# Block Diagonal Preconditioners for Saddle Point Matrices

Doris Schuhmann

July 9, 2008

Bakkalaureatsarbeit aus "Numerik", Johannes Kepler Universität Linz, WS 2007.

Name: Doris Schuhmann
Matr.Nr.: 0555429
StKz.: 201

# Contents

# 1 Introduction

This bachelor thesis is mainly based on the article of Murphy et al.[4] on preconditioning for indefinite linear systems and on the article of Sturler and Liesen [2] on block diagonal preconditioners. A survey over iteration methods and preconditioners for saddle point problems can be found in Benzi et al. [1].
The main focus of this thesis is to give an overview of block diagonal preconditioners for saddle point systems of the following form

$$\mathcal{A}u = \begin{bmatrix} A & B_1^T \\ B_2 & -C \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \tag{1}$$

with $A \in \mathbb{R}^{n\times n}$, $B_1, B_2 \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{m\times m}$ with $n \geq m$.
Let us assume that the matrix $\mathcal{A}$ is invertible and possibly nonsymmetric. In general we deal with an indefinite system. In the case $C = 0$ the saddle point system (1) reduces to the form:

$$\begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \tag{2}$$

After first discussing exact preconditioning in detail and commenting on important convergence results in Chapter 2, we will then continue with approximate preconditioners for systems of the form (2) and generalize the results for exact preconditioning to the approximate case. Finally, in Chapter 3, we will apply our numerical algorithms to mixed finite element equations obtained from fourth-order boundary value problems.

# 2 Block Diagonal Preconditioners

Generally, the role of a preconditioner is to "reduce the number of iterations required for "convergence" while not increasing significantly the amount of computation required at each iteration."(remarked in [4],p.1969). For Krylov subspace methods like CG the number of iterations until convergence depends on the condition number of the system matrix. But if the eigenvalues of the system matrix are clustered, the condition number becomes less important and the method can converge towards the solution after a few iterations. Here, we investigate preconditioners that result in a clustering of eigenvalues. In the ideal case the number of eigenvalues of the system matrix of such a preconditioned system can be reduced to 3 or less.

In the following, block diagonal preconditioners of the form

$$\mathcal{P} = \begin{bmatrix} * & 0 \\ 0 & * \end{bmatrix} \tag{3}$$

will be considered.

These block diagonal preconditioners can be subdivided into:

- **Exact block diagonal preconditioners:**
  In Section 2.1 we will show that certain exact block diagonal preconditioners yield a system with exact three or two distinct eigenvalues, but these exact preconditioners are hardly usable in reality, because they require the solution of systems with the system matrix A and Schur complement matrix S.

- **Approximated block diagonal preconditioners:**
  In Section 2.2 the characteristics of approximated block diagonal preconditioners are described. In particular, it will be shown that the eigenvalues of approximated preconditioners are clustered around the eigenvalues of exact preconditioners.

## 2.1 Exact Preconditioners

In this section exact preconditioners of the form

$$\mathcal{P} = \begin{bmatrix} A & 0 \\ 0 & -S \end{bmatrix}$$

will be considered, where

$$S = -B_2 A^{-1} B_1^T$$

denotes the Schur complement. For such preconditioners the following Proposition 2.1 is valid (see Proposition 1 in [4], p.1970).

**Proposition 2.1** *If the preconditioner*

$$\mathcal{P} = \begin{bmatrix} A & 0 \\ 0 & B_2 A^{-1} B_1^T \end{bmatrix}$$

*is applied to*

$$\mathcal{A} = \begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix},$$

*then the preconditioned matrix $\mathcal{T} = \mathcal{P}^{-1}\mathcal{A}$ satisfies the matrix equation $\mathcal{T}(\mathcal{T}-I)(\mathcal{T}^2-\mathcal{T}-I) = 0$.*

**Proof:** First, it is easy to verify that

$$\begin{aligned}
\mathcal{T} = \mathcal{P}^{-1}\mathcal{A} &= \begin{bmatrix} A^{-1} & 0 \\ 0 & (B_2 A^{-1} B_1^T)^{-1} \end{bmatrix} \begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} = \\
&= \begin{bmatrix} I & A^{-1} B_1^T \\ (B_2 A^{-1} B_1^T)^{-1} B_2 & 0 \end{bmatrix}, \\
\left(\mathcal{T} - \frac{1}{2} I\right)^2 &= \left( \begin{bmatrix} I & A^{-1} B_1^T \\ (B_2 A^{-1} B_1^T)^{-1} B_2 & 0 \end{bmatrix} - \frac{1}{2} I \right)^2 = \begin{bmatrix} \frac{1}{2} I & A^{-1} B_1^T \\ (B_2 A^{-1} B_1^T)^{-1} B_2 & -\frac{1}{2} I \end{bmatrix}^2 = \\
&= \begin{bmatrix} \frac{1}{4} I + A^{-1} B_1^T (B_2 A^{-1} B_1^T)^{-1} B_2 & \frac{1}{2} A^{-1} B^T - \frac{1}{2} A^{-1} B^T \\ 0 & (B_2 A^{-1} B_1^T)^{-1} B_2 A^{-1} B_1^T + \frac{1}{4} I \end{bmatrix} = \\
&= \begin{bmatrix} \frac{1}{4} I + A^{-1} B_1^T (B_2 A^{-1} B_1^T)^{-1} B_2 & 0 \\ 0 & \frac{5}{4} I \end{bmatrix}, \text{ and}
\end{aligned}$$

$$\left[(\mathcal{T}-\frac{1}{2}I)^2-\frac{1}{4}I\right]^2 = \begin{bmatrix} A^{-1}B_1^T(B_2A^{-1}B_1^T)^{-1}B_2 & 0 \\ 0 & I \end{bmatrix}^2 = \begin{bmatrix} \left(A^{-1}B_1^T(B_2A^{-1}B_1^T)^{-1}B_2\right)^2 & 0 \\ 0 & I^2 \end{bmatrix} =$$
$$= \begin{bmatrix} A^{-1}B_1^T(B_2A^{-1}B_1^T)^{-1}B_2A^{-1}B_1^T(B_2A^{-1}B_1^T)^{-1}B_2 & 0 \\ 0 & I \end{bmatrix} =$$
$$= \begin{bmatrix} A^{-1}B_1^T(B_2A^{-1}B_1^T)^{-1}B_2 & 0 \\ 0 & I \end{bmatrix} = (\mathcal{T}-\frac{1}{2}I)^2-\frac{1}{4}I.$$

The following equality

$$\left[(\mathcal{T}-\frac{1}{2}I)^2-\frac{1}{4}I\right]^2 = \left[(\mathcal{T}-\frac{1}{2}I)^2-\frac{1}{4}I\right]$$

results from

$$A^{-1}B_1^T(B_2A^{-1}B_1^T)^{-1}B_2A^{-1}B_1^T(B_2A^{-1}B_1^T)^{-1}B_2 = A^{-1}B_1^T(B_2A^{-1}B_1^T)^{-1}B_2.$$

Therefore, it follows that

$$\begin{aligned} (\mathcal{T}^2-\mathcal{T}I)^2 &= \mathcal{T}^2-\mathcal{T}I \\ (\mathcal{T}^2-\mathcal{T}I)^2-(\mathcal{T}^2-\mathcal{T}I) &= 0 \\ \mathcal{T}(\mathcal{T}-I)\left[\mathcal{T}(\mathcal{T}-I)-I\right] &= 0 \\ \mathcal{T}(\mathcal{T}-I)\left[\mathcal{T}^2-\mathcal{T}-I\right] &= 0. \end{aligned}$$

So $[(\mathcal{T}-\frac{1}{2}I)^2-\frac{1}{4}I]^2 = \left[(\mathcal{T}-\frac{1}{2}I)^2-\frac{1}{4}I\right]$ simplifies to $\mathcal{T}(\mathcal{T}-I)(\mathcal{T}^2-\mathcal{T}-I)=0$.

■

Based on Proposition 2.1, in Proposition 2.2 the location of the eigenvalues of the matrix $\mathcal{T}$ is derived.

**Proposition 2.2** *Since $\mathcal{T}(\mathcal{T}-I)(\mathcal{T}^2-\mathcal{T}-I)=0$ can be factorized into distinct linear factors (over $\mathbb{R}$), it follows that $\mathcal{T}$ is diagonalizable and has at most the three distinct eigenvalues $1, \frac{1}{2}\pm\frac{\sqrt{5}}{2}$.*

As Proposition 2.2 is also mentioned in [4], p.1970, Remark 1, we will not show it directly, but instead we will concentrate on proving that the matrix $\mathcal{T}$ has at most three distinct eigenvalues $(1, \frac{1}{2}\pm\frac{\sqrt{5}}{2})$.
Therefore, we directly consider the eigenvalue problem

$$\mathcal{T}\begin{bmatrix} x \\ y \end{bmatrix} = \lambda\begin{bmatrix} x \\ y \end{bmatrix}. \tag{4}$$

$\mathcal{T}=\mathcal{P}^{-1}\mathcal{A}$ implies that

$$\mathcal{T}\begin{bmatrix} x \\ y \end{bmatrix} = \mathcal{P}^{-1}\mathcal{A}\begin{bmatrix} x \\ y \end{bmatrix} = \lambda\begin{bmatrix} x \\ y \end{bmatrix}.$$

A multiplication of the system (4) by $\mathcal{P}$ results in

$$\begin{aligned} \mathcal{A}\begin{bmatrix} x \\ y \end{bmatrix} &= \lambda\mathcal{P}\begin{bmatrix} x \\ y \end{bmatrix} \\ \begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} &= \lambda\begin{bmatrix} A & 0 \\ 0 & B_2A^{-1}B_1^T \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}. \end{aligned}$$

