Semester Project in TMA4212

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

April 30, 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

In order to achieve a numerical solution to the problem, we 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

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

Discarding terms of $\mathcal{O}(h^2)$ and denoting the approximation of the solution u_m as U_m, we get the finite difference formula

$$\frac{U_{m-1} - 2U_m + U_{m+1}}{h^2} = f_m \qquad (1 \le m \le M - 1). \tag{3}$$

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 $u_x(1) = b$

$$\begin{bmatrix} 1 & & & & & \\ +1/h^2 & -2/h^2 & +1/h^2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & +1/h^2 & -2/h^2 & +1/h^2 \\ & & & +1/2h & -2/h & +3/2h \end{bmatrix} \begin{bmatrix} U_0 \\ U_1 \\ \vdots \\ U_M \\ U_{M+1} \end{bmatrix} = \begin{bmatrix} a \\ f(x_1) \\ \vdots \\ f(x_M) \\ b \end{bmatrix}, \tag{4}$$

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. To the right of each solution plot is the corresponding convergence plot, resulting from refinement of the uniform grid. The convergence plots show the computed relative errors at varying grid resolution, together with the expected convergence rate.

To find the expected convergence rate, we start by insert the exact solution into the finite difference formula (3). This gives rise to an additional term, the *local truncation error* τ_m, since the equality in the formula does not hold for the exact solution. We get that

$$\tau_m = \frac{u_{m-1} - 2u_m + u_{m+1}}{h^2} - f_m. \tag{5}$$

We now Taylor expand the terms around u_m, and the only difference between the expansion of $u_{m-1} = u(x_m - h)$ and the expansion of $u_{m+1} = u(x_m + h)$, is the sign in front of the odd-order terms, which

therefore cancel. The expansion of the truncation error is thus

$$\tau_m = \frac{1}{h^2}\left(-2u_m + 2u_m + 2\frac{h^2}{2}\partial_x^2 u_m + 2\frac{h^4}{24}\partial_x^4 u_m + \mathcal{O}(h^6)\right) - f_m$$

$$= \partial_x^2 u_m + \frac{h^2}{12}\partial_x^4 u_m + \mathcal{O}(h^4) - f_m$$

$$= \frac{h^2}{12}\partial_x^4 u_m + \mathcal{O}(h^4)$$

$$= \mathcal{O}(h^2).$$

For relating the local error τ_m to the global error and the expected convergence rate, we will for brievety just provide a slightly rough discussion. By rearanging (5) and subtracting it from equation (3), we can express the discretization error $e_m = U_m - u_m$ implicitly by

$$\frac{1}{h^2}\delta^2(u_m - U_m) = \frac{1}{h^2}\delta^2 e_m = \tau_m.$$

Defining the vectors $E = [e_1, \ldots, e_m]$ and $\tau = [\tau_1, \ldots, \tau_m]$, we can write this on matrix form so that

$$A'E = \tau.$$

The matrix A' can be shown to be invertible, and for the matrix norms subordinate to both the L_2 continous function norm, and the l_2 discrete vector norm, on can also show that $\|(A')^{-1}\|$ is bounded. This means that

$$\|E\| = \|(A')^{-1}\tau_m\| \leq \|(A')^{-1}\|\|\tau_m\| = C\|\tau\|.$$

Thus we can expect that the global error converges as the local truncation error.

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 $[-1/h, +1/h, 0, \ldots]$ 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

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

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

The job is then once again to solve the system $AU = b$. Note that the stencil reduces to the one in equation (4) 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.

8

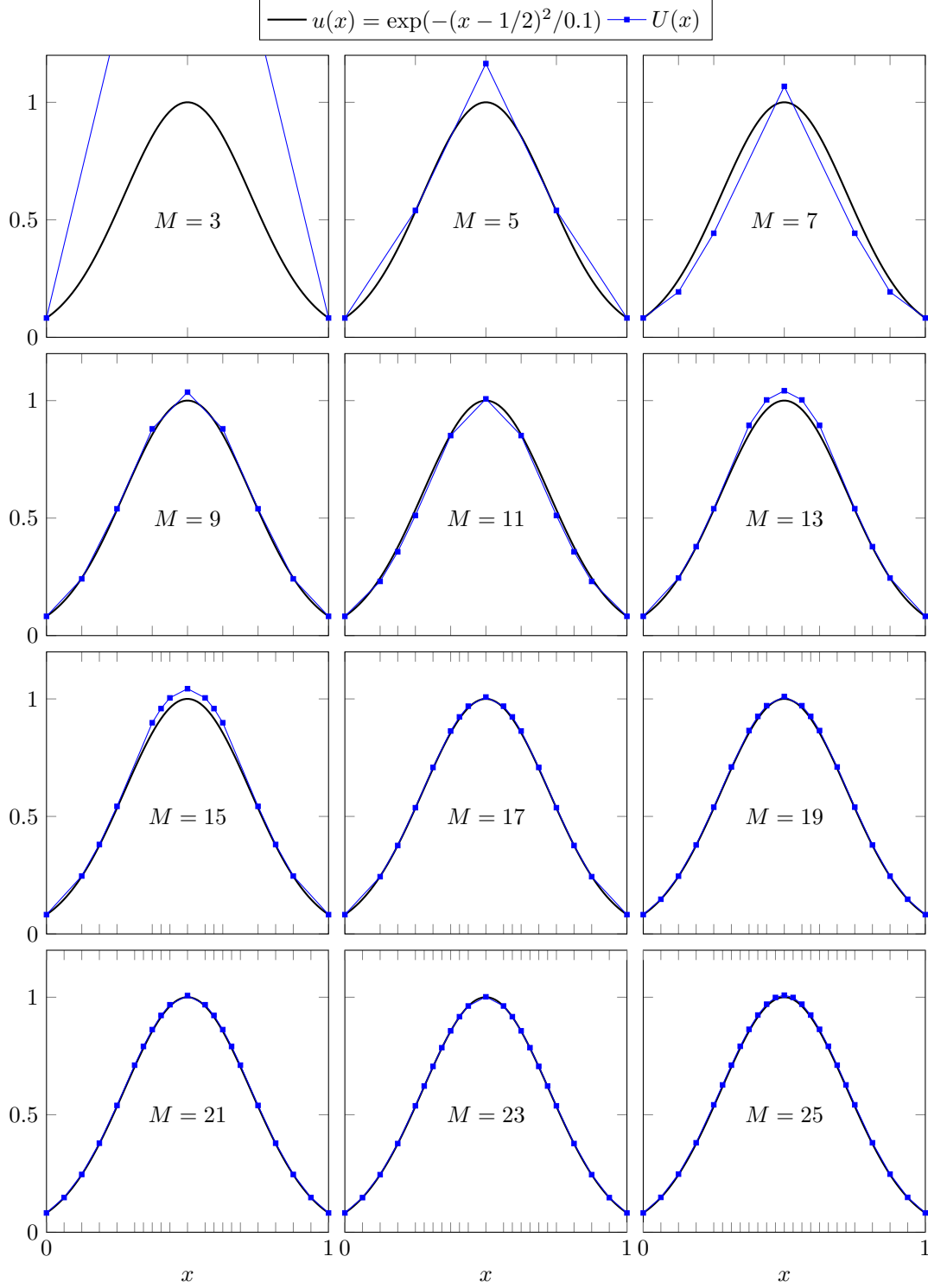**Figure 2:** During adaptive mesh refinement (AMR) with the error strategy (strategy number 1), the interval with the largest error $\int dx |u(x) - U(x)|$ is split in half. Here, $u(x) = \exp(-(x - 1/2)^2/0.1)$ is the symmetric solution to whichever Poisson equation has $f(x) = u_{xx}$ on $x \in [0,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

9

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 $\int dx |f(x)|$. Errors $\|u - U\|_2$ are measured with the continuous and discrete L_2-norm. Note that a cross inside a circle means that the discrete and continous error measurements agree.

2 Heat equation in one dimension

In this section, we consider the heat equation for $u = u(x, t)$ in one spatial dimension,

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

with either Neumann or Dirichlet boundary conditions. We solve it numerically using both the Backward Euler method and the Crank-Nicolson method, which as we will later see, are $\mathcal{O}(k + h^2)$ and $\mathcal{O}(k^2 + h^2)$ methods respectively. Then 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

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

where k is the time step, 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 step size $k = 1/(N - 1)$ in time, where N denotes the number of time steps, and the final uniform grid is illustrated in figure 4. This gives the approximate solution of $v_m(t)$ at $t_n = nk$, where $n = 0, \ldots N-1$, and we denote the fully discretized approximation of $u(x_m, t_n)$ as U_m^n. For the heat equation this results in the following finite difference formula

$$\frac{U_m^{n+1} - U_m^n}{k} = (1 - \theta) \frac{1}{h^2} \delta_x^2 U_m^n + \theta \frac{1}{h^2} \delta_x^2 U_m^{n+1}. \tag{6}$$

After organizing the terms, the θ-method for the 1D heat equation is then written compactely as

$$(1 - \theta r \delta_x^2) U_m^{n+1} = \left(1 + (1 - \theta) r \delta_x^2 \right) U_m^n, \tag{7}$$
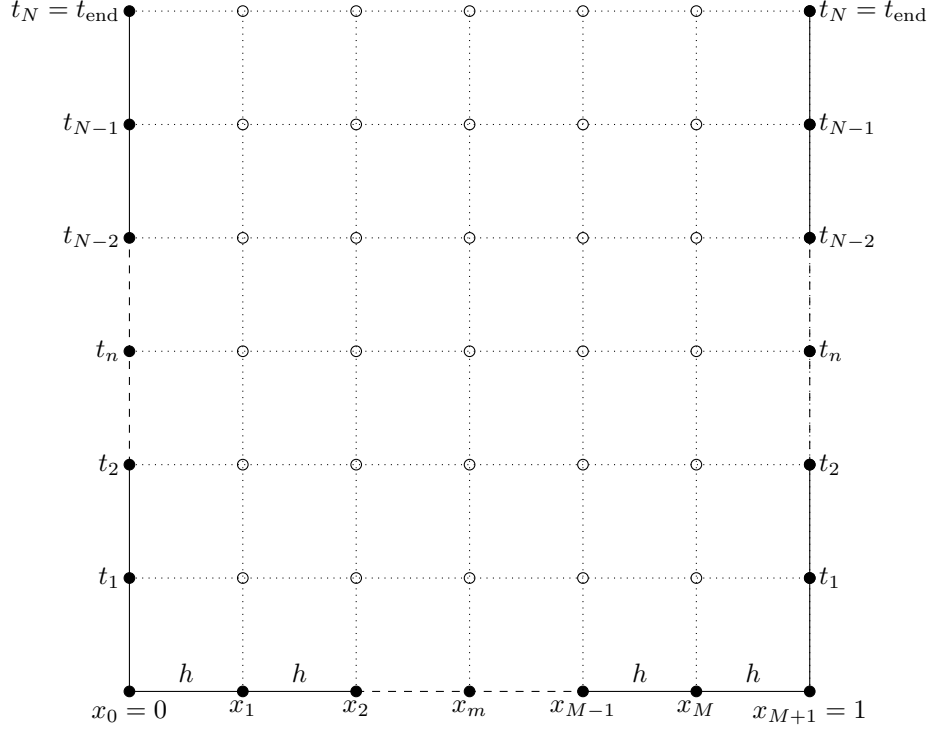
where we have defined $r = k/h^2$.

Figure 4: An illustration of how we divide the temporally and spatially continuous domain into a discrete, uniform grid. The filled circles correspond to values known through the initial and Dirichlet boundary conditions, and the empty circles to unknown values, to be determined using the finite difference methods. In the case of Neumann boundary conditions we will need to compute values at the boundaries as well.

To impose Dirichlet 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 (7) for $m = 1$ and $m = M$ to obtain

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

We combine this with equation 7 for the remaining spatial nodes to write the system of equations in matrix form

$$(I - \theta r A)\,U^{n+1} = \left(I + (1-\theta)rA\right)U^n + \rho, \tag{8}$$

with

$$A = \begin{bmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{bmatrix} \quad \text{and} \quad \rho = \begin{bmatrix} r\sigma \\ 0 \\ \vdots \\ 0 \\ r\beta \end{bmatrix}. \tag{9}$$

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

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

12

We then use these expressions to eliminate the fictitious nodes from equation (7) for $m = 0$ and $m = M + 1$ to get

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

Now we can write the system of equations on the same matrix form (8), but with

$$A = \begin{bmatrix} -2 & 2 & & & & \\ 1 & -2 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & 1 & -2 & 1 \\ & & & 2 & -2 \end{bmatrix} \quad \text{and} \quad \rho = \begin{bmatrix} -2rh\sigma \\ 0 \\ \vdots \\ 0 \\ 2rh\beta \end{bmatrix}. \tag{10}$$

Note that with Dirichlet conditions at both boundaries we only solve the equations for the internal spatial nodes $x_1 \ldots x_M$ so that A in (9) is an $M \times M$ matrix. With Neumann conditions however we also need to solve for the boundary nodes, and A is in (10) an $(M + 2) \times (M + 2)$ matrix. In both cases though, all quantities on the right hand sides in (8) 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 the 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,

$$u_x(0, t) = u_x(1, t) = 0, \quad u(x, 0) = 2\pi x - \sin(2\pi x). \tag{11}$$

The computed solutions for $t \in [0, 0.3]$ is plotted in figure 5a, 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.
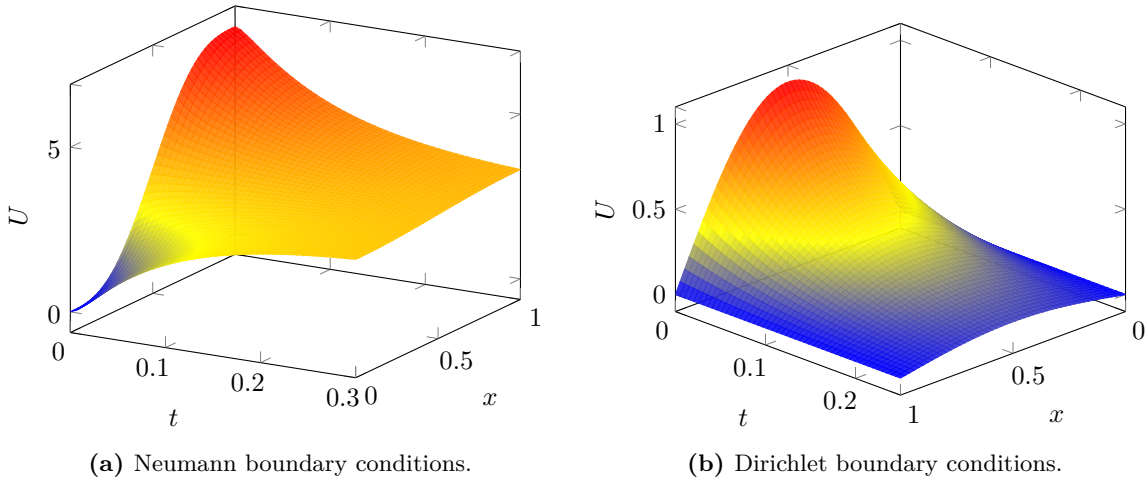

(a) Neumann boundary conditions.

(b) Dirichlet boundary conditions.

Figure 5: Numerical solution of the one dimensional heat equation with different sets of boundary and initial conditions computed with Crank-Nicolson. Subfigure 5a shows the solution with the conditions specified in (11), and subfigure 5b is from solving with the conditions given in (12).

2.2 Convergence and mesh refinement

We will now demonstrate how how the methods used in this section converge, depending on the mesh refinement.

