AILU: A Preconditioner Based on the Analytic Factorization of the Elliptic Operator

Gander, Martin Jakob; Nataf, F.

How to cite

GANDER, Martin Jakob, NATAF, F. AILU: A Preconditioner Based on the Analytic Factorization of the Elliptic Operator. In: Numerical linear algebra with applications, 2000, vol. 7, n° 7-8, p. 543–567. doi: 10.1002/1099-1506(200010/12)7:7/8<505::aid-nla210>3.0.co;2-z

This publication URL: https://archive-ouverte.unige.ch/unige:6298

Publication DOI: 10.1002/1099-1506(200010/12)7:7/8<505::aid-nla210>3.0.co;2-z

Numerical Linear Algebra with Applications, Vol. 1(1), 1–7 (1900)

AILU: A Preconditioner Based on the Analytic Factorization of the Elliptic Operator

Martin J. Gander and Frederic Nataf

Department of Mathematics, McGill University, Montreal, Canada and CMAP, CNRS UMR7641, Ecole Polytechnique, Palaiseau, France

We investigate a new type of preconditioner for large systems of linear equations stemming from the discretization of elliptic symmetric partial differential equations. Instead of working at the matrix level, we construct an analytic factorization of the elliptic operator into two parabolic factors and we identify the two parabolic factors with the LU factors of an exact block LU decomposition at the matrix level. Since these factorizations are nonlocal, we introduce a second order local approximation of the parabolic factors. We analyze the approximate factorization at the continuous level and optimize its performance which leads to the new AILU (Analytic ILU) preconditioner with convergence rate $1 - O(h^{1/3})$ where h denotes the mesh size. Numerical experiments illustrate the effectiveness of the new approach.

KEY WORDS Block preconditioner, ILU, analytic parabolic factorization, frequency filtering

1. Introduction

Given a second order linear elliptic operator $\mathcal{L}(u)$ acting on $u : \mathbb{R}^n \longrightarrow \mathbb{R}$, $n = 2, 3$ we are interested to solve the elliptic partial differential equation

$$\mathcal{L}(u) = f \tag{1.1}$$

in a given domain $\Omega \subset \mathbb{R}^n$ with appropriate boundary conditions. Discretizing the elliptic operator with a finite element or finite difference method on a structured mesh, we obtain a large system of linear equations

$$K\boldsymbol{u} = \boldsymbol{f} \tag{1.2}$$

where the discrete elliptic operator K has the block structure

$$K = \begin{bmatrix} D_1 & L_{1,2} & & \\ L_{2,1} & D_2 & \ddots & \\ & \ddots & \ddots & L_{n-1,n} \\ & & L_{n,n-1} & D_n \end{bmatrix}. \tag{1.3}$$

1070–5325/00/010001–07$8.50

Received 30 November 1999
Revised 7 April 2000

The diagonal blocks D_i represent in our notation the discretization of the y part (and z in three dimensions) of the elliptic operator $\mathcal{L}$ and include on the diagonal a part of the discretization of the x part of the elliptic operator, the rest being contained in $L_{i,j}$. Since K is sparse the linear system $K\boldsymbol{u} = \boldsymbol{f}$ is usually solved by an iterative method and it is necessary to precondition the system to obtain the solution efficiently, because the condition number of K grows like $\frac{1}{h^2}$, h being the mesh parameter.

There are two main groups of matrix based preconditioners: approximate inverses and incomplete factorizations [4]. Both techniques do not relate to the underlying differential operator in general and can thus not take advantage of the underlying partial differential equation. Algebraic multilevel methods are also matrix based, but they try to sense the underlying problem using several levels, which can lead to very effective preconditioners. We derive here a new incomplete factorization preconditioner based on the underlying partial differential equation by factorizing the differential operator itself into two parabolic factors before it is discretized. We then construct local approximations to the non-local parabolic factors to obtain a preconditioner. The continuous analysis permits the optimization of the preconditioner for the given elliptic partial differential equation.

Although the parabolic factorization of elliptic operators has been a topic of interest for a while [7,15] the first use of this approach in an iterative fashion to solve a large system of linear equations was proposed by Nataf in [13] and extended by Nataf, Loheac and Schatzman in [14]. The idea was also picked up by Giladi and Keller to solve a convection dominated convection diffusion equation arising in an asymptotic analysis in [8]. The main difficulties remaining in this approach are the low quality of the approximate factorization and thus the limited applicability. In all the previous work the factored operator was non symmetric and the approximate factorization was only considered for small diffusion coefficients which simplifies this type of approximation. We consider here symmetric operators and we establish a link between the analytic factorization and the exact block LU decomposition which leads to approximate factorizations of high quality. Our approach is related to earlier work at the discrete level by Wittum in [19,20] extended later by Wagner [16,17] and Buzdin [5]. In addition one of the referees pointed out an internal report [6] in which asymptotic results identical to ours are obtained through a very different, fully discrete analysis.

We begin in Section 2. by factorizing the continuous elliptic operator into two parabolic factors. We then extend this factorization to the semi-discrete case and show the equivalence of the parabolic factorization and the exact block LU decomposition in Theorem 2.1.. In Section 3. we show how the parabolic factorization can be approximated to construct a new type of preconditioner we call AILU (Analytic ILU). We show how the performance of the preconditioner can be optimized and prove in Theorem 3.1. that AILU has as a stationary iterative method an asymptotic convergence rate of $1-O(h^{\frac{2}{3}})$ which gives together with a Krylov method $1-O(h^{\frac{1}{3}})$. This is in contrast to matrix based ILU preconditioners which can not improve the asymptotic convergence rate $1-O(h^2)$ of the associated stationary iterative method and give with Krylov acceleration only a convergence rate of $1-O(h)$. In Section 4. we show how boundary conditions of the original problem can be treated avoiding the shooting method used traditionally with the parabolic factorization [8]. Finally in Section 5. we show numerical experiments in two and three space dimensions confirming the theoretical results developed in the previous sections.

2. Analytic Parabolic Factorization

Given an elliptic operator $\mathcal{L}(u)$ we write the operator as a product of two parabolic operators,

$$\mathcal{L}(u) = (\partial_x + \Lambda_1) \circ (\partial_x - \Lambda_2)(u) \tag{2.1}$$

where Λ_1 and Λ_2 are positive operators and thus the first factor represents a parabolic operator acting in the positive x direction and the second one a parabolic operator acting in the negative x direction.

In the sequel we restrict ourselves for the analysis to the case of $\mathcal{L} = (\eta - \Delta)$, where Δ denotes the Laplacian in two dimensions and $\eta \geq 0$. The analysis in three dimensions is similar and we will show numerical results for both the two and three dimensional case.

Our results are based on Fourier analysis. We denote the Fourier transform $\hat{f}(k)$ of $f(y) : \mathbb{R} \longrightarrow \mathbb{R}$ by

$$\hat{f}(k) = \mathcal{F}_y(f)(k) := \int_{-\infty}^{\infty} e^{-iky} f(y) dy$$

and the inverse Fourier transform of $\hat{f}(k)$ by

$$f(y) = \mathcal{F}_y^{-1}(\hat{f})(y) := \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{iky} \hat{f}(k) dk.$$

Our analysis will also involve the Fourier transform of a convolution operator with kernel $h(y)$,

$$\Lambda(u)(y) := \int_{-\infty}^{\infty} h(y - z) u(z) dz$$

whose Fourier transform is given by

$$\mathcal{F}_y(\Lambda(u))(k) = \hat{h}(k)\, \hat{u}(k)$$

or equivalently with $\hat{\Lambda}(k) := \hat{h}(k)$,

$$\Lambda(u) = \mathcal{F}_y^{-1}(\hat{\Lambda}(k)\hat{u}(k)).$$

The function $\hat{\Lambda}(k)$ is called the symbol of the operator Λ. For example, the symbol of the operator $-\partial_{yy}$ is the polynomial k^2. More generally, the symbol of any constant coefficient differential operator is a polynomial in the Fourier variable k and conversely.

