for the finite difference method in figure 3, where adaptive refinement gave errors only comparable and usually larger than those from uniform refinement. It is only in the first problem that all strategies behave identically, as the errors here distribute evenly across the elements.

By splitting multiple intervals between each iteration of the numerical solution, we have eliminated the oscillating error pattern in figure 3. Now the error strictly decreases between each refinement of the grid. This suggests that the oscillating pattern is due to refinements where intervals with large error are present even after refining the element with greatest error. For example, it would be a bad idea to refine only *one* element in the first problem in figure 20, where errors are even across the elements.

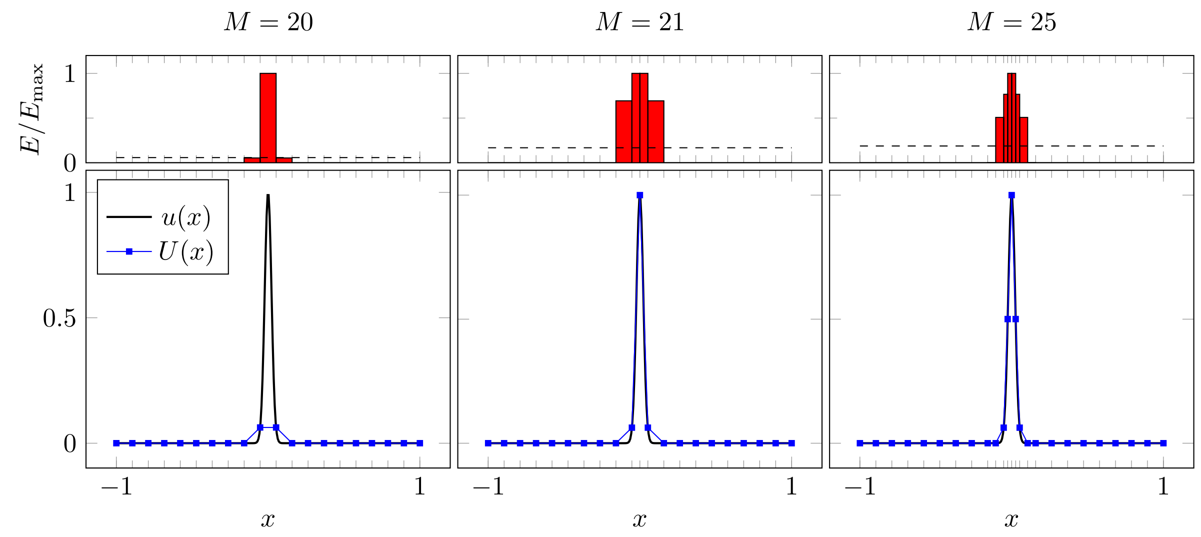
$$f(x) = -2, \quad u(0) = 0, \quad u(1) = 1$$



$$f(x) = -(40000x^2 - 200)\exp(-100x^2), \quad u(-1) = e^{-100}, \quad u(1) = e^{-100}$$



$$f(x) = -(4000000x^2 - 2000)\exp(-1000x^2), \quad u(-1) = e^{-1000}, \quad u(1) = e^{-1000}$$



$$f(x) = 2x^{-4/3}/9, \quad u(0) = 0, \quad u(1) = 1$$

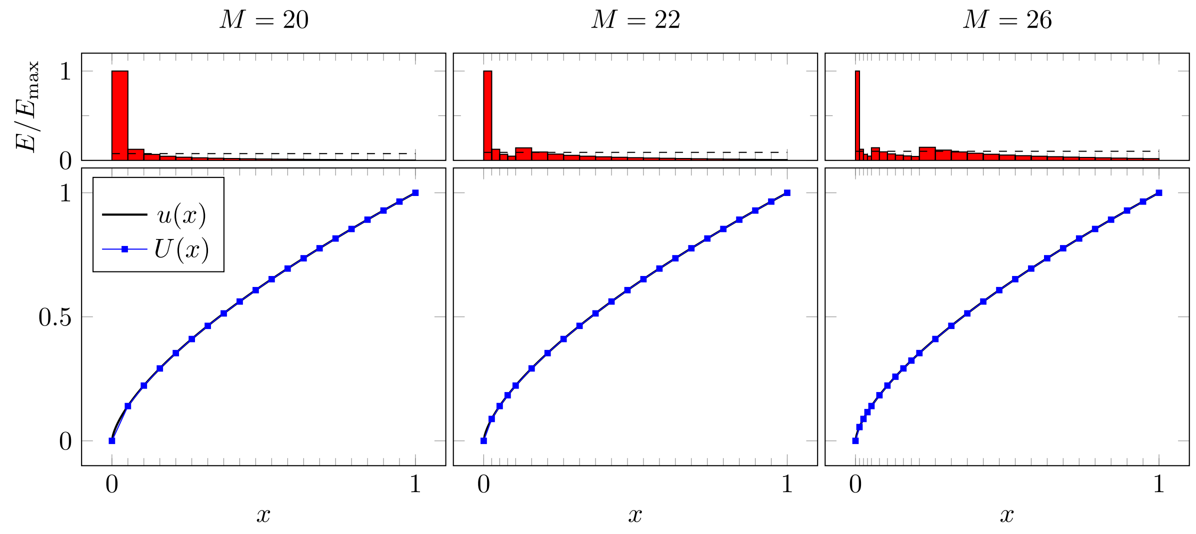


Figure 20: Adaptive mesh refinement on four problems with the average strategy, showing the evolution of an initial uniform mesh as it is refined over three iterations. Elements whose L_2-error lie above the reference error - - - are split in half.