We denote the exact solution evaluated at the discrete grid points as $u_m^n = u(x_m, t_n)$. Inserting this into the finite difference formula (6) gives rise to an additional term τ_m^n, since the difference formula is not satisfied by the exact solution. The term τ_m^n is the *local truncation error*, and for our difference formula it is expressed as follows

$$k\tau_m^n = u_m^{n+1} - u_m^n - (1-\theta)\frac{k}{h^2}\delta_x^2 u_m^n - \theta\frac{k}{h^2}\delta_x^2 u_m^{n+1}$$

$$= (1 - \theta\frac{k}{h^2}\delta_x^2)u_m^{n+1} - u_m^n - \frac{k}{h^2}\delta_x^2 u_m^n + \theta\frac{k}{h^2}\delta_x^2 u_m^n$$

$$= (1 - \theta\frac{k}{h^2}\delta_x^2)(u_m^{n+1} - u_m^n) - \frac{k}{h^2}\delta_x^2 u_m^n.$$

Taylor expansion of all the terms around (x_m, t_n) gives

$$k\tau_m^n = \left(1 - \theta k(\partial_x^2 + \frac{1}{12}h^2\partial_x^4 + \dots)\right)\left(k\partial_t + \frac{1}{2}k^2\partial_t^2 + \frac{1}{6}k^3\partial_t^3\right)u_m^n - k\left(\partial_x^2 + \frac{1}{12}h^2\partial_x^4 + \dots\right)u_m^n$$

$$= \left(k\partial_x^2 + \frac{1}{2}k^2\partial_t^2 + \frac{1}{6}k^3\partial_t - \theta k^2\partial_t^2 - \frac{1}{2}k^3\partial_t^3 - k\partial_x^2 - \frac{1}{12}kh^2\partial_x^4 + \dots\right)u_m^n + \dots$$

$$= \left(\frac{1}{2} - \theta\right)k^2\partial_t^2 - \frac{1}{12}kh^2\partial_x^4 u_m^n + \left(\frac{1}{6} - \frac{1}{2}\theta\right)k^3\partial_t^3 u_m^n + \dots$$

Finally, dividing by k we find that

$$\tau_m^n = \mathcal{O}(k + h^2) \quad \text{for } \theta \neq \frac{1}{2},$$

$$\tau_m^n = \mathcal{O}(k^2 + h^2) \quad \text{for } \theta = \frac{1}{2}.$$

As in section 1 we now analyze the convergence of the numerical solution methods by doing mesh refinement. We restrict the analysis to just Backward Euler and Crank-Nicolson, which have different convergence order in the temporal direction. The reason for which we exclude the forward Euler method is that it is only stable under the condition that $r = k/h^2 < 1/2$, and such a condition would impose impractical restrictions on the grids and the refinement [6]. Backward Euler and Crank-Nicolson however, are unconditionally stable [6]. Later on when we study the Linearized Kortweg-de Vries equation in section 5, we show this by applying Von Neumann analysis on the forward Euler and Crank-Nicolson methods.

We start by refining the spatial grid x_m sepparately, and we compute both the $l2$ discrete and the L_2 continous relative errors. For (11), however, the the analytical solution is not available in closed form, so we cannot compute the exact errors. In order to analyze convergence we therfore compute a reference solution, using a high resolution spatial grid with $M_{ref} = 10000$ spatial nodes, which we use in place of the analytical solution when computing the error. Since we are only refining in the spatial direction, we keep the number of time steps N fixed, and compute the numerical solution and the errors at the same point in time t with different values of M. This way the error in the time step will be constant throughout the refinement, allowing us to analyze the spatial convergence isolated.

The resulting convergence rates from the refinement is plotted in figure 6, together with the expected convergence of $\mathcal{O}(h^2)$ in the spatial direction. We see that the errors for the most part fail to follow the expected curve, instead they quickly flatten and remain about constant throughout the refinement. Crank-Nicolson with $N = 1000$ is the exception, but even it's curve flattens towards the end of the refinement when

Figure 6: Convergence plots from uniformly refining the mesh in the spatial direction while keeping the number of time steps N constant. The equation solved is the one-dimensional heat equation $u_t = u_{xx}$ with the Neumann boundary conditions $u_x(0, t) = u_x(1, t) = 0$ and initial condition as listed in equation (11). The l_2 discrete and the L_2 continous relative errors is computed with respect to a numerical reference solution computed with $M_{ref} = 10000$, since the analytical solution for this problem is not available on closed form.

M gets large enough. A flattend curve means that the error is dominated by the time step error, and further refining the spatial grid gives in that case only diminishing returns. We also see that the error curve of the solution computed with Crank-Nicolson flattens later than those of the solutions computed with Backward Euler for the same value of N. This reflects the fact that Crank-Nicolson is one order more precise in the time step k. Also, with a higher N we would be able to see the spatial $\mathcal{O}(h^2)$ convergence better.

In in order to analyze the convergence further we now switch to a set of boundary and initial conditions for which we can solve the heat equation analytically and compute the error properly. Specifically we consider

$$u(0, t) = u(1, t) = 0, \quad u(x, 0) = \sin(\pi x), \tag{12}$$

on the same domain $x \in [0, 1] := \Omega$ and $t > 0$. Note that we now have Dirichlet boundary conditions, and we have plotted the numerical solution in 5b. The analytical solution, which can be calculated using sepparation of variables [1] [2], is readily available as

$$u(x, t) = \sin(\pi x)e^{-\pi^2 t}. \tag{13}$$

We now do the same spatial refinement for equation (12), however now we compute the errors with respect to the analytical solution. The resulting convergence plots are shown in figure 7a, and here the characteristics we saw in 6 are seen more clearly. Again, Crank-Nicolson with $N = 1000$ performs the best, and displays here second order convergence througout the whole refinement. The others start out with second order convergence, but flattens as the temporal error starts to dominate, and also here we see that the error in Crank-Nicolson flattens later than the error in Backward Euler for the same value of N.

Now we proceed to do refinement in the t-direction, while keeping the spatial step size constant. I.e. we do the same as in the spatial refinement, but instead we fix M and vary N. The resulting convergence

---

[1]We ommit the calculation since this is a very standard introductory problem for solving partial differential equations with sepparation of variables.
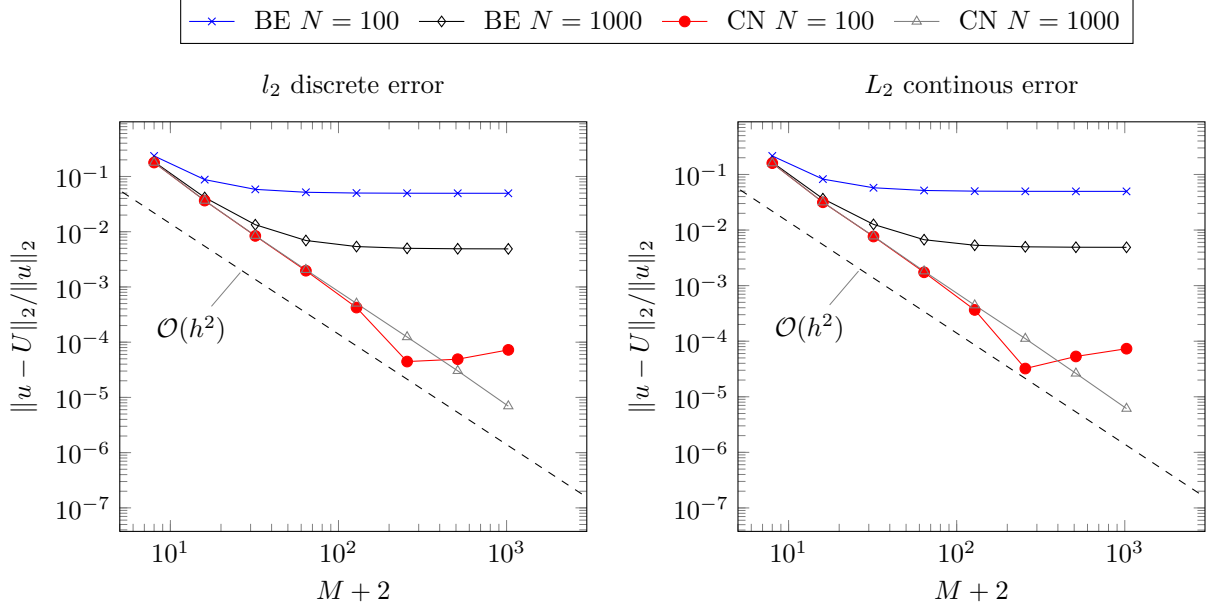
(a) Convergence plots from refinement in the spatial direction.
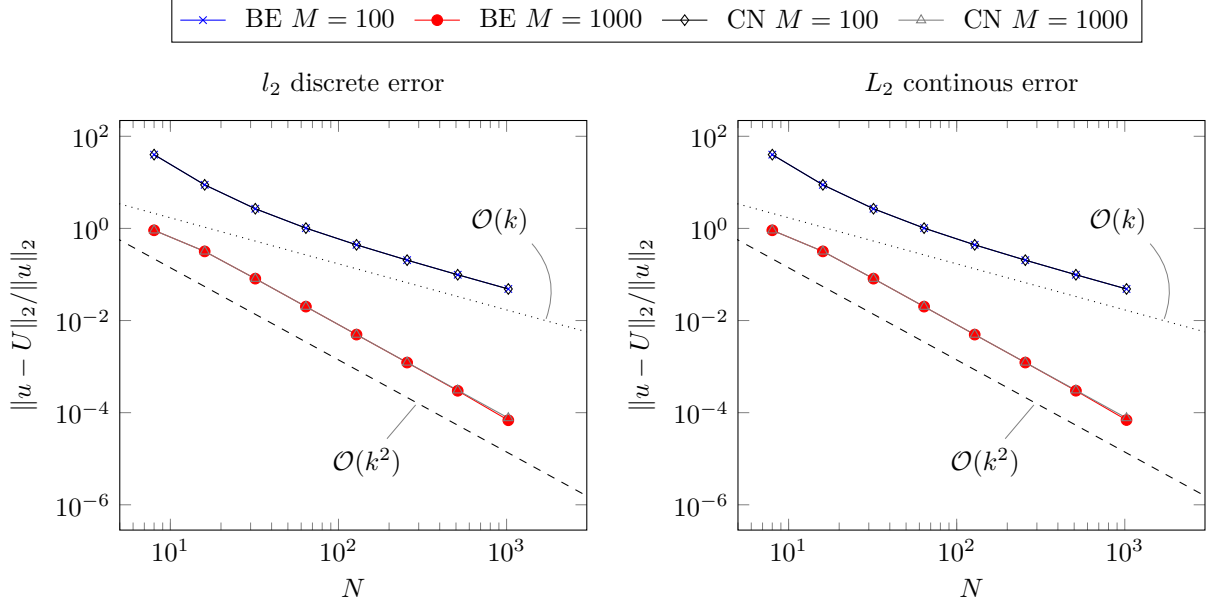
(b) Convergence plots from refinement in the time direction.

Figure 7: Convergence plots from uniform mesh refinement. In 7a we refine in the spatial direction while keeping the number of time steps N constant, then in 7b we refine in the temporal direction while keeping the number of spatial nodes M constant. In both cases both the l_2 discrete and the L_2 continuous relative errors is computed. The equation solved is the one-dimensional heat equation $u_t = u_{xx}$ with a sinusoidal initial temperature distribution and the temperature kept fixed at 0 on the boundaries, corresponding to the conditions listed in equation (12).

plot is show in figure 7b, and here the difference in the temporal convergence of the two methods becomes very apparent. The chosen fixed values for M are large enough so that the spatial error does not dominate, and we avoid the error curves flattening as N gets large. The error curves also seem to follow the expected convergence order.

Having investigated the spatial and temporal convergence sepparately, we now look at the convergence when refining in the x- and t-directions simultaneously. We start with refinement where the time step and spatial step are varied at equal rates, by keeping $c = k/h$ constant. Since we are now refining in both time and space, we want to express the convergence in terms of the system's number of degrees of freedom N_{dof}, which we also will have on the x-axis in the convergence plot to plot the error up against. Defining $M^* = M + 2$ denoting the total number of spatial nodes, we can write $N_{\text{dof}} = M^*N$. Using that $M^* = 1/h + 1$ and $N = 1/k + 1$ as well as our refinement restriction $c = k/h$, we get that

$$
\begin{aligned}
N_{\text{dof}} = M^*N &= \left(\frac{1}{k} + 1\right)\left(\frac{1}{h} + 1\right) \\
&= \left(\frac{1}{h} + 1\right)\left(\frac{1}{ch} + 1\right) \\
&= \frac{1}{ch^2} + \frac{1}{ch} + \frac{1}{h} + 1 \\
&= \mathcal{O}(\frac{1}{h^2}).
\end{aligned}
$$

To relate this to the convergence rates we again use that $c = k/h$ to get

$$\text{Backward Euler:} \quad \mathcal{O}(k + h^2) = \mathcal{O}(ch + h^2) = \mathcal{O}(h) = \mathcal{O}(N_{\text{dof}}^{-1/2})$$
$$\text{Crank-Nicolson:} \quad \mathcal{O}(k^2 + h^2) = \mathcal{O}(c^2h^2 + h^2) = \mathcal{O}(h^2) = \mathcal{O}(N_{\text{dof}}^{-1}).$$

The resulting convergence plots with computed error curves, as well as the expected convergence rates, is shown in figure 8a.

Finally we will do simultaneous refinement where we keep $r = k/h^2$ constant. For the degrees of freedom N_{dof} we now get

$$
\begin{aligned}
N_{\text{dof}} = M^*N &= \left(\frac{1}{k} + 1\right)\left(\frac{1}{h} + 1\right) \\
&= \left(\frac{1}{h} + 1\right)\left(\frac{1}{rh^2} + 1\right) \\
&= \frac{1}{rh^3} + \frac{1}{rh^2} + \frac{1}{h} + 1 \\
&= \mathcal{O}(\frac{1}{h^3}),
\end{aligned}
$$

and for the convergence rates we get

$$\text{Backward Euler:} \quad \mathcal{O}(k + h^2) = \mathcal{O}(rh^2 + h^2) = \mathcal{O}(h^2) = \mathcal{O}(N_{\text{dof}}^{-2/3})$$
$$\text{Crank-Nicolson:} \quad \mathcal{O}(k^2 + h^2) = \mathcal{O}(r^2h^4 + h^2) = \mathcal{O}(h^2) = \mathcal{O}(N_{\text{dof}}^{-2/3}).$$

The resulting convergence plot for this final refinement is shown in figure 8b.

We see from the convergence plots in figure 8a and figure 8b, that our solver gives the expected convergence. The take home point from here is that the order of convergence when doing simultaneous refinement in both directions depends on the refinement strategy. We see that keeping $c = k/h$ constant is appropriate for Crank-Nicolson, which has the same order of convergence in both h and k, while being suboptimal for Backward Euler which is $\mathcal{O}(k + h^2)$. On the other hand, keeping $r = k/h^2$ works well for Backward Euler, while being suboptimal for Crank-Nicolson, and results in the same convergence order for the two methods, despite Crank-Nicolson having better precision in the time step k.
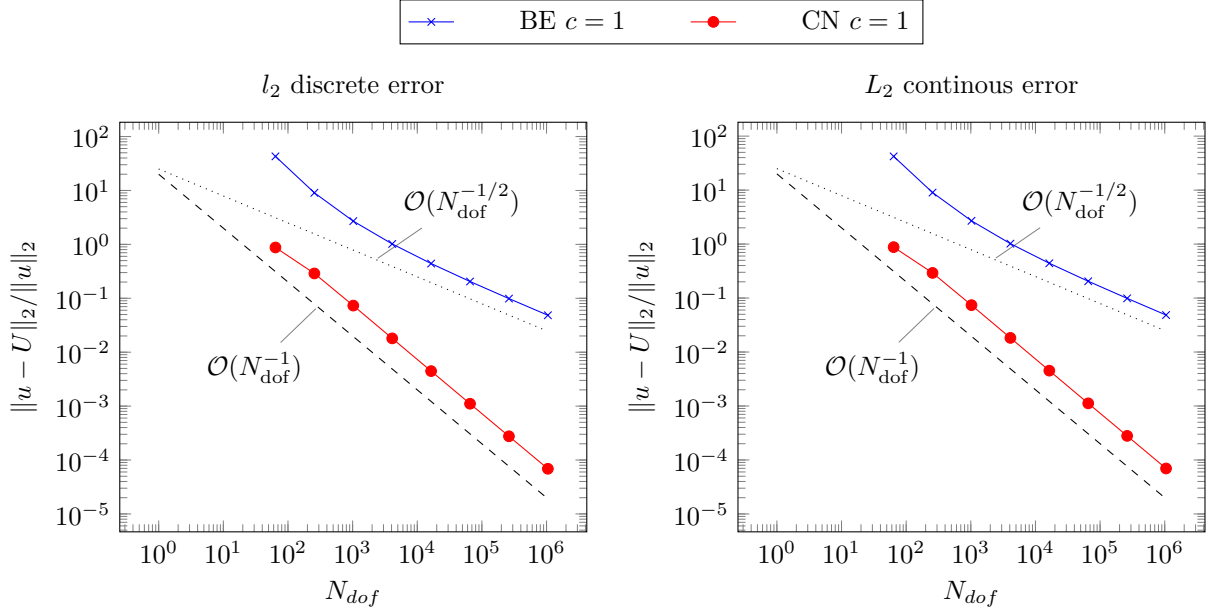
(a) Convergence plots from uniform refinement with constant $c = k/h$.
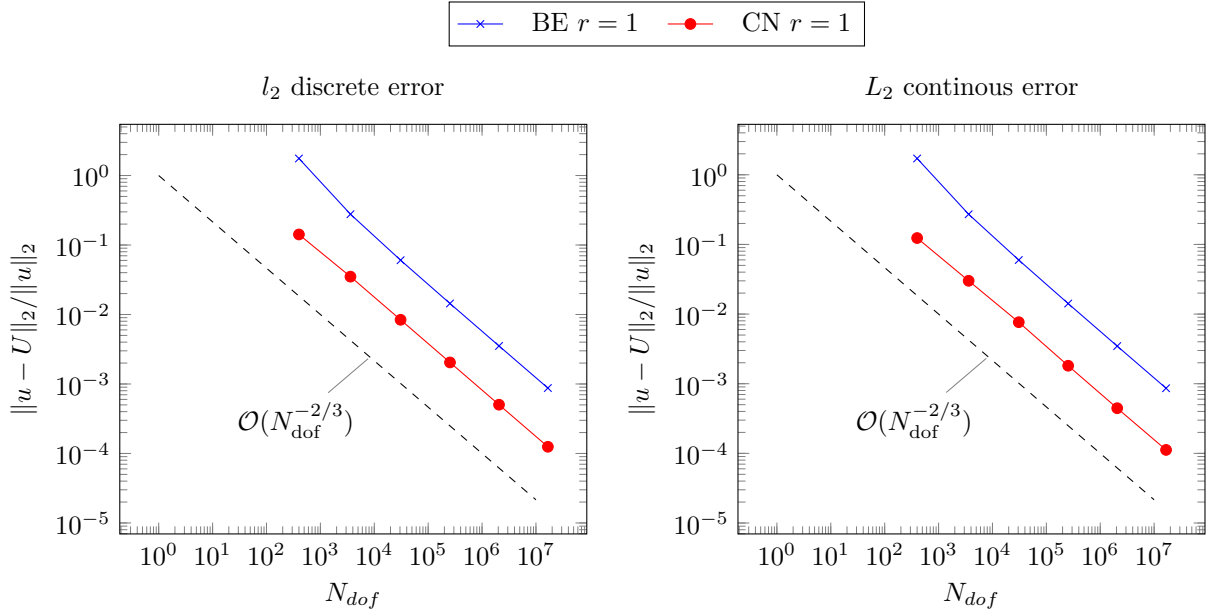
(b) Convergence plots from uniform refinement with constant $r = k/h^2$.

