Numerical methods for Differential equations Report 2

Malte Wegener (4672194)

October 29, 2019

1 Inhomogeneous poisson equation

In the this section, the following Boundary Value Problem (BVP) will be solved using a Finite volume method.

$$
\begin{aligned}
-\nabla \cdot (k \nabla u) &= f, \quad (x,y) \in \Omega = (0,8) \times (0,4) \\
u(x,y) &= 0, \quad (x,y) \in \partial\Omega \\
k(x,y) &= 1 + 4x + 6y, \quad (x,y) \in \bar{\Omega} \\
f(x,y) &= e^{\alpha(x-1)^2+\alpha(y-1)^2} + e^{\alpha(x-3)^2+\alpha(y-1)^2} \\
&\quad + e^{\alpha(x-1)^2+\alpha(y-1)^2} + e^{\alpha(x-3)^2+\alpha(y-3)^2} \\
&\quad + e^{\alpha(x-5)^2+\alpha(y-3)^2} + e^{\alpha(x-7)^2+\alpha(y-3)^2} \\
&\quad \text{with } \alpha = -5, \quad (x,y) \in \bar{\Omega}
\end{aligned} \tag{1}
$$

With $\bar{\Omega} = [0,8] \times [0,4]$. In order to discretize the PDE, the following stencil on a doubly uniform grid with step h is considered.

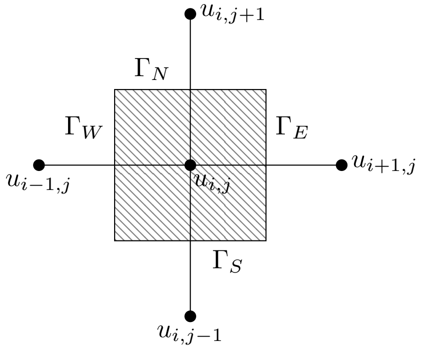


Figure 1: Stencil used for the discretization

In order to discretize the PDE with a finite volume method (FVM), it has to be integrated over a control volume $V_{i,j}$, which is represented by the hatched Area, whose boundary is $\Gamma_{i,j}$.

$$
\iint_{V_{i,j}} [-\nabla \cdot (k \nabla u)] dV = \iint_{V_{i,j}} f dV \tag{2}
$$

Using the Divergence Theorem

$$
\oint_{\Gamma_{i,j}} (-k \nabla u \cdot \mathbf{n}) d\Gamma = \iint_{V_{i,j}} f dV \tag{3}
$$

Which can be rewritten as

$$
\oint_{\Gamma_{ijj}} \left(-k \frac{\partial u}{\partial \mathbf{n}}\right) d\Gamma = \iint_{V_{i,j}} f dV \tag{4}
$$

By splitting the boundary and using a midpoint approximation for the right hand side

$$
\int_{\Gamma_\mathrm{W}} \left(k \frac{\partial u}{\partial x}\right) dy + \int_{\Gamma_\mathrm{E}} \left(-k \frac{\partial u}{\partial x}\right) dy + \int_{\Gamma_\mathrm{S}} \left(k \frac{\partial u}{\partial y}\right) dx + \int_{\Gamma_\mathrm{N}} \left(-k \frac{\partial u}{\partial y}\right) dx = f_{i,j} h^2 \tag{5}
$$

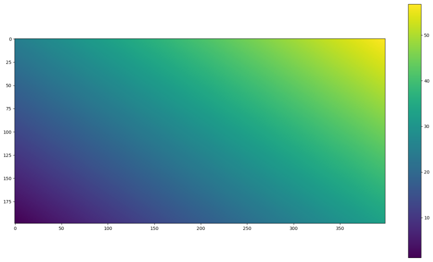
By approximating $\frac{\partial u}{\partial x}$ and $\frac{\partial u}{\partial y}$ with a central difference formula and using a midpoint approximation for the Integral, a discrete approximation can be found.

$$k_{i-1/2,j}\left(u_{i,j}-u_{i-1,j}\right)-k_{i+1/2,j}\left(u_{i+1,j}-u_{i,j}\right)+k_{i,j-1/2}\left(u_{i,j}-u_{i,j-1}\right)-k_{i,j+1/2}\left(u_{i,j+1}-u_{i,j}\right)=h^2 f_{i,j} \tag{6}$$

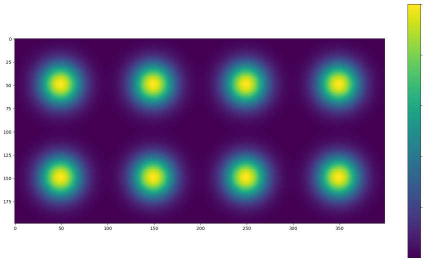
Which can be rearranged to

$$\begin{aligned} f_{ij} = &-\frac{k_{i-1/2,j}}{h^2}u_{i-1,j}-\frac{k_{i,j-1/2}}{h^2}u_{i,j-1} \\ &+\left(\frac{k_{i-1/2,j}}{h^2}+\frac{k_{i,j-1/2}}{h^2}+\frac{k_{i+1/2,j}}{h^2}+\frac{k_{i,j+1/2}}{h^2}\right)u_{i,j} \\ &-\frac{k_{i+1/2,j}}{h^2}u_{i+1,j}-\frac{k_{i,j+1/2}}{h^2}u_{i,j+1} \end{aligned} \tag{7}$$

When assembling a linear system on a lexicographic grid from this equation, a matrix $A \in \mathbb{R}^{(N_x-1)*(N_y-1)\times(N_x-1)*(N_y-1)}$, where $N_x = 8/h$ and $N_y = 4/h$. This matrix has 5 non-zero diagonals, located on the main diagonal, right next to the main diagonal and offset by $\pm(N_x - 1)$. All these diagonals are full, except for the first off diagonals, which have a 0 on every N_x^{th} element. These diagonals can be made, by calculating k for a flattened shifted grid, which can be trimmed to include the proper coefficients in the matrix, which is implemented in the code.



