UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics and Computer Science

Mathematics of Computational Science

On the Multigrid Method for Anisotropic Diffusion

Msc Thesis Applied Mathematics

October 3, 2023

Submitted by:
Marius Albert den Oudsten
University of Twente

Under the guidance of:
dr. M. Schlottbom

Contents

1 Introduction

Anisotropic diffusion is a diffusion process where the rate of diffusion is not the same in all directions. A common problem when solving partial differential equations (PDEs) with anisotropic diffusion numerically is the stability [19]. When the grid is not properly aligned with the direction of anisotropy, this can lead to problems, especially for strongly anisotropic problems.

Strongly anisotropic diffusion problems present a numerical challenge, because errors in the direction where the diffusion is large may have a significant effect on transport in the perpendicular direction. Anisotropic diffusion shows up in a wide range of physical sciences. For instance, in fusion plasmas in magnetic fusion devices, anisotropic diffusion shows up in electron diffusion through the magnetic field [19]. In biology the diffusion of water in neural fibres is anisotropic because of the underlying tissue structure [15]. In image processing anisotropic diffusion is used to remove noise from images while preserving data [7].

Different strategies have been tried to robustly solve strongly anisotropic diffusion problems. A strategy is to try to align the grid in such a way that the problem is well-conditioned [14]. Another possibility is to use custom single-level iterative methods [17]. Also, multi-level iterative solvers are used to solve the problem numerically [6].

Another example of a multilevel method is [1]. In [1], the multigrid method with multiple different smoothers is applied to the anisotropic diffusion problem. The methods are analysed using the local Fourier analysis (LFA). The ILLU method performs the best smoother according to the LFA. As the LFA assumes periodic boundary conditions, and the implementation of the multigrid method and the ILLU smoother in this paper are unclear, we want to investigate the convergence of the multigrid method with the ILLU smoother applied to the anisotropic diffusion problem.

In this report, we will apply the multigrid method with the Jacobi, Gauss-Seidel and ILLU smoother to the anisotropic diffusion problem. We will investigate the convergence of the multigrid method using the eigenvalue analysis. This analysis will show that the multigrid method is not always converging for strongly anisotropic problems. To address this, we will propose a bound on the anisotropy for which the multigrid method with ILLU smoother converges. Also, we will introduce a variation of the multigrid method that converges robustly for strongly anisotropic problems.

The report starts with an example of the multigrid in 1 dimension, to gently introduce the reader to the multigrid method. Then the anisotropic diffusion problem in 2 dimensions is introduced and the ILLU smoother is discussed. In the end, the results will be shown.

2 Introductory example of multigrid

This section contains a simple example of the multigrid method applied to a partial differential equation (PDE) in one dimension. This is used as an introductory example to introduce the multigrid method.

To solve a PDE, several steps are taken. First, the PDE is discretized, and then a linear system is obtained. This linear system is then solved to obtain a numerical solution to the PDE. The linear system can be solved using direct or iterative solvers. One of these iterative solvers in the multigrid method. The multigrid method can be used to solve linear systems that result from discretizing PDEs.

To introduce the multigrid method we consider the Poisson equation in one dimension:

$$\begin{cases} \text{Find } u \in H_0^1(\Omega) \text{ such that} \\ -u''(x) = f(x), x \in \Omega = (0,1), \end{cases} \tag{1}$$

where $f(x) \in L^2(\Omega)$ is a square integrable function on the domain Ω. The Hilbert space $L^2(\Omega)$ contains all square-integrable functions on Ω. The space $H_0^1(\Omega)$ is the subspace of $L^2(\Omega)$ for which the weak derivative u' is square integrable, i.e. $u' \in L^2(\Omega)$ and u vanish on the border $\partial\Omega$ [13, Section 5].

In the remainder of this section, the multigrid method applied to Equation (1) will be shown. Knowledge of the multigrid method is not assumed in this section. If one is already familiar with the multigrid method, one can continue in Section 3.

2.1 Weak formulation

For the multigrid method, the PDE is discretized according to the finite element method (FEM). There are also other methods that can be used to solve PDEs, like finite difference methods. FEM is better at handling complex geometries [10, Introduction], and is thus used for multigrid methods in this report.

The first step to solve a PDE using the FEM is to put the equation into the weak formulation. To get the weak formulation, the PDE from Equation (1) is multiplied with an infinitely differentiable test function $v \in C_0^\infty$, where C_0^∞ with a compact support in $(0, 1)$. Then integration by parts is performed [18, Chapter 24]. We can see that any solution u of Equation (1) satisfies;

$$\int_\Omega u'v'd\Omega = \int_\Omega fvd\Omega. \tag{2}$$

We can see that Equation (2) makes sense for $u, v \in H_0^1(\Omega)$, where $H_0^1(\Omega)$ is the space of functions that are square integrable, have square-integrable weak derivates and vanish on the border $\partial\Omega$ [13, Section 5]. This gives the weak formulation where $u \in H_0^1(\Omega)$ is the weak solution:

$$\begin{cases} \text{Find } u \in H_0^1(\Omega) \text{ such that} \\ a(u, v) = l(v), \forall v \in H_0^1(\Omega), \\ \text{where } a(u, v) = \int_\Omega u'v' d\Omega, \text{ and } l(v) = \int_\Omega fv d\Omega. \end{cases} \tag{3}$$

From Lax-Milgram [18, Lemma 25.2] it follows that Equation (3) is well-posed. Thus there exists a unique solution. In the next subsection, we will obtain a linear system to approximate u with FEM. We will then approximate the solution of this linear system with the multigrid method.

2.2 Galerkin approximation

Now, the domain $\Omega = (0, 1)$ is discretized. A grid with $n + 1$ points is defined such that:

$$\begin{cases} \{x_0, x_1, x_2, \ldots x_n\} \text{ is a partition of } \Omega, \text{ with} \\ x_0 = 0 < x_1 < x_2 < \ldots < x_n = 1. \end{cases} \tag{4}$$

There are different choices that can be made when discretizing a domain in one dimension. For simplicity, the grid is uniformly spaced. This means that $x_{i+1} - x_i = h := \frac{1}{n}$ for $i = 0, 1, \ldots, n - 1$.

By projecting the weak formulation in Equation (3) onto a finite-dimensional subspace, the Galerkin approximation is obtained. For this projection, continuous piecewise linear basis functions $(\varphi_i(x), i = 1, 2, \ldots n - 1)$ are used. These basis functions have the property that

$$\varphi_i(x_j) = \delta_{i,j}, \qquad\qquad i, j = 1, 2, \ldots, n - 1.$$

The finite-dimensional space $V_h(\Omega)$ is a subspace of $H_0^1(\Omega)$ and is spanned by the basis functions φ_i:

$$V_h(\Omega) := \text{span}\{\varphi_i : i = 1, 2, \ldots, n - 1\} \subset H_0^1(\Omega).$$

This gives us the conforming Galerkin approximation [18, Definition 26.2]:

$$\begin{cases} \text{Find } u_h \in V_h \text{ such that} \\ a(u_h, v_h) = l(v_h), \forall v_h \in V_h. \end{cases} \tag{5}$$

This can be solved by substituting $\sum_{i=1}^{n-1} u_i \varphi_i$ for u_h and φ_j for v_h. Then this leads to the problem with unknown $\mathbf{u} = [\mu_1, \mu_2, \dots, \mu_{n-1}]^T \in \mathbb{R}^{n-1}$:

$$\sum_{i=1}^{n-1} \mu_i a(\varphi_i, \varphi_j) = l(\varphi_j), \qquad \text{for } j = 1, 2, \dots, n-1.$$

This can be written as:

$$\mathbf{S}\mathbf{u} = \mathbf{f}, \tag{6}$$

where the $\mathbf{S} \in \mathbb{R}^{(n-1)\times(n-1)}$ is defined by the entries $\mathbf{S}_{i,j} = a(\varphi_i, \varphi_j)$ and $\mathbf{f} \in \mathbb{R}^{n-1}$ is consists of the entries $\mathbf{f}_i = l(\varphi_i)$. This system can be solved using numerical techniques. Some iterative numerical methods will be discussed in the next sections. The coordinate vector $\mathbf{u}$ corresponds directly to the Galerkin solution $u_h = \sum_{i=1}^{n-1} \mu_i \varphi_i$. By the Lax-Milgram theorem [18, Lemma 25.2] both Equation (5) and Equation (6) are well-posed.

2.3 Iterative methods

The linear system Equation (6) can be solved to get the Galerkin solution. For large systems resulting from discretizing PDEs in two or more dimensions, direct solvers can be slow and inefficient [11, Page 125]. In this example of the Poisson equation in one dimension, a direct solver would be efficient. However, as the Poisson equation in one dimension is used as an introduction to the anisotropic diffusion equation in two dimensions, iterative methods are also used here. An example of an iterative method is the Jacobi method. Given $\mathbf{u}^k \in \mathbb{R}^{n-1}$, one iteration of the Jacobi method to solve Equation (6) can be computed with;

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{D}^{-1}\left(\mathbf{f} - \mathbf{S}\mathbf{u}^k\right), \tag{7}$$

where $\mathbf{D} \in \mathbb{R}^{(n-1)\times(n-1)}$ contains the diagonal entries from $\mathbf{S}$. The diagonal matrix $\mathbf{D}$ is invertible because all row vectors are linearly independent [11, Property 1.2]. The row vectors are linearly independent since $\mathbf{D}$ is diagonal and all entries on the diagonal are nonzero by the construction of $\mathbf{S}$.

More generally the stiffness matrix can be split $\mathbf{S} = \mathbf{M} - \mathbf{N}$, such that $\mathbf{M}$ is nonsingular. Then a linear iterative method can be defined as;

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{M}^{-1}\left(\mathbf{f} - \mathbf{S}\mathbf{u}^k\right), \tag{8}$$

where $\mathbf{M}$ is called the preconditioner. The preconditioner $\mathbf{M}$ can be inverted due to its nonsingularity. An iterative method of this form will converge if and only if the spectral radius of $I - \mathbf{M}$ is strictly smaller than 1 [11, Theorem 4.1]. The spectral radius is denoted with ρ and is the maximum absolute value of the set of eigenvalues:

$$\rho\left(\mathbf{M}\right) = \max_{\lambda \in \sigma(\mathbf{M})} |\lambda|\,, \tag{9}$$

where $\sigma\left(\mathbf{M}\right)$ is the set of eigenvalues corresponding to $\mathbf{M}$.

Another iterative method is the Gauss-Seidel method. For this method, $\mathbf{S}$ is decomposed into $\mathbf{S} = \mathbf{L} + \mathbf{D} + \mathbf{U}$, where $\mathbf{D}$ is the same diagonal matrix as for the Jacobi method, $\mathbf{L}$ and $\mathbf{U}$ are strictly lower and upper triangular matrices. The forward Gauss-Seidel method is of the form of Equation (8), with $\mathbf{M} = \mathbf{L} + \mathbf{D}$, while the backward Gauss-Seidel uses $\mathbf{M} = \mathbf{D} + \mathbf{U}$, where $\mathbf{S}, \mathbf{M}, \mathbf{L}, \mathbf{D}, \mathbf{U} \in \mathbb{R}^{(n-1)\times(n-1)}$.

The forward Gauss-Seidel iterative method is applied to the linear system from Equation (6) which originates from Equation (1). The initial error, and the error after 5 and 10 iterations can be seen in Figure 1.

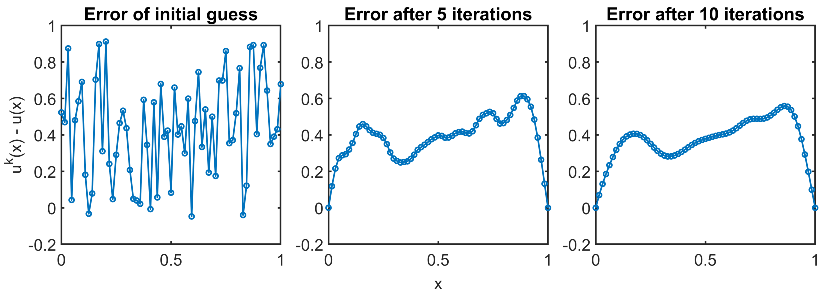

Figure 1: The initial error and after 5 and 10 iterations of the Gauss-Seidel smoother on a grid with 65 points when solving Equation (6) and starting with a random guess.