The symbol of the composition of two convolution operators is the product of their symbols since

$$\Lambda_1 \circ \Lambda_2(u) = \mathcal{F}_y^{-1}(\hat{\Lambda}_1(k)\hat{\Lambda}_2(k)\hat{u}(k))$$

and we write this relation in the sequel in the compact form $\mathcal{F}_y(\Lambda_1 \circ \Lambda_2) = \hat{\Lambda}_1 \hat{\Lambda}_2$.

Lemma 2.1. (Continuous Parabolic Factorization) *The linear operator $\mathcal{L} = (\eta - \Delta)$ admits the continuous parabolic factorization*

$$(\eta - \Delta) = -(\partial_x + \Lambda_1) \circ (\partial_x - \Lambda_2) \tag{2.2}$$

where $\Lambda_1 = \Lambda_2 = \mathcal{F}_y^{-1}(\sqrt{k^2 + \eta})$ are pseudo-differential operators in y.

Proof We take a Fourier transform of $\mathcal{L}$ in y

$$\mathcal{F}_y(\eta - \Delta) = -\partial_{xx} + k^2 + \eta$$

and factor this operator in the transformed domain

$$-\partial_{xx} + k^2 + \eta = -(\partial_x + \sqrt{k^2+\eta})(\partial_x - \sqrt{k^2+\eta}). \tag{2.3}$$

The inverse Fourier transform gives the factorization (2.2) where both factors are parabolic because Λ_j, $j = 1, 2$ are positive operators. ∎

To relate this parabolic factorization to the exact block LU decomposition of the discrete matrix operator, we discretize the x direction of $(\eta - \Delta)$ and compute the analytic factorization (2.3) for the semi discrete operator $(\eta - \Delta_h)$. We have

$$\Delta_h = D_x^- D_x^+ + \partial_{yy} \tag{2.4}$$

where for a vector $\boldsymbol{u}$ the difference operators $D_x^+(\boldsymbol{u})_i := (u_{i+1} - u_i)/h$ and $D_x^-(\boldsymbol{u})_i := (u_i - u_{i-1})/h$, represent the discrete derivatives on a given structured mesh.

Lemma 2.2. (Semi-Discrete Parabolic Factorization) *The semi-discrete operator (2.4) admits the semi-discrete parabolic factorization in the Fourier transformed domain*

$$\mathcal{F}_y(\eta - \Delta_h) = -\left(D_x^- + (\hat{T}h - \frac{1}{h})\right) \frac{1}{h^2\hat{T}} \left(D_x^+ - (\hat{T}h - \frac{1}{h})\right). \tag{2.5}$$

where the pseudo-differential operator T has the symbol

$$\hat{T} = \frac{1}{h^2} + \frac{\eta + k^2}{2} + \frac{1}{2h}\sqrt{(\eta+k^2)^2h^2 + 4(\eta+k^2)}, \tag{2.6}$$

Proof We take the Fourier transform in y of $\eta - \Delta_h$

$$\mathcal{F}_y(\eta - \Delta_h) = -(D_x^- D_x^+ - (\eta + k^2))$$

and look for a factorization of the form

$$-(D_x^- D_x^+ - (\eta+k^2)) = -\frac{1}{h^2\hat{T}}(D_x^- + \hat{\Lambda}_1)(D_x^+ - \hat{\Lambda}_2) = -\frac{1}{h^2\hat{T}}(D_x^- D_x^+ - \hat{\Lambda}_2 D_x^- + \hat{\Lambda}_1 D_x^+ - \hat{\Lambda}_1\hat{\Lambda}_2),$$

with the unknowns $\hat{\Lambda}_1$, $\hat{\Lambda}_2$ and the additional free parameter $\hat{T}$ due to the discretization. Using $D_x^+ - D_x^- = hD_x^- D_x^+$ to replace the term with D_x^- we obtain

$$\mathcal{F}_y(\eta - \Delta_h) = -\frac{1}{h^2\hat{T}}\left((1+\hat{\Lambda}_2 h)D_x^- D_x^+ + (\hat{\Lambda}_1 - \hat{\Lambda}_2)D_x^+ - \hat{\Lambda}_1\hat{\Lambda}_2\right) \equiv -(D_x^- D_x^+ - (\eta+k^2)).$$

Comparing coefficients in the identity above we obtain the system of equations

$$\hat{\Lambda}_1 = \hat{\Lambda}_2, \quad \frac{1 + h\hat{\Lambda}_2}{h^2\hat{T}} = 1, \quad \frac{\hat{\Lambda}_1\hat{\Lambda}_2}{h^2\hat{T}} = \eta + k^2$$

to solve for the unknowns $\hat{\Lambda}_1$, $\hat{\Lambda}_2$ and $\hat{T}$. Substituting the first and second equation into the third one, we find for $\hat{T}$ the quadratic

$$\hat{T}^2 - \left(\eta + k^2 + \frac{2}{h^2}\right)\hat{T} + \frac{1}{h^4} = 0 \tag{2.7}$$

whose solution is precisely (2.6) where we chose the positive root, since we defined $\hat{\Lambda}_1$ and $\hat{\Lambda}_2$ to be positive operators. Substituting $\hat{T}$ back into the second equation we find indeed

$$\hat{\Lambda}_1 = \hat{\Lambda}_2 = \hat{T}h - \frac{1}{h} = h\frac{\eta + k^2}{2} + \frac{1}{2}\sqrt{(\eta + k^2)^2 h^2 + 4(\eta + k^2)}$$

which are positive. The semi discrete analytic parabolic factorization is thus given by (2.5). ■

Note that as we take the limit for $h \longrightarrow 0$ in (2.5) we recover again the continuous parabolic factorization (2.3) since the middle term disappears in the limit ($h^2\hat{T} \longrightarrow 1$ as $h \longrightarrow 0$). For discrete problems it is however important to include the middle factor, which was not the case in previous work on continuous parabolic factorizations.

We now relate the analytic parabolic factorization to the exact block LU decomposition of the discrete elliptic operator K defined in (1.3). This step is important because it allows us to treat general boundary conditions unlike [14] and avoids the shooting method used in [8].

Lemma 2.3. (Block LU Decomposition) *A symmetric positive definite matrix K of the form (1.3) admits the exact block LU decomposition*

$$K = \begin{bmatrix} T_1 & & & \\ L_{2,1} & T_2 & & \\ & \ddots & \ddots & \\ & & L_{n,n-1} & T_n \end{bmatrix} \begin{bmatrix} T_1^{-1} & & & \\ & T_2^{-1} & & \\ & & \ddots & \\ & & & T_n^{-1} \end{bmatrix} \begin{bmatrix} T_1 & L_{1,2} & & \\ & T_2 & \ddots & \\ & & \ddots & L_{n-1,n} \\ & & & T_n \end{bmatrix} \tag{2.8}$$

where the matrices T_i are given by the recurrence relation

$$T_i = \begin{cases} D_1 & i = 1, \\ D_i - L_{i,i-1}T_{i-1}^{-1}L_{i-1,i} & 1 < i \leq n. \end{cases} \tag{2.9}$$

Proof The proof follows by inserting (2.9) into (2.8) and using the positive definiteness of K, see for example Wagner [16]. ■