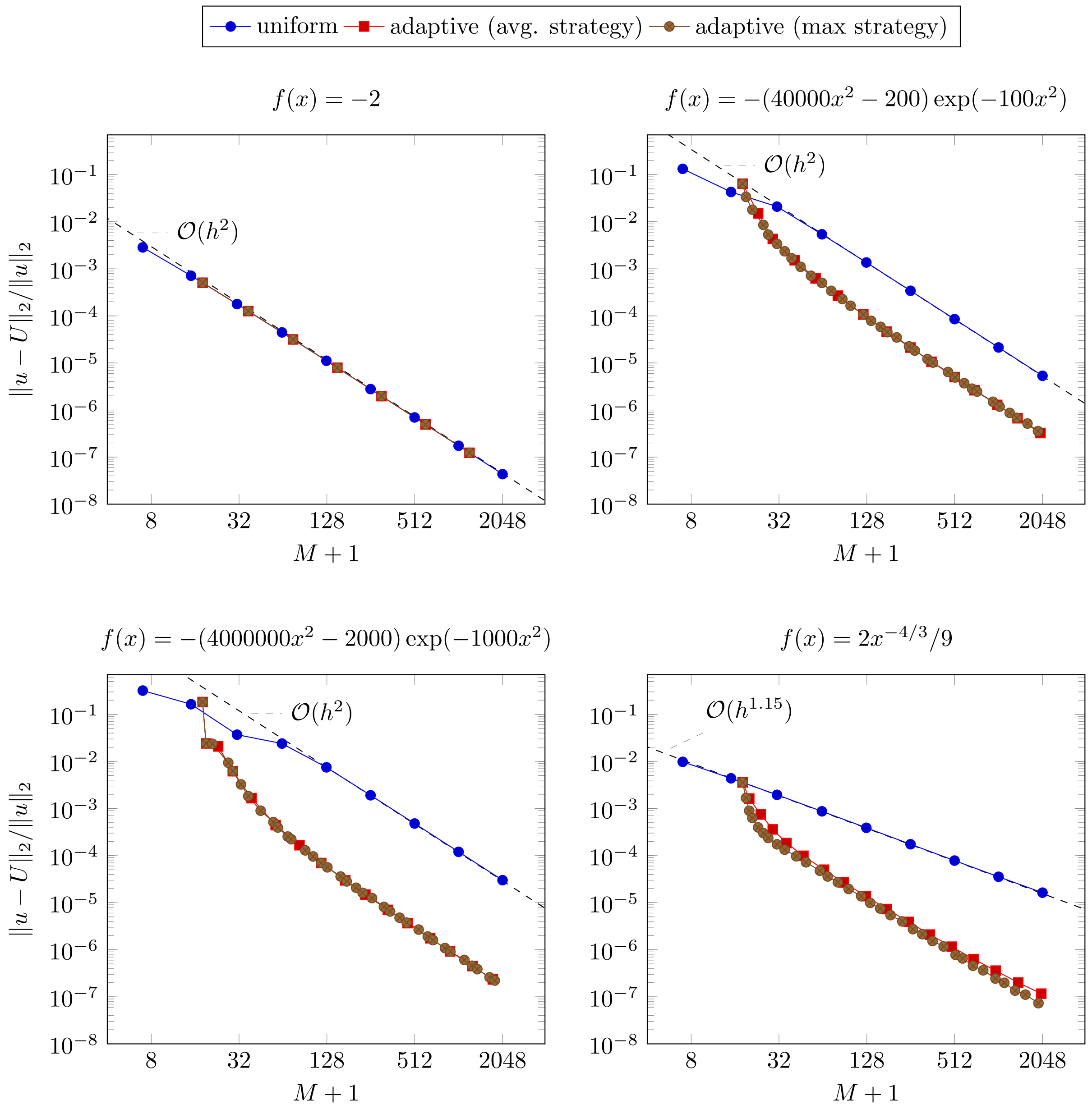


Figure 21: Convergence plots for four problems that compare uniform mesh refinement with the two adaptive mesh refinement strategies.

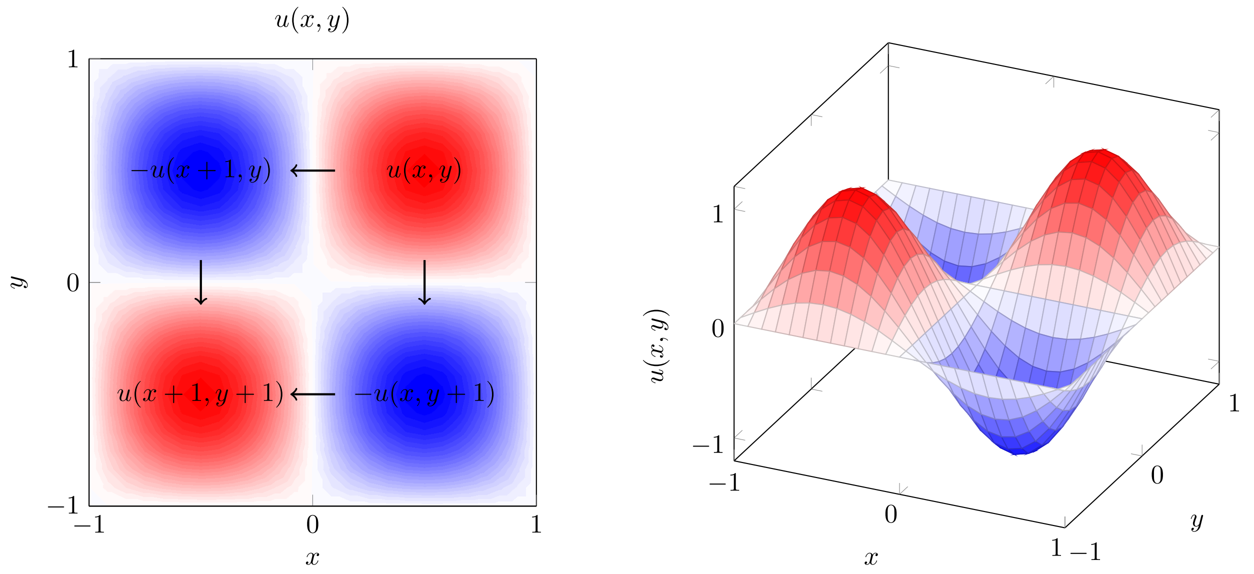


Figure 22: The function $u(x,y)$ is originally defined on $[0,1]\times[0,1]$, but we extend the definition to the full xy-plane with the rules $u(x+1,y)=u(x,y+1)=-u(x,y)$. This makes $u(x,y)$ periodic and permits Fourier analysis. Here is the continuation on $[-1,1]\times[-1,1]$.

7 Biharmonic equation

Consider the inhomogeneous Biharmonic equation with clamped boundary conditions on the unit square $\Omega=[0,1]^2$:

$$\nabla^4 u = f \quad ,(x,y)\in\Omega, \tag{33a}$$

$$u=0, \nabla^2 u = 0 \quad ,(x,y)\in\partial\Omega. \tag{33b}$$

7.1 Analytical solution

To use Fourier analysis, let us extend the definition of $u(x,y)$ on $[0,1]\times[0,1]$ to the full xy-plane $[-\infty,+\infty]\times[-\infty,+\infty]$ as the antisymmetric continuation with the rules $u(x,y+1)=u(x+1,y)=-u(x,y)$. The procedure is illustrated in figure 22. Using the antisymmetry, the conditions $u=-u=0$ and $\nabla^2 u = -\nabla^2 u = 0$ are automatically satisfied at the boundaries. This can also be seen for the simple trial function in figure 22, which vanishes and inflects at the boundaries. Now $u(x,y)=u(x+2,y)=u(x,y+2)$ is periodic in both directions with period 2 and can therefore be written as a Fourier series [2]

$$u(x,y)=\sum_{m=-\infty}^{+\infty}\sum_{n=-\infty}^{+\infty}\hat{u}_{mn}e^{i2\pi mx/2}e^{i2\pi ny/2}.$$