The matrix equation results in two equations, which can be simplified, such that the calculation of the eigenvalues $\lambda_i$ with $i \in \{1,2,3\}$ is possible:

$$\begin{aligned} (I) \quad & Ax + B_1^T y = \lambda Ax \\ (II) \quad & B_2 x \qquad\quad = \lambda B_2 A^{-1} B_1^T y \end{aligned}$$

$$\begin{aligned} (II) \;\Rightarrow\; & x = \lambda B_2^{-1} B_2 A^{-1} B_1^T y \\ (I) \;\Rightarrow\; & 0 = (1-\lambda)Ax + B_1^T y \\ & = (1-\lambda)A\lambda B_2^{-1} B_2 A^{-1} B_1^T y + B_1^T y \\ & = ((1-\lambda)\lambda A B_2^{-1} B_2 A^{-1} + I) B_1^T y \end{aligned}$$

One obtains two cases:

- $((1-\lambda)\lambda A B_2^{-1} B_2 A^{-1} + I) = 0$,
  $AB_2^{-1}B_2A^{-1}$ is a projection such that $(1-\lambda)\lambda I + I = 0$

$$\lambda^2 - \lambda - 1 = 0 \Leftrightarrow \lambda_{1/2} = \frac{1}{2} \pm \frac{\sqrt{5}}{2}$$

- $B_1^T y = 0 \Leftrightarrow B_1^T y = (\lambda - 1)Ax = 0 \Leftrightarrow \lambda - 1 = 0 \Rightarrow \lambda_3 = 1$

■

Until now, Proposition 2.1 and as a consequence Proposition 2.2 are only shown for left preconditioning, but it is important to consider their validity for right and centered preconditioning as well (compare [4], p.1970, Remark 2).

**Proposition 2.3** *For right preconditioning $\mathcal{T} = \mathcal{A}\mathcal{P}^{-1}$, or in general, for any centered preconditioning $\mathcal{T} = \mathcal{P}_1^{-1}\mathcal{A}\mathcal{P}_2^{-1}$, where $\mathcal{P}_1\mathcal{P}_2 = \mathcal{P}$ the same result as in Proposition 2.1 is achieved.*

**Proof:**
In the following, we will show that the right preconditioned matrix $\mathcal{T} = \mathcal{A}\mathcal{P}^{-1}$ as well as the centered preconditioned matrix $\mathcal{T} = \mathcal{P}_1^{-1}\mathcal{A}\mathcal{P}_2^{-1}$ satisfy the equation $\mathcal{T}\left(\mathcal{T} - I\right)\left(\mathcal{T}^2 - \mathcal{T} - I\right) = 0$.

- Right preconditioning:

$$\begin{aligned}
\mathcal{T} = \mathcal{A}\mathcal{P}^{-1} &= \begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & (B_2A^{-1}B_1^T)^{-1} \end{bmatrix} = \\
&= \begin{bmatrix} I & B_1^T(B_2A^{-1}B_1^T)^{-1} \\ B_2A^{-1} & 0 \end{bmatrix} \\
(\mathcal{T} - \frac{1}{2}I)^2 &= \begin{bmatrix} \frac{1}{4}I + B_1^T(B_2A^{-1}B_1^T)^{-1}B_2A^{-1} & 0 \\ 0 & \frac{5}{4}I \end{bmatrix} \\
\left[\left(\mathcal{T} - \frac{1}{2}I\right)^2 - \frac{1}{4}I\right]^2 &= \begin{bmatrix} B_1^T\left(B_2A^{-1}B_1^T\right)^{-1}B_2A^{-1} & 0 \\ 0 & I \end{bmatrix}^2 = \\
&= \begin{bmatrix} B_1^T\left(B_2A^{-1}B_1^T\right)^{-1}B_2A^{-1}B_1^T\left(B_2A^{-1}B_1^T\right)^{-1}B_2A^{-1} & 0 \\ 0 & I \end{bmatrix} = \\
&= \begin{bmatrix} B_1^T\left(B_2A^{-1}B_1^T\right)^{-1}B_2A^{-1} & 0 \\ 0 & I \end{bmatrix} = \\
&= \left[(\mathcal{T} - \frac{1}{2}I)^2 - \frac{1}{4}I\right]
\end{aligned}$$

  Thus, it follows that

$$[(\mathcal{T} - \frac{1}{2}I)^2 - \frac{1}{4}I]^2 = [(\mathcal{T} - \frac{1}{2}I)^2 - \frac{1}{4}I],$$

  this equation was also obtained in the proof for Proposition 2.1, so it is again possible to simplify the equation into

$$\mathcal{T}(\mathcal{T} - I)(\mathcal{T}^2 - \mathcal{T} - I) = 0.$$

  Because of Remark 2.2 the eigenvalues for $\mathcal{T} = \mathcal{A}\mathcal{P}^{-1}$ are equal to the eigenvalues obtained for left preconditioning.

- Centered preconditioning:

  Assume that $\mathcal{P}_1$ and $\mathcal{P}_2$ are invertible:

$$\mathcal{T} = \mathcal{P}_1\mathcal{A}\mathcal{P}_2^{-1}, \quad \text{where} \;\; \mathcal{P}_1\mathcal{P}_2 = \mathcal{P} \Rightarrow \mathcal{P}_2 = \mathcal{P}_1^{-1}\mathcal{P}$$

$$\begin{aligned}
\mathcal{T} &= \mathcal{P}_1^{-1}\mathcal{A}\mathcal{P}_2^{-1} = \mathcal{P}_1^{-1}\mathcal{A}(\mathcal{P}_1^{-1}\mathcal{P})^{-1} = \mathcal{P}_1^{-1}\mathcal{A}\mathcal{P}^{-1}\mathcal{P}_1 = \\
&= \mathcal{P}_1^{-1}\begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & (B_2A^{-1}B_1^T)^{-1} \end{bmatrix} \mathcal{P}_1
\end{aligned}$$

  We define: $\mathcal{K} := \begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} A^{-1} & 0 \\ 0 & (B_2A^{-1}B_1^T)^{-1} \end{bmatrix}$.

  As demonstrated in the first part of this proof, the right preconditioned matrix $\mathcal{K}$ satisfies the equation

$$\mathcal{K}\left(\mathcal{K} - I\right)\left(\mathcal{K}^2 - \mathcal{K} - I\right) = 0.$$

It has to be shown that this property is also true for centered preconditioning, i.e, that

$$\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1\left(\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1 - I\right)\left(\left(\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1\right)^2 - \mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1 - I\right) = 0.$$

It follows:

$$\begin{aligned}
&\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1\left(\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1 - I\right)\left(\left(\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1\right)^2 - \mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1 - I\right) = \\
&= \left(\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1 - \mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1\right)\left(\left(\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1\right)^2 - \mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1 - I\right) = \\
&= \mathcal{P}_1^{-1}\mathcal{K}\left(\mathcal{K} - I\right)\mathcal{P}_1\left(\left(\mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1\right)^2 - \mathcal{P}_1^{-1}\mathcal{K}\mathcal{P}_1 - I\right) = \\
&= \mathcal{P}_1^{-1}\mathcal{K}\left(\mathcal{K} - I\right)\left(\mathcal{K}^2 - \mathcal{K} - I\right)\mathcal{P}_1
\end{aligned}$$

Because $\mathcal{K}\left(\mathcal{K} - I\right)\left(\mathcal{K}^2 - \mathcal{K} - I\right) = 0$ it follows that $\mathcal{P}_1^{-1}\mathcal{K}\left(\mathcal{K} - I\right)\left(\mathcal{K}^2 - \mathcal{K} - I\right)\mathcal{P}_1 = 0$.

■

**Remark 2.4** *Proposition 2.1 implies that for any vector $r_0$ the Krylov subspace span $\left\{r_0, \mathcal{T}r_0, \mathcal{T}^2r_0, \mathcal{T}^3r_0, ...\right\}$ has at most dimension three. Thus, when solving a linear system of the form (2) any Krylov subspace method with Galerkin or orthogonality property stops in at most three iterations. See also [4], p.1970,Remark 3.*

## 2.2 Approximate Preconditioners

As mentioned in [2], it can be shown that solving the above mentioned preconditioned system is more expensive than solving the saddle point problem in a direct way by block elimination. In practice this problem can be avoided by approximating $A$ and $S$ by some matrices $\hat{A}$ and $\hat{S}$ and replacing the exact preconditioner $\mathcal{P}$ by an approximation

$$\hat{\mathcal{P}} = \begin{bmatrix} \hat{A} & 0 \\ 0 & \hat{S} \end{bmatrix}.$$

In literature many different approximations are described, we will concentrate on a quite general framework which occurs in Sturler and Liesen (2005)[2], p. 1600 and is based on a splitting of the matrix $A$.

