Sharpened CG Iteration Bound for High-contrast Heterogeneous Scalar Elliptic PDEs

Going beyond condition number

WI5005: Thesis Project (Interim Thesis)

Philip Soliman

Delft University of Technology

TUDelft

Sharpened CG Iteration Bound for High-contrast Heterogeneous Scalar Elliptic PDEs

Going beyond condition number

by

Philip Soliman

To obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on T.B.A.

Student number:	4945255	
Project duration:	December 2024 – September 2025	
Thesis committee:	Prof. H. Schuttelaars,	TU Delft, responsible supervisor
	Dr. A. Heinlein,	TU Delft, daily supervisor
	F. Cumaru,	TU Delft, daily co-supervisor
Faculty:	Faculty of Electrical Engineering, Mathematics and Computer Science	
Department:	Delft Institute of Applied Mathematics (DIAM)	

Published on Wednesday 21st May, 2025 .

Generative AI was used to assist in the writing of this thesis. The author is responsible for the content and accuracy of the work.

TU Delft

Contents

Nomenclature

Symbols

Table 1: Symbols related to the heterogeneous elliptic problem.

Symbol	Description
Ω	Bounded domain in $\mathbb{R}^d$ with Lipschitz boundary.
$\partial\Omega$	Boundary of Ω.
$\mathbb{R}^d$	d-dimensional Euclidean space.
C	Scalar coefficient in the Darcy problem, assumed to lie in $L^\infty(\Omega)$.
u	Exact solution of the elliptic problem.
f	Source term belonging to $L^2(\Omega)$.
u_D	Dirichlet boundary data.
$\kappa_{\min}$	Lower bound of the scalar coefficient κ.
$\kappa_{\max}$	Upper bound of κ.
u_h	Finite element approximation of u.
V_h	Finite-dimensional subspace of $H_0^1(\Omega)$.
$\{\phi_i\}_{i=1}^n$	Basis functions spanning V_h.
A	Stiffness matrix derived from the Galerkin method.
b	Load vector in the resulting linear system.
v_h	Test function in V_h.

Table 2: Symbols related to the Conjugate Gradient (CG) method.

Symbol	Description
$\mathbf{u}_0$	Initial guess for the solution.
$\mathbf{r}_0$	Initial residual defined as $\mathbf{b} - A\mathbf{u}_0$.
$\mathbf{r}_j$	Residual vector at the j^{th} iteration.
$\mathbf{p}_j$	Search direction at iteration j.
α_j	Step size computed at iteration j.
β_j	Coefficient used to update the search direction in iteration j.
$K_m(A, \mathbf{r}_0)$	Krylov subspace spanned by $\{\mathbf{r}_0, A\mathbf{r}_0, A^2\mathbf{r}_0, \ldots, A^{m-1}\mathbf{r}_0\}$.
T_m	Tridiagonal Hessenberg matrix arising from the Lanczos process.
δ_j	Diagonal entries of T_m.
η_j	Off-diagonal entries of T_m.
$\mathbf{e}_m$	Error at iteration m, defined as $\mathbf{u}^* - \mathbf{u}_m$.
P_{m-1}	Space of polynomials of degree at most $m-1$.
λ_i	Eigenvalues of A.
ξ_i	Components of the initial error in the eigenvector basis of A.
$\sigma(A)$	Spectrum (set of eigenvalues) of A.
C_m	Chebyshev polynomial of degree m.
ρ_0	Initial residual in the eigenvector basis of A.

Table 3: Symbols related to Schwarz preconditioners.

Symbol	Description
Ω_i	Subdomains obtained from partitioning Ω.
R_i	Restriction operator for subdomain Ω_i.

Symbol	Description
D_i	Diagonal matrix representing the partition of unity (weights) for Ω_i.
N_{sub}	Number of subdomains.
M_{ASM}^{-1}	Additive Schwarz preconditioner defined by $M_{\text{ASM}} = \sum_{i=1}^{N_{sub}} R_i^T (R_i A R_i^T)^{-1} R_i$.
M_{RAS}^{-1}	Restrictive additive Schwarz preconditioner defined by $M_{\text{RAS}} = \sum_{i=1}^{N_{sub}} R_i^T D_i (R_i A R_i^T)^{-1} R_i$.
R_0	Restriction operator associated with the coarse space.
P_j	Local projection operator associated with subdomain Ω_j.
P_0	Projection operator for the coarse space.
P_{ad}	Sum of the projection operators, $P_{ad} = \sum_{j=1}^{N_{sub}} P_j$, used in the two-level method.
$\kappa(P_{ad})$	Condition number of the preconditioned system, given by $\frac{\lambda_{\max}}{\lambda_{\min}}$ of P_{ad}.

Table 4: Symbols related to eigenspectra and CG convergence bounds (chapter 5: Preliminary Results).

Symbol	Description
$\sigma_1(A)$	Two-cluster eigenspectrum of A, defined as the union of two intervals $[a,b] \cup [c,d]$.
$\sigma_2(A)$	Eigenspectrum comprising a main cluster and a tail of eigenvalues, with the tail denoted by N_{tail}.
$[a,b]$	Interval containing the first cluster of eigenvalues.
$[c,d]$	Interval containing the second cluster of eigenvalues.
N_{tail}	Number of eigenvalues in the tail cluster (when the spectrum has one cluster plus a tail).
$\hat{r}_p^{(i)}(x)$	Residual polynomial function for the i^{th} cluster, defined piecewise (for $i=1$, via a scaled expression; for $i=2$, as a product over tail eigenvalues).
$\hat{r}_{\bar{m}-p}(x)$	Residual polynomial function based on Chebyshev polynomials, corresponding to the complementary polynomial degree $\bar{m}-p$.
$C_p^{(i)}$	Cluster-specific Chebyshev polynomial of degree p, adapted to the eigenvalue distribution of the i^{th} cluster.
η_1	Upper bound for $\max_{x\in[a,b]} \lvert\hat{r}_p^{(1)}(x)\rvert$, given by $2\left(\frac{\sqrt{b}-\sqrt{a}}{\sqrt{b}+\sqrt{a}}\right)^p$.
η_2	Upper bound for $\max_{x\in[a,b]} \lvert\hat{r}_p^{(2)}(x)\rvert$, expressed as $\left(\frac{b}{a}-1\right)^p$ when $p = N_{\text{tail}}$.
$\bar{m}$	Total degree of the composite residual polynomial $r_{\bar{m}}$, formed as the product $\hat{r}_p^{(i)}(x)\hat{r}_{\bar{m}-p}(x)$.
ϵ	Relative error, defined as $\lVert e_m\rVert_A / \lVert e_0\rVert_A$ at the m^{th} iteration.
$z_1^{(i,j)}, z_2^{(j)}$	Chebyshev coordinates in the frame of reference of the i^{th} cluster
k	Positive integer parameter in the Chebyshev inequality, corresponding to the degree in the bound estimate.
p_i	Chebyshev degree associated with the i^{th} eigenvalue cluster, determining the contraction factor of that cluster's contribution to the residual.
κ_i	Condition number of the i^{th} eigenvalue cluster, defined as the ratio of the largest to smallest eigenvalue within the cluster.

Symbol	Description
f_i	Convergence factor (or spectral measure) for the i^{th} cluster.

Abbreviations

Table 5: List of abbreviations and their full meanings.

Abbreviation	Full Meaning
CG	Conjugate gradient
PCG	Preconditioned conjugate gradient
SPD	Symmetric positive definite
FEM	Finite element method
ASM	Additive Schwarz method
RAS	Restrictive additive Schwarz
ORAS	Optimized restrictive additive Schwarz
MsFEM	Multiscale finite element method
ACMS	Approximate component mode synthesis
GDSW	Generalized Dryja-Smith-Widlund
AMS	Algebraic multiscale solver
DtN	Dirichlet-to-Neumann
DBC	Dirichlet boundary condition
NBC	Neumann boundary condition
PDE	Partial differential equation
DOF(s)	Degree(s) of freedom

1 Introduction

In this thesis we focus on a simple scalar diffusion problem. Let $\Omega \subset \mathbb{R}^d$ be a bounded domain with Lipschitz boundary $\delta\Omega$, $\mathcal{C} \in L^\infty(\Omega)$ be scalar field defined on Ω such that $0 < \mathcal{C}_{\min} \leq \mathcal{C}(x) \leq \mathcal{C}_{\max} < \infty$ for all $x \in \Omega$ and $f \in L^2(\Omega)$ be a source term, then we define

Problem 1.1: High-contrast scalar elliptic problem: strong formulation

Let $u_D \in H^2(\delta\Omega)$ be a Dirichlet boundary condition. Find $u \in H^2(\Omega)$ such that

$$\begin{aligned} -\nabla \cdot (\mathcal{C}\nabla u) &= f \quad \text{in } \Omega, \\ u &= u_D \quad \text{on } \delta\Omega. \end{aligned} \tag{1.1}$$

The first step in solving Problem 1.1 is to reformulate the problem in a way that reduces the regularity constraint on the solution $u \in H^2(\Omega)$ and not requiring to $\nabla \cdot (\mathcal{C}\nabla u)$ to exist pointwise. This leads to the weak formulation of the problem, which is obtained by multiplying equation (1.1) with a test function $v \in H^1_0(\Omega)$ and integrating over Ω. The weak formulation is given by

Problem 1.2: High-contrast scalar elliptic problem: weak formulation

Let $u_D \in H^1(\delta\Omega)$ be a Dirichlet boundary condition. Find $u \in V = \{u \in H^1(\Omega) | u_{\delta\Omega} = u_D\}$ such that $\forall v \in H^1_0(\Omega)$

$$\int_\Omega \mathcal{C}\nabla u \cdot \nabla v \, dx = \int_\Omega f v \, dx. \tag{1.2}$$

To solve this problem numerically, we need to discretize the domain Ω and the solution space V. To that end we consider a triangulation $\mathcal{T}$ of the domain Ω with the $\mathcal{N}$ the set of degrees of freedom (DOFs). Then, we pick a finite dimensional subspace of V, V_h spanned by a set of basis functions ϕ_i defined locally on each of the elements $\tau \in \mathcal{T}$

$$V_h = \text{span}\{\phi_i\}_{i=1}^n,$$

where $n = |\mathcal{N}|$. This leads to the discretized weak formulation

Problem 1.3: High-contrast scalar elliptic problem: discretized weak formulation

Let $u_D \in H^1(\delta\Omega)$ be a Dirichlet boundary condition. Find $u_h \in V_h$ such that $\forall v_h \in V_{h,0} = V_h \cap H_0^1(\Omega)$

$$a(u_h, v_h) = \int_\Omega \mathcal{C}\nabla u_h \cdot \nabla v_h \, dx = \int_\Omega f v_h \, dx = (f, v_h), \tag{1.3}$$

where $a(u_h, v_h)$ is the bilinear form and (f, v_h) is the linear form. equation (1.3) gives rise to the following system of equations

$$A\mathbf{u} = \mathbf{b}, \quad A_{ij} = a(\phi_j, \phi_i), \; b_i = (f, \phi_i) \quad \forall i, j \in \mathcal{N},$$

where $A \in \mathbb{R}^{n\times n}$ is the (by construction) symmetric stiffness matrix, $\mathbf{u} \in \mathbb{R}^n$ the solution vector with components u_i and $\mathbf{b} \in \mathbb{R}^n$ the load vector. The load vector $\mathbf{b}$ is constructed from the source term f and the boundary conditions. The approximate solution u_h is constructed from the basis functions ϕ_i as

$$u_h = \sum_{i\in\mathcal{N}} u_i \phi_i,$$

Note that the bilinear form a in Problem 1.3 is positive definite, meaning that for all $0 \neq w \in H_0^1(\Omega)$ we have

$$a(w, w) = \int_\Omega \mathcal{C}\nabla w \cdot \nabla w \, dx = \int_\Omega \mathcal{C}|\nabla w|^2 \, dx > 0,$$

since $\mathcal{C} > 0$. Moreover, for $w = \sum_{i\in\mathcal{N}} w_i\phi_i$ with $\mathbf{w} = (w_1, w_2, \dots, w_n)^T \neq \mathbf{0}$ we have

$$\mathbf{w}^T A\mathbf{w} = a(w, w) > 0.$$

It follows that A is positive definite, making it symmetric positive definite (SPD). This means that the numerical problem Problem 1.3 is well-posed and has a unique solution $\mathbf{u}$ for any given load vector $\mathbf{b}$.

Apart from possibly complex domains Ω, a major obstacle in solving the linear system $A\mathbf{u} = \mathbf{b}$ from Problem 1.3, comes from its high-contrast coefficient $\mathcal{C}$, which requires a broad range of element sizes $|\tau|$ in the triangulation $\mathcal{T}$ to fully resolve. As a result, the number of DOFs $n = |\mathcal{N}|$ can be very large, leading to a system matrix A that is large and sparse. This makes direct methods like Gaussian elimination, LU- or Cholesky decomposition impractical, as they require storing the entire matrix in memory and generally have complexity $\mathcal{O}(n^3)$.

Though A is large and sparse, it *is* SPD. Therefore, the linear system $A\mathbf{u} = \mathbf{b}$ can be solved using CG. CG requires only the ability to compute matrix-vector products with A (complexity $\mathcal{O}(n)$ for sparse matrices) and does not require storing the entire matrix. Being an iterative method, CG produces a sequence of approximations $\mathbf{u}_i, \; i = 1, \dots, m$ to the solution $\mathbf{u}$ and stops when some convergence criterion depending on a desired tolerance ϵ is met. This means that CG's complexity is given by $\mathcal{O}(mn)$.

Hence, the number of iterations m required for convergence is a crucial factor in the performance of CG. The key subject of section 2.1.3 is to analyze the convergence of CG and how it depends on the properties of the system matrix A. In particular, we will show that m is related to the distribution of the eigenvalues of A. For instance, in exact arithmetic m is equal to the number of distinct eigenvalues of A, say k, from which follows that CG's complexity is given by $\mathcal{O}(kn)$. In general, we can derive, using a well-known bound on CG's convergence rate discussed in Theorem 2.5, an explicit expression for CG's complexity

$$\mathcal{O}\left(\sqrt{\kappa(A)}\log\left(\frac{1}{\epsilon}\right)n\right), \tag{1.4}$$

where $\kappa(A)$ is the condition number of A. Comparing equation (1.4) with the complexity of direct methods, we see that CG is much more efficient for large sparse SPD systems like A.

However, the difficulty of allowing for complex domains and a high-contrast coefficient $\mathcal{C}$ resides in that the condition number $\kappa(A)$ can be very large, which in turn increases CG's iteration count and complexity. Handling of complex domains can be done using *domain decomposition methods* and this is the topic of section 2.2. On the other hand, accounting for the high-contrast coefficient $\mathcal{C}$ is the topic of recent literature and concerns the construction of robust *coarse spaces*, some examples of which are given in chapter 3.

The particular implementation of domain decomposition method and coarse space results in a preconditioner matrix $M \in \mathbb{R}^{n\times n}$, which is used to transform the system $A\mathbf{u} = \mathbf{b}$ into a new system $M^{-1}A\mathbf{u} = M^{-1}\mathbf{b}$ with a (hopefully) smaller condition number. The preconditioned CG method (PCG), described in section 2.1.5, is then used to solve the transformed system. Consequently, using the equivalent of the CG complexity bound equation (1.4) for PCG, we can determine the performance of PCG with the preconditioner M as

$$\mathcal{O}\left(\sqrt{\kappa(M^{-1}A)}\log\left(\frac{1}{\epsilon}\right)n\right). \tag{1.5}$$

This thesis seeks to challenge the applicability of the bound in equation (1.5) to problems like Problem 1.1. The bound relies on an overestimation of the actual number of iterations m required for convergence and overstates the role of the condition number $\kappa(M^{-1}A)$ in determining the convergence rate of CG. We will see in section 2.1.4 that the actual number of iterations m is much smaller than the classical bound that equation (1.5) relies on and that the condition number $\kappa(M^{-1}A)$ is not the only factor influencing the convergence rate of CG.

This thesis answers the following central question:

Can we derive more precise bounds for the convergence rate of the Conjugate Gradient (CG) method in high-contrast scalar elliptic problems, improving upon the classical condition number based bound?

To answer this central question, the thesis explores several sub-questions:

1. How does the distribution of eigenvalues of the stiffness matrix A affect the number of iterations required for CG convergence in high-contrast scalar elliptic problems?

2. What is the impact of preconditioning on the number of CG iterations in high-contrast scalar elliptic problems, and how does it compare to the classical bound based on the condition number?

3. How does the contrast in the coefficient $\mathcal{C}$ influence the condition number of the preconditioned system and the number of CG iterations required for convergence?

4. Can we derive an improved bound for the CG convergence rate in high-contrast scalar elliptic problems that does not rely solely on the condition number of the system?

The hope is that addressing these questions leads to a richer understanding of the convergence rate of the (P)CG method in the specific case of high-contrast problems and, subsequently, aid in the selection of suitable preconditioners M.

This thesis is organized as follows. In chapter 2, the mathematical background of the CG method and Schwarz methods is introduced. chapter 3 reviews related work, providing context for the current study and highlighting differences between existing approaches. Chapter 4 details the research questions, motivation, and challenges associated with refining the CG iteration bound. Chapter 5 presents preliminary results that illustrate the potential benefits of the proposed approach. Finally, chapter 6 summarizes the key insights and discusses directions for future research.

2 Mathematical background

In section 2.1 we discuss the classification of the CG method as both an error projection and a Krylov subspace method. We then derive the convergence rate of CG in section 2.1.3, as well as the influence of the eigenvalues of the system matrix A on that rate in section 2.1.4. The latter is important, since it shows that the condition number $\kappa(A)$ is not the only factor influencing the convergence rate of CG. The first part ends with a brief review of the PCG method in section 2.1.5. Then, the second part of this chapter section 2.2 concerns the Schwarz methods, which are a class of domain decomposition methods. Schwarz methods can be used to construct preconditioners for use in PCG, even though these were originally devised as fixed point iteration methods for solving PDEs on complex domains. In section 2.2.2, we also see some conditioner number estimates that are classically used to estimate the convergence rate of PCG.

2.1. Conjugate gradient method

We seek to solve the linear system of equations $A\mathbf{u} = \mathbf{b}$, where A is a symmetric positive definite (SPD) matrix. These properties of A make the CG method particularly suitable for solving the system, as motivated in the chapter 1.

2.1.1. Projection methods

To further understand the CG method, we need to introduce the class of *projection methods*, which given some initial guess $\mathbf{u}_0$ find an approximation $\mathbf{u}_{\text{new}}$ to the solution of the linear system $A\mathbf{u} = \mathbf{b}$ in a *constrained subspace* $\mathcal{L} \subset \mathbb{R}^n$ using a *correction vector* $\mathbf{c}$ in another subspace $\mathcal{K} \subset \mathbb{R}^n$. That is, projection methods solve the following problem [17, Equation 5.3]

$$\text{Find } \mathbf{u}_{\text{new}} = \mathbf{u}_0 + \mathbf{c} \in \mathbf{u}_0 + \mathcal{K} \text{ such that } \mathbf{r}_{\text{new}} = A\mathbf{u}_{\text{new}} - \mathbf{b} = \mathbf{r}_0 - A\mathbf{c} \perp \mathcal{L}.$$

In doing so, projection methods perform a *projection step*, visualized in figure 2.1.

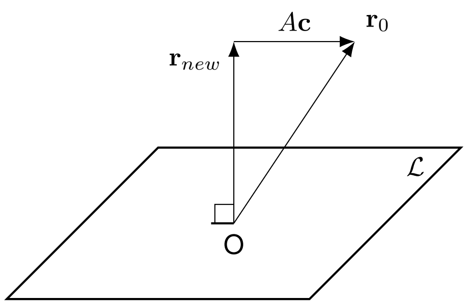

Figure 2.1: Visualization of projection method, based on [17, Figure 5.1]. The projection method finds the solution $\mathbf{u}_{\text{new}}$ in the affine subspace $\mathbf{u}_0 + \mathcal{K}$, such that the new residual $\mathbf{r}_{\text{new}}$ is orthogonal to the constraint subspace $\mathcal{L}$.

Error projection methods

The subclass of *error projection methods* defined by Definition 2.1 sets $\mathcal{K} = \mathcal{L}$.