To relate this decomposition to the semi-discrete parabolic factorization (2.5), we formulate the exact block LU decomposition for $\eta - \Delta_h$ on an infinite mesh in the x component of the operator and Fourier transform the y component to compare with the parabolic factors in (2.5). We obtain the infinite matrix

$$\hat{K} = \begin{bmatrix} \ddots & \ddots & & \\ \ddots & \hat{D} & \hat{L} & \\ & \hat{L} & \hat{D} & \ddots \\ & & \ddots & \ddots \end{bmatrix}$$

where the entries are given by

$$\hat{D} = \eta + k^2 + \frac{2}{h^2}, \quad \hat{L} = -\frac{1}{h^2}.$$

The exact block LU decomposition of $\hat{K}$ is given by

$$\hat{K} = \begin{bmatrix} \ddots & & & \\ \ddots & \hat{T}_\infty & & \\ & \hat{L} & \hat{T}_\infty & \\ & & \ddots & \ddots \end{bmatrix} \begin{bmatrix} \ddots & & & \\ & \hat{T}_\infty^{-1} & & \\ & & \hat{T}_\infty^{-1} & \\ & & & \ddots \end{bmatrix} \begin{bmatrix} \ddots & \ddots & & \\ & \hat{T}_\infty & \hat{L} & \\ & & \hat{T}_\infty & \ddots \\ & & & \ddots \end{bmatrix} \tag{2.10}$$

where $\hat{T}_\infty$ is a solution of

$$\hat{T}_\infty = \hat{D} - \hat{L}\hat{T}_\infty^{-1}\hat{L}, \tag{2.11}$$

which is also the limit of the recurrence relation (2.9). Note that we talk about a block decomposition here, because the scalar entries in the matrix contain Fourier components which will become matrices, once the y direction is discretized. We have the following

Theorem 2.1. (Equivalence of Block LU and Parabolic Factorization) *The block LU decomposition (2.10) of the semi-discretized elliptic operator $\eta - \Delta_h$ on an unbounded domain is identical to the analytic parabolic factorization (2.5) of the same operator and tends to the parabolic factorization (2.3) of the continuous elliptic operator in the limit as h goes to zero.*

Proof Since $\hat{T}_\infty$ satisfies (2.11) it is a solution of

$$\hat{T}_\infty^2 - (\eta + k^2 + \frac{2}{h^2})\hat{T}_\infty + \frac{1}{h^4} = 0, \tag{2.12}$$

the same quadratic found in the analytic parabolic factorization (2.7). Hence $\hat{T}_\infty = \hat{T}$ given by (2.6). The exact block LU factorization of K in the y-Fourier transformed domain is therefore

$$\hat{K} = \begin{bmatrix} \ddots & & & \\ \ddots & \hat{T} & & \\ & \hat{L} & \hat{T} & \\ & & \ddots & \ddots \end{bmatrix} \frac{1}{\hat{T}} \begin{bmatrix} \ddots & \ddots & & \\ & \hat{T} & \hat{L} & \\ & & \hat{T} & \ddots \\ & & & \ddots \end{bmatrix}.$$

By rewriting each entry $\hat{T}$ in the form $1/h^2 + \hat{T} - 1/h^2$ and observing that $\hat{L} = -1/h^2$, each row of the first matrix in the above factorization can be expressed using a finite difference operator, $D_x^-/h + \hat{T} - 1/h^2$. Similarly for the last matrix in the above factorization, we find $-D_x^+/h + \hat{T} - 1/h^2$ for each row. Hence the above factorization can be written entirely in terms of finite differences,

$$\hat{K} = \left(\frac{D_x^-}{h} + \hat{T} - \frac{1}{h^2}\right) \frac{1}{\hat{T}} \left(-\frac{D_x^+}{h} + \hat{T} - \frac{1}{h^2}\right) \tag{2.13}$$

which coincides after rearranging with the analytic parabolic factorization found in (2.5) and hence establishes convergence to the parabolic factorization (2.3) as h goes to zero. ■

Note that the exact block LU decomposition of the matrix K has motivated many people to construct preconditioners before. The first such method called ILLU was proposed by Kettler in [12] where he kept the non-local operators (non-local due to the inverse in the recurrence relation (2.9)) local by neglecting the non-local components. This approach has

been investigated further by Hemker [9] Axelson and Eijkhout [2] and Axelson, Eijkhout, Polman and Vassilevski [3]. Our goal here is to construct the best possible local approximation in the sense of performance of the method by using the relationship with the analytic parabolic factorization.

3. The AILU Preconditioner

One could use the parabolic factorization given in (2.5) to solve the original problem (1.2). Instead of solving the linear system, one would have to solve two lower dimensional parabolic problems, one in the positive and one in the negative x direction, corresponding to a forward and a backward solve of the block LU decomposition. This is however not advisable since the parabolic factorization contains nonlocal operators in y corresponding to dense sub-blocks T_i in the block LU decomposition. We therefore approximate the parabolic factorization by local operators and use the factorization as a preconditioner corresponding to a new type of ILU preconditioner.

Definition 3..1 (AILU Preconditioner) *We call the factored linear operator with the symbol*

$$\hat{\mathcal{L}}_{app} := -\left(D_x^- + (\hat{T}_{app}h - \frac{1}{h})\right)\frac{1}{h^2\hat{T}_{app}}\left(D_x^+ - (\hat{T}_{app}h - \frac{1}{h})\right) \tag{3.1}$$

where $\hat{T}_{app}$ is a local approximation to $\hat{T}$ defined in (2.6) an AILU (Analytic ILU) preconditioner.

Note that the pseudo-differential operator T with the symbol $\hat{T}$ defined in (2.6) is nonlocal, since it involves a convolution in real space. A polynomial approximation $\hat{T}_{app}$ of $\hat{T}$ however would be local, since powers of k correspond to derivatives in real space. As a first representative of the AILU preconditioner class, we replace the nonlocal pseudo-differential operator T by a second order differential approximation whose symbol is

$$\hat{T}_{app} = \frac{1}{h^2} + \frac{\eta + k^2}{2} + \frac{1}{2h}(p + qk^2), \quad p, q > 0. \tag{3.2}$$

This leads to a classical linear second order parabolic problem since k^2 corresponds to a second derivative in y. Note that we did not include a first order term because the underlying problem is symmetric. The parameters p and q are used to optimize the performance of the AILU preconditioner. We insert the approximation $\hat{T}_{app}$ into the factorization (3.1) and obtain the operator

$$\hat{\mathcal{L}}_{app} = -D^-D^+ + \hat{T}_{app} + \frac{1}{\hat{T}_{app}h^4} - \frac{2}{h^2}$$

which approximates the original operator $\hat{\mathcal{L}} = \eta + k^2 - D_x^+ D_x^-$. Now the stationary iterative scheme to solve $\mathcal{L}(u) = f$ using the easy to invert approximate operator $\mathcal{L}_{app}$ is given by

$$\mathcal{L}_{app}(u^{n+1}) = f + \mathcal{L}_{app}(u^n) - \mathcal{L}(u^n). \tag{3.3}$$

By linearity of the operators, the error e^n satisfies

$$e^{n+1} = \mathcal{L}_{app}^{-1}(\mathcal{L} - \mathcal{L}_{app})e^n = (1 - \mathcal{L}_{app}^{-1}\mathcal{L})e^n. \tag{3.4}$$

We want to choose the parameters p and q so that the iterative method converges as fast as possible. For this purpose, we formulate a minimization problem on p and q. Since we consider here an operator discretized in x and continuous in y, we look at the error reduction for the family of vectors $e_{k_x,k} = (e^{ik_x(jh)}e^{iky})_{j\in\mathbb{Z}}$, $i^2 = -1$. Inserting each vector of the familiy into (3.4), we obtain by how much this vector is reduced in one iteration,

$$e^{n+1}_{k_x,k} = \rho(k_x,k,p,q,\eta,h)e^n_{k_x,k} \tag{3.5}$$

with the corresponding damping factor

$$\rho(k_x,k,p,q,\eta,h) = \frac{\hat{T} + \frac{1}{\hat{T}h^4} - (\hat{T}_{app} + \frac{1}{\hat{T}_{app}h^4})}{4\frac{\sin^2(k_xh/2)}{h^2} + \hat{T}_{app} + \frac{1}{\hat{T}_{app}h^4} - \frac{2}{h^2}}.$$

To get fast convergence for all vectors $e_{k_x,k}$ in the family, we have to choose p and q so that ρ is small uniformly for all $e_{k_x,k}$ relevant to our problem. A direct computation shows that

$$\hat{T}_{app} + \frac{1}{\hat{T}_{app}h^4} - \frac{2}{h^2} = \frac{1}{\hat{T}_{app}}\,\frac{(k^2\,h + h\,\eta + q\,k^2 + p)^2}{4h^2} > 0$$

and since

$$4\frac{\sin^2(k_xh/2)}{h^2} \geq 0 \tag{3.6}$$

we obtain a uniform bound on the convergence rate with respect to k_x,

$$|\rho(k_x,k,p,q,\eta,h)| \leq |\rho(0,k,p,q,\eta,h)|.$$

It suffices therefore to minimize the modulus of $\rho(0,k,p,q,\eta,h)$ to minimize the convergence rate and thus optimize the performance of the preconditioner. We obtain after a short calculation

$$\rho(0,k,p,q,\eta,h) = 1 - \frac{2(\eta+k^2)(2+\eta h^2 + ph + h(h+q)k^2)}{(p+\eta h + (q+h)k^2)^2}. \tag{3.7}$$

Remark For particular domains and boundary conditions, the term in (3.6) can be bounded by a positive quantity away from zero related to the lowest mode in x direction. This can be taken into account in the optimization to be closer to the actual optimal performance in the numerical setting and we do this in our numerical experiments.