(a) Coefficient K on the domain



(b) Source on the domain

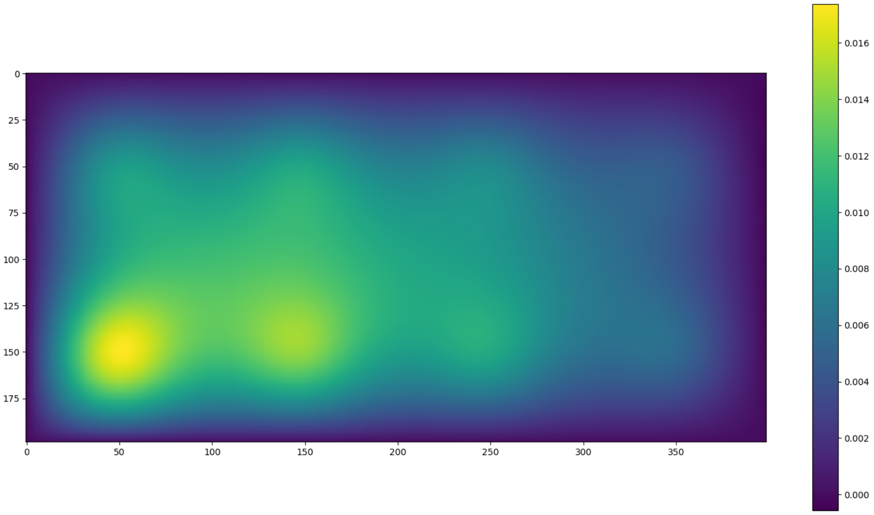


Figure 3: Solution to the BVP on the domain for $h = 0.02$

The solution seems reasonable, as the source function can be interpreted as a steady influx, that is diffused on the domain. As the rate of diffusion is influenced by k it is expected that the upper right corner will experience stronger diffusion than then the lower left corner, which is the case in the solution. Furthermore, zero Dirichlet boundary conditions are applied, which are also clearly visible in the solution.

2 Time evolution of the heat equation

In this section the unsteady heat equation will be solved. This problem can be formulated as the following BVP, with an initial condition.

$$\begin{gathered}\frac{\partial u}{\partial t} - \Delta u = 0, \quad (x,y) \in (0,4) \times (0.4), \quad t \in [0, 0.15] \\ u(x,y,t) = 0, \quad (x,y) \in \partial\Omega \\ u(x,y,0) = e^{\alpha(x-2)^2 + \alpha(y-2)^2} \\ \text{with } \alpha = -5, (x,y) \in \overline{\Omega}\end{gathered} \tag{8}$$

In which $\overline{\Omega} = [0,4] \times [0,4]$. This equation can be discretized on a doubly uniform grid with step h. The negative Laplacian in its discrete form can be expressed as A, such that the equation becomes

$$\frac{\partial \mathbf{u}}{\partial t} \approx -A\mathbf{u} \tag{9}$$

In order to solve this unsteady equation, a time-stepping scheme has to be employed. As a simple explicit scheme, Forward Euler (FE) is considered. In this scheme, $\mathbf{u}^{k+1}$ is calculated as

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta t * \frac{\partial \mathbf{u}^k}{\partial t} \tag{10}$$

By substituting Equation 9

$$\mathbf{u}^{k+1} = \mathbf{u}^k - \Delta t * A\mathbf{u}^k \tag{11}$$

$$\mathbf{u}^{k+1} = (I - \Delta t * A)\mathbf{u}^k \tag{12}$$

As an implicit method, Backward Euler (BE) is used. $\mathbf{u}^{k+1}$ in BE is calculated as follows.

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta t * \frac{\partial \mathbf{u}^{k+1}}{\partial t} \tag{13}$$

By substituting Equation 9

$$\mathbf{u}^{k+1} = \mathbf{u}^k - \Delta t * A\mathbf{u}^{k+1} \tag{14}$$

$$\mathbf{u}^{k+1} + \Delta t * A\mathbf{u}^{k+1} = \mathbf{u}^k \tag{15}$$

$$(I + \Delta t * A)\mathbf{u}^{k+1} = \mathbf{u}^k \tag{16}$$

In order to solve for $\mathbf{u}^{k+1}$ a linear system has to be solved, thus this is an implicit method.

In order to compare these 2 methods, the time evolution is simulate on a grid with $h = 0.08$ and $\delta t = 0.015$.

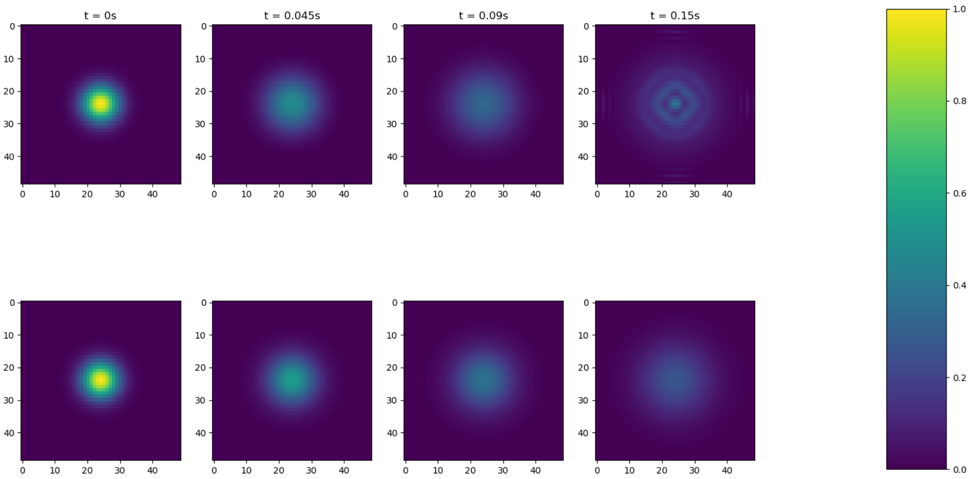
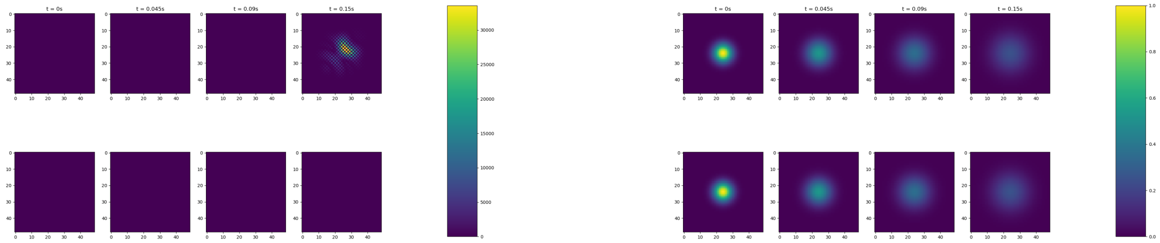


Figure 4: Time evolution of the Heat equation. First Row presents the Forward Euler, second row is computed with Backwards Euler. All plots are normalized to the same colorscale.