Let $\mathcal{A}$ be an invertible matrix.
First the matrix A is splitted into $A = D - E$. Let us assume that $B_2 D^{-1} B_1^T$ is invertible. Then the preconditioner is obtained by choosing $\hat{A} = D$ and $\hat{S} = -B_2 D^{-1} B_1^T$:

$$\hat{\mathcal{P}} = \begin{bmatrix} \hat{A} & 0 \\ 0 & -\hat{S} \end{bmatrix} = \begin{bmatrix} D & 0 \\ 0 & B_2 D^{-1} B_1^T \end{bmatrix}.$$

The inverse of $\hat{\mathcal{P}}$ is given by

$$\mathcal{P}(D) = \hat{\mathcal{P}}^{-1} = \begin{bmatrix} D^{-1} & 0 \\ 0 & (B_2 D^{-1} B_1^T)^{-1} \end{bmatrix}.$$

Left preconditioning results in the matrix

$$\mathcal{P}(D)\mathcal{A} = \begin{bmatrix} D^{-1}A & D^{-1}B_1^T \\ (B_2 D^{-1} B_1^T)^{-1} B_2 & 0 \end{bmatrix} \overset{A=D-E}{=} \begin{bmatrix} I_n - D^{-1}E & D^{-1}B_1^T \\ (B_2 D^{-1} B_1^T)^{-1} B_2 & 0 \end{bmatrix}$$

whereas right preconditioning results in the matrix

$$\mathcal{A}\mathcal{P}(D) = \begin{bmatrix} AD^{-1} & B_1^T (B_2 D^{-1} B_1^T)^{-1} \\ B_2 D^{-1} & 0 \end{bmatrix} \overset{A=D-E}{=} \begin{bmatrix} I_n - ED^{-1} & B_1^T (B_2 D^{-1} B_1^T)^{-1} \\ B_2 D^{-1} & 0 \end{bmatrix}.$$

Both (left and right preconditioned) matrices can be generalized to the form

$$\mathcal{B}(S) = \begin{bmatrix} I_n - S & N \\ M & 0 \end{bmatrix}, \tag{5}$$

where $MN = I_m$, $(NM)^2 = NM$, $S \in \mathbb{R}^{n \times n}$, $M, N \in \mathbb{R}^{m \times n}, n \geq m$.

Applying $\mathcal{P}(D)$ to the general nonsingular linear system of the form

$$\mathcal{A}u = \begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}$$

results in

$$\mathcal{B}(\mathcal{S}) \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}.$$

Concerning the iteration count, the most effective way for solving the system

$$\begin{bmatrix} A & B_1^T \\ B_2 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}$$

is given by the splitting of $A = D - E$ with the choice $D = A$ and $E = 0$, which results in the preconditioner $\mathcal{P}$, which is mentioned in Proposition 2.1. This preconditioner satisfies Proposition 2.2 and Remark 2.4. But as the preconditioner requires multiplications with $A^{-1}$ and the invertibility of the Schur complement, problems can arise. As remarked in [2] approximations to $A^{-1}$ and the Schur complement should lead to clustered eigenvalues. The quality of this clustering depends on the approximations. "So the general strategy must be a splitting that leads to efficiently invertible matrices $D$ and $B_2D^{-1}B_1^T$ and at the same time preserves the properties of the algebraically optimal preconditioner $\mathcal{P}$ as much as possible." ([2], p.1601(Remark 2.1))

## 2.3 Properties of the matrices $\mathcal{B}(S)$

Analogously to Section 2.1, in this section the eigendecomposition of the matrices $\mathcal{B}(0)$ and $\mathcal{B}(S)$ will be considered. As Sturler and Liesen [2] remarked on p.1601, Theorem 3.1 and Remark 3.2, each matrix $\mathcal{B}(0)$ is nonsingular. They argue that nonsingularity of $\mathcal{B}(S)$ is given if and only if one is an eigenvalue of the matrix $(I_n - NM)S$. When assuming that the preconditioner $\mathcal{P}(D)$ exists the matrices $D$ and $B_2D^{-1}B_1^T$ are invertible which implies the nonsingularity of $\mathcal{B}(S)$ and therefore its eigenvalues are different from zero.
By doing the eigendecomposition of the matrices $\mathcal{B}(0)$ and $\mathcal{B}(S)$, one obtains a clustering around the eigenvalues 1 and $\frac{1}{2} \pm \frac{\sqrt{5}}{2}$, which is mentioned in Proposition 2.5 and Proposition 2.6 (compare to Theorem 3.3 and Theorem 3.5 in [2]). As remarked in [4], p.1970, Remark 5 the eigenvalues of the approximated preconditioner are for good approximations the same as in the non-approximate case.

**Proposition 2.5 (Eigendecomposition of $\mathcal{B}(0)$)**
*Let*

$$\mathcal{B}(0) = \begin{bmatrix} I_n & N \\ M & 0 \end{bmatrix},$$

*where* $MN = I_m$, $(NM)^2 = NM$, $M, N \in \mathbb{R}^{m \times n}, n \geq m$.

*Then* $\mathcal{B}(0)$ *is diagonalizable, and it has*

- $n - m$ *eigenpairs of the form*

$$\left(1, \begin{bmatrix} u_j^T, & 0 \end{bmatrix}^T\right),$$

*where* $u_1, ..., u_{n-m}$ *form a basis of the nullspace of* $NM$ .

- *$2m$ eigenpairs of the form*

$$\left(\frac{1}{2} \pm \frac{\sqrt{5}}{2}, \left[ \begin{array}{cc} u_j^T, & (\frac{1}{2} \pm \frac{\sqrt{5}}{2})^{-1}(Mu_j)^T \end{array} \right]^T\right),$$

*where $\frac{1}{2} \pm \frac{\sqrt{5}}{2}$ and $u_{n-m+1}, ..., u_n$ form a basis of the range of $NM$.*

*When denoting $U_1 \equiv \left[ \begin{array}{ccc} u_1, & ..., u_{n-m} \end{array} \right] \in \mathbb{R}^{n\times(n-m)}$ , $U_2 \equiv \left[ \begin{array}{ccc} u_{n-m+1}, & ..., u_n \end{array} \right] \in \mathbb{R}^{n\times m}$, then the eigenvector matrix $\mathcal{Y}(0)$ of $\mathcal{B}(0)$ is given by*

$$\mathcal{Y}(0) = \left[ \begin{array}{ccc} U_1 & U_2 & U_2 \\ 0 & (\frac{1}{2} + \frac{\sqrt{5}}{2})^{-1}MU_2 & (\frac{1}{2} - \frac{\sqrt{5}}{2})^{-1}MU_2 \end{array} \right],$$

*where both, $[U_1, U_2] \in \mathbb{R}^{n\times n}$ and $(\frac{1}{2} - \frac{\sqrt{5}}{2})^{-1} \in \mathbb{R}^{m\times m}$ are nonsingular.*

The matrix $\mathcal{B}(0)$ is equal to the matrix obtained by applying the preconditioner mentioned in Proposition 2.1 to the matrix $\mathcal{A}$. As we proved the location of the eigenvalues of this preconditioned matrix, we will not do the proof of Proposition 3.5, which can be looked up in [2], p.1602. The aim of Proposition 2.6 is to derive bounds on the eigenvalues of each matrix $\mathcal{B}(S)$ concerning the corresponding matrix $\mathcal{B}(0)$.

**Proposition 2.6 (Eigendecomposition of $\mathcal{B}(S)$)**
*Let $\mathcal{B}(S)$ be of the form (4) with $N$ and $M$ fixed. Let the eigendecomposition of $\mathcal{B}(0)$ be denoted by $\mathcal{B}(0) = \mathcal{Y}(0)\mathcal{D}\mathcal{Y}(0)^{-1}$, where the eigenvectormatrix $\mathcal{Y}(0)$ is of the same form as mentioned in Proposition 2.5. Further let $[U_1, U_2]$ denote the corresponding eigenvector matrix of the projection $NM$.*

*Then for each matrix $S$ and each eigenvalues $\lambda_S$ of $\mathcal{B}(S)$ there is an eigenvalue $\lambda$ of $\mathcal{B}(0)$, such that*

$$|\lambda_S - \lambda| \leq \left\| \mathcal{Y}(0)^{-1} \left[ \begin{array}{cc} S & 0 \\ 0 & 0 \end{array} \right] \mathcal{Y}(0) \right\| \leq c_S \left\| [U_1, U_2]^{-1} S [U_1, U_2] \right\|,$$

*where $c_S = (2 + \frac{2}{5}(\frac{1}{2} - \frac{\sqrt{5}}{2})^2)^{\frac{1}{2}} \approx 1.4672$.*

The proof of Proposition 2.6 is done in [2], p. 1603-1604.

# 3 Example

## 3.1 The Beam Bending Problem

Let us consider the beam bending problem, of which the **classical formulation** is given by the following equations:

$$\frac{d^4u}{dx^4} = f(x),\ x \in (0,1), \tag{6}$$

$$u(0) = 0,\ u(1) = 0, \tag{7}$$

$$u'(0) = 0,\ u'(1) = 0. \tag{8}$$

As depicted in Figure 1, the problem (6)-(8) describes a beam which is fixed on both ends and on which a perpendicular force is acting. Thus, the beam equation (6) is subjected to the boundary conditions (7)-(8).

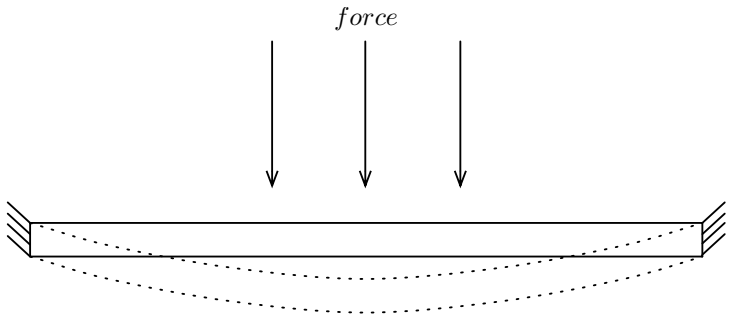


Figure 1: Beam equation.