Definition 2.1: Error projection method

To perform an error projection step, find $\mathbf{u}_{\text{new}} = \mathbf{u}_0 + \mathbf{c} \in \mathbf{u}_0 + \mathcal{K}$ such that

$$(\mathbf{r}_0 - A\mathbf{c}, w) = 0 \quad \forall w \in \mathcal{K}, \tag{2.1}$$

where $(\cdot,\cdot)$ is an inner product on $\mathcal{K}$. Let $V = [\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m]$ be a matrix whose columns span $\mathcal{K}$, then one error projection step is given by

$$\begin{aligned} \mathbf{u}_{\text{new}} &= \mathbf{u}_0 + V\mathbf{v} \\ V^T A V \mathbf{v} &= V^T \mathbf{r}_0, \end{aligned}$$

Error projection methods owe their name to the following central Theorem 2.1.

Theorem 2.1: Error minimization in A-norm

Given a linear system $A\mathbf{u} = \mathbf{b}$ with A SPD and solution $\mathbf{u}^*$. Define the errors $\mathbf{e}_0 = \mathbf{u}^* - \mathbf{u}_0$ and $\mathbf{e}_{\text{new}} = \mathbf{u}^* - \mathbf{u}_{\text{new}}$. Then, an error projection step minimizes the A-norm of the error in the affine subspace $\mathbf{u}_0 + \mathcal{K}$.

Proof. We have

$$\begin{aligned} A\mathbf{e}_{\text{new}} &= A(\mathbf{e}_0 - \mathbf{c}) \\ &= A(\mathbf{u}^* - \mathbf{u}_0 - \mathbf{c}) \\ &= \mathbf{r}_0 - A\mathbf{c} \end{aligned}$$

Hence, the orthogonality condition in equation (2.1) can be written as

$$(A\mathbf{e}_{\text{new}}, w) = (\mathbf{e}_0 - \mathbf{c}, w)_A = 0, \quad \forall w \in \mathcal{K}.$$

In other words, the correction vector $\mathbf{c}$ is the A-orthogonal projection of the error $\mathbf{e}_0$ onto $\mathcal{K}$. Therefore, there exists a projection operator $P_{\mathcal{K}}^A$ such that $\mathbf{c} = P_{\mathcal{K}}^A \mathbf{e}_0$ and we can write

$$\mathbf{e}_{\text{new}} = (I - P_{\mathcal{K}}^A)\mathbf{e}_0.$$

Moreover, we have using symmetry and positive definiteness of A that we can define the A-norm $\|\cdot\|_A$. Then, using the A-orthogonality between the error $\mathbf{e}_{\text{new}}$ and the correction $P_{\mathcal{K}}^A \mathbf{e}_0$, we get

$$\begin{aligned} \|\mathbf{e}_0\|_A^2 &= \|\mathbf{e}_{\text{new}}\|_A^2 + \|P_{\mathcal{K}}^A \mathbf{e}_0\|_A^2 \\ &\geq \|\mathbf{e}_{\text{new}}\|_A^2, \end{aligned}$$

which shows that the new error is smaller than the previous error in the A-norm. To show that the error projection step minimizes the A-norm of the error in the affine subspace $\mathbf{u}_0 + \mathcal{K}$, we let $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbf{u}_0 + \mathcal{K}$ be arbitrary, then using that $P_{\mathcal{K}}^A \mathbf{x} - \mathbf{y} \in \mathbf{u}_0 + \mathcal{K}$ we get

$$\begin{aligned} \|\mathbf{x} - \mathbf{y}\|_A^2 &= \|\mathbf{x} - P_{\mathcal{K}}^A \mathbf{x} + P_{\mathcal{K}}^A \mathbf{x} - \mathbf{y}\|_A^2 \\ &= \|\mathbf{x} - P_{\mathcal{K}}^A \mathbf{x}\|_A^2 + \|P_{\mathcal{K}}^A \mathbf{x} - \mathbf{y}\|_A^2 \\ &\geq \|\mathbf{x} - P_{\mathcal{K}}^A \mathbf{x}\|_A^2, \end{aligned}$$

which yields that [17, Theorem 1.38]

$$\min_{\mathbf{y} \in \mathbf{u}_0 + \mathcal{K}} \|\mathbf{x} - \mathbf{y}\|_A = \|\mathbf{x} - P_{\mathcal{K}}^A \mathbf{x}\|_A. \tag{2.2}$$

Now, substituting $\mathbf{x} = \mathbf{e}_0$ and $\mathbf{y} = \mathbf{c}$ into equation (2.2), we again find $\mathbf{c} = P_{\mathcal{K}}^A \mathbf{e}_0$, giving the desired result. □

General algorithm for error projection methods

By performing multiple (error) projection steps we obtain a sequence of approximations $\{\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_m\}$ to exact the solution $\mathbf{u}^*$ of the linear system $A\mathbf{u} = \mathbf{b}$. Theorem 2.1 ensures that each approximate solution $\mathbf{u}_j$ is closer to the exact solution $\mathbf{u}^*$ than the previous one $\mathbf{u}_{j-1}$. This idea forms the basis for a general error projection method and results in algorithm 1

Algorithm 1 Prototype error projection method [17, Algorithm 5.1]

```
Set $\mathbf{u} = \mathbf{u}_0$
while $\mathbf{u}$ is not converged do
    Choose basis $V$ of $\mathcal{K} = \mathcal{L}$
    $\mathbf{r} = \mathbf{b} - A\mathbf{u}$
    $\mathbf{c} = (V^T A V)^{-1} V^T \mathbf{r}$
    $\mathbf{u} = \mathbf{u} + V\mathbf{c}$
end while
```

Projection methods differ in their choice of the spaces $\mathcal{K}$ and $\mathcal{L}$ as well as in how they obtain the basis V of $\mathcal{K}$, as well as the so-called *Hessenberg matrix* defined in Definition 2.2.

Definition 2.2: Hessenberg matrix

The Hessenberg matrix H is defined as the matrix $H = V^T AV$, where V is a matrix whose columns span the subspace $\mathcal{K}$.

Krylov subspace methods

Krylov subspace methods form yet another subclass of projection methods and are defined by their choice of the space $\mathcal{K}$. Namely,

$$\mathcal{K}_m(A_0, \mathbf{r}_0) = \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, A^2\mathbf{r}_0, \ldots, A^{m-1}\mathbf{r}_0\}, \tag{2.3}$$

or $\mathcal{K}_m$ as a shorthand.

Definition 2.1. The grade of a vector $\mathbf{v}$ with respect to a matrix A is the lowest degree of the polynomial q such that $q(A)\mathbf{v} = 0$.

Consequently,

Theorem 2.1. The Krylov subspace $\mathcal{K}_m$ is of dimension m if and only if the grade μ of $\mathbf{v}$ with respect to A is not less than m [17, proposition 6.2],

$$\dim(\mathcal{K}_m) = m \iff \mu \geq m,$$

such that

$$\dim(\mathcal{K}_m) = \min\{m, \text{grade}(\mathbf{v})\}. \tag{2.4}$$

Definition 2.2. The action of a matrix A can be thought of as a mapping

$$\mathbb{R}^n \to \mathbb{R}^n : v \mapsto Av$$

Thus the domain and codomain of A are $\mathbb{R}^n$. Let $X \subset \mathbb{R}^n$, we can consider the map

$$X \to \mathbb{R}^n : v \mapsto Av$$

instead. The only difference from A is that the domain is X. This map is defined as the restriction $A_{|_X}$ of A to X.

Definition 2.3. Let Q be a projector onto the subspace X. Then the section of the operator A onto X is defined as $QA_{|_X}$.

Theorem 2.2. Let Q_m be any projector onto $\mathcal{K}_m$ and let A_m be the section of A to $\mathcal{K}_m$, that is, $A_m = Q_m A_{|\mathcal{K}_m}$. Then for any polynomial q of degree not exceeding $m-1$ [17, proposition 6.3],

$$q(A)v = q\left(A_m\right)v$$

and for any polynomial of degree $\leq m$,

$$Q_m q(A)v = q\left(A_m\right)v$$

2.1.2. CG algorithm

The CG method exists in the intersection of error projection methods and Krylov subspace methods, as it is a projection method with the choice $\mathcal{L} = \mathcal{K} = \mathcal{K}_m$. We can derive the CG method starting from Arnoldi's method, see algorithm A.1. Arnoldi's method is much like algorithm 1 in that it uses a Gramm-Schmidt orthogonalization procedure to obtain the basis V of the Krylov subspace $\mathcal{K}_m$ and a projection step to update the solution. Where Arnoldi's method is applicable to non-symmetric matrices by performing a full orthogonalization step, CG leverages the symmetry of A by doing a partial orthogonalization step. The latter is possible, since the CG method only needs to maintain the orthogonality of the residuals $\mathbf{r}_j$ with respect to the previous residuals $\mathbf{r}_{j-1}$ and not with respect to all previous residuals $\mathbf{r}_{j-2}, \ldots, \mathbf{r}_0$. The full derivation is discussed in the Appendix, section A and results in algorithm 2.

Algorithm 2 Conjugate Gradient Method

```
$\mathbf{r}_0 = \mathbf{b} - A\mathbf{u}_0$, $\mathbf{p}_0 = \mathbf{r}_0$, $\beta_0 = 0$
for $j = 0, 1, 2, \ldots, m$ do
    $\alpha_j = (\mathbf{r}_j, \mathbf{r}_j)/(A\mathbf{p}_j, \mathbf{p}_j)$
    $\mathbf{u}_{j+1} = \mathbf{u}_j + \alpha_j \mathbf{p}_j$
    $\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j A\mathbf{p}_j$
    $\beta_j = (\mathbf{r}_{j+1}, \mathbf{r}_{j+1})/(\mathbf{r}_j, \mathbf{r}_j)$
    $\mathbf{p}_{j+1} = \mathbf{r}_{j+1} + \beta_j \mathbf{p}_j$
end for
```

A crucial property of the approximate solution that algorithm 2 produces is given in Theorem 2.2

Theorem 2.2: CG approximate solution

The approximate solution at the m^{th} iteration is given by

$$\mathbf{u}_m = \mathbf{u}_0 + \sum_{i=0}^{m-1} c_i A^i \mathbf{r}_0 = \mathbf{u}_0 + q_{m-1}(A)\mathbf{r}_0, \tag{2.5}$$

where $q_{m-1}(A)$ is the solution polynomial of degree $m-1$ in A.

Proof. The CG method is a projection method with the choice $\mathcal{L} = \mathcal{K} = \mathcal{K}_m$. Hence, the approximate solution $\mathbf{u}_m$ is an element of the affine Krylov subspace $\mathbf{u}_0 + \mathcal{K}_m(A, \mathbf{r}_0)$, see Definition 2.1. The result follows from the fact that the Krylov subspace $\mathcal{K}_m(A, \mathbf{r}_0)$ is spanned by the vectors $\{\mathbf{r}_0, A\mathbf{r}_0, A^2\mathbf{r}_0, \ldots, A^{m-1}\mathbf{r}_0\}$ and that the approximate solution $\mathbf{u}_m$ is a linear combination of these vectors. The coefficients of this linear combination are given by the *CG solution coefficients* c_i. □

The Lanczos vectors are related through the Lanczos recurrence relation

$$\eta_{j+1}(A)\mathbf{v}_{j+1} = A\mathbf{v}_j - \delta_j \mathbf{v}_j - \eta_j \mathbf{v}_{j-1}. \tag{2.6}$$

The following relations exist between the entries of T_m and the CG coefficients α_j, β_j

$$\delta_{j+1} = \begin{cases} \dfrac{1}{\alpha_j} + \dfrac{\beta_{j-1}}{\alpha_{j-1}} & j > 0, \\ \dfrac{1}{\alpha_0} & j = 0, \end{cases} \tag{2.7}$$

and

$$\eta_{j+1} = \frac{\sqrt{\beta_{j-1}}}{\alpha_{j-1}}. \tag{2.8}$$

Here we have used the definition of T_m and the fact that the residuals are multiples of the Lanczos vectors $\mathbf{r}_j = \text{scalar} \times \mathbf{v}_j$ [17, Equation 6.103].

2.1.3. CG convergence

We derive a general bound for the error of the CG method in

Theorem 2.3: CG general error bound

Suppose we apply the CG method to the linear system $A\mathbf{u} = \mathbf{b}$ with A SPD and the exact solution $\mathbf{u}^*$. Then, the error of the m^{th} iterate $\mathbf{e}_m = \mathbf{u}^* - \mathbf{u}_m$ is bounded as

$$||\mathbf{e}_m||_A < \min_{r \in \mathcal{P}_{m-1}, r(0)=1} \max_{\lambda \in \sigma(A)} |r(\lambda)|||\mathbf{e}_0||_A. \tag{2.9}$$

Proof. Combining the results of Theorem 2.1 and Theorem 2.2 we get that the error of the m^{th} iterate of the CG algorithm $\mathbf{e}_m = \mathbf{u}^* - \mathbf{u}_m$ satisfies

$$\|\mathbf{e}_m\|_A = \|(I - Aq_{m-1}(A))\mathbf{e}_0\|_A = \min_{q \in \mathcal{P}_{m-1}} \|(I - Aq(A))\mathbf{e}_0\|_A = \min_{r \in \mathcal{P}_m, r(0)=1} \|r(A)\mathbf{e}_0\|_A, \tag{2.10}$$

The right-hand side can be further bounded by letting λ_i, ξ_i be the eigenvalues of A and the components of $\mathbf{e}_0$ in the eigenbasis of A, respectively. Then

$$||r(A)\mathbf{e}_0||_A = \sqrt{\sum_{i=1}^{n} |r(\lambda_i)|^2 |\xi_i|^2} \leq \max_{\lambda \in \sigma(A)} |r(\lambda)|||\mathbf{e}_0||_A,$$

which gives the desired result. □

Convergence criteria

We say that the CG method *converges* to a user-defined, absolute tolerance $\tilde{\epsilon}$ if the error of the m^{th} iterate $\mathbf{e}_m$ satisfies

$$||\mathbf{e}_m||_A \leq \tilde{\epsilon}.$$

Now, Theorem 2.3 allows us to define a criterion based on a *relative tolerance* ϵ, see Definition 2.3

Definition 2.3: Convergence criterion

The CG method is said to have *converged* to a user-defined, relative tolerance ϵ if

$$\frac{||\mathbf{e}_m||_A}{||\mathbf{e}_0||_A} \leq \epsilon,$$

which according to Theorem 2.3 is satisfied when

$$\min_{r \in \mathcal{P}_{m-1}, r(0)=1} \max_{\lambda \in \sigma(A)} |r(\lambda)| \leq \epsilon. \tag{2.11}$$

However, the criterion from Definition 2.3 is not usable a stopping criterion during CG iterations, since it involves the usually unknown error $\mathbf{e}$. Luckily, Theorem 2.4 shows how we can relate the ratio of A-norms of the error to a similar ratio of the 2-norms of the residuals.

Theorem 2.4: Residual convergence criterion

The CG method has converged to a user-defined, relative tolerance ϵ in the sense of Definition 2.3 if

$$\frac{\|\mathbf{r}_m\|_2}{\|\mathbf{r}_0\|_2} \leq \frac{\epsilon}{\sqrt{\kappa}}, \tag{2.12}$$

where $\kappa = \dfrac{\lambda_{\text{max}}}{\lambda_{\text{min}}}$ is the condition number of A.

Proof. We have for $i = 0, m$ that

$$\mathbf{e}_i = A^{-1}\mathbf{b} - \mathbf{u}_i = A^{-1}(\mathbf{b} - A\mathbf{u}_i) = A^{-1}\mathbf{r}_i.$$

Therefore, we can write

$$\frac{\|\mathbf{e}_m\|_A}{\|\mathbf{e}_0\|_A} = \frac{\mathbf{r}_0^T A^{-T}\mathbf{r}_m}{\mathbf{r}_0^T A^{-T}\mathbf{r}_0} = \frac{\|\mathbf{r}_m\|_{A^{-1}}}{\|\mathbf{r}_0\|_{A^{-1}}},$$

where the last equality follows as A^{-1} is SPD. Now, by Theorem C.1 and using that the eigenvalues of A^{-1} are the inverses of the eigenvalues of A, we can bound the A^{-1}-norm of the residuals as

$$\|\mathbf{r}_m\|_{A^{-1}} \leq \frac{1}{\sqrt{\lambda_{\text{min}}}}\|\mathbf{r}_m\|_2,$$

and

$$\|\mathbf{r}_0\|_{A^{-1}} \geq \frac{1}{\sqrt{\lambda_{\text{max}}}}\|\mathbf{r}_0\|_2.$$

Hence, we can write

$$\frac{\|\mathbf{e}_m\|_A}{\|\mathbf{e}_0\|_A} \leq \sqrt{\kappa}\frac{\|\mathbf{r}_m\|_2}{\|\mathbf{r}_0\|_2}.$$

To conclude, if we perform the CG method until the convergence criterion equation (2.12) is satisfied, we have

$$\frac{\|\mathbf{e}_m\|_A}{\|\mathbf{e}_0\|_A} \leq \sqrt{\kappa}\frac{\|\mathbf{r}_m\|_2}{\|\mathbf{r}_0\|_2} \leq \epsilon,$$

which gives the desired result. □

An important conclusion to draw from Theorem 2.4 is that if we set some relative tolerance for the residuals ϵ_r such that we stop iterating when

$$\frac{\|\mathbf{r}_m\|_2}{\|\mathbf{r}_0\|_2} \leq \epsilon_r, \tag{2.13}$$

then we get that the CG method has converged to a relative error tolerance ϵ given by

$$\frac{\|\mathbf{e}_m\|_A}{\|\mathbf{e}_0\|_A} \leq \epsilon = \frac{\epsilon_r}{\sqrt{\kappa}}.$$

This means that the CG method converges to a relative tolerance ϵ that is smaller than the relative tolerance of the residuals ϵ_r by a factor of $\sqrt{\kappa}$. On the one hand, this allows us to set a convergence criterion based on the residuals, which can actually be computed during CG iterations, as is the point of Theorem 2.4. On the other hand, the convergence criterion based on the residuals is also pessimistic by the same factor of $\sqrt{\kappa}$. In other words, the CG method performs more iterations to converge to a stricter tolerance than the user-defined tolerance ϵ_r.

Moreover, using Theorem C.1 we can also bound the absolute error tolerance $\tilde{\epsilon}$ of the CG method in terms of the initial residual. That is, suppose we set ϵ_r as a convergence criterion, then we get

$$\tilde{\epsilon} \leq \frac{\epsilon_r}{\sqrt{\kappa}}\|\mathbf{e}_0\|_A = \frac{\epsilon_r}{\sqrt{\kappa}}\|\mathbf{r}_0\|_{A^{-1}} \leq \frac{\epsilon_r}{\sqrt{\lambda_{\text{max}}}}\|\mathbf{r}_0\|_2. \tag{2.14}$$

From Theorem 2.4 and choosing $\mathbf{u}_0 = \mathbf{0} \in \mathbb{R}^n$, we can derive a commonly implemented convergence criterion

$$\frac{\|\mathbf{r}_m\|_2}{\|\mathbf{b}\|_2} \leq \epsilon_b. \tag{2.15}$$

In this case, the relative error tolerance ϵ achieved by the CG method in the sense of Definition 2.3 still depends on the initial guess. Suppose, we choose a 'good' initial guess such that $\|\mathbf{r}_0\|_2 \leq \|\mathbf{b}\|_2$, then

$$\frac{\|\mathbf{r}_m\|_2}{\|\mathbf{b}\|_2} \leq \frac{\|\mathbf{r}_m\|_2}{\|\mathbf{r}_0\|_2},$$

which means equation (2.15) is more optimistic than equation (2.13), requiring less CG iterations before it is satisfied. However, if we choose a bad initial guess such that $\|\mathbf{r}_0\|_2 \geq \|\mathbf{b}\|_2$, then equation (2.15) is more pessimistic than equation (2.13) and will require more CG iterations before it is satisfied. For a good (bad) initial guess we therefore achieve a relative error tolerance larger (smaller) than $\frac{\epsilon_b}{\sqrt{\kappa}}$. In regard to the absolute error tolerance $\tilde{\epsilon}$, we can write using Theorem C.1 that

$$\textbf{good initial guess } \tilde{\epsilon} \geq \frac{\epsilon_b}{\sqrt{\kappa}}\|\mathbf{e}_0\|_A = \frac{\epsilon_b}{\sqrt{\kappa}}\|\mathbf{r}_0\|_{A^{-1}} \geq \frac{\sqrt{\lambda_{\text{min}}}}{\lambda_{\text{max}}}\epsilon_b\|\mathbf{r}_0\|_2 \geq \frac{\sqrt{\lambda_{\text{min}}}}{\lambda_{\text{max}}}\epsilon_b\|\mathbf{b}\|_2,$$

$$\textbf{bad initial guess } \tilde{\epsilon} \leq \frac{\epsilon_b}{\sqrt{\kappa}}\|\mathbf{e}_0\|_A = \frac{\epsilon_b}{\sqrt{\kappa}}\|\mathbf{r}_0\|_{A^{-1}} \leq \frac{\lambda_{\text{min}}}{\sqrt{\lambda_{\text{max}}}}\epsilon_b\|\mathbf{r}_0\|_2 \leq \frac{\lambda_{\text{min}}}{\sqrt{\lambda_{\text{max}}}}\epsilon_b\|\mathbf{b}\|_2.$$

In other words, when we use equation (2.15) as a convergence criterion, the absolute error tolerance we achieve satisfies

$$\frac{\sqrt{\lambda_{\text{min}}}}{\lambda_{\text{max}}}\epsilon_b\|\mathbf{b}\|_2 \leq \tilde{\epsilon} \leq \frac{\lambda_{\text{min}}}{\sqrt{\lambda_{\text{max}}}}\epsilon_b\|\mathbf{b}\|_2. \tag{2.16}$$

Comparing equation (2.14) and equation (2.16) we gain the insight that when

$$\|\mathbf{r}_0\|_2 < \frac{\epsilon_b}{\epsilon_r\sqrt{\kappa}}\|\mathbf{b}\|_2,$$

using convergence criterion equation (2.12) is guaranteed to give a more accurate result than equation (2.15). On the other hand, if $\|\mathbf{r}_0\|_2$ is larger, then it might be better to use equation (2.15) as a convergence criterion. In practice, criterion equation (2.15) is used, since its accuracy bound in equation (2.16) holds, independent of initial guesses.