It can be seen that FE diffuses faster than BE. Furthermore, FE becomes unstable at $t = 0.15$ as the solution does not settle to reach 0 at all points in the domain, which should be expected from the maximum principle. In order to get stable results from the simulation is run with $\Delta t = 0.005, 0.003$ and 0.0015. For $\Delta t = 0.005 and 0.003$ the solution with FE blows up in the given time, however for $\Delta t = 0.0015$ the solution of FE matches the implicit solution very well.



(a) Unstable solution for $\Delta t = 0.005$, with all plots normalized to the same scale

(b) Stable solution for $\Delta t = 0.0015$, with all plots normalized to the same scale

Figure 5: Comparison of a stable and unstable solution for the BVP

In order to use appropriate timesteps for FE, a stability limit for Δt has to be derived. The stability of a time stepping scheme can be analyzed by looking at the eigenvalues of the stepping scheme. For this λ is considered to be an eigenvalue of the negative Laplacian and σ is considered to be an eigenvalue of the time stepping scheme.

$$C = I - \Delta t A \tag{17}$$

For some eigenvector $\mathbf{x}_m$ it holds that

$$C\mathbf{x}_m = (I - \Delta t A)\mathbf{x}_m \tag{18}$$

$$C\mathbf{x}_m = I\mathbf{x}_m - \Delta t A\mathbf{x}_m \tag{19}$$

$$\sigma_m\mathbf{x}_m = \mathbf{x}_m - \Delta t\lambda_m\mathbf{x}_m \tag{20}$$

$$\sigma_m\mathbf{x}_m = (1 - \Delta t\lambda_m)\mathbf{x}_m \tag{21}$$

$$\tag{22}$$

Thus the relationship for the eigenvalues of the negative Laplacian (λ) and the eigenvalues of the time stepping scheme (σ) are related by $\sigma_m = 1 - \Delta t\lambda_m$. For FE to be stable $|\sigma_m| \leq 1$ for all σ_m. Thus

$$\lambda_{k_x,k_y} = \frac{4}{h^2}\left[\sin^2\left(\frac{\pi k_x}{2N_x}\right) + \sin^2\left(\frac{\pi k_y}{2N_y}\right)\right] \tag{23}$$

Thus $lambda_m$ is bounded.

$$0 \leq \lambda_m \leq \frac{4}{h^2} \tag{24}$$

$$|1 - \Delta t\lambda_m| \leq 1 \tag{25}$$

$$\Delta t\lambda_m \leq 2 \tag{26}$$

$$\Delta t\frac{4}{h^2} \leq 2 \tag{27}$$

$$\Delta t \leq \frac{h^2}{2} \tag{28}$$

Calculating the stability limit for $h = 0.08$ it can be seen that $\Delta t \leq 0.0016$. As previously only one time step was stable, which explains the blow-up of the solution for $\Delta t = 0.005$ and 0.003.

By running the simulation for different values of h, with a stable time step for FE and a time step of 0.015 for BE, the speed of an implicit vs explicit method can be compared.

h	FE	BE
0.08	0.163 s	0.107 s
0.04	0.968 s	0.650 s
0.02	7.986 s	4.412 s

Table 1: Time to solve the BVP for different step-sizes with FE and BE

It can be seen that solving the system implicitly is always faster for this range of h than the explicit solution, as the required time step decreases rapidly such that solving a linear system is less computational effort than performing more steps.

3 Time evolution of the wave equation

In order to examine the time evolution of the wave equation, the following BVP is considered.

$$\begin{aligned}
\frac{\partial^2 u}{\partial t^2} &= c^2\Delta u + f, \quad (x,y) \in (0,4)\times(0,12), \quad t \in [0,8] \\
u(x,y,t) &= 0, \quad (x,y) \in \partial\Omega, \quad t \in [0,8] \\
u(x,y,0) &= 0, \quad (x,y) \in \Omega \\
\left.\frac{\partial u}{\partial t}\right|_{t=0} &= 0, \quad (x,y) \in \Omega \\
f(x,y,t) &= \sin(2\pi\nu t)e^{\alpha(x-2)^2+\alpha(y-2)^2} \quad (x,y) \in \overline{\Omega}, \quad t \in [0,8] \\
\text{with } \alpha = -50, \nu = 4
\end{aligned} \tag{29}$$

With $\overline{\Omega} = [0,4] \times [0,12]$. The discretization of the Laplacian Operator is performed using a central finite difference method on a doubly uniform grid with step h.

$$L = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & 0 & \dots & 0 \\ 1 & -2 & 1 & & \vdots \\ 0 & 1 & -2 & 1 & \\ \vdots & & & \ddots & \\ 0 & \dots & 0 & 1 & -2 \end{bmatrix} \tag{30}$$

From this the space discrete form can be determined to be

$$\frac{\partial^2 \mathbf{u}}{\partial t^2} \approx c^2 L\mathbf{u} + f \tag{31}$$

In order to use time stepping on the second derivative in time, a finite difference time discretization (FDTD) is used.

$$\frac{\partial^2 \mathbf{u}^k}{\partial t^2} = \frac{\mathbf{u}^{k-1} - 2\mathbf{u}^k + \mathbf{u}^{k+1}}{\Delta t^2} + \mathcal{O}(\Delta t^2) \tag{32}$$

$$c^2 L\mathbf{u}^k + f \approx \frac{\mathbf{u}^{k-1} - 2\mathbf{u}^k + \mathbf{u}^{k+1}}{\Delta t^2} \tag{33}$$

$$\mathbf{u}^{k+1} \approx c^2 \Delta t^2 L\mathbf{u}^k + \Delta t^2 f + 2\mathbf{u}^k - \mathbf{u}^{k-1} \tag{34}$$

$$\mathbf{u}^{k+1} \approx (c^2 \Delta t^2 L + 2I)\mathbf{u}^k - \mathbf{u}^{k-1} + \Delta t^2 f \tag{35}$$