Figure 8: Convergence plots from uniform mesh refinement in both the x- and t-direction simultaneously. In 8a $c = k/h$ is kept constant, and in 8b it is $r = k/h^2$ that is kept constant. In both cases both the l_2 discrete and the L_2 continous relative errors is computed. The equation solved is the one-dimensional heat equation $u_t = u_{xx}$ with a sinusoidal initial temperature distribution and the temperature kept fixed at 0 on the boundaries, corresponding to the conditions listed in equation (12).

3 Inviscid Burgers' equation

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

$$u_t = -uu_x, \quad u(0,t) = u(1,t) = 0, \quad u(x,0) = \exp\left(-400(x - 1/2)^2\right). \tag{14}$$

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* [4]. The time t_b before this happens can be found exactly using the method of characteristics, and is given as

$$t_b = \frac{-1}{\min f'(x)}, \tag{15}$$

where $f(x)$ is the given initial condition $u(x,0) = f(x)$ [4].

Numerical solution method

To solve equation (14) 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

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

which we, after imposing the boundary conditions, integrate using an explicit Runge-Kutta method of order 4(5).

3.1 Time of breaking

Insertion of $u(x,0)$ for f in (15), gives $t_b \approx 0.058$. As 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, and then strictly decreasing from the apex towards the right boundary at $x = 1$. When this is no longer the case, or equivalently, when there exists a point U_m such that $U_m < U_{m-1}$ and $U_m < U_{m+1}$, we say that the numerical solution has broken down. The time for which this happened for our solution was at $t^* \approx 0.055$, and figure 9 shows the solution sampled around the time of breaking.
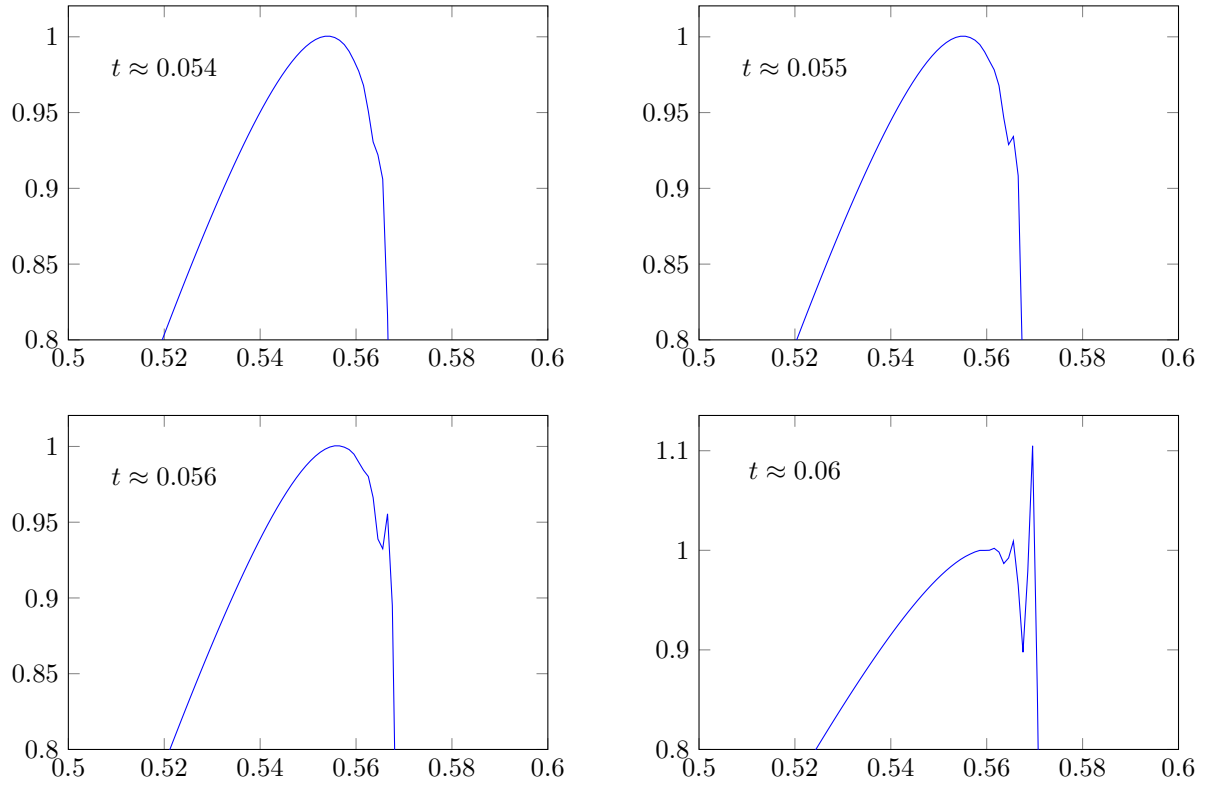
Figure 9: Numerically computed solution to the inviscid Burgers' equation around the time of breaking, displaying the shock formation. The initial condition is $u(x,0) = \exp\left(-400(x - 1/2)^2\right)$, and the boundaries are fixed at 0 by the Dirichlet conditions, as specified in (14). The upper right plot with $t \approx 0.055$ is at 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

$$u_{xx} + u_{yy} = 0, \ (x, y) \in \Omega := [0, 1]^2, \tag{16}$$

with boundary conditions on the edges of Ω

$$
\begin{aligned}
u(0, y) &= 0, \\
u(1, y) &= 0, \\
u(x, 0) &= 0, \\
u(x, 1) &= \sin(2\pi x).
\end{aligned}
\tag{17}
$$

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 16 can be found by separation of variables. First, asssume that we can write

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

which implies that

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

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

$$\partial_x^2 u(x_m, y_n) = \frac{1}{h^2}[u(x_{m-1}, y_n) + 2u(x_m, y_n) + u(x_{m+1}, y_n)] + \mathcal{O}(h^2)$$
$$= \frac{1}{h^2}\delta_x^2 u(x_m, y_n) + \mathcal{O}(h^2),$$

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

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 10,

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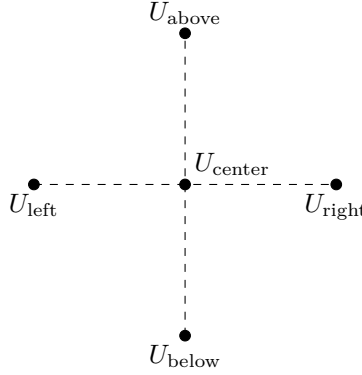

Figure 10: The five-point stencil corresponding to central difference differentiation in both the x- and y-direction. In order to make this more concrete, one can imagine that this stencil is inserted into any point inside a grid such as the one in figure 4. Repeating this process will yield equations for all nodes in the grid, resulting in a solvable system of equations.

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 (17).

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 11.
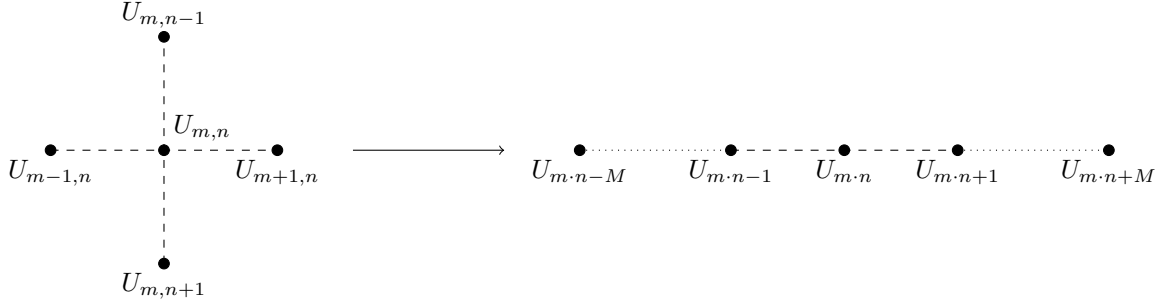

Figure 11: By flattening the five-point stencil, we can write the system of equations, which is then on the form $AU = 0$.

We thus write

$$AU = \begin{bmatrix} \frac{-2}{h^2} + \frac{-2}{k^2} & \frac{1}{h^2} & & \frac{1}{k^2} & & & \\ \frac{1}{h^2} & \frac{-2}{h^2} + \frac{-2}{k^2} & \frac{1}{h^2} & & \frac{1}{k^2} & & \\ & \frac{1}{h^2} & \frac{-2}{h^2} + \frac{-2}{k^2} & 0 & & \frac{1}{k^2} & \\ & \ddots & \ddots & \ddots & & & \\ \frac{1}{k^2} & & 0 & \frac{-2}{h^2} + \frac{-2}{k^2} & \frac{1}{h^2} & & \\ & \frac{1}{k^2} & & \frac{1}{h^2} & \frac{-2}{h^2} + \frac{-2}{k^2} & \frac{1}{h^2} \\ & & \frac{1}{k^2} & & \frac{1}{h^2} & \frac{-2}{h^2} + \frac{-2}{k^2} \end{bmatrix} \begin{bmatrix} U_1 \\ \vdots \\ U_{\text{m}} \\ \vdots \\ U_{N \times m} \\ \vdots \\ U_{N \times M} \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_m \\ \vdots \\ b_{N \times m} \\ \vdots \\ b_{N \times M} \end{bmatrix} = b,$$
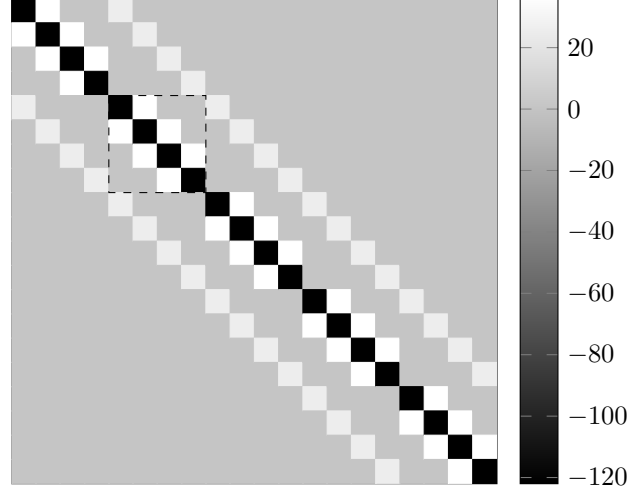
Figure 12: The five point stencil matrix for the case $M = 4$, $N = 5$. Notice the recursive structure, and the fact that some elements along the first off-diagonals are zero. These elements correspond to nodes along the edge of the system.

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, which come from b. As was mentioned briefly above, the values in b are set in points that correspond to the borders of the system. In this way, the edges of the system are determined with the boundary conditions. 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 12, 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. This procedure is discussed in depth in section 7, where we show that the stencil can be represented by the Kroenecker sum 43.

Using the method described above, the solution to equation 16 has been computed, and the results are shown in figure 13. An error analysis showing the error with varying grid resolutions in both the x- and the y-direction is presented in 14. We notice that the error decreases as $\mathcal{O}(h^2)$, but that it flats out to $\mathcal{O}(h)$ when M and N are approximately equal – that is, of the same order of magnitude – and further to order $\mathcal{O}(h^0)$ as the difference grows. The convergence plot shows that varying the grid size in either direction when $M \approx N$, converges to first order. However, when $M \neq N$, varying the smaller one increases accuracy to second order, and increasing the largest one, yields almost no gain in accuracy.

This error behaviour is intuitive: When the resolution is a lot higher in one direction than the other, then the error is almost entirely due to the direction with the lowest resolution. Thus, increasing the resolution in the most precise direction gives only insignificant increases in the overall error.
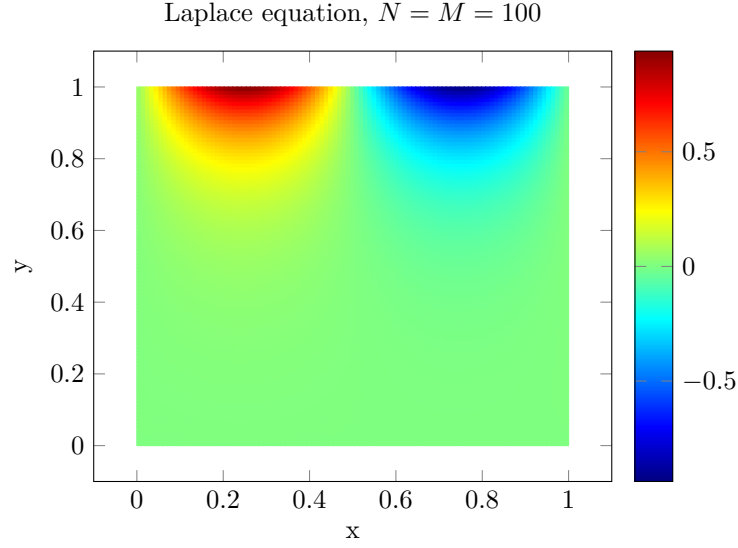
Figure 13: The numerically computed solution to the Laplace equation defined in equation (16) and (17), for a uniform grid with $N = M = 100$.
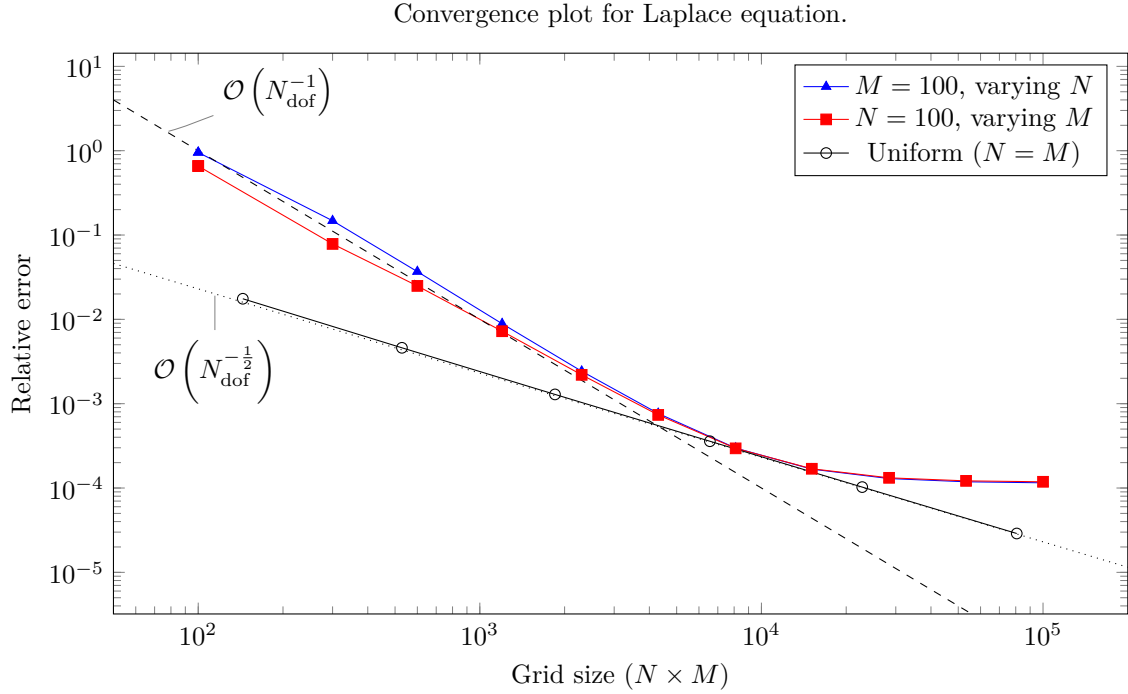

Figure 14: Convergence plot showing the relative error in the computation of the Laplace equation – equation (16) and (17), when varying N and M, corresponding to the y- and the x-direction, respectively.