Since the computations are done with a fully discretized operator, the range of the frequency parameter k is not arbitrary. It is bounded from below by a lowest frequency dependent on the size of the domain in y direction and the boundary conditions, $k^2 > k^2_{\min}$ and from above, k is bounded by the mesh size h_y in y direction, $k^2 < k^2_{max} := (\pi/h_y)^2$.

For given p and q, it is thus possible to estimate the convergence rate of the iterative scheme (3.3) by

$$\max_{k_{\min}<k<k_{\max}} \left| 1 - \frac{2(\eta+k^2)(2+\eta h^2 + ph + h(h+q)k^2)}{(p+\eta h + (q+h)k^2)^2} \right|.$$

To optimize performance we need to choose p and q so that they minimize this expression. Hence the optimal parameters p and q are solutions of the min-max problem

$$\min_{p,q>0} \left(\max_{k_{\min}<k<k_{\max}} \left| 1 - \frac{2(\eta+k^2)(2+\eta h^2 + ph + h(h+q)k^2)}{(p+\eta h + (q+h)k^2)^2} \right| \right) \tag{3.8}$$

where $k_{\min}$ and $k_{\max}$ denote the minimal and maximal frequencies relevant to the problem.

Lemma 3.1. *The solution $p, q > 0$ of the min-max problem (3.8) is given by the solution of the nonlinear system of equations*

$$\rho(0, k_{\min}, p, q, \eta, h) = -\rho(0, k_e, p, q, \eta, h) = \rho(0, k_{\max}, p, q, \eta, h). \tag{3.9}$$

Proof For k fixed, the convergence rate ρ grows monotonically in the parameters $p, q > 0$,

$$\frac{\partial \rho}{\partial p} = 2\frac{(\eta + k^2)(h^2\eta + h^2k^2 + hp + hqk^2 + 4)}{(\eta h + k^2 h + p + qk^2)^3} > 0, \quad \frac{\partial \rho}{\partial q} = k^2 \frac{\partial \rho}{\partial p} \geq 0. \tag{3.10}$$

Hence one can use p and q to balance ρ about zero to minimize its modulus. Using p and q to set ρ to zero for two particular frequencies k_1 and k_2 with $k_{\min} \leq k_1 < k_2 \leq k_{\max}$ we are led to the system of equations

$$p + qk_j^2 = \sqrt{(\eta + k_j^2)^2 h^2 + 4(\eta + k_j^2)}, \quad j = 1, 2 \tag{3.11}$$

obtained from equating $\hat{T}$ given by (2.6) with its approximation $\hat{T}_{app}$ given by (3.2). We find p and q as functions of k_1 and k_2 to be

$$p(k_1, k_2) = \frac{\frac{1}{k_1^2}\sqrt{(\eta + k_1^2)^2 h^2 + 4(\eta + k_1^2)} - \frac{1}{k_2^2}\sqrt{(\eta + k_2^2)^2 h^2 + 4(\eta + k_2^2)}}{\frac{1}{k_1^2} - \frac{1}{k_2^2}} \tag{3.12}$$

$$q(k_1, k_2) = \frac{\sqrt{(\eta + k_1^2)^2 h^2 + 4(\eta + k_1^2)} - \sqrt{(\eta + k_2^2)^2 h^2 + 4(\eta + k_2^2)}}{k_2^2 - k_1^2}. \tag{3.13}$$

Since the function $K \to K^2\sqrt{(\eta + K^{-2})^2 h^2 + 4(\eta + K^{-2})}$ is increasing monotonically as one sees by differentiation we have $p \geq 0$ as required in the optimization. Similarly the function $k \to \sqrt{(\eta + k^2)^2 h^2 + 4(\eta + k^2)}$ is growing monotonically and we obtain $q \geq 0$. Thus there exist p and q in the optimization such that the convergence rate ρ has both signs depending on k. Now ρ has at most one extremum in k, found by differentiation at

$$k_e^2 := \frac{2p - 4q\eta + (p^2 - 2\eta - \eta pq)h + \eta(p - q\eta)h^2}{2q + (\eta q^2 - pq + 2)h - (p + q\eta)h^2}.$$

This extremum must lie between k_1 and k_2 and thus $\rho(k_{\min})$ and $\rho(k_{\max})$ have the same sign which is the opposite of the sign of $\rho(k_e)$. Since ρ is monotone in p and q the modulus of ρ is minimized if the positive maximum of ρ and the negative minimum of ρ are equal in modulus,

$$\max(|\rho(k_{\min})|, |\rho(k_{\max}|) = |\rho(k_e)|.$$

To see that $\rho(k_{\min}) = \rho(k_{\max})$ at the optimum suppose this is not the case and without loss of generality assume that at the optimum we have $\rho(k_{\min}) > \rho(k_{\max}) > 0$. Then arbitrary small changes δp and δq in p and q lead to an arbitrary small change in $\rho(k_{\max})$ that is unimportant since $\rho(k_{\min}) > \rho(k_{\max}) > 0$. The change in $\rho(k_{\min})$ is given by

$$\delta\rho(k_{\min}) = \delta p \frac{\partial \rho}{\partial p}(k_{\min}) + \delta q \frac{\partial \rho}{\partial q}(k_{\min}).$$

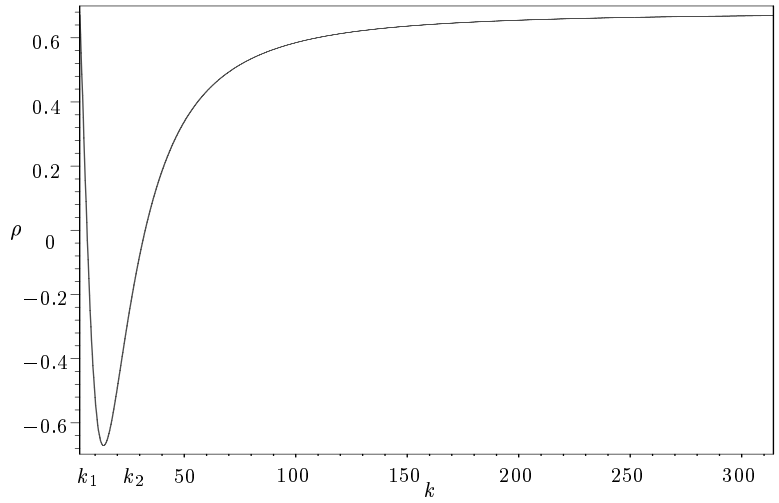

Figure 1. Optimized convergence rate $|\rho(k)|$ for problem parameters $\eta = 0$ and $h = 1/100$.