The 2D analogon to the beam bending problem, the bending of a clamped plate leads to the first biharmonic boundary value problem: $\Delta^2 u = f$ in $\Omega$ with $u = \frac{\partial u}{\partial n} = 0$ on $\partial\Omega$, where $\frac{\partial u}{\partial n}$ denotes the normal derivative.

### 3.1.1 The Primal Variational Formulation

For deriving the **primal variational formulation** of the problem, we first define the space $V_0$ of the test function $v$:

$$V_0 = \{v \in H^2(0,1) : v(0) = 0, v(1) = 0 \text{ and } v'(0) = 0, v'(1) = 0\} = H_0^2(0,1)$$

The multiplication of the differential equation (6) by an arbitrary test function $v \in V_0$ and integration over the computational domain $\Omega = (0,1)$ results in:

$$\int_0^1 \frac{d^4u}{dx^4} v(x)dx = \int_0^1 f(x)v(x)dx \qquad \forall v \in V_0.$$

By integrating by parts and incorporating $v(0) = 0$ and $v(1) = 0$ one obtains:

$$\begin{aligned} u'''(x)v(x)\Big|_0^1 - \int_0^1 u'''(x)v'(x)dx &= \int_0^1 f(x)v(x)dx \qquad \forall v \in V_0, \\ -\int_0^1 u'''(x)v'(x)dx &= \int_0^1 f(x)v(x)dx \qquad \forall v \in V_0. \end{aligned}$$

Integration by parts has to be done a second time:

$$-u''(x)v'(x)\Big|_0^1 + \int_0^1 u''(x)v''(x)dx = \int_0^1 f(x)v(x)dx \qquad \forall v \in V_0.$$

Incorporating $v'(0) = 0$ and $v'(1) = 0$ results in:

$$\int_0^1 u''(x)v''(x)dx = \int_0^1 f(x)v(x)dx \qquad \forall v \in V_0.$$

Finally, the set $V_g$ for the solution $u$ has to be defined:

$$u \in V_g = V_0 = H_0^2(0,1).$$

Summarizing, the **variational formulation** is given by:
Find $u \in V_0$ such that

$$a(u,v) = \langle F, v \rangle \ \ \forall v \in V_0 \tag{9}$$

with

$$a(u,v) = \int_0^1 u''(x)v''(x)dx, \tag{10}$$

and

$$\langle F, v \rangle = \int_0^1 f(x)v(x)dx. \tag{11}$$

### 3.1.2 A Mixed Variational Formulation

The problem (9)-(11) is living in the Sobolev space $H^2(0,1)$, which complicates the finite element approximation as mentioned in [3], p.292, we use the ansatz $w = u''$, which transferrs the $H^2$ - problem (9)-(11) to the $H^1$ - problem (14)-(15). Instead of the fourth-order differential equation (6) we use now the following coupled system of two second-order differential equations for deriving the variational formulation:

$$w - u'' = 0, \tag{12}$$

$$-w'' = -f. \tag{13}$$

By multiplying the differential equation (12) with an arbitrary test function $v \in V = H^1(0,1)$

and integrating over the computational domain $\Omega = (0,1)$ one obtains

$$\int_0^1 w(x)v(x)dx - \int_0^1 u''(x)v(x)dx = 0 \qquad \forall v \in V.$$

An integration by parts and incorporation of $u'(0) = 0$ and $u'(1) = 0$ results in

$$\begin{aligned} \int_0^1 w(x)v(x)dx - u'(x)v(x)\Big|_0^1 + \int_0^1 u'(x)v'(x)dx &= 0 \qquad \forall v \in V, \\ \int_0^1 w(x)v(x)dx + \int_0^1 u'(x)v'(x)dx &= 0 \qquad \forall v \in V. \end{aligned}$$

The same process has to be applied to (13). A multiplication of the differential equation $-w'' = -f$ by an arbitrary test function $q \in W = H_0^1(0,1)$ and integration over the computational domain $\Omega = (0,1)$ leads to

$$-\int_0^1 w''(x)q(x)dx = -\int_0^1 f(x)q(x)dx \qquad \forall q \in W.$$

Integration by parts and an incorporation of $q'(0) = 0$ and $q'(1) = 0$ results in:

$$\begin{aligned} -w'(x)q(x)\Big|_0^1 + \int_0^1 w'(x)q'(x)dx &= -\int_0^1 f(x)q(x)dx \qquad \forall q \in W, \\ \int_0^1 w'(x)q'(x)dx &= -\int_0^1 f(x)q(x)dx \qquad \forall q \in W. \end{aligned}$$

The **mixed variational formulation** is then given by:

Let $w \in V = H^1(0,1)$, $u \in W = H_0^1(0,1)$:

$$a(w,v) + b(u,v) = \langle F, v\rangle_V \qquad \forall v \in V, \tag{14}$$

$$b(w,q) = \langle G, q\rangle_W \qquad \forall q \in W, \tag{15}$$

where

$$a(w,v) = \int_0^1 w(x)v(x)dx, \qquad b(v,u) = \int_0^1 u'(x)v'(x)dx, \qquad \langle F, v\rangle_V = 0,$$

$$b(w,q) = \int_0^1 w'(x)q'(x)dx, \qquad \langle G, q\rangle_W = -\int_0^1 f(x)q(x)dx.$$

In the following, we want to transform the system (14)-(15) to the saddle point system (18) and determine the calculation rule for the appropriate matrices.

### 3.1.3 The Mixed Finite Element Discretization

We use the finite element method with hat functions to discretize the problem (14)-(15). The interval $\Omega = (0,1)$ is subdivided into $N_h$ parts of the length $h_k = \frac{1}{N_h}$ with the nodes $x_i$, $i = 0, ..., N_h$ with $0 = x_0 < x_1 < ... < x_{N_h}$ (Figure 2). This approach implies the subdivision $\mathcal{T}_h = \{T_k : k = 1, ..., N_h\}$ of the computational domain $\Omega$ into a set of subintervals $T_k = (x_{k-1}, x_k)$ for $k = 1, ..., N_h$.

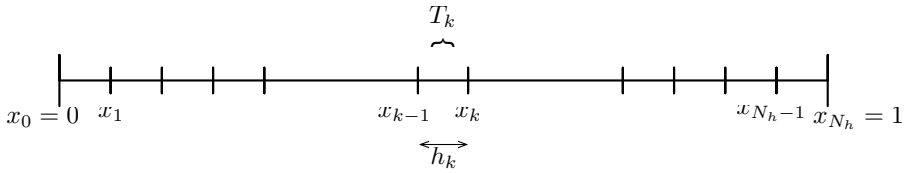


Figure 2: Subdivision of the computational domain.

Let $\mathcal{P}_p = \left\{\sum_{i=0}^{p} c_i x^i\right\}$ denote the set of all polynomials with degree less or equal to p. We define the finite element spaces $V_h$ and $W_h$ by

$$\begin{aligned} V_h &= \left\{ w_h \in C\left(\bar{\Omega}\right) : w_h\Big|_{T_k} \in \mathcal{P}_1 \qquad \forall T_k \in \mathcal{T}_h \right\} = \operatorname{span}\left\{\varphi_0, ..., \varphi_{N_h}\right\}, \\ W_h &= \left\{ u_h \in C\left(\Omega\right) : u_h\Big|_{T_k} \in \mathcal{P}_1 \qquad \forall T_k \in \mathcal{T}_h, u_h\left(0\right) = u_h\left(1\right) = 0 \right\} = \operatorname{span}\left\{\varphi_1, ..., \varphi_{N_h-1}\right\}, \end{aligned}$$

spanned by the nodal basis function $\varphi_i$, where

$$\varphi_i\left(x_j\right) = \delta_{ij}, \text{ for } i, j = 0, ..., N_h.$$

$V_h \subset V$ and $W_h \subset W$, thus, $\dim(V_h) = N_h + 1 = n$ and $\dim(W_h) = N_h - 1 = m$.

We want to find $w_h = \sum_{j=0}^{N_h} w_j \varphi_j \in V_h$ and $u_h = \sum_{k=1}^{N_h - 1} u_k \varphi_k \in W_h$ such that (14) and (15) are fullfilled for all test function from $V_h$ and $W_h$, respectively, i.e., firstly,

$$\begin{aligned} a(w_h, v_h) + b(u_h, v_h) &= \langle F, v_h \rangle \qquad \forall v_h \in V_h, \\ a\left(\sum_{j=0}^{N_h} w_j \varphi_j, v_h\right) + b\left(\sum_{k=1}^{N_h - 1} u_k \varphi_k, v_h\right) &= \langle F, v_h \rangle \qquad \forall v_h \in V_h. \end{aligned}$$