Multiply by $e^{i2\pi m'x/2}e^{i2\pi n'y/2}$, integrate over x and y and use orthogonality to find the coefficients

$$\hat{u}_{mn}=\frac{1}{4}\int_{-1}^{+1}\mathrm{d}x\int_{-1}^{+1}\mathrm{d}y\,u(x,y)e^{-i2\pi mx/2}e^{-i2\pi ny/2}.$$

By the antisymmetry $u(x+1,y)=u(x,y+1)=-u(x,y)$ and the symmetry $u(x+1,y+1)=u(x,y)$, the Fourier coefficients satisfy

$$u_{m,n}=-u_{-m,n}=-u_{m,-n}=u_{-m,-n},$$

so $\hat{u}_{00} = -\hat{u}_{00} = 0$ and we can write the Fourier series as

$$u(x,y) = \sum_{m=1}^{+\infty}\sum_{n=1}^{+\infty} \hat{u}_{mn} \left(e^{+i\pi mx}e^{+i\pi ny} - e^{-i\pi mx}e^{+i\pi ny} - e^{+i\pi mx}e^{-i\pi ny} + e^{-i\pi mx}e^{-i\pi ny}\right)$$
$$= -4\sum_{m=1}^{+\infty}\sum_{n=1}^{+\infty} \hat{u}_{mn} \sin(m\pi x)\sin(n\pi y)$$

after simplifying all complex expontentials using Euler's identity $e^{ix} = \cos x + i\sin x$. Rescaling $\hat{u}_{mn} \to -\hat{u}_{mn}/4$, we then begin by expressing our analytical solution as the double sine series

$$u(x,y) = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} \hat{u}_{mn} \sin(m\pi x)\sin(n\pi y). \tag{34}$$

Plug this Fourier series into equation (33) and act with the biharmonic operator to get

$$\nabla^4 u(x,y) = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} \left((m\pi)^2 + (n\pi)^2\right)^2 \hat{u}_{mn} \sin(m\pi x)\sin(n\pi y) = f(x,y).$$

For simplicity, we **restrict ourselves to sources that can also be written**

$$f(x,y) = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} \hat{f}_{mn} \sin(m\pi x)\sin(n\pi y). \tag{35}$$

The Fourier series for $\nabla^4 u(x,y)$ and $f(x,y)$ can be equal only if their coefficients are equal. This can be seen formally by multiplying both by $\sin(2m'\pi x)\sin(2n'\pi y)$, integrating over x and y and using orthogonality of the sine functions,

$$\int_0^1 \mathrm{d}x \sin(m\pi x)\sin(m'\pi x) = \frac{1}{2}\delta_{mm'}.$$

Therefore, the coefficients of the solution are

$$\hat{u}_{mn} = \frac{\hat{f}_{mn}}{\left((m\pi)^2 + (n\pi)^2\right)^2}, \tag{36}$$

and the solution $u(x,y)$ is available by summing its Fourier series 34.

If we know $\hat{f}_{mn}$, it is straightforward to compute $\hat{u}_{mn}$ and thus the solution $u(x,y)$ itself from its Fourier series. If we only know $f(x,y)$, we can find the coefficients by using the orthogonality of the sine functions again. Multiply the Fourier series by $\sin(m'\pi x)\sin(n'\pi y)$ and integrate over x and y to get

$$\int_0^1 \mathrm{d}x \int_0^1 \mathrm{d}y\, f(x,y)\sin(m'\pi x)\sin(n'\pi y)$$
$$= \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} \hat{f}_{mn} \underbrace{\int_0^1 \mathrm{d}x \sin(m\pi x)\sin(m'\pi x)}_{\delta_{mm'}/2} \underbrace{\int_0^1 \mathrm{d}y \sin(\pi y)\sin(n'\pi y)}_{\delta_{nn'}/2}$$
$$= \hat{f}_{m'n'}/4.$$

Read from bottom to top,

$$\hat{f}_{mn} = 4\int_0^1 \mathrm{d}x \int_0^1 \mathrm{d}y\, f(x,y)\sin(m\pi x)\sin(n\pi y). \tag{37}$$

Note that a general source $f(x, y)$ may only be represented exactly by an infinite Fourier series. To make the Fourier series solution viable, we must cut it off to include only a finite number of terms. In this case, we should analyze $f(x, y)$ to make sure that we exclude only Fourier modes that contribute insignificantly to the solution. However, we can also construct problems with a finite number of terms in the *exact* solution by simply defining the source $f(x, y)$ in terms of a finite number of nonzero Fourier coefficients.

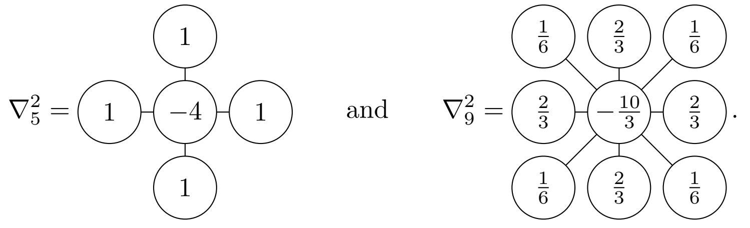
7.2 Numerical solution

We transform the Biharmonic equation 33 into a system of Poisson equations by introducing $g = \nabla^2 u$:

$$\begin{aligned} \nabla^2 g &= f, \\ \nabla^2 u &= g, \\ g = u &= 0, \qquad \text{on } \partial\Omega. \end{aligned} \tag{38}$$

Instead of solving the Biharmonic equation directly with a stencil for ∇^4, we will solve the two Poisson equations successively with a stencil ∇_n^2 for ∇^2 only. First, we solve $\nabla^2 g = f$ to obtain an intermediate numerical solution $G \approx g$. Then we solve $\nabla^2 u = g$ to obtain the final numerical solution $U \approx u$. In the first step we will use the exactly known source f, but in the second step we will only use the approximate source $G \approx g$.

For the Laplacian ∇^2, we will use both a 5-point stencil and a 9-point stencil:



They are defined by their action on some function. For example,

$$\nabla_5^2 u(x, y) = \frac{-4u(x, y) + u(x + h, y) + u(x - h, y) + u(x, y + h) + u(x, y - h)}{h^2},$$

and ∇_9^2 acts similarly but also includes terms like $u(x+h, y+h)$. We will solve Poisson equations with

$$\nabla_5^2 U = F \qquad \text{and} \qquad \nabla_9^2 U = \left(1 + \frac{1}{12}\nabla_5^2\right) F. \tag{39}$$

We will later show that the former is $\mathcal{O}(h^2)$, while the latter is $\mathcal{O}(h^4)$.