In order to approximate the initial conditions with sufficient accuracy, $\mathbf{u}^1$ is expanded as a Taylor series around $\mathbf{u}^0$.

$$\mathbf{u}^1 = \mathbf{u}^0 + \Delta t \frac{\partial \mathbf{u}^0}{\partial t} + \frac{\Delta t^2}{2}\frac{\partial^2 \mathbf{u}^0}{\partial t^2} + \mathcal{O}(\Delta t^3) \tag{36}$$

As it is given that $\frac{\partial \mathbf{u}^0}{\partial t} = 0$ and that $\frac{\partial^2 \mathbf{u}^0}{\partial t^2} \approx c^2 L\mathbf{u}^0 + f$ The first step can be calculated as.

$$\mathbf{u}^1 = \mathbf{u}^0 + \frac{\Delta t^2 c^2}{2} L\mathbf{u}^0 + \frac{\Delta t^2}{2} f + \mathcal{O}(\Delta t^3) \tag{37}$$

In order to guarantee a stable time stepping, the simulation is run at 99% of the corresponding Courant–Friedrichs–Lewy (CFL) number. This number is a important dimensionless number, which relates the speed of the wave to the corresponding spatial and time steps used.

$$C = \frac{c\Delta t}{\sqrt{2}h} \tag{38}$$

Running the simulation for $c = 1$ on a grid with $h = 0.02$ yields the follwing solution.

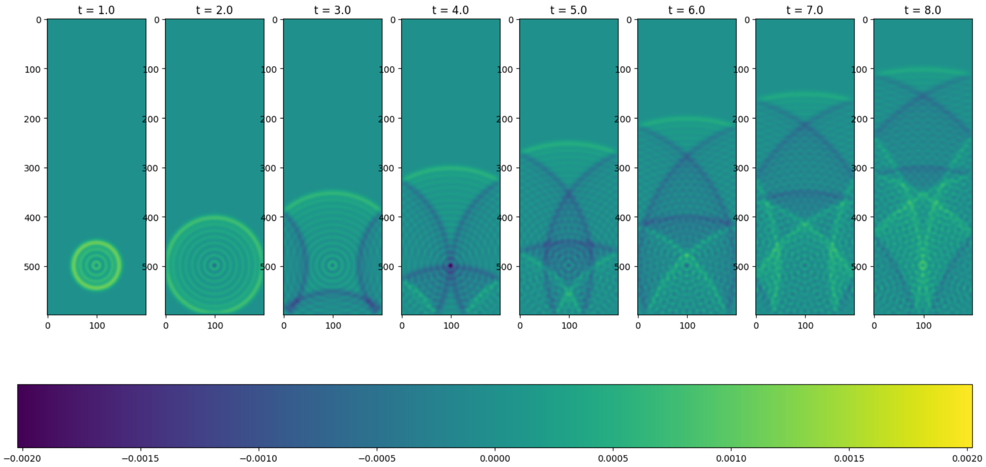


Figure 6: Solution of the wave equation on a uniform grid with step 0.02 and a wavespeed of 1

It can clearly be seen that the wavefront travels at a speed of 1 unit per secodn, which is expected as $c = 1$. Furthermore, reflection can be observed at the boundaries, which however flip the orientation of the wave, whcich would not be expected in a physical system. This behaviour comes from the use of homogeneous dirichlet conditions on the boundary of the domain. When performing a similar the simulation with $c = 2$ it can be seen that the wave travels twice as fast.

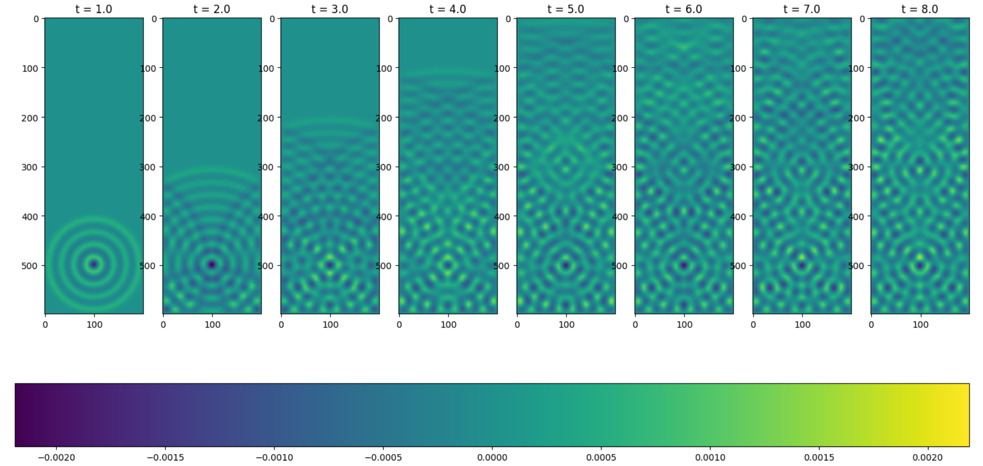
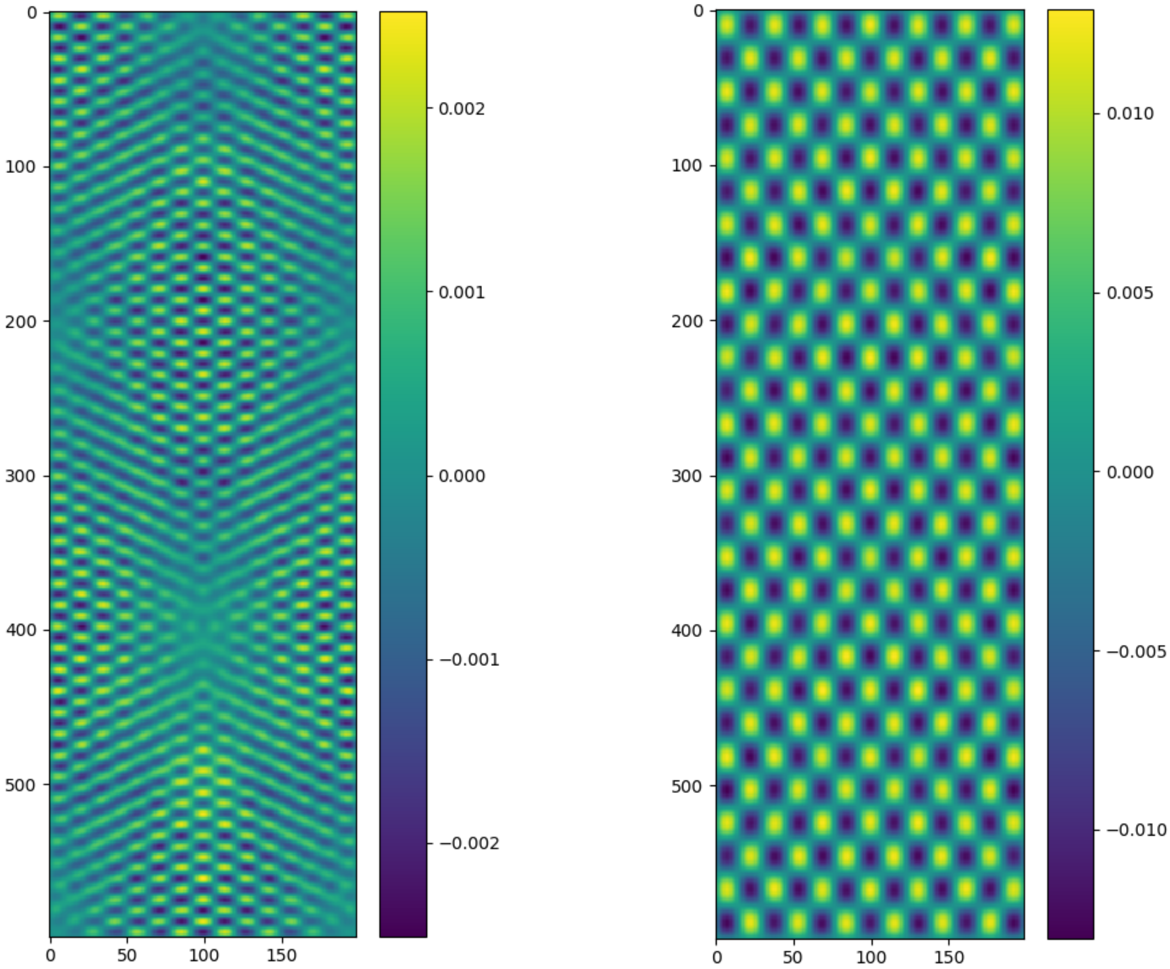