Because of the linearity with respect to $v_h \in V_h$, it suffices to test only with the basis functions $\varphi_i$ with $i \in \{0, ..., N_h\}$:

$$\begin{aligned} a\left(\sum_{j=0}^{N_h} w_j \varphi_j, \varphi_i\right) + b\left(\sum_{k=1}^{N_h-1} u_k \varphi_k, \varphi_i\right) &= \langle F, \varphi_i \rangle \qquad \forall i \in 0, ..., N_h \\ &\Updownarrow \quad a(., \varphi_i) \text{ and } b(., \varphi_i) \text{ linear} \\ \sum_{j=0}^{N_h} w_j a(\varphi_j, \varphi_i) + \sum_{k=1}^{N_h-1} u_k b(\varphi_k, \varphi_i) &= \langle F, \varphi_i \rangle \qquad \forall i = 0, ..., N_h \\ &\Updownarrow \\ \text{Find } \underline{w_h} = (w_j)_{j=0}^{N_h} \text{ and } \underline{u_h} = (u_k)_{k=1}^{N_h-1} &: \quad A_h \underline{w_h} + C_h \underline{u_h} = \underline{f_h} \end{aligned}$$

with the mass matrix $A_h$ and the stiffness matrix $C_h$:

$$\begin{aligned} A_h &= [A_{ij}] = [a(\varphi_j, \varphi_i)], \qquad \text{where} \quad i, j = 0, ..., N_h, \\ C_h &= [C_{ik}] = (B_h)^T = [B_{jl}] = [b(\varphi_l, \varphi_j)], \qquad \text{where} \quad k, l = 1, ..., N_h - 1, \end{aligned}$$

and the vector

$$\underline{f_h} = [f_i]_{i=0}^{N_h} = 0.$$

And, secondly, the same procedure has to be done for equation (15):

$$\begin{aligned} b(w_h, q_h) &= \langle G, q_h \rangle \qquad \forall q_h \in W_h, \\ b\left(\sum_{j=0}^{N_h} w_j \varphi_j, q_h\right) &= \langle G, q_h \rangle \qquad \forall q_h \in W_h. \end{aligned}$$

Due to the linearity with respect to $q_h \in W_h$ it is enough to test only with the basis functions $\varphi_l$ with $l \in \{1, ..., N_h - 1\}$

$$\begin{aligned} b\left(\sum_{j=0}^{N_h} w_j \varphi_j, \varphi_l\right) &= \langle G, \varphi_l \rangle \qquad \forall l = 1, ..., N_h - 1 \\ &\Updownarrow \quad b(., \varphi_l) \text{ linear} \\ \sum_{j=0}^{N_h} w_j b(\varphi_j, \varphi_l) &= \langle G, \varphi_l \rangle \qquad \forall l = 1, ..., N_h - 1 \\ &\Updownarrow \\ \text{Find } \underline{w_h} = (w_j)_{j=0}^{N_h} &: \quad B_h \underline{w_h} = \underline{g_h} \end{aligned}$$

with the stiffness matrix $B_h$:

$$B_h = [B_{lj}] = [b(\varphi_j, \varphi_l)], \text{ where} \qquad l = 1, ..., N_h - 1, j = 0, ..., N_h$$

and

$$\underline{g_h} = [g_l]_{l=1}^{N_h-1} = [\langle G, \varphi_l \rangle]_{l=1}^{N_h-1}.$$

Summarizing, the **system of equations** is given by:

Find $\underline{w_h} \in \mathbb{R}^n$, $\underline{u_h} \in \mathbb{R}^m$ such that

$$A_h \underline{w_h} + B_h^T \underline{u_h} = 0, \tag{16}$$
$$B_h \underline{w_h} = \underline{g_h}. \tag{17}$$

Written in matrix form, the system (16)-(17) leads to the following saddle point problem:

$$\begin{bmatrix} A_h & B_h^T \\ B_h & 0 \end{bmatrix} \begin{bmatrix} \underline{w_h} \\ \underline{u_h} \end{bmatrix} = \begin{bmatrix} 0 \\ \underline{g_h} \end{bmatrix} \tag{18}$$

with a $n \times n$-matrix $A_h$, a $m \times n$-matrix $B_h$, a $m \times m$ zero-matrix, a zero vector with n components and the vector $\underline{g_h}$ with m components.

### 3.1.4 Calculation of the mass matrix $A_h$, the stiffness matrix $B_h$ and the load vector $g_h$

To obtain the matrix $\begin{bmatrix} A_h & B_h^T \\ B_h & 0 \end{bmatrix}$ and the vector $\begin{bmatrix} 0 \\ \underline{g_h} \end{bmatrix}$ the matrices $A_h$ and $B_h$ and the vector $\underline{g_h}$ have to be calculated first. Therefore a transformation from the arbitrary interval to the reference interval $[0, 1]$ is necessary.

**Transformation to the reference element:**

The mapping $F_k^{-1}$ transforms the shape functions $\varphi_k$ and the nodes $x_k$ on the arbitrary domain to the shape functions $\hat{\varphi}_i$ and the nodes $\xi_i$ on the reference domain.

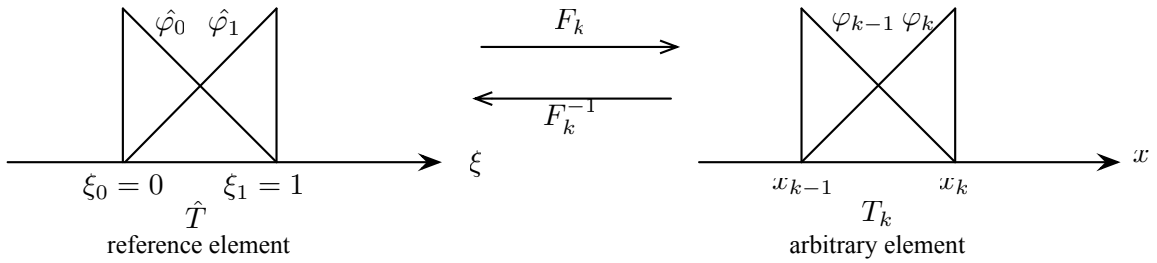


Figure 3: Transformation from the arbitrary element to the reference element.

$$
\begin{aligned}
x &= F_k(\xi) = x_{k-1} + (x_k - x_{k-1})\xi \\
\xi &= F_k^{-1}(x) = \frac{x - x_{k-1}}{x_k - x_{k-1}} = \frac{1}{h_k}(x - x_{k-1}) \\
\varphi_{k-1}(F_k(\xi)) &= 1 - \xi = \hat{\varphi}_0(\xi) \\
\varphi_k(F_k(\xi)) &= \xi = \hat{\varphi}_1(\xi)
\end{aligned}
$$

Further, the following results are obtained:

$$
\begin{aligned}
dx &= \left| det F_k'(\xi) \right| d\xi = |x_k - x_{k+1}|\, d\xi = h_k d\xi, \\
\frac{d}{dx} &= \frac{d}{d\xi}\frac{d\xi}{dx} = \frac{dF_k^{-1}(x)}{dx}\frac{d}{d\xi} = \frac{1}{x_k - x_{k-1}}\frac{d}{d\xi} = \frac{1}{h_k}\frac{d}{d\xi}.
\end{aligned}
$$

**Mass matrix $A_h$:**

$$
(A_h \underline{w_h}, \underline{v_h}) = a(w_h, v_h) = \sum_{T \in \mathcal{T}_h} \sum_{i,j} v_j w_j \int_T \varphi_j \varphi_i dx = \sum_{k=1}^{N_h} \left( A_h^{(k)} \begin{bmatrix} w_{k-1} \\ w_k \end{bmatrix}, \begin{bmatrix} v_{k-1} \\ v_k \end{bmatrix} \right)
$$

where $i, j = 0, ..., N_h$ with the element mass matrix

$$
A_h^{(k)} = \begin{bmatrix} \int_{T_k} \varphi_{k-1}^2(x)dx & \int_{T_k} \varphi_{k-1}(x)\varphi_k(x)dx \\ \int_{T_k} \varphi_k(x)\varphi_{k-1}(x)dx & \int_{T_k} \varphi_k^2(x)dx \end{bmatrix}.
$$

The matrix elements are given by:

$$
\begin{aligned}
\int_{T_k} \varphi_{k-1}^2(x)dx &= \int_{\hat{T}} \hat{\varphi}_0(\xi)^2 h_k d\xi = \int_{\hat{T}} (1-\xi)^2 h_k d\xi = h_k \left( \xi - \xi^2 + \frac{\xi^3}{3}\Big|_0^1 \right) = \frac{h_k}{3} \\
\int_{T_k} \varphi_{k-1}(x)\varphi_k(x)dx &= \int_{\hat{T}} \hat{\varphi}_0(\xi)\hat{\varphi}_1(\xi) h_k d\xi = \int_{\hat{T}} (1-\xi)\xi h_k d\xi = h_k \left( \frac{\xi^2}{2} - \frac{\xi^3}{3} \right) \Big|_0^1 = \frac{h_k}{6} = \\
&= \int_{T_k} \varphi_k(x)\varphi_{k-1}(x)dx \\
\int_{T_k} \varphi_k^2(x)dx &= \int_{T_k} \xi^2 h_k d\xi = h_k \frac{\xi^3}{3}\Big|_0^1 = \frac{h_k}{3}
\end{aligned}
$$

Thus, the element mass matrices $A_h^{(k)}$ are given by:

$$
A_h^{(k)} = \frac{h_k}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix},
$$

and for equidistant subdivision :

$$
A_h^{(k)} = \frac{h}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}.
$$

For $h_k = h$ the mass matrix $A_h$ is of the following form:

$$A_h = \frac{h}{6}\begin{bmatrix} 2 & 1 & 0 & . & . & . & . & . & 0 \\ 1 & 4 & 1 & . & & & & & . \\ 0 & 1 & 4 & 1 & . & & & & . \\ . & . & 1 & 4 & 1 & . & & & . \\ . & & . & . & . & . & . & & . \\ . & & & . & . & . & . & . & . \\ . & & & & . & 1 & 4 & 1 & 0 \\ . & & & & & . & 1 & 4 & 1 \\ 0 & . & . & . & . & . & 0 & 1 & 2 \end{bmatrix}.$$

**Stiffness matrix $B_h$:**

$$\begin{aligned}(B_h\underline{w_h}, \underline{q_h}) &= b(w_h, q_h) = \sum_{T\in\mathcal{T}_h}\sum_{i,j} q_i w_j \int_T \varphi_j'\varphi_i' dx = \\ &= q_1 w_0 \int_T \varphi_0'\varphi_1' dx + q_1 w_1 \int_T \varphi_1'\varphi_1' dx \sum_{k=2}^{N_h-1}\left(B_h^{(k)}\begin{bmatrix} w_{k-1} \\ w_k \end{bmatrix}, \begin{bmatrix} q_{k-1} \\ q_k \end{bmatrix}\right) + \\ &+ q_{N_h-1} w_{N_h-1} \int_T \varphi_{N_h-1}'\varphi_{N_h-1}' dx + q_{N_h-1} w_{N_h} \int_T \varphi_{N_h}'\varphi_{N_h-1}' dx,\end{aligned}$$

where $i = 1, ..., N_h - 1$ and $j = 0, ..., N_h$.