By (3.10), we have

$$\delta\rho(k_{\min}) = (\delta p + k_{\min}^2 \delta q)\frac{\partial \rho}{\partial p}(k_{\min}). \tag{3.14}$$

As for the extremum between k_1 and k_2, it is now located in $k_e + \delta k_e$ and has changed by an amount $\delta\rho_e$

$$\delta\rho_e = \delta k_e \, \frac{\partial \rho}{\partial k}(k_e) + \delta p \, \frac{\partial \rho}{\partial p}(k_e) + \delta q \, \frac{\partial \rho}{\partial q}(k_e)$$

Since k_e is an extremum for ρ, we have $\frac{\partial \rho}{\partial k}(k_e) = 0$. Noticing once more (3.10), we have then

$$\delta\rho_e = (\delta p + k_e^2 \, \delta q) \, \frac{\partial \rho}{\partial p}(k_e). \tag{3.15}$$

From (3.10), (3.14) and (3.15) we can decrease p and increase q such that $\rho(k_e)$ increases, while $\rho(k_{\min})$ decreases. But this contradicts that we had an optimum before. ■

Remark We have chosen to use the semi continuous approach to determine the optimal parameters p and q for its generality and simplicity. A similar approach has been used successfully in domain decomposition methods, see for example [1], [10] or [18]. Figure 4 shows that the semi continuous approach yields values for p and q which are very close to the optimal ones in a fully discrete setting. A fully discrete analysis would be possible as well: for each discretization scheme for the Laplace operator one would need to compute the damping factor for the eigenvectors of the discrete scheme to define an optimization problem for p and q. But such an analysis would need to be done for each scheme and could only give minor improvements over the semi continuous one. Figure 1 shows the optimized convergence rate in modulus $|\rho(k)|$ for problem parameters $\eta = 0$ and $h = 1/100$. The optimal parameters are found to be $p = 10.66$ and $q = 0.05230$ which leads to a convergence rate bounded by $|\rho(k)| < 0.6702$ over all frequencies $k \in [\pi, \frac{\pi}{h}]$. In our example we find $k_1 = 6.395$ and $k_2 = 32.47$. Thus for those two frequencies, our preconditioner is exact. Instead of optimizing, one could choose directly two frequencies and thus obtain a

smoother or a rougher if desired. This observation relates our preconditioner directly to the frequency filtering decomposition by Wittum [20], which was developed on a fully discrete basis and is exact for two test vectors to be chosen by the user. Further development in that direction was done by Wagner who introduced the tangential frequency filtering decomposition in [16] and extended it to the non-symmetric case in [17]. Their work has as a goal a sequence of preconditioners which leads to an iterative method with convergence rates independent of h at the cost of having to add more and more preconditioners as h is refined. Our focus is on the construction of one preconditioner with the best possible performance.

To estimate the convergence rate as a function of the mesh size, we need to know the behavior of p and q as the mesh is refined and h goes to zero. This result is contained in the following

Lemma 3.2. *The optimal coefficients p and q admit the asymptotic expansion*

$$\begin{aligned} p &= (\eta + k_{\min}^2)^{\frac{1}{3}} h^{-\frac{1}{3}} + O(h^{\gamma}), \quad \gamma > -\tfrac{1}{3} \\ q &= \frac{1}{2}(\eta + k_{\min}^2)^{-\frac{1}{3}} h^{\frac{1}{3}} + O(h^{\delta}), \quad \delta > \tfrac{1}{3} \end{aligned} \tag{3.16}$$

as h goes to zero.

Proof From numerical experiments we see that p increases when h decreases whereas q decreases with decreasing h. We therefore make the Ansatz

$$p := Ch^{\alpha},\ \alpha < 0 \quad q := Dh^{\beta},\ \beta > 0$$

where C and D are some constants. Inserting this Ansatz into the system of equations (3.9) and collecting the low order terms in h we find

$$\begin{aligned} 4D^2h^{2\beta}\pi^4(1 + 2D^2h^{2\beta}\eta - 2CDh^{\alpha+\beta}) \quad &+ \quad \dots \quad = \quad 0 \\ -4D^2h^{2\beta}\pi^2(\eta + k_{\min}^2) - 2Dh^{\beta+1}\pi^4(\eta CDh^{\alpha+\beta} - C^2h^{2\alpha}) - & \\ 2Dh^{\beta+1}\pi^4(D^2h^{2\beta}\eta k_{\min}^2 - CDh^{\alpha+\beta}k_{\min}^2 + 4\eta + 4k_{\min}^2) \quad &+ \quad \dots \quad = \quad 0 \end{aligned} \tag{3.17}$$

We thus obtain for α and β by balancing the exponents of the lowest order terms in (3.17) the system of equations

$$\alpha + \beta = 0, \quad 2\beta = \beta + 1 + 2\alpha,$$

with the solution $\alpha = -\frac{1}{3}$ and $\beta = \frac{1}{3}$. For the constants C and D we get the system

$$2C^2D - 4D^2(\eta + k_{\min}^2) = 0, \quad 1 - 2CD = 0,$$

with the solution

$$C = (\eta + k_{\min}^2)^{\frac{1}{3}}, \quad D = \frac{1}{2(\eta + k_{\min}^2)^{\frac{1}{3}}}$$

which concludes the proof. ∎

Theorem 3.1. (Convergence Rate) *A stationary iterative method with AILU and optimized parameters p and q has an asymptotic convergence rate*

$$|\rho| < 1 - O(h^{\frac{2}{3}})$$

as h goes to zero.

Proof We insert p and q from (3.16) into the convergence rate ρ and expand for small h the maximum of ρ attained at $k_{\min}$. We find

$$\rho(k_{\min}) = 1 - 4(\eta + k_{\min}^2)^{\frac{1}{3}} h^{\frac{2}{3}} + O(h^{\frac{4}{3}})$$

which confirms the result. ∎

Having found the optimal parameter p and q, we accelerate the stationary iterative method by applying conjugate gradients, which is equivalent to using the approximate factorization as a preconditioner. The conjugate gradient algorithm improves again the asymptotic convergence rate by a square-root, so that AILU as a preconditioner for conjugate gradients leads to an asymptotic convergence rate

$$|\rho| < 1 - O(h^{\frac{1}{3}}). \tag{3.18}$$

These results are confirmed by the numerical experiments in Section 5..

4. Treating Boundary Conditions

So far our analysis was limited to unbounded domains. Since real calculations are on bounded domains, we need to be able to include the effect of boundary conditions. We do this by using the equivalence between the analytic parabolic factorization and the exact matrix factorization stated in Theorem 2.1..

Instead of approximating the square-root in the limiting $\hat{T}$ (2.6) by a quadratic, we would need to approximate on bounded domains the non-local operators T_i given by the recurrence relation (2.9). After a Fourier transform in y, we obtain for the semi discrete operators the recurrence relation for the symbols,

$$\hat{T}_i = \begin{cases} \eta + k^2 + \frac{2}{h^2} & i = 1, \\ \eta + k^2 + \frac{2}{h^2} - \hat{T}_{i-1}^{-1}/h^4 & 1 < i \leq n. \end{cases} \tag{4.1}$$

Thus for each i one would need to solve an optimization problem to find the optimal parameters p_i and q_i. To see why this overhead can be avoided, we first analyze how the iterates of the recurrence relation (4.1) converge to the limit (2.12). Note that we are not talking about the convergence of AILU here, we are analyzing the effects of imposed boundary conditions on the AILU preconditioner.