Figure 7: Solution of the wave equation on a uniform grid with step 0.02 and a wavespeed of 2

In contrast to the previous simulation, standing waves can be observed. Furthermore shifted replicas of the source are present in later solutions. Changing the speed of the wave slightly destroys these inferences. it can therefore be assumed that these patterns come from standing waves developing between the boundaries. This can be expected as the frequency of the source function permits an integer number of wavelengths to travel to each boundary. This however is also the case for $c = 1$ and no such pattern is visible at 8 seconds. Thus both simulations are run for 300 and 600 seconds to reach a quasi steady state and the results are compared. The simulation for $c = 1$ is run twice as long, to allow the waves to travel the same distance as for $c = 2$.



(a) Quasi-steady solution for c=1 at t=600s

(b) Quasi-steady solution for c=2 at t=300s

Figure 8: Comparison of the quasi-steady solutions

From these 2 figures it can be seen that the patterns emerging in page 7 and page 7, are indeed standing waves.

4 Numerical Solutions of the Fisher equation

This section discusses numerical solutions of the unsteady fisher equation on a 2-dimensional, for this the following BVP with initial conditions is considered.

$$\begin{aligned} \frac{\partial u}{\partial t} &= \Delta u + ku(1-u), \quad (x,y) \in (0,16) \times (0,8), \quad t \in (0,40] \\ u(x,y,t) &= 0, \quad (x,y) \in \partial\Omega, \quad t \in (0,40] \\ u(x,y,0) &= e^{-2(x-1.5)^2-2(y-1.5)^2} \quad (x,y) \in \overline{\Omega} \\ k(x,y) &= \begin{cases} \alpha, & (x,y) \in \Omega_1 \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{39}$$

With $\alpha = 1$, $\overline{\Omega} = [0,16] \times [0,8]$ and $\Omega_1 = \bigcup_{i=1}^{5} R_i$.

$$\begin{aligned} R_1 &= [1,2] \times [1,2] \\ R_2 &= [1,3] \times [3,5] \\ R_3 &= [4,7] \times [4,7] \\ R_4 &= [9,12] \times [4,6] \\ R_5 &= [13,15] \times [1,3] \end{aligned} \tag{40}$$

For the spatial discretisation, a FDM Method is used, in which the Laplacian can be discretized as L, Equation 30. With this, the discrete equation becomes, where $\mathbf{k}$ is the lexicographic vector of coefficients k, and $\circ$ denotes the hadamard product of 2 vectors.

$$\frac{\partial \mathbf{u}}{\partial t} = L\mathbf{u} + \mathbf{k} \circ \mathbf{u} \circ (1 - \mathbf{u}) \tag{41}$$

In order to solve this semi discrete problem, a time stepping scheme has to be employed. As an explicit scheme, Forward Euler can be used. With this scheme, the fully discrete equation becomes

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t * (L\mathbf{u}^n + \mathbf{k} \circ \mathbf{u}^n \circ (1 - \mathbf{u}^n)) \tag{42}$$

In order to achieve a stable solution, Δt is set to be at 99% of the stability margin for the linear part of the equation. $\Delta t = 0.99 * \frac{h^2}{4}$. With this the solution on the domain can be computed for different grid spacings.