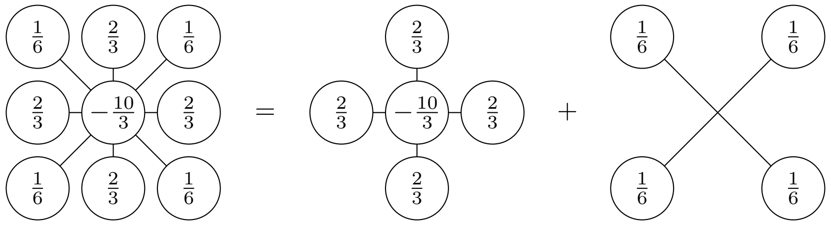
Next, we turn to formulating equation (39) as matrix equations. We define U, G and F as flattened vectors of the discretized functions for u, g and f following the same procedure as in section 4.2. For the 5-point stencil, first define the one-dimensional second order central finite difference matrix,

$$J_5 = \begin{bmatrix} -2 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ & & & & 1 & -2 \end{bmatrix}.$$

Using the Kroenecker product $\otimes$ and the Kroenecker sum $\oplus$, the 5-point stencil is represented by

$$K_5 = \begin{bmatrix} J_5 & 0 & 0 & \\ 0 & J_5 & 0 & \ddots \\ 0 & 0 & J_5 & \ddots \\ & \ddots & \ddots & \ddots \end{bmatrix} + \begin{bmatrix} -2I & I & 0 & \\ I & -2I & I & \ddots \\ 0 & I & -2I & \ddots \\ & \ddots & \ddots & \ddots \end{bmatrix} = I \otimes J_5 + J_5 \otimes I = J_5 \oplus J_5.$$

For the 9-point stencil, note that it can be split into



The first term can be handled like the 5-point stencil and is represented by $J_9 \oplus J_9$ with

$$J_9 = \frac{1}{3} \begin{bmatrix} -5 & 2 & & & & \\ 2 & -5 & 2 & & & \\ & 2 & -5 & 2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 2 & -5 & 2 \\ & & & & 2 & -5 \end{bmatrix}.$$

The second stencil picks out neighbours located diagonally from the center point and is represented by

$$\frac{1}{\sqrt{6}} \begin{bmatrix} 0 & \Sigma & & \\ \Sigma & 0 & \Sigma & \ddots \\ & \Sigma & 0 & \ddots \\ & \ddots & \ddots & \ddots \end{bmatrix} = \Sigma \otimes \Sigma \qquad \text{with} \qquad \Sigma = \frac{1}{\sqrt{6}} \begin{bmatrix} 0 & 1 & & \\ 1 & 0 & 1 & \ddots \\ & 1 & 0 & \ddots \\ & \ddots & \ddots & \ddots \end{bmatrix}.$$

Summarizing, the finite difference matrix equations corresponding to 39 that we will solve are

$$\begin{aligned} K_5 U &= F & \text{with} \quad & K_5 = J_5 \oplus J_5 \\ K_9 U &= \left(I + \frac{h^2}{12} K_5 \right) F & \text{with} \quad & K_9 = J_9 \oplus J_9 + \Sigma \otimes \Sigma. \end{aligned} \tag{40}$$

(TODO: use U or v for discrete approximtae solution?)

(TODO: say the matrices are toeplitz and symmetric) (TODO: Is this correct? I believe it is not Teoplitz, due to the zeros on the off diagonals), where the main diagonal has -4 and the ± 1 and $\pm N$ off diagonals have 1.

(TODO: should we more clearly differentiate between the discrete function and the vector describing it?)

(TODO: Verify order (ie. not I x K + K x I) believe this is a convention thing).

(TODO: check conventions on h^2)

(TODO: This may also be useful when working with sparse matricies, as efficient methods for the Kronecker product are implemented in frameworks such as Scipy[8].)

(TODO: move eigenstuff to FPS section?)

Let $\lambda_k,\ k \in [1,n]$ be the eigenvalues of the matrix A, and let $\mu_l,\ l \in [1,m]$ be the eigenvalues of the matrix B. As shown in for example Laub [3], the eigenvalues of the Kronecker product $A \otimes B$ are the products of the eigenvalues of A and B,

$$\lambda_k \mu_l, \quad k \in [1,n], l \in [1,m].$$

The eigenvalues of the Kronecker sum $A \oplus B$ are the sum of the eigenvalues of A and B

$$\lambda_k + \lambda_l, \quad k \in [1,n], l \in [1,m].$$

Moreover, if $x_1, \ldots, x_n$ are linearly independent eigenvectors of A corresponding to $\lambda_1, \ldots, \lambda_n$, and $z_1, \ldots, z_n$ are linearly independent eigenvectors of B corresponding to $\mu_1, \ldots, \mu_n$, then $x_k \otimes z_l$ are linearly independent eigenvectors of $A \otimes B$ corresponding to $\lambda_k \mu_l$. According to [4] the eigenvalues of a TST matrix where a is the main diagonal and b is the off diagonal are

$$\lambda_k = a + 2b\cos\left(\frac{k\pi}{N+1}\right)$$

and the eigenvectors are

$$y_k(m) = \sin\frac{mk\pi}{N+1}.$$

Thus, $K_{2D}, \tilde{K}_{2D}^{(9)}$, and Σ_{2D} share the same eigenvectors, (TODO: consider to argue more explicitly for this)

$$y_{kl}(m,n) = \sin\frac{mk\pi}{N+1}\sin\frac{nl\pi}{N+1}. \tag{41}$$

As a result, $K_{2D}^{(9)}$ also share these eigenvectors. The eigenvalues of K_{2D} are then

$$\left(-2 + 2\cos\left(\frac{k\pi}{N+1}\right)\right) + \left(-2 + 2\cos\left(\frac{l\pi}{N+1}\right)\right).$$

The eigenvalues of $\tilde{K}_{2D}^{(9)}$ are

$$\frac{1}{3}\left(-10 + 4\cos\left(\frac{k\pi}{N+1}\right)\right) + \frac{1}{3}\left(-10 + 4\cos\left(\frac{l\pi}{N+1}\right)\right)$$

and the eigenvalues of Σ_{2D} are

$$\frac{4}{6}\cos\left(\frac{k\pi}{N+1}\right)\cos\left(\frac{l\pi}{N+1}\right).$$

The eigenvalues of the nine point stencil K_{2D} are thus

$$-\frac{10}{3} + \frac{4}{3}\left(\cos(\frac{k\pi}{N+1}) + \cos(\frac{l\pi}{N+1})\right) + \frac{4}{6}\cos(\frac{k\pi}{N+1})\cos(\frac{l\pi}{N+1}). \tag{42}$$

TODO: Consider writing all of these eigenvalues in a more structured manner.