Convergence rate

Next to convergence criteria based on residuals, we can also try to find a solution to the minimization problem in equation (2.12), which gives us an expected convergence rate. Under the assumption of a uniform distribution of the eigenvalues of A, we can further bound the error of the m^{th} iterate of the CG algorithm by a Chebyshev polynomial. This is done in Theorem 2.5 and is a direct consequence of Theorem B.1.

Theorem 2.5: Convergence rate of CG

Let the linear system $A\mathbf{u} = \mathbf{b}$ be as in Theorem 2.3 and let the eigenvalues of A be uniformly distributed in the interval $[\lambda_{\text{min}}, \lambda_{\text{max}}]$. Then the error of the m^{th} iterate of the CG algorithm satisfies

$$\|\mathbf{e}_m\|_A \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^m \|\mathbf{e}_0\|_A, \tag{2.17}$$

Proof. We use the general expression for CG's error from Theorem 2.3 in combination with the uniform distribution of eigenvalues in $[\lambda_{\text{min}}, \lambda_{\text{max}}]$ to write the error of the m^{th} iterate of the CG algorithm as

$$\|\mathbf{e}_m\|_A \leq \min_{r\in\mathcal{P}_{m-1}, r(0)=1}\ \max_{\lambda\in[\lambda_{\text{min}},\lambda_{\text{max}}]} |r(\lambda)|\|\mathbf{e}_0\|_A, \tag{2.18}$$

which by Theorem B.1 is solved by the real-valued scaled Chebyshev polynomial $\hat{C}_m$ from Definition B.2 with $[a, b] = [\lambda_{\text{min}}, \lambda_{\text{max}}]$ and $\gamma = 0$. We obtain

$$\|\mathbf{e}_m\|_A \leq \frac{1}{d_m(0)}\|\mathbf{e}_0\|_A = \frac{1}{C_m(\frac{\kappa+1}{\kappa-1})}\|\mathbf{e}_0\|_A, \tag{2.19}$$

where $\kappa = \dfrac{\lambda_{\text{max}}}{\lambda_{\text{min}}}$ is the condition number of A. Using the approximation from equation (B.2) and setting $\tilde{z} = \dfrac{\kappa+1}{\kappa-1}$ we can write

$$\begin{aligned}
\frac{1}{d_m(0)} &= \frac{1}{C_m(\tilde{z})} \\
&\leq \frac{2}{\left(\tilde{z} + \sqrt{\tilde{z}^2 - 1}\right)^m} \\
&= 2\left(\tilde{z} - \sqrt{\tilde{z}^2 - 1}\right)^m \\
&= 2\left(\frac{\kappa + 1 - 2\sqrt{\kappa}}{\kappa - 1}\right)^m \\
&= 2\left(\frac{(\sqrt{\kappa} - 1)^2}{(\sqrt{\kappa} - 1)(\sqrt{\kappa} + 1)}\right)^m \\
&= 2\left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^m.
\end{aligned}$$

Substituting this into equation (2.19) gives us the desired result. □

From Theorem 2.5 and Definition 2.3 we get that the CG method converges to a user-defined, relative tolerance ϵ if

$$\left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^m \leq \frac{\epsilon}{2},$$

such that number of iterations m is given by

$$m \leq \left\lceil \log_f \left(\frac{\epsilon}{2}\right) \right\rceil, \tag{2.20}$$

where $f = \dfrac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}$. This could in principle be used as a stopping criterion, but it is not practical, since we generally do not know the condition number κ of A a priori.

At this point, it is necessary to note that even though the Chebyshev polynomial is optimal for the conditions of Theorem 2.5, it is not guaranteed that the eigenvalues of A are uniformly distributed. In fact, the eigenvalues of A are often clustered around some value. This means that the Chebyshev polynomial is not optimal in this case, and we can actually expect a better convergence rate than equation (2.17), as we will see in section 2.1.4. Even though the bounds from equations (2.17) and (2.20) are not sharp for non-uniform distributions, they are still correct as upper bounds.

For general distributions of eigenvalues we can still derive a useful property of the CG method. Instead of the Chebyshev polynomial $\hat{C}_m$ we pose a test polynomial r_{test} of degree m that satisfies the constraints of the minimization problem in equation (5.3). This test polynomial is given by

$$r_{\text{test}}(t) = \prod_{i=1}^{m} \frac{\lambda_i - t}{\lambda_i}.$$

Indeed, note that $r_{\text{test}} \in \mathcal{P}_m$, $r_{\text{test}}(0) = 1$ and $r_{\text{test}}(\lambda_i) = 0$ for $i = 1, 2, \ldots, m$. We obtain for $m = n = |\mathcal{N}|$ that

$$||\mathbf{e}_n||_A = ||\mathbf{e}_0||_A \max_{\lambda \in \sigma(A)} |r_{\text{test}}(\lambda)| = 0,$$

which implies that CG solves the linear system in n iterations. Furthermore, if there are only k distinct eigenvalues, then the CG iteration terminates in at most k iterations.

The strategy of posing a test polynomial r_{test} that satisfies the constraints of the minimization problem to come up with a bound for the error of the CG method is not limited to the Chebyshev polynomial from Theorem 2.5 or the test polynomial r_{test}. In fact, we can use any polynomial r of degree m that satisfies the constraints of the minimization problem. Evaluating the maximum value of this polynomial on the eigenspectrum of A would then result in an error bound. The problem is of course that we want to find a polynomial that gives a *sharp* error bound. In section 3.3 we discuss some recent literature using this strategy to achieve sharper bounds than equation (2.17) and in chapter 5 we apply the same strategy to find a bound for the error of the CG method for the case of clustered eigenvalues.

2.1.4. Influence of eigenvalue distribution on CG convergence

In the derivation of the convergence rate of the CG algorithm, we used the Chebyshev polynomial to bound the error. However, we can find an expression of the error provided the eigendecomposition of A is available. Supposing A is full rank and by its symmetry we can write its diagonalization as $A = VDV^T$, where V is the orthonormal eigenvector matrix and D is the diagonal eigenvalue matrix. Then $r(A) = I - Aq(A) = V(I - Dq(D))V^T = Vr(D)V^T$. Also note that $\mathbf{e}_0 = x^* - \mathbf{u}_0 = A^{-1}b - \mathbf{u}_0 = A^{-1}\mathbf{r}_0$. As seen in equation (2.10), the error of the m^{th} iterate of the CG algorithm is given by

$$||\mathbf{e}_m||_A^2 = ||r_m(A)\mathbf{e}_0||_A^2,$$

and

$$\begin{aligned}||r_m(A)\mathbf{e}_0||_A^2 &= \mathbf{e}_0^T r_m(A)^T A r_m(A)\mathbf{e}_0\\ &= \mathbf{e}_0^T V r_m(D) V^T V D V^T V r_m(D) V^T \mathbf{e}_0\\ &= (V^T\mathbf{e}_0)^T r_m(D) D r_m(D) V^T \mathbf{e}_0.\end{aligned}$$

We also have

$$\begin{aligned}V^T\mathbf{e}_0 &= V^T A^{-1}\mathbf{r}_0\\ &= V^T V D^{-1} V^T \mathbf{r}_0\\ &= D^{-1}\rho_0,\end{aligned}$$

where $\rho_0 = V^T\mathbf{r}_0$ is the initial residual in the eigenvector basis of A. Therefore,

$$\begin{aligned}||r_m(A)\mathbf{e}_0||_A^2 &= \rho_0^T D^{-1} r_m(D) D r_m(D) D^{-1}\rho_0\\ &= \rho_0^T r_m(D) D^{-1} r_m(D)\rho_0\\ &= \sum_{i=1}^{n} \frac{r_m(\lambda_i)^2}{\lambda_i}\rho_{0,i}^2,\end{aligned}$$

which gives

$$||\mathbf{e}_m||_A^2 = \sum_{i=1}^{n} \frac{r_m(\lambda_i)^2}{\lambda_i}\rho_{0,i}^2. \tag{2.21}$$

To obtain the residual polynomial r_m, we can use the recurrence relation between the Lanczos vectors and expressions for the Hessenberg matrix coefficients in equations (2.7) and (2.8). In particular,

$$\begin{aligned}\frac{1}{\eta_{j+1}}\mathbf{v}_{j+1} &= A\mathbf{v}_j - \delta_j\mathbf{v}_j - \eta_j\mathbf{v}_{j-1}\\ &= p_{j+1}(A)\mathbf{v}_1,\end{aligned}$$

where we define $p_{-1}(A) = 0, p_0(A) = I$. This gives

$$\begin{aligned}\eta_{j+1}p_{j+1}(A)\mathbf{v}_1 &= A\mathbf{v}_j - \delta_j\mathbf{v}_j - \eta_j\mathbf{v}_{j-1},\\ &= \left(Ap_j(A) - \delta_j p_j(A) - \eta_j p_{j-1}(A)\right)\mathbf{v}_1,\end{aligned}$$

and therefore

$$p_{j+1}(A) = \frac{1}{\eta_{j+1}} \left((A - \delta_j) p_j(A) - \eta_j p_j(A) \right). \tag{2.22}$$

Furthermore, we have the following relation between the residual polynomial and the Lanczos polynomial [16, Section 3.2]

$$r_j(A) = (I - A q_{j-1}(A)) \mathbf{r}_0 = \frac{p_j(A)}{p_j(0)} \mathbf{r}_0. \tag{2.23}$$

This gives a way of calculating the residual polynomial r_m and thereby the error of the m^{th} iterate of the CG algorithm.

Additionally, the coefficients c_i of the solution polynomial q_m in equation (2.5) can be calculated. First we introduce a function that extracts the coefficients of a polynomial p

Definition 2.4. Let $p(t) = \sum_{i=0}^{n} c_i t^i$ be a polynomial of degree n. Then, the function coeff$(p; i)$ extracts the i^{th} coefficient of p such that coeff$(p; i) = c_i$.

Now using equation (2.23), we can write the solution polynomial as

$$A q_{m-1}(A) = I - r_m(A)$$
$$r_m(\mathbf{0}) = I \implies A \sum_{i=1}^{m-1} c_{i-1} A^i = - \sum_{i=1}^{m} \text{coeff}(r_m; i) A^i,$$

which implies

$$c_i = -\text{coeff}(r_m; i+1), \quad i = 0, 1, \ldots, m-1. \tag{2.24}$$

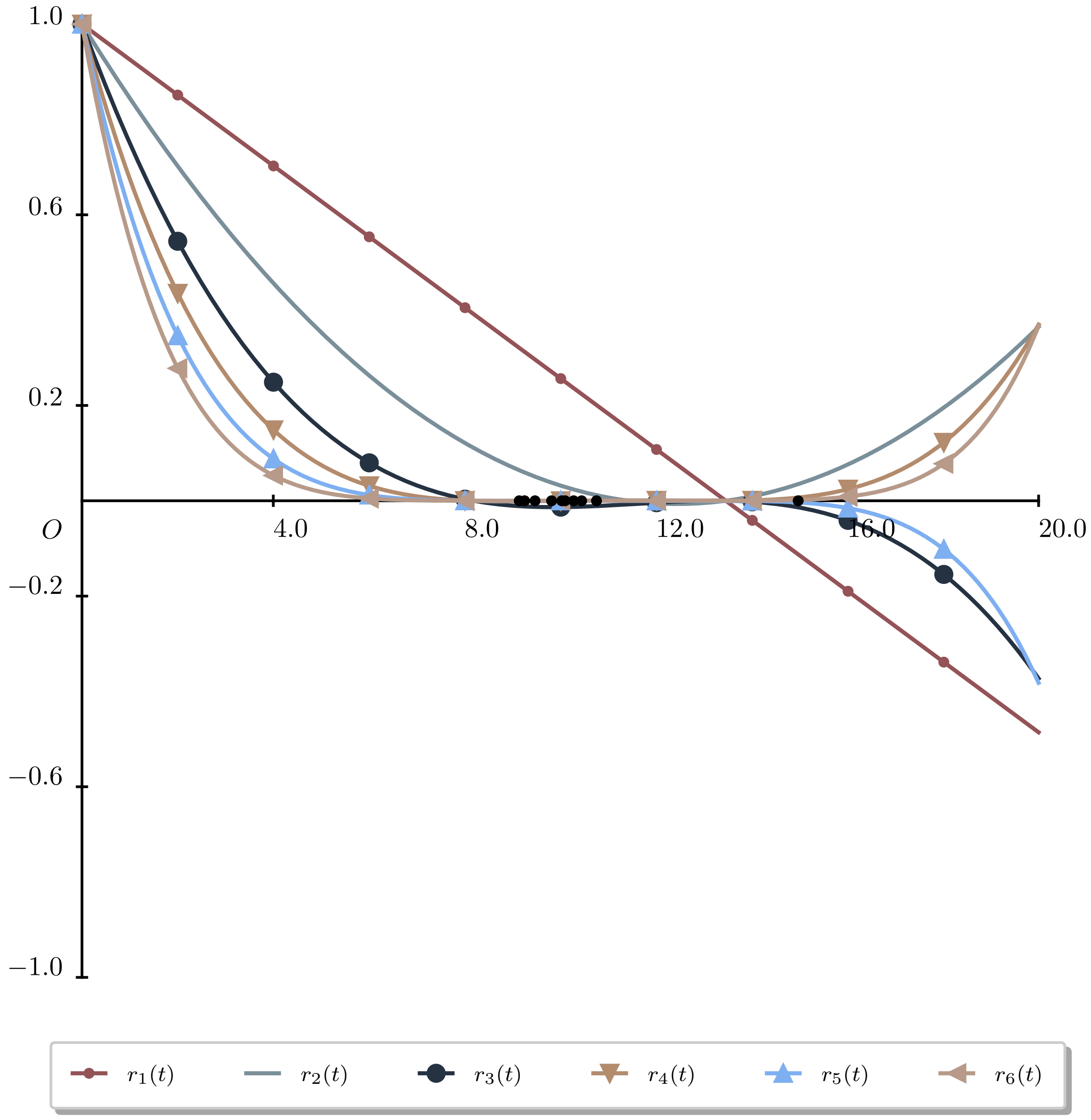

Figure 2.2: Residual polynomials resulting from successive CG iterations

The behavior of the residual polynomials is crucial for understanding the convergence properties of the CG method. In particular, the distribution of the eigenvalues of A significantly affects the convergence rate, as illustrated in figure 2.3. For all plots the lowest and highest eigenvalue in figure 2.3 are $\lambda_{\text{min}} = 0.1$, $\lambda_{\text{max}} = 0.9$ such that $f = \dfrac{\sqrt{\frac{\lambda_{\text{min}}}{\lambda_{\text{max}}}} - 1}{\sqrt{\frac{\lambda_{\text{min}}}{\lambda_{\text{max}}}} + 1}$ and the ratio $\dfrac{\|\mathbf{e}_m\|_A}{\|\mathbf{e}_0\|_A}$ is set to $\dfrac{10^{-6}}{\|\mathbf{u}_{\text{test}} - \mathbf{u}_0\|}$. The system size $N = 360$ is kept small and the system matrix A is diagonal so that it is numerically trivial to determine the exact solution $\mathbf{u}_{\text{test}}$. This results in an overall iteration bound

$$m_{\text{classical}} = \left\lceil \log_f \left(\frac{10^{-6}}{2\|\mathbf{u}_{\text{test}} - \mathbf{u}_0\|} \right) \right\rceil = 26$$

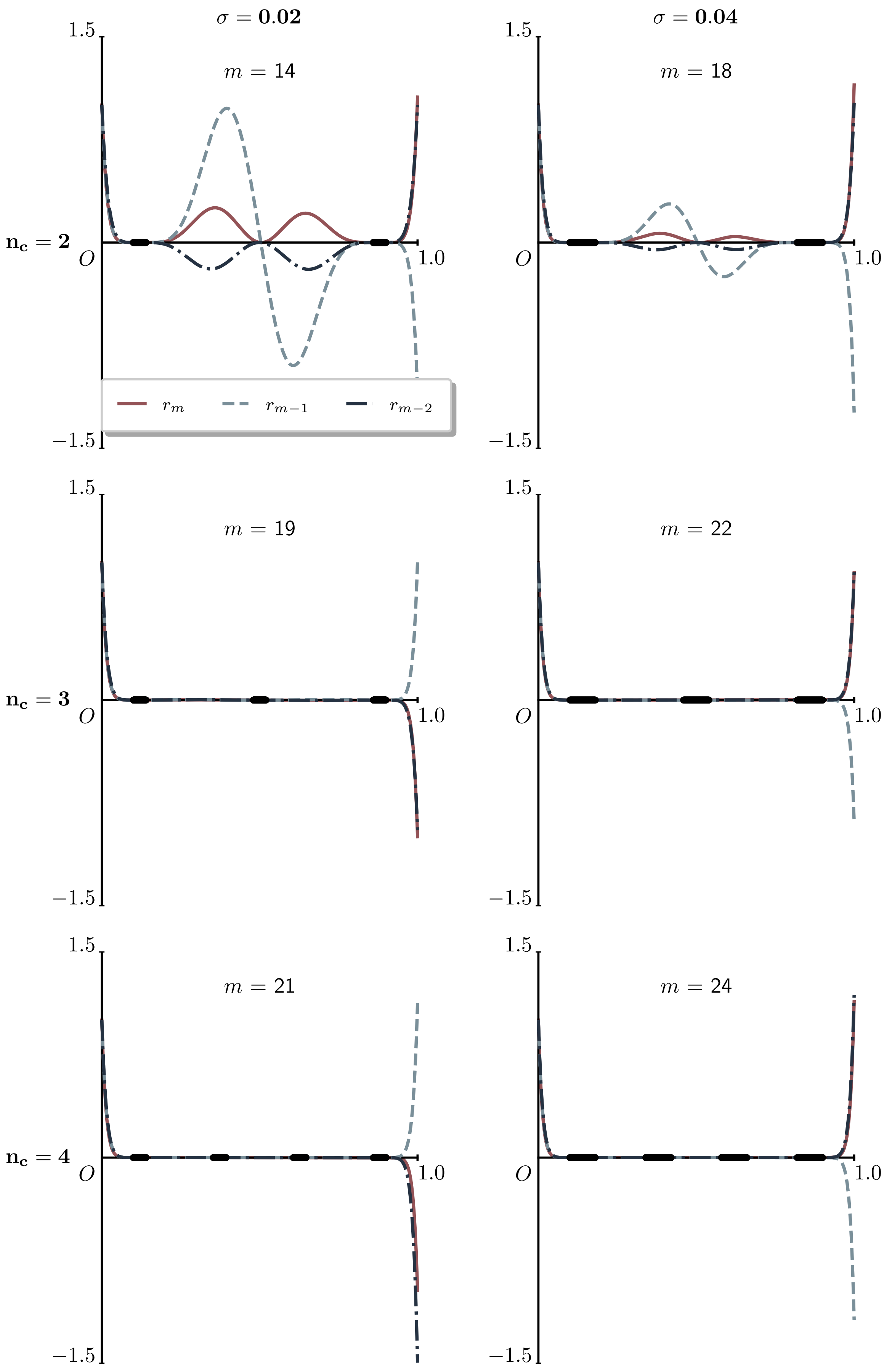

Figure 2.3: Plots of the last three CG residual polynomials for different eigenvalue distributions. n_c indicates the number of clusters and σ is the width of the cluster. The size of the system N and the condition number $\kappa(A)$ are kept constant. m indicates the number of iterations required for convergence.