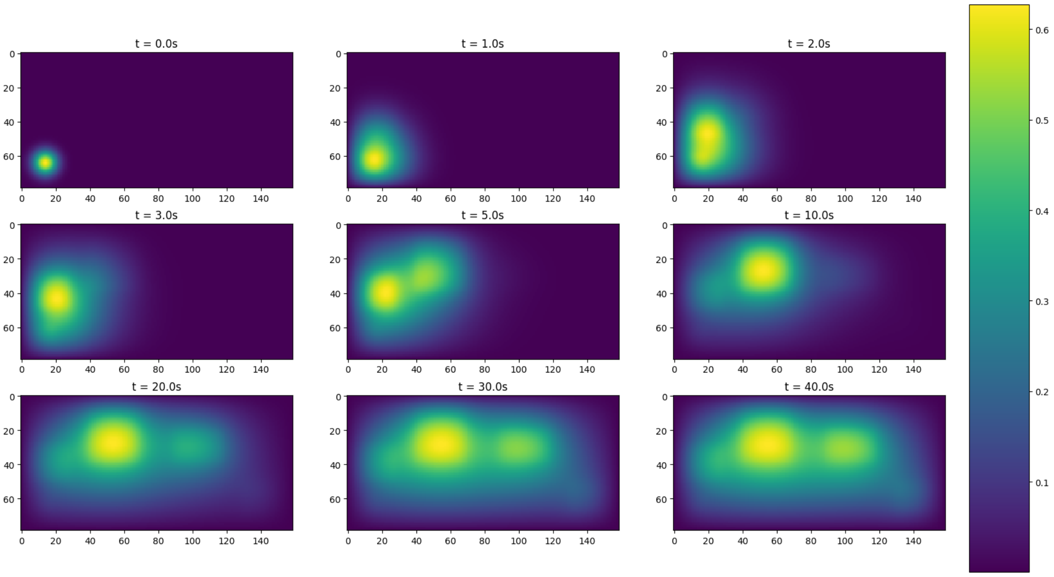


Figure 9: Solution of the Fisher equation on a grid with $h = 0.04$, with FE time stepping

As an implicit time stepping scheme, backwards Euler is used.

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t * f(\mathbf{u}^{n+1}) \tag{43}$$

$$f(\mathbf{u}^{n+1}) = L\mathbf{u}^{n+1} + \mathbf{k} \circ \mathbf{u}^{n+1} \circ \left(1 - \mathbf{u}^{n+1}\right) \tag{44}$$

As this equation is neither explicit nor linear, a non-linear solver has to be implemented. For this Newton-Raphson is chosen. At every inner iteration i of this method, the solution is updated to.

$$\mathbf{u}_{i+1}^{n+1} = \mathbf{u}_i^{n+1} - \left[I - \Delta t J\left(\mathbf{u}_i^{n+1}\right)\right]^{-1} \left[\mathbf{u}_i^{n+1} - \mathbf{u}^n - \Delta t \mathbf{f}\left(\mathbf{u}_i^{n+1}\right)\right] \tag{45}$$

The jacobian at every iteration is calculated as

$$J\left(\mathbf{u}_i^{n+1}\right) = L + I - 2\,\mathrm{diag}(\mathbf{u}_i^{n+1}) \tag{46}$$

in which I is the identity matrix of appropriate size and $\mathrm{diag}(\mathbf{x})$ is the diagonal matrix with $\mathbf{x}$ on its diagonal. The update at each inner iteration $\mathbf{v}_i = \left[I - \Delta t J\left(\mathbf{u}_i^{n+1}\right)\right]^{-1} \left[\mathbf{u}_i^{n+1} - \mathbf{u}^n - \Delta t \mathbf{f}\left(\mathbf{u}_i^{n+1}\right)\right]$, includes the inverse of a matrix, which is computationally very expensive to solve, thus the following sparse linear system is solved instead.

$$\left[I - \Delta t J\left(\mathbf{u}_i^{n+1}\right)\right] \mathbf{v}_i = \left[\mathbf{u}_i^{n+1} - \mathbf{u}^n - \Delta t \mathbf{f}\left(\mathbf{u}_i^{n+1}\right)\right] \tag{47}$$

And the solution is updated to

$$\mathbf{u}_{i+1}^{n+1} = \mathbf{u}_i^{n+1} - \mathbf{v}_i \tag{48}$$

The inner iterations are stopped when $\|\mathbf{v}_i\|^2 < \epsilon$. For this problem ϵ was set to be 10^-3, and $\Delta t = 0.4$, was found to solve the equation the fastest. From the convergence plot, the quadratic convergence of Newton-Raphson can clearly be identified on the residuals. Furthermore it can be seen that all outer iterations take at least 4, but at most 9 inner iterations to converge. This fast convergence in connection with the big time steps that can be used, can make it a viable alternative to FE.

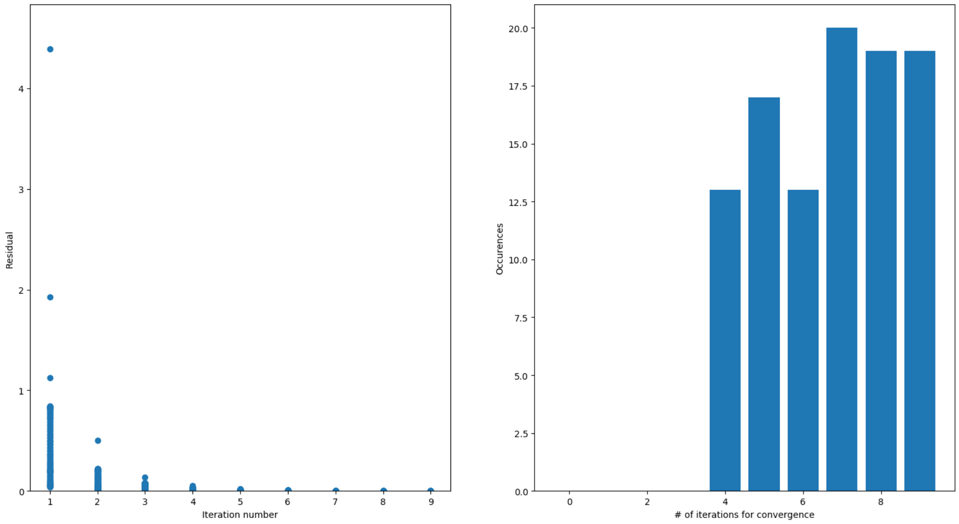


Figure 10: Convergence history of BE-NR method.

