# TMA4212 Numerical Solution of Differential Equations by Difference Methods Project 1

Amandus Omholt Nygaard, Sanne Jamila Razmara Olsen, Yawar Mahmood

February 2023

The purpose of this report is to study and solve the heat distribution in anisotropic and isotropic materials in 2D using finite difference methods in regular and irregular domains. We will be using the model equation:

$$-\nabla \cdot (\kappa \nabla T) = f$$

where T is the temperature distribution, $\kappa$ is the heat conductivity matrix

$$\kappa = \begin{bmatrix} a+1 & r \\ r & r^2 \end{bmatrix}$$

and f is an internal heat source. We define two heat flow directions, $\vec{d_1} = (1, 0)$ and $\vec{d_2} = (1, r)$ where $r \in \mathbb{R}$. In this case:

$$\nabla \cdot (\kappa \nabla T) = a\partial_x^2 u + (\vec{d_2} \cdot \nabla)^2 u$$

Let $\Omega = [0, 1] \times [0, 2]$ with Dirichlet boundary conditions $u = g$ on the boundary $\partial\Omega$.

First let $r = 2$ and the step sizes $h = \frac{1}{M}$ and $k = 2h$ in the $x$ and $y$ directions. Define the set of interior grid points $G = \{p = (x_p, y_p) = (x_i, y_j) = (ih, jk) : i, j = 1, \ldots, M-1\}$, $\partial G$ the set of boundary grid points and $\bar{G} = G \cup \partial G$. We begin by discretizing the problem using second order central differences in $\vec{d_1}$ and $\vec{d_2}$. This gives the difference approximations:

$$\partial_x^2 u_{i,j} \to \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} = \frac{1}{h^2}\delta_x^2 u_{i,j} \qquad \text{and}$$
$$(\vec{d_2} \cdot \nabla)^2 u_{i,j} \to \frac{u_{i+1,j+1} - 2u_{i,j} + u_{i-1,j-1}}{h^2} = \frac{1}{h^2}\delta_{\vec{d_2}}^2 u_{i,j}$$

This gives the scheme:

$$\frac{1}{h^2}(2(a+1)u_{i,j} - au_{i+1,j} - au_{i-1,j} - u_{i+1,j+1} - u_{i-1,j-1}) = f_{i,j},$$

which is solved as $A\vec{U} = h^2\vec{f} + \vec{b}$, where $\vec{U}$ is the vector containing the values on the interior nodes, $A$ contains the factors in the scheme above, $\vec{f}$ contains the function values of the RHS on the interior nodes and $\vec{b}$ contains all needed boundary values.

The scheme was tested on two test solutions, $g(x,y) = \cos(xy)$ and $g(x,y) = e^{x^2-y^2}$, plots of these are in extra plot appendix A in Figure 5 and 6.

Want to now show $L^\infty$ convergence of the scheme presented above. This is done by doing a error analysis, where use the discrete maximum principle to show $L^\infty$-stability and then show that the scheme is consistent.
Since $2(a+1) > 0$, $a > 0$ and $1 > 0$ the scheme has positive coefficients, and thus is monotone and satisfies the Dicrete maximum principle.

The stencil can be determined by looking at the indexes of $u$, and is given in the appendix 8.

To show $L^\infty$-stability, let the operator $\mathcal{L}_h$ be such that:

$$-\mathcal{L}_h V_{i,j} = \frac{1}{h^2}(2(a+1)V_{i,j} - aV_{i+1,j} - aV_{i-1,j} - V_{i+1,j+1} - V_{i-1,j-1})$$

and let $V_{i,j} = V_p$ solve $-\mathcal{L}_h V_p = f_p$ with RHS $f$ and $Vp = 0$ for $p \in \partial G$. Using the operator on our comparison function $\phi = \frac{1}{2}x(1-x)$ gives $-\mathcal{L}_h\phi = a+1$. Define $W_p = V_p - \|f\|_{L^\infty}\phi_p$. Using $-\mathcal{L}_h$ operator on $W_p$ yields:

$$-L_h W_p = -L_h V_p - \|f\|_{L^\infty}(-L_h\phi) = f_p - \|f\|_{L^\infty}(1+a) \leq 0$$