Lemma 4.1. *For fixed $k > 0$ the asymptotic convergence rate of (4.1) is linear.*

Proof The recurrence relation (4.1) is a fixed point iteration of the form

$$\hat{T}_i = g(\hat{T}_{i-1}).$$

Since $g'(\hat{T}_\infty)$ does not vanish the asymptotic convergence rate is linear with the factor

$$\gamma := g'(\hat{T}_\infty) = \frac{4}{\left(2 + \eta h^2 + h^2 k^2 + h\sqrt{(k^2+\eta)(\eta h^2 + 4 + h^2 k^2)}\right)^2} < 1.$$

∎

Note how γ decreases as the frequency parameter k increases. While for a low frequency, say $k = \pi$ and $\eta = 0$, $h = 1/100$ the asymptotic convergence rate is $\gamma = 0.93$, for the

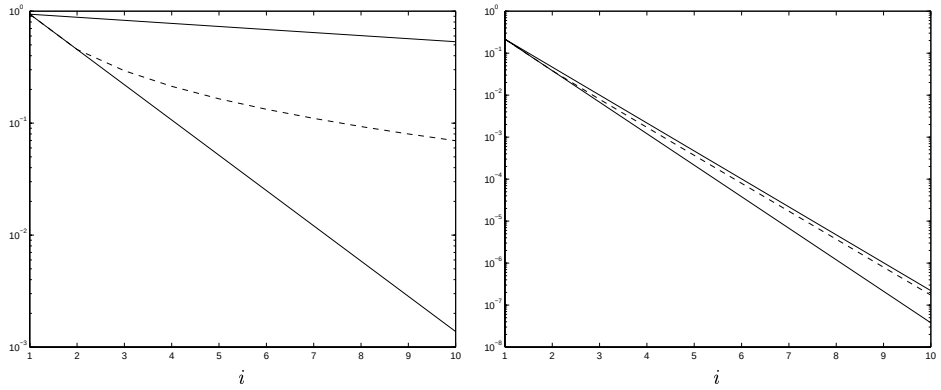

Figure 2. Convergence behavior of the exact factorization including the boundary conditions to the factorization on the unbounded domain as one leaves the boundary for two frequencies $k = \pi$ and $k = \pi/(4h)$ and $\eta = 0$ and $h = 1/100$. The dashed line shows the exact factorization and the solid lines the initial and the asymptotic convergence rates.

largest frequency $k = \pi/h$ it is $\gamma = 0.0072$. In addition the initial convergence is faster than the asymptotic one. Defining the initial convergence rate by

$$\gamma_{ini} := \frac{|\hat{T}_2 - \hat{T}_\infty|}{|\hat{T}_1 - \hat{T}_\infty|}$$

we find

$$\gamma_{ini} = \frac{\eta^2 h^4 + 2\eta h^4 k^2 + 4\eta h^2 + k^4 h^4 + 4k^2 h^2 + 3 - \hat{T}_\infty(h^4\eta + h^4 k^2 + 2h^2)}{(\eta h^2 + k^2 h^2 + 2)(\eta h^2 + k^2 h^2 + 2 - \hat{T}_\infty h^2)}.$$

Figure 2 shows for two frequency parameters k the convergence behavior of the exact factorization including the boundary conditions to the factorization on the unbounded domain as one leaves the boundary. The exact factorization is shown as a dashed line and the initial and asymptotic convergence rates are shown as solid lines. For all except the very low frequencies, the factorization result of the unbounded domain is an excellent approximation to the factorization including the boundary effects a few steps away from the boundary.

This is the motivation for not doing extra optimizations at each step of the recurrence relation. We only optimize the limiting case and use the optimal parameters obtained from the limiting case up to the boundary. To do this, we work with the frequencies k_1 and k_2 for which our factorization is exact. Note that by equation (3.11) every pair of optimized parameters p and q corresponds to two frequencies k_1 and k_2 for which the factorization is exact. For these two frequencies, we can use the recurrence relation (4.1) to obtain the exact operator at that frequency. To remain exact with our approximate factorization at those frequencies, we have to equalize our approximation and the exact factorization at those frequencies, which leads to the system of equations

$$\begin{aligned} \frac{1}{h^2} + \frac{\eta + k_1^2}{2} + \frac{1}{2h}(p_i + q_i k_1^2) &= \hat{T}_i(k_1) \\ \frac{1}{h^2} + \frac{\eta + k_2^2}{2} + \frac{1}{2h}(p_i + q_i k_2^2) &= \hat{T}_i(k_2) \end{aligned}$$

	Iteration count					Solution process only in mega flop			
h	CG	ILU('0')	ILU(1e-4)	Iter	AILU	CG	ILU('0')	ILU(1e-4)	AILU
1/100	221	103	10	48	24	61.8	43.2	16.2	12.6
1/200	451	204	18	82	32	504.9	342.5	125.0	67.1
1/300	683	306	25	113	39	1720.7	1156.3	399.7	183.7
1/400	917	407	33	140	44	4107.3	2734.4	948.9	368.1
1/600	1388	-	-	192	53	13989.2	-	-	996.8
1/800	1862			239	60	33363.8			2005.1
1/1000	2339			283	66	65486.1			3444.9

Table 1. Laplaces equation in two dimensions. AILU as iterative solver (Iter) and as preconditioner (AILU).

for the parameters p_i and q_i. Note that $\hat{T}_i(k_j)$, $j = 1, 2$ can easily be obtained using the recurrence relation (4.1) which is scalar for each frequency k_1 and k_2. We thus construct a preconditioner which is exact for two given frequencies k_1 and k_2 and those frequencies give the best convergence in the case of unbounded domains. Since we do not optimize the boundary part, we expect a small loss in the asymptotic convergence rate. The numerical experiments in the following section confirm that the loss is indeed minor.

5. Numerical Experiments

We first consider a two dimensional problem,

$$\eta u - \partial_x(a(x,y)\partial_x u) - \partial_y(b(x,y)\partial_y u) = f(x,y), \quad 0 < x, y < 1$$

with homogeneous Dirichlet boundary conditions. To conform with the analysis, we first show results for the constant coefficient case $a = b = 1$, $\eta = 0$ and $f(x,y) = 0$. We start the iteration with an initial guess $u^{(0)} = 1$ and we compare the performance to unpreconditioned conjugate gradient and the preconditioners ILU(0) and ILU(1e-4) using conjugate gradient and a tolerance of 1e-6. The choice of the drop tolerance in ILU(1e-4) was motivated by [4] where it lead to the best performance on a similar problem. Table 1 shows the results obtained from numerical experiments for different mesh parameters h and includes a column 'Iter' which shows the performance of AILU as an iterative solver. The new AILU preconditioner shows an excellent reduction in the iteration count, almost as good as ILU(1e-4). In fact it will beat any ILU with a drop tolerance since it performs asymptotically better, as one can see from comparing the case $h = 1/100$ and $h = 1/400$. Unfortunately we could not compute ILU for bigger problems. With the drop tolerance, we ran out of memory and the zero fill in version we used from Matlab took an enormous amount of time, even though the flop counts remained reasonable. Comparing the flop count AILU is by far the most efficient solver, even for small problems.

In Table 2 we show the cost of the preconditioner in mega flop. Clearly the AILU preconditioner is much less expensive than the ILU(1e-4). Its cost is comparable to ILU('0').

Figure 3 shows the asymptotic behavior of the different preconditioners tested. While matrix based preconditioners reduce the iteration count compared to the unpreconditioned CG, their asymptotic behavior is the same as for unpreconditioned conjugate gradient.

h	Preconditioner cost in mega flop		
	ILU('0')	ILU(1e-4)	AILU
1/100	0.09	45.98	0.12
1/200	0.36	348.67	0.48
1/300	0.81	1142.60	1.09
1/400	1.44	2667.92	1.93
1/600	-	-	4.34
1/800			7.70
1/1000			12.03

Table 2. Cost to compute the different preconditioners for the Laplace operator in two dimensions.

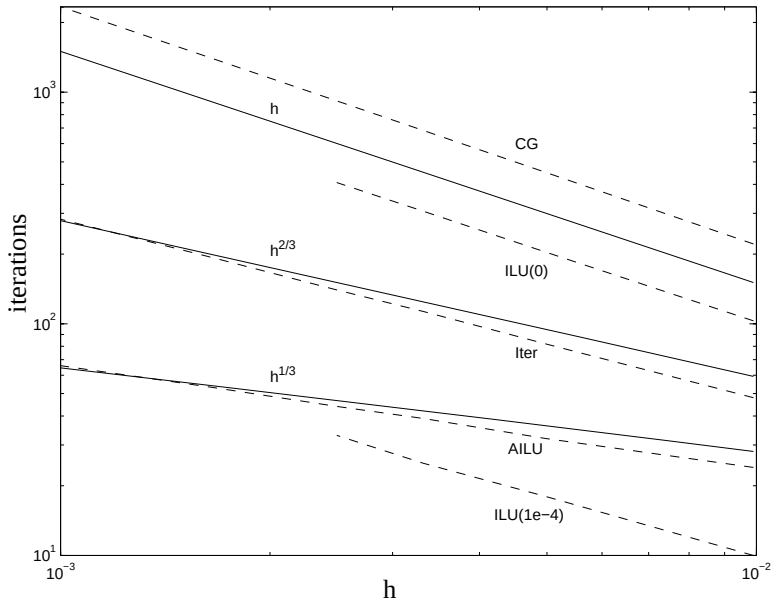

Figure 3. Asymptotic convergence behavior of the different preconditioners compared to AILU with asymptotic rate close to $1 - O(h^{1/3})$.