In Figure 1 we see that the error decreases when multiple steps of the Gauss-Seidel method are applied. We can also see that the error gets ”smoothed” as it decreases. Because of this ”smoothing”, we will also call the iterative methods ”smoothers”.

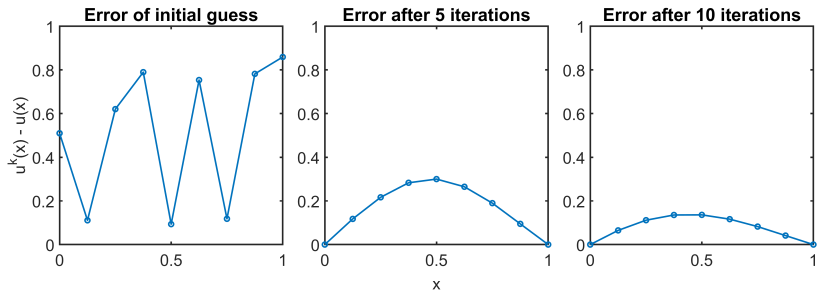

Figure 2: The initial error and after 5 and 10 iterations of the Gauss-Seidel smoother on a grid with 9 points when solving Equation (6) and starting with a random guess.

In Figure 2 we can see that the smoothing also happens when fewer points are used. However because there are fewer points, the error decreases more quickly from the smoothing, and also the iterative method is numerically cheaper to apply for fewer points. The multigrid method leverages both properties by using multiple grids to solve Equation (6).

2.4 Multigrid method

Here will discuss the multigrid method in an abstract setting. Consider a sequence of spaces such that

$$V_1 \subset V_2 \subset \ldots \subset V_m \subset H_0^1(\Omega).$$

For completeness, we recall the Galerkin problem in the linear finite-dimensional subspace V_m here:

$$\begin{cases} \text{Find } u_m \in V_m \text{ such that} \\ a(u_m, v_m) = l(v_m), \forall v_m \in V_m. \end{cases} \tag{10}$$

As we want to have an iterative method, we reformulate the problem such that an initial guess can be used. With an initial guess $u^0 \in V_m$, we want to find $\tilde{u_m} \in V_m$ such that $u^0 + \tilde{u_m}$ solves the Galerkin approximation problem in Equation (10). This gives us

$$\begin{cases} \text{Find } \tilde{u_m} \in V_m \text{ such that} \\ a(\tilde{u_m}, v_m) = l(v_m) - a\left(u^0, v_m\right), \forall v_m \in V_m. \end{cases} \tag{11}$$

To approximate the solution, we start out with $\bar{0}$ which denotes the function that is zero on the entire domain Ω. Then we apply s steps of a smoother, for instance, Equation (8), to obtain an approximation of u_m which we will call $u_m^{pre} \in V_m$. As this is an approximation, it is not perfect. The approximation can be improved with the subspace correction step in the subspace V_{m-1}:

$$\begin{cases} \text{Find } u_{m-1} \in V_{m-1} \text{ such that} \\ a\left(u_{m-1}, v_{m-1}\right) = l\left(v_{m-1}\right) - a\left(u^0 + u_m^{pre}, v_{m-1}\right), \forall v_{m-1} \in V_{m-1}. \end{cases}$$

We will define u_m^{sub} as the summation of u_m^{pre} and the approximation of u_{m-1}. As the approximation of u_{m-1} may not be perfect and the space V_{m-1} is strictly smaller than the space V_m, the new value u_m^{sub} is still an approximation. So we will apply another s steps of a smoother to u_m^{sub} to obtain u_m^{post}. This is one step of a multigrid V-cycle. But we still need to discuss how to approximate u_{m-1}.

We can define the subspace correction step in general. For an approximate solution $u^0 \in V_m$, the subspace correction $u_i \in V_i$ is the weak solution of:

$$\begin{cases} \text{Find } u_i \in V_i \text{ such that} \\ a\left(u_i, v_i\right) = l\left(v_i\right) - a\left(u^0, v_i\right), \forall v_i \in V_i. \end{cases} \tag{12}$$

In the V-cycle of the multigrid the subspace correction on grid i, with $i > 1$, is calculated as before. First s smoother steps are applied to the initial function $\bar{0}$, and then the approximate solution from the subspace correction from grid $i-1$ is added. Finally, s smoother steps are applied again to get u^{post}. On the coarsest grid, u_1 is not approximated but calculated exactly. This completes the V-cycle of the multigrid algorithm, that approximates $u_m \in V_m$ from Equation (10).

Algorithm 1 Multigrid algorithm to approximate u_m from Equation (10)

```
function MULTIGRID($m, tol$)
    $u_m^{approx} = \bar{0}$
    while $\|a(u_m^{approx}, v_m) - l(v_m)\| \geq tol, \forall v_m \in V_m$ do
        $u_m^{approx} = u_m^{approx} + \mathrm{VCycle}(m, u_m^{approx})$        ▷ Equation (11)
    end while
    return $u_m^{approx}$
end function
```

The full multigrid algorithm calls the MULTIGRID function with the updated guesses as input until a convergence criterion is achieved. In Algorithm 1 the

Algorithm 2 V-cycle step to approximate u_i from in Equation (12)

```
function VCycle($i, u^0$)
    if $i > 1$ then
        $u_i^{pre} = \text{Smoother}\big(i, u^0\big)$
        $u_i^{sub} = u_i^{pre} + \text{VCycle}\big(i-1, u^0 + u_i^{pre}\big)$
        $u_i^{post} = u_i^{sub} + \text{Smoother}\big(i, u^0 + u_i^{sub}\big)$
        return $u_i^{post}$
    else
        return $u_i$                    ▷ Return weak solution $u_i$ from Equation (12)
    end if
end function
```

iterations are stopped when the residual is smaller than the tolerance. However, another stopping criterion could be the amount of iterations. The full algorithm is shown in Algorithm 1. The algorithm calls the VCycle algorithm, which is shown in Algorithm 2. Lastly, Smoother$\big(i, u^0\big)$ is called. This returns a function $u \in V_i$ which approximates u_i in Equation (12).

2.5 Numerical implementation

Here, the multigrid method will be applied to Equation (3). This is an example of an implementation of the multigrid method. This will make it easier to understand the multigrid method in two dimensions. For this example, we need to recall some definitions. Consider the discretizations of Ω as in Figure 3. Grid i consist of the points $\big\{0, \frac{1}{n}, \frac{2}{n}, \dots, \frac{n-1}{n}, 1 |$ where $n := 2^i + 1\big\}$.

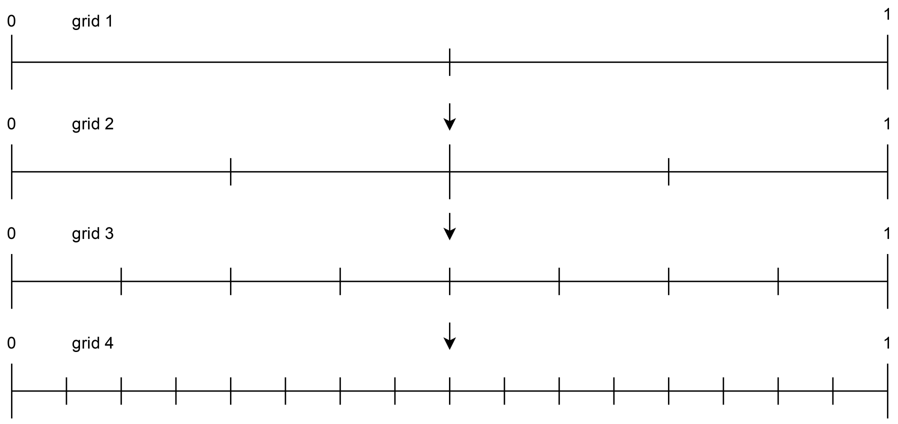

Figure 3: Nested discretizations in 1D

As before, the continuous linear piecewise basis functions are used. We define $x_{i,j}$ as the i-th point on the j-th grid, and $\varphi_{i,j}$ as the i-th basis function on the j-th grid. Then $\varphi_{i,k}\left(x_{j,k}\right) = \delta_{i,j}$. Now we can define the space V_i:

$$V_i = \operatorname{span}\left\{\varphi_{j,i} : j = 2, 3, \ldots, 2^i\right\} \tag{13}$$

These finite-dimensional spaces are nested such that

$$V_1 \subset V_2 \subset \ldots \subset V_m \subset H_0^1\left(\Omega\right).$$

Any function $u_i \in V_i$ is uniquely determined by the corresponding coordinate $\mathbf{u} = \left[\mathbf{u}_2, \mathbf{u}_3, \ldots, \mathbf{u}_{2^i}\right]$ through the following relation:

$$u_i = \sum_{j=2}^{2^i} \mathbf{u}_j \varphi_{j,i}. \tag{14}$$

All functions $u_i \in V_i$ are also elements of V_{i+1} as V_i is a subspace of V_{i+1}. So the embedding $p_i : V_i \hookrightarrow V_{i+1}$ with $p_i\left(u_i\right) = u_i$ exists for all $i < m$. The matrix representation of this embedding is

$$P_i = \begin{bmatrix} 1/2 & & & & & \\ 1 & & & & & \\ 1/2 & 1/2 & & & & \\ & 1 & & & & \\ & 1/2 & 1/2 & & & \\ & & 1 & & & \\ & & & \ddots & & \\ & & & 1/2 & 1/2 & \\ & & & & 1 & \\ & & & & 1/2 & 1/2 \\ & & & & & 1 \\ & & & & & 1/2 \end{bmatrix} \in \mathbb{R}^{\left(2^i-1\right)\times\left(2^i-1\right)}.$$

The matrix P_i is called the prolongation matrix. The coordinate vector $\mathbf{u}_i \in \mathbb{R}^{2^i-1}$ corresponding to a function $u_i \in V_i$ can be mapped to a coordinate vector $\mathbf{u}_{i+1} = P_i\mathbf{u}_i \in \mathbb{R}^{2^{i+1}-1}$ corresponding to to a function in V_{i+1}. The prolongation matrix can also to map a coordinate vector coordinate vector $\mathbf{u}_{i+1} \in \mathbb{R}^{2^{i+1}-1}$ corresponding to a function $u_{i+1} \in V_{i+1}$ to a coordinate vector $u_i = P_i^T u_{i+1} \in \mathbb{R}^{2^i-1}$ corresponding to a function $u_i \in V_i$.

On the the finest grid, grid m, the stiffness matrix $\mathbf{S}_m$ and right hand side vector

f are calculated. The the stiffness matrix $\mathbf{S}_m$ has entries $\mathbf{S}_{i,j} = a\left(\varphi_{i,m}, \varphi_{j,m}\right)$. The right hand side **f** is defined by the entries $\mathbf{f}_i = l\left(\varphi_{i,m}\right)$. The prolongation matrix can be used to calculate the stiffness matrices on coarser grids

$$\mathbf{S}_i = P_i^T \mathbf{S}_{i+1} \in \mathbb{R}^{\left(2^i-1\right)\times\left(2^i-1\right)}. \tag{15}$$

With all elements defined, we can construct the algorithm. This is the same as Algorithm 1 and Algorithm 2, but applied to solving Equation (2). The multigrid algorithm can be seen in Algorithm 3 and the V-cycle can be seen in Algorithm 4.