Both implicit and explicit computation times were recorded for different grid spacings with both methods.

h	FE	BE-NR
0.1	5s	40s
0.08	11s	155s
0.04	198s	976s

Table 2: Comparison of computing times for different grid spacings and time stepping schemes. Run on an Intel Core i5-3570 CPU 4 Cores @ 3.40GHz

It can be seen that Backwards Euler always takes longer than the explicit solution. This most likely originates from the necessity of solving a sparse linear system for every inner iteration of the Newton Raphson Algorithm in comparison to the computationally cheaper matrix multiplication performed in FE.

As a secondary nonlinear solver for BE a Picard method is investigated, in which the update in the inner iterations is calculated with

$$\mathbf{u}_{i+1}^{n+1} = \Delta t \mathbf{f}^{n+1}\left(\mathbf{u}_i^{n+1}\right) + \mathbf{u}^n \tag{49}$$

$$f(\mathbf{u}^{n+1}) = L\mathbf{u}^{n+1} + \mathbf{k} \circ \mathbf{u}^{n+1} \circ \left(1 - \mathbf{u}^{n+1}\right) \tag{50}$$

The inner iterations of Picards method are stopped when $\left\|\mathbf{u}_i^{n+1} - \Delta t \mathbf{f}^{n+1}\left(\mathbf{u}_i^{n+1}\right) - \mathbf{u}^n\right\| < \epsilon$, which is chosen to be 10^{-3}. As a first guess for Δt, the same step as for FE is used. However this time step, makes the solver unstable and the solution explodes rapidly. Thus Δt is halved, which leads to a stable solution. By analyzing the convergence plot, it can be seen that convergence is very fast and most outer iterations converge in 1 or 2 iterations, however Δt is twice as small as for FE, which always takes only one step. Furthermore BE-Picard requires significantly more matrix multiplications than FE, and as such takes about 4 times as long to compute. From these results, it can be concluded that BE-Picard should not be used to solve this equation.

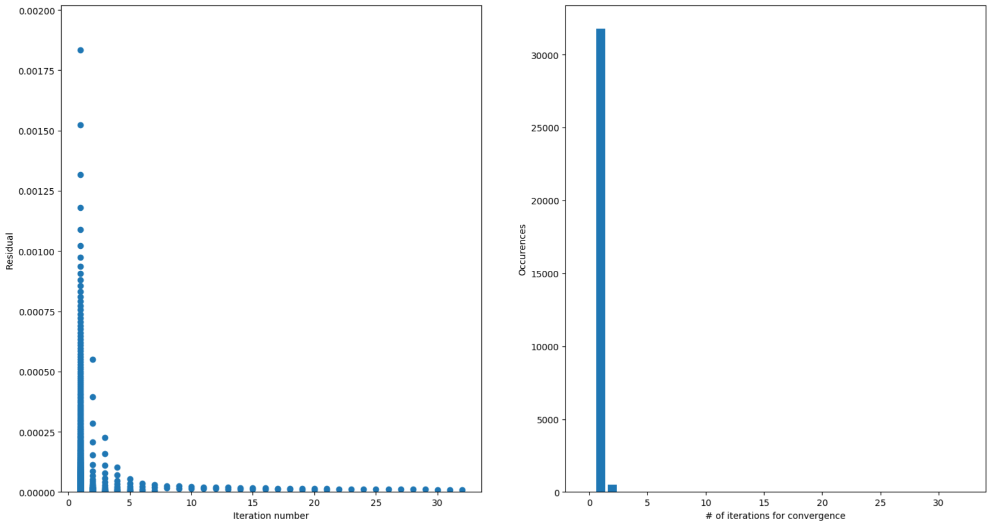
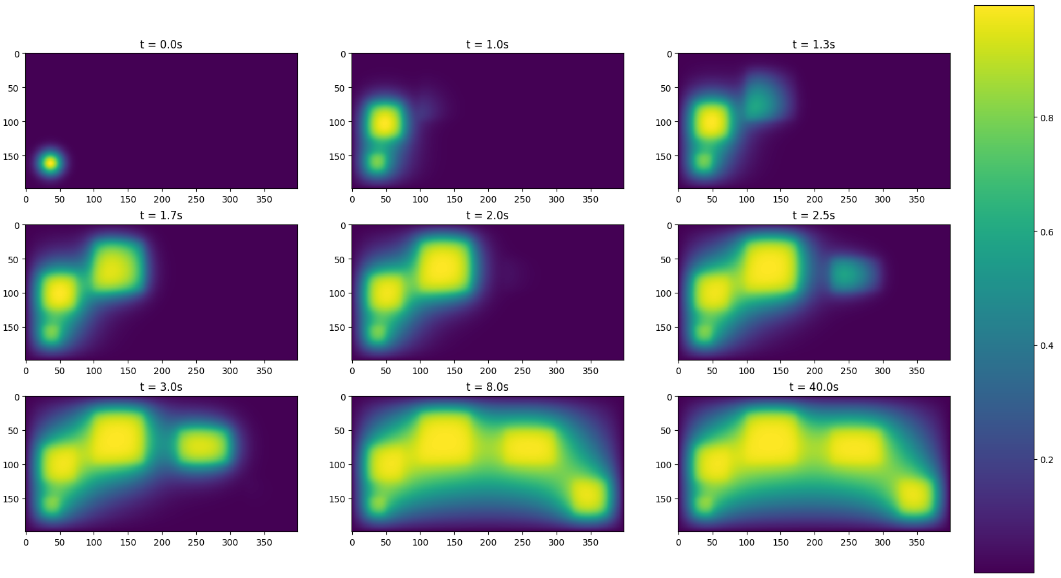


Figure 11: Convergence history of BE-Picard method.