5 Linearized Korteweg-de Vries equation in one dimension

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

$$\frac{\partial u}{\partial t} + \left(1 + \pi^2\right)\frac{\partial u}{\partial x} + \frac{\partial^3 u}{\partial x^3} = 0 \qquad (t \geq 0) \quad (-L/2 \leq x \leq +L/2), \tag{18}$$

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

$$u(x + L, t) = u(x, t). \tag{19}$$

The equation will be solved using the forward Euler and Crank-Nicolson method, and we will conduct a stabiity analysis, showing that – in this paricular case – the choice of method is crucial.

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

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

Now insert $u_n(x,t) = X_n(x)\,T_n(t)$ into equation (18) 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 $X_n(x) = e^{iq_n x}$ 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 (18) and the periodic boundary condition, we can superpose multiple solutions $u_n(x,t)$ into a general solution

$$u(x,t) = \sum_{n=-\infty}^{+\infty} c_n \exp\left(i(q_n x - \omega_n t)\right), \tag{20}$$

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

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-Nicolson method. We will find the solution on the periodic spatial grid of M points. For the first spatial derivative, we use the central difference

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

x_2

h

x_{m-1}

h

x_1

h

x_m

h

$x_0 = -L/2$

h

x_{m+1}

x_{M-1}

h

x_{M-2}

Figure 15: A depiction of the one dimensional periodic grid on which the linearized Korteweg-de Vries equation is computed. Since the boundary conditions – equation (19) – are periodic, a circular grid is equivalent to an arbitrary number of successive linear grids.

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 (18), 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-Nicolson method collectively with the θ-method. This gives the final system of difference equations for the numerical solution

$$\frac{U_m^{n+1} - U_m^n}{k} = (1-\theta)F(U^n) + \theta F(U^{n+1}), \tag{21}$$

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

$$(I - \theta k A)\,U^{n+1} = \left(I + (1-\theta)\,kA\right)U^n, \tag{22}$$

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

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

27

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 (22) repeatedly to step forward in time. Note that with constant steps h and k in both space and time, the matrices in equation (22) are constant. To save both memory and time, we represent them with sparse matrices. [13] In addition, to efficiently solve the same system with different right hand sides many times, we *LU*-factorize the sparse matrix for $I - \theta k A$. [11] 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 20 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-Nicolson method. Note that the Crank-Nicolson 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-Nicolson 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-Nicolson 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]

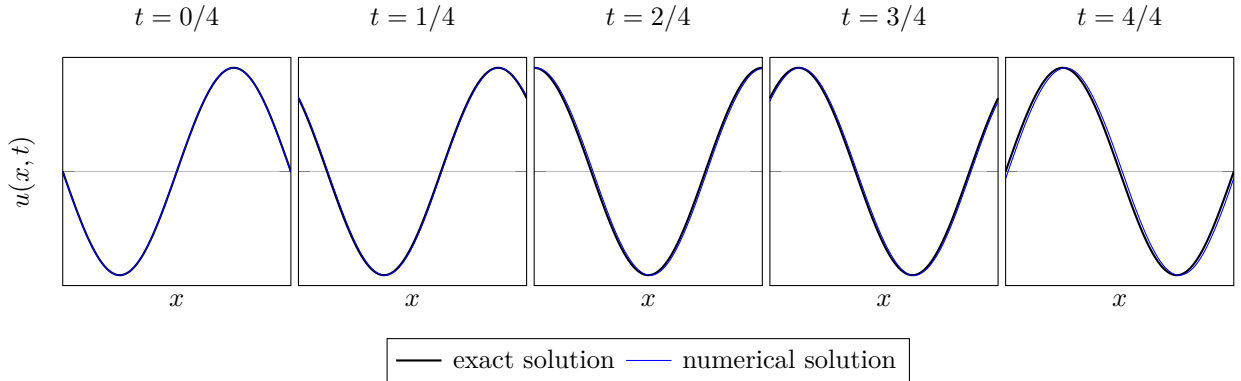$$U_m^n = U(x_m, t_n) = \sum_l C_l^n \exp(iq_l x_m). \tag{23}$$

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-Nicolson method with $h = 1/799$ and $k = 1/99$.
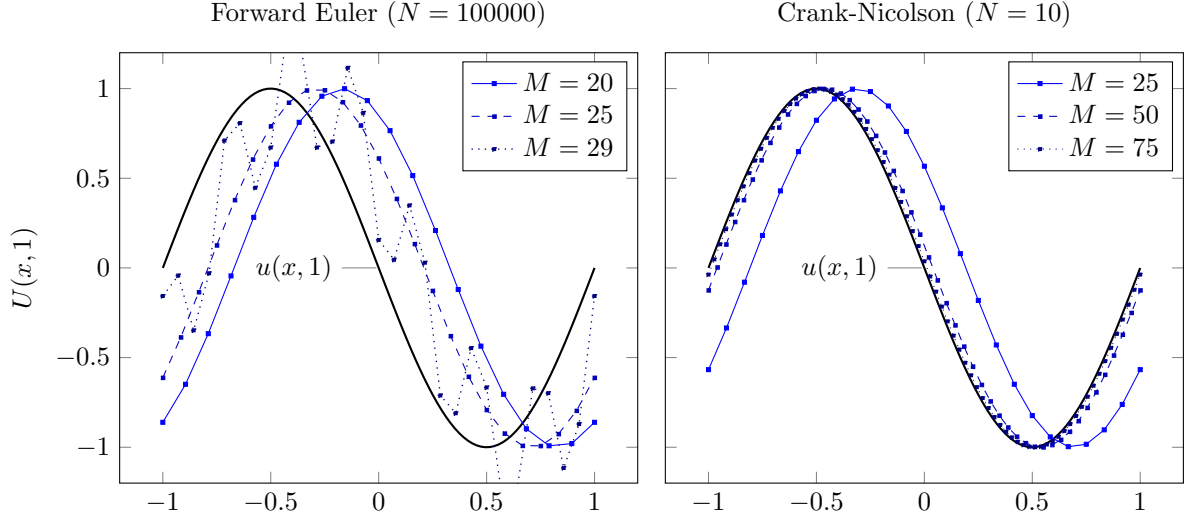
Figure 17: Snapshots of the numerical solution $U(x,1)$ and the exact solution $u(x,1)$ for a constant number of time steps N, but varying number of grid points M with the Forward Euler and Crank-Nicolson method. The left plot is meant to demonstrate the downfall of the Euler method and is not supposed to look pretty.
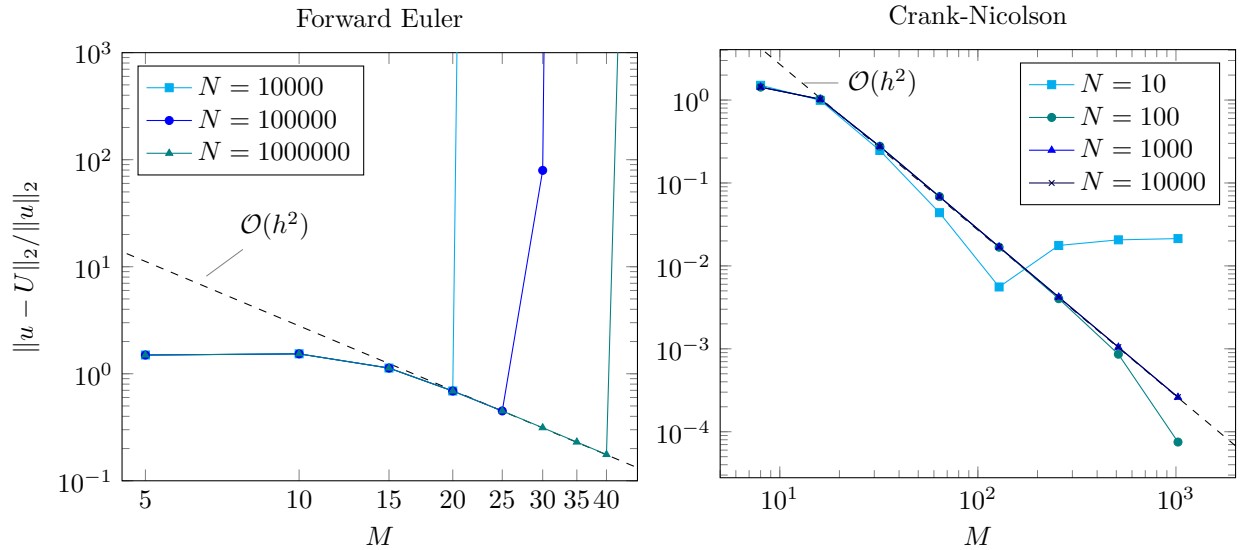

Figure 18: Convergence plots with the discrete L_2 error of the numerical solution $U(x,t)$ for the Forward Euler and Crank-Nicolson method on the problem defined by the exact solution $u(x,t) = \sin(\pi(x-t))$.

Consider now a single Fourier mode $C_l^n \exp(iq_l x_m)$ in this series. Inserting it into equation (21), 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-Nicolson method gives

$$G_l = \frac{1 + ikf(q_l)/2}{1 - ikf(q_l)/2} \quad \Longrightarrow \quad |G_l| = 1. \tag{24}$$

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

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

$$G_l = 1 + ikf(q_l) \quad \Longrightarrow \quad |G_l| = \sqrt{1 + k^2 f(q_l)^2}. \tag{25}$$

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-Nicolson 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 20 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
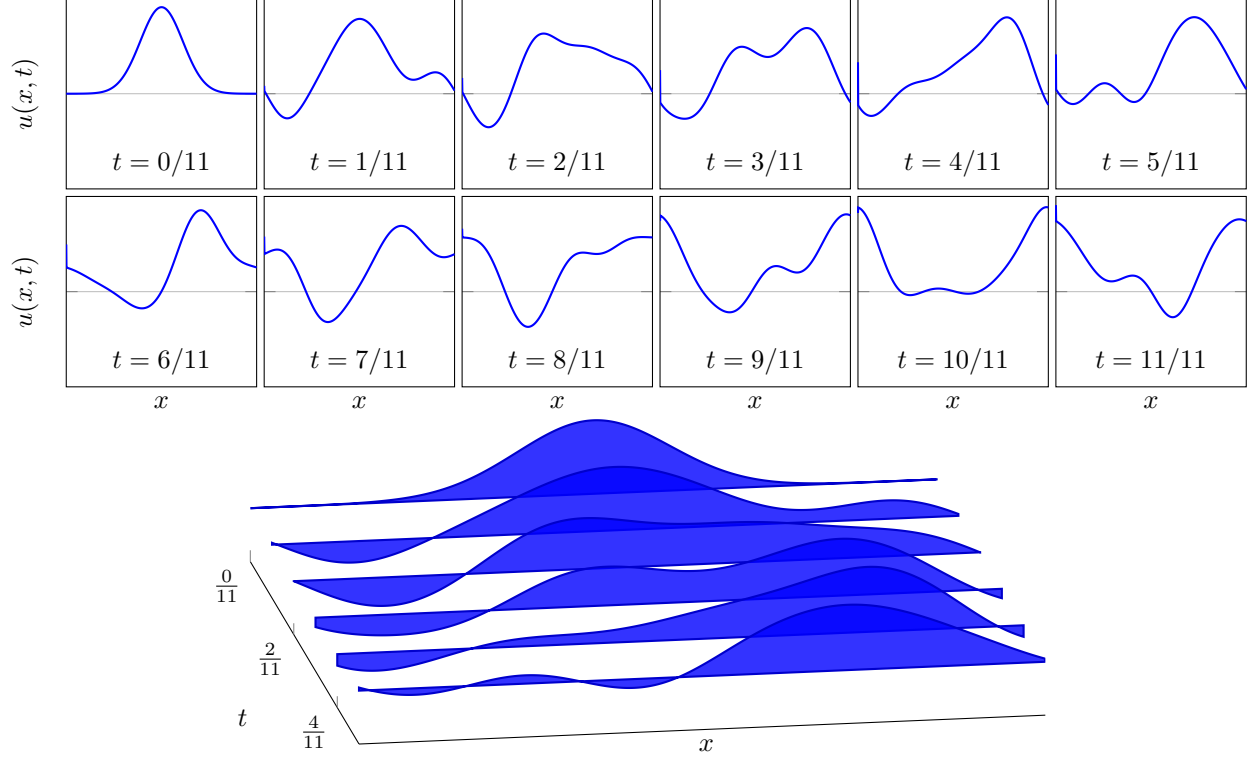
Figure 19: Time evolution of a initial gaussian $u(x,0) = \exp(-x^2/0.1)$ computed from the Crank-Nicolson method on a grid with $M = 800$ points in space and $N = 100$ points in time.
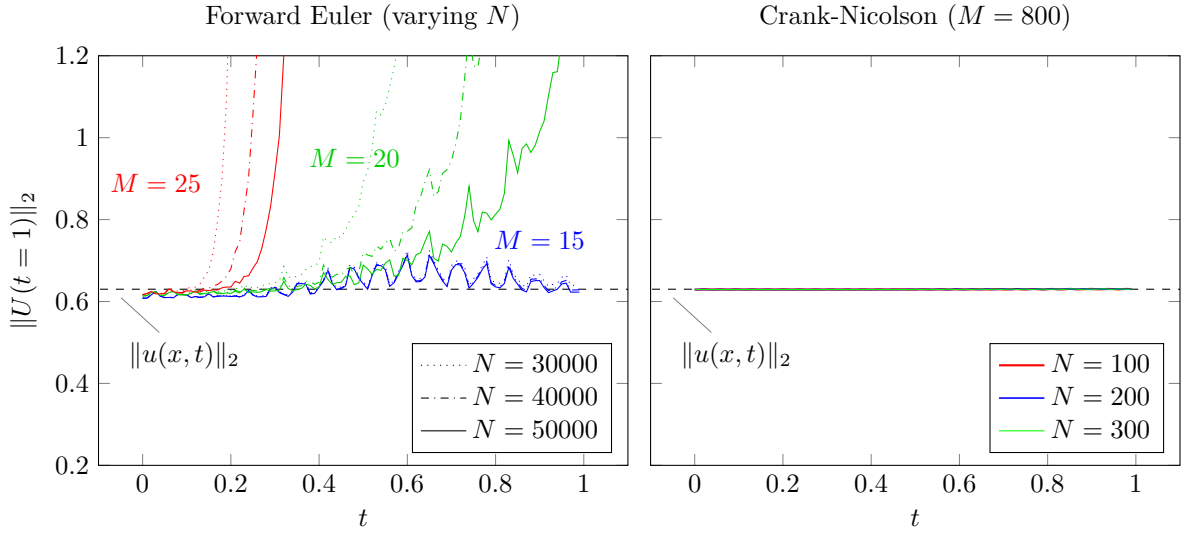

Figure 20: Time evolution of the discrete L_2-norm of the initial gaussian $u(x,0) = \exp(-x^2/0.1)$ computed from the Euler and the Crank-Nicolson method, on different grids.