Using the discrete max principle:

$$\max_{p\in\bar{G}} W_p \leq \max_{p\in\partial G}\{W_p, 0\} = 0,$$

where we use that $-\phi \leq 0$ on $\Omega$ and $V_p = 0$ for $p \in \partial G$ in the last equality. Further

$$\max_{p\in\bar{G}} V_p \leq \|f\|_{L^\infty} \max_{p\in\bar{G}} \phi_p \leq \frac{1}{8}\|f\|_{L^\infty}$$

This also holds for $(V, f) \to (-V, -f)$, thus $\max_{p\in\bar{G}}|V_p| \leq \frac{1}{8}\|f\|_{L^\infty}$. Using linear interpolation of $V_p$, which is an order 2 interpolation, gives $I_h V_p$ and $\|I_h V_p\|_{L^\infty} = \max_{p\in\bar{G}}|V_p|$, thus

$$\|I_h V_p\|_{L^\infty} \leq \frac{1}{8}\|f\|_{L^\infty},$$

and therefore the scheme is $L^\infty$ stable. Letting $V_p = e_p = u_p - U_p$, where $u_p$ is the exact solution and $U_p$ is the numerical solution at point $p$, because of Dirichlet conditions $e_p = 0$ for $p \in \partial G$. Using $-\mathcal{L}_h$ on $e_p$ yields,

$$\begin{aligned} -\mathcal{L}_h e_p &= -\mathcal{L}_h u_p - (-\mathcal{L}_h U_p) = -\mathcal{L}u_p - \tau_p - f_p \\ &= f_p - \tau_p - f_p = -\tau_p \end{aligned}$$

From the $L^\infty$ stability we know that $\|I_h e_p\|_{L^\infty} \leq \frac{1}{8}\|\tau\|_{L^\infty}$. Need to then find an upper-bound for the truncation error $\tau$. It is known that $\partial_x^2 u_p - \frac{1}{h^2}\delta_x^2 u_p = -\frac{1}{12}h^2\partial_x^4 u(\eta_x, y_p)$ for some $\eta_x$ near $x_p$. To find $\tau_{\vec{d}_2} = (\vec{d}_2\cdot\nabla)^2 u_p - \frac{1}{h^2}\delta_{\vec{d}_2}^2 u(x,y)$ we Taylor expand the terms in $\frac{1}{h^2}\delta_{\vec{d}_2}^2 u_p$ around $(x,y)$, this gives

$$
\begin{aligned}
u(x \pm h, y \pm rh) &= u \pm h(u_x + ru_y) + \frac{1}{2}h^2(u_{xx} + 2ru_{xy} + r^2u_{yy}) \\
&\pm \frac{1}{6}h^3(\partial_x^3 u + 3r\partial_x^2\partial_y + 3r^2\partial_x\partial_y^2 u + r^3\partial_y^3 u) \\
&+ \frac{1}{24}h^4(\partial_x^4 u + 4r\partial_x^3\partial_y u + 6r^2\partial_x^2\partial_y^2 u + 4r^3\partial_x\partial_y^2 u + r^4\partial_y^4 u)(\xi_x^{\pm}, \xi_y^{\pm}) \\
&= u \pm h(\vec{d_2} \cdot \nabla)u + \frac{1}{2}h^2(\vec{d_2} \cdot \nabla)^2 u \pm \frac{1}{6}h^3(\vec{d_2} \cdot \nabla)^3 u + \frac{1}{24}h^4(\vec{d_2} \cdot \nabla)^4 u(\xi_x^{\pm}, \xi_y^{\pm}).
\end{aligned}
$$

Using the Taylor expansion and the fact that there exists $\xi_x, \xi_y$ such that $(\vec{d_2}\cdot\nabla)^4 u(\xi_x^+, \xi_y^+) + (\vec{d_2} \cdot \nabla)^4 u(\xi_x^-, \xi_y^-) = 2(\vec{d_2} \cdot \nabla)^4 u(\xi_x, \xi_y)$ we get that $\tau_{\vec{d}_2} = -\frac{1}{12}h^2(\vec{d_2} \cdot \nabla)^4 u(\xi_x, \xi_y)$. The total truncation error then becomes

$$
\tau = -\frac{1}{12}h^2\left(a\partial_x^4 u(\eta_x, y_p) + (\vec{d_2} \cdot \nabla)^4 u(\xi_x, \xi_y)\right)
$$