Hence, the number of iterations required for convergence depends on the specific clustering of the eigenvalues, as pointed out for example in Kelley, Section 2.3.

Based behavior exhibited in figure 2.3 and from theorem 2.1, we can reason what the best and worst possible spectra for CG convergence are. That is, the best possible spectrum is one where eigenvalues are tightly clustered around distinct values, while the worst possible spectrum is one where the eigenvalues are evenly distributed across the whole range of the spectrum. This is illustrated in figure 2.4.

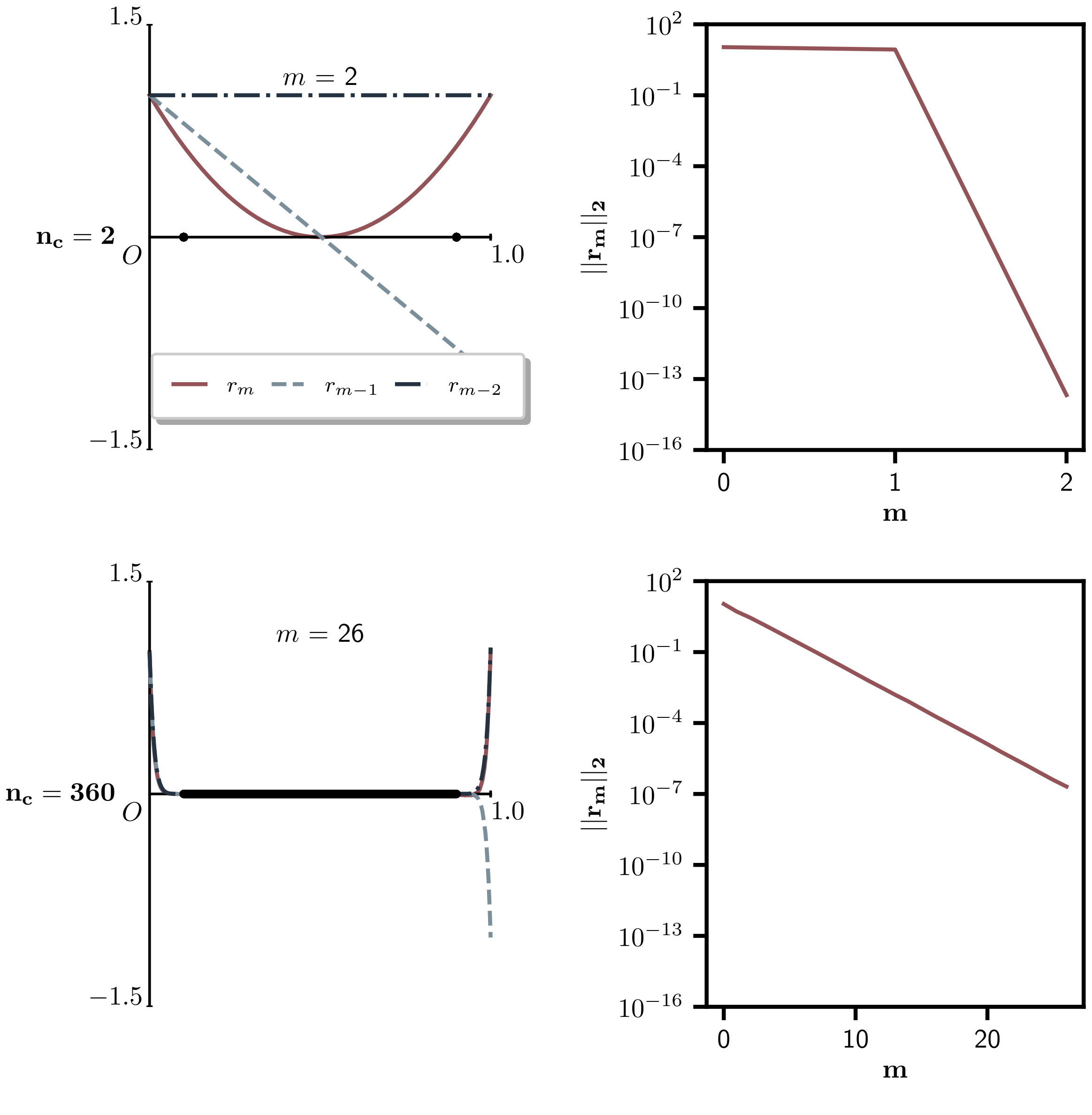

Figure 2.4: Best and worst possible spectra for CG convergence. The top row shows the best possible spectrum, where the eigenvalues are tightly clustered on two distinct values. The bottom row shows the worst possible spectrum, where the eigenvalues are evenly distributed across the whole range of the spectrum. The left column shows the eigenvalue distribution with the residual polynomials corresponding to the iteration. The right column shows the norm of the residual versus the iteration number. Convergence is achieved at a tolerance of 10^{-6}.

The first row in figure 2.4 shows an instance of the super-linear convergence that CG can exhibit, particularly when the eigenvalues are closely clustered. This is characteristic of CG is further touched upon in chapters 3 and 5.

2.1.5. Preconditioned CG

Suppose M is some SPD preconditioner, then variants of CG can be derived by applying M to the system of equations. The three main approaches are

PCG-1 left

$$M^{-1}A\mathbf{u} = M^{-1}\mathbf{b}$$

PCG-2 right

$$\begin{aligned} AM^{-1}\mathbf{x} &= M^{-1}\mathbf{b} \\ \mathbf{u} &= M^{-1}\mathbf{x}; \end{aligned}$$

PCG-3 symmetric or split

$$\begin{aligned} M &= LL^T \\ L^{-1}AL^{-T}\mathbf{x} &= L^{-1}\mathbf{b} \\ \mathbf{u} &= L^{-T}\mathbf{x}. \end{aligned}$$

Furthermore, all these variants are mathematically equivalent. Indeed, for the cases **PCG-type 1** and **PCG-type 2**, we can rewrite the CG algorithm using the $M-$ or $M^{-1}-$inner products, respectively. In either case the iterates are the same. For instance for the left preconditioned CG, we define $\mathbf{z}_j = M^{-1}\mathbf{r}_j$. Note that $M^{-1}A$ is self-adjoint with respect to the $M-$inner product, that is for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ we have

$$(M^{-1}A\mathbf{x}, \mathbf{y})_M = (A\mathbf{x}, \mathbf{y}) = (\mathbf{x}, A\mathbf{y}) = (\mathbf{x}, M^{-1}A\mathbf{y})_M.$$

We use this to get a new expression for α_j. To that end, we write

$$\begin{aligned} 0 &= (\mathbf{r}_{j+1}, \mathbf{r}_j)_M \\ &= (\mathbf{z}_{j+1}, \mathbf{r}_j) \\ &= (\mathbf{z}_j - \alpha_j M^{-1}A\mathbf{p}_j, M^{-1}\mathbf{r}_j)_M \\ &= (\mathbf{z}_j, M^{-1}\mathbf{r}_j)_M - \alpha_j (M^{-1}A\mathbf{p}_j, M^{-1}\mathbf{r}_j)_M \\ &= (\mathbf{z}_j, \mathbf{z}_j)_M - \alpha_j (M^{-1}A\mathbf{p}_j, \mathbf{z}_j)_M \end{aligned}$$

and therefore

$$\alpha_j = \frac{(\mathbf{z}_j, \mathbf{z}_j)_M}{(M^{-1}A\mathbf{p}_j, \mathbf{z}_j)_M}.$$

Using $\mathbf{p}_{j+1} = \mathbf{z}_{j+1} + \beta_j \mathbf{p}_j$ and A-orthogonality of the search directions $\mathbf{p}_j$ with respect to $M-$norm $(A\mathbf{p}_j, \mathbf{p}_k)_M = 0$, we can write

$$\alpha_j = \frac{(\mathbf{z}_j, \mathbf{z}_j)_M}{(M^{-1}A\mathbf{p}_j, \mathbf{p}_j)_M}.$$

Similarly, we can derive the equivalent expression for β_j as

$$\beta_j = \frac{(\mathbf{z}_{j+1}, \mathbf{z}_{j+1})_M}{(\mathbf{z}_j, \mathbf{z}_j)_M}.$$

This gives the left preconditioned CG algorithm in 3.

Algorithm 3 Left preconditioned CG [17, Algorithm 9.1]

```
$\mathbf{r}_0 = b - A\mathbf{u}_0$, $z_0 = M^{-1}\mathbf{r}_0$, $p_0 = z_0$, $\beta_0 = 0$
for $j = 0, 1, 2, \ldots, m$ do
    $\alpha_j = (\mathbf{z}_j, \mathbf{z}_j)_M / (M^{-1}A\mathbf{p}_j, \mathbf{p}_j)_M = (\mathbf{r}_j, \mathbf{z}_j)/(A\mathbf{p}_j, \mathbf{p}_j)$
    $\mathbf{u}_{j+1} = \mathbf{u}_j + \alpha_j \mathbf{p}_j$
    $\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j A\mathbf{p}_j$
    $\mathbf{z}_{j+1} = M^{-1}\mathbf{r}_{j+1}$
    $\beta_j = \frac{(\mathbf{z}_{j+1}, \mathbf{z}_{j+1})_M}{(\mathbf{z}_j, \mathbf{z}_j)_M} = \frac{(\mathbf{r}_{j+1}, \mathbf{z}_{j+1})}{(\mathbf{r}_j, \mathbf{z}_j)}$
    $\mathbf{p}_{j+1} = \mathbf{z}_{j+1} + \beta_j \mathbf{p}_j$
end for
```

Furthermore it can be shown that the iterates of CG applied to the system with **PCG-type 3** results in identical iterates [17, Algorithm 9.2].

2.2. Schwarz methods

The content of this section is largely based on chapters 1, 2, 4 and 5 of Dolean, Jolivet, and Nataf about Schwarz methods.

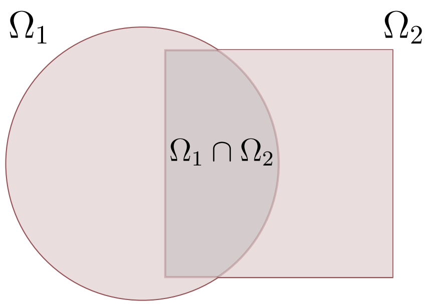

Figure 2.5: Keyhole domain Ω with two overlapping subdomains Ω_1 and Ω_2. The boundary of the keyhole domain is denoted by $\partial\Omega$ and the boundaries of the subdomains are denoted by $\partial\Omega_1$ and $\partial\Omega_2$. The overlapping region is denoted by $\Omega_1 \cap \Omega_2$.

The original Schwarz method was a way of proving that a Poisson problem on some complex domain Ω as in figure 2.5 has a solution.

$$\begin{aligned} -\Delta u &= f \text{ in } \Omega, \\ u &= 0 \text{ on } \partial\Omega. \end{aligned} \tag{2.25}$$

Existence of a solution is proved by splitting up the original complex domain in two (or more) simpler, possibly overlapping subdomains and solving the Poisson problem on each of these subdomains. The solution on the original domain is then the sum of the solutions on the subdomains. The method is named after Hermann Schwarz, who first introduced the method in 1869 [18]. The method has since been extended to more general problems and is now a popular method for solving partial differential equations. Let the alternating Schwarz method be characterised as in definition 2.5.

Definition 2.5. The Alternating Schwarz algorithm is an iterative method based on alternately solving subproblems in domains Ω_1 and Ω_2. It updates $(u_1^n, u_2^n) \to \left(u_1^{n+1}, u_2^{n+1}\right)$ by

$$\begin{aligned} -\Delta\left(u_1^{n+1}\right) &= f && \text{in } \Omega_1, & & -\Delta\left(u_2^{n+1}\right) = f && \text{in } \Omega_2, \\ u_1^{n+1} &= 0 && \text{on } \partial\Omega_1 \cap \partial\Omega, & \text{then} \quad & u_2^{n+1} = 0 && \text{on } \partial\Omega_2 \cap \partial\Omega, \\ u_1^{n+1} &= u_2^n && \text{on } \partial\Omega_1 \cap \overline{\Omega_2}; & & u_2^{n+1} = u_1^{n+1} && \text{on } \partial\Omega_2 \cap \overline{\Omega_1}. \end{aligned}$$

The original Schwarz algorithm is sequential and, thereby, does not allow for parallelization. However, the algorithm can be parallelized. The Jacobi Schwarz method is a generalization of the original Schwarz method, where the subproblems are solved simultaneously and subsequently combined into a global solution. In order to combine local solutions into one global solution, an extension operator E_i, $i = 1, 2$ is used. It is defined as

$$E_i(v) = v \text{ in } \Omega_i, \quad E_i(v) = 0 \text{ in } \Omega \backslash \Omega_i.$$

Instead of solving for local solutions directly, one can also solve for local corrections stemming from a global residual. This is the additive Schwarz method (ASM). It is defined in algorithm 4.

Algorithm 4 Additive Schwarz method [8, Algorithm 1.2]

```
Compute residual $r^n = f - \Delta u^n$.
For $i = 1, 2$ solve for a local correction $v_i^n$:
    $-\Delta v_i^n = r^n \text{ in } \Omega_i, \quad v_i^n = 0 \text{ on } \partial\Omega_i$
Update the solution: $u^{n+1} = u^n + \sum_{i=1}^{2} E_i(v_i)^n$.
```

The restricted additive Schwarz method (RAS) is similar to ASM, but differs in the way local corrections are combined to form a global one. In the overlapping region of the domains it employs a weighted average of the local corrections. In particular, a partition of unity χ_i is used. It is defined as

$$\chi_i(x) = \begin{cases} 1, & x \in \Omega_i \setminus \Omega_{3-i}, \\ 0, & x \in \delta\Omega_i \setminus \delta\Omega \\ \alpha, & 0 \leq \alpha \leq 1, x \in \Omega_i \cap \Omega_{3-i} \end{cases}$$

such that for any function $w : \Omega \to \mathbb{R}$, it holds that

$$w = \sum_{i=1}^{2} E_i(\chi_i w_{\Omega_i}).$$

The RAS algorithm is defined in algorithm 5.

Algorithm 5 Restrictive additive Schwarz method [8, Algorithm 1.1]

```
Compute residual $r^n = f - \Delta u^n$.
For $i = 1, 2$ solve for a local correction $v_i^n$:
    $-\Delta v_i^n = r^n \text{ in } \Omega_i, \quad v_i^n = 0 \text{ on } \partial\Omega_i$
Update the solution: $u^{n+1} = u^n + \sum_{i=1}^{2} E_i(\chi_i v_i^n)$.
```

2.2.1. Schwarz methods as preconditioners

Let $\mathcal{N}$ be set containing all indices of degrees of freedom in the domain Ω and N_{sub} be the number of subdomains such that

$$\mathcal{N} = \sum_{i=1}^{N_{\text{sub}}} \mathcal{N}_i,$$

$\mathcal{N}_i$ is the set of indices of degrees of freedom in the subdomain Ω_i.

Furthermore, let $R_i \in \mathcal{R}^{|\mathcal{N}_i| \times |\mathcal{N}|}$, R_i^T and D_i be the discrete versions of the restriction, extension and partition of unity operators such that

$$\mathcal{R}^{|\mathcal{N}|} \ni U = \sum_{i=1}^{\mathcal{N}_{\text{sub}}} R_i^T D_i R_i U.$$

Note that D_i is a diagonal matrix where the entries are the values of the partition of unity function χ_i evaluated for each degree of freedom. Consider for instance, a multidimensional FEM problem, in which $\mathcal{T}$ is the triangulation of the domain Ω and $\mathcal{T}_i$ is the triangulation of the subdomain Ω_i such that [8, Equation 1.27]

$$\Omega_i = \cup_{\tau \in \mathcal{T}_i} \tau.$$

In this case [8, Equation 1.28]

$$\mathcal{N}_i = \{k \in \mathcal{N} | \text{meas}(\text{supp}(\phi_k) \cap \Omega_i) > 0\},$$

and we can define

$$\mu_k = |\{j | 1 \leq j \leq N_{\text{sub}} \text{ and } k \in \mathcal{N}_j\}|.$$

Finally, this leads to

$$(D_i)_{kk} = \frac{1}{\mu_k}, \; k \in \mathcal{N}_i. \tag{2.26}$$

Although the original Schwarz method is not a preconditioner, the ASM and RAS methods can be used as such. Originally the Schwarz method is a fixed point iteration [8, Definitions 1.12 and 1.13]

$$u^{n+1} = u^n + M^{-1} r^n, \; r^n = f - Au^n,$$

where M equals, but is not limited to, one of the following matrices;

$$M^{-1}_{\text{ASM}} = \sum_{i=1}^{N_{\text{sub}}} R_i^T (R_i A R_i^T)^{-1} R_i, \tag{2.27a}$$

$$M^{-1}_{\text{RAS}} = \sum_{i=1}^{N_{\text{sub}}} R_i^T D_i (R_i A R_i^T)^{-1} R_i. \tag{2.27b}$$

Both M^{-1}_{ASM} and M^{-1}_{RAS} are SPD and can be used as preconditioners for CG.

Optimized Schwarz methods and corresponding preconditioners can also be constructed by including more interface conditions (Robin or Neumann) in the subproblems. One such example is the Optimized Restrictive Additive Schwarz method (ORAS) discussed in [8, Chapter 2].

2.2.2. Convergence of the original Schwarz method

In this section the Schwarz problem stated in definition 2.5 is solved in the one- and two-dimensional case. The convergence of the original Schwarz method is then discussed.

1D case

Let $L > 0$ and the domain $\Omega = (0, L)$. The domain is split into two subdomains $\Omega_1 = (0, L_1)$ and $\Omega_2 = (l_2, L)$ such that $l_2 \leq L_1$. Instead of solving for $u_{1,2}$ directly, we solve for the error $e^n_{1,2} = u^n_{1,2} - u_{|\Omega_i}$, which by linearity of the Poisson problem as well as the original Schwarz algorithm satisfies

$$\begin{aligned} -\frac{e_1^{n+1}}{dx^2} &= f \text{ in } (0, L_1), \\ e_1^{n+1}(0) &= 0, \\ e_1^{n+1}(L_1) &= e_2^n(L_1); \end{aligned} \qquad \text{then} \qquad \begin{aligned} -\frac{e_2^{n+1}}{dx^2} &= f \text{ in } (l_2, L), \\ e_2^{n+1}(l_2) &= e_1^{n+1}(l_2), \\ e_2^{n+1}(L) &= 0. \end{aligned}$$

The solution to the error problem is

$$e_1^{n+1}(x) = \frac{x}{L_1} e_2^n(L_1), \quad e_2^{n+1}(x) = \frac{L - x}{L - l_2} e_1^{n+1}(l_2).$$

These functions increase linearly from the boundary of the domain to the boundary of the overlapping region. The error at $x = L_1$ is updated as

$$e_2^{n+1}(L_1) = \frac{1 - \delta/(L - l_2)}{1 + \delta/l_2} e_2^n(L_1),$$

where $\delta = L_1 - l_2 > 0$ is the overlap. The error is reduced by a factor of

$$\rho_{\text{1D}} = \frac{1 - \delta/(L - l_2)}{1 + \delta/l_2}, \tag{2.28}$$

which indicates the convergence becomes quicker as the overlap increases [8, Section 1.5.1].