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 (25). In contrast, the Crank-Nicolson

metohd is always stable and preserves the norm of the solution, reflecting the amplification factor $G_l = 1$ found in equation (24).

The stability of the Crank-Nicolson 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 elements

In this section, we will again solve the Poisson equation

$$-\frac{\partial^2 u}{\partial x^2} = f(x), \quad u(a) = \alpha, \quad u(b) = \beta, \quad (a \le x \le b) \tag{26}$$

subject to Dirichlet conditions, but this time using finite elements instead of finite differences. Both uniform and adaptive approaches to the grid refinement will be implemented, and we will compare these methods.

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

$$u(x) = C_1 + C_2 x - \int^x \mathrm{d}x' \int^{x'} \mathrm{d}x'' f(x''). \tag{27}$$

6.2 Weak formulation for the exact solution

To derive a finite element method, we will first derive a *weak formulation* of 26 in a way inspired by [1]. First, we split the solution into two terms

$$u(x) = \hat{u}(x) + r(x), \quad \text{with} \quad \hat{u}(a) = \hat{u}(b) = 0 \quad \text{and} \quad r(x) = \alpha \frac{x - b}{a - b} + \beta \frac{x - a}{b - a}. \tag{28}$$

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 26 with homogenuous Dirichlet boundary conditions, while $r(x)$ *lifts* the values at the boundaries to satisfy the inhomogenuous boundary conditions.

Now insert 28 into equation (26), 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:

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

The weak formulation 29 is equivalent to the original boundary value problem 26. Any $u(x)$ that solves 26 also solves 29, 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, but we will now do so, by 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

$$a = x_0 < x_1 < \cdots < x_M < x_{M+1} = b \tag{30}$$

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

$$U(x) = \hat{U}(x) + R(x), \quad \text{with} \quad \hat{U}(a) = \hat{U}(b) = 0 \quad \text{and} \quad R(x) = \begin{cases} \alpha \dfrac{x_1 - x}{x_1 - a} & (a \le x \le x_1) \\ 0 & (x_1 \le x \le x_M) \\ \beta \dfrac{x - x_M}{b - x_M} & (x_M \le x \le b) \end{cases}. \tag{31}$$

Now again insert 31 into 26, 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:
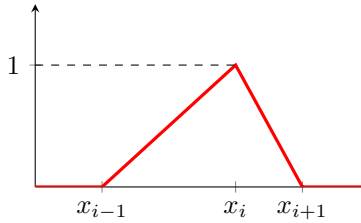
$$\text{Find } \hat{U}(x) \text{ such that} \quad \int_a^b \mathrm{d}x\, \hat{U}'(x)V'(x) = \int_a^b \mathrm{d}x\, f(x)V(x) - \int_a^b \mathrm{d}x\, R'(x)V'(x) \quad \text{for all } V(x). \quad (32)$$

6.4   Numerical solution

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

$$\hat{U}(x) = \sum_{i=0}^{M+1} \hat{U}_i \varphi_i(x) \quad \text{and} \quad V(x) = \sum_{i=0}^{M+1} V_i \varphi_i(x) \quad (33)$$

in a basis for the approximate solution function space, namely the piecewise linear functions on the grid 30. The most natural basis for this space are the functions

$$\varphi_i(x) = \begin{cases} (x - x_{i-1})/(x_i - x_{i-1}) & \text{if } x_{i-1} \le x \le x_i \\ (x_{i+1} - x)/(x_{i+1} - x_i) & \text{if } x_i \le x \le x_{i+1} \\ 0 & \text{otherwise} \end{cases} = \qquad . \quad (34)$$

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 32 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

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

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

$$A\hat{U} = F. \quad (35)$$

This is the matrix equation we will solve to find $U(x)$. After finding $\hat{U}$, we simply sum 33 and add $R(x)$ to find $U(x)$. Note that our particular choice of basis 34 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 35, 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. [8]

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

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

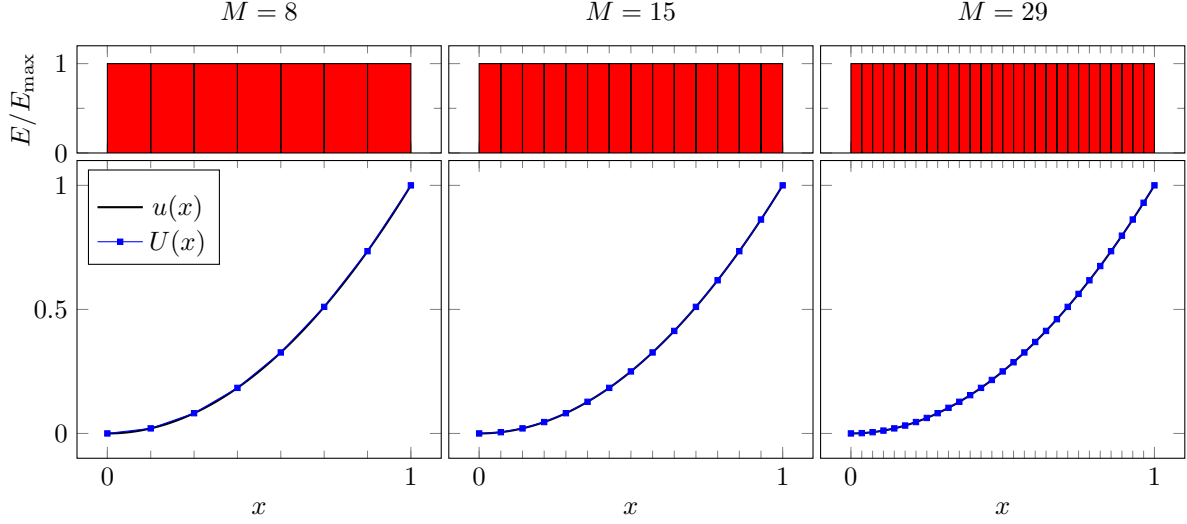$$\|u(x) - U(x)\|_2 > 0.70 \max \|u(x) - U(x)\|_2,$$

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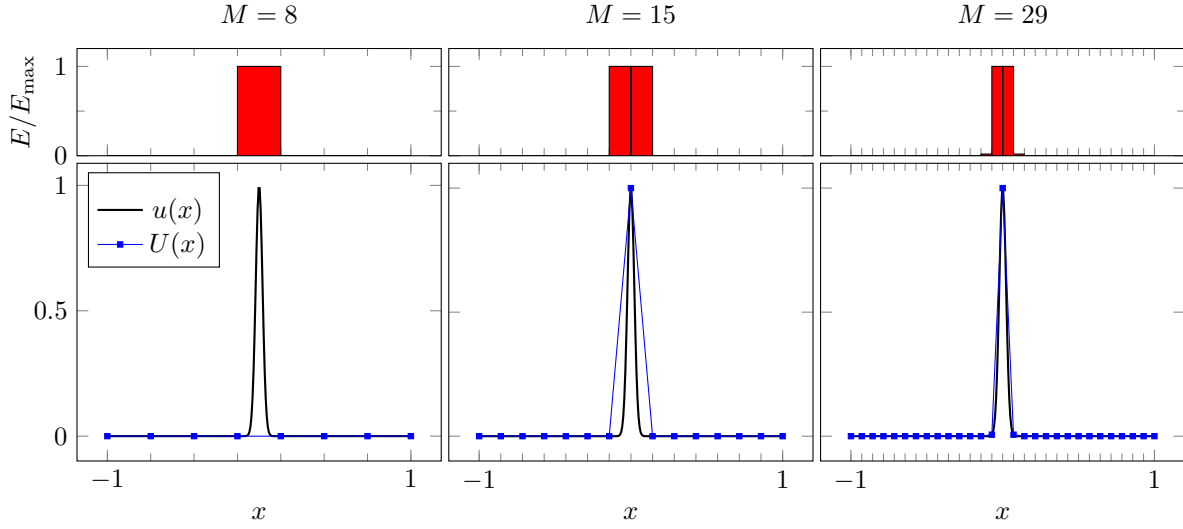
$$f(x) = -2, \quad u(0) = 0, \quad u(1) = 1$$

$M = 8$ $M = 15$ $M = 29$

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

$M = 8$ $M = 15$ $M = 29$
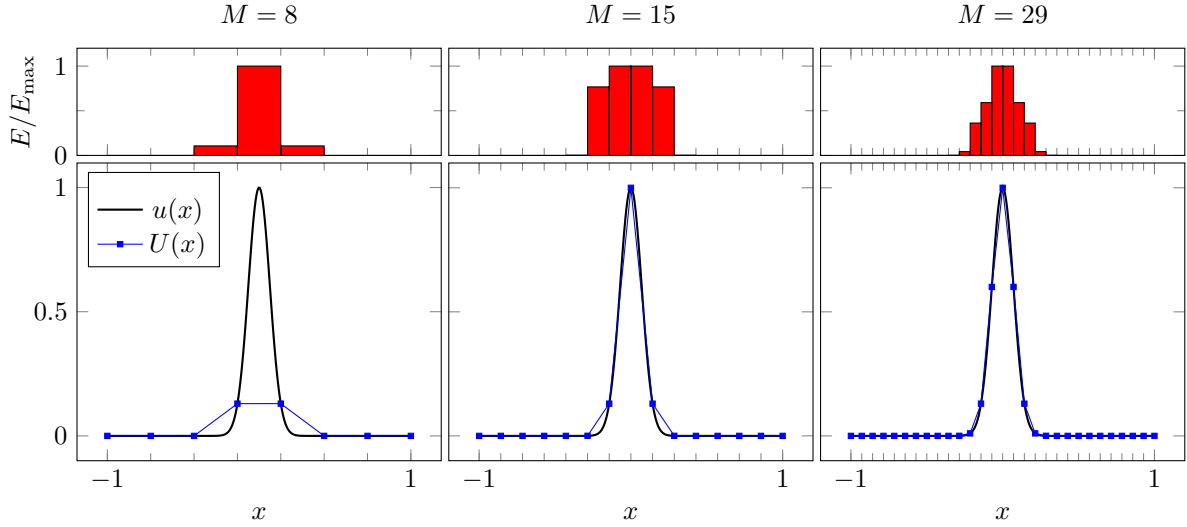
$$f(x) = -(4000000x^2 - 2000)\exp(-1000x^2), \quad u(-1) = e^{-1000}, \quad u(1) = e^{-1000}$$
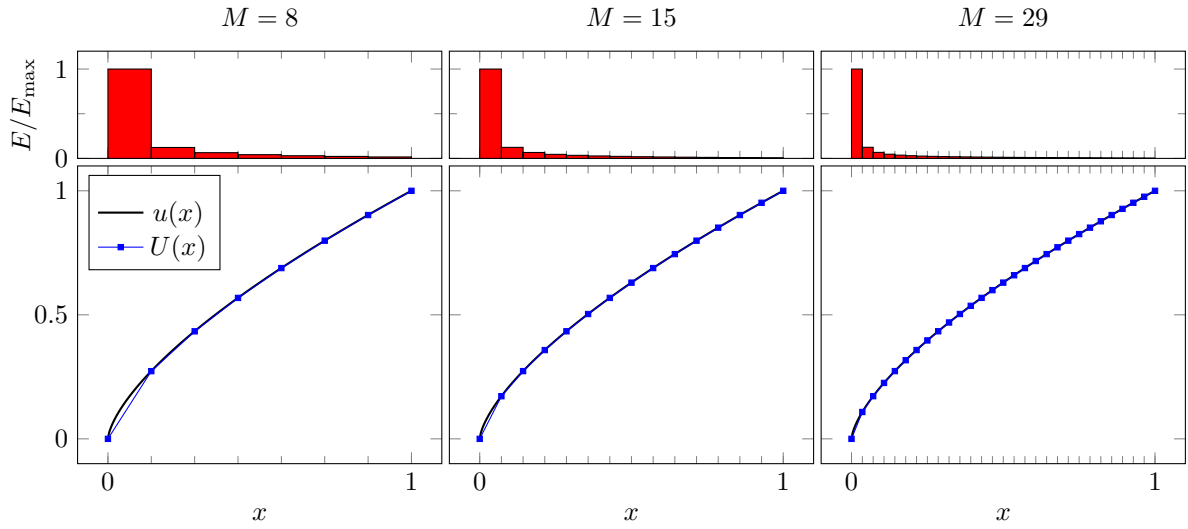

$$f(x) = 2x^{-4/3}/9, \quad u(0) = 0, \quad u(1) = 1$$

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.
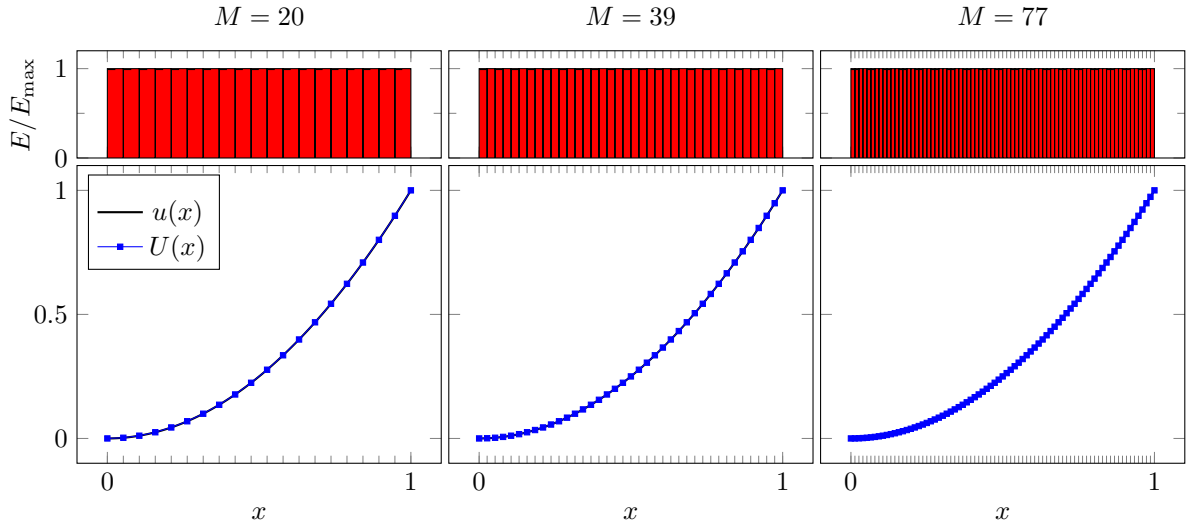
6.4.3 Comparison of convergence