Thus,

$$
\begin{aligned}
\|I_h e_p\|_{L^\infty} &\leq \frac{1}{8}\|\tau\|_{L^\infty} \leq \frac{1}{96}h^2\left(a\left\|\partial_x^4 u\right\|_{L^\infty} + \left\|(\vec{d_2} \cdot \nabla)^4 u\right\|_{L^\infty}\right) \\
&\leq \frac{1}{96}h^2\Big((a+1)\left\|\partial_x^4 u\right\|_{L^\infty} + 4r\left\|\partial_x^3\partial_y u\right\|_{L^\infty} + 6r^2\left\|\partial_x^2\partial_y^2 u\right\|_{L^\infty} \\
&\qquad + 4r^3\left\|\partial_x\partial_y^3 u\right\|_{L^\infty} + r^4\left\|\partial_y^4 u\right\|_{L^\infty}\Big)
\end{aligned}
$$

If $u$ is smooth there exists a constant $M$ such that $\left\|\partial_x^i\partial_y^{4-i}u\right\|_{L^\infty} \leq M$ for $i = 1, 2, 3, 4$. Using this and setting $r = 2$ gives an error bound for our discritisation of the form

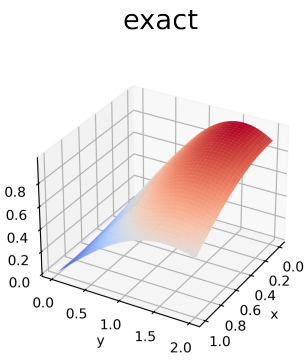
$$
\|I_h e_p\|_{L^\infty} \leq \frac{1}{96}(81 + a)Mh^2 = O(h^2)
$$

From our error bound we see that $e_p = O(h^2)$. The method is therefore convergent, and is 2nd order in space. The rate of convergence for smooth solutions is guaranteed to be of second order, as the derivatives of the function are bounded from above.

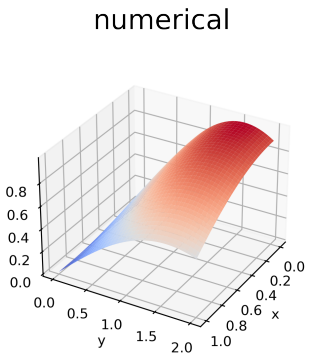
We expect to see a quadratic convergence numerically, as the analytically calculated error is O($h^2$). Quadratic convergence would imply a linear log-log plot of the error with slope equal to 2, as a function of the spacial step h.

The convergence rate is checked by using the test solution $g(x, y) = \cos x \sin y$. In the numerical analysis, the value of $r$ and $a$, is set constant to $r = 2$ and $a = 2$. The code was run for 5 different numbers of grid points, $M = 8, 16, 32, 64, 128$, hence five different step sizes, as the step size is defined as $h = \frac{1}{M}$.

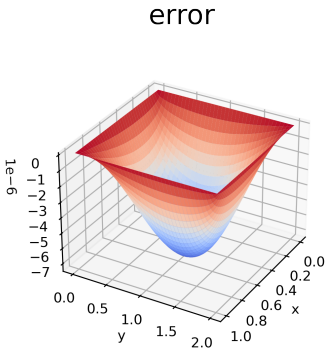
The convergence plot is linear, with a rate of increase equal to 2. Hence quadratic convergence. This is clear from the 1d.



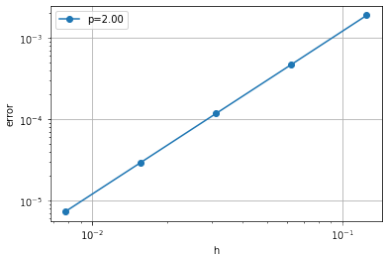

(a) Exact solution




(b) Numerical solution




(c) Error




(d) log-log convergence

Figure 1: Testing the scheme on the solution $g(x, y) = \cos(x)\sin(y)$. (a) plot of exact solution, (b) plot of numerical solution from scheme, (c) plot of error between exact and numerical solution, (d) log-log convergence plot of numerical solution. From (d) we see that $p = 2$, thus the scheme's order of convergence is 2.

Further, we introduce a new grid where $h = \frac{1}{M}$ and $k = |r|h$, where r is irrational. Since $r$ is irrational, the grid will no longer be a exact multiple of $h$, and the grid will not align perfectly with the upper boundary $y = 2$. This means that the solution will not be computed in the boundary, which can lead to inaccuracies of the interior nodes near the boundary.