7.3 Stability and order of the five and nine point stencils

Definition 1. A proper n-point stencil ∇_n^2 with weights a_i has the properties

$$\nabla_n^2 f(x_0) = \sum_{i=0}^{n-1} a_i f(x_i), \qquad a_i > 0 \text{ for } i \geq 1 \qquad \text{and} \qquad \sum_{i=1}^{n-1} a_i = -a_0,$$

where x_i are the neighbouring points of x_0.

Lemma 1 (Discrete maximum principle)**.** *If $\nabla_n^2 f \geq 0$ on Ω and ∇_n^2 is a proper stencil, then f attains its maximum value on $\partial\Omega$, that is*

$$\max_{\Omega} f \leq \max_{\partial\Omega} f.$$

Proof. Suppose instead that $\max_\Omega f > \max_{\partial\Omega} f$. Then there is an internal grid point x_0 on which f attains its maximum value $f(x_0) \geq f(x_i)$, where x_i are the neighbouring points x_i. Then

$$-a_0 f(x_0) = \sum_{i=1}^{n-1} a_i f(x_i) - \nabla_n^2 f(x_0) \leq \sum_{i=1}^{n-1} a_i f(x_i) \leq \sum_{i=1}^{n-1} a_i f(x_0) = -a_0 f(x_0). \tag{43}$$

The right side is equal to the left side, so equality must hold throughout and $\sum_{i=1}^{n-1} a_i(f(x_0) - f(x_i)) = 0$. The stencil is proper, so $a_i > 0$ for $i \geq 1$ and all the are nonnegative, so it can only vanish if $f(x_0) = f(x_1) = \cdots = f(x_{n-1})$. Repeating the argument at each neighbour x_i, and then to their neighbours and so on, we ultimately reach the conclusion that the same value is also attained on $\partial\Omega$, so we have a contradiction. □

Lemma 2. *If there exists a function $\phi_n(x,y) \geq 0$ such that $\nabla_n^2 \phi_n = 1$ on Ω for a proper stencil ∇_n^2, and v is a discrete function that equals zero on $\partial\Omega$, then*

$$\|v\|_\infty \leq \max_{\partial\Omega} |\phi_n| \|\nabla_n^2 v\|_\infty. \tag{44}$$

Proof. Let $\|\nabla_n^2 v\|_\infty = M$. Then

$$\nabla_n^2(v + \phi_n M) = \nabla_n^2 v + M \geq 0.$$

Since ∇_n^2 is a proper stencil, $v + \phi_n M$ attains its maximum value on $\partial\Omega$ by lemma 1. Thus

$$\|v\|_\infty \leq \|v + \phi_n M\|_\infty \leq \max_{\partial\Omega} |v + \phi_n M| = \max_{\partial\Omega} |\phi_n| M = \max_{\partial\Omega} |\phi_n| \|\nabla_n^2 v\|_\infty. \quad \square$$

Remark. For the five and nine point stencil, such functions do exist. Suspecting that the stencils are second and fourth order, we look for second and fourth order polynomials in x and y with this property. We find

$$\phi_5(x,y) = \frac{1}{4}\left(\left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2\right) \tag{45}$$

with the property $\nabla_5^2 \phi_5(x,y) = 1$, taking the values $0 \leq \phi_5(x,y) \leq 1/8$ on Ω and attaining the maximum on $\partial\Omega$. Similarly, the polynomial

$$\phi_9(x,y) = \frac{1}{5}\left(\left(x - \frac{1}{2}\right)^4 + \left(y - \frac{1}{2}\right)^4\right) - \frac{6}{5}\left(x - \frac{1}{2}\right)^2\left(y - \frac{1}{2}\right)^2 + \frac{1}{4}\left(\left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2\right) \tag{46}$$

is such that $\nabla_9^2 \phi_9(x,y) = 1$, takes the values $0 \leq \phi_9(x,y) \leq 3/40$ on Ω and attains the maximum on $\partial\Omega$. Both are shown in figure 23.

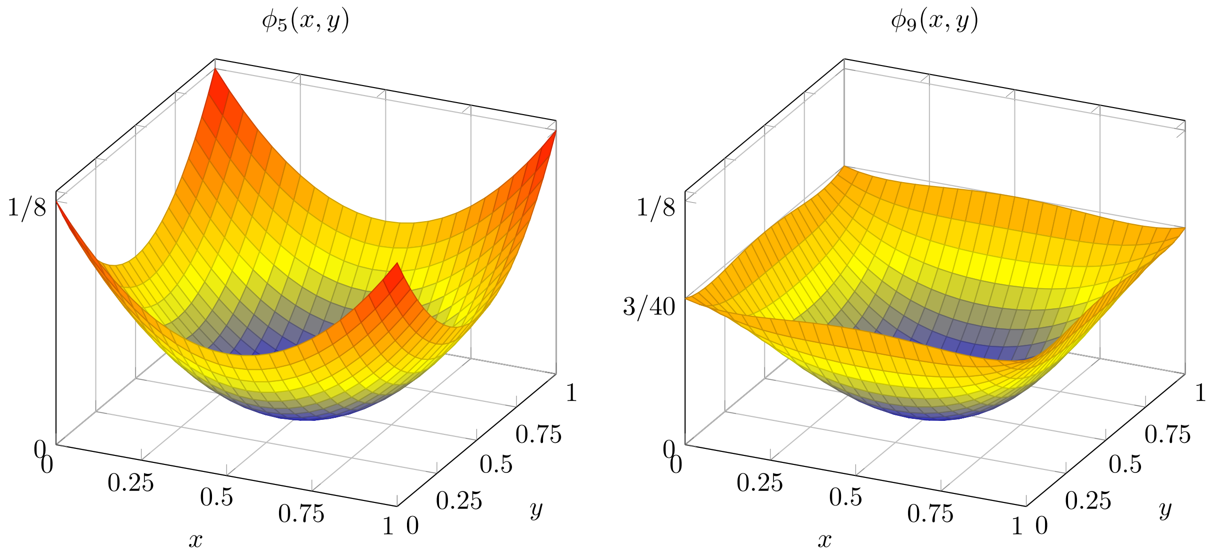


Figure 23: The functions $\phi_5(x,y)$ and $\phi_9(x,y)$ range from $\phi_5(\frac{1}{2},\frac{1}{2}) = \phi_9(\frac{1}{2},\frac{1}{2}) = 0$ at the center to $\phi_5(0,0) = 1/8$ and $\phi_9(0,0) = 3/40$ at the boundaries of $[0,1]\times[0,1]$.