2D case

In the 2D case two half planes are considered $\Omega_1 = (-\infty, \delta) \times \mathbb{R}$ and $\Omega_2 = (\delta, \infty) \times \mathbb{R}$. Following the example of Dolean, Jolivet, and Nataf the problem is

$$-(\eta - \Delta)u = f \text{ in } \mathbb{R}^2,$$
$$u \text{ bounded at infinity},$$

where $\eta > 0$ is a constant. Proceeding in similar fashion as the one-dimensional case, the error $e_{1,2}^{n+1}$ can be solved for in the two subdomains. This is done via a partial Fourier transform of the problem in the y-direction yielding an ODE for the transformed error $\hat{e}_{1,2}^{n+1}$ with the added Fourier constant k, which can be solved explicitly with the ansatz

$$\hat{e}_{1,2}^{n+1}(x,k) = \gamma_1(k)e^{\lambda_+(k)x} + \gamma_2(k)e^{\lambda_-(k)x},$$

where $\lambda_\pm(k) = \pm\sqrt{k^2 + \eta}$. By using the interface conditions we find

$$\gamma_i^{n+1}(k) = \rho(k;\eta,\delta)^2\gamma_i^{n-1}(k),$$

such that the convergence factor is [8, Equation 1.36]

$$\rho_{\text{2D}}(k;\eta,\delta) = e^{-\delta\sqrt{\eta+k^2}} \tag{2.29}$$

which indicates that the convergence is quicker as the overlap increases as before. Next to this, it also shows that the convergence is quicker for higher k.

2.2.3. Need for a coarse space

Following upon the results in the previous section 2.2.2 it is clear that the convergence of the Schwarz method not only depends on the extent of the overlap between various subdomains, but on the frequency components of the solution as well. In a general sense this means that low frequency modes need for instance at least N_{sub} steps to travel from one end of a square domain to the other. This in turns causes plateaus in the convergence of the Schwarz method. To overcome this, we can perform a Galerkin projection of the error onto a coarse space. By solving

$$\min_\beta ||A(\mathbf{u} + R_0^T\beta) - f||^2,$$

where R_0 is a matrix representing the coarse space. The solution to this problem is

$$\beta = (R_0AR_0^T)^{-1}R_0r,$$

where $r = f - A\mathbf{u}$ is the residual.

The coarse space R_0 can be constructed in various ways. The classical way is called the Nicolaides space [8, Section 4.2], which uses the discrete partition of unity operators D_i as examplified in equation (2.26) to get

$$R_0 = \sum_{i=1}^{N_{\text{sub}}} R_i^T D_i R_i. \tag{2.30}$$

Note that the coarse space has a block-diagonal form.

Finally the coarse space correction term can be added to the Schwarz preconditioners equations (2.27a) and (2.27b) to get the following preconditioners

$$M_{\text{ASM,2}} = R_0^T(R_0AR_0^T)^{-1}R_0 + \sum_{i=1}^{N_{\text{sub}}} R_i^T(R_iAR_i^T)^{-1}R_i, \tag{2.31a}$$

$$M_{\text{RAS,2}} = R_0^T(R_0AR_0^T)^{-1}R_0 + \sum_{i=1}^{N_{\text{sub}}} R_i^T D_i(R_iAR_i^T)^{-1}R_i. \tag{2.31b}$$

2.2.4. Two-level additive Schwarz method

In this section the we will construct a coarse space for a Poisson problem with a constant scalar coefficient on arbitrary domain like in problem 2.25. However, the method is applicable to more general (highly) heterogeneous scalar problems, like the Darcy problem. The coarse space is constructed using the eigenfunctions corresponding to the smallest m_j eigenvalues resulting from a local eigenproblem in each subdomain Ω_j. The coarse space is then constructed by taking the union of the m_j eigenvectors corresponding to the smallest eigenvalues in each subdomain glued together by the partition of unity functions χ_j. All of this can be found in [8, Sections 5.1-5.5].

This coarse space is subsequently used to construct the two level additive Schwarz preconditioner, and bounds for its condition number are provided as well.

Slowly convergent modes of the Dirichlet-to-Neumann map

As seen in section 2.2.2 the local error in any subdomain in the Schwarz method satisfies the homogeneous version of the original problem, i.e. right hand side $f = 0$. At the interface the local error has a Dirichlet boundary condition that equals the error of the neighbouring subdomain. Additionally, the convergence factor, e.g. ρ_{2D}, depends on the frequency of the modes in the local error. In particular, small frequencies appear to have slow convergence. The question thus becomes how to get rid of these small frequency modes in the local errors of all subdomains.

One possible answer is the so-called Dirichlet-to-Neumann map [8, Definition 5.1]

Definition 2.6. (Dirichlet-to-Neumann map for a Poisson problem) For any function defined on the interface $u_{\Gamma_j} : \Gamma_j \mapsto \mathbb{R}$, we consider the Dirichlet-to-Neumann map

$$\mathrm{DtN}_{\Omega_j}\left(u_{\Gamma_j}\right) = \left.\frac{\partial v}{\partial \mathbf{n}_j}\right|_{\Gamma_j},$$

where $\Gamma_j := \partial\Omega_j \backslash \partial\Omega$ and v satisfies

$$\begin{aligned} -\Delta v &= 0 && \text{in } \Omega_j, \\ v &= u_{\Gamma_j} && \text{on } \Gamma_j, \\ v &= 0 && \text{on } \partial\Omega_j \cap \partial\Omega. \end{aligned} \tag{2.32}$$

The Dirichlet-to-Neumann map essentially solves for an error-like variable v that satisfies the Dirichlet local interface (or global boundary) conditions. DtN then maps the interface condition to the normal derivative of v on the interface, i.e. the Neumann condition. Now, as stated above and illustrated in [8, Figure 5.2] the low frequency modes of the error correspond to those modes that are nearly constant accross an interface, for which the Neumann condition is close to zero. So the problem of slowly convergent modes in the error of the Schwarz method is equivalent to a problem of finding eigenpairs of the DtN operator.

Hence we aim to solve the eigenvalue problem

$$\mathrm{DtN}_{\Omega_j}(v) = \lambda v,$$

which can be reformulated in the variational form. To that end let w be a test function that is zero on $\delta\Omega$. Multiply both sides of equation (2.32) by w, integrate over Ω_j and apply Green's theorem to get

$$\int_{\Omega_j} \nabla v \cdot \nabla w - \lambda \int_{\Gamma_j} \frac{\partial v}{\partial \mathbf{n}_j} w = 0, \quad \forall w.$$

Then, use the eigen property of v and fact that w is zero on Γ_j to get the eigen problem in the variational form

$$\text{Find } (v, \lambda) \text{ s.t. } \int_{\Omega_j} \nabla v \cdot \nabla w - \lambda \int_{\Gamma_j} vw = 0, \quad \forall w. \tag{2.33}$$

Construction of Two-level additive Schwarz preconditioner

As before we partition Ω into N_{sub} subdomains Ω_j, which overlap each other by one or several layers of elements in the triangulation $\mathcal{T}$. We make the the following observations

D1 For every degree of freedom $k \in \mathcal{N}$, there is a subdomain Ω_j such that ϕ_k has support in Ω_j [8, Lemma 5.3].

D2 The maximum number of subdomains a mesh element can belong to is given by

$$k_0 = \max_{\tau \in \mathcal{T}} \left(|\{j | 1 \leq j \leq N_{\text{sub}} \text{ and } \tau \subset \Omega_j\}| \right).$$

D3 The minimum number of colors needed to color all subdomains so that no two adjacent subdomains have the same color is given by

$$N_c \geq k_0$$

D4 The minimum overlap for any subdomain Ω_j with any of its neighboring subdomains is given by

$$\delta_j = \inf_{x \in \Omega_j \setminus \cup_{i \neq j} \bar{\Omega}_i} \text{dist}(x, \partial\Omega_j \setminus \partial\Omega).$$

D5 The partition of unity functions $\{\chi_j\}_{j=1}^{N_{\text{sub}}} \subset V_h$ are such that

D5.a $\chi_j(x) \in [0,1], \quad \forall x \in \bar{\Omega}, j = 1, \ldots, N_{\text{sub}},$

D5.b $\text{supp}(\chi_j) \subset \bar{\Omega}_j,$

D5.c $\displaystyle\sum_{j=1}^{N_{\text{sub}}} \chi_j(x) = 1, \quad \forall x \in \bar{\Omega},$

D5.d $\displaystyle\|\nabla \chi_j(x)\| \leq \frac{C_\chi}{\delta_j},$

and are given by

$$\chi_j(x) = I_h \left(\frac{d_j(x)}{\sum_{j=1}^{N_{\text{sub}}} d_j(x)} \right),$$

where

$$d_j(x) = \begin{cases} \text{dist}(x, \partial\Omega_j), & x \in \Omega_j, \\ 0, & x \in \Omega \setminus \Omega_j. \end{cases}$$

D6 The overlap region for any subdomain is given by

$$\Omega_j^\delta = \{x \in \Omega_j | \chi_j < 1\}.$$

From item **D1** it follows that the extension operator $E_j : V_{h,0}(\Omega_j) \to V_h$ can defined by

$$V_h = \sum_{j=1}^{N_{\text{sub}}} E_j V_{h,0}(\Omega_j).$$

Note that using the extension operator we can show that all the local bilinear forms are positive definite as

$$a_{\Omega_j}(v, w) = a(E_j v, E_j w) \geq \alpha \|E_j v\|_a^2, \quad \forall v, w \in V_{h,0}(\Omega_j),$$

and a is positive definite.

Finally, we define the a-symmetric projection operators $\tilde{\mathcal{P}}_j : V_{h,0} \to V_h$ and $\mathcal{P}_j : V_h \to V_h$ defined by

$$\begin{aligned} a_{\Omega_j}(\tilde{\mathcal{P}}_j u, v_j) &= a(u, E_j v_j) \quad \forall v_j \in V_{h,0}, \\ \mathcal{P} &= E_j \tilde{\mathcal{P}}_j. \end{aligned}$$

Then their matrix counterparts are given by

$$\begin{aligned} \tilde{P}_j &= A_j^{-1} R_j^T A, \\ P_j &= R_j^T A_j^{-1} R_j^T A, \end{aligned}$$

where $A_j = R_j A R_j^T$. From this we can construct the two-level additive Schwarz method as

$$M_{\text{ASM},2}^{-1} A = \sum_{j=1}^{N_{\text{sub}}} P_j. \tag{2.34}$$

2.2.5. Convergence of two-level additive Schwarz

In the following we denote

$$\mathcal{P}_{\text{ad}} = \sum_{j=1}^{N_{\text{sub}}} \mathcal{P}_j,$$

and correspondingly,

$$P_{\text{ad}} = \sum_{j=1}^{N_{\text{sub}}} P_j.$$

In the context of this thesis the two-level additive Schwarz method is used in combination with a Krylov subspace method, in which case convergence rate depends on the entire spectrum of eigenvalues (section 2.1.4). However, an upperbound for the convergence rate (section 2.1.3) can be derived from the condition number of P_{ad} via equation (2.10).

Using the fact that P_{ad} is symmetric (see [8, Lemma 5.8]) with respect to the a-norm, we can write

$$\kappa(P_{\text{ad}}) = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}},$$

where

$$\lambda_{\text{max}} = \sup_{v \in V_h} \frac{a(\mathcal{P}_{\text{ad}})}{a(v,v)}, \quad \lambda_{\text{min}} = \inf_{v \in V_h} \frac{a(\mathcal{P}_{\text{ad}})}{a(v,v)}.$$

Additionally, we can employ the a-orthogonality of the projection operators to get

$$\frac{a(\mathcal{P}_j u, u)}{\|u\|_a^2} = \frac{a(\mathcal{P}_j u, \mathcal{P}_j u)}{\|u\|_a^2} \leq 1.$$

Going further, we can pose that the projection operators defined by the sum of projection operators $\mathcal{P}_j$ of like-colored subdomains are a-orthogonal to each other. This is due to the fact that the partition of unity functions χ_j are such that they are zero on the interface of like-colored subdomains (see item **D3**). To that end, define

$$\mathcal{P}_{\Theta_i} = \sum_{j \in \Theta_i} \mathcal{P}_j,$$

where Θ_i is the set of indices of subdomains with color i and $i = 1, \dots, N_c$. Then, we can write [8, Lemma 5.9]

$$\begin{aligned} \lambda_{\text{max}}(\mathcal{P}_{\text{ad}}) &= \sup_{v \in V_h} \sum_{i=1}^{N_c} \frac{a(\mathcal{P}_{\Theta_i} v, v)}{a(v,v)} \\ &\leq \sum_{i=1}^{N_c} \sup_{v \in V_h} \frac{a(\mathcal{P}_{\Theta_i} v, v)}{a(v,v)} \\ &\leq N_c + 1, \end{aligned}$$

where the extra one comes from the coarse projection operator $\mathcal{P}_0$. Note that this bound can be made sharper by using item **D2** to get $\lambda_{\text{max}}(\mathcal{P}_{\Theta_i}) \leq k_0 + 1$.

On the other hand, it can be shown that the minimum eigenvalue satisfies provided that $v \in V_h$ admits a C_0-stable decomposition [8, Theorem 5.11]

$$\lambda_{\text{min}}(\mathcal{P}_{\text{ad}}) \geq C_0^{-2}.$$

Finally, we can write the condition number of the two-level additive Schwarz preconditioner as

$$\kappa(P_{\text{ad}}) \leq (N_c + 1)\, C_0^2. \tag{2.35}$$

The value of C_0 depends on the projection operator Π_j onto the chosen coarse space V_0 for each subdomain.

I. **Nicolaides coarse space** The projection operator is defined as

$$\Pi_j^{\text{Nico}} u = \begin{cases} (\frac{1}{|\Omega_j|}\int_{\Omega_j} u)\mathbf{1}_{\Omega_j}, & \delta\Omega_j \cap \delta\Omega = \emptyset, \\ 0, & \text{otherwise}, \end{cases} \tag{2.36}$$

which gives rise to the following basis functions in $V_{h,0}$

$$\Phi_j^{\text{Nico}} = I_h(\chi_j \mathbf{1}_{\Omega_j}).$$

Then,

$$V_0 = \text{span}\{\Phi_j^{\text{Nico}}\}_{j=1}^{N_{\text{sub}}},$$

and

$$\dim V_0 = \text{the number of floating subdomains},$$

that is the number of subdomains that are not connected to the boundary of the domain Ω. In this case [8, Theorem 5.16]

$$C_0^2 = \left(8 + 8C_\chi^2 \max_{j=1}^{N_{\text{sub}}}\left[C_P^2 + C_{\text{tr}}^{-1}\frac{H_j}{\delta_j}\right]k_0 C_{I_h}(k_0+1)+1\right), \tag{2.37}$$

where H_j is the diameter of the subdomain Ω_j, C_p the Poincaré constant following from [8, Lemma 5.18] and C_{tr} is the trace constant.

II. **Local eigenfunctions coarse space** The projection operator is defined as

$$\Pi_j^{\text{spec}} u = \sum_{k=1}^{m_j} a_{\Omega_j}(u, v_k^{(j)})v_k^{(j)},$$

where $v_k^{(j)}$ is the k^{th} eigenfunction resulting from the eigenproblem in equation (2.33). The basis functions in $V_{h,0}$ are then given by

$$\Phi_{j,k}^{\text{spec}} = I_h(\chi_j v_k^{(j)}),$$

resulting in the coarse space

$$V_0 = \text{span}\{\Phi_{j,k}^{\text{spec}}\}_{j=1,k=1}^{N_{\text{sub}},m_j},$$

with dimension

$$\dim V_0 = \sum_{j=1}^{N_{\text{sub}}} m_j.$$

In this case [8, Theorem 5.17]

$$C_0^2 = \left(8 + 8C_\chi^2 \max_{j=1}^{N_{\text{sub}}}\left[C_P^2 + C_{\text{tr}}^{-1}\frac{1}{\delta_j\lambda_{m_j+1}}\right]k_0 C_{I_h}(k_0+1)+1\right). \tag{2.38}$$

3 Related Work

3.1. The spectral gap arising in Darcy problems

In a Darcy problem, high-contrast $\mathcal{C}(x)$ (e.g., 10^6 in conductive channels vs. 10^{-6} in barriers) means flow concentrates in high-permeability regions, while low-permeability zones resist flow. This heterogeneity introduces modes (eigenvectors) that are nearly constant or slowly varying over low-$\mathcal{C}(x)$ regions, contributing small eigenvalues to A. These modes represent "trapped" or "isolated" behaviors disconnected by the contrast.

Small eigenvalues arise when the bilinear form $a(u, v) = \int_\Omega \mathcal{C}(x)\nabla u \cdot \nabla v \, dx$ yields low energy for certain test functions v. In high-contrast cases, if $\mathcal{C}(x)$ is extremely low in a subdomain, ∇u must be large there to balance the equation, but FEM basis functions often cannot resolve this without fine meshes. Instead, coarse bases produce modes where energy is minimized, leading to eigenvalues close to zero. This is exacerbated as contrast grows, adding more such modes.

High-contrast $\mathcal{C}(x)$ can split the spectrum into clusters: large eigenvalues tied to high-$\mathcal{C}(x)$ regions (where gradients dominate) and small eigenvalues tied to low-$\mathcal{C}(x)$ regions (where flow stagnates). In [9], the authors note that standard FEM misses these small eigenvalues unless enriched bases capture fine-scale features.

3.2. Tailored coarse spaces for high-contrast problems

Various methods for constructing a coarse space that are both scalable and robust to high contrast in a problem coefficient.

3.2.1. MsFEM

The Multiscale Finite Element Method (MsFEM), as presented in [9, 10, 13], constructs a coarse space based on five key assumptions (C1-C5). These assumptions ensure stability and accuracy by defining how the coarse space interacts with the fine-scale problem. Local coarse grid basis functions are obtained by solving the homogeneous version of the system equation, meaning they do not include external forcing terms. The construction of these basis functions requires specific boundary conditions, categorized as M1-M4, which control their behavior at interfaces. The method distinguishes between linear and oscillatory boundary conditions for local problems, affecting the resulting coarse space. Coarse grid basis functions are computed as harmonic extensions of basis functions restricted to edges or faces, ensuring continuity across subdomains. The restriction operator R_0 is then derived from these basis functions, as given in Equation 2.12 of [10]. Additionally, the method introduces robustness indicators, $\pi(\alpha)$ and $\gamma(\alpha)$, to quantify the stability of the coarse space and its effectiveness in capturing fine-scale features.

3.2.2. ACMS

The Approximate Component Mode Synthesis (ACMS) method, detailed in [11], introduces a separation of scales with fine and coarse triangulations, denoted as $\mathcal{T}_h$ and $\mathcal{T}_H$. The coarse problem is decomposed into two components: $u_c = u_I + u_\Gamma$, where u_I and u_Γ represent the interior and interface contribution, respectively. This extends MsFEM by incorporating vertex-specific, edge-specific, and fixed-interface basis functions, where MsFEM corresponds solely to the vertex-specific functions. The vertex-specific basis functions are defined as harmonic extensions of trace values on the interface set Γ. Edge-specific basis functions, on the other hand, arise from an eigenvalue problem defined on an edge e, while fixed-interface basis functions correspond to eigenmodes of an eigenvalue problem within a coarse element T.

ACMS supports two types of coarse spaces, depending on whether Dirichlet (DBC) or Neumann (NBC) boundary conditions are applied. Under DBCs, MsFEM basis functions are combined with edge-specific basis functions that match on a shared edge e_{ij} between subdomains Ω_i and Ω_j. These functions are constructed from the harmonic extension of eigenmodes defined on the edge e_{ij}, with a scaled bilinear form on the right-hand side. Only eigenmodes corresponding to eigenfrequencies below a set tolerance are retained. With NBCs, both MsFEM and edge-specific basis functions are modified. MsFEM functions remain defined on an edge e_{ij} and satisfy a Kronecker-delta vertex condition but are now obtained via a generalized eigenvalue problem on a slab of width kh, denoted η_{ij}^{kh}. The edge-specific functions are similarly defined through a generalized eigenvalue problem on the slab but without enforcing DBCs. Solving these eigenvalue problems can be computationally efficient by employing mass matrix lumping techniques.