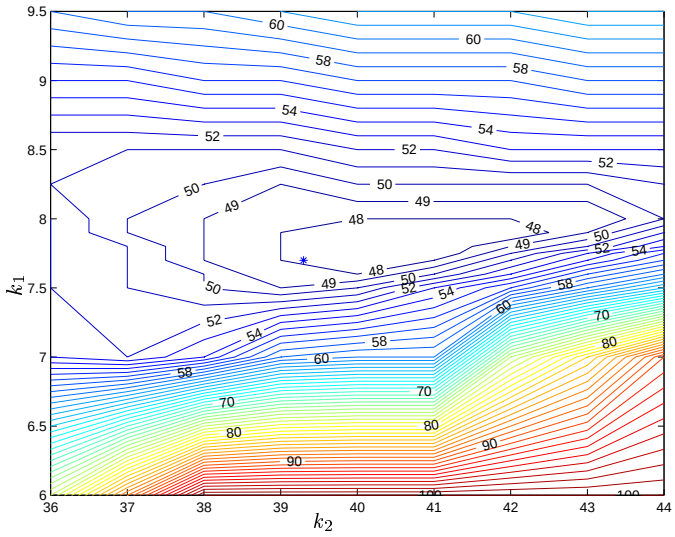

Figure 4. Comparison of the performance of the optimized frequencies k_1 and k_2 marked with a star (*) to other choices of frequencies where AILU is exact. The level sets correspond to the iteration count of AILU as an iterative solver.

AILU however clearly also reduces the iteration count asymptotically to almost $1-O(h^{\frac{1}{3}})$. Fitting the curve gives an asymptotic rate of $1-O(h^{0.4})$.

To test how effective the optimization is, we performed a complete set of experiments with varying frequencies k_1 and k_2 for which AILU is exact on a grid to see where the algorithm really converges fastest. Figure 4 shows a level set plot of the results obtained for the Laplacian with $h = 1/100$, the same test that can be found in Table 1 on the first line. AILU was used as an iterative solver here. The optimization does indeed lead to parameters k_1 and k_2 which are very close to the best performing numerical ones.

As a next example we choose varying coefficients $a(x,y) = x+1/2$, $b(x,y) = 3/2-y$, $f(x,y) = 0$ and $\eta = 0$ and start with an initial guess $u^{(0)} = 1$. With slowly variable coefficients it suffices to optimize using the average of the coefficients and the thus obtained constant coefficient preconditioner. Otherwise our analysis can be applied locally to frozen coefficients, an approach currently under investigation. Table 3 shows the results obtained from numerical experiments for different mesh parameters. The new AILU preconditioner shows a good reduction of the iteration count. In addition it permits the solution of the given problem in less flops than any of the other methods tested. For big problems the savings become significant. Table 4 compares the cost of the computation of the preconditioner. Although AILU performs in the solution process better than ILU(1e-4), the cost of the AILU preconditioner is negligible compared to the solution cost in contrary to ILU(1e-4). Note that for bigger problems we were not able to compute ILU('0') and ILU(1e-4) because of time and memory limitations.

As a next example we consider a three dimensional model problem,

$$\eta u - \partial_x(a(x,y,z)\partial_x u) - \partial_y(b(x,y,z)\partial_y u)\partial_z(c(x,y,z)\partial_z u) = f(x,y,z), \quad 0 < x,y,z < 1$$

again with homogeneous Dirichlet boundary conditions. Like in two dimensions we start

	Iteration count				Solution process only in mega flop			
h	CG	ILU('0')	ILU(1e-4)	AILU	CG	ILU('0')	ILU(1e-4)	AILU
1/100	434	126	11	31	121.3	52.8	18.4	16.5
1/200	901	256	18	45	1008.2	429.8	129.8	95.9
1/300	1372	389	26	55	3455.3	1469.7	431.8	263.5
1/400	1853	523	33	63	8297.6	3513.3	986.5	536.3
1/600	2827	-	-	76	28487.1	-	-	1454.7

Table 3. A variable coefficient case in two spatial dimensions.

	Preconditioner cost in mega flop		
h	ILU('0')	ILU(1e-4)	AILU
1/100	0.089	48.19	0.591
1/200	0.358	365.26	2.362
1/300	0.807	1194.20	5.313
1/400	1.436	2775.76	9.444
1/600	-	-	21.246

Table 4. Cost of constructing the preconditioner for the variable coefficient case in two dimensions.

first with a constant coefficient problem, $a = b = c = 1$. We use $f(x, y, z) = 0$, $\eta = 0$ and start the iteration with the initial guess $u^{(0)} = 1$. We compare the new preconditioner with unpreconditioned CG, ILU('0') and ILU(1e-4) as before using a tolerance of 1e-6. Table 5 shows the iteration and the flop count for five different mesh parameters. The last two problems were too big to compute ILU(1e-4) on our workstation, and for the last one ILU('0') could not be obtained either with Matlab due to time limitations. Note that in three dimensions, AILU is very close in iteration count to ILU(1e-4) and outperforms it flop-wise. Note also that ILU('0') performs very well flop-wise as does unpreconditioned CG due to the eigenvalue clustering in the model problem. Nevertheless for large problems AILU is gaining on all other methods due to the asymptotic behavior. Table 6 shows again the cost of the preconditioner in mega flop. Note that we have chosen here to solve the parabolic problems in AILU exactly by factorization to conform with the analysis. From

	Iteration count					Solution process only in mega flop			
h	CG	ILU('0')	ILU(1e-4)	Iter	AILU	CG	ILU('0')	ILU(1e-4)	AILU
1/16	45	23	6	12	9	4.8	3.8	6.4	4.5
1/29	85	41	10	18	13	59.3	44.6	92.0	51.3
1/35	104	49	11	21	15	130.0	95.6	191.8	108.1
1/55	168	77	-	29	18	843.3	603.5	-	476.8
1/100	317	-		45	25	10122.8	-		5131.0

Table 5. Laplaces equation in three dimensions.

h	Preconditioner cost in mega flop		
	ILU('0')	ILU(1e-4)	AILU
1/16	0.05	87.48	0.40
1/29	0.34	2288.30	5.60
1/35	0.61	6096.90	12.24
1/55	2.46	-	94.75
1/100	-		1258.22

Table 6. Cost of computing the preconditioner for the three dimensional Laplacian.