Algorithm 3 Multigrid method to solve Equation (3)

```
function MULTIGRID1D($m$, $tol$)
    $\mathbf{u}^{approx} = \mathbf{0}$
    while $\|\mathbf{f} - \mathbf{S}_m \mathbf{u}^{approx}\| \geq tol$ do
        $\mathbf{u}^{approx} = \mathbf{u}^{approx} + \text{VCYCLE1D}\left(m, \mathbf{f} - \mathbf{S}_m \mathbf{u}^{approx}\right)$
    end while
    return $\mathbf{u}_m$
end function
```

Algorithm 4 Step of the V-Cycle of the multigrid method to solve Equation (3)

```
function VCYCLE1D($i$, $\mathbf{f}_i$)
    if $i > 1$ then
        $\mathbf{u}^{pre} = \mathbf{D}_i^{-1} \mathbf{f}_i$                                   ▷ 1 step of Jacobi smoother
        $\mathbf{u}^{sub} = \mathbf{u}^{pre} + P_i \text{VCYCLE1D}\left(i-1, P_i^T\left(\mathbf{f}_i - \mathbf{S}_i \mathbf{u}^{pre}\right)\right)$
        $\mathbf{u}^{post} = \mathbf{u}^{sub} + \mathbf{D}_i^{-1}\left(\mathbf{f}_i - \mathbf{S}_i \mathbf{u}^{sub}\right)$           ▷ 1 step of Jacobi smoother
    else
        $\mathbf{u}^{post} = \mathbf{S}_1^{-1} \mathbf{f}_i$                                      ▷ Exact solution
    end if
    return $\mathbf{u}^{post}$
end function
```

3 Anisotropic diffusion in 2D

We will now consider anisotropic diffusion. This is the equation of interest for this report. We will apply the multigrid method to this equation, in a similar fashion as in Section 2.

We consider the following problem for anisotropic diffusion:

$$\begin{cases} \text{Find } u \in H^2(\Omega) \cap C_0(\Omega) \text{ such that} \\ -\nabla \cdot (K \nabla u(x,y)) = f(x,y), (x,y) \in \Omega = (0,1)^2, \end{cases} \tag{16}$$

where the diffusion tensor $K \in \mathbb{R}^{2\times 2}$ is given by:

$$K = R^T \begin{bmatrix} 1 & 0 \\ 0 & \epsilon \end{bmatrix} R, \epsilon > 0,$$

with the rotation matrix $R \in \mathbb{R}^{2\times 2}$;

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \theta \in [0, 2\pi).$$

Again, we convert this problem into a weak formulation. This is done by multiplying with a test function $v \in C_0^\infty(\Omega)$, and then integrating over the domain Ω. Applying integration by parts leads to the weak formulation:

$$\begin{cases} \text{Find } u \in H_0^1(\Omega) \text{ such that} \\ \int_\Omega K \nabla u \cdot \nabla v d\Omega = \int_\Omega f v d\Omega, \forall v \in H_0^1(\Omega). \end{cases} \tag{17}$$

3.1 Domain discretization

In this subsection, the domain will be discretized. This needs to be done, to be able to define basis functions, which in turn are required to calculate the linear system. The domain Ω has 2 dimensions. We discretize the domain into triangles, this is one of the options for domains in two dimensions [12, Section 3]. The multigrid method has been shown to work with unstructured grids in 2 dimensions [9]. However, for the ILLU smoother, which will be discussed in Section 3.3, a uniform grid is required [3]. So we will use structured grids throughout the report.

To save the triangles properly, the edges and vertices are also numbered. In Figure 4, 5 and 6, the numbering of the vertices, edges and triangles on grid $1, 2$

and 3 can be seen. Grid i has $\left(2^{i-1}+1\right)^2$ vertices.

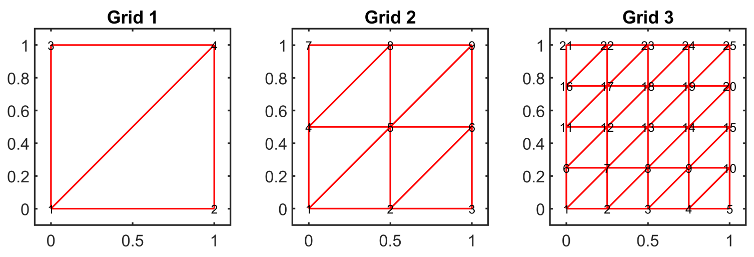

Figure 4: Points on Grid 1, 2 and 3

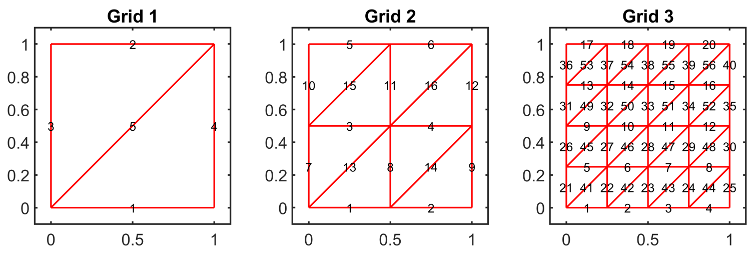

Figure 5: Edges on Grid 1, 2 and 3

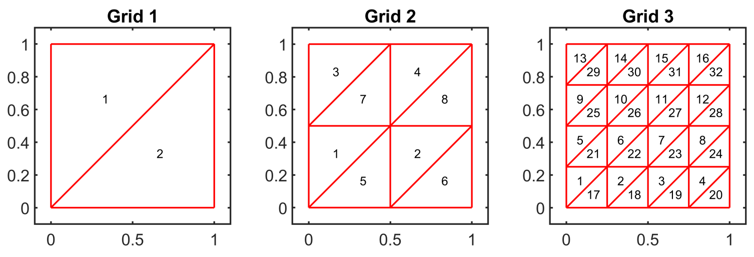

Figure 6: Triangles on Grid 1, 2 and 3

3.2 Linear system

A linear system can be obtained from the weak formulation in Equation (17) and the discretization. Continuous piecewise linear basis functions φ_j are used for the discrete problem. The basis functions are linear on each triangle and correspond to the points from the discretization. The space $V_i \subset H_0^1(\Omega)$ is spanned by all basis functions on grid i, which has $n_i := \left(2^{i-1}+1\right)^2$ points:

$$V_i := span\left\{\varphi_j | j = 1, 2, \ldots, n_i, \right\}. \tag{18}$$

Then the Galerkin problem is:

$$\begin{cases} \text{Find } u_i \in V_i \text{ such that} \\ \int_\Omega K \nabla u_i \cdot \nabla v_i d\Omega = \int_\Omega f v_i d\Omega \forall v_i \in V_i. \end{cases} \tag{19}$$

Any element $u_i \in V_i$ can be represented as a weighted sum of the basis functions; $u_i = \sum_{j=1}^{n_i} \mu_j \varphi_j$. So we can transform Equation (19) to a linear system with unknown $\mathbf{u} = [\mu_1, \mu_2, \ldots, \mu_{n_i}]^T$:

$$\sum_{k=1}^{n_i} \mu_k \int_\Omega K \nabla \varphi_k \cdot \nabla \varphi_j d\Omega = \int_\Omega f \varphi_j d\Omega. \tag{20}$$

This can be written as:

$$\mathbf{S}_m \mathbf{u} = \mathbf{f}, \tag{21}$$

where the entry at k, j of $\mathbf{S}_m \in \mathbb{R}^{n_i \times n_i}$ is given by $\int_\Omega K \nabla \varphi_i \cdot \nabla \varphi_j d\Omega$ and the entry at the j-th position of $\mathbf{f} \in \mathbb{R}^n$ is given by $\int_\Omega f \varphi_j d\Omega$. There exists an isomorphism between the coordinate $\mathbf{u}_m \in \mathbb{R}^{n_i}$ and $u_m \in V_m$. The relation is given by $u_m = \sum_{i=1}^{n_i} \mu_i \varphi_i \in V_m$.

3.3 Iterative methods

This subsection builds on the section on iterative methods in Section 2.3. This subsection introduces multiple iterative methods. These methods can be used to solve linear systems, and are introduced as such. However, they will mainly be used as smoothing operators for the multigrid method. First, two well-known iterative methods are quickly introduced, namely the Jacobi and Gauss-Seidel methods. Then the Incomplete Line LU factorization is introduced.

3.3.1 Matrix splitting methods

In this report, three iterative matrix splitting methods are used. As the name implies, the matrix is split into these methods. To solve the linear system on grid i, $\mathbf{S}_i\mathbf{u} = \mathbf{f}$, the mass matrix $\mathbf{S}_i \in \mathbb{R}^{n_i \times n_i}$ is split such that $\mathbf{S}_i = \mathbf{M} - \mathbf{N}$, with $n_i = 2^{i-1} + 1$. This split can be used in the iterative solver:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{M}^{-1}\left(\mathbf{f} - \mathbf{S}_i\mathbf{u}^k\right). \tag{22}$$

For the Jacobi and Gauss-Seidel methods, the matrix $\mathbf{S}_i$ is decomposed into $\mathbf{S}_i = \mathbf{L} + \mathbf{D} + \mathbf{U}$. Where $\mathbf{L}$ is a strictly lower triangular matrix, $\mathbf{U}$ is a strictly upper triangular matrix and $\mathbf{D}$ contains all diagonal entries of $\mathbf{S}_i$.

The Jacobi method uses $\mathbf{M} = \mathbf{D}$. The Gauss-Seidel method can be executed forwards and backwards. For the forward iteration $\mathbf{M} = \mathbf{L} + \mathbf{D}$, for the backward iteration $\mathbf{M} = \mathbf{D} + \mathbf{U}$ [11, Section 4.2.1].

3.3.2 Incomplete line LU method

The third method that will be used in this report is the Incomplete Line LU (ILLU) factorization. In this method, the stiffness matrix will be factorized using the Line LU Factorization. We will consider the stiffness matrix on grid i, which we call $\mathbf{S}_i \in \mathbb{R}^{n_i^2 \times n_i^2}$, with $n_i = 2^{i-1} + 1$. This ILLU method is explained in multiple different papers [3–5] and we will summarize the method here.

As described in Section 3.1, the grid is structured. Because of this structure, the stiffness matrix $\mathbf{S}_i$ corresponding grid i is of the form [3, Equation 2.20];

$$\mathbf{S}_i = \begin{bmatrix} D_1 & U_1 & & & & & \\ L_2 & D_2 & U_2 & & & & \\ & L_3 & D_3 & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & D_{n_i-2} & U_{n_i-2} & \\ & & & & L_{n_i-1} & D_{n_i-1} & U_{n_i-1} \\ & & & & & L_{n_i} & D_{n_i} \end{bmatrix} \in \mathbb{R}^{n_i^2 \times n_i^2}, \tag{23}$$

where all entries L_j, D_j and U_j, $j = 1, 2, \dots, n_i$ are band matrices of size $n_i \times n_i$. So in L_j, the only entries are along the main diagonal and the lower diagonal. For U_j, the entries are along the main diagonal and the upper diagonal. D_j are tridiagonal matrices. The total size of $\mathbf{S}_i$ is n_i^2 by n_i^2 on the i-th grid. Please note that the notation of [3] is not the same.

With the block matrices L_j, D_j, U_j from above, there exists a factorization of $\mathbf{S}$ such that [3, Equation 2.21];

$$\mathbf{S}_i = (L + B)\, B^{-1}\, (B + U)\,, \tag{24}$$

where;

$$L = \begin{bmatrix} \mathbf{0} & & & \\ L_2 & \ddots & & \\ & \ddots & \ddots & \\ & & L_n & \mathbf{0} \end{bmatrix}, \quad B = \begin{bmatrix} B_1 & & \\ & \ddots & \\ & & B_n \end{bmatrix}, \quad U = \begin{bmatrix} \mathbf{0} & U_1 & & \\ & \ddots & \ddots & \\ & & \ddots & U_{n-1} \\ & & & \mathbf{0} \end{bmatrix}, \tag{25}$$

where $\mathbf{0}$ is a block matrix of size $2^{i-1}+1$ by $2^{i-1}+1$ containing zeros, just like the empty entries. For this factorization to hold the following entries for B_i are required [3];

$$B_1 = D_1, \qquad B_j = D_j - L_j B_{j-1}^{-1} U_{j-1}, \qquad \text{for } j = 2, 3, \ldots, n_i. \tag{26}$$

This factorization is called the Line LU factorization. However, numerically this not very efficient as B_j^{-1} are full matrices. To keep the same sparsity pattern, B_j is approximated by $\bar{B}_j \approx B_j$. This approximation is defined in the following way:

$$\bar{B}_1 = D_1, \quad \bar{B}_j = D_j - tridiag\left(L_j \bar{B}_{j-1}^{-1} U_{j-1}\right), \quad \text{for } j = 2, 3, \ldots, n_i. \tag{27}$$

This approximation does introduce an error. This error is introduced by the *tridiag* operation and is denoted by $\bar{E}$. The matrix $\mathbf{S}_i$ is now split into a part from the ILLU factorization and a part of the error introduced by the same ILLU factorization:

$$\mathbf{S}_i = \left(L + \bar{B}\right) \bar{B}^{-1} \left(\bar{B} + U\right) + \bar{E}. \tag{28}$$

Now that the stiffness matrix $\mathbf{S}_i$ is split, it can be used as a smoother like the matrix splitting methods. This time $\mathbf{M} = \left(L + \bar{B}\right) \bar{B}^{-1} \left(\bar{B} + U\right)$. This leads to the following smoother:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \left(\left(L + \bar{B}\right) \bar{B}^{-1} \left(\bar{B} + U\right)\right)^{-1} \left(\mathbf{f} - \mathbf{S}_i \mathbf{u}^k\right). \tag{29}$$

3.4 Prolongation matrix

Another ingredient for the multigrid method is the prolongation matrix. With the domain in 2 dimensions, this is a bit more complicated than in 1 dimension. However, the idea is the same. The setup will be explained here, without going into too much detail, as this is already thoroughly covered in [8, Section 2.3.4] (where it is called interpolation) and [2, Section 3.4]

Grid $i+1$ has vertices on the vertices of grid i and on the midpoints of the edges of grid i. Using this the prolongation matrix P_i, which prolongates an element from V_i to an element from V_{i+1}, can be constructed. The matrix P_i has a size of $\left(2^i+1\right)^2$ by $\left(2^{i-1}+1\right)^2$. For the vertices on grid $i+1$ that correspond to vertices, the corresponding entry should be set to one 1. For the vertices on grid $i+1$ corresponding to the midpoints of edges on grid i, the values corresponding to the new points on grid $i+1$ and the endpoints of the edge on grid i should be set to $\frac{1}{2}$.