3.2.3. GDSW

The Generalized Dryja-Smith-Widlund (GDSW) method, introduced by [6], partitions the computational domain into non-overlapping subdomains and further divides degrees of freedom (DOFs) into interior and interface nodes. The only required input for the method is a coarse space G. G corresponds to the null space of the problem. In the case of linear elasticity, G spans the rigid body modes, while for the diffusion problem the null space is a constant function. The restriction operators R_Γ and R_I project onto interface and interior DOFs, respectively, with subdomain-specific versions such as R_{Γ_j}.

The coarse problem matrix is [1, Equation 2]

$$A_0 = \Phi^T A \Phi,$$

where, by ordering the DOFs of into interface Γ and interior I nodes we can get [1, Equation 4, 5]

$$\Phi = \begin{pmatrix} -A_{II}^{-1} A_{I\Gamma} \\ I_{\Gamma\Gamma} \end{pmatrix} \Phi_\Gamma,$$

in which Φ_Γ is the prolongation operator of the interface nodes. The coarse solution on the interface set is subsequently defined as

$$u_{0,\Gamma} = \sum_{j}^{N_{\text{sub}}} R_{\Gamma_j}^T G_{\Gamma_j} q_j = \Phi_\Gamma q,$$

where q represents the coarse space coefficients. The complete coarse solution is then given by

$$u_0 = \Phi_\Gamma^T u_{0,\Gamma} + \Phi_I^T u_{0,I},$$

where $\Phi_I = -A_{II}^{-1} A_{I\Gamma} \Phi_\Gamma$ is computed by discrete harmonic extension.

RGDSW

The RGDSW method alters the GDSW preconditioner by reducing the coarse space dimension through a partitioning strategy based on nodal equivalence classes that associates each coarse mesh vertex with interface components formed by adjacent edges and faces, distributed among nearby vertices [7]. This reduction in the dimension of the coarse space is achieved without compromising the robustness of the condition number estimate, ensuring that the preconditioner's convergence properties are maintained [12].

3.2.4. AMS

The Algebraic Multiscale Solver (AMS) method, introduced in [15, 20] and further studied in [1], also relies on domain decomposition into non-overlapping subdomains, followed by a further subdivision of interface nodes into edge, vertex, and face nodes (in 3D). The method eliminates lower diagonal blocks in the system matrix to facilitate efficient computation. Like (R)GDSW, AMS employs the energy minimization principle to obtain Φ_I, ensuring an optimal coarse space representation.

3.3. CG convergence in case of non-uniform spectra

Excessive work has been done on the convergence rate of the CG method, especially in the context of non-uniform spectra. The following papers provide valuable insights into this topic.

First, in [2] a clever use of Chebyshev polynomials is demonstrated to obtain a sharpened CG iteration bound for two kinds of eigenspectra. This technique is further developed in section 5.1, providing additional insights into the convergence behavior of CG.

Second, in [19] the convergence rate of CG is investigated in the case of non-uniform spectra. The authors show that the convergence rate strongly depends on the eigenvalue distribution of the matrix A. By examining a parametrized class of matrices, they study how small perturbations in eigenvalues impact CG performance. Their experiments reveal that there exists a critical value of a parameter at which the number of iterations required for convergence greatly exceeds the degrees of freedom N, even when the condition number is small ($K = 100$) and N is small (e.g., 12 or 24). Moreover, this critical value shifts with increasing precision, so that higher precision reduces the impact of rounding errors. The study further shows that the CG behavior is consistent across different algorithm variants (standard CG, SYMMQL, Jacobi acceleration, etc.). These results suggest that certain eigenvalue distributions make CG highly sensitive to numerical errors, and they underline the importance of preconditioners that can modify the eigenvalue distribution to improve CG robustness in practical applications.

Third, in [5] the authors provide a proof of CG's superlinear convergence. They explain why the conjugate gradient method exhibits faster (superlinear) convergence than the classical bound suggests. In particular, they derive a new asymptotic error bound that is sharper than the standard estimate. This new bound is expressed via the integral [5, Equation 1.8]:

$$\frac{1}{m}\log\left(\min_{r\in\mathcal{P}_m,\ r(0)=1}\ \max_{\lambda\in\sigma(A)}|r(\lambda)|\right)\lesssim -\frac{1}{t}\int_0^t g_{S(\tau)}(0)d\tau, \tag{3.1}$$

where $t = \dfrac{m}{N}$, $S(\tau)$ is a family of sets determined by the eigenvalue distribution and g_S is the Green function for the complement of S with pole at ∞. As shown by equation (3.1) and theorem 2.1 in [3], the error is bounded by a term that decreases more quickly as the number of iterations increases. Under additional separation conditions among eigenvalues (see Theorem 2.2), the bound is proven to be asymptotically sharp. Moreover, for matrices with equidistant eigenvalues, an explicit formula [5, Corollary 3.2 and Equation 3.11] confirms the improved rate and aligns with observed CG error curves. These findings help to explain why, in practice, CG converges faster than predicted by traditional condition number bounds.

Fourth, in [3] the authors present a proof of the sharpness of the CG iteration bound equation (3.1). The paper shows that one cannot beat the asymptotic error estimate bound obtained earlier. It analyzes a strategy in which zeros of the polynomial are set at all eigenvalues outside a chosen set S. The authors prove that any such polynomial cannot yield a better asymptotic error bound than the one given by the earlier formula. They use a constrained energy problem from logarithmic potential theory to define the optimal set $S(t)$. Under conditions that are natural for problems arising from discretized PDEs on the eigenvalue distribution, the bound is demonstrated to be sharp, and the discussion includes cases where equality in the bound is reached.

Finally, in [4] the authors extend the results of [5] to cases where the eigenvalue distribution is asymptotically uniform. They show that even when the asymptotic distribution equals an equilibrium distribution, the CG method can exhibit superlinear convergence. In this work, the superlinearity stems from the particular choice of the right-hand side b. The authors present a family of examples based on the finite difference discretization of the one-dimensional Poisson problem, where they observe superlinear convergence according to the chosen right-hand sides.

4 Research questions

4.1. Main research question

The main research question in this work is as follows:

Research Question. How can we sharpen the CG iteration bound for Schwarz-preconditioned high-contrast heterogeneous scalar-elliptic problems beyond the classical condition number-based bound?

For instance, in [1], the AMS and GDSW preconditioners significantly outperform the RGDSW preconditioner, despite all three having similar condition numbers. The key differences appear in their spectral gap and cluster width, highlighting the need for additional spectral characteristics to refine existing bounds.

4.2. Subsidiary research questions

Subsidiary Questions. To answer the main research question, we address the following subsidiary questions:

Q1 What spectral characteristics, like the condition number, can we define to estimate the distribution of eigenvalues in the eigenspectrum in the case of high-contrast heterogeneous problems?

Q2 How can we estimate any of the spectral characteristics defined in **Q1** for the eigenspectrum in the particular case of a model Darcy problem?

Q3 Given a certain eigenspectrum, how can we sharpen the CG iteration bound?

Q4 How does the sharpened bound from **Q3** perform for an unpreconditioned Darcy problem in comparison with the classical bound in equation (2.10)?

Q5 How does the performance described in **Q4** of a sharpened bound vary with the measures found in **Q1**?

Q6 How can we employ the sharpened bound to distinguish between the performance of Schwarz-like preconditioners?

4.3. Motivation

This research is important because current studies primarily focus on selecting between different Schwarz preconditioners for the Darcy problem, yet condition numbers fail to distinguish them effectively. The ability to differentiate preconditioners based on spectral properties would improve the selection process. Applying sharpened bounds could lead to better predictions of preconditioner performance and improved efficiency in solving high-contrast problems.

4.4. Challenges

The main challenge lies in quantitatively estimating the spectrum of the preconditioned system. The literature provides a priori condition number estimates for various Schwarz preconditioners. For instance, in the simple cases of the additive Schwarz preconditioner with either a **ASM type I coarse space** or **ASM type II coarse space** an a priori estimate for the condition number is given by equation (2.35) in combination with either equation (2.37) or equation (2.38), respectively. The same can be said for the MsFEM and ACMS preconditioners. Despite this, there is no established method for estimating the full eigenspectrum. Without such an estimate, refining the CG iteration bound remains difficult. Overcoming this limitation is central to this work.

5 Preliminary Results

The results described in this chapter are adapted from the ideas discussed in [2, Section 4]. Therein Axelsson presents a sharpened CG iteration bound for two particular eigenspectra, which are described in section 5.1. The sharpened bound is then generalized to multiple clusters in section 5.2. The results are then compared with the classical CG iteration bound in section 5.3. Finally, the implications of the sharpened bound for the research questions are discussed in section 5.4.

5.1. Two cluster case

On the eigenspectrum of A, consider two intervals $[a, b]$ and $[c, d]$ with $0 < a < b < c < d$ such that all eigenvalues of A are contained in the union of these two intervals. Additionally, we have $\kappa(A) = \frac{d}{a}$. We treat the following two cases simultaneously

$$\sigma_1(A) = [a, b] \bigcup [c, d] \tag{5.1}$$

$$\sigma_2(A) = [c, d] \bigcup_{\substack{i=1 \\ \lambda_i \in [a,b]}}^{N_{\text{tail}}} \lambda_i \tag{5.2}$$

where N_{tail} is the number of eigenvalues in the tail. The first case is a two-cluster eigenspectrum, while the second case has one cluster and a tail of eigenvalues.

In order to derive a CG iteration bound for these two cases we proceed as in the classical case laid out in 2.1.3. We know CG is optimal in the A-norm by equation (2.10), from which it follows that the error at the m^{th} iterate can be bounded as

$$||e_m||_A \leq \min_{r \in \mathcal{P}_m, r(0)=1} \max_{\lambda \in \sigma_i(A)} |r(\lambda)|||\mathbf{e}_0||_A, \text{ for } i = 1, 2, \tag{5.3}$$

To get an upper bound for m equation (5.3) suggests we look for a polynomial $r_{\bar{m}}$ of degree $\bar{m}$ that satisfies

$$\min_{r \in \mathcal{P}_{\bar{m}}, r(0)=1} \max_{\lambda \in \sigma_i(A)} |r(\lambda)| \leq \frac{||e_m||_A}{||\mathbf{e}_0||_A} = \epsilon, \text{ for } i = 1, 2,$$

in which ϵ is the relative error.

Axelsson suggests we use not one monolithic residual polynomial function, but a multiplication of two residual polynomial functions $\hat{r}_p^{(i)}$ and $\hat{r}_{\bar{m}-p}$ for the two clusters. The superscript $^{(i)}$ corresponds to the two eigenspectra described above. The residual polynomial functions are defined as

$$\hat{r}_p^{(i)}(x) \begin{cases} C_p\left(\frac{b+a-2x}{b-a}\right) / C_p\left(\frac{b+a}{b-a}\right) & \text{, if } i = 1 \\ \prod_{i=1}^{p}(1 - x/\lambda_i) & \text{, if } i = 2, p = N_{\text{tail}} \end{cases} \tag{5.4}$$

and

$$\hat{r}_{\bar{m}-p}(x) = C_{m-p}\left(\frac{d+c-2x}{d-c}\right) / C_{\bar{m}-p}\left(\frac{d+c}{d-c}\right), \tag{5.5}$$

Indeed, the product $r_{\bar{m}} = \hat{r}_p \hat{r}_{\bar{m}-p} \in \mathcal{P}_{\bar{m}}$. Hence, we can use the residual polynomial functions to bound the error at the m^{th} iterate. Now, we obtain the following intermediate bounds

$$\max_{\lambda\in[a,b]} |r_{\bar{m}}(\lambda)| \leq \max_{\lambda\in[a,b]} |\hat{r}_p^{(i)}(\lambda)| \max_{\lambda\in[a,b]} |\hat{r}_{\bar{m}-p}(\lambda)| \leq \max_{\lambda\in[a,b]} |\hat{r}_p^{(i)}(\lambda)|, \text{ and} \tag{5.6a}$$

$$\max_{\lambda\in[c,d]} |r_{\bar{m}}(\lambda)| \leq \max_{\lambda\in[c,d]} |\hat{r}_p^{(i)}(\lambda)| \max_{\lambda\in[c,d]} |\hat{r}_{\bar{m}-p}(\lambda)| \leq \max_{\lambda\in[c,d]} |\hat{r}_p(\lambda)| / C_{\bar{m}-p}\left(\frac{d+c}{d-c}\right) \tag{5.6b}$$

where the first result follows from the fact that $|\hat{r}_{m-p}(x)| < 1\ \forall x \in [a,b]$ and the second result from

$$\left| C_{m-p}\left(\frac{d+c-2x}{d-c}\right) \right| < 1\ \forall x \in [c,d].$$

Furthermore, using the well-known inequality, we have

$$1/C_k\left(\frac{z_1+z_2}{z_1-z_2}\right) \leq 2\left(\frac{\sqrt{z_2}-\sqrt{z_1}}{\sqrt{z_2}+\sqrt{z_1}}\right)^k, \text{ for } z_1 > z_2 > 0 \text{ and } k \in \mathbb{N}^+, \tag{5.7}$$

and

$$\max_{\lambda\in[a,b]} |\hat{r}_p^{(i)}(\lambda)| \leq \begin{cases} 2\left(\dfrac{\sqrt{b}-\sqrt{a}}{\sqrt{b}+\sqrt{a}}\right)^p = \eta_1 & \text{, if } i=1, \\ \left(\dfrac{b}{a}-1\right)^p = \eta_2 & \text{, if } i=2, p=N_{\text{tail}}, \end{cases}$$

Note that if $i=1$ we can determine p by requiring that the maximum of the residual polynomial function $\hat{r}_p^{(i)}$ in $[a,b]$ is equal to ϵ. This gives the following equation

$$p = \left\lceil \frac{1}{2}\sqrt{\frac{b}{a}} \ln \epsilon + 1 \right\rceil \tag{5.8}$$

Also note that for $i=2$ $\hat{r}_p^{(2)}(\lambda) = 0$ for all eigenvalues $\lambda \in [a,b]$ and thereby, bounded by ϵ.

Next, $\hat{r}_p^{(i)}$ in $[c,d]$ is bounded by its maximum value within $[a,b]$ multiplied by the polynomial that is the fastest growing polynomial outside- and bounded below 1 within $[a,b]$. This polynomial is again the (transformed) Chebyshev polynomial $C_p\left(\frac{2x-b-a}{b-a}\right)$. Therefore,

$$\max_{\lambda\in[c,d]} |\hat{r}_p^{(i)}(\lambda)| \leq \eta_i C_p\left(\frac{2d-b-a}{b+a}\right)$$

At this point we have ensured equation (5.6a) is bounded by ϵ. So it remains to bound equation (5.6b). Using above results we can write

$$\max_{\lambda\in[c,d]} |r_{\bar{m}}(\lambda)| < \epsilon,$$

if we require that

$$\eta_i C_p\left(\frac{2d-b-a}{b-a}\right) / C_{\bar{m}-p}\left(\frac{d+c}{d-c}\right) < \epsilon. \tag{5.9}$$

Using that for $x_1, x_2, x_3 \in \mathbb{R}^+$ with $x_1 > x_3$ and $z = \dfrac{x_1 - x_2}{x_3}$

$$\begin{aligned} C_p(z) &\leq \left(z + \sqrt{z^2 - 1}\right)^p \\ &= \left(\frac{x_1 - x_2}{x_3} + \sqrt{\left[\frac{x_1 - x_2}{x_3}\right]^2 - 1}\right)^p \\ &\leq \left(\frac{x_1}{x_3} + \sqrt{\left[\frac{x_1}{x_3}\right]^2 - 1}\right)^p \\ &\leq \left(\frac{2x_1}{x_3}\right)^p, \end{aligned}$$

and substituting $x_1 = 2d$, $x_2 = b + a$ and $x_3 = b - a$ we obtain the following inequality

$$\eta_i \left(\frac{4d}{b-a}\right)^p / C_{\bar{m}-p}\left(\frac{d+c}{d-c}\right) < \epsilon.$$

Moreover,

$$\begin{aligned} \eta_i \left(\frac{4d}{b-a}\right)^p &= \begin{cases} 2\left(\dfrac{\sqrt{b}-\sqrt{a}}{\sqrt{b}+\sqrt{a}}\dfrac{4d}{b-a}\right)^p & \text{, if } i = 1 \\ \left(\dfrac{b-a}{a}\dfrac{4d}{b-a}\right)^p & \text{, if } i = 2, \end{cases} \\ &= \begin{cases} 2\left(\dfrac{4d}{b+2\sqrt{ab}+a}\right)^p & \text{, if } i = 1 \\ \left(\dfrac{4d}{a}\right)^p & \text{, if } i = 2, \end{cases} \\ &\leq 2\begin{cases} \left(\dfrac{4d}{b}\right)^p & \text{, if } i = 1 \\ \left(\dfrac{4d}{a}\right)^p & \text{, if } i = 2, \end{cases} \end{aligned}$$

We can therefore require that the error bound above is satisfied if we have

$$1/C_{\bar{m}-p}\left(\frac{d+c}{d-c}\right) \leq \frac{\epsilon}{2\left(\frac{4d}{e_i}\right)^p},$$

where

$$e_i = \begin{cases} b & \text{, if } i = 1 \\ a & \text{, if } i = 2. \end{cases}$$

Again using equation (5.7) and solving for the degree $\bar{m} - p$ we obtain

$$\bar{m} - p \geq \frac{1}{2}\sqrt{\frac{d}{c}}\left(\ln \epsilon + p \ln \frac{4d}{e_i}\right),$$

which leads to the following bound for the number of iterations [2, Equation 4.4]

$$\bar{m} = \left\lceil \frac{1}{2}\sqrt{\frac{d}{c}} \ln(2/\epsilon) + \left(1 + \frac{1}{2}\sqrt{\frac{d}{c}} \ln(4d/e_i)\right) p \right\rceil, \tag{5.10}$$

where

$$1 \leq p \leq \min\left\{\left\lceil \frac{1}{2}\sqrt{\frac{b}{a}} \ln \epsilon + 1 \right\rceil, N_{\text{tail}}\right\}.$$

5.2. Generalization to multiple clusters

At this point we assume that we are dealing with an eigenspectrum of the form $\sigma_1(A)$, i.e. we are only treating case 1. In section 5.4, it is reasoned that this is indeed a very applicable case for a discretized Darcy problem.

In this case, the technique outlined in section 5.1 starts at the left most cluster $[a, b]$, finds the Chebyshev degree $p_1 = p$ satisfying inequality 5.8, moves to the neighboring cluster $[c, d]$ and finds the Chebyshev degree $p_2 = \bar{m} - p$ satisfying inequality 5.9. Rewriting inequality 5.9 gives the following equation for p_2:

$$\frac{1}{C_{p_2}\left(\frac{d+c}{d-c}\right)} \leq \frac{\epsilon}{C^{(1)}_{p_1}(d)} = \epsilon_2, \tag{5.11}$$

where

$$C^{(1)}_{p_1}(x) = C_{p_1}\left(\frac{b+a-2x}{b-a}\right)/C_{p_1}\left(\frac{b+a}{b-a}\right),$$

is the Chebyshev polynomial corresponding to the first cluster.