For $k = 2, ..., N_h - 1$ the element stiffness matrices are given by

$$B_h^{(k)} = \begin{bmatrix} \int_{T_k}\varphi_{k-1}'^2(x)dx & \int_{T_k}\varphi_{k-1}'(x)\varphi_k'(x)dx \\ \int_{T_k}\varphi_k'(x)\varphi_{k-1}'(x)dx & \int_{T_k}\varphi_k'^2(x)dx \end{bmatrix}.$$

The matrix elements are calculated with the help of the reference element:

$$\begin{aligned}\int_{T_k}\varphi_{k-1}'^2(x)dx &= \int_{\hat{T}}\hat{\varphi}_0'^2(\xi)\frac{1}{h_k}d\xi = \int_{\hat{T}}\frac{1}{h_k}d\xi = \frac{\xi}{h_k}\Big|_0^1 = \frac{1}{h_k} \\ \int_{T_k}\varphi_{k-1}'(x)\varphi_k'(x)dx &= \int_{\hat{T}}\hat{\varphi}_0'(\xi)\hat{\varphi}_1'(\xi)\frac{1}{h_k}d\xi = \int_{\hat{T}}-\frac{1}{h_k}d\xi = -\frac{\xi}{h_k}\Big|_0^1 = -\frac{1}{h_k} = \\ &= \int_{T_k}\varphi_k'(x)\varphi_{k-1}'(x)dx \\ \int_{T_k}\varphi_k'^2(x)dx &= \int_{\hat{T}}\hat{\varphi}_1'^2(\xi)\frac{1}{h_k}d\xi = \int_{\hat{T}}\frac{1}{h_k}d\xi = \frac{\xi}{h_k}\Big|_0^1 = \frac{1}{h_k}\end{aligned}$$

Thus,

$$B_h^{(k)} = \frac{1}{h_k}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \underbrace{=}_{\text{for equidistant subdivision}} \frac{1}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.$$

The first and the last part of the sum which are resulting from the boundary conditions, are calculated as follows:

$$\begin{aligned} q_1 w_0 \int_T \varphi_0' \varphi_1' dx + q_1 w_1 \int_T \varphi_1' \varphi_1' dx &= \left( \begin{bmatrix} \int_{T_k} \varphi_0'(x)\varphi_1'(x)dx & \int_{T_k} \varphi_1'^2(x)dx \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}, q_1 \right) \\ &= \left( \begin{bmatrix} -\frac{1}{h} & \frac{1}{h} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}, q_1 \right), \\ q_{N_h-1} w_{N_h-1} \int_T \varphi_{N_h-1}' \varphi_{N_h-1}' dx &+ q_{N_h-1} w_{N_h} \int_T \varphi_{N_h}' \varphi_{N_h-1}' dx = \\ &= \left( \begin{bmatrix} \int_{T_k} \varphi_{N_h-1}'^2(x)dx & \int_{T_k} \varphi_{N_h}'(x)\varphi_{N_h-1}'(x)dx \end{bmatrix} \begin{bmatrix} w_{N_h-1} \\ w_{N_h} \end{bmatrix}, q_{N_h-1} \right) \\ &= \left( \begin{bmatrix} \frac{1}{h} & -\frac{1}{h} \end{bmatrix} \begin{bmatrix} w_{N_h-1} \\ w_{N_h} \end{bmatrix}, q_{N_h-1} \right). \end{aligned}$$

Thus, $B_h^{(1)} = \begin{bmatrix} -\frac{1}{h} & \frac{1}{h} \end{bmatrix}$ and $B_h^{(N_h)} = \begin{bmatrix} \frac{1}{h} & -\frac{1}{h} \end{bmatrix}$.

The $m \times n$ matrix $B_h$ is obtained by assembling all element stiffness matrices. For equidistant subdivision it is of the following form:

$$B_h = \frac{1}{h} \begin{bmatrix} -1 & 2 & -1 & 0 & . & . & . & . & . & . & 0 \\ 0 & -1 & 2 & -1 & . & & & & & & . \\ . & . & . & . & . & . & & & & & . \\ . & & . & . & . & . & . & & & & . \\ . & & & . & . & . & . & . & & & . \\ . & & & & . & . & . & . & . & & . \\ . & & & & & . & . & . & . & . & . \\ . & & & & & & . & -1 & 2 & -1 & 0 \\ 0 & . & . & . & . & . & . & 0 & -1 & 2 & -1 \end{bmatrix}.$$

**Load vector $\underline{g_h}$:**

$$\underline{g_h} = \langle G, \varphi_i \rangle \qquad \forall i = 1, ..., N_h - 1$$

with

$$\begin{aligned} \langle G, \varphi_i \rangle &= -\int_\Omega f(x)\varphi_i(x)dx = -\sum_{T_k \in \mathcal{T}_h} \int_{T_k} f(x)\varphi_i(x)dx = \\ &= -\sum_{k=2}^{N_h-1} \int_{x_{k-1}}^{x_k} f(x)\varphi_i(x)dx \qquad \forall i = 1, ..., N_h - 1. \end{aligned}$$

For $k = 2, ..., N_h - 1$:

$$\int_{T_k} f(x)\varphi_i(x)dx = \begin{bmatrix} \int_0^1 f(F_k(\xi))\varphi_{k-1}(F_k(\xi))h_k d\xi \\ \int_0^1 f(F_k(\xi))\varphi_k(F_k(\xi))h_k d\xi \end{bmatrix} = \begin{bmatrix} \int_0^1 f(F_k(\xi))(1-\xi)h_k d\xi \\ \int_0^1 f(F_k(\xi))\xi h_k d\xi \end{bmatrix}$$

$$\underbrace{=}_{\text{trapezoidal rule}} \frac{h_k}{2} \begin{bmatrix} f(F_0(\xi)) \\ f(F_1(\xi)) \end{bmatrix} = \frac{h_k}{2} \begin{bmatrix} f(x_{k-1}) \\ f(x_k) \end{bmatrix} \underbrace{=}_{\text{for equidistant subdivision}} \frac{h}{2} \begin{bmatrix} f(x_{k-1}) \\ f(x_k) \end{bmatrix}.$$

Because of the Dirichlet boundary conditions the first and the last element load vector elements are given as follows:

$$g_h^{(1)} = \frac{h}{2} f(x_1),$$
$$g_h^{(N_h-1)} = \frac{h}{2} f(x_{N_h-1}).$$

For equidistant subdivision the following m-dimensional vector is obtained:

$$\underline{g_h} = \frac{h}{2} \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_{N_h-1}) \end{bmatrix}.$$

## 3.2 Preconditioners

In the following, the solution of the beam bending problem will be determined, first without using a preconditioner (①) and then with the help of an exact preconditioner(②) as well as approximate preconditioners (③ to ⑤). We will test these preconditioners for different mesh sizes h and analyze their convergence behavior.

**Exact Preconditioner:**

The considered block diagonal exact preconditioner (②) is of the form

$$\begin{bmatrix} A & \\ & -S \end{bmatrix},$$

where $S = -BA^{-1}B^T$ denotes the Schur complement. For this choice of the preconditioner the Propositions and Remarks discussed in Section 2.1 are holding.

**Approximate Preconditioners:**

Then, we will continue studying the iteration numbers of three approximate preconditioners of the form

$$\begin{bmatrix} \hat{A} & \\ & -\hat{S} \end{bmatrix}.$$

- In the first case (③) we choose

$$\begin{aligned} \hat{A} &= D = lumpA = diag\left(\sum_{j=1}^{n} a_{ij}\right),\ i = 1, ..., n, \\ \hat{S} &= -BD^{-1}B^T. \end{aligned}$$

- Then we will consider (④) the matrices

$$\begin{aligned} \hat{A} &= D, \\ \hat{S} &= -K^2, \end{aligned}$$

where K denotes the $m \times m$ stiffness matrix received when considering the finite element discretization of the problem

$$\begin{aligned} -u''(x) &= g(x),\ x \in (0,1), \\ u(0) &= u(1) = 0. \end{aligned}$$

After simple but straightforward calculations we obtain for $\underline{u_h} \in \mathbb{R}^m$ and $\underline{w_h} \in \mathbb{R}^m$:

$$(K_h \underline{u_h}, \underline{w_h}) = \sum_{T \in \mathcal{T}_h} \sum_{i,j} u_i w_j \int_T \varphi_j' \varphi_i' dx = \sum_{k=2}^{N_h - 1} \left( K_h^{(k)} \begin{bmatrix} u_{k-1} \\ u_k \end{bmatrix}, \begin{bmatrix} w_{k-1} \\ w_k \end{bmatrix} \right),$$

where $i = 1, ..., N_h - 1$ and $j = 1, ..., N_h - 1$. The summation of the element stiffness matrices $K_h^{(k)}$ finally leads to the $m \times m$ stiffness matrix $K_h$:

$$K_h = \frac{1}{h} \begin{bmatrix} 2 & -1 & 0 & . & . & . & . & . & 0 \\ -1 & 2 & -1 & . & & & & & . \\ 0 & -1 & 2 & -1 & . & & & & . \\ . & . & . & . & . & . & & & . \\ . & & . & . & . & . & . & & . \\ . & & & . & . & . & . & . & . \\ . & & & & . & -1 & 2 & -1 & 0 \\ . & & & & & . & -1 & 2 & -1 \\ 0 & . & . & . & . & . & 0 & -1 & 2 \end{bmatrix}.$$

- And finally (⑤), we have a look at

$$\begin{aligned} \hat{A} &= D, \\ \hat{S} &= -K \tilde{D}^{-1} K, \end{aligned}$$

with $\left( K \tilde{D}^{-1} K e_i, e_i \right) = \left( B D^{-1} B^T e_i, e_i \right)$, where $e_i = (0, ..., 0, 1, 0, ..., 0)^T$.