In order to explore the solution dynamics of the Fisher equation α is set to 10. It can be seen that the solution accumulates on Ω_1 a lot faster and the edges of the accumulation are sharper. Furthermore, smaller regions like $[1, 2] \times [1, 2]$ experience a higher accumulation, than with a smaller α. When observing the later time evolution, it is apparent that a quasi steady state is reached faster with a higher α.


t = 0.0s

t = 1.0s

t = 1.3s

t = 1.7s

t = 2.0s

t = 2.5s

t = 3.0s

t = 8.0s

t = 40.0s

Figure 12: Solution of the Fisher equation for $\alpha = 10$ and $h = 0.04$

5 Convergence of GMRES

The spsolve used in all problems until now is classified as a direct solver, which means that the solution of the system is calculated exactly in a single step, for which the most commonly a LU-Decomposition or an Incomplete-LU is used. Another class of linear solvers are iterative ones, an example for such a solver is the Generalized minimal residual method, which hinges on the subject of expressing the solution inside a Krylov subspace $\mathcal{K}_r(A, b)$ of $\mathbb{R}^n$. In order to investigate the convergence of the GMRES algorithm, the following BVP is considered.

$$\begin{aligned} -\Delta u - \gamma u &= f, \quad (x, y) \in \Omega = (0, 10) \times (0, 10) \\ u(x, y) &= 0, \quad (x, y) \in \partial\Omega \\ f(x, y) &= e^{-10(x-5)^2 - 10(y-5)^2} \quad (x, y) \in \Omega \end{aligned} \tag{51}$$

This BVP is discretized using FDM on a doubly uniform grid. A such the equation in its discrete form can be written as

$$(L - \gamma I)\,\mathbf{u} = \mathbf{f} \tag{52}$$

in which L is the discretization of the negative Laplacian and I is the Identity matrix of appropriate size. With this, the BVP can be solved for $\gamma = -40, 0$ and 40, with spsolve and GMRES. For GMRES, the maximum number of iterations is set to 5000, furthermore restarting is disabled and the tolerance for convergence is set to 10^{-12}.

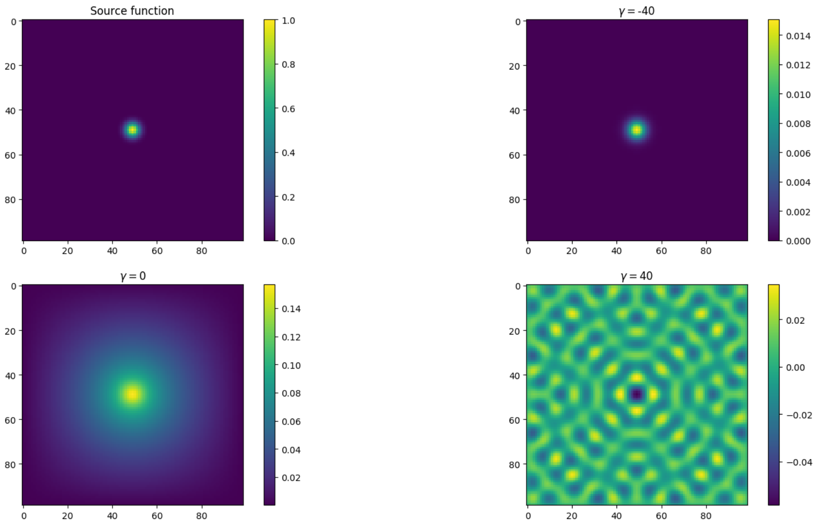


Figure 13: Solution of the BVP on the domain for $h = 0.1$, solved by GMRES

When analyzing the convergence history of GMRES for different γ it can be seen that convergence for $\gamma = -40$ is very fast, while for the other ones, the residual nearly stagnates for the first few iterations. However after an initial stagnation for $\gamma = 0$, it converges with the approximately same rate as the convergence of $\gamma = -40$. In contrast to that the convergence for $\gamma = 40$ plateaus frequently during the convergence after iteration 450, while the rate of convergence is considerably lower than the other two's.

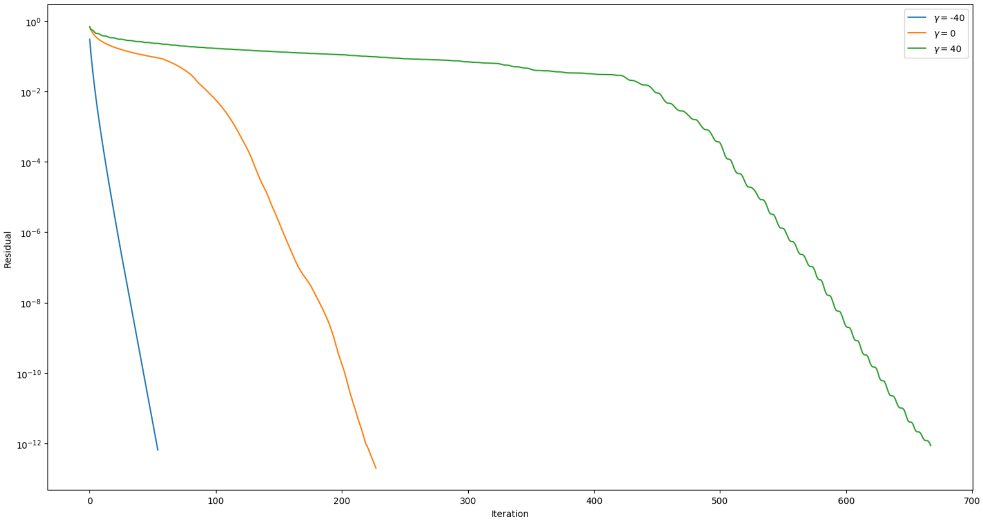


Figure 14: Convergence history of GMRES for different calues of γ

In order to understand this convergence, the eigenvalues of the discrete equation have to be analyzed. The eigenvalues and eigenvectors of the negative Laplacian are known to be λ_m and $\mathbf{x}_m$.

$$L\mathbf{x}_m = \lambda_m \mathbf{x}_m \tag{53}$$

$$\gamma I\mathbf{x}_m = \gamma \mathbf{x}_m \tag{54}$$

$$L\mathbf{x}_m - \gamma I\mathbf{x}_m = \lambda_m \mathbf{x}_m - \gamma \mathbf{x}_m \tag{55}$$

$$\left(L - \gamma I\right)\mathbf{x}_m = \left(\lambda_m - \gamma\right)\mathbf{x}_m \tag{56}$$

Thus the eigenvalues of the discrete BVP (λ') are found to be

$$\lambda'_m = \lambda_m - \gamma \tag{57}$$

The eigenvalues of the negative Laplacian are known to be real and are bounded between 0 and the $max(\lambda_m)$. For $\gamma = 40$ there are eigenvalues on both the positive and negative real axis, while for $\gamma = -40$ all eigenvalues are strictly positive, this suggests that GMRES performs best if the discretization matrix is positive definite or negative definite, while it still works reasonably well for semi positive/negative definite matrices, matrices that are neither however take longer to solve with GMRES.