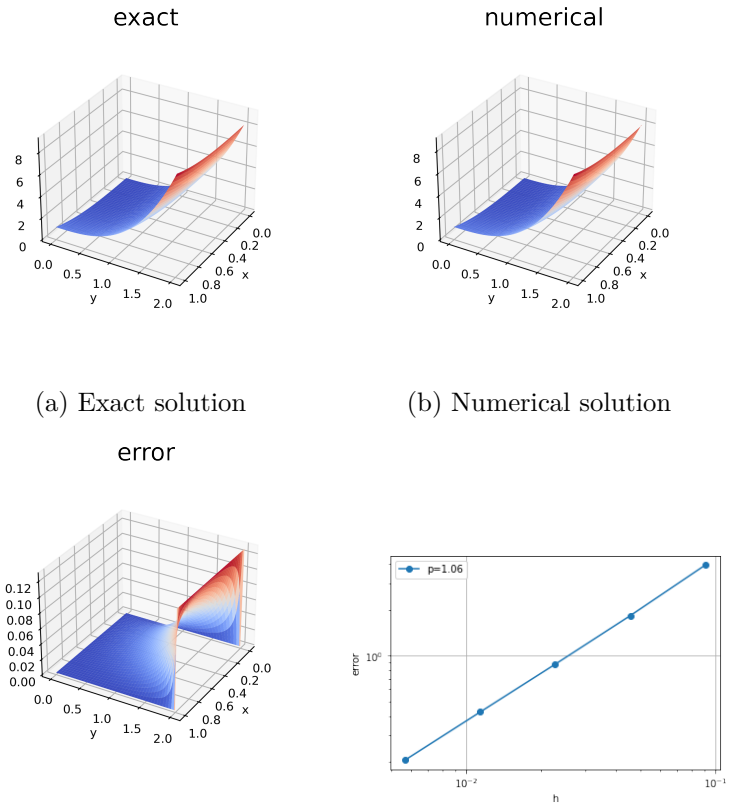
This problem can be solved by fattening the boundary. This is done by adding additional row of grid points above the upper boundary. As the grid points in the fattened boundary are not defined, we have to approximate the values at these grid points. The values are approximated by projecting the fattened boundary node onto the upper boundary $y = 2$ and using the value at the projecting point. Using this approach, we can use the same scheme as before.

It is expected, that using this method for solving the problem, will lead to a greater error at the boundary than elsewhere in the solution. This is evident in the error plot shown in 2c, alongside the plot the numerical solution. The max-norm of the error came

out at $1.28 \cdot 10^{-1}$.

The convergence rate is checked again by using the test solution $g(x, y) = \cos x \sin y$. In the numerical analysis, the value of r and a, is set to $r = \pi$ and $a = 2$. Variations in the step size was done by varying the number of grid points 5 times. The code was run for $M = 11, 22, 44, 88, 176$ grid points, and hence five different step sizes.

The log-log plot of the error-stepsize graph came out to be linear, with a slope of one. The method is convergence of first order.




(a) Exact solution

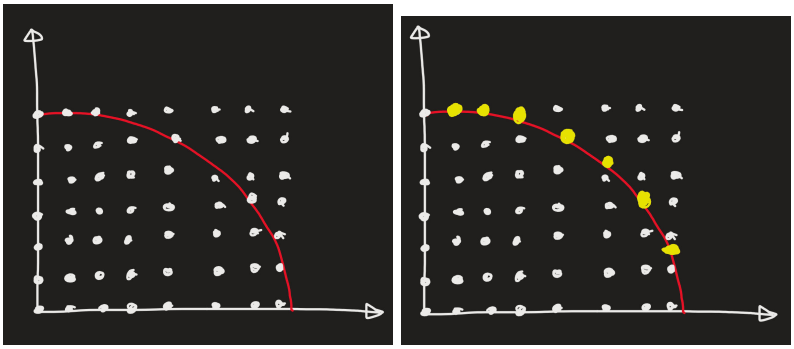
(b) Numerical solution

(c) Error

(d) log-log convergence

Figure 2: Testing the scheme on the solution $g(x, y) = x^3 + y^3$. (a) plot of exact solution, (b) plot of numerical solution from scheme, (c) plot of error between exact and numerical solution, (d) log-log convergence plot of numerical solution. From (d) we see that $p \approx 1$, thus the scheme's order of convergence is 1.

We turn our attention to an irregular domain $\Omega$, which is enclosed in the first quadrant by the parabola $y = 1 - x^2$, and the two axis.