Suppose there is a third cluster next to $[c, d]$, i.e. $[e, f]$. We can repeat the process and find the Chebyshev degree p_3 satisfying a similar inequality as 5.11 for the third cluster.

$$\frac{1}{C_{p_3}\left(\frac{f+e}{f-e}\right)} \leq \frac{\epsilon}{C^{(1)}_{p_1}(f)C^{(2)}_{p_2}(f)} = \epsilon_3,$$

This leads to the general equation for the Chebyshev degree p_i of the i^{th} cluster $[a_i, b_i]$

$$\frac{1}{C_{p_i}\left(\frac{b_i+a_i}{b_i-a_i}\right)} \leq \frac{\epsilon}{\prod_{j=1}^{i-1} C^{(j)}_{p_j}(b_i)} = \epsilon_i. \tag{5.12}$$

Due to the large range of the Chebyshev polynomials $\tilde{C}_p$ a computer is likely to result in floating point number overflow during calculation of the denominator of equation (5.12). Instead, we first apply inequality 5.7 and introduce the cluster condition numbers $\kappa_i = \dfrac{b_i}{a_i}$, where i is the index of the cluster. We can then rewrite equation (5.12) as follows

$$p_i = \left\lceil \ln \frac{\epsilon_i}{2} / \ln \frac{\sqrt{\kappa_i}-1}{\sqrt{\kappa_i}+1} \right\rceil,$$

and

$$\ln \frac{\epsilon_i}{2} = \ln \frac{\epsilon}{2} - \sum_{j=1}^{i-1} \ln C^{(j)}_{p_j}(b_i).$$

Let $z_1^{(i,j)} = \dfrac{b_j + a_j - 2b_i}{b_j - a_j}$ and $z_2^{(j)} = \dfrac{b_j + a_j}{b_j - a_j}$ then

$$\ln C^{(j)}_{p_j}(b_i) = \ln C_{p_j}(z_1^{(i,j)}) - \ln C_{p_j}(z_2^{(j)}).$$

We have, using the definition of the Chebyshev polynomial

$$\ln C_{p_j}(z_1^{(i,j)}) \lessapprox p_j \ln \left[z_1^{(i,j)} - \sqrt{\left(z_1^{(i,j)}\right)^2 - 1} \right] - \ln 2, \tag{5.13}$$

and

$$\ln C_{p_j}(z_2^{(j)}) \gtrapprox p_j \ln \left[z_2^{(j)} + \sqrt{\left(z_2^{(j)}\right)^2 - 1} \right] - \ln 2, \tag{5.14}$$

both of which become more accurate equalities as $z, m \to \infty$. Introducing

$$\zeta_1^{(i,j)} = z_1^{(i,j)} - \sqrt{\left(z_1^{(i,j)}\right)^2 - 1},$$
$$\zeta_2^{(j)} = z_2^{(j)} + \sqrt{\left(z_2^{(j)}\right)^2 - 1}, \text{ and}$$
$$f_i = \frac{\sqrt{\kappa_i} - 1}{\sqrt{\kappa_i} + 1},$$

with κ_i the i^{th} cluster condition number, and substituting the inequalities 5.13 and 5.14 back into the bound for p_i gives

$$\begin{aligned} p_i &\leq \left\lceil \frac{\ln \frac{\epsilon}{2} - \sum_{j=1}^{i-1} p_j \left\{ \ln \zeta_1^{(i,j)} - \ln \zeta_2^{(j)} \right\}}{\ln f_i} \right\rceil \\ &= \left\lceil \log_{f_i} \frac{\epsilon}{2} - \sum_{j=1}^{i-1} p_j \left\{ \log_{f_i} \zeta_1^{(i,j)} - \log_{f_i} \zeta_2^{(j)} \right\} \right\rceil \\ &= \left\lceil \log_{f_i} \frac{\epsilon}{2} - \sum_{j=1}^{i-1} p_j \log_{f_i} \frac{\zeta_1^{(i,j)}}{\zeta_2^{(j)}} \right\rceil \end{aligned}$$

Note that in general $\zeta_1^{(i,j)} < \zeta_2^{(j)}$ and hence $\log_{f_i} \frac{\zeta_2^{(j)}}{\zeta_1^{(i,j)}} > 0$. This prompts us to write

$$p_i \leq \left\lceil \log_{f_i} \frac{\epsilon}{2} + \sum_{j=1}^{i-1} p_j \log_{f_i} \frac{\zeta_2^{(j)}}{\zeta_1^{(i,j)}} \right\rceil \tag{5.15}$$

Evidently, adding more clusters to the left of the interval $[a_i, b_i]$ increases the degree p_i of the Chebyshev polynomial. Next to this, equation (5.15) reduces to the classical CG iteration bound equation (2.17) for a single cluster when $i = N_{\text{clusters}} = 1$.

Equation 5.15 gives us a way to calculate the Chebyshev degree p_i of the i^{th} cluster $[a_i, b_i]$ in terms of the Chebyshev degrees of the previous clusters. To obtain a bound on the number of iterations for the CG method we sum the Chebyshev degrees of all the clusters

$$\bar{m} = \sum_{i=1}^{N_{\text{clusters}}} p_i \tag{5.16}$$

5.3. Numerical experiments

Equations 5.15 and 5.16 give a sequential algorithm for determining an upper bound on the number of iterations for the CG method. Figure 5.1 compares this bound with the classical CG iteration bound equation (2.17). As is the case for figure 2.3, $m_{\text{classical}} = 26$

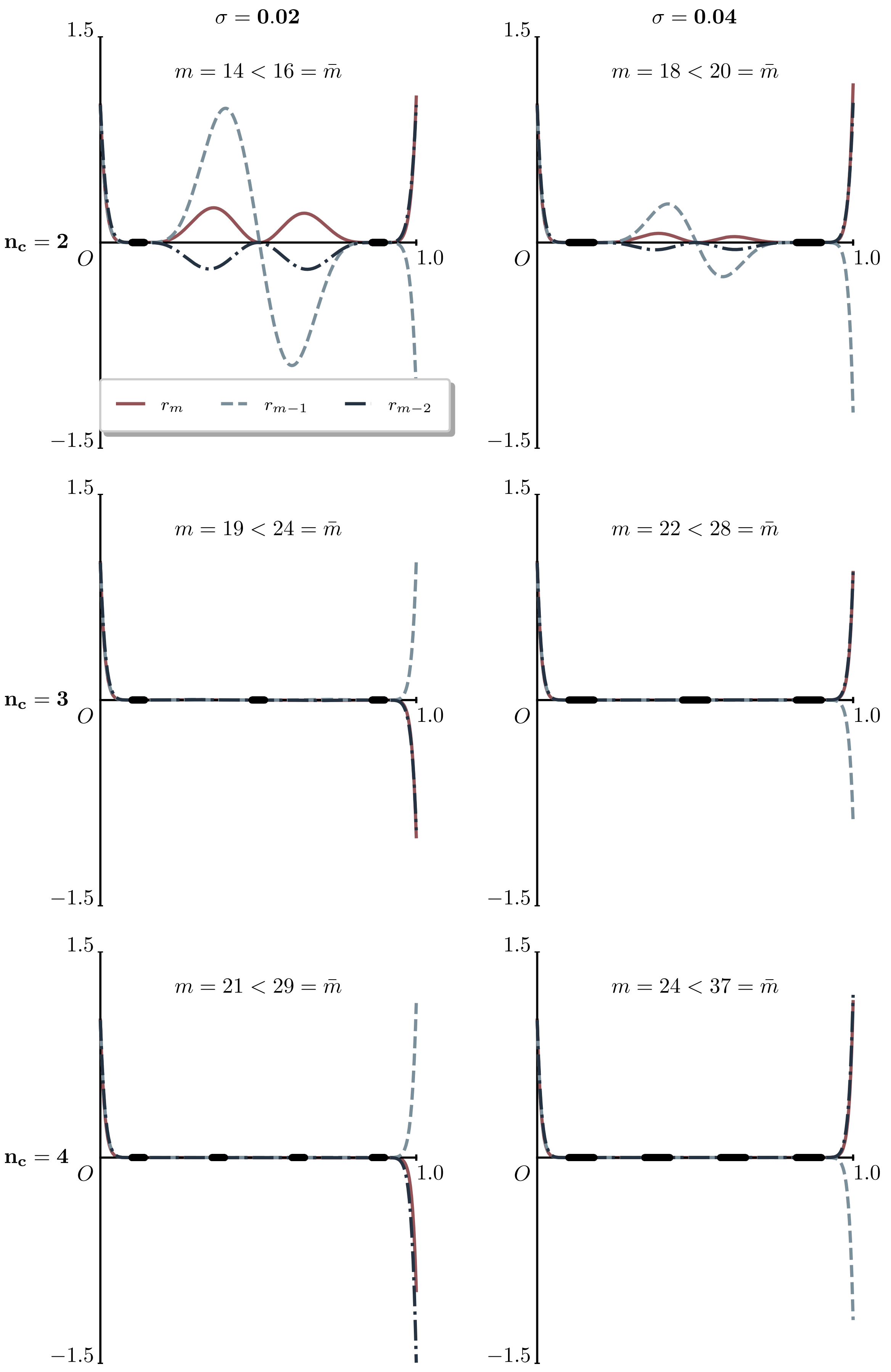

Figure 5.1: As in figure 2.3, plots of the last three CG residual polynomials for different eigenvalue distributions. n_c indicates the number of clusters and σ is the width of the cluster. The size of the system N and the condition number $\kappa(A)$ are kept constant. m indicates the number of iterations required for convergence. However, here $\bar{m}$ is determined by equations (5.15) and (5.16).

Figure 5.1 shows that the sharpened CG iteration bound is significantly lower than the classical CG iteration bound for the two cluster case. The performance of the sharpened bound does decrease as the number of clusters increases, as is evident from equations (5.15) and (5.16). Performance also decreases as clusters become wider. So much so, that the sharpened bound is worse than the classical bound for the three cluster case with spread $\sigma = 0.04$ and for the four cluster case.

Worsening performance for the sharpened bound with increased cluster width is expected. Focussing on the two cluster case, we rediscover the ratios $\frac{d}{c}$ in equation (5.10) as well as $\frac{a}{b}$ in the corresponding equation for p. These ratios grow with increasing cluster width.

5.4. Implications for research

The preliminary results discussed in this chapter show that we can find both an a priori analytic two-cluster (equation (5.10)) and multiple-cluster (equation (5.16)) sharpened iteration bound for the CG method. The latter can also be implemented as a sequential, numerical algorithm that can be applied to artificially constructed spectra in figure 5.1. The sharpened bound appears to perform best in the two-cluster case, which has the most correspondence to the eigenspectrum of a typical (preconditioned) Darcy problem. Hence, **Q3** is answered positively.

With regard to **Q1**, the cluster condition number κ_i is introduced as a measure of the cluster width. This suggests that we can use the cluster condition number to distinguish between different preconditioners which is a promising result for **Q6**. Moreover, inspecting equation (5.10) more closely for the eigenspectrum in equation (5.1) (case $i = 1$) reveals the presence of a sort of spectral gap $\frac{d}{b}$. This serves as yet another candidate for a potential set of spectral characteristics to estimate the distribution of eigenvalues in the eigenspectrum.

A logical next step is to more rigorously investigate how the sharpened bound depends on κ_i as well as the spectral gap (**Q5**). Subsequently, we can simulate the eigenspectrum of a Darcy problem and compare the sharpened bound with the classical bound (**Q4**). This will lead to a clear understanding of the performance of the sharpened bound in the main problem context of this thesis: heterogeneous scalar elliptic problems with high-contrast coefficient.

Furthermore, we can construct, discretize, and precondition a model Darcy problem with the methods outlined in section 3.2. Then, we apply both the sharpened bound and the CG method to the resulting systems, and investigate how sharp the new bound is (**Q6**).

The main challenge described in section 4.4 still stands. More work is needed to be able to use the sharpened bound for spectra that are not known or artificially constructed beforehand. The results in this chapter suggest that the cluster condition number is a good candidate for a measure of the eigenspectrum. However, it is not yet clear how to estimate the cluster condition number for a general eigenspectrum. This is an important step towards answering **Q2**.

6 Conclusion

This thesis stresses that the classical condition number-based CG iteration bound does not fully capture the convergence behavior in high-contrast heterogeneous elliptic problems, particularly when Schwarz preconditioners are employed. By incorporating additional spectral characteristics, such as the distribution, clustering, and gaps of eigenvalues in the eigenspectrum, the refined bound offers a more accurate and discriminative measure of iterative performance. Preliminary results indicate that these spectral properties significantly influence the convergence rate and can be exploited to predict and compare the efficacy of different preconditioners.

Further research is required to test the refined bound on an eigenspectrum typically found in Darcy problems and to develop robust methods for estimating the full eigenspectrum in high-contrast problems found in practice. This research contributes to the theoretical understanding of iterative solvers and has practical importance for improving computational efficiency in solving complex elliptic PDEs.

Appendix

A. Derivation of the CG Method

A.1. Arnoldi's method for linear systems

Arnoldi's method for linear systems $A\mathbf{u} = \mathbf{b}$, where A is a general (possibly non-symmetric) stiffness matrix, is just an instantiation of algorithm 1. It uses a Gramm-Schmidt orthogonalization procedure to simultaneously obtain the basis V of $\mathcal{K}$ and the Hessenberg matrix, see Definition 2.2. Assuming without loss of generality that V has dimension m, we set $V = V_m$ and let $\mathbf{v}_1 = \mathbf{r}_0/||\mathbf{r}_0||_2$ and $\beta = ||\mathbf{r}_0||_2$, then by Definition 2.1 we have

$$V_m^T A V_m = H_m \text{ and } V_m^T \mathbf{r}_0 = V_m^T \beta \mathbf{v}_1 = \beta e_1 \implies \begin{array}{c} \mathbf{u}_m = \mathbf{u}_0 + V_m \mathbf{c}, \\ H_m \mathbf{c} = \beta e_1. \end{array}$$

Substituting this into the template for the error projection methods given in algorithm 1 gives algorithm A.1.

Algorithm A.1 Arnoldi's method for linear systems (FOM) [17, Algorithm 6.4]

```
Compute $\mathbf{r}_0 = \mathbf{b} - A\mathbf{u}_0$, $\beta = ||\mathbf{r}_0||_2$ and $\mathbf{v}_1 = \mathbf{r}_0/\beta$
Define $H_m = \{0\}$
Define $V_1 = \{\mathbf{v}_1\}$
for $j = 1, 2, \dots, m$ do
    $\mathbf{w}_j = A\mathbf{v}_j$
    for $i = 1, 2, \dots, j$ do
        $h_{ij} = (\mathbf{w}_j, \mathbf{v}_i)$ and store $h_{ij}$ in $H_m$
        $\mathbf{w}_j = \mathbf{w}_j - h_{ij}\mathbf{v}_i$
    end for
    $h_{j+1,j} = ||\mathbf{w}_j||_2$
    if $h_{j+1,j} = 0$ then
        $m = j$
        break
    end if
    $\mathbf{v}_{j+1} = \mathbf{w}_j/h_{j+1,j}$ and store $\mathbf{v}_{j+1}$ into $V_{j+1}$
end for
Solve $H_m\mathbf{c} = \beta e_1$ for $\mathbf{c}$
$\mathbf{u}_m = \mathbf{u}_0 + V_m\mathbf{c}$
```

Note that a stopping criterion can be derived from the residual vector $\mathbf{r}_m = \mathbf{b} - A\mathbf{u}_m$. Theorem A.1 gives a way of calculating the size of the residual vector.

Theorem A.1: Arnoldi residual [17, Proposition 6.7]

The residual vector $\mathbf{r}_m = \mathbf{b} - A\mathbf{u}_m$ satisfies

$$||r_m||_2 = h_{m+1,m}|\mathbf{e}_m^T\mathbf{c}|, \tag{A.1}$$

Proof. We have

$$\begin{aligned} \mathbf{r}_m &= \mathbf{b} - A\mathbf{u}_m \\ &= \mathbf{r}_0 - AV_m\mathbf{c} \\ &= \beta v_1 - V_m H_m \mathbf{c} - h_{m+1,m}\mathbf{e}_m^T\mathbf{c}\mathbf{v}_{m+1} \\ &= -h_{m+1,m}\mathbf{e}_m^T\mathbf{c}\mathbf{v}_{m+1}. \end{aligned}$$

The result follows by taking the 2-norm of both sides of the equality and using the fact that $||\mathbf{v}_{m+1}||_2 = 1$. □

A.2. Lanczos' Algorithm

In the special case where A is symmetric, the Arnoldi method can be simplified to the Lanczos algorithm. In particular, for symmetric A, the Hessenberg matrix H_m is tridiagonal

$$H_m = T_m = \begin{pmatrix} \delta_1 & \eta_2 & 0 & \dots & 0 \\ \eta_2 & \delta_3 & \eta_3 & \dots & 0 \\ 0 & \eta_3 & \delta_4 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \eta_m \\ 0 & 0 & 0 & \eta_m & \delta_m \end{pmatrix}, \tag{A.2}$$

where we redefined H_m to be the tridiagonal matrix T_m. The tridiagonality of T_m allows us to reduce the Gramm-Schmidt orthogonalization procedure in the inner for-loop in algorithm A.1 to just two vector subtractions and an inner product, resulting in algorithm A.1

Algorithm A.2 Lanczos algorithm for linear systems [17, Algorithm 6.16]

```
Compute $\mathbf{r}_0 = b - A\mathbf{u}_0$, $\beta = ||\mathbf{r}_0||_2$, $\mathbf{v}_0 = 0$ and $\mathbf{v}_1 = \mathbf{r}_0/\beta$
$V_1 = \{\mathbf{v}_1\}$
for $j = 1, 2, \dots, m$ do
    $\mathbf{w}_j = A\mathbf{v}_j - \eta_j \mathbf{v}_{j-1}$
    $\delta_j = (\mathbf{w}_j, \mathbf{v}_j)$
    $\mathbf{w}_j = \mathbf{w}_j - \delta_j \mathbf{v}_j$
    $\eta_{j+1} = ||\mathbf{w}_j||_2$
    if $\eta_{j+1} = 0$ then
        $m = j$
        Break
    end if
    $\mathbf{v}_{j+1} = \mathbf{w}_j/\eta_{j+1}$ and store $\mathbf{v}_{j+1}$ into $V_{j+1}$
end for
Solve the tridiagonal system $T_m\mathbf{c} = \beta\mathbf{e}_1$ for $\mathbf{c}$
$\mathbf{u}_m = \mathbf{u}_0 + V_m\mathbf{c}$
```

A.3. D-Lanczos

A downside of algorithm A.2 in particular and projections methods like algorithm 1 in general is their reliance on an arbitrary choice of dimension m. The methods run until the basis V_m is constructed and subsequently construct H_m to determine the correction $\mathbf{c}$. This is not ideal, since the resulting solution $\mathbf{u}_m$ may not be close enough to the true solution $\mathbf{u}$. That is, it is not guaranteed that the residual vector $\mathbf{r}_m$ is 'small enough'. Alternately, it might happen m is chosen too large, and the method is unnecessarily expensive. In the specific case of the Arnoldi method Theorem A.1 may be used to determine the residual before calculating $\mathbf{c}$. Though this saves some computational time, it still requires the construction of the basis V_m and the tridiagonal matrix T_m, as well as a restart of the algorithm. This is not ideal, since the construction of V_m and T_m is expensive.

To address the issue of arbitrary m, we construct a version of algorithm A.2 that allows us to incrementally update the solution $\mathbf{u}_m$ and the residual vector $\mathbf{r}_m$. This way, we can stop the algorithm when the residual vector is smaller than some predefined threshold, like $\mathbf{r}_m < \epsilon$.

To that end, we start by performing a LU-factorisation of T_m given by

$$T_m = L_m U_m = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ \tilde{\eta}_2 & 1 & 0 & \dots & 0 \\ 0 & \tilde{\eta}_3 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \tilde{\eta}_m & 1 \end{pmatrix} \times \begin{pmatrix} \tilde{\delta}_1 & \eta_2 & 0 & \dots & 0 \\ 0 & \tilde{\delta}_2 & \eta_3 & \dots & 0 \\ 0 & 0 & \tilde{\delta}_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \eta_m \\ 0 & 0 & 0 & \dots & \tilde{\delta}_m \end{pmatrix} \tag{A.3}$$

Then, the approximate solution is given by

$$\begin{aligned}\mathbf{u}_m &= \mathbf{u}_0 + V_m\mathbf{c}\\ &= \mathbf{u}_0 + V_m U_m^{-1} L_m^{-1}\beta\mathbf{e}_1\\ &= \mathbf{u}_0 + V_m U_m^{-1}(L_m^{-1}\beta\mathbf{e}_1)\\ &= \mathbf{u}_0 + P_m\mathbf{z}_m,\end{aligned}$$

where $P_m = V_m U_m^{-1}$ and $\mathbf{z}_m = L_m^{-1}\beta\mathbf{e}_1$. Considering the definition of U_m in equation (A.3), we have that the m^{th} column of P_m is given by

$$\mathbf{p}_m = \frac{1}{\tilde{\delta}_m}\left[\mathbf{v}_m - \eta_m\mathbf{p}_{m-1}\right]. \tag{A.4}$$

Furthermore, from the LU factorization of T_m we have that

$$\begin{aligned}\tilde{\eta}_m &= \frac{\eta_m}{\tilde{\delta}_{m-1}},\\ \tilde{\delta}_m &= \delta_m - \tilde{\eta}_m\eta_m,\ m > 1.\end{aligned}$$

Now the solution can be incrementally updated by realizing that

$$\mathbf{z}_m = \begin{pmatrix}\mathbf{z}_{m-1}\\ \zeta_m\end{pmatrix} = \begin{pmatrix}\mathbf{z}_{m-2}\\ \zeta_{m-1}\\ \zeta_m\end{pmatrix},$$

and

$$L_m = \begin{pmatrix}L_{m-1} & \mathbf{0}_{m-1}\\ \mathbf{0}_{m-2}^T\quad \tilde{\eta}_m & 1\end{pmatrix}.$$

Then,

$$L_m\mathbf{z}_m = \begin{pmatrix}L_{m-1}\mathbf{z}_{m-1}\\ \tilde{\eta}_m\zeta_{m-1} + \zeta_m\end{pmatrix} = \begin{pmatrix}\beta\mathbf{e}_1\\ 0\end{pmatrix},$$

where the last equality follows from definition of $\mathbf{z}_m$. Consequently, we have that

$$\zeta_m = -\tilde{\eta}_m\zeta_{m-1}.$$

Finally, we obtain

$$\begin{aligned}u_m &= \mathbf{u}_0 + P_m\mathbf{z}_m\\ &= \mathbf{u}_0 + [P_{m-1}\mathbf{p}_m]\begin{pmatrix}\mathbf{z}_{m-1}\\ \zeta_m\end{pmatrix}\\ &= \mathbf{u}_0 + P_{m-1}\mathbf{z}_{m-1} + \mathbf{p}_m\zeta_m\\ &= \mathbf{u}_{m-1} + \mathbf{p}_m\zeta_m.\end{aligned}$$

Putting it all together, we obtain algorithm A.3.