### 3.3 Numerical results

For solving the matrix system (18), we use the minimal residual method. In MATLAB this method is implemented in the function `minres`. Here, we study the behavior of the iteration process, so we fix the tolerance `tol` by the value $10^{-7}$ and the maximum number of iterations `maxit` by $10^6$. However, we mention that in practice we have to adapt the iteration error to the discretization error. To be able to compare the obtained convergence results for different mesh sizes, we choose the function f(x), which is needed for the calculation of the vector g, as constant, here f(x)=8. The starting vector is an all zero vector. The remaining input values for `minres` are generated by the custom function

```
function[G,b,M,PE,P,Q,R] = matrprecsetup(Nh),
```

which calculates the stiffness matrix B, the mass matrix A and the load vector g. Further, it generates the system matrix $G = \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix}$ and the system vector of the form $\begin{bmatrix} 0 \\ g \end{bmatrix}$. Additionally, this function builds the preconditioners ① to ⑤, which are denoted as P1, P2, P3, P4, P5 in the code. One can look up the implementation of the function in Appendix II.

For the mesh sizes $h_i = \frac{1}{25*2^i}$, where $i \in \{1, 2, 3, 4, 5, 6\}$ we obtain the following iteration numbers for the preconditioners ① to ⑤:

| h | ① | ② | ③ | ④ | ⑤ |
|---|---|---|---|---|---|
| $\frac{1}{50}$ | 108 | 1 | 19 | 17 | 31 |
| $\frac{1}{100}$ | 350 | 1 | 21 | 15 | 35 |
| $\frac{1}{200}$ | 1198 | 1 | 21 | 15 | 37 |
| $\frac{1}{400}$ | 4648 | 1 | 22 | 17 | 43 |
| $\frac{1}{800}$ | 19005 | 3 | 23 | 19 | 47 |
| $\frac{1}{1600}$ | 82860 | 3 | 23 | 21 | 53 |

Thus, solving the problem (18), without applying a preconditioner to the matrix system, needs by far the most iterations among the other preconditioners. As we can see in Figure 4, for ① a dependence of the number of iterations on the mesh size h can be detected. In Figure 4, the

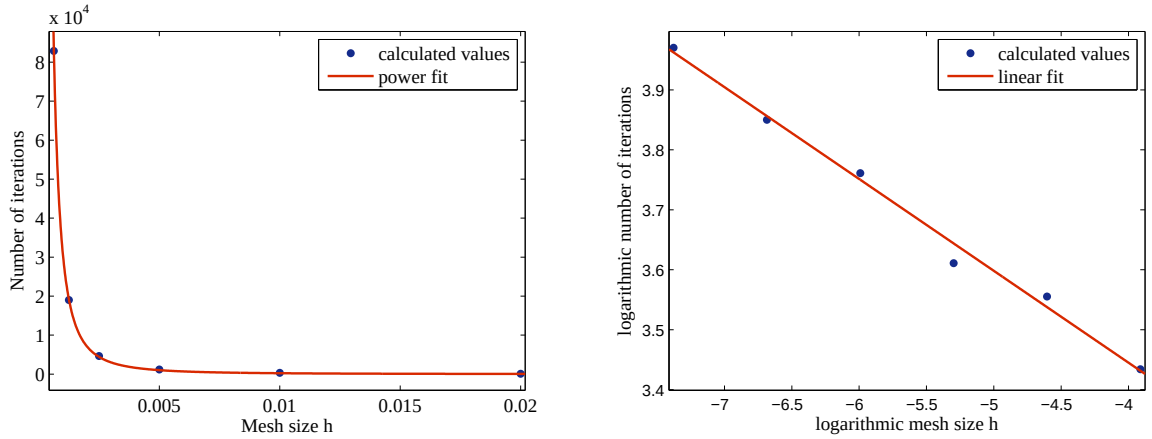


Figure 4: Whereas in the first figure the number of iterations versus the mesh size h for ① are plotted linearily, the second figure makes use of logarithmic axes to show the h-dependence of the iterations.

iterations *iter* for ① concerning the corresponding mesh sizes h as denoted in the table are plotted first with linear axes and then with logarithmic axes. As we want to find out about h-dependence of the iteration numbers, we have to consider the equation: $iter = a * h^b = O(h^b)$, with $a, b \in \mathbb{R}$. Using the MATLAB fitting tool power fit, we obtain the coefficients a,b with $a = 0.014 \pm 0.0034$ and $b = -2.114 \pm 0.033$. When using logarithmic axes and considering a fit through the given points, a linear function is attained. Because $iter = a * h^b$ is equivalent to $log(iter) = log(a) + b * log(h)$, b can be directly found by reading off the slope of the attained linear function. Applying the linear fitting tool in MATLAB results in the right graph of Figure 4 with $b = -1.919 \pm 0.115$ and $a_1 = -2.957 \pm 0.664$ in this case. So summarizing, we found out that $iter = O(h^{-2})$, approximately.

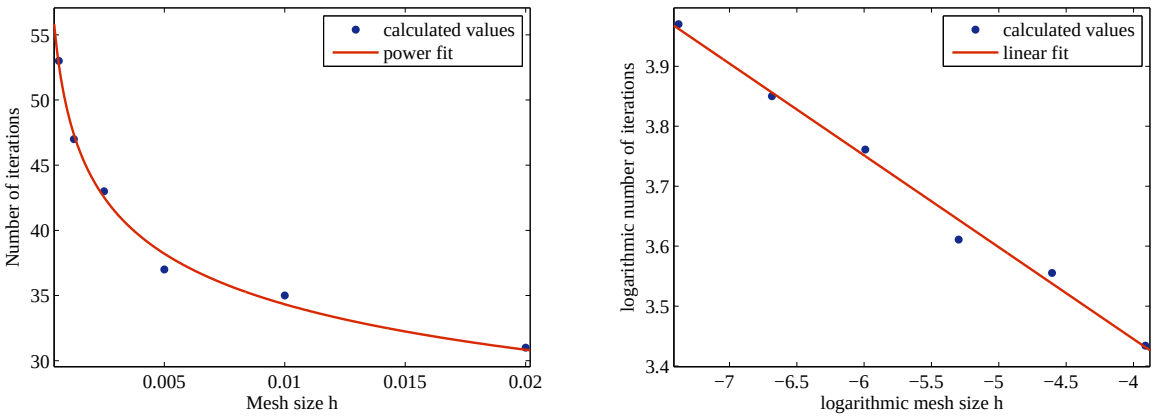


Figure 5: In this figure, we first consider the number of iterations and the associated mesh sizes h linearly and then logarithmically.

As already mentioned in Remark 2.4 the number of iterations for the exact preconditioner ② does not exceed three. Thus, when applying ② to the system (18), `minres` converges for $h = \frac{1}{25*2^i}$ for $i \in \{1,2,3,4\}$ in one iteration and for $i \in \{5,6\}$ in three iterations. When having a closer look to this convergence behavior, we can discover, that the number of iterations changes between $N_h = 409$, and $N_h = 410$. The eigenvalues of the preconditioned system matrix are in this case given by 1 and $\frac{1}{2} \pm \frac{\sqrt{5}}{2}$, but the MATLAB calculation shows slight deviations from zero concerning the imaginary part of the eigenvalues (see Figure 6). This behaviour can be attributed to the rounding error which is a result of matrix inversion. When comparing the eigenvalues obtained for exact preconditioning with those received for approximate preconditioning with preconditioner ③, we can discover three clusters around the eigenvalues calculated for the exact preconditioned system matrix (see Figure 7). The iteration numbers for the approximate preconditioner ③ are located between 19 and 23 for h values from $\frac{1}{50}$ to $\frac{1}{1600}$ and for the preconditioner ④ they are arranged between 15 and 21. Whereas the iteration numbers of ③ and ④ are not of the form $O(h^d)$, this behavior is again fullfilled for the approximate preconditioner ⑤. Analogously to the analysis for ①, we can find out the h-dependence for ⑤. As Figure 5 shows, the power function generated by MATLAB is given by $f(x) = 16.84 * x^{-0.1362}$ and the linear function, which is obtained for using logarithmic axes is of the form $g(x) = -0.1532 * x + 2.832$, thus $iter = O(h^{-0.15})$, approximately.