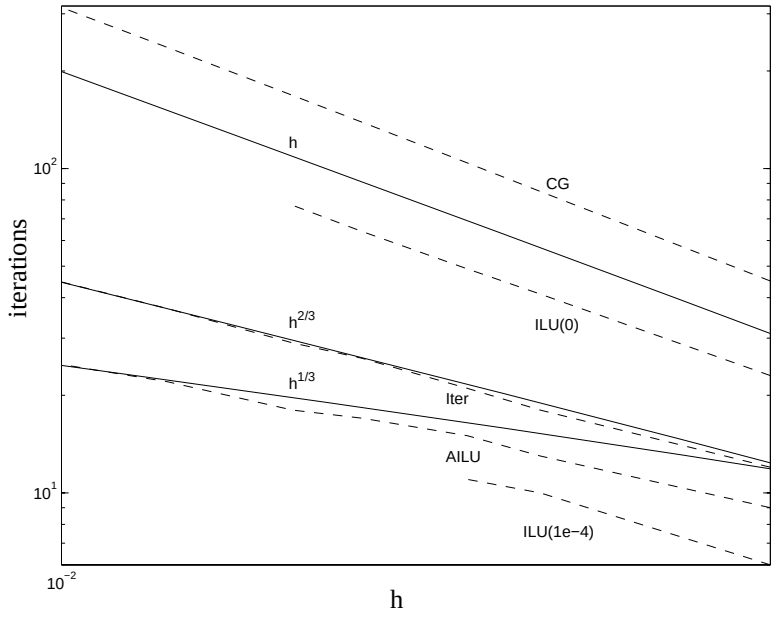

Figure 5. Asymptotic convergence behavior of the different preconditioners compared to AILU with asymptotic rate $1 - O(h^{1/3})$ in three dimensions as well.

a certain problem size on, it will be better to solve the parabolic problems inside AILU iteratively to reduce the flop count. In addition the parabolic problems need not be solved very accurately, since they describe themselves only an approximation.

Figure 5 shows again the asymptotic performance of AILU compared to the other methods tested for Laplaces equation in three dimensions. As in two dimensions, we see that AILU improves greatly the asymptotic performance of the matrix based preconditioners. Used as an iterative solver it is very close to the asymptotic result $1 - O(h^{\frac{2}{3}})$ and together with the conjugate gradient method it approaches $1 - O(h^{\frac{1}{3}})$.

To conclude we show a variable coefficient case in three dimensions. We chose $a = 0.5 + x, b = 1.5 - y^2$ and $c = 3.5/(z+3)$, $\eta = 0$ and $f(x, y, z) = 0$ and start the iteration with the initial guess $u^{(0)} = 1$. As before we compare with the matrix based incomplete

	Iteration count				Solution process only in mega flop			
h	CG	ILU('0')	ILU(1e-2)	AILU	CG	ILU('0')	ILU(1e-2)	AILU
1/16	85	28	17	14	9.0	4.6	3.6	5.2
1/29	168	52	32	18	116.8	56.5	45.2	54.0
1/35	207	63	38	20	258.1	122.7	96.6	113.4
1/46	283	84	51	22	819.9	380.2	302.0	312.5
1/55	344	102	61	24	1724.0	798.5	625.8	638.7
1/76	486	-	84	29	6535.3	-	2317.4	2271.8

Table 7. Variable coefficients in three dimensions.

	Preconditioner cost in mega flop		
h	ILU('0')	ILU(1e-2)	AILU
1/16	0.1	4.9	0.6
1/29	0.3	106.2	6.9
1/35	0.6	278.6	14.6
1/46	1.4	278.6	43.6
1/55	2.5	2788.6	103.9
1/76	-	14354.1	386.7

Table 8. Comparison of the cost of the preconditioner for the variable coefficients case in three dimensions.

factorizations, but this time we used ILU with a higher drop tolerance to be able to deal with larger problems. Table 7 shows the iteration and the flop count for six different mesh parameters. The last problem was too big to compute ILU('0') on our workstation, even though again the flop count is low in Matlab. Note that we changed the drop tolerance in ILU from 1e-4 to 1e-2 to be able to compare bigger problems (using ILU(1e-4) led to similar results as in Table 5). This explains the much lower iteration count of AILU compared to ILU(1e-2). Again for large problems AILU is gaining on all other methods due to its asymptotic superiority.

In Table 8 we show again the cost of the preconditioner for the variable coefficient case. Even though we factorize the parabolic problems within AILU exactly, the AILU preconditioner is much less expensive than ILU(1e-2), but their performance as a solver is comparable.

6. Conclusions

We have derived a new block ILU preconditioner for linear algebra problems stemming from symmetric positive definite elliptic partial differential equations. The new preconditioner AILU is able to improve the asymptotic convergence behavior in contrast to other ILU preconditioners. We achieved this result by relating the matrix factorization to an analytic factorization of the underlying partial differential equation and thus we were able to capture the essentials of the factorization at the continuous level. This led to the precon-

ditioner AILU which has the same sparsity pattern as other block ILU preconditioners but optimal performance for that pattern. Numerical experiments illustrated the analysis.

The new link between the block decomposition and the continuous parabolic factorization could open up the path for this type of preconditioner to a large range of practical applications. So far such preconditioners could only be constructed based on the block structure of the matrix and thus the approach was limited to problems on regular grids and regular domains. The present analysis shows that it suffices to be able to solve parabolic problems on the given grid, and first parabolic finite element solvers in space time have been developed [11].

REFERENCES

1. Y. Achdou and F. Nataf. Preconditionners for the mortar method based on local approximations of the Steklov-Poincaré operator. *Math. Models Methods Appl. Sci.*, 5(7):967–997, 1995.
2. O. Axelson and V. Eijkhout. Robust vectorizable preconditioners for three-dimensional elliptic difference equations with anisotropy. In *Algorithms and applications on vector and parallel computers*. North Holland, 1987.
3. O. Axelson, V. Eijkhout, B. Polman, and P. Vassilevski. Incomplete block-matrix factorization iterative methods for convection-diffusion problems. *BIT*, 29:867–889, 1989.
4. M. Benzi and M. Tuma. A comparative study of sparse approximate inverse preconditioners. *Appl. Num. Math.*, 30:305–340, 1999.
5. A. Buzdin. Tangential decomposition. *Computing*, 61:257–276, 1998.
6. A. Buzdin and G. Wittum. Zwei-Frequenzenzerlegung. Technical report, IWR, Universität Heidelberg, INF 368, 1999.
7. J. F. Claerbout. *Fundamentals of geophysical data processing with applications to petroleum prospecting*. McGraw-Hill, 1976.
8. E. Giladi and J. B. Keller. Iterative solution of elliptic problems by approximate factorization. *Journal of Computational and Applied Mathematics*, 85:287–313, 1997.
9. P. Hemker. Multigrid methods for problems with a small parameter in the highest derivate. In *Numerical Analysis Proceedings, Dundee*. Springer Verlag, Berlin, 1983.
10. C. Japhet. Optimized Krylov-Ventcell method. application to convection-diffusion problems. In P. E. Bjørstad, M. Espedal, and D. Keyes, editors, *Domain Decomposition Methods in Sciences and Engineering*. John Wiley & Sons, 1997. Proceedings from the Ninth International Conference, June 1996, Bergen, Norway.
11. C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Cambridge University Press, 1992.
12. R. Kettler. Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods. In *Multigrid Methods, Proceedings, Köln-Porz*, pages 502–534. Springer-Verlag, 1982.
13. F. Nataf. Résolution de l'equation de convection-diffusion stationaire par une factorisation parabolique. *C. R. Acad. Sci., Paris*, I 310(13):869–872, 1990.
14. F. Nataf, J. Loheac, and M. Schatzman. Parabolic approximation of the convection-diffusion equation. *Math. Comp.*, 60:515–530, 1993.
15. J. Stoer and R. Bulirsch. *Numerical Analysis*. Springer, Berlin, 1993.
16. C. Wagner. Tangential frequency filtering decompositions for symmetric matrices. *Numer. Math.*, 78(1):119–142, 1997.
17. C. Wagner. Tangential frequency filtering decompositions for unsymmetric matrices. *Numer. Math.*, 78(1):143–163, 1997.
18. F. Willien, I. Faille, F. Nataf, and F. Schneider. Domain decomposition methods for fluid flow in porous medium. In *6th European Conference on the Mathematics of Oil Recovery*, sep 1998.
19. G. Wittum. An ILU-based smoothing correction scheme. In *Parallel algorithms for partial differential equations, Proc. 6th GAMM-Semin., Kiel/Ger. , Notes Numer. Fluid Mech.*, volume 31, pages 228–240, 1991.
20. G. Wittum. *Filternde Zerlegungen. Schnelle Löser fuer grosse Gleichungssysteme*. Teubner

Skripten zur Numerik, Stuttgart, 1992.