(a) Rectangular grid, fattening boundary

(b) Irregular grid, modifying the boundary

Figure 3: (a) The grid using fattening the boundary, (b) The grid using modifying the boundary

We are now solving the isotropic heat equation with $\kappa = I$ with Dirichlet boundary conditions

$$\nabla \cdot (\kappa \nabla T) = \Delta T$$

By discretizing both second derivatives, the scheme now becomes:

$$\frac{1}{h^2}(4u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1}) = f_{i,j}$$

The problem is solved in two ways, (i) fattening the boundary and (ii) modifying the discretization near the boundary.

The numerical test is done using $M = 150$ grid points, the function $g(x, y) = -\cos(3x) + e^{2y}$ is used as a test solution, the boundary condition is then given by g and the f becomes:

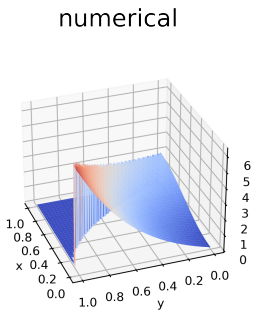
$$f(x, y) = -9\cos(3x) - 4e^{2y}.$$

The method of fattening the boundary (i), introduces additional points outside the boundary. This is the same as earlier, but the difference now is that the domain is not rectangular. The introduction of additional points is done in such a way that the grid becomes rectangular. This is shown in 3a

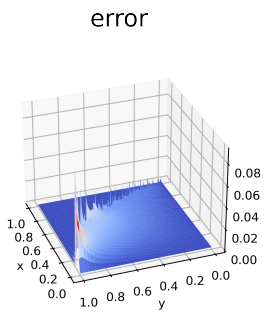
The grid points immediately outside the boundary are the points which are taken in the fattened boundary. The points further away are set to zero.

The value in the additional points is determined by projecting the additional points down on the boundary, by minimizing the distance onto the boundary, and giving them the corresponding value on the boundary.

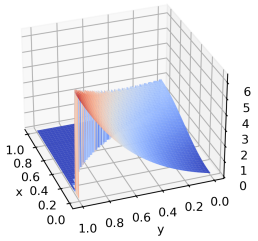
Modifying the boundary (ii), does not introduce additional points that lay outside of the domain, but rather introduces points on the boundary of the domain. These points follow the grid lines, but are not equidistant from the interior points on the grid, as illustrated by the yellow points in 3b. As a result of choosing points that does not align with the underlying grid, is that the scheme must be modified on the boundary. This is due to irregular step sizes. An advantage of this method over fattening the boundary, is that we do not need to approximate the value at the boundary points, as the boundary is given. A disadvantage with this approach, is that the grid is no longer regular.



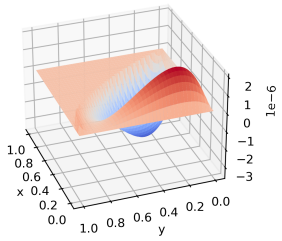

(a) Numerical solution fattening




(b) Error fattening




(c) Numerical solution modified




(d) Error modified solution

Figure 4: Testing the scheme on the solution $g(x) = -\cos(3x) + e^{2y}$ using two different approaches. (a) plot of numerical solution using fattening the boundary, (b) plot of error between exact and numerical solution calculated using fattening the boundary, (c) plot of numerical solution using modifying the boundary, (d) plot of error between exact and numerical solution calculated using modifying the boundary.

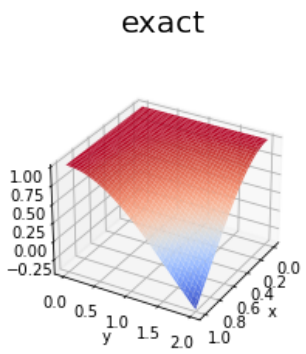
One would expect the error to be greater near the boundary, as the scheme remains unchanged inside the domain. This is evident from the error plots of figure 4b and 4d. Modifying the boundary approach has a error max norm of $3.51 \cdot 10^{-6}$, which is much smaller than the error max norm of obtained while fattening the boundary, which is $7.11 \cdot 10^{-2}$.

It is a fairly expected result that the error when modifying the boundary is smaller than that when fattening the boundary, as the exact value at the boundary is used. That it is so much smaller is a bit unexpected.