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.
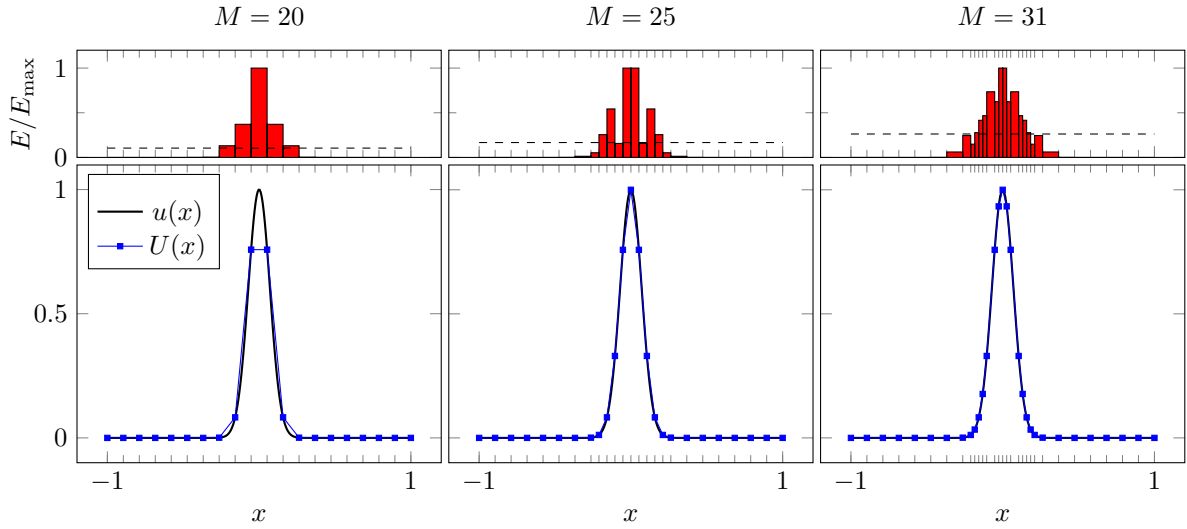
In all problems, adaptive refinement yields greater or equal accuracy for a given number of elements compared to uniform refinement. This is in contrast to what was the case for the finite difference method in figure 3, where adaptive refinement gave errors only comparable and usually larger than those 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.
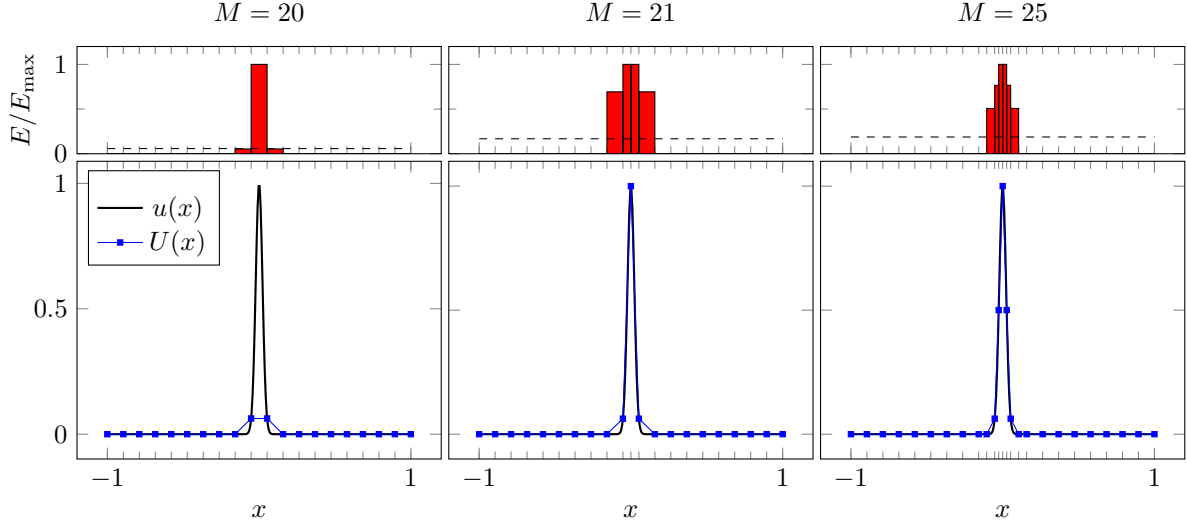
$$f(x) = -2, \quad u(0) = 0, \quad u(1) = 1$$

$M = 20$ $M = 39$ $M = 77$

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

$M = 20$ $M = 25$ $M = 31$

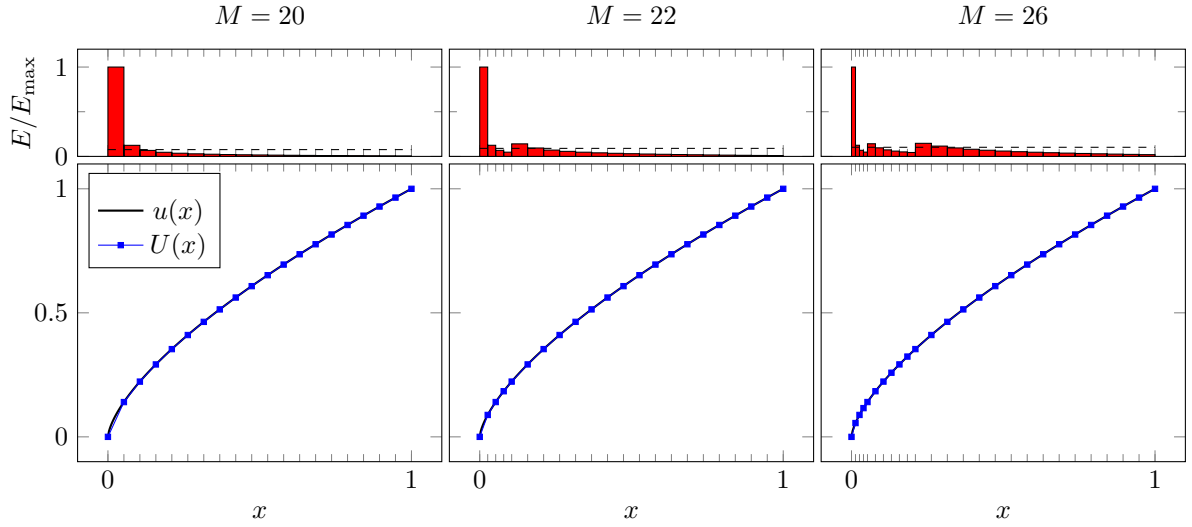**Figure 20:** Adaptive mesh refinement on four problems with the average strategy, showing the evolution of an initial uniform mesh as it is refined over three iterations. Elements whose L_2-error lie above the reference error - - - are split in half.
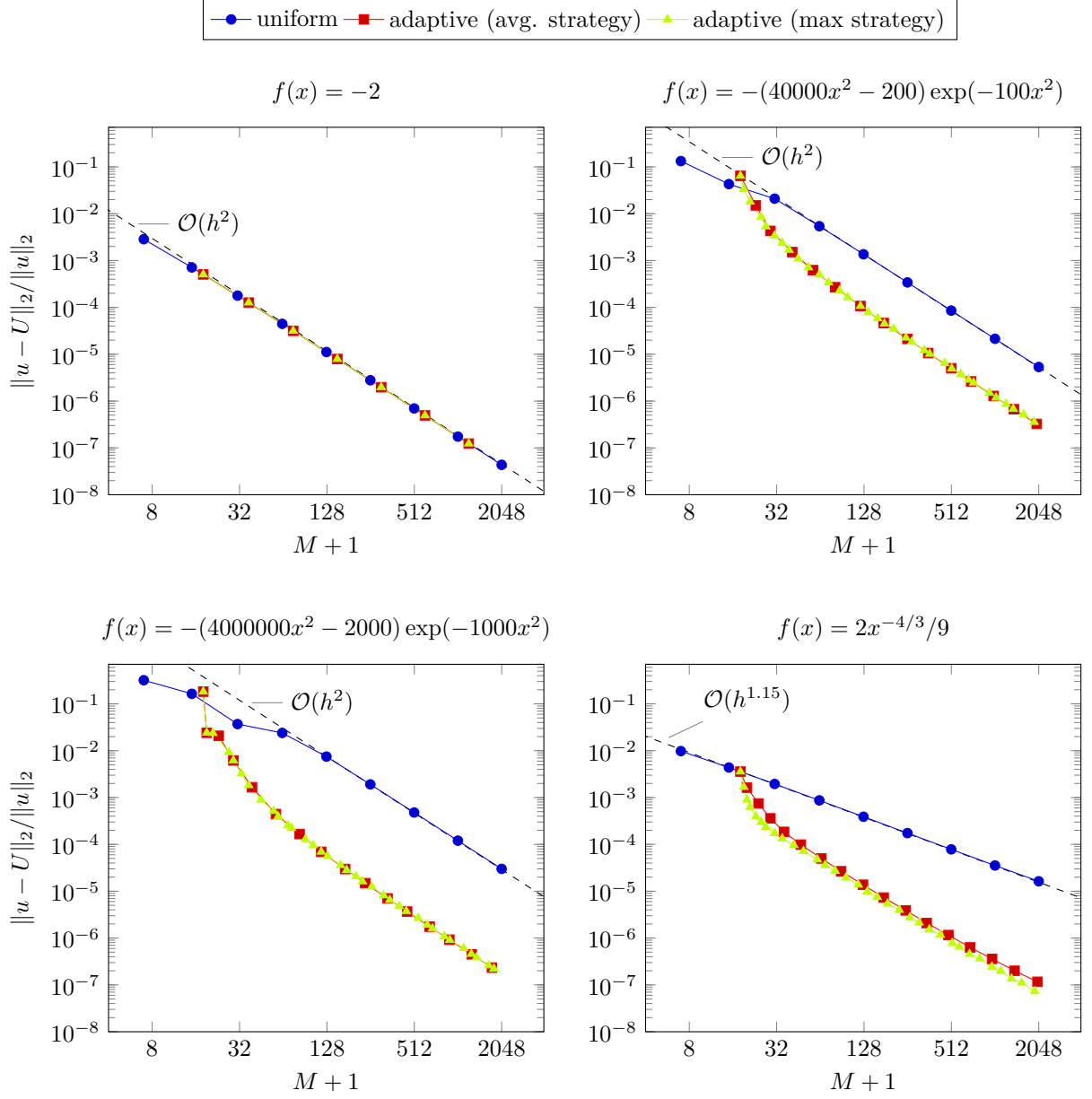
Figure 21: Convergence plots for four problems that compare uniform mesh refinement with the two adaptive mesh refinement strategies. See main text for details.

7 Biharmonic equation

In this section, we will consider the inhomogeneous, two-dimensional Biharmonic equation, with clamped boundary conditions on the domain $\Omega = [0, 1]^2$,

$$\nabla^4 u = f \text{ in } \Omega, \qquad u = \nabla^2 u = 0 \text{ on } \partial\Omega, \tag{36}$$

where $\nabla^2 = \partial_x^2 + \partial_y^2$ and $\nabla^4 = (\nabla^2)^2$. We will solve it with finite difference schemes of both second and fourth order by solving the resulting matrix equations. Then we will describe and implement a specialized *Fast Poisson Solver* that solves the matrix equations very efficiently compared to general solvers.

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

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

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

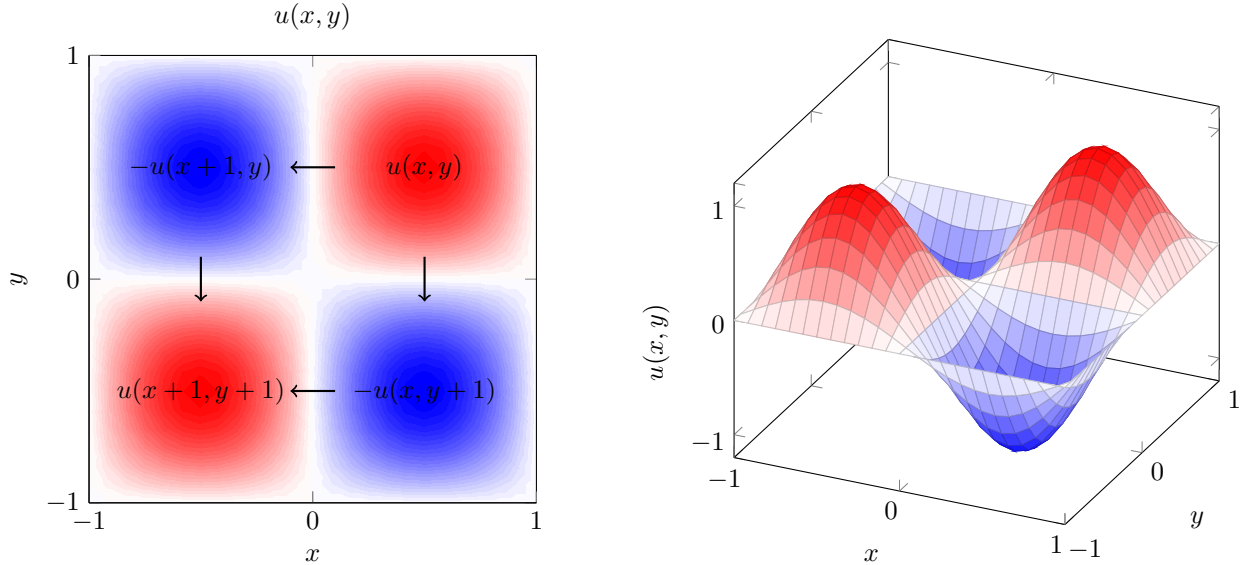$$u_{m,n} = -u_{-m,n} = -u_{m,-n} = u_{-m,-n},$$

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 $[-1, 1] \times [-1, 1]$.

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

$$u(x,y) = \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \hat{u}_{mn} \sin(m\pi x) \sin(n\pi y). \tag{37}$$

Plug this Fourier series into equation (36) 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*

$$f(x,y) = \sum_{m=1}^{\infty} \sum_{n=1}^{\infty} \hat{f}_{mn} \sin(m\pi x) \sin(n\pi y). \tag{38}$$

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

$$\hat{u}_{mn} = \frac{\hat{f}_{mn}}{\left((m\pi)^2 + (n\pi)^2 \right)^2}, \tag{39}$$

and the solution $u(x,y)$ is available by summing its Fourier series 37.

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,

$$\hat{f}_{mn} = 4 \int_0^1 \mathrm{d}x \int_0^1 \mathrm{d}y \, f(x,y) \sin(m\pi x) \sin(n\pi y). \tag{40}$$

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 Finite difference method

The Biharmonic equation 36 can be transformed to a system of Poisson equations

$$\nabla^2 g = f \ \text{ in } \ \Omega, \qquad g = 0 \ \text{ on } \ \partial\Omega \tag{41a}$$

$$\nabla^2 u = g \ \text{ in } \ \Omega, \qquad u = 0 \ \text{ on } \ \partial\Omega. \tag{41b}$$

Instead of solving the Biharmonic equation directly with a stencil for ∇^4, we will solve the two Poisson equations successively with a Poisson stencil $\nabla_n^2 \approx \nabla^2$ only. First, we solve 41a numerically with the exact known source f to obtain an intermediate solution $G \approx g$. Then we solve 41b numerically to obtain the final solution $U \approx u$, but this time we only have access to the approximate source $G \approx g$.

We divide $[0,1] \times [0,1]$ into the uniformly spaced two-dimensional grid
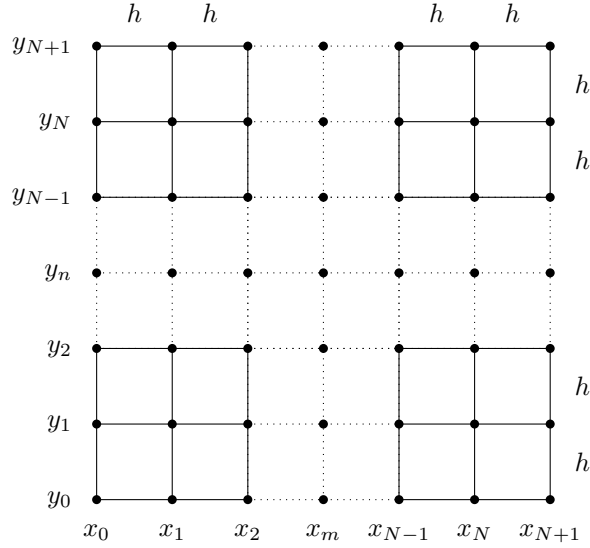
The boundary conditions in 36 already tell us that $g = u = 0$ on $\partial\Omega$, so we will only solve for U in the interior $1 \le m, n \le N$. To do so, we approximate the Laplacian ∇^2 with the 5-point and 9-point stencils
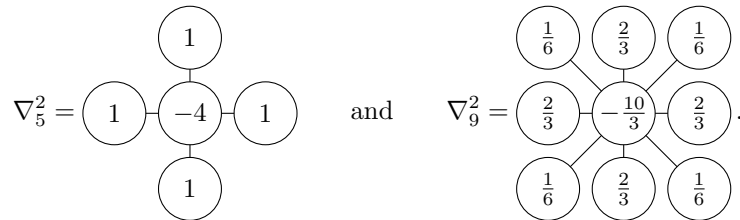
The diagrams define the stencils by their action on some function. For example,

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

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

$$\nabla_5^2 U = F \qquad \text{and} \qquad \nabla_9^2 U = \left(1 + \frac{h^2}{12}\nabla_5^2\right)F, \tag{42}$$

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

Next, we turn to formulating equation (42) 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

$$K_5 = \begin{bmatrix} J_5 & 0 & 0 & \\ 0 & J_5 & 0 & \ddots \\ 0 & 0 & J_5 & \ddots \\ & \ddots & \ddots & \ddots \end{bmatrix} + \begin{bmatrix} -2I & I & 0 & \\ I & -2I & I & \ddots \\ 0 & I & -2I & \ddots \\ & \ddots & \ddots & \ddots \end{bmatrix} = I \otimes J_5 + J_5 \otimes I = J_5 \oplus J_5. \tag{43}$$

For the 9-point stencil, note that it can be split into
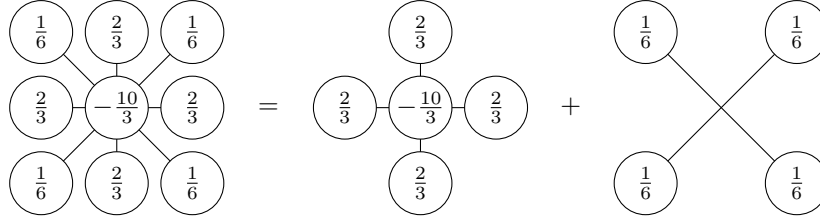
The first term can be handled like the 5-point stencil and is represented by $J_9 \oplus J_9$ with

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

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 42 that we will solve are

$$K_5 U = h^2 F_5 \qquad \text{with} \qquad F_5 = F \qquad \qquad \text{and} \qquad K_5 = J_5 \oplus J_5$$

$$K_9 U = h^2 F_9 \qquad \text{with} \qquad F_9 = \left(I + \frac{1}{12} K_5\right) F \qquad \text{and} \qquad K_9 = J_9 \oplus J_9 + \Sigma \otimes \Sigma. \tag{44}$$

Warning: We know that $U = 0$ on the boundary, so we will only solve for U in the interior $N \times N$ grid. But F need *not* be zero on the boundary. When calculating the right side of the 9-point equation, we must therefore *first* use the $(N+2) \times (N+2)$ grid *with* the boundary to make K_5 sample F there, *then* cut away the boundary and go back to the $N \times N$ grid before solving for U in the interior.