This structure of creating the prolongation matrix will work for a general grid in 2 dimensions. So, it will also work for the structured grids used here. Again, the restriction matrix that restricts an element from grid $i+1$ to an element on grid i, is P_i^T.

3.5 Multigrid V-cycle

Now, that all the required ingredients for the multigrid are known, it is time for the multigrid method. Here the algorithm for the multigrid method will be given. Algorithm 5 returns the coordinate vector $\mathbf{u}_m$ corresoponding to u_m from Equation (19). Algorithm 6 executes the V-cycle, and Algorithm 7 is the smoother algorithm. In each V-cylce we start with an initial guess of $\mathbf{0}$. Therefore the residual that is the input of the presmoothing step is $\mathbf{f}_i - \mathbf{S}_i\mathbf{0} = \mathbf{f}_i$.

Algorithm 5 Multigrid method to solve Equation (19)

```
function MULTIGRID2D($m$, $tol$)
    $\mathbf{u}^{approx} = \mathbf{0}$
    while $\|\mathbf{f} - \mathbf{S}_m\mathbf{u}^{approx}\| \geq tol$ do
        $\mathbf{u}^{approx} = \mathbf{u}^{approx} + \text{VCYCLE2D}\left(m, \mathbf{f} - \mathbf{S}_m\mathbf{u}^{approx}\right)$
    end while
    return $\mathbf{u}^{approx}$
end function
```

Algorithm 6 Step of the V-Cycle of the multigrid method to solve Equation (17)

```
function VCycle2D($i$, $\mathbf{f}_i$)
    if $i > 1$ then
        $\mathbf{u}^{pre} = \text{Smoother2D}\,(i, \mathbf{f}_i)$
        $\mathbf{u}^{sub} = \mathbf{u}^{pre} + P_i \text{VCycle2D}\left(i-1, P_i^T\left(\mathbf{f}_i - \mathbf{S}_i \mathbf{u}^{pre}\right)\right)$
        $\mathbf{u}^{post} = \mathbf{u}^{sub} + \text{Smoother2D}\left(i, \mathbf{f}_i - \mathbf{S}_i \mathbf{u}^{sub}\right)$
    else
        $\mathbf{u}^{post} = \mathbf{S}_1^{-1} \mathbf{f}_i$
    end if
    return $\mathbf{u}^{post}$
end function
```

Algorithm 7 Smoother in 2D

```
function Smoother2D($i$, $\mathbf{f}$)
    $\mathbf{u} = \mathbf{0}$
    for $j = 1 : steps$ do
        $\mathbf{u} = \mathbf{u} + M_i^{-1}\left(\mathbf{f} - \mathbf{S}_i \mathbf{u}\right)$        ▷ $M_i$ depends on the type of smoother
    end for
    return $\mathbf{u}$
end function
```

4 Eigenvalue analysis

In this section, the convergence of the multigrid method applied to the anisotropic diffusion equation in 2 dimensions from Section 3 is analysed. The analysis is done by means of an eigenvalue analysis. We will start with the eigenvalues of the smoothers, and then extend to the full multigrid method.

Consider a linear system of the form $\mathbf{S}\mathbf{u} = \mathbf{f}$, with $\mathbf{u}, \mathbf{f} \in \mathbb{R}^n$, and $\mathbf{S}, \mathbf{M} \in \mathbb{R}^{n \times n}$. Using an iterative solver of the form;

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{M}^{-1}\left(\mathbf{f} - \mathbf{S}u^k\right), \qquad \mathbf{M}^{-1} \in \mathbb{R}^{n \times n}, \tag{30}$$

the sequence of vectors $\left\{\mathbf{u}^k\right\}$ converges to $\mathbf{u}$ for any $\mathbf{u}^0$ if and only if the spectral radius of $\left(I - \mathbf{M}^{-1}\mathbf{S}\right)$ is strictly smaller than one [11, Theorem 4.1]. Recall that the spectral radius is the largest absolute value of the eigenvalues.

4.1 Eigenvalues of iterative methods

We will start with the eigenvalue analysis of the iterative methods as discussed in Section 3.3. We consider the Jacobi, Gauss-Seidel and ILLU methods. These methods are of the form in Equation (30). For the error of $\mathbf{u}^k \in \mathbb{R}^n$ we use [12, Section 0.3]:

$$\mathbf{e}^k = \mathbf{u}^k - \mathbf{u}. \tag{31}$$

We can use the error definition to calculate the error propagation. The error propagation on grid m is

$$\mathbf{e}^{k+1} = \mathbf{u}^{k+1} - \mathbf{u} = \mathbf{u}^k - \mathbf{u} + \mathbf{M}^{-1}\left(\mathbf{f} - \mathbf{S}_m\mathbf{u}^k\right) = \left(I - \mathbf{M}^{-1}\mathbf{S}_m\right)\mathbf{e}^k, \tag{32}$$

where I is the identity matrix of size $n_m^2 \times n_m^2$, $n_m = 2^{m-1}+1$. This clearly shows that the error propagation of a step of the smoother is a linear operation. As seen in Section 3.3, the preconditioner matrix $\mathbf{M}$ equals $\mathbf{D}$ for the Jacobi method, the preconditioner matrix is $\mathbf{L}+\mathbf{D}$ or $\mathbf{D}+\mathbf{U}$ for the Gauss-Seidel method and $\left(L+\bar{B}\right)\bar{B}\left(\bar{B}+U\right)$ for the ILLU method.

Both the Jacobi and Gauss-Seidel iterative methods converge when $\mathbf{S}_i$ is strictly diagonally dominant [11, Theorem 4.2]. For the ILLU method, however, we do not have such a theorem. So we will look at the spectral radii of the error propagation of the iterative methods to infer convergence. We will calculate the spectral radius of $I - \mathbf{M}^{-1}\mathbf{S}_m$, with $\mathbf{S}_m$ from Equation (21). This is done in MATLAB for $\theta \in \{0, 0.1, 0.2, \ldots, 3.1\}$ and $\epsilon \in \left\{1, 0.1, 0.01, \ldots, 10^{-5}\right\}$ and $m = 4, 5, 6$. The eigenvalues can be seen in Figure 7.

We can see in Figure 7 that the spectral radius depends on the type of iterative method, the value of ϵ, the value of θ and the grid.

When $\epsilon = 1$, the diffusion tensor K is just the identity matrix in $\mathbb{R}^{n\times n}$. This means that there is no anisotropy, and thus the spectral radius is uniform for varying the rotation θ. For smaller values of ϵ, the anisotropy is stronger. We can see that for stronger anisotropy, the spectral radius increases. This was expected as solving equations with strong anisotropy is considered challenging [19].

If ϵ is smaller than 1 we can clearly see the influence of the rotation θ on the spectral radius. For values of θ that are close to one of the points of the set $\left\{0, \frac{\pi}{2}, \frac{3\pi}{4}, \pi\right\}$, we can see that the spectral radii are relatively small. At those angles of rotation, the lines in the grid are aligned with the direction of the anisotropy. Therefore the grid is optimally aligned for solving the anisotropic diffusion equation [14].

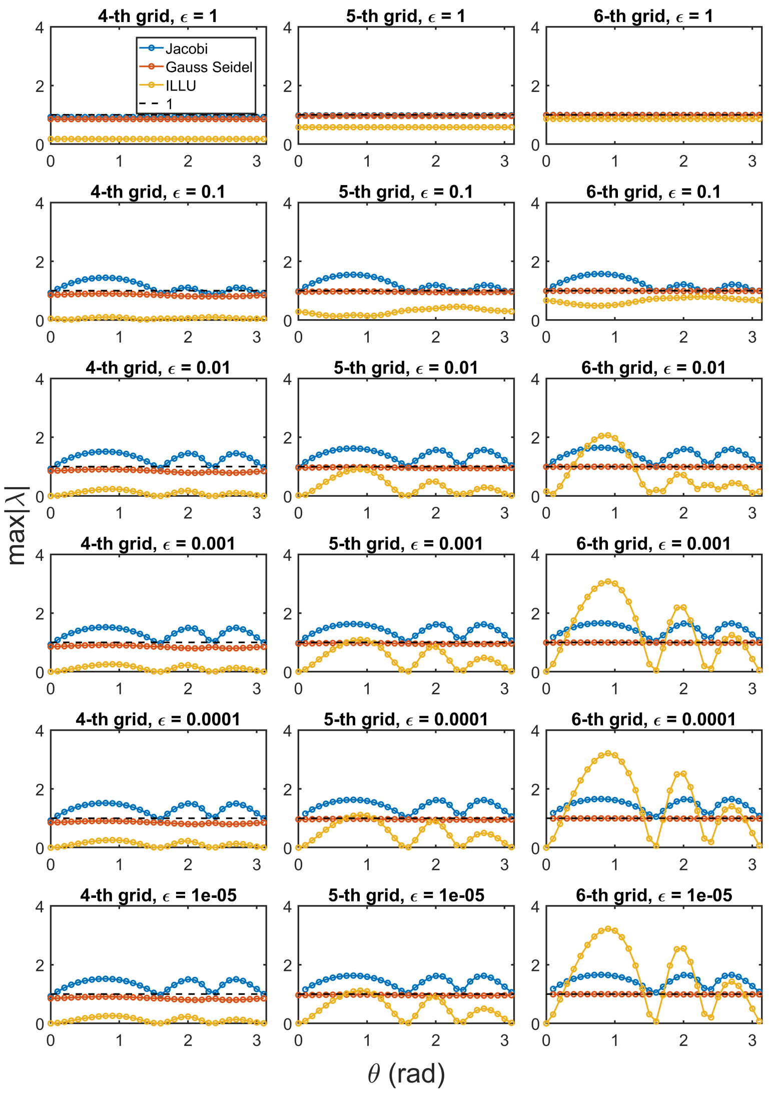

Figure 7: Eigenvalues of iterative methods on different grids for different values of ϵ and θ on grid $m = 4, 5, 6$.

We can see in Figure 7 that on finer grids, the spectral radius of the iterative methods increases. The spectral radius of the Jacobi method is larger than 1 in most cases. Thus, we do not expect the multigrid method with this smoother to converge to the weak solution of the anisotropic diffusion equation. The spectral radius of the Gauss-Seidel method, however, is in all test cases strictly smaller than 1. Thus, we expect the multigrid method with the forward Gauss-Seidel smoother as pre-smoother and the backward Gauss-Seidel method as post-smoother to converge to the weak solution of Equation (17). The spectral radius of the ILLU method is far below 1 in most cases, however, there are also quite some cases where the spectral radius is larger than 1. This happens with small values of ϵ, on finer grids. Because of this, the multigrid method with the ILLU smoother is expected to converge quickly in some cases, but we expect it to not be robust for all values of ϵ and θ, which might only make it usable under certain conditions. We will discuss these conditions in Section 4.6.