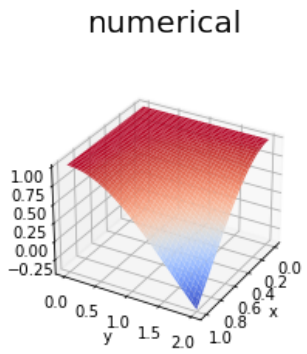
Usually, fattening the boundary is the easiest to implement, since one does not have to change the scheme used, as the grid remains regular. In the case of the domain studied in this report, both methods are relatively straight forward to implement.

# Appendices

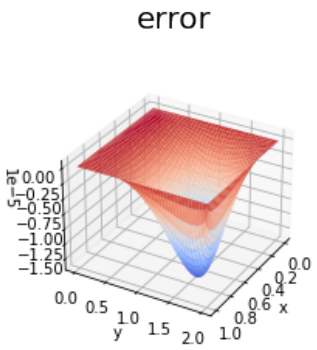
## A Extra plots




(a) Exact solution

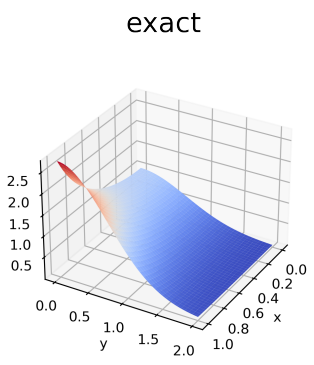



(b) Numerical solution

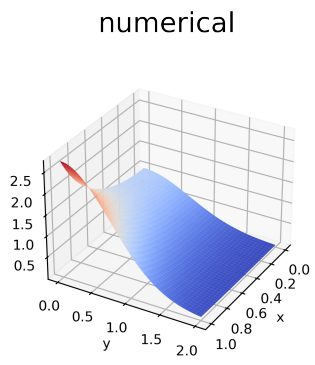



(c) Error plot

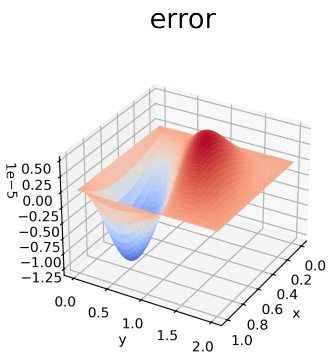
Figure 5: Testing the numerical solution with exact solution $g(x, y) = \cos(xy)$




(a) Exact solution




(b) Numerical solution




(c) Error plot

Figure 6: Testing the numerical solution with exact solution $g(x, y) = e^{x^2-y^2}$

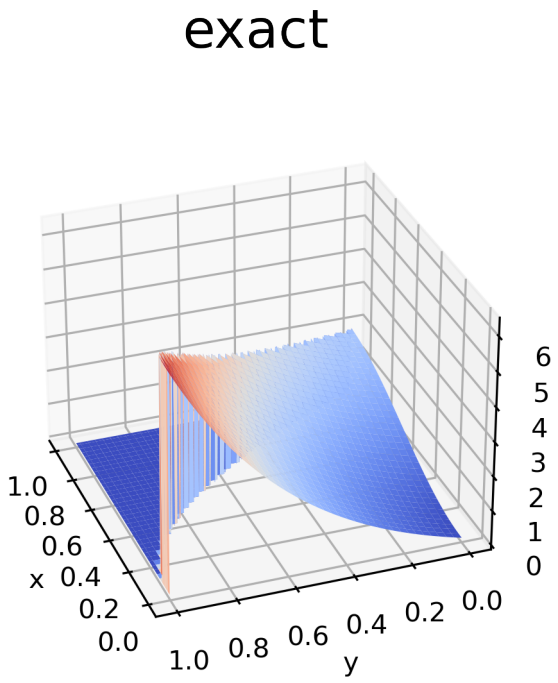



Figure 7: Plot of function $g(x, y) = -\cos 3x + e^{2y}$

# B Stencil

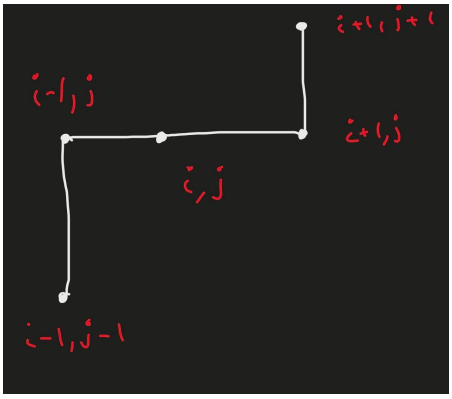


Figure 8: Stencil of numerical scheme