Definition 2. We denote the maximum of all n-index derivatives of u on Ω by

$$|D^n u|_\infty = \max_{m=0}^{n} \|\partial_x^m \partial_y^{n-m} u\|_\infty$$

Theorem 1 (Five point stencil stability). *If $\nabla^2 u = f$ and $\nabla_5^2 v = f$ and $u = v$ on $\partial\Omega$, then*

$$\|u - v\|_\infty \leq \frac{h^2}{48} |D^4 v|_\infty.$$

Proof. By lemma 2 with $\phi_5(x,y)$ from 45,

$$\|u - v\|_\infty \leq \frac{1}{8} \|\nabla_5^2 (u - v)\|_\infty.$$

Taylor expand all terms in $\nabla_5^2 u$ around (x,y) to first order to get

$$\begin{aligned}\nabla_5^2 u(x,y) &= \nabla^2 u(x,y) + \frac{h^2}{12}(\partial_x^4 + \partial_y^4) u(x+\theta h, y+\eta h)\\ &\leq f(x,y) + \frac{h^2}{6} \max_{\Omega} \left\{ \left|\partial_x^4 u\right|, \left|\partial_y^4 u\right| \right\}\\ &\leq f(x,y) + \frac{h^2}{6} \left|D^4 u\right|_\infty.\end{aligned}$$

for some $0 \leq \theta, \eta \leq 1$. Finally, substitute $f = \nabla_5^2 v$ to conclude that

$$\|u - v\|_\infty \leq \frac{1}{8} \|\nabla_5^2 (u - v)\|_\infty \leq \frac{h^2}{48} \left|D^4 v\right|_\infty.$$ □

Theorem 2 (Nine point stencil stability). *If $\nabla^2 u = f$ and $\nabla_9^2 v = f + \frac{h^2}{12}\nabla_5^2 f$ and $u = v$ on $\partial\Omega$, then*

$$\|u - v\|_\infty \leq \frac{11 h^4}{2400} |D^6 v|_\infty.$$

Proof. By lemma 2 with ϕ_9 from 46,

$$\|u - v\|_\infty \le \frac{3}{40} \|\nabla_9^2 (u - v)\|_\infty .$$

Taylor expand all terms in $\nabla_9^2 u$ around (x, y) as in the former proof to get

$$\begin{aligned}\nabla_9^2 u(x,y) &= \nabla^2 u(x,y) + \frac{h^2}{12}\nabla^4 u(x,y) + \frac{h^4}{360}\left(\partial_x^6 + \partial_y^6 + 5\partial_x^2\partial_y^4 + 5\partial_x^4\partial_y^2\right) u(x+\theta h, y+\eta h)\\ &\le f(x,y) + \frac{h^2}{12}\nabla^2 f(x,y) + \frac{h^4}{30}\left|D^6 u\right|_\infty .\end{aligned}$$

for some $0 \le \theta, \eta \le 1$. From the Taylor expansion in the former proof with $u \to f$, we also know that

$$\begin{aligned}\nabla^2 f(x,y) &= \nabla_5^2 f(x,y) + \frac{h^2}{12}(\partial_x^4 + \partial_y^4) f(x+\chi h, y+\psi h)\\ &= \nabla_5^2 f(x,y) + \frac{h^2}{12}(\partial_x^6 + \partial_y^6 + \partial_x^4\partial_y^2 + \partial_x^2\partial_y^4) u(x+\chi h, y+\psi h)\\ &\le \nabla_5^2 f(x,y) + \frac{h^2}{3}\left|D^6 u\right|_\infty\end{aligned}$$

for some $0 \le \chi, \psi \le 1$. Altogether

$$\nabla_9^2 u(x,y) \le f(x,y) + \frac{h^2}{12}\ \nabla_5^2 f(x,y) + \frac{11h^4}{180}\left|D^6 u\right|_\infty .$$

Finally, substitute $f + \frac{h^2}{12}\ \nabla_5^2 f = \nabla_9^2 v$ to conclude that

$$\|u - v\|_\infty \le \frac{3}{40}\|\nabla_9^2(u - v)\|_\infty \le \frac{11h^4}{2400}\left|D^6 u\right|_\infty . \qquad \square$$

(TODO: a word on convergence and stability)

7.4 The Fast Poisson Solver

We are to solve the equation

$$AU = F.$$

Assuming that the eigenvectors of A form a complete set, we may write

$$F = a_1 y_1 + a_2 y_2 + ...,$$

where $y_1, y_2, \ldots$ are the eigenvectors of A. A similar expansion must exist for U, as the eigenvectors are assumed to be complete. One may verify simply by insertion that the solution U is then of course

$$U = \frac{a_1}{\lambda_1} y_1 + \frac{a_2}{\lambda_2} y_2 + \ldots,$$

where λ_i is the eigenvalue corresponding to y_i. In general, however, this is not a viable way to solve the problem, as it requires knowing all the eigenvalues and eigenvectors, as well as finding the coefficients a_i. However, for the set of eigenvectors in this problem, which are sines, we have a very efficient algorithm for computing the coefficients, the discrete fast sine transform. We also have simple analytical expressions for the eigenvalues. The idea behind the Fast Poisson Solver is to given F find its expansion coefficients a_i, divide these by the corresponding eigenvalues λ_i, and then use this to express U.

According to (41) the (m,n) compontent of the eigenvector corresponding to $\lambda_{k,l}$ is

$$\sin\left(\frac{mk\pi}{N+1}\right)\sin\left(\frac{nl\pi}{N+1}\right).$$

The discrete sine transform of type I $x(k)$ of $y(n)$ is defined as[1] [**TODO**]

$$y(n)=\sum_{k=1}^{N}x(k)\sin\left(\frac{\pi kn}{N+1}\right),\ n=1,\dots N. \tag{47}$$

The two dimensional equivalent is simply to apply this transformation to the two directions sequentially.

$$y(m,n)=\sum_{k=1}^{N}\sum_{l=1}^{N}x(k,l)\sin\left(\frac{\pi km}{N+1}\right)\sin\left(\frac{\pi ln}{N+1}\right),\ m=1,\dots N, n=1,\dots N. \tag{48}$$

This prefectly corresonds to the eigenvectors of the stencils! Therefore given $F(m,n)$ the 2D Discrete Sine Transform gives $x(k,l)$, corresponding to the coefficents a_i introduced above (where now the subscript has two indices instead of one). As we know the eigenvalues $\lambda_{k,l}$ of the five and nine point stencils from section 7.2, the solution U is readily computed by the inverse Discrete Sine Transform, which calculates $y(m,n)$ given $x(k,l)$.