4.2 Error propagation of multigrid method

For the error propagation of an iteration of the multigrid method, the same principle is applied. The error after i iterations on grid m is

$$\mathbf{e}_m^i = \mathbf{u}_m^i - \mathbf{u}_m$$

First, the linearity of the error propagation is proven. This is done by first showing it for the 2-grid method and then generalizing the multigrid method. If we have the linearity, we know that eigenvalues do not depend on the input $\mathbf{u}$, but are constant for all $\mathbf{u}^k$.

4.2.1 Linearity of error propagation in 2-grid method

Here the linearity of the error propagation of the V-cycle of the multigrid method with only two grids is shown. Consider Algorithm 8. This algorithm is Algorithm 6 but tailored to the 2-grid method.

Algorithm 8 V-Cycle for the 2-grid method for anisotropic diffusion in 2 dimensions

```
function VCYCLE2GRID($\mathbf{f}_2$)
    $\mathbf{u}^{pre} = \mathbf{0} + \mathbf{M}_{2,\text{pre}}^{-1} (\mathbf{f}_2 - \mathbf{S}_2\mathbf{0})$
    $\mathbf{u}^{sub} = \mathbf{u}^{pre} + P_2\mathbf{S}_1^{-1}P_2^T (\mathbf{f}_2 - \mathbf{S}_2\mathbf{u}^{pre})$
    $\mathbf{u}^{post} = \mathbf{u}^{sub} + \mathbf{M}_{2,\text{post}}^{-1} \left(\mathbf{f}_2 - \mathbf{S}_2\mathbf{u}^{sub}\right)$
    return $\mathbf{u}^{post}$
end function
```

With the same reasoning as for the smoothers, the error propagation for $\mathbf{e}^{sub}$

can be determined;

$$\mathbf{e}^{sub} = \left(I - P_2 \mathbf{S}_1^{-1} P_2^T \mathbf{S}_2\right) \mathbf{e}^{pre}.$$

Combining this with the linearity results of the smoothers leads to:

$$\begin{aligned} \mathbf{e}^{k+1} &= \bar{V}_2 \mathbf{e}^k, \\ \bar{V}_2 &= \left(I - \mathbf{M}_{2,\text{post}}^{-1} \mathbf{S}_2\right) \left(I - P_2 \mathbf{S}_1^{-1} P_2^T \mathbf{S}_2\right) \left(I - \mathbf{M}_{2,\text{pre}}^{-1} \mathbf{S}_2\right). \end{aligned} \tag{33}$$

4.2.2 Linearity of error propagation in multigrid method

In the same way as before, the error propagation of the multigrid method with 3 grids will be calculated here. From Algorithm 6 we see the subspace correction step is

$$\mathbf{u}_3^{sub} = \mathbf{u}_3^{pre} + P_3 \text{VC}{\small\text{YCLE}}\text{2D}(2, P_3^T \left(\mathbf{f}_3 - \mathbf{S}_3 \mathbf{u}_3^{pre}\right),$$

Working this out gives the following term for the error propagation in the subspace correction step;

$$\mathbf{e}_3^{sub} = \left(I - P_3 \left(I - \bar{V}_2\right) P_3^T \mathbf{S}_3\right) \mathbf{e}_3^{pre}.$$

This combines to the full error propagation of the 3-grid cycle:

$$\bar{V}_3 = \left(I - \mathbf{M}_{3,\text{post}}^{-1} \mathbf{S}_3\right) \left(I - P_3 \left(I - \bar{V}_2\right) P_3^T \mathbf{S}_3\right) \left(I - \mathbf{M}_{3,\text{pre}}^{-1} \mathbf{S}_3\right).$$

This can easily be extended to V-cycles with more grids. Then we obtain that the following holds;

$$\begin{aligned} \mathbf{e}_m^{k+1} &= \bar{V}_m \mathbf{e}_m^k, \\ \bar{V}_2 &= \left(I - \mathbf{M}_{2,\text{post}}^{-1} \mathbf{S}_2\right) \left(I - P_2 \mathbf{S}_1^{-1} P_2^T \mathbf{S}_2\right) \left(I - \mathbf{M}_{2,\text{pre}}^{-1} \mathbf{S}_2\right), \\ \bar{V}_j &= \left(I - \mathbf{M}_{j,\text{post}}^{-1} \mathbf{S}_j\right) \left(I - P_j \left(I - \bar{V}_{j-1}\right) P_j^T \mathbf{S}_j\right) \left(I - \mathbf{M}_{j,\text{pre}}^{-1} \mathbf{S}_j\right). \end{aligned} \tag{34}$$

This result shows that the error propagation on every grid is a linear operation.

4.3 Eigenvalue calculation

Here, the method of calculating the eigenvalues will be described. To calculate the eigenvalues of the error propagation of the multigrid method corresponding to the anisotropic diffusion, the eigenvalues of $\bar{V}_m$ from Equation (34) have to be calculated. One way to calculate these eigenvalues would be to construct the matrices $\bar{V}_m$. However, there is another, arguably easier, alternative.

When calling Algorithm 6 with $\mathbf{f} = \mathbf{0}, \mathbf{u}^k \in \mathbb{R}^n$, the resulting output is

$$\text{VCYCLE2D}\left(m, \mathbf{f} - \mathbf{S}_m \mathbf{u}^k\right) = \text{VCYCLE2D}\left(m, -\mathbf{S}_m \mathbf{u}^k\right) = \bar{V}_m \mathbf{u}^k.$$

This means that calling this function with the right-hand side equal to $\mathbf{0}$ results in the error propagation matrix multiplied with $\mathbf{u}^k$. As we are interested in the eigenvalues of the matrix $\bar{V}_m$, the MATLAB function `eigs` can easily be used to compute the eigenvalues of this matrix when given this function handle as an input.

4.4 Eigenvalues of multigrid

In this subsection, the eigenvalues of the error propagation of the multigrid method in Algorithm 5 will be shown and discussed. The eigenvalues are separated into three figures. In Figure 8, the eigenvalues corresponding to the finest grid being grid 4 are shown, in Figure 9 and Figure 10, the finest grid is grid 5 and 6 respectively. All subfigures in each category have the same axes for ease of comparison. Based on the results, the influence of ϵ, θ, the number of grids and the type of smoother will be discussed. The multigrid method with the Gauss-Seidel smoother uses the forward Gauss-Seidel smoother as pre-smoother and the backward Gauss-Seidel smoother as post-smoother.

We can see that the multigrid method with the Jacobi smoother is the worst performing. With this smoother, the multigrid method only converges in a few cases. So for solving the anisotropic diffusion equation with the multigrid method, the Jacobi smoother is not suitable. In most cases tested here, the multigrid method performs better with the ILLU smoother compared to the Gauss-Seidel smoother. However, in the case with six grids and small values of $\epsilon \leq 10^{-3}$, the eigenvalue for the MGM with ILLU smoother is larger for some values of θ. This means that it is not really robust. This was to be expected from the eigenvalues of the smoothers as discussed in Section 4.1.

Again, when $\epsilon = 1$, the spectral radii are uniform for varying rotation values θ. For decreasing values of ϵ, the eigenvalues generally increase. This was expected as solving equations with strong anisotropy is considered challenging [19].

It can also be seen that the eigenvalues depend on θ. for values of θ around

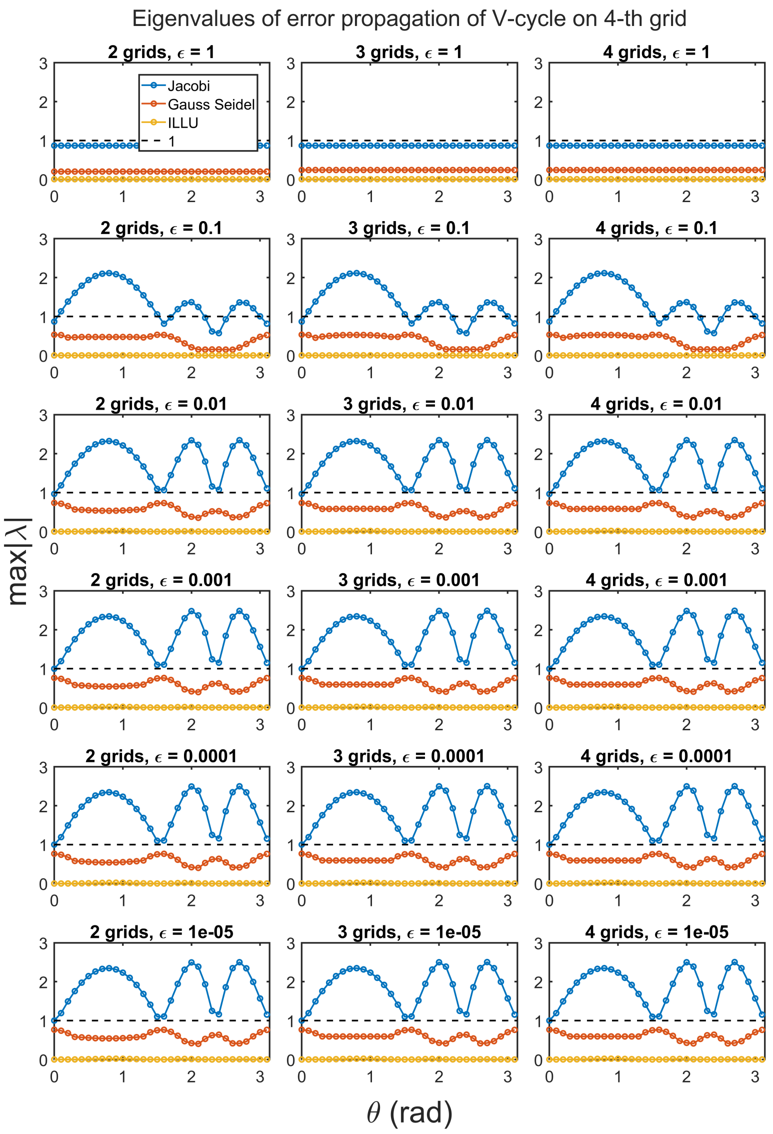

Figure 8: Eigenvalues of error propagation of multigrid method for anisotropic diffusion from Algorithm 8 on the 4-th grid, with different amount of grids in the V-cycle.

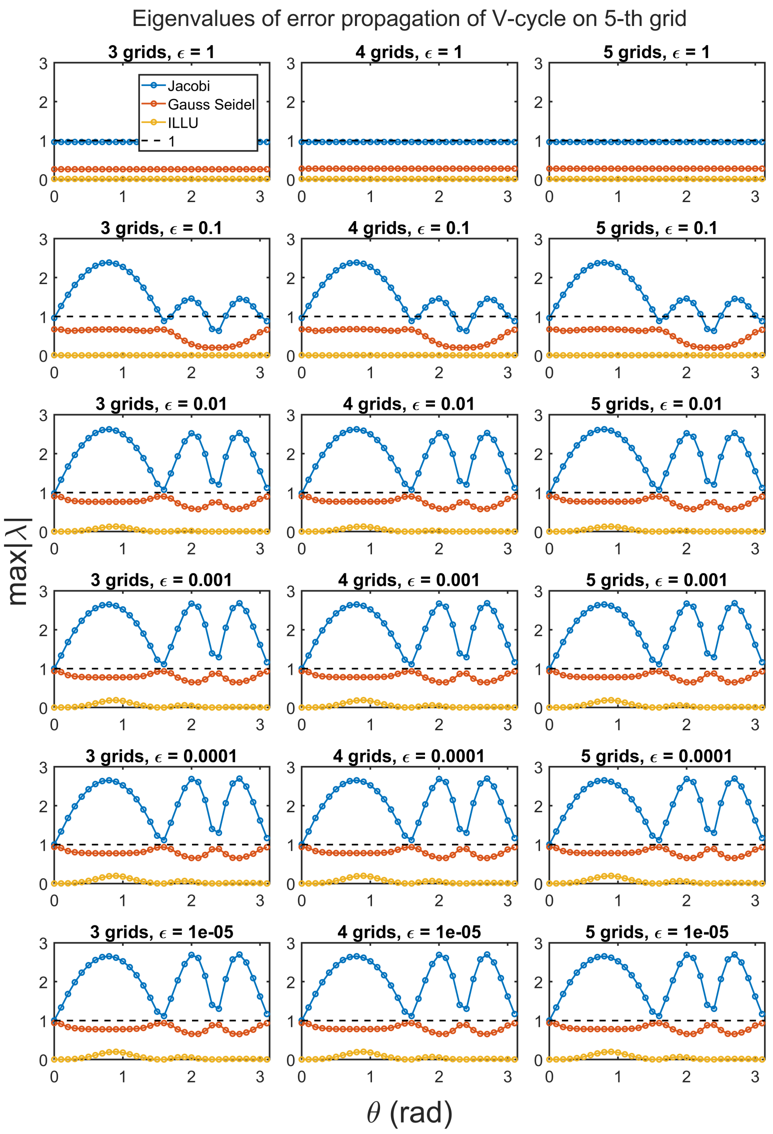

Figure 9: Eigenvalues of error propagation of multigrid method for anisotropic diffusion from Algorithm 8 on the 5-th grid, with different amount of grids in the V-cycle