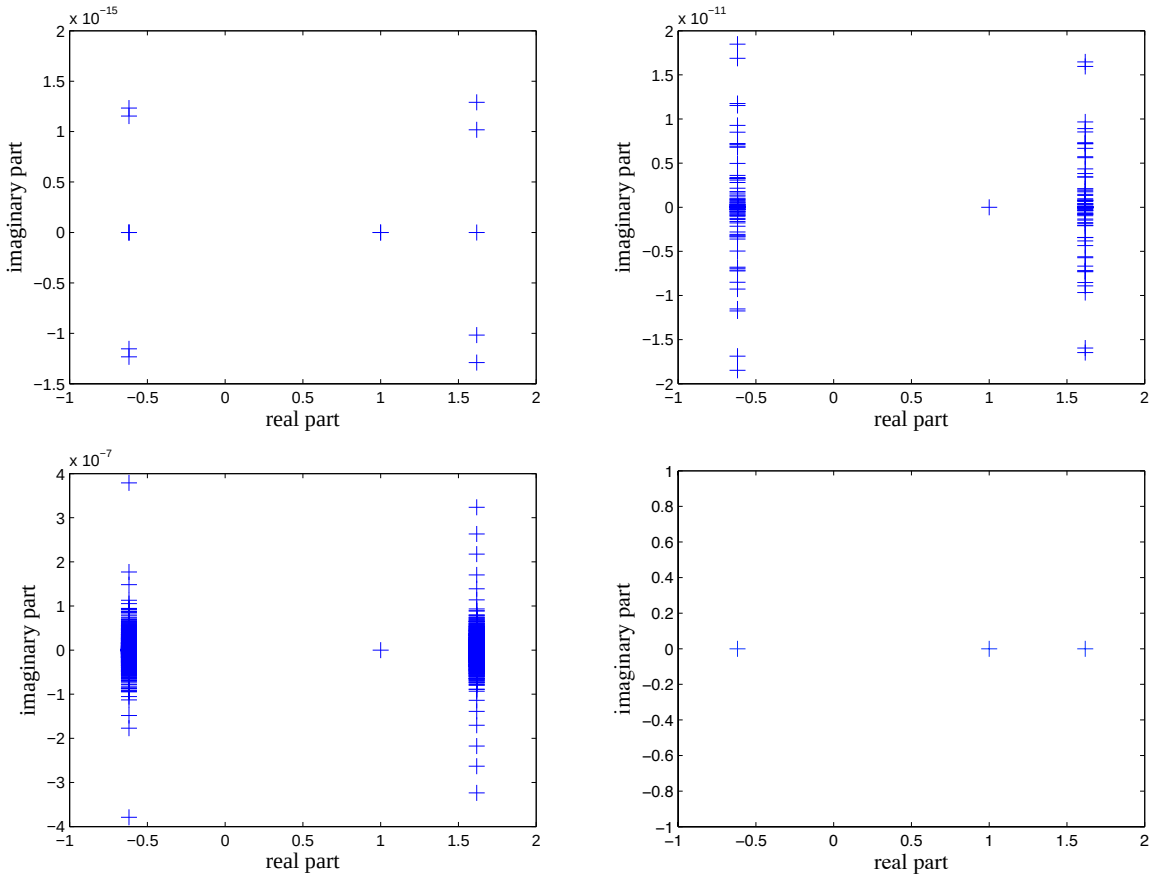


Figure 6: The first row contains eigenvalue plots of the exact preconditioned system matrix for $N_h = 10$ and $N_h = 100$. In the second row we can see two eigenvalue plots for $N_h = 1000$ with different scalings. The x-axis of the graphs describes the real part of the eigenvalues, whereas the y-axis describes the imaginary part.

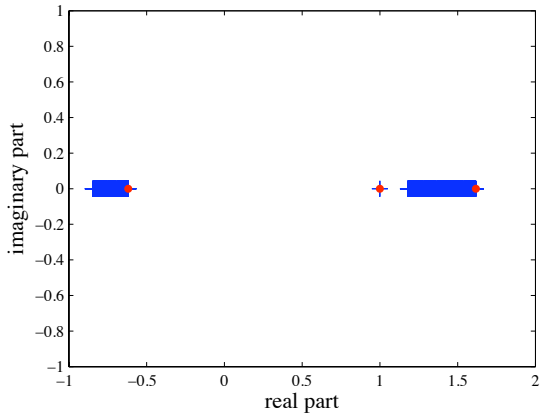


Figure 7: Eigenvalues of the exact preconditioned system matrix (points) in comparison to the eigenvalues for the system matrix, when using preconditioner ③ (crosses) for $N_h = 1000$.

## References


[1] M. Benzi, G. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, 2005.

[2] E. de Sturler and J. Liesen. Block-diagonal and constraint preconditioners for nonsymmetric indefinite linear systems. Part I: Theory. *SIAM J. Sci. Comput.*, 26(5):1598–1619, 2004.

[3] U. Langer. Zur numerischen Lösung des ersten biharmonischen Randwertproblems. *Numerische Mathematik*, 50(3):291–310, 1987.

[4] Malcolm F. Murphy, Gene H. Golub, and Andrew J. Wathen. A note on preconditioning for indefinite linear systems. *SIAM J. Sci. Comput.*, 21(6):1969–1972, 1999.

# A Appendix

## Appendix I

Here, we test the preconditioners ① to ⑤ for the mesh sizes $h_i = \frac{1}{25*2^i}$, where $i \in \{1,2,3,4,5,6\}$ for fixed values `tol` and `maxit` and generate a matrix, containing all obtained numbers of iterations.

```
k = 6;
tol = 0.0000001;
maxit = 1000000;
points(1:k,1:5) = 0;
for i = 1:k
    [G,b,P1,P2,P3,P4,P5] = matrprecsetup(25*2^i);
    [x0,flag0,relres0,iter0] = minres(G,b,tol,maxit,P1);
    [x1,flag1,relres1,iter1] = minres(G,b,tol,maxit,P2);
    [x2,flag2,relres2,iter2] = minres(G,b,tol,maxit,P3);
    [x3,flag3,relres3,iter3] = minres(G,b,tol,maxit,P4);
    [x4,flag4,relres4,iter4] = minres(G,b,tol,maxit,P5);
    points(i,1) = iter0;
    points(i,2) = iter1;
    points(i,3) = iter2;
    points(i,4) = iter3;
    points(i,5) = iter4;
end
```

## Appendix II

As described in Subsection 3.2, this function generates the preconditioners ① to ⑤, as well as the system matrix and the system vector, which are needed as input values for the function `minres`, see Appendix A.

```
function [G,b,P1,P2,P3,P4,P5] = matrprecsetup(Nh)

n = Nh+1;
m = n - 2;
h = 1/Nh;
f = @(x)8;

%stiffness matrix B
B = sparse(n,m);
B(1,1) = -1/h;
B(2,1) = 2/h;
B(2,2) = -1/h;
B(n-1,m-1) = -1/h;
B(n-1,m) = 2/h;
B(n,m) = -1/h;
if n>2
```

```
    for i=3:(n-2)
        B(i,i) = -1/h;
        B(i,i-1) = 2/h;
        B(i,i-2) = B(i,i);
    end
end

%mass matrix A
A = sparse(n,n);
A(1,1) = h/3;
A(1,2) = h/6;
A(n,n) = h/3;
A(n,n-1) = h/6;
if n>2
    for i=2:(n-1)
        A(i,i) = 2*h/3;
        A(i,i+1) = h/6;
        A(i,i-1) = A(i,i+1);
    end
end

%system matrix G
% A  B
% B' 0
G = sparse(n+m,n+m);
G(1:n,1:n) = A;
G(1:n,(n+1):(n+m)) = B;
G((n+1):(n+m),(1:n)) = B';
G((n+1):(n+m),(n+1):(n+m)) = 0;

%load vector g
g(1:m,1)=0;
for j=1:m
    g(j,1)=h*f((j)/(n-1));
end

%system vector
b(1:n,1)=0;
for i=1:m
    b(n+i,1)=g(i,1);
end

%Preconditioners:
%P1:
P1=[];

%P2:
```

```
S = -(B')*inv(A)*B;
P2 = sparse(n+m,n+m);
P2(1:n,1:n) = A;
P2(1:n,(n+1):(n+m)) = 0;
P2((n+1):(n+m),(1:n)) = 0;
P2((n+1):(n+m),(n+1):(n+m)) = -S;

%P3:
P3 = sparse(n+m,n+m);
D = diag(sum(A));
X = -B'*inv(D)*B;
P3(1:n,1:n) = D;
P3(1:n,(n+1):(n+m)) = 0;
P3((n+1):(n+m),(1:n)) = 0;
P3((n+1):(n+m),(n+1):(n+m)) = -X;

%P4:
K = sparse(m,m);
K(1:m,1:m)=B(2:(m+1),1:m);
Y = -K^2;
P4 = sparse(n+m,n+m);
P4(1:n,1:n) = D;
P4(1:n,(n+1):(n+m)) = 0;
P4((n+1):(n+m),(1:n)) = 0;
P4((n+1):(n+m),(n+1):(n+m)) = -Y;

%P5:
br = h^2*diag(B'*inv(D)*B);
T = sparse(m,m);
T(1,1) = 4;
T(1,2) = 1;
T(m,m-1) = 1;
T(m,m) = 4;
for i=2:(m-1)
    T(i,i-1) = 1;
    T(i,i) = 4;
    T(i,i+1) = 1;
end
dloes = T\br;
Dloes = sparse(m,m);
for i=1:m
    Dloes(i,i)= dloes(i);
end
Z = -K*Dloes*K;
P5 = sparse(n+m,n+m);
P5(1:n,1:n) = D;
P5(1:n,(n+1):(n+m)) = 0;
```

```
P5((n+1):(n+m),(1:n)) = 0;
P5((n+1):(n+m),(n+1):(n+m)) = -Z;
```