Formulating the Fast Poisson Solver more directly with regards to implementing the solver, we have that the solution is

$$U=\mathrm{IFST}(\mathrm{FST}(F)/\Lambda),$$

where IFST, FST are the inverse and normal Fast Sine Tranform in two dimensions, and Λ are the eigenvalues of the stencil for which we solve. These eigenvalues were given in section 7.2.

7.5 Demonstration of order

To demonstrate the order of the five and nine point stencils, we will perform UMR on the Poisson equation, with the inhomogenity $f=\sin(m\pi x)\sin(n\pi y),\ m=3,n=4$. From section 7.1 we know that the solution to this manufactured problem is

$$u(x,y)=\frac{\sin(m\pi x)\sin(n\pi y)}{(n\pi)^2+(m\pi)^2},\ m=3,n=4.$$

The mesh refinement is done with the same number of discretization points in x- and y-direction. The values used are $N=N_x=N_y=\{8,16,32,64,128,256\}$. The solution is shown in figure 24. As expected, the error for the five point stencil goes as h^2 while the error for the nine point stencil goes as h^4. The absolute error as a function of N is shown in figure 25.

7.6 Solving the Biharmonic equation

We will now solve (33) numerically on a manufactured problem. Let

$$u(x,y)=(\sin\pi x\sin\pi y)^4\,e^{-(x-0.5)^2-(y-0.5)^2}.$$

The inhomogenity f is simply found by calculating $\nabla^4 u$. The result, which was found using a computer algebra system, is somewhat lengthy, and therefore only included as an appendix for the sake of readability.

[1]Most literature uses a convention where $n\in[0,N-1]$ $k\in[0,N-1]$. We have altered it here to comply with the rest of the report.

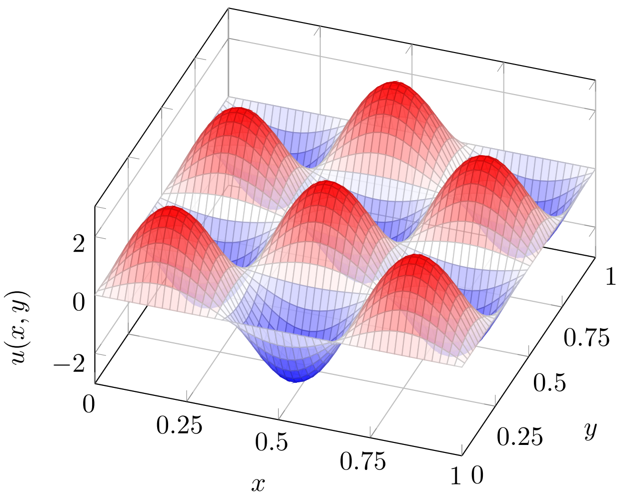


Figure 24: The solution $u(x, y) = \sin(m\pi x)\sin(n\pi y)/((n\pi)^2 + (m\pi)^2),\ m = 3, n = 4$ to the manufactured Poisson equation with inhomogenity $f = \sin(m\pi x)\sin(n\pi y)$.

According to equation (38) we split the equation into a system of poisson equations by introducing a new function g:

$$\begin{aligned}\nabla^2 g &= f,\\ \nabla^2 u &= g,\\ g = u &= 0, \qquad \text{on } \partial\Omega.\end{aligned}$$

It is now simply a matter of applying the described Fast Poisson Solver sequentially for the two equations.

The numerical solution is shown in figure **??**. In figure 26 a convergence plot as a function of the degrees of freedom is shown, and the computation time is shown in figure 27.

Demonstration of order of the five and nine point stencil.

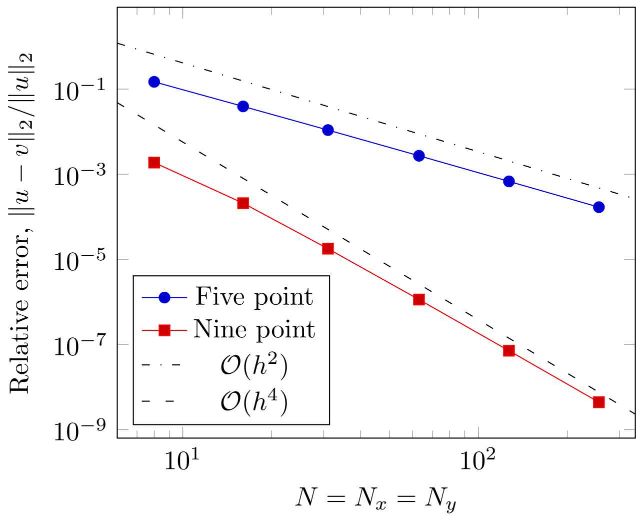


Figure 25: Relative error for the clamped Biharmonic equation on $[0,1]^2$ using the five and nine point stencil. Also shown are h^2 and h^4, which are the expected convergence rates of the five and nine point stencil respectively. First axis shows N, the number of internal discretization points in one direction, so that the total number of grid points is N^2. Solution was found for the manufactured problem $f = \sin(m\pi x)\sin(n\pi y)$, $m = 3, n = 4$.