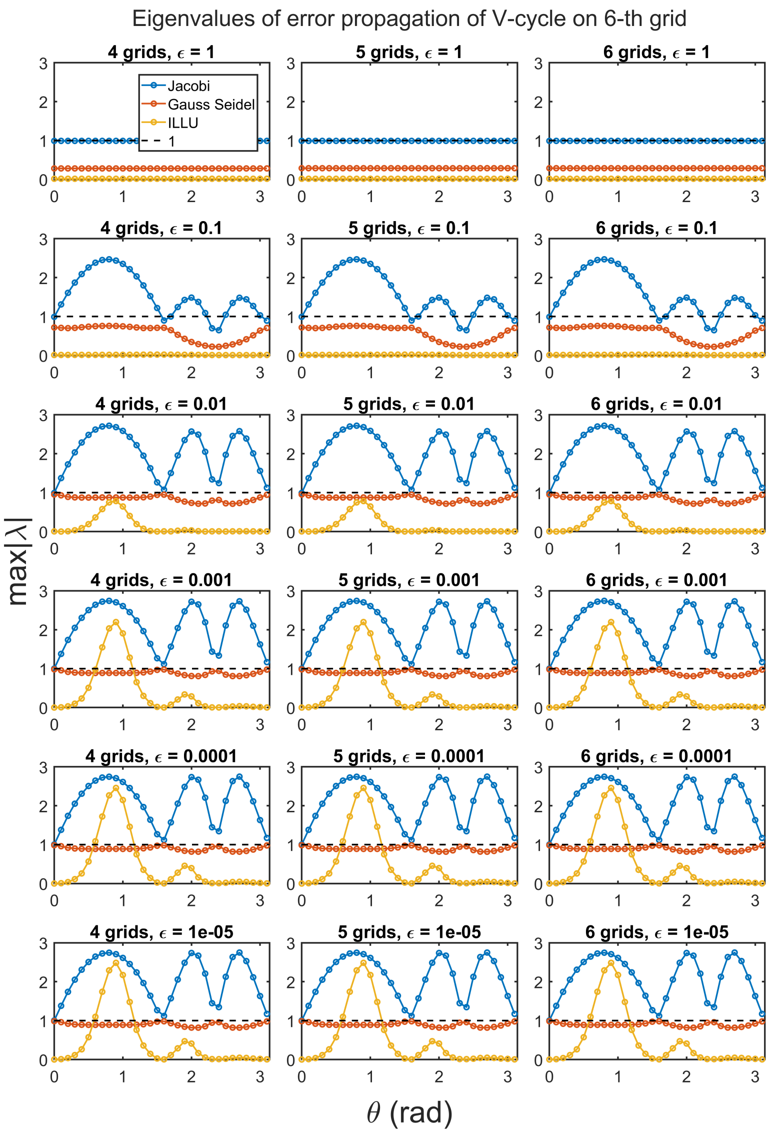

Figure 10: Eigenvalues of error propagation of multigrid method for anisotropic diffusion from Algorithm 8 on the 6-th grid, with different amount of grids in the V-cycle

$\{0, \pi/2, 3\pi/4, \pi\}$ there is a local maximum or local minimum. As explained in Section 3.3, this is because of the alignment with the grid. This shows that the multigrid method with the Gauss-Seidel smoother performs worse when the direction of anisotropy is aligned with the grid lines. The multigrid method with the Jacobi smoother, however, seems to perform better when this is the case.

Another observation is that the eigenvalues in general increase when a finer grid is used. The spectral radius does not show dependence on the amount of grids in this test.

4.5 Comparison with literature

As mentioned before, in the paper by Hemker [1] the Local Fourier Analysis (LFA) has been applied to the anisotropic diffusion problem as seen in Equation (16). The LFA has been applied to the multigrid method with ILLU and Gauss-Seidel smoothers. The local Fourier analysis (LFA) is a different technique than the eigenvalue analysis, but it is also related to the convergence of the method. In the paper, only the values for $\epsilon = 10^{-2}$ and $\epsilon = 10^{-4}$, with θ in steps between 0 and π radians.

To compare the results obtained here with those results, they have been plotted together in Figure 11. The figures show the eigenvalues as calculated before and the results of the referenced LFA. As can clearly be seen in both figures, the LFA and eigenvalue analysis do not match up, especially for the multigrid method with the ILLU smoother this is the case. For the multigrid method with the Gauss-Seidel smoother, the results obtained here and the results from [1] are similar but not the same either. There may be a couple of reasons for this.

The first and obvious difference is the method. The eigenvalue analysis and LFA are not the same. They are both used to show convergence. The Local Fourier Analysis is exact for rectangular grids with period boundary conditions [16]. In other cases, it is also a commonly used method to approximate the convergence factors of the multigrid method [16]. Even though the methods are not the same they should at least give similar results. This is not the case here. Thus, the difference in results should be sought elsewhere.

In the paper from Hemker, there are a lot of things left unclear. While the multigrid method and grid discretization are described properly and unambiguous, some things remain unclear. For instance, the number of grid refinements and the coarsest grid are unknown. Also, the way the smoothers and multigrid are implemented are not explained, and may thus vary from the implementation here. All these differences in implementation and calculation may explain the difference in results.

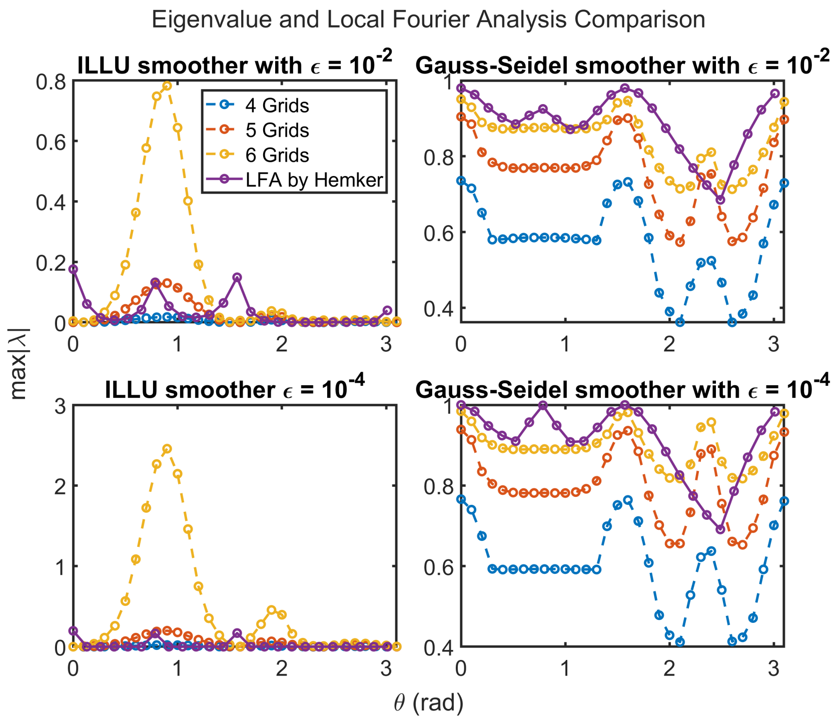

Figure 11: Comparison between the LFA by Hemker [1] and the eigenvalues from the error propagaion from Algorithm 6.

4.6 Proposed bound

We have seen that for values of $\epsilon < 0.01$ the multigrid method with the ILLU smoother on the 6-th grid does not converge for all values of θ. Here we will propose a bound on ϵ to ensure convergence of the multigrid method with the ILLU smoother for all values of θ on all grids.

To find the bound, we calculate the eigenvalue of the error propagation of the ILLU smoother. The eigenvalues are calculated on grids with an element size h ranging between 1 and 10^{-2}, ϵ ranges from 1 down to 10^{-6} and θ is fixed to $\pi/4$. The rotation is set to $\theta = \pi/4$ as the previous tests have shown that this results in the largest spectral radius.

In Figure 12 we can see the spectral radii as calculated. A line at $\epsilon = 0.05$ and

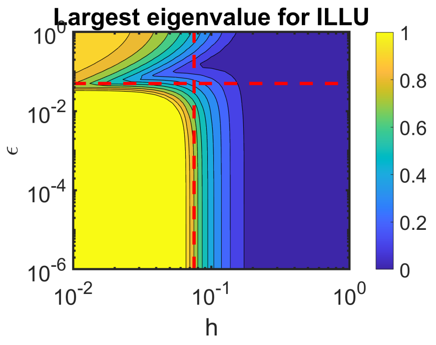

Figure 12: Proposed bound on ϵ and h to ensure convergence of multigrid method with ILLU smoother, with lines at $\epsilon = 0.05$ and $h = 0.075$.

at $h = 0.075$ have been added. These lines show the proposed bounds. Given a value of $\epsilon > 0.05$, or an element size $h > 0.075$, the ILLU method will converge for all values of θ.

Please note that this bound is estimated from numerical experiments and is not proven analytically.

5 Stable multigrid method

We have seen in Section 4, that the multigrid method applied to anisotropic diffusion problem does not always converge to the solution. This happens for small values of ϵ on fine grids. Therefore, we will introduce a variation of the multigrid method which is more stable.

Consider Algorithm 6, this algorithm solves the linear system of Equation (21) ($\mathbf{Su} = \mathbf{f}$). The stiffness matrix $\mathbf{S}$ depends on the grid, and values of θ and ϵ. So the stiffness matrix corresponding to ϵ on grid i can be denoted as $\mathbf{S}_{i,\epsilon}$. The linear system from Equation (21) is equal to

$$\mathbf{S}_{m,1}\mathbf{u} = \mathbf{f} - \left(\mathbf{S}_{m,\epsilon} - \mathbf{S}_{m,1}\right)\mathbf{u}.$$

This equation is solved for $\mathbf{u} = \mathbf{S}_{m,\epsilon}^{-1}\mathbf{f}$. This is exactly the required solution, while still using $\mathbf{S}_{m,1}$. This can be made iterative

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \tau\mathbf{S}_{m,1}^{-1}\left(\mathbf{f} - \mathbf{S}_{m,\epsilon}\mathbf{u}^k\right). \tag{35}$$

Upon convergence, the solution $\mathbf{u}^k$ converges to $\mathbf{S}_{m,\epsilon}^{-1}$.

5.1 Analysis of ideal iteration

Here we will discuss Equation (35), to discover why it should converge. The error propagation for this iterative scheme is:

$$\mathbf{e}^{k+1} = \left(I - \tau\mathbf{S}_{m,1}^{-1}\mathbf{S}_{m,\epsilon}\right)\mathbf{e}^k.$$

Since $\epsilon < 1$, we know that

$$\epsilon\left|\xi\right|^2 \leq \xi^T K \xi \leq \left|\xi\right|^2, \qquad \forall \xi \in \mathbb{R}^2,$$

With K being the diffusion tensor depending on ϵ from Section 3. As K is directly related to $\mathbf{S}_{m,\epsilon}$, we use this to infer

$$\epsilon\left(\mathbf{S}_{m,1}v, v\right) \leq \left(\mathbf{S}_{\epsilon,m}v, v\right) \leq \left(\mathbf{S}_{m,1}v, v\right), \qquad \forall v \in \mathbb{R}^{n_m^2}. \tag{36}$$

We know that the eigenvalues λ and eigenvectors v of $\mathbf{S}_{m,1}^{-1}\mathbf{S}_{m,\epsilon}$ satisfy $\mathbf{S}_{m,\epsilon}v = \lambda\mathbf{S}_{m,1}v$. Multiplying this expression with v^T and dividing by $v^T\mathbf{S}_{m,1}v$ gives

$$\lambda = \frac{\left(\mathbf{S}_{m,\epsilon}v, v\right)}{\left(\mathbf{S}_{m,1}v, v\right)}.$$

Using Equation (36) we get that the eigenvalues of $\mathbf{S}_{m,1}^{-1}\mathbf{S}_{m,\epsilon}$ are bounded such that $\epsilon \leq \lambda \leq 1$. This means that the eigenvalues of the error propagation are bounded such that

$$1 - \tau < \lambda < 1 - \epsilon\tau. \tag{37}$$

So with for instance $\tau = 1$, the iterative scheme Equation (35) converges.