We will solve the matrix equations with sparse matrices. [13] We first represent J_5, J_9 and Σ with sparse matrices and then use [9] to construct sparse representations of Kroenecker products and sums.

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

$$- a_0 f(x_0) = \sum_{i=1}^{n-1} a_i f(x_i) - \nabla_n^2 f(x_0) \leq \sum_{i=1}^{n-1} a_i f(x_i) \leq \sum_{i=1}^{n-1} a_i f(x_0) = -a_0 f(x_0). \tag{45}$$

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. $\square$

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 U is a discrete function that equals zero on $\partial\Omega$, then

$$\|U\|_\infty \leq \max_{\partial\Omega} |\phi_n| \|\nabla_n^2 U\|_\infty. \tag{46}$$

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

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

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

$$\|U\|_\infty \leq \|U + \phi_n M\|_\infty \leq \max_{\partial\Omega} |U + \phi_n M| = \max_{\partial\Omega} |\phi_n| M = \max_{\partial\Omega} |\phi_n| \|\nabla_n^2 U\|_\infty. \qquad \square$$
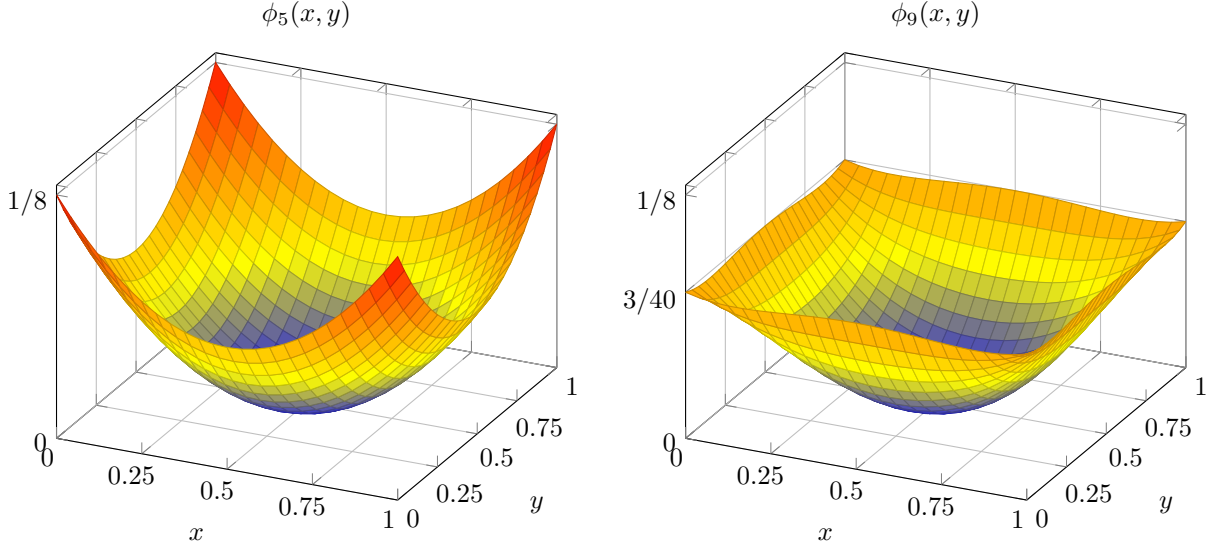
Figure 23: The functions $\phi_5(x,y)$ and $\phi_9(x,y)$ range from $\phi_5(\frac{1}{2},\frac{1}{2}) = \phi_9(\frac{1}{2},\frac{1}{2}) = 0$ at the center to $\phi_5(0,0) = 1/8$ and $\phi_9(0,0) = 3/40$ at the boundaries of $[0,1] \times [0,1]$.

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

$$\phi_5(x,y) = \frac{1}{4}\left(\left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2\right) \tag{47}$$

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

$$\phi_9(x,y) = \frac{1}{5}\left(\left(x - \frac{1}{2}\right)^4 + \left(y - \frac{1}{2}\right)^4\right) - \frac{6}{5}\left(x - \frac{1}{2}\right)^2\left(y - \frac{1}{2}\right)^2 + \frac{1}{4}\left(\left(x - \frac{1}{2}\right)^2 + \left(y - \frac{1}{2}\right)^2\right) \tag{48}$$

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

Theorem 1 (Five-point stencil stability on Poisson equation). If $\nabla^2 u = f$ and $\nabla_5^2 U = f$ in Ω, $u = U$ on $\partial\Omega$ and $D^4 u < \infty$ in Ω, then the 5-point stencil is consistent, stable and convergent of order h^2 with

$$\|u - U\|_\infty \le \frac{1}{8}\|\nabla_5^2(u - U)\|_\infty = Ch^2.$$

Proof. By lemma 2 with ϕ_5 from 47,

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

Taylor expand all terms in $\nabla_5^2 u$ around the center of the stencil to get

$$\begin{aligned}
\nabla_5^2 u &= \nabla^2 u + Ch^2 D^4 u \\
&\le f + Ch^2\left|D^4 u\right|_\infty \\
&\le f + Ch^2.
\end{aligned} \tag{49}$$

47

The second term is the remainder term, where by $D^m u$ we mean *some* combination $\sum_{n=0}^{m} c_n \partial_x^n \partial_y^{m-n} u$ of m-derivatives evaluated *somewhere* inside Ω. By $|D^m u|_\infty$, we mean the maximum value of any m-derivative $\max_{n=0}^{m} \left| \partial_x^n \partial_y^{m-n} u \right|$ in Ω. Note that terms with odd powers of h vanish due to the symmetry of the stencil. Finally, substitute $f = \nabla_5^2 U$ to conclude that

$$\|u - U\|_\infty \leq \frac{1}{8} \|\nabla_5^2 (u - U)\|_\infty \leq Ch^2 \left| D^4 u \right|_\infty \leq Ch^2. \qquad \square$$

Theorem 2 (Nine-point stencil stability on Poisson equation). If $\nabla^2 u = f$ and $\nabla_9^2 U = f + \frac{h^2}{12} \nabla_5^2 f$ in Ω, $u = U$ on $\partial \Omega$ and $D^4 u, D^6 u < \infty$ in Ω, then the 9-point stencil is consistent, stable and convergent of order h^4 with

$$\|u - U\|_\infty \leq \frac{3}{40} \|\nabla_9^2 (u - U)\|_\infty \leq Ch^4.$$

Proof. By lemma 2 with ϕ_9 from 48,

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

Taylor expand all terms in $\nabla_9^2 u$ around the center of the stencil to get

$$\nabla_9^2 u = \nabla^2 u + \frac{h^2}{12} \nabla^4 u + Ch^4 D^6 u$$

$$\leq f + \frac{h^2}{12} \nabla^2 f + Ch^4 \left| D^6 u \right|_\infty$$

$$\leq f + \frac{h^2}{12} \nabla^2 f + Ch^4.$$

From the Taylor expansion 49 with $u \leftrightarrow f$, we also know that

$$\nabla^2 f \leq \nabla_5^2 f + Ch^2 \left| D^4 f \right|_\infty$$

$$\leq \nabla_5^2 f + Ch^2 \left| D^6 u \right|_\infty$$

$$\leq \nabla_5^2 f + Ch^2.$$

Together, the two Taylor expansions imply

$$\nabla_9^2 u \leq f + \frac{h^2}{12} \nabla_5^2 f + Ch^4 \tag{50}$$

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

$$\|u - U\|_\infty \leq \frac{3}{40} \|\nabla_9^2 (u - U)\|_\infty \leq Ch^4. \qquad \square$$

(TODO: using u with f is also weird. using u with g would be more consistent with what we already have)

(TODO: improve biharmonic theorems, make simpler?)

Theorem 3 (Five-point stencil order and stability on Biharmonic equation). If $\nabla^2 u = g$, $\nabla_5^2 U = G$, $\nabla^2 g = f$, $\nabla_5^2 G = f$ in Ω and $u = U$ and $g = G$ on $\partial \Omega$ and derivatives up to 6-th order of u are bounded, then the 5-point stencil $(\nabla_5^2)^2$ is consistent, stable and convergent of order h^2 on the Biharmonic equation.

Proof. We *cannot* apply theorem 1 to $\nabla^2 u = g$ and $\nabla_5^2 U = G$ since $g \neq G$. But we *can* apply theorem 1 to $\nabla^2 u = g$ and $\nabla_5^2 U' = g$. This gives

$$\|u - U\|_\infty = \|(u - U') - (U - U')\|_\infty$$
$$\leq \|u - U'\|_\infty + \|U - U'\|_\infty$$

Now apply theorem 1 to the first term and lemma 2 with ϕ_5 from 47 to the second term to get

$$\|u - U\|_\infty \leq \frac{1}{8}\|\nabla_5^2(u - U')\|_\infty + \frac{1}{8}\|\nabla_5^2(U - U')\|_\infty$$
$$\leq Ch^2 + \frac{1}{8}\|G - g\|_\infty.$$

Finally, apply theorem 1 again to $\nabla^2 g = \nabla_5^2 G = f$ to show that also $\|g - G\|_\infty \leq Ch^2$, so $\|u - U\|_\infty \leq Ch^2$ and $(\nabla_5^2)^2$ is $\mathcal{O}(h^2)$.

To show consistency, note that theorem 1 already ensures consistency for the Poisson equation. But if $\nabla_5^2(u - U) \to 0$ as $h \to 0$, then also $\nabla_5^2 \nabla_5^2(u - U) \to 0$, so $(\nabla_5^2)^2$ is also consistent on the Biharmonic equation. By Lax' equivalence lemma, it is therefore also stable. $\qquad\square$

Theorem 4 (Nine-point stencil order and stability on Biharmonic equation)**.** If $\nabla^2 u = g$, $\nabla_9^2 U = (1 + \frac{h^2}{12}\nabla_5^2)G$, $\nabla^2 g = f$, $\nabla_9^2 G = (1 + \frac{h^2}{12}\nabla_5^2)f$ in Ω and $u = U$ and $g = G$ on $\partial\Omega$ and derivatives up to 8-th order of u are bounded, then the 9-point stencil $(\nabla_9^2)^2$ is consistent, stable and convergent of order h^4 on the Biharmonic equation.

Proof. We *cannot* apply theorem 2 to $\nabla^2 u = g$ and $\nabla_9^2 U = (1 + \frac{h^2}{12}\nabla_5^2)G$, since $g \neq G$. But we *can* apply theorem 2 to $\nabla^2 u = g$ and $\nabla_9^2 U' = (1 + \frac{h^2}{12}\nabla_5^2)g$. This gives

(use lemma 2!)

$$\|u - U\|_\infty = \|(u - U') - (U - U')\|_\infty$$
$$\leq \|u - U'\|_\infty + \|U - U'\|_\infty$$

Now apply theorem 2 to the first term and lemma 2 with ϕ_9 from 48 to the second term to get

$$\|u - U\|_\infty \leq \frac{3}{40}\|\nabla_9^2(u - U')\|_\infty + \frac{3}{40}\|\nabla_9^2(U - U')\|_\infty$$
$$\leq Ch^4 + \frac{3}{40}\|(G - g) + \frac{h^2}{12}\nabla_5^2(G - g)\|_\infty\|_\infty$$
$$\leq Ch^4 + \frac{3}{40}\|(G - g)\|_\infty + \frac{3}{40}\frac{h^2}{12}\|\nabla_5^2(G - g)\|_\infty$$

Finally, apply theorem 2 again on the two last terms to show that they are also $\mathcal{O}(h^4)$, so $\|u - U\|_\infty \leq Ch^4$ and $(\nabla_9^2)^2$ is $\mathcal{O}(h^2)$.

Consistency and stability is shown like in theorem 3. $\qquad\square$

$$\|u - U\|_\infty \leq \frac{1}{8}\|\nabla_5^2(u - U)\|_\infty$$
$$\leq \frac{1}{8}\|\nabla^2 u + Ch^2\|D^4 u\|_\infty - \nabla_5^2 U\|_\infty$$
$$\leq \frac{1}{8}\|g - G\|_\infty + \frac{1}{8}Ch^2\|D^4 u\|_\infty$$
$$\leq Ch^2\|D^4 u\|_\infty + Ch^2\|D^4 u\|_\infty$$
$$\leq Ch^2\|D^4 u\|_\infty.$$

7.3.1 Stability when solving the Biharmonic equation

Theorems 1 and 2 shows that the solutions to the Poisson equation using the five and nine point stencils are stable and of order 2 and 4 respectively. We will here show that the order of the solution to the system of Poisson equations in (41) will be of the same order.

We will begin with the five point stencil. One must show that

$$\|U - u\|_\infty \le Ch^2,$$

where u is the exact solution $\nabla^4 u = f$. Solving the Biharmonic equation as a system of Poisson equations consists of two steps 1) $\nabla^2 g = f$ and 2) $\nabla^2 u = g$. We find an approximation G to g, where by theroem 1

$$\|G - g\|_\infty \le Ch^2.$$

When solving step 2) we would ideally solve

$$\nabla_5^2 U_e = g,$$

in which case theorem 1 implies that

$$\|U_e - u\|_\infty \le Ch^2.$$

However we end up solving

$$\nabla_5^2 U = G = g + \gamma(x, y)h^2.$$

We have that

$$\nabla_5^2 (U - U_e) = \gamma(x, y)h^2, \tag{51}$$

By lemma 2

$$\|U - U_e\|_\infty \le \frac{1}{8}\|\nabla_5^2 (U - U_e)\|_\infty = \frac{1}{8}\|\gamma(x, y)h^2\|_\infty \le Ch^2.$$

Thus,

$$\begin{aligned}
\|U - u\|_\infty &= \|U - U_e + U_e - u\|_\infty \\
&\le \|U - U_e\|_\infty + \|U_e - u\|_\infty \\
&\le Ch^2.
\end{aligned}$$

7.4 The Fast Poisson Solver

Equation (44) can of course be solved directly with (sparse) matrix solvers like before, but there is a more efficient way. The *Fast Poisson Solver* exploits properties of the eigenvectors of K_5 and K_9 to compute the solution efficiently. In this section, we will explain how this works based on [14].

Just like in equation (44), suppose one is to solve the matrix equation

$$KU = h^2 F$$

for U and that one knows the eigenvectors $y_1, \ldots, y_n$ and corresponding eigenvalues $\lambda_1, \ldots, \lambda_n$ of K. If $K = K^T$ is invertible and symmetric, the eigenvectors are complete and orthogonal [6], so we may write

$$F = c_1 y_1 + c_2 y_2 + \cdots + c_n y_n \qquad \text{with} \qquad c_i = y_i \cdot F. \tag{52}$$

One may now easily verify by insertion that the solution is

$$U = \frac{1}{h^2}\left[\frac{c_1}{\lambda_1}y_1 + \frac{c_2}{\lambda_2}y_2 + \cdots + \frac{c_n}{\lambda_n}y_n\right]. \tag{53}$$

We want to solve equation (44), so let us find the eigenvalues and eigenvectors of K_5 and K_9.