A The inhomogenity f

In section 7.6 the Fast Poisson Solver was applied on the manufactured solution u. The inhomogenity $f = \nabla^4 u$ was ommited in the text, due to it being lengthy. It is here shown in full, together with the code used to find it, using the Sage CAS.

```
sage: u(x, y) = (sin(pi * x) * sin(pi * y))^4 * e^(-(x-1/2)^2 - (y-1/2)^2)
sage: (u.diff(x, 4) + u.diff(y, 4) + 2 * u.diff(x, 2).diff(y,
    2)).full_simplify()
>> 4*(6*pi^4*e^(x + y)*sin(pi*x)^4 - 8*(4*pi*y^3*e^x*sin(pi*x)^4 -
    6*pi*y^2*e^x*sin(pi*x)^4 - 6*pi^3*e^x*sin(pi*x)^2 + 4*(2*pi^2*x -
    pi^2)*cos(pi*x)*e^x*sin(pi*x)^3 + (3*pi + 16*pi^3 - 2*pi*x^2 +
    2*pi*x)*e^x*sin(pi*x)^4 + 4*(3*pi^3*e^x*sin(pi*x)^2 - 2*(2*pi^2*x -
    pi^2)*cos(pi*x)*e^x*sin(pi*x)^3 - (pi + 8*pi^3 - pi*x^2 +
    pi*x)*e^x*sin(pi*x)^4)*y)*cos(pi*y)*e^y*sin(pi*y)^3 +
    (4*y^4*e^x*sin(pi*x)^4 - 8*y^3*e^x*sin(pi*x)^4 - 8*(3*pi + 4*pi*x^3 +
    16*pi^3 - 6*pi*x^2 - 4*(pi + 8*pi^3)*x)*cos(pi*x)*e^x*sin(pi*x)^3 +
    (256*pi^4 + 4*x^4 - 8*(16*pi^2 + 1)*x^2 - 8*x^3 + 64*pi^2 + 4*(32*pi^2 +
    3)*x + 1)*e^x*sin(pi*x)^4 + 6*pi^4*e^x - 24*(2*pi^3*x -
    pi^3)*cos(pi*x)*e^x*sin(pi*x) - 12*(13*pi^4 - 6*pi^2*x^2 + 6*pi^2*x +
    2*pi^2)*e^x*sin(pi*x)^2 + 8*(2*(pi - 2*pi*x)*cos(pi*x)*e^x*sin(pi*x)^3 -
    (16*pi^2 - x^2 + x + 1)*e^x*sin(pi*x)^4 + 3*pi^2*e^x*sin(pi*x)^2)*y^2 -
    4*(4*(pi - 2*pi*x)*cos(pi*x)*e^x*sin(pi*x)^3 - (32*pi^2 - 2*x^2 + 2*x +
    3)*e^x*sin(pi*x)^4 + 6*pi^2*e^x*sin(pi*x)^2)*y)*e^y*sin(pi*y)^4 -
    24*(2*pi^3*y*e^x*sin(pi*x)^4 -
    pi^3*e^x*sin(pi*x)^4)*cos(pi*y)*e^y*sin(pi*y) +
    12*(6*pi^2*y^2*e^x*sin(pi*x)^4 - 6*pi^2*y*e^x*sin(pi*x)^4 +
```

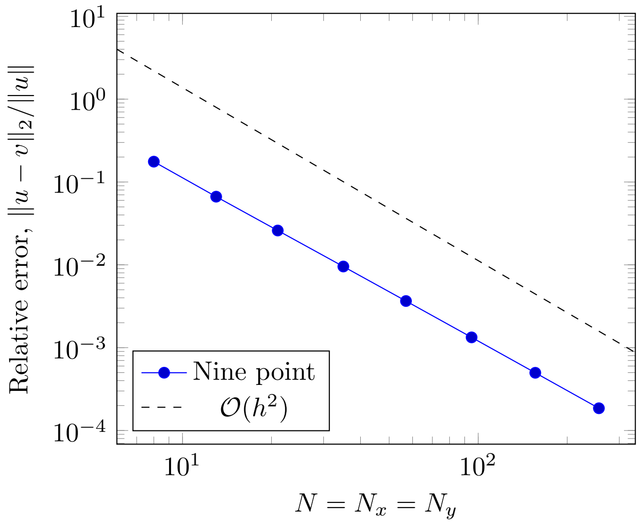


Figure 26: The relative error of the numerical solution to the clamped Biharmonic equation, using the manufactured solution $(\sin \pi x \sin \pi y)^4 \, e^{-(x-0.5)^2-(y-0.5)^2}$. TODO: Could add five point for comparison?

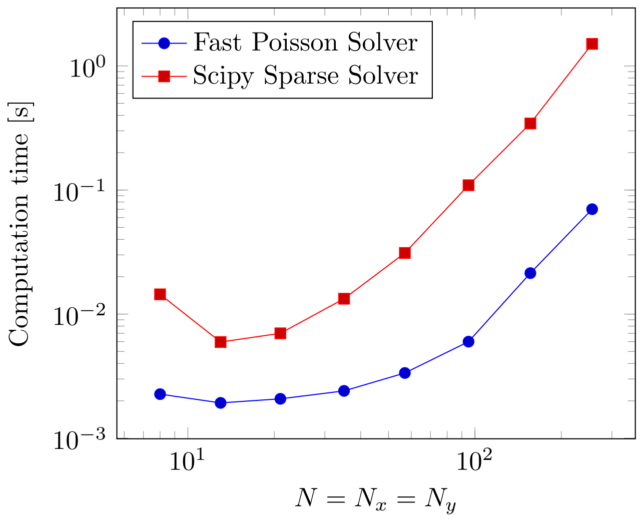


Figure 27: TODO: write

```
6*pi^4*e^x*sin(pi*x)^2 - 4*(2*pi^3*x - pi^3)*cos(pi*x)*e^x*sin(pi*x)^3 -
(13*pi^4 - 2*pi^2*x^2 + 2*pi^2*x +
2*pi^2)*e^x*sin(pi*x)^4)*e^y*sin(pi*y)^2)*e^(-x^2 - y^2 - 1/2)
```

References

[1] Charles Curry. *TMA4212 Part 2: Introduction to finite element methods*. Apr. 30, 2018. URL: `http://www.math.ntnu.no/emner/TMA4212/2020v/notes/master2.pdf` (visited on 04/25/2021).

[2] Erwin Kreyszig. *Advanced Engineering Mathematics (10th edition)*. Wiley, 2011. ISBN: 978-0-470-45836-5.

[3] Alan J. Laub. *Matrix Analysis for Scientists and Engineers*. Philadelphia: SIAM: Society for Industrial and Applied Mathematics, Dec. 29, 2004. 184 pp. ISBN: 978-0-89871-576-7.

[4] Silvia Noschese, Lionello Pasquini, and Lothar Reichel. "Tridiagonal Toeplitz matrices: properties and novel applications". In: *Numerical Linear Algebra with Applications* 20.2 (2013), pp. 302–326. ISSN: 1099-1506. DOI: `https://doi.org/10.1002/nla.1811`.

[5] *numpy.linalg.lstsq — NumPy v1.20 Manual*. URL: `https://numpy.org/doc/stable/reference/generated/numpy.linalg.lstsq.html` (visited on 04/25/2021).

[6] Brynjulf Owren. *TMA4212 Numerical solution of partial differential equations with finite difference methods*. Jan. 31, 2017. URL: `http://www.math.ntnu.no/emner/TMA4212/2020v/notes/master.pdf` (visited on 04/22/2021).

[7] *scipy.integrate.fixed_quad — SciPy v1.6.2 Reference Guide*. URL: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.fixed_quad.html` (visited on 04/25/2021).

[8] *scipy.sparse.kron — SciPy v1.6.2 Reference Guide*. URL: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.kron.html` (visited on 04/22/2021).

[9] *scipy.sparse.linalg.SuperLU — SciPy v1.6.2 Reference Guide*. URL: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.linalg.SuperLU.html` (visited on 04/25/2021).

[10] *Sparse matrices (scipy.sparse) — SciPy v1.6.2 Reference Guide*. URL: `https://docs.scipy.org/doc/scipy/reference/sparse.html` (visited on 04/25/2021).