5.2 Multigrid V-cycle

In general we do not have $\mathbf{S}_{m,1}^{-1}$. Therefore we approximate $\mathbf{S}_{m,1}^{-1}$ with the multigrid method. This can be seen in Algorithm 9, note that it calls VCycle2D1 which an be seen in Algorithm 10.

Algorithm 9 Stable multigrid method to solve Equation (19)

```
function StableMultigrid2D($m$, $tol$)
    $\mathbf{u}^{approx} = \mathbf{0}$
    while $\|\mathbf{f} - \mathbf{S}_{m,\epsilon}\mathbf{u}^{approx}\| \geq tol$ do
        $\mathbf{u}^{approx} = \mathbf{u}^{approx} + \text{VCycle2D1}\left(m, \mathbf{f} - \mathbf{S}_{m,\epsilon}\mathbf{u}^{approx}\right)$
    end while
    return $\mathbf{u}^{approx}$
end function
```

Algorithm 10 Step of the V-Cycle of the multigrid method to solve Equation (17) for $\epsilon = 1$

```
function VCycle2D1($i$, $\mathbf{f}_i$)
    if $i > 1$ then
        $\mathbf{u}^{pre} = \text{Smoother2D}\left(i, \mathbf{f}_i\right)$
        $\mathbf{u}^{sub} = \mathbf{u}^{pre} + P_i \text{VCycle2D1}\left(i-1, P_i^T\left(\mathbf{f}_i - \mathbf{S}_{i,1}\mathbf{u}^{pre}\right)\right)$
        $\mathbf{u}^{post} = \mathbf{u}^{sub} + \text{Smoother2D}\left(i, \mathbf{f}_i - \mathbf{S}_{i,1}\mathbf{u}^{sub}\right)$
    else
        $\mathbf{u}^{post} = \mathbf{S}_{1,1}^{-1}\mathbf{f}_i$
    end if
    return $\mathbf{u}^{post}$
end function
```

As before, we can calculate the error propagation

$$\mathbf{e}^{k+1} = \mathbf{e}^k - \tau \text{VCycle2D1}\left(m, \mathbf{S}_{m,\epsilon}\mathbf{e}^k\right).$$

Due to the linearity of the error propagation as proven in Section 4.2.2, we can define $\mathbf{A}$ as

$$\mathbf{e}^{k+1} = \left(I - \tau\mathbf{A}\mathbf{S}_{m,\epsilon}\right)\mathbf{e}^k.$$

By the linearity of the matrices there exist $\alpha_1, \alpha_2 > 0$, such that

$$\alpha_1 (\mathbf{A}v, v) \leq \left(\mathbf{S}_{m,\epsilon}^{-1} v, v\right) \leq \alpha_2 (\mathbf{A}v, v), \qquad \forall v \in \mathbb{R}^{n_m^2}$$

We can transform this to

$$\alpha_1 \leq \frac{\left(S_{m,\epsilon}^{-1} v, v\right)}{(\mathbf{A}v, v)} \leq \alpha_2.$$

By the definition of the Rayleigh quotient [11, Section 1.7] this holds for the eigenvalues of the eigenvalue problem $\mathbf{S}_{m,\epsilon}^{-1} v = \lambda \mathbf{A} v$. Thus for all eigenvalues of this problem, it holds that $\alpha_1 \leq \lambda \leq \alpha_2$. This leads to the eigenvalues of $\mathbf{AS}_{m,\epsilon}$ being bounded by $1/\alpha_1 < \lambda < 1/\alpha_2$. This gives us the following bounds on the eigenvalues for the error propagation $(I - \tau \mathbf{AS}_{m,\epsilon})$

$$1 - \tau\alpha_2 < \lambda < 1 - \tau\epsilon\alpha_1.$$

Now the value of τ can be chosen such that

$$-1 < 1 - \tau\alpha_2 < 1 - \tau\epsilon\alpha_1 < 1. \tag{38}$$

Then the method will converge.

5.3 Eigenvalue analysis

As before, we can calculate the eigenvalues using the function VCycle2D1. Again we will set the right-hand side to $\mathbf{0}$. Then eigenvalues can be calculated using the MATLAB function `eigs`. This time we call

$$\text{VCycle2D1}\left(m, -\mathbf{S}_{m,\epsilon}\mathbf{u}^k\right). \tag{39}$$

The resulting eigenvalues can be seen in Figure 13, Figure 14 and Figure 15. The first observation is that all the eigenvalues are smaller than 1. This means that Algorithm 9 will converge for all combinations of parameters tested here. As it is hard to see the exact values here, the overall maximal eigenvalue for each smoother method will also be listed here. For the multigrid method with the Gauss-Seidel smoother, the largest eigenvalue of the error propagation is 0.99777, for the ILLU smoother it is 0.99770, and for the Jacobi smoother it

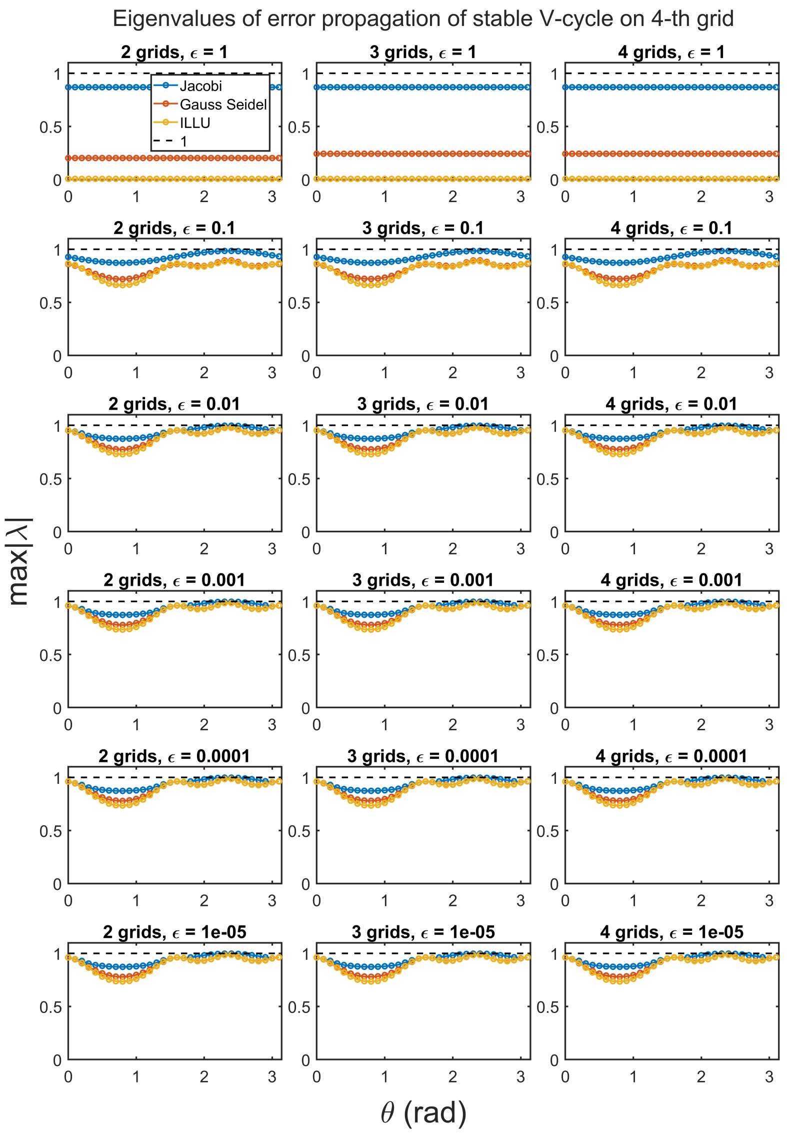

Figure 13: Eigenvalues for error propagation of Stable multigrid method for anisotropic diffusion from Algorithm 9 on the 4-th grid, with different number of grids in the V-cycle.

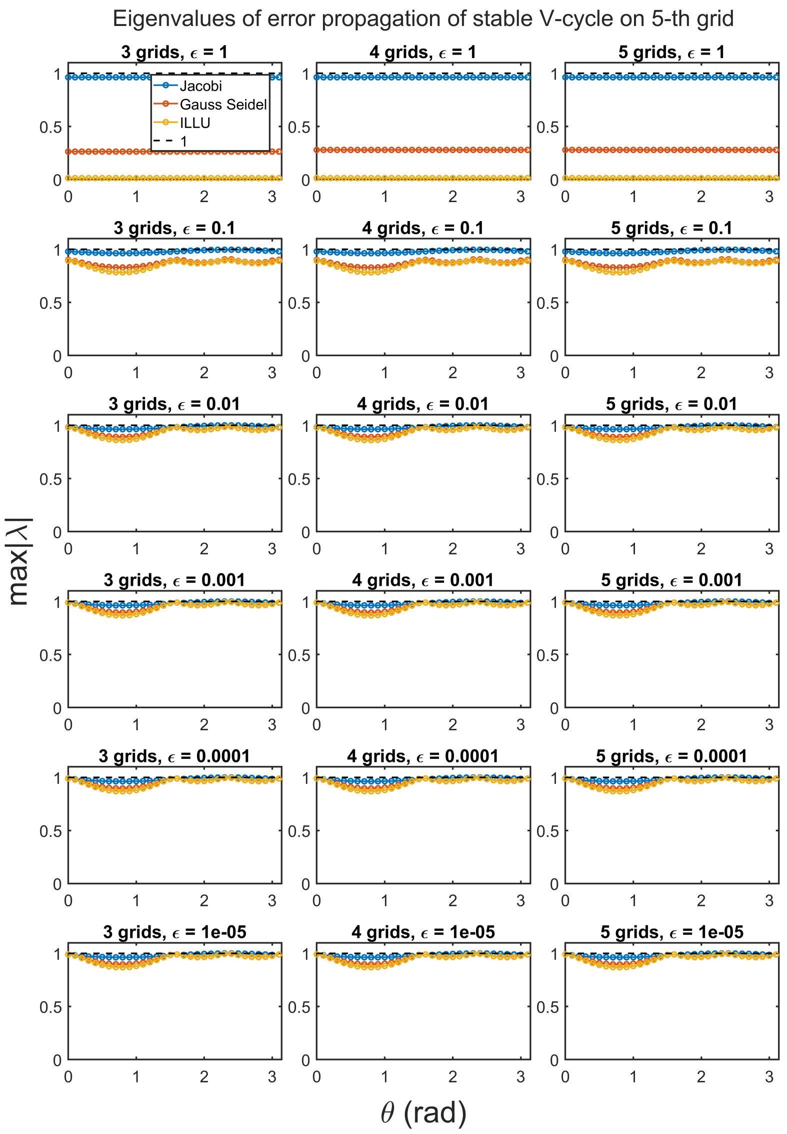

Figure 14: Eigenvalues for error propagation of Stable multigrid method for anisotropic diffusion from Algorithm 9 on the 5-th grid, with different number of grids in the V-cycle.