Lemma 3 (Eigenvalues of Kroenecker product). Let $\alpha_1, \ldots, \alpha_m$ be eigenvalues of A with corresponding eigenvectors $x_1, \ldots, x_m$. Let $\beta_1, \ldots, \beta_n$ be eigenvalues of B with corresponding eigenvectors $y_1, \ldots, y_n$. Then $\alpha_m \beta_n$ are eigenvalues of $A \otimes B$ with corresponding eigenvectors $x_m \otimes y_n$. In other words,

$$A x_m = \alpha_m x_m \quad \text{and} \quad B y_n = \beta_n y_n \quad \implies \quad (A \otimes B)(x_m \otimes y_n) = (\alpha_m \beta_n)(x_m \otimes y_n).$$

Proof. See theorem 13.12 on page 141 in [3]. □

Lemma 4 (Eigenvalues of Kroenecker sum). Let $\alpha_1, \ldots, \alpha_m$ be eigenvalues of A with corresponding eigenvectors $x_1, \ldots, x_m$. Let $\beta_1, \ldots, \beta_n$ be eigenvalues of B with corresponding eigenvectors $y_1, \ldots, y_n$. Then $\alpha_m + \beta_n$ are eigenvalues of $A \oplus B$ with corresponding eigenvectors $x_m \otimes y_n$. In other words,

$$A x_m = \alpha_m x_m \quad \text{and} \quad B y_n = \beta_n y_n \quad \implies \quad (A \oplus B)(x_m \otimes y_n) = (\alpha_m + \beta_n)(x_m \otimes y_n).$$

Proof. See theorem 13.16 on page 143 in [3]. □

Lemma 5 (Eigenvalues of sum of matrices with equal eigenvectors). Let $\alpha_1, \ldots, \alpha_m$ be eigenvalues of A with corresponding eigenvectors $x_1, \ldots, x_m$. Let $\beta_1, \ldots, \beta_m$ be eigenvalues of B with the same corresponding eigenvectors. Then $\alpha_m + \beta_m$ are eigenvalues of $A + B$ with the same eigenvectors.

Proof. $(A + B)x_i = A x_i + B x_i = \alpha_i x_i + \beta_i x_i = (\alpha_i + \beta_i)x_i.$ □

Lemma 6. Let $A = \text{tridiag}(b, a, b)$ be a $N \times N$ TST matrix with a on the diagonal and b on the off-diagonal. Then the eigenvalues $\alpha_1, \ldots, \alpha_N$ and eigenvectors $x_1, \ldots, x_N$ of A are

$$\alpha_m = a + 2b \cos\left(\frac{m\pi}{N+1}\right) \quad \text{and} \quad x_m(k) = \sin\left(\frac{mk\pi}{N+1}\right). \tag{54}$$

Proof. See "The eigenvalues and vectors of a common tridiagonal matrix" on page 154 in [12]. □

With these lemmas, it is now straightforward to obtain the eigenvalues and eigenvectors of K_5 and K_9.

Theorem 5. The matrices K_5 and K_9 in equation (44) have common eigenvectors

$$y_{kl}(m, n) = \sin\left(\frac{mk\pi}{N+1}\right)\sin\left(\frac{nl\pi}{N+1}\right) \qquad (1 \leq k, l, m, n \leq M). \tag{55}$$

Proof. By lemma 6, J_5, J_9 and Σ have common eigenvectors given by equation (54). By lemmas 3 and 4, $J_5 \oplus J_5$, $J_9 \oplus J_9$ and $\Sigma \otimes \Sigma$ have common eigenvectors given by equation (55). By lemma 5, the sum in $(J_9 \oplus J_9) + (\Sigma \otimes \Sigma)$ does not change the eigenvectors. □

Note that it is only J_5, J_9 and Σ that are on the form $\text{tridiag}(b, a, b)$ and are suitable for lemma 6. Like we discussed in figure 12, the matrices K_5 and K_9 themselves are *not*.

Theorem 6. The eigenvalues of K_5 corresponding to the eigenvectors 55 are

$$\lambda_{kl} = -4 + 2\cos\left(\frac{k\pi}{N+1}\right) + 2\cos\left(\frac{l\pi}{N+1}\right). \tag{56}$$

Proof. Use lemma 6 on J_5 with appropriate a and b. Then use lemma 4 on $J_5 \oplus J_5$. □

Theorem 7. The eigenvalues of K_9 corresponding to the eigenvectors 55 are

$$\lambda_{kl} = -\frac{10}{3} + \frac{4}{3}\left(\cos\left(\frac{k\pi}{N+1}\right) + \cos\left(\frac{l\pi}{N+1}\right)\right) + \frac{4}{6}\cos\left(\frac{k\pi}{N+1}\right)\cos\left(\frac{l\pi}{N+1}\right). \tag{57}$$

Proof. Use lemma 6 on J_9 and Σ with appropriate a and b. Then use lemma 3 on $\Sigma \otimes \Sigma$ and lemma 4 on $J_9 \oplus J_9$. Finally, use lemma 5 on the sum in $(J_9 \oplus J_9) + (\Sigma \otimes \Sigma)$. □

(TODO: establish one index convention etc., explain the two indices on the eigenvalues)

It is now straightforward to compute U from 52 and 53. Inserting the eigenvectors 55, we get

$$c_{kl} = \sum_{m,n} F(m,n)y_{kl}(m,n) = \sum_{m,n} F(m,n)\sin\left(\frac{mk\pi}{N+1}\right)\sin\left(\frac{nl\pi}{N+1}\right)$$

and

$$U(m,n) = \sum_{k,l} \frac{c_{kl}}{\lambda_{kl}}y_{kl}(m,n) = \sum_{k,l} \frac{c_{kl}}{\lambda_{kl}}\sin\left(\frac{mk\pi}{N+1}\right)\sin\left(\frac{nl\pi}{N+1}\right)$$

Incidentally, this corresponds *perfectly* to the two-dimensional Discrete Sine Transform of type I (DST-I) and Inverse Discrete Sine Transform (IDST-I), for which very efficient implementations exist in for example [7]. [2] Instead of solving the systems 44, we can instead calculate

$$U = \frac{1}{h^2}\text{IDST}(\text{DST}(F_n)/\Lambda)$$

where Λ is the matrix with the eigenvalues 56 or 57 corresponding to the right stencil on the diagonal. This is the awaited and much more efficient Fast Poisson Solver. Note the strong similarity with the analytical solution 37 – in both cases the solution is expressed as a sum of discrete or continuous sine functions whose coefficients are given by the inner product between the discrete or contiuous source and basis functions.

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 $f = -\sin(m\pi x)\sin(n\pi y)$, $m = 3, n = 4$. (TODO: do we know this from section 7.1?) From section 7.1 we know that the solution to this manufactured problem is

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

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 24a. The relative error as a function of N is shown in figure 24b. 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.

---

[2]Note that [7] uses different normalization factors and summation indices from 55, but this is easy to handle. In addition, it only implements the one-dimensional transform, but this can simply be applied twice to give the two-dimensional transform

$$\sum_{m,n} F(m,n)\sin\left(\frac{mk\pi}{N+1}\right)\sin\left(\frac{nl\pi}{N+1}\right) = \sum_m \underbrace{\left(\underbrace{\sum_n F(m,n)\sin\left(\frac{nl\pi}{N+1}\right)}_{\text{1D transform with } m \text{ fixed}}\right)\sin\left(\frac{mk\pi}{N+1}\right)}_{\text{1D transform}}.$$
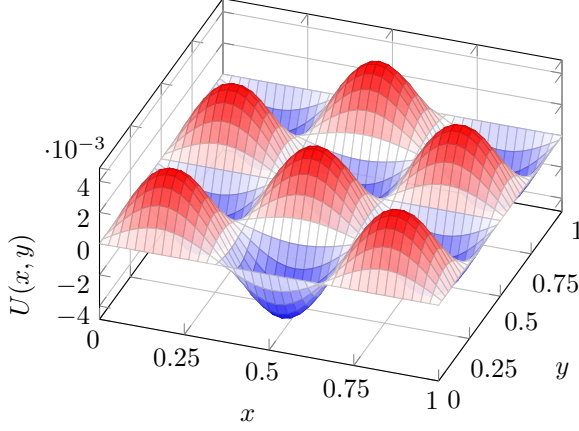
(a) The approximated solution $U(x, y)$.
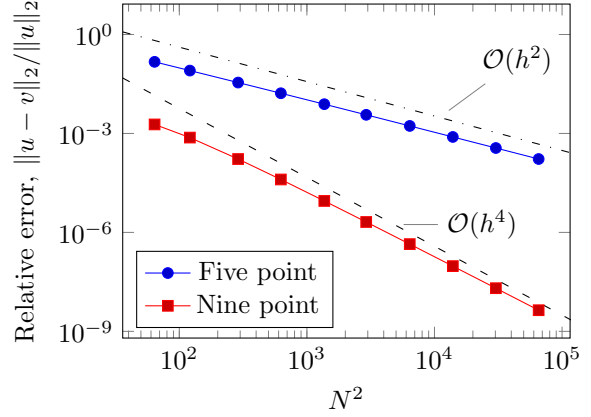
(b) Relative error. Also shown are Ch^2 and Ch^4, which are the expected convergence rates of the five and nine point stencil respectively. The constants are chosen such that the h^2 and h^4 line appears close to the stencil errors. First axis shows degrees of freedom N^2, where N is the number of internal discretization points in one direction.

Figure 24: Solving the Poisson equation on the manufactured problem $f = -\sin(m\pi x)\sin(n\pi y)$, $m = 3, n = 4$, with analytical solution $u(x, y) = \sin(m\pi x)\sin(n\pi y)/((n\pi)^2 + (m\pi)^2)$. Solved with equal number of discretization points in x and y direction. Subfigure a shows the analytical solution u, while b shows the relative error for the five and nine point stencil.

7.6   Solving the Biharmonic equation

We will now solve (36) 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.

According to equation (41) we split the equation into a system of poisson equations by introducing a new function g:

$$
\begin{aligned}
\nabla^2 g &= f, \\
\nabla^2 u &= g, \\
g = u &= 0, \quad \text{on } \partial\Omega.
\end{aligned}
\tag{58}
$$

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 25b. In figure 25c a convergence plot as a function of the degrees of freedom is shown, and the computation time is shown in figure 25d. (TODO: we could probably have solved for higher numbers... maybe change to "we solved for ...") We were able to solve the equation with a maximal discretization of $N = 3000$ in one direction, so about 1×10^7 grid points.

7.6.1 A possible improvement to the Biharmonic solver with nine point stencil

The fourth order nine point stencil may be written as

$$\nabla_9^2 U = \left(I + \frac{h^2}{12}\nabla_5^2\right)F. \tag{59}$$

Recall that the five point stencil on the right hand side emerged as a consequence of $\|(\nabla^2 - \nabla_5^2)u\|_\infty = \frac{h^2}{12}\nabla^2 f$ to order h^2. The laplace operator was later approximated using the five point stencil, after it was shown that this conserved the fourth order convergence, resulting in (59).
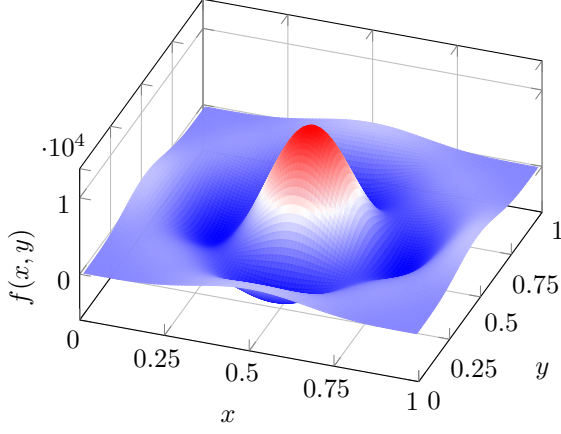
Return now to the Biharmonic equation. Upon solving the second Poisson equation in (58), the nine point stencil equation is

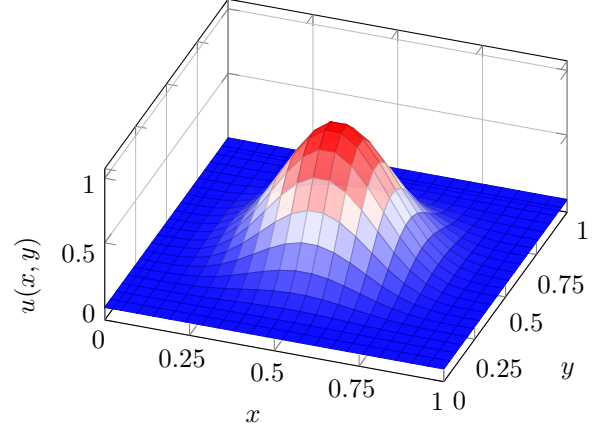$$\nabla_9^2 U = \left(I + \frac{h^2}{12}\nabla_5^2\right)G. \tag{60}$$

However, we already have an exact expression for $\nabla^2 g$ – the inhomogenity f! Thus, we may solve the equation

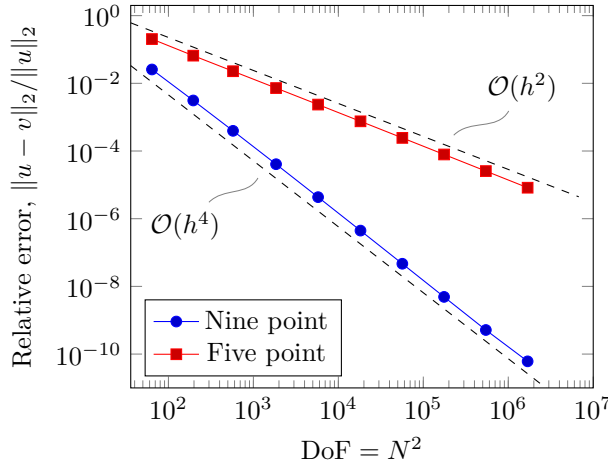$$\nabla_9^2 U = \left(G + \frac{h^2}{12}F\right), \tag{61}$$

and have thus eluded the computation of $\nabla_5^2 G$.

(a) The inhomogenity $f(x, y)$ used when solving the Biharmonic equation. The function is not zero on the entire border of the domain.

(b) The numerical solution $U(x, y)$ to the Biharmonic equation, using the nine point stencil with discretization $N = 20$ in each direction. TODO: does it work to have it here?

(c) The relative error of the numerical solution. TODO: Could add five point for comparison?
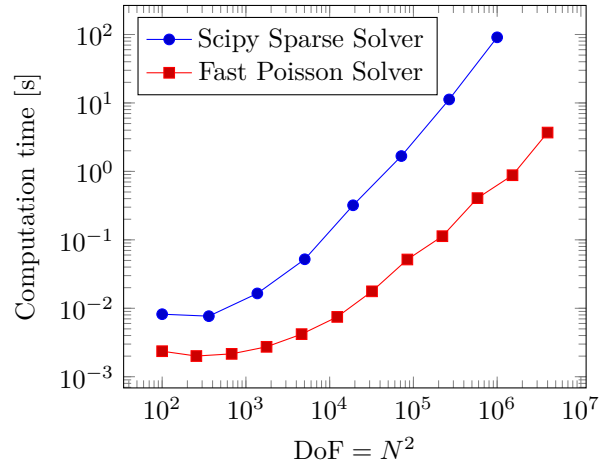
(d) Computation time. Computation times for two solvers shown, the Scipy Sparse Solver[10] and our own Fast Poisson Solver.

Figure 25: Error and computation time solving the clamped Biharmonic equation on the manufactured problem with analytical solution $(\sin \pi x \sin \pi y)^4 \, e^{-(x-0.5)^2-(y-0.5)^2}$. Figure c shows the relative error using the nine point stencil, while figure d shows the computation time. Both are plottet with degrees of freedom on the first axis, the total number of internal points N^2.

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 +
    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.* 30 Apr. 2018. URL: `http://www.math.ntnu.no/emner/TMA4212/2020v/notes/master2.pdf` (visted 25/04/2021).

[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.* SIAM: Society for Industrial and Applied Mathematics, 29 Dec. 2004. ISBN: 978-0-89871-576-7.

[4]   R. J. LeVeque. *Numerical Methods for Conservation Laws.* Second edition. Springer Basel AG, 1992. ISBN: 978-3-7643-2723-1.

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

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

[7]   *scipy.fft.dst — SciPy v1.6.2 Reference Guide.* URL: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.fft.dst.html`.

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

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

[10]   *scipy.sparse.linalg.spsolve — SciPy v1.6.3 Reference Guide.* URL: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.linalg.spsolve.html`.

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

[12]   G. D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods.* Third edition. Oxford University Press, 1985. ISBN: 0-19-859641-3.

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

[14]   Gilbert Strang. *Computational science and engineering.* 2. print. Wellesley-Cambridge Press, 2012. ISBN: 978-0-9614088-1-7.