Algorithm A.3 D-Lanczos [17, Algorithm 6.17]

```
$\mathbf{r}_0 = b - A\mathbf{u}_0$, $\beta = ||\mathbf{r}_0||_2$, $\mathbf{v}_1 = \mathbf{r}_0/\beta$
$\tilde{\eta}_1 = \beta_1 = 0$, $\mathbf{p}_0 = 0$
for $m = 1, 2, \ldots, m$ until convergence do
    $w = A\mathbf{v}_m - \beta_m \mathbf{v}_{m-1}$
    $\delta_m = (w, \mathbf{v}_m)$
    if $m > 1$ then
        $\tilde{\eta}_m = \frac{\beta_m}{\tilde{\delta}_{m-1}}$
        $\zeta_m = -\tilde{\eta}_m \zeta_{m-1}$
    end if
    $\tilde{\delta}_m = \delta_m - \tilde{\eta}_m \beta_m$
    $\mathbf{p}_m = \frac{1}{\tilde{\delta}_m} [\mathbf{v}_m - \beta_m \mathbf{p}_{m-1}]$
    $\mathbf{u}_m = \mathbf{u}_{m-1} + \mathbf{p}_m \zeta_m$
    if $\|\mathbf{r}_{m+1}\|_2 < \epsilon$ then
        break
    end if
    $\mathbf{w} = \mathbf{w} - \delta_m \mathbf{v}_m$
    $\beta_{m+1} = ||\mathbf{w}||_2$
    $\mathbf{v}_{m+1} = \mathbf{w}/\beta_{m+1}$
end for
```

A core aspect of algorithm A.3 is described in

Theorem A.2: A-orthogonality of p_m

The vectors p_m produced in algorithm algorithm A.3 are A-orthogonal to each other.

Proof. We have

$$\begin{aligned} P_m^T A P_m &= U_m^{-T} V_m^T A V_m U_m^{-1} \\ &= U_m^{-T} T_m U_m^{-1} \\ &= U_m^{-T} L_m, \end{aligned}$$

where U_m^{-T} and L_m are both lower diagonal matrices. Their product must be symmetric, since $P_m^T A P_m$ is symmetric (due to the symmetry of A). The result follows from the fact that $U_m^{-T} L_m$ must be a diagonal matrix ☐

A.4. Derivation of CG

From general properties of error projection methods and observations made in the in algorithm A.3, we can derive the CG method. We start by constraining subsequent residuals r_j to be orthogonal. This follows from choosing subspaces $\mathcal{K} = \mathcal{L}$, as in the Arnoldi process. Again, the space $\mathcal{K}$ is spanned by the vectors $\mathbf{v}_m$. Thus setting $\mathbf{v}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|_2$, automatically means subsequent residuals will be orthogonal to each other. Then, as suggested by Theorem A.2, we also require that the vectors p_j are A-orthogonal to each other. From this point on, we use the term *search direction* to refer to the vectors p_j. Next to this we also introduce the CG variables α_j and β_j, which are the step size and the search direction update, respectively. This results in the following update equations

$$\mathbf{u}_{j+1} = \mathbf{u}_j + \alpha_j \mathbf{p}_j, \tag{A.5}$$

and, thereby,

$$\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j A\mathbf{p}_j. \tag{A.6}$$

If the residuals are to be orthogonal, then

$$(\mathbf{r}_{j+1}, \mathbf{r}_j) = 0 \implies (\mathbf{r}_j - \alpha_j A\mathbf{p}_j, \mathbf{r}_j) = 0 \implies \alpha_j = \frac{(\mathbf{r}_j, \mathbf{r}_j)}{(A\mathbf{p}_j, \mathbf{r}_j)}.$$

Now, using the relation between $\mathbf{r}_m$ and $\mathbf{v}_{m+1}$ found in the proof of Theorem A.1 and equation (A.4), we can write the next search direction as a linear combination of the previous search direction and the next residual

$$\mathbf{p}_{j+1} = \mathbf{r}_{j+1} + \beta_j \mathbf{p}_j. \tag{A.7}$$

Substituting equation (A.7), we obtain

$$(A\mathbf{p}_{j+1}, \mathbf{r}_j) = (A\mathbf{p}_j, \mathbf{p}_j - \beta_{j-1}\mathbf{p}_{j-1}) = (A\mathbf{p}_j, \mathbf{p}_j),$$

since p_j is A-orthogonal to p_{j-1}. This allows us to write

$$\alpha_j = \frac{(\mathbf{r}_j, \mathbf{r}_j)}{(A\mathbf{p}_j, \mathbf{p}_j)}. \tag{A.8}$$

Additionally, taking the inner product with $A\mathbf{p}_j$ on both sides of equation (A.7) gives

$$\beta_j = \frac{(\mathbf{r}_{j+1}, A\mathbf{p}_j)}{(\mathbf{p}_j, A\mathbf{p}_j)}.$$

Now, rewriting equation (A.6) gives

$$A\mathbf{p}_j = \frac{1}{\alpha_j}(\mathbf{r}_j - \mathbf{r}_{j+1}),$$

which we substitute into the equation for β_j to obtain

$$\beta_j = \frac{1}{\alpha_j}\frac{(\mathbf{r}_{j+1}, (\mathbf{r}_{j+1} - \mathbf{r}_j))}{(A\mathbf{p}_j, \mathbf{r}_j)} = \frac{(\mathbf{r}_{j+1}, \mathbf{r}_{j+1})}{((\mathbf{r}_j - \mathbf{r}_{j-1}), \mathbf{r}_j)} = \frac{(\mathbf{r}_{j+1}, \mathbf{r}_{j+1})}{(\mathbf{r}_j, \mathbf{r}_j)}. \tag{A.9}$$

Finally, equations (A.5) to (A.9) comprise one iteration of the CG method, as shown in algorithm 2.

A.5. CG relation to Lanczos

To do ►*Describe relation between CG and Lanczos coefficients.*◄

B. Chebyshev approximation

B.1. Chebyshev polynomials

This section introduces Chebyshev polynomials and some of their properties. First, their definition in Definition B.1.

Definition B.1: Chebyshev polynomial

The m^{th} degree Chebyshev polynomial of the first kind is denoted as C_m, for $z \in \mathbb{C}$

$$C_m(z) = \begin{cases} \cos(m\cos^{-1}(z)), & |z| \le 1, \\ \cosh(m\cosh^{-1}(z)), & |z| > 1, \end{cases},$$

as well as through the recurrence relation

$$C_m(z) = 2zC_{m-1}(z) - C_{m-2}(z), \quad m \ge 2,$$

with initial conditions

$$C_0(z) = 1, \quad C_1(z) = z.$$

For $|z| > 1$ we can also write

$$C_m(z) = \frac{1}{2}\left(\left(z + \sqrt{z^2-1}\right)^m + \left(z - \sqrt{z^2-1}\right)^m\right), \tag{B.1}$$

which for large k may be approximated as

$$C_m(z) \gtrsim \frac{1}{2}\left(z + \sqrt{z^2-1}\right)^m. \tag{B.2}$$

The extreme points of C_m are given by

$$z_k = \cos\left(\frac{k\pi}{m}\right), \quad k = 0, 1, \ldots, m. \tag{B.3}$$

Indeed, substituting z_k into C_m gives

$$C_m(z_k) = \cos(k\pi) = \cos(k\pi) = (-1)^k, \quad k = 0, 1, \ldots, m. \tag{B.4}$$

For the optimality proof we need to introduce the real-valued, transformed Chebyshev polynomial in **??**

Definition B.2: Real Transformed Chebyshev polynomial

The transformed Chebyshev polynomial of the first kind is denoted as $\hat{C}_m$, for $x \in \mathbb{R}$ is obtained through an affine change of variables T from the $[a, b] \subset \mathbb{R}$ to the interval $[-1, 1]$ as

$$t \in [a, b] \implies z = T(t) = \frac{2t - (a + b)}{b - a} \in [-1, 1],$$

and a subsequent scaling with the factor $C_m(T(\gamma))$ for $\gamma \in \mathbb{R}$ outside the interval $[a, b]$ as

$$\hat{C}_m(t) = \frac{C_m(T(t))}{C_m(T(\gamma))} = \frac{C_m\left(\frac{2t-(a+b)}{b-a}\right)}{C_m\left(\frac{2\gamma-(a+b)}{b-a}\right)}$$

Lastly, by equation (B.4) we get for $t_k = T^{-1}(z_k)$

$$\hat{C}_m(t_k) = \frac{(-1)^k}{C_m(T(\gamma))} = \frac{(-1)^k}{d_m(\gamma)}, \tag{B.5}$$

where $d_m(\gamma) = C_m(T(\gamma))$.

B.2. Chebyshev optimality

We now show that $\hat{C}_m$ from Definition B.2 is the solution of the following minimization problem

Theorem B.1: Min-max polynomial

The real-valued polynomial $p_m(t)$ of degree m such that for $\gamma \in \mathbb{R}$ outside the interval $[a, b]$ the following holds

$$\min_{p \in \mathcal{P}_m, p(\gamma)=1} \max_{t \in [a,b]} |p(t)|,$$

is given by the Chebyshev polynomial $\hat{C}_m$. Furthermore, we have

$$\min_{p \in \mathcal{P}_m, p(\gamma)=1} \max_{t \in [a,b]} |p_m(t)| = \frac{1}{d_m(\gamma)},$$

where $d_m(\gamma)$ is as in equation (B.5).

Proof. We proof this by contradiction. First, note that by equation (B.5) we have

$$\max_{t \in [a,b]} |\hat{C}_m(t)| = \max_{k=0,1,\ldots,m} |\hat{C}_m(t_k)| = \frac{1}{d_m},$$

Assume that there exists a polynomial $w_m(t)$ of degree m such that

$$\max_{t \in [a,b]} |w_m(t)| < \frac{1}{d_m}.$$

Without loss of generality we can assume w_m, just like $\hat{C}_m$, is a monic polynomial, i.e. $w_m(t) = t^m + \ldots$. We now define the difference polynomial

$$f_m(t) = \hat{C}_m(t) - w_m(t) \in \mathcal{P}_{m-1},$$

where the inclusion in the $m-1$ degree polynomials follows from the fact that f_m is the difference of two monic polynomials of degree m.

Consider the values of f_m at the extreme points $t_k = T^{-1}(z_k)$ of $\hat{C}_m$ with z_k as in equation (B.3) and T as in Definition B.2. We distinguish between even and odd k. By equation (B.5) and the assumption on w_m we then obtain

even k: $f_m(t_k) = \dfrac{1}{d_m} - w_m(t_k) > 0.$

odd k: $f_m(t_k) = -\dfrac{1}{d_m} - w_m(t_k) < 0.$

From this we gather that $f_m(t_k)$ has alternating signs at the extreme points t_k of $\hat{C}_m$.

Now, the sequence $(z_k)_{k=0}^m$ is decreasing, and thus the sequence $(t_k)_{k=0}^m$ is also decreasing. This means that we can construct m distinct intervals $I_k = [t_{k+1}, t_k]$ such that f_m switches sign in each interval. By the intermediate value theorem, we know that f_m must have at least one root in each interval I_k. This leads to the conclusion that f_m has at least m roots in the interval $[a, b]$.

However, by the fundamental theorem of algebra f_m, a polynomial of degree $m-1$, can have at most $m-1$ distinct roots. The only possibility is for $f_m \equiv 0$, but then we have

$$\max_{t\in[a,b]} |w_m(t)| = \max_{t\in[a,b]} |\hat{C}_m(t)| = \frac{1}{d_m},$$

which contradicts our main assumption that w_m is a polynomial such that $\max\limits_{t\in[a,b]} |w_m(t)| < \dfrac{1}{d_m}$. Thus, we conclude that the Chebyshev polynomial $\hat{C}_m$ is indeed the solution to the minimization problem. □

C. Rayleigh quotient

Definition C.1: T

e Rayleigh quotient of a matrix A and a vector $\mathbf{u}$ is defined as

$$R(A, \mathbf{u}) = \frac{\mathbf{u}^T A\mathbf{u}}{\mathbf{u}^T\mathbf{u}}. \tag{C.1}$$

Theorem C.1: Rayleigh quotient bound

Suppose A is symmetric. Then the Rayleigh quotient $R(A, \mathbf{u})$ is bounded by the smallest and largest eigenvalue of A, i.e.

$$\lambda_{\min} \leq R(A, \mathbf{u}) \leq \lambda_{\max}.$$

Proof. A has diagonalization $A = Q\Lambda Q^T$, where Q is an orthonormal eigenbasis and Λ is the diagonal with real and positive eigenvalues λ_i of A. The Rayleigh quotient satisfies with $\mathbf{v} = Q\mathbf{u}$

$$R(A, \mathbf{u}) = \frac{\mathbf{u}^T A\mathbf{u}}{\mathbf{u}^T\mathbf{u}} = \frac{\mathbf{v}^T \Lambda\mathbf{v}}{\mathbf{v}^T\mathbf{v}} = \sum_{i=1}^n \frac{\lambda_i v_i^2}{\|v\|_2},$$

which is a convex combination of the eigenvalues λ_i of A. Thus, we have

$$\lambda_{\min} \leq R(A, \mathbf{u}) \leq \lambda_{\max}.$$

□

Bibliography

[1] Filipe A. C. S. Alves, Alexander Heinlein, and Hadi Hajibeygi. *A computational study of algebraic coarse spaces for two-level overlapping additive Schwarz preconditioners*. 2024. arXiv: 2408.08187 [math.NA]. URL (cit. on pp. 27, 29).

[2] O. Axelsson. “A class of iterative methods for finite element equations”. In: *Computer Methods in Applied Mechanics and Engineering* 9.2 (1976), pp. 123–137. ISSN: 0045-7825. DOI: https://doi.org/10.1016/0045-7825(76)90056-6. URL (cit. on pp. 28, 31, 33).

[3] B. Beckermann and A. B. J. Kuijlaars. “On The Sharpness of an Asymptotic Error Estimate for Conjugate Gradients”. In: *BIT Numerical Mathematics* 41.5 (2001), pp. 856–867 (cit. on p. 28).

[4] Bernhard Beckermann and Arno B. J. Kuijlaars. “Superlinear CG Convergence for Special Right-Hand Sides”. In: *Electronic Transactions on Numerical Analysis* 14 (2002), pp. 1–19. ISSN: 1068-9613. URL (cit. on p. 28).

[5] Bernhard Beckermann and Arno B. J. Kuijlaars. “Superlinear Convergence of Conjugate Gradients”. In: *SIAM Journal on Numerical Analysis* 39.1 (2001), pp. 300–329. DOI: 10.1137/S0036142999363188. eprint: https://doi.org/10.1137/S0036142999363188. URL (cit. on p. 28).

[6] Clark R. Dohrmann, Axel Klawonn, and Olof B. Widlund. “A family of energy minimizing coarse spaces for overlapping Schwarz preconditioners”. In: *Domain Decomposition Methods in Science and Engineering XVII*. Vol. 60. Lecture Notes in Computational Science and Engineering. St. Wolfgang / Strobl, Austria, 2008, pp. 247–254 (cit. on p. 27).

[7] Clark R. Dohrmann and Olof B. Widlund. “On the Design of Small Coarse Spaces for Domain Decomposition Algorithms”. In: *SIAM Journal on Scientific Computing* 39.4 (2017), A1466–A1488. DOI: 10.1137/17M1114272. eprint: https://doi.org/10.1137/17M1114272. URL (cit. on p. 27).

[8] Victorita Dolean, Pierre Jolivet, and Frédéric Nataf. *An Introduction to Domain Decomposition Methods*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2015. DOI: 10.1137/1.9781611974065. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611974065. URL (cit. on pp. 18–25).

[9] Yalchin Efendiev, Juan Galvis, and Xiao-Hui Wu. “Multiscale finite element methods for high-contrast problems using local spectral basis functions”. In: *Journal of Computational Physics* 230.4 (2011), pp. 937–955. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2010.09.026. URL (cit. on p. 26).

[10] I. G. Graham, P. O. Lechner, and R. Scheichl. “Domain decomposition for multiscale PDEs”. In: *Numerische Mathematik* 106 (2007), pp. 589–626. DOI: 10.1007/s00211-007-0074-1. URL (cit. on p. 26).

[11] Alexander Heinlein. “Multiscale coarse spaces for overlapping Schwarz methods based on the ACMS space in 2D”. en. In: (2018). Ed. by Lothar Reichel (Hg.) Ronny Ramlau, pp. 156–182. ISSN: 1068-9613. URL (cit. on p. 26).

[12] Alexander Heinlein et al. “Adaptive GDSW Coarse Spaces of Reduced Dimension for Overlapping Schwarz Methods”. In: *SIAM Journal on Scientific Computing* 44.3 (2022), A1176–A1204. DOI: 10.1137/20M1364540. eprint: https://doi.org/10.1137/20M1364540. URL (cit. on p. 27).

[13] Thomas Y. Hou and Xiao-Hui Wu. “A Multiscale Finite Element Method for Elliptic Problems in Composite Materials and Porous Media”. In: *J. Comput. Phys.* 134.1 (June 1997), pp. 169–189. ISSN: 0021-9991. DOI: 10.1006/jcph.1997.5682. URL (cit. on p. 26).

[14] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1995. DOI: 10.1137/1.9781611970944. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611970944. URL (cit. on p. 16).

[15] Ivan Lunati and Seong H. Lee. “An Operator Formulation of the Multiscale Finite-Volume Method with Correction Function”. In: *Multiscale Modeling & Simulation* 8.1 (2009), pp. 96–109. DOI: 10.1137/080742117. eprint: https://doi.org/10.1137/080742117. URL (cit. on p. 27).

[16] Gérard Meurant and Zdeněk Strakoš. “The Lanczos and conjugate gradient algorithms in finite precision arithmetic”. In: *Acta Numerica* 15 (2006), pp. 471–542. DOI: 10.1017/S096249290626001X (cit. on p. 13).

[17] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Second. Society for Industrial and Applied Mathematics, 2003. DOI: 10.1137/1.9780898718003. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9780898718003. URL (cit. on pp. 4–8, 18, 39, 40, 42).

[18] H.A. Schwarz. In: *Journal für die reine und angewandte Mathematik* 1869.70 (1869), pp. 105–120. DOI: doi:10.1515/crll.1869.70.105. URL (cit. on p. 18).

[19] Z. Strakoš. “On the real convergence rate of the conjugate gradient method”. In: *Linear Algebra and its Applications* 154-156 (1991), pp. 535–549. ISSN: 0024-3795. DOI: https://doi.org/10.1016/0024-3795(91)90393-B. URL (cit. on p. 28).

[20] Y. Wang, H. Hajibeygi, and H.A. Tchelepi. “Algebraic multiscale solver for flow in heterogeneous porous media”. English. In: *Journal of Computational Physics* 259.February (2014). Harvest, pp. 284–303. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2013.11.024 (cit. on p. 27).