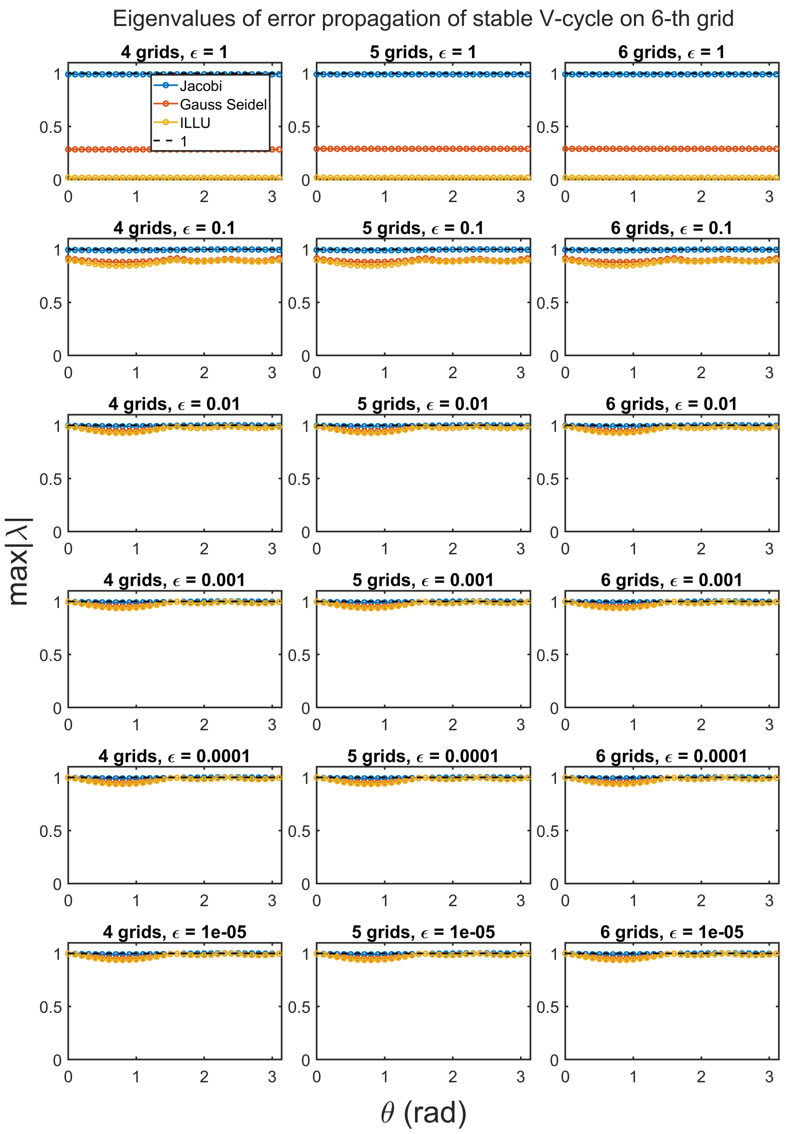

Figure 15: Eigenvalues for error propagation of Stable multigrid method for anisotropic diffusion from Algorithm 9 on the 6-th grid, with different number of grids in the V-cycle.

is 0.99997. These values are very close to 1. But these examples are the most negative case and are still strictly smaller than 1.

There is quite a difference between the eigenvalue when different smoothers are used in the multigrid method. The multigrid method with the Jacobi smoother always has the highest eigenvalues, while the multigrid method using the ILLU smoother always has the lowest eigenvalues.

For decreasing values of ϵ, the eigenvalues increase. However, for values of ϵ that are smaller than 10^{-3} there does not seem to be much difference between the eigenvalues visually. This is another sign that this variant of the multigrid method is stable.

Again, at θ in $\{0, \pi/2, 3\pi/4, \pi\}$ the eigenvalues achieve local maxima. As before, this is due to the grid structure as explained in Section 3.1. Also using finer grids increases the spectral radius, while adding more coarser grids does not do this.

5.4 Comparison

Now the results of eigenvalue analysis from the error propagation of Algorithm 5 and Algorithm 9 can be compared. The first will be called the "normal" multigrid method while the second will be called the "stable" multigrid method for now. The figures containing the results and the discussion can be seen in Section 4.4 and Section 5.3 respectively.

The main difference is that the eigenvalues of the stable multigrid method are always smaller than 1. This is not the case for the normal multigrid method. However, the stable multigrid method usually has slower convergence than the normal multigrid method.

In both the normal and stable case it can be seen that the methods with the Jacobi smoother perform the worst. Also the ILLU smoother performs better than the Gauss-Seidel smoother usually, but not always. For instance when $\epsilon = 10^{-5}, \theta = 0.4$, the finest grid is grid 6 and the coarsest grid is grid 1 in the V-cycle, then the multigrid method with the Gauss-Seidel smoother performs better.

Another similarity between the normal and stable multigrid method is that in both cases the eigenvalues are highly related to the grid alignment. This shows that it pays off to align the grid properly when working with anisotropic diffusion if this is possible. Also lower values of ϵ increase the eigenvalues in both cases.

6 Conclusion

The goal of this thesis was to quantify the convergence of the multigrid method with ILLU smoothers when applied to the anisotropic diffusion equation.

We have shown that we cannot guarantee convergence of the multigrid method with ILLU smoother for strongly elliptic problems. So instead we proposed a bound of $\epsilon = 0.05$. Anisotropic diffusion problems with $0.05 < \epsilon \leq 1$, should always converge for any rotation and grid size.

We have also shown that the Jacobi smoother in the standard multigrid iteration is unsuitable for solving this problem. However, the multigrid method with the Gauss-Seidel smoother seems to be always convergent for all values of anisotropy ϵ and rotation θ.

To deal with the multigrid method with the ILLU smoother diverging, we have proposed an alternative multigrid method. This ”stable” method always converges with either the Jacobi, Gauss-Seidel or ILLU smoother, although it may be relatively slow. The algorithm for this stable multigrid method can be seen in Algorithm 9.

7 Discussion

Even though these results help to get a better understanding of the usage of the multigrid method to solve problems with anisotropic diffusion, there is a lot more that can be researched. So here are some suggestions for future work extended on this report.

The grids used in the multigrid method were very coarse still. This was required for the eigenvalue calculation. Normally on the discretization considered here, one could just use direct solvers. The multigrid is generally used for larger linear systems. One could look at much finer discretizations and see what the convergence rates are in that case.

In Section 4.5 the results of the eigenvalue analysis are compared with the local Fourier analysis. There is a mismatch between the two methods. However, without diving deeper into the local Fourier analysis, the reason for this mismatch cannot be given. But it would be very interesting to see where this difference comes from. The source of this difference can give some insight into methods to analyse the performance of multigrid methods.

Another possibility is to research the eigenvalues more. In Section 4.2.2 the error propagation of the multigrid method is calculated explicitly. This calculation can be dissected to find out which parts of the multigrid method introduce the divergence. This information may then be used to improve the multigrid method applied to anisotropic diffusion.

In the eigenvalue analysis various parameter values for ϵ, θ and the number of grids have been used. But, there are more properties that could change the performance of the multigrid method. In this report, every smoother step executed only a single step of the iterative method. Changing the amount of smoother steps may change the results of the eigenvalue analysis. Also, other smoothers could be considered.

8 Bibliography

References

[1] P. W. Hemker. "Multigrid methods for problems with a small parameter in the highest derivative". In: Springer, Berlin, Heidelberg, 1984, pp. 106–121. DOI: 10.1007/BFb0099520. URL: http://link.springer.com/10.1007/BFb0099520.

[2] Wolfgang Hackbusch. *Multi-Grid Methods and Applications.* Vol. 4. Springer Series in Computational Mathematics. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985. ISBN: 978-3-642-05722-9. DOI: 10.1007/978-3-662-02427-0. URL: http://link.springer.com/10.1007/978-3-662-02427-0.

[3] P. Sonneveld, P. Wesseling, and P. de Zeeuw. "Multigrid and conjugate gradient methods as convergence acceleration techniques". In: (1985).

[4] O. Axelsson. "A general incomplete block-matrix factorization method". In: *Linear Algebra and its Applications* 74 (Feb. 1986), pp. 179–190. ISSN: 00243795. DOI: 10.1016/0024-3795(86)90121-7. URL: https://linkinghub.elsevier.com/retrieve/pii/0024379586901217.

[5] P. M. Zeeuw. "Incomplete Line LU as smoother and as preconditioner". In: *Incomplete Decomposition (ILU) — Algorithms, Theory, and Applications.* Wiesbaden: Vieweg+Teubner Verlag, 1993, pp. 215–224. DOI: 10.1007/978-3-322-85732-3{_}22. URL: http://link.springer.com/10.1007/978-3-322-85732-3_22.

[6] A. Reusken. "On a robust multigrid solver". In: *Computing* 56.3 (Sept. 1996), pp. 303–322. ISSN: 0010-485X. DOI: 10.1007/BF02238516. URL: http://link.springer.com/10.1007/BF02238516.

[7] J. Weickert et al. *Anisotropic Diffusion In Image Processing.* 1996. URL: https://www.researchgate.net/publication/202056812_Anisotropic_Diffusion_In_Image_Processing.

[8] U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid.* Academic Press, 2001, p. 631. ISBN: 9780127010700.

[9] Hong Luo, Joseph D. Baum, and Rainald Löhner. "A p-multigrid discontinuous Galerkin method for the Euler equations on unstructured grids". In: *Journal of Computational Physics* 211.2 (Jan. 2006), pp. 767–783. ISSN: 00219991. DOI: 10.1016/j.jcp.2005.06.019. URL: https://linkinghub.elsevier.com/retrieve/pii/S0021999105003128.

[10] Dietrich Braess. *Finite Elements.* Cambridge University Press, Apr. 2007. ISBN: 9780521705189. DOI: 10.1017/CBO9780511618635. URL: https://www.cambridge.org/core/product/identifier/9780511618635/type/book.

[11] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical Mathematics*. Vol. 37. Texts in Applied Mathematics. New York, NY: Springer New York, 2007. ISBN: 978-1-4757-7394-1. DOI: 10.1007/b98885. URL: http://link.springer.com/10.1007/b98885.

[12] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Vol. 15. Texts in Applied Mathematics. New York, NY: Springer New York, 2008. ISBN: 978-0-387-75933-3. DOI: 10.1007/978-0-387-75934-0. URL: http://link.springer.com/10.1007/978-0-387-75934-0.

[13] L. C. Evans. *Partial Differential Equations: Second Edition*. 2010. URL: https://bookstore.ams.org/gsm-19-r/.

[14] Guozhu Yu, Jinchao Xu, and Ludmil T. Zikatanov. "Analysis of a two-level method for anisotropic diffusion equations on aligned and nonaligned grids". In: *Numerical Linear Algebra with Applications* 20.5 (Oct. 2013), pp. 832–851. ISSN: 1099-1506. DOI: 10.1002/NLA.1847. URL: https://onlinelibrary.wiley.com/doi/full/10.1002/nla.1847%20https://onlinelibrary.wiley.com/doi/abs/10.1002/nla.1847%20https://onlinelibrary.wiley.com/doi/10.1002/nla.1847.

[15] C. Beaulieu. "The Biological Basis of Diffusion Anisotropy". In: *Diffusion MRI: From Quantitative Measurement to In vivo Neuroanatomy: Second Edition* (Jan. 2014), pp. 155–183. DOI: 10.1016/B978-0-12-396460-1.00008-1.

[16] Carmen Rodrigo, Francisco J. Gaspar, and Ludmil T. Zikatanov. "On the validity of the local Fourier analysis". In: (Oct. 2017). URL: http://arxiv.org/abs/1710.00408.

[17] Marcio Pinto et al. "On the Robustness of the xy-Zebra-Gauss-Seidel Smoother on an Anisotropic Diffusion Problem". In: *CMES* 117.2 (2018), pp. 251–270. DOI: 10.31614/cmes.2018.04237. URL: https://www.researchgate.net/publication/329249120.

[18] Alexandre Ern and Jean-Luc Guermond. *Finite Elements II*. Vol. 73. Texts in Applied Mathematics. Cham: Springer International Publishing, 2021. ISBN: 978-3-030-56922-8. DOI: 10.1007/978-3-030-56923-5. URL: https://link.springer.com/10.1007/978-3-030-56923-5.

[19] David Green et al. "An efficient high-order numerical solver for diffusion equations with strong anisotropy". In: *Computer Physics Communications* 276 (July 2022), p. 108333. ISSN: 00104655. DOI: 10.1016/j.cpc.2022.108333. URL: https://linkinghub.elsevier.com/retrieve/pii/S0010465522000510.